

GUIslice

0.11.2

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_GetNameDisp(gslc_tsGui *pGui)	14
7.1.2.3	gslc_GetNameTouch(gslc_tsGui *pGui)	14
7.1.2.4	gslc_GetVer(gslc_tsGui *pGui)	15
7.1.2.5	gslc_GuiRotate(gslc_tsGui *pGui, uint8_t nRotation)	15
7.1.2.6	gslc_Init(gslc_tsGui *pGui, void *pvDriver, gslc_tsPage *asPage, uint8_t nMax↔ Page, gslc_tsFont *asFont, uint8_t nMaxFont)	15

7.1.2.7	<code>gslc_InitDebug(GSLC_CB_DEBUG_OUT pfunc)</code>	16
7.1.2.8	<code>gslc_Quit(gslc_tsGui *pGui)</code>	16
7.1.2.9	<code>gslc_SetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	17
7.1.2.10	<code>gslc_SetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	17
7.1.2.11	<code>gslc_SetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	17
7.1.2.12	<code>gslc_Update(gslc_tsGui *pGui)</code>	18
7.2	Graphics General Functions	19
7.2.1	Detailed Description	19
7.2.2	Function Documentation	20
7.2.2.1	<code>gslc_ClipLine(gslc_tsRect *pClipRect, int16_t *pnX0, int16_t *pnY0, int16_t *pnX1, int16_t *pnY1)</code>	20
7.2.2.2	<code>gslc_ClipPt(gslc_tsRect *pClipRect, int16_t nX, int16_t nY)</code>	20
7.2.2.3	<code>gslc_ClipRect(gslc_tsRect *pClipRect, gslc_tsRect *pRect)</code>	20
7.2.2.4	<code>gslc_ColorBlend2(gslc_tsColor colStart, gslc_tsColor colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)</code>	21
7.2.2.5	<code>gslc_ColorBlend3(gslc_tsColor colStart, gslc_tsColor colMid, gslc_tsColor colEnd, uint16_t nMidAmt, uint16_t nBlendAmt)</code>	21
7.2.2.6	<code>gslc_ColorEqual(gslc_tsColor a, gslc_tsColor b)</code>	21
7.2.2.7	<code>gslc_cosFX(int16_t n64Ang)</code>	22
7.2.2.8	<code>gslc_ExpandRect(gslc_tsRect rRect, int16_t nExpandW, int16_t nExpandH)</code>	22
7.2.2.9	<code>gslc_GetImageFromFile(const char *pFname, gslc_telmgRefFlags eFmt)</code>	22
7.2.2.10	<code>gslc_GetImageFromProg(const unsigned char *plmgBuf, gslc_telmgRefFlags eFmt)</code>	23
7.2.2.11	<code>gslc_GetImageFromRam(unsigned char *plmgBuf, gslc_telmgRefFlags eFmt)</code>	23
7.2.2.12	<code>gslc_GetImageFromSD(const char *pFname, gslc_telmgRefFlags eFmt)</code>	23
7.2.2.13	<code>gslc_IsInRect(int16_t nSelX, int16_t nSelY, gslc_tsRect rRect)</code>	23
7.2.2.14	<code>gslc_IsInWH(int16_t nSelX, int16_t nSelY, uint16_t nWidth, uint16_t nHeight)</code>	24
7.2.2.15	<code>gslc_PolarToXY(uint16_t nRad, int16_t n64Ang, int16_t *nDX, int16_t *nDY)</code>	24
7.2.2.16	<code>gslc_sinFX(int16_t n64Ang)</code>	25
7.3	Graphics Primitive Functions	26
7.3.1	Detailed Description	26
7.3.2	Function Documentation	26

7.3.2.1	<code>gslc_DrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)</code>	26
7.3.2.2	<code>gslc_DrawFillQuad(gslc_tsGui *pGui, gslc_tsPt *psPt, gslc_tsColor nCol)</code>	27
7.3.2.3	<code>gslc_DrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	27
7.3.2.4	<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>	27
7.3.2.5	<code>gslc_DrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)</code>	28
7.3.2.6	<code>gslc_DrawFrameQuad(gslc_tsGui *pGui, gslc_tsPt *psPt, gslc_tsColor nCol)</code>	28
7.3.2.7	<code>gslc_DrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	29
7.3.2.8	<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>	29
7.3.2.9	<code>gslc_DrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)</code>	29
7.3.2.10	<code>gslc_DrawLineH(gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nW, gslc_tsColor nCol)</code>	30
7.3.2.11	<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>	30
7.3.2.12	<code>gslc_DrawLineV(gslc_tsGui *pGui, int16_t nX, int16_t nY, uint16_t nH, gslc_tsColor nCol)</code>	30
7.3.2.13	<code>gslc_DrawSetPixel(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)</code>	31
7.4	Font Functions	32
7.4.1	Detailed Description	32
7.4.2	Function Documentation	32
7.4.2.1	<code>gslc_FontAdd(gslc_tsGui *pGui, int16_t nFontId, gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)</code>	32
7.4.2.2	<code>gslc_FontGet(gslc_tsGui *pGui, int16_t nFontId)</code>	32
7.5	Page Functions	34
7.5.1	Detailed Description	34
7.5.2	Function Documentation	34
7.5.2.1	<code>gslc_GetPageCur(gslc_tsGui *pGui)</code>	34
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>	35
7.5.2.3	<code>gslc_PageFindElemById(gslc_tsGui *pGui, int16_t nPageId, int16_t nElemId)</code>	35
7.5.2.4	<code>gslc_PageRedrawGet(gslc_tsGui *pGui)</code>	36

7.5.2.5	<code>gslc_PageRedrawSet(gslc_tsGui *pGui, bool bRedraw)</code>	36
7.5.2.6	<code>gslc_PopupHide(gslc_tsGui *pGui)</code>	36
7.5.2.7	<code>gslc_PopupShow(gslc_tsGui *pGui, int16_t nPageId, bool bModal)</code>	36
7.5.2.8	<code>gslc_SetPageBase(gslc_tsGui *pGui, int16_t nPageId)</code>	37
7.5.2.9	<code>gslc_SetPageCur(gslc_tsGui *pGui, int16_t nPageId)</code>	37
7.5.2.10	<code>gslc_SetPageOverlay(gslc_tsGui *pGui, int16_t nPageId)</code>	37
7.5.2.11	<code>gslc_SetStackPage(gslc_tsGui *pGui, uint8_t nStackPos, int16_t nPageId)</code>	38
7.5.2.12	<code>gslc_SetStackState(gslc_tsGui *pGui, uint8_t nStackPos, bool bActive, bool b↵ DoDraw)</code>	38
7.6	Element Functions	39
7.6.1	Detailed Description	39
7.7	Element: Creation Functions	40
7.7.1	Detailed Description	40
7.7.2	Function Documentation	40
7.7.2.1	<code>gslc_ElemCreateBox(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_↵ tsRect rElem)</code>	40
7.7.2.2	<code>gslc_ElemCreateBtnImg(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, ↵ gslc_tsRect rElem, gslc_tsImgRef slmgRef, gslc_tsImgRef slmgRefSel, GSLC_↵ _CB_TOUCH cbTouch)</code>	41
7.7.2.3	<code>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)</code>	41
7.7.2.4	<code>gslc_ElemCreateImg(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_↵ tsRect rElem, gslc_tsImgRef slmgRef)</code>	42
7.7.2.5	<code>gslc_ElemCreateLine(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, int16_t ↵ nX0, int16_t nY0, int16_t nX1, int16_t nY1)</code>	42
7.7.2.6	<code>gslc_ElemCreateTxt(gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_↵ tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)</code>	43
7.8	Element: General Functions	44
7.8.1	Detailed Description	44
7.8.2	Function Documentation	44
7.8.2.1	<code>gslc_ElemGetId(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	44
7.9	Element: Update Functions	45
7.9.1	Detailed Description	46

7.9.2	Function Documentation	46
7.9.2.1	<code>gslc_ElemGetGlow(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	46
7.9.2.2	<code>gslc_ElemGetGlowEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	46
7.9.2.3	<code>gslc_ElemGetGroup(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	47
7.9.2.4	<code>gslc_ElemGetRedraw(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	47
7.9.2.5	<code>gslc_ElemGetVisible(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	47
7.9.2.6	<code>gslc_ElemOwnsCoord(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nX, int16_t nY, bool bOnlyClickEn)</code>	48
7.9.2.7	<code>gslc_ElemSetClickEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bClickEn)</code>	48
7.9.2.8	<code>gslc_ElemSetCol(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colFrame, gslc_tsColor colFill, gslc_tsColor colFillGlow)</code>	48
7.9.2.9	<code>gslc_ElemSetDrawFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_DRAW funcCb)</code>	49
7.9.2.10	<code>gslc_ElemSetFillEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFillEn)</code>	49
7.9.2.11	<code>gslc_ElemSetFrameEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFrameEn)</code>	50
7.9.2.12	<code>gslc_ElemSetGlow(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bGlowing)</code>	50
7.9.2.13	<code>gslc_ElemSetGlowCol(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colFrameGlow, gslc_tsColor colFillGlow, gslc_tsColor colTxtGlow)</code>	50
7.9.2.14	<code>gslc_ElemSetGlowEn(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bGlowEn)</code>	51
7.9.2.15	<code>gslc_ElemSetGroup(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int nGroupId)</code>	51
7.9.2.16	<code>gslc_ElemSetRedraw(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tRedrawType eRedraw)</code>	51
7.9.2.17	<code>gslc_ElemSetStyleFrom(gslc_tsGui *pGui, gslc_tsElemRef *pElemRefSrc, gslc_tsElemRef *pElemRefDest)</code>	52
7.9.2.18	<code>gslc_ElemSetTickFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_TICK funcCb)</code>	52
7.9.2.19	<code>gslc_ElemSetTxtAlign(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned nAlign)</code>	52
7.9.2.20	<code>gslc_ElemSetTxtCol(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colVal)</code>	53
7.9.2.21	<code>gslc_ElemSetTxtEnc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tTxtFlags eFlags)</code>	53
7.9.2.22	<code>gslc_ElemSetTxtMargin(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned nMargin)</code>	54

7.9.2.23	<code>gslc_ElemSetTxtMem(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_te↔ TxtFlags eFlags)</code>	54
7.9.2.24	<code>gslc_ElemSetTxtStr(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, const char *pStr)</code>	54
7.9.2.25	<code>gslc_ElemSetVisible(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bVisible)</code>	55
7.9.2.26	<code>gslc_ElemUpdateFont(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int nFontId)</code>	55
7.10	Touchscreen Functions	56
7.10.1	Detailed Description	56
7.10.2	Macro Definition Documentation	56
7.10.2.1	<code>TOUCH_ROTATION_DATA</code>	56
7.10.2.2	<code>TOUCH_ROTATION_DATA</code>	57
7.10.2.3	<code>TOUCH_ROTATION_FLIPX</code>	57
7.10.2.4	<code>TOUCH_ROTATION_FLIPX</code>	57
7.10.2.5	<code>TOUCH_ROTATION_FLIPY</code>	57
7.10.2.6	<code>TOUCH_ROTATION_FLIPY</code>	57
7.10.2.7	<code>TOUCH_ROTATION_SWAPXY</code>	57
7.10.2.8	<code>TOUCH_ROTATION_SWAPXY</code>	57
7.10.3	Function Documentation	57
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>	57
7.10.3.2	<code>gslc_InitTouch(gslc_tsGui *pGui, const char *acDev)</code>	57
7.10.3.3	<code>gslc_SetTouchRemapCal(gslc_tsGui *pGui, uint16_t nXMin, uint16_t nXMax, uint16_t nYMin, uint16_t nYMax)</code>	58
7.10.3.4	<code>gslc_SetTouchRemapEn(gslc_tsGui *pGui, bool bEn)</code>	58
7.10.3.5	<code>gslc_SetTouchRemapYX(gslc_tsGui *pGui, bool bSwap)</code>	58
7.11	Input Mapping Functions	60
7.11.1	Detailed Description	60
7.11.2	Function Documentation	60
7.11.2.1	<code>gslc_InitInputMap(gslc_tsGui *pGui, gslc_tsInputMap *asInputMap, uint8_t n↔ InputMapMax)</code>	60
7.11.2.2	<code>gslc_InputMapAdd(gslc_tsGui *pGui, gslc_teInputRawEvent eInputEvent, int16_t nInputVal, gslc_teAction eAction, int16_t nActionVal)</code>	60
7.11.2.3	<code>gslc_SetPinPollFunc(gslc_tsGui *pGui, GSLC_CB_PIN_POLL pfunc)</code>	60

7.12 General Purpose Macros	61
7.12.1 Detailed Description	61
7.12.2 Macro Definition Documentation	61
7.12.2.1 GSLC_DEBUG_PRINT	61
7.12.2.2 GSLC_DEBUG_PRINT_CONST	61
7.13 Flash-based Element Macros	62
7.13.1 Detailed Description	62
7.13.2 Macro Definition Documentation	62
7.13.2.1 gslc_ElemCreateBox_P	62
7.13.2.2 gslc_ElemCreateBtnTxt_P	63
7.13.2.3 gslc_ElemCreateLine_P	63
7.13.2.4 gslc_ElemCreateTxt_P	63
7.13.2.5 gslc_ElemCreateTxt_P_R	64
7.14 Internal Functions	65
7.14.1 Detailed Description	70
7.14.2 Variable Documentation	70
7.14.2.1 abPageStackActive	70
7.14.2.2 abPageStackDoDraw	70
7.14.2.3 apPageStack	70
7.14.2.4 asElem	70
7.14.2.5 asElemRef	70
7.14.2.6 asFont	70
7.14.2.7 asInputMap	71
7.14.2.8 asPage	71
7.14.2.9 b	71
7.14.2.10 bRedrawPartialEn	71
7.14.2.11 bScreenNeedFlip	71
7.14.2.12 bScreenNeedRedraw	71
7.14.2.13 bTouchRemapEn	71
7.14.2.14 bTouchRemapYX	71

7.14.2.15 colElemFill	71
7.14.2.16 colElemFillGlow	71
7.14.2.17 colElemFrame	72
7.14.2.18 colElemFrameGlow	72
7.14.2.19 colElemText	72
7.14.2.20 colElemTextGlow	72
7.14.2.21 eAction	72
7.14.2.22 eElemFlags	72
7.14.2.23 eEvent	72
7.14.2.24 eFontRefType	72
7.14.2.25 elmgFlags	72
7.14.2.26 eInitStatTouch	72
7.14.2.27 eTouch	73
7.14.2.28 eTxtAlign	73
7.14.2.29 eTxtFlags	73
7.14.2.30 eType	73
7.14.2.31 g	73
7.14.2.32 h	73
7.14.2.33 nActionVal	73
7.14.2.34 nDisp0H	73
7.14.2.35 nDisp0W	73
7.14.2.36 nDispDepth	73
7.14.2.37 nDispH	74
7.14.2.38 nDispW	74
7.14.2.39 nElemAutoldNext	74
7.14.2.40 nElemCnt	74
7.14.2.41 nElemIndFocused	74
7.14.2.42 nElemMax	74
7.14.2.43 nElemRefCnt	74
7.14.2.44 nElemRefMax	74

7.14.2.45 nFeatures	74
7.14.2.46 nFlipX	74
7.14.2.47 nFlipY	75
7.14.2.48 nFontCnt	75
7.14.2.49 nFontMax	75
7.14.2.50 nFrameRateCnt	75
7.14.2.51 nFrameRateStart	75
7.14.2.52 nGroup	75
7.14.2.53 nId	75
7.14.2.54 nId	75
7.14.2.55 nInputMapCnt	75
7.14.2.56 nInputMapMax	75
7.14.2.57 nPageCnt	76
7.14.2.58 nPageId	76
7.14.2.59 nPageMax	76
7.14.2.60 nRotation	76
7.14.2.61 nSize	76
7.14.2.62 nStrBufMax	76
7.14.2.63 nSubType	76
7.14.2.64 nSwapXY	76
7.14.2.65 nTouchCalXMax	76
7.14.2.66 nTouchCalXMin	76
7.14.2.67 nTouchCalYMax	77
7.14.2.68 nTouchCalYMin	77
7.14.2.69 nTouchLastPress	77
7.14.2.70 nTouchLastX	77
7.14.2.71 nTouchLastY	77
7.14.2.72 nTouchRotation	77
7.14.2.73 nTxtMargin	77
7.14.2.74 nType	77

7.14.2.75 nVal	77
7.14.2.76 nX	77
7.14.2.77 nY	78
7.14.2.78 pElem	78
7.14.2.79 pElemRefParent	78
7.14.2.80 pElemRefTracked	78
7.14.2.81 pFname	78
7.14.2.82 pfuncPinPoll	78
7.14.2.83 pfuncXDraw	78
7.14.2.84 pfuncXEvent	78
7.14.2.85 pfuncXTick	78
7.14.2.86 pfuncXTouch	79
7.14.2.87 plmgBuf	79
7.14.2.88 pStrBuf	79
7.14.2.89 pTxtFont	79
7.14.2.90 pvData	79
7.14.2.91 pvDriver	79
7.14.2.92 pvFont	79
7.14.2.93 pvImgRaw	79
7.14.2.94 pvScope	79
7.14.2.95 pXData	79
7.14.2.96 r	80
7.14.2.97 rElem	80
7.14.2.98 sCollect	80
7.14.2.99 sElemTmpProg	80
7.14.2.100sImgRefBkgnd	80
7.14.2.101sImgRefGlow	80
7.14.2.102sImgRefNorm	80
7.14.2.103w	80
7.14.2.104x	80

7.14.2.105x	80
7.14.2.106y	80
7.14.2.107y	80
7.15 Internal: Misc Functions	81
7.15.1 Detailed Description	81
7.15.2 Function Documentation	81
7.15.2.1 gslc_ResetImage()	81
7.16 Internal: Element Functions	82
7.16.1 Detailed Description	82
7.16.2 Function Documentation	82
7.16.2.1 gslc_ElemAdd(gslc_tsGui *pGui, int16_t nPageId, gslc_tsElem *pElem, gslc_teElemRefFlags eFlags)	82
7.16.2.2 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)	83
7.16.2.3 gslc_ElemDraw(gslc_tsGui *pGui, int16_t nPageId, int16_t nElemId)	83
7.16.2.4 gslc_ElemDrawByRef(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)	84
7.16.2.5 gslc_ElemSetImage(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsImgRef sImgRef, gslc_tsImgRef sImgRefSel)	84
7.16.2.6 gslc_GetElemFromRef(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	84
7.16.2.7 gslc_GetElemRefFlag(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nFlagMask)	85
7.16.2.8 gslc_SetElemRefFlag(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nFlagMask, uint8_t nFlagVal)	85
7.17 Internal: Page Functions	86
7.17.1 Detailed Description	86
7.17.2 Function Documentation	86
7.17.2.1 gslc_ElemEvent(void *pvGui, gslc_tsEvent sEvent)	86
7.17.2.2 gslc_ElemSendEventTouch(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teTracked, gslc_teTouch eTouch, int16_t nX, int16_t nY)	87
7.17.2.3 gslc_EventCreate(gslc_tsGui *pGui, gslc_teEventType eType, uint8_t nSubType, void *pvScope, void *pvData)	87
7.17.2.4 gslc_PageEvent(void *pvGui, gslc_tsEvent sEvent)	88
7.17.2.5 gslc_PageFindById(gslc_tsGui *pGui, int16_t nPageId)	88

7.17.2.6	<code>gslc_PageFlipGet(gslc_tsGui *pGui)</code>	88
7.17.2.7	<code>gslc_PageFlipGo(gslc_tsGui *pGui)</code>	88
7.17.2.8	<code>gslc_PageFlipSet(gslc_tsGui *pGui, bool bNeeded)</code>	89
7.17.2.9	<code>gslc_PageFocusStep(gslc_tsGui *pGui, gslc_tsPage *pPage, bool bNext)</code>	89
7.17.2.10	<code>gslc_PageRedrawCalc(gslc_tsGui *pGui)</code>	89
7.17.2.11	<code>gslc_PageRedrawGo(gslc_tsGui *pGui)</code>	90
7.18	Internal: Element Collection Functions	91
7.18.1	Detailed Description	92
7.18.2	Function Documentation	92
7.18.2.1	<code>gslc_CollectElemAdd(gslc_tsGui *pGui, gslc_tsCollect *pCollect, const gslc_tsElem *pElem, gslc_teElemRefFlags eFlags)</code>	92
7.18.2.2	<code>gslc_CollectFindElemById(gslc_tsGui *pGui, gslc_tsCollect *pCollect, int16_t nElemId)</code>	92
7.18.2.3	<code>gslc_CollectFindElemFromCoord(gslc_tsGui *pGui, gslc_tsCollect *pCollect, int16_t nX, int16_t nY)</code>	92
7.18.2.4	<code>gslc_CollectFindFocusStep(gslc_tsGui *pGui, gslc_tsCollect *pCollect, bool bNext, bool *pbWrapped, int16_t *pnElemInd)</code>	93
7.18.2.5	<code>gslc_CollectGetElemRefTracked(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	93
7.18.2.6	<code>gslc_CollectGetFocus(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	93
7.18.2.7	<code>gslc_CollectGetNextId(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	94
7.18.2.8	<code>gslc_CollectGetRedraw(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	94
7.18.2.9	<code>gslc_CollectReset(gslc_tsCollect *pCollect, gslc_tsElem *asElem, uint16_t nElemMax, gslc_tsElemRef *asElemRef, uint16_t nElemRefMax)</code>	94
7.18.2.10	<code>gslc_CollectSetElemTracked(gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsElemRef *pElemRef)</code>	95
7.18.2.11	<code>gslc_CollectSetFocus(gslc_tsGui *pGui, gslc_tsCollect *pCollect, int16_t nElemInd)</code>	95
7.18.2.12	<code>gslc_CollectSetParent(gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsElemRef *pElemRefParent)</code>	95
7.19	Internal: Element Collection Event Functions	97
7.19.1	Detailed Description	97
7.19.2	Function Documentation	97
7.19.2.1	<code>gslc_CollectEvent(void *pvGui, gslc_tsEvent sEvent)</code>	97

7.19.2.2	<code>gslc_CollectInput(gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsEventTouch *pEventTouch)</code>	97
7.19.2.3	<code>gslc_CollectTouch(gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsEventTouch *pEventTouch)</code>	98
7.20	Internal: Tracking Functions	99
7.20.1	Detailed Description	99
7.20.2	Function Documentation	99
7.20.2.1	<code>gslc_InputMapLookup(gslc_tsGui *pGui, gslc_tInputRawEvent eInputEvent, int16_t nInputVal, gslc_teAction *peAction, int16_t *pnActionVal)</code>	99
7.20.2.2	<code>gslc_TrackInput(gslc_tsGui *pGui, gslc_tsPage *pPage, gslc_tInputRawEvent eInputEvent, int16_t nInputVal)</code>	99
7.20.2.3	<code>gslc_TrackTouch(gslc_tsGui *pGui, gslc_tsPage *pPage, int16_t nX, int16_t nY, uint16_t nPress)</code>	100
7.21	Internal: Cleanup Functions	101
7.21.1	Detailed Description	101
7.21.2	Function Documentation	101
7.21.2.1	<code>gslc_CollectDestruct(gslc_tsGui *pGui, gslc_tsCollect *pCollect)</code>	101
7.21.2.2	<code>gslc_ElemDestruct(gslc_tsElem *pElem)</code>	102
7.21.2.3	<code>gslc_GuiDestruct(gslc_tsGui *pGui)</code>	103
7.21.2.4	<code>gslc_PageDestruct(gslc_tsGui *pGui, gslc_tsPage *pPage)</code>	103
7.21.2.5	<code>gslc_ResetElem(gslc_tsElem *pElem)</code>	103
7.21.2.6	<code>gslc_ResetFont(gslc_tsFont *pFont)</code>	104
8	Data Structure Documentation	105
8.1	<code>gslc_tsCollect</code> Struct Reference	105
8.1.1	Detailed Description	106
8.2	<code>gslc_tsColor</code> Struct Reference	106
8.2.1	Detailed Description	106
8.3	<code>gslc_tsDriver</code> Struct Reference	107
8.3.1	Field Documentation	107
8.3.1.1	<code>nColBkgnd</code>	107
8.3.1.2	<code>rClipRect</code>	107
8.4	<code>gslc_tsElem</code> Struct Reference	108

8.4.1	Detailed Description	109
8.5	gslc_tsElemRef Struct Reference	110
8.5.1	Detailed Description	110
8.6	gslc_tsEvent Struct Reference	110
8.6.1	Detailed Description	111
8.7	gslc_tsEventTouch Struct Reference	111
8.7.1	Detailed Description	111
8.8	gslc_tsFont Struct Reference	111
8.8.1	Detailed Description	112
8.9	gslc_tsGui Struct Reference	112
8.9.1	Detailed Description	114
8.10	gslc_tsImgRef Struct Reference	114
8.10.1	Detailed Description	115
8.11	gslc_tsInputMap Struct Reference	115
8.11.1	Detailed Description	115
8.12	gslc_tsPage Struct Reference	116
8.12.1	Detailed Description	116
8.13	gslc_tsPt Struct Reference	116
8.13.1	Detailed Description	117
8.14	gslc_tsRect Struct Reference	117
8.14.1	Detailed Description	117
8.15	gslc_tsXCheckbox Struct Reference	118
8.15.1	Detailed Description	118
8.15.2	Field Documentation	118
8.15.2.1	bChecked	118
8.15.2.2	bRadio	118
8.15.2.3	colCheck	119
8.15.2.4	nStyle	119
8.15.2.5	pfuncXToggle	119
8.16	gslc_tsXGauge Struct Reference	119

8.16.1 Detailed Description	120
8.16.2 Field Documentation	120
8.16.2.1 bFlip	120
8.16.2.2 bIndicFill	120
8.16.2.3 bValLastValid	121
8.16.2.4 bVert	121
8.16.2.5 colGauge	121
8.16.2.6 colTick	121
8.16.2.7 nIndicLen	121
8.16.2.8 nIndicTip	121
8.16.2.9 nMax	121
8.16.2.10 nMin	121
8.16.2.11 nStyle	121
8.16.2.12 nTickCnt	121
8.16.2.13 nTickLen	122
8.16.2.14 nVal	122
8.16.2.15 nValLast	122
8.17 gslc_tsXGraph Struct Reference	122
8.17.1 Detailed Description	123
8.17.2 Field Documentation	123
8.17.2.1 bScrollEn	123
8.17.2.2 colGraph	123
8.17.2.3 eStyle	124
8.17.2.4 nBufCnt	124
8.17.2.5 nBufMax	124
8.17.2.6 nMargin	124
8.17.2.7 nPlotIndMax	124
8.17.2.8 nPlotIndStart	124
8.17.2.9 nPlotValMax	124
8.17.2.10 nPlotValMin	124

8.17.2.11 nScrollPos	124
8.17.2.12 nWndHeight	124
8.17.2.13 nWndWidth	125
8.17.2.14 pBuf	125
8.18 gslc_tsXSlider Struct Reference	125
8.18.1 Detailed Description	126
8.18.2 Field Documentation	126
8.18.2.1 bTrim	126
8.18.2.2 bVert	126
8.18.2.3 colTick	126
8.18.2.4 colTrim	126
8.18.2.5 nPos	126
8.18.2.6 nPosMax	126
8.18.2.7 nPosMin	127
8.18.2.8 nThumbSz	127
8.18.2.9 nTickDiv	127
8.18.2.10 nTickLen	127
8.18.2.11 pfuncXPos	127
8.19 gslc_tsXTextbox Struct Reference	127
8.19.1 Detailed Description	128
8.19.2 Field Documentation	128
8.19.2.1 bScrollEn	128
8.19.2.2 bWrapEn	128
8.19.2.3 nBufCols	128
8.19.2.4 nBufPosX	128
8.19.2.5 nBufPosY	129
8.19.2.6 nBufRows	129
8.19.2.7 nChSizeX	129
8.19.2.8 nChSizeY	129
8.19.2.9 nCurPosX	129

8.19.2.10	nCurPosY	129
8.19.2.11	nMarginX	129
8.19.2.12	nMarginY	129
8.19.2.13	nRedrawRow	129
8.19.2.14	nScrollPos	129
8.19.2.15	nWndCols	130
8.19.2.16	nWndRows	130
8.19.2.17	nWndRowStart	130
8.19.2.18	pBuf	130
8.20	THPoint Class Reference	130
8.20.1	Constructor & Destructor Documentation	131
8.20.1.1	THPoint(void)	131
8.20.1.2	THPoint(int16_t x, int16_t y, int16_t z)	131
8.20.2	Member Function Documentation	131
8.20.2.1	operator!=(THPoint)	131
8.20.2.2	operator==(THPoint)	131
8.20.3	Field Documentation	131
8.20.3.1	x	131
8.20.3.2	y	131
8.20.3.3	z	131
8.21	TouchHandler Class Reference	131
8.21.1	Constructor & Destructor Documentation	132
8.21.1.1	TouchHandler()	132
8.21.2	Member Function Documentation	132
8.21.2.1	begin(void)	132
8.21.2.2	getPoint(void)	132
8.21.2.3	scale(THPoint pln)	132
8.21.2.4	setCalibration(uint16_t ts_xMin, uint16_t ts_xMax, uint16_t ts_yMin, uint16_t ts_yMax)	132
8.21.2.5	setSize(uint16_t _disp_xSize, uint16_t _disp_ySize)	132
8.21.2.6	setSwapFlip(bool _swapXY, bool _flipX, bool _flipY)	132
8.22	TouchHandler_XPT2046 Class Reference	133
8.22.1	Constructor & Destructor Documentation	134
8.22.1.1	TouchHandler_XPT2046(SPIClass &spi, uint8_t spi_cs_pin)	134
8.22.2	Member Function Documentation	134
8.22.2.1	begin(void)	134
8.22.2.2	getPoint(void)	134
8.22.3	Field Documentation	134
8.22.3.1	spi	134
8.22.3.2	touchDriver	134

9	File Documentation	135
9.1	README.md File Reference	135
9.2	src/elem/XCheckbox.c File Reference	135
9.2.1	Function Documentation	136
9.2.1.1	gslc_ElemXCheckboxCreate(gslc_tsGui *pGui, int16_t nElemId, int16_t n↔ Page, gslc_tsXCheckbox *pXData, gslc_tsRect rElem, bool bRadio, gslc_teX↔ CheckboxStyle nStyle, gslc_tsColor colCheck, bool bChecked)	136
9.2.1.2	gslc_ElemXCheckboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)	136
9.2.1.3	gslc_ElemXCheckboxFindChecked(gslc_tsGui *pGui, int16_t nGroupId)	137
9.2.1.4	gslc_ElemXCheckboxGetState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	137
9.2.1.5	gslc_ElemXCheckboxSetState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bChecked)	137
9.2.1.6	gslc_ElemXCheckboxSetStateFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElem↔ Ref, GSLC_CB_XCHECKBOX pfuncCb)	138
9.2.1.7	gslc_ElemXCheckboxSetStateHelp(gslc_tsGui *pGui, gslc_tsElemRef *pElem↔ Ref, bool bChecked)	138
9.2.1.8	gslc_ElemXCheckboxToggleState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	138
9.2.1.9	gslc_ElemXCheckboxTouch(void *pvGui, void *pvElemRef, gslc_teTouch e↔ Touch, int16_t nRelX, int16_t nRelY)	138
9.2.2	Variable Documentation	139
9.2.2.1	ERRSTR_NULL	139
9.2.2.2	ERRSTR_PXD_NULL	139
9.3	src/elem/XCheckbox.h File Reference	139
9.3.1	Macro Definition Documentation	140
9.3.1.1	gslc_ElemXCheckboxCreate_P	140
9.3.1.2	GSLC_TYPEX_CHECKBOX	141
9.3.2	Typedef Documentation	141
9.3.2.1	GSLC_CB_XCHECKBOX	141
9.3.3	Enumeration Type Documentation	141
9.3.3.1	gslc_teXCheckboxStyle	141
9.3.4	Function Documentation	142

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>	142
9.3.4.2	<code>gslc_ElemXCheckboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	143
9.3.4.3	<code>gslc_ElemXCheckboxFindChecked(gslc_tsGui *pGui, int16_t nGroupId)</code>	143
9.3.4.4	<code>gslc_ElemXCheckboxGetState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	144
9.3.4.5	<code>gslc_ElemXCheckboxSetState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bChecked)</code>	144
9.3.4.6	<code>gslc_ElemXCheckboxSetStateFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XCHECKBOX pfuncCb)</code>	144
9.3.4.7	<code>gslc_ElemXCheckboxToggleState(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	144
9.3.4.8	<code>gslc_ElemXCheckboxTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	145
9.4	<code>src/elem/XGauge.c</code> File Reference	145
9.4.1	Function Documentation	147
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>	147
9.4.1.2	<code>gslc_ElemXGaugeDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	147
9.4.1.3	<code>gslc_ElemXGaugeDrawProgressBar(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teRedrawType eRedraw)</code>	148
9.4.1.4	<code>gslc_ElemXGaugeSetFlip(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)</code>	148
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>	148
9.4.1.6	<code>gslc_ElemXGaugeSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teXGaugeStyle nStyle)</code>	149
9.4.1.7	<code>gslc_ElemXGaugeSetTicks(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colTick, uint16_t nTickCnt, uint16_t nTickLen)</code>	149
9.4.1.8	<code>gslc_ElemXGaugeUpdate(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)</code>	150
9.4.2	Variable Documentation	150
9.4.2.1	<code>ERRSTR_NULL</code>	150
9.4.2.2	<code>ERRSTR_PXD_NULL</code>	150
9.5	<code>src/elem/XGauge.h</code> File Reference	150

9.5.1	Macro Definition Documentation	152
9.5.1.1	gslc_ElemXGaugeCreate_P	152
9.5.1.2	GSLC_TYPEX_GAUGE	152
9.5.2	Enumeration Type Documentation	152
9.5.2.1	gslc_teXGaugeStyle	152
9.5.3	Function Documentation	153
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)	153
9.5.3.2	gslc_ElemXGaugeDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔Redraw)	153
9.5.3.3	gslc_ElemXGaugeDrawProgressBar(gslc_tsGui *pGui, gslc_tsElemRef *p↔ElemRef, gslc_teRedrawType eRedraw)	154
9.5.3.4	gslc_ElemXGaugeSetFlip(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bFlip)	154
9.5.3.5	gslc_ElemXGaugeSetIndicator(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)	154
9.5.3.6	gslc_ElemXGaugeSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_teXGaugeStyle nType)	155
9.5.3.7	gslc_ElemXGaugeSetTicks(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor colTick, uint16_t nTickCnt, uint16_t nTickLen)	155
9.5.3.8	gslc_ElemXGaugeUpdate(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16↔_t nVal)	156
9.6	src/elem/XGraph.c File Reference	156
9.6.1	Function Documentation	157
9.6.1.1	gslc_ElemXGraphAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal) 157	
9.6.1.2	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)	157
9.6.1.3	gslc_ElemXGraphDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔Redraw)	158
9.6.1.4	gslc_ElemXGraphScrollSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)	158
9.6.1.5	gslc_ElemXGraphSetRange(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nYMin, int16_t nYMax)	159
9.6.1.6	gslc_ElemXGraphSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc↔_teXGraphStyle eStyle, uint8_t nMargin)	159

9.6.2	Variable Documentation	159
9.6.2.1	ERRSTR_NULL	159
9.6.2.2	ERRSTR_PXD_NULL	159
9.7	src/elem/XGraph.h File Reference	159
9.7.1	Macro Definition Documentation	161
9.7.1.1	GSLC_TYPEX_GRAPH	161
9.7.2	Enumeration Type Documentation	161
9.7.2.1	gslc_teXGraphStyle	161
9.7.3	Function Documentation	161
9.7.3.1	gslc_ElemXGraphAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nVal)	161
9.7.3.2	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)	162
9.7.3.3	gslc_ElemXGraphDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↵ Redraw)	162
9.7.3.4	gslc_ElemXGraphScrollSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)	162
9.7.3.5	gslc_ElemXGraphSetRange(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nYMin, int16_t nYMax)	163
9.7.3.6	gslc_ElemXGraphSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc↵_teXGraphStyle eStyle, uint8_t nMargin)	163
9.8	src/elem/XSelNum.c File Reference	163
9.8.1	Variable Documentation	164
9.8.1.1	ERRSTR_NULL	164
9.8.1.2	ERRSTR_PXD_NULL	164
9.9	src/elem/XSelNum.h File Reference	164
9.10	src/elem/XSlider.c File Reference	165
9.10.1	Function Documentation	166
9.10.1.1	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)	166
9.10.1.2	gslc_ElemXSliderDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↵ Redraw)	166
9.10.1.3	gslc_ElemXSliderGetPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	167

9.10.1.4	<code>gslc_ElemXSliderSetPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nPos)</code>	167
9.10.1.5	<code>gslc_ElemXSliderSetPosFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XSLIDER_POS funcCb)</code>	167
9.10.1.6	<code>gslc_ElemXSliderSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bTrim, gslc_tsColor colTrim, uint16_t nTickDiv, int16_t nTickLen, gslc_tsColor colTick)</code>	168
9.10.1.7	<code>gslc_ElemXSliderTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	168
9.10.2	Variable Documentation	169
9.10.2.1	<code>ERRSTR_NULL</code>	169
9.10.2.2	<code>ERRSTR_PXD_NULL</code>	169
9.11	<code>src/elm/XSlider.h</code> File Reference	169
9.11.1	Macro Definition Documentation	170
9.11.1.1	<code>gslc_ElemXSliderCreate_P</code>	170
9.11.1.2	<code>GSLC_TYPEX_SLIDER</code>	171
9.11.2	Typedef Documentation	171
9.11.2.1	<code>GSLC_CB_XSLIDER_POS</code>	171
9.11.3	Function Documentation	171
9.11.3.1	<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>	171
9.11.3.2	<code>gslc_ElemXSliderDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↵ Redraw)</code>	172
9.11.3.3	<code>gslc_ElemXSliderGetPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	172
9.11.3.4	<code>gslc_ElemXSliderSetPos(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, int16_t nPos)</code>	172
9.11.3.5	<code>gslc_ElemXSliderSetPosFunc(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, GSLC_CB_XSLIDER_POS funcCb)</code>	173
9.11.3.6	<code>gslc_ElemXSliderSetStyle(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bTrim, gslc_tsColor colTrim, uint16_t nTickDiv, int16_t nTickLen, gslc_tsColor colTick)</code>	173
9.11.3.7	<code>gslc_ElemXSliderTouch(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)</code>	173
9.12	<code>src/elm/XTextbox.c</code> File Reference	174
9.12.1	Function Documentation	175

9.12.1.1	<code>gslc_ElemXTextboxAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, char *pTxt)</code>	175
9.12.1.2	<code>gslc_ElemXTextboxBufAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, unsigned char chNew, bool bAdvance)</code>	175
9.12.1.3	<code>gslc_ElemXTextboxColReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	175
9.12.1.4	<code>gslc_ElemXTextboxColSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor nCol)</code>	176
9.12.1.5	<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>	176
9.12.1.6	<code>gslc_ElemXTextboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)</code>	177
9.12.1.7	<code>gslc_ElemXTextboxLineWrAdv(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	177
9.12.1.8	<code>gslc_ElemXTextboxReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	177
9.12.1.9	<code>gslc_ElemXTextboxScrollSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)</code>	177
9.12.1.10	<code>gslc_ElemXTextboxWrapSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bWrapEn)</code>	178
9.12.2	Variable Documentation	178
9.12.2.1	<code>ERRSTR_NULL</code>	178
9.12.2.2	<code>ERRSTR_PXD_NULL</code>	178
9.13	<code>src/elem/XTextbox.h</code> File Reference	178
9.13.1	Macro Definition Documentation	180
9.13.1.1	<code>GSLC_TYPEX_TEXTBOX</code>	180
9.13.1.2	<code>GSLC_XTEXTBOX_CODE_COL_RESET</code>	180
9.13.1.3	<code>GSLC_XTEXTBOX_CODE_COL_SET</code>	180
9.13.1.4	<code>XTEXTBOX_REDRAW_ALL</code>	180
9.13.1.5	<code>XTEXTBOX_REDRAW_NONE</code>	180
9.13.2	Function Documentation	180
9.13.2.1	<code>gslc_ElemXTextboxAdd(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, char *pTxt)</code>	180
9.13.2.2	<code>gslc_ElemXTextboxColReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)</code>	180
9.13.2.3	<code>gslc_ElemXTextboxColSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsColor nCol)</code>	181
9.13.2.4	<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>	181

9.13.2.5	gslc_ElemXTextboxDraw(void *pvGui, void *pvElemRef, gslc_teRedrawType e↔ Redraw)	182
9.13.2.6	gslc_ElemXTextboxReset(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)	182
9.13.2.7	gslc_ElemXTextboxScrollSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)	182
9.13.2.8	gslc_ElemXTextboxWrapSet(gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, bool bWrapEn)	183
9.14	src/GUIslice.c File Reference	183
9.14.1	Enumeration Type Documentation	190
9.14.1.1	gslc_teDebugPrintState	190
9.14.2	Function Documentation	190
9.14.2.1	gslc_OrderCoord(int16_t *pnX0, int16_t *pnY0, int16_t *pnX1, int16_t *pnY1)	190
9.14.2.2	gslc_SwapCoords(int16_t *pnXa, int16_t *pnYa, int16_t *pnXb, int16_t *pnYb)	190
9.14.3	Variable Documentation	190
9.14.3.1	ERRSTR_NULL	190
9.14.3.2	ERRSTR_PXD_NULL	190
9.14.3.3	g_pfDebugOut	190
9.14.3.4	m_nLUTSinF0X16	191
9.15	src/GUIslice.h File Reference	191
9.15.1	Macro Definition Documentation	203
9.15.1.1	GSLC_2PI	203
9.15.1.2	GSLC_ALIGN_BOT_LEFT	203
9.15.1.3	GSLC_ALIGN_BOT_MID	203
9.15.1.4	GSLC_ALIGN_BOT_RIGHT	204
9.15.1.5	GSLC_ALIGN_MID_LEFT	204
9.15.1.6	GSLC_ALIGN_MID_MID	204
9.15.1.7	GSLC_ALIGN_MID_RIGHT	204
9.15.1.8	GSLC_ALIGN_TOP_LEFT	204
9.15.1.9	GSLC_ALIGN_TOP_MID	204
9.15.1.10	GSLC_ALIGN_TOP_RIGHT	204
9.15.1.11	GSLC_ALIGNH_LEFT	204
9.15.1.12	GSLC_ALIGNH_MID	204

9.15.1.13 GSLC_ALIGNH_RIGHT	204
9.15.1.14 GSLC_ALIGNV_BOT	205
9.15.1.15 GSLC_ALIGNV_MID	205
9.15.1.16 GSLC_ALIGNV_TOP	205
9.15.1.17 GSLC_COL_BLACK	205
9.15.1.18 GSLC_COL_BLUE	205
9.15.1.19 GSLC_COL_BLUE_DK1	205
9.15.1.20 GSLC_COL_BLUE_DK2	205
9.15.1.21 GSLC_COL_BLUE_DK3	205
9.15.1.22 GSLC_COL_BLUE_DK4	205
9.15.1.23 GSLC_COL_BLUE_LT1	205
9.15.1.24 GSLC_COL_BLUE_LT2	206
9.15.1.25 GSLC_COL_BLUE_LT3	206
9.15.1.26 GSLC_COL_BLUE_LT4	206
9.15.1.27 GSLC_COL_BROWN	206
9.15.1.28 GSLC_COL_CYAN	206
9.15.1.29 GSLC_COL_GRAY	206
9.15.1.30 GSLC_COL_GRAY_DK1	206
9.15.1.31 GSLC_COL_GRAY_DK2	206
9.15.1.32 GSLC_COL_GRAY_DK3	206
9.15.1.33 GSLC_COL_GRAY_LT1	206
9.15.1.34 GSLC_COL_GRAY_LT2	207
9.15.1.35 GSLC_COL_GRAY_LT3	207
9.15.1.36 GSLC_COL_GREEN	207
9.15.1.37 GSLC_COL_GREEN_DK1	207
9.15.1.38 GSLC_COL_GREEN_DK2	207
9.15.1.39 GSLC_COL_GREEN_DK3	207
9.15.1.40 GSLC_COL_GREEN_DK4	207
9.15.1.41 GSLC_COL_GREEN_LT1	207
9.15.1.42 GSLC_COL_GREEN_LT2	207

9.15.1.43	GSLC_COL_GREEN_LT3	207
9.15.1.44	GSLC_COL_GREEN_LT4	208
9.15.1.45	GSLC_COL_MAGENTA	208
9.15.1.46	GSLC_COL_ORANGE	208
9.15.1.47	GSLC_COL_PURPLE	208
9.15.1.48	GSLC_COL_RED	208
9.15.1.49	GSLC_COL_RED_DK1	208
9.15.1.50	GSLC_COL_RED_DK2	208
9.15.1.51	GSLC_COL_RED_DK3	208
9.15.1.52	GSLC_COL_RED_DK4	208
9.15.1.53	GSLC_COL_RED_LT1	208
9.15.1.54	GSLC_COL_RED_LT2	209
9.15.1.55	GSLC_COL_RED_LT3	209
9.15.1.56	GSLC_COL_RED_LT4	209
9.15.1.57	GSLC_COL_TEAL	209
9.15.1.58	GSLC_COL_WHITE	209
9.15.1.59	GSLC_COL_YELLOW	209
9.15.1.60	GSLC_COL_YELLOW_DK	209
9.15.1.61	GSLC_COLMONO_BLACK	209
9.15.1.62	GSLC_COLMONO_WHITE	209
9.15.1.63	GSLC_ELEM_FEA_CLICK_EN	209
9.15.1.64	GSLC_ELEM_FEA_FILL_EN	210
9.15.1.65	GSLC_ELEM_FEA_FRAME_EN	210
9.15.1.66	GSLC_ELEM_FEA_GLOW_EN	210
9.15.1.67	GSLC_ELEM_FEA_NONE	210
9.15.1.68	GSLC_ELEM_FEA_VALID	210
9.15.1.69	GSLC_ELEMREF_DEFAULT	210
9.15.1.70	GSLC_PMEM	210
9.15.2	Typedef Documentation	210
9.15.2.1	GSLC_CB_DEBUG_OUT	210

9.15.2.2	GSLC_CB_DRAW	210
9.15.2.3	GSLC_CB_EVENT	210
9.15.2.4	GSLC_CB_PIN_POLL	211
9.15.2.5	GSLC_CB_TICK	211
9.15.2.6	GSLC_CB_TOUCH	211
9.15.2.7	gslc_tsColor	211
9.15.2.8	gslc_tsElem	211
9.15.2.9	gslc_tsEvent	211
9.15.2.10	gslc_tsEventTouch	211
9.15.2.11	gslc_tsPt	211
9.15.2.12	gslc_tsRect	212
9.15.3	Enumeration Type Documentation	212
9.15.3.1	gslc_teAction	212
9.15.3.2	gslc_teElemId	212
9.15.3.3	gslc_teElemInd	213
9.15.3.4	gslc_teElemRefFlags	213
9.15.3.5	gslc_teEventSubType	213
9.15.3.6	gslc_teEventType	214
9.15.3.7	gslc_teFontId	214
9.15.3.8	gslc_teFontRefType	214
9.15.3.9	gslc_teGroupId	214
9.15.3.10	gslc_telmgRefFlags	215
9.15.3.11	gslc_telnitStat	215
9.15.3.12	gslc_telInputRawEvent	215
9.15.3.13	gslc_tePageId	216
9.15.3.14	gslc_tePin	216
9.15.3.15	gslc_teRedrawType	216
9.15.3.16	gslc_teStackPage	217
9.15.3.17	gslc_teTouch	217
9.15.3.18	gslc_teTxtFlags	218

9.15.3.19	gslc_teTypeCore	218
9.15.4	Variable Documentation	218
9.15.4.1	g_pfDebugOut	218
9.16	src/GUIslice_config.h File Reference	219
9.17	src/GUIslice_config_ard.h File Reference	219
9.17.1	Macro Definition Documentation	220
9.17.1.1	ADAGFX_PIN_CLK	220
9.17.1.2	ADAGFX_PIN_CS	220
9.17.1.3	ADAGFX_PIN_DC	220
9.17.1.4	ADAGFX_PIN_MISO	220
9.17.1.5	ADAGFX_PIN_MOSI	220
9.17.1.6	ADAGFX_PIN_RD	220
9.17.1.7	ADAGFX_PIN_RST	220
9.17.1.8	ADAGFX_PIN_SDCS	220
9.17.1.9	ADAGFX_PIN_WR	220
9.17.1.10	ADAGFX_SPI_HW	220
9.17.1.11	ADATOUCH_FLIP_X	220
9.17.1.12	ADATOUCH_FLIP_Y	220
9.17.1.13	ADATOUCH_I2C_ADDR	220
9.17.1.14	ADATOUCH_I2C_HW	220
9.17.1.15	ADATOUCH_PIN_CS	220
9.17.1.16	ADATOUCH_SPI_HW	220
9.17.1.17	ADATOUCH_SPI_SW	220
9.17.1.18	ADATOUCH_SWAP_XY	221
9.17.1.19	ADATOUCH_X_MAX	221
9.17.1.20	ADATOUCH_X_MIN	221
9.17.1.21	ADATOUCH_Y_MAX	221
9.17.1.22	ADATOUCH_Y_MIN	221
9.17.1.23	DEBUG_ERR	221
9.17.1.24	DRV_DISP_ADAGFX	221

9.17.1.25 DRV_DISP_ADAGFX_ILI9341	221
9.17.1.26 DRV_TOUCH_ADA_STMPE610	221
9.17.1.27 GSLC_BMP_TRANS_EN	221
9.17.1.28 GSLC_BMP_TRANS_RGB	221
9.17.1.29 GSLC_CLIP_EN	221
9.17.1.30 GSLC_DEV_TOUCH	221
9.17.1.31 GSLC_FEATURE_COMPOUND	221
9.17.1.32 GSLC_FEATURE_INPUT	221
9.17.1.33 GSLC_FEATURE_XGAUGE_RADIAL	221
9.17.1.34 GSLC_FEATURE_XGAUGE_RAMP	221
9.17.1.35 GSLC_FEATURE_XTEXTBOX_EMBED	221
9.17.1.36 GSLC_LOCAL_STR	221
9.17.1.37 GSLC_LOCAL_STR_LEN	221
9.17.1.38 GSLC_ROTATE	221
9.17.1.39 GSLC_SD_BUFFPIXEL	221
9.17.1.40 GSLC_SD_EN	221
9.17.1.41 GSLC_TOUCH_MAX_EVT	222
9.17.1.42 GSLC_USE_FLOAT	222
9.17.1.43 GSLC_USE_PROGMEM	222
9.17.1.44 TOUCH_ROTATION_DATA	222
9.17.1.45 TOUCH_ROTATION_FLIPX	222
9.17.1.46 TOUCH_ROTATION_FLIPY	222
9.17.1.47 TOUCH_ROTATION_SWAPXY	222
9.18 src/GUISlice_config_linux.h File Reference	222
9.18.1 Macro Definition Documentation	223
9.18.1.1 ADATOUCH_FLIP_X	223
9.18.1.2 ADATOUCH_FLIP_Y	223
9.18.1.3 ADATOUCH_SWAP_XY	223
9.18.1.4 DEBUG_ERR	223
9.18.1.5 DRV_DISP_SDL1	223

9.18.1.6	DRV_SDL_FIX_START	223
9.18.1.7	DRV_SDL_MOUSE_SHOW	223
9.18.1.8	DRV_TOUCH_TSLIB	223
9.18.1.9	GSLC_BMP_TRANS_EN	223
9.18.1.10	GSLC_BMP_TRANS_RGB	223
9.18.1.11	GSLC_DEV_FB	223
9.18.1.12	GSLC_DEV_TOUCH	223
9.18.1.13	GSLC_DEV_VID_DRV	223
9.18.1.14	GSLC_FEATURE_COMPOUND	223
9.18.1.15	GSLC_FEATURE_INPUT	223
9.18.1.16	GSLC_FEATURE_XGAUGE_RADIAL	223
9.18.1.17	GSLC_FEATURE_XGAUGE_RAMP	223
9.18.1.18	GSLC_FEATURE_XTEXTBOX_EMBED	223
9.18.1.19	GSLC_LOCAL_STR	223
9.18.1.20	GSLC_LOCAL_STR_LEN	223
9.18.1.21	GSLC_TOUCH_MAX_EVT	223
9.18.1.22	GSLC_USE_FLOAT	223
9.18.1.23	GSLC_USE_PROGMEM	223
9.19	src/GUISlice_drv.h File Reference	223
9.20	src/GUISlice_drv_adagfx.cpp File Reference	224
9.21	src/GUISlice_drv_adagfx.h File Reference	224
9.21.1	Detailed Description	227
9.21.2	Macro Definition Documentation	227
9.21.2.1	DRV_HAS_DRAW_CIRCLE_FILL	227
9.21.2.2	DRV_HAS_DRAW_CIRCLE_FRAME	227
9.21.2.3	DRV_HAS_DRAW_LINE	227
9.21.2.4	DRV_HAS_DRAW_POINT	227
9.21.2.5	DRV_HAS_DRAW_POINTS	227
9.21.2.6	DRV_HAS_DRAW_RECT_FILL	227
9.21.2.7	DRV_HAS_DRAW_RECT_FRAME	227

9.21.2.8	DRV_HAS_DRAW_TEXT	227
9.21.2.9	DRV_HAS_DRAW_TRI_FILL	227
9.21.2.10	DRV_HAS_DRAW_TRI_FRAME	228
9.21.2.11	DRV_OVERRIDE_TXT_ALIGN	228
9.21.3	Function Documentation	228
9.21.3.1	gslc_DrvAdaptColorToRaw(gslc_tsColor nCol)	228
9.21.3.2	gslc_DrvDestruct(gslc_tsGui *pGui)	228
9.21.3.3	gslc_DrvDrawBgnd(gslc_tsGui *pGui)	228
9.21.3.4	gslc_DrvDrawBmp24FromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	228
9.21.3.5	gslc_DrvDrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	229
9.21.3.6	gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	229
9.21.3.7	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)	230
9.21.3.8	gslc_DrvDrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	230
9.21.3.9	gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	230
9.21.3.10	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)	231
9.21.3.11	gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_ts← ImgRef slmgRef)	231
9.21.3.12	gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)	232
9.21.3.13	gslc_DrvDrawMonoFromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	232
9.21.3.14	gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)	232
9.21.3.15	gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc← _tsColor nCol)	233
9.21.3.16	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)	233
9.21.3.17	gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)	233
9.21.3.18	gslc_DrvFontsDestruct(gslc_tsGui *pGui)	234
9.21.3.19	gslc_DrvGetNameDisp(gslc_tsGui *pGui)	234

9.21.3.20	<code>gslc_DrvGetNameTouch(gslc_tsGui *pGui)</code>	234
9.21.3.21	<code>gslc_DrvGetTouch(gslc_tsGui *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pnPress, gslc_teInputRawEvent *peInputEvent, int16_t *pnInputVal)</code>	234
9.21.3.22	<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>	235
9.21.3.23	<code>gslc_DrvImageDestruct(void *pvImg)</code>	235
9.21.3.24	<code>gslc_DrvInit(gslc_tsGui *pGui)</code>	236
9.21.3.25	<code>gslc_DrvInitTouch(gslc_tsGui *pGui, const char *acDev)</code>	236
9.21.3.26	<code>gslc_DrvInitTs(gslc_tsGui *pGui, const char *acDev)</code>	236
9.21.3.27	<code>gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	237
9.21.3.28	<code>gslc_DrvPageFlipNow(gslc_tsGui *pGui)</code>	237
9.21.3.29	<code>gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)</code>	237
9.21.3.30	<code>gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	238
9.21.3.31	<code>gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	238
9.21.3.32	<code>gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	238
9.21.3.33	<code>gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsImgRef sImgRef)</code>	239
9.21.3.34	<code>gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsImgRef sImgRef)</code>	239
9.22	<code>src/GUIslice_drv_m5stack.cpp</code> File Reference	239
9.23	<code>src/GUIslice_drv_m5stack.h</code> File Reference	240
9.23.1	Detailed Description	242
9.23.2	Macro Definition Documentation	242
9.23.2.1	<code>DRV_HAS_DRAW_CIRCLE_FILL</code>	242
9.23.2.2	<code>DRV_HAS_DRAW_CIRCLE_FRAME</code>	242
9.23.2.3	<code>DRV_HAS_DRAW_LINE</code>	243
9.23.2.4	<code>DRV_HAS_DRAW_POINT</code>	243
9.23.2.5	<code>DRV_HAS_DRAW_POINTS</code>	243
9.23.2.6	<code>DRV_HAS_DRAW_RECT_FILL</code>	243
9.23.2.7	<code>DRV_HAS_DRAW_RECT_FRAME</code>	243
9.23.2.8	<code>DRV_HAS_DRAW_TEXT</code>	243
9.23.2.9	<code>DRV_HAS_DRAW_TRI_FILL</code>	243

9.23.2.10	DRV_HAS_DRAW_TRI_FRAME	243
9.23.2.11	DRV_OVERRIDE_TXT_ALIGN	243
9.23.3	Function Documentation	243
9.23.3.1	gslc_DrvAdaptColorToRaw(gslc_tsColor nCol)	243
9.23.3.2	gslc_DrvDestruct(gslc_tsGui *pGui)	243
9.23.3.3	gslc_DrvDrawBkgnd(gslc_tsGui *pGui)	244
9.23.3.4	gslc_DrvDrawBmp24FromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	244
9.23.3.5	gslc_DrvDrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	244
9.23.3.6	gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	245
9.23.3.7	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)	245
9.23.3.8	gslc_DrvDrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	246
9.23.3.9	gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	246
9.23.3.10	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)	246
9.23.3.11	gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_tsGui *pGui, simgRef slmgRef)	247
9.23.3.12	gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)	247
9.23.3.13	gslc_DrvDrawMonoFromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	247
9.23.3.14	gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)	248
9.23.3.15	gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc_tsColor nCol)	248
9.23.3.16	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)	248
9.23.3.17	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)	249
9.23.3.18	gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)	249
9.23.3.19	gslc_DrvFontsDestruct(gslc_tsGui *pGui)	250
9.23.3.20	gslc_DrvGetNameDisp(gslc_tsGui *pGui)	250

9.23.3.21	<code>gslc_DrvGetNameTouch(gslc_tsGui *pGui)</code>	250
9.23.3.22	<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>	250
9.23.3.23	<code>gslc_DrvImageDestruct(void *pvImg)</code>	251
9.23.3.24	<code>gslc_DrvInit(gslc_tsGui *pGui)</code>	251
9.23.3.25	<code>gslc_DrvInitTs(gslc_tsGui *pGui, const char *acDev)</code>	252
9.23.3.26	<code>gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	252
9.23.3.27	<code>gslc_DrvPageFlipNow(gslc_tsGui *pGui)</code>	252
9.23.3.28	<code>gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)</code>	253
9.23.3.29	<code>gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	253
9.23.3.30	<code>gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef sImgRef)</code>	253
9.23.3.31	<code>gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	254
9.23.3.32	<code>gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef sImgRef)</code>	254
9.23.3.33	<code>gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef sImgRef)</code>	254
9.23.4	Variable Documentation	254
9.23.4.1	ERRSTR_NULL	254
9.23.4.2	ERRSTR_PXD_NULL	255
9.24	<code>src/GUISlice_drv_sdl.c</code> File Reference	255
9.25	<code>src/GUISlice_drv_sdl.h</code> File Reference	255
9.25.1	Detailed Description	257
9.25.2	Macro Definition Documentation	257
9.25.2.1	DRV_HAS_DRAW_POINT	257
9.25.2.2	DRV_OVERRIDE_TXT_ALIGN	257
9.25.3	Function Documentation	257
9.25.3.1	<code>gslc_DrvAdaptColor(gslc_tsColor sCol)</code>	257
9.25.3.2	<code>gslc_DrvAdaptRect(gslc_tsRect rRect)</code>	258
9.25.3.3	<code>gslc_DrvCleanStart(const char *sTTY)</code>	258
9.25.3.4	<code>gslc_DrvDestruct(gslc_tsGui *pGui)</code>	258
9.25.3.5	<code>gslc_DrvDrawBkgnd(gslc_tsGui *pGui)</code>	259

9.25.3.6	<code>gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	259
9.25.3.7	<code>gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)</code>	259
9.25.3.8	<code>gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_ts← ImgRef slmgRef)</code>	259
9.25.3.9	<code>gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)</code>	260
9.25.3.10	<code>gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol)</code>	260
9.25.3.11	<code>gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc← _tsColor nCol)</code>	261
9.25.3.12	<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>	262
9.25.3.13	<code>gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)</code>	262
9.25.3.14	<code>gslc_DrvFontsDestruct(gslc_tsGui *pGui)</code>	263
9.25.3.15	<code>gslc_DrvGetNameDisp(gslc_tsGui *pGui)</code>	263
9.25.3.16	<code>gslc_DrvGetNameTouch(gslc_tsGui *pGui)</code>	263
9.25.3.17	<code>gslc_DrvGetTouch(gslc_tsGui *pGui, int16_t *pnX, int16_t *pnY, uint16_t *pn← Press, gslc_telInputRawEvent *peInputEvent, int16_t *pnInputVal)</code>	263
9.25.3.18	<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>	264
9.25.3.19	<code>gslc_DrvImageDestruct(void *pvImg)</code>	264
9.25.3.20	<code>gslc_DrvInit(gslc_tsGui *pGui)</code>	264
9.25.3.21	<code>gslc_DrvInitTouch(gslc_tsGui *pGui, const char *acDev)</code>	265
9.25.3.22	<code>gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)</code>	265
9.25.3.23	<code>gslc_DrvPageFlipNow(gslc_tsGui *pGui)</code>	266
9.25.3.24	<code>gslc_DrvReportInfoPost()</code>	266
9.25.3.25	<code>gslc_DrvReportInfoPre()</code>	266
9.25.3.26	<code>gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)</code>	266
9.25.3.27	<code>gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)</code>	266
9.25.3.28	<code>gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)</code>	267
9.25.3.29	<code>gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)</code>	267
9.25.3.30	<code>gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef slmgRef)</code>	267

9.25.3.31	gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_tsElemRef slmgRef)	268
9.26	src/GUISlice_drv_tft_espi.cpp File Reference	268
9.27	src/GUISlice_drv_tft_espi.h File Reference	268
9.27.1	Detailed Description	271
9.27.2	Macro Definition Documentation	271
9.27.2.1	DRV_HAS_DRAW_CIRCLE_FILL	271
9.27.2.2	DRV_HAS_DRAW_CIRCLE_FRAME	271
9.27.2.3	DRV_HAS_DRAW_LINE	271
9.27.2.4	DRV_HAS_DRAW_POINT	272
9.27.2.5	DRV_HAS_DRAW_POINTS	272
9.27.2.6	DRV_HAS_DRAW_RECT_FILL	272
9.27.2.7	DRV_HAS_DRAW_RECT_FRAME	272
9.27.2.8	DRV_HAS_DRAW_TEXT	272
9.27.2.9	DRV_HAS_DRAW_TRI_FILL	272
9.27.2.10	DRV_HAS_DRAW_TRI_FRAME	272
9.27.2.11	DRV_OVERRIDE_TXT_ALIGN	272
9.27.3	Function Documentation	272
9.27.3.1	gslc_DrvAdaptColorToRaw(gslc_tsColor nCol)	272
9.27.3.2	gslc_DrvDestruct(gslc_tsGui *pGui)	272
9.27.3.3	gslc_DrvDrawBkgnd(gslc_tsGui *pGui)	273
9.27.3.4	gslc_DrvDrawBmp24FromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, const unsigned char *pBitmap, bool bProgMem)	273
9.27.3.5	gslc_DrvDrawFillCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	273
9.27.3.6	gslc_DrvDrawFillRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	274
9.27.3.7	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)	274
9.27.3.8	gslc_DrvDrawFrameCircle(gslc_tsGui *pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)	275
9.27.3.9	gslc_DrvDrawFrameRect(gslc_tsGui *pGui, gslc_tsRect rRect, gslc_tsColor nCol)	275
9.27.3.10	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)	275

9.27.3.11 gslc_DrvDrawImage(gslc_tsGui *pGui, int16_t nDstX, int16_t nDstY, gslc_ts← ImgRef slmgRef)	276
9.27.3.12 gslc_DrvDrawLine(gslc_tsGui *pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, gslc_tsColor nCol)	276
9.27.3.13 gslc_DrvDrawMonoFromMem(gslc_tsGui *pGui, int16_t nDstX, int16_t nDst← Y, const unsigned char *pBitmap, bool bProgMem)	276
9.27.3.14 gslc_DrvDrawPoint(gslc_tsGui *pGui, int16_t nX, int16_t nY, gslc_tsColor nCol) .	277
9.27.3.15 gslc_DrvDrawPoints(gslc_tsGui *pGui, gslc_tsPt *asPt, uint16_t nNumPt, gslc← _tsColor nCol)	277
9.27.3.16 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)	277
9.27.3.17 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)	278
9.27.3.18 gslc_DrvFontAdd(gslc_teFontRefType eFontRefType, const void *pvFontRef, uint16_t nFontSz)	278
9.27.3.19 gslc_DrvFontsDestruct(gslc_tsGui *pGui)	279
9.27.3.20 gslc_DrvGetNameDisp(gslc_tsGui *pGui)	279
9.27.3.21 gslc_DrvGetNameTouch(gslc_tsGui *pGui)	279
9.27.3.22 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)	280
9.27.3.23 gslc_DrvImageDestruct(void *pvImg)	280
9.27.3.24 gslc_DrvInit(gslc_tsGui *pGui)	280
9.27.3.25 gslc_DrvInitTs(gslc_tsGui *pGui, const char *acDev)	281
9.27.3.26 gslc_DrvLoadImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)	281
9.27.3.27 gslc_DrvPageFlipNow(gslc_tsGui *pGui)	281
9.27.3.28 gslc_DrvRotate(gslc_tsGui *pGui, uint8_t nRotation)	282
9.27.3.29 gslc_DrvSetBkgndColor(gslc_tsGui *pGui, gslc_tsColor nCol)	282
9.27.3.30 gslc_DrvSetBkgndImage(gslc_tsGui *pGui, gslc_tsImgRef slmgRef)	282
9.27.3.31 gslc_DrvSetClipRect(gslc_tsGui *pGui, gslc_tsRect *pRect)	283
9.27.3.32 gslc_DrvSetElemImageGlow(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef slmgRef)	283
9.27.3.33 gslc_DrvSetElemImageNorm(gslc_tsGui *pGui, gslc_tsElem *pElem, gslc_ts← ImgRef slmgRef)	283

9.28	src/GUISlice_ex.h File Reference	284
9.29	src/GUISlice_th.cpp File Reference	284
9.29.1	Function Documentation	285
9.29.1.1	gslc_getTouchHandler(void)	285
9.29.1.2	gslc_InitTouchHandler(TouchHandler *pTH)	285
9.29.2	Variable Documentation	285
9.29.2.1	pTouchHandler	285
9.30	src/GUISlice_th.h File Reference	285
9.30.1	Function Documentation	286
9.30.1.1	gslc_getTouchHandler(void)	286
9.30.1.2	gslc_InitTouchHandler(TouchHandler *pTHO)	286
9.31	src/GUISlice_th_XPT2046.h File Reference	286
9.32	src/GUISlice_version.h File Reference	287
9.32.1	Macro Definition Documentation	287
9.32.1.1	GUISLICE_VER	287

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

Features

- Pure C library, no dynamic memory allocation
- *Widgets:*
 - text, images, buttons, checkboxes, radio buttons, sliders, radial controls, scrolling textbox / terminal, graphs, etc. plus extensions and multiple pages.
- Cross-platform **GUIslice Builder** (beta) desktop application to generate layouts
- *Platform-independent* GUI core currently supports:
 - Adafruit-GFX, TFT_eSPI, SDL1.2, SDL2.0
- *Devices:*
 - Raspberry Pi, Arduino, ESP8266 / NodeMCU, ESP32, M5stack, Feather M0 (Cortex-M0), nRF52 (Cortex-M4F), LINUX, Beaglebone Black, STM32
- *Typical displays:*
 - PiTFT, Adafruit TFT 3.5" / 2.8" / 2.4" / 2.2" / 1.44", FeatherWing TFT, OLED 0.96", mcufriend, Wave-share, 4D Cape

- *Display drivers include:*
 - ILI9341, ST7735, SSD1306, HX8347D, HX8357, PCD8544
- *Touchscreen control including:*
 - STMPE610, FT6206, XPT2046, 4-wire, tslib
- Foreign characters / UTF-8 encoding (in SDL mode), anti-aliased fonts (in TFT_eSPI mode)
- Dynamic display rotation
- GPIO / pin / keyboard control for non-touchscreen devices

Screenshots

GUIslice Builder

- Includes cross-platform (Windows & LINUX) desktop application (beta) 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_ElemDraw](#) ([gslc_tsGui](#) *pGui, int16_t nPageId, int16_t nElemId)

Unused?

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	19
Graphics Primitive Functions	26
Font Functions	32
Page Functions	34
Element Functions	39
Element: Creation Functions	40
Element: General Functions	44
Element: Update Functions	45
Touchscreen Functions	56
Input Mapping Functions	60
General Purpose Macros	61
Flash-based Element Macros	62
Internal Functions	65
Internal: Misc Functions	81
Internal: Element Functions	82
Internal: Page Functions	86
Internal: Element Collection Functions	91
Internal: Element Collection Event Functions	97
Internal: Tracking Functions	99
Internal: Cleanup Functions	101

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

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

gslc_tsCollect	105
gslc_tsColor	106
gslc_tsDriver	107
gslc_tsElem	108
gslc_tsElemRef	110
gslc_tsEvent	110
gslc_tsEventTouch	111
gslc_tsFont	111
gslc_tsGui	112
gslc_tsImgRef	114
gslc_tsInputMap	115
gslc_tsPage	116
gslc_tsPt	116
gslc_tsRect	117
gslc_tsXCheckbox	118
gslc_tsXGauge	119
gslc_tsXGraph	122
gslc_tsXSlider	125
gslc_tsXTextbox	127
THPoint	130
TouchHandler	131
TouchHandler_XPT2046	133

Chapter 5

Data Structure Index

5.1 Data Structures

Here are the data structures with brief descriptions:

gslc_tsCollect	Element collection struct	105
gslc_tsColor	Color structure. Defines RGB triplet	106
gslc_tsDriver	107
gslc_tsElem	Element Struct	108
gslc_tsElemRef	Element reference structure	110
gslc_tsEvent	Event structure	110
gslc_tsEventTouch	Structure used to pass touch data through event	111
gslc_tsFont	Font reference structure	111
gslc_tsGui	GUI structure	112
gslc_tsImgRef	Image reference structure	114
gslc_tsInputMap	Input mapping	115
gslc_tsPage	Page structure	116
gslc_tsPt	Define point coordinates	116
gslc_tsRect	Rectangular region. Defines X,Y corner coordinates plus dimensions	117
gslc_tsXCheckbox	Extended data for Checkbox element	118
gslc_tsXGauge	Extended data for Gauge element	119
gslc_tsXGraph	Extended data for Graph element	122
gslc_tsXSlider	Extended data for Slider element	125

gslc_tsXTextbox	
Extended data for Textbox element	127
THPoint	130
TouchHandler	131
TouchHandler_XPT2046	133

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

src/GUISlice.c	183
src/GUISlice.h	191
src/GUISlice_config.h	219
src/GUISlice_config_ard.h	219
src/GUISlice_config_linux.h	222
src/GUISlice_drv.h	223
src/GUISlice_drv_adagfx.cpp	224
src/GUISlice_drv_adagfx.h	
GUISlice library (driver layer for Adafruit-GFX)	224
src/GUISlice_drv_m5stack.cpp	239
src/GUISlice_drv_m5stack.h	
GUISlice library (driver layer for M5stack)	240
src/GUISlice_drv_sdl.c	255
src/GUISlice_drv_sdl.h	
GUISlice library (driver layer for LINUX / SDL)	255
src/GUISlice_drv_tft_espi.cpp	268
src/GUISlice_drv_tft_espi.h	
GUISlice library (driver layer for TFT-eSPI)	268
src/GUISlice_ex.h	284
src/GUISlice_th.cpp	284
src/GUISlice_th.h	285
src/GUISlice_th_XPT2046.h	286
src/GUISlice_version.h	287
src/elem/XCheckbox.c	135
src/elem/XCheckbox.h	139
src/elem/XGauge.c	145
src/elem/XGauge.h	150
src/elem/XGraph.c	156
src/elem/XGraph.h	159
src/elem/XSelNum.c	163
src/elem/XSelNum.h	164
src/elem/XSlider.c	165
src/elem/XSlider.h	169
src/elem/XTextbox.c	174
src/elem/XTextbox.h	178

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.
- 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_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_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 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

7.1.2.3 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

7.1.2.4 char* gslc_GetVer (gslc_tsGui * pGui)

Get the GUIslice version number.

Parameters

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

Returns

String containing version number

7.1.2.5 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

7.1.2.6 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

7.1.2.7 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

7.1.2.8 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

7.1.2.9 `bool gslc_SetBgndColor (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

7.1.2.10 `bool gslc_SetBgndImage (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

7.1.2.11 `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

7.1.2.12 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.
- `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.

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.
- $\text{gslc_cosFX}(\text{nAngDeg} * 64) / 32768.0 = \cos(\text{nAngDeg} * 2\pi / 360)$

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 `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

7.2.2.14 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

7.2.2.15 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

7.2.2.16 `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.

7.3 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_DrawFillRect` (`gslc_tsGui` *pGui, `gslc_tsRect` rRect, `gslc_tsColor` nCol)
Draw a filled 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.

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_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

7.3.2.3 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

7.3.2.4 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
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

7.3.2.5 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

7.3.2.6 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

7.3.2.7 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

7.3.2.8 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

7.3.2.9 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

7.3.2.10 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

7.3.2.11 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
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

7.3.2.12 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

7.3.2.13 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_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.

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_teFontRefType eFontRefType, const void * pvFontRef, uint16_t nFontSz)`

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

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 <code>gslc_FontAdd()</code>)

Returns

A pointer to the font structure or NULL if error

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 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.

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

<code>in</code>	<code>pGui</code>	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← PageId</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 *nPageId*)

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>nPageId</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).

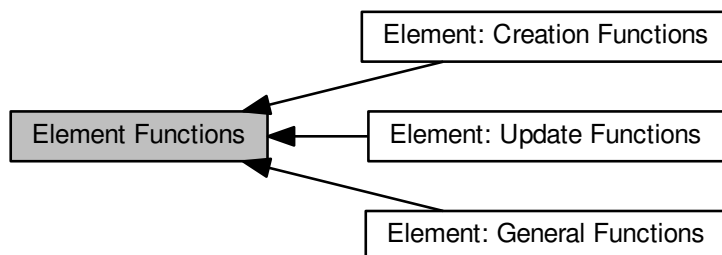
Returns

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_ElemCreatelm (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

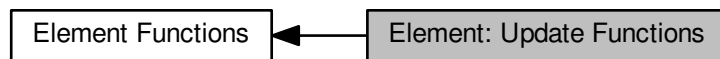
Returns

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_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_ElemSetTxtStr` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pElemRef, const char *pStr)
Update the text string associated with an Element ID.
- 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_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.*
- 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 `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

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

Get the visibility status for an element.

Parameters

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

Returns

True if element is shown, false if hidden

7.9.2.6 `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

7.9.2.7 `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

7.9.2.8 `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

7.9.2.9 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

7.9.2.10 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

7.9.2.11 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

7.9.2.12 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

7.9.2.13 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

7.9.2.14 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

7.9.2.15 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

7.9.2.16 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

7.9.2.17 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

7.9.2.18 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

7.9.2.19 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

Parameters

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

7.9.2.20 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

7.9.2.21 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

7.9.2.22 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

7.9.2.23 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

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

Update the text string associated with an Element ID.

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

7.9.2.25 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

7.9.2.26 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_tInputRawEvent *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_FLIPX(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_tsInputRawEvent * 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_DEBUG_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_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

in	sFmt	Format string for debug message
----	------	---------------------------------

7.12.2.2 #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_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)

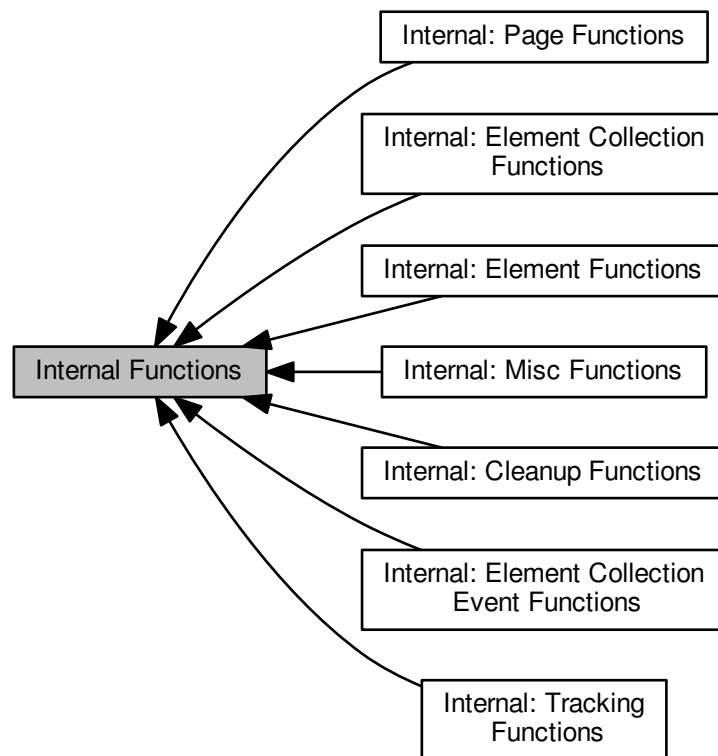
Parameters

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

7.14 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.
- `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_teImgRefFlags 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.
- [uint8_t gslc_tsElem::nTxtMargin](#)
Margin of overlay text within rect region.
- [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_teInputRawEvent gslc_tsInputMap::eEvent`
The input event.
 - `int16_t gslc_tsInputMap::nVal`
The value associated with the input event.
 - `gslc_teAction 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 y 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.
- `gslc_tsElem gslc_tsGui::sElemTmpProg`
Temporary element for Flash compatibility.
- `gslc_telnitStat gslc_tsGui::elnitStatTouch`
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::sImgRefBkgnd`
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.
- `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::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.

7.14.2.11 bool gslc_tsGui::bScreenNeedFlip

Screen requires a page flip.

7.14.2.12 bool gslc_tsGui::bScreenNeedRedraw

Screen requires a redraw.

7.14.2.13 bool gslc_tsGui::bTouchRemapEn

Enable touch remapping?

7.14.2.14 bool gslc_tsGui::bTouchRemapYX

Enable touch controller swapping of X & Y.

7.14.2.15 gslc_tsColor gslc_tsElem::colElemFill

Color for background fill.

7.14.2.16 gslc_tsColor gslc_tsElem::colElemFillGlow

Color to use for fill when glowing.

7.14.2.17 gslc_tsColor gslc_tsElem::colElemFrame

Color for frame.

7.14.2.18 gslc_tsColor gslc_tsElem::colElemFrameGlow

Color to use for frame when glowing.

7.14.2.19 gslc_tsColor gslc_tsElem::colElemText

Color of overlay text.

7.14.2.20 gslc_tsColor gslc_tsElem::colElemTextGlow

Color of overlay text when glowing.

7.14.2.21 gslc_teAction gslc_tsInputMap::eAction

Resulting action.

7.14.2.22 gslc_teElemRefFlags gslc_tsElemRef::eElemFlags

Element reference flags.

7.14.2.23 gslc_telInputRawEvent gslc_tsInputMap::eEvent

The input event.

7.14.2.24 gslc_teFontRefType gslc_tsFont::eFontRefType

Font reference type.

7.14.2.25 gslc_telmgRefFlags gslc_tslmgRef::elmngFlags

Image reference flags.

7.14.2.26 gslc_telnitStat gslc_tsGui::elnitStatTouch

Status of touch initialization.

7.14.2.27 `gslc_teTouch` `gslc_tsEventTouch::eTouch`

Touch state.

7.14.2.28 `int8_t` `gslc_tsElem::eTxtAlign`

Alignment of overlay text.

7.14.2.29 `gslc_teTxtFlags` `gslc_tsElem::eTxtFlags`

Flags associated with text buffer.

7.14.2.30 `gslc_teEventType` `gslc_tsEvent::eType`

Event type.

7.14.2.31 `uint8_t` `gslc_tsColor::g`

RGB green value.

7.14.2.32 `uint16_t` `gslc_tsRect::h`

Height of region.

7.14.2.33 `int16_t` `gslc_tsInputMap::nActionVal`

The value for the output action.

7.14.2.34 `uint16_t` `gslc_tsGui::nDisp0H`

Height of the display (pixels) in native orientation.

7.14.2.35 `uint16_t` `gslc_tsGui::nDisp0W`

Width of the display (pixels) in native orientation.

7.14.2.36 `uint8_t` `gslc_tsGui::nDispDepth`

Bit depth of display (bits per pixel)

7.14.2.37 `uint16_t gslc_tsGui::nDispH`

Height of the display (pixels)

7.14.2.38 `uint16_t gslc_tsGui::nDispW`

Width of the display (pixels)

7.14.2.39 `int16_t gslc_tsCollect::nElemAutoldNext`

Next Element ID for auto-assignment.

7.14.2.40 `uint16_t gslc_tsCollect::nElemCnt`

Number of elements allocated.

7.14.2.41 `int16_t gslc_tsCollect::nElemIndFocused`

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

7.14.2.42 `uint16_t gslc_tsCollect::nElemMax`

Maximum number of elements to allocate (in RAM)

7.14.2.43 `uint16_t gslc_tsCollect::nElemRefCnt`

Number of element references allocated.

7.14.2.44 `uint16_t gslc_tsCollect::nElemRefMax`

Maximum number of element references to allocate.

7.14.2.45 `uint8_t gslc_tsElem::nFeatures`

Element feature vector (appearance/behavior))

7.14.2.46 `uint8_t gslc_tsGui::nFlipX`

Adafruit GFX Touch Flip x axis.

7.14.2.47 uint8_t gslc_tsGui::nFlipY

Adafruit GFX Touch Flip x axis.

7.14.2.48 uint8_t gslc_tsGui::nFontCnt

Number of fonts allocated.

7.14.2.49 uint8_t gslc_tsGui::nFontMax

Maximum number of fonts to allocate.

7.14.2.50 uint8_t gslc_tsGui::nFrameRateCnt

Diagnostic frame rate count.

7.14.2.51 uint8_t gslc_tsGui::nFrameRateStart

Diagnostic frame rate timestamp.

7.14.2.52 int16_t gslc_tsElem::nGroup

Group ID that the element belongs to.

7.14.2.53 int16_t gslc_tsFont::nId

Font ID specified by user.

7.14.2.54 int16_t gslc_tsElem::nId

Element ID specified by user.

7.14.2.55 uint8_t gslc_tsGui::nInputMapCnt

Current number of input maps.

7.14.2.56 uint8_t gslc_tsGui::nInputMapMax

Maximum number of input maps.

7.14.2.57 uint8_t gslc_tsGui::nPageCnt

Current number of pages defined.

7.14.2.58 int16_t gslc_tsPage::nPageld

Page identifier.

7.14.2.59 uint8_t gslc_tsGui::nPageMax

Maximum number of pages that can be defined.

7.14.2.60 uint8_t gslc_tsGui::nRotation

Adafruit GFX Rotation of display.

7.14.2.61 uint16_t gslc_tsFont::nSize

Font size.

7.14.2.62 uint8_t gslc_tsElem::nStrBufMax

Size of string buffer.

7.14.2.63 uint8_t gslc_tsEvent::nSubType

Event sub-type.

7.14.2.64 uint8_t gslc_tsGui::nSwapXY

Adafruit GFX Touch Swap x and y axes.

7.14.2.65 uint16_t gslc_tsGui::nTouchCalXMax

Calibration X maximum reading.

7.14.2.66 uint16_t gslc_tsGui::nTouchCalXMin

Calibration X minimum reading.

7.14.2.67 uint16_t gslc_tsGui::nTouchCalYMax

Calibration Y maximum reading.

7.14.2.68 uint16_t gslc_tsGui::nTouchCalYMin

Calibration Y minimum reading.

7.14.2.69 uint16_t gslc_tsGui::nTouchLastPress

Last touch event pressure (0=none))

7.14.2.70 int16_t gslc_tsGui::nTouchLastX

Last touch event X coord.

7.14.2.71 int16_t gslc_tsGui::nTouchLastY

Last touch event Y coord.

7.14.2.72 uint8_t gslc_tsGui::nTouchRotation

Touchscreen rotation offset vs display.

7.14.2.73 uint8_t gslc_tsElem::nTxtMargin

Margin of overlay text within rect region.

7.14.2.74 int16_t gslc_tsElem::nType

Element type enumeration.

7.14.2.75 int16_t gslc_tsInputMap::nVal

The value associated with the input event.

7.14.2.76 int16_t gslc_tsEventTouch::nX

Touch X coordinate (or param1)

7.14.2.77 int16_t gslc_tsEventTouch::nY

Touch Y coordinate (or param2)

7.14.2.78 gslc_tsElem* gslc_tsElemRef::pElem

Pointer to element in memory [RAM,FLASH].

7.14.2.79 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.

7.14.2.80 gslc_tsElemRef* gslc_tsCollect::pElemRefTracked

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

7.14.2.81 const char* gslc_tsImgRef::pFname

Pathname to input image file [FILE,SD].

7.14.2.82 GSLC_CB_PIN_POLL gslc_tsGui::pfuncPinPoll

Callback func ptr for pin polling.

7.14.2.83 GSLC_CB_DRAW gslc_tsElem::pfuncXDraw

Callback func ptr for custom drawing.

7.14.2.84 GSLC_CB_EVENT gslc_tsElem::pfuncXEvent

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

7.14.2.85 GSLC_CB_TICK gslc_tsElem::pfuncXTick

Callback func ptr for timer/main loop tick.

7.14.2.86 GSLC_CB_TOUCH `gslc_tsElem::pfuncXTouch`

Callback func ptr for touch.

7.14.2.87 `const unsigned char* gslc_tsImgRef::plmgBuf`

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

7.14.2.88 `char* gslc_tsElem::pStrBuf`

Ptr to text string buffer to overlay.

7.14.2.89 `gslc_tsFont* gslc_tsElem::pTxtFont`

Ptr to Font for overlay text.

7.14.2.90 `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.

7.14.2.91 `void* gslc_tsGui::pvDriver`

Driver-specific members (`gslc_tsDriver*`)

7.14.2.92 `const void* gslc_tsFont::pvFont`

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

7.14.2.93 `void* gslc_tsImgRef::pvImgRaw`

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

7.14.2.94 `void* gslc_tsEvent::pvScope`

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

7.14.2.95 `void* gslc_tsElem::pXData`

Ptr to extended data structure.

7.14.2.96 `uint8_t gslc_tsColor::r`

RGB red value.

7.14.2.97 `gslc_tsRect gslc_tsElem::rElem`

Rect region containing element.

7.14.2.98 `gslc_tsCollect gslc_tsPage::sCollect`

Collection of elements on page.

7.14.2.99 `gslc_tsElem gslc_tsGui::sElemTmpProg`

Temporary element for Flash compatibility.

7.14.2.100 `gslc_tsImgRef gslc_tsGui::sImgRefBkgnd`

Image reference for background.

7.14.2.101 `gslc_tsImgRef gslc_tsElem::sImgRefGlow`

Image reference to draw (glowing)

7.14.2.102 `gslc_tsImgRef gslc_tsElem::sImgRefNorm`

Image reference to draw (normal)

7.14.2.103 `uint16_t gslc_tsRect::w`

Width of region.

7.14.2.104 `int16_t gslc_tsRect::x`

X coordinate of corner.

7.14.2.105 `int16_t gslc_tsPt::x`

X coordinate.

7.14.2.106 `int16_t gslc_tsRect::y`

Y coordinate of corner.

7.14.2.107 `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 \(\)](#)

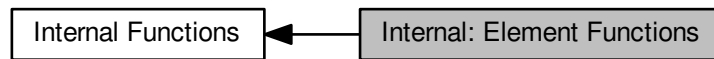
Create a blank image reference structure.

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 nPageld, 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 nPageld, [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)
- 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_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 nPageld, int16_t nElemId)
Draw an element to the active display.

7.16.1 Detailed Description

7.16.2 Function Documentation

7.16.2.1 [gslc_tsElemRef](#)* [gslc_ElemAdd](#) ([gslc_tsGui](#) * pGui, int16_t nPageld, [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</i> ↔ <i>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

7.16.2.2 `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

7.16.2.3 `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

Parameters

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

Returns

none

Todo Unused?

7.16.2.4 `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

7.16.2.5 `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

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

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

7.16.2.7 `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)

7.16.2.8 `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

7.17 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	$n \leftrightarrow$ <i>Pageld</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

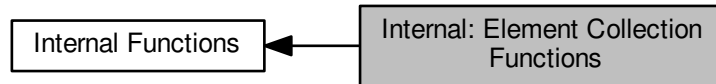
in	<i>pGui</i>	Pointer to GUI
----	-------------	----------------

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

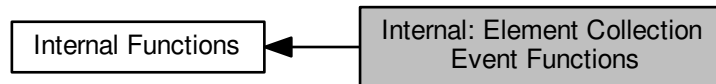
in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to the collection
in	<i>pElemRefParent</i>	Ptr to element reference that is the parent

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.
- 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

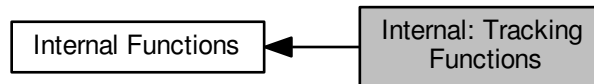
in	<i>pGui</i>	Pointer to the GUI
in	<i>pCollect</i>	Ptr to the element collection
in	<i>pEventTouch</i>	Ptr to the touch event structure

Returns

none

7.20 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

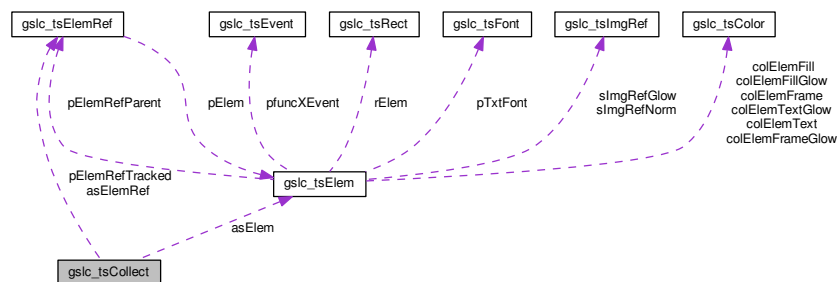
Data Structure Documentation

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.

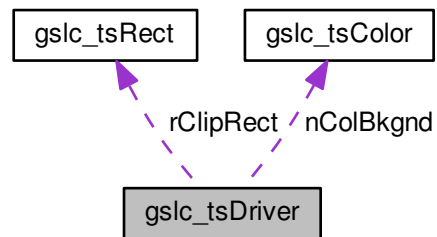
The documentation for this struct was generated from the following file:

- `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:

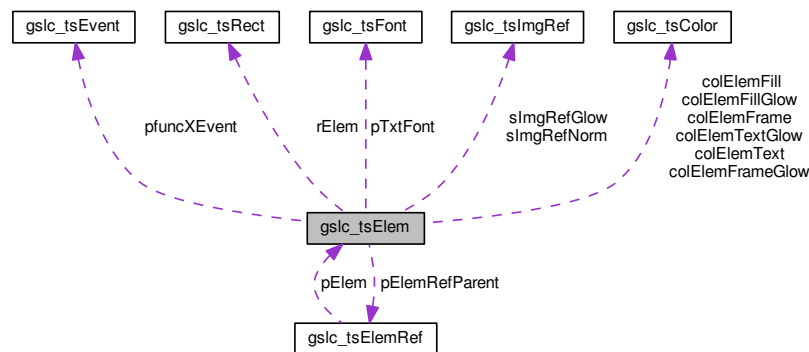
- [src/GUIslice_drv_adagfx.h](#)
- [src/GUIslice_drv_m5stack.h](#)
- [src/GUIslice_drv_tft_espi.h](#)

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_tsTextFlags 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.
- [uint8_t nTxtMargin](#)
Margin of overlay text within rect region.
- [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:

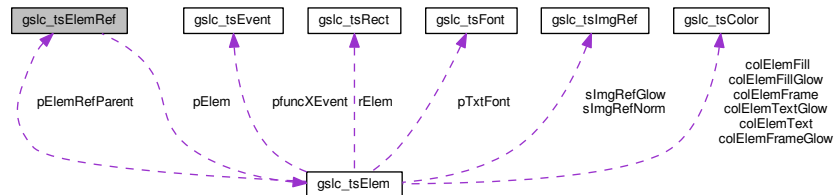
- [src/GUIslice.h](#)

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.
- `const void *` [pvFont](#)
Void ptr to the font reference (type defined by driver)
- [uint16_t nSize](#)
Font size.

8.8.1 Detailed Description

Font reference structure.

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

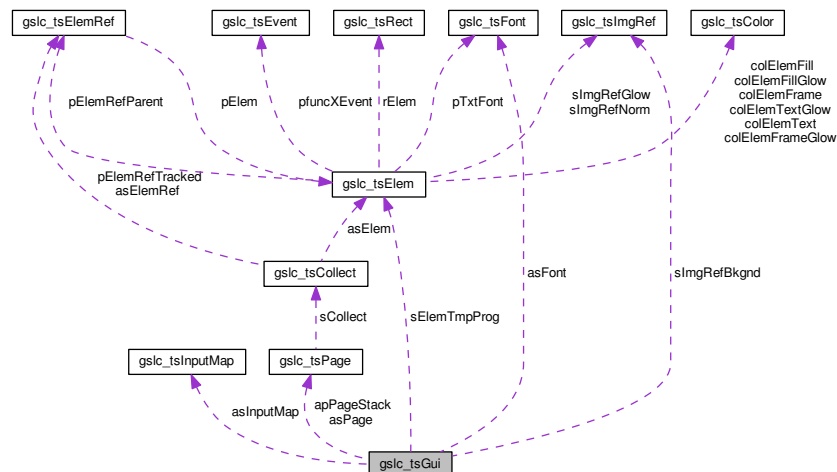
- [src/GUISlice.h](#)

8.9 gslc_tsGui Struct Reference

GUI structure.

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

Collaboration diagram for gslc_tsGui:



Data Fields

- `uint16_t nDispW`
Width of the display (pixels)
- `uint16_t nDispH`
Height of the display (pixels)
- `uint16_t nDisp0W`
Width of the display (pixels) in native orientation.
- `uint16_t nDisp0H`
Height of the display (pixels) in native orientation.
- `uint8_t nDispDepth`
Bit depth of display (bits per pixel)
- `uint8_t nRotation`
Adafruit GFX Rotation of display.
- `uint8_t nTouchRotation`

- Touchscreen rotation offset vs display.*

 - uint8_t [nSwapXY](#)

Adafruit GFX Touch Swap x and y axes.
- uint8_t [nFlipX](#)

Adafruit GFX Touch Flip x axis.
- uint8_t [nFlipY](#)

Adafruit GFX Touch Flip y 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.
- [gslc_tsElem](#) [sElemTmpProg](#)

Temporary element for Flash compatibility.
- [gslc_telnitStat](#) [elnitStatTouch](#)

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](#) [slmgRefBkgnd](#)

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.
- [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_tsImgRefFlags](#) [elmFlags](#)
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:

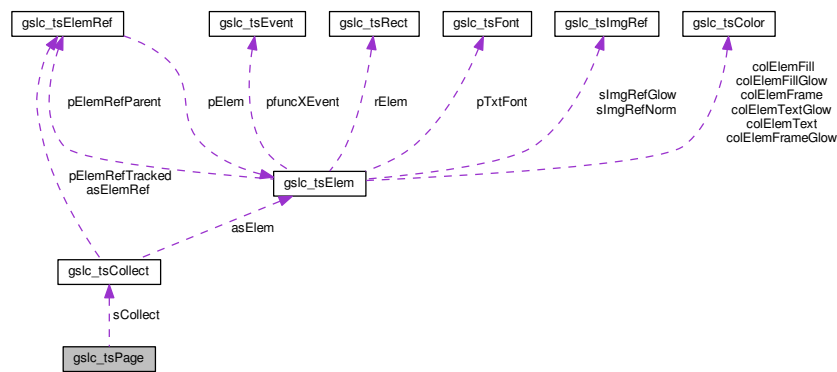
- [src/GUIslice.h](#)

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.

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:

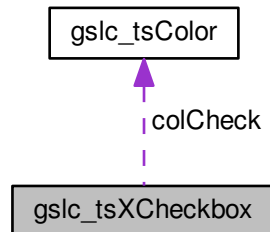
- [src/GUIslice.h](#)

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:

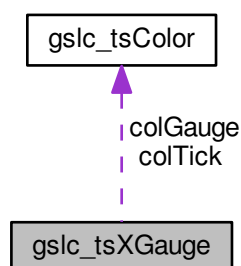
- [src/elem/XCheckbox.h](#)

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:

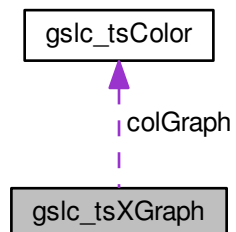
- [src/elem/XGauge.h](#)

8.17 gslc_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.

8.17.1 Detailed Description

Extended data for Graph element.

8.17.2 Field Documentation

8.17.2.1 `bool gslc_tsXGraph::bScrollEn`

Enable for scrollbar.

8.17.2.2 `gslc_tsColor gslc_tsXGraph::colGraph`

Color of the graph.

8.17.2.3 gslc_tXGraphStyle gslc_tsXGraph::eStyle

Style of the graph.

8.17.2.4 uint16_t gslc_tsXGraph::nBufCnt

Number of points in buffer.

8.17.2.5 uint16_t gslc_tsXGraph::nBufMax

Maximum number of points in buffer.

8.17.2.6 uint8_t gslc_tsXGraph::nMargin

Margin for graph area within element rect.

8.17.2.7 uint16_t gslc_tsXGraph::nPlotIndMax

Number of data points to show in window.

8.17.2.8 uint16_t gslc_tsXGraph::nPlotIndStart

First row of current window.

8.17.2.9 int16_t gslc_tsXGraph::nPlotValMax

Visible window maximum value.

8.17.2.10 int16_t gslc_tsXGraph::nPlotValMin

Visible window minimum value.

8.17.2.11 uint16_t gslc_tsXGraph::nScrollPos

Current scrollbar position.

8.17.2.12 uint16_t gslc_tsXGraph::nWndHeight

Visible window height.

8.17.2.13 uint16_t gslc_tsXGraph::nWndWidth

Visible window width.

8.17.2.14 int16_t* gslc_tsXGraph::pBuf

Ptr to the data buffer (circular buffer)

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

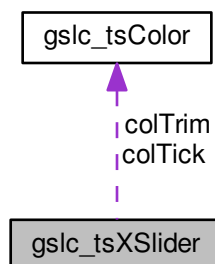
- [src/elem/XGraph.h](#)

8.18 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.*

8.18.1 Detailed Description

Extended data for Slider element.

8.18.2 Field Documentation

8.18.2.1 bool [gslc_tsXSlider::bTrim](#)

Style: show a trim color.

8.18.2.2 bool [gslc_tsXSlider::bVert](#)

Orientation: true if vertical, else horizontal.

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

Style: color of ticks.

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

Style: color of trim.

8.18.2.5 int16_t [gslc_tsXSlider::nPos](#)

Current position value of the slider.

8.18.2.6 int16_t [gslc_tsXSlider::nPosMax](#)

Maximum position value of the slider.

8.18.2.7 int16_t gslc_tsXSlider::nPosMin

Minimum position value of the slider.

8.18.2.8 int16_t gslc_tsXSlider::nThumbSz

Size of the thumb control.

8.18.2.9 uint16_t gslc_tsXSlider::nTickDiv

Style: number of tickmark divisions (0 for none)

8.18.2.10 int16_t gslc_tsXSlider::nTickLen

Style: length of tickmarks.

8.18.2.11 GSLC_CB_XSLIDER_POS gslc_tsXSlider::pfuncXPos

Callback func ptr for position update.

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

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

8.19 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)

8.19.1 Detailed Description

Extended data for Textbox element.

8.19.2 Field Documentation

8.19.2.1 bool `gslc_tsXTextbox::bScrollEn`

Enable for scrollbar.

8.19.2.2 bool `gslc_tsXTextbox::bWrapEn`

Enable for line wrapping.

8.19.2.3 uint16_t `gslc_tsXTextbox::nBufCols`

Number of columns in buffer.

8.19.2.4 uint8_t `gslc_tsXTextbox::nBufPosX`

Buffer X position.

8.19.2.5 uint8_t gslc_tsXTextbox::nBufPosY

Buffer Y position.

8.19.2.6 uint16_t gslc_tsXTextbox::nBufRows

Number of rows in buffer.

8.19.2.7 uint8_t gslc_tsXTextbox::nChSizeX

Width of characters (pixels)

8.19.2.8 uint8_t gslc_tsXTextbox::nChSizeY

Height of characters (pixels)

8.19.2.9 uint8_t gslc_tsXTextbox::nCurPosX

Cursor X position.

8.19.2.10 uint8_t gslc_tsXTextbox::nCurPosY

Cursor Y position.

8.19.2.11 int8_t gslc_tsXTextbox::nMarginX

Margin for text area within element rect (X)

8.19.2.12 int8_t gslc_tsXTextbox::nMarginY

Margin for text area within element rect (Y)

8.19.2.13 int16_t gslc_tsXTextbox::nRedrawRow

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

8.19.2.14 uint16_t gslc_tsXTextbox::nScrollPos

Current scrollbar position.

8.19.2.15 `uint8_t gslc_tsXTextbox::nWndCols`

Window X size.

8.19.2.16 `uint8_t gslc_tsXTextbox::nWndRows`

Window Y size.

8.19.2.17 `uint8_t gslc_tsXTextbox::nWndRowStart`

First row of current window.

8.19.2.18 `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](#)

8.20 THPoint Class Reference

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

Public Member Functions

- [THPoint](#) (void)
- [THPoint](#) (int16_t x, int16_t y, int16_t z)
- bool [operator==](#) (THPoint)
- bool [operator!=](#) (THPoint)

Data Fields

- int16_t x
- int16_t y
- int16_t z

8.20.1 Constructor & Destructor Documentation

8.20.1.1 THPoint::THPoint (void)

8.20.1.2 THPoint::THPoint (int16_t x, int16_t y, int16_t z)

8.20.2 Member Function Documentation

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

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

8.20.3 Field Documentation

8.20.3.1 int16_t THPoint::x

8.20.3.2 int16_t THPoint::y

8.20.3.3 int16_t THPoint::z

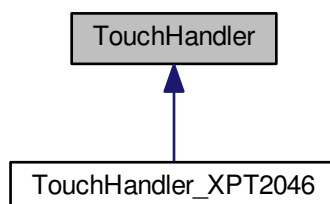
The documentation for this class was generated from the following files:

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

8.21 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)

8.21.1 Constructor & Destructor Documentation

8.21.1.1 [TouchHandler::TouchHandler](#) () `[inline]`

8.21.2 Member Function Documentation

8.21.2.1 void [TouchHandler::begin](#) (void) `[virtual]`

Reimplemented in [TouchHandler_XPT2046](#).

8.21.2.2 [THPoint](#) [TouchHandler::getPoint](#) (void) `[virtual]`

Reimplemented in [TouchHandler_XPT2046](#).

8.21.2.3 [THPoint](#) [TouchHandler::scale](#) ([THPoint](#) pIn)

8.21.2.4 void [TouchHandler::setCalibration](#) (uint16_t ts_xMin, uint16_t ts_xMax, uint16_t ts_yMin, uint16_t ts_yMax)

8.21.2.5 void [TouchHandler::setSize](#) (uint16_t _disp_xSize, uint16_t _disp_ySize)

8.21.2.6 void [TouchHandler::setSwapFlip](#) (bool _swapXY, bool _flipX, bool _flipY)

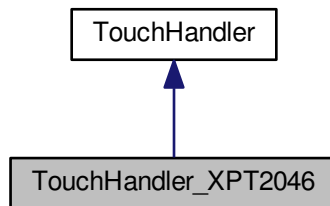
The documentation for this class was generated from the following files:

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

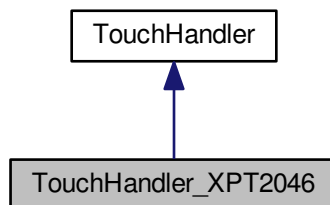
8.22 TouchHandler_XPT2046 Class Reference

```
#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](#)

8.22.1 Constructor & Destructor Documentation

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

8.22.2 Member Function Documentation

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

Reimplemented from [TouchHandler](#).

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

Reimplemented from [TouchHandler](#).

8.22.3 Field Documentation

8.22.3.1 `SPIClass TouchHandler_XPT2046::spi`

8.22.3.2 `XPT2046_touch TouchHandler_XPT2046::touchDriver`

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

- [src/GUIslice_th_XPT2046.h](#)

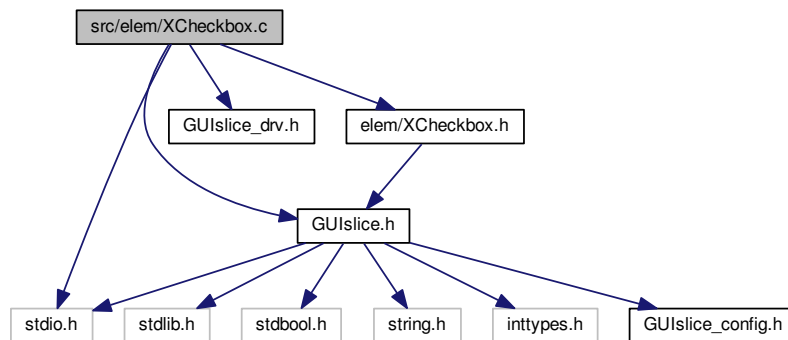
Chapter 9

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 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.
- `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_XCHECKBOX](#) pfuncCb)

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

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

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

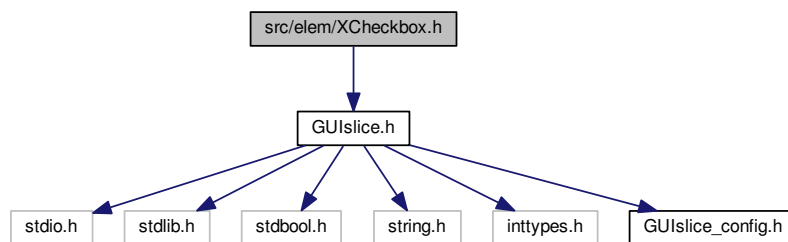
9.2.2.1 `const char ERRSTR_NULL`

9.2.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

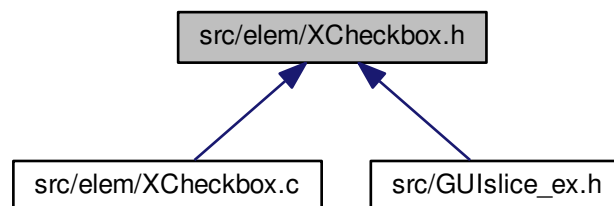
9.3 src/elem/XCheckbox.h File Reference

```
#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 * pvGui, void * pElemRef, 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>pElemRef</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

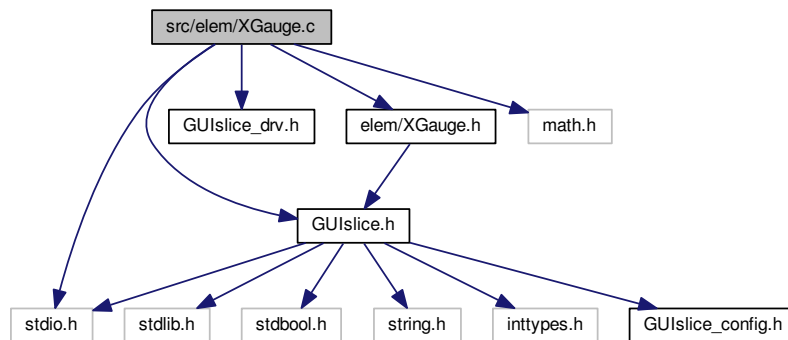
Returns

true if success, false otherwise

9.4 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

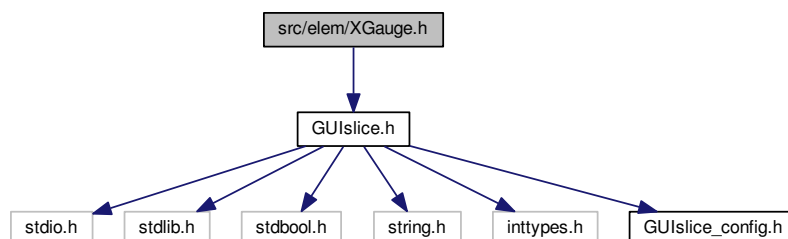
9.4.2.1 const char GSLC_PMEM_ERRSTR_NULL[]

9.4.2.2 const char GSLC_PMEM_ERRSTR_PXD_NULL[]

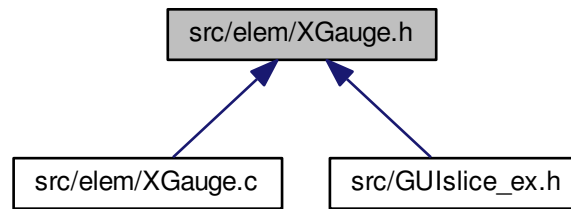
9.5 src/elem/XGauge.h File Reference

```
#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	<code>pGui</code>	Pointer to GUI
in	<code>nElemId</code>	Unique element ID to assign
in	<code>nPage</code>	Page ID to attach element to
in	<code>nX</code>	X coordinate of element
in	<code>nY</code>	Y coordinate of element
in	<code>nW</code>	Width of element
in	<code>nH</code>	Height of element
in	<code>nMin_</code>	Minimum value of gauge for nVal comparison
in	<code>nMax_</code>	Maximum value of gauge for nVal comparison
in	<code>nVal_</code>	Starting value of gauge
in	<code>colFrame_</code>	Color for the gauge frame
in	<code>colFill_</code>	Color for the gauge background fill
in	<code>colGauge_</code>	Color for the gauge indicator
in	<code>bVert_</code>	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

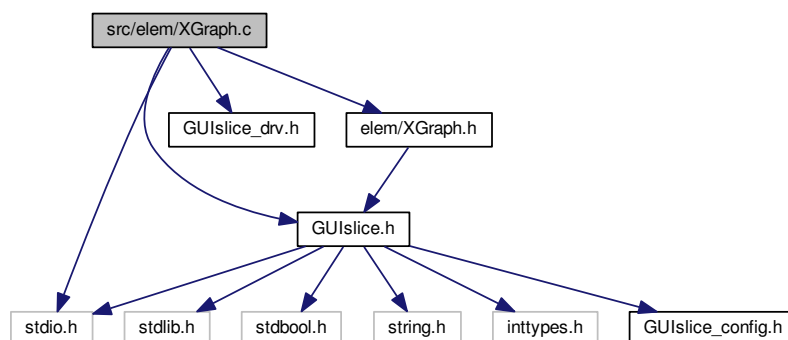
in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>nVal</i>	New value to show in gauge

Returns

none

9.6 src/elem/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_tsX↵Graph](#) * *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 *pvGui, void *pvElemRef, [gslc_teRedrawType](#) eRedraw)

Draw a Graph element on the screen.

Variables

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

9.6.1 Function Documentation

9.6.1.1 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

9.6.1.2 [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

Parameters

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

9.6.1.3 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 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

9.6.1.4 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


```
9.6.1.5 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

```
9.6.1.6 void gslc_ElemXGraphSetStyle ( gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_teXGraphStyle 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

9.6.2 Variable Documentation

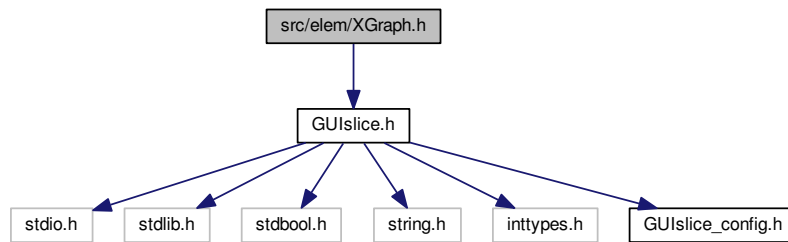
```
9.6.2.1 const char GSLC_PMEM ERRSTR_NULL[]
```

```
9.6.2.2 const char GSLC_PMEM ERRSTR_PXD_NULL[]
```

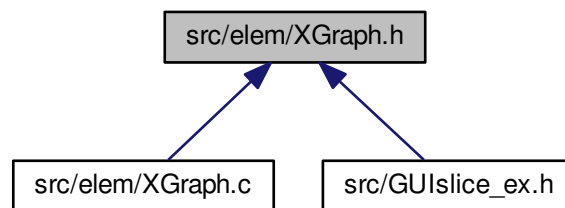
9.7 src/elem/XGraph.h File Reference

```
#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`, `GSLCX_GRAPH_STYLE_FILL` }
Gauge drawing style.

Functions

- [gslc_tsElemRef](#) * [gslc_ElemXGraphCreate](#) ([gslc_tsGui](#) *pGui, int16_t nElemId, int16_t nPage, [gslc_tsX↔Graph](#) *pXData, [gslc_tsRect](#) rElem, int16_t nFontId, int16_t *pBuf, uint16_t nBufRows, [gslc_tsColor](#) col↔Graph)
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 n↔YMax)
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.

9.7.1 Macro Definition Documentation

9.7.1.1 #define GSLC_TYPEX_GRAPH

9.7.2 Enumeration Type Documentation

9.7.2.1 enum gslc_teXGraphStyle

Gauge drawing style.

Enumerator

GSLCX_GRAPH_STYLE_DOT Dot.
GSLCX_GRAPH_STYLE_LINE Line.
GSLCX_GRAPH_STYLE_FILL Filled.

9.7.3 Function Documentation

9.7.3.1 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

9.7.3.2 `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

9.7.3.3 `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

9.7.3.4 `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

9.7.3.5 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

9.7.3.6 void gslc_ElemXGraphSetStyle (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, gslc_teXGraphStyle *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

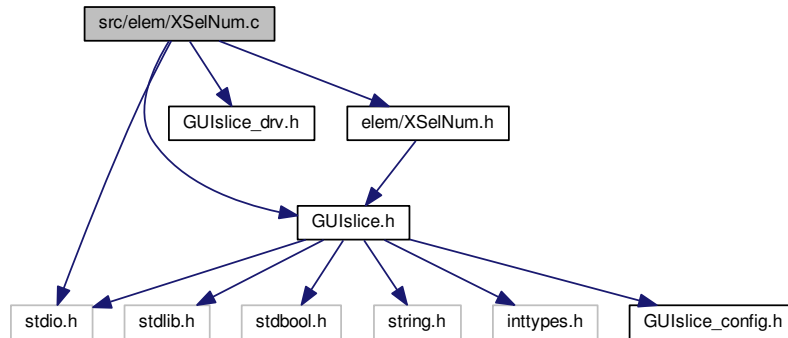
none

9.8 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:
```



Variables

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

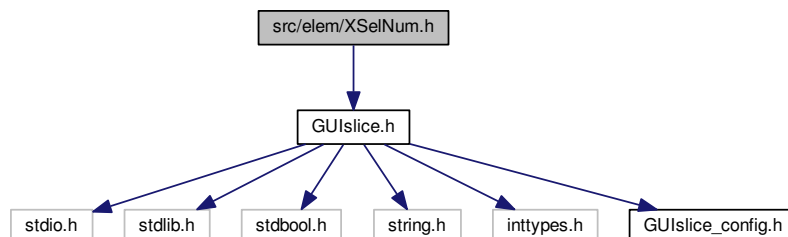
9.8.1 Variable Documentation

9.8.1.1 const char [GSLC_PMEM_ERRSTR_NULL](#) []

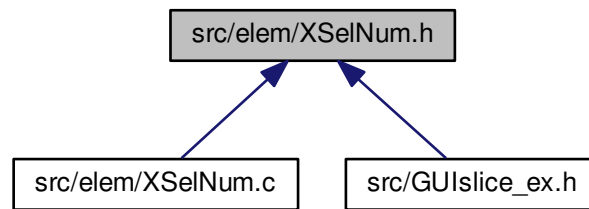
9.8.1.2 const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.9 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:



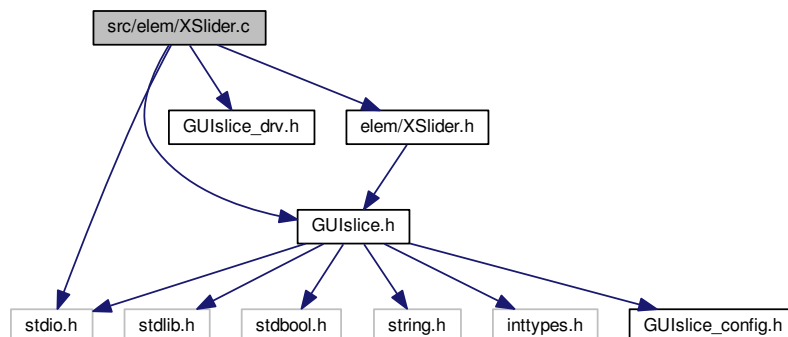
9.10 src/elem/XSlider.c File Reference

```

#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](#) []

9.10.1 Function Documentation

- 9.10.1.1 [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

- 9.10.1.2 [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 <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

9.10.1.3 `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

9.10.1.4 `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

9.10.1.5 `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

9.10.1.6 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

9.10.1.7 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 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

9.10.2 Variable Documentation

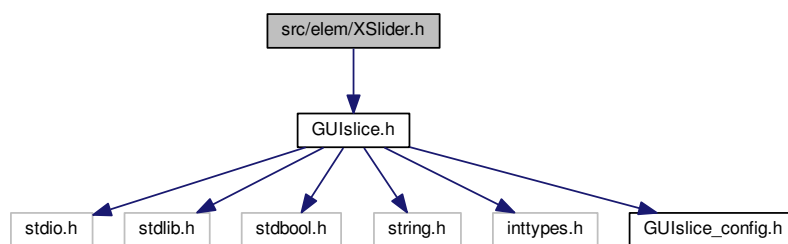
9.10.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.10.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

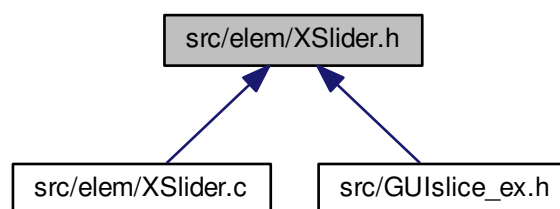
9.11 src/elem/XSlider.h File Reference

```
#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_tsX↔Slider *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 col↔Trim, 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.

9.11.1 Macro Definition Documentation

- 9.11.1.1 `#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

Parameters

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)
in	<i>colFrame</i> ↔ —	Color of the element frame
in	<i>colFill_</i>	Color of the element fill

Returns

none

9.11.1.2 #define GSLC_TYPEX_SLIDER

9.11.2 Typedef Documentation

9.11.2.1 typedef bool(* GSLC_CB_XSLIDER_POS)(void *pvGui, void *pvElem, int16_t nPos)

Callback function for slider feedback.

9.11.3 Function Documentation

9.11.3.1 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

9.11.3.2 `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

9.11.3.3 `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

9.11.3.4 `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

9.11.3.5 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

9.11.3.6 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

9.11.3.7 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

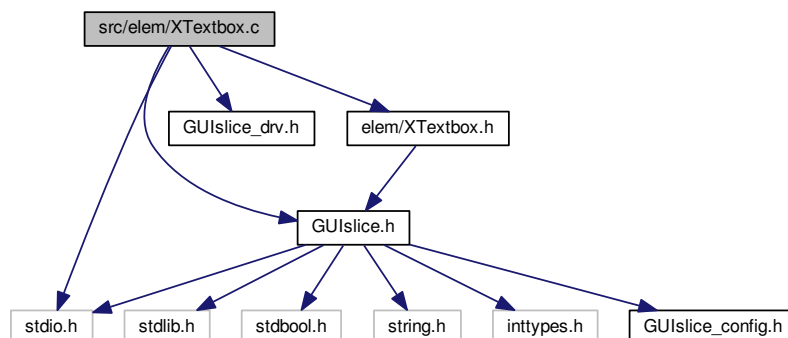
Returns

true if success, false otherwise

9.12 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](#) []

9.12.1 Function Documentation

9.12.1.1 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

9.12.1.2 void [gslc_ElemXTextboxBufAdd](#) ([gslc_tsGui](#) * *pGui*, [gslc_tsElemRef](#) * *pElemRef*, unsigned char *chNew*, bool *bAdvance*)

9.12.1.3 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

9.12.1.4 `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

9.12.1.5 `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 (<code>nBufRows*nBufCols</code>) 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

9.12.1.6 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 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

9.12.1.7 void gslc_ElemXTextboxLineWrAdv (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*)

9.12.1.8 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

9.12.1.9 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

9.12.1.10 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

9.12.2 Variable Documentation

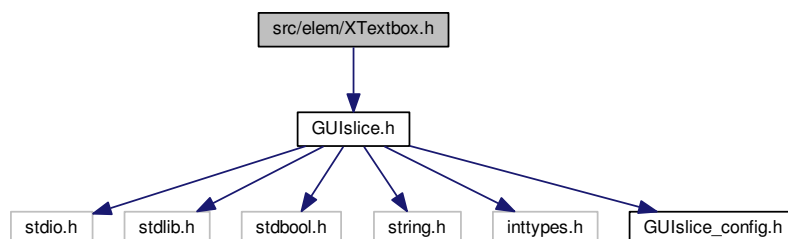
9.12.2.1 const char GSLC_PMEM ERRSTR_NULL[]

9.12.2.2 const char GSLC_PMEM ERRSTR_PXD_NULL[]

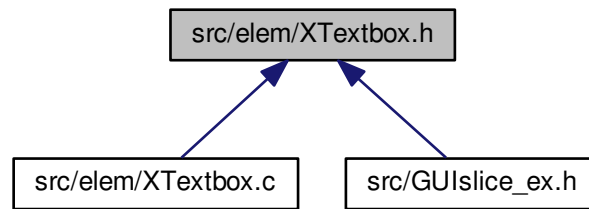
9.13 src/elem/XTextbox.h File Reference

```
#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_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.
- 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 nScrollMax)
Set the textbox scroll position (nScrollPos) as a fraction of nScrollMax.

9.13.1 Macro Definition Documentation

9.13.1.1 `#define GSLC_TEXPEX_TEXTBOX`

9.13.1.2 `#define GSLC_XTEXTBOX_CODE_COL_RESET`

9.13.1.3 `#define GSLC_XTEXTBOX_CODE_COL_SET`

Definitions for textbox special inline codes.

9.13.1.4 `#define XTEXTBOX_REDRAW_ALL`

9.13.1.5 `#define XTEXTBOX_REDRAW_NONE`

9.13.2 Function Documentation

9.13.2.1 `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

9.13.2.2 `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

9.13.2.3 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

9.13.2.4 `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 (<code>nBufRows*nBufCols</code>) 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

9.13.2.5 `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

9.13.2.6 `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

9.13.2.7 `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

9.13.2.8 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

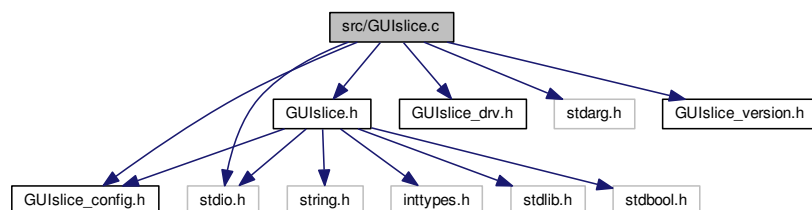
Returns

none

9.14 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_DEBUG_PRINT_NORM`, `GSLC_DEBUG_PRINT_TOKEN`, `GSLC_DEBUG_PRINT_UINT16`, `GSLC_DEBUG_PRINT_STR`,
`GSLC_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.
- 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_telInputRawEvent](#) eInputEvent, int16_t nInputVal, [gslc_teAction](#) eAction, int16_t nActionVal)
- bool [gslc_InputMapLookup](#) ([gslc_tsGui](#) *pGui, [gslc_telInputRawEvent](#) 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_DrawFillRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
- Draw a filled 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_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.*

 - bool [gslc_FontAdd](#) ([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_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)

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

Get an Element ID from an element structure.

- 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)

- `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.
- bool [gslc_ElemDrawByRef](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teRedrawType](#) 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_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_ElemSetTxtStr](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, const char *pStr)
Update the text string associated with an Element ID.
- 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.*

 - 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_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.*

 - 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_tsImgRef sImgRefSel)`
- Set an element to use a bitmap image.*
- `bool gslc_SetBgndImage (gslc_tsGui *pGui, gslc_tsImgRef sImgRef)`
- Configure the background to use a bitmap image.*
- `bool gslc_SetBgndColor (gslc_tsGui *pGui, gslc_tsColor nCol)`
- Configure the background to use a solid color.*
- `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 eTouch, 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.

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](#) []

9.14.1 Enumeration Type Documentation

9.14.1.1 enum [gslc_teDebugPrintState](#)

Enumerator

[GSLC_DEBUG_PRINT_NORM](#)
[GSLC_DEBUG_PRINT_TOKEN](#)
[GSLC_DEBUG_PRINT_UINT16](#)
[GSLC_DEBUG_PRINT_STR](#)
[GSLC_DEBUG_PRINT_STR_P](#)

9.14.2 Function Documentation

9.14.2.1 void [gslc_OrderCoord](#) (int16_t * [pnX0](#), int16_t * [pnY0](#), int16_t * [pnX1](#), int16_t * [pnY1](#))

9.14.2.2 void [gslc_SwapCoords](#) (int16_t * [pnXa](#), int16_t * [pnYa](#), int16_t * [pnXb](#), int16_t * [pnYb](#))

9.14.3 Variable Documentation

9.14.3.1 const char [ERRSTR_NULL](#) []

9.14.3.2 const char [GSLC_PMEM_ERRSTR_PXD_NULL](#) []

9.14.3.3 [GSLC_CB_DEBUG_OUT](#) [g_pfDebugOut](#)

Global debug output function.

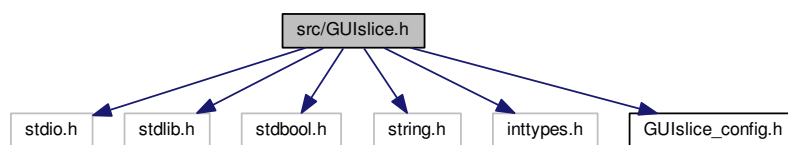
- The user assigns this function via [gslc_InitDebug\(\)](#)

9.14.3.4 uint16_t m_nLUTSinF0X16

9.15 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_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_DEBUG_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_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 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_telInputRawEvent` {
`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_FOCUS_NEXT`,
`GSLC_ACTION_SELECT`, `GSLC_ACTION_SET_REL`, `GSLC_ACTION_SET_ABS`, `GSLC_ACTION_DEBUG` }
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` }
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` }
Processed event from input raw events and actions.
- enum `gslc_telInitStat` { `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`,
`GSLC_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_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_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.
- 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_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.
- 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.
- 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_DrawFillRect](#) ([gslc_tsGui](#) *pGui, [gslc_tsRect](#) rRect, [gslc_tsColor](#) nCol)
Draw a filled 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.
- bool [gslc_FontAdd](#) ([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.
- 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](#) 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.
- 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_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_ElemSetTxtStr](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, const char *pStr)
Update the text string associated with an Element ID.
- 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_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.
- 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 b↵ OnlyClickEn)

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_te↵ Action](#) 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_ts↵ Rect](#) 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_teElem↵ RefFlags](#) 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 n↵ FlagVal)

Set the flags associated with an element reference.
- [gslc_tsElem](#) * [gslc_GetElemFromRef](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef)
- void [gslc_ElemSetImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_tslmgRef](#) sImgRef, [gslc_ts↵ ImgRef](#) 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.*

 - 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.*

 - 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](#) *p↔ Elem, [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)
- void [gslc_CollectSetParent](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsElemRef](#) *pElemRefParent)

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

 - 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.
- void [gslc_CollectInput](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsEventTouch](#) *pEventTouch)

Handle direct input events within the element collection.
- 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)
- 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.

Variables

- [GSLC_CB_DEBUG_OUT](#) [g_pfDebugOut](#)
- Global debug output function.*

9.15.1 Macro Definition Documentation

9.15.1.1 #define GSLC_2PI

9.15.1.2 #define GSLC_ALIGN_BOT_LEFT

Align to bottom-left.

9.15.1.3 #define GSLC_ALIGN_BOT_MID

Align to middle of bottom.

9.15.1.4 `#define GSLC_ALIGN_BOT_RIGHT`

Align to bottom-right.

9.15.1.5 `#define GSLC_ALIGN_MID_LEFT`

Align to middle of left side.

9.15.1.6 `#define GSLC_ALIGN_MID_MID`

Align to center.

9.15.1.7 `#define GSLC_ALIGN_MID_RIGHT`

Align to middle of right side.

9.15.1.8 `#define GSLC_ALIGN_TOP_LEFT`

Align to top-left.

9.15.1.9 `#define GSLC_ALIGN_TOP_MID`

Align to middle of top.

9.15.1.10 `#define GSLC_ALIGN_TOP_RIGHT`

Align to top-right.

9.15.1.11 `#define GSLC_ALIGNH_LEFT`

Horizontal align to left.

9.15.1.12 `#define GSLC_ALIGNH_MID`

Horizontal align to middle.

9.15.1.13 `#define GSLC_ALIGNH_RIGHT`

Horizontal align to right.

9.15.1.14 `#define GSLC_ALIGNV_BOT`

Vertical align to bottom.

9.15.1.15 `#define GSLC_ALIGNV_MID`

Vertical align to middle.

9.15.1.16 `#define GSLC_ALIGNV_TOP`

Element text alignment.

Vertical align to top

9.15.1.17 `#define GSLC_COL_BLACK`

Black.

9.15.1.18 `#define GSLC_COL_BLUE`

Blue.

9.15.1.19 `#define GSLC_COL_BLUE_DK1`

Blue (dark1)

9.15.1.20 `#define GSLC_COL_BLUE_DK2`

Blue (dark2)

9.15.1.21 `#define GSLC_COL_BLUE_DK3`

Blue (dark3)

9.15.1.22 `#define GSLC_COL_BLUE_DK4`

Blue (dark4)

9.15.1.23 `#define GSLC_COL_BLUE_LT1`

Blue (light1)

9.15.1.24 `#define GSLC_COL_BLUE_LT2`

Blue (light2)

9.15.1.25 `#define GSLC_COL_BLUE_LT3`

Blue (light3)

9.15.1.26 `#define GSLC_COL_BLUE_LT4`

Blue (light4)

9.15.1.27 `#define GSLC_COL_BROWN`

Brown.

9.15.1.28 `#define GSLC_COL_CYAN`

Cyan.

9.15.1.29 `#define GSLC_COL_GRAY`

Gray.

9.15.1.30 `#define GSLC_COL_GRAY_DK1`

Gray (dark)

9.15.1.31 `#define GSLC_COL_GRAY_DK2`

Gray (dark)

9.15.1.32 `#define GSLC_COL_GRAY_DK3`

Gray (dark)

9.15.1.33 `#define GSLC_COL_GRAY_LT1`

Gray (light1)

9.15.1.34 `#define GSLC_COL_GRAY_LT2`

Gray (light2)

9.15.1.35 `#define GSLC_COL_GRAY_LT3`

Gray (light3)

9.15.1.36 `#define GSLC_COL_GREEN`

Green.

9.15.1.37 `#define GSLC_COL_GREEN_DK1`

Green (dark1)

9.15.1.38 `#define GSLC_COL_GREEN_DK2`

Green (dark2)

9.15.1.39 `#define GSLC_COL_GREEN_DK3`

Green (dark3)

9.15.1.40 `#define GSLC_COL_GREEN_DK4`

Green (dark4)

9.15.1.41 `#define GSLC_COL_GREEN_LT1`

Green (light1)

9.15.1.42 `#define GSLC_COL_GREEN_LT2`

Green (light2)

9.15.1.43 `#define GSLC_COL_GREEN_LT3`

Green (light3)

9.15.1.44 `#define GSLC_COL_GREEN_LT4`

Green (light4)

9.15.1.45 `#define GSLC_COL_MAGENTA`

Magenta.

9.15.1.46 `#define GSLC_COL_ORANGE`

Orange.

9.15.1.47 `#define GSLC_COL_PURPLE`

Purple.

9.15.1.48 `#define GSLC_COL_RED`

Red.

9.15.1.49 `#define GSLC_COL_RED_DK1`

Red (dark1)

9.15.1.50 `#define GSLC_COL_RED_DK2`

Red (dark2)

9.15.1.51 `#define GSLC_COL_RED_DK3`

Red (dark3)

9.15.1.52 `#define GSLC_COL_RED_DK4`

Basic color definition.

Red (dark4)

9.15.1.53 `#define GSLC_COL_RED_LT1`

Red (light1)

9.15.1.54 `#define GSLC_COL_RED_LT2`

Red (light2)

9.15.1.55 `#define GSLC_COL_RED_LT3`

Red (light3)

9.15.1.56 `#define GSLC_COL_RED_LT4`

Red (light4)

9.15.1.57 `#define GSLC_COL_TEAL`

Teal.

9.15.1.58 `#define GSLC_COL_WHITE`

White.

9.15.1.59 `#define GSLC_COL_YELLOW`

Yellow.

9.15.1.60 `#define GSLC_COL_YELLOW_DK`

Yellow (dark)

9.15.1.61 `#define GSLC_COLMONO_BLACK`

Black.

9.15.1.62 `#define GSLC_COLMONO_WHITE`

White.

9.15.1.63 `#define GSLC_ELEM_FEA_CLICK_EN`

Element accepts touch presses.

9.15.1.64 `#define GSLC_ELEM_FEA_FILL_EN`

Element is drawn with a fill.

9.15.1.65 `#define GSLC_ELEM_FEA_FRAME_EN`

Element is drawn with a frame.

9.15.1.66 `#define GSLC_ELEM_FEA_GLOW_EN`

Element supports glowing state.

9.15.1.67 `#define GSLC_ELEM_FEA_NONE`

Element default (no features set))

9.15.1.68 `#define GSLC_ELEM_FEA_VALID`

Element features type.

Element record is valid

9.15.1.69 `#define GSLC_ELEMREF_DEFAULT`

Define the default element reference flags for new elements.

9.15.1.70 `#define GSLC_PMEM`

9.15.2 Typedef Documentation

9.15.2.1 `typedef int16_t(* GSLC_CB_DEBUG_OUT)(char ch)`

9.15.2.2 `typedef bool(* GSLC_CB_DRAW)(void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`

Callback function for element drawing.

9.15.2.3 `typedef bool(* GSLC_CB_EVENT)(void *pvGui, gslc_tsEvent sEvent)`

Callback function for element drawing.

9.15.2.4 `typedef bool(* GSLC_CB_PIN_POLL)(void *pvGui, int16_t *pnPinInd, int16_t *pnPinVal)`

Callback function for pin polling.

9.15.2.5 `typedef bool(* GSLC_CB_TICK)(void *pvGui, void *pvElemRef)`

Callback function for element tick.

9.15.2.6 `typedef bool(* GSLC_CB_TOUCH)(void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

Callback function for element touch tracking.

9.15.2.7 `typedef struct gslc_tsColor gslc_tsColor`

Color structure. Defines RGB triplet.

9.15.2.8 `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.

9.15.2.9 `typedef struct gslc_tsEvent gslc_tsEvent`

Event structure.

9.15.2.10 `typedef struct gslc_tsEventTouch gslc_tsEventTouch`

Structure used to pass touch data through event.

9.15.2.11 `typedef struct gslc_tsPt gslc_tsPt`

Define point coordinates.

9.15.2.12 typedef struct **gslc_tsRect** **gslc_tsRect**

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

9.15.3 Enumeration Type Documentation

9.15.3.1 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.

9.15.3.2 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)

9.15.3.3 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.

9.15.3.4 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.

9.15.3.5 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.

9.15.3.6 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.

9.15.3.7 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.

9.15.3.8 enum `gslc_teFontRefType`

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_FONTRF_FNAME Font reference is a filename (full path)
GSLC_FONTRF_PTR Font reference is a pointer to a font structure.

9.15.3.9 enum `gslc_teGroupId`

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.

9.15.3.10 enum gslc_telmgRefFlags

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_SRC Mask for Source flags.
GSLC_IMGREF_FMT Mask for Format flags.

9.15.3.11 enum gslc_telnitStat

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.

9.15.3.12 enum gslc_telInputRawEvent

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)

9.15.3.13 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.

9.15.3.14 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)

9.15.3.15 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.

9.15.3.16 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.

9.15.3.17 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.

9.15.3.18 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

9.15.3.19 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.

9.15.4 Variable Documentation

9.15.4.1 GSLC_CB_DEBUG_OUT g_pfDebugOut

Global debug output function.

- The user assigns this function via [gslc_InitDebug\(\)](#)

- `#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`

9.17.1 Macro Definition Documentation

9.17.1.1 `#define ADAGFX_PIN_CLK`

9.17.1.2 `#define ADAGFX_PIN_CS`

9.17.1.3 `#define ADAGFX_PIN_DC`

9.17.1.4 `#define ADAGFX_PIN_MISO`

9.17.1.5 `#define ADAGFX_PIN_MOSI`

9.17.1.6 `#define ADAGFX_PIN_RD`

9.17.1.7 `#define ADAGFX_PIN_RST`

9.17.1.8 `#define ADAGFX_PIN_SDCS`

9.17.1.9 `#define ADAGFX_PIN_WR`

9.17.1.10 `#define ADAGFX_SPI_HW`

9.17.1.11 `#define ADATOUCH_FLIP_X`

9.17.1.12 `#define ADATOUCH_FLIP_Y`

9.17.1.13 `#define ADATOUCH_I2C_ADDR`

9.17.1.14 `#define ADATOUCH_I2C_HW`

9.17.1.15 `#define ADATOUCH_PIN_CS`

9.17.1.16 `#define ADATOUCH_SPI_HW`

9.17.1.17 `#define ADATOUCH_SPI_SW`

- 9.17.1.18 `#define ADATOUCH_SWAP_XY`
- 9.17.1.19 `#define ADATOUCH_X_MAX`
- 9.17.1.20 `#define ADATOUCH_X_MIN`
- 9.17.1.21 `#define ADATOUCH_Y_MAX`
- 9.17.1.22 `#define ADATOUCH_Y_MIN`
- 9.17.1.23 `#define DEBUG_ERR`
- 9.17.1.24 `#define DRV_DISP_ADAGFX`
- 9.17.1.25 `#define DRV_DISP_ADAGFX_ILI9341`
- 9.17.1.26 `#define DRV_TOUCH_ADA_STMPE610`
- 9.17.1.27 `#define GSLC_BMP_TRANS_EN`
- 9.17.1.28 `#define GSLC_BMP_TRANS_RGB`
- 9.17.1.29 `#define GSLC_CLIP_EN`
- 9.17.1.30 `#define GSLC_DEV_TOUCH`
- 9.17.1.31 `#define GSLC_FEATURE_COMPOUND`
- 9.17.1.32 `#define GSLC_FEATURE_INPUT`
- 9.17.1.33 `#define GSLC_FEATURE_XGAUGE_RADIAL`
- 9.17.1.34 `#define GSLC_FEATURE_XGAUGE_RAMP`
- 9.17.1.35 `#define GSLC_FEATURE_XTEXTBOX_EMBED`
- 9.17.1.36 `#define GSLC_LOCAL_STR`
- 9.17.1.37 `#define GSLC_LOCAL_STR_LEN`
- 9.17.1.38 `#define GSLC_ROTATE`
- 9.17.1.39 `#define GSLC_SD_BUFFPIXEL`
- 9.17.1.40 `#define GSLC_SD_EN`

9.17.1.41 `#define GSLC_TOUCH_MAX_EVT`

9.17.1.42 `#define GSLC_USE_FLOAT`

9.17.1.43 `#define GSLC_USE_PROGMEM`

9.17.1.44 `#define TOUCH_ROTATION_DATA`

9.17.1.45 `#define TOUCH_ROTATION_FLIPX(rotation)`

9.17.1.46 `#define TOUCH_ROTATION_FLIPY(rotation)`

9.17.1.47 `#define TOUCH_ROTATION_SWAPXY(rotation)`

9.18 `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`

9.18.1 Macro Definition Documentation

9.18.1.1 `#define ADATOUCH_FLIP_X`

9.18.1.2 `#define ADATOUCH_FLIP_Y`

9.18.1.3 `#define ADATOUCH_SWAP_XY`

9.18.1.4 `#define DEBUG_ERR`

9.18.1.5 `#define DRV_DISP_SDL1`

9.18.1.6 `#define DRV_SDL_FIX_START`

9.18.1.7 `#define DRV_SDL_MOUSE_SHOW`

9.18.1.8 `#define DRV_TOUCH_TSLIB`

9.18.1.9 `#define GSLC_BMP_TRANS_EN`

9.18.1.10 `#define GSLC_BMP_TRANS_RGB`

9.18.1.11 `#define GSLC_DEV_FB`

9.18.1.12 `#define GSLC_DEV_TOUCH`

9.18.1.13 `#define GSLC_DEV_VID_DRV`

9.18.1.14 `#define GSLC_FEATURE_COMPOUND`

9.18.1.15 `#define GSLC_FEATURE_INPUT`

9.18.1.16 `#define GSLC_FEATURE_XGAUGE_RADIAL`

9.18.1.17 `#define GSLC_FEATURE_XGAUGE_RAMP`

9.18.1.18 `#define GSLC_FEATURE_XTEXTBOX_EMBED`

9.18.1.19 `#define GSLC_LOCAL_STR`

9.18.1.20 `#define GSLC_LOCAL_STR_LEN`

9.18.1.21 `#define GSLC_TOUCH_MAX_EVT`

9.18.1.22 `#define GSLC_USE_FLOAT`

9.18.1.23 `#define GSLC_USE_PROGMEM`

9.19 src/GUISlice_drv.h File Reference

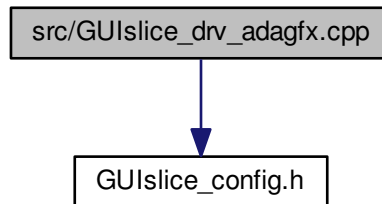
This graph shows which files directly or indirectly include this file:



9.20 src/GUISlice_drv_adagfx.cpp File Reference

```
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice_drv_adagfx.cpp:



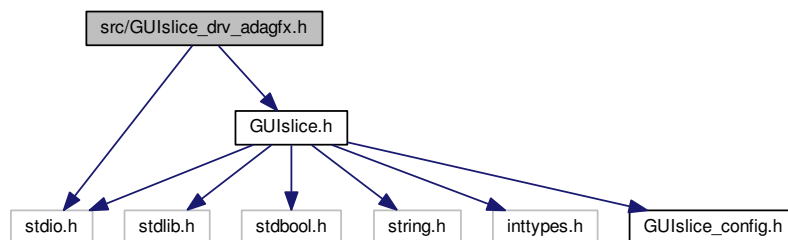
9.21 src/GUISlice_drv_adagfx.h File Reference

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_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_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_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_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)

9.21.1 Detailed Description

GUISlice library (driver layer for Adafruit-GFX)

9.21.2 Macro Definition Documentation

9.21.2.1 `#define DRV_HAS_DRAW_CIRCLE_FILL`

Support [gslc_DrvDrawFillCircle\(\)](#)

9.21.2.2 `#define DRV_HAS_DRAW_CIRCLE_FRAME`

Support [gslc_DrvDrawFrameCircle\(\)](#)

9.21.2.3 `#define DRV_HAS_DRAW_LINE`

Support [gslc_DrvDrawLine\(\)](#)

9.21.2.4 `#define DRV_HAS_DRAW_POINT`

Support [gslc_DrvDrawPoint\(\)](#)

9.21.2.5 `#define DRV_HAS_DRAW_POINTS`

Support [gslc_DrvDrawPoints\(\)](#)

9.21.2.6 `#define DRV_HAS_DRAW_RECT_FILL`

Support [gslc_DrvDrawFillRect\(\)](#)

9.21.2.7 `#define DRV_HAS_DRAW_RECT_FRAME`

Support [gslc_DrvDrawFrameRect\(\)](#)

9.21.2.8 `#define DRV_HAS_DRAW_TEXT`

Support [gslc_DrvDrawTxt\(\)](#)

9.21.2.9 `#define DRV_HAS_DRAW_TRI_FILL`

Support [gslc_DrvDrawFillTriangle\(\)](#)

9.21.2.10 `#define DRV_HAS_DRAW_TRI_FRAME`

Support [gslc_DrvDrawFrameTriangle\(\)](#)

9.21.2.11 `#define DRV_OVERRIDE_TXT_ALIGN`

Driver provides text alignment.

9.21.3 Function Documentation

9.21.3.1 `uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)`

9.21.3.2 `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

9.21.3.3 `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

9.21.3.4 `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

9.21.3.5 `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

9.21.3.6 `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

9.21.3.7 `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

9.21.3.8 `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

9.21.3.9 `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

9.21.3.10 `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

9.21.3.11 `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

9.21.3.12 `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

9.21.3.13 `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

9.21.3.14 `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

9.21.3.15 `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>nNumPt</i>	Number of points in array
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.21.3.16 `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

9.21.3.17 `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

9.21.3.18 void gslc_DrvFontsDestruct (gslc_tsGui * pGui)

Release all fonts defined in the GUI.

Parameters

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

Returns

none

9.21.3.19 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

9.21.3.20 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

9.21.3.21 bool gslc_DrvGetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress, gslc_tInputRawEvent * pInputEvent, 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

9.21.3.22 `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

9.21.3.23 `void gslc_DrvImageDestruct (void * pVImg)`

Release an image surface.

Parameters

in	<i>pVImg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.21.3.24 `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

9.21.3.25 `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

9.21.3.26 `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

9.21.3.27 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

9.21.3.28 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

9.21.3.29 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

9.21.3.30 `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

9.21.3.31 `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

9.21.3.32 `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

9.21.3.33 `bool gslc_DrvSetElemlImageGlow (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

9.21.3.34 `bool gslc_DrvSetElemlImageNorm (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

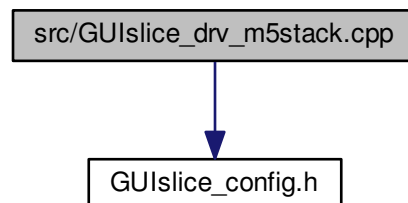
Returns

true if success, false if error

9.22 src/GUISlice_drv_m5stack.cpp File Reference

```
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice_drv_m5stack.cpp:



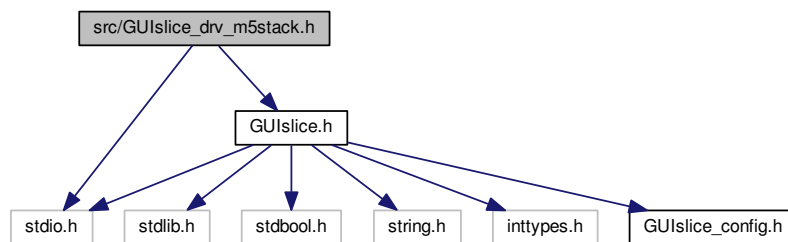
9.23 src/GUISlice_drv_m5stack.h File Reference

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_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_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_DrvLoadImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Load a bitmap (.bmp) and create a new image resource.*
- bool [gslc_DrvSetBgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tsImgRef](#) sImgRef)
Configure the background to use a bitmap image.
- bool [gslc_DrvSetBgndColor](#) ([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.
- 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.
- 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_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](#) 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)

Variables

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

9.23.1 Detailed Description

GUIslice library (driver layer for M5stack)

9.23.2 Macro Definition Documentation

9.23.2.1 #define DRV_HAS_DRAW_CIRCLE_FILL

Support [gslc_DrvDrawFillCircle\(\)](#)

9.23.2.2 #define DRV_HAS_DRAW_CIRCLE_FRAME

Support [gslc_DrvDrawFrameCircle\(\)](#)

9.23.2.3 #define DRV_HAS_DRAW_LINE

Support [gslc_DrvDrawLine\(\)](#)

9.23.2.4 #define DRV_HAS_DRAW_POINT

Support [gslc_DrvDrawPoint\(\)](#)

9.23.2.5 #define DRV_HAS_DRAW_POINTS

Support [gslc_DrvDrawPoints\(\)](#)

9.23.2.6 #define DRV_HAS_DRAW_RECT_FILL

Support [gslc_DrvDrawFillRect\(\)](#)

9.23.2.7 #define DRV_HAS_DRAW_RECT_FRAME

Support [gslc_DrvDrawFrameRect\(\)](#)

9.23.2.8 #define DRV_HAS_DRAW_TEXT

Support [gslc_DrvDrawTxt\(\)](#)

9.23.2.9 #define DRV_HAS_DRAW_TRI_FILL

Support [gslc_DrvDrawFillTriangle\(\)](#)

9.23.2.10 #define DRV_HAS_DRAW_TRI_FRAME

Support [gslc_DrvDrawFrameTriangle\(\)](#)

9.23.2.11 #define DRV_OVERRIDE_TXT_ALIGN

Driver provides text alignment.

9.23.3 Function Documentation

9.23.3.1 uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor *nCol*)

9.23.3.2 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

9.23.3.3 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

9.23.3.4 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

9.23.3.5 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

9.23.3.6 `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

9.23.3.7 `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

9.23.3.8 `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

9.23.3.9 `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

9.23.3.10 `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

9.23.3.11 `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

9.23.3.12 `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

9.23.3.13 `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

9.23.3.14 `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

9.23.3.15 `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↔ NumPt</i>	Number of points in array
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.23.3.16 `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

9.23.3.17 `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

9.23.3.18 `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

9.23.3.19 void gslc_DrvFontsDestruct (gslc_tsGui * *pGui*)

Release all fonts defined in the GUI.

Parameters

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

Returns

none

9.23.3.20 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

9.23.3.21 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

9.23.3.22 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

9.23.3.23 void gslc_DrvImageDestruct (void * *pvlmg*)

Release an image surface.

Parameters

in	<i>pvlmg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.23.3.24 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

9.23.3.25 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

9.23.3.26 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

9.23.3.27 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

9.23.3.28 `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

9.23.3.29 `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

9.23.3.30 `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

9.23.3.31 `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

9.23.3.32 `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

9.23.3.33 `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

9.23.4 Variable Documentation

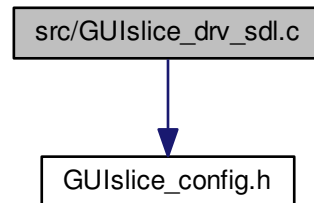
9.23.4.1 `const char GSLC_PMEM ERRSTR_NULL[]`

9.23.4.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

9.24 src/GUISlice_drv_sdl.c File Reference

```
#include "GUISlice_config.h"
```

Include dependency graph for GUISlice_drv_sdl.c:



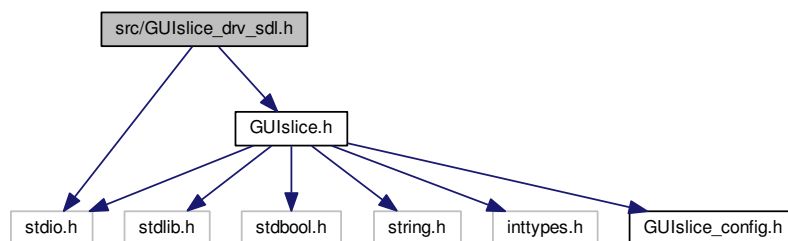
9.25 src/GUISlice_drv_sdl.h File Reference

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_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_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_tsInputRawEvent](#) *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.

9.25.1 Detailed Description

GUISlice library (driver layer for LINUX / SDL)

9.25.2 Macro Definition Documentation

9.25.2.1 #define DRV_HAS_DRAW_POINT

Support [gslc_DrvDrawPoint\(\)](#)

9.25.2.2 #define DRV_OVERRIDE_TXT_ALIGN

Driver provides text alignment.

9.25.3 Function Documentation

9.25.3.1 SDL_Color [gslc_DrvAdaptColor](#) ([gslc_tsColor](#) sCol)

Translate a [gslc_tsColor](#) into an SDL_Color.

Parameters

in	sCol	gslc_tsColor
----	------	------------------------------

Returns

Converted SDL_Color

9.25.3.2 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

9.25.3.3 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

9.25.3.4 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

9.25.3.5 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

9.25.3.6 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

9.25.3.7 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

9.25.3.8 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

9.25.3.9 `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

9.25.3.10 `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

9.25.3.11 `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

9.25.3.12 `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

9.25.3.13 `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

9.25.3.14 void gslc_DrvFontsDestruct (gslc_tsGui * pGui)

Release all fonts defined in the GUI.

Parameters

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

Returns

none

9.25.3.15 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

9.25.3.16 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

9.25.3.17 bool gslc_DrvGetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress, gslc_telInputRawEvent * pnInputEvent, 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

Parameters

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

9.25.3.18 `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

9.25.3.19 `void gslc_DrvImageDestruct (void * pvlmg)`

Release an image surface.

Parameters

in	<i>pvlmg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.25.3.20 `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

9.25.3.21 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

9.25.3.22 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/path) or NULL if error

9.25.3.23 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

9.25.3.24 void gslc_DrvReportInfoPost ()

Report driver debug info (after initialization)

Returns

none

9.25.3.25 void gslc_DrvReportInfoPre ()

Report driver debug info (before initialization)

Returns

none

9.25.3.26 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

9.25.3.27 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

9.25.3.28 `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

9.25.3.29 `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

9.25.3.30 `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

9.25.3.31 `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

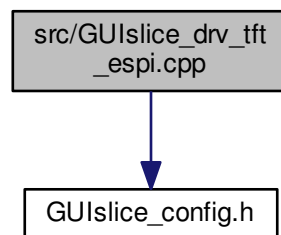
Returns

true if success, false if error

9.26 src/GUISlice_drv_tft_espi.cpp File Reference

```
#include "GUISlice_config.h"
```

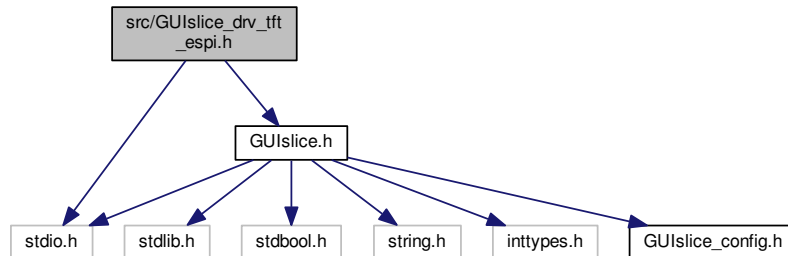
Include dependency graph for GUISlice_drv_tft_espi.cpp:

**9.27 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 [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_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_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_DrvLoadImage](#) ([gslc_tsGui](#) *pGui, [gslc_tslmgRef](#) slmgRef)
Load a bitmap (.bmp) and create a new image resource.*
- bool [gslc_DrvSetBkgndImage](#) ([gslc_tsGui](#) *pGui, [gslc_tslmgRef](#) slmgRef)
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_tslmgRef](#) slmgRef)
Set an element's normal-state image.
- bool [gslc_DrvSetElemImageGlow](#) ([gslc_tsGui](#) *pGui, [gslc_tsElem](#) *pElem, [gslc_tslmgRef](#) slmgRef)
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.
- 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.
- 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_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)

9.27.1 Detailed Description

GUISlice library (driver layer for TFT-eSPI)

9.27.2 Macro Definition Documentation

9.27.2.1 #define DRV_HAS_DRAW_CIRCLE_FILL

Support [gslc_DrvDrawFillCircle\(\)](#)

9.27.2.2 #define DRV_HAS_DRAW_CIRCLE_FRAME

Support [gslc_DrvDrawFrameCircle\(\)](#)

9.27.2.3 #define DRV_HAS_DRAW_LINE

Support [gslc_DrvDrawLine\(\)](#)

9.27.2.4 `#define DRV_HAS_DRAW_POINT`

Support [gslc_DrvDrawPoint\(\)](#)

9.27.2.5 `#define DRV_HAS_DRAW_POINTS`

Support [gslc_DrvDrawPoints\(\)](#)

9.27.2.6 `#define DRV_HAS_DRAW_RECT_FILL`

Support [gslc_DrvDrawFillRect\(\)](#)

9.27.2.7 `#define DRV_HAS_DRAW_RECT_FRAME`

Support [gslc_DrvDrawFrameRect\(\)](#)

9.27.2.8 `#define DRV_HAS_DRAW_TEXT`

Support [gslc_DrvDrawTxt\(\)](#)

9.27.2.9 `#define DRV_HAS_DRAW_TRI_FILL`

Support [gslc_DrvDrawFillTriangle\(\)](#)

9.27.2.10 `#define DRV_HAS_DRAW_TRI_FRAME`

Support [gslc_DrvDrawFrameTriangle\(\)](#)

9.27.2.11 `#define DRV_OVERRIDE_TXT_ALIGN`

Driver provides text alignment.

9.27.3 Function Documentation

9.27.3.1 `uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)`

9.27.3.2 `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

9.27.3.3 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

9.27.3.4 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

9.27.3.5 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

9.27.3.6 `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

9.27.3.7 `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

9.27.3.8 `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

9.27.3.9 `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

9.27.3.10 `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

9.27.3.11 `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

9.27.3.12 `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

9.27.3.13 `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

9.27.3.14 `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

9.27.3.15 `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↔ NumPt</i>	Number of points in array
in	<i>nCol</i>	Color RGB value to draw

Returns

true if success, false if error

9.27.3.16 `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

9.27.3.17 `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

9.27.3.18 `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

9.27.3.19 void gslc_DrvFontsDestruct (gslc_tsGui * *pGui*)

Release all fonts defined in the GUI.

Parameters

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

Returns

none

9.27.3.20 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

9.27.3.21 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

9.27.3.22 `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

9.27.3.23 `void gslc_DrvImageDestruct (void * pvlmg)`

Release an image surface.

Parameters

in	<i>pvlmg</i>	Void ptr to image
----	--------------	-------------------

Returns

none

9.27.3.24 `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

9.27.3.25 `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

9.27.3.26 `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

9.27.3.27 `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

9.27.3.28 `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

9.27.3.29 `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

9.27.3.30 `bool gslc_DrvSetBkgndImage (gslc_tsGui * pGui, gslc_tslmgRef slmgRef)`

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>slmgRef</i>	Image reference

Returns

true if success, false if fail

9.27.3.31 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

9.27.3.32 bool gslc_DrvSetElemImageGlow (gslc_tsGui * *pGui*, gslc_tsElem * *pElem*, gslc_tslmgRef *slmgRef*)

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>slmgRef</i>	Image reference

Returns

true if success, false if error

9.27.3.33 bool gslc_DrvSetElemImageNorm (gslc_tsGui * *pGui*, gslc_tsElem * *pElem*, gslc_tslmgRef *slmgRef*)

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>slmgRef</i>	Image reference

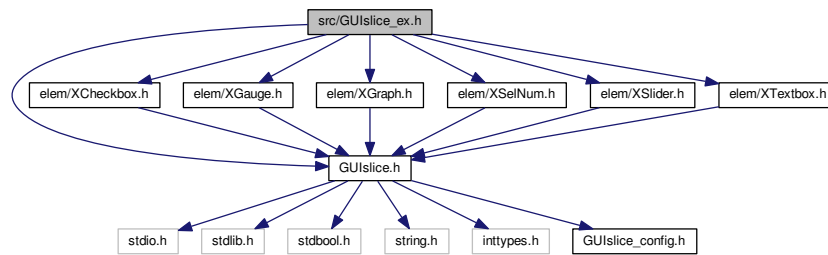
Returns

true if success, false if error

9.28 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"
```

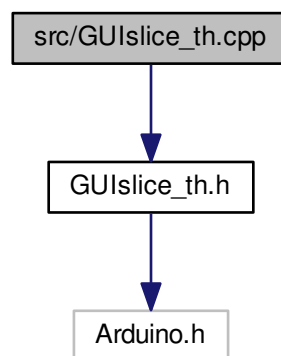
Include dependency graph for GUISlice_ex.h:



9.29 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](#)

9.29.1 Function Documentation

9.29.1.1 `TouchHandler* gslc_getTouchHandler (void)`

9.29.1.2 `void gslc_InitTouchHandler (TouchHandler * pTH)`

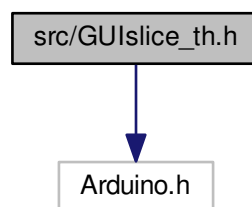
9.29.2 Variable Documentation

9.29.2.1 `TouchHandler* pTouchHandler`

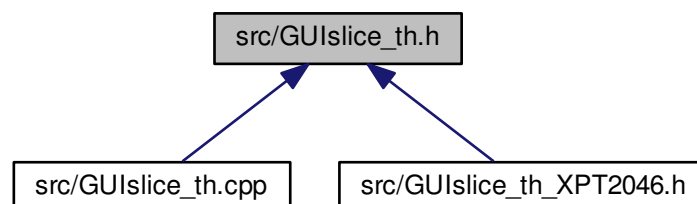
9.30 src/GUIslice_th.h File Reference

```
#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)

9.30.1 Function Documentation

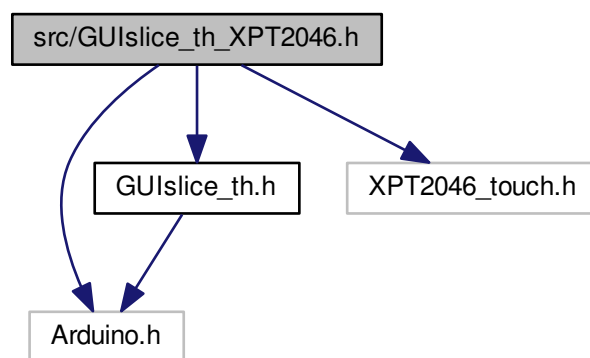
9.30.1.1 [TouchHandler](#)* [gslc_getTouchHandler](#) (void)

9.30.1.2 void [gslc_InitTouchHandler](#) ([TouchHandler](#) * *pTHO*)

9.31 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:

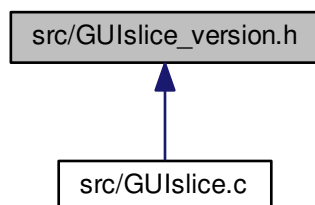


Data Structures

- class [TouchHandler_XPT2046](#)

9.32 src/GUISlice_version.h File Reference

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



Macros

- `#define` [GUISLICE_VER](#)

9.32.1 Macro Definition Documentation

9.32.1.1 `#define` GUISLICE_VER

