

GUIslice

0.14.0

Generated by Doxygen 1.8.11

Contents

1	GUIslice library	1
2	Todo List	3
3	Module Index	5
3.1	Modules	5
4	Hierarchical Index	7
4.1	Class Hierarchy	7
5	Data Structure Index	9
5.1	Data Structures	9
6	File Index	11
6.1	File List	11
7	Module Documentation	13
7.1	General Functions	13
7.1.1	Detailed Description	14
7.1.2	Function Documentation	14
7.1.2.1	gslc_DebugPrintf(const char *pFmt,...)	14
7.1.2.2	gslc_GetDriverDisp(gslc_tsGui *pGui)	14
7.1.2.3	gslc_GetDriverTouch(gslc_tsGui *pGui)	14
7.1.2.4	gslc_GetNameDisp(gslc_tsGui *pGui)	15
7.1.2.5	gslc_GetNameTouch(gslc_tsGui *pGui)	15
7.1.2.6	gslc_GetVer(gslc_tsGui *pGui)	15

7.1.2.7	<code>gslc_GuiRotate(gslc_tsGui *pGui, uint8_t nRotation)</code>	16
7.1.2.8	<code>gslc_Init(gslc_tsGui *pGui, void *pvDriver, gslc_tsPage *asPage, uint8_t nMax↵ Page, gslc_tsFont *asFont, uint8_t nMaxFont)</code>	16
7.1.2.9	<code>gslc_InitDebug(GSLC_CB_DEBUG_OUT pfunc)</code>	16
7.1.2.10	<code>gslc_Quit(gslc_tsGui *pGui)</code>	17
7.1.2.11	<code>gslc_SetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	17
7.1.2.12	<code>gslc_SetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	18
7.1.2.13	<code>gslc_SetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	18
7.1.2.14	<code>gslc_SetTransparentColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	18
7.1.2.15	<code>gslc_Update(gslc_tsGui *pGui)</code>	19
7.2	Graphics General Functions	20
7.2.1	Detailed Description	21
7.2.2	Function Documentation	21
7.2.2.1	<code>gslc_ClipLine(gslc_tsRect *pClipRect, int16_t *pnX0, int16_t *pnY0, int16_t↵ t *pnX1, int16_t *pnY1)</code>	21
7.2.2.2	<code>gslc_ClipPt(gslc_tsRect *pClipRect, int16_t nX, int16_t nY)</code>	21
7.2.2.3	<code>gslc_ClipRect(gslc_tsRect *pClipRect, gslc_tsRect *pRect)</code>	21
7.2.2.4	<code>gslc_ColorBlend2(gslc_tsColor colStart, gslc_tsColor colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)</code>	22
7.2.2.5	<code>gslc_ColorBlend3(gslc_tsColor colStart, gslc_tsColor colMid, gslc_tsColor col↵ End, uint16_t nMidAmt, uint16_t nBlendAmt)</code>	22
7.2.2.6	<code>gslc_ColorEqual(gslc_tsColor a, gslc_tsColor b)</code>	23
7.2.2.7	<code>gslc_cosFX(int16_t n64Ang)</code>	23
7.2.2.8	<code>gslc_ExpandRect(gslc_tsRect rRect, int16_t nExpandW, int16_t nExpandH)</code>	23
7.2.2.9	<code>gslc_GetImageFromFile(const char *pFname, gslc_telmgRefFlags eFmt)</code>	24
7.2.2.10	<code>gslc_GetImageFromProg(const unsigned char *plmgBuf, gslc_telmgRefFlags eFmt)</code>	24
7.2.2.11	<code>gslc_GetImageFromRam(unsigned char *plmgBuf, gslc_telmgRefFlags eFmt)</code> . .	24
7.2.2.12	<code>gslc_GetImageFromSD(const char *pFname, gslc_telmgRefFlags eFmt)</code>	24
7.2.2.13	<code>gslc_InvalidateRgnAdd(gslc_tsGui *pGui, gslc_tsRect rAddRect)</code>	25
7.2.2.14	<code>gslc_InvalidateRgnPage(gslc_tsGui *pGui, gslc_tsPage *pPage)</code>	25
7.2.2.15	<code>gslc_InvalidateRgnReset(gslc_tsGui *pGui)</code>	25
7.2.2.16	<code>gslc_InvalidateRgnScreen(gslc_tsGui *pGui)</code>	26

7.2.2.17	<code>gslc_IsInRect(int16_t nSelX, int16_t nSelY, gslc_tsRect rRect)</code>	26
7.2.2.18	<code>gslc_IsInWH(int16_t nSelX, int16_t nSelY, uint16_t nWidth, uint16_t nHeight)</code>	26
7.2.2.19	<code>gslc_PolarToXY(uint16_t nRad, int16_t n64Ang, int16_t *nDX, int16_t *nDY)</code>	27
7.2.2.20	<code>gslc_sinFX(int16_t n64Ang)</code>	27
7.2.2.21	<code>gslc_UnionRect(gslc_tsRect *pRect, gslc_tsRect rAddRect)</code>	27
7.3	Graphics Primitive Functions	29
7.3.1	Detailed Description	30
7.3.2	Function Documentation	30
7.3.2.1	<code>gslc_DrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)</code>	30
7.3.2.2	<code>gslc_DrawFillGradSector(gslc_tsGui *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, gslc_tsColor cArcStart, gslc_tsColor cArcEnd, int16_t nAngSecStart, int16_t nAngSecEnd, int16_t nAngGradStart, int16_t nAngGradRange)</code>	30
7.3.2.3	<code>gslc_DrawFillQuad(gslc_tsGui *pGui, gslc_tsPt *psPt, gslc_tsColor nCol)</code>	31
7.3.2.4	<code>gslc_DrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	31
7.3.2.5	<code>gslc_DrawFillRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)</code>	31
7.3.2.6	<code>gslc_DrawFillSector(gslc_tsGui *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, gslc_tsColor cArc, int16_t nAngSecStart, int16_t nAngSecEnd)</code>	32
7.3.2.7	<code>gslc_DrawFillTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)</code>	32
7.3.2.8	<code>gslc_DrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)</code>	33
7.3.2.9	<code>gslc_DrawFrameQuad(gslc_tsGui *pGui, gslc_tsPt *psPt, gslc_tsColor nCol)</code>	33
7.3.2.10	<code>gslc_DrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	33
7.3.2.11	<code>gslc_DrawFrameRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)</code>	34
7.3.2.12	<code>gslc_DrawFrameTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)</code>	34
7.3.2.13	<code>gslc_DrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)</code>	35
7.3.2.14	<code>gslc_DrawLineH(gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nW, gslc_tsColor nCol)</code>	35
7.3.2.15	<code>gslc_DrawLinePolar(gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nRadStart, uint16_t nRadEnd, int16_t n64Ang, gslc_tsColor nCol)</code>	35

7.3.2.16	<code>gslc_DrawLineV(gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nH, gslc_tsColor nCol)</code>	36
7.3.2.17	<code>gslc_DrawSetPixel(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)</code>	36
7.4	Font Functions	38
7.4.1	Detailed Description	38
7.4.2	Function Documentation	38
7.4.2.1	<code>gslc_FontAdd(gslc_tsGui *pGui, int16_t nFontId, gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)</code>	38
7.4.2.2	<code>gslc_FontGet(gslc_tsGui *pGui, int16_t nFontId)</code>	39
7.4.2.3	<code>gslc_FontSet(gslc_tsGui *pGui, int16_t nFontId, gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)</code>	39
7.4.2.4	<code>gslc_FontSetMode(gslc_tsGui *pGui, int16_t nFontId, gslc_teFontRefMode eFontMode)</code>	39
7.5	Page Functions	41
7.5.1	Detailed Description	41
7.5.2	Function Documentation	41
7.5.2.1	<code>gslc_GetPageCur(gslc_tsGui *pGui)</code>	41
7.5.2.2	<code>gslc_PageAdd(gslc_tsGui *pGui, int16_t nPageId, gslc_tsElem *psElem, uint16_t nMaxElem, gslc_tsElemRef *psElemRef, uint16_t nMaxElemRef)</code>	42
7.5.2.3	<code>gslc_PageFindElemById(gslc_tsGui *pGui, int16_t nPageId, int16_t nElemId)</code>	42
7.5.2.4	<code>gslc_PageRedrawGet(gslc_tsGui *pGui)</code>	43
7.5.2.5	<code>gslc_PageRedrawSet(gslc_tsGui *pGui, bool bRedraw)</code>	43
7.5.2.6	<code>gslc_PopupHide(gslc_tsGui *pGui)</code>	43
7.5.2.7	<code>gslc_PopupShow(gslc_tsGui *pGui, int16_t nPageId, bool bModal)</code>	43
7.5.2.8	<code>gslc_SetPageBase(gslc_tsGui *pGui, int16_t nPageId)</code>	44
7.5.2.9	<code>gslc_SetPageCur(gslc_tsGui *pGui, int16_t nPageId)</code>	44
7.5.2.10	<code>gslc_SetPageOverlay(gslc_tsGui *pGui, int16_t nPageId)</code>	44
7.5.2.11	<code>gslc_SetStackPage(gslc_tsGui *pGui, uint8_t nStackPos, int16_t nPageId)</code>	45
7.5.2.12	<code>gslc_SetStackState(gslc_tsGui *pGui, uint8_t nStackPos, bool bActive, bool bDoDraw)</code>	45
7.6	Element Functions	46
7.6.1	Detailed Description	46
7.7	Element: Creation Functions	47

7.7.1	Detailed Description	47
7.7.2	Function Documentation	47
7.7.2.1	gslc_ElemCreateBox(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_↵ tsRect rElem)	47
7.7.2.2	gslc_ElemCreateBtnImg(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, gslc_tsImgRef slmgRef, gslc_tsImgRef slmgRefSel, GSLC_↵ _CB_TOUCH cbTouch)	48
7.7.2.3	gslc_ElemCreateBtnTxt(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_↵ tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId, GSLC_CB_↵ _TOUCH cbTouch)	48
7.7.2.4	gslc_ElemCreateImg(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_↵ tsRect rElem, gslc_tsImgRef slmgRef)	49
7.7.2.5	gslc_ElemCreateLine(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1)	49
7.7.2.6	gslc_ElemCreateTxt(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_ts_↵ Rect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)	50
7.8	Element: General Functions	51
7.8.1	Detailed Description	51
7.8.2	Function Documentation	51
7.8.2.1	gslc_ElemGetId(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	51
7.9	Element: Update Functions	52
7.9.1	Detailed Description	53
7.9.2	Function Documentation	53
7.9.2.1	gslc_ElemGetGlow(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	53
7.9.2.2	gslc_ElemGetGlowEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	54
7.9.2.3	gslc_ElemGetGroup(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	54
7.9.2.4	gslc_ElemGetOnScreen(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	54
7.9.2.5	gslc_ElemGetRedraw(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	55
7.9.2.6	gslc_ElemGetTxtStr(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	55
7.9.2.7	gslc_ElemGetVisible(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	55
7.9.2.8	gslc_ElemOwnsCoord(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nX, int16_t nY, bool bOnlyClickEn)	56
7.9.2.9	gslc_ElemSetClickEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool b_↵ ClickEn)	56

7.9.2.10	<code>gslc_ElemSetCol(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colFrame, gslc_tsColor colFill, gslc_tsColor colFillGlow)</code>	56
7.9.2.11	<code>gslc_ElemSetDrawFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_DRAW funcCb)</code>	57
7.9.2.12	<code>gslc_ElemSetFillEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFillEn)</code>	57
7.9.2.13	<code>gslc_ElemSetFrameEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFrameEn)</code>	58
7.9.2.14	<code>gslc_ElemSetGlow(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bGlowing)</code>	58
7.9.2.15	<code>gslc_ElemSetGlowCol(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colFrameGlow, gslc_tsColor colFillGlow, gslc_tsColor colTxtGlow)</code>	58
7.9.2.16	<code>gslc_ElemSetGlowEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bGlowEn)</code>	59
7.9.2.17	<code>gslc_ElemSetGroup(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int nGroupId)</code>	59
7.9.2.18	<code>gslc_ElemSetRedraw(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tRedrawType eRedraw)</code>	59
7.9.2.19	<code>gslc_ElemSetRoundEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bRoundEn)</code>	60
7.9.2.20	<code>gslc_ElemSetStyleFrom(gslc_tsGui *pGui, gslc_tsElemRef *pElemRefSrc, gslc_tsElemRef *pElemRefDest)</code>	60
7.9.2.21	<code>gslc_ElemSetTickFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_TICK funcCb)</code>	60
7.9.2.22	<code>gslc_ElemSetTouchFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_TOUCH funcCb)</code>	61
7.9.2.23	<code>gslc_ElemSetTxtAlign(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned nAlign)</code>	61
7.9.2.24	<code>gslc_ElemSetTxtCol(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colVal)</code>	62
7.9.2.25	<code>gslc_ElemSetTxtEnc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tTxtFlags eFlags)</code>	63
7.9.2.26	<code>gslc_ElemSetTxtMargin(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned nMargin)</code>	63
7.9.2.27	<code>gslc_ElemSetTxtMarginXY(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nMarginX, int8_t nMarginY)</code>	63
7.9.2.28	<code>gslc_ElemSetTxtMem(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tTxtFlags eFlags)</code>	64
7.9.2.29	<code>gslc_ElemSetTxtStr(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, const char *pStr)</code>	64
7.9.2.30	<code>gslc_ElemSetVisible(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bVisible)</code>	64

7.9.2.31	<code>gslc_ElemUpdateFont(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int nFontId)</code>	65
7.10	Touchscreen Functions	66
7.10.1	Detailed Description	66
7.10.2	Macro Definition Documentation	66
7.10.2.1	<code>TOUCH_ROTATION_DATA</code>	66
7.10.2.2	<code>TOUCH_ROTATION_DATA</code>	67
7.10.2.3	<code>TOUCH_ROTATION_FLIPX</code>	67
7.10.2.4	<code>TOUCH_ROTATION_FLIPX</code>	67
7.10.2.5	<code>TOUCH_ROTATION_FLIPY</code>	67
7.10.2.6	<code>TOUCH_ROTATION_FLIPY</code>	67
7.10.2.7	<code>TOUCH_ROTATION_SWAPXY</code>	67
7.10.2.8	<code>TOUCH_ROTATION_SWAPXY</code>	67
7.10.3	Function Documentation	67
7.10.3.1	<code>gslc_GetTouch(gslc_tsGui *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, gslc_teInputRawEvent *peInputEvent, int16_t *pnInputVal)</code>	67
7.10.3.2	<code>gslc_InitTouch(gslc_tsGui *pGui, const char *acDev)</code>	67
7.10.3.3	<code>gslc_SetTouchRemapCal(gslc_tsGui *pGui, uint16_t nXMin, uint16_t nXMax, uint16_t nYMin, uint16_t nYMax)</code>	68
7.10.3.4	<code>gslc_SetTouchRemapEn(gslc_tsGui *pGui, bool bEn)</code>	68
7.10.3.5	<code>gslc_SetTouchRemapYX(gslc_tsGui *pGui, bool bSwap)</code>	68
7.11	Input Mapping Functions	70
7.11.1	Detailed Description	70
7.11.2	Function Documentation	70
7.11.2.1	<code>gslc_InitInputMap(gslc_tsGui *pGui, gslc_tsInputMap *asInputMap, uint8_t n↔InputMapMax)</code>	70
7.11.2.2	<code>gslc_InputMapAdd(gslc_tsGui *pGui, gslc_teInputRawEvent eInputEvent, int16_t nInputVal, gslc_teAction eAction, int16_t nActionVal)</code>	70
7.11.2.3	<code>gslc_SetPinPollFunc(gslc_tsGui *pGui, GSLC_CB_PIN_POLL pfunc)</code>	70
7.12	General Purpose Macros	71
7.12.1	Detailed Description	71
7.12.2	Macro Definition Documentation	71
7.12.2.1	<code>GSLC_DEBUG2_PRINT</code>	71

7.12.2.2	GSLC_DEBUG2_PRINT_CONST	71
7.12.2.3	GSLC_DEBUG_PRINT	71
7.12.2.4	GSLC_DEBUG_PRINT_CONST	71
7.13	Flash-based Element Macros	72
7.13.1	Detailed Description	72
7.13.2	Macro Definition Documentation	72
7.13.2.1	gslc_ElemCreateBox_P	72
7.13.2.2	gslc_ElemCreateBtnTxt_P	73
7.13.2.3	gslc_ElemCreateLine_P	73
7.13.2.4	gslc_ElemCreateTxt_P	73
7.13.2.5	gslc_ElemCreateTxt_P_R	74
7.13.2.6	gslc_ElemCreateTxt_P_R_ext	74
7.14	Internal Functions	76
7.14.1	Detailed Description	81
7.14.2	Variable Documentation	81
7.14.2.1	abPageStackActive	81
7.14.2.2	abPageStackDoDraw	81
7.14.2.3	apPageStack	81
7.14.2.4	asElem	82
7.14.2.5	asElemRef	82
7.14.2.6	asFont	82
7.14.2.7	asInputMap	82
7.14.2.8	asPage	82
7.14.2.9	b	82
7.14.2.10	bInvalidateEn	82
7.14.2.11	bRedrawPartialEn	82
7.14.2.12	bScreenNeedFlip	82
7.14.2.13	bScreenNeedRedraw	82
7.14.2.14	bTouchRemapEn	83
7.14.2.15	bTouchRemapYX	83

7.14.2.16 colElemFill	83
7.14.2.17 colElemFillGlow	83
7.14.2.18 colElemFrame	83
7.14.2.19 colElemFrameGlow	83
7.14.2.20 colElemText	83
7.14.2.21 colElemTextGlow	83
7.14.2.22 eAction	83
7.14.2.23 eElemFlags	83
7.14.2.24 eEvent	84
7.14.2.25 eFontRefMode	84
7.14.2.26 eFontRefType	84
7.14.2.27 eImgFlags	84
7.14.2.28 eInitStatTouch	84
7.14.2.29 eTouch	84
7.14.2.30 eTxtAlign	84
7.14.2.31 eTxtFlags	84
7.14.2.32 eType	84
7.14.2.33 g	84
7.14.2.34 h	85
7.14.2.35 nActionVal	85
7.14.2.36 nDisp0H	85
7.14.2.37 nDisp0W	85
7.14.2.38 nDispDepth	85
7.14.2.39 nDispH	85
7.14.2.40 nDispW	85
7.14.2.41 nElemAutoldNext	85
7.14.2.42 nElemCnt	85
7.14.2.43 nElemIndFocused	85
7.14.2.44 nElemMax	86
7.14.2.45 nElemRefCnt	86

7.14.2.46 nElemRefMax	86
7.14.2.47 nFeatures	86
7.14.2.48 nFlipX	86
7.14.2.49 nFlipY	86
7.14.2.50 nFontCnt	86
7.14.2.51 nFontMax	86
7.14.2.52 nFrameRateCnt	86
7.14.2.53 nFrameRateStart	86
7.14.2.54 nGroup	87
7.14.2.55 nId	87
7.14.2.56 nId	87
7.14.2.57 nInputMapCnt	87
7.14.2.58 nInputMapMax	87
7.14.2.59 nPageCnt	87
7.14.2.60 nPageId	87
7.14.2.61 nPageMax	87
7.14.2.62 nRotation	87
7.14.2.63 nRoundRadius	87
7.14.2.64 nSize	88
7.14.2.65 nStrBufMax	88
7.14.2.66 nSubType	88
7.14.2.67 nSwapXY	88
7.14.2.68 nTouchCalXMax	88
7.14.2.69 nTouchCalXMin	88
7.14.2.70 nTouchCalYMax	88
7.14.2.71 nTouchCalYMin	88
7.14.2.72 nTouchLastPress	88
7.14.2.73 nTouchLastX	88
7.14.2.74 nTouchLastY	89
7.14.2.75 nTouchRotation	89

7.14.2.76 nTxtMarginX	89
7.14.2.77 nTxtMarginY	89
7.14.2.78 nType	89
7.14.2.79 nVal	89
7.14.2.80 nX	89
7.14.2.81 nY	89
7.14.2.82 pElem	89
7.14.2.83 pElemRefParent	89
7.14.2.84 pElemRefTracked	90
7.14.2.85 pFname	90
7.14.2.86 pfuncPinPoll	90
7.14.2.87 pfuncXDraw	90
7.14.2.88 pfuncXEvent	90
7.14.2.89 pfuncXTick	90
7.14.2.90 pfuncXTouch	90
7.14.2.91 plmgBuf	90
7.14.2.92 pStrBuf	90
7.14.2.93 pTxtFont	90
7.14.2.94 pvData	91
7.14.2.95 pvDriver	91
7.14.2.96 pvFont	91
7.14.2.97 pvImgRaw	91
7.14.2.98 pvScope	91
7.14.2.99 pXData	91
7.14.2.100r	91
7.14.2.101rBounds	91
7.14.2.102rElem	91
7.14.2.103rInvalidateRect	91
7.14.2.104sCollect	92
7.14.2.105sElemRefTmp	92

7.14.2.106	ElemTmp	92
7.14.2.107	ElemTmpProg	92
7.14.2.108	ImgRefBkgnd	92
7.14.2.109	ImgRefGlow	92
7.14.2.110	ImgRefNorm	92
7.14.2.111	TransCol	92
7.14.2.112	w	92
7.14.2.113	x	92
7.14.2.114	x	92
7.14.2.115	y	92
7.14.2.116	y	92
7.15	Internal: Misc Functions	93
7.15.1	Detailed Description	93
7.15.2	Function Documentation	93
7.15.2.1	gslc_ResetImage()	93
7.16	Internal: Element Functions	94
7.16.1	Detailed Description	95
7.16.2	Function Documentation	95
7.16.2.1	gslc_DrawTxtBase(gslc_tsGui *pGui, char *pStrBuf, gslc_tsRect rTxt, gslc_tsFont *pTxtFont, gslc_teTxtFlags eTxtFlags, int8_t eTxtAlign, gslc_tsColor colTxt, gslc_tsColor colBg, int16_t nMarginW, int16_t nMarginH)	95
7.16.2.2	gslc_ElemAdd(gslc_tsGui *pGui, int16_t nPageId, gslc_tsElem *pElem, gslc_teElemRefFlags eFlags)	95
7.16.2.3	gslc_ElemCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPageId, int16_t nType, gslc_tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)	96
7.16.2.4	gslc_ElemDraw(gslc_tsGui *pGui, int16_t nPageId, int16_t nElemId)	96
7.16.2.5	gslc_ElemDrawByRef(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)	97
7.16.2.6	gslc_ElemSetImage(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsImgRef sImgRef, gslc_tsImgRefSel sImgRefSel)	97
7.16.2.7	gslc_GetElemFromRef(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	97
7.16.2.8	gslc_GetElemFromRefD(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nLineNum)	98

7.16.2.9	<code>gslc_GetElemRefFlag(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t n↔ FlagMask)</code>	98
7.16.2.10	<code>gslc_GetXDataFromRef(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t↔ nType, int16_t nLineNum)</code>	98
7.16.2.11	<code>gslc_SetElemRefFlag(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t n↔ FlagMask, uint8_t nFlagVal)</code>	99
7.16.2.12	<code>gslc_SetRoundRadius(gslc_tsGui *pGui, uint8_t nRadius)</code>	99
7.17	Internal: Page Functions	100
7.17.1	Detailed Description	100
7.17.2	Function Documentation	100
7.17.2.1	<code>gslc_ElemEvent(void *pvGui, gslc_tsEvent sEvent)</code>	100
7.17.2.2	<code>gslc_ElemSendEventTouch(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef↔ Tracked, gslc_teTouch eTouch, int16_t nX, int16_t nY)</code>	101
7.17.2.3	<code>gslc_EventCreate(gslc_tsGui *pGui, gslc_teEventType eType, uint8_t nSubType, void *pvScope, void *pvData)</code>	101
7.17.2.4	<code>gslc_PageEvent(void *pvGui, gslc_tsEvent sEvent)</code>	102
7.17.2.5	<code>gslc_PageFindById(gslc_tsGui *pGui, int16_t nPageId)</code>	102
7.17.2.6	<code>gslc_PageFlipGet(gslc_tsGui *pGui)</code>	102
7.17.2.7	<code>gslc_PageFlipGo(gslc_tsGui *pGui)</code>	102
7.17.2.8	<code>gslc_PageFlipSet(gslc_tsGui *pGui, bool bNeeded)</code>	103
7.17.2.9	<code>gslc_PageFocusStep(gslc_tsGui *pGui, gslc_tsPage *pPage, bool bNext)</code>	103
7.17.2.10	<code>gslc_PageRedrawCalc(gslc_tsGui *pGui)</code>	103
7.17.2.11	<code>gslc_PageRedrawGo(gslc_tsGui *pGui)</code>	104
7.18	Internal: Element Collection Functions	105
7.18.1	Detailed Description	106
7.18.2	Function Documentation	106
7.18.2.1	<code>gslc_CollectElemAdd(gslc_tsGui *pGui, gslc_tsCollect *pCollect, const gslc_ts↔ Elem *pElem, gslc_teElemRefFlags eFlags)</code>	106
7.18.2.2	<code>gslc_CollectFindElemById(gslc_tsGui *pGui, gslc_tsCollect *pCollect, int16_t↔ nElemId)</code>	106
7.18.2.3	<code>gslc_CollectFindElemFromCoord(gslc_tsGui *pGui, gslc_tsCollect *pCollect, int16_t nX, int16_t nY)</code>	106
7.18.2.4	<code>gslc_CollectFindFocusStep(gslc_tsGui *pGui, gslc_tsCollect *pCollect, bool b↔ Next, bool *pbWrapped, int16_t *pnElemInd)</code>	107

7.18.2.5	<code>gslc_CollectGetElemRefTracked(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	107
7.18.2.6	<code>gslc_CollectGetFocus(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	107
7.18.2.7	<code>gslc_CollectGetNextId(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	108
7.18.2.8	<code>gslc_CollectGetRedraw(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	108
7.18.2.9	<code>gslc_CollectReset(gslc_tsCollect *pCollect, gslc_tsElem *asElem, uint16_t nElemMax, gslc_tsElemRef *asElemRef, uint16_t nElemRefMax)</code>	108
7.18.2.10	<code>gslc_CollectSetElemTracked(gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsElemRef *pElemRef)</code>	109
7.18.2.11	<code>gslc_CollectSetFocus(gslc_tsGui *pGui, gslc_tsCollect *pCollect, int16_t nElemInd)</code>	109
7.18.2.12	<code>gslc_CollectSetParent(gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsElemRef *pElemRefParent)</code>	109
7.19	Internal: Element Collection Event Functions	111
7.19.1	Detailed Description	111
7.19.2	Function Documentation	111
7.19.2.1	<code>gslc_CollectEvent(void *pvGui, gslc_tsEvent sEvent)</code>	111
7.19.2.2	<code>gslc_CollectInput(gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsEventTouch *pEventTouch)</code>	111
7.19.2.3	<code>gslc_CollectTouch(gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsEventTouch *pEventTouch)</code>	112
7.19.2.4	<code>gslc_CollectTouchCompound(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY, gslc_tsCollect *pCollect)</code>	112
7.20	Internal: Tracking Functions	113
7.20.1	Detailed Description	113
7.20.2	Function Documentation	113
7.20.2.1	<code>gslc_InputMapLookup(gslc_tsGui *pGui, gslc_teInputRawEvent eInputEvent, int16_t nInputVal, gslc_teAction *peAction, int16_t *pnActionVal)</code>	113
7.20.2.2	<code>gslc_TrackInput(gslc_tsGui *pGui, gslc_tsPage *pPage, gslc_teInputRawEvent eInputEvent, int16_t nInputVal)</code>	113
7.20.2.3	<code>gslc_TrackTouch(gslc_tsGui *pGui, gslc_tsPage *pPage, int16_t nX, int16_t nY, uint16_t nPress)</code>	114
7.21	Internal: Cleanup Functions	115
7.21.1	Detailed Description	115
7.21.2	Function Documentation	115
7.21.2.1	<code>gslc_CollectDestruct(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	115
7.21.2.2	<code>gslc_ElemDestruct(gslc_tsElem *pElem)</code>	116
7.21.2.3	<code>gslc_GuiDestruct(gslc_tsGui *pGui)</code>	117
7.21.2.4	<code>gslc_PageDestruct(gslc_tsGui *pGui, gslc_tsPage *pPage)</code>	117
7.21.2.5	<code>gslc_ResetElem(gslc_tsElem *pElem)</code>	117
7.21.2.6	<code>gslc_ResetFont(gslc_tsFont *pFont)</code>	118

8 Data Structure Documentation	119
8.1 gslc_tsCollect Struct Reference	119
8.1.1 Detailed Description	120
8.2 gslc_tsColor Struct Reference	120
8.2.1 Detailed Description	120
8.3 gslc_tsDriver Struct Reference	121
8.3.1 Field Documentation	121
8.3.1.1 nColBkgnd	121
8.3.1.2 rClipRect	121
8.4 gslc_tsElem Struct Reference	122
8.4.1 Detailed Description	123
8.5 gslc_tsElemRef Struct Reference	124
8.5.1 Detailed Description	124
8.6 gslc_tsEvent Struct Reference	124
8.6.1 Detailed Description	125
8.7 gslc_tsEventTouch Struct Reference	125
8.7.1 Detailed Description	125
8.8 gslc_tsFont Struct Reference	125
8.8.1 Detailed Description	126
8.9 gslc_tsGui Struct Reference	126
8.9.1 Detailed Description	128
8.10 gslc_tsImgRef Struct Reference	128
8.10.1 Detailed Description	129
8.11 gslc_tsInputMap Struct Reference	129
8.11.1 Detailed Description	129
8.12 gslc_tsPage Struct Reference	130
8.12.1 Detailed Description	130
8.13 gslc_tsPt Struct Reference	131
8.13.1 Detailed Description	131
8.14 gslc_tsRect Struct Reference	131

8.14.1 Detailed Description	131
8.15 gslc_tsXCheckbox Struct Reference	132
8.15.1 Detailed Description	132
8.15.2 Field Documentation	132
8.15.2.1 bChecked	132
8.15.2.2 bRadio	132
8.15.2.3 colCheck	133
8.15.2.4 nStyle	133
8.15.2.5 pfuncXToggle	133
8.16 gslc_tsXGauge Struct Reference	133
8.16.1 Detailed Description	134
8.16.2 Field Documentation	134
8.16.2.1 bFlip	134
8.16.2.2 bIndicFill	134
8.16.2.3 bValLastValid	135
8.16.2.4 bVert	135
8.16.2.5 colGauge	135
8.16.2.6 colTick	135
8.16.2.7 nIndicLen	135
8.16.2.8 nIndicTip	135
8.16.2.9 nMax	135
8.16.2.10 nMin	135
8.16.2.11 nStyle	135
8.16.2.12 nTickCnt	135
8.16.2.13 nTickLen	136
8.16.2.14 nVal	136
8.16.2.15 nValLast	136
8.17 gslc_tsXGlowball Struct Reference	136
8.17.1 Detailed Description	137
8.17.2 Field Documentation	137

8.17.2.1	colBg	137
8.17.2.2	nAngEnd	137
8.17.2.3	nAngStart	137
8.17.2.4	nMidX	137
8.17.2.5	nMidY	138
8.17.2.6	nNumRings	138
8.17.2.7	nQuality	138
8.17.2.8	nVal	138
8.17.2.9	nValLast	138
8.17.2.10	pRings	138
8.18	gslc_tsXGlowballRing Struct Reference	138
8.18.1	Field Documentation	139
8.18.1.1	cCol	139
8.18.1.2	nRad1	139
8.18.1.3	nRad2	139
8.19	gslc_tsXGraph Struct Reference	139
8.19.1	Detailed Description	140
8.19.2	Field Documentation	140
8.19.2.1	bScrollEn	140
8.19.2.2	colGraph	140
8.19.2.3	eStyle	141
8.19.2.4	nBufCnt	141
8.19.2.5	nBufMax	141
8.19.2.6	nMargin	141
8.19.2.7	nPlotIndMax	141
8.19.2.8	nPlotIndStart	141
8.19.2.9	nPlotValMax	141
8.19.2.10	nPlotValMin	141
8.19.2.11	nScrollPos	141
8.19.2.12	nWndHeight	141

8.19.2.13 nWndWidth	142
8.19.2.14 pBuf	142
8.20 gslc_tsXKeyPad Struct Reference	142
8.20.1 Detailed Description	143
8.20.2 Field Documentation	143
8.20.2.1 acValStr	143
8.20.2.2 bValDecimalPt	143
8.20.2.3 bValPositive	143
8.20.2.4 nSubElemMax	143
8.20.2.5 nTargetId	143
8.20.2.6 nValStrPos	143
8.20.2.7 pacKeys	144
8.20.2.8 pfuncCb	144
8.20.2.9 pfuncLookup	144
8.20.2.10 psElem	144
8.20.2.11 psElemRef	144
8.20.2.12 sCollect	144
8.20.2.13 sConfig	144
8.21 gslc_tsXKeyPad_Alpha Struct Reference	144
8.21.1 Field Documentation	145
8.21.1.1 asElem	145
8.21.1.2 asElemRef	145
8.21.1.3 sKeyPad	145
8.22 gslc_tsXKeyPad_Num Struct Reference	145
8.22.1 Field Documentation	146
8.22.1.1 asElem	146
8.22.1.2 asElemRef	146
8.22.1.3 sKeyPad	146
8.23 gslc_tsXKeyPadCfg Struct Reference	146
8.23.1 Detailed Description	147

8.23.2	Field Documentation	147
8.23.2.1	bFloatEn	147
8.23.2.2	bRoundEn	147
8.23.2.3	bSignEn	147
8.23.2.4	nButtonSzH	148
8.23.2.5	nButtonSzW	148
8.23.2.6	nFontId	148
8.23.2.7	nFrameMargin	148
8.23.2.8	nMaxCols	148
8.23.2.9	nMaxRows	148
8.23.2.10	nOffsetX	148
8.23.2.11	nOffsetY	148
8.23.2.12	pacKeys	148
8.24	gslc_tsXKeyPadData Struct Reference	149
8.24.1	Detailed Description	149
8.24.2	Field Documentation	149
8.24.2.1	nTargetId	149
8.24.2.2	pStr	149
8.25	gslc_tsXListbox Struct Reference	149
8.25.1	Detailed Description	150
8.25.2	Field Documentation	151
8.25.2.1	bItemAutoSizeH	151
8.25.2.2	bItemAutoSizeW	151
8.25.2.3	bNeedRecalc	151
8.25.2.4	colGap	151
8.25.2.5	nBufItemsMax	151
8.25.2.6	nBufItemsPos	151
8.25.2.7	nCols	151
8.25.2.8	nItemCnt	151
8.25.2.9	nItemCurSel	151

8.25.2.10 nItemCurSelLast	151
8.25.2.11 nItemGap	152
8.25.2.12 nItemH	152
8.25.2.13 nItemSavedSel	152
8.25.2.14 nItemTop	152
8.25.2.15 nItemW	152
8.25.2.16 nMarginH	152
8.25.2.17 nMarginW	152
8.25.2.18 nRows	152
8.25.2.19 pBufItems	152
8.25.2.20 pfuncXSel	152
8.26 gslc_tsXProgress Struct Reference	153
8.26.1 Detailed Description	153
8.26.2 Field Documentation	153
8.26.2.1 bFlip	153
8.26.2.2 bValLastValid	154
8.26.2.3 bVert	154
8.26.2.4 colGauge	154
8.26.2.5 nMax	154
8.26.2.6 nMin	154
8.26.2.7 nVal	154
8.26.2.8 nValLast	154
8.27 gslc_tsXRadial Struct Reference	154
8.27.1 Detailed Description	155
8.27.2 Field Documentation	155
8.27.2.1 bFlip	155
8.27.2.2 bIndicFill	155
8.27.2.3 bValLastValid	155
8.27.2.4 colGauge	156
8.27.2.5 colTick	156

8.27.2.6	nIndicLen	156
8.27.2.7	nIndicTip	156
8.27.2.8	nMax	156
8.27.2.9	nMin	156
8.27.2.10	nTickCnt	156
8.27.2.11	nTickLen	156
8.27.2.12	nVal	156
8.27.2.13	nValLast	156
8.28	gslc_tsXRamp Struct Reference	157
8.28.1	Detailed Description	157
8.28.2	Field Documentation	157
8.28.2.1	bValLastValid	157
8.28.2.2	nMax	157
8.28.2.3	nMin	157
8.28.2.4	nVal	157
8.28.2.5	nValLast	158
8.29	gslc_tsXRingGauge Struct Reference	158
8.29.1	Detailed Description	159
8.29.2	Field Documentation	159
8.29.2.1	acStrLast	159
8.29.2.2	bGradient	159
8.29.2.3	colRing1	159
8.29.2.4	colRing2	159
8.29.2.5	colRingRemain	159
8.29.2.6	nAngRange	159
8.29.2.7	nAngStart	159
8.29.2.8	nQuality	159
8.29.2.9	nSegGap	159
8.29.2.10	nThickness	159
8.29.2.11	nVal	159

8.29.2.12 nValLast	159
8.29.2.13 nValMax	159
8.29.2.14 nValMin	159
8.30 gslc_tsXSelNum Struct Reference	160
8.30.1 Detailed Description	160
8.30.2 Field Documentation	160
8.30.2.1 acElemTxt	160
8.30.2.2 asElem	160
8.30.2.3 asElemRef	161
8.30.2.4 nCounter	161
8.30.2.5 sCollect	161
8.31 gslc_tsXSlider Struct Reference	161
8.31.1 Detailed Description	162
8.31.2 Field Documentation	162
8.31.2.1 bTrim	162
8.31.2.2 bVert	162
8.31.2.3 colTick	162
8.31.2.4 colTrim	162
8.31.2.5 nPos	163
8.31.2.6 nPosMax	163
8.31.2.7 nPosMin	163
8.31.2.8 nThumbSz	163
8.31.2.9 nTickDiv	163
8.31.2.10 nTickLen	163
8.31.2.11 pfuncXPos	163
8.32 gslc_tsXSpinner Struct Reference	163
8.32.1 Detailed Description	164
8.32.2 Field Documentation	164
8.32.2.1 acDecr	164
8.32.2.2 acElemTxt	164

8.32.2.3	acIncr	164
8.32.2.4	asElem	165
8.32.2.5	asElemRef	165
8.32.2.6	nCounter	165
8.32.2.7	nIncr	165
8.32.2.8	nMax	165
8.32.2.9	nMin	165
8.32.2.10	pElemRef	165
8.32.2.11	pfuncXInput	165
8.32.2.12	sCollect	165
8.33	gslc_tsXTemplate Struct Reference	166
8.33.1	Detailed Description	166
8.34	gslc_tsXTextbox Struct Reference	166
8.34.1	Detailed Description	167
8.34.2	Field Documentation	167
8.34.2.1	bScrollEn	167
8.34.2.2	bWrapEn	167
8.34.2.3	nBufCols	167
8.34.2.4	nBufPosX	167
8.34.2.5	nBufPosY	167
8.34.2.6	nBufRows	167
8.34.2.7	nChSizeX	168
8.34.2.8	nChSizeY	168
8.34.2.9	nCurPosX	168
8.34.2.10	nCurPosY	168
8.34.2.11	nMarginX	168
8.34.2.12	nMarginY	168
8.34.2.13	nRedrawRow	168
8.34.2.14	nScrollPos	168
8.34.2.15	nWndCols	168

8.34.2.16 nWndRows	168
8.34.2.17 nWndRowStart	169
8.34.2.18 pBuf	169
8.35 THPoint Class Reference	169
8.35.1 Constructor & Destructor Documentation	169
8.35.1.1 THPoint(void)	169
8.35.1.2 THPoint(uint16_t x, uint16_t y, uint16_t z)	169
8.35.2 Member Function Documentation	169
8.35.2.1 operator!=(THPoint)	169
8.35.2.2 operator==(THPoint)	169
8.35.3 Field Documentation	169
8.35.3.1 x	169
8.35.3.2 y	169
8.35.3.3 z	169
8.36 TouchHandler Class Reference	170
8.36.1 Constructor & Destructor Documentation	170
8.36.1.1 TouchHandler()	170
8.36.2 Member Function Documentation	170
8.36.2.1 begin(void)	170
8.36.2.2 getPoint(void)	170
8.36.2.3 scale(THPoint pln)	171
8.36.2.4 setCalibration(uint16_t ts_xMin, uint16_t ts_xMax, uint16_t ts_yMin, uint16_t ts_yMax)	171
8.36.2.5 setSize(uint16_t _disp_xSize, uint16_t _disp_ySize)	171
8.36.2.6 setSwapFlip(bool _swapXY, bool _flipX, bool _flipY)	171
8.37 TouchHandler_XPT2046 Class Reference	171
8.37.1 Constructor & Destructor Documentation	172
8.37.1.1 TouchHandler_XPT2046(SPIClass &spi, uint8_t spi_cs_pin)	172
8.37.2 Member Function Documentation	172
8.37.2.1 begin(void)	172
8.37.2.2 getPoint(void)	172
8.37.3 Field Documentation	172
8.37.3.1 spi	172
8.37.3.2 touchDriver	172

9	File Documentation	173
9.1	README.md File Reference	173
9.2	src/elem/XCheckbox.c File Reference	173
9.2.1	Function Documentation	174
9.2.1.1	gslc_ElemXCheckboxCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXCheckbox *pXData, gslc_tsRect rElem, bool bRadio, gslc_teXCheckboxStyle nStyle, gslc_tsColor colCheck, bool bChecked)	174
9.2.1.2	gslc_ElemXCheckboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	174
9.2.1.3	gslc_ElemXCheckboxFindChecked(gslc_tsGui *pGui, int16_t nGroupId)	175
9.2.1.4	gslc_ElemXCheckboxGetState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	175
9.2.1.5	gslc_ElemXCheckboxSetState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bChecked)	175
9.2.1.6	gslc_ElemXCheckboxSetStateFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XCHECKBOX pfuncCb)	176
9.2.1.7	gslc_ElemXCheckboxSetStateHelp(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bChecked, bool bDoCb)	176
9.2.1.8	gslc_ElemXCheckboxToggleState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	176
9.2.1.9	gslc_ElemXCheckboxTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)	176
9.2.2	Variable Documentation	178
9.2.2.1	ERRSTR_NULL	178
9.2.2.2	ERRSTR_PXD_NULL	178
9.3	src/elem/XCheckbox.h File Reference	178
9.3.1	Macro Definition Documentation	179
9.3.1.1	gslc_ElemXCheckboxCreate_P	179
9.3.1.2	GSLC_TYPEX_CHECKBOX	180
9.3.2	Typedef Documentation	180
9.3.2.1	GSLC_CB_XCHECKBOX	180
9.3.3	Enumeration Type Documentation	180
9.3.3.1	gslc_teXCheckboxStyle	180
9.3.4	Function Documentation	181

9.3.4.1	<code>gslc_ElemXCheckboxCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXCheckbox *pXData, gslc_tsRect rElem, bool bRadio, gslc_teXCheckboxStyle nStyle, gslc_tsColor colCheck, bool bChecked)</code>	181
9.3.4.2	<code>gslc_ElemXCheckboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	182
9.3.4.3	<code>gslc_ElemXCheckboxFindChecked(gslc_tsGui *pGui, int16_t nGroupId)</code>	182
9.3.4.4	<code>gslc_ElemXCheckboxGetState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	183
9.3.4.5	<code>gslc_ElemXCheckboxSetState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bChecked)</code>	183
9.3.4.6	<code>gslc_ElemXCheckboxSetStateFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XCHECKBOX pfuncCb)</code>	183
9.3.4.7	<code>gslc_ElemXCheckboxToggleState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	183
9.3.4.8	<code>gslc_ElemXCheckboxTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	184
9.4	<code>src/elem/XGauge.c</code> File Reference	184
9.4.1	Function Documentation	186
9.4.1.1	<code>gslc_ElemXGaugeCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXGauge *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)</code>	186
9.4.1.2	<code>gslc_ElemXGaugeDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	186
9.4.1.3	<code>gslc_ElemXGaugeDrawProgressBar(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)</code>	187
9.4.1.4	<code>gslc_ElemXGaugeSetFlip(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)</code>	187
9.4.1.5	<code>gslc_ElemXGaugeSetIndicator(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)</code>	187
9.4.1.6	<code>gslc_ElemXGaugeSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teXGaugeStyle nStyle)</code>	188
9.4.1.7	<code>gslc_ElemXGaugeSetTicks(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colTick, uint16_t nTickCnt, uint16_t nTickLen)</code>	188
9.4.1.8	<code>gslc_ElemXGaugeUpdate(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)</code>	189
9.4.2	Variable Documentation	189
9.4.2.1	<code>ERRSTR_NULL</code>	189
9.4.2.2	<code>ERRSTR_PXD_NULL</code>	189
9.5	<code>src/elem/XGauge.h</code> File Reference	189

9.5.1	Macro Definition Documentation	191
9.5.1.1	gslc_ElemXGaugeCreate_P	191
9.5.1.2	GSLC_TYPEX_GAUGE	191
9.5.2	Enumeration Type Documentation	191
9.5.2.1	gslc_teXGaugeStyle	191
9.5.3	Function Documentation	192
9.5.3.1	gslc_ElemXGaugeCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXGauge *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)	192
9.5.3.2	gslc_ElemXGaugeDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	192
9.5.3.3	gslc_ElemXGaugeDrawProgressBar(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)	193
9.5.3.4	gslc_ElemXGaugeSetFlip(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)	193
9.5.3.5	gslc_ElemXGaugeSetIndicator(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)	193
9.5.3.6	gslc_ElemXGaugeSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teXGaugeStyle nType)	194
9.5.3.7	gslc_ElemXGaugeSetTicks(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colTick, uint16_t nTickCnt, uint16_t nTickLen)	194
9.5.3.8	gslc_ElemXGaugeUpdate(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	195
9.6	src/elem/XGlowball.c File Reference	195
9.6.1	Function Documentation	196
9.6.1.1	drawXGlowball(gslc_tsGui *pGui, gslc_tsXGlowball *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nVal, uint16_t nAngStart, uint16_t nAngEnd)	196
9.6.1.2	drawXGlowballArc(gslc_tsGui *pGui, gslc_tsXGlowball *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, gslc_tsColor cArc, uint16_t nAngStart, uint16_t nAngEnd)	196
9.6.1.3	drawXGlowballRing(gslc_tsGui *pGui, gslc_tsXGlowball *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nVal, uint16_t nAngStart, uint16_t nAngEnd, bool bErase)	196
9.6.1.4	gslc_ElemXGlowballCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXGlowball *pXData, int16_t nMidX, int16_t nMidY, gslc_tsXGlowballRing *pRings, uint8_t nNumRings)	196
9.6.1.5	gslc_ElemXGlowballDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	197

9.6.1.6	gslc_ElemXGlowballSetAngles(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nAngStart, int16_t nAngEnd)	197
9.6.1.7	gslc_ElemXGlowballSetColorBack(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colBg)	197
9.6.1.8	gslc_ElemXGlowballSetQuality(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint16_t nQuality)	197
9.6.1.9	gslc_ElemXGlowballSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	197
9.6.2	Variable Documentation	197
9.6.2.1	ERRSTR_NULL	197
9.6.2.2	ERRSTR_PXD_NULL	197
9.7	src/elem/XGlowball.h File Reference	197
9.7.1	Macro Definition Documentation	199
9.7.1.1	GSLC_TYPEX_GLOW	199
9.7.2	Function Documentation	199
9.7.2.1	drawXGlowball(gslc_tsGui *pGui, gslc_tsXGlowball *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nVal, uint16_t nAngStart, uint16_t nAngEnd)	199
9.7.2.2	drawXGlowballArc(gslc_tsGui *pGui, gslc_tsXGlowball *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, gslc_tsColor cArc, uint16_t nAngStart, uint16_t nAngEnd)	199
9.7.2.3	drawXGlowballRing(gslc_tsGui *pGui, gslc_tsXGlowball *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nVal, uint16_t nAngStart, uint16_t nAngEnd, bool bErase)	199
9.7.2.4	gslc_ElemXGlowballCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXGlowball *pXData, int16_t nMidX, int16_t nMidY, gslc_tsXGlowballRing *pRings, uint8_t nNumRings)	199
9.7.2.5	gslc_ElemXGlowballDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	199
9.7.2.6	gslc_ElemXGlowballSetAngles(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nAngStart, int16_t nAngEnd)	200
9.7.2.7	gslc_ElemXGlowballSetColorBack(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colBg)	200
9.7.2.8	gslc_ElemXGlowballSetQuality(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint16_t nQuality)	200
9.7.2.9	gslc_ElemXGlowballSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	200
9.8	src/elem/XGraph.c File Reference	200
9.8.1	Function Documentation	201

9.8.1.1	gslc_ElemXGraphAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	201
9.8.1.2	gslc_ElemXGraphCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXGraph *pXData, gslc_tsRect rElem, int16_t nFontId, int16_t *pBuf, uint16_t nBufMax, gslc_tsColor colGraph)	201
9.8.1.3	gslc_ElemXGraphDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔Redraw)	201
9.8.1.4	gslc_ElemXGraphScrollSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)	203
9.8.1.5	gslc_ElemXGraphSetRange(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nYMin, int16_t nYMax)	203
9.8.1.6	gslc_ElemXGraphSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc↔_teXGraphStyle eStyle, uint8_t nMargin)	203
9.8.2	Variable Documentation	204
9.8.2.1	ERRSTR_NULL	204
9.8.2.2	ERRSTR_PXD_NULL	204
9.9	src/elem/XGraph.h File Reference	204
9.9.1	Macro Definition Documentation	205
9.9.1.1	GSLC_TYPEX_GRAPH	205
9.9.2	Enumeration Type Documentation	205
9.9.2.1	gslc_teXGraphStyle	205
9.9.3	Function Documentation	205
9.9.3.1	gslc_ElemXGraphAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	205
9.9.3.2	gslc_ElemXGraphCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXGraph *pXData, gslc_tsRect rElem, int16_t nFontId, int16_t *pBuf, uint16_t nBufRows, gslc_tsColor colGraph)	206
9.9.3.3	gslc_ElemXGraphDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔Redraw)	206
9.9.3.4	gslc_ElemXGraphScrollSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)	207
9.9.3.5	gslc_ElemXGraphSetRange(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nYMin, int16_t nYMax)	207
9.9.3.6	gslc_ElemXGraphSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc↔_teXGraphStyle eStyle, uint8_t nMargin)	207
9.10	src/elem/XKeyPad.c File Reference	208
9.10.1	Function Documentation	209
9.10.1.1	ElemXKeyPadAddChar(gslc_tsGui *pGui, gslc_tsXKeyPad *pKeyPad, char ch)	209

9.10.1.2	<code>gslc_ElemXKeyPadCfgSetButtonSz(gslc_tsXKeyPadCfg *pConfig, int8_t nButtonSzW, int8_t nButtonSzH)</code>	209
9.10.1.3	<code>gslc_ElemXKeyPadCfgSetFloatEn(gslc_tsXKeyPadCfg *pConfig, bool bEn)</code>	210
9.10.1.4	<code>gslc_ElemXKeyPadCfgSetRoundEn(gslc_tsXKeyPadCfg *pConfig, bool bEn)</code>	210
9.10.1.5	<code>gslc_ElemXKeyPadCfgSetSignEn(gslc_tsXKeyPadCfg *pConfig, bool bEn)</code>	210
9.10.1.6	<code>gslc_ElemXKeyPadClick(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)</code>	211
9.10.1.7	<code>gslc_ElemXKeyPadCreateBase(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg *pConfig, XKEYPAD_CREATE pfuncCreate, XKEYPAD_LOOKUP pfuncLookup)</code>	211
9.10.1.8	<code>gslc_ElemXKeyPadDataTargetIdGet(gslc_tsGui *pGui, void *pvData)</code>	212
9.10.1.9	<code>gslc_ElemXKeyPadDataValGet(gslc_tsGui *pGui, void *pvData)</code>	212
9.10.1.10	<code>gslc_ElemXKeyPadDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	212
9.10.1.11	<code>gslc_ElemXKeyPadInputAsk(gslc_tsGui *pGui, gslc_tsElemRef *pKeyPadRef, int16_t nPgPopup, gslc_tsElemRef *pTxtRef)</code>	212
9.10.1.12	<code>gslc_ElemXKeyPadInputGet(gslc_tsGui *pGui, gslc_tsElemRef *pTxtRef, void *pvCbData)</code>	213
9.10.1.13	<code>gslc_ElemXKeyPadSetFloatEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bEn)</code>	213
9.10.1.14	<code>gslc_ElemXKeyPadSetSignEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bEn)</code>	213
9.10.1.15	<code>gslc_ElemXKeyPadTargetIdSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nTargetId)</code>	214
9.10.1.16	<code>gslc_ElemXKeyPadTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	214
9.10.1.17	<code>gslc_ElemXKeyPadValGet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, char *pStrBuf, uint8_t nStrBufLen)</code>	215
9.10.1.18	<code>gslc_ElemXKeyPadValSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, const char *pStrBuf)</code>	215
9.10.1.19	<code>gslc_ElemXKeyPadValSetCb(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, G_SLC_CB_INPUT pfuncCb)</code>	215
9.10.1.20	<code>gslc_ElemXKeyPadValSetSign(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bPositive)</code>	216
9.10.1.21	<code>XKeyPadAddKeyElem(gslc_tsGui *pGui, gslc_tsXKeyPad *pXData, int16_t nKeyId, bool bTxtField, int16_t nRow, int16_t nCol, int8_t nRowSpan, int8_t nColSpan, gslc_tsColor cColFill, gslc_tsColor cColGlow, bool bVisible)</code>	216
9.10.2	Variable Documentation	216

9.10.2.1	ERRSTR_NULL	216
9.10.2.2	ERRSTR_PXD_NULL	216
9.10.2.3	KEYPAD_DISP_DECIMAL_PT	216
9.10.2.4	KEYPAD_DISP_NEGATIVE	216
9.11	src/elem/XKeyPad.h File Reference	216
9.11.1	Macro Definition Documentation	219
9.11.1.1	GSLC_TYPEX_KEYPAD	219
9.11.1.2	XKEYPAD_CB_STATE_CANCEL	219
9.11.1.3	XKEYPAD_CB_STATE_DONE	219
9.11.1.4	XKEYPAD_CB_STATE_UPDATE	219
9.11.1.5	XKEYPAD_VAL_LEN	219
9.11.2	Typedef Documentation	219
9.11.2.1	XKEYPAD_CREATE	219
9.11.2.2	XKEYPAD_LOOKUP	219
9.11.3	Enumeration Type Documentation	219
9.11.3.1	anonymous enum	219
9.11.4	Function Documentation	219
9.11.4.1	gslc_ElemXKeyPadCfgSetButtonSz(gslc_tsXKeyPadCfg *pConfig, int8_t nButtonSzW, int8_t nButtonSzH)	219
9.11.4.2	gslc_ElemXKeyPadCfgSetFloatEn(gslc_tsXKeyPadCfg *pConfig, bool bEn)	220
9.11.4.3	gslc_ElemXKeyPadCfgSetRoundEn(gslc_tsXKeyPadCfg *pConfig, bool bEn)	220
9.11.4.4	gslc_ElemXKeyPadCfgSetSignEn(gslc_tsXKeyPadCfg *pConfig, bool bEn)	220
9.11.4.5	gslc_ElemXKeyPadClick(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)	221
9.11.4.6	gslc_ElemXKeyPadCreateBase(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg *pConfig, XKEYPAD_CREATE pfuncCreate, XKEYPAD_LOOKUP pfuncLookup)	221
9.11.4.7	gslc_ElemXKeyPadDataTargetIdGet(gslc_tsGui *pGui, void *pvData)	222
9.11.4.8	gslc_ElemXKeyPadDataValGet(gslc_tsGui *pGui, void *pvData)	222
9.11.4.9	gslc_ElemXKeyPadDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	222

9.11.4.10	<code>gslc_ElemXKeyPadInputAsk(gslc_tsGui *pGui, gslc_tsElemRef *pKeyPadRef, int16_t nPgPopup, gslc_tsElemRef *pTxtRef)</code>	223
9.11.4.11	<code>gslc_ElemXKeyPadInputGet(gslc_tsGui *pGui, gslc_tsElemRef *pTxtRef, void *pvCbData)</code>	223
9.11.4.12	<code>gslc_ElemXKeyPadSetFloatEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bEn)</code>	223
9.11.4.13	<code>gslc_ElemXKeyPadSetSignEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bEn)</code>	224
9.11.4.14	<code>gslc_ElemXKeyPadTargetIdSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nId)</code>	224
9.11.4.15	<code>gslc_ElemXKeyPadTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	224
9.11.4.16	<code>gslc_ElemXKeyPadValGet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, char *pStrBuf, uint8_t nStrBufMax)</code>	225
9.11.4.17	<code>gslc_ElemXKeyPadValSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, const char *pStrBuf)</code>	225
9.11.4.18	<code>gslc_ElemXKeyPadValSetCb(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, G↵ SLC_CB_INPUT pfuncCb)</code>	225
9.11.4.19	<code>XKeyPadAddKeyElem(gslc_tsGui *pGui, gslc_tsXKeyPad *pXData, int16_t n↵ KeyId, bool bTxtField, int16_t nRow, int16_t nCol, int8_t nRowSpan, int8_t n↵ ColSpan, gslc_tsColor cColFill, gslc_tsColor cColGlow, bool bVisible)</code>	226
9.12	<code>src/elem/XKeyPad_Alpha.c</code> File Reference	226
9.12.1	Enumeration Type Documentation	227
9.12.1.1	anonymous enum	227
9.12.2	Function Documentation	228
9.12.2.1	<code>gslc_ElemXKeyPadCfgInit_Alpha()</code>	228
9.12.2.2	<code>gslc_ElemXKeyPadCreate_Alpha(gslc_tsGui *pGui, int16_t nElemId, int16_t n↵ Page, gslc_tsXKeyPad_Alpha *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg *pConfig)</code>	228
9.12.2.3	<code>XKeyPadCreateKeys_Alpha(gslc_tsGui *pGui, gslc_tsXKeyPad *pXData)</code>	228
9.12.2.4	<code>XKeyPadLookup_Alpha(gslc_tsGui *pGui, int16_t nKeyId)</code>	228
9.12.3	Variable Documentation	228
9.12.3.1	<code>ERRSTR_NULL</code>	228
9.12.3.2	<code>ERRSTR_PXD_NULL</code>	228
9.12.3.3	<code>KEYPAD_LABEL_STRINGS</code>	229
9.13	<code>src/elem/XKeyPad_Alpha.h</code> File Reference	229

9.13.1	Macro Definition Documentation	230
9.13.1.1	XKEYPADALPHA_BTN_BASIC	230
9.13.1.2	XKEYPADALPHA_ELEM_MAX	230
9.13.2	Function Documentation	230
9.13.2.1	gslc_ElemXKeyPadCfgInit_Alpha()	230
9.13.2.2	gslc_ElemXKeyPadCreate_Alpha(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad_Alpha *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg *pConfig)	230
9.14	src/elem/XKeyPad_Num.c File Reference	231
9.14.1	Enumeration Type Documentation	232
9.14.1.1	anonymous enum	232
9.14.2	Function Documentation	232
9.14.2.1	gslc_ElemXKeyPadCfgInit_Num()	232
9.14.2.2	gslc_ElemXKeyPadCreate_Num(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad_Num *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg *pConfig)	232
9.14.2.3	XKeyPadCreateKeys_Num(gslc_tsGui *pGui, gslc_tsXKeyPad *pXData)	233
9.14.2.4	XKeyPadLookup_Num(gslc_tsGui *pGui, int16_t nKeyId)	233
9.14.3	Variable Documentation	233
9.14.3.1	ERRSTR_NULL	233
9.14.3.2	ERRSTR_PXD_NULL	233
9.14.3.3	KEYPAD_LABEL_STRINGS	233
9.15	src/elem/XKeyPad_Num.h File Reference	233
9.15.1	Macro Definition Documentation	234
9.15.1.1	XKEYPADNUM_BTN_BASIC	234
9.15.1.2	XKEYPADNUM_ELEM_MAX	234
9.15.2	Function Documentation	234
9.15.2.1	gslc_ElemXKeyPadCfgInit_Num()	234
9.15.2.2	gslc_ElemXKeyPadCreate_Num(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad_Num *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg *pConfig)	234
9.16	src/elem/XListbox.c File Reference	235
9.16.1	Macro Definition Documentation	236

9.16.1.1	XLISTBOX_MAX_STR	236
9.16.2	Function Documentation	236
9.16.2.1	gslc_ElemXListboxAddItem(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, const char *pStrItem)	236
9.16.2.2	gslc_ElemXListboxCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXListbox *pXData, gslc_tsRect rElem, int16_t nFontId, uint8_t *pBufItems, uint16_t nBufItemsMax, int16_t nItemDefault)	237
9.16.2.3	gslc_ElemXListboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	237
9.16.2.4	gslc_ElemXListboxGetItem(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nItemCurSel, char *pStrItem, uint8_t nStrItemLen)	237
9.16.2.5	gslc_ElemXListboxGetItemCnt(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	238
9.16.2.6	gslc_ElemXListboxGetSel(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	238
9.16.2.7	gslc_ElemXListboxItemsSetGap(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nGap, gslc_tsColor colGap)	238
9.16.2.8	gslc_ElemXListboxItemsSetSize(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nItemW, int16_t nItemH)	239
9.16.2.9	gslc_ElemXListboxRecalcSize(gslc_tsXListbox *pListbox, gslc_tsRect rElem)	239
9.16.2.10	gslc_ElemXListboxReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	239
9.16.2.11	gslc_ElemXListboxSetMargin(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nMarginW, int8_t nMarginH)	239
9.16.2.12	gslc_ElemXListboxSetScrollPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint16_t nScrollPos)	240
9.16.2.13	gslc_ElemXListboxSetSel(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nItemCurSel)	240
9.16.2.14	gslc_ElemXListboxSetSelFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XLISTBOX_SEL funcCb)	240
9.16.2.15	gslc_ElemXListboxSetSize(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nRows, int8_t nCols)	241
9.16.2.16	gslc_ElemXListboxTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)	241
9.16.3	Variable Documentation	241
9.16.3.1	ERRSTR_NULL	241
9.16.3.2	ERRSTR_PXD_NULL	241
9.17	src/elem/XListbox.h File Reference	242
9.17.1	Macro Definition Documentation	243

9.17.1.1	GSLC_TYPEX_LISTBOX	243
9.17.1.2	XLISTBOX_BUF_OH_R	243
9.17.1.3	XLISTBOX_SEL_NONE	243
9.17.1.4	XLISTBOX_SIZE_AUTO	243
9.17.2	Typedef Documentation	243
9.17.2.1	GSLC_CB_XLISTBOX_SEL	243
9.17.3	Function Documentation	244
9.17.3.1	gslc_ElemXListboxAddItem(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, const char *pStrItem)	244
9.17.3.2	gslc_ElemXListboxCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXListbox *pXData, gslc_tsRect rElem, int16_t nFontId, uint8_t *pBufItems, uint16_t nBufItemsMax, int16_t nSelDefault)	244
9.17.3.3	gslc_ElemXListboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	244
9.17.3.4	gslc_ElemXListboxGetItem(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nItemCurSel, char *pStrItem, uint8_t nStrItemLen)	245
9.17.3.5	gslc_ElemXListboxGetItemCnt(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	245
9.17.3.6	gslc_ElemXListboxGetSel(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	245
9.17.3.7	gslc_ElemXListboxItemsSetGap(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nGap, gslc_tsColor colGap)	246
9.17.3.8	gslc_ElemXListboxItemsSetSize(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nItemW, int16_t nItemH)	246
9.17.3.9	gslc_ElemXListboxReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	246
9.17.3.10	gslc_ElemXListboxSetMargin(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nMarginW, int8_t nMarginH)	247
9.17.3.11	gslc_ElemXListboxSetScrollPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint16_t nScrollPos)	247
9.17.3.12	gslc_ElemXListboxSetSel(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nItemCurSel)	247
9.17.3.13	gslc_ElemXListboxSetSelFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XLISTBOX_SEL funcCb)	248
9.17.3.14	gslc_ElemXListboxSetSize(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nRows, int8_t nCols)	248
9.17.3.15	gslc_ElemXListboxTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)	248
9.18	src/elem/XProgress.c File Reference	249

9.18.1	Function Documentation	250
9.18.1.1	gslc_ElemXProgressCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXProgress *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)	250
9.18.1.2	gslc_ElemXProgressDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	250
9.18.1.3	gslc_ElemXProgressDrawHelp(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)	251
9.18.1.4	gslc_ElemXProgressSetFlip(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)	251
9.18.1.5	gslc_ElemXProgressSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	251
9.18.2	Variable Documentation	252
9.18.2.1	ERRSTR_NULL	252
9.18.2.2	ERRSTR_PXD_NULL	252
9.19	src/elem/XProgress.h File Reference	252
9.19.1	Macro Definition Documentation	253
9.19.1.1	gslc_ElemXProgressCreate_P	253
9.19.1.2	GSLC_TYPEX_PROGRESS	254
9.19.2	Function Documentation	254
9.19.2.1	gslc_ElemXProgressCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXProgress *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)	254
9.19.2.2	gslc_ElemXProgressDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	254
9.19.2.3	gslc_ElemXProgressDrawHelp(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)	255
9.19.2.4	gslc_ElemXProgressSetFlip(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)	255
9.19.2.5	gslc_ElemXProgressSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	255
9.20	src/elem/XRadial.c File Reference	256
9.20.1	Function Documentation	257
9.20.1.1	gslc_ElemXRadialCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXRadial *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge)	257

9.20.1.2	<code>gslc_ElemXRadialDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔ Redraw)</code>	257
9.20.1.3	<code>gslc_ElemXRadialDrawRadial(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)</code>	258
9.20.1.4	<code>gslc_ElemXRadialDrawRadialHelp(gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nArrowLen, uint16_t nArrowSz, int16_t n64Ang, bool bFill, gslc_tsColor colFrame)</code>	258
9.20.1.5	<code>gslc_ElemXRadialSetFlip(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)</code>	258
9.20.1.6	<code>gslc_ElemXRadialSetIndicator(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)</code>	259
9.20.1.7	<code>gslc_ElemXRadialSetTicks(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc↔ _tsColor colTick, uint16_t nTickCnt, uint16_t nTickLen)</code>	259
9.20.1.8	<code>gslc_ElemXRadialSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)</code>	259
9.20.2	Variable Documentation	260
9.20.2.1	ERRSTR_NULL	260
9.20.2.2	ERRSTR_PXD_NULL	260
9.21	src/elem/XRadial.h File Reference	260
9.21.1	Macro Definition Documentation	261
9.21.1.1	<code>gslc_ElemXRadialCreate_P</code>	261
9.21.1.2	GSLC_TYPEX_RADIAL	262
9.21.2	Function Documentation	262
9.21.2.1	<code>gslc_ElemXRadialCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXRadial *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge)</code>	262
9.21.2.2	<code>gslc_ElemXRadialDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔ Redraw)</code>	262
9.21.2.3	<code>gslc_ElemXRadialDrawRadial(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)</code>	263
9.21.2.4	<code>gslc_ElemXRadialSetFlip(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)</code>	263
9.21.2.5	<code>gslc_ElemXRadialSetIndicator(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)</code>	264
9.21.2.6	<code>gslc_ElemXRadialSetTicks(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc↔ _tsColor colTick, uint16_t nTickCnt, uint16_t nTickLen)</code>	264
9.21.2.7	<code>gslc_ElemXRadialSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)</code>	264

9.22 src/elem/XRamp.c File Reference	265
9.22.1 Function Documentation	266
9.22.1.1 gslc_ElemXRampCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXRamp *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)	266
9.22.1.2 gslc_ElemXRampDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔Redraw)	266
9.22.1.3 gslc_ElemXRampDrawHelp(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)	267
9.22.1.4 gslc_ElemXRampSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	267
9.22.2 Variable Documentation	267
9.22.2.1 ERRSTR_NULL	267
9.22.2.2 ERRSTR_PXD_NULL	267
9.23 src/elem/XRamp.h File Reference	268
9.23.1 Macro Definition Documentation	269
9.23.1.1 gslc_ElemXRampCreate_P	269
9.23.1.2 GSLC_TYPEX_RAMP	269
9.23.2 Function Documentation	269
9.23.2.1 gslc_ElemXRampCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXRamp *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)	269
9.23.2.2 gslc_ElemXRampDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔Redraw)	271
9.23.2.3 gslc_ElemXRampDrawHelp(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)	271
9.23.2.4 gslc_ElemXRampSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	272
9.24 src/elem/XRingGauge.c File Reference	272
9.24.1 Function Documentation	273
9.24.1.1 gslc_ElemXRingGaugeCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXRingGauge *pXData, gslc_tsRect rElem, char *pStrBuf, uint8_t nStr↔BufMax, int16_t nFontId)	273
9.24.1.2 gslc_ElemXRingGaugeDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	274
9.24.1.3 gslc_ElemXRingGaugeSetAngleRange(gslc_tsGui *pGui, gslc_tsElemRef *p↔ElemRef, int16_t nStart, int16_t nRange, bool bClockwise)	274

9.24.1.4	<code>gslc_ElemXRingGaugeSetColorActiveFlat(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colActive)</code>	275
9.24.1.5	<code>gslc_ElemXRingGaugeSetColorActiveGradient(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colStart, gslc_tsColor colEnd)</code>	275
9.24.1.6	<code>gslc_ElemXRingGaugeSetColorInactive(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor collnactive)</code>	275
9.24.1.7	<code>gslc_ElemXRingGaugeSetQuality(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint16_t nSegments)</code>	276
9.24.1.8	<code>gslc_ElemXRingGaugeSetThickness(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nThickness)</code>	276
9.24.1.9	<code>gslc_ElemXRingGaugeSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)</code>	277
9.24.1.10	<code>gslc_ElemXRingGaugeSetValRange(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nValMin, int16_t nValMax)</code>	277
9.24.2	Variable Documentation	277
9.24.2.1	<code>ERRSTR_NULL</code>	277
9.24.2.2	<code>ERRSTR_PXD_NULL</code>	277
9.25	<code>src/elem/XRingGauge.h</code> File Reference	278
9.25.1	Macro Definition Documentation	279
9.25.1.1	<code>GSLC_TYPEX_RING</code>	279
9.25.1.2	<code>XRING_STR_MAX</code>	279
9.25.2	Function Documentation	279
9.25.2.1	<code>gslc_ElemXRingGaugeCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXRingGauge *pXData, gslc_tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)</code>	279
9.25.2.2	<code>gslc_ElemXRingGaugeDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	280
9.25.2.3	<code>gslc_ElemXRingGaugeSetAngleRange(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nStart, int16_t nRange, bool bClockwise)</code>	280
9.25.2.4	<code>gslc_ElemXRingGaugeSetColorActiveFlat(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colActive)</code>	281
9.25.2.5	<code>gslc_ElemXRingGaugeSetColorActiveGradient(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colStart, gslc_tsColor colEnd)</code>	281
9.25.2.6	<code>gslc_ElemXRingGaugeSetColorInactive(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor collnactive)</code>	281
9.25.2.7	<code>gslc_ElemXRingGaugeSetQuality(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint16_t nSegments)</code>	282

9.25.2.8	<code>gslc_ElemXRingGaugeSetThickness(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nThickness)</code>	282
9.25.2.9	<code>gslc_ElemXRingGaugeSetVal(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)</code>	283
9.25.2.10	<code>gslc_ElemXRingGaugeSetValRange(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nValMin, int16_t nValMax)</code>	283
9.26	<code>src/elem/XSelNum.c</code> File Reference	283
9.26.1	Function Documentation	284
9.26.1.1	<code>gslc_ElemXSelNumClick(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)</code>	284
9.26.1.2	<code>gslc_ElemXSelNumCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSelNum *pXData, gslc_tsRect rElem, int8_t nFontId)</code>	285
9.26.1.3	<code>gslc_ElemXSelNumDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	285
9.26.1.4	<code>gslc_ElemXSelNumGetCounter(gslc_tsGui *pGui, gslc_tsXSelNum *pSelNum)</code>	286
9.26.1.5	<code>gslc_ElemXSelNumSetCounter(gslc_tsGui *pGui, gslc_tsXSelNum *pSelNum, int16_t nCount)</code>	286
9.26.1.6	<code>gslc_ElemXSelNumTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	286
9.26.2	Variable Documentation	287
9.26.2.1	<code>ERRSTR_NULL</code>	287
9.26.2.2	<code>ERRSTR_PXD_NULL</code>	287
9.26.2.3	<code>SELNUM_ID_BTN_DEC</code>	287
9.26.2.4	<code>SELNUM_ID_BTN_INC</code>	287
9.26.2.5	<code>SELNUM_ID_TXT</code>	287
9.27	<code>src/elem/XSelNum.h</code> File Reference	287
9.27.1	Macro Definition Documentation	288
9.27.1.1	<code>GSLC_TYPEX_SELNUM</code>	288
9.27.1.2	<code>SELNUM_STR_LEN</code>	288
9.27.2	Function Documentation	288
9.27.2.1	<code>gslc_ElemXSelNumClick(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)</code>	288
9.27.2.2	<code>gslc_ElemXSelNumCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSelNum *pXData, gslc_tsRect rElem, int8_t nFontId)</code>	289

9.27.2.3	<code>gslc_ElemXSelNumDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↵ Redraw)</code>	289
9.27.2.4	<code>gslc_ElemXSelNumGetCounter(gslc_tsGui *pGui, gslc_tsXSelNum *pSelNum)</code>	289
9.27.2.5	<code>gslc_ElemXSelNumSetCounter(gslc_tsGui *pGui, gslc_tsXSelNum *pSelNum, int16_t nCount)</code>	290
9.27.2.6	<code>gslc_ElemXSelNumTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	290
9.28	<code>src/elem/XSlider.c</code> File Reference	290
9.28.1	Function Documentation	291
9.28.1.1	<code>gslc_ElemXSliderCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSlider *pXData, gslc_tsRect rElem, int16_t nPosMin, int16_t nPosMax, int16_t nPos, uint16_t nThumbSz, bool bVert)</code>	291
9.28.1.2	<code>gslc_ElemXSliderDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↵ Redraw)</code>	292
9.28.1.3	<code>gslc_ElemXSliderGetPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	292
9.28.1.4	<code>gslc_ElemXSliderSetPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nPos)</code>	293
9.28.1.5	<code>gslc_ElemXSliderSetPosFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XSLIDER_POS funcCb)</code>	293
9.28.1.6	<code>gslc_ElemXSliderSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bTrim, gslc_tsColor colTrim, uint16_t nTickDiv, int16_t nTickLen, gslc_tsColor colTick)</code>	293
9.28.1.7	<code>gslc_ElemXSliderTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	294
9.28.2	Variable Documentation	294
9.28.2.1	<code>ERRSTR_NULL</code>	294
9.28.2.2	<code>ERRSTR_PXD_NULL</code>	294
9.29	<code>src/elem/XSlider.h</code> File Reference	294
9.29.1	Macro Definition Documentation	296
9.29.1.1	<code>gslc_ElemXSliderCreate_P</code>	296
9.29.1.2	<code>GSLC_TYPEX_SLIDER</code>	296
9.29.2	Typedef Documentation	296
9.29.2.1	<code>GSLC_CB_XSLIDER_POS</code>	296
9.29.3	Function Documentation	296

9.29.3.1	<code>gslc_ElemXSliderCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSlider *pXData, gslc_tsRect rElem, int16_t nPosMin, int16_t nPosMax, int16_t nPos, uint16_t nThumbSz, bool bVert)</code>	296
9.29.3.2	<code>gslc_ElemXSliderDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↵Redraw)</code>	297
9.29.3.3	<code>gslc_ElemXSliderGetPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	297
9.29.3.4	<code>gslc_ElemXSliderSetPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nPos)</code>	298
9.29.3.5	<code>gslc_ElemXSliderSetPosFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XSLIDER_POS funcCb)</code>	298
9.29.3.6	<code>gslc_ElemXSliderSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bTrim, gslc_tsColor colTrim, uint16_t nTickDiv, int16_t nTickLen, gslc_tsColor colTick)</code>	298
9.29.3.7	<code>gslc_ElemXSliderTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	299
9.30	<code>src/elem/XSpinner.c</code> File Reference	299
9.30.1	Function Documentation	300
9.30.1.1	<code>gslc_ElemXSpinnerClick(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)</code>	300
9.30.1.2	<code>gslc_ElemXSpinnerCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSpinner *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16↵_t nVal, int16_t nIncr, int8_t nFontId, int8_t nButtonSz, GSLC_CB_INPUT cbInput)</code>	301
9.30.1.3	<code>gslc_ElemXSpinnerDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↵Redraw)</code>	301
9.30.1.4	<code>gslc_ElemXSpinnerGetCounter(gslc_tsGui *pGui, gslc_tsXSpinner *pSpinner)</code>	301
9.30.1.5	<code>gslc_ElemXSpinnerSetChars(void *pvGui, gslc_tsElemRef *pElemRef, uint8_t↵cIncr, uint8_t cDecr)</code>	302
9.30.1.6	<code>gslc_ElemXSpinnerSetCounter(gslc_tsGui *pGui, gslc_tsXSpinner *pSpinner, int16_t nCount)</code>	302
9.30.1.7	<code>gslc_ElemXSpinnerTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	302
9.30.2	Variable Documentation	304
9.30.2.1	<code>ERRSTR_NULL</code>	304
9.30.2.2	<code>ERRSTR_PXD_NULL</code>	304
9.30.2.3	<code>SPINNER_ID_BTN_DEC</code>	304
9.30.2.4	<code>SPINNER_ID_BTN_INC</code>	304
9.30.2.5	<code>SPINNER_ID_TXT</code>	304

9.31	src/elem/XSpinner.h File Reference	304
9.31.1	Macro Definition Documentation	306
9.31.1.1	GSLC_TYPEX_SPINNER	306
9.31.1.2	XSPINNER_CB_STATE_UPDATE	306
9.31.1.3	XSPINNER_COMP_CNT	306
9.31.1.4	XSPINNER_STR_LEN	306
9.31.2	Function Documentation	306
9.31.2.1	gslc_ElemXSpinnerClick(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)	306
9.31.2.2	gslc_ElemXSpinnerCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSpinner *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, int16_t nIncr, int8_t nFontId, int8_t nButtonSz, GSLC_CB_INPUT cbInput)	306
9.31.2.3	gslc_ElemXSpinnerDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	307
9.31.2.4	gslc_ElemXSpinnerGetCounter(gslc_tsGui *pGui, gslc_tsXSpinner *pSpinner)	307
9.31.2.5	gslc_ElemXSpinnerSetChars(void *pvGui, gslc_tsElemRef *pElemRef, uint8_t cIncr, uint8_t cDecr)	307
9.31.2.6	gslc_ElemXSpinnerSetCounter(gslc_tsGui *pGui, gslc_tsXSpinner *pSpinner, int16_t nCount)	308
9.31.2.7	gslc_ElemXSpinnerTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)	308
9.32	src/elem/XTemplate.c File Reference	309
9.32.1	Function Documentation	309
9.32.1.1	gslc_ElemXTemplateCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXTemplate *pXData, gslc_tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)	309
9.32.1.2	gslc_ElemXTemplateDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	310
9.32.1.3	gslc_ElemXTemplateTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)	310
9.32.2	Variable Documentation	311
9.32.2.1	ERRSTR_NULL	311
9.32.2.2	ERRSTR_PXD_NULL	311
9.33	src/elem/XTemplate.h File Reference	311
9.33.1	Macro Definition Documentation	312

9.33.1.1	<code>GSLC_TYPEX_TEMPLATE</code>	312
9.33.2	Function Documentation	312
9.33.2.1	<code>gslc_ElemXTemplateCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXTemplate *pXData, gslc_tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)</code>	312
9.33.2.2	<code>gslc_ElemXTemplateDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	312
9.33.2.3	<code>gslc_ElemXTemplateTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	313
9.34	<code>src/elem/XTextbox.c</code> File Reference	313
9.34.1	Function Documentation	315
9.34.1.1	<code>gslc_ElemXTextboxAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, char *pTxt)</code>	315
9.34.1.2	<code>gslc_ElemXTextboxBufAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned char chNew, bool bAdvance)</code>	315
9.34.1.3	<code>gslc_ElemXTextboxColReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	315
9.34.1.4	<code>gslc_ElemXTextboxColSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor nCol)</code>	315
9.34.1.5	<code>gslc_ElemXTextboxCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXTextbox *pXData, gslc_tsRect rElem, int16_t nFontId, char *pBuf, uint16_t nBufRows, uint16_t nBufCols)</code>	316
9.34.1.6	<code>gslc_ElemXTextboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	316
9.34.1.7	<code>gslc_ElemXTextboxLineWrAdv(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	317
9.34.1.8	<code>gslc_ElemXTextboxReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	317
9.34.1.9	<code>gslc_ElemXTextboxScrollSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)</code>	317
9.34.1.10	<code>gslc_ElemXTextboxWrapSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bWrapEn)</code>	317
9.34.2	Variable Documentation	318
9.34.2.1	<code>ERRSTR_NULL</code>	318
9.34.2.2	<code>ERRSTR_PXD_NULL</code>	318
9.35	<code>src/elem/XTextbox.h</code> File Reference	318
9.35.1	Macro Definition Documentation	319
9.35.1.1	<code>GSLC_TYPEX_TEXTBOX</code>	319
9.35.1.2	<code>GSLC_XTEXTBOX_CODE_COL_RESET</code>	319

9.35.1.3	gslc_XTEXTBOX_CODE_COL_SET	319
9.35.1.4	XTEXTBOX_REDRAW_ALL	319
9.35.1.5	XTEXTBOX_REDRAW_NONE	319
9.35.2	Function Documentation	319
9.35.2.1	gslc_ElemXTextboxAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, char *pTxt)	319
9.35.2.2	gslc_ElemXTextboxColReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef) . .	320
9.35.2.3	gslc_ElemXTextboxColSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor nCol)	320
9.35.2.4	gslc_ElemXTextboxCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXTextbox *pXData, gslc_tsRect rElem, int16_t nFontId, char *pBuf, uint16_t nBufRows, uint16_t nBufCols)	320
9.35.2.5	gslc_ElemXTextboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	321
9.35.2.6	gslc_ElemXTextboxReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	321
9.35.2.7	gslc_ElemXTextboxScrollSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)	322
9.35.2.8	gslc_ElemXTextboxWrapSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bWrapEn)	322
9.36	src/GUIslice.c File Reference	322
9.36.1	Enumeration Type Documentation	331
9.36.1.1	gslc_teDebugPrintState	331
9.36.2	Function Documentation	331
9.36.2.1	gslc_DrawFillSectorBase(gslc_tsGui *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, gslc_tsColor cArcStart, gslc_tsColor cArcEnd, bool bGradient, int16_t nAngGradStart, int16_t nAngGradRange, int16_t nAngSecStart, int16_t nAngSecEnd)	331
9.36.2.2	gslc_FontSetBase(gslc_tsGui *pGui, uint8_t nFontInd, int16_t nFontId, gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)	331
9.36.2.3	gslc_OrderCoord(int16_t *pnX0, int16_t *pnY0, int16_t *pnX1, int16_t *pnY1) .	331
9.36.2.4	gslc_SwapCoords(int16_t *pnXa, int16_t *pnYa, int16_t *pnXb, int16_t *pnYb) .	331
9.36.3	Variable Documentation	331
9.36.3.1	ERRSTR_NULL	331
9.36.3.2	ERRSTR_PXD_NULL	331
9.36.3.3	g_pfDebugOut	331
9.36.3.4	m_nLUTSinF0X16	332

9.37 src/GUISlice.h File Reference	332
9.37.1 Macro Definition Documentation	346
9.37.1.1 GSLC_2PI	346
9.37.1.2 GSLC_ALIGN_BOT_LEFT	346
9.37.1.3 GSLC_ALIGN_BOT_MID	346
9.37.1.4 GSLC_ALIGN_BOT_RIGHT	346
9.37.1.5 GSLC_ALIGN_MID_LEFT	346
9.37.1.6 GSLC_ALIGN_MID_MID	346
9.37.1.7 GSLC_ALIGN_MID_RIGHT	346
9.37.1.8 GSLC_ALIGN_TOP_LEFT	346
9.37.1.9 GSLC_ALIGN_TOP_MID	346
9.37.1.10 GSLC_ALIGN_TOP_RIGHT	346
9.37.1.11 GSLC_ALIGHN_LEFT	347
9.37.1.12 GSLC_ALIGHN_MID	347
9.37.1.13 GSLC_ALIGHN_RIGHT	347
9.37.1.14 GSLC_ALIGNV_BOT	347
9.37.1.15 GSLC_ALIGNV_MID	347
9.37.1.16 GSLC_ALIGNV_TOP	347
9.37.1.17 GSLC_COL_BLACK	347
9.37.1.18 GSLC_COL_BLUE	347
9.37.1.19 GSLC_COL_BLUE_DK1	347
9.37.1.20 GSLC_COL_BLUE_DK2	347
9.37.1.21 GSLC_COL_BLUE_DK3	348
9.37.1.22 GSLC_COL_BLUE_DK4	348
9.37.1.23 GSLC_COL_BLUE_LT1	348
9.37.1.24 GSLC_COL_BLUE_LT2	348
9.37.1.25 GSLC_COL_BLUE_LT3	348
9.37.1.26 GSLC_COL_BLUE_LT4	348
9.37.1.27 GSLC_COL_BROWN	348
9.37.1.28 GSLC_COL_CYAN	348

9.37.1.29 GSLC_COL_GRAY	348
9.37.1.30 GSLC_COL_GRAY_DK1	348
9.37.1.31 GSLC_COL_GRAY_DK2	349
9.37.1.32 GSLC_COL_GRAY_DK3	349
9.37.1.33 GSLC_COL_GRAY_LT1	349
9.37.1.34 GSLC_COL_GRAY_LT2	349
9.37.1.35 GSLC_COL_GRAY_LT3	349
9.37.1.36 GSLC_COL_GREEN	349
9.37.1.37 GSLC_COL_GREEN_DK1	349
9.37.1.38 GSLC_COL_GREEN_DK2	349
9.37.1.39 GSLC_COL_GREEN_DK3	349
9.37.1.40 GSLC_COL_GREEN_DK4	349
9.37.1.41 GSLC_COL_GREEN_LT1	350
9.37.1.42 GSLC_COL_GREEN_LT2	350
9.37.1.43 GSLC_COL_GREEN_LT3	350
9.37.1.44 GSLC_COL_GREEN_LT4	350
9.37.1.45 GSLC_COL_MAGENTA	350
9.37.1.46 GSLC_COL_ORANGE	350
9.37.1.47 GSLC_COL_PURPLE	350
9.37.1.48 GSLC_COL_RED	350
9.37.1.49 GSLC_COL_RED_DK1	350
9.37.1.50 GSLC_COL_RED_DK2	350
9.37.1.51 GSLC_COL_RED_DK3	351
9.37.1.52 GSLC_COL_RED_DK4	351
9.37.1.53 GSLC_COL_RED_LT1	351
9.37.1.54 GSLC_COL_RED_LT2	351
9.37.1.55 GSLC_COL_RED_LT3	351
9.37.1.56 GSLC_COL_RED_LT4	351
9.37.1.57 GSLC_COL_TEAL	351
9.37.1.58 GSLC_COL_WHITE	351

9.37.1.59	GSLC_COL_YELLOW	351
9.37.1.60	GSLC_COL_YELLOW_DK	351
9.37.1.61	GSLC_COLMONO_BLACK	352
9.37.1.62	GSLC_COLMONO_WHITE	352
9.37.1.63	GSLC_ELEM_FEA_CLICK_EN	352
9.37.1.64	GSLC_ELEM_FEA_FILL_EN	352
9.37.1.65	GSLC_ELEM_FEA_FRAME_EN	352
9.37.1.66	GSLC_ELEM_FEA_GLOW_EN	352
9.37.1.67	GSLC_ELEM_FEA_NONE	352
9.37.1.68	GSLC_ELEM_FEA_ROUND_EN	352
9.37.1.69	GSLC_ELEM_FEA_VALID	352
9.37.1.70	GSLC_ELEMREF_DEFAULT	352
9.37.1.71	GSLC_PMEM	353
9.37.2	Typedef Documentation	353
9.37.2.1	GSLC_CB_DEBUG_OUT	353
9.37.2.2	GSLC_CB_DRAW	353
9.37.2.3	GSLC_CB_EVENT	353
9.37.2.4	GSLC_CB_INPUT	353
9.37.2.5	GSLC_CB_PIN_POLL	353
9.37.2.6	GSLC_CB_TICK	353
9.37.2.7	GSLC_CB_TOUCH	353
9.37.2.8	gslc_tsColor	353
9.37.2.9	gslc_tsElem	353
9.37.2.10	gslc_tsEvent	354
9.37.2.11	gslc_tsEventTouch	354
9.37.2.12	gslc_tsPt	354
9.37.2.13	gslc_tsRect	354
9.37.3	Enumeration Type Documentation	354
9.37.3.1	gslc_teAction	354
9.37.3.2	gslc_teElemId	354

9.37.3.3	gslc_teElemInd	355
9.37.3.4	gslc_teElemRefFlags	355
9.37.3.5	gslc_teEventSubType	355
9.37.3.6	gslc_teEventType	356
9.37.3.7	gslc_teFontId	356
9.37.3.8	gslc_teFontRefMode	356
9.37.3.9	gslc_teFontRefType	357
9.37.3.10	gslc_teGroupId	357
9.37.3.11	gslc_telmgRefFlags	357
9.37.3.12	gslc_telnitStat	357
9.37.3.13	gslc_telInputRawEvent	358
9.37.3.14	gslc_tePageId	358
9.37.3.15	gslc_tePin	358
9.37.3.16	gslc_teRedrawType	359
9.37.3.17	gslc_teStackPage	359
9.37.3.18	gslc_teTouch	359
9.37.3.19	gslc_teTxtFlags	360
9.37.3.20	gslc_teTypeCore	360
9.37.4	Variable Documentation	360
9.37.4.1	g_pfDebugOut	360
9.38	src/GUIslice_config.h File Reference	361
9.39	src/GUIslice_config_ard.h File Reference	361
9.39.1	Macro Definition Documentation	362
9.39.1.1	ADAGFX_PIN_CLK	362
9.39.1.2	ADAGFX_PIN_CS	362
9.39.1.3	ADAGFX_PIN_DC	362
9.39.1.4	ADAGFX_PIN_MISO	362
9.39.1.5	ADAGFX_PIN_MOSI	362
9.39.1.6	ADAGFX_PIN_RD	362
9.39.1.7	ADAGFX_PIN_RST	362

9.39.1.8	ADAGFX_PIN_SDCS	362
9.39.1.9	ADAGFX_PIN_WR	362
9.39.1.10	ADAGFX_SPI_HW	362
9.39.1.11	ADATOUCH_FLIP_X	362
9.39.1.12	ADATOUCH_FLIP_Y	362
9.39.1.13	ADATOUCH_SWAP_XY	362
9.39.1.14	DEBUG_ERR	362
9.39.1.15	DRV_DISP_ADAGFX	362
9.39.1.16	DRV_DISP_ADAGFX_ILI9341	362
9.39.1.17	DRV_TOUCH_NONE	362
9.39.1.18	GSLC_BMP_TRANS_EN	362
9.39.1.19	GSLC_BMP_TRANS_RGB	362
9.39.1.20	GSLC_CLIP_EN	362
9.39.1.21	GSLC_DEV_TOUCH	362
9.39.1.22	GSLC_FEATURE_COMPOUND	362
9.39.1.23	GSLC_FEATURE_INPUT	363
9.39.1.24	GSLC_FEATURE_XGAUGE_RADIAL	363
9.39.1.25	GSLC_FEATURE_XGAUGE_RAMP	363
9.39.1.26	GSLC_FEATURE_XTEXTBOX_EMBED	363
9.39.1.27	GSLC_LOCAL_STR	363
9.39.1.28	GSLC_LOCAL_STR_LEN	363
9.39.1.29	GSLC_ROTATE	363
9.39.1.30	GSLC_SD_BUFFPIXEL	363
9.39.1.31	GSLC_SD_EN	363
9.39.1.32	GSLC_TOUCH_MAX_EVT	363
9.39.1.33	GSLC_USE_FLOAT	363
9.39.1.34	GSLC_USE_PROGMEM	363
9.39.1.35	TOUCH_ROTATION_DATA	363
9.39.1.36	TOUCH_ROTATION_FLIPX	363
9.39.1.37	TOUCH_ROTATION_FLIPY	363

9.39.1.38 TOUCH_ROTATION_SWAPXY	363
9.40 src/GUISlice_config_linux.h File Reference	363
9.40.1 Macro Definition Documentation	364
9.40.1.1 ADATOUCH_FLIP_X	364
9.40.1.2 ADATOUCH_FLIP_Y	364
9.40.1.3 ADATOUCH_SWAP_XY	364
9.40.1.4 DEBUG_ERR	364
9.40.1.5 DRV_DISP_SDL1	364
9.40.1.6 DRV_SDL_FIX_START	364
9.40.1.7 DRV_SDL_MOUSE_SHOW	364
9.40.1.8 DRV_TOUCH_TSLIB	364
9.40.1.9 GSLC_BMP_TRANS_EN	364
9.40.1.10 GSLC_BMP_TRANS_RGB	364
9.40.1.11 GSLC_DEV_FB	364
9.40.1.12 GSLC_DEV_TOUCH	364
9.40.1.13 GSLC_DEV_VID_DRV	364
9.40.1.14 GSLC_FEATURE_COMPOUND	364
9.40.1.15 GSLC_FEATURE_INPUT	364
9.40.1.16 GSLC_FEATURE_XGAUGE_RADIAL	364
9.40.1.17 GSLC_FEATURE_XGAUGE_RAMP	364
9.40.1.18 GSLC_FEATURE_XTEXTBOX_EMBED	364
9.40.1.19 GSLC_LOCAL_STR	364
9.40.1.20 GSLC_LOCAL_STR_LEN	364
9.40.1.21 GSLC_TOUCH_MAX_EVT	364
9.40.1.22 GSLC_USE_FLOAT	364
9.40.1.23 GSLC_USE_PROGMEM	364
9.41 src/GUISlice_drv.h File Reference	364
9.42 src/GUISlice_drv_adagfx.cpp File Reference	365
9.43 src/GUISlice_drv_adagfx.h File Reference	365
9.43.1 Detailed Description	368

9.43.2 Macro Definition Documentation	368
9.43.2.1 DRV_HAS_DRAW_BMP_MEM	368
9.43.2.2 DRV_HAS_DRAW_CIRCLE_FILL	368
9.43.2.3 DRV_HAS_DRAW_CIRCLE_FRAME	368
9.43.2.4 DRV_HAS_DRAW_LINE	368
9.43.2.5 DRV_HAS_DRAW_POINT	368
9.43.2.6 DRV_HAS_DRAW_POINTS	368
9.43.2.7 DRV_HAS_DRAW_RECT_FILL	369
9.43.2.8 DRV_HAS_DRAW_RECT_FRAME	369
9.43.2.9 DRV_HAS_DRAW_RECT_ROUND_FILL	369
9.43.2.10 DRV_HAS_DRAW_RECT_ROUND_FRAME	369
9.43.2.11 DRV_HAS_DRAW_TEXT	369
9.43.2.12 DRV_HAS_DRAW_TRI_FILL	369
9.43.2.13 DRV_HAS_DRAW_TRI_FRAME	369
9.43.2.14 DRV_OVERRIDE_TXT_ALIGN	369
9.43.3 Function Documentation	369
9.43.3.1 gslc_DrvAdaptColorToRaw(gslc_tsColor nCol)	369
9.43.3.2 gslc_DrvDestruct(gslc_tsGui *pGui)	369
9.43.3.3 gslc_DrvDrawBkgnd(gslc_tsGui *pGui)	370
9.43.3.4 gslc_DrvDrawBmp24FromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	370
9.43.3.5 gslc_DrvDrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	370
9.43.3.6 gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	371
9.43.3.7 gslc_DrvDrawFillRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t n↵ Radius, gslc_tsColor nCol)	371
9.43.3.8 gslc_DrvDrawFillTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)	371
9.43.3.9 gslc_DrvDrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMid↵ Y, uint16_t nRadius, gslc_tsColor nCol)	372
9.43.3.10 gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	372
9.43.3.11 gslc_DrvDrawFrameRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t n↵ Radius, gslc_tsColor nCol)	373

9.43.3.12 gslc_DrvDrawFrameTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)	373
9.43.3.13 gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_tsColor nCol, gslc_tsImgRef slmgRef)	373
9.43.3.14 gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)	374
9.43.3.15 gslc_DrvDrawMonoFromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	374
9.43.3.16 gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)	374
9.43.3.17 gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc_tsColor nCol)	375
9.43.3.18 gslc_DrvDrawTxt(gslc_tsGui *pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)	375
9.43.3.19 gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)	376
9.43.3.20 gslc_DrvFontsDestruct(gslc_tsGui *pGui)	376
9.43.3.21 gslc_DrvGetDriverDisp(gslc_tsGui *pGui)	376
9.43.3.22 gslc_DrvGetDriverTouch(gslc_tsGui *pGui)	376
9.43.3.23 gslc_DrvGetNameDisp(gslc_tsGui *pGui)	378
9.43.3.24 gslc_DrvGetNameTouch(gslc_tsGui *pGui)	378
9.43.3.25 gslc_DrvGetTouch(gslc_tsGui *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, gslc_teInputRawEvent *peInputEvent, int16_t *pnInputVal)	378
9.43.3.26 gslc_DrvGetTxtSize(gslc_tsGui *pGui, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)	379
9.43.3.27 gslc_DrvImageDestruct(void *pvImg)	379
9.43.3.28 gslc_DrvInit(gslc_tsGui *pGui)	379
9.43.3.29 gslc_DrvInitTouch(gslc_tsGui *pGui, const char *acDev)	380
9.43.3.30 gslc_DrvInitTs(gslc_tsGui *pGui, const char *acDev)	380
9.43.3.31 gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)	380
9.43.3.32 gslc_DrvPageFlipNow(gslc_tsGui *pGui)	381
9.43.3.33 gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)	381
9.43.3.34 gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)	381
9.43.3.35 gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)	382

9.43.3.36	<code>gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	382
9.43.3.37	<code>gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef slmgRef)</code>	382
9.43.3.38	<code>gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef slmgRef)</code>	383
9.44	<code>src/GUIslice_drv_m5stack.cpp</code> File Reference	383
9.45	<code>src/GUIslice_drv_m5stack.h</code> File Reference	383
9.45.1	Detailed Description	386
9.45.2	Macro Definition Documentation	386
9.45.2.1	<code>DRV_HAS_DRAW_BMP_MEM</code>	386
9.45.2.2	<code>DRV_HAS_DRAW_CIRCLE_FILL</code>	387
9.45.2.3	<code>DRV_HAS_DRAW_CIRCLE_FRAME</code>	387
9.45.2.4	<code>DRV_HAS_DRAW_LINE</code>	387
9.45.2.5	<code>DRV_HAS_DRAW_POINT</code>	387
9.45.2.6	<code>DRV_HAS_DRAW_POINTS</code>	387
9.45.2.7	<code>DRV_HAS_DRAW_RECT_FILL</code>	387
9.45.2.8	<code>DRV_HAS_DRAW_RECT_FRAME</code>	387
9.45.2.9	<code>DRV_HAS_DRAW_RECT_ROUND_FILL</code>	387
9.45.2.10	<code>DRV_HAS_DRAW_RECT_ROUND_FRAME</code>	387
9.45.2.11	<code>DRV_HAS_DRAW_TEXT</code>	387
9.45.2.12	<code>DRV_HAS_DRAW_TRI_FILL</code>	388
9.45.2.13	<code>DRV_HAS_DRAW_TRI_FRAME</code>	388
9.45.2.14	<code>DRV_OVERRIDE_TXT_ALIGN</code>	388
9.45.3	Function Documentation	388
9.45.3.1	<code>gslc_DrvAdaptColorToRaw(gslc_tsColor nCol)</code>	388
9.45.3.2	<code>gslc_DrvDestruct(gslc_tsGui *pGui)</code>	388
9.45.3.3	<code>gslc_DrvDrawBkgnd(gslc_tsGui *pGui)</code>	388
9.45.3.4	<code>gslc_DrvDrawBmp24FromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)</code>	389
9.45.3.5	<code>gslc_DrvDrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)</code>	389
9.45.3.6	<code>gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	389

9.45.3.7	<code>gslc_DrvDrawFillRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t n← Radius, gslc_tsColor nCol)</code>	390
9.45.3.8	<code>gslc_DrvDrawFillTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)</code>	390
9.45.3.9	<code>gslc_DrvDrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMid← Y, uint16_t nRadius, gslc_tsColor nCol)</code>	390
9.45.3.10	<code>gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	391
9.45.3.11	<code>gslc_DrvDrawFrameRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t n← Radius, gslc_tsColor nCol)</code>	391
9.45.3.12	<code>gslc_DrvDrawFrameTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)</code>	391
9.45.3.13	<code>gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_ts← ImgRef slmgRef)</code>	392
9.45.3.14	<code>gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)</code>	392
9.45.3.15	<code>gslc_DrvDrawMonoFromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDst← Y, const unsigned char *pBitmap, bool bProgMem)</code>	393
9.45.3.16	<code>gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)</code> .	393
9.45.3.17	<code>gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc← _tsColor nCol)</code>	393
9.45.3.18	<code>gslc_DrvDrawTxt(gslc_tsGui *pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc← _tsColor colBg)</code>	394
9.45.3.19	<code>gslc_DrvDrawTxtAlign(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int8_t eTxtAlign, gslc_tsFont *pFont, const char *pStr, gslc_teTxt← Flags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)</code>	394
9.45.3.20	<code>gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)</code>	395
9.45.3.21	<code>gslc_DrvFontsDestruct(gslc_tsGui *pGui)</code>	395
9.45.3.22	<code>gslc_DrvGetDriverDisp(gslc_tsGui *pGui)</code>	395
9.45.3.23	<code>gslc_DrvGetDriverTouch(gslc_tsGui *pGui)</code>	396
9.45.3.24	<code>gslc_DrvGetNameDisp(gslc_tsGui *pGui)</code>	396
9.45.3.25	<code>gslc_DrvGetNameTouch(gslc_tsGui *pGui)</code>	396
9.45.3.26	<code>gslc_DrvGetTxtSize(gslc_tsGui *pGui, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxt← SzW, uint16_t *pnTxtSzH)</code>	396
9.45.3.27	<code>gslc_DrvImageDestruct(void *pvImg)</code>	397
9.45.3.28	<code>gslc_DrvInit(gslc_tsGui *pGui)</code>	397

9.45.3.29	<code>gslc_DrvInitTs(gslc_tsGui *pGui, const char *acDev)</code>	398
9.45.3.30	<code>gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)</code>	398
9.45.3.31	<code>gslc_DrvPageFlipNow(gslc_tsGui *pGui)</code>	398
9.45.3.32	<code>gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)</code>	399
9.45.3.33	<code>gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	399
9.45.3.34	<code>gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)</code>	399
9.45.3.35	<code>gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	400
9.45.3.36	<code>gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts↔ ImgRef slmgRef)</code>	400
9.45.3.37	<code>gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts↔ ImgRef slmgRef)</code>	400
9.45.4	Variable Documentation	400
9.45.4.1	<code>ERRSTR_NULL</code>	400
9.45.4.2	<code>ERRSTR_PXD_NULL</code>	401
9.46	<code>src/GUIslice_drv_sdl.c</code> File Reference	401
9.47	<code>src/GUIslice_drv_sdl.h</code> File Reference	401
9.47.1	Detailed Description	403
9.47.2	Macro Definition Documentation	403
9.47.2.1	<code>DRV_HAS_DRAW_POINT</code>	403
9.47.2.2	<code>DRV_OVERRIDE_TXT_ALIGN</code>	403
9.47.3	Function Documentation	403
9.47.3.1	<code>gslc_DrvAdaptColor(gslc_tsColor sCol)</code>	403
9.47.3.2	<code>gslc_DrvAdaptRect(gslc_tsRect rRect)</code>	404
9.47.3.3	<code>gslc_DrvCleanStart(const char *sTTY)</code>	404
9.47.3.4	<code>gslc_DrvDestruct(gslc_tsGui *pGui)</code>	404
9.47.3.5	<code>gslc_DrvDrawBkgnd(gslc_tsGui *pGui)</code>	405
9.47.3.6	<code>gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	405
9.47.3.7	<code>gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	405
9.47.3.8	<code>gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_ts↔ ImgRef slmgRef)</code>	405
9.47.3.9	<code>gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)</code>	406

9.47.3.10	<code>gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)</code>	406
9.47.3.11	<code>gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc_tsColor nCol)</code>	407
9.47.3.12	<code>gslc_DrvDrawTxt(gslc_tsGui *pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)</code>	408
9.47.3.13	<code>gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)</code>	408
9.47.3.14	<code>gslc_DrvFontsDestruct(gslc_tsGui *pGui)</code>	409
9.47.3.15	<code>gslc_DrvGetDriverDisp(gslc_tsGui *pGui)</code>	409
9.47.3.16	<code>gslc_DrvGetDriverTouch(gslc_tsGui *pGui)</code>	409
9.47.3.17	<code>gslc_DrvGetNameDisp(gslc_tsGui *pGui)</code>	410
9.47.3.18	<code>gslc_DrvGetNameTouch(gslc_tsGui *pGui)</code>	411
9.47.3.19	<code>gslc_DrvGetTouch(gslc_tsGui *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, gslc_telInputRawEvent *peInputEvent, int16_t *pnInputVal)</code>	411
9.47.3.20	<code>gslc_DrvGetTxtSize(gslc_tsGui *pGui, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)</code>	411
9.47.3.21	<code>gslc_DrvImageDestruct(void *pvImg)</code>	412
9.47.3.22	<code>gslc_DrvInit(gslc_tsGui *pGui)</code>	412
9.47.3.23	<code>gslc_DrvInitTouch(gslc_tsGui *pGui, const char *acDev)</code>	413
9.47.3.24	<code>gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	413
9.47.3.25	<code>gslc_DrvPageFlipNow(gslc_tsGui *pGui)</code>	413
9.47.3.26	<code>gslc_DrvReportInfoPost()</code>	414
9.47.3.27	<code>gslc_DrvReportInfoPre()</code>	414
9.47.3.28	<code>gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)</code>	414
9.47.3.29	<code>gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	414
9.47.3.30	<code>gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	415
9.47.3.31	<code>gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	415
9.47.3.32	<code>gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsImgRef sImgRef)</code>	415
9.47.3.33	<code>gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsImgRef sImgRef)</code>	415
9.48	<code>src/GUISlice_drv_tft_espi.cpp</code> File Reference	416

9.49	src/GUISlice_drv_tft_espi.h File Reference	416
9.49.1	Detailed Description	419
9.49.2	Macro Definition Documentation	419
9.49.2.1	DRV_HAS_DRAW_BMP_MEM	419
9.49.2.2	DRV_HAS_DRAW_CIRCLE_FILL	419
9.49.2.3	DRV_HAS_DRAW_CIRCLE_FRAME	419
9.49.2.4	DRV_HAS_DRAW_LINE	419
9.49.2.5	DRV_HAS_DRAW_POINT	420
9.49.2.6	DRV_HAS_DRAW_POINTS	420
9.49.2.7	DRV_HAS_DRAW_RECT_FILL	420
9.49.2.8	DRV_HAS_DRAW_RECT_FRAME	420
9.49.2.9	DRV_HAS_DRAW_RECT_ROUND_FILL	420
9.49.2.10	DRV_HAS_DRAW_RECT_ROUND_FRAME	420
9.49.2.11	DRV_HAS_DRAW_TEXT	420
9.49.2.12	DRV_HAS_DRAW_TRI_FILL	420
9.49.2.13	DRV_HAS_DRAW_TRI_FRAME	420
9.49.2.14	DRV_OVERRIDE_TXT_ALIGN	420
9.49.2.15	GSLC_SPIFFS_EN	420
9.49.3	Function Documentation	420
9.49.3.1	gslc_DrvAdaptColorToRaw(gslc_tsColor nCol)	420
9.49.3.2	gslc_DrvDestruct(gslc_tsGui *pGui)	420
9.49.3.3	gslc_DrvDrawBkgnd(gslc_tsGui *pGui)	421
9.49.3.4	gslc_DrvDrawBmp24FromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	421
9.49.3.5	gslc_DrvDrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	421
9.49.3.6	gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	422
9.49.3.7	gslc_DrvDrawFillRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)	422
9.49.3.8	gslc_DrvDrawFillTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)	422
9.49.3.9	gslc_DrvDrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	423

9.49.3.10 gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	423
9.49.3.11 gslc_DrvDrawFrameRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)	424
9.49.3.12 gslc_DrvDrawFrameTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)	424
9.49.3.13 gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_tsImgRef slmgRef)	424
9.49.3.14 gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)	425
9.49.3.15 gslc_DrvDrawMonoFromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	425
9.49.3.16 gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)	425
9.49.3.17 gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc_tsColor nCol)	426
9.49.3.18 gslc_DrvDrawTxt(gslc_tsGui *pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)	426
9.49.3.19 gslc_DrvDrawTxtAlign(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int8_t eTxtAlign, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)	427
9.49.3.20 gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)	427
9.49.3.21 gslc_DrvFontsDestruct(gslc_tsGui *pGui)	427
9.49.3.22 gslc_DrvGetDriverDisp(gslc_tsGui *pGui)	428
9.49.3.23 gslc_DrvGetDriverTouch(gslc_tsGui *pGui)	428
9.49.3.24 gslc_DrvGetNameDisp(gslc_tsGui *pGui)	428
9.49.3.25 gslc_DrvGetNameTouch(gslc_tsGui *pGui)	429
9.49.3.26 gslc_DrvGetTxtSize(gslc_tsGui *pGui, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)	429
9.49.3.27 gslc_DrvImageDestruct(void *pvImg)	429
9.49.3.28 gslc_DrvInit(gslc_tsGui *pGui)	430
9.49.3.29 gslc_DrvInitTs(gslc_tsGui *pGui, const char *acDev)	430
9.49.3.30 gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)	431
9.49.3.31 gslc_DrvPageFlipNow(gslc_tsGui *pGui)	431
9.49.3.32 gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)	431

9.49.3.33	<code>gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	431
9.49.3.34	<code>gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	432
9.49.3.35	<code>gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	432
9.49.3.36	<code>gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsImgRef sImgRef)</code>	432
9.49.3.37	<code>gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsImgRef sImgRef)</code>	433
9.50	<code>src/GUISlice_drv_utf8.cpp</code> File Reference	433
9.51	<code>src/GUISlice_drv_utf8.h</code> File Reference	433
9.51.1	Detailed Description	436
9.51.2	Macro Definition Documentation	436
9.51.2.1	<code>DRV_HAS_DRAW_BMP_MEM</code>	436
9.51.2.2	<code>DRV_HAS_DRAW_CIRCLE_FILL</code>	437
9.51.2.3	<code>DRV_HAS_DRAW_CIRCLE_FRAME</code>	437
9.51.2.4	<code>DRV_HAS_DRAW_LINE</code>	437
9.51.2.5	<code>DRV_HAS_DRAW_POINT</code>	437
9.51.2.6	<code>DRV_HAS_DRAW_POINTS</code>	437
9.51.2.7	<code>DRV_HAS_DRAW_RECT_FILL</code>	437
9.51.2.8	<code>DRV_HAS_DRAW_RECT_FRAME</code>	437
9.51.2.9	<code>DRV_HAS_DRAW_RECT_ROUND_FILL</code>	437
9.51.2.10	<code>DRV_HAS_DRAW_RECT_ROUND_FRAME</code>	437
9.51.2.11	<code>DRV_HAS_DRAW_TEXT</code>	437
9.51.2.12	<code>DRV_HAS_DRAW_TRI_FILL</code>	438
9.51.2.13	<code>DRV_HAS_DRAW_TRI_FRAME</code>	438
9.51.2.14	<code>DRV_OVERRIDE_TXT_ALIGN</code>	438
9.51.3	Function Documentation	438
9.51.3.1	<code>gslc_DrvAdaptColorToRaw(gslc_tsColor nCol)</code>	438
9.51.3.2	<code>gslc_DrvDestruct(gslc_tsGui *pGui)</code>	438
9.51.3.3	<code>gslc_DrvDrawBkgnd(gslc_tsGui *pGui)</code>	438
9.51.3.4	<code>gslc_DrvDrawBmp24FromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)</code>	439

9.51.3.5	<code>gslc_DrvDrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)</code>	439
9.51.3.6	<code>gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	439
9.51.3.7	<code>gslc_DrvDrawFillRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)</code>	440
9.51.3.8	<code>gslc_DrvDrawFillTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)</code>	440
9.51.3.9	<code>gslc_DrvDrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)</code>	440
9.51.3.10	<code>gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	441
9.51.3.11	<code>gslc_DrvDrawFrameRoundRect(gslc_tsGui *pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)</code>	441
9.51.3.12	<code>gslc_DrvDrawFrameTriangle(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)</code>	441
9.51.3.13	<code>gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_tsImgRef sImgRef)</code>	442
9.51.3.14	<code>gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)</code>	442
9.51.3.15	<code>gslc_DrvDrawMonoFromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)</code>	443
9.51.3.16	<code>gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)</code>	443
9.51.3.17	<code>gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc_tsColor nCol)</code>	443
9.51.3.18	<code>gslc_DrvDrawTxt(gslc_tsGui *pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)</code>	444
9.51.3.19	<code>gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)</code>	444
9.51.3.20	<code>gslc_DrvFontsDestruct(gslc_tsGui *pGui)</code>	445
9.51.3.21	<code>gslc_DrvGetDriverDisp(gslc_tsGui *pGui)</code>	445
9.51.3.22	<code>gslc_DrvGetDriverTouch(gslc_tsGui *pGui)</code>	445
9.51.3.23	<code>gslc_DrvGetNameDisp(gslc_tsGui *pGui)</code>	446
9.51.3.24	<code>gslc_DrvGetNameTouch(gslc_tsGui *pGui)</code>	447
9.51.3.25	<code>gslc_DrvGetTouch(gslc_tsGui *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, gslc_teInputRawEvent *peInputEvent, int16_t *pnInputVal)</code>	447
9.51.3.26	<code>gslc_DrvGetTxtSize(gslc_tsGui *pGui, gslc_tsFont *pFont, const char *pStr, gslc_teTxtFlags eTxtFlags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)</code>	447

9.51.3.27	<code>gslc_DrvImageDestruct(void *pvImg)</code>	448
9.51.3.28	<code>gslc_DrvInit(gslc_tsGui *pGui)</code>	448
9.51.3.29	<code>gslc_DrvInitTouch(gslc_tsGui *pGui, const char *acDev)</code>	449
9.51.3.30	<code>gslc_DrvInitTs(gslc_tsGui *pGui, const char *acDev)</code>	449
9.51.3.31	<code>gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	449
9.51.3.32	<code>gslc_DrvPageFlipNow(gslc_tsGui *pGui)</code>	450
9.51.3.33	<code>gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)</code>	450
9.51.3.34	<code>gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	450
9.51.3.35	<code>gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	450
9.51.3.36	<code>gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	451
9.51.3.37	<code>gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef sImgRef)</code>	451
9.51.3.38	<code>gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef sImgRef)</code>	451
9.52	<code>src/GUISlice_ex.h</code> File Reference	452
9.53	<code>src/GUISlice_th.cpp</code> File Reference	452
9.53.1	Function Documentation	453
9.53.1.1	<code>gslc_getTouchHandler(void)</code>	453
9.53.1.2	<code>gslc_InitTouchHandler(TouchHandler *pTH)</code>	453
9.53.2	Variable Documentation	453
9.53.2.1	<code>pTouchHandler</code>	453
9.54	<code>src/GUISlice_th.h</code> File Reference	453
9.54.1	Function Documentation	454
9.54.1.1	<code>gslc_getTouchHandler(void)</code>	454
9.54.1.2	<code>gslc_InitTouchHandler(TouchHandler *pTHO)</code>	454
9.55	<code>src/GUISlice_th_XPT2046.h</code> File Reference	454
9.56	<code>src/GUISlice_version.h</code> File Reference	455
9.56.1	Macro Definition Documentation	455
9.56.1.1	<code>GUISLICE_VER</code>	455

Chapter 1

GUIslice library

A lightweight GUI framework for embedded displays

Design your GUI with a **drag & drop builder**, then apply the same code to a wide range of displays, libraries and controllers with the **cross-platform framework**. Open source **MIT license** grants free commercial usage.

- Extensive [Documentation](#) guides available
- [GUIslice API documentation \(online\) & \(PDF\)](#)
- Active development: see [latest updates & work in progress](#)
- [Release history](#)
- [Website \(www.impulseadventure.com\)](#)
- **Support email:** guislice@gmail.com
- GUIslice by Calvin Hass and [GitHub contributors](#), Builder by Paul Conti

Features

- Pure C library, no dynamic memory allocation
- *Widgets:*
 - text, images, buttons, checkboxes, radio buttons, sliders, keypad, listbox, radial controls, scrolling textbox / terminal, graphs, etc. plus extensions and multiple pages.
- Cross-platform **GUIslice Builder** application to generate layouts
- *Platform-independent* GUI core currently supports:
 - Adafruit-GFX, TFT_eSPI, mcufriend, UTFT, SDL1.2, SDL2.0
- *Devices:*
 - Raspberry Pi, Arduino, ATmega2560, ESP8266 / NodeMCU, ESP32, M5stack, Teensy 3, Feather M0 (Cortex-M0), nRF52 (Cortex-M4F), LINUX, Beaglebone Black, STM32, Due, etc.
- *Typical displays:*

- PiTFT, Adafruit TFT 3.5" / 2.8" / 2.4" / 2.2" / 1.44", FeatherWing TFT, OLED 0.96", mcufriend, BuyDisplay / EastRising 4.3" 5" 7", Waveshare, 4D Cape
- *Display drivers include:*
 - ILI9341, ST7735, SSD1306, HX8347D, HX8357, PCD8544, RA8875, RA8876, ILI9341_t3, ILI9341_↔
due
- *Touchscreen control including:*
 - STMPE610, FT6206, FT5206, XPT2046, 4-wire, tslib, URTouch, Adafruit Seesaw
- Foreign characters / UTF-8 encoding (in SDL mode), anti-aliased fonts (in TFT_eSPI mode)
- Dynamic display rotation
- GPIO / pin / keyboard / Adafruit Seesaw control for non-touchscreen devices

Screenshots

GUIslice Builder

- Includes cross-platform (Windows, LINUX and Mac) desktop application to generate GUIslice layouts
- Please refer to [GUIslice Builder wiki](#) for documentation

Chapter 2

Todo List

Global [gslc_CollectFindFocusStep](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, bool bNext, bool *pb↵
Wrapped, int16_t *pnElemInd)

Doc. This API is experimental and subject to change

Global [gslc_InitInputMap](#) ([gslc_tsGui](#) *pGui, [gslc_tsInputMap](#) *asInputMap, uint8_t nInputMapMax)

Doc. This API is experimental and subject to change

Global [gslc_InputMapAdd](#) ([gslc_tsGui](#) *pGui, [gslc_tInputRawEvent](#) eInputEvent, int16_t nInputVal, [gslc_↵
_teAction](#) eAction, int16_t nActionVal)

Doc. This API is experimental and subject to change

Global [gslc_InputMapLookup](#) ([gslc_tsGui](#) *pGui, [gslc_tInputRawEvent](#) eInputEvent, int16_t nInputVal,
[gslc_teAction](#) *peAction, int16_t *pnActionVal)

Doc. This API is experimental and subject to change

Global [gslc_PageFocusStep](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage, bool bNext)

Doc. This API is experimental and subject to change

Global [gslc_SetPinPollFunc](#) ([gslc_tsGui](#) *pGui, [GSLC_CB_PIN_POLL](#) pfunc)

Doc. This API is experimental and subject to change

Chapter 3

Module Index

3.1 Modules

Here is a list of all modules:

General Functions	13
Graphics General Functions	20
Graphics Primitive Functions	29
Font Functions	38
Page Functions	41
Element Functions	46
Element: Creation Functions	47
Element: General Functions	51
Element: Update Functions	52
Touchscreen Functions	66
Input Mapping Functions	70
General Purpose Macros	71
Flash-based Element Macros	72
Internal Functions	76
Internal: Misc Functions	93
Internal: Element Functions	94
Internal: Page Functions	100
Internal: Element Collection Functions	105
Internal: Element Collection Event Functions	111
Internal: Tracking Functions	113
Internal: Cleanup Functions	115

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

gslc_tsCollect	119
gslc_tsColor	120
gslc_tsDriver	121
gslc_tsElem	122
gslc_tsElemRef	124
gslc_tsEvent	124
gslc_tsEventTouch	125
gslc_tsFont	125
gslc_tsGui	126
gslc_tsImgRef	128
gslc_tsInputMap	129
gslc_tsPage	130
gslc_tsPt	131
gslc_tsRect	131
gslc_tsXCheckbox	132
gslc_tsXGauge	133
gslc_tsXGlowball	136
gslc_tsXGlowballRing	138
gslc_tsXGraph	139
gslc_tsXKeyPad	142
gslc_tsXKeyPad_Alpha	144
gslc_tsXKeyPad_Num	145
gslc_tsXKeyPadCfg	146
gslc_tsXKeyPadData	149
gslc_tsXListbox	149
gslc_tsXProgress	153
gslc_tsXRadial	154
gslc_tsXRamp	157
gslc_tsXRingGauge	158
gslc_tsXSelNum	160
gslc_tsXSlider	161
gslc_tsXSpinner	163
gslc_tsXTemplate	166
gslc_tsXTextbox	166
THPoint	169
TouchHandler	170
TouchHandler_XPT2046	171

Chapter 5

Data Structure Index

5.1 Data Structures

Here are the data structures with brief descriptions:

gslc_tsCollect	Element collection struct	119
gslc_tsColor	Color structure. Defines RGB triplet	120
gslc_tsDriver	121
gslc_tsElem	Element Struct	122
gslc_tsElemRef	Element reference structure	124
gslc_tsEvent	Event structure	124
gslc_tsEventTouch	Structure used to pass touch data through event	125
gslc_tsFont	Font reference structure	125
gslc_tsGui	GUI structure	126
gslc_tsImgRef	Image reference structure	128
gslc_tsInputMap	Input mapping	129
gslc_tsPage	Page structure	130
gslc_tsPt	Define point coordinates	131
gslc_tsRect	Rectangular region. Defines X,Y corner coordinates plus dimensions	131
gslc_tsXCheckbox	Extended data for Checkbox element	132
gslc_tsXGauge	Extended data for Gauge element	133
gslc_tsXGlowball	Extended data for Slider element	136
gslc_tsXGlowballRing	138
gslc_tsXGraph	Extended data for Graph element	139

gslc_tsXKeyPad	
Extended data for KeyPad element	142
gslc_tsXKeyPad_Alpha	144
gslc_tsXKeyPad_Num	145
gslc_tsXKeyPadCfg	
Configuration for the KeyPad	146
gslc_tsXKeyPadData	
Input callback data structure	149
gslc_tsXListbox	
Extended data for Listbox element	149
gslc_tsXProgress	
Extended data for Gauge element	153
gslc_tsXRadial	
Extended data for Gauge element	154
gslc_tsXRamp	
Extended data for Gauge element	157
gslc_tsXRingGauge	
Extended data for XRingGauge element	158
gslc_tsXSelNum	
Extended data for SelNum element	160
gslc_tsXSlider	
Extended data for Slider element	161
gslc_tsXSpinner	
Extended data for Spinner element	163
gslc_tsXTemplate	
Callback function for slider feedback	166
gslc_tsXTextbox	
Extended data for Textbox element	166
THPoint	169
TouchHandler	170
TouchHandler_XPT2046	171

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

src/GUISlice.c	322
src/GUISlice.h	332
src/GUISlice_config.h	361
src/GUISlice_config_ard.h	361
src/GUISlice_config_linux.h	363
src/GUISlice_drv.h	364
src/GUISlice_drv_adagfx.cpp	365
src/GUISlice_drv_adagfx.h	
GUISlice library (driver layer for Adafruit-GFX)	365
src/GUISlice_drv_m5stack.cpp	383
src/GUISlice_drv_m5stack.h	
GUISlice library (driver layer for M5stack)	383
src/GUISlice_drv_sdl.c	401
src/GUISlice_drv_sdl.h	
GUISlice library (driver layer for LINUX / SDL)	401
src/GUISlice_drv_tft_espi.cpp	416
src/GUISlice_drv_tft_espi.h	
GUISlice library (driver layer for TFT-eSPI)	416
src/GUISlice_drv_utft.cpp	433
src/GUISlice_drv_utft.h	
GUISlice library (driver layer for UTFT)	433
src/GUISlice_ex.h	452
src/GUISlice_th.cpp	452
src/GUISlice_th.h	453
src/GUISlice_th_XPT2046.h	454
src/GUISlice_version.h	455
src/elem/XCheckbox.c	173
src/elem/XCheckbox.h	178
src/elem/XGauge.c	184
src/elem/XGauge.h	189
src/elem/XGlowball.c	195
src/elem/XGlowball.h	197
src/elem/XGraph.c	200
src/elem/XGraph.h	204
src/elem/XKeyPad.c	208

src/elem/XKeyPad.h	216
src/elem/XKeyPad_Alpha.c	226
src/elem/XKeyPad_Alpha.h	229
src/elem/XKeyPad_Num.c	231
src/elem/XKeyPad_Num.h	233
src/elem/XListbox.c	235
src/elem/XListbox.h	242
src/elem/XProgress.c	249
src/elem/XProgress.h	252
src/elem/XRadial.c	256
src/elem/XRadial.h	260
src/elem/XRamp.c	265
src/elem/XRamp.h	268
src/elem/XRingGauge.c	272
src/elem/XRingGauge.h	278
src/elem/XSelNum.c	283
src/elem/XSelNum.h	287
src/elem/XSlider.c	290
src/elem/XSlider.h	294
src/elem/XSpinner.c	299
src/elem/XSpinner.h	304
src/elem/XTemplate.c	309
src/elem/XTemplate.h	311
src/elem/XTextbox.c	313
src/elem/XTextbox.h	318

Chapter 7

Module Documentation

7.1 General Functions

General functions for configuring the GUI.

Functions

- char * [gslc_GetVer](#) ([gslc_tsGui](#) *pGui)
Get the GUIslice version number.
- const char * [gslc_GetNameDisp](#) ([gslc_tsGui](#) *pGui)
Get the GUIslice display driver name.
- const char * [gslc_GetNameTouch](#) ([gslc_tsGui](#) *pGui)
Get the GUIslice touch driver name.
- void * [gslc_GetDriverDisp](#) ([gslc_tsGui](#) *pGui)
Get the native display driver instance.
- void * [gslc_GetDriverTouch](#) ([gslc_tsGui](#) *pGui)
Get the native touch driver instance.
- bool [gslc_Init](#) ([gslc_tsGui](#) *pGui, void *pvDriver, [gslc_tsPage](#) *asPage, uint8_t nMaxPage, [gslc_tsFont](#) *asFont, uint8_t nMaxFont)
Initialize the GUIslice library.
- void [gslc_InitDebug](#) ([GSLC_CB_DEBUG_OUT](#) pfunc)
Initialize debug output.
- void [gslc_DebugPrintf](#) (const char *pFmt,...)
Optimized printf routine for GUIslice debug/error output.
- bool [gslc_GuiRotate](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation)
Dynamically change rotation, automatically adapt touchscreen axes swap/flip.
- void [gslc_Quit](#) ([gslc_tsGui](#) *pGui)
Exit the GUIslice environment.
- void [gslc_Update](#) ([gslc_tsGui](#) *pGui)
Perform main GUIslice handling functions.
- bool [gslc_SetBkgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tslmgRef](#) slmgRef)
Configure the background to use a bitmap image.
- bool [gslc_SetBkgndColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)
Configure the background to use a solid color.
- bool [gslc_SetTransparentColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)
Configure the color to use for image transparency.
- bool [gslc_SetClipRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) *pRect)
Set the clipping rectangle for further drawing.

7.1.1 Detailed Description

General functions for configuring the GUI.

7.1.2 Function Documentation

7.1.2.1 void `gslc_DebugPrintf` (const char * *pFmt*, ...)

Optimized printf routine for GUIslice debug/error output.

- Only supports 's','d','u' tokens
- Calls on the output function configured in [gslc_InitDebug\(\)](#)

Parameters

in	<i>pFmt</i>	Format string to use for printing
in	...	Variable parameter list

Returns

none

7.1.2.2 void* `gslc_GetDriverDisp` (`gslc_tsGui` * *pGui*)

Get the native display driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the display driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

7.1.2.3 void* `gslc_GetDriverTouch` (`gslc_tsGui` * *pGui*)

Get the native touch driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the touch driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

7.1.2.4 const char* gslc_GetNameDisp (gslc_tsGui * pGui)

Get the GUIslice display driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

7.1.2.5 const char* gslc_GetNameTouch (gslc_tsGui * pGui)

Get the GUIslice touch driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

7.1.2.6 char* gslc_GetVer (gslc_tsGui * pGui)

Get the GUIslice version number.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing version number

7.1.2.7 `bool gslc_GuiRotate (gslc_tsGui * pGui, uint8_t nRotation)`

Dynamically change rotation, automatically adapt touchscreen axes swap/flip.

The function assumes that the touchscreen settings for swap and flip in the GUIslice config are valid for the configured `GSLC_ROTATE`.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)

Returns

true if success, false otherwise

7.1.2.8 `bool gslc_Init (gslc_tsGui * pGui, void * pvDriver, gslc_tsPage * asPage, uint8_t nMaxPage, gslc_tsFont * asFont, uint8_t nMaxFont)`

Initialize the GUIslice library.

- Configures the primary screen surface(s)
- Initializes font support

PRE:

- The environment variables should be configured before calling [gslc_Init\(\)](#).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pvDriver</i>	Void pointer to Driver struct (gslc_tsDriver*)
in	<i>asPage</i>	Pointer to Page array
in	<i>nMaxPage</i>	Size of Page array
in	<i>asFont</i>	Pointer to Font array
in	<i>nMaxFont</i>	Size of Font array

Returns

true if success, false if fail

7.1.2.9 `void gslc_InitDebug (GSLC_CB_DEBUG_OUT pfunc)`

Initialize debug output.

- Defines the user function used for debug/error output
- `pfunc` is responsible for outputting a single character
- For Arduino, this user function would typically call `Serial.print()`

Parameters

in	<i>pfunc</i>	Pointer to user character-out function
----	--------------	--

Returns

none

7.1.2.10 void gslc_Quit (gslc_tsGui * pGui)

Exit the GUIslice environment.

- Calls lower-level destructors to clean up any initialized subsystems and deletes any created elements or fonts

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

None

7.1.2.11 bool gslc_SetBkgndColor (gslc_tsGui * pGui, gslc_tsColor nCol)

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nCol</i>	RGB Color to use

Returns

true if success, false if fail

7.1.2.12 `bool gslc_SetBkgndImage (gslc_tsGui * pGui, gslc_tsImgRef sImgRef)`

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

7.1.2.13 `bool gslc_SetClipRect (gslc_tsGui * pGui, gslc_tsRect * pRect)`

Set the clipping rectangle for further drawing.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pRect</i>	Pointer to Rect for clipping (or NULL for entire screen)

Returns

true if success, false if error

7.1.2.14 `bool gslc_SetTransparentColor (gslc_tsGui * pGui, gslc_tsColor nCol)`

Configure the color to use for image transparency.

- Drawing a BMP with transparency enabled will cause regions in this specific color to appear transparent
- This API overrides the config option `GSLC_BMP_TRANS_RGB`

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nCol</i>	RGB Color to use

Returns

true if success, false if fail

7.1.2.15 void gslc_Update (gslc_tsGui * *pGui*)

Perform main GUIslice handling functions.

- Handles any touch events
- Performs any necessary screen redraw

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

None

7.2 Graphics General Functions

Helper functions that support graphics operations.

Functions

- bool [gslc_IsInRect](#) (int16_t nSelX, int16_t nSelY, [gslc_tsRect](#) rRect)
Determine if a coordinate is inside of a rectangular region.
- [gslc_tsRect](#) [gslc_ExpandRect](#) ([gslc_tsRect](#) rRect, int16_t nExpandW, int16_t nExpandH)
Expand or contract a rectangle in width and/or height (equal amounts on both side), based on the centerpoint of the rectangle.
- bool [gslc_IsInWH](#) (int16_t nSelX, int16_t nSelY, uint16_t nWidth, uint16_t nHeight)
Determine if a coordinate is inside of a width x height region.
- void [gslc_UnionRect](#) ([gslc_tsRect](#) *pRect, [gslc_tsRect](#) rAddRect)
Expand a rect to include another rect.
- void [gslc_InvalidateRgnReset](#) ([gslc_tsGui](#) *pGui)
Reset the invalidation region.
- void [gslc_InvalidateRgnPage](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage)
Include an entire page (eg.
- void [gslc_InvalidateRgnScreen](#) ([gslc_tsGui](#) *pGui)
Mark the entire screen as invalidated.
- void [gslc_InvalidateRgnAdd](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rAddRect)
Add a rectangular region to the invalidation region.
- bool [gslc_ClipPt](#) ([gslc_tsRect](#) *pClipRect, int16_t nX, int16_t nY)
Perform basic clipping of a single point to a clipping region.
- bool [gslc_ClipLine](#) ([gslc_tsRect](#) *pClipRect, int16_t *pnX0, int16_t *pnY0, int16_t *pnX1, int16_t *pnY1)
Perform basic clipping of a line to a clipping region.
- bool [gslc_ClipRect](#) ([gslc_tsRect](#) *pClipRect, [gslc_tsRect](#) *pRect)
Perform basic clipping of a rectangle to a clipping region.
- [gslc_tslmgRef](#) [gslc_GetImageFromFile](#) (const char *pFname, [gslc_telmgRefFlags](#) eFmt)
Create an image reference to a bitmap file in LINUX filesystem.
- [gslc_tslmgRef](#) [gslc_GetImageFromSD](#) (const char *pFname, [gslc_telmgRefFlags](#) eFmt)
Create an image reference to a bitmap file in SD card.
- [gslc_tslmgRef](#) [gslc_GetImageFromRam](#) (unsigned char *plmgBuf, [gslc_telmgRefFlags](#) eFmt)
Create an image reference to a bitmap in SRAM.
- [gslc_tslmgRef](#) [gslc_GetImageFromProg](#) (const unsigned char *plmgBuf, [gslc_telmgRefFlags](#) eFmt)
Create an image reference to a bitmap in program memory (PROGMEM)
- void [gslc_PolarToXY](#) (uint16_t nRad, int16_t n64Ang, int16_t *nDX, int16_t *nDY)
Convert polar coordinate to cartesian.
- int16_t [gslc_sinFX](#) (int16_t n64Ang)
Calculate fixed-point sine function from fractional degrees.
- int16_t [gslc_cosFX](#) (int16_t n64Ang)
Calculate fixed-point cosine function from fractional degrees.
- [gslc_tsColor](#) [gslc_ColorBlend2](#) ([gslc_tsColor](#) colStart, [gslc_tsColor](#) colEnd, uint16_t nMidAmt, uint16_t n↔BlendAmt)
Create a color based on a blend between two colors.
- [gslc_tsColor](#) [gslc_ColorBlend3](#) ([gslc_tsColor](#) colStart, [gslc_tsColor](#) colMid, [gslc_tsColor](#) colEnd, uint16_t n↔MidAmt, uint16_t nBlendAmt)
Create a color based on a blend between three colors.
- bool [gslc_ColorEqual](#) ([gslc_tsColor](#) a, [gslc_tsColor](#) b)
Check whether two colors are equal.

7.2.1 Detailed Description

Helper functions that support graphics operations.

7.2.2 Function Documentation

7.2.2.1 `bool gslc_ClipLine (gslc_tsRect * pClipRect, int16_t * pnX0, int16_t * pnY0, int16_t * pnX1, int16_t * pnY1)`

Perform basic clipping of a line to a clipping region.

- Implements Cohen-Sutherland algorithm
- Coordinates in parameter list are modified to fit the region

Parameters

in	<i>pClipRect</i>	Pointer to clipping region
in, out	<i>pnX0</i>	Ptr to X coordinate of line start
in, out	<i>pnY0</i>	Ptr to Y coordinate of line start
in, out	<i>pnX1</i>	Ptr to X coordinate of line end
in, out	<i>pnY1</i>	Ptr to Y coordinate of line end

Returns

true if line is visible, false if it should be discarded

7.2.2.2 `bool gslc_ClipPt (gslc_tsRect * pClipRect, int16_t nX, int16_t nY)`

Perform basic clipping of a single point to a clipping region.

Parameters

in	<i>pClipRect</i>	Pointer to clipping region
in	<i>nX</i>	X coordinate of point
in	<i>nY</i>	Y coordinate of point

Returns

true if point is visible, false if it should be discarded

7.2.2.3 `bool gslc_ClipRect (gslc_tsRect * pClipRect, gslc_tsRect * pRect)`

Perform basic clipping of a rectangle to a clipping region.

- Coordinates in parameter rect are modified to fit the region

Parameters

in	<i>pClipRect</i>	Pointer to clipping region
in, out	<i>pRect</i>	Ptr to rectangle

Returns

true if rect is visible, false if it should be discarded

7.2.2.4 `gslc_tsColor gslc_ColorBlend2 (gslc_tsColor colStart, gslc_tsColor colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)`

Create a color based on a blend between two colors.

Parameters

in	<i>colStart</i>	Starting color
in	<i>colEnd</i>	Ending color
in	<i>nMidAmt</i>	Position (0..1000) between start and end color at which the midpoint between colors should appear. Normally set to 500 (half-way).
in	<i>nBlendAmt</i>	The position (0..1000) between start and end at which we want to calculate the resulting blended color.

Returns

Blended color

7.2.2.5 `gslc_tsColor gslc_ColorBlend3 (gslc_tsColor colStart, gslc_tsColor colMid, gslc_tsColor colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)`

Create a color based on a blend between three colors.

Parameters

in	<i>colStart</i>	Starting color
in	<i>colMid</i>	Intermediate color
in	<i>colEnd</i>	Ending color
in	<i>nMidAmt</i>	Position (0..1000) between start and end color at which the intermediate color should appear.
in	<i>nBlendAmt</i>	The position (0..1000) between start and end at which we want to calculate the resulting blended color.

Returns

Blended color

7.2.2.6 `bool gslc_ColorEqual (gslc_tsColor a, gslc_tsColor b)`

Check whether two colors are equal.

Parameters

in	<i>a</i>	First color
in	<i>b</i>	Second color

Returns

True iff *a* and *b* are the same color.

7.2.2.7 `int16_t gslc_cosFX (int16_t n64Ang)`

Calculate fixed-point cosine function from fractional degrees.

- Depending on configuration, the result is derived from either floating point math library or fixed point lookup table.
- $\text{gslc_cosFX}(n\text{AngDeg} \cdot 64) / 32768.0 = \cos(n\text{AngDeg} \cdot 2\pi / 360)$

Parameters

in	<i>n64Ang</i>	Angle (in units of 1/64 degrees)
----	---------------	----------------------------------

Returns

Fixed-point cosine result. Signed 16-bit; divide by 32768 to get the actual value.

7.2.2.8 `gslc_tsRect gslc_ExpandRect (gslc_tsRect rRect, int16_t nExpandW, int16_t nExpandH)`

Expand or contract a rectangle in width and/or height (equal amounts on both side), based on the centerpoint of the rectangle.

Parameters

in	<i>rRect</i>	Rectangular region before resizing
in	<i>nExpandW</i>	Number of pixels to expand the width (if positive) or contract the width (if negative)
in	<i>nExpandH</i>	Number of pixels to expand the height (if positive) or contract the height (if negative)

Returns

`gslc_tsRect()` with resized dimensions

7.2.2.9 gslc_tslmgRef gslc_GetImageFromFile (const char * *pFname*, gslc_telmgRefFlags *eFmt*)

Create an image reference to a bitmap file in LINUX filesystem.

Parameters

in	<i>pFname</i>	Pointer to filename string of image in filesystem
in	<i>eFmt</i>	Image format

Returns

Loaded image reference

7.2.2.10 gslc_tslmgRef gslc_GetImageFromProg (const unsigned char * *plmgBuf*, gslc_telmgRefFlags *eFmt*)

Create an image reference to a bitmap in program memory (PROGMEM)

Parameters

in	<i>plmgBuf</i>	Pointer to image buffer in memory
in	<i>eFmt</i>	Image format

Returns

Loaded image reference

7.2.2.11 gslc_tslmgRef gslc_GetImageFromRam (unsigned char * *plmgBuf*, gslc_telmgRefFlags *eFmt*)

Create an image reference to a bitmap in SRAM.

Parameters

in	<i>plmgBuf</i>	Pointer to image buffer in memory
in	<i>eFmt</i>	Image format

Returns

Loaded image reference

7.2.2.12 gslc_tslmgRef gslc_GetImageFromSD (const char * *pFname*, gslc_telmgRefFlags *eFmt*)

Create an image reference to a bitmap file in SD card.

Parameters

in	<i>pFname</i>	Pointer to filename string of image in SD card
in	<i>eFmt</i>	Image format

Returns

Loaded image reference

7.2.2.13 void `gslc_InvalidateRgnAdd (gslc_tsGui * pGui, gslc_tsRect rAddRect)`

Add a rectangular region to the invalidation region.

- This is usually called when an element has been modified

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rAddRect</i>	Rectangle to add to the invalidation region

Returns

none

7.2.2.14 void `gslc_InvalidateRgnPage (gslc_tsGui * pGui, gslc_tsPage * pPage)`

Include an entire page (eg.

from a page stack) in the invalidation region

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pPage</i>	Pointer to page

Returns

none

7.2.2.15 void `gslc_InvalidateRgnReset (gslc_tsGui * pGui)`

Reset the invalidation region.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

7.2.2.16 void gslc_InvalidateRgnScreen (gslc_tsGui * pGui)

Mark the entire screen as invalidated.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

7.2.2.17 bool gslc_IsInRect (int16_t nSelX, int16_t nSelY, gslc_tsRect rRect)

Determine if a coordinate is inside of a rectangular region.

- This routine is useful in determining if a touch coordinate is inside of a button.

Parameters

in	<i>nSelX</i>	X coordinate to test
in	<i>nSelY</i>	X coordinate to test
in	<i>rRect</i>	Rectangular region to compare against

Returns

true if inside region, false otherwise

7.2.2.18 bool gslc_IsInWH (int16_t nSelX, int16_t nSelY, uint16_t nWidth, uint16_t nHeight)

Determine if a coordinate is inside of a width x height region.

- This routine is useful in determining if a relative coordinate is within a given W x H dimension

Parameters

in	<i>nSelX</i>	X coordinate to test
in	<i>nSelY</i>	X coordinate to test
in	<i>nWidth</i>	Width to test against
in	<i>nHeight</i>	Height to test against

Returns

true if inside region, false otherwise

7.2.2.19 void gslc_PolarToXY (uint16_t *nRad*, int16_t *n64Ang*, int16_t * *nDX*, int16_t * *nDY*)

Convert polar coordinate to cartesian.

Parameters

in	<i>nRad</i>	Radius of ray
in	<i>n64Ang</i>	Angle of ray (in units of 1/64 degrees, 0 is up)
out	<i>nDX</i>	X offset for ray end
out	<i>nDY</i>	Y offset for ray end

Returns

none

7.2.2.20 int16_t gslc_sinFX (int16_t *n64Ang*)

Calculate fixed-point sine function from fractional degrees.

- Depending on configuration, the result is derived from either floating point math library or fixed point lookup table.
- $\text{gslc_sinFX}(\text{nAngDeg} * 64) / 32768.0 = \sin(\text{nAngDeg} * 2\pi / 360)$

Parameters

in	<i>n64Ang</i>	Angle (in units of 1/64 degrees)
----	---------------	----------------------------------

Returns

Fixed-point sine result. Signed 16-bit; divide by 32768 to get the actual value.

7.2.2.21 void gslc_UnionRect (gslc_tsRect * *pRect*, gslc_tsRect *rAddRect*)

Expand a rect to include another rect.

- This routine can be useful to modify an invalidation region to include another modified element

Parameters

in	<i>pRect</i>	Initial rect region
in	<i>rAddRect</i>	Rectangle to add to the rect region

Returns

none

7.3 Graphics Primitive Functions

These routines cause immediate drawing to occur on the primary screen.

Functions

- void `gslc_DrawSetPixel` (`gslc_tsGui` *pGui, int16_t nX, int16_t nY, `gslc_tsColor` nCol)
Set a pixel on the active screen to the given color with lock.
- void `gslc_DrawLine` (`gslc_tsGui` *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, `gslc_tsColor` nCol)
Draw an arbitrary line using Bresenham's algorithm.
- void `gslc_DrawLineH` (`gslc_tsGui` *pGui, int16_t nX, int16_t nY, uint16_t nW, `gslc_tsColor` nCol)
Draw a horizontal line.
- void `gslc_DrawLineV` (`gslc_tsGui` *pGui, int16_t nX, int16_t nY, uint16_t nH, `gslc_tsColor` nCol)
Draw a vertical line.
- void `gslc_DrawLinePolar` (`gslc_tsGui` *pGui, int16_t nX, int16_t nY, uint16_t nRadStart, uint16_t nRadEnd, int16_t n64Ang, `gslc_tsColor` nCol)
Draw a polar ray segment.
- void `gslc_DrawFrameRect` (`gslc_tsGui` *pGui, `gslc_tsRect` rRect, `gslc_tsColor` nCol)
Draw a framed rectangle.
- void `gslc_DrawFrameRoundRect` (`gslc_tsGui` *pGui, `gslc_tsRect` rRect, int16_t nRadius, `gslc_tsColor` nCol)
Draw a framed rounded rectangle.
- void `gslc_DrawFillRect` (`gslc_tsGui` *pGui, `gslc_tsRect` rRect, `gslc_tsColor` nCol)
Draw a filled rectangle.
- void `gslc_DrawFillRoundRect` (`gslc_tsGui` *pGui, `gslc_tsRect` rRect, int16_t nRadius, `gslc_tsColor` nCol)
Draw a filled rounded rectangle.
- void `gslc_DrawFrameCircle` (`gslc_tsGui` *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, `gslc_tsColor` nCol)
Draw a framed circle.
- void `gslc_DrawFillCircle` (`gslc_tsGui` *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, `gslc_tsColor` nCol)
Draw a filled circle.
- void `gslc_DrawFrameTriangle` (`gslc_tsGui` *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, `gslc_tsColor` nCol)
Draw a framed triangle.
- void `gslc_DrawFillTriangle` (`gslc_tsGui` *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, `gslc_tsColor` nCol)
Draw a filled triangle.
- void `gslc_DrawFrameQuad` (`gslc_tsGui` *pGui, `gslc_tsPt` *psPt, `gslc_tsColor` nCol)
Draw a framed quadrilateral.
- void `gslc_DrawFillQuad` (`gslc_tsGui` *pGui, `gslc_tsPt` *psPt, `gslc_tsColor` nCol)
Draw a filled quadrilateral.
- void `gslc_DrawFillGradSector` (`gslc_tsGui` *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, `gslc_tsColor` cArcStart, `gslc_tsColor` cArcEnd, int16_t nAngSecStart, int16_t nAngSecEnd, int16_t nAngGradStart, int16_t nAngGradRange)
Draw a gradient filled sector of a circle with support for inner and outer radius.
- void `gslc_DrawFillSector` (`gslc_tsGui` *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, `gslc_tsColor` cArc, int16_t nAngSecStart, int16_t nAngSecEnd)
Draw a flat filled sector of a circle with support for inner and outer radius.

7.3.1 Detailed Description

These routines cause immediate drawing to occur on the primary screen.

7.3.2 Function Documentation

7.3.2.1 void `gslc_DrawFillCircle` (`gslc_tsGui * pGui`, `int16_t nMidX`, `int16_t nMidY`, `uint16_t nRadius`, `gslc_tsColor nCol`)

Draw a filled circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nMidX</i>	Center X coordinate
in	<i>nMidY</i>	Center Y coordinate
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value for the fill

Returns

none

7.3.2.2 void `gslc_DrawFillGradSector` (`gslc_tsGui * pGui`, `int16_t nQuality`, `int16_t nMidX`, `int16_t nMidY`, `int16_t nRad1`, `int16_t nRad2`, `gslc_tsColor cArcStart`, `gslc_tsColor cArcEnd`, `int16_t nAngSecStart`, `int16_t nAngSecEnd`, `int16_t nAngGradStart`, `int16_t nAngGradRange`)

Draw a gradient filled sector of a circle with support for inner and outer radius.

- Can be used to create a ring or pie chart
- Note that the gradient fill is defined by both the color stops (`cArcStart`..`cArcEnd`) as well as a gradient angular range (`nAngGradStart`..`nAngGradStart`+`nAngGradRange`). This gradient angular range can be differeng from the drawing angular range (`nAngSegStart`..`nAngSecEnd`) to enable more advanced control styling / updates.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nQuality</i>	Number of segments used to depict a full circle. The higher the value, the smoother the resulting arcs. A value of 72 provides 360/72=5 degrees per segment which is a reasonable compromise between smoothness and performance. Note that 360/nQuality should be an integer result, thus the allowable quality settings are: 360 (max quality), 180, 120, 90, 72, 60, 45, 40, 36 (low quality), etc.
in	<i>nMidX</i>	Midpoint X coordinate of circle
in	<i>nMidY</i>	Midpoint Y coordinate of circle
in	<i>nRad1</i>	Inner sector radius (0 for sector / pie, non-zero for ring)
in	<i>nRad2</i>	Outer sector radius. Delta from <i>nRad1</i> defines ring thickness.
in	<i>cArcStart</i>	Start color for gradient fill (with angular range defined by <i>nAngGradStart</i> , <i>nAngGradRange</i>)

Parameters

in	<i>cArcEnd</i>	End color for gradient fill
in	<i>nAngSecStart</i>	Angle of start of sector drawing (0 at top), measured in degrees.
in	<i>nAngSecEnd</i>	Angle of end of sector drawing (0 at top), measured in degrees.
in	<i>nAngGradStart</i>	For gradient fill, defines the starting angle associated with the starting color (cArcStart)
in	<i>nAngGradRange</i>	For gradient fill, defines the angular range associated with the start-to-end color range (cArcStart..cArcEnd)

Returns

none

7.3.2.3 void gslc_DrawFillQuad (gslc_tsGui * *pGui*, gslc_tsPt * *psPt*, gslc_tsColor *nCol*)

Draw a filled quadrilateral.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>psPt</i>	Pointer to array of 4 points
in	<i>nCol</i>	Color RGB value for the frame

Returns

true if success, false if error

7.3.2.4 void gslc_DrawFillRect (gslc_tsGui * *pGui*, gslc_tsRect *rRect*, gslc_tsColor *nCol*)

Draw a filled rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nCol</i>	Color RGB value to fill

Returns

none

7.3.2.5 void gslc_DrawFillRoundRect (gslc_tsGui * *pGui*, gslc_tsRect *rRect*, int16_t *nRadius*, gslc_tsColor *nCol*)

Draw a filled rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nRadius</i>	Radius for the rounded corners
in	<i>nCol</i>	Color RGB value to fill

Returns

none

7.3.2.6 void `gslc_DrawFillSector (gslc_tsGui * pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, gslc_tsColor cArc, int16_t nAngSecStart, int16_t nAngSecEnd)`

Draw a flat filled sector of a circle with support for inner and outer radius.

- Can be used to create a ring or pie chart

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nQuality</i>	Number of segments used to depict a full circle. The higher the value, the smoother the resulting arcs. A value of 72 provides 360/72=5 degrees per segment which is a reasonable compromise between smoothness and performance.
in	<i>nMidX</i>	Midpoint X coordinate of circle
in	<i>nMidY</i>	Midpoint Y coordinate of circle
in	<i>nRad1</i>	Inner sector radius (0 for sector / pie, non-zero for ring)
in	<i>nRad2</i>	Outer sector radius. Delta from nRad1 defines ring thickness.
in	<i>cArc</i>	Color for flat fill
in	<i>nAngSecStart</i>	Angle of start of sector drawing (0 at top), measured in degrees.
in	<i>nAngSecEnd</i>	Angle of end of sector drawing (0 at top), measured in degrees.

Returns

none

7.3.2.7 void `gslc_DrawFillTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a filled triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1

Parameters

in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value for the fill

Returns

true if success, false if error

7.3.2.8 void `gslc_DrawFrameCircle` (`gslc_tsGui * pGui`, `int16_t nMidX`, `int16_t nMidY`, `uint16_t nRadius`, `gslc_tsColor nCol`)

Draw a framed circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nMidX</i>	Center X coordinate
in	<i>nMidY</i>	Center Y coordinate
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value for the frame

Returns

none

7.3.2.9 void `gslc_DrawFrameQuad` (`gslc_tsGui * pGui`, `gslc_tsPt * psPt`, `gslc_tsColor nCol`)

Draw a framed quadrilateral.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>psPt</i>	Pointer to array of 4 points
in	<i>nCol</i>	Color RGB value for the frame

Returns

true if success, false if error

7.3.2.10 void `gslc_DrawFrameRect` (`gslc_tsGui * pGui`, `gslc_tsRect rRect`, `gslc_tsColor nCol`)

Draw a framed rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nCol</i>	Color RGB value for the frame

Returns

none

7.3.2.11 void `gslc_DrawFrameRoundRect` (`gslc_tsGui * pGui`, `gslc_tsRect rRect`, `int16_t nRadius`, `gslc_tsColor nCol`)

Draw a framed rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nRadius</i>	Radius for the rounded corners
in	<i>nCol</i>	Color RGB value for the frame

Returns

none

7.3.2.12 void `gslc_DrawFrameTriangle` (`gslc_tsGui * pGui`, `int16_t nX0`, `int16_t nY0`, `int16_t nX1`, `int16_t nY1`, `int16_t nX2`, `int16_t nY2`, `gslc_tsColor nCol`)

Draw a framed triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value for the frame

Returns

true if success, false if error

7.3.2.13 void `gslc_DrawLine` (`gslc_tsGui * pGui`, `int16_t nX0`, `int16_t nY0`, `int16_t nX1`, `int16_t nY1`, `gslc_tsColor nCol`)

Draw an arbitrary line using Bresenham's algorithm.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X coordinate of line startpoint
in	<i>nY0</i>	Y coordinate of line startpoint
in	<i>nX1</i>	X coordinate of line endpoint
in	<i>nY1</i>	Y coordinate of line endpoint
in	<i>nCol</i>	Color RGB value for the line

Returns

none

7.3.2.14 void `gslc_DrawLineH` (`gslc_tsGui * pGui`, `int16_t nX`, `int16_t nY`, `uint16_t nW`, `gslc_tsColor nCol`)

Draw a horizontal line.

- Note that direction of line is in +ve X axis

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	X coordinate of line startpoint
in	<i>nY</i>	Y coordinate of line startpoint
in	<i>nW</i>	Width of line (in +X direction)
in	<i>nCol</i>	Color RGB value for the line

Returns

none

7.3.2.15 void `gslc_DrawLinePolar` (`gslc_tsGui * pGui`, `int16_t nX`, `int16_t nY`, `uint16_t nRadStart`, `uint16_t nRadEnd`, `int16_t n64Ang`, `gslc_tsColor nCol`)

Draw a polar ray segment.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	X coordinate of line startpoint
in	<i>nY</i>	Y coordinate of line startpoint
in	<i>nRadStart</i>	Starting radius of line

Parameters

in	<i>nRadEnd</i>	Ending radius of line
in	<i>n64Ang</i>	Angle of ray (degrees * 64). 0 is up, +90*64 is to right From -180*64 to +180*64
in	<i>nCol</i>	Color RGB value for the line

Returns

none

7.3.2.16 void gslc_DrawLineV (gslc_tsGui * *pGui*, int16_t *nX*, int16_t *nY*, uint16_t *nH*, gslc_tsColor *nCol*)

Draw a vertical line.

- Note that direction of line is in +ve Y axis

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	X coordinate of line startpoint
in	<i>nY</i>	Y coordinate of line startpoint
in	<i>nH</i>	Height of line (in +Y direction)
in	<i>nCol</i>	Color RGB value for the line

Returns

none

7.3.2.17 void gslc_DrawSetPixel (gslc_tsGui * *pGui*, int16_t *nX*, int16_t *nY*, gslc_tsColor *nCol*)

Set a pixel on the active screen to the given color with lock.

- Calls upon gslc_DrvDrawSetPixelRaw() but wraps with a surface lock lock
- If repeated access is needed, use gslc_DrvDrawSetPixelRaw() instead

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	Pixel X coordinate to set
in	<i>nY</i>	Pixel Y coordinate to set
in	<i>nCol</i>	Color pixel value to assign

Returns

none

7.4 Font Functions

Functions that load fonts.

Functions

- `bool gslc_FontAdd (gslc_tsGui *pGui, int16_t nFontId, gslc_tFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)`
Load a font into the local font cache and assign font ID (nFontId).
- `bool gslc_FontSet (gslc_tsGui *pGui, int16_t nFontId, gslc_tFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)`
Load a font into the local font cache and store as font ID (nFontId)
- `gslc_tsFont * gslc_FontGet (gslc_tsGui *pGui, int16_t nFontId)`
Fetch a font from its ID value.
- `bool gslc_FontSetMode (gslc_tsGui *pGui, int16_t nFontId, gslc_tFontRefMode eFontMode)`
Set the font operating mode.

7.4.1 Detailed Description

Functions that load fonts.

7.4.2 Function Documentation

7.4.2.1 `bool gslc_FontAdd (gslc_tsGui * pGui, int16_t nFontId, gslc_tFontRefType eFontRefType, const void * pvFontRef, uint16_t nFontSz)`

Load a font into the local font cache and assign font ID (nFontId).

- Font is stored into next available internal array element
- NOTE: Use FontSet() instead

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nFontId</i>	ID to use when referencing this font
in	<i>eFontRefType</i>	Font reference type (eg. filename or pointer)
in	<i>pvFontRef</i>	Reference pointer to identify the font. In the case of SDL mode, it is a filepath to the font file. In the case of Arduino it is a pointer value to the font bitmap array (GFXFont)
in	<i>nFontSz</i>	Typeface size to use (only used in SDL mode)

Returns

true if load was successful, false otherwise

7.4.2.2 `gslc_tsFont*` `gslc_FontGet (gslc_tsGui * pGui, int16_t nFontId)`

Fetch a font from its ID value.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nFontId</i>	ID value used to reference the font (supplied originally to gslc_FontAdd())

Returns

A pointer to the font structure or NULL if error

7.4.2.3 `bool` `gslc_FontSet (gslc_tsGui * pGui, int16_t nFontId, gslc_teFontRefType eFontRefType, const void * pvFontRef, uint16_t nFontSz)`

Load a font into the local font cache and store as font ID (*nFontId*)

- Font is stored into index *nFontId*, so *nFontId* must be from separate font enum (0-based).
- Example: enum { E_FONT_BTN, E_FONT_TXT, MAX_FONT };

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nFontId</i>	ID to use when referencing this font
in	<i>eFontRefType</i>	Font reference type (eg. filename or pointer)
in	<i>pvFontRef</i>	Reference pointer to identify the font. In the case of SDL mode, it is a filepath to the font file. In the case of Arduino it is a pointer value to the font bitmap array (GFXFont)
in	<i>nFontSz</i>	Typeface size to use (only used in SDL mode)

Returns

true if load was successful, false otherwise

7.4.2.4 `bool` `gslc_FontSetMode (gslc_tsGui * pGui, int16_t nFontId, gslc_teFontRefMode eFontMode)`

Set the font operating mode.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nFontId</i>	ID value used to reference the font (supplied originally to gslc_FontAdd())
in, out	<i>eFontMode</i>	Font mode to assign to this font

Returns

true if success

7.5 Page Functions

Functions that operate at the page level.

Functions

- `int gslc_GetPageCur (gslc_tsGui *pGui)`
Fetch the current page ID.
- `void gslc_SetStackPage (gslc_tsGui *pGui, uint8_t nStackPos, int16_t nPageld)`
Assign a page to the page stack.
- `void gslc_SetStackState (gslc_tsGui *pGui, uint8_t nStackPos, bool bActive, bool bDoDraw)`
Change the status of a page in a page stack.
- `void gslc_SetPageBase (gslc_tsGui *pGui, int16_t nPageld)`
Assigns a page for the base layer in the page stack.
- `void gslc_SetPageCur (gslc_tsGui *pGui, int16_t nPageld)`
Select a page for the current layer in the page stack.
- `void gslc_SetPageOverlay (gslc_tsGui *pGui, int16_t nPageld)`
Select a page for the overlay layer in the page stack.
- `void gslc_PopupShow (gslc_tsGui *pGui, int16_t nPageld, bool bModal)`
Show a popup dialog.
- `void gslc_PopupHide (gslc_tsGui *pGui)`
Hides the currently active popup dialog.
- `void gslc_PageRedrawSet (gslc_tsGui *pGui, bool bRedraw)`
Update the need-redraw status for the current page.
- `bool gslc_PageRedrawGet (gslc_tsGui *pGui)`
Get the need-redraw status for the current page.
- `void gslc_PageAdd (gslc_tsGui *pGui, int16_t nPageld, gslc_tsElem *psElem, uint16_t nMaxElem, gslc_tsElemRef *psElemRef, uint16_t nMaxElemRef)`
Add a page to the GUI.
- `gslc_tsElemRef * gslc_PageFindElemById (gslc_tsGui *pGui, int16_t nPageld, int16_t nElemId)`
Find an element in the GUI by its Page ID and Element ID.

7.5.1 Detailed Description

Functions that operate at the page level.

7.5.2 Function Documentation

7.5.2.1 `int gslc_GetPageCur (gslc_tsGui * pGui)`

Fetch the current page ID.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Page ID

```
7.5.2.2 void gslc_PageAdd ( gslc_tsGui * pGui, int16_t nPageld, gslc_tsElem * psElem, uint16_t nMaxElem,
gslc_tsElemRef * psElemRef, uint16_t nMaxElemRef )
```

Add a page to the GUI.

- This call associates an element array with the collection within the page
- Once a page has been added to the GUI, elements can be added to the page by specifying the same page ID

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nPageld</i>	Page ID to assign
in	<i>psElem</i>	Internal element array storage to associate with the page
in	<i>nMaxElem</i>	Maximum number of elements that can be added to the internal element array (ie. RAM))
in	<i>psElemRef</i>	Internal element reference array storage to associate with the page. All elements, whether they are located in the internal element array or in external Flash (PROGMEM) storage, require an entry in the element reference array.
in	<i>nMaxElemRef</i>	Maximum number of elements in the reference array. This is effectively the maximum number of elements that can appear on a page, irrespective of whether it is stored in RAM or Flash (PROGMEM).

Returns

none

```
7.5.2.3 gslc_tsElemRef* gslc_PageFindElemById ( gslc_tsGui * pGui, int16_t nPageld, int16_t nElemId )
```

Find an element in the GUI by its Page ID and Element ID.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ Pageld</i>	Page ID to search
in	<i>n↔ ElemId</i>	Element ID to search

Returns

Ptr to an element or NULL if none found

7.5.2.4 bool gslc_PageRedrawGet (gslc_tsGui * *pGui*)

Get the need-redraw status for the current page.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

True if redraw required, false otherwise

7.5.2.5 void gslc_PageRedrawSet (gslc_tsGui * *pGui*, bool *bRedraw*)

Update the need-redraw status for the current page.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>bRedraw</i>	True if redraw required, false otherwise

Returns

none

7.5.2.6 void gslc_PopupHide (gslc_tsGui * *pGui*)

Hides the currently active popup dialog.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

7.5.2.7 void gslc_PopupShow (gslc_tsGui * *pGui*, int16_t *nPageId*, bool *bModal*)

Show a popup dialog.

- Popup dialogs use the overlay layer in the page stack

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ Pageld</i>	Page ID to use as the popup dialog
in	<i>bModal</i>	If true, popup is modal (other layers won't accept touch). If false, popup is modeless (other layers still accept touch)

Returns

none

7.5.2.8 void gslc_SetPageBase (gslc_tsGui * *pGui*, int16_t *nPageld*)

Assigns a page for the base layer in the page stack.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ Pageld</i>	Page ID to select (or GSLC_PAGE_NONE to disable)

Returns

none

7.5.2.9 void gslc_SetPageCur (gslc_tsGui * *pGui*, int16_t *nPageld*)

Select a page for the current layer in the page stack.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ Pageld</i>	Page ID to select

Returns

none

7.5.2.10 void gslc_SetPageOverlay (gslc_tsGui * *pGui*, int16_t *nPageld*)

Select a page for the overlay layer in the page stack.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n← Pageld</i>	Page ID to select (or GSLC_PAGE_NONE to disable)

Returns

none

7.5.2.11 void gslc_SetStackPage (gslc_tsGui * *pGui*, uint8_t *nStackPos*, int16_t *nPageld*)

Assign a page to the page stack.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nStackPos</i>	Position to update in the page stack (0..GSLC_STACK__MAX-1)
in	<i>nPageld</i>	Page ID to select as current

Returns

none

7.5.2.12 void gslc_SetStackState (gslc_tsGui * *pGui*, uint8_t *nStackPos*, bool *bActive*, bool *bDoDraw*)

Change the status of a page in a page stack.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nStackPos</i>	Position to update in the page stack (0..GSLC_STACK__MAX-1)
in	<i>bActive</i>	Indicate if page should receive touch events
in	<i>bDoDraw</i>	Indicate if page should continue to be redrawn. If pages in the stack are overlapping and an element in a lower layer continues to receive updates, then the element may "show through" the layers above it. In such cases where pages in the stack are overlapping and lower pages contain dynamically updating elements, it may be best to disable redraw while the overlapping page is visible (by setting bDoDraw to false).

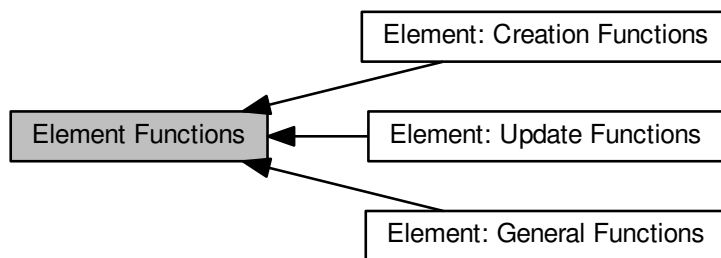
Returns

none

7.6 Element Functions

Functions that are used to create and manipulate elements.

Collaboration diagram for Element Functions:



Modules

- [Element: Creation Functions](#)
Functions that create GUI elements.
- [Element: General Functions](#)
General-purpose functions that operate on Elements.
- [Element: Update Functions](#)
Functions that configure or modify an existing element.

7.6.1 Detailed Description

Functions that are used to create and manipulate elements.

7.7 Element: Creation Functions

Functions that create GUI elements.

Collaboration diagram for Element: Creation Functions:



Functions

- [gslc_tsElemRef](#) * [gslc_ElemCreateTxt](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)
Create a Text Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateBtnTxt](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId, [GSLC_CB_TOUCH](#) cbTouch)
Create a textual Button Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateBtnImg](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem, [gslc_tsImgRef](#) sImgRef, [gslc_tsImgRef](#) sImgRefSel, [GSLC_CB_TOUCH](#) cbTouch)
Create a graphical Button Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateBox](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem)
Create a Box Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateLine](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1)
Create a Line Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateImg](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem, [gslc_tsImgRef](#) sImgRef)
Create an image Element.

7.7.1 Detailed Description

Functions that create GUI elements.

7.7.2 Function Documentation

7.7.2.1 [gslc_tsElemRef](#)* [gslc_ElemCreateBox](#) ([gslc_tsGui](#) * pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem)

Create a Box Element.

- Draws a box with frame and fill

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>rElem</i>	Rectangle coordinates defining box size

Returns

Pointer to the Element reference or NULL if failure

7.7.2.2 `gslc_tsElemRef* gslc_ElemCreateBtnImg (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, gslc_tsImgRef sImgRef, gslc_tsImgRef sImgRefSel, GSLC_CB_TOUCH cbTouch)`

Create a graphical Button Element.

- Creates a clickable element that uses a BMP image with no frame or fill
- Transparency is supported by bitmap color (0xFF00FF)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>rElem</i>	Rectangle coordinates defining image size
in	<i>sImgRef</i>	Image reference to load (unselected state)
in	<i>sImgRefSel</i>	Image reference to load (selected state)
in	<i>cbTouch</i>	Callback for touch events

Returns

Pointer to the Element reference or NULL if failure

7.7.2.3 `gslc_tsElemRef* gslc_ElemCreateBtnTxt (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, char * pStrBuf, uint8_t nStrBufMax, int16_t nFontId, GSLC_CB_TOUCH cbTouch)`

Create a textual Button Element.

- Creates a clickable element that has a textual label with frame and fill

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)

Parameters

in	<i>nPage</i>	Page ID to attach element to
in	<i>rElem</i>	Rectangle coordinates defining text background size
in	<i>pStrBuf</i>	String to copy into element
in	<i>nStrBufMax</i>	Maximum length of string buffer (<i>pStrBuf</i>). Only applicable if GSLC_LOCAL_STR=0. Ignored if GSLC_LOCAL_STR=1.)
in	<i>nFontId</i>	Font ID to use for text display
in	<i>cbTouch</i>	Callback for touch events

Returns

Pointer to the Element reference or NULL if failure

7.7.2.4 `gslc_tsElemRef* gslc_ElemCreateImg (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, gslc_tsImgRef sImgRef)`

Create an image Element.

- Draws an image

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ ElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>rElem</i>	Rectangle coordinates defining box size
in	<i>sImgRef</i>	Image reference to load

Returns

Pointer to the Element reference or NULL if failure

7.7.2.5 `gslc_tsElemRef* gslc_ElemCreateLine (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1)`

Create a Line Element.

- Draws a line with fill color

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ ElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)

Parameters

in	<i>nPage</i>	Page ID to attach element to
in	<i>nX0</i>	X coordinate of line startpoint
in	<i>nY0</i>	Y coordinate of line startpoint
in	<i>nX1</i>	X coordinate of line endpoint
in	<i>nY1</i>	Y coordinate of line endpoint

Returns

Pointer to the Element reference or NULL if failure

7.7.2.6 `gslc_tsElemRef* gslc_ElemCreateTxt (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, char * pStrBuf, uint8_t nStrBufMax, int16_t nFontId)`

Create a Text Element.

- Draws a text string with filled background

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>rElem</i>	Rectangle coordinates defining text background size
in	<i>pStrBuf</i>	String to copy into element
in	<i>nStrBufMax</i>	Maximum length of string buffer (pStrBuf). Only applicable if GSLC_LOCAL_STR=0. Ignored if GSLC_LOCAL_STR=1.)
in	<i>nFontId</i>	Font ID to use for text display

Returns

Pointer to the Element reference or NULL if failure

7.8 Element: General Functions

General-purpose functions that operate on Elements.

Collaboration diagram for Element: General Functions:



Functions

- `int gslc_ElemGetId (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
Get an Element ID from an element structure.

7.8.1 Detailed Description

General-purpose functions that operate on Elements.

7.8.2 Function Documentation

7.8.2.1 `int gslc_ElemGetId (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get an Element ID from an element structure.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference structure

Returns

ID of element or `GSLC_ID_NONE` if not found

7.9 Element: Update Functions

Functions that configure or modify an existing element.

Collaboration diagram for Element: Update Functions:



Functions

- void [gslc_ElemSetFillEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bFillEn)
Set the fill state for an Element.
- void [gslc_ElemSetFrameEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bFrameEn)
Set the frame state for an Element.
- void [gslc_ElemSetRoundEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bRoundEn)
Set the rounded frame/fill state for an Element.
- void [gslc_ElemSetCol](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colFrame, [gslc_tsColor](#) colFill, [gslc_tsColor](#) colFillGlow)
Update the common color selection for an Element.
- void [gslc_ElemSetGlowCol](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colFrameGlow, [gslc_tsColor](#) colFillGlow, [gslc_tsColor](#) colTxtGlow)
Update the common color selection for glowing state of an Element.
- void [gslc_ElemSetGroup](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int nGroupId)
Set the group ID for an element.
- int [gslc_ElemGetGroup](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the group ID for an element.
- void [gslc_ElemSetTxtAlign](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, unsigned nAlign)
Set the alignment of a textual element (horizontal and vertical)
- void [gslc_ElemSetTxtMargin](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, unsigned nMargin)
Set the margin around of a textual element.
- void [gslc_ElemSetTxtMarginXY](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int8_t nMarginX, int8_t nMarginY)
Set the margin around of a textual element (X & Y offsets can be different)
- void [gslc_ElemSetTxtStr](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, const char *pStr)
Update the text string associated with an Element.
- char * [gslc_ElemGetTxtStr](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Fetch the current text string associated with an Element.
- void [gslc_ElemSetTxtCol](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colVal)
Update the text string color associated with an Element ID.
- void [gslc_ElemSetTxtMem](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teTxtFlags](#) eFlags)
Update the text string location in memory.
- void [gslc_ElemSetTxtEnc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teTxtFlags](#) eFlags)
Update the text string encoding mode.

- void [gslc_ElemUpdateFont](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int nFontId)
Update the Font selected for an Element's text.
- void [gslc_ElemSetRedraw](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) eRedraw)
Update the need-redraw status for an element.
- [gslc_teRedrawType](#) [gslc_ElemGetRedraw](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the need-redraw status for an element.
- void [gslc_ElemSetGlowEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bGlowEn)
Update the glowing enable for an element.
- void [gslc_ElemSetClickEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bClickEn)
Update the click enable for an element.
- void [gslc_ElemSetTouchFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_TOUCH](#) funcCb)
Update the touch function callback for an element.
- void [gslc_ElemSetStyleFrom](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRefSrc, [gslc_tsElemRef](#) *pElemRefDest)
Copy style settings from one element to another.
- bool [gslc_ElemGetGlowEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the glowing enable for an element.
- void [gslc_ElemSetGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bGlowing)
Update the glowing indicator for an element.
- bool [gslc_ElemGetGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the glowing indicator for an element.
- void [gslc_ElemSetVisible](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bVisible)
Update the visibility status for an element.
- bool [gslc_ElemGetVisible](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the visibility status for an element.
- bool [gslc_ElemGetOnScreen](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Determine whether an element is visible on the screen.
- void [gslc_ElemSetDrawFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_DRAW](#) funcCb)
Assign the drawing callback function for an element.
- void [gslc_ElemSetTickFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_TICK](#) funcCb)
Assign the tick callback function for an element.
- bool [gslc_ElemOwnsCoord](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nX, int16_t nY, bool bOnlyClickEn)
Determine if a coordinate is inside of an element.

7.9.1 Detailed Description

Functions that configure or modify an existing element.

7.9.2 Function Documentation

7.9.2.1 bool [gslc_ElemGetGlow](#) ([gslc_tsGui](#) * pGui, [gslc_tsElemRef](#) * pElemRef)

Get the glowing indicator for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

True if element is glowing

7.9.2.2 `bool gslc_ElemGetGlowEn (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get the glowing enable for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

True if element supports glowing

7.9.2.3 `int gslc_ElemGetGroup (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get the group ID for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Group ID or GSLC_GROUP_ID_NONE if unassigned

7.9.2.4 `bool gslc_ElemGetOnScreen (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Determine whether an element is visible on the screen.

- This function takes into account both the element's "Visible" state as well as whether the element's associated page is active in the page stack.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

True if element appears on the screen, false otherwise

7.9.2.5 `gslc_teRedrawType` `gslc_ElemGetRedraw (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get the need-redraw status for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Redraw status

7.9.2.6 `char*` `gslc_ElemGetTxtStr (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Fetch the current text string associated with an Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Pointer to character array string

7.9.2.7 `bool` `gslc_ElemGetVisible (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get the visibility status for an element.

- Note that the visibility state is independent of whether or not the page associated with the element is actively displayed.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

True if element is marked as visible, false if hidden

7.9.2.8 `bool gslc_ElemOwnsCoord (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nX, int16_t nY, bool bOnlyClickEn)`

Determine if a coordinate is inside of an element.

- This routine is useful in determining if a touch coordinate is inside of a button.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Element reference used for boundary test
in	<i>nX</i>	X coordinate to test
in	<i>nY</i>	Y coordinate to test
in	<i>bOnlyClickEn</i>	Only output true if element was also marked as "clickable" (eg. bClickEn=true)

Returns

true if inside element, false otherwise

7.9.2.9 `void gslc_ElemSetClickEn (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bClickEn)`

Update the click enable for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bClickEn</i>	True if element should support click events

Returns

none

7.9.2.10 `void gslc_ElemSetCol (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tsColor colFrame, gslc_tsColor colFill, gslc_tsColor colFillGlow)`

Update the common color selection for an Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colFrame</i>	Color for the frame
in	<i>colFill</i>	Color for the fill
in	<i>colFillGlow</i>	Color for the fill when glowing

Returns

none

7.9.2.11 void `gslc_ElemSetDrawFunc (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, GSLC_CB_DRAW funcCb)`

Assign the drawing callback function for an element.

- This allows the user to override the default rendering for an element, enabling the creation of a custom element

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>funcCb</i>	Function pointer to drawing routine (or NULL for default))

Returns

none

7.9.2.12 void `gslc_ElemSetFillEn (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bFillEn)`

Set the fill state for an Element.

- If not filled, the element can support transparency against an arbitrary background, but this can require full screen redraws if the element is updated.
- If filled, the background fill color can be changed by [gslc_ElemSetCol\(\)](#)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bFillEn</i>	True if filled, false otherwise

Returns

none

7.9.2.13 void gslc_ElemSetFrameEn (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bFrameEn*)

Set the frame state for an Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bFrameEn</i>	True if framed, false otherwise

Returns

none

7.9.2.14 void gslc_ElemSetGlow (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bGlowing*)

Update the glowing indicator for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bGlowing</i>	True if element is glowing

Returns

none

7.9.2.15 void gslc_ElemSetGlowCol (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colFrameGlow*, gslc_tsColor *colFillGlow*, gslc_tsColor *colTxtGlow*)

Update the common color selection for glowing state of an Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colFrameGlow</i>	Color for the frame when glowing
in	<i>colFillGlow</i>	Color for the fill when glowing
in	<i>colTxtGlow</i>	Color for the text when glowing

Returns

none

7.9.2.16 void `gslc_ElemSetGlowEn` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, bool *bGlowEn*)

Update the glowing enable for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bGlowEn</i>	True if element should support glowing

Returns

none

7.9.2.17 void `gslc_ElemSetGroup` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, int *nGroupId*)

Set the group ID for an element.

- Typically used to associate radio button elements together

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nGroupId</i>	Group ID to assign

Returns

none

7.9.2.18 void `gslc_ElemSetRedraw` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `gslc_teRedrawType` *eRedraw*)

Update the need-redraw status for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>eRedraw</i>	Redraw state to set

Returns

none

7.9.2.19 void `gslc_ElemSetRoundEn (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bRoundEn)`

Set the rounded frame/fill state for an Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bRoundEn</i>	True if rounded, false otherwise

Returns

none

7.9.2.20 void `gslc_ElemSetStyleFrom (gslc_tsGui * pGui, gslc_tsElemRef * pElemRefSrc, gslc_tsElemRef * pElemRefDest)`

Copy style settings from one element to another.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRefSrc</i>	Pointer to source Element reference
in	<i>pElemRefDest</i>	Pointer to destination Element reference

Returns

none

7.9.2.21 void `gslc_ElemSetTickFunc (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, GSLC_CB_TICK funcCb)`

Assign the tick callback function for an element.

- This allows the user to provide background updates to an element triggered by the main loop call to [gslc_↔ Update\(\)](#)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>funcCb</i>	Function pointer to tick routine (or NULL for none))

Returns

none

7.9.2.22 void gslc_ElemSetTouchFunc (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, GSLC_CB_TOUCH *funcCb*)

Update the touch function callback for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>funcCb</i>	Pointer to the touch callback function

Returns

none

7.9.2.23 void gslc_ElemSetTxtAlign (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, unsigned *nAlign*)

Set the alignment of a textual element (horizontal and vertical)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nAlign</i>	Alignment to specify: <ul style="list-style-type: none"> • GSLC_ALIGN_TOP_LEFT • GSLC_ALIGN_TOP_MID • GSLC_ALIGN_TOP_RIGHT • GSLC_ALIGN_MID_LEFT • GSLC_ALIGN_MID_MID • GSLC_ALIGN_MID_RIGHT • GSLC_ALIGN_BOT_LEFT • GSLC_ALIGN_BOT_MID • GSLC_ALIGN_BOT_RIGHT

Returns

none

7.9.2.24 `void gslc_ElemSetTxtCol (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tsColor colVal)`

Update the text string color associated with an Element ID.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colVal</i>	RGB color to change to

Returns

none

7.9.2.25 void gslc_ElemSetTxtEnc (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_teTxtFlags *eFlags*)

Update the text string encoding mode.

- This function can be used to indicate that the element's text string is encoded in UTF-8, which supports extended / foreign character maps

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>eFlags</i>	Flags associated with text encoding (GSLC_TXT_ENC_*)

Returns

none

7.9.2.26 void gslc_ElemSetTxtMargin (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, unsigned *nMargin*)

Set the margin around of a textual element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nMargin</i>	Number of pixels gap to leave surrounding text

Returns

none

7.9.2.27 void gslc_ElemSetTxtMarginXY (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int8_t *nMarginX*, int8_t *nMarginY*)

Set the margin around of a textual element (X & Y offsets can be different)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nMarginX</i>	Number of pixels gap to offset text horizontally
in	<i>nMarginY</i>	Number of pixels gap to offset text vertically

Returns

none

7.9.2.28 void gslc_ElemSetTxtMem (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_teTxtFlags *eFlags*)

Update the text string location in memory.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>eFlags</i>	Flags associated with text memory location (GSLC_TXT_MEM_*)

Returns

none

7.9.2.29 void gslc_ElemSetTxtStr (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, const char * *pStr*)

Update the text string associated with an Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>pStr</i>	String to copy into element

Returns

none

7.9.2.30 void gslc_ElemSetVisible (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bVisible*)

Update the visibility status for an element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bVisible</i>	True if element is shown, false if hidden

Returns

none

7.9.2.31 void gslc_ElemUpdateFont (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int *nFontId*)

Update the Font selected for an Element's text.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nFontId</i>	Font ID to select

Returns

none

7.10 Touchscreen Functions

Functions that configure and respond to a touch device.

Macros

- `#define TOUCH_ROTATION_DATA`
Additional definitions for Touch Handling These macros define the transforms used in remapping the touchscreen inputs on the basis of the GUI nRotation setting.
- `#define TOUCH_ROTATION_DATA`
Additional definitions for Touch Handling These macros define the transforms used in remapping the touchscreen inputs on the basis of the GUI nRotation setting.
- `#define TOUCH_ROTATION_SWAPXY(rotation)`
- `#define TOUCH_ROTATION_SWAPXY(rotation)`
- `#define TOUCH_ROTATION_FLIPX(rotation)`
- `#define TOUCH_ROTATION_FLIPX(rotation)`
- `#define TOUCH_ROTATION_FLIPY(rotation)`
- `#define TOUCH_ROTATION_FLIPY(rotation)`

Functions

- `bool gslc_InitTouch (gslc_tsGui *pGui, const char *acDev)`
Initialize the touchscreen device driver.
- `bool gslc_GetTouch (gslc_tsGui *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, gslc_telInputRawEvent *peInputEvent, int16_t *pnInputVal)`
Initialize the touchscreen device driver.
- `void gslc_SetTouchRemapEn (gslc_tsGui *pGui, bool bEn)`
Configure touchscreen remapping.
- `void gslc_SetTouchRemapCal (gslc_tsGui *pGui, uint16_t nXMin, uint16_t nXMax, uint16_t nYMin, uint16_t nYMax)`
Configure touchscreen calibration values.
- `void gslc_SetTouchRemapYX (gslc_tsGui *pGui, bool bSwap)`
Configure touchscreen XY swap.

7.10.1 Detailed Description

Functions that configure and respond to a touch device.

7.10.2 Macro Definition Documentation

7.10.2.1 `#define TOUCH_ROTATION_DATA`

Additional definitions for Touch Handling These macros define the transforms used in remapping the touchscreen inputs on the basis of the GUI nRotation setting.

7.10.2.2 `#define TOUCH_ROTATION_DATA`

Additional definitions for Touch Handling These macros define the transforms used in remapping the touchscreen inputs on the basis of the GUI `nRotation` setting.

7.10.2.3 `#define TOUCH_ROTATION_FLIPX(rotation)`7.10.2.4 `#define TOUCH_ROTATION_FLIPY(rotation)`7.10.2.5 `#define TOUCH_ROTATION_FLIPY(rotation)`7.10.2.6 `#define TOUCH_ROTATION_FLIPY(rotation)`7.10.2.7 `#define TOUCH_ROTATION_SWAPXY(rotation)`7.10.2.8 `#define TOUCH_ROTATION_SWAPXY(rotation)`

7.10.3 Function Documentation

7.10.3.1 `bool gslc_GetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress, gslc_tInputRawEvent * pInputEvent, int16_t * pnInputVal)`

Initialize the touchscreen device driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
out	<i>pnX</i>	Ptr to int to contain latest touch X coordinate
out	<i>pnY</i>	Ptr to int to contain latest touch Y coordinate
out	<i>pnPress</i>	Ptr to int to contain latest touch pressure value
out	<i>pInputEvent</i>	Indication of event type
out	<i>pnInputVal</i>	Additional data for event type

Returns

true if touch event, false otherwise

7.10.3.2 `bool gslc_InitTouch (gslc_tsGui * pGui, const char * acDev)`

Initialize the touchscreen device driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen (or "" if not applicable)) eg. "/dev/input/touchscreen"

Returns

true if successful

7.10.3.3 void `gslc_SetTouchRemapCal` (`gslc_tsGui` * *pGui*, uint16_t *nXMin*, uint16_t *nXMax*, uint16_t *nYMin*, uint16_t *nYMax*)

Configure touchscreen calibration values.

- Only used if calibration remapping has been enabled

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nXMin</i>	Resistive touchscreen X_MIN calibration value
in	<i>nXMax</i>	Resistive touchscreen X_MAX calibration value
in	<i>nYMin</i>	Resistive touchscreen Y_MIN calibration value
in	<i>nYMax</i>	Resistive touchscreen Y_MAX calibration value

Returns

none

7.10.3.4 void `gslc_SetTouchRemapEn` (`gslc_tsGui` * *pGui*, bool *bEn*)

Configure touchscreen remapping.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>bEn</i>	Enable touchscreen remapping?

Returns

none

7.10.3.5 void `gslc_SetTouchRemapYX` (`gslc_tsGui` * *pGui*, bool *bSwap*)

Configure touchscreen XY swap.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>bSwap</i>	Enable touchscreen XY swap

Returns

none

7.11 Input Mapping Functions

Functions that handle GPIO / pin and keyboard input.

Functions

- void [gslc_SetPinPollFunc](#) ([gslc_tsGui](#) *pGui, [GSLC_CB_PIN_POLL](#) pfunc)
- void [gslc_InitInputMap](#) ([gslc_tsGui](#) *pGui, [gslc_tsInputMap](#) *asInputMap, uint8_t nInputMapMax)
- void [gslc_InputMapAdd](#) ([gslc_tsGui](#) *pGui, [gslc_teInputRawEvent](#) eInputEvent, int16_t nInputVal, [gslc_teAction](#) eAction, int16_t nActionVal)

7.11.1 Detailed Description

Functions that handle GPIO / pin and keyboard input.

7.11.2 Function Documentation

7.11.2.1 void [gslc_InitInputMap](#) ([gslc_tsGui](#) * *pGui*, [gslc_tsInputMap](#) * *asInputMap*, uint8_t *nInputMapMax*)

Todo Doc. This API is experimental and subject to change

7.11.2.2 void [gslc_InputMapAdd](#) ([gslc_tsGui](#) * *pGui*, [gslc_teInputRawEvent](#) *eInputEvent*, int16_t *nInputVal*, [gslc_teAction](#) *eAction*, int16_t *nActionVal*)

Todo Doc. This API is experimental and subject to change

7.11.2.3 void [gslc_SetPinPollFunc](#) ([gslc_tsGui](#) * *pGui*, [GSLC_CB_PIN_POLL](#) *pfunc*)

Todo Doc. This API is experimental and subject to change

7.12 General Purpose Macros

Macros that are used throughout the GUI for debug.

Macros

- `#define GSLC_DEBUG_PRINT(sFmt, ...)`
Macro to enable optional debug output.
- `#define GSLC_DEBUG2_PRINT(sFmt, ...)`
- `#define GSLC_DEBUG_PRINT_CONST(sFmt, ...)`
- `#define GSLC_DEBUG2_PRINT_CONST(sFmt, ...)`

7.12.1 Detailed Description

Macros that are used throughout the GUI for debug.

7.12.2 Macro Definition Documentation

7.12.2.1 `#define GSLC_DEBUG2_PRINT(sFmt, ...)`

7.12.2.2 `#define GSLC_DEBUG2_PRINT_CONST(sFmt, ...)`

7.12.2.3 `#define GSLC_DEBUG_PRINT(sFmt, ...)`

Macro to enable optional debug output.

- Supports printf formatting via `gslc_DebugPrintf()`
- Supports storing the format string in PROGMEM
- Note that at least one variable argument must be provided to the macro after the format string. This is a limitation of the macro definition. If no parameters are needed, then simply pass 0. For example: `GSLC_DEBUG_PRINT("Loaded OK",0);`

Parameters

<code>in</code>	<code>sFmt</code>	Format string for debug message
-----------------	-------------------	---------------------------------

7.12.2.4 `#define GSLC_DEBUG_PRINT_CONST(sFmt, ...)`

7.13 Flash-based Element Macros

Macros that represent element creation routines based in FLASH memory.

Macros

- #define `gslc_ElemCreateTxt_P`(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, pFont, colTxt, colFrame, colFill, nAlignTxt, bFrameEn, bFillEn)
Create a read-only text element.
- #define `gslc_ElemCreateTxt_P_R`(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, strLength, pFont, colTxt, colFrame, colFill, nAlignTxt, bFrameEn, bFillEn)
Create a read-write text element (element in Flash, string in RAM)
- #define `gslc_ElemCreateTxt_P_R_ext`(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, strLength, pFont, colTxt, colTxtGlow, colFrame, colFill, nAlignTxt, nMarginX, nMarginY, bFrameEn, bFillEn, bClickEn, bGlowEn, pfuncXEvent, pfuncXDraw, pfuncXTouch, pfuncXTick)
Create a read-write text element (element in Flash, string in RAM) with extended customization options.
- #define `gslc_ElemCreateBox_P`(pGui, nElemId, nPage, nX, nY, nW, nH, colFrame, colFill, bFrameEn, bFillEn, pfuncXDraw, pfuncXTick)
Create a read-only box element.
- #define `gslc_ElemCreateLine_P`(pGui, nElemId, nPage, nX0, nY0, nX1, nY1, colFill)
Create a read-only line element.
- #define `gslc_ElemCreateBtnTxt_P`(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, pFont, colTxt, colFrame, colFill, colFrameGlow, colFillGlow, nAlignTxt, bFrameEn, bFillEn, callFunc, extraData)
Create a text button element.

7.13.1 Detailed Description

Macros that represent element creation routines based in FLASH memory.

7.13.2 Macro Definition Documentation

7.13.2.1 #define `gslc_ElemCreateBox_P`(pGui, nElemId, nPage, nX, nY, nW, nH, colFrame, colFill, bFrameEn, bFillEn, pfuncXDraw, pfuncXTick)

Create a read-only box element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>colFrame</i>	Color for the frame
in	<i>colFill</i>	Color for the fill
in	<i>bFrameEn</i>	True if framed, false otherwise
in	<i>bFillEn</i>	True if filled, false otherwise
in	<i>pfuncXDraw</i>	Pointer to custom draw callback (or NULL if default)
in	<i>pfuncXTick</i>	Pointer to custom tick callback (or NULL if default)

7.13.2.2 `#define gslc_ElemCreateBtnTxt_P(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, pFont, colTxt, colFrame, colFill, colFrameGlow, colFillGlow, nAlignTxt, bFrameEn, bFillEn, callFunc, extraData)`

Create a text button element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>strTxt</i>	Text string to display
in	<i>pFont</i>	Pointer to font resource
in	<i>colTxt</i>	Color for the text
in	<i>colFrame</i>	Color for the frame
in	<i>colFill</i>	Color for the fill
in	<i>colFrameGlow</i>	Color for the frame when glowing
in	<i>colFillGlow</i>	Color for the fill when glowing
in	<i>nAlignTxt</i>	Text alignment
in	<i>bFrameEn</i>	True if framed, false otherwise
in	<i>bFillEn</i>	True if filled, false otherwise
in	<i>callFunc</i>	Callback function for button press
in	<i>extraData</i>	Ptr to extended data structure

7.13.2.3 `#define gslc_ElemCreateLine_P(pGui, nElemId, nPage, nX0, nY0, nX1, nY1, colFill)`

Create a read-only line element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX0</i>	X coordinate of line start
in	<i>nY0</i>	Y coordinate of line start
in	<i>nX1</i>	X coordinate of line end
in	<i>nY1</i>	Y coordinate of line end
in	<i>colFill</i>	Color for the line

7.13.2.4 `#define gslc_ElemCreateTxt_P(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, pFont, colTxt, colFrame, colFill, nAlignTxt, bFrameEn, bFillEn)`

Create a read-only text element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>strTxt</i>	Text string to display
in	<i>pFont</i>	Pointer to font resource
in	<i>colTxt</i>	Color for the text
in	<i>colFrame</i>	Color for the frame
in	<i>colFill</i>	Color for the fill
in	<i>nAlignTxt</i>	Text alignment
in	<i>bFrameEn</i>	True if framed, false otherwise
in	<i>bFillEn</i>	True if filled, false otherwise

7.13.2.5 `#define gslc_ElemCreateTxt_P_R(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, strLength, pFont, colTxt, colFrame, colFill, nAlignTxt, bFrameEn, bFillEn)`

Create a read-write text element (element in Flash, string in RAM)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>strTxt</i>	Text string to display
in	<i>strLength</i>	Length of text string
in	<i>pFont</i>	Pointer to font resource
in	<i>colTxt</i>	Color for the text
in	<i>colFrame</i>	Color for the frame
in	<i>colFill</i>	Color for the fill
in	<i>nAlignTxt</i>	Text alignment
in	<i>bFrameEn</i>	True if framed, false otherwise
in	<i>bFillEn</i>	True if filled, false otherwise

7.13.2.6 `#define gslc_ElemCreateTxt_P_R_ext(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, strLength, pFont, colTxt, colTxtGlow, colFrame, colFill, nAlignTxt, nMarginX, nMarginY, bFrameEn, bFillEn, bClickEn, bGlowEn, pfuncXEvent, pfuncXDraw, pfuncXTouch, pfuncXTick)`

Create a read-write text element (element in Flash, string in RAM) with extended customization options.

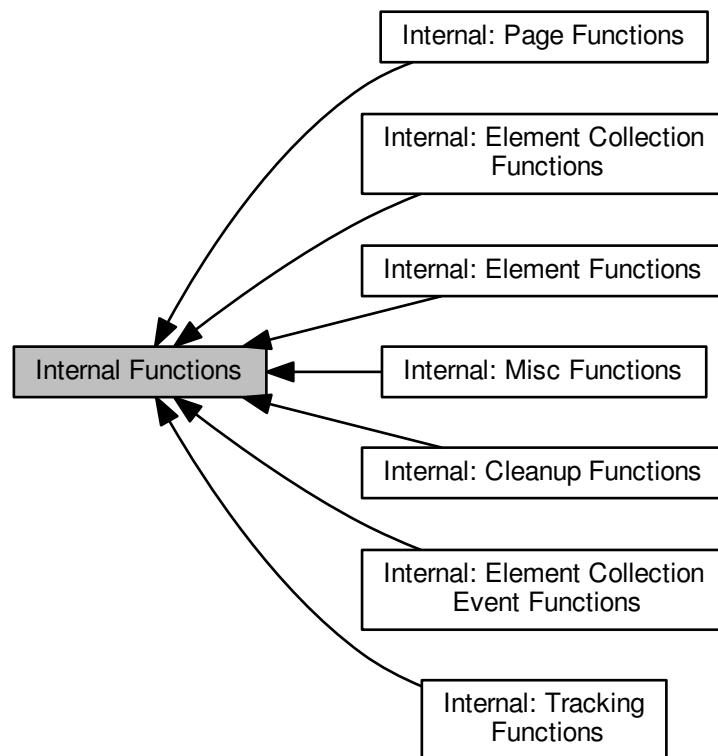
Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>strTxt</i>	Text string to display
in	<i>strLength</i>	Length of text string
in	<i>pFont</i>	Pointer to font resource
in	<i>colTxt</i>	Color for the text
in	<i>colTxtGlow</i>	Color for the text when glowing
in	<i>colFrame</i>	Color for the frame
in	<i>colFill</i>	Color for the fill
in	<i>nAlignTxt</i>	Text alignment
in	<i>nMarginX</i>	Text margin (X offset)
in	<i>nMarginY</i>	Text margin (Y offset)
in	<i>bFrameEn</i>	True if framed, false otherwise
in	<i>bFillEn</i>	True if filled, false otherwise
in	<i>bClickEn</i>	True if accept click events, false otherwise
in	<i>bGlowEn</i>	True if supports glow state, false otherwise
in	<i>pfuncXEvent</i>	Callback function ptr for Event
in	<i>pfuncXDraw</i>	Callback function ptr for Redraw
in	<i>pfuncXTouch</i>	Callback function ptr for Touch
in	<i>pfuncXTick</i>	Callback function ptr for Timer tick

7.14 Internal Functions

These functions are internal to the GUIslice implementation and are not intended to be called by user code and subject to change even in minor releases.

Collaboration diagram for Internal Functions:



Modules

- [Internal: Misc Functions](#)
- [Internal: Element Functions](#)
- [Internal: Page Functions](#)
- [Internal: Element Collection Functions](#)
- [Internal: Element Collection Event Functions](#)
- [Internal: Tracking Functions](#)
- [Internal: Cleanup Functions](#)

Variables

- [int16_t gslc_tsRect::x](#)
X coordinate of corner.
- [int16_t gslc_tsRect::y](#)

- Y coordinate of corner.*
- `uint16_t gslc_tsRect::w`
 - Width of region.*
- `uint16_t gslc_tsRect::h`
 - Height of region.*
- `int16_t gslc_tsPt::x`
 - X coordinate.*
- `int16_t gslc_tsPt::y`
 - Y coordinate.*
- `uint8_t gslc_tsColor::r`
 - RGB red value.*
- `uint8_t gslc_tsColor::g`
 - RGB green value.*
- `uint8_t gslc_tsColor::b`
 - RGB blue value.*
- `gslc_teEventType gslc_tsEvent::eType`
 - Event type.*
- `uint8_t gslc_tsEvent::nSubType`
 - Event sub-type.*
- `void * gslc_tsEvent::pvScope`
 - Event target scope (eg. Page,Collection,Event)*
- `void * gslc_tsEvent::pvData`
 - Generic data pointer for event.*
- `gslc_teTouch gslc_tsEventTouch::eTouch`
 - Touch state.*
- `int16_t gslc_tsEventTouch::nX`
 - Touch X coordinate (or param1)*
- `int16_t gslc_tsEventTouch::nY`
 - Touch Y coordinate (or param2)*
- `int16_t gslc_tsFont::nId`
 - Font ID specified by user.*
- `gslc_teFontRefType gslc_tsFont::eFontRefType`
 - Font reference type.*
- `gslc_teFontRefMode gslc_tsFont::eFontRefMode`
 - Font reference mode.*
- `const void * gslc_tsFont::pvFont`
 - Void ptr to the font reference (type defined by driver)*
- `uint16_t gslc_tsFont::nSize`
 - Font size.*
- `const unsigned char * gslc_tsImgRef::plmgBuf`
 - Pointer to input image buffer in memory [RAM,FLASH].*
- `const char * gslc_tsImgRef::pFname`
 - Pathname to input image file [FILE,SD].*
- `gslc_telmgRefFlags gslc_tsImgRef::eImgFlags`
 - Image reference flags.*
- `void * gslc_tsImgRef::pvImgRaw`
 - Ptr to raw output image data (for pre-loaded images)*
- `gslc_tsElem * gslc_tsElemRef::pElem`
 - Pointer to element in memory [RAM,FLASH].*
- `gslc_teElemRefFlags gslc_tsElemRef::eElemFlags`
 - Element reference flags.*

- [int16_t gslc_tsElem::nId](#)
Element ID specified by user.
- [uint8_t gslc_tsElem::nFeatures](#)
Element feature vector (appearance/behavior))
- [int16_t gslc_tsElem::nType](#)
Element type enumeration.
- [gslc_tsRect gslc_tsElem::rElem](#)
Rect region containing element.
- [int16_t gslc_tsElem::nGroup](#)
Group ID that the element belongs to.
- [gslc_tsColor gslc_tsElem::colElemFrame](#)
Color for frame.
- [gslc_tsColor gslc_tsElem::colElemFill](#)
Color for background fill.
- [gslc_tsColor gslc_tsElem::colElemFrameGlow](#)
Color to use for frame when glowing.
- [gslc_tsColor gslc_tsElem::colElemFillGlow](#)
Color to use for fill when glowing.
- [gslc_tsImgRef gslc_tsElem::sImgRefNorm](#)
Image reference to draw (normal)
- [gslc_tsImgRef gslc_tsElem::sImgRefGlow](#)
Image reference to draw (glowing)
- [gslc_tsElemRef * gslc_tsElem::pElemRefParent](#)
Parent element reference.
- [char * gslc_tsElem::pStrBuf](#)
Ptr to text string buffer to overlay.
- [uint8_t gslc_tsElem::nStrBufMax](#)
Size of string buffer.
- [gslc_teTxtFlags gslc_tsElem::eTxtFlags](#)
Flags associated with text buffer.
- [gslc_tsColor gslc_tsElem::colElemText](#)
Color of overlay text.
- [gslc_tsColor gslc_tsElem::colElemTextGlow](#)
Color of overlay text when glowing.
- [int8_t gslc_tsElem::eTxtAlign](#)
Alignment of overlay text.
- [int8_t gslc_tsElem::nTxtMarginX](#)
Margin of overlay text within rect region (x offset)
- [int8_t gslc_tsElem::nTxtMarginY](#)
Margin of overlay text within rect region (y offset)
- [gslc_tsFont * gslc_tsElem::pTxtFont](#)
Ptr to Font for overlay text.
- [void * gslc_tsElem::pXData](#)
Ptr to extended data structure.
- [GSLC_CB_EVENT gslc_tsElem::pfuncXEvent](#)
UNUSED: Callback func ptr for event tree (draw,touch,tick)
- [GSLC_CB_DRAW gslc_tsElem::pfuncXDraw](#)
Callback func ptr for custom drawing.
- [GSLC_CB_TOUCH gslc_tsElem::pfuncXTouch](#)
Callback func ptr for touch.
- [GSLC_CB_TICK gslc_tsElem::pfuncXTick](#)

- Callback func ptr for timer/main loop tick.*
- `gslc_tsElem * gslc_tsCollect::asElem`
Array of elements.
- `uint16_t gslc_tsCollect::nElemMax`
Maximum number of elements to allocate (in RAM)
- `uint16_t gslc_tsCollect::nElemCnt`
Number of elements allocated.
- `int16_t gslc_tsCollect::nElemAutoldNext`
Next Element ID for auto-assignment.
- `gslc_tsElemRef * gslc_tsCollect::asElemRef`
Array of element references.
- `uint16_t gslc_tsCollect::nElemRefMax`
Maximum number of element references to allocate.
- `uint16_t gslc_tsCollect::nElemRefCnt`
Number of element references allocated.
- `gslc_tsElemRef * gslc_tsCollect::pElemRefTracked`
Element reference currently being touch-tracked (NULL for none)
- `int16_t gslc_tsCollect::nElemIndFocused`
Element index currently in focus (eg. by keyboard/pin control), GSLC_IND_NONE for none.
- `gslc_tsCollect gslc_tsPage::sCollect`
Collection of elements on page.
- `int16_t gslc_tsPage::nPageId`
Page identifier.
- `gslc_tsRect gslc_tsPage::rBounds`
Bounding rect for page elements.
- `gslc_tsInputRawEvent gslc_tsInputMap::eEvent`
The input event.
- `int16_t gslc_tsInputMap::nVal`
The value associated with the input event.
- `gslc_tsAction gslc_tsInputMap::eAction`
Resulting action.
- `int16_t gslc_tsInputMap::nActionVal`
The value for the output action.
- `uint16_t gslc_tsGui::nDispW`
Width of the display (pixels)
- `uint16_t gslc_tsGui::nDispH`
Height of the display (pixels)
- `uint16_t gslc_tsGui::nDisp0W`
Width of the display (pixels) in native orientation.
- `uint16_t gslc_tsGui::nDisp0H`
Height of the display (pixels) in native orientation.
- `uint8_t gslc_tsGui::nDispDepth`
Bit depth of display (bits per pixel)
- `uint8_t gslc_tsGui::nRotation`
Adafruit GFX Rotation of display.
- `uint8_t gslc_tsGui::nTouchRotation`
Touchscreen rotation offset vs display.
- `uint8_t gslc_tsGui::nSwapXY`
Adafruit GFX Touch Swap x and y axes.
- `uint8_t gslc_tsGui::nFlipX`
Adafruit GFX Touch Flip x axis.

- [uint8_t gslc_tsGui::nFlipY](#)
Adafruit GFX Touch Flip x axis.
- [uint16_t gslc_tsGui::nTouchCalXMin](#)
Calibration X minimum reading.
- [uint16_t gslc_tsGui::nTouchCalXMax](#)
Calibration X maximum reading.
- [uint16_t gslc_tsGui::nTouchCalYMin](#)
Calibration Y minimum reading.
- [uint16_t gslc_tsGui::nTouchCalYMax](#)
Calibration Y maximum reading.
- [gslc_tsFont * gslc_tsGui::asFont](#)
Collection of loaded fonts.
- [uint8_t gslc_tsGui::nFontMax](#)
Maximum number of fonts to allocate.
- [uint8_t gslc_tsGui::nFontCnt](#)
Number of fonts allocated.
- [uint8_t gslc_tsGui::nRoundRadius](#)
Radius for rounded elements.
- [gslc_tsColor gslc_tsGui::sTransCol](#)
Color used for transparent image regions (GSLC_BMP_TRANS_EN=1)
- [gslc_tsElem gslc_tsGui::sElemTmp](#)
Temporary element.
- [gslc_tsElemRef gslc_tsGui::sElemRefTmp](#)
Temporary element reference.
- [gslc_tsElem gslc_tsGui::sElemTmpProg](#)
Temporary element for Flash compatibility.
- [gslc_tInitStat gslc_tsGui::eInitStatTouch](#)
Status of touch initialization.
- [int16_t gslc_tsGui::nTouchLastX](#)
Last touch event X coord.
- [int16_t gslc_tsGui::nTouchLastY](#)
Last touch event Y coord.
- [uint16_t gslc_tsGui::nTouchLastPress](#)
Last touch event pressure (0=none)
- [bool gslc_tsGui::bTouchRemapEn](#)
Enable touch remapping?
- [bool gslc_tsGui::bTouchRemapYX](#)
Enable touch controller swapping of X & Y.
- [void * gslc_tsGui::pvDriver](#)
Driver-specific members (gslc_tsDriver)*
- [bool gslc_tsGui::bRedrawPartialEn](#)
Driver supports partial page redraw.
- [gslc_tsImgRef gslc_tsGui::sImgRefBgnd](#)
Image reference for background.
- [uint8_t gslc_tsGui::nFrameRateCnt](#)
Diagnostic frame rate count.
- [uint8_t gslc_tsGui::nFrameRateStart](#)
Diagnostic frame rate timestamp.
- [gslc_tsPage * gslc_tsGui::asPage](#)
Array of all pages defined in system.
- [uint8_t gslc_tsGui::nPageMax](#)

- Maximum number of pages that can be defined.*

 - `uint8_t gslc_tsGui::nPageCnt`

Current number of pages defined.
- `gslc_tsPage * gslc_tsGui::apPageStack [GSLC_STACK__MAX]`

Stack of pages.
- `bool gslc_tsGui::abPageStackActive [GSLC_STACK__MAX]`

Whether page in stack can receive touch events.
- `bool gslc_tsGui::abPageStackDoDraw [GSLC_STACK__MAX]`

Whether page in stack is still actively drawn.
- `bool gslc_tsGui::bScreenNeedRedraw`

Screen requires a redraw.
- `bool gslc_tsGui::bScreenNeedFlip`

Screen requires a page flip.
- `bool gslc_tsGui::bInvalidateEn`

A region of the display has been invalidated.
- `gslc_tsRect gslc_tsGui::rInvalidateRect`

The rect region that has been invalidated.
- `GSLC_CB_PIN_POLL gslc_tsGui::pfuncPinPoll`

Callback func ptr for pin polling.
- `gslc_tsInputMap * gslc_tsGui::asInputMap`

Array of input maps.
- `uint8_t gslc_tsGui::nInputMapMax`

Maximum number of input maps.
- `uint8_t gslc_tsGui::nInputMapCnt`

Current number of input maps.

7.14.1 Detailed Description

These functions are internal to the GUIslice implementation and are not intended to be called by user code and subject to change even in minor releases.

- The following functions are generally not required for typical users of GUIslice. However, for advanced usage more direct access may be required.

7.14.2 Variable Documentation

7.14.2.1 `bool gslc_tsGui::abPageStackActive[GSLC_STACK__MAX]`

Whether page in stack can receive touch events.

7.14.2.2 `bool gslc_tsGui::abPageStackDoDraw[GSLC_STACK__MAX]`

Whether page in stack is still actively drawn.

7.14.2.3 `gslc_tsPage* gslc_tsGui::apPageStack[GSLC_STACK__MAX]`

Stack of pages.

7.14.2.4 `gslc_tsElem*` `gslc_tsCollect::asElem`

Array of elements.

7.14.2.5 `gslc_tsElemRef*` `gslc_tsCollect::asElemRef`

Array of element references.

7.14.2.6 `gslc_tsFont*` `gslc_tsGui::asFont`

Collection of loaded fonts.

7.14.2.7 `gslc_tsInputMap*` `gslc_tsGui::asInputMap`

Array of input maps.

7.14.2.8 `gslc_tsPage*` `gslc_tsGui::asPage`

Array of all pages defined in system.

7.14.2.9 `uint8_t` `gslc_tsColor::b`

RGB blue value.

7.14.2.10 `bool` `gslc_tsGui::bInvalidateEn`

A region of the display has been invalidated.

7.14.2.11 `bool` `gslc_tsGui::bRedrawPartialEn`

Driver supports partial page redraw.

If true, only changed elements are redrawn during next page redraw command. If false, entire page is redrawn when any element has been updated prior to next page redraw command.

7.14.2.12 `bool` `gslc_tsGui::bScreenNeedFlip`

Screen requires a page flip.

7.14.2.13 `bool` `gslc_tsGui::bScreenNeedRedraw`

Screen requires a redraw.

7.14.2.14 `bool gslc_tsGui::bTouchRemapEn`

Enable touch remapping?

7.14.2.15 `bool gslc_tsGui::bTouchRemapYX`

Enable touch controller swapping of X & Y.

7.14.2.16 `gslc_tsColor gslc_tsElem::colElemFill`

Color for background fill.

7.14.2.17 `gslc_tsColor gslc_tsElem::colElemFillGlow`

Color to use for fill when glowing.

7.14.2.18 `gslc_tsColor gslc_tsElem::colElemFrame`

Color for frame.

7.14.2.19 `gslc_tsColor gslc_tsElem::colElemFrameGlow`

Color to use for frame when glowing.

7.14.2.20 `gslc_tsColor gslc_tsElem::colElemText`

Color of overlay text.

7.14.2.21 `gslc_tsColor gslc_tsElem::colElemTextGlow`

Color of overlay text when glowing.

7.14.2.22 `gslc_teAction gslc_tsInputMap::eAction`

Resulting action.

7.14.2.23 `gslc_teElemRefFlags gslc_tsElemRef::eElemFlags`

Element reference flags.

7.14.2.24 gslc_telInputRawEvent gslc_tsInputMap::eEvent

The input event.

7.14.2.25 gslc_teFontRefMode gslc_tsFont::eFontRefMode

Font reference mode.

7.14.2.26 gslc_teFontRefType gslc_tsFont::eFontRefType

Font reference type.

7.14.2.27 gslc_telmgRefFlags gslc_tsImgRef::elmgFlags

Image reference flags.

7.14.2.28 gslc_telnitStat gslc_tsGui::elnitStatTouch

Status of touch initialization.

7.14.2.29 gslc_teTouch gslc_tsEventTouch::eTouch

Touch state.

7.14.2.30 int8_t gslc_tsElem::eTxtAlign

Alignment of overlay text.

7.14.2.31 gslc_teTxtFlags gslc_tsElem::eTxtFlags

Flags associated with text buffer.

7.14.2.32 gslc_teEventType gslc_tsEvent::eType

Event type.

7.14.2.33 uint8_t gslc_tsColor::g

RGB green value.

7.14.2.34 uint16_t gslc_tsRect::h

Height of region.

7.14.2.35 int16_t gslc_tsInputMap::nActionVal

The value for the output action.

7.14.2.36 uint16_t gslc_tsGui::nDisp0H

Height of the display (pixels) in native orientation.

7.14.2.37 uint16_t gslc_tsGui::nDisp0W

Width of the display (pixels) in native orientation.

7.14.2.38 uint8_t gslc_tsGui::nDispDepth

Bit depth of display (bits per pixel)

7.14.2.39 uint16_t gslc_tsGui::nDispH

Height of the display (pixels)

7.14.2.40 uint16_t gslc_tsGui::nDispW

Width of the display (pixels)

7.14.2.41 int16_t gslc_tsCollect::nElemAutoldNext

Next Element ID for auto-assignment.

7.14.2.42 uint16_t gslc_tsCollect::nElemCnt

Number of elements allocated.

7.14.2.43 int16_t gslc_tsCollect::nElemIndFocused

Element index currently in focus (eg. by keyboard/pin control), GSLC_IND_NONE for none.

7.14.2.44 `uint16_t gslc_tsCollect::nElemMax`

Maximum number of elements to allocate (in RAM)

7.14.2.45 `uint16_t gslc_tsCollect::nElemRefCnt`

Number of element references allocated.

7.14.2.46 `uint16_t gslc_tsCollect::nElemRefMax`

Maximum number of element references to allocate.

7.14.2.47 `uint8_t gslc_tsElem::nFeatures`

Element feature vector (appearance/behavior))

7.14.2.48 `uint8_t gslc_tsGui::nFlipX`

Adafruit GFX Touch Flip x axis.

7.14.2.49 `uint8_t gslc_tsGui::nFlipY`

Adafruit GFX Touch Flip y axis.

7.14.2.50 `uint8_t gslc_tsGui::nFontCnt`

Number of fonts allocated.

7.14.2.51 `uint8_t gslc_tsGui::nFontMax`

Maximum number of fonts to allocate.

7.14.2.52 `uint8_t gslc_tsGui::nFrameRateCnt`

Diagnostic frame rate count.

7.14.2.53 `uint8_t gslc_tsGui::nFrameRateStart`

Diagnostic frame rate timestamp.

7.14.2.54 int16_t gslc_tsElem::nGroup

Group ID that the element belongs to.

7.14.2.55 int16_t gslc_tsFont::nId

Font ID specified by user.

7.14.2.56 int16_t gslc_tsElem::nId

Element ID specified by user.

7.14.2.57 uint8_t gslc_tsGui::nInputMapCnt

Current number of input maps.

7.14.2.58 uint8_t gslc_tsGui::nInputMapMax

Maximum number of input maps.

7.14.2.59 uint8_t gslc_tsGui::nPageCnt

Current number of pages defined.

7.14.2.60 int16_t gslc_tsPage::nPageId

Page identifier.

7.14.2.61 uint8_t gslc_tsGui::nPageMax

Maximum number of pages that can be defined.

7.14.2.62 uint8_t gslc_tsGui::nRotation

Adafruit GFX Rotation of display.

7.14.2.63 uint8_t gslc_tsGui::nRoundRadius

Radius for rounded elements.

7.14.2.64 `uint16_t gslc_tsFont::nSize`

Font size.

7.14.2.65 `uint8_t gslc_tsElem::nStrBufMax`

Size of string buffer.

7.14.2.66 `uint8_t gslc_tsEvent::nSubType`

Event sub-type.

7.14.2.67 `uint8_t gslc_tsGui::nSwapXY`

Adafruit GFX Touch Swap x and y axes.

7.14.2.68 `uint16_t gslc_tsGui::nTouchCalXMax`

Calibration X maximum reading.

7.14.2.69 `uint16_t gslc_tsGui::nTouchCalXMin`

Calibration X minimum reading.

7.14.2.70 `uint16_t gslc_tsGui::nTouchCalYMax`

Calibration Y maximum reading.

7.14.2.71 `uint16_t gslc_tsGui::nTouchCalYMin`

Calibration Y minimum reading.

7.14.2.72 `uint16_t gslc_tsGui::nTouchLastPress`

Last touch event pressure (0=none))

7.14.2.73 `int16_t gslc_tsGui::nTouchLastX`

Last touch event X coord.

7.14.2.74 int16_t gslc_tsGui::nTouchLastY

Last touch event Y coord.

7.14.2.75 uint8_t gslc_tsGui::nTouchRotation

Touchscreen rotation offset vs display.

7.14.2.76 int8_t gslc_tsElem::nTxtMarginX

Margin of overlay text within rect region (x offset)

7.14.2.77 int8_t gslc_tsElem::nTxtMarginY

Margin of overlay text within rect region (y offset)

7.14.2.78 int16_t gslc_tsElem::nType

Element type enumeration.

7.14.2.79 int16_t gslc_tsInputMap::nVal

The value associated with the input event.

7.14.2.80 int16_t gslc_tsEventTouch::nX

Touch X coordinate (or param1)

7.14.2.81 int16_t gslc_tsEventTouch::nY

Touch Y coordinate (or param2)

7.14.2.82 gslc_tsElem* gslc_tsElemRef::pElem

Pointer to element in memory [RAM,FLASH].

7.14.2.83 gslc_tsElemRef* gslc_tsElem::pElemRefParent

Parent element reference.

Used during redraw to notify parent elements that they require redraw as well. Primary usage is in compound elements. NOTE: Although this field is only used in GLSC_COMPOUND mode, it is not wrapped in an ifdef because the ElemCreate*_P() function macros currently initialize this field.

7.14.2.84 gslc_tsElemRef* gslc_tsCollect::pElemRefTracked

Element reference currently being touch-tracked (NULL for none)

7.14.2.85 const char* gslc_tsImgRef::pFname

Pathname to input image file [FILE,SD].

7.14.2.86 GSLC_CB_PIN_POLL gslc_tsGui::pfuncPinPoll

Callback func ptr for pin polling.

7.14.2.87 GSLC_CB_DRAW gslc_tsElem::pfuncXDraw

Callback func ptr for custom drawing.

7.14.2.88 GSLC_CB_EVENT gslc_tsElem::pfuncXEvent

UNUSED: Callback func ptr for event tree (draw,touch,tick)

7.14.2.89 GSLC_CB_TICK gslc_tsElem::pfuncXTick

Callback func ptr for timer/main loop tick.

7.14.2.90 GSLC_CB_TOUCH gslc_tsElem::pfuncXTouch

Callback func ptr for touch.

7.14.2.91 const unsigned char* gslc_tsImgRef::pImgBuf

Pointer to input image buffer in memory [RAM,FLASH].

7.14.2.92 char* gslc_tsElem::pStrBuf

Ptr to text string buffer to overlay.

7.14.2.93 gslc_tsFont* gslc_tsElem::pTxtFont

Ptr to Font for overlay text.

7.14.2.94 void* gslc_tsEvent::pvData

Generic data pointer for event.

This member is used to either pass a pointer to a simple data datatype (such as Element or Collection) or to a another structure that contains multiple fields.

7.14.2.95 void* gslc_tsGui::pvDriver

Driver-specific members (gslc_tsDriver*)

7.14.2.96 const void* gslc_tsFont::pvFont

Void ptr to the font reference (type defined by driver)

7.14.2.97 void* gslc_tsImgRef::pvImgRaw

Ptr to raw output image data (for pre-loaded images)

7.14.2.98 void* gslc_tsEvent::pvScope

Event target scope (eg. Page,Collection,Event)

7.14.2.99 void* gslc_tsElem::pXData

Ptr to extended data structure.

7.14.2.100 uint8_t gslc_tsColor::r

RGB red value.

7.14.2.101 gslc_tsRect gslc_tsPage::rBounds

Bounding rect for page elements.

7.14.2.102 gslc_tsRect gslc_tsElem::rElem

Rect region containing element.

7.14.2.103 gslc_tsRect gslc_tsGui::rInvalidateRect

The rect region that has been invalidated.

7.14.2.104 gslc_tsCollect gslc_tsPage::sCollect

Collection of elements on page.

7.14.2.105 gslc_tsElemRef gslc_tsGui::sElemRefTmp

Temporary element reference.

7.14.2.106 gslc_tsElem gslc_tsGui::sElemTmp

Temporary element.

7.14.2.107 gslc_tsElem gslc_tsGui::sElemTmpProg

Temporary element for Flash compatibility.

7.14.2.108 gslc_tsImgRef gslc_tsGui::sImgRefBkgnd

Image reference for background.

7.14.2.109 gslc_tsImgRef gslc_tsElem::sImgRefGlow

Image reference to draw (glowing)

7.14.2.110 gslc_tsImgRef gslc_tsElem::sImgRefNorm

Image reference to draw (normal)

7.14.2.111 gslc_tsColor gslc_tsGui::sTransCol

Color used for transparent image regions (GSLC_BMP_TRANS_EN=1)

7.14.2.112 uint16_t gslc_tsRect::w

Width of region.

7.14.2.113 int16_t gslc_tsRect::x

X coordinate of corner.

7.14.2.114 int16_t gslc_tsPt::x

X coordinate.

7.14.2.115 int16_t gslc_tsRect::y

Y coordinate of corner.

7.14.2.116 int16_t gslc_tsPt::y

Y coordinate.

7.15 Internal: Misc Functions

Collaboration diagram for Internal: Misc Functions:



Functions

- [gslc_tsImgRef gslc_ResetImage \(\)](#)
Create a blank image reference structure.

7.15.1 Detailed Description

7.15.2 Function Documentation

7.15.2.1 [gslc_tsImgRef gslc_ResetImage \(\)](#)

Create a blank image reference structure.

Returns

Image reference struct

7.16 Internal: Element Functions

Collaboration diagram for Internal: Element Functions:



Functions

- [gslc_tsElem](#) [gslc_ElemCreate](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nElemId, [int16_t](#) nPageId, [int16_t](#) nType, [gslc_tsRect](#) rElem, [char](#) *pStrBuf, [uint8_t](#) nStrBufMax, [int16_t](#) nFontId)

Create a new element with default styling.
- [gslc_tsElemRef](#) * [gslc_ElemAdd](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nPageId, [gslc_tsElem](#) *pElem, [gslc_teElemRefFlags](#) eFlags)

Add the Element to the list of generated elements in the GUI environment.
- [uint8_t](#) [gslc_GetElemRefFlag](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [uint8_t](#) nFlagMask)

Get the flags associated with an element reference.
- void [gslc_SetElemRefFlag](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [uint8_t](#) nFlagMask, [uint8_t](#) nFlagVal)

Set the flags associated with an element reference.
- [gslc_tsElem](#) * [gslc_GetElemFromRef](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Returns a pointer to an element from an element reference, copying from FLASH to RAM if element is stored in PROGMEM.
- [gslc_tsElem](#) * [gslc_GetElemFromRefD](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [int16_t](#) nLineNum)

Returns a pointer to an element from an element reference.
- void * [gslc_GetXDataFromRef](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [int16_t](#) nType, [int16_t](#) nLineNum)

Returns a pointer to the data structure associated with an extended element.
- void [gslc_ElemSetImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsImgRef](#) sImgRef, [gslc_tsImgRef](#) sImgRefSel)

Set an element to use a bitmap image.
- bool [gslc_ElemDrawByRef](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) eRedraw)

Draw an element to the active display.
- void [gslc_ElemDraw](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nPageId, [int16_t](#) nElemId)

Draw an element to the active display.
- void [gslc_DrawTxtBase](#) ([gslc_tsGui](#) *pGui, [char](#) *pStrBuf, [gslc_tsRect](#) rTxt, [gslc_tsFont](#) *pTxtFont, [gslc_teTxtFlags](#) eTxtFlags, [int8_t](#) eTxtAlign, [gslc_tsColor](#) colTxt, [gslc_tsColor](#) colBg, [int16_t](#) nMarginW, [int16_t](#) nMarginH)

Draw text with full text justification.
- void [gslc_SetRoundRadius](#) ([gslc_tsGui](#) *pGui, [uint8_t](#) nRadius)

Set the global rounded radius.

7.16.1 Detailed Description

7.16.2 Function Documentation

7.16.2.1 void `gslc_DrawTxtBase` (`gslc_tsGui` * *pGui*, `char` * *pStrBuf*, `gslc_tsRect` *rTxt*, `gslc_tsFont` * *pTxtFont*, `gslc_teTxtFlags` *eTxtFlags*, `int8_t` *eTxtAlign*, `gslc_tsColor` *colTxt*, `gslc_tsColor` *colBg*, `int16_t` *nMarginW*, `int16_t` *nMarginH*)

Draw text with full text justification.

- This function is usually only required by internal GUIslice rendering operations but is made available for custom element usage as well

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pStrBuf</i>	Pointer to text string buffer
in	<i>rTxt</i>	Rectangle region to contain the text
in	<i>pTxtFont</i>	Pointer to the font
in	<i>eTxtFlags</i>	Text string attributes
in	<i>eTxtAlign</i>	Text alignment / justification mode
in	<i>colTxt</i>	Text foreground color
in	<i>colBg</i>	Text background color
in	<i>nMarginW</i>	Horizontal margin within rect region to keep text away
in	<i>nMarginH</i>	Vertical margin within rect region to keep text away

Returns

none

7.16.2.2 `gslc_tsElemRef*` `gslc_ElemAdd` (`gslc_tsGui` * *pGui*, `int16_t` *nPageId*, `gslc_tsElem` * *pElem*, `gslc_teElemRefFlags` *eFlags*)

Add the Element to the list of generated elements in the GUI environment.

- NOTE: The content of *pElem* is copied so the pointer can be released after the call.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ PageId</i>	Page ID to add element to (GSLC_PAGE_NONE to skip in case of temporary creation for compound elements)
in	<i>pElem</i>	Pointer to Element to add
in	<i>eFlags</i>	Flags describing the element (eg. whether the element should be stored in internal RAM array or is located in Flash/PROGMEM).

Returns

Pointer to Element reference or NULL if fail

7.16.2.3 `gslc_tsElem gslc_ElemCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPageId, int16_t nType, gslc_tsRect rElem, char * pStrBuf, uint8_t nStrBufMax, int16_t nFontId)`

Create a new element with default styling.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	User-supplied ID for referencing this element (or GSLC_ID_AUTO to auto-generate)
in	<i>nPageId</i>	The page ID on which this page should be associated
in	<i>nType</i>	Enumeration that indicates the type of element that is requested for creation. The type adjusts the visual representation and default styling.
in	<i>rElem</i>	Rectangle region framing the element
in	<i>pStrBuf</i>	String to copy into element
in	<i>nStrBufMax</i>	Maximum length of string buffer (pStrBuf). Only applicable if GSLC_LOCAL_STR=0. Ignored if GSLC_LOCAL_STR=1.)
in	<i>nFontId</i>	Font ID for textual elements

Returns

Initialized structure

7.16.2.4 `void gslc_ElemDraw (gslc_tsGui * pGui, int16_t nPageId, int16_t nElemId)`

Draw an element to the active display.

- Element is referenced by a page ID and element ID
- Provides similar functionality as ElemDrawByRef() but accepts page and element IDs

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n_↔PageId</i>	ID of page containing element
in	<i>n_↔ElemId</i>	ID of element

Returns

none

7.16.2.5 `bool gslc_ElemDrawByRef (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_teRedrawType eRedraw)`

Draw an element to the active display.

- Element is referenced by an element pointer

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element reference to draw
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

7.16.2.6 `void gslc_ElemSetImage (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tsImgRef sImgRef, gslc_tsImgRef sImgRefSel)`

Set an element to use a bitmap image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference to update
in	<i>sImgRef</i>	Image reference (normal state)
in	<i>sImgRefSel</i>	Image reference (glowing state)

Returns

none

7.16.2.7 `gslc_tsElem* gslc_GetElemFromRef (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Returns a pointer to an element from an element reference, copying from FLASH to RAM if element is stored in PROGMEM.

This function enables all APIs to work with Elements irrespective of whether they were created in RAM or Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element Reference

Returns

Pointer to Element after ensuring that it is accessible from RAM

7.16.2.8 `gslc_tsElem* gslc_GetElemFromRefD (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nLineNum)`

Returns a pointer to an element from an element reference.

This is a wrapper for GetElemFromRef() including debug checking for invalid pointers.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element Reference
in	<i>nLineNum</i>	Line number from calling function (ie. LINE)

Returns

Pointer to Element after ensuring that it is accessible from RAM

7.16.2.9 `uint8_t gslc_GetElemRefFlag (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, uint8_t nFlagMask)`

Get the flags associated with an element reference.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Element reference pointer
in	<i>nFlagMask</i>	Flags to read

Returns

Values associated with the element reference flags (subject to the flag mask)

7.16.2.10 `void* gslc_GetXDataFromRef (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nType, int16_t nLineNum)`

Returns a pointer to the data structure associated with an extended element.

- Example usage: `gslc_tsXListbox* pListbox = (gslc_tsXListbox*)gslc_GetXDataFromRef(pGui, pElemRef, GSLC_TYPEX_LISTBOX, LINE);`

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element Reference
in	<i>nType</i>	Expected type indicator (ie. <code>GSLC_TYPEX_*</code>)
in	<i>nLineNum</i>	Line number from calling function (ie. LINE)

Returns

Void pointer to extended data (pXData), or NULL if error. Needs to be typecasted accordingly.

7.16.2.11 void gslc_SetElemRefFlag (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, uint8_t *nFlagMask*, uint8_t *nFlagVal*)

Set the flags associated with an element reference.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Element reference pointer
in	<i>nFlagMask</i>	Flags to read
in	<i>nFlagVal</i>	Values to assign to masked flags

Returns

none

7.16.2.12 void gslc_SetRoundRadius (gslc_tsGui * *pGui*, uint8_t *nRadius*)

Set the global rounded radius.

- Used for rounded rectangles

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRadius</i>	Radius for rounded elements

Returns

none

7.17 Internal: Page Functions

Collaboration diagram for Internal: Page Functions:



Functions

- bool [gslc_PageEvent](#) (void *pvGui, [gslc_tsEvent](#) sEvent)
Common event handler function for a page.
- void [gslc_PageRedrawGo](#) ([gslc_tsGui](#) *pGui)
Redraw all elements on the active page.
- void [gslc_PageFlipSet](#) ([gslc_tsGui](#) *pGui, bool bNeeded)
Indicate whether the screen requires page flip.
- bool [gslc_PageFlipGet](#) ([gslc_tsGui](#) *pGui)
Get state of pending page flip state.
- void [gslc_PageFlipGo](#) ([gslc_tsGui](#) *pGui)
Update the visible screen if page has been marked for flipping.
- [gslc_tsPage](#) * [gslc_PageFindById](#) ([gslc_tsGui](#) *pGui, int16_t nPageId)
Find a page in the GUI by its ID.
- void [gslc_PageRedrawCalc](#) ([gslc_tsGui](#) *pGui)
Perform a redraw calculation on the page to determine if additional elements should also be redrawn.
- int16_t [gslc_PageFocusStep](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage, bool bNext)
- [gslc_tsEvent](#) [gslc_EventCreate](#) ([gslc_tsGui](#) *pGui, [gslc_teEventType](#) eType, uint8_t nSubType, void *pv↔
Scope, void *pvData)
Create an event structure.
- bool [gslc_ElemEvent](#) (void *pvGui, [gslc_tsEvent](#) sEvent)
Common event handler function for an element.
- bool [gslc_ElemSendEventTouch](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRefTracked, [gslc_teTouch](#) e↔
Touch, int16_t nX, int16_t nY)
Trigger an element's touch event.

7.17.1 Detailed Description

7.17.2 Function Documentation

7.17.2.1 bool [gslc_ElemEvent](#) (void * pvGui, [gslc_tsEvent](#) sEvent)

Common event handler function for an element.

Parameters

in	<i>pVGui</i>	Void pointer to GUI
in	<i>sEvent</i>	Event data structure

Returns

true if success, false if fail

7.17.2.2 `bool gslc_ElemSendEventTouch (gslc_tsGui * pGui, gslc_tsElemRef * pElemRefTracked, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

Trigger an element's touch event.

This is an optional behavior useful in some extended element types.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRefTracked</i>	Pointer to tracked Element reference (or NULL for none))
in	<i>eTouch</i>	Touch event type
in	<i>nX</i>	X coordinate of event (absolute coordinate)
in	<i>nY</i>	Y coordinate of event (absolute coordinate)

Returns

true if success, false if error

7.17.2.3 `gslc_tsEvent gslc_EventCreate (gslc_tsGui * pGui, gslc_teEventType eType, uint8_t nSubType, void * pvScope, void * pvData)`

Create an event structure.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>eType</i>	Event type (draw, touch, tick, etc.)
in	<i>nSubType</i>	Refinement of event type (or 0 if unused)
in	<i>pvScope</i>	Void ptr to object receiving event so that the event handler will have the context
in	<i>pvData</i>	Void ptr to additional data associated with the event (eg. coordinates for touch events)

Returns

None

7.17.2.4 `bool gslc_PageEvent (void * pvGui, gslc_tsEvent sEvent)`

Common event handler function for a page.

Parameters

in	<i>pvGui</i>	Void pointer to GUI
in	<i>sEvent</i>	Event data structure

Returns

true if success, false if fail

7.17.2.5 `gslc_tsPage* gslc_PageFindById (gslc_tsGui * pGui, int16_t nPageld)`

Find a page in the GUI by its ID.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nPageld</i>	Page ID to search

Returns

Ptr to a page or NULL if none found

7.17.2.6 `bool gslc_PageFlipGet (gslc_tsGui * pGui)`

Get state of pending page flip state.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

True if screen requires page flip

7.17.2.7 `void gslc_PageFlipGo (gslc_tsGui * pGui)`

Update the visible screen if page has been marked for flipping.

- On some hardware this can trigger a double-buffering page flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

None

7.17.2.8 void `gslc_PageFlipSet (gslc_tsGui * pGui, bool bNeeded)`

Indicate whether the screen requires page flip.

- This is generally called with `bNeeded=true` whenever drawing has been done to the active page. Page flip is actually performed later when calling `PageFlipGo()`.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>bNeeded</i>	True if screen requires page flip

Returns

None

7.17.2.9 int16_t `gslc_PageFocusStep (gslc_tsGui * pGui, gslc_tsPage * pPage, bool bNext)`

Todo Doc. This API is experimental and subject to change

7.17.2.10 void `gslc_PageRedrawCalc (gslc_tsGui * pGui)`

Perform a redraw calculation on the page to determine if additional elements should also be redrawn.

This routine checks to see if any transparent elements have been marked as needing redraw. If so, the whole page may be marked as needing redraw (or at least the other elements that have been exposed underneath).

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

7.17.2.11 void gslc_PageRedrawGo (gslc_tsGui * *pGui*)

Redraw all elements on the active page.

Only the elements that have been marked as needing redraw are rendered unless the entire page has been marked as needing redraw (in which case everything is drawn)

Parameters

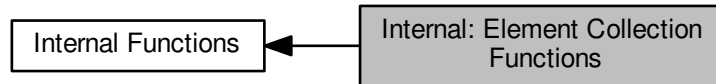
in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

7.18 Internal: Element Collection Functions

Collaboration diagram for Internal: Element Collection Functions:



Functions

- void `gslc_CollectReset` (`gslc_tsCollect` *pCollect, `gslc_tsElem` *asElem, `uint16_t` nElemMax, `gslc_tsElemRef` *asElemRef, `uint16_t` nElemRefMax)
Reset the members of an element collection.
- `gslc_tsElemRef` * `gslc_CollectElemAdd` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect, const `gslc_tsElem` *pElem, `gslc_teElemRefFlags` eFlags)
Add an element to a collection.
- bool `gslc_CollectGetRedraw` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect)
Determine if any elements in a collection need redraw.
- `gslc_tsElemRef` * `gslc_CollectFindElemById` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect, `int16_t` nElemId)
Find an element in a collection by its Element ID.
- `gslc_tsElemRef` * `gslc_CollectFindElemFromCoord` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect, `int16_t` nX, `int16_t` nY)
Find an element in a collection by a coordinate coordinate.
- int `gslc_CollectGetNextId` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect)
Allocate the next available Element ID in a collection.
- `gslc_tsElemRef` * `gslc_CollectGetElemRefTracked` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect)
Get the element within a collection that is currently being tracked.
- void `gslc_CollectSetElemTracked` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect, `gslc_tsElemRef` *pElemRef)
Set the element within a collection that is currently being tracked.
- `int16_t` `gslc_CollectGetFocus` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect)
Get the element index within a collection that is currently in focus.
- void `gslc_CollectSetFocus` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect, `int16_t` nElemInd)
Set the element index within a collection that is currently in focus.
- bool `gslc_CollectFindFocusStep` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect, bool bNext, bool *pbWrapped, `int16_t` *pnElemInd)
Find the next element in focus.
- void `gslc_CollectSetParent` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect, `gslc_tsElemRef` *pElemRefParent)
Assign the parent element reference to all elements within a collection.

7.18.1 Detailed Description

7.18.2 Function Documentation

7.18.2.1 `gslc_tsElemRef* gslc_CollectElemAdd (gslc_tsGui * pGui, gslc_tsCollect * pCollect, const gslc_tsElem * pElem, gslc_teElemRefFlags eFlags)`

Add an element to a collection.

- Note that the contents of *pElem* are copied to the collection's element array so the *pElem* pointer can be discarded after the call is complete.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection
in	<i>pElem</i>	Ptr to the element to add
in	<i>eFlags</i>	Flags describing the element (eg. whether the element should be stored in internal RAM array or is located in Flash/PROGMEM).

Returns

Pointer to the element reference in the collection that has been added or NULL if there was an error

7.18.2.2 `gslc_tsElemRef* gslc_CollectFindElemById (gslc_tsGui * pGui, gslc_tsCollect * pCollect, int16_t nElemId)`

Find an element in a collection by its Element ID.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection
in	<i>nElemId</i>	Element ID to search for

Returns

Pointer to the element reference in the collection that was found or NULL if no matches found

7.18.2.3 `gslc_tsElemRef* gslc_CollectFindElemFromCoord (gslc_tsGui * pGui, gslc_tsCollect * pCollect, int16_t nX, int16_t nY)`

Find an element in a collection by a coordinate.

- A match is found if the element is "clickable" (*bClickEn*=true) and the coordinate falls within the element's bounds (*rElem*).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection
in	<i>nX</i>	Absolute X coordinate to use for search
in	<i>nY</i>	Absolute Y coordinate to use for search

Returns

Pointer to the element reference in the collection that was found or NULL if no matches found

7.18.2.4 `bool gslc_CollectFindFocusStep (gslc_tsGui * pGui, gslc_tsCollect * pCollect, bool bNext, bool * pbWrapped, int16_t * pnElemInd)`

Todo Doc. This API is experimental and subject to change

7.18.2.5 `gslc_tsElemRef* gslc_CollectGetElemRefTracked (gslc_tsGui * pGui, gslc_tsCollect * pCollect)`

Get the element within a collection that is currently being tracked.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection

Returns

Pointer to the element reference in the collection that is currently being tracked or NULL if no elements are being tracked

7.18.2.6 `int16_t gslc_CollectGetFocus (gslc_tsGui * pGui, gslc_tsCollect * pCollect)`

Get the element index within a collection that is currently in focus.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection

Returns

Element index or GSLC_IND_NONE for none

7.18.2.7 `int gslc_CollectGetNextId (gslc_tsGui * pGui, gslc_tsCollect * pCollect)`

Allocate the next available Element ID in a collection.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection

Returns

Element ID that is reserved for use

7.18.2.8 `bool gslc_CollectGetRedraw (gslc_tsGui * pGui, gslc_tsCollect * pCollect)`

Determine if any elements in a collection need redraw.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to Element collection

Returns

True if redraw required, false otherwise

7.18.2.9 `void gslc_CollectReset (gslc_tsCollect * pCollect, gslc_tsElem * asElem, uint16_t nElemMax, gslc_tsElemRef * asElemRef, uint16_t nElemRefMax)`

Reset the members of an element collection.

Parameters

in	<i>pCollect</i>	Pointer to the collection
in	<i>asElem</i>	Internal element array storage to associate with the collection
in	<i>nElemMax</i>	Maximum number of elements that can be added to the internal element array (ie. RAM))
in	<i>asElemRef</i>	Internal element reference array storage to associate with the collection. All elements, whether they are located in the internal element array or in external Flash (PROGMEM) storage, require an entry in the element reference array.
in	<i>nElemRefMax</i>	Maximum number of elements in the reference array. This is effectively the maximum number of elements that can appear in the collection, irrespective of whether it is stored in RAM or Flash (PROGMEM).

Returns

none

7.18.2.10 void `gslc_CollectSetElemTracked (gslc_tsGui * pGui, gslc_tsCollect * pCollect, gslc_tsElemRef * pElemRef)`

Set the element within a collection that is currently being tracked.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection
in	<i>pElemRef</i>	Ptr to element reference to mark as being tracked

Returns

none

7.18.2.11 void `gslc_CollectSetFocus (gslc_tsGui * pGui, gslc_tsCollect * pCollect, int16_t nElemInd)`

Set the element index within a collection that is currently in focus.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection
in	<i>nElemInd</i>	Element index to set in focus, GSLC_IND_NONE for none

Returns

none

7.18.2.12 void `gslc_CollectSetParent (gslc_tsGui * pGui, gslc_tsCollect * pCollect, gslc_tsElemRef * pElemRefParent)`

Assign the parent element reference to all elements within a collection.

- This is generally used in the case of compound elements where updates to a sub-element should cause the parent (compound element) to be redrawn as well.)

Parameters

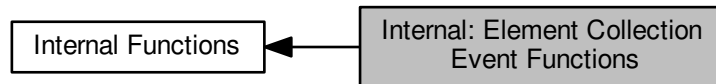
in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection
in	<i>pElemRefParent</i>	Ptr to element reference that is the parent

Returns

none

7.19 Internal: Element Collection Event Functions

Collaboration diagram for Internal: Element Collection Event Functions:



Functions

- `bool gslc_CollectEvent (void *pvGui, gslc_tsEvent sEvent)`
Common event handler function for an element collection.
- `void gslc_CollectTouch (gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsEventTouch *pEventTouch)`
Handle touch events within the element collection.
- `bool gslc_CollectTouchCompound (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY, gslc_tsCollect *pCollect)`
Handle dispatch of touch (up,down,move) events to compound elements sub elements.
- `void gslc_CollectInput (gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsEventTouch *pEventTouch)`
Handle direct input events within the element collection.

7.19.1 Detailed Description

7.19.2 Function Documentation

7.19.2.1 `bool gslc_CollectEvent (void * pvGui, gslc_tsEvent sEvent)`

Common event handler function for an element collection.

Parameters

in	<i>pvGui</i>	Void pointer to GUI
in	<i>sEvent</i>	Event data structure

Returns

true if success, false if fail

7.19.2.2 `void gslc_CollectInput (gslc_tsGui * pGui, gslc_tsCollect * pCollect, gslc_tsEventTouch * pEventTouch)`

Handle direct input events within the element collection.

Parameters

in	<i>pGui</i>	Pointer to the GUI
in	<i>pCollect</i>	Ptr to the element collection
in	<i>pEventTouch</i>	Ptr to the touch event structure

Returns

none

7.19.2.3 void gslc_CollectTouch (gslc_tsGui * *pGui*, gslc_tsCollect * *pCollect*, gslc_tsEventTouch * *pEventTouch*)

Handle touch events within the element collection.

Parameters

in	<i>pGui</i>	Pointer to the GUI
in	<i>pCollect</i>	Ptr to the element collection
in	<i>pEventTouch</i>	Ptr to the touch event structure

Returns

none

7.19.2.4 bool gslc_CollectTouchCompound (void * *pvGui*, void * *pvElemRef*, gslc_teTouch *eTouch*, int16_t *nRelX*, int16_t *nRelY*, gslc_tsCollect * *pCollect*)

Handle dispatch of touch (up,down,move) events to compound elements sub elements.

Parameters

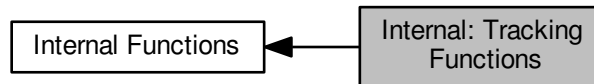
in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element Reference(typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element
in	<i>pCollect</i>	Collection containing sub elements

Returns

true if success, false otherwise

7.20 Internal: Tracking Functions

Collaboration diagram for Internal: Tracking Functions:



Functions

- void `gslc_TrackTouch` (`gslc_tsGui` *pGui, `gslc_tsPage` *pPage, int16_t nX, int16_t nY, uint16_t nPress)
Handles a touch event and performs the necessary tracking, glowing and selection actions depending on the press state.
- void `gslc_TrackInput` (`gslc_tsGui` *pGui, `gslc_tsPage` *pPage, `gslc_telInputRawEvent` eInputEvent, int16_t nInputVal)
Handles a direct input event and performs the necessary tracking, glowing and selection actions depending on the state.
- bool `gslc_InputMapLookup` (`gslc_tsGui` *pGui, `gslc_telInputRawEvent` eInputEvent, int16_t nInputVal, `gslc_teAction` *peAction, int16_t *pnActionVal)

7.20.1 Detailed Description

7.20.2 Function Documentation

7.20.2.1 bool `gslc_InputMapLookup` (`gslc_tsGui` * pGui, `gslc_telInputRawEvent` eInputEvent, int16_t nInputVal, `gslc_teAction` * peAction, int16_t * pnActionVal)

Todo Doc. This API is experimental and subject to change

7.20.2.2 void `gslc_TrackInput` (`gslc_tsGui` * pGui, `gslc_tsPage` * pPage, `gslc_telInputRawEvent` eInputEvent, int16_t nInputVal)

Handles a direct input event and performs the necessary tracking, glowing and selection actions depending on the state.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pPage</i>	Pointer to current page
in	<i>eInputEvent</i>	Indication of event type
in	<i>nInputVal</i>	Additional data for event type

Returns

none

7.20.2.3 void `gslc_TrackTouch (gslc_tsGui * pGui, gslc_tsPage * pPage, int16_t nX, int16_t nY, uint16_t nPress)`

Handles a touch event and performs the necessary tracking, glowing and selection actions depending on the press state.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pPage</i>	Pointer to current page
in	<i>nX</i>	X coordinate of touch event
in	<i>nY</i>	Y coordinate of touch event
in	<i>nPress</i>	Pressure level of touch event (0 for none, else touch)

Returns

none

7.21 Internal: Cleanup Functions

Collaboration diagram for Internal: Cleanup Functions:



Functions

- void `gslc_GuiDestruct` (`gslc_tsGui` *pGui)
Free up any surfaces associated with the GUI, pages, collections and elements.
- void `gslc_PageDestruct` (`gslc_tsGui` *pGui, `gslc_tsPage` *pPage)
Free up any members associated with a page.
- void `gslc_CollectDestruct` (`gslc_tsGui` *pGui, `gslc_tsCollect` *pCollect)
Free up any members associated with an element collection.
- void `gslc_ElemDestruct` (`gslc_tsElem` *pElem)
Free up any members associated with an element.
- void `gslc_ResetFont` (`gslc_tsFont` *pFont)
Initialize a Font struct.
- void `gslc_ResetElem` (`gslc_tsElem` *pElem)
Initialize an Element struct.

7.21.1 Detailed Description

7.21.2 Function Documentation

7.21.2.1 void `gslc_CollectDestruct` (`gslc_tsGui` * *pGui*, `gslc_tsCollect` * *pCollect*)

Free up any members associated with an element collection.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to collection

Returns

none

7.21.2.2 void `gslc_ElemDestruct (gslc_tsElem * pElem)`

Free up any members associated with an element.

Parameters

in	<i>pElem</i>	Pointer to element
----	--------------	--------------------

Returns

none

7.21.2.3 void gslc_GuiDestruct (gslc_tsGui * *pGui*)

Free up any surfaces associated with the GUI, pages, collections and elements.

Also frees up any fonts.

- Called by [gslc_Quit\(\)](#)

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

7.21.2.4 void gslc_PageDestruct (gslc_tsGui * *pGui*, gslc_tsPage * *pPage*)

Free up any members associated with a page.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pPage</i>	Pointer to Page

Returns

none

7.21.2.5 void gslc_ResetElem (gslc_tsElem * *pElem*)

Initialize an Element struct.

Parameters

in	<i>pElem</i>	Pointer to Element
----	--------------	--------------------

Returns

none

7.21.2.6 void gslc_ResetFont (gslc_tsFont * *pFont*)

Initialize a Font struct.

Parameters

in	<i>pFont</i>	Pointer to Font
----	--------------	-----------------

Returns

none

Chapter 8

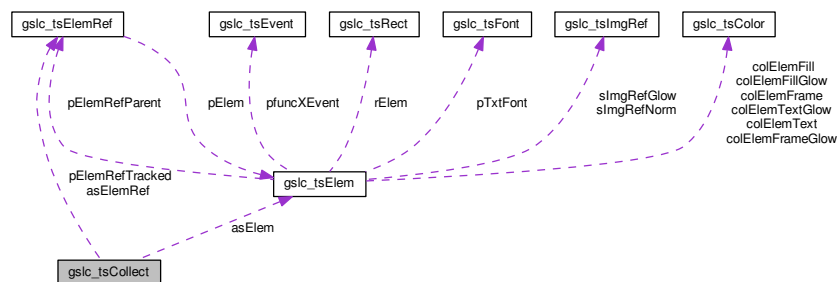
Data Structure Documentation

8.1 gslc_tsCollect Struct Reference

Element collection struct.

```
#include <GUIslice.h>
```

Collaboration diagram for gslc_tsCollect:



Data Fields

- `gslc_tsElem * asElem`
Array of elements.
- `uint16_t nElemMax`
Maximum number of elements to allocate (in RAM)
- `uint16_t nElemCnt`
Number of elements allocated.
- `int16_t nElemAutoldNext`
Next Element ID for auto-assignment.
- `gslc_tsElemRef * asElemRef`
Array of element references.
- `uint16_t nElemRefMax`
Maximum number of element references to allocate.

- `uint16_t nElemRefCnt`
Number of element references allocated.
- `gslc_tsElemRef * pElemRefTracked`
Element reference currently being touch-tracked (NULL for none)
- `int16_t nElemIndFocused`
Element index currently in focus (eg. by keyboard/pin control), `GSLC_IND_NONE` for none.

8.1.1 Detailed Description

Element collection struct.

- Collections are used to maintain a list of elements and any touch tracking status.
- Pages and Compound Elements both instantiate a Collection

The documentation for this struct was generated from the following file:

- `src/GUISlice.h`

8.2 gslc_tsColor Struct Reference

Color structure. Defines RGB triplet.

```
#include <GUISlice.h>
```

Data Fields

- `uint8_t r`
RGB red value.
- `uint8_t g`
RGB green value.
- `uint8_t b`
RGB blue value.

8.2.1 Detailed Description

Color structure. Defines RGB triplet.

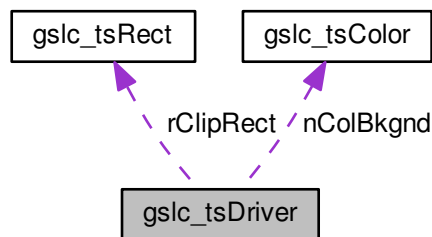
The documentation for this struct was generated from the following file:

- `src/GUISlice.h`

8.3 gslc_tsDriver Struct Reference

```
#include <GUIslice_drv_adagfx.h>
```

Collaboration diagram for gslc_tsDriver:



Data Fields

- [gslc_tsColor nColBkgnd](#)
Background color (if not image-based)
- [gslc_tsRect rClipRect](#)
Clipping rectangle.

8.3.1 Field Documentation

8.3.1.1 [gslc_tsColor](#) [gslc_tsDriver::nColBkgnd](#)

Background color (if not image-based)

8.3.1.2 [gslc_tsRect](#) [gslc_tsDriver::rClipRect](#)

Clipping rectangle.

The documentation for this struct was generated from the following files:

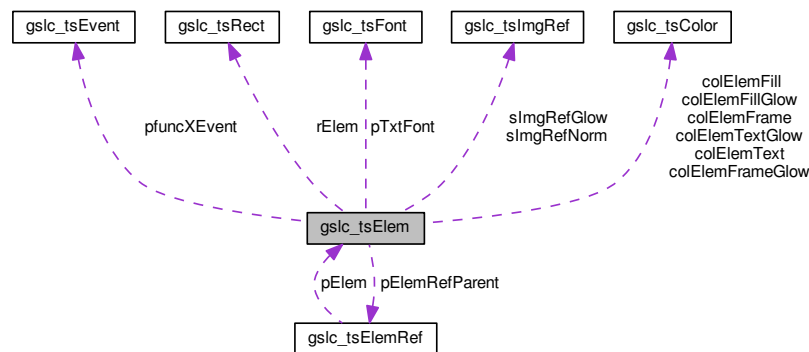
- [src/GUIslice_drv_adagfx.h](#)
- [src/GUIslice_drv_m5stack.h](#)
- [src/GUIslice_drv_tft_espi.h](#)
- [src/GUIslice_drv_utft.h](#)

8.4 gslc_tsElem Struct Reference

Element Struct.

```
#include <GUIslice.h>
```

Collaboration diagram for gslc_tsElem:



Data Fields

- `int16_t nId`
Element ID specified by user.
- `uint8_t nFeatures`
Element feature vector (appearance/behavior)
- `int16_t nType`
Element type enumeration.
- `gslc_tsRect rElem`
Rect region containing element.
- `int16_t nGroup`
Group ID that the element belongs to.
- `gslc_tsColor colElemFrame`
Color for frame.
- `gslc_tsColor colElemFill`
Color for background fill.
- `gslc_tsColor colElemFrameGlow`
Color to use for frame when glowing.
- `gslc_tsColor colElemFillGlow`
Color to use for fill when glowing.
- `gslc_tsImgRef sImgRefNorm`
Image reference to draw (normal)
- `gslc_tsImgRef sImgRefGlow`
Image reference to draw (glowing)
- `gslc_tsElemRef * pElemRefParent`
Parent element reference.
- `char * pStrBuf`

- Ptr to text string buffer to overlay.*
- [uint8_t nStrBufMax](#)
Size of string buffer.
- [gslc_teTxtFlags eTxtFlags](#)
Flags associated with text buffer.
- [gslc_tsColor colElemText](#)
Color of overlay text.
- [gslc_tsColor colElemTextGlow](#)
Color of overlay text when glowing.
- [int8_t eTxtAlign](#)
Alignment of overlay text.
- [int8_t nTxtMarginX](#)
Margin of overlay text within rect region (x offset)
- [int8_t nTxtMarginY](#)
Margin of overlay text within rect region (y offset)
- [gslc_tsFont * pTxtFont](#)
Ptr to Font for overlay text.
- [void * pXData](#)
Ptr to extended data structure.
- [GSLC_CB_EVENT pfuncXEvent](#)
UNUSED: Callback func ptr for event tree (draw,touch,tick)
- [GSLC_CB_DRAW pfuncXDraw](#)
Callback func ptr for custom drawing.
- [GSLC_CB_TOUCH pfuncXTouch](#)
Callback func ptr for touch.
- [GSLC_CB_TICK pfuncXTick](#)
Callback func ptr for timer/main loop tick.

8.4.1 Detailed Description

Element Struct.

- Represents a single graphic element in the GUIslice environment
- A page is made up of a number of elements
- Each element is created with a user-specified ID for further accesses (or `GSLC_ID_AUTO` for it to be auto-generated)
- Display order of elements in a page is based upon the creation order
- Extensions to the core element types is provided through the `pXData` reference and `pfuncX*` callback functions.

The documentation for this struct was generated from the following file:

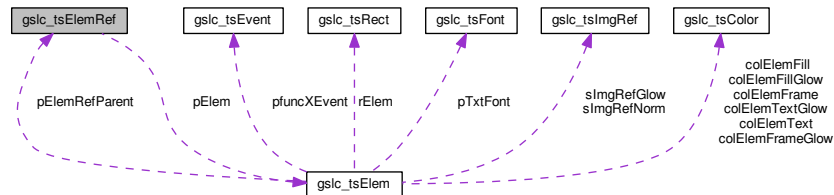
- [src/GUIslice.h](#)

8.5 gslc_tsElemRef Struct Reference

Element reference structure.

```
#include <GUIslice.h>
```

Collaboration diagram for gslc_tsElemRef:



Data Fields

- [gslc_tsElem](#) * [pElem](#)
Pointer to element in memory [RAM,FLASH].
- [gslc_teElemRefFlags](#) [eElemFlags](#)
Element reference flags.

8.5.1 Detailed Description

Element reference structure.

The documentation for this struct was generated from the following file:

- [src/GUIslice.h](#)

8.6 gslc_tsEvent Struct Reference

Event structure.

```
#include <GUIslice.h>
```

Data Fields

- [gslc_teEventType](#) [eType](#)
Event type.
- [uint8_t](#) [nSubType](#)
Event sub-type.
- [void](#) * [pvScope](#)
Event target scope (eg. Page,Collection,Event)
- [void](#) * [pvData](#)
Generic data pointer for event.

8.6.1 Detailed Description

Event structure.

The documentation for this struct was generated from the following file:

- [src/GUISlice.h](#)

8.7 gslc_tsEventTouch Struct Reference

Structure used to pass touch data through event.

```
#include <GUISlice.h>
```

Data Fields

- [gslc_teTouch eTouch](#)
Touch state.
- [int16_t nX](#)
Touch X coordinate (or param1)
- [int16_t nY](#)
Touch Y coordinate (or param2)

8.7.1 Detailed Description

Structure used to pass touch data through event.

The documentation for this struct was generated from the following file:

- [src/GUISlice.h](#)

8.8 gslc_tsFont Struct Reference

Font reference structure.

```
#include <GUISlice.h>
```

Data Fields

- [int16_t nId](#)
Font ID specified by user.
- [gslc_teFontRefType eFontRefType](#)
Font reference type.
- [gslc_teFontRefMode eFontRefMode](#)
Font reference mode.
- `const void *` [pvFont](#)
Void ptr to the font reference (type defined by driver)
- [uint16_t nSize](#)
Font size.

- Adafruit GFX Touch Flip x axis.
- [uint8_t nFlipY](#)
- Adafruit GFX Touch Flip x axis.
- [uint16_t nTouchCalXMin](#)
- Calibration X minimum reading.
- [uint16_t nTouchCalXMax](#)
- Calibration X maximum reading.
- [uint16_t nTouchCalYMin](#)
- Calibration Y minimum reading.
- [uint16_t nTouchCalYMax](#)
- Calibration Y maximum reading.
- [gslc_tsFont * asFont](#)
- Collection of loaded fonts.
- [uint8_t nFontMax](#)
- Maximum number of fonts to allocate.
- [uint8_t nFontCnt](#)
- Number of fonts allocated.
- [uint8_t nRoundRadius](#)
- Radius for rounded elements.
- [gslc_tsColor sTransCol](#)
- Color used for transparent image regions (GSLC_BMP_TRANS_EN=1)
- [gslc_tsElem sElemTmp](#)
- Temporary element.
- [gslc_tsElemRef sElemRefTmp](#)
- Temporary element reference.
- [gslc_tsElem sElemTmpProg](#)
- Temporary element for Flash compatibility.
- [gslc_tInitStat eInitStatTouch](#)
- Status of touch initialization.
- [int16_t nTouchLastX](#)
- Last touch event X coord.
- [int16_t nTouchLastY](#)
- Last touch event Y coord.
- [uint16_t nTouchLastPress](#)
- Last touch event pressure (0=none)
- [bool bTouchRemapEn](#)
- Enable touch remapping?
- [bool bTouchRemapYX](#)
- Enable touch controller swapping of X & Y.
- [void * pvDriver](#)
- Driver-specific members (gslc_tsDriver*)
- [bool bRedrawPartialEn](#)
- Driver supports partial page redraw.
- [gslc_tsImgRef sImgRefBkgnd](#)
- Image reference for background.
- [uint8_t nFrameRateCnt](#)
- Diagnostic frame rate count.
- [uint8_t nFrameRateStart](#)
- Diagnostic frame rate timestamp.
- [gslc_tsPage * asPage](#)
- Array of all pages defined in system.

- `uint8_t nPageMax`
Maximum number of pages that can be defined.
- `uint8_t nPageCnt`
Current number of pages defined.
- `gslc_tsPage * apPageStack [GSLC_STACK__MAX]`
Stack of pages.
- `bool abPageStackActive [GSLC_STACK__MAX]`
Whether page in stack can receive touch events.
- `bool abPageStackDoDraw [GSLC_STACK__MAX]`
Whether page in stack is still actively drawn.
- `bool bScreenNeedRedraw`
Screen requires a redraw.
- `bool bScreenNeedFlip`
Screen requires a page flip.
- `bool bInvalidateEn`
A region of the display has been invalidated.
- `gslc_tsRect rInvalidateRect`
The rect region that has been invalidated.
- `GSLC_CB_PIN_POLL pfuncPinPoll`
Callback func ptr for pin polling.
- `gslc_tsInputMap * asInputMap`
Array of input maps.
- `uint8_t nInputMapMax`
Maximum number of input maps.
- `uint8_t nInputMapCnt`
Current number of input maps.

8.9.1 Detailed Description

GUI structure.

- Contains all GUI state and content
- Maintains list of one or more pages

The documentation for this struct was generated from the following file:

- `src/GUIslice.h`

8.10 gslc_tsImgRef Struct Reference

Image reference structure.

```
#include <GUIslice.h>
```

Data Fields

- `const unsigned char * pImgBuf`
Pointer to input image buffer in memory [RAM,FLASH].
- `const char * pFname`
Pathname to input image file [FILE,SD].
- `gslc_telmgRefFlags elmngFlags`
Image reference flags.
- `void * pvImgRaw`
Ptr to raw output image data (for pre-loaded images)

8.10.1 Detailed Description

Image reference structure.

The documentation for this struct was generated from the following file:

- `src/GUIslice.h`

8.11 gslc_tsInputMap Struct Reference

Input mapping.

```
#include <GUIslice.h>
```

Data Fields

- `gslc_telInputRawEvent eEvent`
The input event.
- `int16_t nVal`
The value associated with the input event.
- `gslc_teAction eAction`
Resulting action.
- `int16_t nActionVal`
The value for the output action.

8.11.1 Detailed Description

Input mapping.

- Describes mapping from keyboard or GPIO input to a GUI action (such as changing the current element focus)
- This is generally used to support keyboard or GPIO control over the GUI operation

The documentation for this struct was generated from the following file:

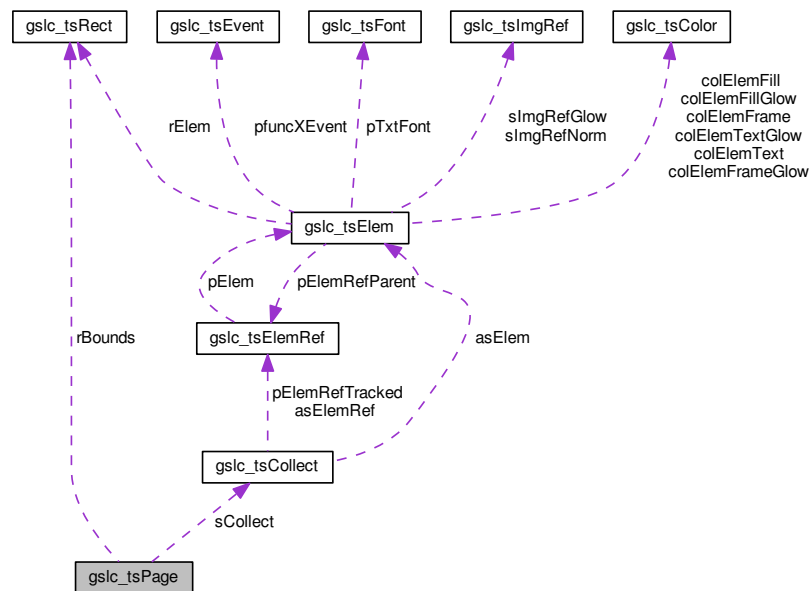
- `src/GUIslice.h`

8.12 gslc_tsPage Struct Reference

Page structure.

```
#include <GUIslice.h>
```

Collaboration diagram for gslc_tsPage:



Data Fields

- [gslc_tsCollect sCollect](#)
Collection of elements on page.
- [int16_t nPageId](#)
Page identifier.
- [gslc_tsRect rBounds](#)
Bounding rect for page elements.

8.12.1 Detailed Description

Page structure.

- A page contains a collection of elements
- Many redraw functions operate at a page level
- Maintains state as to whether redraw or screen flip is required

The documentation for this struct was generated from the following file:

- [src/GUIslice.h](#)

8.13 gslc_tsPt Struct Reference

Define point coordinates.

```
#include <GUIslice.h>
```

Data Fields

- `int16_t x`
X coordinate.
- `int16_t y`
Y coordinate.

8.13.1 Detailed Description

Define point coordinates.

The documentation for this struct was generated from the following file:

- `src/GUIslice.h`

8.14 gslc_tsRect Struct Reference

Rectangular region. Defines X,Y corner coordinates plus dimensions.

```
#include <GUIslice.h>
```

Data Fields

- `int16_t x`
X coordinate of corner.
- `int16_t y`
Y coordinate of corner.
- `uint16_t w`
Width of region.
- `uint16_t h`
Height of region.

8.14.1 Detailed Description

Rectangular region. Defines X,Y corner coordinates plus dimensions.

The documentation for this struct was generated from the following file:

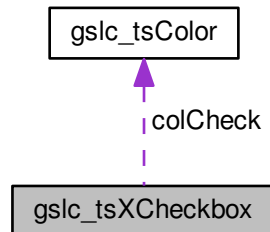
- `src/GUIslice.h`

8.15 gslc_tsXCheckbox Struct Reference

Extended data for Checkbox element.

```
#include <XCheckbox.h>
```

Collaboration diagram for gslc_tsXCheckbox:



Data Fields

- bool `bRadio`
Radio-button operation if true.
- `gslc_teXCheckboxStyle` `nStyle`
Drawing style for element.
- bool `bChecked`
Indicates if it is selected (checked)
- `gslc_tsColor` `colCheck`
Color of checked inner fill.
- `GSLC_CB_XCHECKBOX` `pfuncXToggle`
Callback event to say element has changed.

8.15.1 Detailed Description

Extended data for Checkbox element.

8.15.2 Field Documentation

8.15.2.1 bool `gslc_tsXCheckbox::bChecked`

Indicates if it is selected (checked)

8.15.2.2 bool `gslc_tsXCheckbox::bRadio`

Radio-button operation if true.

8.15.2.3 gslc_tsColor gslc_tsXCheckbox::colCheck

Color of checked inner fill.

8.15.2.4 gslc_teXCheckboxStyle gslc_tsXCheckbox::nStyle

Drawing style for element.

8.15.2.5 GSLC_CB_XCHECKBOX gslc_tsXCheckbox::pfuncXToggle

Callback event to say element has changed.

The documentation for this struct was generated from the following file:

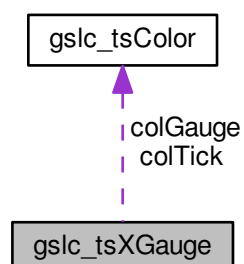
- [src/elem/XCheckbox.h](#)

8.16 gslc_tsXGauge Struct Reference

Extended data for Gauge element.

```
#include <XGauge.h>
```

Collaboration diagram for gslc_tsXGauge:



Data Fields

- `int16_t nMin`
Minimum control value.
- `int16_t nMax`
Maximum control value.
- `int16_t nVal`
Current control value.
- `int16_t nValLast`
Last value.
- `bool bValLastValid`
Last value valid?
- `gslc_tsXGaugeStyle nStyle`
Gauge sub-type.
- `gslc_tsColor colGauge`
Color of gauge fill bar.
- `gslc_tsColor colTick`
Color of gauge tick marks.
- `uint16_t nTickCnt`
Number of gauge tick marks.
- `uint16_t nTickLen`
Length of gauge tick marks.
- `bool bVert`
Vertical if true, else Horizontal.
- `bool bFlip`
Reverse direction of gauge.
- `uint16_t nIndicLen`
Indicator length.
- `uint16_t nIndicTip`
Size of tip at end of indicator.
- `bool bIndicFill`
Fill the indicator if true.

8.16.1 Detailed Description

Extended data for Gauge element.

8.16.2 Field Documentation

8.16.2.1 `bool gslc_tsXGauge::bFlip`

Reverse direction of gauge.

8.16.2.2 `bool gslc_tsXGauge::bIndicFill`

Fill the indicator if true.

8.16.2.3 `bool gslc_tsXGauge::bValLastValid`

Last value valid?

8.16.2.4 `bool gslc_tsXGauge::bVert`

Vertical if true, else Horizontal.

8.16.2.5 `gslc_tsColor gslc_tsXGauge::colGauge`

Color of gauge fill bar.

8.16.2.6 `gslc_tsColor gslc_tsXGauge::colTick`

Color of gauge tick marks.

8.16.2.7 `uint16_t gslc_tsXGauge::nIndicLen`

Indicator length.

8.16.2.8 `uint16_t gslc_tsXGauge::nIndicTip`

Size of tip at end of indicator.

8.16.2.9 `int16_t gslc_tsXGauge::nMax`

Maximum control value.

8.16.2.10 `int16_t gslc_tsXGauge::nMin`

Minimum control value.

8.16.2.11 `gslc_tsXGaugeStyle gslc_tsXGauge::nStyle`

Gauge sub-type.

8.16.2.12 `uint16_t gslc_tsXGauge::nTickCnt`

Number of gauge tick marks.

8.16.2.13 uint16_t gslc_tsXGauge::nTickLen

Length of gauge tick marks.

8.16.2.14 int16_t gslc_tsXGauge::nVal

Current control value.

8.16.2.15 int16_t gslc_tsXGauge::nValLast

Last value.

The documentation for this struct was generated from the following file:

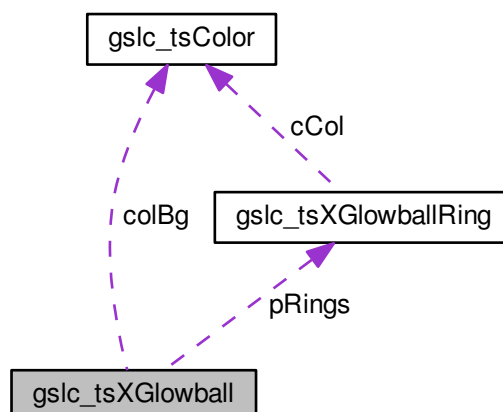
- [src/elem/XGauge.h](#)

8.17 gslc_tsXGlowball Struct Reference

Extended data for Slider element.

```
#include <XGlowball.h>
```

Collaboration diagram for gslc_tsXGlowball:



Data Fields

- `int16_t nMidX`
Gauge midpoint X coord.
- `int16_t nMidY`
Gauge midpoint Y coord.
- `gslc_tsXGlowballRing * pRings`
Ring definition array.
- `uint8_t nNumRings`
Number of rings in definition.
- `uint16_t nQuality`
Rendering quality (number of segments / rotation)
- `int16_t nAngStart`
Starting angle (0..510 degrees)
- `int16_t nAngEnd`
Ending angle (0..510 degrees)
- `gslc_tsColor colBg`
Background color (for redraw)
- `int16_t nVal`
Current value.
- `int16_t nValLast`
Previous value.

8.17.1 Detailed Description

Extended data for Slider element.

8.17.2 Field Documentation

8.17.2.1 `gslc_tsColor gslc_tsXGlowball::colBg`

Background color (for redraw)

8.17.2.2 `int16_t gslc_tsXGlowball::nAngEnd`

Ending angle (0..510 degrees)

8.17.2.3 `int16_t gslc_tsXGlowball::nAngStart`

Starting angle (0..510 degrees)

8.17.2.4 `int16_t gslc_tsXGlowball::nMidX`

Gauge midpoint X coord.

8.17.2.5 `int16_t gslc_tsXGlowball::nMidY`

Gauge midpoint Y coord.

8.17.2.6 `uint8_t gslc_tsXGlowball::nNumRings`

Number of rings in definition.

8.17.2.7 `uint16_t gslc_tsXGlowball::nQuality`

Rendering quality (number of segments / rotation)

8.17.2.8 `int16_t gslc_tsXGlowball::nVal`

Current value.

8.17.2.9 `int16_t gslc_tsXGlowball::nValLast`

Previous value.

8.17.2.10 `gslc_tsXGlowballRing* gslc_tsXGlowball::pRings`

Ring definition array.

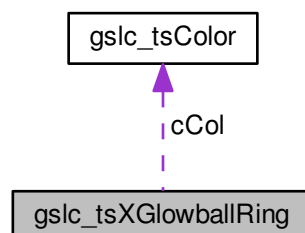
The documentation for this struct was generated from the following file:

- [src/elem/XGlowball.h](#)

8.18 `gslc_tsXGlowballRing` Struct Reference

```
#include <XGlowball.h>
```

Collaboration diagram for `gslc_tsXGlowballRing`:



Data Fields

- [uint8_t nRad1](#)
- [uint8_t nRad2](#)
- [gslc_tsColor cCol](#)

8.18.1 Field Documentation

8.18.1.1 [gslc_tsColor](#) [gslc_tsXGlowballRing::cCol](#)

8.18.1.2 [uint8_t](#) [gslc_tsXGlowballRing::nRad1](#)

8.18.1.3 [uint8_t](#) [gslc_tsXGlowballRing::nRad2](#)

The documentation for this struct was generated from the following file:

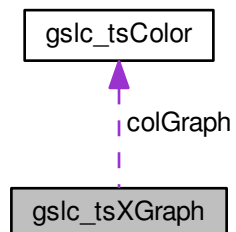
- [src/elem/XGlowball.h](#)

8.19 gslc_tsXGraph Struct Reference

Extended data for Graph element.

```
#include <XGraph.h>
```

Collaboration diagram for [gslc_tsXGraph](#):



Data Fields

- `int16_t * pBuf`
Ptr to the data buffer (circular buffer)
- `uint8_t nMargin`
Margin for graph area within element rect.
- `gslc_tsColor colGraph`
Color of the graph.
- `gslc_teXGraphStyle eStyle`
Style of the graph.
- `uint16_t nBufMax`
Maximum number of points in buffer.
- `bool bScrollEn`
Enable for scrollbar.
- `uint16_t nScrollPos`
Current scrollbar position.
- `uint16_t nWndHeight`
Visible window height.
- `uint16_t nWndWidth`
Visible window width.
- `int16_t nPlotValMax`
Visible window maximum value.
- `int16_t nPlotValMin`
Visible window minimum value.
- `uint16_t nPlotIndMax`
Number of data points to show in window.
- `uint16_t nBufCnt`
Number of points in buffer.
- `uint16_t nPlotIndStart`
First row of current window.

8.19.1 Detailed Description

Extended data for Graph element.

8.19.2 Field Documentation

8.19.2.1 `bool gslc_tsXGraph::bScrollEn`

Enable for scrollbar.

8.19.2.2 `gslc_tsColor gslc_tsXGraph::colGraph`

Color of the graph.

8.19.2.3 `gslc_tsXGraphStyle` `gslc_tsXGraph::eStyle`

Style of the graph.

8.19.2.4 `uint16_t` `gslc_tsXGraph::nBufCnt`

Number of points in buffer.

8.19.2.5 `uint16_t` `gslc_tsXGraph::nBufMax`

Maximum number of points in buffer.

8.19.2.6 `uint8_t` `gslc_tsXGraph::nMargin`

Margin for graph area within element rect.

8.19.2.7 `uint16_t` `gslc_tsXGraph::nPlotIndMax`

Number of data points to show in window.

8.19.2.8 `uint16_t` `gslc_tsXGraph::nPlotIndStart`

First row of current window.

8.19.2.9 `int16_t` `gslc_tsXGraph::nPlotValMax`

Visible window maximum value.

8.19.2.10 `int16_t` `gslc_tsXGraph::nPlotValMin`

Visible window minimum value.

8.19.2.11 `uint16_t` `gslc_tsXGraph::nScrollPos`

Current scrollbar position.

8.19.2.12 `uint16_t` `gslc_tsXGraph::nWndHeight`

Visible window height.

8.19.2.13 uint16_t gslc_tsXGraph::nWndWidth

Visible window width.

8.19.2.14 int16_t* gslc_tsXGraph::pBuf

Ptr to the data buffer (circular buffer)

The documentation for this struct was generated from the following file:

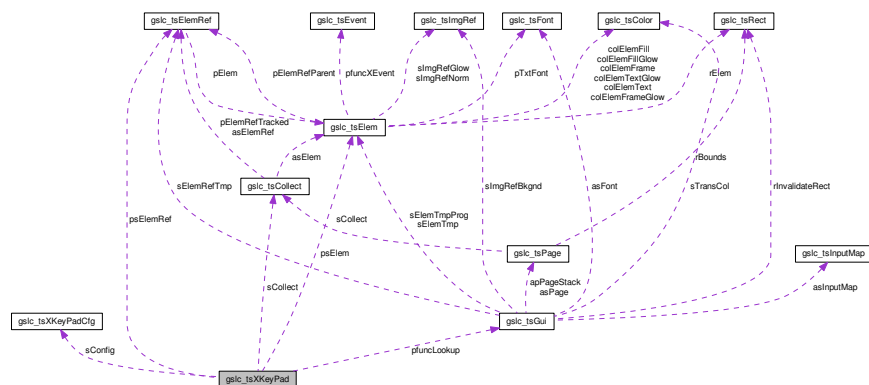
- [src/elem/XGraph.h](#)

8.20 gslc_tsXKeyPad Struct Reference

Extended data for KeyPad element.

```
#include <XKeyPad.h>
```

Collaboration diagram for gslc_tsXKeyPad:



Data Fields

- `uint8_t nValStrPos`
Current number of characters stored in edit value string.
- `char acValStr [XKEYPAD_VAL_LEN]`
Storage for edit value string.
- `bool bValPositive`
Value is positive if true, negative if false.
- `bool bValDecimalPt`
Value string includes decimal point.
- `char ** pacKeys`
Array of character strings for KeyPad labels.
- `gslc_tsXKeyPadCfg sConfig`
Configuration options.

- [GSLC_CB_INPUT pfuncCb](#)
Callback function for KeyPad actions.
- [XKEYPAD_LOOKUP pfuncLookup](#)
Callback function for converting key into key label.
- [int16_t nTargetId](#)
Target element ID associated with keypad (GSLC_CB_INPUT)
- [gslc_tsCollect sCollect](#)
Collection management for sub-elements.
- [uint8_t nSubElemMax](#)
Maximum number of sub-elements to create within KeyPad container.
- [gslc_tsElemRef * psElemRef](#)
Ptr to storage for sub-element references.
- [gslc_tsElem * psElem](#)
Ptr to storage for sub-elements.

8.20.1 Detailed Description

Extended data for KeyPad element.

8.20.2 Field Documentation

8.20.2.1 `char gslc_tsXKeyPad::acValStr[XKEYPAD_VAL_LEN]`

Storage for edit value string.

8.20.2.2 `bool gslc_tsXKeyPad::bValDecimalPt`

Value string includes decimal point.

8.20.2.3 `bool gslc_tsXKeyPad::bValPositive`

Value is positive if true, negative if false.

8.20.2.4 `uint8_t gslc_tsXKeyPad::nSubElemMax`

Maximum number of sub-elements to create within KeyPad container.

8.20.2.5 `int16_t gslc_tsXKeyPad::nTargetId`

Target element ID associated with keypad (GSLC_CB_INPUT)

8.20.2.6 `uint8_t gslc_tsXKeyPad::nValStrPos`

Current number of characters stored in edit value string.

- `gslc_tsXKeyPad sKeyPad`
Base XKeyPad element.
- `gslc_tsElemRef asElemRef [XKEYPADALPHA_ELEM_MAX]`
Storage for sub-element references.
- `gslc_tsElem asElem [XKEYPADALPHA_ELEM_MAX]`
Storage for sub-elements.

8.21.1.1 gslc_tsElem gslc_tsXKeyPad_Alpha::asElem[XKEYPADALPHA_ELEM_MAX]

8.21.1.2 gslc_tsElemRef gslc_tsXKeyPad_Alpha::asElemRef[XKEYPADALPHA_ELEM_MAX]

8.21.1.3 gslc_tsXKeyPad gslc_tsXKeyPad_Alpha::sKeyPad

The documentation for this struct was generated from the following file:

- `src/elem/XKeyPad_Alpha.h`

```
#include <XKeyPad_Num.h>
```

[illegible]

Data Fields

- [gslc_tsXKeyPad sKeyPad](#)
Base XKeyPad element.
- [gslc_tsElemRef asElemRef \[XKEYPADNUM_ELEM_MAX\]](#)
Storage for sub-element references.
- [gslc_tsElem asElem \[XKEYPADNUM_ELEM_MAX\]](#)
Storage for sub-elements.

8.22.1 Field Documentation

8.22.1.1 `gslc_tsElem gslc_tsXKeyPad_Num::asElem[XKEYPADNUM_ELEM_MAX]`

Storage for sub-elements.

8.22.1.2 `gslc_tsElemRef gslc_tsXKeyPad_Num::asElemRef[XKEYPADNUM_ELEM_MAX]`

Storage for sub-element references.

8.22.1.3 `gslc_tsXKeyPad gslc_tsXKeyPad_Num::sKeyPad`

Base XKeyPad element.

The documentation for this struct was generated from the following file:

- [src/elem/XKeyPad_Num.h](#)

8.23 `gslc_tsXKeyPadCfg` Struct Reference

Configuration for the KeyPad.

```
#include <XKeyPad.h>
```

Data Fields

- bool [bFloatEn](#)
Enable floating point (ie. decimal point)
- bool [bSignEn](#)
Enable negative numbers.
- bool [bRoundEn](#)
Enable rounded corners.
- int8_t [nButtonSzW](#)
Button width (in pixels)
- int8_t [nButtonSzH](#)
Button height (in pixels)
- char ** [pacKeys](#)
Array of character strings for KeyPad labels.
- int16_t [nFontId](#)
Configured font for KeyPad labels.
- int16_t [nOffsetX](#)
Configured offset (X direction) for buttons from parent container.
- int16_t [nOffsetY](#)
Configured offset (Y direction) for buttons from parent container.
- int8_t [nFrameMargin](#)
Margin around text value field.
- uint8_t [nMaxCols](#)
Maximum number of columns to occupy.
- uint8_t [nMaxRows](#)
Maximum number of rows to occupy.

8.23.1 Detailed Description

Configuration for the KeyPad.

8.23.2 Field Documentation

8.23.2.1 bool gslc_tsXKeyPadCfg::bFloatEn

Enable floating point (ie. decimal point)

8.23.2.2 bool gslc_tsXKeyPadCfg::bRoundEn

Enable rounded corners.

8.23.2.3 bool gslc_tsXKeyPadCfg::bSignEn

Enable negative numbers.

8.23.2.4 `int8_t gslc_tsXKeyPadCfg::nButtonSzH`

Button height (in pixels)

8.23.2.5 `int8_t gslc_tsXKeyPadCfg::nButtonSzW`

Button width (in pixels)

8.23.2.6 `int16_t gslc_tsXKeyPadCfg::nFontId`

Configured font for KeyPad labels.

8.23.2.7 `int8_t gslc_tsXKeyPadCfg::nFrameMargin`

Margin around text value field.

8.23.2.8 `uint8_t gslc_tsXKeyPadCfg::nMaxCols`

Maximum number of columns to occupy.

8.23.2.9 `uint8_t gslc_tsXKeyPadCfg::nMaxRows`

Maximum number of rows to occupy.

8.23.2.10 `int16_t gslc_tsXKeyPadCfg::nOffsetX`

Configured offset (X direction) for buttons from parent container.

8.23.2.11 `int16_t gslc_tsXKeyPadCfg::nOffsetY`

Configured offset (Y direction) for buttons from parent container.

8.23.2.12 `char** gslc_tsXKeyPadCfg::pacKeys`

Array of character strings for KeyPad labels.

The documentation for this struct was generated from the following file:

- [src/elem/XKeyPad.h](#)

8.24 gslc_tsXKeyPadData Struct Reference

Input callback data structure.

```
#include <XKeyPad.h>
```

Data Fields

- char * [pStr](#)
Final value of edited value field.
- int16_t [nTargetId](#)
Target element ID to receive the value.

8.24.1 Detailed Description

Input callback data structure.

- This struct is returned in GSLC_CB_INPUT when the KeyPad edits are complete, and is used to provide the resulting edited value.

8.24.2 Field Documentation

8.24.2.1 int16_t gslc_tsXKeyPadData::nTargetId

Target element ID to receive the value.

8.24.2.2 char* gslc_tsXKeyPadData::pStr

Final value of edited value field.

The documentation for this struct was generated from the following file:

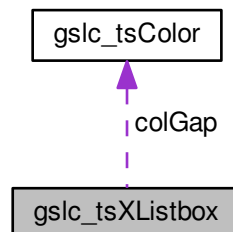
- [src/elem/XKeyPad.h](#)

8.25 gslc_tsXListbox Struct Reference

Extended data for Listbox element.

```
#include <XListbox.h>
```

Collaboration diagram for gslc_tsXListbox:



Data Fields

- `uint8_t * pBufItems`
Buffer containing items.
- `uint16_t nBufItemsMax`
Max size of buffer containing items.
- `uint16_t nBufItemsPos`
Current buffer position.
- `int16_t nItemCnt`
Number of items in the list.
- `int8_t nCols`
Number of columns.
- `int8_t nRows`
Number of columns (or XLSITBOX_SIZE_AUTO to calculate)
- `bool bNeedRecalc`
Determine if sizing may need recalc.
- `int8_t nMarginW`
Margin inside main listbox area (X offset)
- `int8_t nMarginH`
Margin inside main listbox area (Y offset)
- `int16_t nItemW`
Width of listbox item.
- `int16_t nItemH`
Height of listbox item.
- `int8_t nItemGap`
Gap between listbox items.
- `gslc_tsColor colGap`
Gap color.
- `bool blItemAutoSizeW`
Enable auto-sizing of items (in width)
- `bool blItemAutoSizeH`
Enable auto-sizing of items (in height)
- `int16_t nItemCurSel`
Currently selected item (XLISTBOX_SEL_NONE for none)
- `int16_t nItemCurSelLast`
Old selected item to redraw (XLISTBOX_SEL_NONE for none)
- `int16_t nItemSavedSel`
Persistent selected item (ie. saved selection)
- `int16_t nItemTop`
Item to show at top of list after scrolling (0 is default)
- `GSLC_CB_XLISTBOX_SEL pfuncXSel`
Callback func ptr for selection update.

8.25.1 Detailed Description

Extended data for Listbox element.

8.25.2 Field Documentation

8.25.2.1 `bool gslc_tsXListbox::bltemAutoSizeH`

Enable auto-sizing of items (in height)

8.25.2.2 `bool gslc_tsXListbox::bltemAutoSizeW`

Enable auto-sizing of items (in width)

8.25.2.3 `bool gslc_tsXListbox::bNeedRecalc`

Determine if sizing may need recalc.

8.25.2.4 `gslc_tsColor gslc_tsXListbox::colGap`

Gap color.

8.25.2.5 `uint16_t gslc_tsXListbox::nBufItemsMax`

Max size of buffer containing items.

8.25.2.6 `uint16_t gslc_tsXListbox::nBufItemsPos`

Current buffer position.

8.25.2.7 `int8_t gslc_tsXListbox::nCols`

Number of columns.

8.25.2.8 `int16_t gslc_tsXListbox::nItemCnt`

Number of items in the list.

8.25.2.9 `int16_t gslc_tsXListbox::nItemCurSel`

Currently selected item (XLISTBOX_SEL_NONE for none)

8.25.2.10 `int16_t gslc_tsXListbox::nItemCurSelLast`

Old selected item to redraw (XLISTBOX_SEL_NONE for none)

8.25.2.11 int8_t gslc_tsXListBox::nItemGap

Gap between listbox items.

8.25.2.12 int16_t gslc_tsXListBox::nItemH

Height of listbox item.

8.25.2.13 int16_t gslc_tsXListBox::nItemSavedSel

Persistent selected item (ie. saved selection)

8.25.2.14 int16_t gslc_tsXListBox::nItemTop

Item to show at top of list after scrolling (0 is default)

8.25.2.15 int16_t gslc_tsXListBox::nItemW

Width of listbox item.

8.25.2.16 int8_t gslc_tsXListBox::nMarginH

Margin inside main listbox area (Y offset)

8.25.2.17 int8_t gslc_tsXListBox::nMarginW

Margin inside main listbox area (X offset)

8.25.2.18 int8_t gslc_tsXListBox::nRows

Number of columns (or XLSITBOX_SIZE_AUTO to calculate)

8.25.2.19 uint8_t* gslc_tsXListBox::pBufItems

Buffer containing items.

8.25.2.20 GSLC_CB_XLISTBOX_SEL gslc_tsXListBox::pfuncXSel

Callback func ptr for selection update.

The documentation for this struct was generated from the following file:

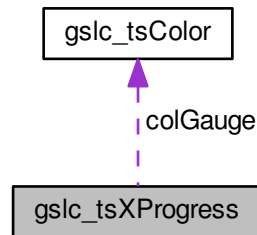
- [src/elem/XListBox.h](#)

8.26 gslc_tsXProgress Struct Reference

Extended data for Gauge element.

```
#include <XProgress.h>
```

Collaboration diagram for gslc_tsXProgress:



Data Fields

- `int16_t nMin`
Minimum control value.
- `int16_t nMax`
Maximum control value.
- `int16_t nVal`
Current control value.
- `int16_t nValLast`
Last value.
- `bool bValLastValid`
Last value valid?
- `gslc_tsColor colGauge`
Color of gauge fill bar.
- `bool bVert`
Vertical if true, else Horizontal.
- `bool bFlip`
Reverse direction of gauge.

8.26.1 Detailed Description

Extended data for Gauge element.

8.26.2 Field Documentation

8.26.2.1 `bool gslc_tsXProgress::bFlip`

Reverse direction of gauge.

8.26.2.2 `bool gslc_tsXProgress::bValLastValid`

Last value valid?

8.26.2.3 `bool gslc_tsXProgress::bVert`

Vertical if true, else Horizontal.

8.26.2.4 `gslc_tsColor gslc_tsXProgress::colGauge`

Color of gauge fill bar.

8.26.2.5 `int16_t gslc_tsXProgress::nMax`

Maximum control value.

8.26.2.6 `int16_t gslc_tsXProgress::nMin`

Minimum control value.

8.26.2.7 `int16_t gslc_tsXProgress::nVal`

Current control value.

8.26.2.8 `int16_t gslc_tsXProgress::nValLast`

Last value.

The documentation for this struct was generated from the following file:

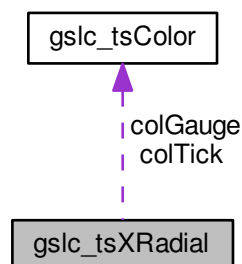
- [src/elem/XProgress.h](#)

8.27 `gslc_tsXRadial` Struct Reference

Extended data for Gauge element.

```
#include <XRadial.h>
```

Collaboration diagram for `gslc_tsXRadial`:



Data Fields

- `int16_t nMin`
Minimum control value.
- `int16_t nMax`
Maximum control value.
- `int16_t nVal`
Current control value.
- `int16_t nValLast`
Last value.
- `bool bValLastValid`
Last value valid?
- `gslc_tsColor colGauge`
Color of gauge fill bar.
- `gslc_tsColor colTick`
Color of gauge tick marks.
- `uint16_t nTickCnt`
Number of gauge tick marks.
- `uint16_t nTickLen`
Length of gauge tick marks.
- `bool bFlip`
Reverse direction of gauge.
- `uint16_t nIndicLen`
Indicator length.
- `uint16_t nIndicTip`
Size of tip at end of indicator.
- `bool bIndicFill`
Fill the indicator if true.

8.27.1 Detailed Description

Extended data for Gauge element.

8.27.2 Field Documentation

8.27.2.1 `bool gslc_tsXRadial::bFlip`

Reverse direction of gauge.

8.27.2.2 `bool gslc_tsXRadial::bIndicFill`

Fill the indicator if true.

8.27.2.3 `bool gslc_tsXRadial::bValLastValid`

Last value valid?

8.27.2.4 gslc_tsColor gslc_tXRadial::colGauge

Color of gauge fill bar.

8.27.2.5 gslc_tsColor gslc_tXRadial::colTick

Color of gauge tick marks.

8.27.2.6 uint16_t gslc_tXRadial::nIndicLen

Indicator length.

8.27.2.7 uint16_t gslc_tXRadial::nIndicTip

Size of tip at end of indicator.

8.27.2.8 int16_t gslc_tXRadial::nMax

Maximum control value.

8.27.2.9 int16_t gslc_tXRadial::nMin

Minimum control value.

8.27.2.10 uint16_t gslc_tXRadial::nTickCnt

Number of gauge tick marks.

8.27.2.11 uint16_t gslc_tXRadial::nTickLen

Length of gauge tick marks.

8.27.2.12 int16_t gslc_tXRadial::nVal

Current control value.

8.27.2.13 int16_t gslc_tXRadial::nValLast

Last value.

The documentation for this struct was generated from the following file:

- [src/ele/XRadial.h](#)

8.28 gslc_tsXRamp Struct Reference

Extended data for Gauge element.

```
#include <XRamp.h>
```

Data Fields

- `int16_t nMin`
Minimum control value.
- `int16_t nMax`
Maximum control value.
- `int16_t nVal`
Current control value.
- `int16_t nValLast`
Last value.
- `bool bValLastValid`
Last value valid?

8.28.1 Detailed Description

Extended data for Gauge element.

8.28.2 Field Documentation

8.28.2.1 `bool gslc_tsXRamp::bValLastValid`

Last value valid?

8.28.2.2 `int16_t gslc_tsXRamp::nMax`

Maximum control value.

8.28.2.3 `int16_t gslc_tsXRamp::nMin`

Minimum control value.

8.28.2.4 `int16_t gslc_tsXRamp::nVal`

Current control value.

8.28.2.5 int16_t gslc_tsXRamp::nValLast

Last value.

The documentation for this struct was generated from the following file:

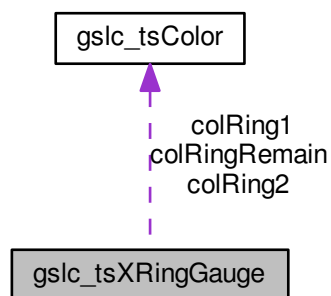
- [src/elm/XRamp.h](#)

8.29 gslc_tsXRingGauge Struct Reference

Extended data for XRingGauge element.

```
#include <XRingGauge.h>
```

Collaboration diagram for gslc_tsXRingGauge:



Data Fields

- [int16_t nValMin](#)
- [int16_t nValMax](#)
- [int16_t nAngStart](#)
- [int16_t nAngRange](#)
- [int16_t nQuality](#)
- [int8_t nThickness](#)
- [bool bGradient](#)
- [uint8_t nSegGap](#)
- [gslc_tsColor colRing1](#)
- [gslc_tsColor colRing2](#)
- [gslc_tsColor colRingRemain](#)
- [int16_t nVal](#)
Current position value.
- [int16_t nValLast](#)
Previous position value.
- [char acStrLast \[XRING_STR_MAX\]](#)

8.29.1 Detailed Description

Extended data for XRingGauge element.

8.29.2 Field Documentation

8.29.2.1 `char gslc_tsXRingGauge::acStrLast[XRING_STR_MAX]`

8.29.2.2 `bool gslc_tsXRingGauge::bGradient`

8.29.2.3 `gslc_tsColor gslc_tsXRingGauge::colRing1`

8.29.2.4 `gslc_tsColor gslc_tsXRingGauge::colRing2`

8.29.2.5 `gslc_tsColor gslc_tsXRingGauge::colRingRemain`

8.29.2.6 `int16_t gslc_tsXRingGauge::nAngRange`

8.29.2.7 `int16_t gslc_tsXRingGauge::nAngStart`

8.29.2.8 `int16_t gslc_tsXRingGauge::nQuality`

8.29.2.9 `uint8_t gslc_tsXRingGauge::nSegGap`

8.29.2.10 `int8_t gslc_tsXRingGauge::nThickness`

8.29.2.11 `int16_t gslc_tsXRingGauge::nVal`

Current position value.

8.29.2.12 `int16_t gslc_tsXRingGauge::nValLast`

Previous position value.

8.29.2.13 `int16_t gslc_tsXRingGauge::nValMax`

8.29.2.14 `int16_t gslc_tsXRingGauge::nValMin`

The documentation for this struct was generated from the following file:

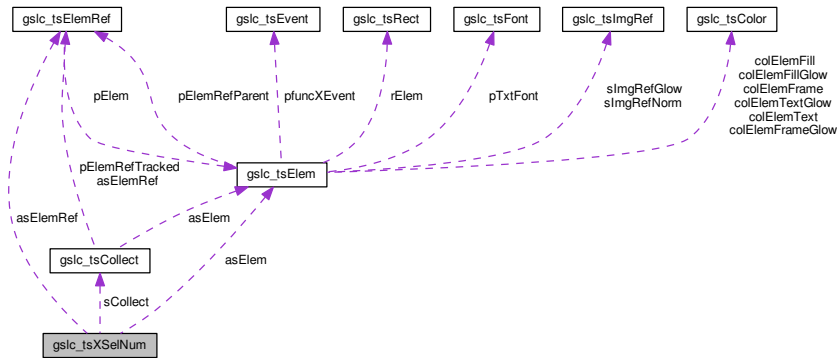
- `src/elem/XRingGauge.h`

8.30 gslc_tsXSelNum Struct Reference

Extended data for SelNum element.

```
#include <XSelNum.h>
```

Collaboration diagram for gslc_tsXSelNum:



Data Fields

- `int16_t nCounter`
Counter for demo purposes.
- `gslc_tsCollect sCollect`
Collection management for sub-elements.
- `gslc_tsElemRef asElemRef` [4]
Storage for sub-element references.
- `gslc_tsElem asElem` [4]
Storage for sub-elements.
- `char acElemTxt` [4][`SELNUM_STR_LEN`]
Storage for strings.

8.30.1 Detailed Description

Extended data for SelNum element.

8.30.2 Field Documentation

8.30.2.1 `char gslc_tsXSelNum::acElemTxt`[4][`SELNUM_STR_LEN`]

Storage for strings.

8.30.2.2 `gslc_tsElem gslc_tsXSelNum::asElem`[4]

Storage for sub-elements.

8.30.2.3 gslc_tsElemRef gslc_tsXSelNum::asElemRef[4]

Storage for sub-element references.

8.30.2.4 int16_t gslc_tsXSelNum::nCounter

Counter for demo purposes.

8.30.2.5 gslc_tsCollect gslc_tsXSelNum::sCollect

Collection management for sub-elements.

The documentation for this struct was generated from the following file:

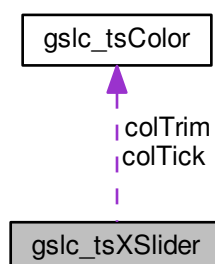
- [src/ele/XSelNum.h](#)

8.31 gslc_tsXSlider Struct Reference

Extended data for Slider element.

```
#include <XSlider.h>
```

Collaboration diagram for gslc_tsXSlider:



Data Fields

- bool [bVert](#)
Orientation: true if vertical, else horizontal.
- int16_t [nThumbSz](#)
Size of the thumb control.
- int16_t [nPosMin](#)
Minimum position value of the slider.
- int16_t [nPosMax](#)
Maximum position value of the slider.
- uint16_t [nTickDiv](#)
Style: number of tickmark divisions (0 for none)
- int16_t [nTickLen](#)
Style: length of tickmarks.
- [gslc_tsColor](#) [colTick](#)
Style: color of ticks.
- bool [bTrim](#)
Style: show a trim color.
- [gslc_tsColor](#) [colTrim](#)
Style: color of trim.
- int16_t [nPos](#)
Current position value of the slider.
- [GSLC_CB_XSLIDER_POS](#) [pfuncXPos](#)
Callback func ptr for position update.

8.31.1 Detailed Description

Extended data for Slider element.

8.31.2 Field Documentation

8.31.2.1 bool [gslc_tsXSlider::bTrim](#)

Style: show a trim color.

8.31.2.2 bool [gslc_tsXSlider::bVert](#)

Orientation: true if vertical, else horizontal.

8.31.2.3 [gslc_tsColor](#) [gslc_tsXSlider::colTick](#)

Style: color of ticks.

8.31.2.4 [gslc_tsColor](#) [gslc_tsXSlider::colTrim](#)

Style: color of trim.

Data Fields

- `int16_t nMin`
Minimum control value.
- `int16_t nMax`
Maximum control value.
- `int16_t nIncr`
Increment by value.
- `int16_t nCounter`
Current value.
- `GSLC_CB_INPUT pfuncXInput`
Callback func ptr for input ready.
- `gslc_tsElemRef * pElemRef`
Save our ElemRef for the callback.
- `gslc_tsCollect sCollect`
Collection management for sub-elements.
- `gslc_tsElemRef asElemRef [XSPINNER_COMP_CNT]`
Storage for sub-element references.
- `gslc_tsElem asElem [XSPINNER_COMP_CNT]`
Storage for sub-elements.
- `char acElemTxt [1][XSPINNER_STR_LEN]`
Storage for strings.
- `char acIncr [2]`
Increment character string.
- `char acDecr [2]`
Decrement character string.

8.32.1 Detailed Description

Extended data for Spinner element.

8.32.2 Field Documentation

8.32.2.1 `char gslc_tsXSpinner::acDecr[2]`

Decrement character string.

8.32.2.2 `char gslc_tsXSpinner::acElemTxt[1][XSPINNER_STR_LEN]`

Storage for strings.

8.32.2.3 `char gslc_tsXSpinner::acIncr[2]`

Increment character string.

8.32.2.4 `gslc_tsElem` `gslc_tsXSpinner::asElem[XSPINNER_COMP_CNT]`

Storage for sub-elements.

8.32.2.5 `gslc_tsElemRef` `gslc_tsXSpinner::asElemRef[XSPINNER_COMP_CNT]`

Storage for sub-element references.

8.32.2.6 `int16_t` `gslc_tsXSpinner::nCounter`

Current value.

8.32.2.7 `int16_t` `gslc_tsXSpinner::nIncr`

Increment by value.

8.32.2.8 `int16_t` `gslc_tsXSpinner::nMax`

Maximum control value.

8.32.2.9 `int16_t` `gslc_tsXSpinner::nMin`

Minimum control value.

8.32.2.10 `gslc_tsElemRef*` `gslc_tsXSpinner::pElemRef`

Save our ElemRef for the callback.

8.32.2.11 `GSLC_CB_INPUT` `gslc_tsXSpinner::pfuncXInput`

Callback func ptr for input ready.

8.32.2.12 `gslc_tsCollect` `gslc_tsXSpinner::sCollect`

Collection management for sub-elements.

The documentation for this struct was generated from the following file:

- `src/elem/XSpinner.h`

8.33 gslc_tsXTemplate Struct Reference

Callback function for slider feedback.

```
#include <XTemplate.h>
```

8.33.1 Detailed Description

Callback function for slider feedback.

Extended data for Slider element

The documentation for this struct was generated from the following file:

- [src/elem/XTemplate.h](#)

8.34 gslc_tsXTextbox Struct Reference

Extended data for Textbox element.

```
#include <XTextbox.h>
```

Data Fields

- `char * pBuf`
Ptr to the text buffer (circular buffer)
- `int8_t nMarginX`
Margin for text area within element rect (X)
- `int8_t nMarginY`
Margin for text area within element rect (Y)
- `bool bWrapEn`
Enable for line wrapping.
- `uint16_t nBufRows`
Number of rows in buffer.
- `uint16_t nBufCols`
Number of columns in buffer.
- `bool bScrollEn`
Enable for scrollbar.
- `uint16_t nScrollPos`
Current scrollbar position.
- `uint8_t nChSizeX`
Width of characters (pixels)
- `uint8_t nChSizeY`
Height of characters (pixels)
- `uint8_t nWndCols`
Window X size.
- `uint8_t nWndRows`

- Window Y size.*
- uint8_t [nCurPosX](#)
Cursor X position.
- uint8_t [nCurPosY](#)
Cursor Y position.
- uint8_t [nBufPosX](#)
Buffer X position.
- uint8_t [nBufPosY](#)
Buffer Y position.
- uint8_t [nWndRowStart](#)
First row of current window.
- int16_t [nRedrawRow](#)
Specific row to update in redraw (if not -1)

8.34.1 Detailed Description

Extended data for Textbox element.

8.34.2 Field Documentation

8.34.2.1 bool gslc_tsXTextbox::bScrollEn

Enable for scrollbar.

8.34.2.2 bool gslc_tsXTextbox::bWrapEn

Enable for line wrapping.

8.34.2.3 uint16_t gslc_tsXTextbox::nBufCols

Number of columns in buffer.

8.34.2.4 uint8_t gslc_tsXTextbox::nBufPosX

Buffer X position.

8.34.2.5 uint8_t gslc_tsXTextbox::nBufPosY

Buffer Y position.

8.34.2.6 uint16_t gslc_tsXTextbox::nBufRows

Number of rows in buffer.

8.34.2.7 `uint8_t gslc_tsXTextbox::nChSizeX`

Width of characters (pixels)

8.34.2.8 `uint8_t gslc_tsXTextbox::nChSizeY`

Height of characters (pixels)

8.34.2.9 `uint8_t gslc_tsXTextbox::nCurPosX`

Cursor X position.

8.34.2.10 `uint8_t gslc_tsXTextbox::nCurPosY`

Cursor Y position.

8.34.2.11 `int8_t gslc_tsXTextbox::nMarginX`

Margin for text area within element rect (X)

8.34.2.12 `int8_t gslc_tsXTextbox::nMarginY`

Margin for text area within element rect (Y)

8.34.2.13 `int16_t gslc_tsXTextbox::nRedrawRow`

Specific row to update in redraw (if not -1)

8.34.2.14 `uint16_t gslc_tsXTextbox::nScrollPos`

Current scrollbar position.

8.34.2.15 `uint8_t gslc_tsXTextbox::nWndCols`

Window X size.

8.34.2.16 `uint8_t gslc_tsXTextbox::nWndRows`

Window Y size.

8.34.2.17 uint8_t gslc_tsXTextbox::nWndRowStart

First row of current window.

8.34.2.18 char* gslc_tsXTextbox::pBuf

Ptr to the text buffer (circular buffer)

The documentation for this struct was generated from the following file:

- [src/elem/XTextbox.h](#)

8.35 THPoint Class Reference

```
#include <GUIslice_th.h>
```

Public Member Functions

- [THPoint](#) (void)
- [THPoint](#) (uint16_t x, uint16_t y, uint16_t z)
- bool [operator==](#) (THPoint)
- bool [operator!=](#) (THPoint)

Data Fields

- uint16_t x
- uint16_t y
- uint16_t z

8.35.1 Constructor & Destructor Documentation

8.35.1.1 THPoint::THPoint (void)

8.35.1.2 THPoint::THPoint (uint16_t x, uint16_t y, uint16_t z)

8.35.2 Member Function Documentation

8.35.2.1 bool THPoint::operator!= (THPoint p1)

8.35.2.2 bool THPoint::operator== (THPoint p1)

8.35.3 Field Documentation

8.35.3.1 uint16_t THPoint::x

8.35.3.2 uint16_t THPoint::y

8.35.3.3 uint16_t THPoint::z

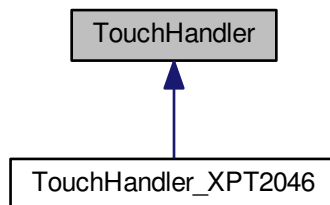
The documentation for this class was generated from the following files:

- [src/GUIslice_th.h](#)
- [src/GUIslice_th.cpp](#)

8.36 TouchHandler Class Reference

```
#include <GUIslice_th.h>
```

Inheritance diagram for TouchHandler:



Public Member Functions

- [TouchHandler](#) ()
- void [setSize](#) (uint16_t _disp_xSize, uint16_t _disp_ySize)
- void [setCalibration](#) (uint16_t ts_xMin, uint16_t ts_xMax, uint16_t ts_yMin, uint16_t ts_yMax)
- void [setSwapFlip](#) (bool _swapXY, bool _flipX, bool _flipY)
- [THPoint](#) [scale](#) ([THPoint](#) pIn)
- virtual void [begin](#) (void)
- virtual [THPoint](#) [getPoint](#) (void)

8.36.1 Constructor & Destructor Documentation

8.36.1.1 [TouchHandler::TouchHandler](#) () [inline]

8.36.2 Member Function Documentation

8.36.2.1 void [TouchHandler::begin](#) (void) [virtual]

Reimplemented in [TouchHandler_XPT2046](#).

8.36.2.2 [THPoint](#) [TouchHandler::getPoint](#) (void) [virtual]

Reimplemented in [TouchHandler_XPT2046](#).

8.36.2.3 `THPoint TouchHandler::scale (THPoint pln)`

8.36.2.4 `void TouchHandler::setCalibration (uint16_t ts_xMin, uint16_t ts_xMax, uint16_t ts_yMin, uint16_t ts_yMax)`

8.36.2.5 `void TouchHandler::setSize (uint16_t disp_xSize, uint16_t disp_ySize)`

8.36.2.6 `void TouchHandler::setSwapFlip (bool _swapXY, bool _flipX, bool _flipY)`

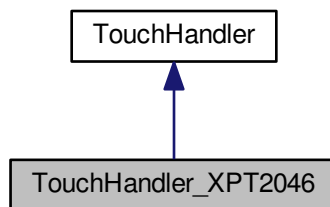
The documentation for this class was generated from the following files:

- [src/GUISlice_th.h](#)
- [src/GUISlice_th.cpp](#)

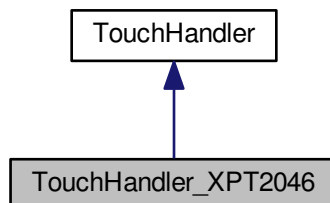
8.37 TouchHandler_XPT2046 Class Reference

```
#include <GUISlice_th_XPT2046.h>
```

Inheritance diagram for TouchHandler_XPT2046:



Collaboration diagram for TouchHandler_XPT2046:



Public Member Functions

- [TouchHandler_XPT2046](#) (SPIClass &[spi](#), uint8_t [spi_cs_pin](#))
- void [begin](#) (void)
- [THPoint](#) [getPoint](#) (void)

Data Fields

- SPIClass [spi](#)
- XPT2046_touch [touchDriver](#)

8.37.1 Constructor & Destructor Documentation

8.37.1.1 `TouchHandler_XPT2046::TouchHandler_XPT2046 (SPIClass & spi, uint8_t spi_cs_pin)` `[inline]`

8.37.2 Member Function Documentation

8.37.2.1 `void TouchHandler_XPT2046::begin (void)` `[inline],[virtual]`

Reimplemented from [TouchHandler](#).

8.37.2.2 `THPoint TouchHandler_XPT2046::getPoint (void)` `[inline],[virtual]`

Reimplemented from [TouchHandler](#).

8.37.3 Field Documentation

8.37.3.1 `SPIClass TouchHandler_XPT2046::spi`

8.37.3.2 `XPT2046_touch TouchHandler_XPT2046::touchDriver`

The documentation for this class was generated from the following file:

- [src/GUIslice_th_XPT2046.h](#)

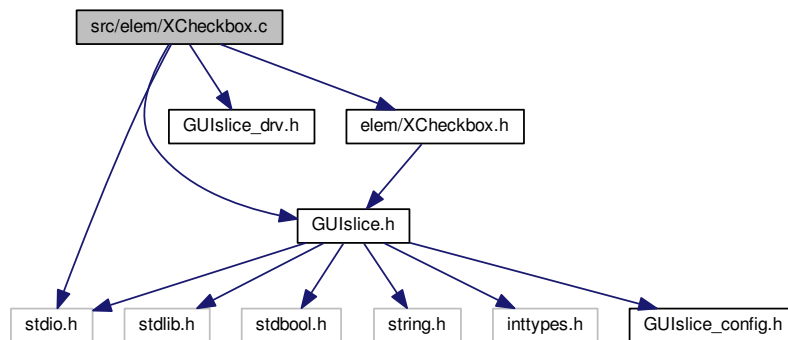
Chapter 9

File Documentation

9.1 README.md File Reference

9.2 src/elem/XCheckbox.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XCheckbox.h"
#include <stdio.h>
Include dependency graph for XCheckbox.c:
```



Functions

- `gslc_tsElemRef * gslc_ElemXCheckboxCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXCheckbox *pXData, gslc_tsRect rElem, bool bRadio, gslc_teXCheckboxStyle nStyle, gslc_tsColor col, bool bChecked)`
Create a Checkbox or Radio button Element.
- `bool gslc_ElemXCheckboxGetState (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
Get a Checkbox element's current state.
- `gslc_tsElemRef * gslc_ElemXCheckboxFindChecked (gslc_tsGui *pGui, int16_t nGroupId)`

Find the checkbox within a group that has been checked.

- void [gslc_ElemXCheckboxSetStateFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_XCH↵ECKBOX](#) pfuncCb)

Assign the state callback function for a checkbox/radio button.

- void [gslc_ElemXCheckboxSetStateHelp](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bChecked, bool bDoCb)
- void [gslc_ElemXCheckboxSetState](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bChecked)

Set a Checkbox element's current state.

- void [gslc_ElemXCheckboxToggleState](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Toggle a Checkbox element's current state.

- bool [gslc_ElemXCheckboxDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)

Draw a Checkbox element on the screen.

- bool [gslc_ElemXCheckboxTouch](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY)

Handle touch events to Checkbox element.

Variables

- const char [GSLC_PMEM ERRSTR_NULL](#) []
- const char [GSLC_PMEM ERRSTR_PXD_NULL](#) []

9.2.1 Function Documentation

- 9.2.1.1 [gslc_tsElemRef*](#) [gslc_ElemXCheckboxCreate](#) ([gslc_tsGui](#) * *pGui*, int16_t *nElemId*, int16_t *nPage*, [gslc_tsXCheckbox](#) * *pXData*, [gslc_tsRect](#) *rElem*, bool *bRadio*, [gslc_teXCheckboxStyle](#) *nStyle*, [gslc_tsColor](#) *colCheck*, bool *bChecked*)

Create a Checkbox or Radio button Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining checkbox size
in	<i>bRadio</i>	Radio-button functionality if true
in	<i>nStyle</i>	Drawing style for checkbox / radio button
in	<i>colCheck</i>	Color for inner fill when checked
in	<i>bChecked</i>	Default state

Returns

Pointer to Element reference or NULL if failure

- 9.2.1.2 [bool](#) [gslc_ElemXCheckboxDraw](#) (void * *pvGui*, void * *pvElemRef*, [gslc_teRedrawType](#) *eRedraw*)

Draw a Checkbox element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.2.1.3 `gslc_tsElemRef* gslc_ElemXCheckboxFindChecked (gslc_tsGui * pGui, int16_t nGroupId)`

Find the checkbox within a group that has been checked.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ GroupId</i>	Group ID to search

Returns

Element Ptr or NULL if none checked

9.2.1.4 `bool gslc_ElemXCheckboxGetState (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get a Checkbox element's current state.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Current state

9.2.1.5 `void gslc_ElemXCheckboxSetState (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bChecked)`

Set a Checkbox element's current state.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bChecked</i>	New state

Returns

none

9.2.1.6 void gslc_ElemXCheckboxSetStateFunc (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*,
GSLC_CB_XCHECKBOX *pfuncCb*)

Assign the state callback function for a checkbox/radio button.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>pfuncCb</i>	Function pointer to callback routine (or NULL for none)

Returns

none

9.2.1.7 void gslc_ElemXCheckboxSetStateHelp (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bChecked*, bool
bDoCb)

9.2.1.8 void gslc_ElemXCheckboxToggleState (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

Toggle a Checkbox element's current state.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

none

9.2.1.9 bool gslc_ElemXCheckboxTouch (void * *pvGui*, void * *pvElemRef*, gslc_teTouch *eTouch*, int16_t *nRelX*, int16_t
nRelY)

Handle touch events to Checkbox element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

Returns

true if success, false otherwise

9.2.2 Variable Documentation

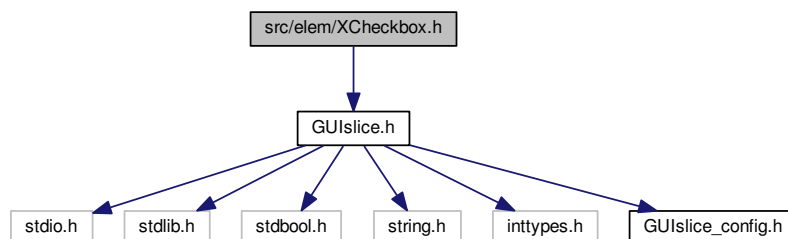
9.2.2.1 `const char ERRSTR_NULL`

9.2.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

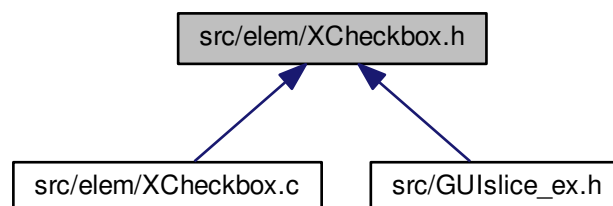
9.3 src/elem/XCheckbox.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XCheckbox.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXCheckbox](#)
Extended data for Checkbox element.

Macros

- #define [GSLC_TYPEX_CHECKBOX](#)
- #define [gslc_ElemXCheckboxCreate_P](#)(pGui, nElemId, nPage, nX, nY, nW, nH, colFill, bFillEn, nGroup, bRadio_, nStyle_, colCheck_, bChecked_)
Create a Checkbox or Radio button Element in Flash.

Typedefs

- typedef bool(* [GSLC_CB_XCHECKBOX](#)) (void *pvGui, void *pvElemRef, int16_t nSelId, bool bChecked)
Callback function for checkbox/radio element state change.

Enumerations

- enum [gslc_teXCheckboxStyle](#) { [GSLCX_CHECKBOX_STYLE_BOX](#), [GSLCX_CHECKBOX_STYLE_X](#), [GSLCX_CHECKBOX_STYLE_ROUND](#) }
Checkbox drawing style.

Functions

- [gslc_tsElemRef * gslc_ElemXCheckboxCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXCheckbox](#) *pXData, [gslc_tsRect](#) rElem, bool bRadio, [gslc_teXCheckboxStyle](#) nStyle, [gslc_tsColor](#) colCheck, bool bChecked)
Create a Checkbox or Radio button Element.
- bool [gslc_ElemXCheckboxGetState](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get a Checkbox element's current state.
- void [gslc_ElemXCheckboxSetState](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bChecked)
Set a Checkbox element's current state.
- [gslc_tsElemRef * gslc_ElemXCheckboxFindChecked](#) ([gslc_tsGui](#) *pGui, int16_t nGroupId)
Find the checkbox within a group that has been checked.
- void [gslc_ElemXCheckboxToggleState](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Toggle a Checkbox element's current state.
- void [gslc_ElemXCheckboxSetStateFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_XCHECKBOX](#) pfuncCb)
Assign the state callback function for a checkbox/radio button.
- bool [gslc_ElemXCheckboxDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a Checkbox element on the screen.
- bool [gslc_ElemXCheckboxTouch](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY)
Handle touch events to Checkbox element.

9.3.1 Macro Definition Documentation

- 9.3.1.1 #define [gslc_ElemXCheckboxCreate_P](#)(*pGui*, *nElemId*, *nPage*, *nX*, *nY*, *nW*, *nH*, *colFill*, *bFillEn*, *nGroup*, *bRadio*_, *nStyle*_, *colCheck*_, *bChecked*_)

Create a Checkbox or Radio button Element in Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>colFill</i>	Color for the control background fill
in	<i>bFillEn</i>	True if background filled, false otherwise (recommend True)
in	<i>nGroup</i>	Group ID that radio buttons belong to (else GSLC_GROUP_NONE)
in	<i>bRadio_</i>	Radio-button functionality if true
in	<i>nStyle_</i>	Drawing style for checkbox / radio button
in	<i>col↔ Check_</i>	Color for inner fill when checked
in	<i>b↔ Checked↔ _</i>	Default state

Returns

none

9.3.1.2 #define GSLC_TYPEX_CHECKBOX

9.3.2 Typedef Documentation

9.3.2.1 typedef bool(* GSLC_CB_XCHECKBOX) (void *pvGui, void *pvElemRef, int16_t nSelId, bool bChecked)

Callback function for checkbox/radio element state change.

- nSelId: Selected element's ID or GSLC_ID_NONE
- bChecked: Element was selected if true, false otherwise

9.3.3 Enumeration Type Documentation

9.3.3.1 enum gslc_teXCheckboxStyle

Checkbox drawing style.

Enumerator

GSLCX_CHECKBOX_STYLE_BOX Inner box.
GSLCX_CHECKBOX_STYLE_X Crossed.
GSLCX_CHECKBOX_STYLE_ROUND Circular.

9.3.4 Function Documentation

9.3.4.1 `gslc_tsElemRef* gslc_ElemXCheckboxCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage,
gslc_tsXCheckbox * pXData, gslc_tsRect rElem, bool bRadio, gslc_teXCheckboxStyle nStyle,
gslc_tsColor colCheck, bool bChecked)`

Create a Checkbox or Radio button Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining checkbox size
in	<i>bRadio</i>	Radio-button functionality if true
in	<i>nStyle</i>	Drawing style for checkbox / radio button
in	<i>colCheck</i>	Color for inner fill when checked
in	<i>bChecked</i>	Default state

Returns

Pointer to Element reference or NULL if failure

9.3.4.2 `bool gslc_ElemXCheckboxDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a Checkbox element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.3.4.3 `gslc_tsElemRef* gslc_ElemXCheckboxFindChecked (gslc_tsGui * pGui, int16_t nGroupId)`

Find the checkbox within a group that has been checked.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ GroupId</i>	Group ID to search

Returns

Element Ptr or NULL if none checked

9.3.4.4 bool gslc_ElemXCheckboxGetState (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

Get a Checkbox element's current state.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Current state

9.3.4.5 void gslc_ElemXCheckboxSetState (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bChecked*)

Set a Checkbox element's current state.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bChecked</i>	New state

Returns

none

9.3.4.6 void gslc_ElemXCheckboxSetStateFunc (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, GSLC_CB_XCHECKBOX *pfuncCb*)

Assign the state callback function for a checkbox/radio button.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>pfuncCb</i>	Function pointer to callback routine (or NULL for none)

Returns

none

9.3.4.7 void gslc_ElemXCheckboxToggleState (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

Toggle a Checkbox element's current state.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

none

9.3.4.8 `bool gslc_ElemXCheckboxTouch (void * pGui, void * pElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch events to Checkbox element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

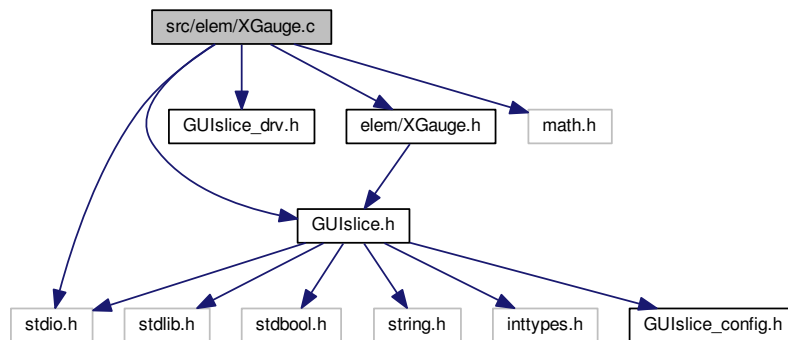
Returns

true if success, false otherwise

9.4 `src/elem/XGauge.c` File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XGauge.h"
#include <stdio.h>
#include <math.h>
```

Include dependency graph for XGauge.c:



Functions

- [gslc_tsElemRef](#) * [gslc_ElemXGaugeCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXGauge](#) *pXData, [gslc_tsRect](#) rElem, int16_t nMin, int16_t nMax, int16_t nVal, [gslc_tsColor](#) colGauge, bool bVert)
Create a Gauge Element.
- void [gslc_ElemXGaugeSetStyle](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsXGaugeStyle](#) nStyle)
Configure the style of a Gauge element.
- void [gslc_ElemXGaugeSetIndicator](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)
Configure the appearance of the Gauge indicator.
- void [gslc_ElemXGaugeSetTicks](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colTick, uint16_t nTickCnt, uint16_t nTickLen)
Configure the appearance of the Gauge ticks.
- void [gslc_ElemXGaugeUpdate](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)
Update a Gauge element's current value.
- void [gslc_ElemXGaugeSetFlip](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bFlip)
Set a Gauge element's fill direction.
- bool [gslc_ElemXGaugeDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a gauge element on the screen.
- bool [gslc_ElemXGaugeDrawProgressBar](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) eRedraw)
Helper function to draw a gauge with style: progress bar.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.4.1 Function Documentation

9.4.1.1 `gslc_tsElemRef* gslc_ElemXGaugeCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXGauge * pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)`

Create a Gauge Element.

- Draws a gauge element that represents a proportion (nVal) between nMin and nMax.
- Support gauge sub-types:
 - `GSLC_TYPEX_GAUGE_PROG_BAR`: Horizontal or vertical box with filled region
 - `GSLC_TYPEX_GAUGE_RADIAL`: Radial / compass indicator
- Default appearance is a horizontal progress bar, but can be changed with [gslc_ElemXGaugeSetStyle\(\)](#)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or <code>GSLC_ID_AUTO</code> to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for nVal comparison
in	<i>nMax</i>	Maximum value of gauge for nVal comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator
in	<i>bVert</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element reference or NULL if failure

9.4.1.2 `bool gslc_ElemXGaugeDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a gauge element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.4.1.3 `bool gslc_ElemXGaugeDrawProgressBar (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef,
gslc_teRedrawType eRedraw)`

Helper function to draw a gauge with style: progress bar.

- Called from [gslc_ElemXGaugeDraw\(\)](#)

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pElemRef</i>	Ptr to Element reference
in	<i>eRedraw</i>	Redraw status

Returns

true if success, false otherwise

9.4.1.4 `void gslc_ElemXGaugeSetFlip (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bFlip)`

Set a Gauge element's fill direction.

- Setting *bFlip* reverses the default fill direction
- Default fill direction for horizontal gauges: left-to-right
- Default fill direction for vertical gauges: bottom-to-top

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bFlip</i>	If set, reverse direction of fill from default

Returns

none

9.4.1.5 `void gslc_ElemXGaugeSetIndicator (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tsColor colGauge,
uint16_t nIndicLen, uint16_t nIndicTip, bool blndicFill)`

Configure the appearance of the Gauge indicator.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colGauge</i>	Color of the indicator
in	<i>nIndicLen</i>	Length of the indicator
in	<i>nIndicTip</i>	Size of the indicator tip
in	<i>bIndicFill</i>	Fill in the indicator if true

Returns

none

9.4.1.6 void gslc_ElemXGaugeSetStyle (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_teXGaugeStyle *nType*)

Configure the style of a Gauge element.

- This function is used to select between one of several gauge types (eg. progress bar, radial dial, etc.)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nType</i>	Gauge style enumeration

Returns

none

9.4.1.7 void gslc_ElemXGaugeSetTicks (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colTick*, uint16_t *nTickCnt*, uint16_t *nTickLen*)

Configure the appearance of the Gauge ticks.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colTick</i>	Color of the gauge ticks
in	<i>nTickCnt</i>	Number of ticks to draw around / along gauge
in	<i>nTickLen</i>	Length of the tick marks to draw

Returns

none

9.4.1.8 void gslc_ElemXGaugeUpdate (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nVal*)

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

Returns

none

9.4.2 Variable Documentation

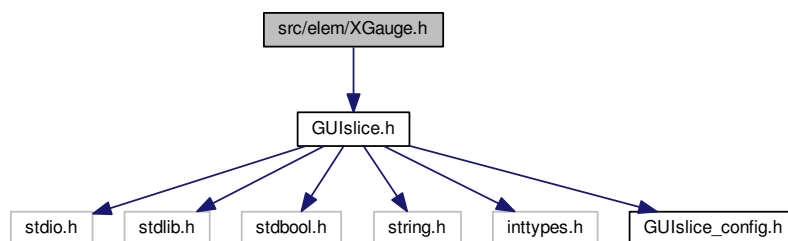
9.4.2.1 const char GSLC_PMEM_ERRSTR_NULL[]

9.4.2.2 const char GSLC_PMEM_ERRSTR_PXD_NULL[]

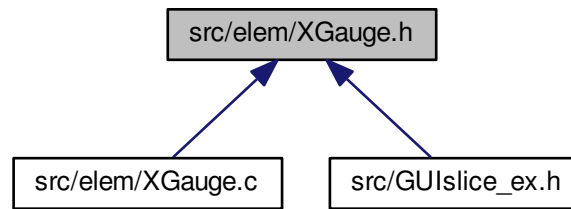
9.5 src/elem/XGauge.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XGauge.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXGauge](#)
Extended data for Gauge element.

Macros

- #define [GSLC_TYPEX_GAUGE](#)
- #define [gslc_ElemXGaugeCreate_P](#)(pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, col←
Frame_, colFill_, colGauge_, bVert_)
Create a Gauge Element in Flash.

Enumerations

- enum [gslc_teXGaugeStyle](#) { [GSLCX_GAUGE_STYLE_PROG_BAR](#), [GSLCX_GAUGE_STYLE_RADIAL](#),
[GSLCX_GAUGE_STYLE_RAMP](#) }
Gauge drawing style.

Functions

- [gslc_tsElemRef](#) * [gslc_ElemXGaugeCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsX←
Gauge](#) *pXData, [gslc_tsRect](#) rElem, int16_t nMin, int16_t nMax, int16_t nVal, [gslc_tsColor](#) colGauge, bool
bVert)
Create a Gauge Element.
- void [gslc_ElemXGaugeSetStyle](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teXGaugeStyle](#) nType)
Configure the style of a Gauge element.
- void [gslc_ElemXGaugeSetIndicator](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colGauge,
uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)
Configure the appearance of the Gauge indicator.
- void [gslc_ElemXGaugeSetTicks](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colTick,
uint16_t nTickCnt, uint16_t nTickLen)
Configure the appearance of the Gauge ticks.
- void [gslc_ElemXGaugeUpdate](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)
Update a Gauge element's current value.

- void [gslc_ElemXGaugeSetFlip](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bFlip)
Set a Gauge element's fill direction.
- bool [gslc_ElemXGaugeDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a gauge element on the screen.
- bool [gslc_ElemXGaugeDrawProgressBar](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) eRedraw)
Helper function to draw a gauge with style: progress bar.

9.5.1 Macro Definition Documentation

9.5.1.1 `#define gslc_ElemXGaugeCreate_P(pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, colFrame_, colFill_, colGauge_, bVert_)`

Create a Gauge Element in Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>nMin_</i>	Minimum value of gauge for nVal comparison
in	<i>nMax_</i>	Maximum value of gauge for nVal comparison
in	<i>nVal_</i>	Starting value of gauge
in	<i>colFrame_</i>	Color for the gauge frame
in	<i>colFill_</i>	Color for the gauge background fill
in	<i>colGauge_</i>	Color for the gauge indicator
in	<i>bVert_</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

none

9.5.1.2 `#define GSLC_TYPEX_GAUGE`

9.5.2 Enumeration Type Documentation

9.5.2.1 `enum gslc_teXGaugeStyle`

Gauge drawing style.

Enumerator

GSLCX_GAUGE_STYLE_PROG_BAR Progress bar.
GSLCX_GAUGE_STYLE_RADIAL Radial indicator.
GSLCX_GAUGE_STYLE_RAMP Ramp indicator.

9.5.3 Function Documentation

9.5.3.1 `gslc_tsElemRef* gslc_ElemXGaugeCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXGauge * pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)`

Create a Gauge Element.

- Draws a gauge element that represents a proportion (nVal) between nMin and nMax.
- Support gauge sub-types:
 - `GSLC_TYPEX_GAUGE_PROG_BAR`: Horizontal or vertical box with filled region
 - `GSLC_TYPEX_GAUGE_RADIAL`: Radial / compass indicator
- Default appearance is a horizontal progress bar, but can be changed with [gslc_ElemXGaugeSetStyle\(\)](#)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or <code>GSLC_ID_AUTO</code> to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for nVal comparison
in	<i>nMax</i>	Maximum value of gauge for nVal comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator
in	<i>bVert</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element reference or NULL if failure

9.5.3.2 `bool gslc_ElemXGaugeDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a gauge element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.5.3.3 `bool gslc_ElemXGaugeDrawProgressBar (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef,
gslc_teRedrawType eRedraw)`

Helper function to draw a gauge with style: progress bar.

- Called from [gslc_ElemXGaugeDraw\(\)](#)

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pElemRef</i>	Ptr to Element reference
in	<i>eRedraw</i>	Redraw status

Returns

true if success, false otherwise

9.5.3.4 `void gslc_ElemXGaugeSetFlip (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bFlip)`

Set a Gauge element's fill direction.

- Setting *bFlip* reverses the default fill direction
- Default fill direction for horizontal gauges: left-to-right
- Default fill direction for vertical gauges: bottom-to-top

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bFlip</i>	If set, reverse direction of fill from default

Returns

none

9.5.3.5 `void gslc_ElemXGaugeSetIndicator (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tsColor colGauge,
uint16_t nIndicLen, uint16_t nIndicTip, bool blndicFill)`

Configure the appearance of the Gauge indicator.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colGauge</i>	Color of the indicator
in	<i>nIndicLen</i>	Length of the indicator
in	<i>nIndicTip</i>	Size of the indicator tip
in	<i>bIndicFill</i>	Fill in the indicator if true

Returns

none

9.5.3.6 void gslc_ElemXGaugeSetStyle (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_teXGaugeStyle *nType*)

Configure the style of a Gauge element.

- This function is used to select between one of several gauge types (eg. progress bar, radial dial, etc.)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nType</i>	Gauge style enumeration

Returns

none

9.5.3.7 void gslc_ElemXGaugeSetTicks (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colTick*, uint16_t *nTickCnt*, uint16_t *nTickLen*)

Configure the appearance of the Gauge ticks.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colTick</i>	Color of the gauge ticks
in	<i>nTickCnt</i>	Number of ticks to draw around / along gauge
in	<i>nTickLen</i>	Length of the tick marks to draw

Returns

none

9.5.3.8 void gslc_ElemXGaugeUpdate (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nVal*)

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

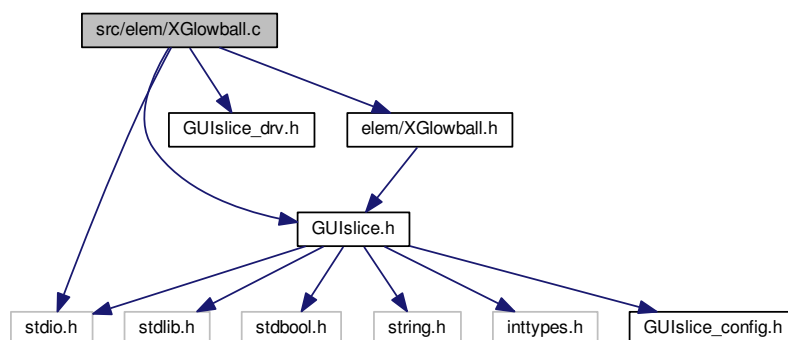
Returns

none

9.6 src/elem/XGlowball.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XGlowball.h"
#include <stdio.h>
```

Include dependency graph for XGlowball.c:



Functions

- [gslc_tsElemRef * gslc_ElemXGlowballCreate](#) ([gslc_tsGui](#) **pGui*, int16_t *nElemId*, int16_t *nPage*, [gslc_tsXGlowball](#) **pXData*, int16_t *nMidX*, int16_t *nMidY*, [gslc_tsXGlowballRing](#) **pRings*, uint8_t *nNumRings*)

Create a XGlowball element.

- void [drawXGlowballArc](#) ([gslc_tsGui](#) *pGui, [gslc_tsXGlowball](#) *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, [gslc_tsColor](#) cArc, uint16_t nAngStart, uint16_t nAngEnd)
- void [drawXGlowballRing](#) ([gslc_tsGui](#) *pGui, [gslc_tsXGlowball](#) *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nVal, uint16_t nAngStart, uint16_t nAngEnd, bool bErase)
- void [drawXGlowball](#) ([gslc_tsGui](#) *pGui, [gslc_tsXGlowball](#) *pGlowball, int16_t nMidX, int16_t nMidY, int16_t nVal, uint16_t nAngStart, uint16_t nAngEnd)
- void [gslc_ElemXGlowballSetVal](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)
- void [gslc_ElemXGlowballSetAngles](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nAngStart, int16_t nAngEnd)
- void [gslc_ElemXGlowballSetQuality](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint16_t nQuality)
- void [gslc_ElemXGlowballSetColorBack](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colBg)
- bool [gslc_ElemXGlowballDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)

Draw the XGlowball element on the screen.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.6.1 Function Documentation

9.6.1.1 void [drawXGlowball](#) ([gslc_tsGui](#) * *pGui*, [gslc_tsXGlowball](#) * *pGlowball*, int16_t *nMidX*, int16_t *nMidY*, int16_t *nVal*, uint16_t *nAngStart*, uint16_t *nAngEnd*)

9.6.1.2 void [drawXGlowballArc](#) ([gslc_tsGui](#) * *pGui*, [gslc_tsXGlowball](#) * *pGlowball*, int16_t *nMidX*, int16_t *nMidY*, int16_t *nRad1*, int16_t *nRad2*, [gslc_tsColor](#) *cArc*, uint16_t *nAngStart*, uint16_t *nAngEnd*)

9.6.1.3 void [drawXGlowballRing](#) ([gslc_tsGui](#) * *pGui*, [gslc_tsXGlowball](#) * *pGlowball*, int16_t *nMidX*, int16_t *nMidY*, int16_t *nVal*, uint16_t *nAngStart*, uint16_t *nAngEnd*, bool *bErase*)

9.6.1.4 [gslc_tsElemRef](#)* [gslc_ElemXGlowballCreate](#) ([gslc_tsGui](#) * *pGui*, int16_t *nElemId*, int16_t *nPage*, [gslc_tsXGlowball](#) * *pXData*, int16_t *nMidX*, int16_t *nMidY*, [gslc_tsXGlowballRing](#) * *pRings*, uint8_t *nNumRings*)

Create a XGlowball element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nMidX</i>	Center X coordinate
in	<i>nMidY</i>	Center Y coordinate
in	<i>pRings</i>	Pointer to tsXGlowballRing structure array defining appearance
in	<i>nNumRings</i>	Number of rings in pRings array

Returns

Pointer to Element reference or NULL if failure

9.6.1.5 `bool gslc_ElemXGlowballDraw (void * pvGui, void * pElemRef, gslc_teRedrawType eRedraw)`

Draw the XGlowball element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pElemRef</i>	Void ptr to Element (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.6.1.6 `void gslc_ElemXGlowballSetAngles (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nAngStart, int16_t nAngEnd)`

9.6.1.7 `void gslc_ElemXGlowballSetColorBack (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tsColor colBg)`

9.6.1.8 `void gslc_ElemXGlowballSetQuality (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, uint16_t nQuality)`

9.6.1.9 `void gslc_ElemXGlowballSetVal (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nVal)`

9.6.2 Variable Documentation

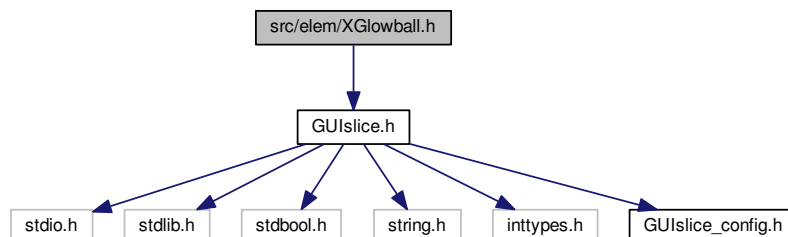
9.6.2.1 `const char GSLC_PMEM_ERRSTR_NULL[]`

9.6.2.2 `const char GSLC_PMEM_ERRSTR_PXD_NULL[]`

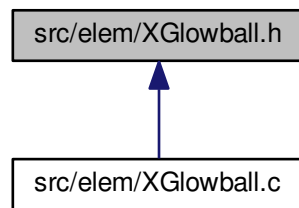
9.7 src/elem/XGlowball.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XGlowball.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXGlowballRing](#)
- struct [gslc_tsXGlowball](#)

Extended data for Slider element.

Macros

- `#define` [GSLC_TYPEX_GLOW](#)

Functions

- [gslc_tsElemRef * gslc_ElemXGlowballCreate](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nElemId, [int16_t](#) nPage, [gslc_tsXGlowball](#) *pXData, [int16_t](#) nMidX, [int16_t](#) nMidY, [gslc_tsXGlowballRing](#) *pRings, [uint8_t](#) nNumRings)
- Create a XGlowball element.*
- [bool gslc_ElemXGlowballDraw](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teRedrawType](#) eRedraw)
- Draw the XGlowball element on the screen.*
- [void drawXGlowballArc](#) ([gslc_tsGui](#) *pGui, [gslc_tsXGlowball](#) *pGlowball, [int16_t](#) nMidX, [int16_t](#) nMidY, [int16_t](#) nRad1, [int16_t](#) nRad2, [gslc_tsColor](#) cArc, [uint16_t](#) nAngStart, [uint16_t](#) nAngEnd)
- [void drawXGlowballRing](#) ([gslc_tsGui](#) *pGui, [gslc_tsXGlowball](#) *pGlowball, [int16_t](#) nMidX, [int16_t](#) nMidY, [int16_t](#) nVal, [uint16_t](#) nAngStart, [uint16_t](#) nAngEnd, [bool](#) bErase)
- [void drawXGlowball](#) ([gslc_tsGui](#) *pGui, [gslc_tsXGlowball](#) *pGlowball, [int16_t](#) nMidX, [int16_t](#) nMidY, [int16_t](#) nVal, [uint16_t](#) nAngStart, [uint16_t](#) nAngEnd)
- [void gslc_ElemXGlowballSetAngles](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [int16_t](#) nAngStart, [int16_t](#) nAngEnd)
- [void gslc_ElemXGlowballSetVal](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [int16_t](#) nVal)
- [void gslc_ElemXGlowballSetQuality](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [uint16_t](#) nQuality)
- [void gslc_ElemXGlowballSetColorBack](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colBg)

9.7.1 Macro Definition Documentation

9.7.1.1 #define GSLC_TYPEX_GLOW

9.7.2 Function Documentation

9.7.2.1 void drawXGlowball (gslc_tsGui * *pGui*, gslc_tsXGlowball * *pGlowball*, int16_t *nMidX*, int16_t *nMidY*, int16_t *nVal*, uint16_t *nAngStart*, uint16_t *nAngEnd*)

9.7.2.2 void drawXGlowballArc (gslc_tsGui * *pGui*, gslc_tsXGlowball * *pGlowball*, int16_t *nMidX*, int16_t *nMidY*, int16_t *nRad1*, int16_t *nRad2*, gslc_tsColor *cArc*, uint16_t *nAngStart*, uint16_t *nAngEnd*)

9.7.2.3 void drawXGlowballRing (gslc_tsGui * *pGui*, gslc_tsXGlowball * *pGlowball*, int16_t *nMidX*, int16_t *nMidY*, int16_t *nVal*, uint16_t *nAngStart*, uint16_t *nAngEnd*, bool *bErase*)

9.7.2.4 gslc_tsElemRef* gslc_ElemXGlowballCreate (gslc_tsGui * *pGui*, int16_t *nElemId*, int16_t *nPage*, gslc_tsXGlowball * *pXData*, int16_t *nMidX*, int16_t *nMidY*, gslc_tsXGlowballRing * *pRings*, uint8_t *nNumRings*)

Create a XGlowball element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nMidX</i>	Center X coordinate
in	<i>nMidY</i>	Center Y coordinate
in	<i>pRings</i>	Pointer to tsXGlowballRing structure array defining appearance
in	<i>nNumRings</i>	Number of rings in pRings array

Returns

Pointer to Element reference or NULL if failure

9.7.2.5 bool gslc_ElemXGlowballDraw (void * *pvGui*, void * *pvElemRef*, gslc_teRedrawType *eRedraw*)

Draw the XGlowball element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.7.2.6 void `gslc_ElemXGlowballSetAngles` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `int16_t` *nAngStart*, `int16_t` *nAngEnd*)

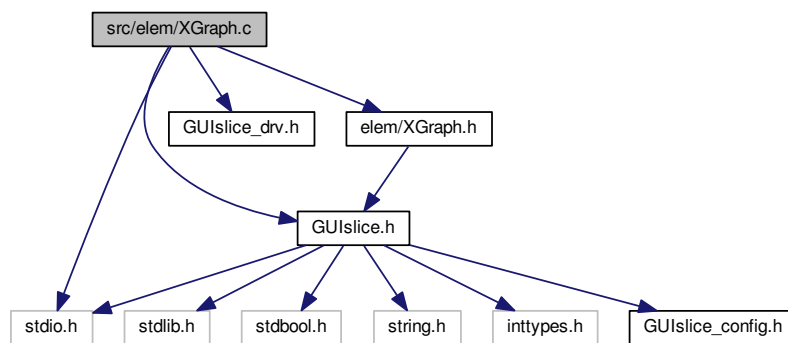
9.7.2.7 void `gslc_ElemXGlowballSetColorBack` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `gslc_tsColor` *colBg*)

9.7.2.8 void `gslc_ElemXGlowballSetQuality` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `uint16_t` *nQuality*)

9.7.2.9 void `gslc_ElemXGlowballSetVal` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `int16_t` *nVal*)

9.8 src/elem/XGraph.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XGraph.h"
#include <stdio.h>
Include dependency graph for XGraph.c:
```



Functions

- `gslc_tsElemRef` * `gslc_ElemXGraphCreate` (`gslc_tsGui` * *pGui*, `int16_t` *nElemId*, `int16_t` *nPage*, `gslc_tsXGraph` * *pXData*, `gslc_tsRect` *rElem*, `int16_t` *nFontId*, `int16_t` * *pBuf*, `uint16_t` *nBufMax*, `gslc_tsColor` *colGraph*)
Create a Graph Element.
- void `gslc_ElemXGraphSetStyle` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `gslc_teXGraphStyle` *eStyle*, `uint8_t` *nMargin*)
Set the graph's additional drawing characteristics.
- void `gslc_ElemXGraphSetRange` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `int16_t` *nYMin*, `int16_t` *nYMax*)
Set the graph's drawing range.
- void `gslc_ElemXGraphScrollSet` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `uint8_t` *nScrollPos*, `uint8_t` *nScrollMax*)
Set the graph scroll position (nScrollPos) as a fraction of nScrollMax.
- void `gslc_ElemXGraphAdd` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, `int16_t` *nVal*)
Add a value to the graph at the latest position.
- bool `gslc_ElemXGraphDraw` (void * *pGui*, void * *pElemRef*, `gslc_teRedrawType` *eRedraw*)
Draw a Graph element on the screen.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.8.1 Function Documentation

9.8.1.1 void [gslc_ElemXGraphAdd](#) ([gslc_tsGui](#) * *pGui*, [gslc_tsElemRef](#) * *pElemRef*, int16_t *nVal*)

Add a value to the graph at the latest position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	Data value to add

Returns

none

9.8.1.2 [gslc_tsElemRef](#)* [gslc_ElemXGraphCreate](#) ([gslc_tsGui](#) * *pGui*, int16_t *nElemId*, int16_t *nPage*, [gslc_tsXGraph](#) * *pXData*, [gslc_tsRect](#) *rElem*, int16_t *nFontId*, int16_t * *pBuf*, uint16_t *nBufRows*, [gslc_tsColor](#) *colGraph*)

Create a Graph Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining checkbox size
in	<i>nFontId</i>	Font ID to use for graph area
in	<i>pBuf</i>	Ptr to data buffer (already allocated) with size (nBufMax) int16_t
in	<i>nBufRows</i>	Maximum number of points in buffer
in	<i>colGraph</i>	Color of the graph

Returns

Pointer to Element reference or NULL if failure

9.8.1.3 bool [gslc_ElemXGraphDraw](#) (void * *pvGui*, void * *pvElemRef*, [gslc_teRedrawType](#) *eRedraw*)

Draw a Graph element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.8.1.4 void `gslc_ElemXGraphScrollSet (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)`

Set the graph scroll position (*nScrollPos*) as a fraction of *nScrollMax*.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nScrollPos</i>	New scroll position
in	<i>nScrollMax</i>	Maximum scroll position

Returns

none

9.8.1.5 void `gslc_ElemXGraphSetRange (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nYMin, int16_t nYMax)`

Set the graph's drawing range.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nYMin</i>	Minimum Y value to draw
in	<i>nYMax</i>	Maximum Y value to draw

Returns

none

9.8.1.6 void `gslc_ElemXGraphSetStyle (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tXGraphStyle eStyle, uint8_t nMargin)`

Set the graph's additional drawing characteristics.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>eStyle</i>	Drawing style for the graph
in	<i>nMargin</i>	Margin to provide around graph area inside frame

Returns

none

9.8.2 Variable Documentation

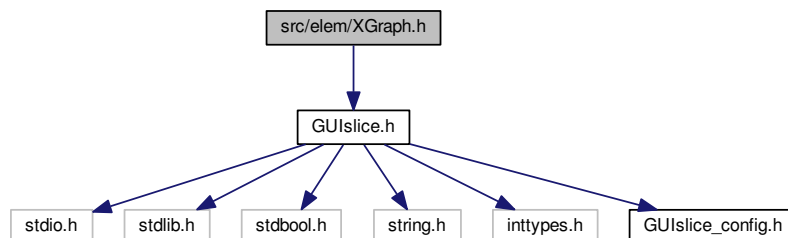
9.8.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.8.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

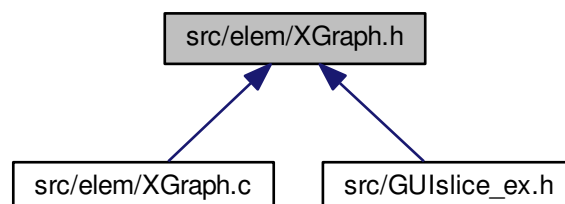
9.9 `src/elem/XGraph.h` File Reference

```
#include "GUIslice.h"
```

Include dependency graph for `XGraph.h`:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXGraph](#)
Extended data for Graph element.

Macros

- #define [GSLC_TYPEX_GRAPH](#)

Enumerations

- enum [gslc_teXGraphStyle](#) { [GSLCX_GRAPH_STYLE_DOT](#), [GSLCX_GRAPH_STYLE_LINE](#), [GSLCX_GRAPH_STYLE_FILL](#) }
Gauge drawing style.

Functions

- [gslc_tsElemRef](#) * [gslc_ElemXGraphCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXGraph](#) *pXData, [gslc_tsRect](#) rElem, int16_t nFontId, int16_t *pBuf, uint16_t nBufRows, [gslc_tsColor](#) col, [gslc_tsXGraph](#) *pGraph)
Create a Graph Element.
- void [gslc_ElemXGraphSetStyle](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teXGraphStyle](#) eStyle, uint8_t nMargin)
Set the graph's additional drawing characteristics.
- void [gslc_ElemXGraphSetRange](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nYMin, int16_t nYMax)
Set the graph's drawing range.
- bool [gslc_ElemXGraphDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a Graph element on the screen.
- void [gslc_ElemXGraphAdd](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)
Add a value to the graph at the latest position.
- void [gslc_ElemXGraphScrollSet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)
Set the graph scroll position (nScrollPos) as a fraction of nScrollMax.

9.9.1 Macro Definition Documentation

9.9.1.1 #define GSLC_TYPEX_GRAPH

9.9.2 Enumeration Type Documentation

9.9.2.1 enum [gslc_teXGraphStyle](#)

Gauge drawing style.

Enumerator

[GSLCX_GRAPH_STYLE_DOT](#) Dot.
[GSLCX_GRAPH_STYLE_LINE](#) Line.
[GSLCX_GRAPH_STYLE_FILL](#) Filled.

9.9.3 Function Documentation

9.9.3.1 void [gslc_ElemXGraphAdd](#) ([gslc_tsGui](#) * pGui, [gslc_tsElemRef](#) * pElemRef, int16_t nVal)

Add a value to the graph at the latest position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	Data value to add

Returns

none

9.9.3.2 `gslc_tsElemRef* gslc_ElemXGraphCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXGraph * pXData, gslc_tsRect rElem, int16_t nFontId, int16_t * pBuf, uint16_t nBufRows, gslc_tsColor colGraph)`

Create a Graph Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining checkbox size
in	<i>nFontId</i>	Font ID to use for graph area
in	<i>pBuf</i>	Ptr to data buffer (already allocated) with size (nBufMax) int16_t
in	<i>nBufRows</i>	Maximum number of points in buffer
in	<i>colGraph</i>	Color of the graph

Returns

Pointer to Element reference or NULL if failure

9.9.3.3 `bool gslc_ElemXGraphDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a Graph element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.9.3.4 void gslc_ElemXGraphScrollSet (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, uint8_t *nScrollPos*, uint8_t *nScrollMax*)

Set the graph scroll position (*nScrollPos*) as a fraction of *nScrollMax*.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nScrollPos</i>	New scroll position
in	<i>nScrollMax</i>	Maximum scroll position

Returns

none

9.9.3.5 void gslc_ElemXGraphSetRange (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nYMin*, int16_t *nYMax*)

Set the graph's drawing range.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nYMin</i>	Minimum Y value to draw
in	<i>nYMax</i>	Maximum Y value to draw

Returns

none

9.9.3.6 void gslc_ElemXGraphSetStyle (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tXGraphStyle *eStyle*, uint8_t *nMargin*)

Set the graph's additional drawing characteristics.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>eStyle</i>	Drawing style for the graph
in	<i>nMargin</i>	Margin to provide around graph area inside frame

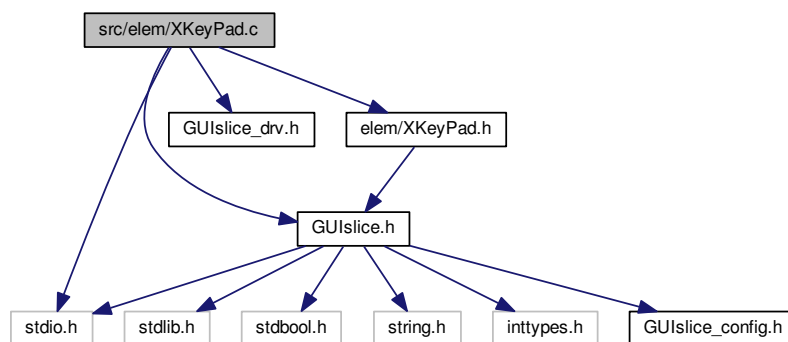
Returns

none

9.10 src/elem/XKeyPad.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XKeyPad.h"
#include <stdio.h>
```

Include dependency graph for XKeyPad.c:



Functions

- void [XKeyPadAddKeyElem](#) ([gslc_tsGui](#) *pGui, [gslc_tsXKeyPad](#) *pXData, int16_t nKeyId, bool bTxtField, int16_t nRow, int16_t nCol, int8_t nRowSpan, int8_t nColSpan, [gslc_tsColor](#) cColFill, [gslc_tsColor](#) cColGlow, bool bVisible)
Add a key to the KeyPad control.
- [gslc_tsElemRef](#) * [gslc_ElemXKeyPadCreateBase](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXKeyPad](#) *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, [gslc_tsXKeyPadCfg](#) *pConfig, [XKEYPAD_CREATE](#) pfuncCreate, [XKEYPAD_LOOKUP](#) pfuncLookup)
Create a KeyPad Element.
- void [gslc_ElemXKeyPadValSet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, const char *pStrBuf)
Set the current value for the editable text field.
- void [gslc_ElemXKeyPadTargetIdSet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nTargetId)
Set target element ID for KeyPad return value.
- int16_t [gslc_ElemXKeyPadDataTargetIdGet](#) ([gslc_tsGui](#) *pGui, void *pvData)
Fetch the element target ID associated with this KeyPad.
- char * [gslc_ElemXKeyPadDataValGet](#) ([gslc_tsGui](#) *pGui, void *pvData)
Fetch the edited value string from the KeyPad.
- bool [gslc_ElemXKeyPadValGet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, char *pStrBuf, uint8_t nStrBufLen)
Set the current output string buffer associated with NumericInput element.
- bool [gslc_ElemXKeyPadDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a KeyPad element on the screen.

- void [gslc_ElemXKeyPadValSetCb](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_INPUT](#) pfuncCb)
Set the callback function associated with the KeyPad.
- void [gslc_ElemXKeyPadValSetSign](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bPositive)
- bool [ElemXKeyPadAddChar](#) ([gslc_tsGui](#) *pGui, [gslc_tsXKeyPad](#) *pKeyPad, char ch)
- bool [gslc_ElemXKeyPadClick](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nX, int16_t nY)
Handle a click event within the KeyPad.
- bool [gslc_ElemXKeyPadTouch](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY)
Handle touch (up,down,move) events to KeyPad element.
- void [gslc_ElemXKeyPadCfgSetButtonSz](#) ([gslc_tsXKeyPadCfg](#) *pConfig, int8_t nButtonSzW, int8_t nButtonSzH)
Update the KeyPad configuration to define the KeyPad button sizing.
- void [gslc_ElemXKeyPadCfgSetFloatEn](#) ([gslc_tsXKeyPadCfg](#) *pConfig, bool bEn)
Update the KeyPad configuration to enable floating point numbers.
- void [gslc_ElemXKeyPadCfgSetSignEn](#) ([gslc_tsXKeyPadCfg](#) *pConfig, bool bEn)
Update the KeyPad configuration to enable negative numbers.
- void [gslc_ElemXKeyPadCfgSetRoundEn](#) ([gslc_tsXKeyPadCfg](#) *pConfig, bool bEn)
Update the KeyPad configuration to enable rounded button corners.
- void [gslc_ElemXKeyPadSetFloatEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bEn)
Update the KeyPad active configuration to enable negative numbers.
- void [gslc_ElemXKeyPadSetSignEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bEn)
Update the KeyPad active configuration to enable negative numbers.
- void [gslc_ElemXKeyPadInputAsk](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pKeyPadRef, int16_t nPgPopup, [gslc_tsElemRef](#) *pTxtRef)
Trigger a KeyPad popup and associate it with a text element.
- void [gslc_ElemXKeyPadInputGet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pTxtRef, void *pvCbData)
Complete a KeyPad popup by retrieving the input data and storing it in the text element.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []
- const char [KEYPAD_DISP_NEGATIVE](#)
- const char [KEYPAD_DISP_DECIMAL_PT](#)

9.10.1 Function Documentation

9.10.1.1 [bool ElemXKeyPadAddChar \(\[gslc_tsGui\]\(#\) * pGui, \[gslc_tsXKeyPad\]\(#\) * pKeyPad, char ch \)](#)

9.10.1.2 [void gslc_ElemXKeyPadCfgSetButtonSz \(\[gslc_tsXKeyPadCfg\]\(#\) * pConfig, int8_t nButtonSzW, int8_t nButtonSzH \)](#)

Update the KeyPad configuration to define the KeyPad button sizing.

Parameters

in	<i>pConfig</i>	Pointer to the XKeyPad config structure
in	<i>nButtonSzW</i>	Width of buttons in pixels
in	<i>nButtonSzH</i>	Width of buttons in pixels

Returns

none

9.10.1.3 void gslc_ElemXKeyPadCfgSetFloatEn (gslc_tsXKeyPadCfg * *pConfig*, bool *bEn*)

Update the KeyPad configuration to enable floating point numbers.

- Effectively disables/enables the decimal point button & handling

Parameters

in	<i>pConfig</i>	Pointer to the XKeyPad config structure
in	<i>bEn</i>	Enable flag (true if floating point enabled)

Returns

none

9.10.1.4 void gslc_ElemXKeyPadCfgSetRoundEn (gslc_tsXKeyPadCfg * *pConfig*, bool *bEn*)

Update the KeyPad configuration to enable rounded button corners.

Parameters

in	<i>pConfig</i>	Pointer to the XKeyPad config structure
in	<i>bEn</i>	Enable rounded corners

Returns

none

9.10.1.5 void gslc_ElemXKeyPadCfgSetSignEn (gslc_tsXKeyPadCfg * *pConfig*, bool *bEn*)

Update the KeyPad configuration to enable negative numbers.

- Effectively disables/enables the sign button & handling

Parameters

in	<i>pConfig</i>	Pointer to the XKeyPad config structure
in	<i>bEn</i>	Enable flag (true if negative numbers enabled)

Returns

none

9.10.1.6 `bool gslc_ElemXKeyPadClick (void * pvGui, void * pElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

Handle a click event within the KeyPad.

- This is called internally by the KeyPad touch handler

Parameters

in	<i>pGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pElemRef</i>	Void ptr to Element Ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nX</i>	Touch X coord
in	<i>nY</i>	Touch Y coord

Returns

none

9.10.1.7 `gslc_tsElemRef* gslc_ElemXKeyPadCreateBase (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad * pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg * pConfig, XKEYPAD_CREATE pfuncCreate, XKEYPAD_LOOKUP pfuncLookup)`

Create a KeyPad Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or <code>GSLC_ID_AUTO</code> to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nX0</i>	X KeyPad Starting Coordinate
in	<i>nY0</i>	Y KeyPad Starting Coordinate
in	<i>nFontId</i>	Font ID to use for drawing the element
in	<i>pConfig</i>	Ptr to Config options
in	<i>pfuncCreate</i>	Ptr to callback function for creation
in	<i>pfuncLookup</i>	Ptr to callback function for button lookups

Returns

Pointer to Element or NULL if failure

9.10.1.8 `int16_t gslc_ElemXKeyPadDataTargetIdGet (gslc_tsGui * pGui, void * pvData)`

Fetch the element target ID associated with this KeyPad.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pvData</i>	: Void ptr to callback data structure

Returns

Target Element ID or GSLC_ID_NONE if unspecified

9.10.1.9 `char* gslc_ElemXKeyPadDataValGet (gslc_tsGui * pGui, void * pvData)`

Fetch the edited value string from the KeyPad.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pvData</i>	: Void ptr to callback data structure

Returns

Pointer to edited character string

9.10.1.10 `bool gslc_ElemXKeyPadDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a KeyPad element on the screen.

- Called during redraw

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.10.1.11 `void gslc_ElemXKeyPadInputAsk (gslc_tsGui * pGui, gslc_tsElemRef * pKeyPadRef, int16_t nPgPopup, gslc_tsElemRef * pTxtRef)`

Trigger a KeyPad popup and associate it with a text element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pKeyPadRef</i>	Pointer to KeyPad element reference
in	<i>nPgPopup</i>	Page enum that contains the popup to show
in	<i>pTxtRef</i>	Pointer to associated text field element reference

Returns

none

9.10.1.12 void gslc_ElemXKeyPadInputGet (gslc_tsGui * *pGui*, gslc_tsElemRef * *pTxtRef*, void * *pvCbData*)

Complete a KeyPad popup by retrieving the input data and storing it in the text element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pTxtRef</i>	Pointer to associated text field element reference
in	<i>pvCbData</i>	Void pointer to callback function's pvData

Returns

none

9.10.1.13 void gslc_ElemXKeyPadSetFloatEn (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bEn*)

Update the KeyPad active configuration to enable negative numbers.

- Effectively disables/enables the sign button & handling

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to KeyPad element reference
in	<i>bEn</i>	Enable flag (true if negative numbers enabled)

Returns

none

9.10.1.14 void gslc_ElemXKeyPadSetSignEn (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bEn*)

Update the KeyPad active configuration to enable negative numbers.

- Effectively disables/enables the sign button & handling

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to KeyPad element reference
in	<i>bEn</i>	Enable flag (true if negative numbers enabled)

Returns

none

9.10.1.15 void `gslc_ElemXKeyPadTargetIdSet` (`gslc_tsGui` * *pGui*, `gslc_tsElemRef` * *pElemRef*, int16_t *nId*)

Set target element ID for KeyPad return value.

- The Target ID is used in the `GSLC_CB_INPUT` callback so that the user has the context needed to determine which field should be edited with the contents of the KeyPad edit field
- It is expected that the user will call this function when showing the KeyPad popup dialog

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to KeyPad Element reference
in	<i>nId</i>	Element enum ID for target of KeyPad value

Returns

none

9.10.1.16 bool `gslc_ElemXKeyPadTouch` (void * *pvGui*, void * *pvElemRef*, `gslc_teTouch` *eTouch*, int16_t *nRelX*, int16_t *nRelY*)

Handle touch (up,down,move) events to KeyPad element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

Returns

true if success, false otherwise

9.10.1.17 `bool gslc_ElemXKeyPadValGet (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, char * pStrBuf, uint8_t nStrBufMax)`

Set the current output string buffer associated with NumericInput element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to KeyPad Element reference
in	<i>pStrBuf</i>	String to copy into element
in	<i>nStrBufMax</i>	Maximum length of string buffer (pStrBuf)

Returns

none

9.10.1.18 `void gslc_ElemXKeyPadValSet (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, const char * pStrBuf)`

Set the current value for the editable text field.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to KeyPad Element reference
in	<i>pStrBuf</i>	String to copy into keypad

Returns

none

9.10.1.19 `void gslc_ElemXKeyPadValSetCb (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, GSLC_CB_INPUT pfuncCb)`

Set the callback function associated with the KeyPad.

- This function will be called during updates and OK / Cancel

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element Reference for KeyPad
in	<i>pfuncCb</i>	Callback function pointer

Returns

none

9.10.1.20 void `gslc_ElemXKeyPadValSetSign (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bPositive)`

9.10.1.21 void `XKeyPadAddKeyElem (gslc_tsGui * pGui, gslc_tsXKeyPad * pXData, int16_t nKeyId, bool bTxtField, int16_t nRow, int16_t nCol, int8_t nRowSpan, int8_t nColSpan, gslc_tsColor cColFill, gslc_tsColor cColGlow, bool bVisible)`

Add a key to the KeyPad control.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nKeyId</i>	ID to associated with the key
in	<i>bTxtField</i>	Is this the text value field?
in	<i>nRow</i>	Element placement position (row index, 0 at top)
in	<i>nCol</i>	Element placement position (column index, 0 at left)
in	<i>nRowSpan</i>	Number of columns to occupy by element (1 for normal size, 2 for double width)
in	<i>nColSpan</i>	Number of rows to occupy by element (1 for normal size, 2 for double height)
in	<i>cColFill</i>	Fill color for element
in	<i>cColGlow</i>	Fill color for element when glowing
in	<i>bVisible</i>	Initial key visibility state

Returns

none

9.10.2 Variable Documentation

9.10.2.1 const char `GSLC_PMEM ERRSTR_NULL[]`

9.10.2.2 const char `GSLC_PMEM ERRSTR_PXD_NULL[]`

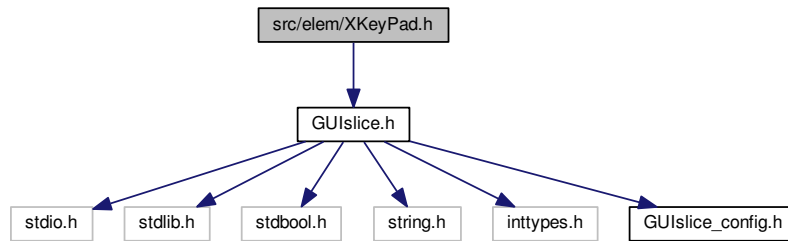
9.10.2.3 const char `KEYPAD_DISP_DECIMAL_PT`

9.10.2.4 const char `KEYPAD_DISP_NEGATIVE`

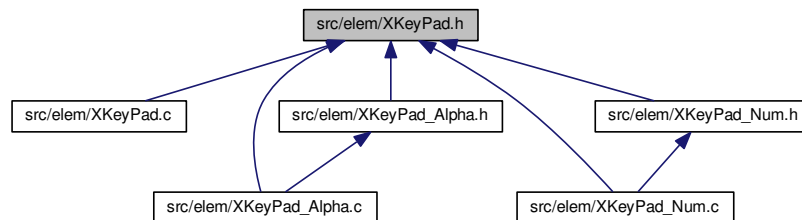
9.11 src/elem/XKeyPad.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XKeyPad.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXKeyPadCfg](#)
Configuration for the KeyPad.
- struct [gslc_tsXKeyPadData](#)
Input callback data structure.
- struct [gslc_tsXKeyPad](#)
Extended data for KeyPad element.

Macros

- `#define` [GSLC_TYPEX_KEYPAD](#)
- `#define` [XKEYPAD_VAL_LEN](#)
- `#define` [XKEYPAD_CB_STATE_DONE](#)
- `#define` [XKEYPAD_CB_STATE_CANCEL](#)
- `#define` [XKEYPAD_CB_STATE_UPDATE](#)

Typedefs

- typedef int16_t(* [XKEYPAD_LOOKUP](#)) (gslc_tsGui *pGui, int16_t nKeyId)
Function for KeyPad creation.
- typedef void(* [XKEYPAD_CREATE](#)) (gslc_tsGui *pGui, [gslc_tsXKeyPad](#) *pXData)

Enumerations

- enum {
[KEYPAD_ID_BACKSPACE](#), [KEYPAD_ID_PERIOD](#), [KEYPAD_ID_SPACE](#), [KEYPAD_ID_DECIMAL](#),
[KEYPAD_ID_MINUS](#), [KEYPAD_ID_ESC](#), [KEYPAD_ID_ENTER](#), [KEYPAD_ID_BASIC_START](#),
[KEYPAD_ID_TXT](#) }

Functions

- void [XKeyPadAddKeyElem](#) ([gslc_tsGui](#) *pGui, [gslc_tsXKeyPad](#) *pXData, int16_t nKeyId, bool bTxtField, int16_t nRow, int16_t nCol, int8_t nRowSpan, int8_t nColSpan, [gslc_tsColor](#) cColFill, [gslc_tsColor](#) cColGlow, bool bVisible)
 Add a key to the KeyPad control.
- [gslc_tsElemRef](#) * [gslc_ElemXKeyPadCreateBase](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXKeyPad](#) *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, [gslc_tsXKeyPadCfg](#) *pConfig, [XKEYPAD_CREATE](#) pfuncCreate, [XKEYPAD_LOOKUP](#) pfuncLookup)
 Create a KeyPad Element.
- void [gslc_ElemXKeyPadValSet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, const char *pStrBuf)
 Set the current value for the editable text field.
- void [gslc_ElemXKeyPadTargetIdSet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nId)
 Set target element ID for KeyPad return value.
- bool [gslc_ElemXKeyPadValGet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, char *pStrBuf, uint8_t nStrBufMax)
 Set the current output string buffer associated with NumericInput element.
- char * [gslc_ElemXKeyPadDataValGet](#) ([gslc_tsGui](#) *pGui, void *pvData)
 Fetch the edited value string from the KeyPad.
- int16_t [gslc_ElemXKeyPadDataTargetIdGet](#) ([gslc_tsGui](#) *pGui, void *pvData)
 Fetch the element target ID associated with this KeyPad.
- bool [gslc_ElemXKeyPadDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
 Draw a KeyPad element on the screen.
- bool [gslc_ElemXKeyPadClick](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nX, int16_t nY)
 Handle a click event within the KeyPad.
- bool [gslc_ElemXKeyPadTouch](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY)
 Handle touch (up,down,move) events to KeyPad element.
- void [gslc_ElemXKeyPadValSetCb](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_INPUT](#) pfuncCb)
 Set the callback function associated with the KeyPad.
- void [gslc_ElemXKeyPadCfgSetFloatEn](#) ([gslc_tsXKeyPadCfg](#) *pConfig, bool bEn)
 Update the KeyPad configuration to enable floating point numbers.
- void [gslc_ElemXKeyPadCfgSetSignEn](#) ([gslc_tsXKeyPadCfg](#) *pConfig, bool bEn)
 Update the KeyPad configuration to enable negative numbers.
- void [gslc_ElemXKeyPadCfgSetRoundEn](#) ([gslc_tsXKeyPadCfg](#) *pConfig, bool bEn)
 Update the KeyPad configuration to enable rounded button corners.
- void [gslc_ElemXKeyPadCfgSetButtonSz](#) ([gslc_tsXKeyPadCfg](#) *pConfig, int8_t nButtonSzW, int8_t nButtonSzH)
 Update the KeyPad configuration to define the KeyPad button sizing.
- void [gslc_ElemXKeyPadSetFloatEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bEn)
 Update the KeyPad active configuration to enable negative numbers.
- void [gslc_ElemXKeyPadSetSignEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bEn)
 Update the KeyPad active configuration to enable negative numbers.

- void `gslc_ElemXKeyPadInputAsk` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pKeyPadRef, `int16_t` nPgPopup, `gslc_tsElemRef` *pTxtRef)
Trigger a KeyPad popup and associate it with a text element.
- void `gslc_ElemXKeyPadInputGet` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pTxtRef, void *pvCbData)
Complete a KeyPad popup by retrieving the input data and storing it in the text element.

9.11.1 Macro Definition Documentation

9.11.1.1 `#define GSLC_TYPEX_KEYPAD`

9.11.1.2 `#define XKEYPAD_CB_STATE_CANCEL`

9.11.1.3 `#define XKEYPAD_CB_STATE_DONE`

9.11.1.4 `#define XKEYPAD_CB_STATE_UPDATE`

9.11.1.5 `#define XKEYPAD_VAL_LEN`

9.11.2 Typedef Documentation

9.11.2.1 `typedef void(* XKEYPAD_CREATE) (gslc_tsGui *pGui, gslc_tsXKeyPad *pXData)`

9.11.2.2 `typedef int16_t(* XKEYPAD_LOOKUP) (gslc_tsGui *pGui, int16_t nKeyId)`

Function for KeyPad creation.

9.11.3 Enumeration Type Documentation

9.11.3.1 anonymous enum

Enumerator

`KEYPAD_ID_BACKSPACE`
`KEYPAD_ID_PERIOD`
`KEYPAD_ID_SPACE`
`KEYPAD_ID_DECIMAL`
`KEYPAD_ID_MINUS`
`KEYPAD_ID_ESC`
`KEYPAD_ID_ENTER`
`KEYPAD_ID_BASIC_START`
`KEYPAD_ID_TXT`

9.11.4 Function Documentation

9.11.4.1 void `gslc_ElemXKeyPadCfgSetButtonSz` (`gslc_tsXKeyPadCfg` * pConfig, `int8_t` nButtonSzW, `int8_t` nButtonSzH)

Update the KeyPad configuration to define the KeyPad button sizing.

Parameters

in	<i>pConfig</i>	Pointer to the XKeyPad config structure
in	<i>nButtonSzW</i>	Width of buttons in pixels
in	<i>nButtonSzH</i>	Width of buttons in pixels

Returns

none

9.11.4.2 void gslc_ElemXKeyPadCfgSetFloatEn (gslc_tsXKeyPadCfg * *pConfig*, bool *bEn*)

Update the KeyPad configuration to enable floating point numbers.

- Effectively disables/enables the decimal point button & handling

Parameters

in	<i>pConfig</i>	Pointer to the XKeyPad config structure
in	<i>bEn</i>	Enable flag (true if floating point enabled)

Returns

none

9.11.4.3 void gslc_ElemXKeyPadCfgSetRoundEn (gslc_tsXKeyPadCfg * *pConfig*, bool *bEn*)

Update the KeyPad configuration to enable rounded button corners.

Parameters

in	<i>pConfig</i>	Pointer to the XKeyPad config structure
in	<i>bEn</i>	Enable rounded corners

Returns

none

9.11.4.4 void gslc_ElemXKeyPadCfgSetSignEn (gslc_tsXKeyPadCfg * *pConfig*, bool *bEn*)

Update the KeyPad configuration to enable negative numbers.

- Effectively disables/enables the sign button & handling

Parameters

in	<i>pConfig</i>	Pointer to the XKeyPad config structure
in	<i>bEn</i>	Enable flag (true if negative numbers enabled)

Returns

none

9.11.4.5 `bool gslc_ElemXKeyPadClick (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

Handle a click event within the KeyPad.

- This is called internally by the KeyPad touch handler

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element Ref (typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nX</i>	Touch X coord
in	<i>nY</i>	Touch Y coord

Returns

none

9.11.4.6 `gslc_tsElemRef* gslc_ElemXKeyPadCreateBase (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad * pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg * pConfig, XKEYPAD_CREATE pfuncCreate, XKEYPAD_LOOKUP pfuncLookup)`

Create a KeyPad Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nX0</i>	X KeyPad Starting Coordinate
in	<i>nY0</i>	Y KeyPad Starting Coordinate
in	<i>nFontId</i>	Font ID to use for drawing the element
in	<i>pConfig</i>	Ptr to Config options
in	<i>pfuncCreate</i>	Ptr to callback function for creation
in	<i>pfuncLookup</i>	Ptr to callback function for button lookups

Returns

Pointer to Element or NULL if failure

9.11.4.7 `int16_t gslc_ElemXKeyPadDataTargetIdGet (gslc_tsGui * pGui, void * pvData)`

Fetch the element target ID associated with this KeyPad.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pvData</i>	: Void ptr to callback data structure

Returns

Target Element ID or GSLC_ID_NONE if unspecified

9.11.4.8 `char* gslc_ElemXKeyPadDataValGet (gslc_tsGui * pGui, void * pvData)`

Fetch the edited value string from the KeyPad.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pvData</i>	: Void ptr to callback data structure

Returns

Pointer to edited character string

9.11.4.9 `bool gslc_ElemXKeyPadDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a KeyPad element on the screen.

- Called during redraw

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.11.4.10 void gslc_ElemXKeyPadInputAsk (gslc_tsGui * *pGui*, gslc_tsElemRef * *pKeyPadRef*, int16_t *nPgPopup*, gslc_tsElemRef * *pTxtRef*)

Trigger a KeyPad popup and associate it with a text element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pKeyPadRef</i>	Pointer to KeyPad element reference
in	<i>nPgPopup</i>	Page enum that contains the popup to show
in	<i>pTxtRef</i>	Pointer to associated text field element reference

Returns

none

9.11.4.11 void gslc_ElemXKeyPadInputGet (gslc_tsGui * *pGui*, gslc_tsElemRef * *pTxtRef*, void * *pvCbData*)

Complete a KeyPad popup by retrieving the input data and storing it in the text element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pTxtRef</i>	Pointer to associated text field element reference
in	<i>pvCbData</i>	Void pointer to callback function's pvData

Returns

none

9.11.4.12 void gslc_ElemXKeyPadSetFloatEn (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bEn*)

Update the KeyPad active configuration to enable negative numbers.

- Effectively disables/enables the sign button & handling

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to KeyPad element reference
in	<i>bEn</i>	Enable flag (true if negative numbers enabled)

Returns

none

9.11.4.13 `void gslc_ElemXKeyPadSetSignEn (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bEn)`

Update the KeyPad active configuration to enable negative numbers.

- Effectively disables/enables the sign button & handling

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to KeyPad element reference
in	<i>bEn</i>	Enable flag (true if negative numbers enabled)

Returns

none

9.11.4.14 `void gslc_ElemXKeyPadTargetIdSet (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nId)`

Set target element ID for KeyPad return value.

- The Target ID is used in the GSLC_CB_INPUT callback so that the user has the context needed to determine which field should be edited with the contents of the KeyPad edit field
- It is expected that the user will call this function when showing the KeyPad popup dialog

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to KeyPad Element reference
in	<i>nId</i>	Element enum ID for target of KeyPad value

Returns

none

9.11.4.15 `bool gslc_ElemXKeyPadTouch (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch (up,down,move) events to KeyPad element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pVGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pElemRef</i>	Void ptr to Element ref (typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

Returns

true if success, false otherwise

9.11.4.16 `bool gslc_ElemXKeyPadValGet (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, char * pStrBuf, uint8_t nStrBufMax)`

Set the current output string buffer associated with NumericInput element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to KeyPad Element reference
in	<i>pStrBuf</i>	String to copy into element
in	<i>nStrBufMax</i>	Maximum length of string buffer (pStrBuf)

Returns

none

9.11.4.17 `void gslc_ElemXKeyPadValSet (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, const char * pStrBuf)`

Set the current value for the editable text field.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to KeyPad Element reference
in	<i>pStrBuf</i>	String to copy into keypad

Returns

none

9.11.4.18 `void gslc_ElemXKeyPadValSetCb (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, GSLC_CB_INPUT pfuncCb)`

Set the callback function associated with the KeyPad.

- This function will be called during updates and OK / Cancel

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element Reference for KeyPad
in	<i>pfuncCb</i>	Callback function pointer

Returns

none

9.11.4.19 void XKeyPadAddKeyElem (gslc_tsGui * *pGui*, gslc_tsXKeyPad * *pXData*, int16_t *nKeyId*, bool *bTxtField*, int16_t *nRow*, int16_t *nCol*, int8_t *nRowSpan*, int8_t *nColSpan*, gslc_tsColor *cColFill*, gslc_tsColor *cColGlow*, bool *bVisible*)

Add a key to the KeyPad control.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nKeyId</i>	ID to associated with the key
in	<i>bTxtField</i>	Is this the text value field?
in	<i>nRow</i>	Element placement position (row index, 0 at top)
in	<i>nCol</i>	Element placement position (column index, 0 at left)
in	<i>nRowSpan</i>	Number of columns to occupy by element (1 for normal size, 2 for double width)
in	<i>nColSpan</i>	Number of rows to occupy by element (1 for normal size, 2 for double height)
in	<i>cColFill</i>	Fill color for element
in	<i>cColGlow</i>	Fill color for element when glowing
in	<i>bVisible</i>	Initial key visibility state

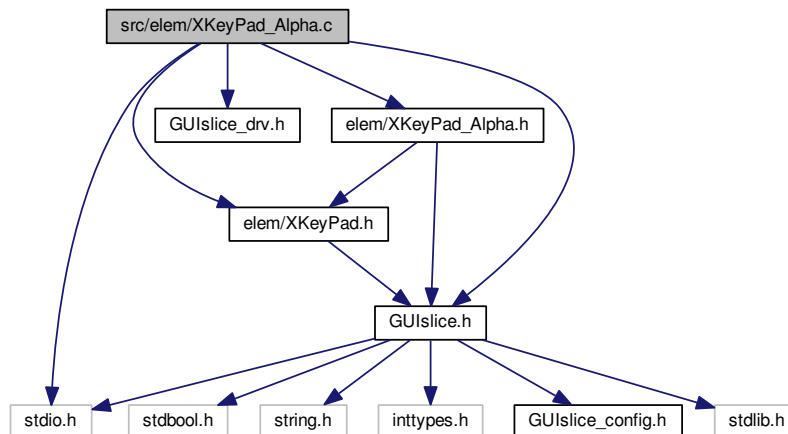
Returns

none

9.12 src/elem/XKeyPad_Alpha.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XKeyPad.h"
#include "elem/XKeyPad_Alpha.h"
#include <stdio.h>
```

Include dependency graph for XKeyPad_Alpha.c:



Enumerations

- enum {
[KEYPAD_LBL_BACKSPACE](#), [KEYPAD_LBL_PERIOD](#), [KEYPAD_LBL_SPACE](#), [KEYPAD_LBL_ESC](#),
[KEYPAD_LBL_ENTER](#), [KEYPAD_LBL_BASIC_START](#) }

Functions

- void [XKeyPadCreateKeys_Alpha](#) ([gslc_tsGui](#) *pGui, [gslc_tsXKeyPad](#) *pXData)
- int16_t [XKeyPadLookup_Alpha](#) ([gslc_tsGui](#) *pGui, int16_t nKeyId)
- [gslc_tsElemRef](#) * [gslc_ElemXKeyPadCreate_Alpha](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXKeyPad_Alpha](#) *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, [gslc_tsXKeyPadCfg](#) *pConfig)
Create a KeyPad Element.
- [gslc_tsXKeyPadCfg](#) [gslc_ElemXKeyPadCfgInit_Alpha](#) ()
Initialize the KeyPad config structure.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []
- static char * [KEYPAD_LABEL_STRINGS](#) []

9.12.1 Enumeration Type Documentation

9.12.1.1 anonymous enum

Enumerator

[KEYPAD_LBL_BACKSPACE](#)
[KEYPAD_LBL_PERIOD](#)
[KEYPAD_LBL_SPACE](#)
[KEYPAD_LBL_ESC](#)
[KEYPAD_LBL_ENTER](#)
[KEYPAD_LBL_BASIC_START](#)

9.12.2 Function Documentation

9.12.2.1 `gslc_tsXKeyPadCfg gslc_ElemXKeyPadCfgInit_Alpha ()`

Initialize the KeyPad config structure.

- This routine should be called to initialize the configuration data structure before calling any of the KeyPad config APIs

Returns

Initialized KeyPad config structure

9.12.2.2 `gslc_tsElemRef* gslc_ElemXKeyPadCreate_Alpha (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad_Alpha * pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg * pConfig)`

Create a KeyPad Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nX0</i>	X KeyPad Starting Coordinate
in	<i>nY0</i>	Y KeyPad Starting Coordinate
in	<i>nFontId</i>	Font ID to use for drawing the element
in	<i>pConfig</i>	Ptr to config options

Returns

Pointer to Element or NULL if failure

9.12.2.3 `void XKeyPadCreateKeys_Alpha (gslc_tsGui * pGui, gslc_tsXKeyPad * pXData)`

9.12.2.4 `int16_t XKeyPadLookup_Alpha (gslc_tsGui * pGui, int16_t nKeyId)`

9.12.3 Variable Documentation

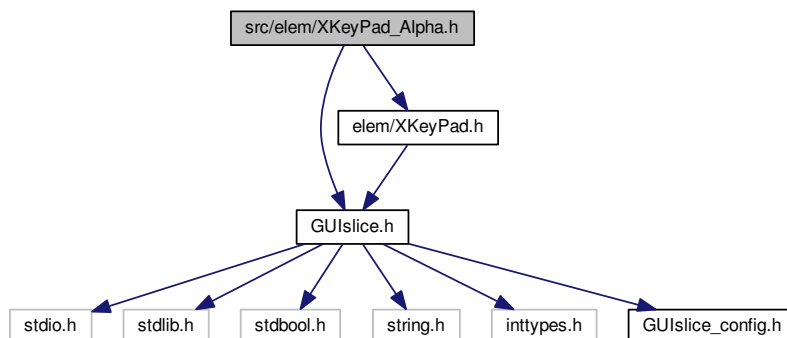
9.12.3.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.12.3.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

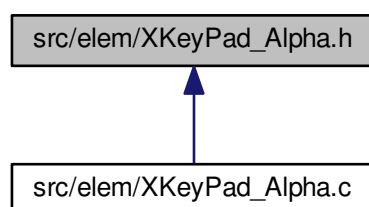
9.12.3.3 `char* KEYPAD_LABEL_STRINGS[]` `[static]`

9.13 src/elem/XKeyPad_Alpha.h File Reference

```
#include "GUIslice.h"
#include "elem/XKeyPad.h"
Include dependency graph for XKeyPad_Alpha.h:
```



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXKeyPad_Alpha](#)

Macros

- `#define` [XKEYPADALPHA_BTN_BASIC](#)
- `#define` [XKEYPADALPHA_ELEM_MAX](#)

Functions

- `gslc_tsElemRef * gslc_ElemXKeyPadCreate_Alpha (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad_Alpha *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg *pConfig)`
Create a KeyPad Element.
- `gslc_tsXKeyPadCfg gslc_ElemXKeyPadCfgInit_Alpha ()`
Initialize the KeyPad config structure.

9.13.1 Macro Definition Documentation

9.13.1.1 `#define XKEYPADALPHA_BTN_BASIC`

9.13.1.2 `#define XKEYPADALPHA_ELEM_MAX`

9.13.2 Function Documentation

9.13.2.1 `gslc_tsXKeyPadCfg gslc_ElemXKeyPadCfgInit_Alpha ()`

Initialize the KeyPad config structure.

- This routine should be called to initialize the configuration data structure before calling any of the KeyPad config APIs

Returns

Initialized KeyPad config structure

9.13.2.2 `gslc_tsElemRef* gslc_ElemXKeyPadCreate_Alpha (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad_Alpha * pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg * pConfig)`

Create a KeyPad Element.

Parameters

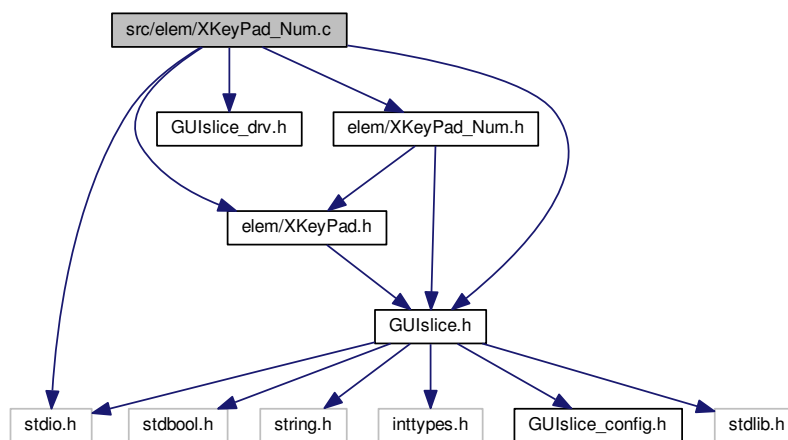
in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nX0</i>	X KeyPad Starting Coordinate
in	<i>nY0</i>	Y KeyPad Starting Coordinate
in	<i>nFontId</i>	Font ID to use for drawing the element
in	<i>pConfig</i>	Ptr to config options

Returns

Pointer to Element or NULL if failure

9.14 src/elem/XKeyPad_Num.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XKeyPad.h"
#include "elem/XKeyPad_Num.h"
#include <stdio.h>
Include dependency graph for XKeyPad_Num.c:
```



Enumerations

- enum {
[KEYPAD_LBL_BACKSPACE](#), [KEYPAD_LBL_DECIMAL](#), [KEYPAD_LBL_MINUS](#), [KEYPAD_LBL_ESC](#),
[KEYPAD_LBL_ENTER](#), [KEYPAD_LBL_BASIC_START](#) }

Functions

- void [XKeyPadCreateKeys_Num](#) ([gslc_tsGui](#) *pGui, [gslc_tsXKeyPad](#) *pXData)
- int16_t [XKeyPadLookup_Num](#) ([gslc_tsGui](#) *pGui, int16_t nKeyId)
- [gslc_tsElemRef](#) * [gslc_ElemXKeyPadCreate_Num](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXKeyPad_Num](#) *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, [gslc_tsXKeyPadCfg](#) *pConfig)
Create a KeyPad Element.
- [gslc_tsXKeyPadCfg](#) [gslc_ElemXKeyPadCfgInit_Num](#) ()
Initialize the KeyPad config structure.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []
- static char * [KEYPAD_LABEL_STRINGS](#) []

9.14.1 Enumeration Type Documentation

9.14.1.1 anonymous enum

Enumerator

KEYPAD_LBL_BACKSPACE
KEYPAD_LBL_DECIMAL
KEYPAD_LBL_MINUS
KEYPAD_LBL_ESC
KEYPAD_LBL_ENTER
KEYPAD_LBL_BASIC_START

9.14.2 Function Documentation

9.14.2.1 gslc_tsXKeyPadCfg gslc_ElemXKeyPadCfgInit_Num ()

Initialize the KeyPad config structure.

- This routine should be called to initialize the configuration data structure before calling any of the KeyPad config APIs

Returns

Initialized KeyPad config structure

9.14.2.2 gslc_tsElemRef* gslc_ElemXKeyPadCreate_Num (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXKeyPad_Num * pXData, int16_t nX0, int16_t nY0, int8_t nFontId, gslc_tsXKeyPadCfg * pConfig)

Create a KeyPad Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nX0</i>	X KeyPad Starting Coordinate
in	<i>nY0</i>	Y KeyPad Starting Coordinate
in	<i>nFontId</i>	Font ID to use for drawing the element
in	<i>pConfig</i>	Ptr to config options

Returns

Pointer to Element or NULL if failure

9.14.2.3 void XKeyPadCreateKeys_Num (gslc_tsGui * pGui, gslc_tsXKeyPad * pXData)

9.14.2.4 int16_t XKeyPadLookup_Num (gslc_tsGui * pGui, int16_t nKeyld)

9.14.3 Variable Documentation

9.14.3.1 const char GSLC_PMEM ERRSTR_NULL[]

9.14.3.2 const char GSLC_PMEM ERRSTR_PXD_NULL[]

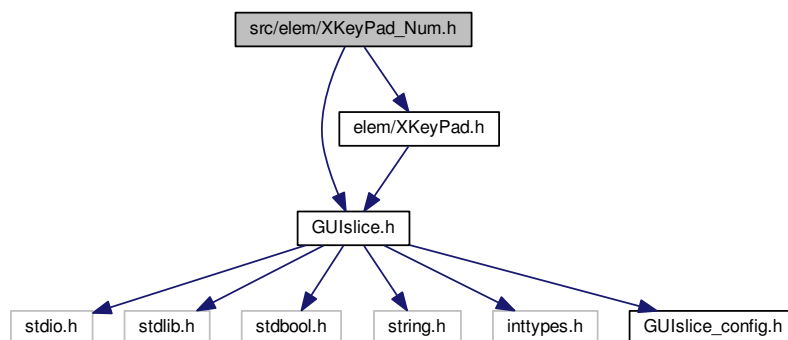
9.14.3.3 char* KEYPAD_LABEL_STRINGS[] [static]

9.15 src/elem/XKeyPad_Num.h File Reference

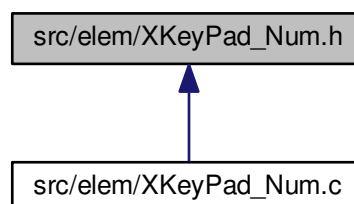
```
#include "GUIslice.h"
```

```
#include "elem/XKeyPad.h"
```

Include dependency graph for XKeyPad_Num.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXKeyPad_Num](#)

Macros

- #define [XKEYPADNUM_BTN_BASIC](#)
- #define [XKEYPADNUM_ELEM_MAX](#)

Functions

- [gslc_tsElemRef](#) * [gslc_ElemXKeyPadCreate_Num](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXKeyPad_Num](#) *pXData, int16_t nX0, int16_t nY0, int8_t nFontId, [gslc_tsXKeyPadCfg](#) *pConfig)
Create a KeyPad Element.
- [gslc_tsXKeyPadCfg](#) [gslc_ElemXKeyPadCfgInit_Num](#) ()
Initialize the KeyPad config structure.

9.15.1 Macro Definition Documentation

9.15.1.1 #define XKEYPADNUM_BTN_BASIC

9.15.1.2 #define XKEYPADNUM_ELEM_MAX

9.15.2 Function Documentation

9.15.2.1 [gslc_tsXKeyPadCfg](#) [gslc_ElemXKeyPadCfgInit_Num](#) ()

Initialize the KeyPad config structure.

- This routine should be called to initialize the configuration data structure before calling any of the KeyPad config APIs

Returns

Initialized KeyPad config structure

9.15.2.2 [gslc_tsElemRef](#)* [gslc_ElemXKeyPadCreate_Num](#) ([gslc_tsGui](#) * pGui, int16_t nElemId, int16_t nPage, [gslc_tsXKeyPad_Num](#) * pXData, int16_t nX0, int16_t nY0, int8_t nFontId, [gslc_tsXKeyPadCfg](#) * pConfig)

Create a KeyPad Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>n↔ ElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>nX0</i>	X KeyPad Starting Coordinate
in	<i>nY0</i>	Y KeyPad Starting Coordinate

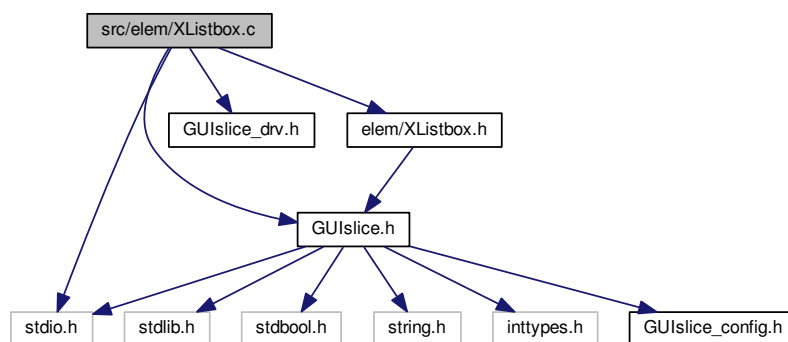
Returns

Pointer to Element or NULL if failure

9.16 src/elem/XListbox.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XListbox.h"
#include <stdio.h>
```

Include dependency graph for XListbox.c:



Macros

- `#define XLISTBOX_MAX_STR`

Functions

- `bool gslc_ElemXListboxRecalcSize (gslc_tsXListbox *pListbox, gslc_tsRect rElem)`
- `void gslc_ElemXListboxSetSize (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nRows, int8_t nCols)`
Configure the number of rows & columns to display in the listbox.
- `void gslc_ElemXListboxSetMargin (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nMarginW, int8_t nMarginH)`
Configure the margin inside the listbox.
- `void gslc_ElemXListboxItemsSetSize (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nItemW, int16_t nItemH)`
Configure the size of the listbox items.
- `void gslc_ElemXListboxItemsSetGap (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nGap, gslc_tsColor colGap)`
Configure the gap between listbox items.
- `void gslc_ElemXListboxReset (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
Empty the listbox of all items.
- `bool gslc_ElemXListboxAddItem (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, const char *pStrItem)`
Add an item to the listbox.

- bool [gslc_ElemXListboxGetItem](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nItemCurSel, char *pStrItem, uint8_t nStrItemLen)
Get the indexed listbox item.
- int16_t [gslc_ElemXListboxGetItemCnt](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the number of items in the listbox.
- [gslc_tsElemRef](#) * [gslc_ElemXListboxCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsElemRef](#) *pXData, [gslc_tsRect](#) rElem, int16_t nFontId, uint8_t *pBufItems, uint16_t nBufItemsMax, int16_t nItemDefault)
Create a Listbox Element.
- bool [gslc_ElemXListboxDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a Listbox element on the screen.
- bool [gslc_ElemXListboxTouch](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY)
Handle touch events to Listbox element.
- int16_t [gslc_ElemXListboxGetSel](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get a Listbox element's current selection.
- bool [gslc_ElemXListboxSetSel](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nItemCurSel)
Set a Listbox element's current selection.
- bool [gslc_ElemXListboxSetScrollPos](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint16_t nScrollPos)
Set the Listbox scroll position.
- void [gslc_ElemXListboxSetSelFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_XLISTBOX_SEL](#) funcCb)
Assign the selection callback function for a Listbox.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.16.1 Macro Definition Documentation

9.16.1.1 #define XLISTBOX_MAX_STR

9.16.2 Function Documentation

9.16.2.1 bool [gslc_ElemXListboxAddItem](#) ([gslc_tsGui](#) * pGui, [gslc_tsElemRef](#) * pElemRef, const char * pStrItem)

Add an item to the listbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>pStrItem</i>	String to use when creating the listbox item

Returns

true if OK, false if fail (eg. insufficient buffer storage)

9.16.2.2 **gslc_tsElemRef*** **gslc_ElemXListboxCreate** (**gslc_tsGui** * *pGui*, **int16_t** *nElemId*, **int16_t** *nPage*, **gslc_tsXListbox** * *pXData*, **gslc_tsRect** *rElem*, **int16_t** *nFontId*, **uint8_t** * *pBufItems*, **uint16_t** *nBufItemsMax*, **int16_t** *nSelDefault*)

Create a Listbox Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining checkbox size
in	<i>nFontId</i>	Font ID for item display
in	<i>pBufItems</i>	Pointer to buffer that will contain list of items
in	<i>nBufItemsMax</i>	Max size of buffer for list of items (pBufItems)
in	<i>nSelDefault</i>	Default item to select

Returns

Pointer to Element reference or NULL if failure

9.16.2.3 **bool** **gslc_ElemXListboxDraw** (**void** * *pvGui*, **void** * *pvElemRef*, **gslc_teRedrawType** *eRedraw*)

Draw a Listbox element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.16.2.4 **bool** **gslc_ElemXListboxGetItem** (**gslc_tsGui** * *pGui*, **gslc_tsElemRef** * *pElemRef*, **int16_t** *nItemCurSel*, **char** * *pStrItem*, **uint8_t** *nStrItemLen*)

Get the indexed listbox item.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nItemCurSel</i>	Item index to fetch
out	<i>pStrItem</i>	Ptr to the string buffer to receive the item
in	<i>nStrItemLen</i>	Maximum buffer length of pStrItem

Returns

true if success, false if fail (eg. can't locate item)

9.16.2.5 `int16_t gslc_ElemXListboxGetItemCnt (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get the number of items in the listbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update

Returns

Number of items

9.16.2.6 `int16_t gslc_ElemXListboxGetSel (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get a Listbox element's current selection.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Current Listbox selection (or -1 if none)

9.16.2.7 `void gslc_ElemXListboxItemsSetGap (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int8_t nGap, gslc_tsColor colGap)`

Configure the gap between listbox items.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nGap</i>	Set the gap between listbox items (0 for none)
in	<i>colGap</i>	Set the color of the gap between listbox items

Returns

none

9.16.2.8 void gslc_ElemXListboxItemsSetSize (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nItemW*, int16_t *nItemH*)

Configure the size of the listbox items.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nItemW</i>	Set the width of a listbox item (or -1 to auto-size)
in	<i>nItemH</i>	Set the height of a listbox item

Returns

none

9.16.2.9 bool gslc_ElemXListboxRecalcSize (gslc_tsXListbox * *pListbox*, gslc_tsRect *rElem*)

9.16.2.10 void gslc_ElemXListboxReset (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

Empty the listbox of all items.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update

Returns

none

9.16.2.11 void gslc_ElemXListboxSetMargin (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int8_t *nMarginW*, int8_t *nMarginH*)

Configure the margin inside the listbox.

- Defines the region bewteen the element rect and the inner listbox items

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nMarginW</i>	Set the margin (horizontal) inside the listbox (0 for none)
in	<i>nMarginH</i>	Set the margin (horizontal) inside the listbox (0 for none)

Returns

none

9.16.2.12 `bool gslc_ElemXListboxSetScrollPos (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, uint16_t nScrollPos)`

Set the Listbox scroll position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nScrollPos</i>	Scroll the listbox so that the <i>nScrollPos</i> item is at the top (0 default)

Returns

true if success, false if fail

9.16.2.13 `bool gslc_ElemXListboxSetSel (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nItemCurSel)`

Set a Listbox element's current selection.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nItemCurSel</i>	Listbox item to select (or -1 for none)

Returns

true if success, false if fail

9.16.2.14 `void gslc_ElemXListboxSetSelFunc (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef,
GSLC_CB_XLISTBOX_SEL funcCb)`

Assign the selection callback function for a Listbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>funcCb</i>	Function pointer to selection routine (or NULL for none)

Returns

none

9.16.2.15 void gslc_ElemXListboxSetSize (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int8_t *nRows*, int8_t *nCols*)

Configure the number of rows & columns to display in the listbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nRows</i>	Number of rows (>= 1, or XLISTBOX_SIZE_AUTO to base on content)
in	<i>nCols</i>	Number of columns (>= 1)

Returns

none

9.16.2.16 bool gslc_ElemXListboxTouch (void * *pvGui*, void * *pvElemRef*, gslc_teTouch *eTouch*, int16_t *nRelX*, int16_t *nRelY*)

Handle touch events to Listbox element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

Returns

true if success, false otherwise

9.16.3 Variable Documentation

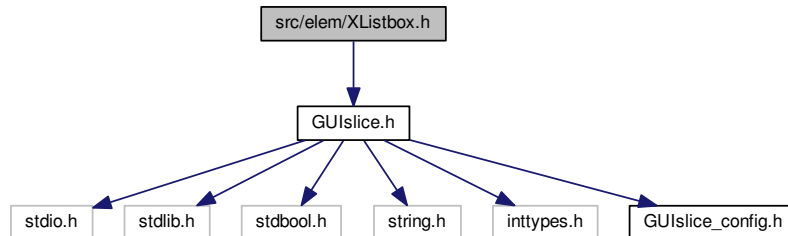
9.16.3.1 const char GSLC_PMEM ERRSTR_NULL[]

9.16.3.2 const char GSLC_PMEM ERRSTR_PXD_NULL[]

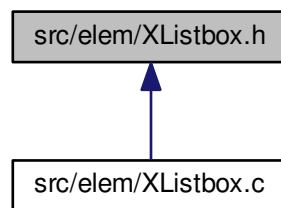
9.17 src/elem/XListbox.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XListbox.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXListbox](#)
Extended data for Listbox element.

Macros

- #define [GSLC_TYPEX_LISTBOX](#)
- #define [XLISTBOX_SEL_NONE](#)
- #define [XLISTBOX_SIZE_AUTO](#)
- #define [XLISTBOX_BUF_OH_R](#)

Typedefs

- typedef bool(* [GSLC_CB_XLISTBOX_SEL](#)) (void *pvGui, void *pvElem, int16_t nSel)
Callback function for Listbox feedback.

Functions

- [gslc_tsElemRef](#) * [gslc_ElemXListboxCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsElemRef](#) *pXData, [gslc_tsRect](#) rElem, int16_t nFontId, uint8_t *pBufItems, uint16_t nBufItemsMax, int16_t nSelDefault)
Create a Listbox Element.
- void [gslc_ElemXListboxSetSize](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int8_t nRows, int8_t nCols)
Configure the number of rows & columns to display in the listbox.
- void [gslc_ElemXListboxSetMargin](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int8_t nMarginW, int8_t nMarginH)
Configure the margin inside the listbox.
- void [gslc_ElemXListboxItemsSetSize](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nItemW, int16_t nItemH)
Configure the size of the listbox items.
- void [gslc_ElemXListboxItemsSetGap](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int8_t nGap, [gslc_tsColor](#) colGap)
Configure the gap between listbox items.
- void [gslc_ElemXListboxReset](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Empty the listbox of all items.
- bool [gslc_ElemXListboxAddItem](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, const char *pStrItem)
Add an item to the listbox.
- bool [gslc_ElemXListboxGetItem](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nItemCurSel, char *pStrItem, uint8_t nStrItemLen)
Get the indexed listbox item.
- int16_t [gslc_ElemXListboxGetItemCnt](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the number of items in the listbox.
- bool [gslc_ElemXListboxDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a Listbox element on the screen.
- bool [gslc_ElemXListboxTouch](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY)
Handle touch events to Listbox element.
- int16_t [gslc_ElemXListboxGetSel](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get a Listbox element's current selection.
- bool [gslc_ElemXListboxSetSel](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nItemCurSel)
Set a Listbox element's current selection.
- bool [gslc_ElemXListboxSetScrollPos](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint16_t nScrollPos)
Set the Listbox scroll position.
- void [gslc_ElemXListboxSetSelFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_XLISTBOX_SEL](#) funcCb)
Assign the selection callback function for a Listbox.

9.17.1 Macro Definition Documentation

9.17.1.1 `#define GSLC_TYPEX_LISTBOX`

9.17.1.2 `#define XLISTBOX_BUF_OH_R`

9.17.1.3 `#define XLISTBOX_SEL_NONE`

9.17.1.4 `#define XLISTBOX_SIZE_AUTO`

9.17.2 Typedef Documentation

9.17.2.1 `typedef bool(* GSLC_CB_XLISTBOX_SEL)(void *pvGui, void *pvElem, int16_t nSel)`

Callback function for Listbox feedback.

9.17.3 Function Documentation

9.17.3.1 `bool gslc_ElemXListboxAddItem (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, const char * pStrItem)`

Add an item to the listbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>pStrItem</i>	String to use when creating the listbox item

Returns

true if OK, false if fail (eg. insufficient buffer storage)

9.17.3.2 `gslc_tsElemRef* gslc_ElemXListboxCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXListbox * pXData, gslc_tsRect rElem, int16_t nFontId, uint8_t * pBufItems, uint16_t nBufItemsMax, int16_t nSelDefault)`

Create a Listbox Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining checkbox size
in	<i>nFontId</i>	Font ID for item display
in	<i>pBufItems</i>	Pointer to buffer that will contain list of items
in	<i>nBufItemsMax</i>	Max size of buffer for list of items (pBufItems)
in	<i>nSelDefault</i>	Default item to select

Returns

Pointer to Element reference or NULL if failure

9.17.3.3 `bool gslc_ElemXListboxDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a Listbox element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pVGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pVElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.17.3.4 `bool gslc_ElemXListboxGetItem (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nItemCurSel, char * pStrItem, uint8_t nStrItemLen)`

Get the indexed listbox item.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nItemCurSel</i>	Item index to fetch
out	<i>pStrItem</i>	Ptr to the string buffer to receive the item
in	<i>nStrItemLen</i>	Maximum buffer length of pStrItem

Returns

true if success, false if fail (eg. can't locate item)

9.17.3.5 `int16_t gslc_ElemXListboxGetItemCnt (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get the number of items in the listbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update

Returns

Number of items

9.17.3.6 `int16_t gslc_ElemXListboxGetSel (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get a Listbox element's current selection.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Current Listbox selection (or -1 if none)

9.17.3.7 void gslc_ElemXListboxItemsSetGap (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int8_t *nGap*, gslc_tsColor *colGap*)

Configure the gap between listbox items.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nGap</i>	Set the gap between listbox items (0 for none)
in	<i>colGap</i>	Set the color of the gap between listbox items

Returns

none

9.17.3.8 void gslc_ElemXListboxItemsSetSize (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nItemW*, int16_t *nItemH*)

Configure the size of the listbox items.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nItemW</i>	Set the width of a listbox item (or -1 to auto-size)
in	<i>nItemH</i>	Set the height of a listbox item

Returns

none

9.17.3.9 void gslc_ElemXListboxReset (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

Empty the listbox of all items.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update

Returns

none

9.17.3.10 void gslc_ElemXListboxSetMargin (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int8_t *nMarginW*, int8_t *nMarginH*)

Configure the margin inside the listbox.

- Defines the region bewteen the element rect and the inner listbox items

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nMarginW</i>	Set the margin (horizontal) inside the listbox (0 for none)
in	<i>nMarginH</i>	Set the margin (horizontal) inside the listbox (0 for none)

Returns

none

9.17.3.11 bool gslc_ElemXListboxSetScrollPos (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, uint16_t *nScrollPos*)

Set the Listbox scroll position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nScrollPos</i>	Scroll the listbox so that the nScrollPos item is at the top (0 default)

Returns

true if success, false if fail

9.17.3.12 bool gslc_ElemXListboxSetSel (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nItemCurSel*)

Set a Listbox element's current selection.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nItemCurSel</i>	Listbox item to select (or -1 for none)

Returns

true if success, false if fail

9.17.3.13 void `gslc_ElemXListboxSetSelFunc (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, GSLC_CB_XLISTBOX_SEL funcCb)`

Assign the selection callback function for a Listbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>funcCb</i>	Function pointer to selection routine (or NULL for none)

Returns

none

9.17.3.14 void `gslc_ElemXListboxSetSize (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int8_t nRows, int8_t nCols)`

Configure the number of rows & columns to display in the listbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Ptr to Element Reference to update
in	<i>nRows</i>	Number of rows (>= 1, or XLISTBOX_SIZE_AUTO to base on content)
in	<i>nCols</i>	Number of columns (>= 1)

Returns

none

9.17.3.15 bool `gslc_ElemXListboxTouch (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch events to Listbox element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

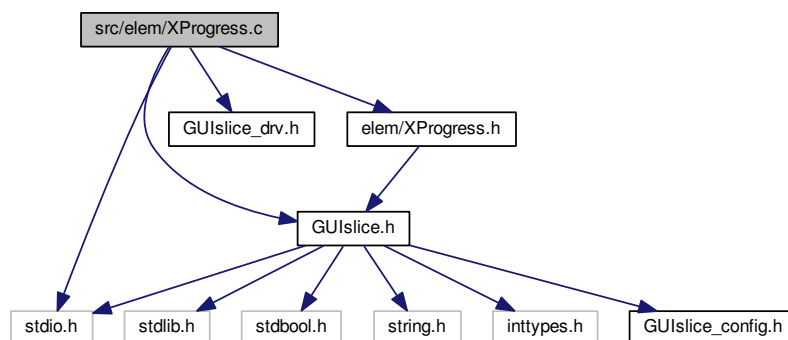
Returns

true if success, false otherwise

9.18 src/elem/XProgress.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XProgress.h"
#include <stdio.h>
```

Include dependency graph for XProgress.c:



Functions

- `gslc_tsElemRef * gslc_ElemXProgressCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsX↔Progress *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)`
Create a Progress Bar Element.
- void `gslc_ElemXProgressSetVal (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)`
Update a Gauge element's current value.
- void `gslc_ElemXProgressSetFlip (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)`
Set a Gauge element's fill direction.
- bool `gslc_ElemXProgressDraw (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
Draw a gauge element on the screen.
- bool `gslc_ElemXProgressDrawHelp (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType e↔Redraw)`
Helper function to draw a gauge with style: progress bar.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.18.1 Function Documentation

9.18.1.1 `gslc_tsElemRef* gslc_ElemXProgressCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage,
gslc_tsXProgress * pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor
colGauge, bool bVert)`

Create a Progress Bar Element.

- Draws a gauge element that represents a proportion (*nVal*) between *nMin* and *nMax*.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for <i>nVal</i> comparison
in	<i>nMax</i>	Maximum value of gauge for <i>nVal</i> comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator
in	<i>bVert</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element reference or NULL if failure

9.18.1.2 `bool gslc_ElemXProgressDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a gauge element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.18.1.3 `bool gslc_ElemXProgressDrawHelp (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_teRedrawType eRedraw)`

Helper function to draw a gauge with style: progress bar.

- Called from [gslc_ElemXProgressDraw\(\)](#)

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pElemRef</i>	Ptr to Element reference
in	<i>eRedraw</i>	Redraw status

Returns

true if success, false otherwise

9.18.1.4 `void gslc_ElemXProgressSetFlip (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bFlip)`

Set a Gauge element's fill direction.

- Setting bFlip reverses the default fill direction
- Default fill direction for horizontal gauges: left-to-right
- Default fill direction for vertical gauges: bottom-to-top

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bFlip</i>	If set, reverse direction of fill from default

Returns

none

9.18.1.5 `void gslc_ElemXProgressSetVal (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nVal)`

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

Returns

none

9.18.2 Variable Documentation

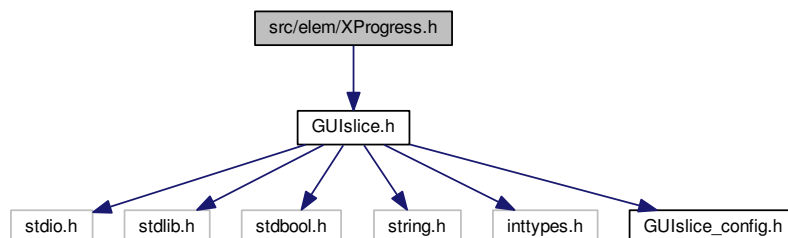
9.18.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.18.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

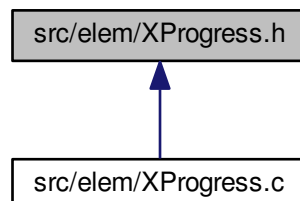
9.19 src/elem/XProgress.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XProgress.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXProgress](#)
Extended data for Gauge element.

Macros

- #define [GSLC_TYPEX_PROGRESS](#)
- #define [gslc_ElemXProgressCreate_P](#)(pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, col↵Frame_, colFill_, colGauge_, bVert_)
Create a Gauge Element in Flash.

Functions

- [gslc_tsElemRef * gslc_ElemXProgressCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsX↵Progress](#) *pXData, [gslc_tsRect](#) rElem, int16_t nMin, int16_t nMax, int16_t nVal, [gslc_tsColor](#) colGauge, bool bVert)
Create a Progress Bar Element.
- void [gslc_ElemXProgressSetVal](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)
Update a Gauge element's current value.
- void [gslc_ElemXProgressSetFlip](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bFlip)
Set a Gauge element's fill direction.
- bool [gslc_ElemXProgressDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a gauge element on the screen.
- bool [gslc_ElemXProgressDrawHelp](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) e↵Redraw)
Helper function to draw a gauge with style: progress bar.

9.19.1 Macro Definition Documentation

- 9.19.1.1 #define [gslc_ElemXProgressCreate_P](#)(*pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, col↵Frame_, colFill_, colGauge_, bVert_*)

Create a Gauge Element in Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>nMin_</i>	Minimum value of gauge for nVal comparison
in	<i>nMax_</i>	Maximum value of gauge for nVal comparison
in	<i>nVal_</i>	Starting value of gauge
in	<i>col↵Frame_</i>	Color for the gauge frame
in	<i>colFill_</i>	Color for the gauge background fill
Generated by Doxygen	<i>colGauge_</i>	Color for the gauge indicator
in	<i>bVert_</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

none

9.19.1.2 #define GSLC_TYPEX_PROGRESS**9.19.2 Function Documentation**

9.19.2.1 `gslc_tsElemRef* gslc_ElemXProgressCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage,
gslc_tsXProgress * pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor
colGauge, bool bVert)`

Create a Progress Bar Element.

- Draws a gauge element that represents a proportion (*nVal*) between *nMin* and *nMax*.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for <i>nVal</i> comparison
in	<i>nMax</i>	Maximum value of gauge for <i>nVal</i> comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator
in	<i>bVert</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element reference or NULL if failure

9.19.2.2 `bool gslc_ElemXProgressDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a gauge element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.19.2.3 `bool gslc_ElemXProgressDrawHelp (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_teRedrawType eRedraw)`

Helper function to draw a gauge with style: progress bar.

- Called from [gslc_ElemXProgressDraw\(\)](#)

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pElemRef</i>	Ptr to Element reference
in	<i>eRedraw</i>	Redraw status

Returns

true if success, false otherwise

9.19.2.4 `void gslc_ElemXProgressSetFlip (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bFlip)`

Set a Gauge element's fill direction.

- Setting bFlip reverses the default fill direction
- Default fill direction for horizontal gauges: left-to-right
- Default fill direction for vertical gauges: bottom-to-top

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bFlip</i>	If set, reverse direction of fill from default

Returns

none

9.19.2.5 `void gslc_ElemXProgressSetVal (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nVal)`

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

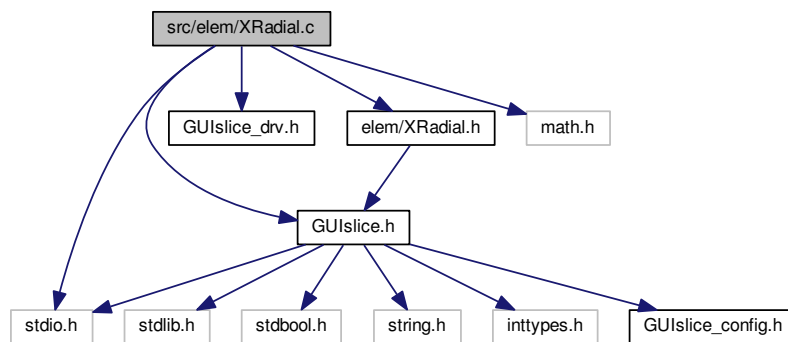
in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

Returns

none

9.20 src/elem/XRadial.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XRadial.h"
#include <stdio.h>
#include <math.h>
Include dependency graph for XRadial.c:
```



Functions

- [gslc_tsElemRef * gslc_ElemXRadialCreate](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nElemId, [int16_t](#) nPage, [gslc_tsXRadial](#) *pXData, [gslc_tsRect](#) rElem, [int16_t](#) nMin, [int16_t](#) nMax, [int16_t](#) nVal, [gslc_tsColor](#) colGauge)
Create a Radial Gauge Element.
- void [gslc_ElemXRadialSetIndicator](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colGauge, [uint16_t](#) nIndicLen, [uint16_t](#) nIndicTip, bool bIndicFill)
Configure the appearance of the Gauge indicator.
- void [gslc_ElemXRadialSetTicks](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colTick, [uint16_t](#) nTickCnt, [uint16_t](#) nTickLen)
Configure the appearance of the Gauge ticks.
- void [gslc_ElemXRadialSetVal](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [int16_t](#) nVal)
Update a Gauge element's current value.
- void [gslc_ElemXRadialSetFlip](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bFlip)
Set a Gauge element's rotation direction.

- bool [gslc_ElemXRadialDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a gauge element on the screen.
- void [gslc_ElemXRadialDrawRadialHelp](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, uint16_t nArrowLen, uint16_t nArrowSz, int16_t n64Ang, bool bFill, [gslc_tsColor](#) colFrame)
- bool [gslc_ElemXRadialDrawRadial](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) eRedraw)
Helper function to draw a gauge with style: radial.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.20.1 Function Documentation

9.20.1.1 [gslc_tsElemRef*](#) [gslc_ElemXRadialCreate](#) ([gslc_tsGui](#) * *pGui*, int16_t *nElemId*, int16_t *nPage*, [gslc_tsXRadial](#) * *pXData*, [gslc_tsRect](#) *rElem*, int16_t *nMin*, int16_t *nMax*, int16_t *nVal*, [gslc_tsColor](#) *colGauge*)

Create a Radial Gauge Element.

- Draws a gauge element that represents a proportion (nVal) between nMin and nMax.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for nVal comparison
in	<i>nMax</i>	Maximum value of gauge for nVal comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator

Returns

Pointer to Element reference or NULL if failure

9.20.1.2 bool [gslc_ElemXRadialDraw](#) (void * *pvGui*, void * *pvElemRef*, [gslc_teRedrawType](#) *eRedraw*)

Draw a gauge element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pVGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.20.1.3 `bool gslc_ElemXRadialDrawRadial (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_teRedrawType eRedraw)`

Helper function to draw a gauge with style: radial.

- Called from [gslc_ElemXRadialDraw\(\)](#)

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pElemRef</i>	Ptr to Element reference
in	<i>eRedraw</i>	Redraw status

Returns

true if success, false otherwise

9.20.1.4 `void gslc_ElemXRadialDrawRadialHelp (gslc_tsGui * pGui, int16_t nX, int16_t nY, uint16_t nArrowLen, uint16_t nArrowSz, int16_t n64Ang, bool bFill, gslc_tsColor colFrame)`

9.20.1.5 `void gslc_ElemXRadialSetFlip (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bFlip)`

Set a Gauge element's rotation direction.

- Setting bFlip reverses the rotation direction
- Default rotation is clockwise. When bFlip is set, uses counter-clockwise

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bFlip</i>	If set, reverse direction of rotation from default

Returns

none

9.20.1.6 void gslc_ElemXRadialSetIndicator (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colGauge*, uint16_t *nIndicLen*, uint16_t *nIndicTip*, bool *blndicFill*)

Configure the appearance of the Gauge indicator.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colGauge</i>	Color of the indicator
in	<i>nIndicLen</i>	Length of the indicator
in	<i>nIndicTip</i>	Size of the indicator tip
in	<i>blndicFill</i>	Fill in the indicator if true

Returns

none

9.20.1.7 void gslc_ElemXRadialSetTicks (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colTick*, uint16_t *nTickCnt*, uint16_t *nTickLen*)

Configure the appearance of the Gauge ticks.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colTick</i>	Color of the gauge ticks
in	<i>nTickCnt</i>	Number of ticks to draw around / along gauge
in	<i>nTickLen</i>	Length of the tick marks to draw

Returns

none

9.20.1.8 void gslc_ElemXRadialSetVal (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nVal*)

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

Returns

none

9.20.2 Variable Documentation

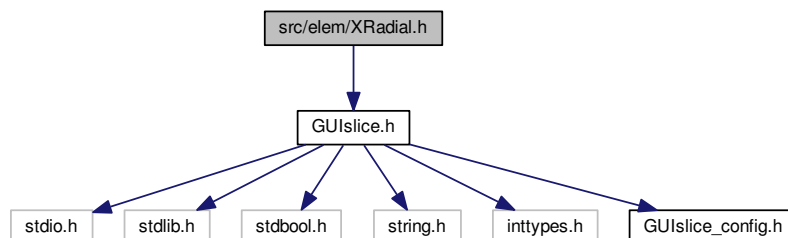
9.20.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.20.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

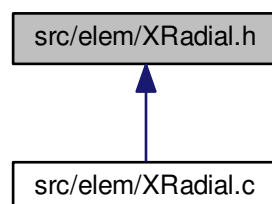
9.21 src/elem/XRadial.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XRadial.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXRadial](#)
Extended data for Gauge element.

Macros

- #define [GSLC_TYPEX_RADIAL](#)
- #define [gslc_ElemXRadialCreate_P](#)(pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, colFrame_, colFill_, colGauge_)
Create a Gauge Element in Flash.

Functions

- [gslc_tsElemRef * gslc_ElemXRadialCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXRadial](#) *pXData, [gslc_tsRect](#) rElem, int16_t nMin, int16_t nMax, int16_t nVal, [gslc_tsColor](#) colGauge)
Create a Radial Gauge Element.
- void [gslc_ElemXRadialSetIndicator](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)
Configure the appearance of the Gauge indicator.
- void [gslc_ElemXRadialSetTicks](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colTick, uint16_t nTickCnt, uint16_t nTickLen)
Configure the appearance of the Gauge ticks.
- void [gslc_ElemXRadialSetVal](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)
Update a Gauge element's current value.
- void [gslc_ElemXRadialSetFlip](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bFlip)
Set a Gauge element's rotation direction.
- bool [gslc_ElemXRadialDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a gauge element on the screen.
- bool [gslc_ElemXRadialDrawRadial](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) eRedraw)
Helper function to draw a gauge with style: radial.

9.21.1 Macro Definition Documentation

- 9.21.1.1 #define [gslc_ElemXRadialCreate_P](#)(*pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, colFrame_, colFill_, colGauge_*)

Create a Gauge Element in Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element

Parameters

in	<i>nMin_</i>	Minimum value of gauge for nVal comparison
in	<i>nMax_</i>	Maximum value of gauge for nVal comparison
in	<i>nVal_</i>	Starting value of gauge
in	<i>col↔ Frame_</i>	Color for the gauge frame
in	<i>colFill_</i>	Color for the gauge background fill
in	<i>col↔ Gauge_</i>	Color for the gauge indicator

Returns

none

9.21.1.2 #define GSLC_TYPEX_RADIAL**9.21.2 Function Documentation**

9.21.2.1 `gslc_tsElemRef* gslc_ElemXRadialCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXRadial * pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge)`

Create a Radial Gauge Element.

- Draws a gauge element that represents a proportion (nVal) between nMin and nMax.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for nVal comparison
in	<i>nMax</i>	Maximum value of gauge for nVal comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator

Returns

Pointer to Element reference or NULL if failure

9.21.2.2 `bool gslc_ElemXRadialDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a gauge element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pVGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.21.2.3 `bool gslc_ElemXRadialDrawRadial (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_teRedrawType eRedraw)`

Helper function to draw a gauge with style: radial.

- Called from [gslc_ElemXRadialDraw\(\)](#)

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pElemRef</i>	Ptr to Element reference
in	<i>eRedraw</i>	Redraw status

Returns

true if success, false otherwise

9.21.2.4 `void gslc_ElemXRadialSetFlip (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bFlip)`

Set a Gauge element's rotation direction.

- Setting `bFlip` reverses the rotation direction
- Default rotation is clockwise. When `bFlip` is set, uses counter-clockwise

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bFlip</i>	If set, reverse direction of rotation from default

Returns

none

9.21.2.5 void gslc_ElemXRadialSetIndicator (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colGauge*, uint16_t *nIndicLen*, uint16_t *nIndicTip*, bool *blndicFill*)

Configure the appearance of the Gauge indicator.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colGauge</i>	Color of the indicator
in	<i>nIndicLen</i>	Length of the indicator
in	<i>nIndicTip</i>	Size of the indicator tip
in	<i>blndicFill</i>	Fill in the indicator if true

Returns

none

9.21.2.6 void gslc_ElemXRadialSetTicks (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colTick*, uint16_t *nTickCnt*, uint16_t *nTickLen*)

Configure the appearance of the Gauge ticks.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colTick</i>	Color of the gauge ticks
in	<i>nTickCnt</i>	Number of ticks to draw around / along gauge
in	<i>nTickLen</i>	Length of the tick marks to draw

Returns

none

9.21.2.7 void gslc_ElemXRadialSetVal (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nVal*)

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

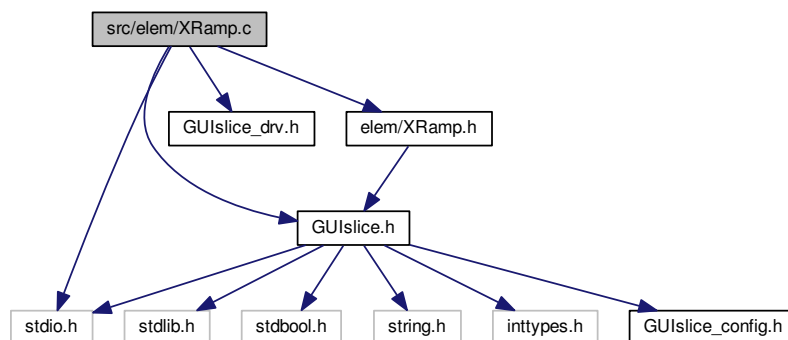
Returns

none

9.22 src/elem/XRamp.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XRamp.h"
#include <stdio.h>
```

Include dependency graph for XRamp.c:



Functions

- [gslc_tsElemRef](#) * [gslc_ElemXRampCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXRamp](#) *pXData, [gslc_tsRect](#) rElem, int16_t nMin, int16_t nMax, int16_t nVal, [gslc_tsColor](#) colGauge, bool bVert)
Create a Ramp Gauge Element.
- void [gslc_ElemXRampSetVal](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)
Update a Gauge element's current value.
- bool [gslc_ElemXRampDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a gauge element on the screen.
- bool [gslc_ElemXRampDrawHelp](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) eRedraw)
Helper function to draw a gauge with style: ramp.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.22.1 Function Documentation

9.22.1.1 `gslc_tsElemRef* gslc_ElemXRampCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXRamp * pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)`

Create a Ramp Gauge Element.

- Draws a gauge element that represents a proportion (*nVal*) between *nMin* and *nMax*.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for <i>nVal</i> comparison
in	<i>nMax</i>	Maximum value of gauge for <i>nVal</i> comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator
in	<i>bVert</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element reference or NULL if failure

9.22.1.2 `bool gslc_ElemXRampDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a gauge element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.22.1.3 `bool gslc_ElemXRampDrawHelp (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_teRedrawType eRedraw)`

Helper function to draw a gauge with style: ramp.

- Called from [gslc_ElemXRampDraw\(\)](#)

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pElemRef</i>	Ptr to Element reference
in	<i>eRedraw</i>	Redraw status

Returns

true if success, false otherwise

9.22.1.4 `void gslc_ElemXRampSetVal (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nVal)`

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

Returns

none

9.22.2 Variable Documentation

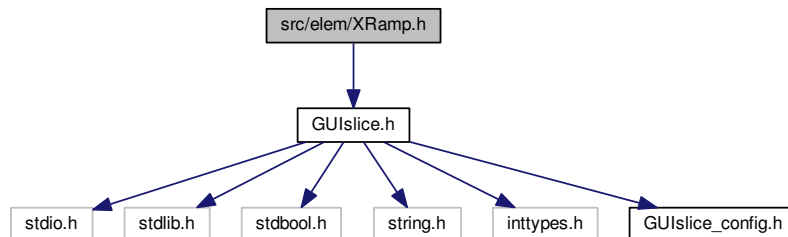
9.22.2.1 `const char GSLC_PMEM_ERRSTR_NULL[]`

9.22.2.2 `const char GSLC_PMEM_ERRSTR_PXD_NULL[]`

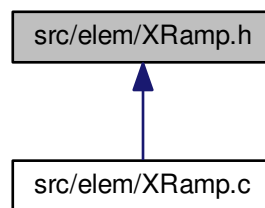
9.23 src/elem/XRamp.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XRamp.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXRamp](#)
Extended data for Gauge element.

Macros

- #define [GSLC_TYPEX_RAMP](#)
- #define [gslc_ElemXRampCreate_P](#)(pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, col↵
Frame_, colFill_)
Create a Gauge Element in Flash.

Functions

- `gslc_tsElemRef * gslc_ElemXRampCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXRamp *pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)`
Create a Ramp Gauge Element.
- `void gslc_ElemXRampSetVal (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)`
Update a Gauge element's current value.
- `bool gslc_ElemXRampDraw (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
Draw a gauge element on the screen.
- `bool gslc_ElemXRampDrawHelp (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)`
Helper function to draw a gauge with style: ramp.

9.23.1 Macro Definition Documentation

9.23.1.1 `#define gslc_ElemXRampCreate_P(pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, colFrame_, colFill_)`

Create a Gauge Element in Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>nMin_</i>	Minimum value of gauge for nVal comparison
in	<i>nMax_</i>	Maximum value of gauge for nVal comparison
in	<i>nVal_</i>	Starting value of gauge
in	<i>colFrame_</i>	Color for the gauge frame
in	<i>colFill_</i>	Color for the gauge background fill

Returns

none

9.23.1.2 `#define GSLC_TYPEX_RAMP`

9.23.2 Function Documentation

9.23.2.1 `gslc_tsElemRef* gslc_ElemXRampCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXRamp * pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, gslc_tsColor colGauge, bool bVert)`

Create a Ramp Gauge Element.

- Draws a gauge element that represents a proportion (nVal) between nMin and nMax.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for nVal comparison
in	<i>nMax</i>	Maximum value of gauge for nVal comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator
in	<i>bVert</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element reference or NULL if failure

9.23.2.2 bool gslc_ElemXRampDraw (void * *pvGui*, void * *pvElemRef*, gslc_teRedrawType *eRedraw*)

Draw a gauge element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.23.2.3 bool gslc_ElemXRampDrawHelp (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_teRedrawType *eRedraw*)

Helper function to draw a gauge with style: ramp.

- Called from [gslc_ElemXRampDraw\(\)](#)

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pElemRef</i>	Ptr to Element reference
in	<i>eRedraw</i>	Redraw status

Returns

true if success, false otherwise

9.23.2.4 void gslc_ElemXRampSetVal (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nVal*)

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

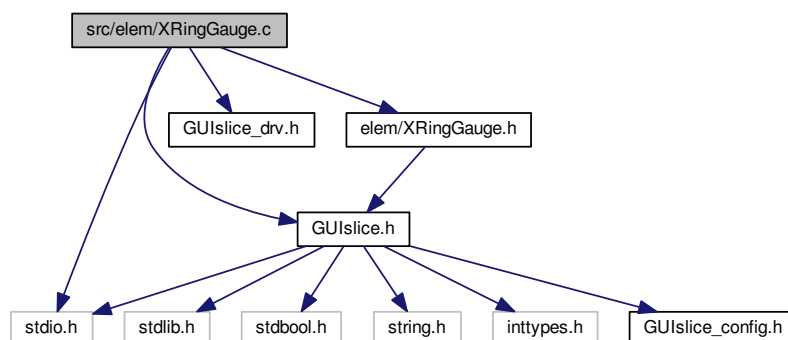
in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

Returns

none

9.24 src/elem/XRingGauge.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XRingGauge.h"
#include <stdio.h>
Include dependency graph for XRingGauge.c:
```

**Functions**

- [gslc_tsElemRef * gslc_ElemXRingGaugeCreate](#) (gslc_tsGui **pGui*, int16_t *nElemId*, int16_t *nPage*, [gslc_tsXRingGauge](#) **pXData*, [gslc_tsRect](#) *rElem*, char **pStrBuf*, uint8_t *nStrBufMax*, int16_t *nFontId*)

- Create an XRingGauge element.*
- bool [gslc_ElemXRingGaugeDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
- Draw the template element on the screen.*
- void [gslc_ElemXRingGaugeSetVal](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)
- Set an Ring Gauge current indicator value.*
- void [gslc_ElemXRingGaugeSetValRange](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nValMin, int16_t nValMax)
- Defines the range of values that may be passed into SetVal(), used to scale the input to SetVal().*
- void [gslc_ElemXRingGaugeSetAngleRange](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nStart, int16_t nRange, bool bClockwise)
- Defines the angular range of the gauge, including both the active and inactive regions.*
- void [gslc_ElemXRingGaugeSetThickness](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int8_t nThickness)
- Defines the thickness of the ring arcs.*
- void [gslc_ElemXRingGaugeSetColorActiveFlat](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colActive)
- Defines the color of the active region to be a flat (constant) color.*
- void [gslc_ElemXRingGaugeSetColorActiveGradient](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colStart, [gslc_tsColor](#) colEnd)
- Defines the color of the active region to be a gradient using two color stops.*
- void [gslc_ElemXRingGaugeSetColorInactive](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colInactive)
- Defines the color of the inactive region to be a flat (constant) color.*
- void [gslc_ElemXRingGaugeSetQuality](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint16_t nSegments)
- Sets the quality of the ring drawing by defining the number of segments that are used when rendering a 360 degree gauge. The larger the number, the more segments are used and the smoother the curve.*

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.24.1 Function Documentation

- 9.24.1.1 [gslc_tsElemRef* gslc_ElemXRingGaugeCreate](#) ([gslc_tsGui](#) * pGui, int16_t nElemId, int16_t nPage, [gslc_tsXRingGauge](#) * pXData, [gslc_tsRect](#) rElem, char * pStrBuf, uint8_t nStrBufMax, int16_t nFontId)

Create an XRingGauge element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	The square box that bounds the ring element. If a rectangular region is provided, then the ring control will be centered in the long axis.
in	<i>pStrBuf</i>	String buffer to use for gauge inner text
in	<i>nStrBufMax</i>	Maximum length of string buffer (pStrBuf)
in	<i>nFontId</i>	Font ID to use for text display

Returns

Pointer to Element reference or NULL if failure

9.24.1.2 `bool gslc_ElemXRingGaugeDraw (void * pvGui, void * pElemRef, gslc_teRedrawType eRedraw)`

Draw the template element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.24.1.3 `void gslc_ElemXRingGaugeSetAngleRange (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nStart, int16_t nRange, bool bClockwise)`

Defines the angular range of the gauge, including both the active and inactive regions.

- *nStart* defines the angle at the beginning of the active region.
- The current position marks the end of the active region and the beginning of the inactive region.
- *nRange* defines the angular range from the start of the active region to the end of the inactive region. In most cases, a range of 360 degrees is used.
- All angles are measured in units of degrees.
- Angles are measured with 0 at the top, 90 towards the right, 180 towards the bottom, 270 towards the left, etc.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nStart</i>	Define angle of start of active region (measured in degrees)
in	<i>nRange</i>	Define angular range from strt of active region to end of the inactive region (measured in degrees)
in	<i>bClockwise</i>	Defines the direction in which the active region grows (true for clockwise) [FORCED TRUE, FOR FUTURE IMPLEMENTATION]

Returns

none

9.24.1.4 void gslc_ElemXRingGaugeSetColorActiveFlat (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colActive*)

Defines the color of the active region to be a flat (constant) color.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colActive</i>	Color of active region

Returns

none

9.24.1.5 void gslc_ElemXRingGaugeSetColorActiveGradient (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colStart*, gslc_tsColor *colEnd*)

Defines the color of the active region to be a gradient using two color stops.

The active region will be filled according to the proportion between nMin and nMax. The gradient is defined by a linear RGB blend between the two color stops(*colStart* and *colEnd*)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colStart</i>	Starting color of gradient fill
in	<i>colEnd</i>	Ending color of gradient fill

Returns

none

9.24.1.6 void gslc_ElemXRingGaugeSetColorInactive (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colInactive*)

Defines the color of the inactive region to be a flat (constant) color.

The inactive color is often set to be the same as the background but it can be set to a different color to indicate the remainder of the value range that is yet to be filled.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>collnactive</i>	Color of inactive region

Returns

none

9.24.1.7 void gslc_ElemXRingGaugeSetQuality (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, uint16_t *nSegments*)

Sets the quality of the ring drawing by defining the number of segments that are used when rendering a 360 degree gauge. The larger the number, the more segments are used and the smoother the curve.

A larger ring gauge may need a higher quality number to maintain a smoothed curve appearance.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nSegments</i>	Number of arc segments to render a complete circle. The higher the value, the smoother the ring. Note that 360/nSegments should be an integer result, thus the allowable quality settings are: 360 (max quality), 180, 120, 90, 72, 60, 45, 40, 36 (low quality), etc.

Returns

none

9.24.1.8 void gslc_ElemXRingGaugeSetThickness (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int8_t *nThickness*)

Defines the thickness of the ring arcs.

More specifically, it defines the reduction in radius from the outer radius to the inner radius in pixels.

- Default thickness is 10 pixels

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nThickness</i>	Thickness of ring

Returns

none

9.24.1.9 void gslc_ElemXRingGaugeSetVal (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nVal*)

Set an Ring Gauge current indicator value.

Updates the current value of the ring gauge. The active region will be drawn up to the position defined by *nVal* within the value range defined by SetValRange(*nMin*,*nMax*). A SetVal() close to *nMin* will cause a very small active region to be drawn and a large remainder drawn in the inactive color, whereas a SetVal() close to *nMax* will cause a more complete active region to be drawn. When SetVal() equals *nMax*, the entire angular range will be drawn in the active color (and no inactive region).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New position value

Returns

none

9.24.1.10 void gslc_ElemXRingGaugeSetValRange (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nValMin*, int16_t *nValMax*)

Defines the range of values that may be passed into SetVal(), used to scale the input to SetVal().

- Default is 0..100.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nValMin</i>	Minimum value
in	<i>nValMax</i>	Maximum value

Returns

none

9.24.2 Variable Documentation

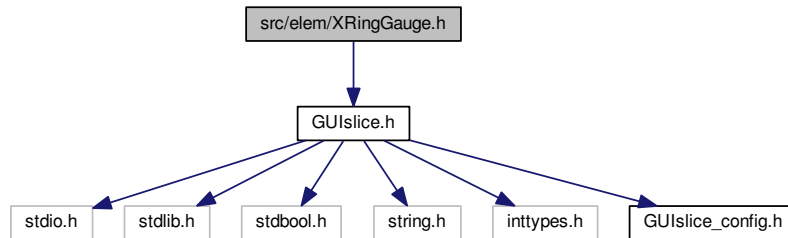
9.24.2.1 const char GSLC_PMEM_ERRSTR_NULL[]

9.24.2.2 const char GSLC_PMEM_ERRSTR_PXD_NULL[]

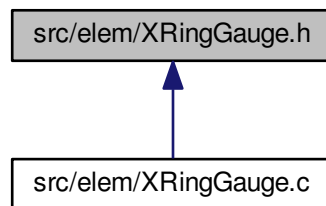
9.25 src/elem/XRingGauge.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XRingGauge.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXRingGauge](#)
Extended data for XRingGauge element.

Macros

- #define [GSLC_TYPEX_RING](#)
- #define [XRING_STR_MAX](#)

Functions

- [gslc_tsElemRef](#) * [gslc_ElemXRingGaugeCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXRingGauge](#) *pXData, [gslc_tsRect](#) rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)
 - bool [gslc_ElemXRingGaugeDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
- Create an XRingGauge element.*

Draw the template element on the screen.

- void [gslc_ElemXRingGaugeSetVal](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)

Set an Ring Gauge current indicator value.

- void [gslc_ElemXRingGaugeSetAngleRange](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nStart, int16_t nRange, bool bClockwise)

Defines the angular range of the gauge, including both the active and inactive regions.

- void [gslc_ElemXRingGaugeSetValRange](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nValMin, int16_t nValMax)

Defines the range of values that may be passed into SetVal(), used to scale the input to SetVal().

- void [gslc_ElemXRingGaugeSetThickness](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int8_t nThickness)

Defines the thickness of the ring arcs.

- void [gslc_ElemXRingGaugeSetQuality](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint16_t nSegments)

Sets the quality of the ring drawing by defining the number of segments that are used when rendering a 360 degree gauge. The larger the number, the more segments are used and the smoother the curve.

- void [gslc_ElemXRingGaugeSetColorInactive](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colInactive)

Defines the color of the inactive region to be a flat (constant) color.

- void [gslc_ElemXRingGaugeSetColorActiveFlat](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colActive)

Defines the color of the active region to be a flat (constant) color.

- void [gslc_ElemXRingGaugeSetColorActiveGradient](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colStart, [gslc_tsColor](#) colEnd)

Defines the color of the active region to be a gradient using two color stops.

9.25.1 Macro Definition Documentation

9.25.1.1 #define GSLC_TYPEX_RING

9.25.1.2 #define XRING_STR_MAX

9.25.2 Function Documentation

9.25.2.1 [gslc_tsElemRef*](#) [gslc_ElemXRingGaugeCreate](#) ([gslc_tsGui](#) * pGui, int16_t nElemId, int16_t nPage, [gslc_tsXRingGauge](#) * pXData, [gslc_tsRect](#) rElem, char * pStrBuf, uint8_t nStrBufMax, int16_t nFontId)

Create an XRingGauge element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	The square box that bounds the ring element. If a rectangular region is provided, then the ring control will be centered in the long axis.
in	<i>pStrBuf</i>	String buffer to use for gauge inner text
in	<i>nStrBufMax</i>	Maximum length of string buffer (pStrBuf)
in	<i>nFontId</i>	Font ID to use for text display

Returns

Pointer to Element reference or NULL if failure

9.25.2.2 `bool gslc_ElemXRingGaugeDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw the template element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.25.2.3 `void gslc_ElemXRingGaugeSetAngleRange (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, int16_t nStart, int16_t nRange, bool bClockwise)`

Defines the angular range of the gauge, including both the active and inactive regions.

- *nStart* defines the angle at the beginning of the active region.
- The current position marks the end of the active region and the beginning of the inactive region.
- *nRange* defines the angular range from the start of the active region to the end of the inactive region. In most cases, a range of 360 degrees is used.
- All angles are measured in units of degrees.
- Angles are measured with 0 at the top, 90 towards the right, 180 towards the bottom, 270 towards the left, etc.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nStart</i>	Define angle of start of active region (measured in degrees)
in	<i>nRange</i>	Define angular range from strt of active region to end of the inactive region (measured in degrees)
in	<i>bClockwise</i>	Defines the direction in which the active region grows (true for clockwise) [FORCED TRUE, FOR FUTURE IMPLEMENTATION]

Returns

none

9.25.2.4 void gslc_ElemXRingGaugeSetColorActiveFlat (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colActive*)

Defines the color of the active region to be a flat (constant) color.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colActive</i>	Color of active region

Returns

none

9.25.2.5 void gslc_ElemXRingGaugeSetColorActiveGradient (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colStart*, gslc_tsColor *colEnd*)

Defines the color of the active region to be a gradient using two color stops.

The active region will be filled according to the proportion between nMin and nMax. The gradient is defined by a linear RGB blend between the two color stops(*colStart* and *colEnd*)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>colStart</i>	Starting color of gradient fill
in	<i>colEnd</i>	Ending color of gradient fill

Returns

none

9.25.2.6 void gslc_ElemXRingGaugeSetColorInactive (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_tsColor *colInactive*)

Defines the color of the inactive region to be a flat (constant) color.

The inactive color is often set to be the same as the background but it can be set to a different color to indicate the remainder of the value range that is yet to be filled.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>collnactive</i>	Color of inactive region

Returns

none

9.25.2.7 void gslc_ElemXRingGaugeSetQuality (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, uint16_t *nSegments*)

Sets the quality of the ring drawing by defining the number of segments that are used when rendering a 360 degree gauge. The larger the number, the more segments are used and the smoother the curve.

A larger ring gauge may need a higher quality number to maintain a smoothed curve appearance.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nSegments</i>	Number of arc segments to render a complete circle. The higher the value, the smoother the ring. Note that 360/nSegments should be an integer result, thus the allowable quality settings are: 360 (max quality), 180, 120, 90, 72, 60, 45, 40, 36 (low quality), etc.

Returns

none

9.25.2.8 void gslc_ElemXRingGaugeSetThickness (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int8_t *nThickness*)

Defines the thickness of the ring arcs.

More specifically, it defines the reduction in radius from the outer radius to the inner radius in pixels.

- Default thickness is 10 pixels

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nThickness</i>	Thickness of ring

Returns

none

9.25.2.9 void gslc_ElemXRingGaugeSetVal (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nVal*)

Set an Ring Gauge current indicator value.

Updates the current value of the ring gauge. The active region will be drawn up to the position defined by *nVal* within the value range defined by SetValRange(*nMin*,*nMax*). A SetVal() close to *nMin* will cause a very small active region to be drawn and a large remainder drawn in the inactive color, whereas a SetVal() close to *nMax* will cause a more complete active region to be drawn. When SetVal() equals *nMax*, the entire angular range will be drawn in the active color (and no inactive region).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New position value

Returns

none

9.25.2.10 void gslc_ElemXRingGaugeSetValRange (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nValMin*, int16_t *nValMax*)

Defines the range of values that may be passed into SetVal(), used to scale the input to SetVal().

- Default is 0..100.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nValMin</i>	Minimum value
in	<i>nValMax</i>	Maximum value

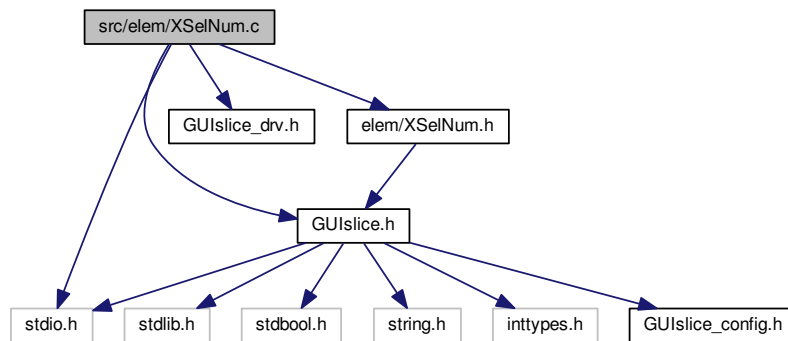
Returns

none

9.26 src/elem/XSelNum.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XSelNum.h"
#include <stdio.h>
```

Include dependency graph for XSelNum.c:



Functions

- `gslc_tsElemRef * gslc_ElemXSelNumCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSelNum *pXData, gslc_tsRect rElem, int8_t nFontId)`
Create a SelNum Element.
- `bool gslc_ElemXSelNumDraw (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
Draw a SelNum element on the screen.
- `int gslc_ElemXSelNumGetCounter (gslc_tsGui *pGui, gslc_tsXSelNum *pSelNum)`
Get the current counter associated with SelNum.
- `void gslc_ElemXSelNumSetCounter (gslc_tsGui *pGui, gslc_tsXSelNum *pSelNum, int16_t nCount)`
Set the current counter associated with SelNum.
- `bool gslc_ElemXSelNumClick (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
Handle a click event within the SelNum.
- `bool gslc_ElemXSelNumTouch (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`
Handle touch (up,down,move) events to SelNum element.

Variables

- `const char GSLC_PMEM_ERRSTR_NULL []`
- `const char GSLC_PMEM_ERRSTR_PXD_NULL []`
- `static const int16_t SELNUM_ID_BTN_INC`
- `static const int16_t SELNUM_ID_BTN_DEC`
- `static const int16_t SELNUM_ID_TXT`

9.26.1 Function Documentation

9.26.1.1 `bool gslc_ElemXSelNumClick (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

Handle a click event within the SelNum.

- This is called internally by the SelNum touch handler

Parameters

in	<i>pVGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pVElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nX</i>	Touch X coord
in	<i>nY</i>	Touch Y coord

Returns

none

9.26.1.2 `gslc_tsElemRef* gslc_ElemXSelNumCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXSelNum * pXData, gslc_tsRect rElem, int8_t nFontId)`

Create a SelNum Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or <code>GSLC_ID_AUTO</code> to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining element size
in	<i>nFontId</i>	Font ID to use for drawing the element

Returns

Pointer to Element or NULL if failure

9.26.1.3 `bool gslc_ElemXSelNumDraw (void * pVGui, void * pVElemRef, gslc_teRedrawType eRedraw)`

Draw a SelNum element on the screen.

- Called during redraw

Parameters

in	<i>pVGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pVElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.26.1.4 `int gslc_ElemXSelNumGetCounter (gslc_tsGui * pGui, gslc_tsXSelNum * pSelNum)`

Get the current counter associated with SelNum.

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pSelNum</i>	Ptr to Element

Returns

Current counter value

9.26.1.5 `void gslc_ElemXSelNumSetCounter (gslc_tsGui * pGui, gslc_tsXSelNum * pSelNum, int16_t nCount)`

Set the current counter associated with SelNum.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pSelNum</i>	Ptr to Element
in	<i>nCount</i>	New counter value

Returns

none

9.26.1.6 `bool gslc_ElemXSelNumTouch (void * pGui, void * pElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch (up,down,move) events to SelNum element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pElemRef</i>	Void ptr to Element ref (typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

Returns

true if success, false otherwise

9.26.2 Variable Documentation

9.26.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.26.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

9.26.2.3 `const int16_t SELNUM_ID_BTN_DEC` `[static]`

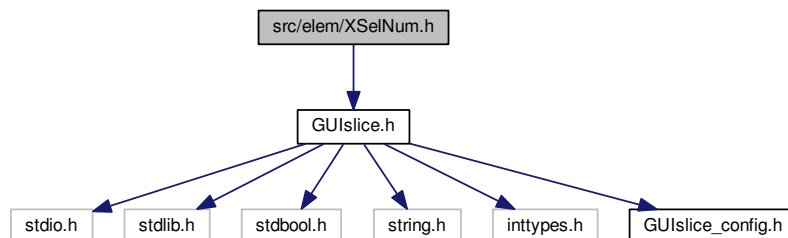
9.26.2.4 `const int16_t SELNUM_ID_BTN_INC` `[static]`

9.26.2.5 `const int16_t SELNUM_ID_TXT` `[static]`

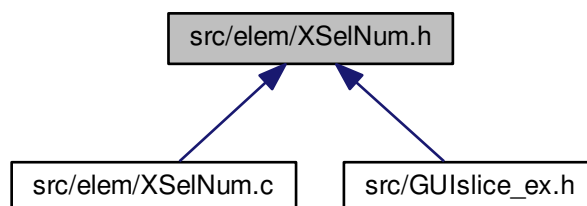
9.27 src/elem/XSelNum.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XSelNum.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXSelNum](#)

Extended data for SelNum element.

Macros

- #define [GSLC_TYPEX_SELNUM](#)
- #define [SELNUM_STR_LEN](#)

Functions

- [gslc_tsElemRef * gslc_ElemXSelNumCreate](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nElemId, [int16_t](#) nPage, [gslc_tsXSelNum](#) *pXData, [gslc_tsRect](#) rElem, [int8_t](#) nFontId)
Create a SelNum Element.
- [bool gslc_ElemXSelNumDraw](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a SelNum element on the screen.
- [int gslc_ElemXSelNumGetCounter](#) ([gslc_tsGui](#) *pGui, [gslc_tsXSelNum](#) *pSelNum)
Get the current counter associated with SelNum.
- [void gslc_ElemXSelNumSetCounter](#) ([gslc_tsGui](#) *pGui, [gslc_tsXSelNum](#) *pSelNum, [int16_t](#) nCount)
Set the current counter associated with SelNum.
- [bool gslc_ElemXSelNumClick](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teTouch](#) eTouch, [int16_t](#) nX, [int16_t](#) nY)
Handle a click event within the SelNum.
- [bool gslc_ElemXSelNumTouch](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teTouch](#) eTouch, [int16_t](#) nRelX, [int16_t](#) nRelY)
Handle touch (up,down,move) events to SelNum element.

9.27.1 Macro Definition Documentation

9.27.1.1 #define GSLC_TYPEX_SELNUM

9.27.1.2 #define SELNUM_STR_LEN

9.27.2 Function Documentation

9.27.2.1 bool gslc_ElemXSelNumClick (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)

Handle a click event within the SelNum.

- This is called internally by the SelNum touch handler

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nX</i>	Touch X coord
in	<i>nY</i>	Touch Y coord

Returns

none

9.27.2.2 `gslc_tsElemRef* gslc_ElemXSelNumCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage,
gslc_tsXSelNum * pXData, gslc_tsRect rElem, int8_t nFontId)`

Create a SelNum Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining element size
in	<i>nFontId</i>	Font ID to use for drawing the element

Returns

Pointer to Element or NULL if failure

9.27.2.3 `bool gslc_ElemXSelNumDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a SelNum element on the screen.

- Called during redraw

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.27.2.4 `int gslc_ElemXSelNumGetCounter (gslc_tsGui * pGui, gslc_tsXSelNum * pSelNum)`

Get the current counter associated with SelNum.

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pSelNum</i>	Ptr to Element

Returns

Current counter value

9.27.2.5 void gslc_ElemXSelNumSetCounter (gslc_tsGui * *pGui*, gslc_tsXSelNum * *pSelNum*, int16_t *nCount*)

Set the current counter associated with SelNum.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pSelNum</i>	Ptr to Element
in	<i>nCount</i>	New counter value

Returns

none

9.27.2.6 bool gslc_ElemXSelNumTouch (void * *pvGui*, void * *pvElemRef*, gslc_teTouch *eTouch*, int16_t *nRelX*, int16_t *nRelY*)

Handle touch (up,down,move) events to SelNum element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

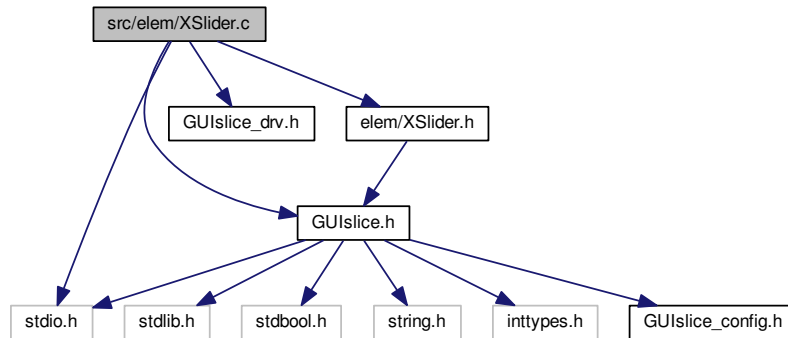
Returns

true if success, false otherwise

9.28 src/elem/XSlider.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XSlider.h"
#include <stdio.h>
```

Include dependency graph for XSlider.c:



Functions

- `gslc_tsElemRef * gslc_ElemXSliderCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSlider *pXData, gslc_tsRect rElem, int16_t nPosMin, int16_t nPosMax, int16_t nPos, uint16_t nThumbSz, bool bVert)`
Create a Slider Element.
- `void gslc_ElemXSliderSetStyle (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bTrim, gslc_tsColor colTrim, uint16_t nTickDiv, int16_t nTickLen, gslc_tsColor colTick)`
Set a Slider element's current position.
- `int gslc_ElemXSliderGetPos (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
Get a Slider element's current position.
- `void gslc_ElemXSliderSetPos (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nPos)`
Set a Slider element's current position.
- `void gslc_ElemXSliderSetPosFunc (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XSLIDER_POS funcCb)`
Assign the position callback function for a slider.
- `bool gslc_ElemXSliderDraw (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
Draw a Slider element on the screen.
- `bool gslc_ElemXSliderTouch (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`
Handle touch events to Slider element.

Variables

- `const char GSLC_PMEM ERRSTR_NULL []`
- `const char GSLC_PMEM ERRSTR_PXD_NULL []`

9.28.1 Function Documentation

- 9.28.1.1 `gslc_tsElemRef* gslc_ElemXSliderCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXSlider *pXData, gslc_tsRect rElem, int16_t nPosMin, int16_t nPosMax, int16_t nPos, uint16_t nThumbSz, bool bVert)`

Create a Slider Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining checkbox size
in	<i>nPosMin</i>	Minimum position value
in	<i>nPosMax</i>	Maximum position value
in	<i>nPos</i>	Starting position value
in	<i>nThumbSz</i>	Size of the thumb control
in	<i>bVert</i>	Orientation (true for vertical)

Returns

Pointer to Element reference or NULL if failure

9.28.1.2 `bool gslc_ElemXSliderDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a Slider element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.28.1.3 `int gslc_ElemXSliderGetPos (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Get a Slider element's current position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Current slider position

9.28.1.4 void gslc_ElemXSliderSetPos (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nPos*)

Set a Slider element's current position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nPos</i>	New position value

Returns

none

9.28.1.5 void gslc_ElemXSliderSetPosFunc (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, GSLC_CB_XSLIDER_POS *funcCb*)

Assign the position callback function for a slider.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>funcCb</i>	Function pointer to position routine (or NULL for none)

Returns

none

9.28.1.6 void gslc_ElemXSliderSetStyle (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bTrim*, gslc_tsColor *colTrim*, uint16_t *nTickDiv*, int16_t *nTickLen*, gslc_tsColor *colTick*)

Set a Slider element's current position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bTrim</i>	Show a colored trim?
in	<i>colTrim</i>	Color of trim
in	<i>nTickDiv</i>	Number of tick divisions to show (0 for none)
in	<i>nTickLen</i>	Length of tickmarks
in	<i>colTick</i>	Color of ticks

Returns

none

9.28.1.7 `bool gslc_ElemXSliderTouch (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch events to Slider element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

Returns

true if success, false otherwise

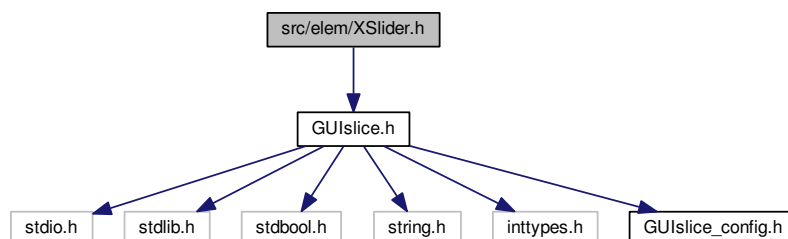
9.28.2 Variable Documentation

9.28.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

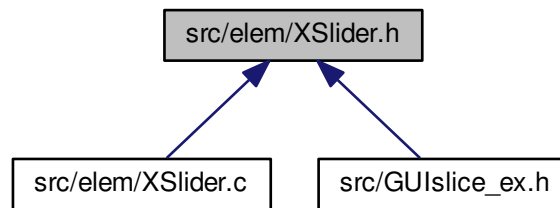
9.28.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

9.29 src/elem/XSlider.h File Reference

`#include "GUIslice.h"`
 Include dependency graph for XSlider.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXSlider](#)
Extended data for Slider element.

Macros

- #define [GSLC_TYPEX_SLIDER](#)
- #define [gslc_ElemXSliderCreate_P](#)(pGui, nElemId, nPage, nX, nY, nW, nH, nPosMin_, nPosMax_, nPos_, nThumbSz_, bVert_, colFrame_, colFill_)
Create a Slider Element in Flash.

Typedefs

- typedef bool(* [GSLC_CB_XSLIDER_POS](#))(void *pvGui, void *pvElem, int16_t nPos)
Callback function for slider feedback.

Functions

- [gslc_tsElemRef](#) * [gslc_ElemXSliderCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXSlider](#) *pXData, [gslc_tsRect](#) rElem, int16_t nPosMin, int16_t nPosMax, int16_t nPos, uint16_t nThumbSz, bool bVert)
Create a Slider Element.
- void [gslc_ElemXSliderSetStyle](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bTrim, [gslc_tsColor](#) colTrim, uint16_t nTickDiv, int16_t nTickLen, [gslc_tsColor](#) colTick)
Set a Slider element's current position.
- int [gslc_ElemXSliderGetPos](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get a Slider element's current position.
- void [gslc_ElemXSliderSetPos](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nPos)
Set a Slider element's current position.
- void [gslc_ElemXSliderSetPosFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_XSLIDER_POS](#) funcCb)
Assign the position callback function for a slider.
- bool [gslc_ElemXSliderDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a Slider element on the screen.
- bool [gslc_ElemXSliderTouch](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY)
Handle touch events to Slider element.

9.29.1 Macro Definition Documentation

9.29.1.1 `#define gslc_ElemXSliderCreate_P(pGui, nElemId, nPage, nX, nY, nW, nH, nPosMin_, nPosMax_, nPos_, nThumbSz_, bVert_, colFrame_, colFill_)`

Create a Slider Element in Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>nPosMin</i> ↔ —	Minimum position value
in	<i>nPosMax</i> ↔ —	Maximum position value
in	<i>nPos_</i>	Starting position value
in	<i>nThumb</i> ↔ <i>Sz_</i>	Size of the thumb control
in	<i>bVert_</i>	Orientation (true for vertical)
in	<i>colFrame</i> ↔ —	Color of the element frame
in	<i>colFill_</i>	Color of the element fill

Returns

none

9.29.1.2 `#define GSLC_TYPEX_SLIDER`

9.29.2 Typedef Documentation

9.29.2.1 `typedef bool(* GSLC_CB_XSLIDER_POS)(void *pvGui, void *pvElem, int16_t nPos)`

Callback function for slider feedback.

9.29.3 Function Documentation

9.29.3.1 `gslc_tsElemRef* gslc_ElemXSliderCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXSlider * pXData, gslc_tsRect rElem, int16_t nPosMin, int16_t nPosMax, int16_t nPos, uint16_t nThumbSz, bool bVert)`

Create a Slider Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining checkbox size
in	<i>nPosMin</i>	Minimum position value
in	<i>nPosMax</i>	Maximum position value
in	<i>nPos</i>	Starting position value
in	<i>nThumbSz</i>	Size of the thumb control
in	<i>bVert</i>	Orientation (true for vertical)

Returns

Pointer to Element reference or NULL if failure

9.29.3.2 bool gslc_ElemXSliderDraw (void * *pvGui*, void * *pvElemRef*, gslc_teRedrawType *eRedraw*)

Draw a Slider element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.29.3.3 int gslc_ElemXSliderGetPos (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

Get a Slider element's current position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

Current slider position

9.29.3.4 void gslc_ElemXSliderSetPos (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, int16_t *nPos*)

Set a Slider element's current position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nPos</i>	New position value

Returns

none

9.29.3.5 void gslc_ElemXSliderSetPosFunc (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, GSLC_CB_XSLIDER_POS *funcCb*)

Assign the position callback function for a slider.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>funcCb</i>	Function pointer to position routine (or NULL for none)

Returns

none

9.29.3.6 void gslc_ElemXSliderSetStyle (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bTrim*, gslc_tsColor *colTrim*, uint16_t *nTickDiv*, int16_t *nTickLen*, gslc_tsColor *colTick*)

Set a Slider element's current position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bTrim</i>	Show a colored trim?
in	<i>colTrim</i>	Color of trim
in	<i>nTickDiv</i>	Number of tick divisions to show (0 for none)
in	<i>nTickLen</i>	Length of tickmarks
in	<i>colTick</i>	Color of ticks

Returns

none

9.29.3.7 `bool gslc_ElemXSliderTouch (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch events to Slider element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

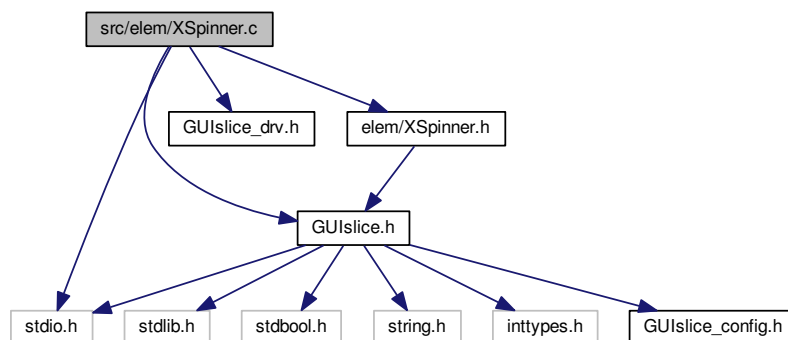
Returns

true if success, false otherwise

9.30 src/elem/XSpinner.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XSpinner.h"
#include <stdio.h>
```

Include dependency graph for XSpinner.c:



Functions

- [gslc_tsElemRef](#) * [gslc_ElemXSpinnerCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsXSpinner](#) *pXData, [gslc_tsRect](#) rElem, int16_t nMin, int16_t nMax, int16_t nVal, int16_t nIncr, int8_t nFontId, int8_t nButtonSz, [GSLC_CB_INPUT](#) cbInput)
Create a Spinner Element.
- bool [gslc_ElemXSpinnerSetChars](#) (void *pvGui, [gslc_tsElemRef](#) *pElemRef, uint8_t cIncr, uint8_t cDecr)
Set Up and Down characters for the Spinner element.
- bool [gslc_ElemXSpinnerDraw](#) (void *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a Spinner element on the screen.
- int [gslc_ElemXSpinnerGetCounter](#) ([gslc_tsGui](#) *pGui, [gslc_tsXSpinner](#) *pSpinner)
Get the current counter associated with Spinner.
- void [gslc_ElemXSpinnerSetCounter](#) ([gslc_tsGui](#) *pGui, [gslc_tsXSpinner](#) *pSpinner, int16_t nCount)
Set the current counter associated with Spinner.
- bool [gslc_ElemXSpinnerClick](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nX, int16_t nY)
Handle a click event within the Spinner.
- bool [gslc_ElemXSpinnerTouch](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY)
Handle touch (up,down,move) events to Spinner element.

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []
- static const int16_t [SPINNER_ID_BTN_INC](#)
- static const int16_t [SPINNER_ID_BTN_DEC](#)
- static const int16_t [SPINNER_ID_TXT](#)

9.30.1 Function Documentation

9.30.1.1 bool [gslc_ElemXSpinnerClick](#) (void * *pvGui*, void * *pvElemRef*, [gslc_teTouch](#) *eTouch*, int16_t *nX*, int16_t *nY*)

Handle a click event within the Spinner.

- This is called internally by the Spinner touch handler

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui *)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to gslc_tsElemRef *)
in	<i>eTouch</i>	Touch event type
in	<i>nX</i>	Touch X coord
in	<i>nY</i>	Touch Y coord

Returns

none

9.30.1.2 **gslc_tsElemRef*** **gslc_ElemXSpinnerCreate** (**gslc_tsGui** * *pGui*, **int16_t** *nElemId*, **int16_t** *nPage*, **gslc_tsXSpinner** * *pXData*, **gslc_tsRect** *rElem*, **int16_t** *nMin*, **int16_t** *nMax*, **int16_t** *nVal*, **int16_t** *nIncr*, **int8_t** *nFontId*, **int8_t** *nButtonSz*, **GSLC_CB_INPUT** *cbInput*)

Create a Spinner Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining overall size
in	<i>nMin</i>	Minimum value of Spinner
in	<i>nMax</i>	Maximum value of Spinner
in	<i>nVal</i>	Starting value of Spinner
in	<i>nIncr</i>	Increment Spinner by this value
in	<i>nFontId</i>	Font ID to use for drawing the element
in	<i>nButtonSz</i>	Size of individual buttons
in	<i>cbInput</i>	Callback for touch events

Returns

Pointer to Element or NULL if failure

9.30.1.3 **bool** **gslc_ElemXSpinnerDraw** (**void** * *pvGui*, **void** * *pvElemRef*, **gslc_teRedrawType** *eRedraw*)

Draw a Spinner element on the screen.

- Called during redraw

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.30.1.4 **int** **gslc_ElemXSpinnerGetCounter** (**gslc_tsGui** * *pGui*, **gslc_tsXSpinner** * *pSpinner*)

Get the current counter associated with Spinner.

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pSpinner</i>	Ptr to Element

Returns

Current counter value

9.30.1.5 `bool gslc_ElemXSpinnerSetChars (void * pvGui, gslc_tsElemRef * pElemRef, uint8_t cIncr, uint8_t cDecr)`

Set Up and Down characters for the Spinner element.

- Called during redraw

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui *)
in	<i>pElemRef</i>	Ptr to ElementRef
in	<i>cIncr</i>	Character to use to indicate incrementing the spinner
in	<i>cDecr</i>	Character to use to indicate decrementing the spinner

Returns

true if success, false otherwise

9.30.1.6 `void gslc_ElemXSpinnerSetCounter (gslc_tsGui * pGui, gslc_tsXSpinner * pSpinner, int16_t nCount)`

Set the current counter associated with Spinner.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pSpinner</i>	Ptr to Element
in	<i>nCount</i>	New counter value

Returns

none

9.30.1.7 `bool gslc_ElemXSpinnerTouch (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch (up,down,move) events to Spinner element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

Returns

true if success, false otherwise

9.30.2 Variable Documentation

9.30.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.30.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

9.30.2.3 `const int16_t SPINNER_ID_BTN_DEC [static]`

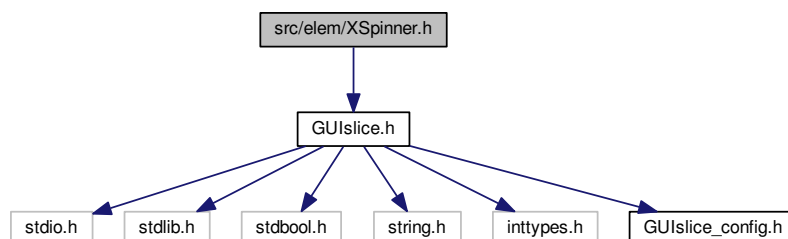
9.30.2.4 `const int16_t SPINNER_ID_BTN_INC [static]`

9.30.2.5 `const int16_t SPINNER_ID_TXT [static]`

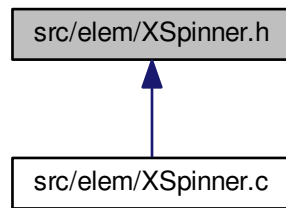
9.31 `src/elem/XSpinner.h` File Reference

```
#include "GUIslice.h"
```

Include dependency graph for `XSpinner.h`:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXSpinner](#)
Extended data for Spinner element.

Macros

- #define [GSLC_TYPEX_SPINNER](#)
- #define [XSPINNER_COMP_CNT](#)
- #define [XSPINNER_STR_LEN](#)
- #define [XSPINNER_CB_STATE_UPDATE](#)

Functions

- [gslc_tsElemRef * gslc_ElemXSpinnerCreate](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nElemId, [int16_t](#) nPage, [gslc_tsXSpinner](#) *pXData, [gslc_tsRect](#) rElem, [int16_t](#) nMin, [int16_t](#) nMax, [int16_t](#) nVal, [int16_t](#) nIncr, [int8_t](#) nFontId, [int8_t](#) nButtonSz, [GSLC_CB_INPUT](#) cbInput)
Create a Spinner Element.
- [bool gslc_ElemXSpinnerSetChars](#) ([void](#) *pvGui, [gslc_tsElemRef](#) *pElemRef, [uint8_t](#) cIncr, [uint8_t](#) cDecr)
Set Up and Down characters for the Spinner element.
- [bool gslc_ElemXSpinnerDraw](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw a Spinner element on the screen.
- [int gslc_ElemXSpinnerGetCounter](#) ([gslc_tsGui](#) *pGui, [gslc_tsXSpinner](#) *pSpinner)
Get the current counter associated with Spinner.
- [void gslc_ElemXSpinnerSetCounter](#) ([gslc_tsGui](#) *pGui, [gslc_tsXSpinner](#) *pSpinner, [int16_t](#) nCount)
Set the current counter associated with Spinner.
- [bool gslc_ElemXSpinnerClick](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teTouch](#) eTouch, [int16_t](#) nX, [int16_t](#) nY)
Handle a click event within the Spinner.
- [bool gslc_ElemXSpinnerTouch](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teTouch](#) eTouch, [int16_t](#) nRelX, [int16_t](#) nRelY)
Handle touch (up,down,move) events to Spinner element.

9.31.1 Macro Definition Documentation

9.31.1.1 `#define GSLC_TYPEX_SPINNER`

9.31.1.2 `#define XSPINNER_CB_STATE_UPDATE`

9.31.1.3 `#define XSPINNER_COMP_CNT`

9.31.1.4 `#define XSPINNER_STR_LEN`

9.31.2 Function Documentation

9.31.2.1 `bool gslc_ElemXSpinnerClick (void * pvGui, void * pvElemRef, gslc_tsTouch eTouch, int16_t nX, int16_t nY)`

Handle a click event within the Spinner.

- This is called internally by the Spinner touch handler

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nX</i>	Touch X coord
in	<i>nY</i>	Touch Y coord

Returns

none

9.31.2.2 `gslc_tsElemRef* gslc_ElemXSpinnerCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXSpinner * pXData, gslc_tsRect rElem, int16_t nMin, int16_t nMax, int16_t nVal, int16_t nIncr, int8_t nFontId, int8_t nButtonSz, GSLC_CB_INPUT cbInput)`

Create a Spinner Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or <code>GSLC_ID_AUTO</code> to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining overall size
in	<i>nMin</i>	Minimum value of Spinner
in	<i>nMax</i>	Maximum value of Spinner
in	<i>nVal</i>	Starting value of Spinner

Parameters

in	<i>nIncr</i>	Increment Spinner by this value
in	<i>nFontId</i>	Font ID to use for drawing the element
in	<i>nButtonSz</i>	Size of individual buttons
in	<i>cbInput</i>	Callback for touch events

Returns

Pointer to Element or NULL if failure

9.31.2.3 bool gslc_ElemXSpinnerDraw (void * *pvGui*, void * *pvElemRef*, gslc_teRedrawType *eRedraw*)

Draw a Spinner element on the screen.

- Called during redraw

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pvElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.31.2.4 int gslc_ElemXSpinnerGetCounter (gslc_tsGui * *pGui*, gslc_tsXSpinner * *pSpinner*)

Get the current counter associated with Spinner.

Parameters

in	<i>pGui</i>	Ptr to GUI
in	<i>pSpinner</i>	Ptr to Element

Returns

Current counter value

9.31.2.5 bool gslc_ElemXSpinnerSetChars (void * *pvGui*, gslc_tsElemRef * *pElemRef*, uint8_t *cIncr*, uint8_t *cDecr*)

Set Up and Down characters for the Spinner element.

- Called during redraw

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pElemRef</i>	Ptr to ElementRef
in	<i>cIncr</i>	Character to use to indicate incrementing the spinner
in	<i>cDecr</i>	Character to use to indicate decrementing the spinner

Returns

true if success, false otherwise

9.31.2.6 void `gslc_ElemXSpinnerSetCounter (gslc_tsGui * pGui, gslc_tsXSpinner * pSpinner, int16_t nCount)`

Set the current counter associated with Spinner.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pSpinner</i>	Ptr to Element
in	<i>nCount</i>	New counter value

Returns

none

9.31.2.7 bool `gslc_ElemXSpinnerTouch (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch (up,down,move) events to Spinner element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

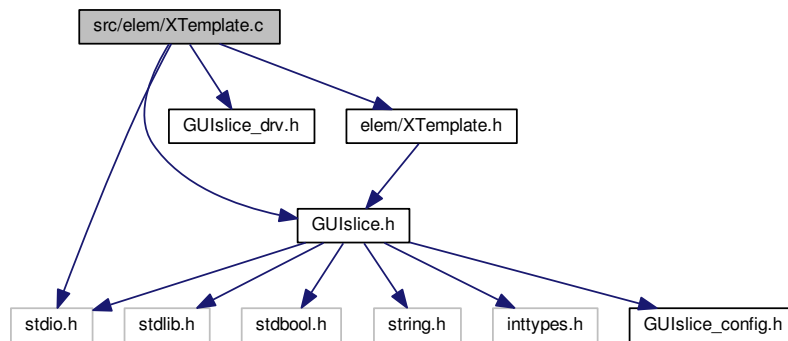
Returns

true if success, false otherwise

9.32 src/elem/XTemplate.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XTemplate.h"
#include <stdio.h>
```

Include dependency graph for XTemplate.c:



Functions

- [gslc_tsElemRef * gslc_ElemXTemplateCreate](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nElemId, [int16_t](#) nPage, [gslc_tsXTemplate](#) *pXData, [gslc_tsRect](#) rElem, [char](#) *pStrBuf, [uint8_t](#) nStrBufMax, [int16_t](#) nFontId)
Create an Extended Text Field Element.
- [bool gslc_ElemXTemplateDraw](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teRedrawType](#) eRedraw)
Draw the template element on the screen.
- [bool gslc_ElemXTemplateTouch](#) ([void](#) *pvGui, [void](#) *pvElemRef, [gslc_teTouch](#) eTouch, [int16_t](#) nRelX, [int16_t](#) nRelY)
Handle touch events to template element.

Variables

- [const char](#) [GSLC_PMEM ERRSTR_NULL](#) []
- [const char](#) [GSLC_PMEM ERRSTR_PXD_NULL](#) []

9.32.1 Function Documentation

9.32.1.1 [gslc_tsElemRef* gslc_ElemXTemplateCreate](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nElemId, [int16_t](#) nPage, [gslc_tsXTemplate](#) *pXData, [gslc_tsRect](#) rElem, [char](#) *pStrBuf, [uint8_t](#) nStrBufMax, [int16_t](#) nFontId)

Create an Extended Text Field Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining element size
in	<i>pStrBuf</i>	Ptr to string buffer
in	<i>nStrBufMax</i>	Maximum buffer alength allocated to pStrBuf
in	<i>nFontId</i>	ID of font to use for text output

Returns

Pointer to Element reference or NULL if failure

9.32.1.2 bool gslc_ElemXTemplateDraw (void * *pvGui*, void * *pElemRef*, gslc_teRedrawType *eRedraw*)

Draw the template element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pElemRef</i>	Void ptr to Element (typecast to gslc_tsElemRef*)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.32.1.3 bool gslc_ElemXTemplateTouch (void * *pGui*, void * *pElemRef*, gslc_teTouch *eTouch*, int16_t *nRelX*, int16_t *nRelY*)

Handle touch events to template element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pGui</i>	Void ptr to GUI (typecast to gslc_tsGui*)
in	<i>pElemRef</i>	Void ptr to Element ref (typecast to gslc_tsElemRef*)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

Returns

true if success, false otherwise

9.32.2 Variable Documentation

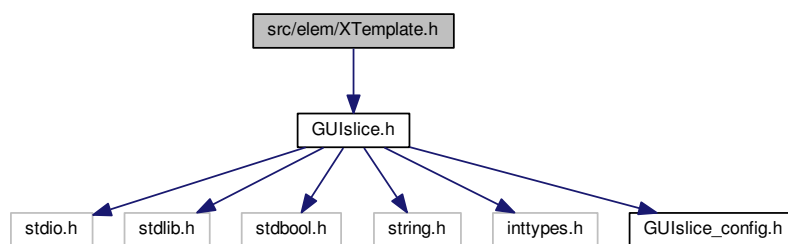
9.32.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.32.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

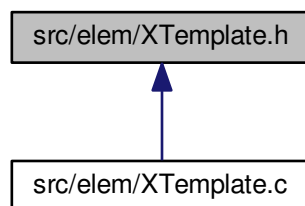
9.33 src/elem/XTemplate.h File Reference

```
#include "GUIslice.h"
```

Include dependency graph for XTemplate.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsXTemplate](#)

Callback function for slider feedback.

Macros

- `#define GSLC_TYPEX_TEMPLATE`

Functions

- `gslc_tsElemRef * gslc_ElemXTemplateCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXTemplate *pXData, gslc_tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)`
Create an Extended Text Field Element.
- `bool gslc_ElemXTemplateDraw (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
Draw the template element on the screen.
- `bool gslc_ElemXTemplateTouch (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`
Handle touch events to template element.

9.33.1 Macro Definition Documentation

9.33.1.1 `#define GSLC_TYPEX_TEMPLATE`

9.33.2 Function Documentation

9.33.2.1 `gslc_tsElemRef* gslc_ElemXTemplateCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXTemplate * pXData, gslc_tsRect rElem, char * pStrBuf, uint8_t nStrBufMax, int16_t nFontId)`

Create an Extended Text Field Element.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining element size
in	<i>pStrBuf</i>	Ptr to string buffer
in	<i>nStrBufMax</i>	Maximum buffer alength allocated to pStrBuf
in	<i>nFontId</i>	ID of font to use for text output

Returns

Pointer to Element reference or NULL if failure

9.33.2.2 `bool gslc_ElemXTemplateDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw the template element on the screen.

- Called from `gslc_ElemDraw()`

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.33.2.3 `bool gslc_ElemXTemplateTouch (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)`

Handle touch events to template element.

- Called from [gslc_ElemSendEventTouch\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element ref (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eTouch</i>	Touch event type
in	<i>nRelX</i>	Touch X coord relative to element
in	<i>nRelY</i>	Touch Y coord relative to element

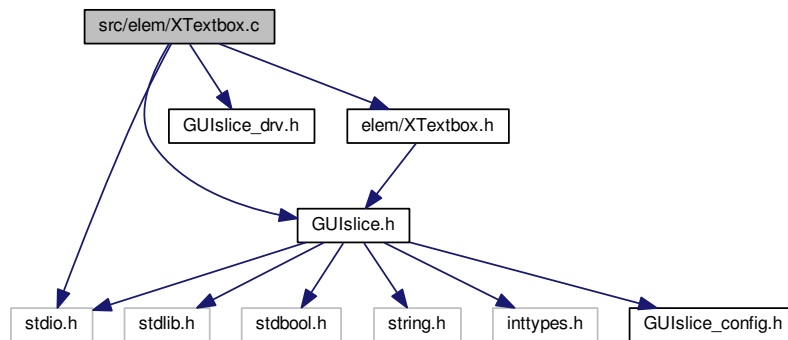
Returns

true if success, false otherwise

9.34 src/elem/XTextbox.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include "elem/XTextbox.h"
#include <stdio.h>
```

Include dependency graph for XTextbox.c:



Functions

- `gslc_tsElemRef * gslc_ElemXTextboxCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsXTextbox *pXData, gslc_tsRect rElem, int16_t nFontId, char *pBuf, uint16_t nBufRows, uint16_t nBufCols)`
Create a Textbox Element.
- `void gslc_ElemXTextboxReset (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
Reset the contents of the textbox.
- `void gslc_ElemXTextboxLineWrAdv (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
- `void gslc_ElemXTextboxScrollSet (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)`
Set the textbox scroll position (nScrollPos) as a fraction of nScrollMax.
- `void gslc_ElemXTextboxBufAdd (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned char chNew, bool bAdvance)`
- `void gslc_ElemXTextboxColSet (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor nCol)`
Insert a color set code into the current buffer position.
- `void gslc_ElemXTextboxColReset (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
Insert a color reset code into the current buffer position.
- `void gslc_ElemXTextboxWrapSet (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bWrapEn)`
Enable or disable line wrap within textbox.
- `void gslc_ElemXTextboxAdd (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, char *pTxt)`
Add a text string to the textbox.
- `bool gslc_ElemXTextboxDraw (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
Draw a Textbox element on the screen.

Variables

- `const char GSLC_PMEM_ERRSTR_NULL []`
- `const char GSLC_PMEM_ERRSTR_PXD_NULL []`

9.34.1 Function Documentation

9.34.1.1 void `gslc_ElemXTextboxAdd (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, char * pTxt)`

Add a text string to the textbox.

- If it includes a newline then the buffer will advance to the next row
- If wrap has been enabled, then a newline will be forced

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>pTxt</i>	Pointer to text string (null-terminated)

Returns

none

9.34.1.2 void `gslc_ElemXTextboxBufAdd (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, unsigned char chNew, bool bAdvance)`

9.34.1.3 void `gslc_ElemXTextboxColReset (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Insert a color reset code into the current buffer position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

none

9.34.1.4 void `gslc_ElemXTextboxColSet (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tsColor nCol)`

Insert a color set code into the current buffer position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nCol</i>	Color to assign for next text written to textbox

Returns

none

9.34.1.5 `gslc_tsElemRef* gslc_ElemXTextboxCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsXTextbox * pXData, gslc_tsRect rElem, int16_t nFontId, char * pBuf, uint16_t nBufRows, uint16_t nBufCols)`

Create a Textbox Element.

- The textbox is a scrolling window designed for displaying multi-line text using a monospaced font. A character buffer is defined by nBufRows*nBufCols to capture the added text. If the allocation buffer is larger than the display size (defined by rElem), then a scrollbar will be shown.
- Support for changing color within a row can be enabled with `GSLC_FEATURE_XTEXTBOX_EMBED 1`
- Note that each color change command will consume 4 of the available "column" bytes.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or <code>GSLC_ID_AUTO</code> to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining textbox size
in	<i>nFontId</i>	Font ID to use for text area
in	<i>pBuf</i>	Ptr to text buffer (already allocated) with size (nBufRows*nBufCols) chars
in	<i>nBufRows</i>	Number of rows in buffer
in	<i>nBufCols</i>	Number of columns in buffer (incl special codes)

Returns

Pointer to Element reference or NULL if failure

9.34.1.6 `bool gslc_ElemXTextboxDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a Textbox element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.34.1.7 void gslc_ElemXTextboxLineWrAdv (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

9.34.1.8 void gslc_ElemXTextboxReset (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

Reset the contents of the textbox.

- Clears the buffer and resets the position

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

none

9.34.1.9 void gslc_ElemXTextboxScrollSet (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, uint8_t *nScrollPos*, uint8_t *nScrollMax*)

Set the textbox scroll position (*nScrollPos*) as a fraction of *nScrollMax*.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nScrollPos</i>	New scroll position
in	<i>nScrollMax</i>	Maximum scroll position

Returns

none

9.34.1.10 void gslc_ElemXTextboxWrapSet (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bWrapEn*)

Enable or disable line wrap within textbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bWrapEn</i>	Enable line wrap if true

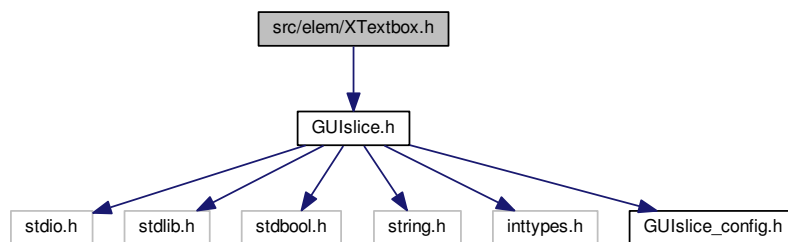
Returns

none

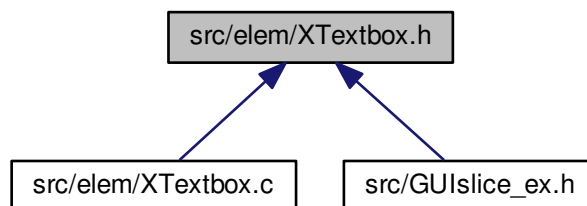
9.34.2 Variable Documentation**9.34.2.1** `const char GSLC_PMEM ERRSTR_NULL[]`**9.34.2.2** `const char GSLC_PMEM ERRSTR_PXD_NULL[]`**9.35 src/elem/XTextbox.h File Reference**

```
#include "GUIslice.h"
```

Include dependency graph for XTextbox.h:



This graph shows which files directly or indirectly include this file:

**Data Structures**

- struct [gslc_tsXTextbox](#)

Extended data for Textbox element.

Macros

- `#define GSLC_TYPEX_TEXTBOX`
- `#define GSLC_XTEXTBOX_CODE_COL_SET`
Definitions for textbox special inline codes.
- `#define GSLC_XTEXTBOX_CODE_COL_RESET`
- `#define XTEXTBOX_REDRAW_NONE`
- `#define XTEXTBOX_REDRAW_ALL`

Functions

- `gslc_tsElemRef * gslc_ElemXTextboxCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsX↵
Textbox *pXData, gslc_tsRect rElem, int16_t nFontId, char *pBuf, uint16_t nBufRows, uint16_t nBufCols)`
Create a Textbox Element.
- `void gslc_ElemXTextboxReset (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
Reset the contents of the textbox.
- `bool gslc_ElemXTextboxDraw (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
Draw a Textbox element on the screen.
- `void gslc_ElemXTextboxAdd (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, char *pTxt)`
Add a text string to the textbox.
- `void gslc_ElemXTextboxColSet (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor nCol)`
Insert a color set code into the current buffer position.
- `void gslc_ElemXTextboxColReset (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`
Insert a color reset code into the current buffer position.
- `void gslc_ElemXTextboxWrapSet (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bWrapEn)`
Enable or disable line wrap within textbox.
- `void gslc_ElemXTextboxScrollSet (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t↵
_t nScrollMax)`
Set the textbox scroll position (nScrollPos) as a fraction of nScrollMax.

9.35.1 Macro Definition Documentation

9.35.1.1 `#define GSLC_TYPEX_TEXTBOX`

9.35.1.2 `#define GSLC_XTEXTBOX_CODE_COL_RESET`

9.35.1.3 `#define GSLC_XTEXTBOX_CODE_COL_SET`

Definitions for textbox special inline codes.

9.35.1.4 `#define XTEXTBOX_REDRAW_ALL`

9.35.1.5 `#define XTEXTBOX_REDRAW_NONE`

9.35.2 Function Documentation

9.35.2.1 `void gslc_ElemXTextboxAdd (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, char * pTxt)`

Add a text string to the textbox.

- If it includes a newline then the buffer will advance to the next row
- If wrap has been enabled, then a newline will be forced

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>pTxt</i>	Pointer to text string (null-terminated)

Returns

none

9.35.2.2 void `gslc_ElemXTextboxColReset (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Insert a color reset code into the current buffer position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

none

9.35.2.3 void `gslc_ElemXTextboxColSet (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tsColor nCol)`

Insert a color set code into the current buffer position.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nCol</i>	Color to assign for next text written to textbox

Returns

none

9.35.2.4 `gslc_tsElemRef*` `gslc_ElemXTextboxCreate (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage,
gslc_tsXTextbox * pXData, gslc_tsRect rElem, int16_t nFontId, char * pBuf, uint16_t nBufRows, uint16_t
nBufCols)`

Create a Textbox Element.

- The textbox is a scrolling window designed for displaying multi-line text using a monospaced font. A character buffer is defined by *nBufRows***nBufCols* to capture the added text. If the allocation buffer is larger than the display size (defined by *rElem*), then a scrollbar will be shown.

- Support for changing color within a row can be enabled with `GSLC_FEATURE_XTEXTBOX_EMBED 1`
- Note that each color change command will consume 4 of the available "column" bytes.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Element ID to assign (0..16383 or <code>GSLC_ID_AUTO</code> to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining textbox size
in	<i>nFontId</i>	Font ID to use for text area
in	<i>pBuf</i>	Ptr to text buffer (already allocated) with size (nBufRows*nBufCols) chars
in	<i>nBufRows</i>	Number of rows in buffer
in	<i>nBufCols</i>	Number of columns in buffer (incl special codes)

Returns

Pointer to Element reference or NULL if failure

9.35.2.5 `bool gslc_ElemXTextboxDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a Textbox element on the screen.

- Called from [gslc_ElemDraw\(\)](#)

Parameters

in	<i>pvGui</i>	Void ptr to GUI (typecast to <code>gslc_tsGui*</code>)
in	<i>pvElemRef</i>	Void ptr to Element reference (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

9.35.2.6 `void gslc_ElemXTextboxReset (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef)`

Reset the contents of the textbox.

- Clears the buffer and resets the position

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference

Returns

none

9.35.2.7 void gslc_ElemXTextboxScrollSet (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, uint8_t *nScrollPos*, uint8_t *nScrollMax*)

Set the textbox scroll position (*nScrollPos*) as a fraction of *nScrollMax*.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nScrollPos</i>	New scroll position
in	<i>nScrollMax</i>	Maximum scroll position

Returns

none

9.35.2.8 void gslc_ElemXTextboxWrapSet (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, bool *bWrapEn*)

Enable or disable line wrap within textbox.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>bWrapEn</i>	Enable line wrap if true

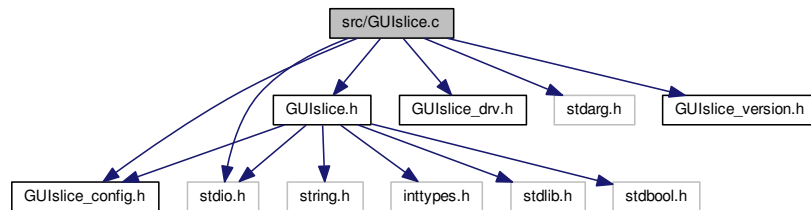
Returns

none

9.36 src/GUISlice.c File Reference

```
#include "GUISlice_config.h"
#include "GUISlice.h"
#include "GUISlice_drv.h"
#include <stdio.h>
#include <stdarg.h>
#include "GUISlice_version.h"
```

Include dependency graph for GUISlice.c:



Enumerations

- enum [gslc_teDebugPrintState](#) {
[GSLC_S_DEBUG_PRINT_NORM](#), [GSLC_S_DEBUG_PRINT_TOKEN](#), [GSLC_S_DEBUG_PRINT_UINT16](#),
[GSLC_S_DEBUG_PRINT_CHAR](#),
[GSLC_S_DEBUG_PRINT_STR](#), [GSLC_S_DEBUG_PRINT_STR_P](#) }

Functions

- char * [gslc_GetVer](#) ([gslc_tsGui](#) *pGui)
Get the GUISlice version number.
- const char * [gslc_GetNameDisp](#) ([gslc_tsGui](#) *pGui)
Get the GUISlice display driver name.
- const char * [gslc_GetNameTouch](#) ([gslc_tsGui](#) *pGui)
Get the GUISlice touch driver name.
- void * [gslc_GetDriverDisp](#) ([gslc_tsGui](#) *pGui)
Get the native display driver instance.
- void * [gslc_GetDriverTouch](#) ([gslc_tsGui](#) *pGui)
Get the native touch driver instance.
- bool [gslc_Init](#) ([gslc_tsGui](#) *pGui, void *pvDriver, [gslc_tsPage](#) *asPage, uint8_t nMaxPage, [gslc_tsFont](#) *asFont, uint8_t nMaxFont)
Initialize the GUISlice library.
- void [gslc_SetPinPollFunc](#) ([gslc_tsGui](#) *pGui, [GSLC_CB_PIN_POLL](#) pfunc)
- void [gslc_InitInputMap](#) ([gslc_tsGui](#) *pGui, [gslc_tsInputMap](#) *asInputMap, uint8_t nInputMapMax)
- void [gslc_InputMapAdd](#) ([gslc_tsGui](#) *pGui, [gslc_teInputRawEvent](#) eInputEvent, int16_t nInputVal, [gslc_teAction](#) eAction, int16_t nActionVal)
- bool [gslc_InputMapLookup](#) ([gslc_tsGui](#) *pGui, [gslc_teInputRawEvent](#) eInputEvent, int16_t nInputVal, [gslc_teAction](#) *peAction, int16_t *pnActionVal)
- void [gslc_InitDebug](#) ([GSLC_CB_DEBUG_OUT](#) pfunc)
Initialize debug output.
- void [gslc_DebugPrintf](#) (const char *pFmt,...)
Optimized printf routine for GUISlice debug/error output.
- void [gslc_Quit](#) ([gslc_tsGui](#) *pGui)
Exit the GUISlice environment.
- void [gslc_Update](#) ([gslc_tsGui](#) *pGui)
Perform main GUISlice handling functions.
- [gslc_tsEvent](#) [gslc_EventCreate](#) ([gslc_tsGui](#) *pGui, [gslc_teEventType](#) eType, uint8_t nSubType, void *pvScope, void *pvData)

- Create an event structure.*

 - bool [gslc_IsInRect](#) (int16_t nSelX, int16_t nSelY, [gslc_tsRect](#) rRect)

Determine if a coordinate is inside of a rectangular region.
- bool [gslc_IsInWH](#) (int16_t nSelX, int16_t nSelY, uint16_t nWidth, uint16_t nHeight)

Determine if a coordinate is inside of a width x height region.
- void [gslc_OrderCoord](#) (int16_t *pnX0, int16_t *pnY0, int16_t *pnX1, int16_t *pnY1)
- bool [gslc_ClipPt](#) ([gslc_tsRect](#) *pClipRect, int16_t nX, int16_t nY)

Perform basic clipping of a single point to a clipping region.
- bool [gslc_ClipLine](#) ([gslc_tsRect](#) *pClipRect, int16_t *pnX0, int16_t *pnY0, int16_t *pnX1, int16_t *pnY1)

Perform basic clipping of a line to a clipping region.
- bool [gslc_ClipRect](#) ([gslc_tsRect](#) *pClipRect, [gslc_tsRect](#) *pRect)

Perform basic clipping of a rectangle to a clipping region.
- [gslc_tslmgRef](#) [gslc_ResetImage](#) ()

Create a blank image reference structure.
- [gslc_tslmgRef](#) [gslc_GetImageFromFile](#) (const char *pFname, [gslc_telmgRefFlags](#) eFmt)

Create an image reference to a bitmap file in LINUX filesystem.
- [gslc_tslmgRef](#) [gslc_GetImageFromSD](#) (const char *pFname, [gslc_telmgRefFlags](#) eFmt)

Create an image reference to a bitmap file in SD card.
- [gslc_tslmgRef](#) [gslc_GetImageFromRam](#) (unsigned char *plmgBuf, [gslc_telmgRefFlags](#) eFmt)

Create an image reference to a bitmap in SRAM.
- [gslc_tslmgRef](#) [gslc_GetImageFromProg](#) (const unsigned char *plmgBuf, [gslc_telmgRefFlags](#) eFmt)

Create an image reference to a bitmap in program memory (PROGMEM)
- int16_t [gslc_sinFX](#) (int16_t n64Ang)

Calculate fixed-point sine function from fractional degrees.
- int16_t [gslc_cosFX](#) (int16_t n64Ang)

Calculate fixed-point cosine function from fractional degrees.
- void [gslc_PolarToXY](#) (uint16_t nRad, int16_t n64Ang, int16_t *nDX, int16_t *nDY)

Convert polar coordinate to cartesian.
- [gslc_tsColor](#) [gslc_ColorBlend2](#) ([gslc_tsColor](#) colStart, [gslc_tsColor](#) colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)

Create a color based on a blend between two colors.
- [gslc_tsColor](#) [gslc_ColorBlend3](#) ([gslc_tsColor](#) colStart, [gslc_tsColor](#) colMid, [gslc_tsColor](#) colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)

Create a color based on a blend between three colors.
- bool [gslc_ColorEqual](#) ([gslc_tsColor](#) a, [gslc_tsColor](#) b)

Check whether two colors are equal.
- void [gslc_DrawSetPixel](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, [gslc_tsColor](#) nCol)

Set a pixel on the active screen to the given color with lock.
- void [gslc_DrawLine](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, [gslc_tsColor](#) nCol)

Draw an arbitrary line using Bresenham's algorithm.
- void [gslc_DrawLineH](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, uint16_t nW, [gslc_tsColor](#) nCol)

Draw a horizontal line.
- void [gslc_DrawLineV](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, uint16_t nH, [gslc_tsColor](#) nCol)

Draw a vertical line.
- void [gslc_DrawLinePolar](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, uint16_t nRadStart, uint16_t nRadEnd, int16_t n64Ang, [gslc_tsColor](#) nCol)

Draw a polar ray segment.
- void [gslc_DrawFrameRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)

Draw a framed rectangle.
- void [gslc_DrawFrameRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)

Draw a framed rounded rectangle.

- void [gslc_DrawFillRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
Draw a filled rectangle.
- void [gslc_DrawFillRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)
Draw a filled rounded rectangle.
- [gslc_tsRect](#) [gslc_ExpandRect](#) ([gslc_tsRect](#) rRect, int16_t nExpandW, int16_t nExpandH)
Expand or contract a rectangle in width and/or height (equal amounts on both side), based on the centerpoint of the rectangle.
- void [gslc_UnionRect](#) ([gslc_tsRect](#) *pRect, [gslc_tsRect](#) rAddRect)
Expand a rect to include another rect.
- void [gslc_InvalidateRgnReset](#) ([gslc_tsGui](#) *pGui)
Reset the invalidation region.
- void [gslc_InvalidateRgnScreen](#) ([gslc_tsGui](#) *pGui)
Mark the entire screen as invalidated.
- void [gslc_InvalidateRgnPage](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage)
Include an entire page (eg.
- void [gslc_InvalidateRgnAdd](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rAddRect)
Add a rectangular region to the invalidation region.
- void [gslc_DrawFrameCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)
Draw a framed circle.
- void [gslc_DrawFillCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)
Draw a filled circle.
- void [gslc_DrawFrameTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)
Draw a framed triangle.
- void [gslc_SwapCoords](#) (int16_t *pnXa, int16_t *pnYa, int16_t *pnXb, int16_t *pnYb)
- void [gslc_DrawFillTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)
Draw a filled triangle.
- void [gslc_DrawFrameQuad](#) ([gslc_tsGui](#) *pGui, [gslc_tsPt](#) *psPt, [gslc_tsColor](#) nCol)
Draw a framed quadrilateral.
- void [gslc_DrawFillQuad](#) ([gslc_tsGui](#) *pGui, [gslc_tsPt](#) *psPt, [gslc_tsColor](#) nCol)
Draw a filled quadrilateral.
- void [gslc_DrawFillSectorBase](#) ([gslc_tsGui](#) *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, [gslc_tsColor](#) cArcStart, [gslc_tsColor](#) cArcEnd, bool bGradient, int16_t nAngGradStart, int16_t nAngGradRange, int16_t nAngSecStart, int16_t nAngSecEnd)
Draw a gradient filled sector of a circle with support for inner and outer radius.
- void [gslc_DrawFillGradSector](#) ([gslc_tsGui](#) *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, [gslc_tsColor](#) cArcStart, [gslc_tsColor](#) cArcEnd, int16_t nAngSecStart, int16_t nAngSecEnd, int16_t nAngGradStart, int16_t nAngGradRange)
Draw a flat filled sector of a circle with support for inner and outer radius.
- void [gslc_DrawFillSector](#) ([gslc_tsGui](#) *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, [gslc_tsColor](#) cArc, int16_t nAngSecStart, int16_t nAngSecEnd)
Draw a flat filled sector of a circle with support for inner and outer radius.
- bool [gslc_FontSetBase](#) ([gslc_tsGui](#) *pGui, uint8_t nFontInd, int16_t nFontId, [gslc_teFontRefType](#) eFontRefType, const void *pvFontRef, uint16_t nFontSz)
Load a font into the local font cache and store as font ID (nFontId)
- bool [gslc_FontSet](#) ([gslc_tsGui](#) *pGui, int16_t nFontId, [gslc_teFontRefType](#) eFontRefType, const void *pvFontRef, uint16_t nFontSz)
Load a font into the local font cache and assign font ID (nFontId).
- [gslc_tsFont](#) * [gslc_FontGet](#) ([gslc_tsGui](#) *pGui, int16_t nFontId)

- Fetch a font from its ID value.*

 - bool `gslc_FontSetMode` (`gslc_tsGui` *pGui, int16_t nFontId, `gslc_teFontRefMode` eFontMode)
- Set the font operating mode.*

 - bool `gslc_PageEvent` (void *pvGui, `gslc_tsEvent` sEvent)
- Common event handler function for a page.*

 - void `gslc_PageAdd` (`gslc_tsGui` *pGui, int16_t nPageId, `gslc_tsElem` *psElem, uint16_t nMaxElem, `gslc_tsElemRef` *psElemRef, uint16_t nMaxElemRef)
- Add a page to the GUI.*

 - int `gslc_GetPageCur` (`gslc_tsGui` *pGui)
- Fetch the current page ID.*

 - void `gslc_SetStackPage` (`gslc_tsGui` *pGui, uint8_t nStackPos, int16_t nPageId)
- Assign a page to the page stack.*

 - void `gslc_SetStackState` (`gslc_tsGui` *pGui, uint8_t nStackPos, bool bActive, bool bDoDraw)
- Change the status of a page in a page stack.*

 - void `gslc_SetPageBase` (`gslc_tsGui` *pGui, int16_t nPageId)
- Assigns a page for the base layer in the page stack.*

 - void `gslc_SetPageCur` (`gslc_tsGui` *pGui, int16_t nPageId)
- Select a page for the current layer in the page stack.*

 - void `gslc_SetPageOverlay` (`gslc_tsGui` *pGui, int16_t nPageId)
- Select a page for the overlay layer in the page stack.*

 - void `gslc_PopupShow` (`gslc_tsGui` *pGui, int16_t nPageId, bool bModal)
- Show a popup dialog.*

 - void `gslc_PopupHide` (`gslc_tsGui` *pGui)
- Hides the currently active popup dialog.*

 - void `gslc_PageRedrawSet` (`gslc_tsGui` *pGui, bool bRedraw)
- Update the need-redraw status for the current page.*

 - bool `gslc_PageRedrawGet` (`gslc_tsGui` *pGui)
- Get the need-redraw status for the current page.*

 - void `gslc_PageRedrawCalc` (`gslc_tsGui` *pGui)
- Perform a redraw calculation on the page to determine if additional elements should also be redrawn.*

 - void `gslc_PageRedrawGo` (`gslc_tsGui` *pGui)
- Redraw all elements on the active page.*

 - void `gslc_PageFlipSet` (`gslc_tsGui` *pGui, bool bNeeded)
- Indicate whether the screen requires page flip.*

 - bool `gslc_PageFlipGet` (`gslc_tsGui` *pGui)
- Get state of pending page flip state.*

 - void `gslc_PageFlipGo` (`gslc_tsGui` *pGui)
- Update the visible screen if page has been marked for flipping.*

 - `gslc_tsPage` * `gslc_PageFindById` (`gslc_tsGui` *pGui, int16_t nPageId)
- Find a page in the GUI by its ID.*

 - `gslc_tsElemRef` * `gslc_PageFindElemById` (`gslc_tsGui` *pGui, int16_t nPageId, int16_t nElemId)
- Find an element in the GUI by its Page ID and Element ID.*

 - int16_t `gslc_PageFocusStep` (`gslc_tsGui` *pGui, `gslc_tsPage` *pPage, bool bNext)
- Get an Element ID from an element structure.*

 - int `gslc_ElemGetId` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pElemRef)
- Get the flags associated with an element reference.*

 - uint8_t `gslc_GetElemRefFlag` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pElemRef, uint8_t nFlagMask)
- Set the flags associated with an element reference.*

 - void `gslc_SetElemRefFlag` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pElemRef, uint8_t nFlagMask, uint8_t nFlagVal)
- Set the flags associated with an element reference.*

 - `gslc_tsElem` * `gslc_GetElemFromRef` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pElemRef)

Returns a pointer to an element from an element reference, copying from FLASH to RAM if element is stored in PROGMEM.

- `gslc_tsElem * gslc_GetElemFromRefD (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nLineNum)`

Returns a pointer to an element from an element reference.

- `void * gslc_GetXDataFromRef (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nType, int16_t nLineNum)`

Returns a pointer to the data structure associated with an extended element.

- `void gslc_SetRoundRadius (gslc_tsGui *pGui, uint8_t nRadius)`

Set the global rounded radius.

- `gslc_tsElemRef * gslc_ElemCreateTxt (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)`

Create a Text Element.

- `gslc_tsElemRef * gslc_ElemCreateBtnTxt (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId, GSLC_CB_TOUCH cbTouch)`

Create a textual Button Element.

- `gslc_tsElemRef * gslc_ElemCreateBtnImg (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, gslc_tsImgRef sImgRef, gslc_tsImgRef sImgRefSel, GSLC_CB_TOUCH cbTouch)`

Create a graphical Button Element.

- `gslc_tsElemRef * gslc_ElemCreateBox (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem)`

Create a Box Element.

- `gslc_tsElemRef * gslc_ElemCreateLine (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1)`

Create a Line Element.

- `gslc_tsElemRef * gslc_ElemCreateImg (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, gslc_tsImgRef sImgRef)`

Create an image Element.

- `bool gslc_ElemEvent (void *pvGui, gslc_tsEvent sEvent)`

Common event handler function for an element.

- `void gslc_ElemDraw (gslc_tsGui *pGui, int16_t nPageId, int16_t nElemId)`

Draw an element to the active display.

- `void gslc_DrawTxtBase (gslc_tsGui *pGui, char *pStrBuf, gslc_tsRect rTxt, gslc_tsFont *pTxtFont, gslc_tsTextFlags eTxtFlags, int8_t eTxtAlign, gslc_tsColor colTxt, gslc_tsColor colBg, int16_t nMarginW, int16_t nMarginH)`

Draw text with full text justification.

- `bool gslc_ElemDrawByRef (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsRedrawType eRedraw)`

Draw an element to the active display.

- `void gslc_ElemSetFillEn (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFillEn)`

Set the fill state for an Element.

- `void gslc_ElemSetFrameEn (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFrameEn)`

Set the frame state for an Element.

- `void gslc_ElemSetRoundEn (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bRoundEn)`

Set the rounded frame/fill state for an Element.

- `void gslc_ElemSetCol (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colFrame, gslc_tsColor colFill, gslc_tsColor colFillGlow)`

Update the common color selection for an Element.

- `void gslc_ElemSetGlowCol (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colFrameGlow, gslc_tsColor colFillGlow, gslc_tsColor colTxtGlow)`

Update the common color selection for glowing state of an Element.

- `void gslc_ElemSetGroup (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int nGroupId)`

Set the group ID for an element.

- `int gslc_ElemGetGroup (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`

- Get the group ID for an element.*

 - void [gslc_ElemSetTxtAlign](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, unsigned nAlign)

Set the alignment of a textual element (horizontal and vertical)
- void [gslc_ElemSetTxtMargin](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, unsigned nMargin)

Set the margin around of a textual element.
- void [gslc_ElemSetTxtMarginXY](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int8_t nMarginX, int8_t nMarginY)

Set the margin around of a textual element (X & Y offsets can be different)
- void [gslc_ElemSetTxtStr](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, const char *pStr)

Update the text string associated with an Element.
- char * [gslc_ElemGetTxtStr](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Fetch the current text string associated with an Element.
- void [gslc_ElemSetTxtCol](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsColor](#) colVal)

Update the text string color associated with an Element ID.
- void [gslc_ElemSetTxtMem](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teTxtFlags](#) eFlags)

Update the text string location in memory.
- void [gslc_ElemSetTxtEnc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teTxtFlags](#) eFlags)

Update the text string encoding mode.
- void [gslc_ElemUpdateFont](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int nFontId)

Update the Font selected for an Element's text.
- void [gslc_ElemSetRedraw](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) eRedraw)

Update the need-redraw status for an element.
- [gslc_teRedrawType](#) [gslc_ElemGetRedraw](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Get the need-redraw status for an element.
- void [gslc_ElemSetGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bGlowing)

Update the glowing indicator for an element.
- bool [gslc_ElemGetGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Get the glowing indicator for an element.
- void [gslc_ElemSetVisible](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bVisible)

Update the visibility status for an element.
- bool [gslc_ElemGetVisible](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Get the visibility status for an element.
- bool [gslc_ElemGetOnScreen](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Determine whether an element is visible on the screen.
- void [gslc_ElemSetGlowEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bGlowEn)

Update the glowing enable for an element.
- bool [gslc_ElemGetGlowEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Get the glowing enable for an element.
- void [gslc_ElemSetClickEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bClickEn)

Update the click enable for an element.
- void [gslc_ElemSetTouchFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_TOUCH](#) funcCb)

Update the touch function callback for an element.
- void [gslc_ElemSetStyleFrom](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRefSrc, [gslc_tsElemRef](#) *pElemRefDest)

Copy style settings from one element to another.
- void [gslc_ElemSetDrawFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_DRAW](#) funcCb)

Assign the drawing callback function for an element.
- void [gslc_ElemSetTickFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_TICK](#) funcCb)

Assign the tick callback function for an element.
- bool [gslc_ElemOwnsCoord](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nX, int16_t nY, bool bOnlyClickEn)

- Determine if a coordinate is inside of an element.*

 - void [gslc_CollectInput](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsEventTouch](#) *pEventTouch)

Handle direct input events within the element collection.
- void [gslc_CollectTouch](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsEventTouch](#) *pEventTouch)

Handle touch events within the element collection.
- bool [gslc_CollectTouchCompound](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY, [gslc_tsCollect](#) *pCollect)

Handle dispatch of touch (up,down,move) events to compound elements sub elements.
- void [gslc_TrackInput](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage, [gslc_telInputRawEvent](#) eInputEvent, int16_t nInputVal)

Handles a direct input event and performs the necessary tracking, glowing and selection actions depending on the state.
- void [gslc_TrackTouch](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage, int16_t nX, int16_t nY, uint16_t nPress)

Handles a touch event and performs the necessary tracking, glowing and selection actions depending on the press state.
- bool [gslc_InitTouch](#) ([gslc_tsGui](#) *pGui, const char *acDev)

Initialize the touchscreen device driver.
- bool [gslc_GetTouch](#) ([gslc_tsGui](#) *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, [gslc_telInputRawEvent](#) *peInputEvent, int16_t *pnInputVal)

Initialize the touchscreen device driver.
- void [gslc_SetTouchRemapEn](#) ([gslc_tsGui](#) *pGui, bool bEn)

Configure touchscreen remapping.
- void [gslc_SetTouchRemapCal](#) ([gslc_tsGui](#) *pGui, uint16_t nXMin, uint16_t nXMax, uint16_t nYMin, uint16_t nYMax)

Configure touchscreen calibration values.
- void [gslc_SetTouchRemapYX](#) ([gslc_tsGui](#) *pGui, bool bSwap)

Configure touchscreen XY swap.
- [gslc_tsElem](#) [gslc_ElemCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPageId, int16_t nType, [gslc_tsRect](#) rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)

Create a new element with default styling.
- bool [gslc_CollectEvent](#) (void *pvGui, [gslc_tsEvent](#) sEvent)

Common event handler function for an element collection.
- [gslc_tsElemRef](#) * [gslc_CollectElemAdd](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, const [gslc_tsElem](#) *pElem, [gslc_teElemRefFlags](#) eFlags)

Add an element to a collection.
- bool [gslc_CollectGetRedraw](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)

Determine if any elements in a collection need redraw.
- [gslc_tsElemRef](#) * [gslc_ElemAdd](#) ([gslc_tsGui](#) *pGui, int16_t nPageId, [gslc_tsElem](#) *pElem, [gslc_teElemRefFlags](#) eFlags)

Add the Element to the list of generated elements in the GUI environment.
- bool [gslc_SetClipRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) *pRect)

Set the clipping rectangle for further drawing.
- void [gslc_ElemSetImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tsImgRef](#) sImgRef, [gslc_tsImgRefSel](#) sImgRefSel)

Set an element to use a bitmap image.
- bool [gslc_SetBkgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)

Configure the background to use a bitmap image.
- bool [gslc_SetBkgndColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)

Configure the background to use a solid color.
- bool [gslc_SetTransparentColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)

Configure the color to use for image transparency.
- bool [gslc_GuiRotate](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation)

Dynamically change rotation, automatically adapt touchscreen axes swap/flip.

- bool [gslc_ElemSendEventTouch](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRefTracked, [gslc_teTouch](#) e↵ Touch, int16_t nX, int16_t nY)

Trigger an element's touch event.

- void [gslc_ResetElem](#) ([gslc_tsElem](#) *pElem)

Initialize an Element struct.

- void [gslc_ResetFont](#) ([gslc_tsFont](#) *pFont)

Initialize a Font struct.

- void [gslc_ElemDestruct](#) ([gslc_tsElem](#) *pElem)

Free up any members associated with an element.

- void [gslc_CollectDestruct](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)

Free up any members associated with an element collection.

- void [gslc_PageDestruct](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage)

Free up any members associated with a page.

- void [gslc_GuiDestruct](#) ([gslc_tsGui](#) *pGui)

Free up any surfaces associated with the GUI, pages, collections and elements.

- void [gslc_CollectReset](#) ([gslc_tsCollect](#) *pCollect, [gslc_tsElem](#) *asElem, uint16_t nElemMax, [gslc_tsElemRef](#) *asElemRef, uint16_t nElemRefMax)

Reset the members of an element collection.

- bool [gslc_CollectFindFocusStep](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, bool bNext, bool *pbWrapped, int16_t *pnElemInd)

- [gslc_tsElemRef](#) * [gslc_CollectFindElemById](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, int16_t nElemId)

Find an element in a collection by its Element ID.

- int [gslc_CollectGetNextId](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)

Allocate the next available Element ID in a collection.

- [gslc_tsElemRef](#) * [gslc_CollectGetElemRefTracked](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)

Get the element within a collection that is currently being tracked.

- void [gslc_CollectSetElemTracked](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsElemRef](#) *pElemRef)

Set the element within a collection that is currently being tracked.

- [gslc_tsElemRef](#) * [gslc_CollectFindElemFromCoord](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, int16_t nX, int16_t nY)

Find an element in a collection by a coordinate coordinate.

- int16_t [gslc_CollectGetFocus](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)

Get the element index within a collection that is currently in focus.

- void [gslc_CollectSetFocus](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, int16_t nElemInd)

Set the element index within a collection that is currently in focus.

- void [gslc_CollectSetParent](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsElemRef](#) *pElemRefParent)

Assign the parent element reference to all elements within a collection.

Variables

- [GSLC_CB_DEBUG_OUT](#) [g_pfDebugOut](#)

Global debug output function.

- uint16_t [m_nLUTSinFOX16](#) [257]

- const char [GSLC_PMEM_ERRSTR_NULL](#) []

- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.36.1 Enumeration Type Documentation

9.36.1.1 enum gslc_teDebugPrintState

Enumerator

GSLC_S_DEBUG_PRINT_NORM
GSLC_S_DEBUG_PRINT_TOKEN
GSLC_S_DEBUG_PRINT_UINT16
GSLC_S_DEBUG_PRINT_CHAR
GSLC_S_DEBUG_PRINT_STR
GSLC_S_DEBUG_PRINT_STR_P

9.36.2 Function Documentation

9.36.2.1 void `gslc_DrawFillSectorBase` (`gslc_tsGui` * *pGui*, `int16_t` *nQuality*, `int16_t` *nMidX*, `int16_t` *nMidY*, `int16_t` *nRad1*, `int16_t` *nRad2*, `gslc_tsColor` *cArcStart*, `gslc_tsColor` *cArcEnd*, `bool` *bGradient*, `int16_t` *nAngGradStart*, `int16_t` *nAngGradRange*, `int16_t` *nAngSecStart*, `int16_t` *nAngSecEnd*)

9.36.2.2 `bool` `gslc_FontSetBase` (`gslc_tsGui` * *pGui*, `uint8_t` *nFontInd*, `int16_t` *nFontId*, `gslc_teFontRefType` *eFontRefType*, `const void` * *pvFontRef*, `uint16_t` *nFontSz*)

9.36.2.3 void `gslc_OrderCoord` (`int16_t` * *pnX0*, `int16_t` * *pnY0*, `int16_t` * *pnX1*, `int16_t` * *pnY1*)

9.36.2.4 void `gslc_SwapCoords` (`int16_t` * *pnXa*, `int16_t` * *pnYa*, `int16_t` * *pnXb*, `int16_t` * *pnYb*)

9.36.3 Variable Documentation

9.36.3.1 `const char` `ERRSTR_NULL`[]

9.36.3.2 `const char` `GSLC_PMEM_ERRSTR_PXD_NULL`[]

9.36.3.3 `GSLC_CB_DEBUG_OUT` `g_pfDebugOut`

Global debug output function.

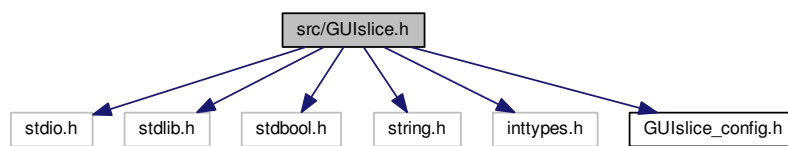
- The user assigns this function via [gslc_InitDebug\(\)](#)

9.36.3.4 uint16_t m_nLUTSinF0X16

9.37 src/GUISlice.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <string.h>
#include <inttypes.h>
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [gslc_tsRect](#)
Rectangular region. Defines X,Y corner coordinates plus dimensions.
- struct [gslc_tsPt](#)
Define point coordinates.
- struct [gslc_tsColor](#)
Color structure. Defines RGB triplet.
- struct [gslc_tsEvent](#)
Event structure.
- struct [gslc_tsEventTouch](#)
Structure used to pass touch data through event.
- struct [gslc_tsFont](#)
Font reference structure.
- struct [gslc_tsImgRef](#)
Image reference structure.
- struct [gslc_tsElemRef](#)
Element reference structure.
- struct [gslc_tsElem](#)
Element Struct.
- struct [gslc_tsCollect](#)
Element collection struct.

- struct [gslc_tsPage](#)
Page structure.
- struct [gslc_tsInputMap](#)
Input mapping.
- struct [gslc_tsGui](#)
GUI structure.

Macros

- #define [GSLC_PMEM](#)
- #define [GSLC_2PI](#)
- #define [GSLC_ELEM_FEA_VALID](#)
Element features type.
- #define [GSLC_ELEM_FEA_ROUND_EN](#)
Element is drawn with a rounded profile.
- #define [GSLC_ELEM_FEA_CLICK_EN](#)
Element accepts touch presses.
- #define [GSLC_ELEM_FEA_GLOW_EN](#)
Element supports glowing state.
- #define [GSLC_ELEM_FEA_FRAME_EN](#)
Element is drawn with a frame.
- #define [GSLC_ELEM_FEA_FILL_EN](#)
Element is drawn with a fill.
- #define [GSLC_ELEM_FEA_NONE](#)
Element default (no features set)
- #define [GSLC_ALIGNV_TOP](#)
Element text alignment.
- #define [GSLC_ALIGNV_MID](#)
Vertical align to middle.
- #define [GSLC_ALIGNV_BOT](#)
Vertical align to bottom.
- #define [GSLC_ALIGNH_LEFT](#)
Horizontal align to left.
- #define [GSLC_ALIGNH_MID](#)
Horizontal align to middle.
- #define [GSLC_ALIGNH_RIGHT](#)
Horizontal align to right.
- #define [GSLC_ALIGN_TOP_LEFT](#)
Align to top-left.
- #define [GSLC_ALIGN_TOP_MID](#)
Align to middle of top.
- #define [GSLC_ALIGN_TOP_RIGHT](#)
Align to top-right.
- #define [GSLC_ALIGN_MID_LEFT](#)
Align to middle of left side.
- #define [GSLC_ALIGN_MID_MID](#)
Align to center.
- #define [GSLC_ALIGN_MID_RIGHT](#)
Align to middle of right side.
- #define [GSLC_ALIGN_BOT_LEFT](#)

- Align to bottom-left.*
- #define `GSLC_ALIGN_BOT_MID`
 - Align to middle of bottom.*
- #define `GSLC_ALIGN_BOT_RIGHT`
 - Align to bottom-right.*
- #define `GSLC_COL_RED_DK4`
 - Basic color definition.*
- #define `GSLC_COL_RED_DK3`
 - Red (dark3)*
- #define `GSLC_COL_RED_DK2`
 - Red (dark2)*
- #define `GSLC_COL_RED_DK1`
 - Red (dark1)*
- #define `GSLC_COL_RED`
 - Red.*
- #define `GSLC_COL_RED_LT1`
 - Red (light1)*
- #define `GSLC_COL_RED_LT2`
 - Red (light2)*
- #define `GSLC_COL_RED_LT3`
 - Red (light3)*
- #define `GSLC_COL_RED_LT4`
 - Red (light4)*
- #define `GSLC_COL_GREEN_DK4`
 - Green (dark4)*
- #define `GSLC_COL_GREEN_DK3`
 - Green (dark3)*
- #define `GSLC_COL_GREEN_DK2`
 - Green (dark2)*
- #define `GSLC_COL_GREEN_DK1`
 - Green (dark1)*
- #define `GSLC_COL_GREEN`
 - Green.*
- #define `GSLC_COL_GREEN_LT1`
 - Green (light1)*
- #define `GSLC_COL_GREEN_LT2`
 - Green (light2)*
- #define `GSLC_COL_GREEN_LT3`
 - Green (light3)*
- #define `GSLC_COL_GREEN_LT4`
 - Green (light4)*
- #define `GSLC_COL_BLUE_DK4`
 - Blue (dark4)*
- #define `GSLC_COL_BLUE_DK3`
 - Blue (dark3)*
- #define `GSLC_COL_BLUE_DK2`
 - Blue (dark2)*
- #define `GSLC_COL_BLUE_DK1`
 - Blue (dark1)*
- #define `GSLC_COL_BLUE`
 - Blue.*

- #define `GSLC_COL_BLUE_LT1`
Blue (light1)
- #define `GSLC_COL_BLUE_LT2`
Blue (light2)
- #define `GSLC_COL_BLUE_LT3`
Blue (light3)
- #define `GSLC_COL_BLUE_LT4`
Blue (light4)
- #define `GSLC_COL_BLACK`
Black.
- #define `GSLC_COL_GRAY_DK3`
Gray (dark)
- #define `GSLC_COL_GRAY_DK2`
Gray (dark)
- #define `GSLC_COL_GRAY_DK1`
Gray (dark)
- #define `GSLC_COL_GRAY`
Gray.
- #define `GSLC_COL_GRAY_LT1`
Gray (light1)
- #define `GSLC_COL_GRAY_LT2`
Gray (light2)
- #define `GSLC_COL_GRAY_LT3`
Gray (light3)
- #define `GSLC_COL_WHITE`
White.
- #define `GSLC_COL_YELLOW`
Yellow.
- #define `GSLC_COL_YELLOW_DK`
Yellow (dark)
- #define `GSLC_COL_PURPLE`
Purple.
- #define `GSLC_COL_CYAN`
Cyan.
- #define `GSLC_COL_MAGENTA`
Magenta.
- #define `GSLC_COL_TEAL`
Teal.
- #define `GSLC_COL_ORANGE`
Orange.
- #define `GSLC_COL_BROWN`
Brown.
- #define `GSLC_COLMONO_BLACK`
Black.
- #define `GSLC_COLMONO_WHITE`
White.
- #define `TOUCH_ROTATION_DATA`
Additional definitions for Touch Handling These macros define the transforms used in remapping the touchscreen inputs on the basis of the GUI nRotation setting.
- #define `TOUCH_ROTATION_SWAPXY`(rotation)
- #define `TOUCH_ROTATION_FLIPX`(rotation)

- `#define TOUCH_ROTATION_FLIPY(rotation)`
- `#define GSLC_ELEMREF_DEFAULT`
Define the default element reference flags for new elements.
- `#define TOUCH_ROTATION_DATA`
Additional definitions for Touch Handling These macros define the transforms used in remapping the touchscreen inputs on the basis of the GUI nRotation setting.
- `#define TOUCH_ROTATION_SWAPXY(rotation)`
- `#define TOUCH_ROTATION_FLIPX(rotation)`
- `#define TOUCH_ROTATION_FLIPY(rotation)`
- `#define GSLC_DEBUG_PRINT(sFmt, ...)`
Macro to enable optional debug output.
- `#define GSLC_DEBUG2_PRINT(sFmt, ...)`
- `#define GSLC_DEBUG_PRINT_CONST(sFmt, ...)`
- `#define GSLC_DEBUG2_PRINT_CONST(sFmt, ...)`
- `#define gslc_ElemCreateTxt_P(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, pFont, colTxt, colFrame, colFill, nAlignTxt, bFrameEn, bFillEn)`
Create a read-only text element.
- `#define gslc_ElemCreateTxt_P_R(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, strLength, pFont, colTxt, colFrame, colFill, nAlignTxt, bFrameEn, bFillEn)`
Create a read-write text element (element in Flash, string in RAM)
- `#define gslc_ElemCreateTxt_P_R_ext(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, strLength, pFont, colTxt, colTxtGlow, colFrame, colFill, nAlignTxt, nMarginX, nMarginY, bFrameEn, bFillEn, bClickEn, bGlowEn, pfuncXEvent, pfuncXDraw, pfuncXTouch, pfuncXTick)`
Create a read-write text element (element in Flash, string in RAM) with extended customization options.
- `#define gslc_ElemCreateBox_P(pGui, nElemId, nPage, nX, nY, nW, nH, colFrame, colFill, bFrameEn, bFillEn, pfuncXDraw, pfuncXTick)`
Create a read-only box element.
- `#define gslc_ElemCreateLine_P(pGui, nElemId, nPage, nX0, nY0, nX1, nY1, colFill)`
Create a read-only line element.
- `#define gslc_ElemCreateBtnTxt_P(pGui, nElemId, nPage, nX, nY, nW, nH, strTxt, pFont, colTxt, colFrame, colFill, colFrameGlow, colFillGlow, nAlignTxt, bFrameEn, bFillEn, callFunc, extraData)`
Create a text button element.

Typedefs

- `typedef int16_t(* GSLC_CB_DEBUG_OUT) (char ch)`
- `typedef struct gslc_tsElem gslc_tsElem`
Element Struct.
- `typedef struct gslc_tsEvent gslc_tsEvent`
Event structure.
- `typedef bool(* GSLC_CB_EVENT) (void *pvGui, gslc_tsEvent sEvent)`
Callback function for element drawing.
- `typedef bool(* GSLC_CB_DRAW) (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
Callback function for element drawing.
- `typedef bool(* GSLC_CB_TOUCH) (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
Callback function for element touch tracking.
- `typedef bool(* GSLC_CB_TICK) (void *pvGui, void *pvElemRef)`
Callback function for element tick.
- `typedef bool(* GSLC_CB_PIN_POLL) (void *pvGui, int16_t *pnPinInd, int16_t *pnPinVal)`
Callback function for pin polling.
- `typedef bool(* GSLC_CB_INPUT) (void *pvGui, void *pvElemRef, int16_t nStatus, void *pvData)`

- Callback function for element input ready.
- typedef struct [gslc_tsRect](#) [gslc_tsRect](#)
Rectangular region. Defines X,Y corner coordinates plus dimensions.
- typedef struct [gslc_tsPt](#) [gslc_tsPt](#)
Define point coordinates.
- typedef struct [gslc_tsColor](#) [gslc_tsColor](#)
Color structure. Defines RGB triplet.
- typedef struct [gslc_tsEventTouch](#) [gslc_tsEventTouch](#)
Structure used to pass touch data through event.

Enumerations

- enum [gslc_teElemId](#) {
GSLC_ID_USER_BASE, GSLC_ID_NONE, GSLC_ID_AUTO, GSLC_ID_TEMP,
GSLC_ID_AUTO_BASE }
Element ID enumerations.
- enum [gslc_tePageId](#) { GSLC_PAGE_USER_BASE, GSLC_PAGE_NONE }
Page ID enumerations.
- enum [gslc_teStackPage](#) { GSLC_STACK_BASE, GSLC_STACK_CUR, GSLC_STACK_OVERLAY, GSLC_↵
_STACK_MAX }
Define page stack.
- enum [gslc_teGroupId](#) { GSLC_GROUP_ID_USER_BASE, GSLC_GROUP_ID_NONE }
Group ID enumerations.
- enum [gslc_teFontId](#) { GSLC_FONT_USER_BASE, GSLC_FONT_NONE }
Font ID enumerations.
- enum [gslc_teElemInd](#) { GSLC_IND_NONE, GSLC_IND_FIRST }
Element Index enumerations.
- enum [gslc_teTypeCore](#) {
GSLC_TYPE_NONE, GSLC_TYPE_BKGND, GSLC_TYPE_BTN, GSLC_TYPE_TXT,
GSLC_TYPE_BOX, GSLC_TYPE_LINE, GSLC_TYPE_BASE_EXTEND }
Element type.
- enum [gslc_teInputRawEvent](#) {
GSLC_INPUT_NONE, GSLC_INPUT_TOUCH, GSLC_INPUT_KEY_DOWN, GSLC_INPUT_KEY_UP,
GSLC_INPUT_PIN_ASSERT, GSLC_INPUT_PIN_DEASSERT }
Raw input event types: touch, key, GPIOs.
- enum [gslc_teAction](#) {
GSLC_ACTION_UNDEF, GSLC_ACTION_NONE, GSLC_ACTION_FOCUS_PREV, GSLC_ACTION_FO↵
CUS_NEXT,
GSLC_ACTION_SELECT, GSLC_ACTION_SET_REL, GSLC_ACTION_SET_ABS, GSLC_ACTION_DE↵
BUG }
GUI Action Requested These actions are usually the result of an InputMap lookup.
- enum [gslc_tePin](#) {
GSLC_PIN_BTN_A, GSLC_PIN_BTN_A_LONG, GSLC_PIN_BTN_B, GSLC_PIN_BTN_B_LONG,
GSLC_PIN_BTN_C, GSLC_PIN_BTN_C_LONG, GSLC_PIN_BTN_D, GSLC_PIN_BTN_D_LONG,
GSLC_PIN_BTN_E, GSLC_PIN_BTN_E_LONG, GSLC_PIN_BTN_UP, GSLC_PIN_BTN_DOWN,
GSLC_PIN_BTN_LEFT, GSLC_PIN_BTN_RIGHT, GSLC_PIN_BTN_SEL }
General purpose pin/button constants.
- enum [gslc_teTouch](#) {
GSLC_TOUCH_NONE, GSLC_TOUCH_TYPE_MASK, GSLC_TOUCH_COORD, GSLC_TOUCH_DIRECT,
GSLC_TOUCH_SUBTYPE_MASK, GSLC_TOUCH_DOWN, GSLC_TOUCH_DOWN_IN, GSLC_TOUCH_↵
_DOWN_OUT,
GSLC_TOUCH_UP, GSLC_TOUCH_UP_IN, GSLC_TOUCH_UP_OUT, GSLC_TOUCH_MOVE,
GSLC_TOUCH_MOVE_IN, GSLC_TOUCH_MOVE_OUT, GSLC_TOUCH_FOCUS_ON, GSLC_TOUCH_↵
FOCUS_OFF,
GSLC_TOUCH_FOCUS_SELECT, GSLC_TOUCH_SET_REL, GSLC_TOUCH_SET_ABS }
Touch event types.

Processed event from input raw events and actions.

- enum `gslc_telnitStat` { `GSLC_INITSTAT_UNDEF`, `GSLC_INITSTAT_INACTIVE`, `GSLC_INITSTAT_FAIL`, `GSLC_INITSTAT_ACTIVE` }

Status of a module's initialization.

- enum `gslc_teEventType` { `GSLC_EVT_NONE`, `GSLC_EVT_DRAW`, `GSLC_EVT_TOUCH`, `GSLC_EVT_TICK`, `GSLV_EVT_CUSTOM` }

Event types.

- enum `gslc_teEventSubType` { `GSLC_EVTSUB_NONE`, `GSLC_EVTSUB_DRAW_NEEDED`, `GSLC_EVTSUB_DRAW_FORCE` }

Event sub-types.

- enum `gslc_teRedrawType` { `GSLC_REDRAW_NONE`, `GSLC_REDRAW_FULL`, `GSLC_REDRAW_INC` }

Redraw types.

- enum `gslc_teFontRefType` { `GSLC_FONTREF_FNAME`, `GSLC_FONTREF_PTR` }

Font Reference types.

- enum `gslc_teFontRefMode` { `GSLC_FONTREF_MODE_DEFAULT`, `GSLC_FONTREF_MODE_1`, `GSLC_FONTREF_MODE_2`, `GSLC_FONTREF_MODE_3` }

Font Reference modes.

- enum `gslc_teElemRefFlags` { `GSLC_ELEMREF_NONE`, `GSLC_ELEMREF_SRC_RAM`, `GSLC_ELEMREF_SRC_PROG`, `GSLC_ELEMREF_SRC_CONST`, `GSLC_ELEMREF_REDRAW_NONE`, `GSLC_ELEMREF_REDRAW_FULL`, `GSLC_ELEMREF_REDRAW_INC`, `GSLC_ELEMREF_GLOWING`, `GSLC_ELEMREF_VISIBLE`, `GSLC_ELEMREF_SRC`, `GSLC_ELEMREF_REDRAW_MASK` }

Element reference flags: Describes characteristics of an element.

- enum `gslc_telmgRefFlags` { `GSLC_IMGREF_NONE`, `GSLC_IMGREF_SRC_FILE`, `GSLC_IMGREF_SRC_SD`, `GSLC_IMGREF_SRC_RAM`, `GSLC_IMGREF_SRC_PROG`, `GSLC_IMGREF_FMT_BMP24`, `GSLC_IMGREF_FMT_BMP16`, `GSLC_IMGREF_FMT_RAW1`, `GSLC_IMGREF_FMT_JPG`, `GSLC_IMGREF_SRC`, `GSLC_IMGREF_FMT` }

Image reference flags: Describes characteristics of an image reference.

- enum `gslc_teTxtFlags` { `GSLC_TXT_MEM_RAM`, `GSLC_TXT_MEM_PROG`, `GSLC_TXT_ALLOC_NONE`, `GSLC_TXT_ALLOC_INT`, `GSLC_TXT_ALLOC_EXT`, `GSLC_TXT_ENC_PLAIN`, `GSLC_TXT_ENC_UTF8`, `GSLC_TXT_MEM`, `GSLC_TXT_ALLOC`, `GSLC_TXT_ENC`, `GSLC_TXT_DEFAULT` }

Text reference flags: Describes the characteristics of a text string (ie.

Functions

- char * `gslc_GetVer` (`gslc_tsGui` *pGui)
Get the GUIslice version number.
- const char * `gslc_GetNameDisp` (`gslc_tsGui` *pGui)
Get the GUIslice display driver name.
- const char * `gslc_GetNameTouch` (`gslc_tsGui` *pGui)
Get the GUIslice touch driver name.
- void * `gslc_GetDriverDisp` (`gslc_tsGui` *pGui)
Get the native display driver instance.
- void * `gslc_GetDriverTouch` (`gslc_tsGui` *pGui)
Get the native touch driver instance.
- bool `gslc_Init` (`gslc_tsGui` *pGui, void *pvDriver, `gslc_tsPage` *asPage, uint8_t nMaxPage, `gslc_tsFont` *asFont, uint8_t nMaxFont)

- Initialize the GUISlice library.*

 - void [gslc_InitDebug](#) ([GSLC_CB_DEBUG_OUT](#) pfunc)

Initialize debug output.
- void [gslc_DebugPrintf](#) (const char *pFmt,...)

Optimized printf routine for GUISlice debug/error output.
- bool [gslc_GuiRotate](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation)

Dynamically change rotation, automatically adapt touchscreen axes swap/flip.
- void [gslc_Quit](#) ([gslc_tsGui](#) *pGui)

Exit the GUISlice environment.
- void [gslc_Update](#) ([gslc_tsGui](#) *pGui)

Perform main GUISlice handling functions.
- bool [gslc_SetBkgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tslmgRef](#) slmgRef)

Configure the background to use a bitmap image.
- bool [gslc_SetBkgndColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)

Configure the background to use a solid color.
- bool [gslc_SetTransparentColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)

Configure the color to use for image transparency.
- bool [gslc_SetClipRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) *pRect)

Set the clipping rectangle for further drawing.
- bool [gslc_IsInRect](#) (int16_t nSelX, int16_t nSelY, [gslc_tsRect](#) rRect)

Determine if a coordinate is inside of a rectangular region.
- [gslc_tsRect](#) [gslc_ExpandRect](#) ([gslc_tsRect](#) rRect, int16_t nExpandW, int16_t nExpandH)

Expand or contract a rectangle in width and/or height (equal amounts on both side), based on the centerpoint of the rectangle.
- bool [gslc_IsInWH](#) (int16_t nSelX, int16_t nSelY, uint16_t nWidth, uint16_t nHeight)

Determine if a coordinate is inside of a width x height region.
- void [gslc_UnionRect](#) ([gslc_tsRect](#) *pRect, [gslc_tsRect](#) rAddRect)

Expand a rect to include another rect.
- void [gslc_InvalidateRgnReset](#) ([gslc_tsGui](#) *pGui)

Reset the invalidation region.
- void [gslc_InvalidateRgnPage](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage)

Include an entire page (eg.
- void [gslc_InvalidateRgnScreen](#) ([gslc_tsGui](#) *pGui)

Mark the entire screen as invalidated.
- void [gslc_InvalidateRgnAdd](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rAddRect)

Add a rectangular region to the invalidation region.
- bool [gslc_ClipPt](#) ([gslc_tsRect](#) *pClipRect, int16_t nX, int16_t nY)

Perform basic clipping of a single point to a clipping region.
- bool [gslc_ClipLine](#) ([gslc_tsRect](#) *pClipRect, int16_t *pnX0, int16_t *pnY0, int16_t *pnX1, int16_t *pnY1)

Perform basic clipping of a line to a clipping region.
- bool [gslc_ClipRect](#) ([gslc_tsRect](#) *pClipRect, [gslc_tsRect](#) *pRect)

Perform basic clipping of a rectangle to a clipping region.
- [gslc_tslmgRef](#) [gslc_GetImageFromFile](#) (const char *pFname, [gslc_telmgRefFlags](#) eFmt)

Create an image reference to a bitmap file in LINUX filesystem.
- [gslc_tslmgRef](#) [gslc_GetImageFromSD](#) (const char *pFname, [gslc_telmgRefFlags](#) eFmt)

Create an image reference to a bitmap file in SD card.
- [gslc_tslmgRef](#) [gslc_GetImageFromRam](#) (unsigned char *plmgBuf, [gslc_telmgRefFlags](#) eFmt)

Create an image reference to a bitmap in SRAM.
- [gslc_tslmgRef](#) [gslc_GetImageFromProg](#) (const unsigned char *plmgBuf, [gslc_telmgRefFlags](#) eFmt)

Create an image reference to a bitmap in program memory (PROGMEM)
- void [gslc_PolarToXY](#) (uint16_t nRad, int16_t n64Ang, int16_t *nDX, int16_t *nDY)

- Convert polar coordinate to cartesian.*

 - `int16_t gslc_sinFX (int16_t n64Ang)`

Calculate fixed-point sine function from fractional degrees.
- `int16_t gslc_cosFX (int16_t n64Ang)`

Calculate fixed-point cosine function from fractional degrees.
- `gslc_tsColor gslc_ColorBlend2 (gslc_tsColor colStart, gslc_tsColor colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)`

Create a color based on a blend between two colors.
- `gslc_tsColor gslc_ColorBlend3 (gslc_tsColor colStart, gslc_tsColor colMid, gslc_tsColor colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)`

Create a color based on a blend between three colors.
- `bool gslc_ColorEqual (gslc_tsColor a, gslc_tsColor b)`

Check whether two colors are equal.
- `void gslc_DrawSetPixel (gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)`

Set a pixel on the active screen to the given color with lock.
- `void gslc_DrawLine (gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)`

Draw an arbitrary line using Bresenham's algorithm.
- `void gslc_DrawLineH (gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nW, gslc_tsColor nCol)`

Draw a horizontal line.
- `void gslc_DrawLineV (gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nH, gslc_tsColor nCol)`

Draw a vertical line.
- `void gslc_DrawLinePolar (gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nRadStart, uint16_t nRadEnd, int16_t n64Ang, gslc_tsColor nCol)`

Draw a polar ray segment.
- `void gslc_DrawFrameRect (gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)`

Draw a framed rectangle.
- `void gslc_DrawFrameRoundRect (gslc_tsGui *pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a framed rounded rectangle.
- `void gslc_DrawFillRect (gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)`

Draw a filled rectangle.
- `void gslc_DrawFillRoundRect (gslc_tsGui *pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a filled rounded rectangle.
- `void gslc_DrawFrameCircle (gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)`

Draw a framed circle.
- `void gslc_DrawFillCircle (gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)`

Draw a filled circle.
- `void gslc_DrawFrameTriangle (gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a framed triangle.
- `void gslc_DrawFillTriangle (gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a filled triangle.
- `void gslc_DrawFrameQuad (gslc_tsGui *pGui, gslc_tsPt *psPt, gslc_tsColor nCol)`

Draw a framed quadrilateral.
- `void gslc_DrawFillQuad (gslc_tsGui *pGui, gslc_tsPt *psPt, gslc_tsColor nCol)`

Draw a filled quadrilateral.
- `void gslc_DrawFillGradSector (gslc_tsGui *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, gslc_tsColor cArcStart, gslc_tsColor cArcEnd, int16_t nAngSecStart, int16_t nAngSecEnd, int16_t nAngGradStart, int16_t nAngGradRange)`

Draw a gradient filled sector of a circle with support for inner and outer radius.

- void [gslc_DrawFillSector](#) ([gslc_tsGui](#) *pGui, int16_t nQuality, int16_t nMidX, int16_t nMidY, int16_t nRad1, int16_t nRad2, [gslc_tsColor](#) cArc, int16_t nAngSecStart, int16_t nAngSecEnd)
Draw a flat filled sector of a circle with support for inner and outer radius.
- bool [gslc_FontAdd](#) ([gslc_tsGui](#) *pGui, int16_t nFontId, [gslc_teFontRefType](#) eFontRefType, const void *pv↔FontRef, uint16_t nFontSz)
Load a font into the local font cache and assign font ID (nFontId).
- bool [gslc_FontSet](#) ([gslc_tsGui](#) *pGui, int16_t nFontId, [gslc_teFontRefType](#) eFontRefType, const void *pv↔FontRef, uint16_t nFontSz)
Load a font into the local font cache and store as font ID (nFontId)
- [gslc_tsFont](#) * [gslc_FontGet](#) ([gslc_tsGui](#) *pGui, int16_t nFontId)
Fetch a font from its ID value.
- bool [gslc_FontSetMode](#) ([gslc_tsGui](#) *pGui, int16_t nFontId, [gslc_teFontRefMode](#) eFontMode)
Set the font operating mode.
- int [gslc_GetPageCur](#) ([gslc_tsGui](#) *pGui)
Fetch the current page ID.
- void [gslc_SetStackPage](#) ([gslc_tsGui](#) *pGui, uint8_t nStackPos, int16_t nPageId)
Assign a page to the page stack.
- void [gslc_SetStackState](#) ([gslc_tsGui](#) *pGui, uint8_t nStackPos, bool bActive, bool bDoDraw)
Change the status of a page in a page stack.
- void [gslc_SetPageBase](#) ([gslc_tsGui](#) *pGui, int16_t nPageId)
Assigns a page for the base layer in the page stack.
- void [gslc_SetPageCur](#) ([gslc_tsGui](#) *pGui, int16_t nPageId)
Select a page for the current layer in the page stack.
- void [gslc_SetPageOverlay](#) ([gslc_tsGui](#) *pGui, int16_t nPageId)
Select a page for the overlay layer in the page stack.
- void [gslc_PopupShow](#) ([gslc_tsGui](#) *pGui, int16_t nPageId, bool bModal)
Show a popup dialog.
- void [gslc_PopupHide](#) ([gslc_tsGui](#) *pGui)
Hides the currently active popup dialog.
- void [gslc_PageRedrawSet](#) ([gslc_tsGui](#) *pGui, bool bRedraw)
Update the need-redraw status for the current page.
- bool [gslc_PageRedrawGet](#) ([gslc_tsGui](#) *pGui)
Get the need-redraw status for the current page.
- void [gslc_PageAdd](#) ([gslc_tsGui](#) *pGui, int16_t nPageId, [gslc_tsElem](#) *psElem, uint16_t nMaxElem, [gslc_tsElemRef](#) *psElemRef, uint16_t nMaxElemRef)
Add a page to the GUI.
- [gslc_tsElemRef](#) * [gslc_PageFindElemById](#) ([gslc_tsGui](#) *pGui, int16_t nPageId, int16_t nElemId)
Find an element in the GUI by its Page ID and Element ID.
- [gslc_tsElemRef](#) * [gslc_ElemCreateTxt](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)
Create a Text Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateBtnTxt](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId, [GSLC_CB_TOUCH](#) cbTouch)
Create a textual Button Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateBtnImg](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) rElem, [gslc_tsImgRef](#) slmgRef, [gslc_tsImgRef](#) slmgRefSel, [GSLC_CB_TOUCH](#) cbTouch)
Create a graphical Button Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateBox](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsRect](#) r↔Elem)
Create a Box Element.
- [gslc_tsElemRef](#) * [gslc_ElemCreateLine](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1)

Create a Line Element.

- `gslc_tsElemRef * gslc_ElemCreateImg (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsRect r←Elem, gslc_tsImgRef sImgRef)`

Create an image Element.

- `int gslc_ElemGetId (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`

Get an Element ID from an element structure.

- `void gslc_ElemSetFillEn (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFillEn)`

Set the fill state for an Element.

- `void gslc_ElemSetFrameEn (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFrameEn)`

Set the frame state for an Element.

- `void gslc_ElemSetRoundEn (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bRoundEn)`

Set the rounded frame/fill state for an Element.

- `void gslc_ElemSetCol (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colFrame, gslc_tsColor colFill, gslc_tsColor colFillGlow)`

Update the common color selection for an Element.

- `void gslc_ElemSetGlowCol (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colFrameGlow, gslc_tsColor colFillGlow, gslc_tsColor colTxtGlow)`

Update the common color selection for glowing state of an Element.

- `void gslc_ElemSetGroup (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int nGroupId)`

Set the group ID for an element.

- `int gslc_ElemGetGroup (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`

Get the group ID for an element.

- `void gslc_ElemSetTxtAlign (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned nAlign)`

Set the alignment of a textual element (horizontal and vertical)

- `void gslc_ElemSetTxtMargin (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned nMargin)`

Set the margin around of a textual element.

- `void gslc_ElemSetTxtMarginXY (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int8_t nMarginX, int8_t n←MarginY)`

Set the margin around of a textual element (X & Y offsets can be different)

- `void gslc_ElemSetTxtStr (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, const char *pStr)`

Update the text string associated with an Element.

- `char * gslc_ElemGetTxtStr (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`

Fetch the current text string associated with an Element.

- `void gslc_ElemSetTxtCol (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colVal)`

Update the text string color associated with an Element ID.

- `void gslc_ElemSetTxtMem (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teTxtFlags eFlags)`

Update the text string location in memory.

- `void gslc_ElemSetTxtEnc (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teTxtFlags eFlags)`

Update the text string encoding mode.

- `void gslc_ElemUpdateFont (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int nFontId)`

Update the Font selected for an Element's text.

- `void gslc_ElemSetRedraw (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)`

Update the need-redraw status for an element.

- `gslc_teRedrawType gslc_ElemGetRedraw (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`

Get the need-redraw status for an element.

- `void gslc_ElemSetGlowEn (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bGlowEn)`

Update the glowing enable for an element.

- `void gslc_ElemSetClickEn (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bClickEn)`

Update the click enable for an element.

- `void gslc_ElemSetTouchFunc (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_TOUCH funcCb)`

Update the touch function callback for an element.

- void [gslc_ElemSetStyleFrom](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRefSrc, [gslc_tsElemRef](#) *pElemRefDest)
Copy style settings from one element to another.
- bool [gslc_ElemGetGlowEn](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the glowing enable for an element.
- void [gslc_ElemSetGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bGlowing)
Update the glowing indicator for an element.
- bool [gslc_ElemGetGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the glowing indicator for an element.
- void [gslc_ElemSetVisible](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, bool bVisible)
Update the visibility status for an element.
- bool [gslc_ElemGetVisible](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Get the visibility status for an element.
- bool [gslc_ElemGetOnScreen](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
Determine whether an element is visible on the screen.
- void [gslc_ElemSetDrawFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_DRAW](#) funcCb)
Assign the drawing callback function for an element.
- void [gslc_ElemSetTickFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_TICK](#) funcCb)
Assign the tick callback function for an element.
- bool [gslc_ElemOwnsCoord](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nX, int16_t nY, bool bOnlyClickEn)
Determine if a coordinate is inside of an element.
- bool [gslc_InitTouch](#) ([gslc_tsGui](#) *pGui, const char *acDev)
Initialize the touchscreen device driver.
- bool [gslc_GetTouch](#) ([gslc_tsGui](#) *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, [gslc_telInputRawEvent](#) *peInputEvent, int16_t *pnInputVal)
Initialize the touchscreen device driver.
- void [gslc_SetTouchRemapEn](#) ([gslc_tsGui](#) *pGui, bool bEn)
Configure touchscreen remapping.
- void [gslc_SetTouchRemapCal](#) ([gslc_tsGui](#) *pGui, uint16_t nXMin, uint16_t nXMax, uint16_t nYMin, uint16_t nYMax)
Configure touchscreen calibration values.
- void [gslc_SetTouchRemapYX](#) ([gslc_tsGui](#) *pGui, bool bSwap)
Configure touchscreen XY swap.
- void [gslc_SetPinPollFunc](#) ([gslc_tsGui](#) *pGui, [GSLC_CB_PIN_POLL](#) pfunc)
- void [gslc_InitInputMap](#) ([gslc_tsGui](#) *pGui, [gslc_tsInputMap](#) *asInputMap, uint8_t nInputMapMax)
- void [gslc_InputMapAdd](#) ([gslc_tsGui](#) *pGui, [gslc_telInputRawEvent](#) eInputEvent, int16_t nInputVal, [gslc_telAction](#) eAction, int16_t nActionVal)
- [gslc_tslmgRef](#) [gslc_ResetImage](#) ()
Create a blank image reference structure.
- [gslc_tsElem](#) [gslc_ElemCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPageId, int16_t nType, [gslc_tsRect](#) rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)
Create a new element with default styling.
- [gslc_tsElemRef](#) * [gslc_ElemAdd](#) ([gslc_tsGui](#) *pGui, int16_t nPageId, [gslc_tsElem](#) *pElem, [gslc_telElemRefFlags](#) eFlags)
Add the Element to the list of generated elements in the GUI environment.
- uint8_t [gslc_GetElemRefFlag](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint8_t nFlagMask)
Get the flags associated with an element reference.
- void [gslc_SetElemRefFlag](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint8_t nFlagMask, uint8_t nFlagVal)
Set the flags associated with an element reference.
- [gslc_tsElem](#) * [gslc_GetElemFromRef](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)

Returns a pointer to an element from an element reference, copying from FLASH to RAM if element is stored in PROGMEM.

- `gslc_tsElem * gslc_GetElemFromRefD (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nLineNum)`

Returns a pointer to an element from an element reference.

- `void * gslc_GetXDataFromRef (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nType, int16_t nLineNum)`

Returns a pointer to the data structure associated with an extended element.

- `void gslc_ElemSetImage (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsImgRef sImgRef, gslc_tsImgRef sImgRefSel)`

Set an element to use a bitmap image.

- `bool gslc_ElemDrawByRef (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)`

Draw an element to the active display.

- `void gslc_ElemDraw (gslc_tsGui *pGui, int16_t nPageId, int16_t nElemId)`

Draw an element to the active display.

- `void gslc_DrawTxtBase (gslc_tsGui *pGui, char *pStrBuf, gslc_tsRect rTxt, gslc_tsFont *pTxtFont, gslc_teTxtFlags eTxtFlags, int8_t eTxtAlign, gslc_tsColor colTxt, gslc_tsColor colBg, int16_t nMarginW, int16_t nMarginH)`

Draw text with full text justification.

- `void gslc_SetRoundRadius (gslc_tsGui *pGui, uint8_t nRadius)`

Set the global rounded radius.

- `bool gslc_PageEvent (void *pvGui, gslc_tsEvent sEvent)`

Common event handler function for a page.

- `void gslc_PageRedrawGo (gslc_tsGui *pGui)`

Redraw all elements on the active page.

- `void gslc_PageFlipSet (gslc_tsGui *pGui, bool bNeeded)`

Indicate whether the screen requires page flip.

- `bool gslc_PageFlipGet (gslc_tsGui *pGui)`

Get state of pending page flip state.

- `void gslc_PageFlipGo (gslc_tsGui *pGui)`

Update the visible screen if page has been marked for flipping.

- `gslc_tsPage * gslc_PageFindById (gslc_tsGui *pGui, int16_t nPageId)`

Find a page in the GUI by its ID.

- `void gslc_PageRedrawCalc (gslc_tsGui *pGui)`

Perform a redraw calculation on the page to determine if additional elements should also be redrawn.

- `int16_t gslc_PageFocusStep (gslc_tsGui *pGui, gslc_tsPage *pPage, bool bNext)`

- `gslc_tsEvent gslc_EventCreate (gslc_tsGui *pGui, gslc_teEventType eType, uint8_t nSubType, void *pvScope, void *pvData)`

Create an event structure.

- `bool gslc_ElemEvent (void *pvGui, gslc_tsEvent sEvent)`

Common event handler function for an element.

- `bool gslc_ElemSendEventTouch (gslc_tsGui *pGui, gslc_tsElemRef *pElemRefTracked, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

Trigger an element's touch event.

- `void gslc_CollectReset (gslc_tsCollect *pCollect, gslc_tsElem *asElem, uint16_t nElemMax, gslc_tsElemRef *asElemRef, uint16_t nElemRefMax)`

Reset the members of an element collection.

- `gslc_tsElemRef * gslc_CollectElemAdd (gslc_tsGui *pGui, gslc_tsCollect *pCollect, const gslc_tsElem *pElem, gslc_teElemRefFlags eFlags)`

Add an element to a collection.

- `bool gslc_CollectGetRedraw (gslc_tsGui *pGui, gslc_tsCollect *pCollect)`

Determine if any elements in a collection need redraw.

- `gslc_tsElemRef * gslc_CollectFindElemById (gslc_tsGui *pGui, gslc_tsCollect *pCollect, int16_t nElemId)`

- Find an element in a collection by its Element ID.*

 - [gslc_tsElemRef](#) * [gslc_CollectFindElemFromCoord](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, int16_t nX, int16_t nY)
- Find an element in a collection by a coordinate coordinate.*

 - int [gslc_CollectGetNextId](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)
- Allocate the next available Element ID in a collection.*

 - [gslc_tsElemRef](#) * [gslc_CollectGetElemRefTracked](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)
- Get the element within a collection that is currently being tracked.*

 - void [gslc_CollectSetElemTracked](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsElemRef](#) *pElemRef)
- Set the element within a collection that is currently being tracked.*

 - int16_t [gslc_CollectGetFocus](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)
- Get the element index within a collection that is currently in focus.*

 - void [gslc_CollectSetFocus](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, int16_t nElemInd)
- Set the element index within a collection that is currently in focus.*

 - bool [gslc_CollectFindFocusStep](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, bool bNext, bool *pbWrapped, int16_t *pnElemInd)
- Assign the parent element reference to all elements within a collection.*

 - void [gslc_CollectSetParent](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsElemRef](#) *pElemRefParent)
- Common event handler function for an element collection.*

 - bool [gslc_CollectEvent](#) (void *pvGui, [gslc_tsEvent](#) sEvent)
- Handle touch events within the element collection.*

 - void [gslc_CollectTouch](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsEventTouch](#) *pEventTouch)
- Handle dispatch of touch (up,down,move) events to compound elements sub elements.*

 - bool [gslc_CollectTouchCompound](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nRelX, int16_t nRelY, [gslc_tsCollect](#) *pCollect)
- Handle direct input events within the element collection.*

 - void [gslc_CollectInput](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsEventTouch](#) *pEventTouch)
- Handles a touch event and performs the necessary tracking, glowing and selection actions depending on the press state.*

 - void [gslc_TrackTouch](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage, int16_t nX, int16_t nY, uint16_t nPress)
- Handles a direct input event and performs the necessary tracking, glowing and selection actions depending on the state.*

 - void [gslc_TrackInput](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage, [gslc_teInputRawEvent](#) eInputEvent, int16_t nInputVal)
- Free up any surfaces associated with the GUI, pages, collections and elements.*

 - bool [gslc_InputMapLookup](#) ([gslc_tsGui](#) *pGui, [gslc_teInputRawEvent](#) eInputEvent, int16_t nInputVal, [gslc_teAction](#) *peAction, int16_t *pnActionVal)
- Free up any members associated with a page.*

 - void [gslc_GuiDestruct](#) ([gslc_tsGui](#) *pGui)
- Free up any members associated with an element collection.*

 - void [gslc_PageDestruct](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage)
- Free up any members associated with an element.*

 - void [gslc_CollectDestruct](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)
- Initialize a Font struct.*

 - void [gslc_ElemDestruct](#) ([gslc_tsElem](#) *pElem)
- Initialize an Element struct.*

 - void [gslc_ResetFont](#) ([gslc_tsFont](#) *pFont)
- Initialize an Element struct.*

 - void [gslc_ResetElem](#) ([gslc_tsElem](#) *pElem)

Variables

- [GSLC_CB_DEBUG_OUT](#) [g_pfDebugOut](#)
- Global debug output function.*

9.37.1 Macro Definition Documentation

9.37.1.1 `#define GSLC_2PI`

9.37.1.2 `#define GSLC_ALIGN_BOT_LEFT`

Align to bottom-left.

9.37.1.3 `#define GSLC_ALIGN_BOT_MID`

Align to middle of bottom.

9.37.1.4 `#define GSLC_ALIGN_BOT_RIGHT`

Align to bottom-right.

9.37.1.5 `#define GSLC_ALIGN_MID_LEFT`

Align to middle of left side.

9.37.1.6 `#define GSLC_ALIGN_MID_MID`

Align to center.

9.37.1.7 `#define GSLC_ALIGN_MID_RIGHT`

Align to middle of right side.

9.37.1.8 `#define GSLC_ALIGN_TOP_LEFT`

Align to top-left.

9.37.1.9 `#define GSLC_ALIGN_TOP_MID`

Align to middle of top.

9.37.1.10 `#define GSLC_ALIGN_TOP_RIGHT`

Align to top-right.

9.37.1.11 `#define GSLC_ALIGNH_LEFT`

Horizontal align to left.

9.37.1.12 `#define GSLC_ALIGNH_MID`

Horizontal align to middle.

9.37.1.13 `#define GSLC_ALIGNH_RIGHT`

Horizontal align to right.

9.37.1.14 `#define GSLC_ALIGNV_BOT`

Vertical align to bottom.

9.37.1.15 `#define GSLC_ALIGNV_MID`

Vertical align to middle.

9.37.1.16 `#define GSLC_ALIGNV_TOP`

Element text alignment.

Vertical align to top

9.37.1.17 `#define GSLC_COL_BLACK`

Black.

9.37.1.18 `#define GSLC_COL_BLUE`

Blue.

9.37.1.19 `#define GSLC_COL_BLUE_DK1`

Blue (dark1)

9.37.1.20 `#define GSLC_COL_BLUE_DK2`

Blue (dark2)

9.37.1.21 `#define GSLC_COL_BLUE_DK3`

Blue (dark3)

9.37.1.22 `#define GSLC_COL_BLUE_DK4`

Blue (dark4)

9.37.1.23 `#define GSLC_COL_BLUE_LT1`

Blue (light1)

9.37.1.24 `#define GSLC_COL_BLUE_LT2`

Blue (light2)

9.37.1.25 `#define GSLC_COL_BLUE_LT3`

Blue (light3)

9.37.1.26 `#define GSLC_COL_BLUE_LT4`

Blue (light4)

9.37.1.27 `#define GSLC_COL_BROWN`

Brown.

9.37.1.28 `#define GSLC_COL_CYAN`

Cyan.

9.37.1.29 `#define GSLC_COL_GRAY`

Gray.

9.37.1.30 `#define GSLC_COL_GRAY_DK1`

Gray (dark)

9.37.1.31 `#define GSLC_COL_GRAY_DK2`

Gray (dark)

9.37.1.32 `#define GSLC_COL_GRAY_DK3`

Gray (dark)

9.37.1.33 `#define GSLC_COL_GRAY_LT1`

Gray (light1)

9.37.1.34 `#define GSLC_COL_GRAY_LT2`

Gray (light2)

9.37.1.35 `#define GSLC_COL_GRAY_LT3`

Gray (light3)

9.37.1.36 `#define GSLC_COL_GREEN`

Green.

9.37.1.37 `#define GSLC_COL_GREEN_DK1`

Green (dark1)

9.37.1.38 `#define GSLC_COL_GREEN_DK2`

Green (dark2)

9.37.1.39 `#define GSLC_COL_GREEN_DK3`

Green (dark3)

9.37.1.40 `#define GSLC_COL_GREEN_DK4`

Green (dark4)

9.37.1.41 `#define GSLC_COL_GREEN_LT1`

Green (light1)

9.37.1.42 `#define GSLC_COL_GREEN_LT2`

Green (light2)

9.37.1.43 `#define GSLC_COL_GREEN_LT3`

Green (light3)

9.37.1.44 `#define GSLC_COL_GREEN_LT4`

Green (light4)

9.37.1.45 `#define GSLC_COL_MAGENTA`

Magenta.

9.37.1.46 `#define GSLC_COL_ORANGE`

Orange.

9.37.1.47 `#define GSLC_COL_PURPLE`

Purple.

9.37.1.48 `#define GSLC_COL_RED`

Red.

9.37.1.49 `#define GSLC_COL_RED_DK1`

Red (dark1)

9.37.1.50 `#define GSLC_COL_RED_DK2`

Red (dark2)

9.37.1.51 `#define GSLC_COL_RED_DK3`

Red (dark3)

9.37.1.52 `#define GSLC_COL_RED_DK4`

Basic color definition.

Red (dark4)

9.37.1.53 `#define GSLC_COL_RED_LT1`

Red (light1)

9.37.1.54 `#define GSLC_COL_RED_LT2`

Red (light2)

9.37.1.55 `#define GSLC_COL_RED_LT3`

Red (light3)

9.37.1.56 `#define GSLC_COL_RED_LT4`

Red (light4)

9.37.1.57 `#define GSLC_COL_TEAL`

Teal.

9.37.1.58 `#define GSLC_COL_WHITE`

White.

9.37.1.59 `#define GSLC_COL_YELLOW`

Yellow.

9.37.1.60 `#define GSLC_COL_YELLOW_DK`

Yellow (dark)

9.37.1.61 `#define GSLC_COLMONO_BLACK`

Black.

9.37.1.62 `#define GSLC_COLMONO_WHITE`

White.

9.37.1.63 `#define GSLC_ELEM_FEA_CLICK_EN`

Element accepts touch presses.

9.37.1.64 `#define GSLC_ELEM_FEA_FILL_EN`

Element is drawn with a fill.

9.37.1.65 `#define GSLC_ELEM_FEA_FRAME_EN`

Element is drawn with a frame.

9.37.1.66 `#define GSLC_ELEM_FEA_GLOW_EN`

Element supports glowing state.

9.37.1.67 `#define GSLC_ELEM_FEA_NONE`

Element default (no features set))

9.37.1.68 `#define GSLC_ELEM_FEA_ROUND_EN`

Element is drawn with a rounded profile.

9.37.1.69 `#define GSLC_ELEM_FEA_VALID`

Element features type.

Element record is valid

9.37.1.70 `#define GSLC_ELEMREF_DEFAULT`

Define the default element reference flags for new elements.

9.37.1.71 `#define GSLC_PMEM`

9.37.2 Typedef Documentation

9.37.2.1 `typedef int16_t(* GSLC_CB_DEBUG_OUT)(char ch)`

9.37.2.2 `typedef bool(* GSLC_CB_DRAW)(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`

Callback function for element drawing.

9.37.2.3 `typedef bool(* GSLC_CB_EVENT)(void *pvGui, gslc_tsEvent sEvent)`

Callback function for element drawing.

9.37.2.4 `typedef bool(* GSLC_CB_INPUT)(void *pvGui, void *pvElemRef, int16_t nStatus, void *pvData)`

Callback function for element input ready.

9.37.2.5 `typedef bool(* GSLC_CB_PIN_POLL)(void *pvGui, int16_t *pnPinInd, int16_t *pnPinVal)`

Callback function for pin polling.

9.37.2.6 `typedef bool(* GSLC_CB_TICK)(void *pvGui, void *pvElemRef)`

Callback function for element tick.

9.37.2.7 `typedef bool(* GSLC_CB_TOUCH)(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

Callback function for element touch tracking.

9.37.2.8 `typedef struct gslc_tsColor gslc_tsColor`

Color structure. Defines RGB triplet.

9.37.2.9 `typedef struct gslc_tsElem gslc_tsElem`

Element Struct.

- Represents a single graphic element in the GUISlice environment
- A page is made up of a number of elements
- Each element is created with a user-specified ID for further accesses (or `GSLC_ID_AUTO` for it to be auto-generated)
- Display order of elements in a page is based upon the creation order
- Extensions to the core element types is provided through the `pXData` reference and `pfuncX*` callback functions.

9.37.2.10 typedef struct **gslc_tsEvent** **gslc_tsEvent**

Event structure.

9.37.2.11 typedef struct **gslc_tsEventTouch** **gslc_tsEventTouch**

Structure used to pass touch data through event.

9.37.2.12 typedef struct **gslc_tsPt** **gslc_tsPt**

Define point coordinates.

9.37.2.13 typedef struct **gslc_tsRect** **gslc_tsRect**

Rectangular region. Defines X,Y corner coordinates plus dimensions.

9.37.3 Enumeration Type Documentation

9.37.3.1 enum **gslc_teAction**

GUI Action Requested These actions are usually the result of an InputMap lookup.

Enumerator

GSLC_ACTION_UNDEF Invalid action.
GSLC_ACTION_NONE No action to perform.
GSLC_ACTION_FOCUS_PREV Advance focus to the previous GUI element.
GSLC_ACTION_FOCUS_NEXT Advance focus to the next GUI element.
GSLC_ACTION_SELECT Select the currently focused GUI element.
GSLC_ACTION_SET_REL Adjust value (relative) of focused element.
GSLC_ACTION_SET_ABS Adjust value (absolute) of focused element.
GSLC_ACTION_DEBUG Internal debug action.

9.37.3.2 enum **gslc_teElemId**

Element ID enumerations.

- The Element ID is the primary means for user code to reference a graphic element.
- Application code can assign arbitrary Element ID values in the range of 0...16383
- Specifying **GSLC_ID_AUTO** to ElemCreate() requests that GUIslice auto-assign an ID value for the Element. These auto-assigned values will begin at **GSLC_ID_AUTO_BASE**.
- Negative Element ID values are reserved

Enumerator

GSLC_ID_USER_BASE Starting Element ID for user assignments.
GSLC_ID_NONE No Element ID has been assigned.
GSLC_ID_AUTO Auto-assigned Element ID requested.
GSLC_ID_TEMP ID for Temporary Element.
GSLC_ID_AUTO_BASE Starting Element ID to start auto-assignment (when **GSLC_ID_AUTO** is specified)

9.37.3.3 enum gslc_teElemInd

Element Index enumerations.

- The Element Index is used for internal purposes as an offset

Enumerator

GSLC_IND_NONE No Element Index is available.

GSLC_IND_FIRST User elements start at index 0.

9.37.3.4 enum gslc_teElemRefFlags

Element reference flags: Describes characteristics of an element.

- Primarily used to support relocation of elements to Flash memory (PROGMEM)

Enumerator

GSLC_ELEMREF_NONE No element defined.

GSLC_ELEMREF_SRC_RAM Element is read/write Stored in RAM (internal element array)) Access directly.

GSLC_ELEMREF_SRC_PROG Element is read-only / const Stored in FLASH (external to element array)
Access via PROGMEM.

GSLC_ELEMREF_SRC_CONST Element is read-only / const Stored in FLASH (external to element array)
Access directly.

GSLC_ELEMREF_REDRAW_NONE No redraw requested.

GSLC_ELEMREF_REDRAW_FULL Full redraw of element requested.

GSLC_ELEMREF_REDRAW_INC Incremental redraw of element requested.

GSLC_ELEMREF_GLOWING Element state is glowing.

GSLC_ELEMREF_VISIBLE Element is currently shown (ie. visible)

GSLC_ELEMREF_SRC Mask for Source flags.

GSLC_ELEMREF_REDRAW_MASK Mask for Redraw flags.

9.37.3.5 enum gslc_teEventSubType

Event sub-types.

Enumerator

GSLC_EVTSUB_NONE

GSLC_EVTSUB_DRAW_NEEDED Incremental redraw (as needed)

GSLC_EVTSUB_DRAW_FORCE Force a full redraw.

9.37.3.6 enum `gslc_teEventType`

Event types.

Enumerator

`GSLC_EVT_NONE` No event; ignore.
`GSLC_EVT_DRAW` Perform redraw.
`GSLC_EVT_TOUCH` Track touch event.
`GSLC_EVT_TICK` Perform background tick handling.
`GSLV_EVT_CUSTOM` Custom event.

9.37.3.7 enum `gslc_teFontId`

Font ID enumerations.

- The Font ID is the primary means for user code to reference a specific font.
- Application code can assign arbitrary Font ID values in the range of 0...16383
- Negative Font ID values are reserved

Enumerator

`GSLC_FONT_USER_BASE` Starting Font ID for user assignments.
`GSLC_FONT_NONE` No Font ID has been assigned.

9.37.3.8 enum `gslc_teFontRefMode`

Font Reference modes.

- The Font Reference mode defines the source for the selected font. For graphics libraries that offer multiple types of fonts, this can be used to differentiate between a default font, hardware fonts, software fonts, etc.
- The encoding between the different modes is driver-specific.

Enumerator

`GSLC_FONTREF_MODE_DEFAULT` Default font mode.
`GSLC_FONTREF_MODE_1` Font mode 1.
`GSLC_FONTREF_MODE_2` Font mode 2.
`GSLC_FONTREF_MODE_3` Font mode 3.

9.37.3.9 enum `gslc_tFontRefType`

Font Reference types.

- The Font Reference type defines the way in which a font is selected. In some device targets (such as LINUX SDL) a filename to a font file is provided. In others (such as Arduino, ESP8266), a pointer is given to a font structure (or NULL for default).

Enumerator

GSLC_FONTREF_FNAME Font reference is a filename (full path)

GSLC_FONTREF_PTR Font reference is a pointer to a font structure.

9.37.3.10 enum `gslc_tGroupId`

Group ID enumerations.

Enumerator

GSLC_GROUP_ID_USER_BASE Starting Group ID for user assignments.

GSLC_GROUP_ID_NONE No Group ID has been assigned.

9.37.3.11 enum `gslc_tImgRefFlags`

Image reference flags: Describes characteristics of an image reference.

Enumerator

GSLC_IMGREF_NONE No image defined.

GSLC_IMGREF_SRC_FILE Image is stored in file system.

GSLC_IMGREF_SRC_SD Image is stored on SD card.

GSLC_IMGREF_SRC_RAM Image is stored in RAM.

GSLC_IMGREF_SRC_PROG Image is stored in program memory (PROGMEM)

GSLC_IMGREF_FMT_BMP24 Image format is BMP (24-bit)

GSLC_IMGREF_FMT_BMP16 Image format is BMP (16-bit RGB565)

GSLC_IMGREF_FMT_RAW1 Image format is raw monochrome (1-bit)

GSLC_IMGREF_FMT_JPG Image format is JPG (ESP32/ESP8366)

GSLC_IMGREF_SRC Mask for Source flags.

GSLC_IMGREF_FMT Mask for Format flags.

9.37.3.12 enum `gslc_tInitStat`

Status of a module's initialization.

Enumerator

GSLC_INITSTAT_UNDEF Module status has not been defined yet.

GSLC_INITSTAT_INACTIVE Module is not enabled.

GSLC_INITSTAT_FAIL Module is enabled but failed to init.

GSLC_INITSTAT_ACTIVE Module is enabled and initialized OK.

9.37.3.13 enum `gslc_teInputRawEvent`

Raw input event types: touch, key, GPIOs.

Enumerator

GSLC_INPUT_NONE No input event.
GSLC_INPUT_TOUCH Touch / mouse event.
GSLC_INPUT_KEY_DOWN Key press down / pin input asserted.
GSLC_INPUT_KEY_UP Key press up (released)
GSLC_INPUT_PIN_ASSERT GPIO pin input asserted (eg. set to 1 / High)
GSLC_INPUT_PIN_DEASSERT GPIO pin input deasserted (eg. set to 0 / Low)

9.37.3.14 enum `gslc_tePageld`

Page ID enumerations.

- The Page ID is the primary means for user code to reference a specific page of elements.
- Application code can assign arbitrary Page ID values in the range of 0...16383
- Negative Page ID values are reserved

Enumerator

GSLC_PAGE_USER_BASE Starting Page ID for user assignments.
GSLC_PAGE_NONE No Page ID has been assigned.

9.37.3.15 enum `gslc_tePin`

General purpose pin/button constants.

Enumerator

GSLC_PIN_BTN_A Button A (short press)
GSLC_PIN_BTN_A_LONG Button A (long press)
GSLC_PIN_BTN_B Button B (short press)
GSLC_PIN_BTN_B_LONG Button B (long press)
GSLC_PIN_BTN_C Button C (short press)
GSLC_PIN_BTN_C_LONG Button C (long press)
GSLC_PIN_BTN_D Button D (short press)
GSLC_PIN_BTN_D_LONG Button D (long press)
GSLC_PIN_BTN_E Button E (short press)
GSLC_PIN_BTN_E_LONG Button E (long press)
GSLC_PIN_BTN_UP Button Up (short press)
GSLC_PIN_BTN_DOWN Button Down (short press)
GSLC_PIN_BTN_LEFT Button Left (short press)
GSLC_PIN_BTN_RIGHT Button Right (short press)
GSLC_PIN_BTN_SEL Button Select (short press)

9.37.3.16 enum `gslc_teRedrawType`

Redraw types.

Enumerator

GSLC_REDRAW_NONE No redraw requested.
GSLC_REDRAW_FULL Full redraw of element requested.
GSLC_REDRAW_INC Incremental redraw of element requested.

9.37.3.17 enum `gslc_teStackPage`

Define page stack.

Enumerator

GSLC_STACK_BASE Base page.
GSLC_STACK_CUR Current page.
GSLC_STACK_OVERLAY Overlay page (eg. popups)
GSLC_STACK_MAX Defines maximum number of pages in stack.

9.37.3.18 enum `gslc_teTouch`

Processed event from input raw events and actions.

Enumerator

GSLC_TOUCH_NONE No touch event active.
GSLC_TOUCH_TYPE_MASK Mask for type: coord/direct mode.
GSLC_TOUCH_COORD Event based on touch coordinate.
GSLC_TOUCH_DIRECT Event based on specific element index (keyboard/GPIO action)
GSLC_TOUCH_SUBTYPE_MASK Mask for subtype.
GSLC_TOUCH_DOWN Touch event (down)
GSLC_TOUCH_DOWN_IN Touch event (down inside tracked element)
GSLC_TOUCH_DOWN_OUT Touch event (down outside tracked element)
GSLC_TOUCH_UP Touch event (up)
GSLC_TOUCH_UP_IN Touch event (up inside tracked element)
GSLC_TOUCH_UP_OUT Touch event (up inside tracked element)
GSLC_TOUCH_MOVE Touch event (move)
GSLC_TOUCH_MOVE_IN Touch event (move inside tracked element)
GSLC_TOUCH_MOVE_OUT Touch event (move outside tracked element)
GSLC_TOUCH_FOCUS_ON Direct event focus on element.
GSLC_TOUCH_FOCUS_OFF Direct event focus away from focused element.
GSLC_TOUCH_FOCUS_SELECT Direct event select focus element.
GSLC_TOUCH_SET_REL Direct event set value (relative) on focus element.
GSLC_TOUCH_SET_ABS Direct event set value (absolute) on focus element.

9.37.3.19 enum gslc_teTxtFlags

Text reference flags: Describes the characteristics of a text string (ie.

whether internal to element or external and RAM vs Flash).)

Supported flag combinations are:

- `ALLOC_NONE`
- `ALLOC_INT | MEM_RAM`
- `ALLOC_EXT | MEM_RAM`
- `ALLOC_EXT | MEM_PROG`

Enumerator

GSLC_TXT_MEM_RAM Text string is in SRAM (read-write)

GSLC_TXT_MEM_PROG Text string is in PROGMEM (read-only)

GSLC_TXT_ALLOC_NONE No text string present.

GSLC_TXT_ALLOC_INT Text string allocated in internal element memory (`GSLC_STR_LOCAL=1`)

GSLC_TXT_ALLOC_EXT Text string allocated in external memory (`GSLC_STR_LOCAL=0`), ie. user code.

GSLC_TXT_ENC_PLAIN Encoding is plain text (LATIN1))

GSLC_TXT_ENC_UTF8 Encoding is UTF-8.

GSLC_TXT_MEM Mask for updating text memory type.

GSLC_TXT_ALLOC Mask for updating location of text string buffer allocation.

GSLC_TXT_ENC Mask for updating text encoding.

GSLC_TXT_DEFAULT

9.37.3.20 enum gslc_teTypeCore

Element type.

Enumerator

GSLC_TYPE_NONE No element type specified.

GSLC_TYPE_BKGND Background element type.

GSLC_TYPE_BTN Button element type.

GSLC_TYPE_TXT Text label element type.

GSLC_TYPE_BOX Box / frame element type.

GSLC_TYPE_LINE Line element type.

GSLC_TYPE_BASE_EXTEND Base value for extended type enumerations.

9.37.4 Variable Documentation

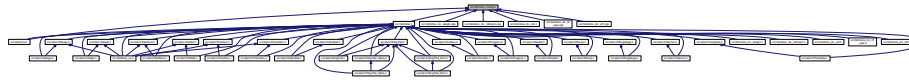
9.37.4.1 GSLC_CB_DEBUG_OUT g_pfDebugOut

Global debug output function.

- The user assigns this function via [gslc_InitDebug\(\)](#)

9.38 src/GUISlice_config.h File Reference

This graph shows which files directly or indirectly include this file:



9.39 src/GUISlice_config_ard.h File Reference

Macros

- #define [DRV_DISP_ADAGFX](#)
- #define [DRV_TOUCH_NONE](#)
- #define [DRV_DISP_ADAGFX_ILI9341](#)
- #define [ADAGFX_PIN_CS](#)
- #define [ADAGFX_PIN_DC](#)
- #define [ADAGFX_PIN_RST](#)
- #define [ADAGFX_PIN_SDCS](#)
- #define [ADAGFX_PIN_WR](#)
- #define [ADAGFX_PIN_RD](#)
- #define [ADAGFX_SPI_HW](#)
- #define [ADAGFX_PIN_MOSI](#)
- #define [ADAGFX_PIN_MISO](#)
- #define [ADAGFX_PIN_CLK](#)
- #define [GSLC_ROTATE](#)
- #define [TOUCH_ROTATION_DATA](#)
- #define [TOUCH_ROTATION_SWAPXY](#)(rotation)
- #define [TOUCH_ROTATION_FLIPX](#)(rotation)
- #define [TOUCH_ROTATION_FLIPY](#)(rotation)
- #define [ADATOUCH_SWAP_XY](#)
- #define [ADATOUCH_FLIP_X](#)
- #define [ADATOUCH_FLIP_Y](#)
- #define [GSLC_TOUCH_MAX_EVT](#)
- #define [DEBUG_ERR](#)
- #define [GSLC_FEATURE_COMPOUND](#)
- #define [GSLC_FEATURE_XGAUGE_RADIAL](#)
- #define [GSLC_FEATURE_XGAUGE_RAMP](#)
- #define [GSLC_FEATURE_XTEXTBOX_EMBED](#)
- #define [GSLC_FEATURE_INPUT](#)
- #define [GSLC_SD_EN](#)
- #define [GSLC_SD_BUFFPIXEL](#)
- #define [GSLC_CLIP_EN](#)
- #define [GSLC_BMP_TRANS_EN](#)
- #define [GSLC_BMP_TRANS_RGB](#)
- #define [GSLC_LOCAL_STR](#)
- #define [GSLC_LOCAL_STR_LEN](#)
- #define [GSLC_USE_FLOAT](#)
- #define [GSLC_DEV_TOUCH](#)
- #define [GSLC_USE_PROGMEM](#)

9.39.1 Macro Definition Documentation

9.39.1.1 `#define ADAGFX_PIN_CLK`

9.39.1.2 `#define ADAGFX_PIN_CS`

9.39.1.3 `#define ADAGFX_PIN_DC`

9.39.1.4 `#define ADAGFX_PIN_MISO`

9.39.1.5 `#define ADAGFX_PIN_MOSI`

9.39.1.6 `#define ADAGFX_PIN_RD`

9.39.1.7 `#define ADAGFX_PIN_RST`

9.39.1.8 `#define ADAGFX_PIN_SDCS`

9.39.1.9 `#define ADAGFX_PIN_WR`

9.39.1.10 `#define ADAGFX_SPI_HW`

9.39.1.11 `#define ADATOUCH_FLIP_X`

9.39.1.12 `#define ADATOUCH_FLIP_Y`

9.39.1.13 `#define ADATOUCH_SWAP_XY`

9.39.1.14 `#define DEBUG_ERR`

9.39.1.15 `#define DRV_DISP_ADAGFX`

9.39.1.16 `#define DRV_DISP_ADAGFX_ILI9341`

9.39.1.17 `#define DRV_TOUCH_NONE`

9.39.1.18 `#define GSLC_BMP_TRANS_EN`

9.39.1.19 `#define GSLC_BMP_TRANS_RGB`

9.39.1.20 `#define GSLC_CLIP_EN`

9.39.1.21 `#define GSLC_DEV_TOUCH`

9.39.1.22 `#define GSLC_FEATURE_COMPOUND`

- 9.39.1.23 `#define GSLC_FEATURE_INPUT`
- 9.39.1.24 `#define GSLC_FEATURE_XGAUGE_RADIAL`
- 9.39.1.25 `#define GSLC_FEATURE_XGAUGE_RAMP`
- 9.39.1.26 `#define GSLC_FEATURE_XTEXTBOX_EMBED`
- 9.39.1.27 `#define GSLC_LOCAL_STR`
- 9.39.1.28 `#define GSLC_LOCAL_STR_LEN`
- 9.39.1.29 `#define GSLC_ROTATE`
- 9.39.1.30 `#define GSLC_SD_BUFFPIXEL`
- 9.39.1.31 `#define GSLC_SD_EN`
- 9.39.1.32 `#define GSLC_TOUCH_MAX_EVT`
- 9.39.1.33 `#define GSLC_USE_FLOAT`
- 9.39.1.34 `#define GSLC_USE_PROGMEM`
- 9.39.1.35 `#define TOUCH_ROTATION_DATA`
- 9.39.1.36 `#define TOUCH_ROTATION_FLIPX(rotation)`
- 9.39.1.37 `#define TOUCH_ROTATION_FLIPY(rotation)`
- 9.39.1.38 `#define TOUCH_ROTATION_SWAPXY(rotation)`

9.40 src/GUISlice_config_linux.h File Reference

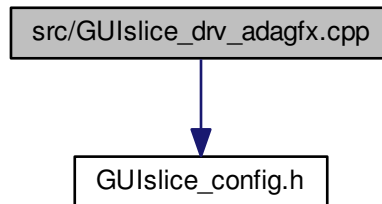
Macros

- `#define DRV_DISP_SDL1`
- `#define DRV_TOUCH_TSLIB`
- `#define GSLC_FEATURE_COMPOUND`
- `#define GSLC_FEATURE_XGAUGE_RADIAL`
- `#define GSLC_FEATURE_XGAUGE_RAMP`
- `#define GSLC_FEATURE_XTEXTBOX_EMBED`
- `#define GSLC_FEATURE_INPUT`
- `#define DEBUG_ERR`
- `#define GSLC_DEV_FB`
- `#define GSLC_DEV_TOUCH`
- `#define GSLC_DEV_VID_DRV`
- `#define DRV_SDL_FIX_START`
- `#define DRV_SDL_MOUSE_SHOW`
- `#define GSLC_LOCAL_STR`
- `#define GSLC_USE_FLOAT`
- `#define ADATOUCH_SWAP_XY`
- `#define ADATOUCH_FLIP_X`
- `#define ADATOUCH_FLIP_Y`
- `#define GSLC_TOUCH_MAX_EVT`
- `#define GSLC_LOCAL_STR_LEN`
- `#define GSLC_BMP_TRANS_EN`
- `#define GSLC_BMP_TRANS_RGB`
- `#define GSLC_USE_PROGMEM`

9.42 src/GUISlice_drv_adagfx.cpp File Reference

```
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice_drv_adagfx.cpp:



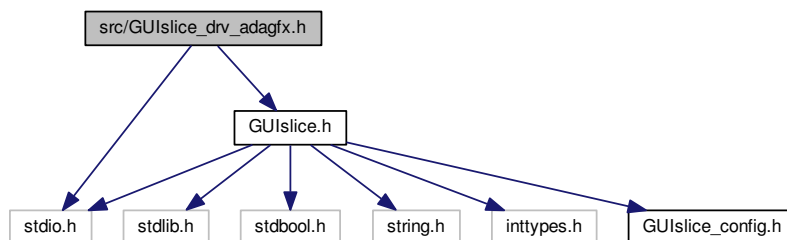
9.43 src/GUISlice_drv_adagfx.h File Reference

GUISlice library (driver layer for Adafruit-GFX)

```
#include "GUISlice.h"
```

```
#include <stdio.h>
```

Include dependency graph for GUISlice_drv_adagfx.h:



Data Structures

- struct [gslc_tsDriver](#)

Macros

- #define [DRV_HAS_DRAW_POINT](#)
Support [gslc_DrvDrawPoint\(\)](#)
- #define [DRV_HAS_DRAW_POINTS](#)
Support [gslc_DrvDrawPoints\(\)](#)

- `#define DRV_HAS_DRAW_LINE`
Support `gslc_DrvDrawLine()`
- `#define DRV_HAS_DRAW_RECT_FRAME`
Support `gslc_DrvDrawFrameRect()`
- `#define DRV_HAS_DRAW_RECT_FILL`
Support `gslc_DrvDrawFillRect()`
- `#define DRV_HAS_DRAW_RECT_ROUND_FRAME`
Support `gslc_DrvDrawFrameRoundRect()`
- `#define DRV_HAS_DRAW_RECT_ROUND_FILL`
Support `gslc_DrvDrawFillRoundRect()`
- `#define DRV_HAS_DRAW_CIRCLE_FRAME`
Support `gslc_DrvDrawFrameCircle()`
- `#define DRV_HAS_DRAW_CIRCLE_FILL`
Support `gslc_DrvDrawFillCircle()`
- `#define DRV_HAS_DRAW_TRI_FRAME`
Support `gslc_DrvDrawFrameTriangle()`
- `#define DRV_HAS_DRAW_TRI_FILL`
Support `gslc_DrvDrawFillTriangle()`
- `#define DRV_HAS_DRAW_TEXT`
Support `gslc_DrvDrawTxt()`
- `#define DRV_HAS_DRAW_BMP_MEM`
Support `gslc_DrvDrawBmp24FromMem()`
- `#define DRV_OVERRIDE_TXT_ALIGN`
Driver provides text alignment.

Functions

- `bool gslc_DrvInit (gslc_tsGui *pGui)`
Initialize the SDL library.
- `bool gslc_DrvInitTs (gslc_tsGui *pGui, const char *acDev)`
Perform any touchscreen-specific initialization.
- `void gslc_DrvDestruct (gslc_tsGui *pGui)`
Free up any members associated with the driver.
- `const char * gslc_DrvGetNameDisp (gslc_tsGui *pGui)`
Get the display driver name.
- `const char * gslc_DrvGetNameTouch (gslc_tsGui *pGui)`
Get the touch driver name.
- `void * gslc_DrvGetDriverDisp (gslc_tsGui *pGui)`
Get the native display driver instance.
- `void * gslc_DrvGetDriverTouch (gslc_tsGui *pGui)`
Get the native touch driver instance.
- `void * gslc_DrvLoadImage (gslc_tsGui *pGui, gslc_tsImgRef sImgRef)`
Load a bitmap (*.bmp) and create a new image resource.
- `bool gslc_DrvSetBkgndImage (gslc_tsGui *pGui, gslc_tsImgRef sImgRef)`
Configure the background to use a bitmap image.
- `bool gslc_DrvSetBkgndColor (gslc_tsGui *pGui, gslc_tsColor nCol)`
Configure the background to use a solid color.
- `bool gslc_DrvSetElemImageNorm (gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsImgRef sImgRef)`
Set an element's normal-state image.
- `bool gslc_DrvSetElemImageGlow (gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsImgRef sImgRef)`

- Set an element's glow-state image.*

 - void [gslc_DrvImageDestruct](#) (void *pVImg)
- Release an image surface.*

 - bool [gslc_DrvSetClipRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) *pRect)
- Set the clipping rectangle for future drawing updates.*

 - const void * [gslc_DrvFontAdd](#) ([gslc_teFontRefType](#) eFontRefType, const void *pvFontRef, uint16_t nFontSz)
- Load a font from a resource and return pointer to it.*

 - void [gslc_DrvFontsDestruct](#) ([gslc_tsGui](#) *pGui)
- Release all fonts defined in the GUI.*

 - bool [gslc_DrvGetTxtSize](#) ([gslc_tsGui](#) *pGui, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)
- Get the extent (width and height) of a text string.*

 - bool [gslc_DrvDrawTxt](#) ([gslc_tsGui](#) *pGui, int16_t nTxtX, int16_t nTxtY, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, [gslc_tsColor](#) colTxt, [gslc_tsColor](#) colBg)
- Draw a text string at the given coordinate.*

 - void [gslc_DrvPageFlipNow](#) ([gslc_tsGui](#) *pGui)
- Force a page flip to occur.*

 - bool [gslc_DrvDrawPoint](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, [gslc_tsColor](#) nCol)
- Draw a point.*

 - bool [gslc_DrvDrawPoints](#) ([gslc_tsGui](#) *pGui, [gslc_tsPt](#) *asPt, uint16_t nNumPt, [gslc_tsColor](#) nCol)
- Draw a point.*

 - bool [gslc_DrvDrawFrameRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
- Draw a framed rectangle.*

 - bool [gslc_DrvDrawFillRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
- Draw a filled rectangle.*

 - bool [gslc_DrvDrawFrameRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)
- Draw a framed rounded rectangle.*

 - bool [gslc_DrvDrawFillRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)
- Draw a filled rounded rectangle.*

 - bool [gslc_DrvDrawLine](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, [gslc_tsColor](#) nCol)
- Draw a line.*

 - bool [gslc_DrvDrawFrameCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)
- Draw a framed circle.*

 - bool [gslc_DrvDrawFillCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)
- Draw a filled circle.*

 - bool [gslc_DrvDrawFrameTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)
- Draw a framed triangle.*

 - bool [gslc_DrvDrawFillTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)
- Draw a filled triangle.*

 - bool [gslc_DrvDrawImage](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, [gslc_tsImgRef](#) slmgRef)
- Copy all of source image to destination screen at specified coordinate.*

 - void [gslc_DrvDrawMonoFromMem](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)
- Draw a monochrome bitmap from a memory array.*

 - void [gslc_DrvDrawBmp24FromMem](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)

- Draw a color 24-bit depth bitmap from a memory array.*
- void [gslc_DrvDrawBkgnd](#) ([gslc_tsGui](#) *pGui)
- Copy the background image to destination screen.*
- bool [gslc_DrvInitTouch](#) ([gslc_tsGui](#) *pGui, const char *acDev)
- Perform any touchscreen-specific initialization.*
- bool [gslc_DrvGetTouch](#) ([gslc_tsGui](#) *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, [gslc_teInputRawEvent](#) *peInputEvent, int16_t *pnInputVal)
- Get the last touch event from the internal touch handler.*
- bool [gslc_DrvRotate](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation)
- Change rotation, automatically adapt touchscreen axes swap/flip.*
- uint16_t [gslc_DrvAdaptColorToRaw](#) ([gslc_tsColor](#) nCol)

9.43.1 Detailed Description

GUIslice library (driver layer for Adafruit-GFX)

9.43.2 Macro Definition Documentation

9.43.2.1 #define DRV_HAS_DRAW_BMP_MEM

Support [gslc_DrvDrawBmp24FromMem\(\)](#)

9.43.2.2 #define DRV_HAS_DRAW_CIRCLE_FILL

Support [gslc_DrvDrawFillCircle\(\)](#)

9.43.2.3 #define DRV_HAS_DRAW_CIRCLE_FRAME

Support [gslc_DrvDrawFrameCircle\(\)](#)

9.43.2.4 #define DRV_HAS_DRAW_LINE

Support [gslc_DrvDrawLine\(\)](#)

9.43.2.5 #define DRV_HAS_DRAW_POINT

Support [gslc_DrvDrawPoint\(\)](#)

9.43.2.6 #define DRV_HAS_DRAW_POINTS

Support [gslc_DrvDrawPoints\(\)](#)

9.43.2.7 #define DRV_HAS_DRAW_RECT_FILL

Support [gslc_DrvDrawFillRect\(\)](#)

9.43.2.8 #define DRV_HAS_DRAW_RECT_FRAME

Support [gslc_DrvDrawFrameRect\(\)](#)

9.43.2.9 #define DRV_HAS_DRAW_RECT_ROUND_FILL

Support [gslc_DrvDrawFillRoundRect\(\)](#)

9.43.2.10 #define DRV_HAS_DRAW_RECT_ROUND_FRAME

Support [gslc_DrvDrawFrameRoundRect\(\)](#)

9.43.2.11 #define DRV_HAS_DRAW_TEXT

Support [gslc_DrvDrawTxt\(\)](#)

9.43.2.12 #define DRV_HAS_DRAW_TRI_FILL

Support [gslc_DrvDrawFillTriangle\(\)](#)

9.43.2.13 #define DRV_HAS_DRAW_TRI_FRAME

Support [gslc_DrvDrawFrameTriangle\(\)](#)

9.43.2.14 #define DRV_OVERRIDE_TXT_ALIGN

Driver provides text alignment.

9.43.3 Function Documentation

9.43.3.1 uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor *nCol*)

9.43.3.2 void gslc_DrvDestruct (gslc_tsGui * *pGui*)

Free up any members associated with the driver.

- Eg. renderers, windows, background surfaces, etc.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.43.3.3 void gslc_DrvDrawBkgnd (gslc_tsGui * pGui)

Copy the background image to destination screen.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.43.3.4 void gslc_DrvDrawBmp24FromMem (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, const unsigned char * pBitmap, bool bProgMem)

Draw a color 24-bit depth bitmap from a memory array.

- Note that users must convert images from their native format (eg. BMP, PNG, etc.) into a C array. Please refer to the following guide for details: <https://github.com/ImpulseAdventure/GUIslice/wiki/Display-Images-from-FLASH>
- The converted file (c array) can then be included in the sketch.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	X coord for copy
in	<i>nDstY</i>	Y coord for copy
in	<i>pBitmap</i>	Pointer to bitmap buffer
in	<i>bProgMem</i>	Bitmap is stored in Flash if true, RAM otherwise

Returns

none

9.43.3.5 bool gslc_DrvDrawFillCircle (gslc_tsGui * pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)

Draw a filled circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nMidX</i>	Center of circle (X coordinate)
in	<i>nMidY</i>	Center of circle (Y coordinate)
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.43.3.6 `bool gslc_DrvDrawFillRect (gslc_tsGui * pGui, gslc_tsRect rRect, gslc_tsColor nCol)`

Draw a filled rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.43.3.7 `bool gslc_DrvDrawFillRoundRect (gslc_tsGui * pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a filled rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nRadius</i>	Radius for rounded corners
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.43.3.8 `bool gslc_DrvDrawFillTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a filled triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.43.3.9 `bool gslc_DrvDrawFrameCircle (gslc_tsGui * pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)`

Draw a framed circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nMidX</i>	Center of circle (X coordinate)
in	<i>nMidY</i>	Center of circle (Y coordinate)
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.43.3.10 `bool gslc_DrvDrawFrameRect (gslc_tsGui * pGui, gslc_tsRect rRect, gslc_tsColor nCol)`

Draw a framed rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.43.3.11 `bool gslc_DrvDrawFrameRoundRect (gslc_tsGui * pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a framed rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nRadius</i>	Radius for rounded corners
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.43.3.12 `bool gslc_DrvDrawFrameTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a framed triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.43.3.13 `bool gslc_DrvDrawImage (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, gslc_tsImgRef sImgRef)`

Copy all of source image to destination screen at specified coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.43.3.14 `bool gslc_DrvDrawLine (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)`

Draw a line.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	Line start (X coordinate)
in	<i>nY0</i>	Line start (Y coordinate)
in	<i>nX1</i>	Line finish (X coordinate)
in	<i>nY1</i>	Line finish (Y coordinate)
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.43.3.15 `void gslc_DrvDrawMonoFromMem (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, const unsigned char * pBitmap, bool bProgMem)`

Draw a monochrome bitmap from a memory array.

- Draw from the bitmap buffer using the foreground color defined in the header (unset bits are transparent)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>pBitmap</i>	Pointer to bitmap buffer
in	<i>bProgMem</i>	Bitmap is stored in Flash if true, RAM otherwise

Returns

none

9.43.3.16 `bool gslc_DrvDrawPoint (gslc_tsGui * pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)`

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	X coordinate of point
in	<i>nY</i>	Y coordinate of point
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.43.3.17 `bool gslc_DrvDrawPoints (gslc_tsGui * pGui, gslc_tsPt * asPt, uint16_t nNumPt, gslc_tsColor nCol)`

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>asPt</i>	Array of points to draw
in	<i>n</i> ↔ <i>NumPt</i>	Number of points in array
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.43.3.18 `bool gslc_DrvDrawTxt (gslc_tsGui * pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)`

Draw a text string at the given coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nTxtX</i>	X coordinate of top-left text string
in	<i>nTxtY</i>	Y coordinate of top-left text string
in	<i>pFont</i>	Ptr to Font
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
in	<i>colTxt</i>	Color to draw text
in	<i>colBg</i>	unused in ADAGFX, defaults to black

Returns

true if success, false if failure

9.43.3.19 `const void* gslc_DrvFontAdd (gslc_teFontRefType eFontRefType, const void * pvFontRef, uint16_t nFontSz)`

Load a font from a resource and return pointer to it.

Parameters

in	<i>eFontRefType</i>	Font reference type (GSLC_FONTREF_PTR for Arduino)
in	<i>pvFontRef</i>	Font reference pointer (Pointer to the GFXFont array)
in	<i>nFontSz</i>	Typeface size to use

Returns

Void ptr to driver-specific font if load was successful, NULL otherwise

9.43.3.20 `void gslc_DrvFontsDestruct (gslc_tsGui * pGui)`

Release all fonts defined in the GUI.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.43.3.21 `void* gslc_DrvGetDriverDisp (gslc_tsGui * pGui)`

Get the native display driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the display driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.43.3.22 `void* gslc_DrvGetDriverTouch (gslc_tsGui * pGui)`

Get the native touch driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the touch driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.43.3.23 `const char* gslc_DrvGetNameDisp (gslc_tsGui * pGui)`

Get the display driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.43.3.24 `const char* gslc_DrvGetNameTouch (gslc_tsGui * pGui)`

Get the touch driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.43.3.25 `bool gslc_DrvGetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress, gslc_telInputRawEvent * pelInputEvent, int16_t * pnInputVal)`

Get the last touch event from the internal touch handler.

Parameters

in	<i>pGui</i>	Pointer to GUI
out	<i>pnX</i>	Ptr to X coordinate of last touch event
out	<i>pnY</i>	Ptr to Y coordinate of last touch event
out	<i>pnPress</i>	Ptr to Pressure level of last touch event (0 for none, 1 for touch)
out	<i>pelInputEvent</i>	Indication of event type
out	<i>pnInputVal</i>	Additional data for event type

Returns

true if an event was detected or false otherwise

9.43.3.26 `bool gslc_DrvGetTxtSize (gslc_tsGui * pGui, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, int16_t * pnTxtX, int16_t * pnTxtY, uint16_t * pnTxtSzW, uint16_t * pnTxtSzH)`

Get the extent (width and height) of a text string.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pFont</i>	Ptr to Font structure
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
out	<i>pnTxtX</i>	Ptr to offset X of text
out	<i>pnTxtY</i>	Ptr to offset Y of text
out	<i>pnTxtSzW</i>	Ptr to width of text
out	<i>pnTxtSzH</i>	Ptr to height of text

Returns

true if success, false if failure

9.43.3.27 `void gslc_DrvImageDestruct (void * pvImg)`

Release an image surface.

Parameters

in	<i>pvImg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.43.3.28 `bool gslc_DrvInit (gslc_tsGui * pGui)`

Initialize the SDL library.

- Performs clean startup workaround (if enabled)
- Configures video mode
- Initializes font support

PRE:

- The environment variables should be configured before calling `gslc_DrvInit()`. This can be done with `gslc_DrvInitEnv()` or manually in user function.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.43.3.29 `bool gslc_DrvInitTouch (gslc_tsGui * pGui, const char * acDev)`

Perform any touchscreen-specific initialization.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

9.43.3.30 `bool gslc_DrvInitTs (gslc_tsGui * pGui, const char * acDev)`

Perform any touchscreen-specific initialization.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

9.43.3.31 `void* gslc_DrvLoadImage (gslc_tsGui * pGui, gslc_tsImgRef sImgRef)`

Load a bitmap (*.bmp) and create a new image resource.

Transparency is enabled by GSLC_BMP_TRANS_EN through use of color (GSLC_BMP_TRANS_RGB).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

Image pointer (surface/texture) or NULL if error

9.43.3.32 void gslc_DrvPageFlipNow (gslc_tsGui * *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.43.3.33 bool gslc_DrvRotate (gslc_tsGui * *pGui*, uint8_t *nRotation*)

Change rotation, automatically adapt touchscreen axes swap/flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)

Returns

true if successful

9.43.3.34 bool gslc_DrvSetBkgndColor (gslc_tsGui * *pGui*, gslc_tsColor *nCol*)

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nCol</i>	RGB Color to use

Returns

true if success, false if fail

9.43.3.35 `bool gslc_DrvSetBkgndImage (gslc_tsGui * pGui, gslc_tsImgRef sImgRef)`

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.43.3.36 `bool gslc_DrvSetClipRect (gslc_tsGui * pGui, gslc_tsRect * pRect)`

Set the clipping rectangle for future drawing updates.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pRect</i>	Rectangular region to constrain edits

Returns

true if success, false if error

9.43.3.37 `bool gslc_DrvSetElemImageGlow (gslc_tsGui * pGui, gslc_tsElem * pElem, gslc_tsImgRef sImgRef)`

Set an element's glow-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if error

9.43.3.38 `bool gslc_DrvSetElemImageNorm (gslc_tsGui * pGui, gslc_tsElem * pElem, gslc_tsImgRef sImgRef)`

Set an element's normal-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

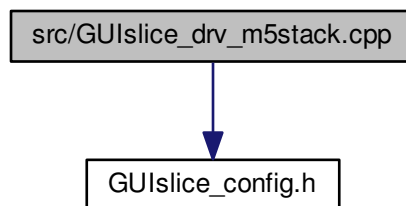
Returns

true if success, false if error

9.44 src/GUISlice_drv_m5stack.cpp File Reference

```
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice_drv_m5stack.cpp:

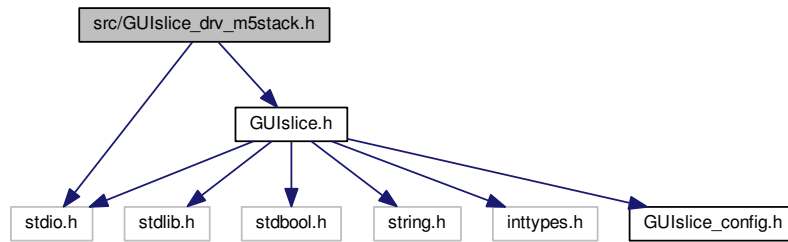


9.45 src/GUISlice_drv_m5stack.h File Reference

GUISlice library (driver layer for M5stack)

```
#include "GUISlice.h"
#include <stdio.h>
```

Include dependency graph for `GUIslice_drv_m5stack.h`:



Data Structures

- struct [gslc_tsDriver](#)

Macros

- `#define DRV_HAS_DRAW_POINT`
Support [gslc_DrvDrawPoint\(\)](#)
- `#define DRV_HAS_DRAW_POINTS`
Support [gslc_DrvDrawPoints\(\)](#)
- `#define DRV_HAS_DRAW_LINE`
Support [gslc_DrvDrawLine\(\)](#)
- `#define DRV_HAS_DRAW_RECT_FRAME`
Support [gslc_DrvDrawFrameRect\(\)](#)
- `#define DRV_HAS_DRAW_RECT_FILL`
Support [gslc_DrvDrawFillRect\(\)](#)
- `#define DRV_HAS_DRAW_RECT_ROUND_FRAME`
Support [gslc_DrvDrawFrameRoundRect\(\)](#)
- `#define DRV_HAS_DRAW_RECT_ROUND_FILL`
Support [gslc_DrvDrawFillRoundRect\(\)](#)
- `#define DRV_HAS_DRAW_CIRCLE_FRAME`
Support [gslc_DrvDrawFrameCircle\(\)](#)
- `#define DRV_HAS_DRAW_CIRCLE_FILL`
Support [gslc_DrvDrawFillCircle\(\)](#)
- `#define DRV_HAS_DRAW_TRI_FRAME`
Support [gslc_DrvDrawFrameTriangle\(\)](#)
- `#define DRV_HAS_DRAW_TRI_FILL`
Support [gslc_DrvDrawFillTriangle\(\)](#)
- `#define DRV_HAS_DRAW_TEXT`
Support [gslc_DrvDrawTxt\(\)](#)
- `#define DRV_HAS_DRAW_BMP_MEM`
Support [gslc_DrvDrawBmp24FromMem\(\)](#)
- `#define DRV_OVERRIDE_TXT_ALIGN`
Driver provides text alignment.

Functions

- bool [gslc_DrvInit](#) ([gslc_tsGui](#) *pGui)
Initialize the SDL library.
- bool [gslc_DrvInitTs](#) ([gslc_tsGui](#) *pGui, const char *acDev)
Perform any touchscreen-specific initialization.
- void [gslc_DrvDestruct](#) ([gslc_tsGui](#) *pGui)
Free up any members associated with the driver.
- const char * [gslc_DrvGetNameDisp](#) ([gslc_tsGui](#) *pGui)
Get the display driver name.
- const char * [gslc_DrvGetNameTouch](#) ([gslc_tsGui](#) *pGui)
Get the touch driver name.
- void * [gslc_DrvGetDriverDisp](#) ([gslc_tsGui](#) *pGui)
Get the native display driver instance.
- void * [gslc_DrvGetDriverTouch](#) ([gslc_tsGui](#) *pGui)
Get the native touch driver instance.
- void * [gslc_DrvLoadImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Load a bitmap (.bmp) and create a new image resource.*
- bool [gslc_DrvSetBkgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Configure the background to use a bitmap image.
- bool [gslc_DrvSetBkgndColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)
Configure the background to use a solid color.
- bool [gslc_DrvSetElemImageNorm](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tsImgRef](#) sImgRef)
Set an element's normal-state image.
- bool [gslc_DrvSetElemImageGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tsImgRef](#) sImgRef)
Set an element's glow-state image.
- void [gslc_DrvImageDestruct](#) (void *pvImg)
Release an image surface.
- bool [gslc_DrvSetClipRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) *pRect)
Set the clipping rectangle for future drawing updates.
- const void * [gslc_DrvFontAdd](#) ([gslc_teFontRefType](#) eFontRefType, const void *pvFontRef, uint16_t nFontSz)
Load a font from a resource and return pointer to it.
- void [gslc_DrvFontsDestruct](#) ([gslc_tsGui](#) *pGui)
Release all fonts defined in the GUI.
- bool [gslc_DrvGetTxtSize](#) ([gslc_tsGui](#) *pGui, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxt↵Flags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)
Get the extent (width and height) of a text string.
- bool [gslc_DrvDrawTxt](#) ([gslc_tsGui](#) *pGui, int16_t nTxtX, int16_t nTxtY, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, [gslc_tsColor](#) colTxt, [gslc_tsColor](#) colBg)
Draw a text string at the given coordinate.
- bool [gslc_DrvDrawTxtAlign](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int8_t e↵TxtAlign, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, [gslc_tsColor](#) colTxt, [gslc_tsColor](#) colBg)
Draw a text string in a bounding box using the specified alignment.
- void [gslc_DrvPageFlipNow](#) ([gslc_tsGui](#) *pGui)
Force a page flip to occur.
- bool [gslc_DrvDrawPoint](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, [gslc_tsColor](#) nCol)
Draw a point.
- bool [gslc_DrvDrawPoints](#) ([gslc_tsGui](#) *pGui, [gslc_tsPt](#) *asPt, uint16_t nNumPt, [gslc_tsColor](#) nCol)
Draw a point.
- bool [gslc_DrvDrawFrameRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)

Draw a framed rectangle.

- bool [gslc_DrvDrawFillRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)

Draw a filled rectangle.

- bool [gslc_DrvDrawFrameRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)

Draw a framed rounded rectangle.

- bool [gslc_DrvDrawFillRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)

Draw a filled rounded rectangle.

- bool [gslc_DrvDrawLine](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, [gslc_tsColor](#) nCol)

Draw a line.

- bool [gslc_DrvDrawFrameCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)

Draw a framed circle.

- bool [gslc_DrvDrawFillCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)

Draw a filled circle.

- bool [gslc_DrvDrawFrameTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)

Draw a framed triangle.

- bool [gslc_DrvDrawFillTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)

Draw a filled triangle.

- bool [gslc_DrvDrawImage](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, [gslc_tsImgRef](#) slmgRef)

Copy all of source image to destination screen at specified coordinate.

- void [gslc_DrvDrawMonoFromMem](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)

Draw a monochrome bitmap from a memory array.

- void [gslc_DrvDrawBmp24FromMem](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)

Draw a color 24-bit depth bitmap from a memory array.

- void [gslc_DrvDrawBkgnd](#) ([gslc_tsGui](#) *pGui)

Copy the background image to destination screen.

- bool [gslc_DrvRotate](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation)

Change rotation, automatically adapt touchscreen axes swap/flip.

- uint16_t [gslc_DrvAdaptColorToRaw](#) ([gslc_tsColor](#) nCol)

Variables

- const char [GSLC_PMEM_ERRSTR_NULL](#) []
- const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.45.1 Detailed Description

GUIslice library (driver layer for M5stack)

9.45.2 Macro Definition Documentation

9.45.2.1 #define DRV_HAS_DRAW_BMP_MEM

Support [gslc_DrvDrawBmp24FromMem\(\)](#)

9.45.2.2 #define DRV_HAS_DRAW_CIRCLE_FILL

Support [gslc_DrvDrawFillCircle\(\)](#)

9.45.2.3 #define DRV_HAS_DRAW_CIRCLE_FRAME

Support [gslc_DrvDrawFrameCircle\(\)](#)

9.45.2.4 #define DRV_HAS_DRAW_LINE

Support [gslc_DrvDrawLine\(\)](#)

9.45.2.5 #define DRV_HAS_DRAW_POINT

Support [gslc_DrvDrawPoint\(\)](#)

9.45.2.6 #define DRV_HAS_DRAW_POINTS

Support [gslc_DrvDrawPoints\(\)](#)

9.45.2.7 #define DRV_HAS_DRAW_RECT_FILL

Support [gslc_DrvDrawFillRect\(\)](#)

9.45.2.8 #define DRV_HAS_DRAW_RECT_FRAME

Support [gslc_DrvDrawFrameRect\(\)](#)

9.45.2.9 #define DRV_HAS_DRAW_RECT_ROUND_FILL

Support [gslc_DrvDrawFillRoundRect\(\)](#)

9.45.2.10 #define DRV_HAS_DRAW_RECT_ROUND_FRAME

Support [gslc_DrvDrawFrameRoundRect\(\)](#)

9.45.2.11 #define DRV_HAS_DRAW_TEXT

Support [gslc_DrvDrawTxt\(\)](#)

9.45.2.12 `#define DRV_HAS_DRAW_TRI_FILL`

Support [gslc_DrvDrawFillTriangle\(\)](#)

9.45.2.13 `#define DRV_HAS_DRAW_TRI_FRAME`

Support [gslc_DrvDrawFrameTriangle\(\)](#)

9.45.2.14 `#define DRV_OVERRIDE_TXT_ALIGN`

Driver provides text alignment.

9.45.3 Function Documentation

9.45.3.1 `uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)`

9.45.3.2 `void gslc_DrvDestruct (gslc_tsGui * pGui)`

Free up any members associated with the driver.

- Eg. renderers, windows, background surfaces, etc.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.45.3.3 `void gslc_DrvDrawBkgnd (gslc_tsGui * pGui)`

Copy the background image to destination screen.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.45.3.4 void gslc_DrvDrawBmp24FromMem (gslc_tsGui * *pGui*, int16_t *nDstX*, int16_t *nDstY*, const unsigned char * *pBitmap*, bool *bProgMem*)

Draw a color 24-bit depth bitmap from a memory array.

- Note that users must convert images from their native format (eg. BMP, PNG, etc.) into a C array. Please refer to the following guide for details: <https://github.com/ImpulseAdventure/GUIslice/wiki/Display-Images-from-FLASH>
- The converted file (c array) can then be included in the sketch.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	X coord for copy
in	<i>nDstY</i>	Y coord for copy
in	<i>pBitmap</i>	Pointer to bitmap buffer
in	<i>bProgMem</i>	Bitmap is stored in Flash if true, RAM otherwise

Returns

none

9.45.3.5 bool gslc_DrvDrawFillCircle (gslc_tsGui * *pGui*, int16_t *nMidX*, int16_t *nMidY*, uint16_t *nRadius*, gslc_tsColor *nCol*)

Draw a filled circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nMidX</i>	Center of circle (X coordinate)
in	<i>nMidY</i>	Center of circle (Y coordinate)
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.45.3.6 bool gslc_DrvDrawFillRect (gslc_tsGui * *pGui*, gslc_tsRect *rRect*, gslc_tsColor *nCol*)

Draw a filled rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.45.3.7 `bool gslc_DrvDrawFillRoundRect (gslc_tsGui * pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a filled rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nRadius</i>	Radius for rounded corners
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.45.3.8 `bool gslc_DrvDrawFillTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a filled triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.45.3.9 `bool gslc_DrvDrawFrameCircle (gslc_tsGui * pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)`

Draw a framed circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>nMidX</i>	Center of circle (X coordinate)
in	<i>nMidY</i>	Center of circle (Y coordinate)
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.45.3.10 `bool gslc_DrvDrawFrameRect (gslc_tsGui * pGui, gslc_tsRect rRect, gslc_tsColor nCol)`

Draw a framed rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.45.3.11 `bool gslc_DrvDrawFrameRoundRect (gslc_tsGui * pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a framed rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nRadius</i>	Radius for rounded corners
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.45.3.12 `bool gslc_DrvDrawFrameTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a framed triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.45.3.13 `bool gslc_DrvDrawImage (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, gslc_tsImgRef slmgRef)`

Copy all of source image to destination screen at specified coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>slmgRef</i>	Image reference

Returns

true if success, false if fail

9.45.3.14 `bool gslc_DrvDrawLine (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)`

Draw a line.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	Line start (X coordinate)
in	<i>nY0</i>	Line start (Y coordinate)
in	<i>nX1</i>	Line finish (X coordinate)
in	<i>nY1</i>	Line finish (Y coordinate)
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.45.3.15 void gslc_DrvDrawMonoFromMem (gslc_tsGui * *pGui*, int16_t *nDstX*, int16_t *nDstY*, const unsigned char * *pBitmap*, bool *bProgMem*)

Draw a monochrome bitmap from a memory array.

- Draw from the bitmap buffer using the foreground color defined in the header (unset bits are transparent)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>pBitmap</i>	Pointer to bitmap buffer
in	<i>bProgMem</i>	Bitmap is stored in Flash if true, RAM otherwise

Returns

none

9.45.3.16 bool gslc_DrvDrawPoint (gslc_tsGui * *pGui*, int16_t *nX*, int16_t *nY*, gslc_tsColor *nCol*)

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	X coordinate of point
in	<i>nY</i>	Y coordinate of point
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.45.3.17 bool gslc_DrvDrawPoints (gslc_tsGui * *pGui*, gslc_tsPt * *asPt*, uint16_t *nNumPt*, gslc_tsColor *nCol*)

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>asPt</i>	Array of points to draw
in	<i>n</i> ↔ <i>NumPt</i>	Number of points in array
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.45.3.18 `bool gslc_DrvDrawTxt (gslc_tsGui * pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)`

Draw a text string at the given coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nTxtX</i>	X coordinate of top-left text string
in	<i>nTxtY</i>	Y coordinate of top-left text string
in	<i>pFont</i>	Ptr to Font
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
in	<i>colTxt</i>	Color to draw text
in	<i>colBg</i>	unused in m5stack, defaults to black

Returns

true if success, false if failure

9.45.3.19 `bool gslc_DrvDrawTxtAlign (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int8_t eTxtAlign, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)`

Draw a text string in a bounding box using the specified alignment.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X coordinate of top-left of bounding box
in	<i>nY0</i>	Y coordinate of top-left of bounding box
in	<i>nX1</i>	X coordinate of bot-right of bounding box
in	<i>nY1</i>	Y coordinate of bot-right of bounding box
in	<i>eTxtAlign</i>	Alignment mode]
in	<i>pFont</i>	Ptr to Font
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
in	<i>colTxt</i>	Color to draw text
in	<i>colBg</i>	unused in m5stack, defaults to black

Returns

true if success, false if failure

9.45.3.20 `const void* gslc_DrvFontAdd (gslc_teFontRefType eFontRefType, const void * pvFontRef, uint16_t nFontSz)`

Load a font from a resource and return pointer to it.

Parameters

in	<i>eFontRefType</i>	Font reference type (GSLC_FONTREF_PTR for Arduino)
in	<i>pvFontRef</i>	Font reference pointer (Pointer to the GFXFont array)
in	<i>nFontSz</i>	Typeface size to use

Returns

Void ptr to driver-specific font if load was successful, NULL otherwise

9.45.3.21 `void gslc_DrvFontsDestruct (gslc_tsGui * pGui)`

Release all fonts defined in the GUI.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.45.3.22 `void* gslc_DrvGetDriverDisp (gslc_tsGui * pGui)`

Get the native display driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the display driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.45.3.23 void* gslc_DrvGetDriverTouch (gslc_tsGui * pGui)

Get the native touch driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the touch driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.45.3.24 const char* gslc_DrvGetNameDisp (gslc_tsGui * pGui)

Get the display driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.45.3.25 const char* gslc_DrvGetNameTouch (gslc_tsGui * pGui)

Get the touch driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.45.3.26 bool gslc_DrvGetTxtSize (gslc_tsGui * pGui, gslc_tsFont * pFont, const char * pStr, gslc_teTextFlags eTxtFlags, int16_t * pnTxtX, int16_t * pnTxtY, uint16_t * pnTxtSzW, uint16_t * pnTxtSzH)

Get the extent (width and height) of a text string.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pFont</i>	Ptr to Font structure
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
out	<i>pnTxtX</i>	Ptr to offset X of text
out	<i>pnTxtY</i>	Ptr to offset Y of text
out	<i>pnTxtSzW</i>	Ptr to width of text
out	<i>pnTxtSzH</i>	Ptr to height of text

Returns

true if success, false if failure

9.45.3.27 void gslc_DrvImageDestruct (void * *pvlmg*)

Release an image surface.

Parameters

in	<i>pvlmg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.45.3.28 bool gslc_DrvInit (gslc_tsGui * *pGui*)

Initialize the SDL library.

- Performs clean startup workaround (if enabled)
- Configures video mode
- Initializes font support

PRE:

- The environment variables should be configured before calling [gslc_DrvInit\(\)](#). This can be done with [gslc_DrvInitEnv\(\)](#) or manually in user function.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.45.3.29 bool gslc_DrvInitTs (gslc_tsGui * *pGui*, const char * *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

9.45.3.30 void* gslc_DrvLoadImage (gslc_tsGui * *pGui*, gslc_tslmgRef *slmgRef*)

Load a bitmap (*.bmp) and create a new image resource.

Transparency is enabled by GSLC_BMP_TRANS_EN through use of color (GSLC_BMP_TRANS_RGB).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>slmgRef</i>	Image reference

Returns

Image pointer (surface/texture) or NULL if error

9.45.3.31 void gslc_DrvPageFlipNow (gslc_tsGui * *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.45.3.32 `bool gslc_DrvRotate (gslc_tsGui * pGui, uint8_t nRotation)`

Change rotation, automatically adapt touchscreen axes swap/flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)

Returns

true if successful

9.45.3.33 `bool gslc_DrvSetBkgndColor (gslc_tsGui * pGui, gslc_tsColor nCol)`

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nCol</i>	RGB Color to use

Returns

true if success, false if fail

9.45.3.34 `bool gslc_DrvSetBkgndImage (gslc_tsGui * pGui, gslc_tsImgRef sImgRef)`

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.45.3.35 `bool gslc_DrvSetClipRect (gslc_tsGui * pGui, gslc_tsRect * pRect)`

Set the clipping rectangle for future drawing updates.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pRect</i>	Rectangular region to constrain edits

Returns

true if success, false if error

9.45.3.36 `bool gslc_DrvSetElemImageGlow (gslc_tsGui * pGui, gslc_tsElem * pElem, gslc_tsImgRef sImgRef)`

Set an element's glow-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if error

9.45.3.37 `bool gslc_DrvSetElemImageNorm (gslc_tsGui * pGui, gslc_tsElem * pElem, gslc_tsImgRef sImgRef)`

Set an element's normal-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if error

9.45.4 Variable Documentation

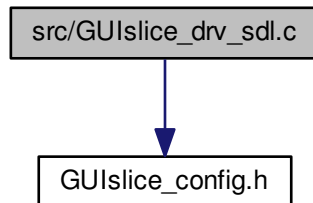
9.45.4.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.45.4.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

9.46 src/GUISlice_drv_sdl.c File Reference

```
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice_drv_sdl.c:



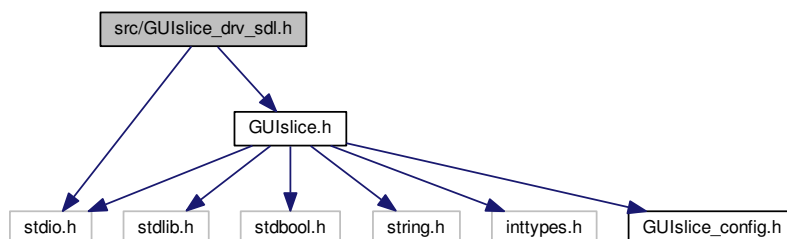
9.47 src/GUISlice_drv_sdl.h File Reference

GUISlice library (driver layer for LINUX / SDL)

```
#include "GUISlice.h"
```

```
#include <stdio.h>
```

Include dependency graph for GUISlice_drv_sdl.h:



Data Structures

- struct [gslc_tsDriver](#)

Macros

- `#define DRV_HAS_DRAW_POINT`
Support [gslc_DrvDrawPoint\(\)](#)
- `#define DRV_OVERRIDE_TXT_ALIGN`
Driver provides text alignment.

Functions

- bool [gslc_DrvInit](#) ([gslc_tsGui](#) *pGui)
Initialize the SDL library.
- void [gslc_DrvDestruct](#) ([gslc_tsGui](#) *pGui)
Free up any members associated with the driver.
- const char * [gslc_DrvGetNameDisp](#) ([gslc_tsGui](#) *pGui)
Get the display driver name.
- const char * [gslc_DrvGetNameTouch](#) ([gslc_tsGui](#) *pGui)
Get the touch driver name.
- void * [gslc_DrvGetDriverDisp](#) ([gslc_tsGui](#) *pGui)
Get the native display driver instance.
- void * [gslc_DrvGetDriverTouch](#) ([gslc_tsGui](#) *pGui)
Get the native touch driver instance.
- void * [gslc_DrvLoadImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Load a bitmap (.bmp) and create a new image resource.*
- bool [gslc_DrvSetBkgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Configure the background to use a bitmap image.
- bool [gslc_DrvSetBkgndColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)
Configure the background to use a solid color.
- bool [gslc_DrvSetElemImageNorm](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tsImgRef](#) sImgRef)
Set an element's normal-state image.
- bool [gslc_DrvSetElemImageGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tsImgRef](#) sImgRef)
Set an element's glow-state image.
- void [gslc_DrvImageDestruct](#) (void *pvImg)
Release an image surface.
- bool [gslc_DrvSetClipRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) *pRect)
Set the clipping rectangle for future drawing updates.
- const void * [gslc_DrvFontAdd](#) ([gslc_teFontRefType](#) eFontRefType, const void *pvFontRef, uint16_t nFontSz)
Load a font from a resource and return pointer to it.
- void [gslc_DrvFontsDestruct](#) ([gslc_tsGui](#) *pGui)
Release all fonts defined in the GUI.
- bool [gslc_DrvGetTxtSize](#) ([gslc_tsGui](#) *pGui, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxt↵
Flags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)
Get the extent (width and height) of a text string.
- bool [gslc_DrvDrawTxt](#) ([gslc_tsGui](#) *pGui, int16_t nTxtX, int16_t nTxtY, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, [gslc_tsColor](#) colTxt, [gslc_tsColor](#) colBg)
Draw a text string at the given coordinate.
- void [gslc_DrvPageFlipNow](#) ([gslc_tsGui](#) *pGui)
Force a page flip to occur.
- bool [gslc_DrvDrawPoint](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, [gslc_tsColor](#) nCol)
Draw a point.
- bool [gslc_DrvDrawPoints](#) ([gslc_tsGui](#) *pGui, [gslc_tsPt](#) *asPt, uint16_t nNumPt, [gslc_tsColor](#) nCol)
Draw a point.
- bool [gslc_DrvDrawFrameRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
Draw a framed rectangle.
- bool [gslc_DrvDrawFillRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
Draw a filled rectangle.
- bool [gslc_DrvDrawLine](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, [gslc_tsColor](#) nCol)
Draw a line.

- bool [gslc_DrvDrawImage](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, [gslc_tsImgRef](#) sImgRef)
Copy all of source image to destination screen at specified coordinate.
- void [gslc_DrvDrawBkgnd](#) ([gslc_tsGui](#) *pGui)
Copy the background image to destination screen.
- bool [gslc_DrvGetTouch](#) ([gslc_tsGui](#) *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, [gslc_telInputRawEvent](#) *peInputEvent, int16_t *pnInputVal)
Get the last touch event from the SDL_Event handler.
- bool [gslc_DrvRotate](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation)
Change rotation, automatically adapt touchscreen axes swap/flip.
- bool [gslc_DrvCleanStart](#) (const char *sTTY)
Ensure SDL initializes cleanly to workaround possible issues if previous SDL application failed to close down gracefully.
- void [gslc_DrvReportInfoPre](#) ()
Report driver debug info (before initialization)
- void [gslc_DrvReportInfoPost](#) ()
Report driver debug info (after initialization)
- SDL_Rect [gslc_DrvAdaptRect](#) ([gslc_tsRect](#) rRect)
Translate a [gslc_tsRect](#) into an SDL_Rect.
- SDL_Color [gslc_DrvAdaptColor](#) ([gslc_tsColor](#) sCol)
Translate a [gslc_tsColor](#) into an SDL_Color.
- bool [gslc_DrvInitTouch](#) ([gslc_tsGui](#) *pGui, const char *acDev)
Perform any touchscreen-specific initialization.

9.47.1 Detailed Description

GUISlice library (driver layer for LINUX / SDL)

9.47.2 Macro Definition Documentation

9.47.2.1 #define DRV_HAS_DRAW_POINT

Support [gslc_DrvDrawPoint\(\)](#)

9.47.2.2 #define DRV_OVERRIDE_TXT_ALIGN

Driver provides text alignment.

9.47.3 Function Documentation

9.47.3.1 SDL_Color [gslc_DrvAdaptColor](#) ([gslc_tsColor](#) sCol)

Translate a [gslc_tsColor](#) into an SDL_Color.

Parameters

in	sCol	gslc_tsColor
----	------	------------------------------

Returns

Converted SDL_Color

9.47.3.2 SDL_Rect gslc_DrvAdaptRect (gslc_tsRect *rRect*)

Translate a [gslc_tsRect](#) into an SDL_Rect.

Parameters

in	<i>rRect</i>	gslc_tsRect
----	--------------	-----------------------------

Returns

Converted SDL_Rect

9.47.3.3 bool gslc_DrvCleanStart (const char * *sTTY*)

Ensure SDL initializes cleanly to workaround possible issues if previous SDL application failed to close down gracefully.

Parameters

in	<i>sTTY</i>	Terminal device (eg. "/dev/tty0")
----	-------------	-----------------------------------

Returns

true if success

9.47.3.4 void gslc_DrvDestruct (gslc_tsGui * *pGui*)

Free up any members associated with the driver.

- Eg. renderers, windows, background surfaces, etc.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.47.3.5 void gslc_DrvDrawBkgnd (gslc_tsGui * *pGui*)

Copy the background image to destination screen.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.47.3.6 bool gslc_DrvDrawFillRect (gslc_tsGui * *pGui*, gslc_tsRect *rRect*, gslc_tsColor *nCol*)

Draw a filled rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.47.3.7 bool gslc_DrvDrawFrameRect (gslc_tsGui * *pGui*, gslc_tsRect *rRect*, gslc_tsColor *nCol*)

Draw a framed rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.47.3.8 bool gslc_DrvDrawImage (gslc_tsGui * *pGui*, int16_t *nDstX*, int16_t *nDstY*, gslc_tsImgRef *sImgRef*)

Copy all of source image to destination screen at specified coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.47.3.9 `bool gslc_DrvDrawLine (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)`

Draw a line.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	Line start (X coordinate)
in	<i>nY0</i>	Line start (Y coordinate)
in	<i>nX1</i>	Line finish (X coordinate)
in	<i>nY1</i>	Line finish (Y coordinate)
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.47.3.10 `bool gslc_DrvDrawPoint (gslc_tsGui * pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)`

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	X coordinate of point
in	<i>nY</i>	Y coordinate of point
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.47.3.11 `bool gslc_DrvDrawPoints (gslc_tsGui * pGui, gslc_tsPt * asPt, uint16_t nNumPt, gslc_tsColor nCol)`

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>asPt</i>	Array of points to draw
in	<i>n</i> ↔ <i>NumPt</i>	Number of points in array
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.47.3.12 `bool gslc_DrvDrawTxt (gslc_tsGui * pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)`

Draw a text string at the given coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nTxtX</i>	X coordinate of top-left text string
in	<i>nTxtY</i>	Y coordinate of top-left text string
in	<i>pFont</i>	Ptr to Font
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
in	<i>colTxt</i>	Color to draw text
in	<i>colBg</i>	unused in SDL, defaults to black

Returns

true if success, false if failure

9.47.3.13 `const void* gslc_DrvFontAdd (gslc_teFontRefType eFontRefType, const void * pvFontRef, uint16_t nFontSz)`

Load a font from a resource and return pointer to it.

Parameters

in	<i>eFontRefType</i>	Font reference type (GSLC_FONTREF_FNAME for SDL)
in	<i>pvFontRef</i>	Font reference pointer (Pointer to the font filename)
in	<i>nFontSz</i>	Typeface size to use

Returns

Void ptr to driver-specific font if load was successful, NULL otherwise

9.47.3.14 void gslc_DrvFontsDestruct (gslc_tsGui * pGui)

Release all fonts defined in the GUI.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.47.3.15 void* gslc_DrvGetDriverDisp (gslc_tsGui * pGui)

Get the native display driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the display driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.47.3.16 void* gslc_DrvGetDriverTouch (gslc_tsGui * pGui)

Get the native touch driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the touch driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.47.3.17 `const char* gslc_DrvGetNameDisp (gslc_tsGui * pGui)`

Get the display driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.47.3.18 `const char* gslc_DrvGetNameTouch (gslc_tsGui * pGui)`

Get the touch driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.47.3.19 `bool gslc_DrvGetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress, gslc_teInputRawEvent * peInputEvent, int16_t * pnInputVal)`

Get the last touch event from the SDL_Event handler.

Parameters

in	<i>pGui</i>	Pointer to GUI
out	<i>pnX</i>	Ptr to X coordinate of last touch event
out	<i>pnY</i>	Ptr to Y coordinate of last touch event
out	<i>pnPress</i>	Ptr to Pressure level of last touch event (0 for none, 1 for touch)
out	<i>peInputEvent</i>	Indication of event type
out	<i>pnInputVal</i>	Additional data for event type

Returns

true if an event was detected or false otherwise

9.47.3.20 `bool gslc_DrvGetTxtSize (gslc_tsGui * pGui, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, int16_t * pnTxtX, int16_t * pnTxtY, uint16_t * pnTxtSzW, uint16_t * pnTxtSzH)`

Get the extent (width and height) of a text string.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>pFont</i>	Ptr to Font structure
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
out	<i>pnTxtX</i>	Ptr to offset X of text
out	<i>pnTxtY</i>	Ptr to offset Y of text
out	<i>pnTxtSzW</i>	Ptr to width of text
out	<i>pnTxtSzH</i>	Ptr to height of text

Returns

true if success, false if failure

9.47.3.21 void gslc_DrvImageDestruct (void * *pvlmg*)

Release an image surface.

Parameters

in	<i>pvlmg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.47.3.22 bool gslc_DrvInit (gslc_tsGui * *pGui*)

Initialize the SDL library.

- Performs clean startup workaround (if enabled)
- Configures video mode
- Initializes font support

PRE:

- The environment variables should be configured before calling [gslc_DrvInit\(\)](#).

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.47.3.23 bool gslc_DrvInitTouch (gslc_tsGui * *pGui*, const char * *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

9.47.3.24 void* gslc_DrvLoadImage (gslc_tsGui * *pGui*, gslc_tslmgRef *slmgRef*)

Load a bitmap (*.bmp) and create a new image resource.

Transparency is enabled by GSLC_BMP_TRANS_EN through use of color (GSLC_BMP_TRANS_RGB).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>slmgRef</i>	Image reference

Returns

Image pointer (surface/texture/path) or NULL if error

9.47.3.25 void gslc_DrvPageFlipNow (gslc_tsGui * *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.47.3.26 void gslc_DrvReportInfoPost ()

Report driver debug info (after initialization)

Returns

none

9.47.3.27 void gslc_DrvReportInfoPre ()

Report driver debug info (before initialization)

Returns

none

9.47.3.28 bool gslc_DrvRotate (gslc_tsGui * *pGui*, uint8_t *nRotation*)

Change rotation, automatically adapt touchscreen axes swap/flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)

Returns

true if successful

9.47.3.29 bool gslc_DrvSetBkgndColor (gslc_tsGui * *pGui*, gslc_tsColor *nCol*)

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nCol</i>	RGB Color to use

Returns

true if success, false if fail

9.47.3.30 bool gslc_DrvSetBkgndImage (gslc_tsGui * *pGui*, gslc_tsImgRef *sImgRef*)

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.47.3.31 bool gslc_DrvSetClipRect (gslc_tsGui * *pGui*, gslc_tsRect * *pRect*)

Set the clipping rectangle for future drawing updates.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pRect</i>	Rectangular region to constrain edits

Returns

true if success, false if error

9.47.3.32 bool gslc_DrvSetElemImageGlow (gslc_tsGui * *pGui*, gslc_tsElem * *pElem*, gslc_tsImgRef *sImgRef*)

Set an element's glow-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if error

9.47.3.33 bool gslc_DrvSetElemImageNorm (gslc_tsGui * *pGui*, gslc_tsElem * *pElem*, gslc_tsImgRef *sImgRef*)

Set an element's normal-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

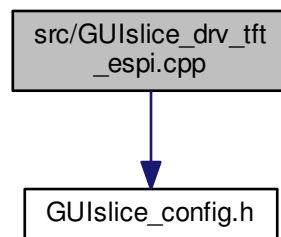
Returns

true if success, false if error

9.48 src/GUISlice_drv_tft_espi.cpp File Reference

```
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice_drv_tft_espi.cpp:

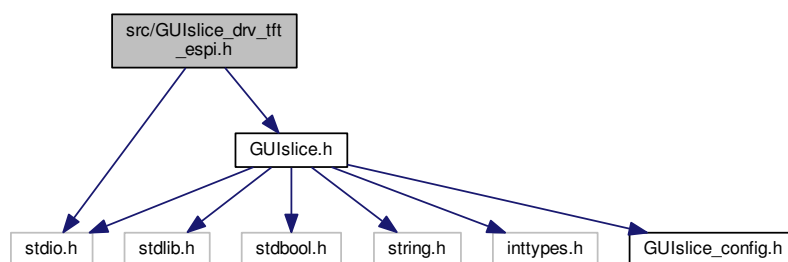
**9.49 src/GUISlice_drv_tft_espi.h File Reference**

GUISlice library (driver layer for TFT-eSPI)

```
#include "GUISlice.h"
```

```
#include <stdio.h>
```

Include dependency graph for GUISlice_drv_tft_espi.h:



Data Structures

- struct [gslc_tsDriver](#)

Macros

- #define [GSLC_SPIFFS_EN](#)
- #define [DRV_HAS_DRAW_POINT](#)
Support [gslc_DrvDrawPoint\(\)](#)
- #define [DRV_HAS_DRAW_POINTS](#)
Support [gslc_DrvDrawPoints\(\)](#)
- #define [DRV_HAS_DRAW_LINE](#)
Support [gslc_DrvDrawLine\(\)](#)
- #define [DRV_HAS_DRAW_RECT_FRAME](#)
Support [gslc_DrvDrawFrameRect\(\)](#)
- #define [DRV_HAS_DRAW_RECT_FILL](#)
Support [gslc_DrvDrawFillRect\(\)](#)
- #define [DRV_HAS_DRAW_RECT_ROUND_FRAME](#)
Support [gslc_DrvDrawFrameRoundRect\(\)](#)
- #define [DRV_HAS_DRAW_RECT_ROUND_FILL](#)
Support [gslc_DrvDrawFillRoundRect\(\)](#)
- #define [DRV_HAS_DRAW_CIRCLE_FRAME](#)
Support [gslc_DrvDrawFrameCircle\(\)](#)
- #define [DRV_HAS_DRAW_CIRCLE_FILL](#)
Support [gslc_DrvDrawFillCircle\(\)](#)
- #define [DRV_HAS_DRAW_TRI_FRAME](#)
Support [gslc_DrvDrawFrameTriangle\(\)](#)
- #define [DRV_HAS_DRAW_TRI_FILL](#)
Support [gslc_DrvDrawFillTriangle\(\)](#)
- #define [DRV_HAS_DRAW_TEXT](#)
Support [gslc_DrvDrawTxt\(\)](#)
- #define [DRV_HAS_DRAW_BMP_MEM](#)
Support [gslc_DrvDrawBmp24FromMem\(\)](#)
- #define [DRV_OVERRIDE_TXT_ALIGN](#)
Driver provides text alignment.

Functions

- bool [gslc_DrvInit](#) ([gslc_tsGui](#) *pGui)
Initialize the SDL library.
- bool [gslc_DrvInitTs](#) ([gslc_tsGui](#) *pGui, const char *acDev)
Perform any touchscreen-specific initialization.
- void [gslc_DrvDestruct](#) ([gslc_tsGui](#) *pGui)
Free up any members associated with the driver.
- const char * [gslc_DrvGetNameDisp](#) ([gslc_tsGui](#) *pGui)
Get the display driver name.
- const char * [gslc_DrvGetNameTouch](#) ([gslc_tsGui](#) *pGui)
Get the touch driver name.
- void * [gslc_DrvGetDriverDisp](#) ([gslc_tsGui](#) *pGui)
Get the native display driver instance.

- void * [gslc_DrvGetDriverTouch](#) ([gslc_tsGui](#) *pGui)
Get the native touch driver instance.
- void * [gslc_DrvLoadImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Load a bitmap (.bmp) and create a new image resource.*
- bool [gslc_DrvSetBkgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Configure the background to use a bitmap image.
- bool [gslc_DrvSetBkgndColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)
Configure the background to use a solid color.
- bool [gslc_DrvSetElemImageNorm](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tsImgRef](#) sImgRef)
Set an element's normal-state image.
- bool [gslc_DrvSetElemImageGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tsImgRef](#) sImgRef)
Set an element's glow-state image.
- void [gslc_DrvImageDestruct](#) (void *pvImg)
Release an image surface.
- bool [gslc_DrvSetClipRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) *pRect)
Set the clipping rectangle for future drawing updates.
- const void * [gslc_DrvFontAdd](#) ([gslc_teFontRefType](#) eFontRefType, const void *pvFontRef, uint16_t nFontSz)
Load a font from a resource and return pointer to it.
- void [gslc_DrvFontsDestruct](#) ([gslc_tsGui](#) *pGui)
Release all fonts defined in the GUI.
- bool [gslc_DrvGetTxtSize](#) ([gslc_tsGui](#) *pGui, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxt↵
Flags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)
Get the extent (width and height) of a text string.
- bool [gslc_DrvDrawTxt](#) ([gslc_tsGui](#) *pGui, int16_t nTxtX, int16_t nTxtY, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, [gslc_tsColor](#) colTxt, [gslc_tsColor](#) colBg)
Draw a text string at the given coordinate.
- bool [gslc_DrvDrawTxtAlign](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int8_t e↵
TxtAlign, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, [gslc_tsColor](#) colTxt, [gslc_tsColor](#) colBg)
Draw a text string in a bounding box using the specified alignment.
- void [gslc_DrvPageFlipNow](#) ([gslc_tsGui](#) *pGui)
Force a page flip to occur.
- bool [gslc_DrvDrawPoint](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, [gslc_tsColor](#) nCol)
Draw a point.
- bool [gslc_DrvDrawPoints](#) ([gslc_tsGui](#) *pGui, [gslc_tsPt](#) *asPt, uint16_t nNumPt, [gslc_tsColor](#) nCol)
Draw a point.
- bool [gslc_DrvDrawFrameRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
Draw a framed rectangle.
- bool [gslc_DrvDrawFillRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
Draw a filled rectangle.
- bool [gslc_DrvDrawFrameRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) n↵
Col)
Draw a framed rounded rectangle.
- bool [gslc_DrvDrawFillRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)
Draw a filled rounded rectangle.
- bool [gslc_DrvDrawLine](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, [gslc_tsColor](#) nCol)
Draw a line.
- bool [gslc_DrvDrawFrameCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_ts↵
Color](#) nCol)
Draw a framed circle.

- bool [gslc_DrvDrawFillCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)
Draw a filled circle.
- bool [gslc_DrvDrawFrameTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)
Draw a framed triangle.
- bool [gslc_DrvDrawFillTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)
Draw a filled triangle.
- bool [gslc_DrvDrawImage](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, [gslc_tsImgRef](#) sImgRef)
Copy all of source image to destination screen at specified coordinate.
- void [gslc_DrvDrawMonoFromMem](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)
Draw a monochrome bitmap from a memory array.
- void [gslc_DrvDrawBmp24FromMem](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)
Draw a color 24-bit depth bitmap from a memory array.
- void [gslc_DrvDrawBkgnd](#) ([gslc_tsGui](#) *pGui)
Copy the background image to destination screen.
- bool [gslc_DrvRotate](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation)
Change rotation, automatically adapt touchscreen axes swap/flip.
- uint16_t [gslc_DrvAdaptColorToRaw](#) ([gslc_tsColor](#) nCol)

9.49.1 Detailed Description

GUISlice library (driver layer for TFT-eSPI)

9.49.2 Macro Definition Documentation

9.49.2.1 #define DRV_HAS_DRAW_BMP_MEM

Support [gslc_DrvDrawBmp24FromMem\(\)](#)

9.49.2.2 #define DRV_HAS_DRAW_CIRCLE_FILL

Support [gslc_DrvDrawFillCircle\(\)](#)

9.49.2.3 #define DRV_HAS_DRAW_CIRCLE_FRAME

Support [gslc_DrvDrawFrameCircle\(\)](#)

9.49.2.4 #define DRV_HAS_DRAW_LINE

Support [gslc_DrvDrawLine\(\)](#)

9.49.2.5 `#define DRV_HAS_DRAW_POINT`

Support [gslc_DrvDrawPoint\(\)](#)

9.49.2.6 `#define DRV_HAS_DRAW_POINTS`

Support [gslc_DrvDrawPoints\(\)](#)

9.49.2.7 `#define DRV_HAS_DRAW_RECT_FILL`

Support [gslc_DrvDrawFillRect\(\)](#)

9.49.2.8 `#define DRV_HAS_DRAW_RECT_FRAME`

Support [gslc_DrvDrawFrameRect\(\)](#)

9.49.2.9 `#define DRV_HAS_DRAW_RECT_ROUND_FILL`

Support [gslc_DrvDrawFillRoundRect\(\)](#)

9.49.2.10 `#define DRV_HAS_DRAW_RECT_ROUND_FRAME`

Support [gslc_DrvDrawFrameRoundRect\(\)](#)

9.49.2.11 `#define DRV_HAS_DRAW_TEXT`

Support [gslc_DrvDrawTxt\(\)](#)

9.49.2.12 `#define DRV_HAS_DRAW_TRI_FILL`

Support [gslc_DrvDrawFillTriangle\(\)](#)

9.49.2.13 `#define DRV_HAS_DRAW_TRI_FRAME`

Support [gslc_DrvDrawFrameTriangle\(\)](#)

9.49.2.14 `#define DRV_OVERRIDE_TXT_ALIGN`

Driver provides text alignment.

9.49.2.15 `#define GSLC_SPIFFS_EN`

9.49.3 Function Documentation

9.49.3.1 `uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)`

9.49.3.2 `void gslc_DrvDestruct (gslc_tsGui * pGui)`

Free up any members associated with the driver.

- Eg. renderers, windows, background surfaces, etc.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.49.3.3 void gslc_DrvDrawBkgnd (gslc_tsGui * *pGui*)

Copy the background image to destination screen.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.49.3.4 void gslc_DrvDrawBmp24FromMem (gslc_tsGui * *pGui*, int16_t *nDstX*, int16_t *nDstY*, const unsigned char * *pBitmap*, bool *bProgMem*)

Draw a color 24-bit depth bitmap from a memory array.

- Note that users must convert images from their native format (eg. BMP, PNG, etc.) into a C array. Please refer to the following guide for details: <https://github.com/ImpulseAdventure/GUISlice/wiki/Display-Images-from-FLASH>
- The converted file (c array) can then be included in the sketch.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	X coord for copy
in	<i>nDstY</i>	Y coord for copy
in	<i>pBitmap</i>	Pointer to bitmap buffer
in	<i>bProgMem</i>	Bitmap is stored in Flash if true, RAM otherwise

Returns

none

9.49.3.5 bool gslc_DrvDrawFillCircle (gslc_tsGui * *pGui*, int16_t *nMidX*, int16_t *nMidY*, uint16_t *nRadius*, gslc_tsColor *nCol*)

Draw a filled circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nMidX</i>	Center of circle (X coordinate)
in	<i>nMidY</i>	Center of circle (Y coordinate)
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.49.3.6 `bool gslc_DrvDrawFillRect (gslc_tsGui * pGui, gslc_tsRect rRect, gslc_tsColor nCol)`

Draw a filled rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.49.3.7 `bool gslc_DrvDrawFillRoundRect (gslc_tsGui * pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a filled rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nRadius</i>	Radius for rounded corners
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.49.3.8 `bool gslc_DrvDrawFillTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a filled triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.49.3.9 `bool gslc_DrvDrawFrameCircle (gslc_tsGui * pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)`

Draw a framed circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nMidX</i>	Center of circle (X coordinate)
in	<i>nMidY</i>	Center of circle (Y coordinate)
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.49.3.10 `bool gslc_DrvDrawFrameRect (gslc_tsGui * pGui, gslc_tsRect rRect, gslc_tsColor nCol)`

Draw a framed rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.49.3.11 `bool gslc_DrvDrawFrameRoundRect (gslc_tsGui * pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a framed rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nRadius</i>	Radius for rounded corners
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.49.3.12 `bool gslc_DrvDrawFrameTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a framed triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.49.3.13 `bool gslc_DrvDrawImage (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, gslc_tsImgRef sImgRef)`

Copy all of source image to destination screen at specified coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.49.3.14 `bool gslc_DrvDrawLine (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)`

Draw a line.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	Line start (X coordinate)
in	<i>nY0</i>	Line start (Y coordinate)
in	<i>nX1</i>	Line finish (X coordinate)
in	<i>nY1</i>	Line finish (Y coordinate)
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.49.3.15 `void gslc_DrvDrawMonoFromMem (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, const unsigned char * pBitmap, bool bProgMem)`

Draw a monochrome bitmap from a memory array.

- Draw from the bitmap buffer using the foreground color defined in the header (unset bits are transparent)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>pBitmap</i>	Pointer to bitmap buffer
in	<i>bProgMem</i>	Bitmap is stored in Flash if true, RAM otherwise

Returns

none

9.49.3.16 `bool gslc_DrvDrawPoint (gslc_tsGui * pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)`

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	X coordinate of point
in	<i>nY</i>	Y coordinate of point
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.49.3.17 `bool gslc_DrvDrawPoints (gslc_tsGui * pGui, gslc_tsPt * asPt, uint16_t nNumPt, gslc_tsColor nCol)`

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>asPt</i>	Array of points to draw
in	<i>n</i> ↔ <i>NumPt</i>	Number of points in array
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.49.3.18 `bool gslc_DrvDrawTxt (gslc_tsGui * pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)`

Draw a text string at the given coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nTxtX</i>	X coordinate of top-left text string
in	<i>nTxtY</i>	Y coordinate of top-left text string
in	<i>pFont</i>	Ptr to Font
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
in	<i>colTxt</i>	Color to draw text
in	<i>colBg</i>	Color of Background for antialias blending

Returns

true if success, false if failure

9.49.3.19 `bool gslc_DrvDrawTxtAlign (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int8_t eTxtAlign, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)`

Draw a text string in a bounding box using the specified alignment.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X coordinate of top-left of bounding box
in	<i>nY0</i>	Y coordinate of top-left of bounding box
in	<i>nX1</i>	X coordinate of bot-right of bounding box
in	<i>nY1</i>	Y coordinate of bot-right of bounding box
in	<i>eTxtAlign</i>	Alignment mode]
in	<i>pFont</i>	Ptr to Font
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
in	<i>colTxt</i>	Color to draw text
in	<i>colBg</i>	Color of Background for antialias blending

Returns

true if success, false if failure

9.49.3.20 `const void* gslc_DrvFontAdd (gslc_teFontRefType eFontRefType, const void * pvFontRef, uint16_t nFontSz)`

Load a font from a resource and return pointer to it.

Parameters

in	<i>eFontRefType</i>	Font reference type: <ul style="list-style-type: none"> • GSLC_FONTREF_PTR for Standard TFT_eSPI Fonts • GSLC_FONTREF_FNAME for antialiased Font in SPIFFS
in	<i>pvFontRef</i>	Font reference pointer / SPIFFS font filename without ext.
in	<i>nFontSz</i>	Typeface size to use, ignored for SPIFFS font

Returns

Void ptr to driver-specific font if load was successful, NULL otherwise

9.49.3.21 `void gslc_DrvFontsDestruct (gslc_tsGui * pGui)`

Release all fonts defined in the GUI.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.49.3.22 void* gslc_DrvGetDriverDisp (gslc_tsGui * pGui)

Get the native display driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the display driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.49.3.23 void* gslc_DrvGetDriverTouch (gslc_tsGui * pGui)

Get the native touch driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the touch driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.49.3.24 const char* gslc_DrvGetNameDisp (gslc_tsGui * pGui)

Get the display driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.49.3.25 `const char* gslc_DrvGetNameTouch (gslc_tsGui * pGui)`

Get the touch driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.49.3.26 `bool gslc_DrvGetTxtSize (gslc_tsGui * pGui, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, int16_t * pnTxtX, int16_t * pnTxtY, uint16_t * pnTxtSzW, uint16_t * pnTxtSzH)`

Get the extent (width and height) of a text string.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pFont</i>	Ptr to Font structure
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
out	<i>pnTxtX</i>	Ptr to offset X of text
out	<i>pnTxtY</i>	Ptr to offset Y of text
out	<i>pnTxtSzW</i>	Ptr to width of text
out	<i>pnTxtSzH</i>	Ptr to height of text

Returns

true if success, false if failure

9.49.3.27 `void gslc_DrvImageDestruct (void * pVImg)`

Release an image surface.

Parameters

in	<i>pVImg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.49.3.28 bool gslc_DrvInit (gslc_tsGui * *pGui*)

Initialize the SDL library.

- Performs clean startup workaround (if enabled)
- Configures video mode
- Initializes font support

PRE:

- The environment variables should be configured before calling [gslc_DrvInit\(\)](#). This can be done with [gslc_DrvInitEnv\(\)](#) or manually in user function.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.49.3.29 bool gslc_DrvInitTs (gslc_tsGui * *pGui*, const char * *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

9.49.3.30 void* gslc_DrvLoadImage (gslc_tsGui * *pGui*, gslc_tsImgRef *sImgRef*)

Load a bitmap (*.bmp) and create a new image resource.

Transparency is enabled by GSLC_BMP_TRANS_EN through use of color (GSLC_BMP_TRANS_RGB).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

Image pointer (surface/texture) or NULL if error

9.49.3.31 void gslc_DrvPageFlipNow (gslc_tsGui * *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.49.3.32 bool gslc_DrvRotate (gslc_tsGui * *pGui*, uint8_t *nRotation*)

Change rotation, automatically adapt touchscreen axes swap/flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)

Returns

true if successful

9.49.3.33 bool gslc_DrvSetBkgndColor (gslc_tsGui * *pGui*, gslc_tsColor *nCol*)

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nCol</i>	RGB Color to use

Returns

true if success, false if fail

9.49.3.34 `bool gslc_DrvSetBkgndImage (gslc_tsGui * pGui, gslc_tsImgRef sImgRef)`

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.49.3.35 `bool gslc_DrvSetClipRect (gslc_tsGui * pGui, gslc_tsRect * pRect)`

Set the clipping rectangle for future drawing updates.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pRect</i>	Rectangular region to constrain edits

Returns

true if success, false if error

9.49.3.36 `bool gslc_DrvSetElemImageGlow (gslc_tsGui * pGui, gslc_tsElem * pElem, gslc_tsImgRef sImgRef)`

Set an element's glow-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if error

9.49.3.37 `bool gslc_DrvSetElemImageNorm (gslc_tsGui * pGui, gslc_tsElem * pElem, gslc_tsImgRef sImgRef)`

Set an element's normal-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

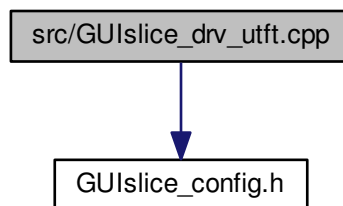
Returns

true if success, false if error

9.50 src/GUISlice_drv_utft.cpp File Reference

```
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice_drv_utft.cpp:

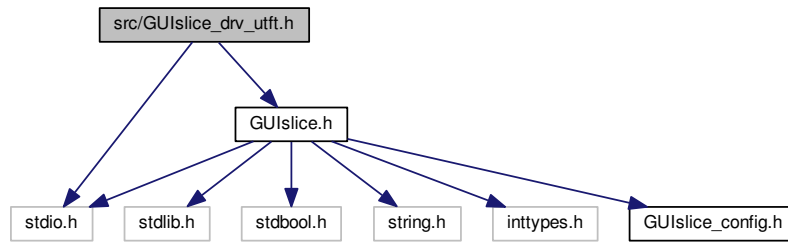


9.51 src/GUISlice_drv_utft.h File Reference

GUISlice library (driver layer for UTFT)

```
#include "GUISlice.h"
#include <stdio.h>
```

Include dependency graph for `GUIslice_drv_utf.h`:



Data Structures

- struct [gslc_tsDriver](#)

Macros

- `#define DRV_HAS_DRAW_POINT`
Support [gslc_DrvDrawPoint\(\)](#)
- `#define DRV_HAS_DRAW_POINTS`
Support [gslc_DrvDrawPoints\(\)](#)
- `#define DRV_HAS_DRAW_LINE`
Support [gslc_DrvDrawLine\(\)](#)
- `#define DRV_HAS_DRAW_RECT_FRAME`
Support [gslc_DrvDrawFrameRect\(\)](#)
- `#define DRV_HAS_DRAW_RECT_FILL`
Support [gslc_DrvDrawFillRect\(\)](#)
- `#define DRV_HAS_DRAW_RECT_ROUND_FRAME`
Support [gslc_DrvDrawFrameRoundRect\(\)](#)
- `#define DRV_HAS_DRAW_RECT_ROUND_FILL`
Support [gslc_DrvDrawFillRoundRect\(\)](#)
- `#define DRV_HAS_DRAW_CIRCLE_FRAME`
Support [gslc_DrvDrawFrameCircle\(\)](#)
- `#define DRV_HAS_DRAW_CIRCLE_FILL`
Support [gslc_DrvDrawFillCircle\(\)](#)
- `#define DRV_HAS_DRAW_TRI_FRAME`
Support [gslc_DrvDrawFrameTriangle\(\)](#)
- `#define DRV_HAS_DRAW_TRI_FILL`
Support [gslc_DrvDrawFillTriangle\(\)](#)
- `#define DRV_HAS_DRAW_TEXT`
Support [gslc_DrvDrawTxt\(\)](#)
- `#define DRV_HAS_DRAW_BMP_MEM`
Support [gslc_DrvDrawBmp24FromMem\(\)](#)
- `#define DRV_OVERRIDE_TXT_ALIGN`
Driver provides text alignment.

Functions

- bool [gslc_DrvInit](#) ([gslc_tsGui](#) *pGui)
Initialize the SDL library.
- bool [gslc_DrvInitTs](#) ([gslc_tsGui](#) *pGui, const char *acDev)
Perform any touchscreen-specific initialization.
- void [gslc_DrvDestruct](#) ([gslc_tsGui](#) *pGui)
Free up any members associated with the driver.
- const char * [gslc_DrvGetNameDisp](#) ([gslc_tsGui](#) *pGui)
Get the display driver name.
- const char * [gslc_DrvGetNameTouch](#) ([gslc_tsGui](#) *pGui)
Get the touch driver name.
- void * [gslc_DrvGetDriverDisp](#) ([gslc_tsGui](#) *pGui)
Get the native display driver instance.
- void * [gslc_DrvGetDriverTouch](#) ([gslc_tsGui](#) *pGui)
Get the native touch driver instance.
- void * [gslc_DrvLoadImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Load a bitmap (.bmp) and create a new image resource.*
- bool [gslc_DrvSetBkgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Configure the background to use a bitmap image.
- bool [gslc_DrvSetBkgndColor](#) ([gslc_tsGui](#) *pGui, [gslc_tsColor](#) nCol)
Configure the background to use a solid color.
- bool [gslc_DrvSetElemImageNorm](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tsImgRef](#) sImgRef)
Set an element's normal-state image.
- bool [gslc_DrvSetElemImageGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tsImgRef](#) sImgRef)
Set an element's glow-state image.
- void [gslc_DrvImageDestruct](#) (void *pvImg)
Release an image surface.
- bool [gslc_DrvSetClipRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) *pRect)
Set the clipping rectangle for future drawing updates.
- const void * [gslc_DrvFontAdd](#) ([gslc_teFontRefType](#) eFontRefType, const void *pvFontRef, uint16_t nFontSz)
Load a font from a resource and return pointer to it.
- void [gslc_DrvFontsDestruct](#) ([gslc_tsGui](#) *pGui)
Release all fonts defined in the GUI.
- bool [gslc_DrvGetTxtSize](#) ([gslc_tsGui](#) *pGui, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, int16_t *pnTxtX, int16_t *pnTxtY, uint16_t *pnTxtSzW, uint16_t *pnTxtSzH)
Get the extent (width and height) of a text string.
- bool [gslc_DrvDrawTxt](#) ([gslc_tsGui](#) *pGui, int16_t nTxtX, int16_t nTxtY, [gslc_tsFont](#) *pFont, const char *pStr, [gslc_teTxtFlags](#) eTxtFlags, [gslc_tsColor](#) colTxt, [gslc_tsColor](#) colBg)
Draw a text string at the given coordinate.
- void [gslc_DrvPageFlipNow](#) ([gslc_tsGui](#) *pGui)
Force a page flip to occur.
- bool [gslc_DrvDrawPoint](#) ([gslc_tsGui](#) *pGui, int16_t nX, int16_t nY, [gslc_tsColor](#) nCol)
Draw a point.
- bool [gslc_DrvDrawPoints](#) ([gslc_tsGui](#) *pGui, [gslc_tsPt](#) *asPt, uint16_t nNumPt, [gslc_tsColor](#) nCol)
Draw a point.
- bool [gslc_DrvDrawFrameRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
Draw a framed rectangle.
- bool [gslc_DrvDrawFillRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
Draw a filled rectangle.

- bool [gslc_DrvDrawFrameRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)
 - Draw a framed rounded rectangle.*
- bool [gslc_DrvDrawFillRoundRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, int16_t nRadius, [gslc_tsColor](#) nCol)
 - Draw a filled rounded rectangle.*
- bool [gslc_DrvDrawLine](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, [gslc_tsColor](#) nCol)
 - Draw a line.*
- bool [gslc_DrvDrawFrameCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)
 - Draw a framed circle.*
- bool [gslc_DrvDrawFillCircle](#) ([gslc_tsGui](#) *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, [gslc_tsColor](#) nCol)
 - Draw a filled circle.*
- bool [gslc_DrvDrawFrameTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)
 - Draw a framed triangle.*
- bool [gslc_DrvDrawFillTriangle](#) ([gslc_tsGui](#) *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, [gslc_tsColor](#) nCol)
 - Draw a filled triangle.*
- bool [gslc_DrvDrawImage](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, [gslc_tsImgRef](#) slmgRef)
 - Copy all of source image to destination screen at specified coordinate.*
- void [gslc_DrvDrawMonoFromMem](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)
 - Draw a monochrome bitmap from a memory array.*
- void [gslc_DrvDrawBmp24FromMem](#) ([gslc_tsGui](#) *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)
 - Draw a color 24-bit depth bitmap from a memory array.*
- void [gslc_DrvDrawBkgnd](#) ([gslc_tsGui](#) *pGui)
 - Copy the background image to destination screen.*
- bool [gslc_DrvInitTouch](#) ([gslc_tsGui](#) *pGui, const char *acDev)
 - Perform any touchscreen-specific initialization.*
- bool [gslc_DrvGetTouch](#) ([gslc_tsGui](#) *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, [gslc_tsInputRawEvent](#) *pInputEvent, int16_t *pnInputVal)
 - Get the last touch event from the internal touch handler.*
- bool [gslc_DrvRotate](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation)
 - Change rotation, automatically adapt touchscreen axes swap/flip.*
- uint16_t [gslc_DrvAdaptColorToRaw](#) ([gslc_tsColor](#) nCol)

9.51.1 Detailed Description

GUIslice library (driver layer for UTFT)

9.51.2 Macro Definition Documentation

9.51.2.1 #define DRV_HAS_DRAW_BMP_MEM

Support [gslc_DrvDrawBmp24FromMem\(\)](#)

9.51.2.2 #define DRV_HAS_DRAW_CIRCLE_FILL

Support [gslc_DrvDrawFillCircle\(\)](#)

9.51.2.3 #define DRV_HAS_DRAW_CIRCLE_FRAME

Support [gslc_DrvDrawFrameCircle\(\)](#)

9.51.2.4 #define DRV_HAS_DRAW_LINE

Support [gslc_DrvDrawLine\(\)](#)

9.51.2.5 #define DRV_HAS_DRAW_POINT

Support [gslc_DrvDrawPoint\(\)](#)

9.51.2.6 #define DRV_HAS_DRAW_POINTS

Support [gslc_DrvDrawPoints\(\)](#)

9.51.2.7 #define DRV_HAS_DRAW_RECT_FILL

Support [gslc_DrvDrawFillRect\(\)](#)

9.51.2.8 #define DRV_HAS_DRAW_RECT_FRAME

Support [gslc_DrvDrawFrameRect\(\)](#)

9.51.2.9 #define DRV_HAS_DRAW_RECT_ROUND_FILL

Support [gslc_DrvDrawFillRoundRect\(\)](#)

9.51.2.10 #define DRV_HAS_DRAW_RECT_ROUND_FRAME

Support [gslc_DrvDrawFrameRoundRect\(\)](#)

9.51.2.11 #define DRV_HAS_DRAW_TEXT

Support [gslc_DrvDrawTxt\(\)](#)

9.51.2.12 `#define DRV_HAS_DRAW_TRI_FILL`

Support [gslc_DrvDrawFillTriangle\(\)](#)

9.51.2.13 `#define DRV_HAS_DRAW_TRI_FRAME`

Support [gslc_DrvDrawFrameTriangle\(\)](#)

9.51.2.14 `#define DRV_OVERRIDE_TXT_ALIGN`

Driver provides text alignment.

9.51.3 Function Documentation

9.51.3.1 `uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)`

9.51.3.2 `void gslc_DrvDestruct (gslc_tsGui * pGui)`

Free up any members associated with the driver.

- Eg. renderers, windows, background surfaces, etc.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.51.3.3 `void gslc_DrvDrawBkgnd (gslc_tsGui * pGui)`

Copy the background image to destination screen.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.51.3.4 void gslc_DrvDrawBmp24FromMem (gslc_tsGui * *pGui*, int16_t *nDstX*, int16_t *nDstY*, const unsigned char * *pBitmap*, bool *bProgMem*)

Draw a color 24-bit depth bitmap from a memory array.

- Note that users must convert images from their native format (eg. BMP, PNG, etc.) into a C array. Please refer to the following guide for details: <https://github.com/ImpulseAdventure/GUIslice/wiki/Display-Images-from-FLASH>
- The converted file (c array) can then be included in the sketch.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	X coord for copy
in	<i>nDstY</i>	Y coord for copy
in	<i>pBitmap</i>	Pointer to bitmap buffer
in	<i>bProgMem</i>	Bitmap is stored in Flash if true, RAM otherwise

Returns

none

9.51.3.5 bool gslc_DrvDrawFillCircle (gslc_tsGui * *pGui*, int16_t *nMidX*, int16_t *nMidY*, uint16_t *nRadius*, gslc_tsColor *nCol*)

Draw a filled circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nMidX</i>	Center of circle (X coordinate)
in	<i>nMidY</i>	Center of circle (Y coordinate)
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.51.3.6 bool gslc_DrvDrawFillRect (gslc_tsGui * *pGui*, gslc_tsRect *rRect*, gslc_tsColor *nCol*)

Draw a filled rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.51.3.7 `bool gslc_DrvDrawFillRoundRect (gslc_tsGui * pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a filled rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to fill
in	<i>nRadius</i>	Radius for rounded corners
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.51.3.8 `bool gslc_DrvDrawFillTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a filled triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value to fill

Returns

true if success, false if error

9.51.3.9 `bool gslc_DrvDrawFrameCircle (gslc_tsGui * pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)`

Draw a framed circle.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>nMidX</i>	Center of circle (X coordinate)
in	<i>nMidY</i>	Center of circle (Y coordinate)
in	<i>nRadius</i>	Radius of circle
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.51.3.10 `bool gslc_DrvDrawFrameRect (gslc_tsGui * pGui, gslc_tsRect rRect, gslc_tsColor nCol)`

Draw a framed rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.51.3.11 `bool gslc_DrvDrawFrameRoundRect (gslc_tsGui * pGui, gslc_tsRect rRect, int16_t nRadius, gslc_tsColor nCol)`

Draw a framed rounded rectangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>rRect</i>	Rectangular region to frame
in	<i>nRadius</i>	Radius for rounded corners
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.51.3.12 `bool gslc_DrvDrawFrameTriangle (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)`

Draw a framed triangle.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	X Coordinate #1
in	<i>nY0</i>	Y Coordinate #1
in	<i>nX1</i>	X Coordinate #2
in	<i>nY1</i>	Y Coordinate #2
in	<i>nX2</i>	X Coordinate #3
in	<i>nY2</i>	Y Coordinate #3
in	<i>nCol</i>	Color RGB value to frame

Returns

true if success, false if error

9.51.3.13 `bool gslc_DrvDrawImage (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, gslc_tsImgRef sImgRef)`

Copy all of source image to destination screen at specified coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.51.3.14 `bool gslc_DrvDrawLine (gslc_tsGui * pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)`

Draw a line.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX0</i>	Line start (X coordinate)
in	<i>nY0</i>	Line start (Y coordinate)
in	<i>nX1</i>	Line finish (X coordinate)
in	<i>nY1</i>	Line finish (Y coordinate)
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.51.3.15 void gslc_DrvDrawMonoFromMem (gslc_tsGui * *pGui*, int16_t *nDstX*, int16_t *nDstY*, const unsigned char * *pBitmap*, bool *bProgMem*)

Draw a monochrome bitmap from a memory array.

- Draw from the bitmap buffer using the foreground color defined in the header (unset bits are transparent)

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nDstX</i>	Destination X coord for copy
in	<i>nDstY</i>	Destination Y coord for copy
in	<i>pBitmap</i>	Pointer to bitmap buffer
in	<i>bProgMem</i>	Bitmap is stored in Flash if true, RAM otherwise

Returns

none

9.51.3.16 bool gslc_DrvDrawPoint (gslc_tsGui * *pGui*, int16_t *nX*, int16_t *nY*, gslc_tsColor *nCol*)

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nX</i>	X coordinate of point
in	<i>nY</i>	Y coordinate of point
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.51.3.17 bool gslc_DrvDrawPoints (gslc_tsGui * *pGui*, gslc_tsPt * *asPt*, uint16_t *nNumPt*, gslc_tsColor *nCol*)

Draw a point.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>asPt</i>	Array of points to draw
in	<i>n</i> ↔ <i>NumPt</i>	Number of points in array
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.51.3.18 `bool gslc_DrvDrawTxt (gslc_tsGui * pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt, gslc_tsColor colBg)`

Draw a text string at the given coordinate.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nTxtX</i>	X coordinate of top-left text string
in	<i>nTxtY</i>	Y coordinate of top-left text string
in	<i>pFont</i>	Ptr to Font
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
in	<i>colTxt</i>	Color to draw text
in	<i>colBg</i>	unused in ADAGFX, defaults to black

Returns

true if success, false if failure

9.51.3.19 `const void* gslc_DrvFontAdd (gslc_teFontRefType eFontRefType, const void * pvFontRef, uint16_t nFontSz)`

Load a font from a resource and return pointer to it.

Parameters

in	<i>eFontRefType</i>	Font reference type (GSLC_FONTREF_PTR for Arduino)
in	<i>pvFontRef</i>	Font reference pointer (Pointer to the GFXFont array)
in	<i>nFontSz</i>	Typeface size to use

Returns

Void ptr to driver-specific font if load was successful, NULL otherwise

9.51.3.20 void gslc_DrvFontsDestruct (gslc_tsGui * pGui)

Release all fonts defined in the GUI.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.51.3.21 void* gslc_DrvGetDriverDisp (gslc_tsGui * pGui)

Get the native display driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the display driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.51.3.22 void* gslc_DrvGetDriverTouch (gslc_tsGui * pGui)

Get the native touch driver instance.

- This can be useful to access special commands available in the selected driver.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

Void pointer to the touch driver instance. This pointer should be typecast to the particular driver being used. If no driver was created then this function will return NULL.

9.51.3.23 `const char* gslc_DrvGetNameDisp (gslc_tsGui * pGui)`

Get the display driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.51.3.24 `const char* gslc_DrvGetNameTouch (gslc_tsGui * pGui)`

Get the touch driver name.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

String containing driver name

9.51.3.25 `bool gslc_DrvGetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress, gslc_teInputRawEvent * peInputEvent, int16_t * pnInputVal)`

Get the last touch event from the internal touch handler.

Parameters

in	<i>pGui</i>	Pointer to GUI
out	<i>pnX</i>	Ptr to X coordinate of last touch event
out	<i>pnY</i>	Ptr to Y coordinate of last touch event
out	<i>pnPress</i>	Ptr to Pressure level of last touch event (0 for none, 1 for touch)
out	<i>peInputEvent</i>	Indication of event type
out	<i>pnInputVal</i>	Additional data for event type

Returns

true if an event was detected or false otherwise

9.51.3.26 `bool gslc_DrvGetTxtSize (gslc_tsGui * pGui, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, int16_t * pnTxtX, int16_t * pnTxtY, uint16_t * pnTxtSzW, uint16_t * pnTxtSzH)`

Get the extent (width and height) of a text string.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Parameters

in	<i>pFont</i>	Ptr to Font structure
in	<i>pStr</i>	String to display
in	<i>eTxtFlags</i>	Flags associated with text string
out	<i>pnTxtX</i>	Ptr to offset X of text
out	<i>pnTxtY</i>	Ptr to offset Y of text
out	<i>pnTxtSzW</i>	Ptr to width of text
out	<i>pnTxtSzH</i>	Ptr to height of text

Returns

true if success, false if failure

9.51.3.27 void gslc_DrvImageDestruct (void * *pvlmg*)

Release an image surface.

Parameters

in	<i>pvlmg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.51.3.28 bool gslc_DrvInit (gslc_tsGui * *pGui*)

Initialize the SDL library.

- Performs clean startup workaround (if enabled)
- Configures video mode
- Initializes font support

PRE:

- The environment variables should be configured before calling [gslc_DrvInit\(\)](#). This can be done with `gslc_↵` `DrvInitEnv()` or manually in user function.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

true if success, false if fail

9.51.3.29 bool gslc_DrvInitTouch (gslc_tsGui * *pGui*, const char * *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

9.51.3.30 bool gslc_DrvInitTs (gslc_tsGui * *pGui*, const char * *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

9.51.3.31 void* gslc_DrvLoadImage (gslc_tsGui * *pGui*, gslc_tsImgRef *sImgRef*)

Load a bitmap (*.bmp) and create a new image resource.

Transparency is enabled by GSLC_BMP_TRANS_EN through use of color (GSLC_BMP_TRANS_RGB).

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

Image pointer (surface/texture) or NULL if error

9.51.3.32 void gslc_DrvPageFlipNow (gslc_tsGui * *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

Returns

none

9.51.3.33 bool gslc_DrvRotate (gslc_tsGui * *pGui*, uint8_t *nRotation*)

Change rotation, automatically adapt touchscreen axes swap/flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)

Returns

true if successful

9.51.3.34 bool gslc_DrvSetBkgndColor (gslc_tsGui * *pGui*, gslc_tsColor *nCol*)

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nCol</i>	RGB Color to use

Returns

true if success, false if fail

9.51.3.35 bool gslc_DrvSetBkgndImage (gslc_tsGui * *pGui*, gslc_tsImgRef *sImgRef*)

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if fail

9.51.3.36 bool gslc_DrvSetClipRect (gslc_tsGui * *pGui*, gslc_tsRect * *pRect*)

Set the clipping rectangle for future drawing updates.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pRect</i>	Rectangular region to constrain edits

Returns

true if success, false if error

9.51.3.37 bool gslc_DrvSetElemImageGlow (gslc_tsGui * *pGui*, gslc_tsElem * *pElem*, gslc_tsImgRef *sImgRef*)

Set an element's glow-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

Returns

true if success, false if error

9.51.3.38 bool gslc_DrvSetElemImageNorm (gslc_tsGui * *pGui*, gslc_tsElem * *pElem*, gslc_tsImgRef *sImgRef*)

Set an element's normal-state image.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElem</i>	Pointer to Element to update
in	<i>sImgRef</i>	Image reference

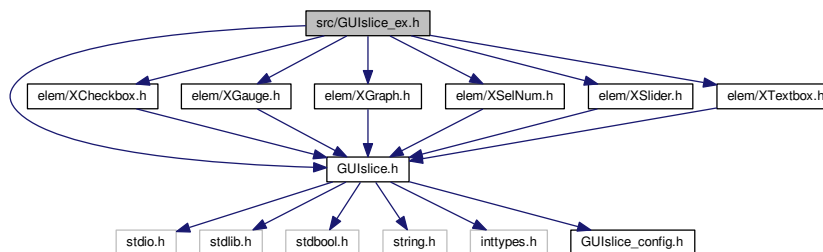
Returns

true if success, false if error

9.52 src/GUISlice_ex.h File Reference

```
#include "GUISlice.h"
#include "elem/XCheckbox.h"
#include "elem/XGauge.h"
#include "elem/XGraph.h"
#include "elem/XSelNum.h"
#include "elem/XSlider.h"
#include "elem/XTextbox.h"
```

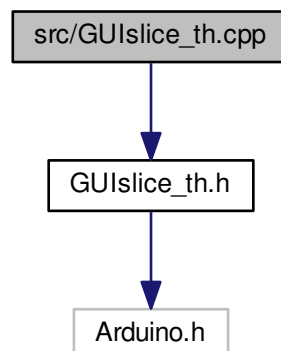
Include dependency graph for GUISlice_ex.h:



9.53 src/GUISlice_th.cpp File Reference

```
#include "GUISlice_th.h"
```

Include dependency graph for GUISlice_th.cpp:



Functions

- void `gslc_InitTouchHandler` (`TouchHandler *pTH`)
- `TouchHandler *` `gslc_getTouchHandler` (`void`)

Variables

- `TouchHandler *` `pTouchHandler`

9.53.1 Function Documentation

9.53.1.1 `TouchHandler*` `gslc_getTouchHandler` (`void`)

9.53.1.2 `void` `gslc_InitTouchHandler` (`TouchHandler *` *pTH*)

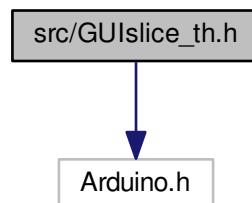
9.53.2 Variable Documentation

9.53.2.1 `TouchHandler*` `pTouchHandler`

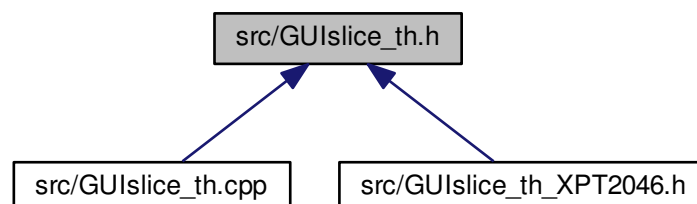
9.54 src/GUIslice_th.h File Reference

```
#include <Arduino.h>
```

Include dependency graph for GUIslice_th.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- class [THPoint](#)
- class [TouchHandler](#)

Functions

- void [gslc_InitTouchHandler](#) ([TouchHandler](#) *pTHO)
- [TouchHandler](#) * [gslc_getTouchHandler](#) (void)

9.54.1 Function Documentation

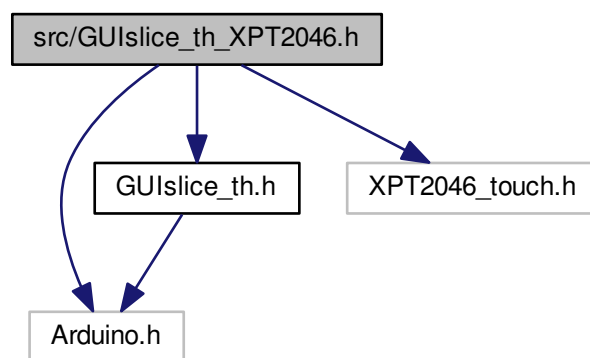
9.54.1.1 [TouchHandler](#)* [gslc_getTouchHandler](#) (void)

9.54.1.2 void [gslc_InitTouchHandler](#) ([TouchHandler](#) * *pTHO*)

9.55 src/GUIslice_th_XPT2046.h File Reference

```
#include <Arduino.h>
#include <GUIslice_th.h>
#include <XPT2046_touch.h>
```

Include dependency graph for GUIslice_th_XPT2046.h:

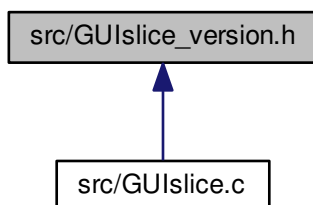


Data Structures

- class [TouchHandler_XPT2046](#)

9.56 src/GUISlice_version.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- `#define` [GUISLICE_VER](#)

9.56.1 Macro Definition Documentation

9.56.1.1 `#define` GUISLICE_VER

