

GUIslice

0.10.0

Generated by Doxygen 1.8.8

Tue Jan 2 2018 07:02:26

Contents

1	GUIslice library	1
2	Class Index	3
2.1	Class List	3
3	File Index	5
3.1	File List	5
4	Class Documentation	7
4.1	gslc_tsCollect Struct Reference	7
4.1.1	Detailed Description	8
4.1.2	Member Data Documentation	8
4.1.2.1	asElem	8
4.1.2.2	asElemRef	8
4.1.2.3	nElemAutoldNext	8
4.1.2.4	nElemCnt	8
4.1.2.5	nElemMax	8
4.1.2.6	nElemRefCnt	8
4.1.2.7	nElemRefMax	8
4.1.2.8	pElemRefTracked	9
4.1.2.9	pfuncXEvent	9
4.2	gslc_tsColor Struct Reference	9
4.2.1	Detailed Description	9
4.2.2	Member Data Documentation	9
4.2.2.1	b	9
4.2.2.2	g	9
4.2.2.3	r	9
4.3	gslc_tsDriver Struct Reference	10
4.3.1	Member Data Documentation	10
4.3.1.1	nColRawBkgnd	10
4.3.1.2	rClipRect	10
4.4	gslc_tsElem Struct Reference	10

4.4.1	Detailed Description	12
4.4.2	Member Data Documentation	12
4.4.2.1	colElemFill	12
4.4.2.2	colElemFillGlow	12
4.4.2.3	colElemFrame	12
4.4.2.4	colElemFrameGlow	12
4.4.2.5	colElemText	13
4.4.2.6	colElemTextGlow	13
4.4.2.7	eTxtAlign	13
4.4.2.8	eTxtFlags	13
4.4.2.9	nFeatures	13
4.4.2.10	nGroup	13
4.4.2.11	nId	13
4.4.2.12	nStrBufMax	13
4.4.2.13	nTxtMargin	13
4.4.2.14	nType	13
4.4.2.15	pElemRefParent	13
4.4.2.16	pfuncXDraw	13
4.4.2.17	pfuncXEvent	14
4.4.2.18	pfuncXTick	14
4.4.2.19	pfuncXTouch	14
4.4.2.20	pStrBuf	14
4.4.2.21	pTxtFont	14
4.4.2.22	pXData	14
4.4.2.23	rElem	14
4.4.2.24	slmgRefGlow	14
4.4.2.25	slmgRefNorm	14
4.5	gslc_tsElemRef Struct Reference	14
4.5.1	Detailed Description	15
4.5.2	Member Data Documentation	15
4.5.2.1	eElemFlags	15
4.5.2.2	pElem	15
4.6	gslc_tsEvent Struct Reference	15
4.6.1	Detailed Description	16
4.6.2	Member Data Documentation	16
4.6.2.1	eType	16
4.6.2.2	nSubType	16
4.6.2.3	pvData	16
4.6.2.4	pvScope	16
4.7	gslc_tsEventTouch Struct Reference	16

4.7.1	Detailed Description	16
4.7.2	Member Data Documentation	17
4.7.2.1	eTouch	17
4.7.2.2	nX	17
4.7.2.3	nY	17
4.8	gslc_tsFont Struct Reference	17
4.8.1	Detailed Description	17
4.8.2	Member Data Documentation	17
4.8.2.1	eFontRefType	17
4.8.2.2	nId	17
4.8.2.3	nSize	18
4.8.2.4	pvFont	18
4.9	gslc_tsGui Struct Reference	18
4.9.1	Detailed Description	19
4.9.2	Member Data Documentation	19
4.9.2.1	asFont	19
4.9.2.2	asPage	19
4.9.2.3	bRedrawPartialEn	20
4.9.2.4	nDispDepth	20
4.9.2.5	nDispH	20
4.9.2.6	nDispW	20
4.9.2.7	nFontCnt	20
4.9.2.8	nFontMax	20
4.9.2.9	nFrameRateCnt	20
4.9.2.10	nFrameRateStart	20
4.9.2.11	nPageCnt	20
4.9.2.12	nPageMax	20
4.9.2.13	nTouchLastPress	20
4.9.2.14	nTouchLastX	20
4.9.2.15	nTouchLastY	21
4.9.2.16	pCurPage	21
4.9.2.17	pCurPageCollect	21
4.9.2.18	pfuncXEvent	21
4.9.2.19	pvDriver	21
4.9.2.20	sElemTmpProg	21
4.9.2.21	sImgRefBkgnd	21
4.10	gslc_tsImgRef Struct Reference	21
4.10.1	Detailed Description	22
4.10.2	Member Data Documentation	22
4.10.2.1	eImgFlags	22

4.10.2.2	pFname	22
4.10.2.3	pImgBuf	22
4.10.2.4	pVImgRaw	22
4.11	gslc_tsPage Struct Reference	22
4.11.1	Detailed Description	23
4.11.2	Member Data Documentation	23
4.11.2.1	bPageNeedFlip	23
4.11.2.2	bPageNeedRedraw	24
4.11.2.3	nPageId	24
4.11.2.4	pfuncXEvent	24
4.11.2.5	sCollect	24
4.12	gslc_tsPt Struct Reference	24
4.12.1	Detailed Description	24
4.12.2	Member Data Documentation	24
4.12.2.1	x	24
4.12.2.2	y	24
4.13	gslc_tsRect Struct Reference	25
4.13.1	Detailed Description	25
4.13.2	Member Data Documentation	25
4.13.2.1	h	25
4.13.2.2	w	25
4.13.2.3	x	25
4.13.2.4	y	25
4.14	gslc_tsXCheckbox Struct Reference	25
4.14.1	Detailed Description	26
4.14.2	Member Data Documentation	26
4.14.2.1	bChecked	26
4.14.2.2	bRadio	26
4.14.2.3	colCheck	26
4.14.2.4	nStyle	26
4.15	gslc_tsXGauge Struct Reference	27
4.15.1	Detailed Description	28
4.15.2	Member Data Documentation	28
4.15.2.1	bFlip	28
4.15.2.2	bIndicFill	28
4.15.2.3	bValLastValid	28
4.15.2.4	bVert	28
4.15.2.5	colGauge	28
4.15.2.6	colTick	28
4.15.2.7	nIndicLen	28

4.15.2.8	nIndicTip	28
4.15.2.9	nMax	28
4.15.2.10	nMin	28
4.15.2.11	nStyle	29
4.15.2.12	nTickCnt	29
4.15.2.13	nTickLen	29
4.15.2.14	nVal	29
4.15.2.15	nValLast	29
4.16	gslc_tsXGraph Struct Reference	29
4.16.1	Detailed Description	30
4.16.2	Member Data Documentation	30
4.16.2.1	bScrollEn	30
4.16.2.2	colGraph	30
4.16.2.3	eStyle	30
4.16.2.4	nBufCnt	30
4.16.2.5	nBufMax	30
4.16.2.6	nMargin	31
4.16.2.7	nPlotIndMax	31
4.16.2.8	nPlotIndStart	31
4.16.2.9	nPlotValMax	31
4.16.2.10	nPlotValMin	31
4.16.2.11	nScrollPos	31
4.16.2.12	nWndHeight	31
4.16.2.13	nWndWidth	31
4.16.2.14	pBuf	31
4.17	gslc_tsXSlider Struct Reference	31
4.17.1	Detailed Description	32
4.17.2	Member Data Documentation	32
4.17.2.1	bTrim	32
4.17.2.2	bVert	33
4.17.2.3	colTick	33
4.17.2.4	colTrim	33
4.17.2.5	nPos	33
4.17.2.6	nPosMax	33
4.17.2.7	nPosMin	33
4.17.2.8	nThumbSz	33
4.17.2.9	nTickDiv	33
4.17.2.10	nTickLen	33
4.17.2.11	pfuncXPos	33
4.18	gslc_tsXTextbox Struct Reference	33

4.18.1	Detailed Description	34
4.18.2	Member Data Documentation	34
4.18.2.1	bScrollEn	34
4.18.2.2	bWrapEn	34
4.18.2.3	nBufCols	35
4.18.2.4	nBufPosX	35
4.18.2.5	nBufPosY	35
4.18.2.6	nBufRows	35
4.18.2.7	nChSizeX	35
4.18.2.8	nChSizeY	35
4.18.2.9	nCurPosX	35
4.18.2.10	nCurPosY	35
4.18.2.11	nMargin	35
4.18.2.12	nScrollPos	35
4.18.2.13	nWndCols	35
4.18.2.14	nWndRows	35
4.18.2.15	nWndRowStart	36
4.18.2.16	pBuf	36
5	File Documentation	37
5.1	linux/gslc-ex01.c File Reference	37
5.1.1	Macro Definition Documentation	38
5.1.1.1	MAX_ELEM_PG_MAIN	38
5.1.1.2	MAX_PAGE	38
5.1.2	Enumeration Type Documentation	38
5.1.2.1	anonymous enum	38
5.1.2.2	anonymous enum	38
5.1.3	Function Documentation	38
5.1.3.1	DebugOut	38
5.1.3.2	main	38
5.1.3.3	UserInitEnv	38
5.1.4	Variable Documentation	38
5.1.4.1	m_asPage	38
5.1.4.2	m_asPageElem	38
5.1.4.3	m_asPageElemRef	38
5.1.4.4	m_drv	38
5.1.4.5	m_gui	38
5.2	linux/gslc-ex02.c File Reference	38
5.2.1	Macro Definition Documentation	39
5.2.1.1	FONT_DROID_SANS	39

5.2.1.2	MAX_ELEM_PG_MAIN	39
5.2.1.3	MAX_FONT	39
5.2.1.4	MAX_PAGE	40
5.2.2	Enumeration Type Documentation	40
5.2.2.1	anonymous enum	40
5.2.2.2	anonymous enum	40
5.2.2.3	anonymous enum	40
5.2.3	Function Documentation	40
5.2.3.1	CbBtnQuit	40
5.2.3.2	DebugOut	40
5.2.3.3	main	40
5.2.3.4	UserInitEnv	40
5.2.4	Variable Documentation	40
5.2.4.1	m_asFont	40
5.2.4.2	m_asPage	40
5.2.4.3	m_asPageElem	40
5.2.4.4	m_asPageElemRef	40
5.2.4.5	m_bQuit	40
5.2.4.6	m_drv	40
5.2.4.7	m_gui	40
5.3	linux/gslc-ex03.c File Reference	40
5.3.1	Macro Definition Documentation	42
5.3.1.1	IMG_BTN_QUIT	42
5.3.1.2	IMG_BTN_QUIT_SEL	42
5.3.1.3	MAX_ELEM_PG_MAIN	42
5.3.1.4	MAX_PAGE	42
5.3.1.5	MAX_PATH	42
5.3.2	Enumeration Type Documentation	42
5.3.2.1	anonymous enum	42
5.3.2.2	anonymous enum	42
5.3.3	Function Documentation	42
5.3.3.1	CbBtnQuit	42
5.3.3.2	DebugOut	42
5.3.3.3	InitOverlays	42
5.3.3.4	main	42
5.3.3.5	UserInitEnv	42
5.3.4	Variable Documentation	42
5.3.4.1	m_asPage	42
5.3.4.2	m_asPageElem	42
5.3.4.3	m_asPageElemRef	42

5.3.4.4	m_bQuit	42
5.3.4.5	m_drv	42
5.3.4.6	m_gui	42
5.3.4.7	m_strImgQuit	42
5.3.4.8	m_strImgQuitSel	43
5.4	linux/gslc-ex04.c File Reference	43
5.4.1	Macro Definition Documentation	44
5.4.1.1	FONT_DROID_SANS	44
5.4.1.2	MAX_ELEM_PG_MAIN	44
5.4.1.3	MAX_FONT	44
5.4.1.4	MAX_PAGE	44
5.4.1.5	MAX_STR	44
5.4.1.6	TEST_UPDATE_RATE	44
5.4.2	Enumeration Type Documentation	44
5.4.2.1	anonymous enum	44
5.4.2.2	anonymous enum	44
5.4.2.3	anonymous enum	44
5.4.2.4	anonymous enum	45
5.4.3	Function Documentation	45
5.4.3.1	CbBtnQuit	45
5.4.3.2	DebugOut	45
5.4.3.3	InitOverlays	45
5.4.3.4	main	45
5.4.3.5	UserInitEnv	45
5.4.4	Variable Documentation	45
5.4.4.1	m_asFont	45
5.4.4.2	m_asPage	45
5.4.4.3	m_asPageElem	45
5.4.4.4	m_asPageElemRef	45
5.4.4.5	m_asXCheck	45
5.4.4.6	m_bQuit	45
5.4.4.7	m_drv	45
5.4.4.8	m_gui	45
5.4.4.9	m_nCount	45
5.4.4.10	m_sXGauge	45
5.4.4.11	m_sXGauge1	45
5.4.4.12	m_sXSlider	45
5.5	linux/gslc-ex05.c File Reference	45
5.5.1	Macro Definition Documentation	47
5.5.1.1	FONT_DROID_SANS	47

5.5.1.2	IMG_BKGND	47
5.5.1.3	MAX_ELEM_PG_EXTRA	47
5.5.1.4	MAX_ELEM_PG_MAIN	47
5.5.1.5	MAX_FONT	47
5.5.1.6	MAX_PAGE	47
5.5.1.7	MAX_PATH	47
5.5.1.8	MAX_STR	47
5.5.2	Enumeration Type Documentation	47
5.5.2.1	anonymous enum	47
5.5.2.2	anonymous enum	47
5.5.2.3	anonymous enum	47
5.5.3	Function Documentation	48
5.5.3.1	CbBtnCommon	48
5.5.3.2	DebugOut	48
5.5.3.3	InitOverlays	48
5.5.3.4	main	48
5.5.3.5	UserInitEnv	48
5.5.4	Variable Documentation	48
5.5.4.1	m_asExtraElem	48
5.5.4.2	m_asExtraElemRef	48
5.5.4.3	m_asFont	48
5.5.4.4	m_asMainElem	48
5.5.4.5	m_asMainElemRef	48
5.5.4.6	m_asPage	48
5.5.4.7	m_bQuit	48
5.5.4.8	m_drv	48
5.5.4.9	m_gui	48
5.5.4.10	m_nCount	48
5.5.4.11	m_strImgBkgnd	48
5.5.4.12	m_sXGauge	48
5.5.4.13	m_sXSelNum	48
5.6	linux/gslc-ex06.c File Reference	48
5.6.1	Macro Definition Documentation	50
5.6.1.1	FONT_DROID_SANS	50
5.6.1.2	IMG_LOGO	50
5.6.1.3	MAX_ELEM_PG_MAIN	50
5.6.1.4	MAX_FONT	50
5.6.1.5	MAX_PAGE	50
5.6.1.6	MAX_PATH	50
5.6.1.7	MAX_STR	50

5.6.2	Enumeration Type Documentation	50
5.6.2.1	anonymous enum	50
5.6.2.2	anonymous enum	50
5.6.2.3	anonymous enum	50
5.6.3	Function Documentation	51
5.6.3.1	CbBtnQuit	51
5.6.3.2	CbDrawScanner	51
5.6.3.3	CbTickScanner	51
5.6.3.4	DebugOut	51
5.6.3.5	InitOverlays	51
5.6.3.6	main	51
5.6.3.7	UserInitEnv	51
5.6.4	Variable Documentation	51
5.6.4.1	m_asFont	51
5.6.4.2	m_asPage	51
5.6.4.3	m_asPageElem	51
5.6.4.4	m_asPageElemRef	51
5.6.4.5	m_asXCheck	51
5.6.4.6	m_bQuit	51
5.6.4.7	m_drv	51
5.6.4.8	m_fCoordX	51
5.6.4.9	m_fCoordY	51
5.6.4.10	m_fCoordZ	51
5.6.4.11	m_gui	51
5.6.4.12	m_nCount	51
5.6.4.13	m_nOriginX	51
5.6.4.14	m_nOriginY	51
5.6.4.15	m_strImgLogo	51
5.6.4.16	m_sXGauge	51
5.7	linux/gslc-ex07.c File Reference	51
5.7.1	Macro Definition Documentation	53
5.7.1.1	FONT_DROID_SANS	53
5.7.1.2	MAX_ELEM_PG_MAIN	53
5.7.1.3	MAX_FONT	53
5.7.1.4	MAX_PAGE	53
5.7.2	Enumeration Type Documentation	53
5.7.2.1	anonymous enum	53
5.7.2.2	anonymous enum	53
5.7.2.3	anonymous enum	53
5.7.3	Function Documentation	53

5.7.3.1	CbBtnQuit	53
5.7.3.2	CbSlidePos	53
5.7.3.3	DebugOut	53
5.7.3.4	InitOverlays	53
5.7.3.5	main	54
5.7.3.6	UserInitEnv	54
5.7.4	Variable Documentation	54
5.7.4.1	m_asFont	54
5.7.4.2	m_asPage	54
5.7.4.3	m_asPageElem	54
5.7.4.4	m_asPageElemRef	54
5.7.4.5	m_bQuit	54
5.7.4.6	m_drv	54
5.7.4.7	m_gui	54
5.7.4.8	m_nCount	54
5.7.4.9	m_nPosB	54
5.7.4.10	m_nPosG	54
5.7.4.11	m_nPosR	54
5.7.4.12	m_sXSlider_B	54
5.7.4.13	m_sXSlider_G	54
5.7.4.14	m_sXSlider_R	54
5.8	linux/gslc-ex08.c File Reference	54
5.8.1	Macro Definition Documentation	56
5.8.1.1	FONT_DROID_SANS	56
5.8.1.2	IMG_GRAD_BACK	56
5.8.1.3	IMG_GRADBAR_BOT	56
5.8.1.4	IMG_GRADBAR_TOP	56
5.8.1.5	MAX_ELEM_PG_MAIN	56
5.8.1.6	MAX_FONT	56
5.8.1.7	MAX_PAGE	56
5.8.1.8	MAX_PATH	56
5.8.2	Enumeration Type Documentation	56
5.8.2.1	anonymous enum	56
5.8.2.2	anonymous enum	56
5.8.2.3	anonymous enum	57
5.8.3	Function Documentation	57
5.8.3.1	CbBtnQuit	57
5.8.3.2	InitOverlays	57
5.8.3.3	main	57
5.8.3.4	UserInitEnv	57

5.8.4	Variable Documentation	57
5.8.4.1	m_asFont	57
5.8.4.2	m_asPage	57
5.8.4.3	m_asPageElem	57
5.8.4.4	m_asPageElemRef	57
5.8.4.5	m_bQuit	57
5.8.4.6	m_drv	57
5.8.4.7	m_gui	57
5.8.4.8	m_nCount	57
5.8.4.9	m_strImgGradBack	57
5.8.4.10	m_strImgGradBarBot	57
5.8.4.11	m_strImgGradBarTop	57
5.9	linux/gslc-ex09.c File Reference	57
5.9.1	Macro Definition Documentation	59
5.9.1.1	FONT_DROID_SANS	59
5.9.1.2	MAX_ELEM_PG_MAIN	59
5.9.1.3	MAX_FONT	59
5.9.1.4	MAX_PAGE	59
5.9.2	Enumeration Type Documentation	59
5.9.2.1	anonymous enum	59
5.9.2.2	anonymous enum	59
5.9.2.3	anonymous enum	59
5.9.3	Function Documentation	59
5.9.3.1	CbBtnQuit	59
5.9.3.2	CbSlideRadial	59
5.9.3.3	DebugOut	59
5.9.3.4	InitOverlays	59
5.9.3.5	main	60
5.9.3.6	UserInitEnv	60
5.9.4	Variable Documentation	60
5.9.4.1	m_asFont	60
5.9.4.2	m_asPage	60
5.9.4.3	m_asPageElem	60
5.9.4.4	m_asPageElemRef	60
5.9.4.5	m_bQuit	60
5.9.4.6	m_drv	60
5.9.4.7	m_gui	60
5.9.4.8	m_nCount	60
5.9.4.9	m_sXRadial	60
5.9.4.10	m_sXRamp	60

5.9.4.11	m_sXSlider	60
5.10	linux/gslc-ex10.c File Reference	60
5.10.1	Macro Definition Documentation	61
5.10.1.1	FONT_DROID_SANS	61
5.10.1.2	MAX_ELEM_PG_MAIN	61
5.10.1.3	MAX_FONT	61
5.10.1.4	MAX_PAGE	61
5.10.1.5	TBOX_COLS	61
5.10.1.6	TBOX_ROWS	61
5.10.2	Enumeration Type Documentation	62
5.10.2.1	anonymous enum	62
5.10.2.2	anonymous enum	62
5.10.2.3	anonymous enum	62
5.10.3	Function Documentation	62
5.10.3.1	CbBtnQuit	62
5.10.3.2	CbControls	62
5.10.3.3	DebugOut	62
5.10.3.4	InitOverlays	62
5.10.3.5	main	62
5.10.3.6	UserInitEnv	62
5.10.4	Variable Documentation	62
5.10.4.1	m_acTextboxBuf	62
5.10.4.2	m_asFont	62
5.10.4.3	m_asPage	62
5.10.4.4	m_asPageElem	62
5.10.4.5	m_asPageElemRef	62
5.10.4.6	m_bQuit	63
5.10.4.7	m_drv	63
5.10.4.8	m_gui	63
5.10.4.9	m_nCount	63
5.10.4.10	m_sTextbox	63
5.10.4.11	m_sXSlider	63
5.10.4.12	m_sXSliderText	63
5.11	linux/gslc-ex11.c File Reference	63
5.11.1	Macro Definition Documentation	64
5.11.1.1	FONT_DROID_SANS	64
5.11.1.2	GRAPH_ROWS	64
5.11.1.3	MAX_ELEM_PG_MAIN	64
5.11.1.4	MAX_FONT	64
5.11.1.5	MAX_PAGE	64

5.11.2	Enumeration Type Documentation	64
5.11.2.1	anonymous enum	64
5.11.2.2	anonymous enum	64
5.11.2.3	anonymous enum	65
5.11.3	Function Documentation	65
5.11.3.1	CbBtnQuit	65
5.11.3.2	CbControls	65
5.11.3.3	DebugOut	65
5.11.3.4	InitOverlays	65
5.11.3.5	main	65
5.11.3.6	UserInitEnv	65
5.11.4	Variable Documentation	65
5.11.4.1	m_anGraphBuf	65
5.11.4.2	m_asFont	65
5.11.4.3	m_asPage	65
5.11.4.4	m_asPageElem	65
5.11.4.5	m_asPageElemRef	65
5.11.4.6	m_bQuit	65
5.11.4.7	m_drv	65
5.11.4.8	m_gui	65
5.11.4.9	m_nCount	65
5.11.4.10	m_sGraph	65
5.11.4.11	m_sXSlider	65
5.11.4.12	m_sXSliderGraph	65
5.12	linux/test-sdl1.c File Reference	65
5.12.1	Function Documentation	66
5.12.1.1	main	66
5.12.2	Variable Documentation	66
5.12.2.1	scrMain	66
5.13	linux/test-sdl2.c File Reference	66
5.13.1	Function Documentation	67
5.13.1.1	main	67
5.13.2	Variable Documentation	67
5.13.2.1	pRender	67
5.13.2.2	pWind	67
5.14	README.md File Reference	67
5.15	src/GUISlice.c File Reference	67
5.15.1	Macro Definition Documentation	73
5.15.1.1	GUISLICE_VER	73
5.15.2	Enumeration Type Documentation	73

5.15.2.1	gslc_teDebugPrintState	73
5.15.3	Function Documentation	73
5.15.3.1	gslc_ClipLine	73
5.15.3.2	gslc_ClipPt	74
5.15.3.3	gslc_ClipRect	74
5.15.3.4	gslc_CollectDestruct	74
5.15.3.5	gslc_CollectElemAdd	75
5.15.3.6	gslc_CollectEvent	75
5.15.3.7	gslc_CollectFindElemById	75
5.15.3.8	gslc_CollectFindElemFromCoord	75
5.15.3.9	gslc_CollectGetElemRefTracked	76
5.15.3.10	gslc_CollectGetNextId	76
5.15.3.11	gslc_CollectGetRedraw	76
5.15.3.12	gslc_CollectReset	76
5.15.3.13	gslc_CollectSetElemTracked	77
5.15.3.14	gslc_CollectSetEventFunc	77
5.15.3.15	gslc_CollectTouch	77
5.15.3.16	gslc_ColorBlend2	78
5.15.3.17	gslc_ColorBlend3	78
5.15.3.18	gslc_ColorEqual	78
5.15.3.19	gslc_cosFX	79
5.15.3.20	gslc_DebugPrintf	79
5.15.3.21	gslc_DrawFillCircle	79
5.15.3.22	gslc_DrawFillQuad	80
5.15.3.23	gslc_DrawFillRect	80
5.15.3.24	gslc_DrawFillTriangle	80
5.15.3.25	gslc_DrawFrameCircle	81
5.15.3.26	gslc_DrawFrameQuad	82
5.15.3.27	gslc_DrawFrameRect	82
5.15.3.28	gslc_DrawFrameTriangle	82
5.15.3.29	gslc_DrawLine	83
5.15.3.30	gslc_DrawLineH	83
5.15.3.31	gslc_DrawLinePolar	83
5.15.3.32	gslc_DrawLineV	84
5.15.3.33	gslc_DrawSetPixel	84
5.15.3.34	gslc_ElemAdd	84
5.15.3.35	gslc_ElemCreate	85
5.15.3.36	gslc_ElemCreateBox	85
5.15.3.37	gslc_ElemCreateBtnImg	86
5.15.3.38	gslc_ElemCreateBtnTxt	86

5.15.3.39 gslc_ElemCreateImg	86
5.15.3.40 gslc_ElemCreateLine	87
5.15.3.41 gslc_ElemCreateTxt	87
5.15.3.42 gslc_ElemDestruct	88
5.15.3.43 gslc_ElemDraw	88
5.15.3.44 gslc_ElemDrawByRef	88
5.15.3.45 gslc_ElemEvent	88
5.15.3.46 gslc_ElemGetGlow	89
5.15.3.47 gslc_ElemGetGlowEn	89
5.15.3.48 gslc_ElemGetGroup	89
5.15.3.49 gslc_ElemGetId	89
5.15.3.50 gslc_ElemGetRedraw	90
5.15.3.51 gslc_ElemOwnsCoord	90
5.15.3.52 gslc_ElemSendEventTouch	90
5.15.3.53 gslc_ElemSetCol	91
5.15.3.54 gslc_ElemSetDrawFunc	91
5.15.3.55 gslc_ElemSetEventFunc	91
5.15.3.56 gslc_ElemSetFillEn	91
5.15.3.57 gslc_ElemSetFrameEn	92
5.15.3.58 gslc_ElemSetGlow	92
5.15.3.59 gslc_ElemSetGlowCol	92
5.15.3.60 gslc_ElemSetGlowEn	92
5.15.3.61 gslc_ElemSetGroup	93
5.15.3.62 gslc_ElemSetImage	93
5.15.3.63 gslc_ElemSetRedraw	93
5.15.3.64 gslc_ElemSetStyleFrom	94
5.15.3.65 gslc_ElemSetTickFunc	95
5.15.3.66 gslc_ElemSetTxtAlign	95
5.15.3.67 gslc_ElemSetTxtCol	96
5.15.3.68 gslc_ElemSetTxtMargin	97
5.15.3.69 gslc_ElemSetTxtMem	97
5.15.3.70 gslc_ElemSetTxtStr	97
5.15.3.71 gslc_ElemUpdateFont	97
5.15.3.72 gslc_EventCreate	98
5.15.3.73 gslc_ExpandRect	98
5.15.3.74 gslc_FontAdd	98
5.15.3.75 gslc_FontGet	99
5.15.3.76 gslc_GetElemFromRef	99
5.15.3.77 gslc_GetElemRefFlag	99
5.15.3.78 gslc_GetImageFromFile	99

5.15.3.79 gslc_GetImageFromProg	99
5.15.3.80 gslc_GetImageFromRam	100
5.15.3.81 gslc_GetImageFromSD	100
5.15.3.82 gslc_GetPageCur	100
5.15.3.83 gslc_GetTouch	100
5.15.3.84 gslc_GetVer	101
5.15.3.85 gslc_GuiDestruct	101
5.15.3.86 gslc_Init	101
5.15.3.87 gslc_InitDebug	102
5.15.3.88 gslc_InitTouch	102
5.15.3.89 gslc_IsInRect	102
5.15.3.90 gslc_IsInWH	103
5.15.3.91 gslc_OrderCoord	103
5.15.3.92 gslc_PageAdd	103
5.15.3.93 gslc_PageDestruct	104
5.15.3.94 gslc_PageEvent	105
5.15.3.95 gslc_PageFindById	105
5.15.3.96 gslc_PageFindElemById	105
5.15.3.97 gslc_PageFlipGet	105
5.15.3.98 gslc_PageFlipGo	106
5.15.3.99 gslc_PageFlipSet	106
5.15.3.100 gslc_PageRedrawCalc	106
5.15.3.101 gslc_PageRedrawGet	106
5.15.3.102 gslc_PageRedrawGo	107
5.15.3.103 gslc_PageRedrawSet	107
5.15.3.104 gslc_PageSetEventFunc	107
5.15.3.105 gslc_PolarToXY	107
5.15.3.106 gslc_Quit	108
5.15.3.107 gslc_ResetElem	108
5.15.3.108 gslc_ResetFont	108
5.15.3.109 gslc_ResetImage	108
5.15.3.110 gslc_SetBkgndColor	109
5.15.3.111 gslc_SetBkgndImage	109
5.15.3.112 gslc_SetClipRect	109
5.15.3.113 gslc_SetElemRefFlag	109
5.15.3.114 gslc_SetPageCur	109
5.15.3.115 gslc_sinFX	110
5.15.3.116 gslc_SwapCoords	110
5.15.3.117 gslc_TrackTouch	110
5.15.3.118 gslc_Update	110

5.15.4	Variable Documentation	111
5.15.4.1	ERRSTR_NULL	111
5.15.4.2	ERRSTR_PXD_NULL	111
5.15.4.3	g_pfDebugOut	111
5.15.4.4	m_nLUTSinF0X16	111
5.16	src/GUISlice.h File Reference	112
5.16.1	Macro Definition Documentation	122
5.16.1.1	GSLC_2PI	122
5.16.1.2	GSLC_ALIGN_BOT_LEFT	122
5.16.1.3	GSLC_ALIGN_BOT_MID	123
5.16.1.4	GSLC_ALIGN_BOT_RIGHT	123
5.16.1.5	GSLC_ALIGN_MID_LEFT	123
5.16.1.6	GSLC_ALIGN_MID_MID	123
5.16.1.7	GSLC_ALIGN_MID_RIGHT	123
5.16.1.8	GSLC_ALIGN_TOP_LEFT	123
5.16.1.9	GSLC_ALIGN_TOP_MID	123
5.16.1.10	GSLC_ALIGN_TOP_RIGHT	123
5.16.1.11	GSLC_ALIGNH_LEFT	123
5.16.1.12	GSLC_ALIGNH_MID	123
5.16.1.13	GSLC_ALIGNH_RIGHT	123
5.16.1.14	GSLC_ALIGNV_BOT	123
5.16.1.15	GSLC_ALIGNV_MID	124
5.16.1.16	GSLC_ALIGNV_TOP	124
5.16.1.17	GSLC_COL_BLACK	124
5.16.1.18	GSLC_COL_BLUE	124
5.16.1.19	GSLC_COL_BLUE_DK1	124
5.16.1.20	GSLC_COL_BLUE_DK2	124
5.16.1.21	GSLC_COL_BLUE_DK3	124
5.16.1.22	GSLC_COL_BLUE_DK4	124
5.16.1.23	GSLC_COL_BLUE_LT1	124
5.16.1.24	GSLC_COL_BLUE_LT2	124
5.16.1.25	GSLC_COL_BLUE_LT3	124
5.16.1.26	GSLC_COL_BLUE_LT4	124
5.16.1.27	GSLC_COL_BROWN	125
5.16.1.28	GSLC_COL_CYAN	125
5.16.1.29	GSLC_COL_GRAY	125
5.16.1.30	GSLC_COL_GRAY_DK1	125
5.16.1.31	GSLC_COL_GRAY_DK2	125
5.16.1.32	GSLC_COL_GRAY_DK3	125
5.16.1.33	GSLC_COL_GRAY_LT1	125

5.16.1.34 GSLC_COL_GRAY_LT2	125
5.16.1.35 GSLC_COL_GRAY_LT3	125
5.16.1.36 GSLC_COL_GREEN	125
5.16.1.37 GSLC_COL_GREEN_DK1	125
5.16.1.38 GSLC_COL_GREEN_DK2	125
5.16.1.39 GSLC_COL_GREEN_DK3	126
5.16.1.40 GSLC_COL_GREEN_DK4	126
5.16.1.41 GSLC_COL_GREEN_LT1	126
5.16.1.42 GSLC_COL_GREEN_LT2	126
5.16.1.43 GSLC_COL_GREEN_LT3	126
5.16.1.44 GSLC_COL_GREEN_LT4	126
5.16.1.45 GSLC_COL_MAGENTA	126
5.16.1.46 GSLC_COL_ORANGE	126
5.16.1.47 GSLC_COL_PURPLE	126
5.16.1.48 GSLC_COL_RED	126
5.16.1.49 GSLC_COL_RED_DK1	126
5.16.1.50 GSLC_COL_RED_DK2	126
5.16.1.51 GSLC_COL_RED_DK3	127
5.16.1.52 GSLC_COL_RED_DK4	127
5.16.1.53 GSLC_COL_RED_LT1	127
5.16.1.54 GSLC_COL_RED_LT2	127
5.16.1.55 GSLC_COL_RED_LT3	127
5.16.1.56 GSLC_COL_RED_LT4	127
5.16.1.57 GSLC_COL_TEAL	127
5.16.1.58 GSLC_COL_WHITE	127
5.16.1.59 GSLC_COL_YELLOW	127
5.16.1.60 GSLC_COL_YELLOW_DK	127
5.16.1.61 GSLC_COLMONO_BLACK	127
5.16.1.62 GSLC_COLMONO_WHITE	127
5.16.1.63 GSLC_DEBUG_PRINT	128
5.16.1.64 GSLC_DEBUG_PRINT_CONST	128
5.16.1.65 GSLC_ELEM_FEA_CLICK_EN	128
5.16.1.66 GSLC_ELEM_FEA_FILL_EN	128
5.16.1.67 GSLC_ELEM_FEA_FRAME_EN	128
5.16.1.68 GSLC_ELEM_FEA_GLOW_EN	128
5.16.1.69 GSLC_ELEM_FEA_NONE	128
5.16.1.70 GSLC_ELEM_FEA_VALID	129
5.16.1.71 gslc_ElemCreateBox_P	129
5.16.1.72 gslc_ElemCreateBtnTxt_P	129
5.16.1.73 gslc_ElemCreateTxt_P	130

5.16.1.74	gslc_ElemCreateTxt_P_R	131
5.16.1.75	GSLC_PMEM	132
5.16.2	Typedef Documentation	132
5.16.2.1	GSLC_CB_DEBUG_OUT	132
5.16.2.2	GSLC_CB_DRAW	132
5.16.2.3	GSLC_CB_EVENT	132
5.16.2.4	GSLC_CB_TICK	132
5.16.2.5	GSLC_CB_TOUCH	132
5.16.2.6	gslc_tsColor	133
5.16.2.7	gslc_tsElem	133
5.16.2.8	gslc_tsEvent	133
5.16.2.9	gslc_tsEventTouch	133
5.16.2.10	gslc_tsPt	133
5.16.2.11	gslc_tsRect	133
5.16.3	Enumeration Type Documentation	133
5.16.3.1	gslc_teElemId	133
5.16.3.2	gslc_teElemInd	134
5.16.3.3	gslc_teElemRefFlags	134
5.16.3.4	gslc_teEventSubType	134
5.16.3.5	gslc_teEventType	135
5.16.3.6	gslc_teFontId	135
5.16.3.7	gslc_teFontRefType	135
5.16.3.8	gslc_teGroupId	135
5.16.3.9	gslc_teImgRefFlags	136
5.16.3.10	gslc_tePageId	136
5.16.3.11	gslc_teRedrawType	136
5.16.3.12	gslc_teTouch	136
5.16.3.13	gslc_teTxtFlags	137
5.16.3.14	gslc_teTypeCore	137
5.16.4	Function Documentation	138
5.16.4.1	gslc_ClipLine	138
5.16.4.2	gslc_ClipPt	138
5.16.4.3	gslc_ClipRect	138
5.16.4.4	gslc_CollectDestruct	138
5.16.4.5	gslc_CollectElemAdd	139
5.16.4.6	gslc_CollectEvent	139
5.16.4.7	gslc_CollectFindElemById	139
5.16.4.8	gslc_CollectFindElemFromCoord	140
5.16.4.9	gslc_CollectGetElemRefTracked	140
5.16.4.10	gslc_CollectGetNextId	140

5.16.4.11	gslc_CollectGetRedraw	140
5.16.4.12	gslc_CollectReset	141
5.16.4.13	gslc_CollectSetElemTracked	142
5.16.4.14	gslc_CollectSetEventFunc	142
5.16.4.15	gslc_CollectSetParent	142
5.16.4.16	gslc_CollectTouch	143
5.16.4.17	gslc_ColorBlend2	143
5.16.4.18	gslc_ColorBlend3	143
5.16.4.19	gslc_ColorEqual	144
5.16.4.20	gslc_cosFX	144
5.16.4.21	gslc_DebugPrintf	144
5.16.4.22	gslc_DrawFillCircle	144
5.16.4.23	gslc_DrawFillQuad	145
5.16.4.24	gslc_DrawFillRect	145
5.16.4.25	gslc_DrawFillTriangle	145
5.16.4.26	gslc_DrawFrameCircle	146
5.16.4.27	gslc_DrawFrameQuad	146
5.16.4.28	gslc_DrawFrameRect	146
5.16.4.29	gslc_DrawFrameTriangle	146
5.16.4.30	gslc_DrawLine	147
5.16.4.31	gslc_DrawLineH	147
5.16.4.32	gslc_DrawLinePolar	147
5.16.4.33	gslc_DrawLineV	148
5.16.4.34	gslc_DrawSetPixel	148
5.16.4.35	gslc_ElemAdd	149
5.16.4.36	gslc_ElemCreate	150
5.16.4.37	gslc_ElemCreateBox	150
5.16.4.38	gslc_ElemCreateBtnImg	151
5.16.4.39	gslc_ElemCreateBtnTxt	151
5.16.4.40	gslc_ElemCreateImg	151
5.16.4.41	gslc_ElemCreateLine	152
5.16.4.42	gslc_ElemCreateTxt	152
5.16.4.43	gslc_ElemDestruct	153
5.16.4.44	gslc_ElemDraw	154
5.16.4.45	gslc_ElemDrawByRef	154
5.16.4.46	gslc_ElemEvent	154
5.16.4.47	gslc_ElemGetGlow	154
5.16.4.48	gslc_ElemGetGlowEn	155
5.16.4.49	gslc_ElemGetGroup	155
5.16.4.50	gslc_ElemGetId	155

5.16.4.51 gslc_ElemGetRedraw	155
5.16.4.52 gslc_ElemOwnsCoord	156
5.16.4.53 gslc_ElemSendEventTouch	156
5.16.4.54 gslc_ElemSetCol	156
5.16.4.55 gslc_ElemSetDrawFunc	157
5.16.4.56 gslc_ElemSetEventFunc	157
5.16.4.57 gslc_ElemSetFillEn	157
5.16.4.58 gslc_ElemSetFrameEn	158
5.16.4.59 gslc_ElemSetGlow	159
5.16.4.60 gslc_ElemSetGlowCol	159
5.16.4.61 gslc_ElemSetGlowEn	159
5.16.4.62 gslc_ElemSetGroup	160
5.16.4.63 gslc_ElemSetImage	160
5.16.4.64 gslc_ElemSetRedraw	160
5.16.4.65 gslc_ElemSetStyleFrom	160
5.16.4.66 gslc_ElemSetTickFunc	161
5.16.4.67 gslc_ElemSetTxtAlign	161
5.16.4.68 gslc_ElemSetTxtCol	162
5.16.4.69 gslc_ElemSetTxtMargin	163
5.16.4.70 gslc_ElemSetTxtMem	163
5.16.4.71 gslc_ElemSetTxtStr	163
5.16.4.72 gslc_ElemUpdateFont	163
5.16.4.73 gslc_EventCreate	164
5.16.4.74 gslc_ExpandRect	164
5.16.4.75 gslc_FontAdd	164
5.16.4.76 gslc_FontGet	165
5.16.4.77 gslc_GetElemFromRef	165
5.16.4.78 gslc_GetElemRefFlag	165
5.16.4.79 gslc_GetImageFromFile	165
5.16.4.80 gslc_GetImageFromProg	165
5.16.4.81 gslc_GetImageFromRam	166
5.16.4.82 gslc_GetImageFromSD	167
5.16.4.83 gslc_GetPageCur	167
5.16.4.84 gslc_GetTouch	167
5.16.4.85 gslc_GetVer	167
5.16.4.86 gslc_GuiDestruct	168
5.16.4.87 gslc_Init	168
5.16.4.88 gslc_InitDebug	168
5.16.4.89 gslc_InitTouch	169
5.16.4.90 gslc_IsInRect	169

5.16.4.91	gslc_IsInWH	169
5.16.4.92	gslc_PageAdd	170
5.16.4.93	gslc_PageDestruct	170
5.16.4.94	gslc_PageEvent	170
5.16.4.95	gslc_PageFindById	171
5.16.4.96	gslc_PageFindElemById	171
5.16.4.97	gslc_PageFlipGet	171
5.16.4.98	gslc_PageFlipGo	171
5.16.4.99	gslc_PageFlipSet	172
5.16.4.100	gslc_PageRedrawCalc	172
5.16.4.101	gslc_PageRedrawGet	172
5.16.4.102	gslc_PageRedrawGo	172
5.16.4.103	gslc_PageRedrawSet	173
5.16.4.104	gslc_PageSetEventFunc	173
5.16.4.105	gslc_PolarToXY	173
5.16.4.106	gslc_Quit	173
5.16.4.107	gslc_ResetElem	174
5.16.4.108	gslc_ResetFont	174
5.16.4.109	gslc_ResetImage	174
5.16.4.110	gslc_SetBkgndColor	174
5.16.4.111	gslc_SetBkgndImage	175
5.16.4.112	gslc_SetClipRect	175
5.16.4.113	gslc_SetElemRefFlag	175
5.16.4.114	gslc_SetPageCur	175
5.16.4.115	gslc_sinFX	175
5.16.4.116	gslc_TrackTouch	176
5.16.4.117	gslc_Update	176
5.16.5	Variable Documentation	176
5.16.5.1	g_pfDebugOut	176
5.17	src/GUIslice_config.h File Reference	177
5.18	src/GUIslice_config_ard.h File Reference	177
5.18.1	Macro Definition Documentation	178
5.18.1.1	ADAGFX_PIN_CLK	178
5.18.1.2	ADAGFX_PIN_CS	178
5.18.1.3	ADAGFX_PIN_DC	178
5.18.1.4	ADAGFX_PIN_MISO	178
5.18.1.5	ADAGFX_PIN_MOSI	178
5.18.1.6	ADAGFX_PIN_RD	178
5.18.1.7	ADAGFX_PIN_RST	178
5.18.1.8	ADAGFX_PIN_SDCS	178

5.18.1.9	ADAGFX_PIN_WR	178
5.18.1.10	ADAGFX_SPI_HW	178
5.18.1.11	ADATOUCH_FLIP_X	178
5.18.1.12	ADATOUCH_FLIP_Y	178
5.18.1.13	ADATOUCH_I2C_ADDR	178
5.18.1.14	ADATOUCH_I2C_HW	178
5.18.1.15	ADATOUCH_PIN_CS	178
5.18.1.16	ADATOUCH_SPI_HW	178
5.18.1.17	ADATOUCH_SPI_SW	178
5.18.1.18	ADATOUCH_SWAP_XY	178
5.18.1.19	ADATOUCH_X_MAX	178
5.18.1.20	ADATOUCH_X_MIN	178
5.18.1.21	ADATOUCH_Y_MAX	178
5.18.1.22	ADATOUCH_Y_MIN	178
5.18.1.23	DEBUG_ERR	178
5.18.1.24	DRV_DISP_ADAGFX	178
5.18.1.25	DRV_DISP_ADAGFX_ILI9341	178
5.18.1.26	DRV_TOUCH_ADA_STMPE610	178
5.18.1.27	GSLC_BMP_TRANS_EN	178
5.18.1.28	GSLC_BMP_TRANS_RGB	179
5.18.1.29	GSLC_CLIP_EN	179
5.18.1.30	GSLC_DEV_TOUCH	179
5.18.1.31	GSLC_FEATURE_COMPOUND	179
5.18.1.32	GSLC_FEATURE_XGAUGE_RADIAL	179
5.18.1.33	GSLC_FEATURE_XGAUGE_RAMP	179
5.18.1.34	GSLC_LOCAL_STR	179
5.18.1.35	GSLC_LOCAL_STR_LEN	179
5.18.1.36	GSLC_ROTATE	179
5.18.1.37	GSLC_SD_BUFFPIXEL	179
5.18.1.38	GSLC_SD_EN	179
5.18.1.39	GSLC_TOUCH_MAX_EVT	179
5.18.1.40	GSLC_USE_FLOAT	179
5.18.1.41	GSLC_USE_PROGMEM	179
5.19	src/GUIslice_config_esp.h File Reference	179
5.19.1	Macro Definition Documentation	180
5.19.1.1	ADATOUCH_FLIP_X	180
5.19.1.2	ADATOUCH_FLIP_Y	180
5.19.1.3	ADATOUCH_I2C_ADDR	180
5.19.1.4	ADATOUCH_I2C_HW	180
5.19.1.5	ADATOUCH_PIN_CS	180

5.19.1.6	ADATOUCH_SPI_HW	180
5.19.1.7	ADATOUCH_SPI_SW	180
5.19.1.8	ADATOUCH_SWAP_XY	180
5.19.1.9	ADATOUCH_X_MAX	180
5.19.1.10	ADATOUCH_X_MIN	180
5.19.1.11	ADATOUCH_Y_MAX	180
5.19.1.12	ADATOUCH_Y_MIN	180
5.19.1.13	DEBUG_ERR	180
5.19.1.14	DRV_DISP_TFT_ESPI	180
5.19.1.15	DRV_TOUCH_ADA_STMPE610	180
5.19.1.16	GSLC_BMP_TRANS_EN	180
5.19.1.17	GSLC_BMP_TRANS_RGB	180
5.19.1.18	GSLC_CLIP_EN	180
5.19.1.19	GSLC_DEV_TOUCH	180
5.19.1.20	GSLC_FEATURE_COMPOUND	180
5.19.1.21	GSLC_FEATURE_XGAUGE_RADIAL	180
5.19.1.22	GSLC_FEATURE_XGAUGE_RAMP	180
5.19.1.23	GSLC_LOCAL_STR	180
5.19.1.24	GSLC_LOCAL_STR_LEN	181
5.19.1.25	GSLC_ROTATE	181
5.19.1.26	GSLC_SD_EN	181
5.19.1.27	GSLC_TOUCH_MAX_EVT	181
5.19.1.28	GSLC_USE_FLOAT	181
5.19.1.29	GSLC_USE_PROGMEM	181
5.20	src/GUIslice_config_linux.h File Reference	181
5.20.1	Macro Definition Documentation	181
5.20.1.1	ADATOUCH_FLIP_X	181
5.20.1.2	ADATOUCH_FLIP_Y	181
5.20.1.3	ADATOUCH_SWAP_XY	181
5.20.1.4	DEBUG_ERR	181
5.20.1.5	DRV_DISP_SDL1	181
5.20.1.6	DRV_SDL_FIX_START	181
5.20.1.7	DRV_SDL_MOUSE_SHOW	181
5.20.1.8	DRV_TOUCH_TSLIB	182
5.20.1.9	GSLC_BMP_TRANS_EN	182
5.20.1.10	GSLC_BMP_TRANS_RGB	182
5.20.1.11	GSLC_DEV_FB	182
5.20.1.12	GSLC_DEV_TOUCH	182
5.20.1.13	GSLC_DEV_VID_DRV	182
5.20.1.14	GSLC_FEATURE_COMPOUND	182

5.20.1.15	GSLC_FEATURE_XGAUGE_RADIAL	182
5.20.1.16	GSLC_FEATURE_XGAUGE_RAMP	182
5.20.1.17	GSLC_LOCAL_STR	182
5.20.1.18	GSLC_LOCAL_STR_LEN	182
5.20.1.19	GSLC_TOUCH_MAX_EVT	182
5.20.1.20	GSLC_USE_FLOAT	182
5.20.1.21	GSLC_USE_PROGMEM	182
5.21	src/GUISlice_drv.h File Reference	182
5.22	src/GUISlice_drv_adagfx.cpp File Reference	183
5.23	src/GUISlice_drv_adagfx.h File Reference	183
5.23.1	Macro Definition Documentation	185
5.23.1.1	DRV_HAS_DRAW_CIRCLE_FILL	185
5.23.1.2	DRV_HAS_DRAW_CIRCLE_FRAME	185
5.23.1.3	DRV_HAS_DRAW_LINE	185
5.23.1.4	DRV_HAS_DRAW_POINT	186
5.23.1.5	DRV_HAS_DRAW_POINTS	186
5.23.1.6	DRV_HAS_DRAW_RECT_FILL	186
5.23.1.7	DRV_HAS_DRAW_RECT_FRAME	186
5.23.1.8	DRV_HAS_DRAW_TEXT	186
5.23.1.9	DRV_HAS_DRAW_TRI_FILL	186
5.23.1.10	DRV_HAS_DRAW_TRI_FRAME	186
5.23.1.11	DRV_OVERRIDE_TXT_ALIGN	186
5.23.2	Function Documentation	186
5.23.2.1	gslc_DrvAdaptColorToRaw	186
5.23.2.2	gslc_DrvDestruct	186
5.23.2.3	gslc_DrvDrawBkgnd	187
5.23.2.4	gslc_DrvDrawFillCircle	188
5.23.2.5	gslc_DrvDrawFillRect	188
5.23.2.6	gslc_DrvDrawFillTriangle	188
5.23.2.7	gslc_DrvDrawFrameCircle	189
5.23.2.8	gslc_DrvDrawFrameRect	189
5.23.2.9	gslc_DrvDrawFrameTriangle	189
5.23.2.10	gslc_DrvDrawImage	189
5.23.2.11	gslc_DrvDrawLine	190
5.23.2.12	gslc_DrvDrawMonoFromMem	190
5.23.2.13	gslc_DrvDrawPoint	190
5.23.2.14	gslc_DrvDrawPoints	191
5.23.2.15	gslc_DrvDrawTxt	191
5.23.2.16	gslc_DrvFontAdd	191
5.23.2.17	gslc_DrvFontsDestruct	192

5.23.2.18	gslc_DrvGetTouch	192
5.23.2.19	gslc_DrvGetTxtSize	192
5.23.2.20	gslc_DrvImageDestruct	193
5.23.2.21	gslc_DrvInit	194
5.23.2.22	gslc_DrvInitTouch	194
5.23.2.23	gslc_DrvInitTs	194
5.23.2.24	gslc_DrvLoadImage	195
5.23.2.25	gslc_DrvPageFlipNow	195
5.23.2.26	gslc_DrvRotateSwapFlip	195
5.23.2.27	gslc_DrvSetBkgndColor	195
5.23.2.28	gslc_DrvSetBkgndImage	196
5.23.2.29	gslc_DrvSetClipRect	196
5.23.2.30	gslc_DrvSetElemImageGlow	196
5.23.2.31	gslc_DrvSetElemImageNorm	196
5.24	src/GUIslice_drv_m5stack.cpp File Reference	197
5.25	src/GUIslice_drv_m5stack.h File Reference	197
5.25.1	Macro Definition Documentation	199
5.25.1.1	DRV_HAS_DRAW_CIRCLE_FILL	199
5.25.1.2	DRV_HAS_DRAW_CIRCLE_FRAME	200
5.25.1.3	DRV_HAS_DRAW_LINE	200
5.25.1.4	DRV_HAS_DRAW_POINT	200
5.25.1.5	DRV_HAS_DRAW_POINTS	200
5.25.1.6	DRV_HAS_DRAW_RECT_FILL	200
5.25.1.7	DRV_HAS_DRAW_RECT_FRAME	200
5.25.1.8	DRV_HAS_DRAW_TEXT	200
5.25.1.9	DRV_HAS_DRAW_TRI_FILL	200
5.25.1.10	DRV_HAS_DRAW_TRI_FRAME	200
5.25.1.11	DRV_OVERRIDE_TXT_ALIGN	200
5.25.2	Function Documentation	200
5.25.2.1	gslc_DrvAdaptColorToRaw	200
5.25.2.2	gslc_DrvDestruct	200
5.25.2.3	gslc_DrvDrawBkgnd	201
5.25.2.4	gslc_DrvDrawFillCircle	201
5.25.2.5	gslc_DrvDrawFillRect	201
5.25.2.6	gslc_DrvDrawFillTriangle	201
5.25.2.7	gslc_DrvDrawFrameCircle	202
5.25.2.8	gslc_DrvDrawFrameRect	202
5.25.2.9	gslc_DrvDrawFrameTriangle	202
5.25.2.10	gslc_DrvDrawImage	203
5.25.2.11	gslc_DrvDrawLine	203

5.25.2.12	gslc_DrvDrawMonoFromMem	203
5.25.2.13	gslc_DrvDrawPoint	204
5.25.2.14	gslc_DrvDrawPoints	204
5.25.2.15	gslc_DrvDrawTxt	204
5.25.2.16	gslc_DrvDrawTxtAlign	205
5.25.2.17	gslc_DrvFontAdd	205
5.25.2.18	gslc_DrvFontsDestruct	205
5.25.2.19	gslc_DrvGetTxtSize	205
5.25.2.20	gslc_DrvImageDestruct	206
5.25.2.21	gslc_DrvInit	206
5.25.2.22	gslc_DrvInitTs	206
5.25.2.23	gslc_DrvLoadImage	207
5.25.2.24	gslc_DrvPageFlipNow	207
5.25.2.25	gslc_DrvRotateSwapFlip	207
5.25.2.26	gslc_DrvSetBkgndColor	208
5.25.2.27	gslc_DrvSetBkgndImage	208
5.25.2.28	gslc_DrvSetClipRect	208
5.25.2.29	gslc_DrvSetElemImageGlow	208
5.25.2.30	gslc_DrvSetElemImageNorm	209
5.26	src/GUISlice_drv_sdl.c File Reference	210
5.27	src/GUISlice_drv_sdl.h File Reference	210
5.27.1	Macro Definition Documentation	212
5.27.1.1	DRV_HAS_DRAW_POINT	212
5.27.1.2	DRV_OVERRIDE_TXT_ALIGN	212
5.27.2	Function Documentation	212
5.27.2.1	gslc_DrvAdaptColor	212
5.27.2.2	gslc_DrvAdaptRect	212
5.27.2.3	gslc_DrvCleanStart	213
5.27.2.4	gslc_DrvDestruct	214
5.27.2.5	gslc_DrvDrawBkgnd	214
5.27.2.6	gslc_DrvDrawFillRect	214
5.27.2.7	gslc_DrvDrawFrameRect	214
5.27.2.8	gslc_DrvDrawImage	215
5.27.2.9	gslc_DrvDrawLine	215
5.27.2.10	gslc_DrvDrawPoint	215
5.27.2.11	gslc_DrvDrawPoints	216
5.27.2.12	gslc_DrvDrawTxt	217
5.27.2.13	gslc_DrvFontAdd	217
5.27.2.14	gslc_DrvFontsDestruct	217
5.27.2.15	gslc_DrvGetTouch	218

5.27.2.16	gslc_DrvGetTxtSize	218
5.27.2.17	gslc_DrvImageDestruct	218
5.27.2.18	gslc_DrvInit	219
5.27.2.19	gslc_DrvInitTouch	219
5.27.2.20	gslc_DrvLoadImage	219
5.27.2.21	gslc_DrvPageFlipNow	219
5.27.2.22	gslc_DrvReportInfoPost	220
5.27.2.23	gslc_DrvReportInfoPre	220
5.27.2.24	gslc_DrvSetBkgndColor	220
5.27.2.25	gslc_DrvSetBkgndImage	220
5.27.2.26	gslc_DrvSetClipRect	220
5.27.2.27	gslc_DrvSetElemImageGlow	221
5.27.2.28	gslc_DrvSetElemImageNorm	221
5.28	src/GUIslice_drv_tft_espi.cpp File Reference	221
5.29	src/GUIslice_drv_tft_espi.h File Reference	222
5.29.1	Macro Definition Documentation	224
5.29.1.1	DRV_HAS_DRAW_CIRCLE_FILL	224
5.29.1.2	DRV_HAS_DRAW_CIRCLE_FRAME	224
5.29.1.3	DRV_HAS_DRAW_LINE	224
5.29.1.4	DRV_HAS_DRAW_POINT	224
5.29.1.5	DRV_HAS_DRAW_POINTS	224
5.29.1.6	DRV_HAS_DRAW_RECT_FILL	224
5.29.1.7	DRV_HAS_DRAW_RECT_FRAME	225
5.29.1.8	DRV_HAS_DRAW_TEXT	225
5.29.1.9	DRV_HAS_DRAW_TRI_FILL	225
5.29.1.10	DRV_HAS_DRAW_TRI_FRAME	225
5.29.1.11	DRV_OVERRIDE_TXT_ALIGN	225
5.29.2	Function Documentation	225
5.29.2.1	gslc_DrvAdaptColorToRaw	225
5.29.2.2	gslc_DrvDestruct	225
5.29.2.3	gslc_DrvDrawBkgnd	225
5.29.2.4	gslc_DrvDrawFillCircle	225
5.29.2.5	gslc_DrvDrawFillRect	226
5.29.2.6	gslc_DrvDrawFillTriangle	226
5.29.2.7	gslc_DrvDrawFrameCircle	226
5.29.2.8	gslc_DrvDrawFrameRect	227
5.29.2.9	gslc_DrvDrawFrameTriangle	227
5.29.2.10	gslc_DrvDrawImage	227
5.29.2.11	gslc_DrvDrawLine	228
5.29.2.12	gslc_DrvDrawMonoFromMem	228

5.29.2.13	gslc_DrvDrawPoint	228
5.29.2.14	gslc_DrvDrawPoints	229
5.29.2.15	gslc_DrvDrawTxt	229
5.29.2.16	gslc_DrvDrawTxtAlign	229
5.29.2.17	gslc_DrvFontAdd	230
5.29.2.18	gslc_DrvFontsDestruct	230
5.29.2.19	gslc_DrvGetTouch	230
5.29.2.20	gslc_DrvGetTxtSize	231
5.29.2.21	gslc_DrvImageDestruct	232
5.29.2.22	gslc_DrvInit	232
5.29.2.23	gslc_DrvInitTouch	232
5.29.2.24	gslc_DrvInitTs	233
5.29.2.25	gslc_DrvLoadImage	233
5.29.2.26	gslc_DrvPageFlipNow	233
5.29.2.27	gslc_DrvRotateSwapFlip	233
5.29.2.28	gslc_DrvSetBkgndColor	234
5.29.2.29	gslc_DrvSetBkgndImage	234
5.29.2.30	gslc_DrvSetClipRect	234
5.29.2.31	gslc_DrvSetElemImageGlow	235
5.29.2.32	gslc_DrvSetElemImageNorm	236
5.30	src/GUISlice_ex.c File Reference	236
5.30.1	Function Documentation	238
5.30.1.1	gslc_ElemXCheckboxCreate	238
5.30.1.2	gslc_ElemXCheckboxDraw	239
5.30.1.3	gslc_ElemXCheckboxFindChecked	239
5.30.1.4	gslc_ElemXCheckboxGetState	239
5.30.1.5	gslc_ElemXCheckboxSetState	240
5.30.1.6	gslc_ElemXCheckboxSetStateHelp	241
5.30.1.7	gslc_ElemXCheckboxToggleState	241
5.30.1.8	gslc_ElemXCheckboxTouch	241
5.30.1.9	gslc_ElemXGaugeCreate	241
5.30.1.10	gslc_ElemXGaugeDraw	242
5.30.1.11	gslc_ElemXGaugeDrawProgressBar	242
5.30.1.12	gslc_ElemXGaugeSetFlip	242
5.30.1.13	gslc_ElemXGaugeSetIndicator	243
5.30.1.14	gslc_ElemXGaugeSetStyle	243
5.30.1.15	gslc_ElemXGaugeSetTicks	243
5.30.1.16	gslc_ElemXGaugeUpdate	244
5.30.1.17	gslc_ElemXGraphAdd	244
5.30.1.18	gslc_ElemXGraphCreate	244

5.30.1.19	gslc_ElemXGraphDraw	245
5.30.1.20	gslc_ElemXGraphScrollSet	245
5.30.1.21	gslc_ElemXGraphSetRange	245
5.30.1.22	gslc_ElemXGraphSetStyle	246
5.30.1.23	gslc_ElemXSliderCreate	246
5.30.1.24	gslc_ElemXSliderDraw	246
5.30.1.25	gslc_ElemXSliderGetPos	247
5.30.1.26	gslc_ElemXSliderSetPos	247
5.30.1.27	gslc_ElemXSliderSetPosFunc	247
5.30.1.28	gslc_ElemXSliderSetStyle	247
5.30.1.29	gslc_ElemXSliderTouch	248
5.30.1.30	gslc_ElemXTextboxAdd	248
5.30.1.31	gslc_ElemXTextboxBufAdd	248
5.30.1.32	gslc_ElemXTextboxColReset	248
5.30.1.33	gslc_ElemXTextboxColSet	249
5.30.1.34	gslc_ElemXTextboxCreate	249
5.30.1.35	gslc_ElemXTextboxDraw	249
5.30.1.36	gslc_ElemXTextboxLineWrAdv	250
5.30.1.37	gslc_ElemXTextboxReset	250
5.30.1.38	gslc_ElemXTextboxScrollSet	250
5.30.1.39	gslc_ElemXTextboxWrapSet	250
5.30.2	Variable Documentation	251
5.30.2.1	ERRSTR_NULL	251
5.30.2.2	ERRSTR_PXD_NULL	251
5.31	src/GUIslice_ex.h File Reference	251
5.31.1	Macro Definition Documentation	254
5.31.1.1	gslc_ElemXCheckboxCreate_P	254
5.31.1.2	gslc_ElemXGaugeCreate_P	254
5.31.1.3	gslc_ElemXSliderCreate_P	255
5.31.1.4	GSLC_XTEXTBOX_CODE_COL_RESET	255
5.31.1.5	GSLC_XTEXTBOX_CODE_COL_SET	255
5.31.2	Typedef Documentation	255
5.31.2.1	GSLC_CB_XSLIDER_POS	256
5.31.3	Enumeration Type Documentation	256
5.31.3.1	gslc_teTypeExtend	256
5.31.3.2	gslc_teXCheckboxStyle	256
5.31.3.3	gslc_teXGaugeStyle	256
5.31.3.4	gslc_teXGraphStyle	256
5.31.4	Function Documentation	257
5.31.4.1	gslc_ElemXCheckboxCreate	257

5.31.4.2	gslc_ElemXCheckboxDraw	258
5.31.4.3	gslc_ElemXCheckboxFindChecked	258
5.31.4.4	gslc_ElemXCheckboxGetState	258
5.31.4.5	gslc_ElemXCheckboxSetState	259
5.31.4.6	gslc_ElemXCheckboxToggleState	259
5.31.4.7	gslc_ElemXCheckboxTouch	259
5.31.4.8	gslc_ElemXGaugeCreate	260
5.31.4.9	gslc_ElemXGaugeDraw	260
5.31.4.10	gslc_ElemXGaugeDrawProgressBar	260
5.31.4.11	gslc_ElemXGaugeSetFlip	261
5.31.4.12	gslc_ElemXGaugeSetIndicator	261
5.31.4.13	gslc_ElemXGaugeSetStyle	261
5.31.4.14	gslc_ElemXGaugeSetTicks	262
5.31.4.15	gslc_ElemXGaugeUpdate	262
5.31.4.16	gslc_ElemXGraphAdd	262
5.31.4.17	gslc_ElemXGraphCreate	263
5.31.4.18	gslc_ElemXGraphDraw	263
5.31.4.19	gslc_ElemXGraphScrollSet	263
5.31.4.20	gslc_ElemXGraphSetRange	264
5.31.4.21	gslc_ElemXGraphSetStyle	264
5.31.4.22	gslc_ElemXSliderCreate	264
5.31.4.23	gslc_ElemXSliderDraw	265
5.31.4.24	gslc_ElemXSliderGetPos	265
5.31.4.25	gslc_ElemXSliderSetPos	265
5.31.4.26	gslc_ElemXSliderSetPosFunc	265
5.31.4.27	gslc_ElemXSliderSetStyle	266
5.31.4.28	gslc_ElemXSliderTouch	267
5.31.4.29	gslc_ElemXTextboxAdd	267
5.31.4.30	gslc_ElemXTextboxColReset	267
5.31.4.31	gslc_ElemXTextboxColSet	268
5.31.4.32	gslc_ElemXTextboxCreate	268
5.31.4.33	gslc_ElemXTextboxDraw	268
5.31.4.34	gslc_ElemXTextboxReset	269
5.31.4.35	gslc_ElemXTextboxScrollSet	269
5.31.4.36	gslc_ElemXTextboxWrapSet	269

Chapter 1

GUIslice library

A lightweight GUI framework suitable for embedded displays

- [Website \(www.impulseedventure.com\)](http://www.impulseedventure.com)
- [Documentation wiki \(github\)](#)
- [Release notes](#)

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.
- *Platform-independent GUI core currently supports*: SDL1.2, SDL2.0, Adafruit-GFX, TFT_eSPI
- *Devices*: Raspberry Pi, Arduino, ESP8266 / NodeMCU, ESP32, Cortex M0 (Feather M0), LINUX, Beaglebone Black, M5Stack
- *Typical displays*: PiTFT, Waveshare, Adafruit TFT 3.5" / 2.8" / 2.4" / 2.2" / 1.44", OLED 0.96", 4D Cape
- *Display drivers include*: ILI9341, ST7735, SSD1306, HX8357
- *Touchscreen control including*: STMPE610, FT6206, XPT2046, tslib
- No GUIslice installation – just add include files and go!
- *LINUX Dependencies*: sdl, sdl-ttf, optional: tslib
- *Arduino Dependencies*: TFT_eSPI or Adafruit-GFX plus display / touch driver libraries

Screenshots

Device Configuration

- The following table lists a number of devices that have been tested with GUIslice and the recommended configuration modes and test examples.

Important Note for Arduino Users

- The baseline Arduino (**ATmega328P**) devices have very limited SRAM memory (2KB SRAM, 32KB FLASH). Therefore, it is important that GUI elements are stored in FLASH whenever possible. A set of examples that demonstrate this method are located in `\arduino_min`. The examples in `\arduino` don't use the FLASH optimizations and are less likely to run on these limited devices.
- Other Arduino variants and devices such as **ATmega2560** (8KB SRAM, 256KB FLASH), **ESP8266**, **NodeMCU**, **Feather M0**, etc. tend to work much better as there is far more SRAM and FLASH available.
- By default, `DEBUG_ERR` is enabled on many devices to provide error messages via the Serial Monitor interface. However, if further reduction of FLASH memory is necessary, disable `DEBUG_ERR` in the `GUIslice_config*.h`.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

gslc_tsCollect	Element collection struct	7
gslc_tsColor	Color structure. Defines RGB triplet	9
gslc_tsDriver	10
gslc_tsElem	Element Struct	10
gslc_tsElemRef	Element reference structure	14
gslc_tsEvent	Event structure	15
gslc_tsEventTouch	Structure used to pass touch data through event	16
gslc_tsFont	Font reference structure	17
gslc_tsGui	GUI structure	18
gslc_tsImgRef	Image reference structure	21
gslc_tsPage	Page structure	22
gslc_tsPt	Define point coordinates	24
gslc_tsRect	Rectangular region. Defines X,Y corner coordinates plus dimensions	25
gslc_tsXCheckbox	Extended data for Checkbox element	25
gslc_tsXGauge	Extended data for Gauge element	27
gslc_tsXGraph	Extended data for Graph element	29
gslc_tsXSlider	Extended data for Slider element	31
gslc_tsXTextbox	Extended data for Textbox element	33

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

linux/gslc-ex01.c	37
linux/gslc-ex02.c	38
linux/gslc-ex03.c	40
linux/gslc-ex04.c	43
linux/gslc-ex05.c	45
linux/gslc-ex06.c	48
linux/gslc-ex07.c	51
linux/gslc-ex08.c	54
linux/gslc-ex09.c	57
linux/gslc-ex10.c	60
linux/gslc-ex11.c	63
linux/test-sdl1.c	65
linux/test-sdl2.c	66
src/GUISlice.c	67
src/GUISlice.h	112
src/GUISlice_config.h	177
src/GUISlice_config_ard.h	177
src/GUISlice_config_esp.h	179
src/GUISlice_config_linux.h	181
src/GUISlice_drv.h	182
src/GUISlice_drv_adagfx.cpp	183
src/GUISlice_drv_adagfx.h	183
src/GUISlice_drv_m5stack.cpp	197
src/GUISlice_drv_m5stack.h	197
src/GUISlice_drv_sdl.c	210
src/GUISlice_drv_sdl.h	210
src/GUISlice_drv_tft_espi.cpp	221
src/GUISlice_drv_tft_espi.h	222
src/GUISlice_ex.c	236
src/GUISlice_ex.h	251

Chapter 4

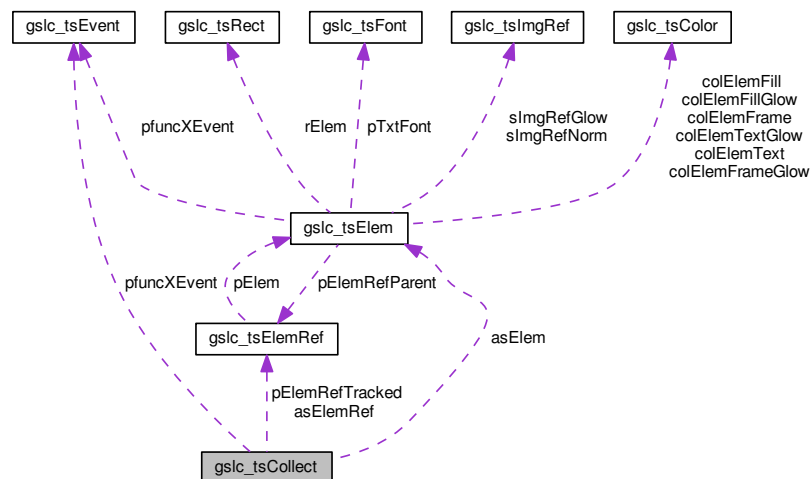
Class Documentation

4.1 gslc_tsCollect Struct Reference

Element collection struct.

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

Collaboration diagram for gslc_tsCollect:



Public Attributes

- `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)
- [GSLC_CB_EVENT](#) [pfuncXEvent](#)
Callback func ptr for events.

4.1.1 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

4.1.2 Member Data Documentation

4.1.2.1 [gslc_tsElem](#)* [gslc_tsCollect::asElem](#)

Array of elements.

4.1.2.2 [gslc_tsElemRef](#)* [gslc_tsCollect::asElemRef](#)

Array of element references.

4.1.2.3 int16_t [gslc_tsCollect::nElemAutoldNext](#)

Next Element ID for auto-assignment.

4.1.2.4 uint16_t [gslc_tsCollect::nElemCnt](#)

Number of elements allocated.

4.1.2.5 uint16_t [gslc_tsCollect::nElemMax](#)

Maximum number of elements to allocate (in RAM)

4.1.2.6 uint16_t [gslc_tsCollect::nElemRefCnt](#)

Number of element references allocated.

4.1.2.7 uint16_t [gslc_tsCollect::nElemRefMax](#)

Maximum number of element references to allocate.

4.1.2.8 gslc_tsElemRef* gslc_tsCollect::pElemRefTracked

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

4.1.2.9 GSLC_CB_EVENT gslc_tsCollect::pfuncXEvent

Callback func ptr for events.

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

- [src/GUIslice.h](#)

4.2 gslc_tsColor Struct Reference

Color structure. Defines RGB triplet.

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

Public Attributes

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

4.2.1 Detailed Description

Color structure. Defines RGB triplet.

4.2.2 Member Data Documentation

4.2.2.1 `uint8_t gslc_tsColor::b`

RGB blue value.

4.2.2.2 `uint8_t gslc_tsColor::g`

RGB green value.

4.2.2.3 `uint8_t gslc_tsColor::r`

RGB red value.

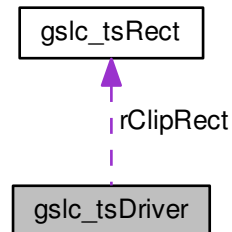
The documentation for this struct was generated from the following file:

- [src/GUIslice.h](#)

4.3 gslc_tsDriver Struct Reference

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

Collaboration diagram for `gslc_tsDriver`:



Public Attributes

- `uint16_t nColRawBkgnd`
Background color (if not image-based)
- `gslc_tsRect rClipRect`
Clipping rectangle.

4.3.1 Member Data Documentation

4.3.1.1 `uint16_t gslc_tsDriver::nColRawBkgnd`

Background color (if not image-based)

4.3.1.2 `gslc_tsRect gslc_tsDriver::rClipRect`

Clipping rectangle.

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

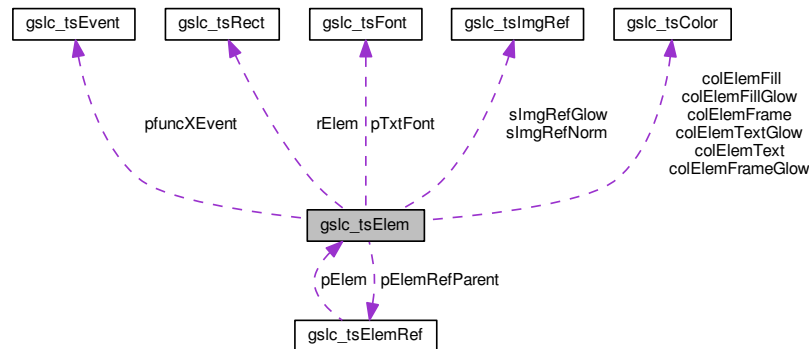
- `src/GUIslice_drv_adagfx.h`
- `src/GUIslice_drv_m5stack.h`
- `src/GUIslice_drv_tft_espi.h`

4.4 gslc_tsElem Struct Reference

Element Struct.

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

Collaboration diagram for gslc_tsElem:



Public Attributes

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

- [gslc_tsColor colElemTextGlow](#)
Color of overlay text when glowing.
- [int8_t eTxtAlign](#)
Alignment of overlay text.
- [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](#)
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.

4.4.1 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.

4.4.2 Member Data Documentation

4.4.2.1 `gslc_tsColor gslc_tsElem::colElemFill`

Color for background fill.

4.4.2.2 `gslc_tsColor gslc_tsElem::colElemFillGlow`

Color to use for fill when glowing.

4.4.2.3 `gslc_tsColor gslc_tsElem::colElemFrame`

Color for frame.

4.4.2.4 `gslc_tsColor gslc_tsElem::colElemFrameGlow`

Color to use for frame when glowing.

4.4.2.5 gslc_tsColor gslc_tsElem::colElemText

Color of overlay text.

4.4.2.6 gslc_tsColor gslc_tsElem::colElemTextGlow

Color of overlay text when glowing.

4.4.2.7 int8_t gslc_tsElem::eTxtAlign

Alignment of overlay text.

4.4.2.8 gslc_teTxtFlags gslc_tsElem::eTxtFlags

Flags associated with text buffer.

4.4.2.9 uint8_t gslc_tsElem::nFeatures

Element feature vector (appearance/behavior)

4.4.2.10 int16_t gslc_tsElem::nGroup

Group ID that the element belongs to.

4.4.2.11 int16_t gslc_tsElem::nId

Element ID specified by user.

4.4.2.12 uint8_t gslc_tsElem::nStrBufMax

Size of string buffer.

4.4.2.13 uint8_t gslc_tsElem::nTxtMargin

Margin of overlay text within rect region.

4.4.2.14 int16_t gslc_tsElem::nType

Element type enumeration.

4.4.2.15 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.

4.4.2.16 GSLC_CB_DRAW gslc_tsElem::pfuncXDraw

Callback func ptr for custom drawing.

4.4.2.17 **GSLC_CB_EVENT** `gslc_tsElem::pfuncXEvent`

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

4.4.2.18 **GSLC_CB_TICK** `gslc_tsElem::pfuncXTick`

Callback func ptr for timer/main loop tick.

4.4.2.19 **GSLC_CB_TOUCH** `gslc_tsElem::pfuncXTouch`

Callback func ptr for touch.

4.4.2.20 `char*` `gslc_tsElem::pStrBuf`

Ptr to text string buffer to overlay.

4.4.2.21 `gslc_tsFont*` `gslc_tsElem::pTxtFont`

Ptr to Font for overlay text.

4.4.2.22 `void*` `gslc_tsElem::pXData`

Ptr to extended data structure.

4.4.2.23 `gslc_tsRect` `gslc_tsElem::rElem`

Rect region containing element.

4.4.2.24 `gslc_tsImgRef` `gslc_tsElem::sImgRefGlow`

Image reference to draw (glowing)

4.4.2.25 `gslc_tsImgRef` `gslc_tsElem::sImgRefNorm`

Image reference to draw (normal)

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

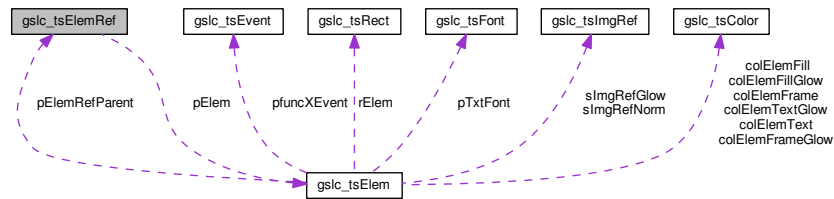
- [src/GUISlice.h](#)

4.5 `gslc_tsElemRef` Struct Reference

Element reference structure.

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


Collaboration diagram for gslc_tsElemRef:



Public Attributes

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

4.5.1 Detailed Description

Element reference structure.

4.5.2 Member Data Documentation

4.5.2.1 [gslc_teElemRefFlags](#) [gslc_tsElemRef::eElemFlags](#)

Element reference flags.

4.5.2.2 [gslc_tsElem](#)* [gslc_tsElemRef::pElem](#)

Pointer to element in memory [RAM,FLASH].

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

- [src/GUIslice.h](#)

4.6 gslc_tsEvent Struct Reference

Event structure.

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

Public Attributes

- [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.

4.6.1 Detailed Description

Event structure.

4.6.2 Member Data Documentation

4.6.2.1 `gslc_teEventType` `gslc_tsEvent::eType`

Event type.

4.6.2.2 `uint8_t` `gslc_tsEvent::nSubType`

Event sub-type.

4.6.2.3 `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.

4.6.2.4 `void*` `gslc_tsEvent::pvScope`

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

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

- `src/GUIslice.h`

4.7 `gslc_tsEventTouch` Struct Reference

Structure used to pass touch data through event.

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

Public Attributes

- `gslc_teTouch eTouch`
Touch state.
- `int16_t nX`
Touch X coordinate (absolute)
- `int16_t nY`
Touch Y coordinate (absolute)

4.7.1 Detailed Description

Structure used to pass touch data through event.

4.7.2 Member Data Documentation

4.7.2.1 gslc_teTouch gslc_tsEventTouch::eTouch

Touch state.

4.7.2.2 int16_t gslc_tsEventTouch::nX

Touch X coordinate (absolute)

4.7.2.3 int16_t gslc_tsEventTouch::nY

Touch Y coordinate (absolute)

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

- [src/GUISlice.h](#)

4.8 gslc_tsFont Struct Reference

Font reference structure.

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

Public Attributes

- [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.

4.8.1 Detailed Description

Font reference structure.

4.8.2 Member Data Documentation

4.8.2.1 gslc_teFontRefType gslc_tsFont::eFontRefType

Font reference type.

4.8.2.2 int16_t gslc_tsFont::nId

Font ID specified by user.

- Number of fonts allocated.*
- [gslc_tsElem sElemTmpProg](#)
Temporary element for Flash compatibility.
- `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)
- `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 pages.
- `uint8_t nPageMax`
Maximum number of pages.
- `uint8_t nPageCnt`
Current page index.
- [gslc_tsPage * pCurPage](#)
Currently active page.
- [gslc_tsCollect * pCurPageCollect](#)
Ptr to active page collection.
- [GSLC_CB_EVENT pfuncXEvent](#)
Callback func ptr for events.

4.9.1 Detailed Description

GUI structure.

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

4.9.2 Member Data Documentation

4.9.2.1 `gslc_tsFont* gslc_tsGui::asFont`

Collection of loaded fonts.

4.9.2.2 `gslc_tsPage* gslc_tsGui::asPage`

Array of pages.

4.9.2.3 `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.

4.9.2.4 `uint8_t gslc_tsGui::nDispDepth`

Bit depth of display (bits per pixel)

4.9.2.5 `uint16_t gslc_tsGui::nDispH`

Height of the display (pixels)

4.9.2.6 `uint16_t gslc_tsGui::nDispW`

Width of the display (pixels)

4.9.2.7 `uint8_t gslc_tsGui::nFontCnt`

Number of fonts allocated.

4.9.2.8 `uint8_t gslc_tsGui::nFontMax`

Maximum number of fonts to allocate.

4.9.2.9 `uint8_t gslc_tsGui::nFrameRateCnt`

Diagnostic frame rate count.

4.9.2.10 `uint8_t gslc_tsGui::nFrameRateStart`

Diagnostic frame rate timestamp.

4.9.2.11 `uint8_t gslc_tsGui::nPageCnt`

Current page index.

4.9.2.12 `uint8_t gslc_tsGui::nPageMax`

Maximum number of pages.

4.9.2.13 `uint16_t gslc_tsGui::nTouchLastPress`

Last touch event pressure (0=none))

4.9.2.14 `int16_t gslc_tsGui::nTouchLastX`

Last touch event X coord.

4.9.2.15 int16_t gslc_tsGui::nTouchLastY

Last touch event Y coord.

4.9.2.16 gslc_tsPage* gslc_tsGui::pCurPage

Currently active page.

4.9.2.17 gslc_tsCollect* gslc_tsGui::pCurPageCollect

Ptr to active page collection.

4.9.2.18 GSLC_CB_EVENT gslc_tsGui::pfuncXEvent

Callback func ptr for events.

4.9.2.19 void* gslc_tsGui::pvDriver

Driver-specific members (gslc_tsDriver*)

4.9.2.20 gslc_tsElem gslc_tsGui::sElemTmpProg

Temporary element for Flash compatibility.

4.9.2.21 gslc_tsImgRef gslc_tsGui::sImgRefBkgnd

Image reference for background.

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

- [src/GUISlice.h](#)

4.10 gslc_tsImgRef Struct Reference

Image reference structure.

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

Public Attributes

- 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](#) [elmgFlags](#)
Image reference flags.
- void * [pvImgRaw](#)
Ptr to raw output image data (for pre-loaded images)

4.10.1 Detailed Description

Image reference structure.

4.10.2 Member Data Documentation

4.10.2.1 `gslc_telmRefFlags` `gslc_tslmgRef::elmRefFlags`

Image reference flags.

4.10.2.2 `const char*` `gslc_tslmgRef::pFname`

Pathname to input image file [FILE,SD].

4.10.2.3 `const unsigned char*` `gslc_tslmgRef::pImgBuf`

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

4.10.2.4 `void*` `gslc_tslmgRef::pVImgRaw`

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

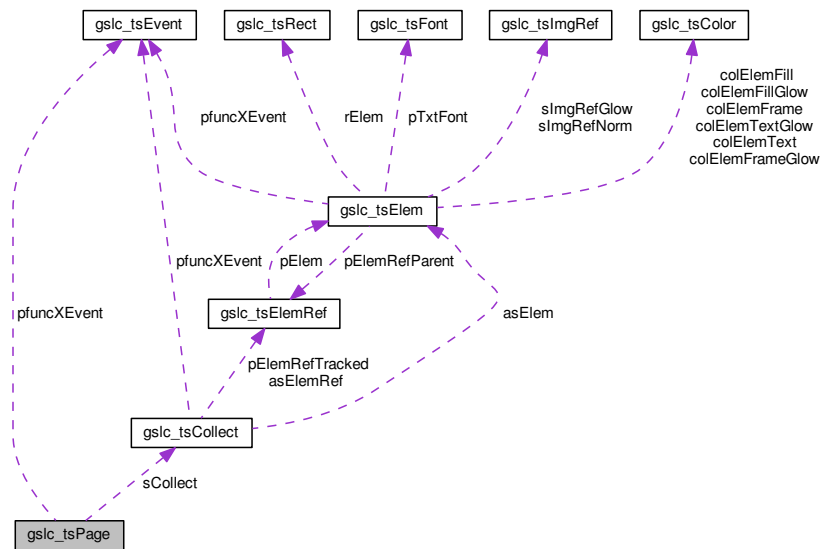
The documentation for this struct was generated from the following file:

- [src/GUIslice.h](#)

4.11 `gslc_tsPage` Struct Reference

Page structure.

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

- `gslc_tsCollect sCollect`
Collection of elements on page.
- `int8_t nPageId`
Page identifier.
- `bool bPageNeedRedraw`
Page require a redraw.
- `bool bPageNeedFlip`
Screen requires a page flip.
- `GSLC_CB_EVENT pfuncXEvent`
Callback func ptr for events.

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

4.11.2.1 bool gslc_tsPage::bPageNeedFlip

Screen requires a page flip.

4.11.2.2 `bool gslc_tsPage::bPageNeedRedraw`

Page require a redraw.

4.11.2.3 `int8_t gslc_tsPage::nPageId`

Page identifier.

4.11.2.4 `GSLC_CB_EVENT gslc_tsPage::pfuncXEvent`

Callback func ptr for events.

4.11.2.5 `gslc_tsCollect gslc_tsPage::sCollect`

Collection of elements on page.

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

- [src/GUISlice.h](#)

4.12 `gslc_tsPt` Struct Reference

Define point coordinates.

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

Public Attributes

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

4.12.1 Detailed Description

Define point coordinates.

4.12.2 Member Data Documentation

4.12.2.1 `int16_t gslc_tsPt::x`

X coordinate.

4.12.2.2 `int16_t gslc_tsPt::y`

Y coordinate.

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

- [src/GUISlice.h](#)

4.13 gslc_tsRect Struct Reference

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

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

Public Attributes

- `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.

4.13.1 Detailed Description

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

4.13.2 Member Data Documentation

4.13.2.1 `uint16_t gslc_tsRect::h`

Height of region.

4.13.2.2 `uint16_t gslc_tsRect::w`

Width of region.

4.13.2.3 `int16_t gslc_tsRect::x`

X coordinate of corner.

4.13.2.4 `int16_t gslc_tsRect::y`

Y coordinate of corner.

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

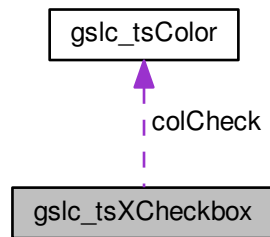
- `src/GUIslice.h`

4.14 gslc_tsXCheckbox Struct Reference

Extended data for Checkbox element.

```
#include <GUIslice_ex.h>
```

Collaboration diagram for `gslc_tsXCheckbox`:



Public Attributes

- 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.

4.14.1 Detailed Description

Extended data for Checkbox element.

4.14.2 Member Data Documentation

4.14.2.1 bool `gslc_tsXCheckbox::bChecked`

Indicates if it is selected (checked)

4.14.2.2 bool `gslc_tsXCheckbox::bRadio`

Radio-button operation if true.

4.14.2.3 `gslc_tsColor` `gslc_tsXCheckbox::colCheck`

Color of checked inner fill.

4.14.2.4 `gslc_teXCheckboxStyle` `gslc_tsXCheckbox::nStyle`

Drawing style for element.

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

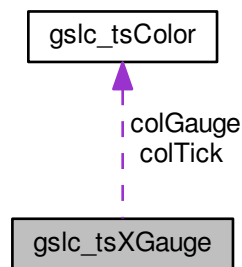
- `src/GUIslice_ex.h`

4.15 gslc_tsXGauge Struct Reference

Extended data for Gauge element.

```
#include <GUIslice_ex.h>
```

Collaboration diagram for gslc_tsXGauge:



Public Attributes

- `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.

4.15.1 Detailed Description

Extended data for Gauge element.

4.15.2 Member Data Documentation

4.15.2.1 bool `gslc_tsXGauge::bFlip`

Reverse direction of gauge.

4.15.2.2 bool `gslc_tsXGauge::bIndicFill`

Fill the indicator if true.

4.15.2.3 bool `gslc_tsXGauge::bValLastValid`

Last value valid?

4.15.2.4 bool `gslc_tsXGauge::bVert`

Vertical if true, else Horizontal.

4.15.2.5 `gslc_tsColor` `gslc_tsXGauge::colGauge`

Color of gauge fill bar.

4.15.2.6 `gslc_tsColor` `gslc_tsXGauge::colTick`

Color of gauge tick marks.

4.15.2.7 `uint16_t` `gslc_tsXGauge::nIndicLen`

Indicator length.

4.15.2.8 `uint16_t` `gslc_tsXGauge::nIndicTip`

Size of tip at end of indicator.

4.15.2.9 `int16_t` `gslc_tsXGauge::nMax`

Maximum control value.

4.15.2.10 `int16_t` `gslc_tsXGauge::nMin`

Minimum control value.

4.15.2.11 `gslc_tsXGaugeStyle` `gslc_tsXGauge::nStyle`

Gauge sub-type.

4.15.2.12 `uint16_t` `gslc_tsXGauge::nTickCnt`

Number of gauge tick marks.

4.15.2.13 `uint16_t` `gslc_tsXGauge::nTickLen`

Length of gauge tick marks.

4.15.2.14 `int16_t` `gslc_tsXGauge::nVal`

Current control value.

4.15.2.15 `int16_t` `gslc_tsXGauge::nValLast`

Last value.

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

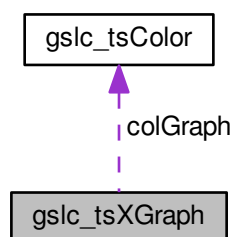
- [src/GUISlice_ex.h](#)

4.16 `gslc_tsXGraph` Struct Reference

Extended data for Graph element.

```
#include <GUISlice_ex.h>
```

Collaboration diagram for `gslc_tsXGraph`:



Public Attributes

- `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.

4.16.1 Detailed Description

Extended data for Graph element.

4.16.2 Member Data Documentation

4.16.2.1 `bool gslc_tsXGraph::bScrollEn`

Enable for scrollbar.

4.16.2.2 `gslc_tsColor gslc_tsXGraph::colGraph`

Color of the graph.

4.16.2.3 `gslc_teXGraphStyle gslc_tsXGraph::eStyle`

Style of the graph.

4.16.2.4 `uint16_t gslc_tsXGraph::nBufCnt`

Number of points in buffer.

4.16.2.5 `uint16_t gslc_tsXGraph::nBufMax`

Maximum number of points in buffer.

4.16.2.6 uint8_t gslc_tsXGraph::nMargin

Margin for graph area within element rect.

4.16.2.7 uint16_t gslc_tsXGraph::nPlotIndMax

Number of data points to show in window.

4.16.2.8 uint16_t gslc_tsXGraph::nPlotIndStart

First row of current window.

4.16.2.9 int16_t gslc_tsXGraph::nPlotValMax

Visible window maximum value.

4.16.2.10 int16_t gslc_tsXGraph::nPlotValMin

Visible window minimum value.

4.16.2.11 uint16_t gslc_tsXGraph::nScrollPos

Current scrollbar position.

4.16.2.12 uint16_t gslc_tsXGraph::nWndHeight

Visible window height.

4.16.2.13 uint16_t gslc_tsXGraph::nWndWidth

Visible window width.

4.16.2.14 int16_t* gslc_tsXGraph::pBuf

Ptr to the data buffer (circular buffer)

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

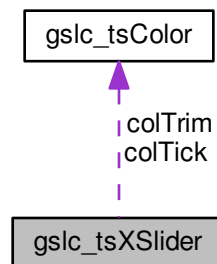
- [src/GUISlice_ex.h](#)

4.17 gslc_tsXSlider Struct Reference

Extended data for Slider element.

```
#include <GUISlice_ex.h>
```

Collaboration diagram for `gslc_tsXSlider`:



Public Attributes

- 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.

4.17.1 Detailed Description

Extended data for Slider element.

4.17.2 Member Data Documentation

4.17.2.1 bool `gslc_tsXSlider::bTrim`

Style: show a trim color.

4.17.2.2 bool gslc_tsXSlider::bVert

Orientation: true if vertical, else horizontal.

4.17.2.3 gslc_tsColor gslc_tsXSlider::colTick

Style: color of ticks.

4.17.2.4 gslc_tsColor gslc_tsXSlider::colTrim

Style: color of trim.

4.17.2.5 int16_t gslc_tsXSlider::nPos

Current position value of the slider.

4.17.2.6 int16_t gslc_tsXSlider::nPosMax

Maximum position value of the slider.

4.17.2.7 int16_t gslc_tsXSlider::nPosMin

Minimum position value of the slider.

4.17.2.8 int16_t gslc_tsXSlider::nThumbSz

Size of the thumb control.

4.17.2.9 uint16_t gslc_tsXSlider::nTickDiv

Style: number of tickmark divisions (0 for none)

4.17.2.10 int16_t gslc_tsXSlider::nTickLen

Style: length of tickmarks.

4.17.2.11 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/GUISlice_ex.h](#)

4.18 gslc_tsXTextbox Struct Reference

Extended data for Textbox element.

```
#include <GUISlice_ex.h>
```

Public Attributes

- char * [pBuf](#)
Ptr to the text buffer (circular buffer)
- uint8_t [nMargin](#)
Margin for text area within element rect.
- 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.

4.18.1 Detailed Description

Extended data for Textbox element.

4.18.2 Member Data Documentation

4.18.2.1 bool gslc_tsXTextbox::bScrollEn

Enable for scrollbar.

4.18.2.2 bool gslc_tsXTextbox::bWrapEn

Enable for line wrapping.

4.18.2.3 uint16_t gslc_tsXTextbox::nBufCols

Number of columns in buffer.

4.18.2.4 uint8_t gslc_tsXTextbox::nBufPosX

Buffer X position.

4.18.2.5 uint8_t gslc_tsXTextbox::nBufPosY

Buffer Y position.

4.18.2.6 uint16_t gslc_tsXTextbox::nBufRows

Number of rows in buffer.

4.18.2.7 uint8_t gslc_tsXTextbox::nChSizeX

Width of characters (pixels)

4.18.2.8 uint8_t gslc_tsXTextbox::nChSizeY

Height of characters (pixels)

4.18.2.9 uint8_t gslc_tsXTextbox::nCurPosX

Cursor X position.

4.18.2.10 uint8_t gslc_tsXTextbox::nCurPosY

Cursor Y position.

4.18.2.11 uint8_t gslc_tsXTextbox::nMargin

Margin for text area within element rect.

4.18.2.12 uint16_t gslc_tsXTextbox::nScrollPos

Current scrollbar position.

4.18.2.13 uint8_t gslc_tsXTextbox::nWndCols

Window X size.

4.18.2.14 uint8_t gslc_tsXTextbox::nWndRows

Window Y size.

4.18.2.15 `uint8_t gslc_tsXTextbox::nWndRowStart`

First row of current window.

4.18.2.16 `char* gslc_tsXTextbox::pBuf`

Ptr to the text buffer (circular buffer)

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

- [src/GUIslice_ex.h](#)

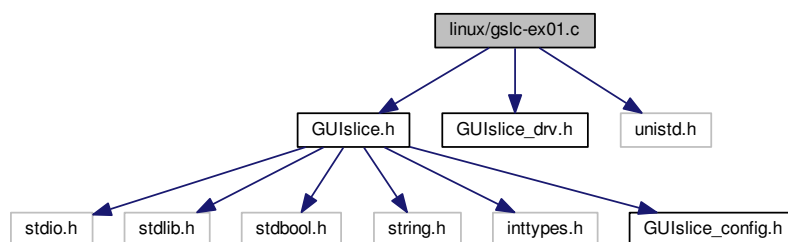
Chapter 5

File Documentation

5.1 linux/gslc-ex01.c File Reference

```
#include "GUIslice.h"  
#include "GUIslice_drv.h"  
#include <unistd.h>
```

Include dependency graph for gslc-ex01.c:



Macros

- `#define MAX_PAGE 1`
- `#define MAX_ELEM_PG_MAIN 5`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_ELEM_BOX }`

Functions

- `void UserInitEnv ()`
- `static int16_t DebugOut (char ch)`
- `int main (int argc, char *args[])`

Variables

- [gslc_tsGui m_gui](#)
- [gslc_tsDriver m_drv](#)
- [gslc_tsPage m_asPage \[MAX_PAGE\]](#)
- [gslc_tsElem m_asPageElem \[MAX_ELEM_PG_MAIN\]](#)
- [gslc_tsElemRef m_asPageElemRef \[MAX_ELEM_PG_MAIN\]](#)

5.1.1 Macro Definition Documentation

5.1.1.1 `#define MAX_ELEM_PG_MAIN 5`

5.1.1.2 `#define MAX_PAGE 1`

5.1.2 Enumeration Type Documentation

5.1.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.1.2.2 anonymous enum

Enumerator

E_ELEM_BOX

5.1.3 Function Documentation

5.1.3.1 `static int16_t DebugOut (char ch) [static]`

5.1.3.2 `int main (int argc, char * args[])`

5.1.3.3 `void UserInitEnv ()`

5.1.4 Variable Documentation

5.1.4.1 [gslc_tsPage m_asPage\[MAX_PAGE\]](#)

5.1.4.2 [gslc_tsElem m_asPageElem\[MAX_ELEM_PG_MAIN\]](#)

5.1.4.3 [gslc_tsElemRef m_asPageElemRef\[MAX_ELEM_PG_MAIN\]](#)

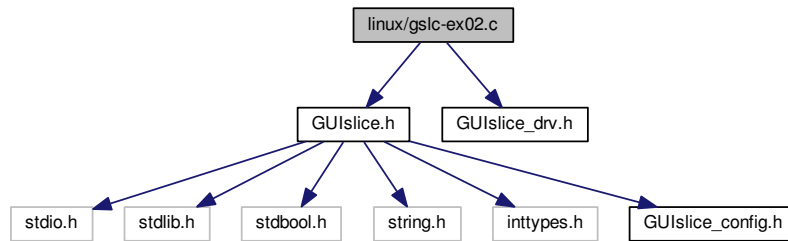
5.1.4.4 [gslc_tsDriver m_drv](#)

5.1.4.5 [gslc_tsGui m_gui](#)

5.2 linux/gslc-ex02.c File Reference

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


Include dependency graph for gslc-ex02.c:



Macros

- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`
- `#define MAX_PAGE 1`
- `#define MAX_FONT 1`
- `#define MAX_ELEM_PG_MAIN 2`

Enumerations

- enum { `E_PG_MAIN` }
- enum { `E_ELEM_BOX`, `E_ELEM_BTN_QUIT` }
- enum { `E_FONT_BTN` }

Functions

- void `UserInitEnv` ()
- static int16_t `DebugOut` (char ch)
- bool `CbBtnQuit` (void *pvGui, void *pvElemRef, `gslc_teTouch` eTouch, int16_t nX, int16_t nY)
- int `main` (int argc, char *args[])

Variables

- bool `m_bQuit` = false
- `gslc_tsGui` `m_gui`
- `gslc_tsDriver` `m_drv`
- `gslc_tsFont` `m_asFont` [`MAX_FONT`]
- `gslc_tsPage` `m_asPage` [`MAX_PAGE`]
- `gslc_tsElem` `m_asPageElem` [`MAX_ELEM_PG_MAIN`]
- `gslc_tsElemRef` `m_asPageElemRef` [`MAX_ELEM_PG_MAIN`]

5.2.1 Macro Definition Documentation

5.2.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.2.1.2 `#define MAX_ELEM_PG_MAIN 2`

5.2.1.3 `#define MAX_FONT 1`

5.2.1.4 `#define MAX_PAGE 1`

5.2.2 Enumeration Type Documentation

5.2.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.2.2.2 anonymous enum

Enumerator

E_ELEM_BOX

E_ELEM_BTN_QUIT

5.2.2.3 anonymous enum

Enumerator

E_FONT_BTN

5.2.3 Function Documentation

5.2.3.1 `bool CbBtnQuit (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

5.2.3.2 `static int16_t DebugOut (char ch)` `[static]`

5.2.3.3 `int main (int argc, char * args[])`

5.2.3.4 `void UserInitEnv ()`

5.2.4 Variable Documentation

5.2.4.1 `gslc_tsFont m_asFont[MAX_FONT]`

5.2.4.2 `gslc_tsPage m_asPage[MAX_PAGE]`

5.2.4.3 `gslc_tsElem m_asPageElem[MAX_ELEM_PG_MAIN]`

5.2.4.4 `gslc_tsElemRef m_asPageElemRef[MAX_ELEM_PG_MAIN]`

5.2.4.5 `bool m_bQuit = false`

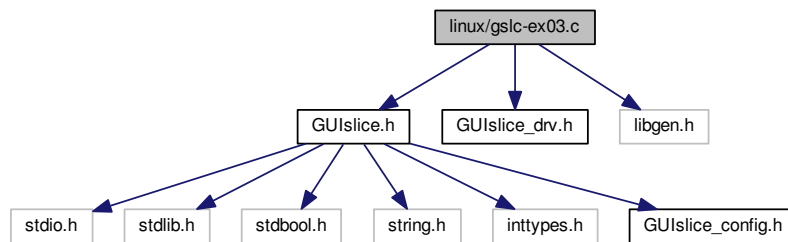
5.2.4.6 `gslc_tsDriver m_drv`

5.2.4.7 `gslc_tsGui m_gui`

5.3 linux/gslc-ex03.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_drv.h"
#include <libgen.h>
```

Include dependency graph for gslc-ex03.c:



Macros

- `#define MAX_PATH 255`
- `#define IMG_BTN_QUIT "/res/btn-exit32x32.bmp"`
- `#define IMG_BTN_QUIT_SEL "/res/btn-exit_sel32x32.bmp"`
- `#define MAX_PAGE 1`
- `#define MAX_ELEM_PG_MAIN 2`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_ELEM_BOX, E_ELEM_BTN_QUIT }`

Functions

- `void UserInitEnv ()`
- `static int16_t DebugOut (char ch)`
- `bool CbBtnQuit (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
- `bool InitOverlays (const char *strPath)`
- `int main (int argc, char *args[])`

Variables

- `char m_strImgQuit [MAX_PATH]`
- `char m_strImgQuitSel [MAX_PATH]`
- `bool m_bQuit = false`
- `gslc_tsGui m_gui`
- `gslc_tsDriver m_drv`
- `gslc_tsPage m_asPage [MAX_PAGE]`
- `gslc_tsElem m_asPageElem [MAX_ELEM_PG_MAIN]`
- `gslc_tsElemRef m_asPageElemRef [MAX_ELEM_PG_MAIN]`

5.3.1 Macro Definition Documentation

5.3.1.1 `#define IMG_BTN_QUIT "/res/btn-exit32x32.bmp"`

5.3.1.2 `#define IMG_BTN_QUIT_SEL "/res/btn-exit_sel32x32.bmp"`

5.3.1.3 `#define MAX_ELEM_PG_MAIN 2`

5.3.1.4 `#define MAX_PAGE 1`

5.3.1.5 `#define MAX_PATH 255`

5.3.2 Enumeration Type Documentation

5.3.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.3.2.2 anonymous enum

Enumerator

E_ELEM_BOX

E_ELEM_BTN_QUIT

5.3.3 Function Documentation

5.3.3.1 `bool CbBtnQuit (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

5.3.3.2 `static int16_t DebugOut (char ch) [static]`

5.3.3.3 `bool InitOverlays (const char * strPath)`

5.3.3.4 `int main (int argc, char * args[])`

5.3.3.5 `void UserInitEnv ()`

5.3.4 Variable Documentation

5.3.4.1 `gslc_tsPage m_asPage[MAX_PAGE]`

5.3.4.2 `gslc_tsElem m_asPageElem[MAX_ELEM_PG_MAIN]`

5.3.4.3 `gslc_tsElemRef m_asPageElemRef[MAX_ELEM_PG_MAIN]`

5.3.4.4 `bool m_bQuit = false`

5.3.4.5 `gslc_tsDriver m_drv`

5.3.4.6 `gslc_tsGui m_gui`

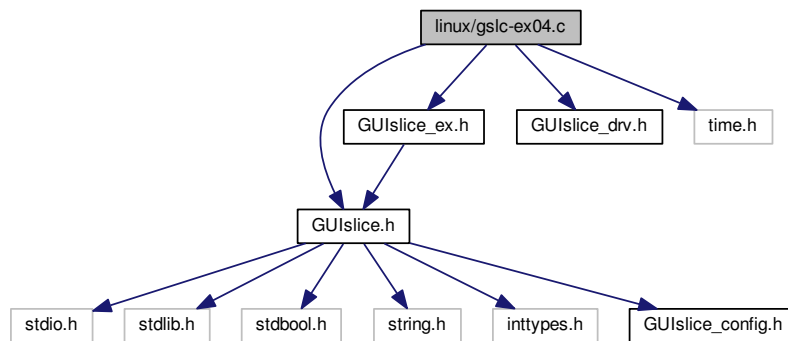
5.3.4.7 `char m_strImgQuit[MAX_PATH]`

5.3.4.8 char m_strlmgQuitSel[MAX_PATH]

5.4 linux/gslc-ex04.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
#include <time.h>
```

Include dependency graph for gslc-ex04.c:



Macros

- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`
- `#define MAX_PAGE 1`
- `#define MAX_FONT 5`
- `#define MAX_ELEM_PG_MAIN 21`
- `#define MAX_STR 100`
- `#define TEST_UPDATE_RATE 0`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_ELEM_BOX, E_ELEM_BTN_QUIT, E_ELEM_TXT_COUNT, E_ELEM_PROGRESS, E_ELEM_PROGRESS1, E_ELEM_CHECK1, E_ELEM_RADIO1, E_ELEM_RADIO2, E_ELEM_SLIDER, E_ELEM_TXT_SLIDER }`
- `enum { E_FONT_BTN, E_FONT_TXT }`
- `enum { E_GROUP1 }`

Functions

- void `UserInitEnv ()`
- static int16_t `DebugOut (char ch)`
- bool `CbBtnQuit (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
- bool `InitOverlays ()`
- int `main (int argc, char *args[])`

Variables

- bool `m_bQuit` = false
- unsigned `m_nCount` = 0
- `gslc_tsGui m_gui`
- `gslc_tsDriver m_drv`
- `gslc_tsFont m_asFont` [MAX_FONT]
- `gslc_tsPage m_asPage` [MAX_PAGE]
- `gslc_tsElem m_asPageElem` [MAX_ELEM_PG_MAIN]
- `gslc_tsElemRef m_asPageElemRef` [MAX_ELEM_PG_MAIN]
- `gslc_tsXGauge m_sXGauge`
- `gslc_tsXGauge m_sXGauge1`
- `gslc_tsXCheckbox m_asXCheck` [3]
- `gslc_tsXSlider m_sXSlider`

5.4.1 Macro Definition Documentation

5.4.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.4.1.2 `#define MAX_ELEM_PG_MAIN 21`

5.4.1.3 `#define MAX_FONT 5`

5.4.1.4 `#define MAX_PAGE 1`

5.4.1.5 `#define MAX_STR 100`

5.4.1.6 `#define TEST_UPDATE_RATE 0`

5.4.2 Enumeration Type Documentation

5.4.2.1 anonymous enum

Enumerator

E_GROUP1

5.4.2.2 anonymous enum

Enumerator

E_PG_MAIN

5.4.2.3 anonymous enum

Enumerator

E_ELEM_BOX

E_ELEM_BTN_QUIT

E_ELEM_TXT_COUNT

E_ELEM_PROGRESS

E_ELEM_PROGRESS1

E_ELEM_CHECK1

E_ELEM_RADIO1

E_ELEM_RADIO2
E_ELEM_SLIDER
E_ELEM_TXT_SLIDER

5.4.2.4 anonymous enum

Enumerator

E_FONT_BTN
E_FONT_TXT

5.4.3 Function Documentation

5.4.3.1 **bool** CbBtnQuit (**void** * *pvGui*, **void** * *pvElemRef*, **gslc_teTouch** *eTouch*, **int16_t** *nX*, **int16_t** *nY*)

5.4.3.2 **static int16_t** DebugOut (**char** *ch*) [static]

5.4.3.3 **bool** InitOverlays ()

5.4.3.4 **int** main (**int** *argc*, **char** * *args*[])

5.4.3.5 **void** UserInitEnv ()

5.4.4 Variable Documentation

5.4.4.1 **gslc_tsFont** *m_asFont*[MAX_FONT]

5.4.4.2 **gslc_tsPage** *m_asPage*[MAX_PAGE]

5.4.4.3 **gslc_tsElem** *m_asPageElem*[MAX_ELEM_PG_MAIN]

5.4.4.4 **gslc_tsElemRef** *m_asPageElemRef*[MAX_ELEM_PG_MAIN]

5.4.4.5 **gslc_tsXCheckbox** *m_asXCheck*[3]

5.4.4.6 **bool** *m_bQuit* = false

5.4.4.7 **gslc_tsDriver** *m_drv*

5.4.4.8 **gslc_tsGui** *m_gui*

5.4.4.9 **unsigned** *m_nCount* = 0

5.4.4.10 **gslc_tsXGauge** *m_sXGauge*

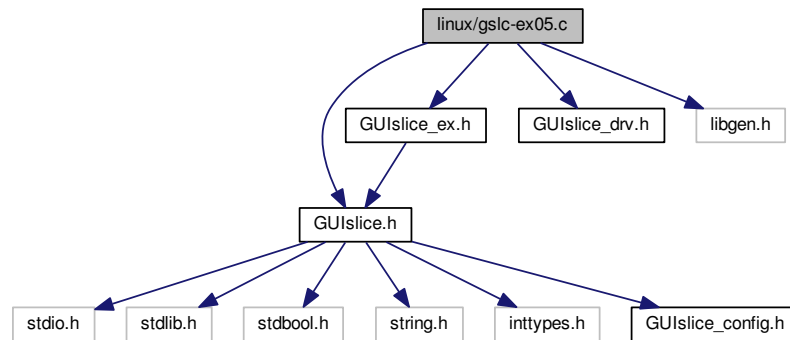
5.4.4.11 **gslc_tsXGauge** *m_sXGauge1*

5.4.4.12 **gslc_tsXSlider** *m_sXSlider*

5.5 linux/gslc-ex05.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
#include <libgen.h>
```

Include dependency graph for `gslc-ex05.c`:



Macros

- `#define MAX_PATH 255`
- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`
- `#define IMG_BKGND "/res/bkgnd1_320x240.bmp"`
- `#define MAX_PAGE 2`
- `#define MAX_FONT 5`
- `#define MAX_ELEM_PG_MAIN 10`
- `#define MAX_ELEM_PG_EXTRA 10`
- `#define MAX_STR 100`

Enumerations

- `enum { E_PG_MAIN, E_PG_EXTRA }`
- `enum { E_ELEM_BTN_QUIT, E_ELEM_BTN_EXTRA, E_ELEM_BTN_BACK, E_ELEM_TXT_COUNT, E_ELEM_PROGRESS, E_ELEM_COMP1, E_ELEM_COMP2, E_ELEM_COMP3 }`
- `enum { E_FONT_BTN, E_FONT_TXT, E_FONT_TITLE }`

Functions

- `void UserInitEnv ()`
- `static int16_t DebugOut (char ch)`
- `bool CbBtnCommon (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
- `bool InitOverlays (char *strPath)`
- `int main (int argc, char *args[])`

Variables

- `char m_strImgBkgnd [MAX_PATH]`
- `bool m_bQuit = false`
- `unsigned m_nCount = 0`
- `gslc_tsGui m_gui`
- `gslc_tsDriver m_drv`
- `gslc_tsFont m_asFont [MAX_FONT]`

- [gslc_tsPage m_asPage](#) [MAX_PAGE]
- [gslc_tsElem m_asMainElem](#) [MAX_ELEM_PG_MAIN]
- [gslc_tsElemRef m_asMainElemRef](#) [MAX_ELEM_PG_MAIN]
- [gslc_tsElem m_asExtraElem](#) [MAX_ELEM_PG_EXTRA]
- [gslc_tsElemRef m_asExtraElemRef](#) [MAX_ELEM_PG_EXTRA]
- [gslc_tsXGauge m_sXGauge](#)
- [gslc_tsXSelNum m_sXSelNum](#) [3]

5.5.1 Macro Definition Documentation

5.5.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.5.1.2 `#define IMG_BKGND "/res/bkgnd1_320x240.bmp"`

5.5.1.3 `#define MAX_ELEM_PG_EXTRA 10`

5.5.1.4 `#define MAX_ELEM_PG_MAIN 10`

5.5.1.5 `#define MAX_FONT 5`

5.5.1.6 `#define MAX_PAGE 2`

5.5.1.7 `#define MAX_PATH 255`

5.5.1.8 `#define MAX_STR 100`

5.5.2 Enumeration Type Documentation

5.5.2.1 anonymous enum

Enumerator

E_PG_MAIN

E_PG_EXTRA

5.5.2.2 anonymous enum

Enumerator

E_ELEM_BTN_QUIT

E_ELEM_BTN_EXTRA

E_ELEM_BTN_BACK

E_ELEM_TXT_COUNT

E_ELEM_PROGRESS

E_ELEM_COMP1

E_ELEM_COMP2

E_ELEM_COMP3

5.5.2.3 anonymous enum

Enumerator

E_FONT_BTN

E_FONT_TXT

E_FONT_TITLE

5.5.3 Function Documentation

5.5.3.1 `bool CbBtnCommon (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

5.5.3.2 `static int16_t DebugOut (char ch) [static]`

5.5.3.3 `bool InitOverlays (char * strPath)`

5.5.3.4 `int main (int argc, char * args[])`

5.5.3.5 `void UserInitEnv ()`

5.5.4 Variable Documentation

5.5.4.1 `gslc_tsElem m_asExtraElem[MAX_ELEM_PG_EXTRA]`

5.5.4.2 `gslc_tsElemRef m_asExtraElemRef[MAX_ELEM_PG_EXTRA]`

5.5.4.3 `gslc_tsFont m_asFont[MAX_FONT]`

5.5.4.4 `gslc_tsElem m_asMainElem[MAX_ELEM_PG_MAIN]`

5.5.4.5 `gslc_tsElemRef m_asMainElemRef[MAX_ELEM_PG_MAIN]`

5.5.4.6 `gslc_tsPage m_asPage[MAX_PAGE]`

5.5.4.7 `bool m_bQuit = false`

5.5.4.8 `gslc_tsDriver m_drv`

5.5.4.9 `gslc_tsGui m_gui`

5.5.4.10 `unsigned m_nCount = 0`

5.5.4.11 `char m_strImgBkgnd[MAX_PATH]`

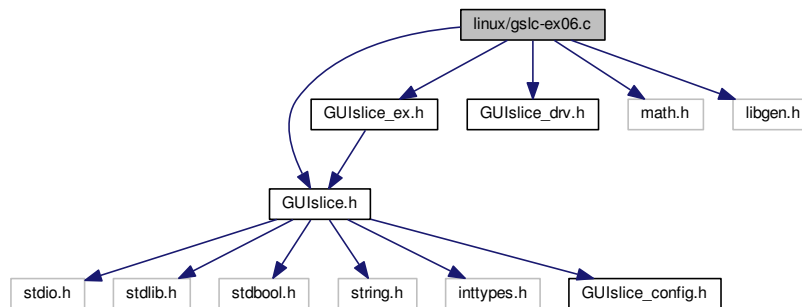
5.5.4.12 `gslc_tsXGauge m_sXGauge`

5.5.4.13 `gslc_tsXSelNum m_sXSelNum[3]`

5.6 linux/gslc-ex06.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
#include <math.h>
#include <libgen.h>
```

Include dependency graph for gslc-ex06.c:



Macros

- `#define MAX_PATH 255`
- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`
- `#define IMG_LOGO "/res/logo1-200x40.bmp"`
- `#define MAX_PAGE 1`
- `#define MAX_FONT 5`
- `#define MAX_ELEM_PG_MAIN 20`
- `#define MAX_STR 100`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_ELEM_BTN_QUIT, E_ELEM_TXT_COUNT, E_ELEM_PROGRESS, E_ELEM_DATAX, E_ELEM_DATAY, E_ELEM_DATAZ, E_ELEM_SCAN, E_ELEM_CHECK1 }`
- `enum { E_FONT_BTN, E_FONT_TXT }`

Functions

- `void UserInitEnv ()`
- `static int16_t DebugOut (char ch)`
- `bool CbDrawScanner (void *pvGui, void *pvElemRef, gslc_teRedrawType eRedraw)`
- `bool CbTickScanner (void *pvGui, void *pvScope)`
- `bool CbBtnQuit (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
- `bool InitOverlays (char *strPath)`
- `int main (int argc, char *args[])`

Variables

- `char m_strImgLogo [MAX_PATH]`
- `bool m_bQuit = false`
- `unsigned m_nCount = 0`
- `float m_fCoordX = 0`
- `float m_fCoordY = 0`
- `float m_fCoordZ = 0`

- [gslc_tsGui m_gui](#)
- [gslc_tsDriver m_drv](#)
- [gslc_tsFont m_asFont \[MAX_FONT\]](#)
- [gslc_tsPage m_asPage \[MAX_PAGE\]](#)
- [gslc_tsElem m_asPageElem \[MAX_ELEM_PG_MAIN\]](#)
- [gslc_tsElemRef m_asPageElemRef \[MAX_ELEM_PG_MAIN\]](#)
- [gslc_tsXGauge m_sXGauge](#)
- [gslc_tsXCheckbox m_asXCheck \[2\]](#)
- `int16_t m_nOriginX = 0`
- `int16_t m_nOriginY = 0`

5.6.1 Macro Definition Documentation

5.6.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.6.1.2 `#define IMG_LOGO "/res/logo1-200x40.bmp"`

5.6.1.3 `#define MAX_ELEM_PG_MAIN 20`

5.6.1.4 `#define MAX_FONT 5`

5.6.1.5 `#define MAX_PAGE 1`

5.6.1.6 `#define MAX_PATH 255`

5.6.1.7 `#define MAX_STR 100`

5.6.2 Enumeration Type Documentation

5.6.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.6.2.2 anonymous enum

Enumerator

E_ELEM_BTN_QUIT

E_ELEM_TXT_COUNT

E_ELEM_PROGRESS

E_ELEM_DATAX

E_ELEM_DATAY

E_ELEM_DATAZ

E_ELEM_SCAN

E_ELEM_CHECK1

5.6.2.3 anonymous enum

Enumerator

E_FONT_BTN

E_FONT_TXT

5.6.3 Function Documentation

5.6.3.1 `bool CbBtnQuit (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

5.6.3.2 `bool CbDrawScanner (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

5.6.3.3 `bool CbTickScanner (void * pvGui, void * pvScope)`

5.6.3.4 `static int16_t DebugOut (char ch) [static]`

5.6.3.5 `bool InitOverlays (char * strPath)`

5.6.3.6 `int main (int argc, char * args[])`

5.6.3.7 `void UserInitEnv ()`

5.6.4 Variable Documentation

5.6.4.1 `gslc_tsFont m_asFont[MAX_FONT]`

5.6.4.2 `gslc_tsPage m_asPage[MAX_PAGE]`

5.6.4.3 `gslc_tsElem m_asPageElem[MAX_ELEM_PG_MAIN]`

5.6.4.4 `gslc_tsElemRef m_asPageElemRef[MAX_ELEM_PG_MAIN]`

5.6.4.5 `gslc_tsXCheckbox m_asXCheck[2]`

5.6.4.6 `bool m_bQuit = false`

5.6.4.7 `gslc_tsDriver m_drv`

5.6.4.8 `float m_fCoordX = 0`

5.6.4.9 `float m_fCoordY = 0`

5.6.4.10 `float m_fCoordZ = 0`

5.6.4.11 `gslc_tsGui m_gui`

5.6.4.12 `unsigned m_nCount = 0`

5.6.4.13 `int16_t m_nOriginX = 0`

5.6.4.14 `int16_t m_nOriginY = 0`

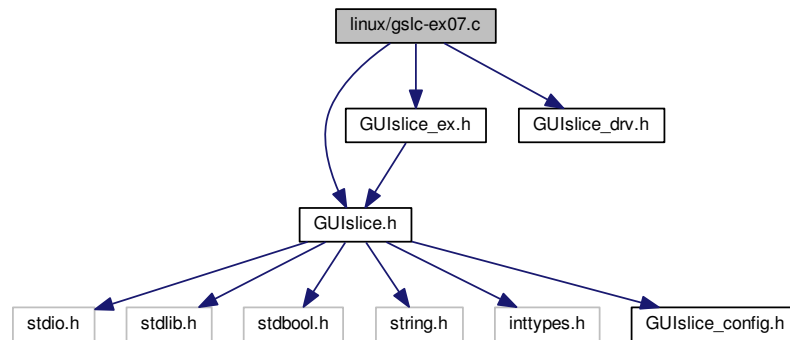
5.6.4.15 `char m_strImgLogo[MAX_PATH]`

5.6.4.16 `gslc_tsXGauge m_sXGauge`

5.7 linux/gslc-ex07.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
```

Include dependency graph for gslc-ex07.c:



Macros

- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`
- `#define MAX_PAGE 1`
- `#define MAX_FONT 5`
- `#define MAX_ELEM_PG_MAIN 17`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_ELEM_BOX, E_ELEM_BTN_QUIT, E_ELEM_COLOR, E_SLIDER_R, E_SLIDER_G, E_SLIDER_B }`
- `enum { E_FONT_BTN, E_FONT_TXT, E_FONT_HEAD, E_FONT_TITLE }`

Functions

- `void UserInitEnv ()`
- `static int16_t DebugOut (char ch)`
- `bool CbBtnQuit (void *pvGui, void *pvElem, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
- `bool CbSlidePos (void *pvGui, void *pvElemRef, int16_t nPos)`
- `bool InitOverlays ()`
- `int main (int argc, char *args[])`

Variables

- `bool m_bQuit = false`
- `unsigned m_nCount = 0`
- `gslc_tsGui m_gui`
- `gslc_tsDriver m_drv`
- `gslc_tsFont m_asFont [MAX_FONT]`
- `gslc_tsPage m_asPage [MAX_PAGE]`
- `gslc_tsElem m_asPageElem [MAX_ELEM_PG_MAIN]`
- `gslc_tsElemRef m_asPageElemRef [MAX_ELEM_PG_MAIN]`
- `gslc_tsXSlider m_sXSlider_R`

- [gslc_tsXSlider m_sXSlider_G](#)
- [gslc_tsXSlider m_sXSlider_B](#)
- uint16_t [m_nPosR](#) = 255
- uint16_t [m_nPosG](#) = 128
- uint16_t [m_nPosB](#) = 0

5.7.1 Macro Definition Documentation

5.7.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.7.1.2 `#define MAX_ELEM_PG_MAIN 17`

5.7.1.3 `#define MAX_FONT 5`

5.7.1.4 `#define MAX_PAGE 1`

5.7.2 Enumeration Type Documentation

5.7.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.7.2.2 anonymous enum

Enumerator

E_ELEM_BOX

E_ELEM_BTN_QUIT

E_ELEM_COLOR

E_SLIDER_R

E_SLIDER_G

E_SLIDER_B

5.7.2.3 anonymous enum

Enumerator

E_FONT_BTN

E_FONT_TXT

E_FONT_HEAD

E_FONT_TITLE

5.7.3 Function Documentation

5.7.3.1 `bool CbBtnQuit (void * pvGui, void * pvElem, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

5.7.3.2 `bool CbSlidePos (void * pvGui, void * pvElemRef, int16_t nPos)`

5.7.3.3 `static int16_t DebugOut (char ch) [static]`

5.7.3.4 `bool InitOverlays ()`

5.7.3.5 `int main (int argc, char * args[])`

5.7.3.6 `void UserInitEnv ()`

5.7.4 Variable Documentation

5.7.4.1 `gslc_tsFont m_asFont[MAX_FONT]`

5.7.4.2 `gslc_tsPage m_asPage[MAX_PAGE]`

5.7.4.3 `gslc_tsElem m_asPageElem[MAX_ELEM_PG_MAIN]`

5.7.4.4 `gslc_tsElemRef m_asPageElemRef[MAX_ELEM_PG_MAIN]`

5.7.4.5 `bool m_bQuit = false`

5.7.4.6 `gslc_tsDriver m_drv`

5.7.4.7 `gslc_tsGui m_gui`

5.7.4.8 `unsigned m_nCount = 0`

5.7.4.9 `uint16_t m_nPosB = 0`

5.7.4.10 `uint16_t m_nPosG = 128`

5.7.4.11 `uint16_t m_nPosR = 255`

5.7.4.12 `gslc_tsXSlider m_sXSlider_B`

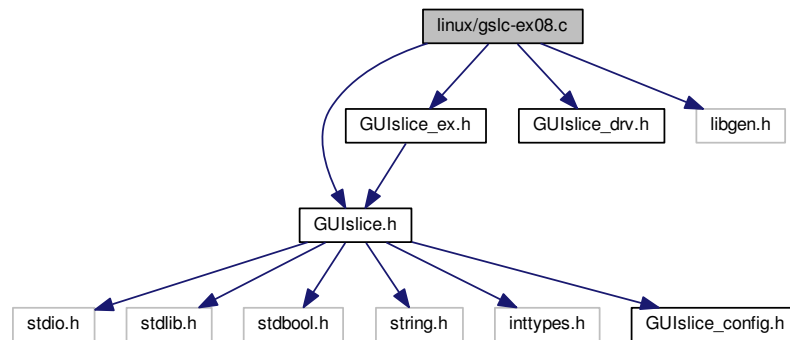
5.7.4.13 `gslc_tsXSlider m_sXSlider_G`

5.7.4.14 `gslc_tsXSlider m_sXSlider_R`

5.8 linux/gslc-ex08.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
#include <libgen.h>
```


Include dependency graph for gslc-ex08.c:



Macros

- `#define MAX_PATH 255`
- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`
- `#define IMG_GRAD_BACK "/res/grad-blue1.bmp"`
- `#define IMG_GRADBAR_TOP "/res/gradbar-purple-top.bmp"`
- `#define IMG_GRADBAR_BOT "/res/gradbar-purple-bot.bmp"`
- `#define MAX_PAGE 1`
- `#define MAX_FONT 10`
- `#define MAX_ELEM_PG_MAIN 30`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_IMG_BACK, E_IMG_BAR_TOP, E_IMG_BAR_BOT, E_BOX_CURRENT, E_BOX_PRESETS, E_BTN1, E_BTN2, E_BTN3, E_BTN4, E_BTN5, E_BTN6, E_BTN7, E_BTN8, E_BTN_P1, E_BTN_P2, E_BTN_P3, E_BTN_P4, E_BTN_P5, E_BTN_QUIT }`
- `enum { E_FONT_BTN, E_FONT_TXT, E_FONT_HEAD, E_FONT_TITLE }`

Functions

- `void UserInitEnv ()`
- `bool CbBtnQuit (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
- `bool InitOverlays (char *strBasePath)`
- `int main (int argc, char *args[])`

Variables

- `char m_strImgGradBack [MAX_PATH]`
- `char m_strImgGradBarTop [MAX_PATH]`
- `char m_strImgGradBarBot [MAX_PATH]`
- `bool m_bQuit = false`

- unsigned `m_nCount` = 0
- `gslc_tsGui m_gui`
- `gslc_tsDriver m_drv`
- `gslc_tsFont m_asFont` [`MAX_FONT`]
- `gslc_tsPage m_asPage` [`MAX_PAGE`]
- `gslc_tsElem m_asPageElem` [`MAX_ELEM_PG_MAIN`]
- `gslc_tsElemRef m_asPageElemRef` [`MAX_ELEM_PG_MAIN`]

5.8.1 Macro Definition Documentation

5.8.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.8.1.2 `#define IMG_GRAD_BACK "/res/grad-blue1.bmp"`

5.8.1.3 `#define IMG_GRADBAR_BOT "/res/gradbar-purple-bot.bmp"`

5.8.1.4 `#define IMG_GRADBAR_TOP "/res/gradbar-purple-top.bmp"`

5.8.1.5 `#define MAX_ELEM_PG_MAIN 30`

5.8.1.6 `#define MAX_FONT 10`

5.8.1.7 `#define MAX_PAGE 1`

5.8.1.8 `#define MAX_PATH 255`

5.8.2 Enumeration Type Documentation

5.8.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.8.2.2 anonymous enum

Enumerator

E_IMG_BACK

E_IMG_BAR_TOP

E_IMG_BAR_BOT

E_BOX_CURRENT

E_BOX_PRESETS

E_BTN1

E_BTN2

E_BTN3

E_BTN4

E_BTN5

E_BTN6

E_BTN7

E_BTN8

E_BTN_P1

E_BTN_P2
E_BTN_P3
E_BTN_P4
E_BTN_P5
E_BTN_QUIT

5.8.2.3 anonymous enum

Enumerator

E_FONT_BTN
E_FONT_TXT
E_FONT_HEAD
E_FONT_TITLE

5.8.3 Function Documentation

- 5.8.3.1 **bool** CbBtnQuit (void * *pvGui*, void * *pvElemRef*, **gslc_teTouch** *eTouch*, **int16_t** *nX*, **int16_t** *nY*)
- 5.8.3.2 **bool** InitOverlays (char * *strBasePath*)
- 5.8.3.3 **int** main (**int** *argc*, char * *args[]*)
- 5.8.3.4 **void** UserInitEnv ()

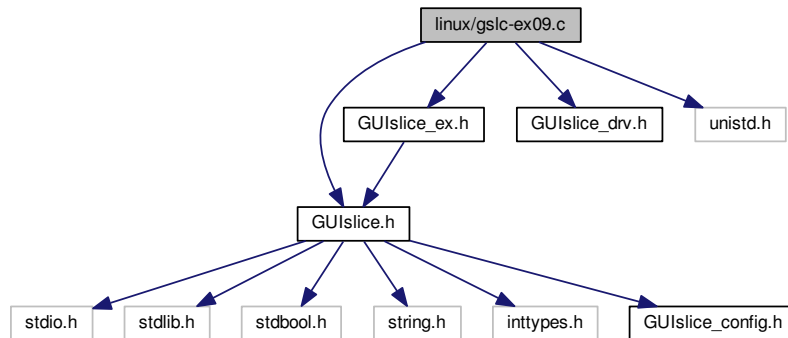
5.8.4 Variable Documentation

- 5.8.4.1 **gslc_tsFont** *m_asFont*[**MAX_FONT**]
- 5.8.4.2 **gslc_tsPage** *m_asPage*[**MAX_PAGE**]
- 5.8.4.3 **gslc_tsElem** *m_asPageElem*[**MAX_ELEM_PG_MAIN**]
- 5.8.4.4 **gslc_tsElemRef** *m_asPageElemRef*[**MAX_ELEM_PG_MAIN**]
- 5.8.4.5 **bool** *m_bQuit* = false
- 5.8.4.6 **gslc_tsDriver** *m_drv*
- 5.8.4.7 **gslc_tsGui** *m_gui*
- 5.8.4.8 **unsigned** *m_nCount* = 0
- 5.8.4.9 **char** *m_strImgGradBack*[**MAX_PATH**]
- 5.8.4.10 **char** *m_strImgGradBarBot*[**MAX_PATH**]
- 5.8.4.11 **char** *m_strImgGradBarTop*[**MAX_PATH**]

5.9 linux/gslc-ex09.c File Reference

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

```
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
#include "unistd.h"
Include dependency graph for gslc-ex09.c:
```



Macros

- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`
- `#define MAX_PAGE 1`
- `#define MAX_FONT 3`
- `#define MAX_ELEM_PG_MAIN 8`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_ELEM_BOX, E_ELEM_BTN_QUIT, E_ELEM_COLOR, E_RADIAL, E_RAMP, E_SLIDER, E_ELEM_TXT_COUNT }`
- `enum { E_FONT_BTN, E_FONT_TXT, E_FONT_TITLE }`

Functions

- `void UserInitEnv ()`
- `static int16_t DebugOut (char ch)`
- `bool CbBtnQuit (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
- `bool CbSlideRadial (void *pvGui, void *pvElemRef, int16_t nPos)`
- `bool InitOverlays ()`
- `int main (int argc, char *args[])`

Variables

- `bool m_bQuit = false`
- `unsigned m_nCount = 0`
- `gslc_tsGui m_gui`
- `gslc_tsDriver m_drv`
- `gslc_tsFont m_asFont [MAX_FONT]`
- `gslc_tsPage m_asPage [MAX_PAGE]`

- [gslc_tsElem m_asPageElem](#) [[MAX_ELEM_PG_MAIN](#)]
- [gslc_tsElemRef m_asPageElemRef](#) [[MAX_ELEM_PG_MAIN](#)]
- [gslc_tsXGauge m_sXRadial](#)
- [gslc_tsXGauge m_sXRamp](#)
- [gslc_tsXSlider m_sXSlider](#)

5.9.1 Macro Definition Documentation

5.9.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.9.1.2 `#define MAX_ELEM_PG_MAIN 8`

5.9.1.3 `#define MAX_FONT 3`

5.9.1.4 `#define MAX_PAGE 1`

5.9.2 Enumeration Type Documentation

5.9.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.9.2.2 anonymous enum

Enumerator

E_ELEM_BOX

E_ELEM_BTN_QUIT

E_ELEM_COLOR

E_RADIAL

E_RAMP

E_SLIDER

E_ELEM_TXT_COUNT

5.9.2.3 anonymous enum

Enumerator

E_FONT_BTN

E_FONT_TXT

E_FONT_TITLE

5.9.3 Function Documentation

5.9.3.1 `bool CbBtnQuit (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

5.9.3.2 `bool CbSlideRadial (void * pvGui, void * pvElemRef, int16_t nPos)`

5.9.3.3 `static int16_t DebugOut (char ch)` [`static`]

5.9.3.4 `bool InitOverlays ()`

5.9.3.5 `int main (int argc, char * args[])`

5.9.3.6 `void UserInitEnv ()`

5.9.4 Variable Documentation

5.9.4.1 `gslc_tsFont m_asFont[MAX_FONT]`

5.9.4.2 `gslc_tsPage m_asPage[MAX_PAGE]`

5.9.4.3 `gslc_tsElem m_asPageElem[MAX_ELEM_PG_MAIN]`

5.9.4.4 `gslc_tsElemRef m_asPageElemRef[MAX_ELEM_PG_MAIN]`

5.9.4.5 `bool m_bQuit = false`

5.9.4.6 `gslc_tsDriver m_drv`

5.9.4.7 `gslc_tsGui m_gui`

5.9.4.8 `unsigned m_nCount = 0`

5.9.4.9 `gslc_tsXGauge m_sXRadial`

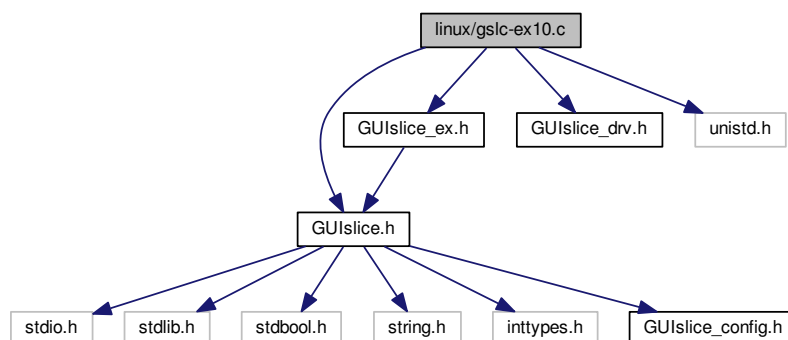
5.9.4.10 `gslc_tsXGauge m_sXRamp`

5.9.4.11 `gslc_tsXSlider m_sXSlider`

5.10 linux/gslc-ex10.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
#include "unistd.h"
```

Include dependency graph for `gslc-ex10.c`:



Macros

- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

- `#define MAX_PAGE 1`
- `#define MAX_FONT 3`
- `#define MAX_ELEM_PG_MAIN 9`
- `#define TBOX_ROWS 20`
- `#define TBOX_COLS 20`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_ELEM_BOX, E_ELEM_BTN_QUIT, E_ELEM_COLOR, E_SLIDER, E_ELEM_TXT_COUNT, E_ELEM_TEXTBOX, E_SCROLLBAR }`
- `enum { E_FONT_BTN, E_FONT_TXT, E_FONT_TITLE }`

Functions

- `void UserInitEnv ()`
- `static int16_t DebugOut (char ch)`
- `bool CbBtnQuit (void *pvGui, void *pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`
- `bool CbControls (void *pvGui, void *pvElemRef, int16_t nPos)`
- `bool InitOverlays ()`
- `int main (int argc, char *args[])`

Variables

- `bool m_bQuit = false`
- `unsigned m_nCount = 0`
- `gslc_tsGui m_gui`
- `gslc_tsDriver m_drv`
- `gslc_tsFont m_asFont [MAX_FONT]`
- `gslc_tsPage m_asPage [MAX_PAGE]`
- `gslc_tsElem m_asPageElem [MAX_ELEM_PG_MAIN]`
- `gslc_tsElemRef m_asPageElemRef [MAX_ELEM_PG_MAIN]`
- `gslc_tsXSlider m_sXSlider`
- `gslc_tsXSlider m_sXSliderText`
- `gslc_tsXTextbox m_sTextbox`
- `char m_acTextboxBuf [TBOX_ROWS * TBOX_COLS]`

5.10.1 Macro Definition Documentation

5.10.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.10.1.2 `#define MAX_ELEM_PG_MAIN 9`

5.10.1.3 `#define MAX_FONT 3`

5.10.1.4 `#define MAX_PAGE 1`

5.10.1.5 `#define TBOX_COLS 20`

5.10.1.6 `#define TBOX_ROWS 20`

5.10.2 Enumeration Type Documentation

5.10.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.10.2.2 anonymous enum

Enumerator

E_ELEM_BOX

E_ELEM_BTN_QUIT

E_ELEM_COLOR

E_SLIDER

E_ELEM_TXT_COUNT

E_ELEM_TEXTBOX

E_SCROLLBAR

5.10.2.3 anonymous enum

Enumerator

E_FONT_BTN

E_FONT_TXT

E_FONT_TITLE

5.10.3 Function Documentation

5.10.3.1 `bool CbBtnQuit (void * pvGui, void * pvElemRef, gslc_teTouch eTouch, int16_t nX, int16_t nY)`

5.10.3.2 `bool CbControls (void * pvGui, void * pvElemRef, int16_t nPos)`

5.10.3.3 `static int16_t DebugOut (char ch) [static]`

5.10.3.4 `bool InitOverlays ()`

5.10.3.5 `int main (int argc, char * args[])`

5.10.3.6 `void UserInitEnv ()`

5.10.4 Variable Documentation

5.10.4.1 `char m_acTextboxBuf[TBOX_ROWS * TBOX_COLS]`

5.10.4.2 `gslc_tsFont m_asFont[MAX_FONT]`

5.10.4.3 `gslc_tsPage m_asPage[MAX_PAGE]`

5.10.4.4 `gslc_tsElem m_asPageElem[MAX_ELEM_PG_MAIN]`

5.10.4.5 `gslc_tsElemRef m_asPageElemRef[MAX_ELEM_PG_MAIN]`

5.10.4.6 `bool m_bQuit = false`

5.10.4.7 `gslc_tsDriver m_drv`

5.10.4.8 `gslc_tsGui m_gui`

5.10.4.9 `unsigned m_nCount = 0`

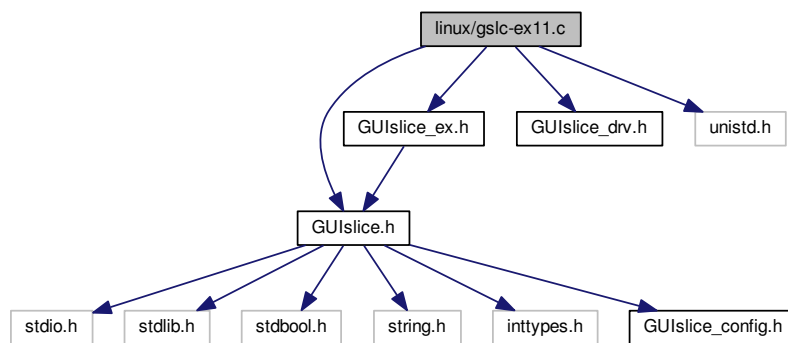
5.10.4.10 `gslc_tsXTextbox m_sTextbox`

5.10.4.11 `gslc_tsXSlider m_sSlider`

5.10.4.12 `gslc_tsXSlider m_sSliderText`

5.11 linux/gslc-ex11.c File Reference

```
#include "GUIslice.h"
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
#include "unistd.h"
Include dependency graph for gslc-ex11.c:
```



Macros

- `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`
- `#define MAX_PAGE 1`
- `#define MAX_FONT 3`
- `#define MAX_ELEM_PG_MAIN 9`
- `#define GRAPH_ROWS 200`

Enumerations

- `enum { E_PG_MAIN }`
- `enum { E_ELEM_BOX, E_ELEM_BTN_QUIT, E_ELEM_COLOR, E_SLIDER, E_ELEM_TXT_COUNT, E_ELEM_GRAPH, E_SCROLLBAR }`
- `enum { E_FONT_BTN, E_FONT_TXT, E_FONT_TITLE }`

Functions

- void [UserInitEnv](#) ()
- static int16_t [DebugOut](#) (char ch)
- bool [CbBtnQuit](#) (void *pvGui, void *pvElemRef, [gslc_teTouch](#) eTouch, int16_t nX, int16_t nY)
- bool [CbControls](#) (void *pvGui, void *pvElemRef, int16_t nPos)
- bool [InitOverlays](#) ()
- int [main](#) (int argc, char *args[])

Variables

- bool [m_bQuit](#) = false
- unsigned [m_nCount](#) = 0
- [gslc_tsGui](#) [m_gui](#)
- [gslc_tsDriver](#) [m_drv](#)
- [gslc_tsFont](#) [m_asFont](#) [MAX_FONT]
- [gslc_tsPage](#) [m_asPage](#) [MAX_PAGE]
- [gslc_tsElem](#) [m_asPageElem](#) [MAX_ELEM_PG_MAIN]
- [gslc_tsElemRef](#) [m_asPageElemRef](#) [MAX_ELEM_PG_MAIN]
- [gslc_tsXSlider](#) [m_sXSlider](#)
- [gslc_tsXSlider](#) [m_sXSliderGraph](#)
- [gslc_tsXGraph](#) [m_sGraph](#)
- int16_t [m_anGraphBuf](#) [GRAPH_ROWS]

5.11.1 Macro Definition Documentation

5.11.1.1 `#define FONT_DROID_SANS "/usr/share/fonts/truetype/droid/DroidSans.ttf"`

5.11.1.2 `#define GRAPH_ROWS 200`

5.11.1.3 `#define MAX_ELEM_PG_MAIN 9`

5.11.1.4 `#define MAX_FONT 3`

5.11.1.5 `#define MAX_PAGE 1`

5.11.2 Enumeration Type Documentation

5.11.2.1 anonymous enum

Enumerator

E_PG_MAIN

5.11.2.2 anonymous enum

Enumerator

E_ELEM_BOX

E_ELEM_BTN_QUIT

E_ELEM_COLOR

E_SLIDER

E_ELEM_TXT_COUNT

E_ELEM_GRAPH

E_SCROLLBAR

5.11.2.3 anonymous enum

Enumerator

E_FONT_BTN***E_FONT_TXT******E_FONT_TITLE***

5.11.3 Function Documentation

5.11.3.1 **bool** CbBtnQuit (void * *pvGui*, void * *pvElemRef*, gslc_teTouch *eTouch*, int16_t *nX*, int16_t *nY*)5.11.3.2 **bool** CbControls (void * *pvGui*, void * *pvElemRef*, int16_t *nPos*)5.11.3.3 **static** int16_t DebugOut (char *ch*) [static]5.11.3.4 **bool** InitOverlays ()5.11.3.5 **int** main (int *argc*, char * *args*[])5.11.3.6 **void** UserInitEnv ()

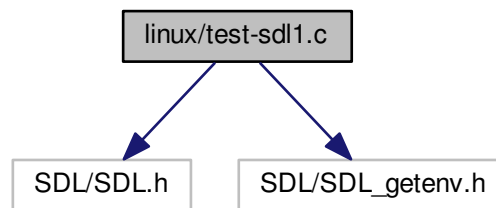
5.11.4 Variable Documentation

5.11.4.1 int16_t *m_anGraphBuf*[GRAPH_ROWS]5.11.4.2 **gslc_tsFont** *m_asFont*[MAX_FONT]5.11.4.3 **gslc_tsPage** *m_asPage*[MAX_PAGE]5.11.4.4 **gslc_tsElem** *m_asPageElem*[MAX_ELEM_PG_MAIN]5.11.4.5 **gslc_tsElemRef** *m_asPageElemRef*[MAX_ELEM_PG_MAIN]5.11.4.6 **bool** *m_bQuit* = false5.11.4.7 **gslc_tsDriver** *m_drv*5.11.4.8 **gslc_tsGui** *m_gui*5.11.4.9 **unsigned** *m_nCount* = 05.11.4.10 **gslc_tsXGraph** *m_sGraph*5.11.4.11 **gslc_tsXSlider** *m_sXSlider*5.11.4.12 **gslc_tsXSlider** *m_sXSliderGraph*

5.12 linux/test-sdl1.c File Reference

```
#include "SDL/SDL.h"
#include "SDL/SDL_getenv.h"
```

Include dependency graph for test-sdl1.c:



Functions

- int `main` (int argc, char *args[])

Variables

- SDL_Surface * `scrMain` = NULL

5.12.1 Function Documentation

5.12.1.1 int main (int argc, char * args[])

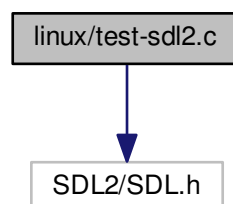
5.12.2 Variable Documentation

5.12.2.1 SDL_Surface* scrMain = NULL

5.13 linux/test-sdl2.c File Reference

```
#include <SDL2/SDL.h>
```

Include dependency graph for test-sdl2.c:



Functions

- int [main](#) (int argc, char *args[])

Variables

- SDL_Window * [pWind](#) = NULL
- SDL_Renderer * [pRender](#) = NULL

5.13.1 Function Documentation

5.13.1.1 int main (int *argc*, char * *args*[])

5.13.2 Variable Documentation

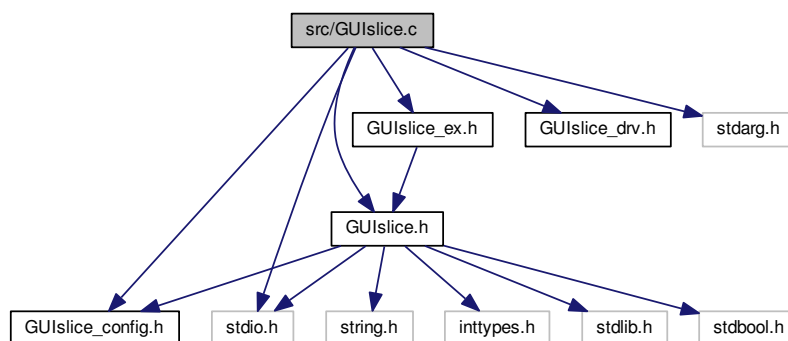
5.13.2.1 SDL_Renderer* pRender = NULL

5.13.2.2 SDL_Window* pWind = NULL

5.14 README.md File Reference

5.15 src/GUISlice.c File Reference

```
#include "GUISlice_config.h"
#include "GUISlice.h"
#include "GUISlice_ex.h"
#include "GUISlice_drv.h"
#include <stdio.h>
#include <stdarg.h>
Include dependency graph for GUISlice.c:
```



Macros

- #define [GUISLICE_VER](#) "0.10.0"

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.
- 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.
- 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_tsFontRefType 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_SetPageCur](#) ([gslc_tsGui](#) *pGui, [int16_t](#) nPageId)
Select a new page for display.
- 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.
- void [gslc_PageSetEventFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage, [GSLC_CB_EVENT](#) funcCb)
Assign the event callback function for a page.
- [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)
- void [gslc_SetElemRefFlag](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [uint8_t](#) nFlagMask, [uint8_t](#) nFlagVal)
- [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_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_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_ElemSetStyleFrom](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRefSrc, [gslc_tsElemRef](#) *pElemRefDest)
Copy style settings from one element to another.
 - void [gslc_ElemSetEventFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_EVENT](#) funcCb)
Assign the event callback function for a 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.*
- void [gslc_CollectTouch](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect, [gslc_tsEventTouch](#) *pEventTouch)
- Handle touch 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.*
- 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)
- Initialize the touchscreen device driver.*
- [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.*
- 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](#) *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_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.*
- 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_ts↵ ImgRef](#) 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_ElemSendEventTouch](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRefTracked, [gslc_teTouch](#) e↵ Touch, int16_t nX, int16_t nY)
- Trigger an element's touch event.*
- void [gslc_ResetElem](#) ([gslc_tsElem](#) *pElem)
- Initialize an Element struct.*
- void [gslc_ResetFont](#) ([gslc_tsFont](#) *pFont)
- Initialize a Font struct.*
- void [gslc_ElemDestruct](#) ([gslc_tsElem](#) *pElem)
- Free up any members associated with an element.*
- void [gslc_CollectDestruct](#) ([gslc_tsGui](#) *pGui, [gslc_tsCollect](#) *pCollect)
- Free up any members associated with an element collection.*
- void [gslc_PageDestruct](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage)
- Free up any members associated with a page.*
- void [gslc_GuiDestruct](#) ([gslc_tsGui](#) *pGui)
- Free up any surfaces associated with the GUI, pages, collections and elements.*
- void [gslc_CollectReset](#) ([gslc_tsCollect](#) *pCollect, [gslc_tsElem](#) *asElem, uint16_t nElemMax, [gslc_tsElemRef](#) *asElemRef, uint16_t nElemRefMax)

- Reset the members of an element collection.*
- `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.
- `void gslc_CollectSetEventFunc (gslc_tsGui *pGui, gslc_tsCollect *pCollect, GSLC_CB_EVENT funcCb)`
Assign the event callback function for an element collection.

Variables

- `GSLC_CB_DEBUG_OUT g_pfDebugOut = NULL`
Global debug output function.
- `uint16_t m_nLUTSinFOX16 [257]`
- `const char GSLC_PMEM_ERRSTR_NULL [] = "ERROR: %z() called with NULL ptr\n"`
- `const char GSLC_PMEM_ERRSTR_PXD_NULL [] = "ERROR: %z() pXData NULL\n"`

5.15.1 Macro Definition Documentation

- 5.15.1.1 `#define GUISLICE_VER "0.10.0"`

5.15.2 Enumeration Type Documentation

- 5.15.2.1 `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`

5.15.3 Function Documentation

- 5.15.3.1 `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

5.15.3.2 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

5.15.3.3 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

5.15.3.4 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

5.15.3.5 `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

5.15.3.6 `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

5.15.3.7 `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

5.15.3.8 `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

5.15.3.9 `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

5.15.3.10 `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

5.15.3.11 `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

5.15.3.12 `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

5.15.3.13 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

5.15.3.14 void `gslc_CollectSetEventFunc (gslc_tsGui * pGui, gslc_tsCollect * pCollect, GSLC_CB_EVENT funcCb)`

Assign the event callback function for an element collection.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to collection
in	<i>funcCb</i>	Function pointer to event routine (or NULL for default))

Returns

none

5.15.3.15 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

5.15.3.16 `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

5.15.3.17 `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

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

Check whether two colors are equal.

Parameters

in	<i>a</i>	First color
----	----------	-------------

<i>in</i>	<i>b</i>	Second color
-----------	----------	--------------

Returns

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

5.15.3.19 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

<i>in</i>	<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.

5.15.3.20 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

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

Returns

none

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

Draw a filled circle.

Parameters

<i>in</i>	<i>pGui</i>	Pointer to GUI
<i>in</i>	<i>nMidX</i>	Center X coordinate
<i>in</i>	<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

5.15.3.22 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

5.15.3.23 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

5.15.3.24 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

5.15.3.25 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

5.15.3.26 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

5.15.3.27 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

5.15.3.28 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

5.15.3.29 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

5.15.3.30 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

5.15.3.31 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

5.15.3.32 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

5.15.3.33 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

5.15.3.34 gslc_tsElemRef* gslc_ElemAdd (gslc_tsGui * *pGui*, int16_t *nPagId*, 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>nPageld</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

5.15.3.35 `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.

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>nPageld</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

5.15.3.36 `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

5.15.3.37 `gslc_tsElemRef* gslc_ElemCreateBtnImg (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, gslc_tsImgRef slmgRef, gslc_tsImgRef slmgRefSel, GSLC_CB_TOUCH cbTouch)`

Create a graphical Button Element.

- 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>slmgRef</i>	Image reference to load (unselected state)
in	<i>slmgRefSel</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

5.15.3.38 `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)
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
in	<i>cbTouch</i>	Callback for touch events

Returns

Pointer to the Element reference or NULL if failure

5.15.3.39 `gslc_tsElemRef* gslc_ElemCreateImg (gslc_tsGui * pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem, gslc_tsImgRef slmgRef)`

Create an image Element.

- Draws an image

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
in	<i>sImgRef</i>	Image reference to load

Returns

Pointer to the Element reference or NULL if failure

5.15.3.40 **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>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>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

5.15.3.41 **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

5.15.3.42 void gslc_ElemDestruct (gslc_tsElem * pElem)

Free up any members associated with an element.

Parameters

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

Returns

none

5.15.3.43 void gslc_ElemDraw (gslc_tsGui * pGui, int16_t nPageld, 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>nPageld</i>	ID of page containing element
in	<i>nElemId</i>	ID of element

Returns

none

5.15.3.44 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

5.15.3.45 bool gslc_ElemEvent (void * pvGui, gslc_tsEvent sEvent)

Common event handler function for an element.

Parameters

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

Returns

true if success, false if fail

5.15.3.46 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

5.15.3.47 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

5.15.3.48 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

5.15.3.49 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

5.15.3.50 `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

5.15.3.51 `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

5.15.3.52 `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

5.15.3.53 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
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

5.15.3.54 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

5.15.3.55 void gslc_ElemSetEventFunc (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, GSLC_CB_EVENT *funcCb*)

Assign the event callback function for a element.

Parameters

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

Returns

none

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

Set the fill state for an Element.

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

5.15.3.57 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

5.15.3.58 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

5.15.3.59 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

5.15.3.60 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

5.15.3.61 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

5.15.3.62 void `gslc_ElemSetImage` (`gslc_tsGui * pGui`, `gslc_tsElemRef * pElemRef`, `gslc_tslmgRef slmgRef`, `gslc_tslmgRef slmgRefSel`)

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>slmgRef</i>	Image reference (normal state)
in	<i>slmgRefSel</i>	Image reference (glowing state)

Returns

none

5.15.3.63 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

5.15.3.64 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

5.15.3.65 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

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

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

Parameters

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

Returns

none

5.15.3.67 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

5.15.3.68 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

5.15.3.69 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

5.15.3.70 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

5.15.3.71 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

5.15.3.72 **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

5.15.3.73 **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) of contract the width (if negative)
in	<i>nExpandH</i>	Number of pixels to expand the height (if positive) of contract the height (if negative)

Returns

[gslc_tsRect\(\)](#) with resized dimensions

5.15.3.74 **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

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

Fetch a font from its ID value.

Parameters

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

Returns

A pointer to the font structure or NULL if error

5.15.3.76 `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

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

5.15.3.78 `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

5.15.3.79 `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>pImgBuf</i>	Pointer to image buffer in memory
in	<i>eFmt</i>	Image format

Returns

Loaded image reference

5.15.3.80 `gslc_tslmgRef gslc_GetImageFromRam (unsigned char * pImgBuf, gslc_telmgRefFlags eFmt)`

Create an image reference to a bitmap in SRAM.

Parameters

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

Returns

Loaded image reference

5.15.3.81 `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

5.15.3.82 `int gslc_GetPageCur (gslc_tsGui * pGui)`

Fetch the current page ID.

Parameters

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

Returns

Page ID

5.15.3.83 `bool gslc_GetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress)`

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

Returns

true if touch event, false otherwise

5.15.3.84 char* gslc_GetVer (gslc_tsGui * pGui)

Get the GUIslice version number.

Parameters

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

Returns

String containing version number

5.15.3.85 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

5.15.3.86 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

5.15.3.87 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

5.15.3.88 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

5.15.3.89 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

5.15.3.90 `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

5.15.3.91 `void gslc_OrderCoord (int16_t * pnX0, int16_t * pnY0, int16_t * pnX1, int16_t * pnY1)`

5.15.3.92 `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.

- 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>nPageId</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

5.15.3.93 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

5.15.3.94 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

5.15.3.95 gslc_tsPage* gslc_PageFindById (gslc_tsGui * *pGui*, int16_t *nPageId*)

Find a page in the GUI by its ID.

Parameters

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

Returns

Ptr to a page or NULL if none found

5.15.3.96 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.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nPageId</i>	Page ID to search
in	<i>nElemId</i>	Element ID to search

Returns

Ptr to an element or NULL if none found

5.15.3.97 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

5.15.3.98 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

5.15.3.99 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

5.15.3.100 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

5.15.3.101 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

5.15.3.102 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

5.15.3.103 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

5.15.3.104 void gslc_PageSetEventFunc (gslc_tsGui * *pGui*, gslc_tsPage * *pPage*, GSLC_CB_EVENT *funcCb*)

Assign the event callback function for a page.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pPage</i>	Pointer to page
in	<i>funcCb</i>	Function pointer to event routine (or NULL for default)

Returns

none

5.15.3.105 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

5.15.3.106 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

5.15.3.107 void `gslc_ResetElem` (`gslc_tsElem` * *pElem*)

Initialize an Element struct.

Parameters

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

Returns

none

5.15.3.108 void `gslc_ResetFont` (`gslc_tsFont` * *pFont*)

Initialize a Font struct.

Parameters

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

Returns

none

5.15.3.109 `gslc_tslmgRef gslc_ResetImage ()`

Create a blank image reference structure.

Returns

Image reference struct

5.15.3.110 `bool gslc_SetBkgndColor (gslc_tsGui * pGui, gslc_tsColor nCol)`

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

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

Returns

true if success, false if fail

5.15.3.111 `bool gslc_SetBkgndImage (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

5.15.3.112 `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

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

5.15.3.114 void `gslc_SetPageCur` (`gslc_tsGui` * *pGui*, int16_t *nPageId*)

Select a new page for display.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nPageId</i>	Page ID to select as current

Returns

none

5.15.3.115 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.

5.15.3.116 void `gslc_SwapCoords` (int16_t * *pnXa*, int16_t * *pnYa*, int16_t * *pnXb*, int16_t * *pnYb*)

5.15.3.117 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

5.15.3.118 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

5.15.4 Variable Documentation**5.15.4.1** const char ERRSTR_NULL = "ERROR: %z() called with NULL ptr\n"**5.15.4.2** const char GSLC_PMEM ERRSTR_PXD_NULL[] = "ERROR: %z() pXData NULL\n"**5.15.4.3** GSLC_CB_DEBUG_OUT g_pfDebugOut = NULL

Global debug output function.

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

5.15.4.4 uint16_t m_nLUTSinF0X16**Initial value:**

```
= {
    0x0000, 0x0192, 0x0324, 0x04B6, 0x0648, 0x07DA, 0x096C, 0x0AFD, 0x0C8F, 0x0E21, 0x0FB2, 0x1143, 0x12D5, 0x1465, 0x15F6,
    0x1787,
    0x1917, 0x1AA7, 0x1C37, 0x1DC6, 0x1F56, 0x20E5, 0x2273, 0x2402, 0x258F, 0x271D, 0x28AA, 0x2A37, 0x2BC3, 0x2D4F, 0x2EDB,
    0x3066,
    0x31F1, 0x337B, 0x3505, 0x368E, 0x3816, 0x399E, 0x3B26, 0x3CAD, 0x3E33, 0x3FB9, 0x413E, 0x42C3, 0x4447, 0x45CA, 0x474C,
    0x48CE,
    0x4A4F, 0x4BD0, 0x4D4F, 0x4ECE, 0x504D, 0x51CA, 0x5347, 0x54C3, 0x563E, 0x57B8, 0x5931, 0x5AAA, 0x5C21, 0x5D98, 0x5F0E,
    0x6083,
    0x61F7, 0x636A, 0x64DC, 0x664D, 0x67BD, 0x692C, 0x6A9A, 0x6C07, 0x6D73, 0x6EDE, 0x7048, 0x71B1, 0x7319, 0x747F, 0x75E5,
    0x7749,
    0x78AC, 0x7A0F, 0x7B6F, 0x7CCF, 0x7E2E, 0x7F8B, 0x80E7, 0x8242, 0x839B, 0x84F3, 0x864A, 0x87A0, 0x88F5, 0x8A48, 0x8B99,
    0x8CEA,
    0x8E39, 0x8F86, 0x90D3, 0x921E, 0x9367, 0x94AF, 0x95F6, 0x973B, 0x987F, 0x99C1, 0x9B02, 0x9C41, 0x9D7F, 0x9EBB, 0x9FF6,
    0xA12F,
    0xA266, 0xA39D, 0xA4D1, 0xA604, 0xA735, 0xA865, 0xA993, 0xAABF, 0xABEA, 0xAD13, 0xAE3B, 0xAF60, 0xB085, 0xB1A7, 0xB2C8,
    0xB3E7,
    0xB504, 0xB61F, 0xB739, 0xB851, 0xB967, 0xBA7B, 0xBB8E, 0xBC9F, 0xBDAE, 0xBEBB, 0xBF6C, 0xC0D0, 0xC1D7, 0xC2DD, 0xC3E1,
    0xC4E3,
    0xC5E3, 0xC6E1, 0xC7DD, 0xC8D7, 0xC9D0, 0xCAC6, 0xCBBB, 0xCCAD, 0xCD9E, 0xCE8C, 0xCF79, 0xD063, 0xD14C, 0xD232, 0xD317,
    0xD3F9,
    0xD4DA, 0xD5B8, 0xD695, 0xD76F, 0xD847, 0xD91D, 0xD9F1, 0xDAC3, 0xDB93, 0xDC60, 0xDD2C, 0xDDF5, 0xDEBD, 0xDF82, 0xE045,
    0xE106,
    0xE1C4, 0xE281, 0xE33B, 0xE3F3, 0xE4A9, 0xE55D, 0xE60E, 0xE6BD, 0xE76A, 0xE815, 0xE8BE, 0xE964, 0xEA08, 0xEAAA, 0xEB4A,
    0xEBE7,
    0xEC82, 0xED1B, 0xEDB1, 0xEE45, 0xEED7, 0xEF67, 0xEFF4, 0xF07F, 0xF108, 0xF18E, 0xF212, 0xF294, 0xF313, 0xF390, 0xF40A,
    0xF483,
    0xF4F9, 0xF56C, 0xF5DD, 0xF64C, 0xF6B9, 0xF723, 0xF78A, 0xF7F0, 0xF853, 0xF8B3, 0xF911, 0xF96D, 0xF9C6, 0xFA1D, 0xFA72,
```


- struct [gslc_tsElem](#)
Element Struct.
- struct [gslc_tsCollect](#)
Element collection struct.
- struct [gslc_tsPage](#)
Page structure.
- struct [gslc_tsGui](#)
GUI structure.

Macros

- #define [GSLC_PMEM](#)
- #define [GSLC_2PI](#) 6.28318530718
- #define [GSLC_ELEM_FEA_VALID](#) 0x80
Element features type.
- #define [GSLC_ELEM_FEA_CLICK_EN](#) 0x08
Element accepts touch presses.
- #define [GSLC_ELEM_FEA_GLOW_EN](#) 0x04
Element supports glowing state.
- #define [GSLC_ELEM_FEA_FRAME_EN](#) 0x02
Element is drawn with a frame.
- #define [GSLC_ELEM_FEA_FILL_EN](#) 0x01
Element is drawn with a fill.
- #define [GSLC_ELEM_FEA_NONE](#) 0x00
Element default (no features set)
- #define [GSLC_ALIGNV_TOP](#) 0x10
Element text alignment.
- #define [GSLC_ALIGNV_MID](#) 0x20
Vertical align to middle.
- #define [GSLC_ALIGNV_BOT](#) 0x40
Vertical align to bottom.
- #define [GSLC_ALIGNH_LEFT](#) 0x01
Horizontal align to left.
- #define [GSLC_ALIGNH_MID](#) 0x02
Horizontal align to middle.
- #define [GSLC_ALIGNH_RIGHT](#) 0x04
Horizontal align to right.
- #define [GSLC_ALIGN_TOP_LEFT](#) [GSLC_ALIGNH_LEFT](#) | [GSLC_ALIGNV_TOP](#)
Align to top-left.
- #define [GSLC_ALIGN_TOP_MID](#) [GSLC_ALIGNH_MID](#) | [GSLC_ALIGNV_TOP](#)
Align to middle of top.
- #define [GSLC_ALIGN_TOP_RIGHT](#) [GSLC_ALIGNH_RIGHT](#) | [GSLC_ALIGNV_TOP](#)
Align to top-right.
- #define [GSLC_ALIGN_MID_LEFT](#) [GSLC_ALIGNH_LEFT](#) | [GSLC_ALIGNV_MID](#)
Align to middle of left side.
- #define [GSLC_ALIGN_MID_MID](#) [GSLC_ALIGNH_MID](#) | [GSLC_ALIGNV_MID](#)
Align to center.
- #define [GSLC_ALIGN_MID_RIGHT](#) [GSLC_ALIGNH_RIGHT](#) | [GSLC_ALIGNV_MID](#)
Align to middle of right side.
- #define [GSLC_ALIGN_BOT_LEFT](#) [GSLC_ALIGNH_LEFT](#) | [GSLC_ALIGNV_BOT](#)

- Align to bottom-left.*

 - #define `GSLC_ALIGN_BOT_MID GSLC_ALIGNH_MID | GSLC_ALIGNV_BOT`
- Align to middle of bottom.*

 - #define `GSLC_ALIGN_BOT_RIGHT GSLC_ALIGNH_RIGHT | GSLC_ALIGNV_BOT`
- Align to bottom-right.*

 - #define `GSLC_COL_RED_DK4 (gslc_tsColor) {128, 0, 0}`
- Basic color definition.*

 - #define `GSLC_COL_RED_DK3 (gslc_tsColor) {160, 0, 0}`
- Red (dark3)*

 - #define `GSLC_COL_RED_DK2 (gslc_tsColor) {192, 0, 0}`
- Red (dark2)*

 - #define `GSLC_COL_RED_DK1 (gslc_tsColor) {224, 0, 0}`
- Red (dark1)*

 - #define `GSLC_COL_RED (gslc_tsColor) {255, 0, 0}`
- Red.*

 - #define `GSLC_COL_RED_LT1 (gslc_tsColor) {255, 32, 32}`
- Red (light1)*

 - #define `GSLC_COL_RED_LT2 (gslc_tsColor) {255, 64, 64}`
- Red (light2)*

 - #define `GSLC_COL_RED_LT3 (gslc_tsColor) {255, 96, 96}`
- Red (light3)*

 - #define `GSLC_COL_RED_LT4 (gslc_tsColor) {255, 128, 128}`
- Red (light4)*

 - #define `GSLC_COL_GREEN_DK4 (gslc_tsColor) { 0, 128, 0}`
- Green (dark4)*

 - #define `GSLC_COL_GREEN_DK3 (gslc_tsColor) { 0, 160, 0}`
- Green (dark3)*

 - #define `GSLC_COL_GREEN_DK2 (gslc_tsColor) { 0, 192, 0}`
- Green (dark2)*

 - #define `GSLC_COL_GREEN_DK1 (gslc_tsColor) { 0, 224, 0}`
- Green (dark1)*

 - #define `GSLC_COL_GREEN (gslc_tsColor) { 0, 255, 0}`
- Green.*

 - #define `GSLC_COL_GREEN_LT1 (gslc_tsColor) { 32, 255, 32}`
- Green (light1)*

 - #define `GSLC_COL_GREEN_LT2 (gslc_tsColor) { 64, 255, 64}`
- Green (light2)*

 - #define `GSLC_COL_GREEN_LT3 (gslc_tsColor) { 96, 255, 96}`
- Green (light3)*

 - #define `GSLC_COL_GREEN_LT4 (gslc_tsColor) {128, 255, 128}`
- Green (light4)*

 - #define `GSLC_COL_BLUE_DK4 (gslc_tsColor) { 0, 0, 128}`
- Blue (dark4)*

 - #define `GSLC_COL_BLUE_DK3 (gslc_tsColor) { 0, 0, 160}`
- Blue (dark3)*

 - #define `GSLC_COL_BLUE_DK2 (gslc_tsColor) { 0, 0, 192}`
- Blue (dark2)*

 - #define `GSLC_COL_BLUE_DK1 (gslc_tsColor) { 0, 0, 224}`
- Blue (dark1)*

 - #define `GSLC_COL_BLUE (gslc_tsColor) { 0, 0, 255}`
- Blue.*

- #define `GSLC_COL_BLUE_LT1` (`gslc_tsColor`) { 32, 32,255}
Blue (light1)
- #define `GSLC_COL_BLUE_LT2` (`gslc_tsColor`) { 64, 64,255}
Blue (light2)
- #define `GSLC_COL_BLUE_LT3` (`gslc_tsColor`) { 96, 96,255}
Blue (light3)
- #define `GSLC_COL_BLUE_LT4` (`gslc_tsColor`) {128,128,255}
Blue (light4)
- #define `GSLC_COL_BLACK` (`gslc_tsColor`) { 0, 0, 0}
Black.
- #define `GSLC_COL_GRAY_DK3` (`gslc_tsColor`) { 32, 32, 32}
Gray (dark)
- #define `GSLC_COL_GRAY_DK2` (`gslc_tsColor`) { 64, 64, 64}
Gray (dark)
- #define `GSLC_COL_GRAY_DK1` (`gslc_tsColor`) { 96, 96, 96}
Gray (dark)
- #define `GSLC_COL_GRAY` (`gslc_tsColor`) {128,128,128}
Gray.
- #define `GSLC_COL_GRAY_LT1` (`gslc_tsColor`) {160,160,160}
Gray (light1)
- #define `GSLC_COL_GRAY_LT2` (`gslc_tsColor`) {192,192,192}
Gray (light2)
- #define `GSLC_COL_GRAY_LT3` (`gslc_tsColor`) {224,224,224}
Gray (light3)
- #define `GSLC_COL_WHITE` (`gslc_tsColor`) {255,255,255}
White.
- #define `GSLC_COL_YELLOW` (`gslc_tsColor`) {255,255,0}
Yellow.
- #define `GSLC_COL_YELLOW_DK` (`gslc_tsColor`) {64,64,0}
Yellow (dark)
- #define `GSLC_COL_PURPLE` (`gslc_tsColor`) {128,0,128}
Purple.
- #define `GSLC_COL_CYAN` (`gslc_tsColor`) {0,255,255}
Cyan.
- #define `GSLC_COL_MAGENTA` (`gslc_tsColor`) {255,0,255}
Magenta.
- #define `GSLC_COL_TEAL` (`gslc_tsColor`) {0,128,128}
Teal.
- #define `GSLC_COL_ORANGE` (`gslc_tsColor`) {255,165,0}
Orange.
- #define `GSLC_COL_BROWN` (`gslc_tsColor`) {165,42,42}
Brown.
- #define `GSLC_COLMONO_BLACK` (`gslc_tsColor`) {255,255,255}
Black.
- #define `GSLC_COLMONO_WHITE` (`gslc_tsColor`) { 0, 0, 0}
White.
- #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, col←
Fill, 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_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 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](#) = 0, [GSLC_ID_NONE](#) = -1999, [GSLC_ID_AUTO](#), [GSLC_ID_TEMP](#),
[GSLC_ID_AUTO_BASE](#) = 16384 }

Element ID enumerations.

- enum [gslc_tePageId](#) { [GSLC_PAGE_USER_BASE](#) = 0, [GSLC_PAGE_NONE](#) = -2999 }

Page ID enumerations.

- enum [gslc_teGroupId](#) { [GSLC_GROUP_ID_USER_BASE](#) = 0, [GSLC_GROUP_ID_NONE](#) = -6999 }

Group ID enumerations.

- enum [gslc_teFontId](#) { [GSLC_FONT_USER_BASE](#) = 0, [GSLC_FONT_NONE](#) = -4999 }

Font ID enumerations.

- enum [gslc_teElemInd](#) { [GSLC_IND_NONE](#) = -9999, [GSLC_IND_FIRST](#) = 0 }

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` = 0x1000 }
Element type.
- enum `gslc_teTouch` {
`GSLC_TOUCH_NONE` = 0, `GSLC_TOUCH_DOWN` = (1<<4), `GSLC_TOUCH_MOVE` = (1<<5), `GSLC_TOUCH_UP` = (1<<6),
`GSLC_TOUCH_IN` = (1<<0), `GSLC_TOUCH_OUT` = (1<<1), `GSLC_TOUCH_INOUT_MASK` = `GSLC_TOUCH_IN` | `GSLC_TOUCH_OUT`,
`GSLC_TOUCH_DOWN_IN` = `GSLC_TOUCH_DOWN` | `GSLC_TOUCH_IN`,
`GSLC_TOUCH_MOVE_IN` = `GSLC_TOUCH_MOVE` | `GSLC_TOUCH_IN`, `GSLC_TOUCH_MOVE_OUT` = `GSLC_TOUCH_MOVE` | `GSLC_TOUCH_OUT`,
`GSLC_TOUCH_UP_IN` = `GSLC_TOUCH_UP` | `GSLC_TOUCH_IN`, `GSLC_TOUCH_UP_OUT` = `GSLC_TOUCH_UP` | `GSLC_TOUCH_OUT` }
Touch event type for element touch tracking.
- enum `gslc_teEventType` {
`GSLC_EVT_NONE`, `GSLC_EVT_DRAW`, `GSLC_EVT_TOUCH`, `GSLC_EVT_TICK`,
`GSLV_EVT_CUSTOM` }
Event types.
- enum `gslc_teEventSubType` { `GSLC_EVTSUB_NONE`, `GSLC_EVTSUB_DRAW_NEEDED`, `GSLC_EVTSUB_DRAW_FORCE` }
Event sub-types.
- enum `gslc_teRedrawType` { `GSLC_REDRAW_NONE`, `GSLC_REDRAW_FULL`, `GSLC_REDRAW_INC` }
Redraw types.
- enum `gslc_teFontRefType` { `GSLC_FONTREF_FNAME`, `GSLC_FONTREF_PTR` }
Font Reference types.
- enum `gslc_teElemRefFlags` {
`GSLC_ELEMREF_NONE` = 0, `GSLC_ELEMREF_SRC_RAM` = (1<<0), `GSLC_ELEMREF_SRC_PROG` = (2<<0), `GSLC_ELEMREF_SRC_CONST` = (3<<0),
`GSLC_ELEMREF_REDRAW_NONE` = (0<<4), `GSLC_ELEMREF_REDRAW_FULL` = (1<<4), `GSLC_ELEMREF_REDRAW_INC` = (2<<4),
`GSLC_ELEMREF_GLOWING` = (1<<6),
`GSLC_ELEMREF_SRC` = (3<<0), `GSLC_ELEMREF_REDRAW_MASK` = (3<<4) }
Element reference flags: Describes characteristics of an element.
- enum `gslc_telmgRefFlags` {
`GSLC_IMGREF_NONE` = 0, `GSLC_IMGREF_SRC_FILE` = (1<<0), `GSLC_IMGREF_SRC_SD` = (2<<0),
`GSLC_IMGREF_SRC_RAM` = (3<<0),
`GSLC_IMGREF_SRC_PROG` = (4<<0), `GSLC_IMGREF_FMT_BMP24` = (1<<4), `GSLC_IMGREF_FMT_BMP16` = (2<<4),
`GSLC_IMGREF_FMT_RAW1` = (3<<4),
`GSLC_IMGREF_SRC` = (7<<0), `GSLC_IMGREF_FMT` = (7<<4) }
Image reference flags: Describes characteristics of an image reference.
- enum `gslc_teTxtFlags` {
`GSLC_TXT_MEM_RAM` = (0<<0), `GSLC_TXT_MEM_PROG` = (1<<0), `GSLC_TXT_ALLOC_NONE` = (0<<2),
`GSLC_TXT_ALLOC_INT` = (1<<2),
`GSLC_TXT_ALLOC_EXT` = (2<<2), `GSLC_TXT_MEM` = (3<<0), `GSLC_TXT_ALLOC` = (3<<2), `GSLC_TXT_DEFAULT` = `GSLC_TXT_MEM_RAM` | `GSLC_TXT_ALLOC_NONE` }
Text reference flags: Describes the characteristics of a text string (ie.

Functions

- char * `gslc_GetVer` (`gslc_tsGui` *pGui)
Get the GUISlice version number.
- 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.
- 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 *pv↵ Scope, 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.
- [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_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)
- 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.
- bool [gslc_PageEvent](#) (void *pvGui, [gslc_tsEvent](#) sEvent)
Common event handler function for a page.
- void [gslc_PageSetEventFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsPage](#) *pPage, [GSLC_CB_EVENT](#) funcCb)
Assign the event callback function for a page.
- int [gslc_GetPageCur](#) ([gslc_tsGui](#) *pGui)
Fetch the current page ID.
- void [gslc_SetPageCur](#) ([gslc_tsGui](#) *pGui, int16_t nPageId)
Select a new page for display.
- 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_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.

- 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_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.

- void `gslc_PageRedrawCalc` (`gslc_tsGui` *pGui)

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

- uint8_t `gslc_GetElemRefFlag` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pElemRef, uint8_t nFlagMask)
- void `gslc_SetElemRefFlag` (`gslc_tsGui` *pGui, `gslc_tsElemRef` *pElemRef, uint8_t nFlagMask, uint8_t nFlagVal)
- `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.

- 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_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_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_ElemSetEventFunc](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [GSLC_CB_EVENT](#) funcCb)
Assign the event callback function for a 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.
- bool [gslc_ElemEvent](#) (void *pvGui, [gslc_tsEvent](#) sEvent)
Common event handler function for an element.
- void [gslc_ElemDraw](#) ([gslc_tsGui](#) *pGui, int16_t nPageId, int16_t nElemId)
Draw an element to the active display.
- void [gslc_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.
- `void gslc_CollectSetParent (gslc_tsGui *pGui, gslc_tsCollect *pCollect, gslc_tsElemRef *pElemRefParent)`
Assign the parent element reference to all elements within a collection.
- `void gslc_CollectSetEventFunc (gslc_tsGui *pGui, gslc_tsCollect *pCollect, GSLC_CB_EVENT funcCb)`
Assign the event callback function for an element 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_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)`
Initialize the touchscreen device driver.
- `gslc_tsElem gslc_ElemCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPageId, int16_t nType, gslc_tsRect rElem, char *pStrBuf, uint8_t nStrBufMax, int16_t nFontId)`
Create a new element with default styling.
- `gslc_tsElemRef * gslc_ElemAdd (gslc_tsGui *pGui, int16_t nPageId, gslc_tsElem *pElem, gslc_tsElemRefFlags eFlags)`
Add the Element to the list of generated elements in the GUI environment.
- `bool gslc_SetClipRect (gslc_tsGui *pGui, gslc_tsRect *pRect)`
Set the clipping rectangle for further drawing.
- `void gslc_ElemSetImage (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsImgRef sImgRef, gslc_tsImgRefSel sImgRefSel)`
Set an element to use a bitmap image.
- `bool gslc_SetBkgndImage (gslc_tsGui *pGui, gslc_tsImgRef sImgRef)`
Configure the background to use a bitmap image.
- `bool gslc_SetBkgndColor (gslc_tsGui *pGui, gslc_tsColor nCol)`
Configure the background to use a solid color.
- `bool gslc_ElemDrawByRef (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef, gslc_tsRedrawType eRedraw)`
Draw an element to the active display.
- `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.
- `bool gslc_ElemSendEventTouch (gslc_tsGui *pGui, gslc_tsElemRef *pElemRefTracked, gslc_tsEventTouch eTouch, int16_t nX, int16_t nY)`
Trigger an element's touch event.
- `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.

5.16.1 Macro Definition Documentation

5.16.1.1 `#define GSLC_2PI 6.28318530718`

5.16.1.2 `#define GSLC_ALIGN_BOT_LEFT GSLC_ALIGNH_LEFT | GSLC_ALIGNV_BOT`

Align to bottom-left.

5.16.1.3 `#define GSLC_ALIGN_BOT_MID GSLC_ALIGNH_MID | GSLC_ALIGNV_BOT`

Align to middle of bottom.

5.16.1.4 `#define GSLC_ALIGN_BOT_RIGHT GSLC_ALIGNH_RIGHT | GSLC_ALIGNV_BOT`

Align to bottom-right.

5.16.1.5 `#define GSLC_ALIGN_MID_LEFT GSLC_ALIGNH_LEFT | GSLC_ALIGNV_MID`

Align to middle of left side.

5.16.1.6 `#define GSLC_ALIGN_MID_MID GSLC_ALIGNH_MID | GSLC_ALIGNV_MID`

Align to center.

5.16.1.7 `#define GSLC_ALIGN_MID_RIGHT GSLC_ALIGNH_RIGHT | GSLC_ALIGNV_MID`

Align to middle of right side.

5.16.1.8 `#define GSLC_ALIGN_TOP_LEFT GSLC_ALIGNH_LEFT | GSLC_ALIGNV_TOP`

Align to top-left.

5.16.1.9 `#define GSLC_ALIGN_TOP_MID GSLC_ALIGNH_MID | GSLC_ALIGNV_TOP`

Align to middle of top.

5.16.1.10 `#define GSLC_ALIGN_TOP_RIGHT GSLC_ALIGNH_RIGHT | GSLC_ALIGNV_TOP`

Align to top-right.

5.16.1.11 `#define GSLC_ALIGNH_LEFT 0x01`

Horizontal align to left.

5.16.1.12 `#define GSLC_ALIGNH_MID 0x02`

Horizontal align to middle.

5.16.1.13 `#define GSLC_ALIGNH_RIGHT 0x04`

Horizontal align to right.

5.16.1.14 `#define GSLC_ALIGNV_BOT 0x40`

Vertical align to bottom.

5.16.1.15 `#define GSLC_ALIGNV_MID 0x20`

Vertical align to middle.

5.16.1.16 `#define GSLC_ALIGNV_TOP 0x10`

Element text alignment.

Vertical align to top

5.16.1.17 `#define GSLC_COL_BLACK (gslc_tsColor) { 0, 0, 0}`

Black.

5.16.1.18 `#define GSLC_COL_BLUE (gslc_tsColor) { 0, 0, 255}`

Blue.

5.16.1.19 `#define GSLC_COL_BLUE_DK1 (gslc_tsColor) { 0, 0, 224}`

Blue (dark1)

5.16.1.20 `#define GSLC_COL_BLUE_DK2 (gslc_tsColor) { 0, 0, 192}`

Blue (dark2)

5.16.1.21 `#define GSLC_COL_BLUE_DK3 (gslc_tsColor) { 0, 0, 160}`

Blue (dark3)

5.16.1.22 `#define GSLC_COL_BLUE_DK4 (gslc_tsColor) { 0, 0, 128}`

Blue (dark4)

5.16.1.23 `#define GSLC_COL_BLUE_LT1 (gslc_tsColor) { 32, 32, 255}`

Blue (light1)

5.16.1.24 `#define GSLC_COL_BLUE_LT2 (gslc_tsColor) { 64, 64, 255}`

Blue (light2)

5.16.1.25 `#define GSLC_COL_BLUE_LT3 (gslc_tsColor) { 96, 96, 255}`

Blue (light3)

5.16.1.26 `#define GSLC_COL_BLUE_LT4 (gslc_tsColor) {128, 128, 255}`

Blue (light4)

5.16.1.27 `#define GSLC_COL_BROWN (gslc_tsColor) {165, 42, 42}`

Brown.

5.16.1.28 `#define GSLC_COL_CYAN (gslc_tsColor) {0, 255, 255}`

Cyan.

5.16.1.29 `#define GSLC_COL_GRAY (gslc_tsColor) {128, 128, 128}`

Gray.

5.16.1.30 `#define GSLC_COL_GRAY_DK1 (gslc_tsColor) { 96, 96, 96}`

Gray (dark)

5.16.1.31 `#define GSLC_COL_GRAY_DK2 (gslc_tsColor) { 64, 64, 64}`

Gray (dark)

5.16.1.32 `#define GSLC_COL_GRAY_DK3 (gslc_tsColor) { 32, 32, 32}`

Gray (dark)

5.16.1.33 `#define GSLC_COL_GRAY_LT1 (gslc_tsColor) {160, 160, 160}`

Gray (light1)

5.16.1.34 `#define GSLC_COL_GRAY_LT2 (gslc_tsColor) {192, 192, 192}`

Gray (light2)

5.16.1.35 `#define GSLC_COL_GRAY_LT3 (gslc_tsColor) {224, 224, 224}`

Gray (light3)

5.16.1.36 `#define GSLC_COL_GREEN (gslc_tsColor) { 0,255, 0}`

Green.

5.16.1.37 `#define GSLC_COL_GREEN_DK1 (gslc_tsColor) { 0,224, 0}`

Green (dark1)

5.16.1.38 `#define GSLC_COL_GREEN_DK2 (gslc_tsColor) { 0,192, 0}`

Green (dark2)

5.16.1.39 `#define GSLC_COL_GREEN_DK3 (gslc_tsColor) { 0,160, 0}`

Green (dark3)

5.16.1.40 `#define GSLC_COL_GREEN_DK4 (gslc_tsColor) { 0,128, 0}`

Green (dark4)

5.16.1.41 `#define GSLC_COL_GREEN_LT1 (gslc_tsColor) { 32,255, 32}`

Green (light1)

5.16.1.42 `#define GSLC_COL_GREEN_LT2 (gslc_tsColor) { 64,255, 64}`

Green (light2)

5.16.1.43 `#define GSLC_COL_GREEN_LT3 (gslc_tsColor) { 96,255, 96}`

Green (light3)

5.16.1.44 `#define GSLC_COL_GREEN_LT4 (gslc_tsColor) {128,255,128}`

Green (light4)

5.16.1.45 `#define GSLC_COL_MAGENTA (gslc_tsColor) {255,0,255}`

Magenta.

5.16.1.46 `#define GSLC_COL_ORANGE (gslc_tsColor) {255,165,0}`

Orange.

5.16.1.47 `#define GSLC_COL_PURPLE (gslc_tsColor) {128,0,128}`

Purple.

5.16.1.48 `#define GSLC_COL_RED (gslc_tsColor) {255, 0, 0}`

Red.

5.16.1.49 `#define GSLC_COL_RED_DK1 (gslc_tsColor) {224, 0, 0}`

Red (dark1)

5.16.1.50 `#define GSLC_COL_RED_DK2 (gslc_tsColor) {192, 0, 0}`

Red (dark2)

5.16.1.51 `#define GSLC_COL_RED_DK3 (gslc_tsColor) {160, 0, 0}`

Red (dark3)

5.16.1.52 `#define GSLC_COL_RED_DK4 (gslc_tsColor) {128, 0, 0}`

Basic color definition.

Red (dark4)

5.16.1.53 `#define GSLC_COL_RED_LT1 (gslc_tsColor) {255, 32, 32}`

Red (light1)

5.16.1.54 `#define GSLC_COL_RED_LT2 (gslc_tsColor) {255, 64, 64}`

Red (light2)

5.16.1.55 `#define GSLC_COL_RED_LT3 (gslc_tsColor) {255, 96, 96}`

Red (light3)

5.16.1.56 `#define GSLC_COL_RED_LT4 (gslc_tsColor) {255,128,128}`

Red (light4)

5.16.1.57 `#define GSLC_COL_TEAL (gslc_tsColor) {0,128,128}`

Teal.

5.16.1.58 `#define GSLC_COL_WHITE (gslc_tsColor) {255,255,255}`

White.

5.16.1.59 `#define GSLC_COL_YELLOW (gslc_tsColor) {255,255,0}`

Yellow.

5.16.1.60 `#define GSLC_COL_YELLOW_DK (gslc_tsColor) {64,64,0}`

Yellow (dark)

5.16.1.61 `#define GSLC_COLMONO_BLACK (gslc_tsColor) {255,255,255}`

Black.

5.16.1.62 `#define GSLC_COLMONO_WHITE (gslc_tsColor) { 0, 0, 0}`

White.

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

Value:

```
do {
    if (DEBUG_ERR) {
        gslc_DebugPrintf(sFmt, __VA_ARGS__);
    }
} while (0)
```

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	<i>sFmt</i>	Format string for debug message
----	-------------	---------------------------------

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

Value:

```
do {
    if (DEBUG_ERR) {
        gslc_DebugPrintf(sFmt, __VA_ARGS__);
    }
} while (0)
```

5.16.1.65 `#define GSLC_ELEM_FEA_CLICK_EN 0x08`

Element accepts touch presses.

5.16.1.66 `#define GSLC_ELEM_FEA_FILL_EN 0x01`

Element is drawn with a fill.

5.16.1.67 `#define GSLC_ELEM_FEA_FRAME_EN 0x02`

Element is drawn with a frame.

5.16.1.68 `#define GSLC_ELEM_FEA_GLOW_EN 0x04`

Element supports glowing state.

5.16.1.69 `#define GSLC_ELEM_FEA_NONE 0x00`

Element default (no features set))

5.16.1.70 `#define GSLC_ELEM_FEA_VALID 0x80`

Element features type.

Element record is valid

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

Value:

```
static const uint8_t nFeatures##nElemId = GSLC_ELEM_FEA_VALID | \
    (bFrameEn?GSLC_ELEM_FEA_FRAME_EN:0) | (bFillEn?
    GSLC_ELEM_FEA_FILL_EN:0); \
static const gslc_tsElem sElem##nElemId = {
    nElemId,
    nFeatures##nElemId,
    GSLC_TYPE_BOX,
    (gslc_tsRect){nX, nY, nW, nH},
    GSLC_GROUP_ID_NONE,
    colFrame, colFill, GSLC_COL_BLACK, GSLC_COL_BLACK,
    (gslc_tsImgRef){NULL, NULL, GSLC_IMGREF_NONE, NULL},
    (gslc_tsImgRef){NULL, NULL, GSLC_IMGREF_NONE, NULL},
    NULL,
    NULL,
    0,
    GSLC_TXT_DEFAULT,
    \
    GSLC_COL_WHITE,
    \
    GSLC_COL_WHITE,
    \
    GSLC_ALIGN_MID_MID,
    0,
    NULL,
    NULL,
    NULL,
    pfuncXDraw,
    NULL,
    pfuncXTick,
};
gslc_ElemAdd(pGui, nPage, (gslc_tsElem*)&sElem##nElemId,
    (gslc_tElemRefFlags)(GSLC_ELEMREF_SRC_CONST |
    GSLC_ELEMREF_REDRAW_FULL));
```

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)

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

Value:

```
static const char str##nElemId[] = strTxt;
static const uint8_t nFeatures##nElemId = GSLC_ELEM_FEA_VALID |
    \
    GSLC_ELEM_FEA_CLICK_EN | GSLC_ELEM_FEA_GLOW_EN |
    \
    (bFrameEn?GSLC_ELEM_FEA_FRAME_EN:0) | (bFillEn?
    GSLC_ELEM_FEA_FILL_EN:0); \
static const gslc_tsElem sElem##nElemId = {
    nElemId,
    nFeatures##nElemId,
    GSLC_TYPE_BTN,
    (gslc_tsRect){nX,nY,nW,nH},
    GSLC_GROUP_ID_NONE,
    colFrame,colFill,colFrameGlow,colFillGlow,
    (gslc_tsImgRef){NULL,NULL,GSLC_IMGREF_NONE,NULL},
    (gslc_tsImgRef){NULL,NULL,GSLC_IMGREF_NONE,NULL},
    NULL,
    (char*)str##nElemId,
    0,
    (gslc_teTxtFlags)(GSLC_TXT_MEM_RAM |
    GSLC_TXT_ALLOC_EXT), \
    colTxt,
    colTxt,
    nAlignTxt,
    0,
    pFont,
    (void*)extraData,
    NULL,
    NULL,
    callFunc,
    NULL,
};
gslc_ElemAdd(pGui,nPage,(gslc_tsElem*)&sElem##nElemId,
(gslc_teElemRefFlags)(GSLC_ELEMREF_SRC_CONST |
    GSLC_ELEMREF_REDRAW_FULL));
```

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

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

Value:

```
static const char str##nElemId[] = strTxt;
static const uint8_t nFeatures##nElemId = GSLC_ELEM_FEA_VALID | \
(bFrameEn?GSLC_ELEM_FEA_FRAME_EN:0) | (bFillEn?
GSLC_ELEM_FEA_FILL_EN:0); \
static const gslc_tsElem sElem##nElemId = {
    nElemId,
    nFeatures##nElemId,
    GSLC_TYPE_TXT,
    (gslc_tsRect){nX,nY,nW,nH},
    GSLC_GROUP_ID_NONE,
    colFrame,colFill,GSLC_COL_BLACK,GSLC_COL_BLACK,
    (gslc_tsImgRef){NULL,NULL,GSLC_IMGREF_NONE,NULL},
    (gslc_tsImgRef){NULL,NULL,GSLC_IMGREF_NONE,NULL},
    NULL,
    (char*)str##nElemId,
    0,
    (gslc_teTxtFlags)(GSLC_TXT_MEM_RAM |
GSLC_TXT_ALLOC_EXT), \
    colTxt,
    colTxt,
    nAlignTxt,
    0,
    pFont,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
};
gslc_ElemAdd(pGui,nPage,(gslc_tsElem*)&sElem##nElemId,
(gslc_teElemRefFlags)(GSLC_ELEMREF_SRC_CONST |
GSLC_ELEMREF_REDRAW_FULL));
```

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

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

Value:

```
static const uint8_t nFeatures##nElemId = GSLC_ELEM_FEA_VALID | \
(bFrameEn?GSLC_ELEM_FEA_FRAME_EN:0) | (bFillEn? \
GSLC_ELEM_FEA_FILL_EN:0); \
static const gslc_tsElem sElem##nElemId = {
    nElemId,
    nFeatures##nElemId,
    GSLC_TYPE_TXT,
    (gslc_tsRect){nX,nY,nW,nH},
    GSLC_GROUP_ID_NONE,
    colFrame,colFill,GSLC_COL_BLACK,GSLC_COL_BLACK,
    (gslc_tsImgRef){NULL,NULL,GSLC_IMGREF_NONE,NULL},
    (gslc_tsImgRef){NULL,NULL,GSLC_IMGREF_NONE,NULL},
    NULL,
    (char*)strTxt,
    strLength,
    (gslc_teTxtFlags)(GSLC_TXT_MEM_RAM |
GSLC_TXT_ALLOC_EXT),
    colTxt,
    colTxt,
    nAlignTxt,
    0,
    pFont,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
};
gslc_ElemAdd(pGui,nPage,(gslc_tsElem*)&sElem##nElemId,
(gslc_teElemRefFlags)(GSLC_ELEMREF_SRC_CONST |
GSLC_ELEMREF_REDRAW_FULL));
```

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

5.16.1.75 `#define GSLC_PMEM`

5.16.2 Typedef Documentation

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

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

Callback function for element drawing.

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

Callback function for element drawing.

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

Callback function for element tick.

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

Callback function for element touch tracking.

5.16.2.6 `typedef struct gslc_tsColor gslc_tsColor`

Color structure. Defines RGB triplet.

5.16.2.7 `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.

5.16.2.8 `typedef struct gslc_tsEvent gslc_tsEvent`

Event structure.

5.16.2.9 `typedef struct gslc_tsEventTouch gslc_tsEventTouch`

Structure used to pass touch data through event.

5.16.2.10 `typedef struct gslc_tsPt gslc_tsPt`

Define point coordinates.

5.16.2.11 `typedef struct gslc_tsRect gslc_tsRect`

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

5.16.3 Enumeration Type Documentation

5.16.3.1 `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)

5.16.3.2 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.

5.16.3.3 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_SRC` Mask for Source flags.

`GSLC_ELEMREF_REDRAW_MASK` Mask for Redraw flags.

5.16.3.4 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.

5.16.3.5 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.

5.16.3.6 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.

5.16.3.7 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.

5.16.3.8 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.

5.16.3.9 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.

5.16.3.10 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.

5.16.3.11 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.

5.16.3.12 enum `gslc_teTouch`

Touch event type for element touch tracking.

Enumerator

`GSLC_TOUCH_NONE` No touch event active.

`GSLC_TOUCH_DOWN` Touch event (down)

`GSLC_TOUCH_MOVE` Touch event (move)

`GSLC_TOUCH_UP` Touch event (up)

`GSLC_TOUCH_IN` Touch event inside element.

`GSLC_TOUCH_OUT` Touch event outside element.

`GSLC_TOUCH_INOUT_MASK` Mask for in/out state.

`GSLC_TOUCH_DOWN_IN` Touch down inside element (start tracking)

`GSLC_TOUCH_MOVE_IN` Touch move inside tracked element.

`GSLC_TOUCH_MOVE_OUT` Touch move outside tracked element.

`GSLC_TOUCH_UP_IN` Touch up inside tracked element.

`GSLC_TOUCH_UP_OUT` Touch up outside tracked element.

5.16.3.13 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_MEM Mask for updating text memory type.

GSLC_TXT_ALLOC Mask for updating location of text string buffer allocation.

GSLC_TXT_DEFAULT

5.16.3.14 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.

5.16.4 Function Documentation

5.16.4.1 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

5.16.4.2 `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

5.16.4.3 `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

5.16.4.4 `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

5.16.4.5 `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

5.16.4.6 `bool gslc_CollectEvent (void * pGui, gslc_tsEvent sEvent)`

Common event handler function for an element collection.

Parameters

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

Returns

true if success, false if fail

5.16.4.7 **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

5.16.4.8 **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.

- 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

5.16.4.9 **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

5.16.4.10 `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

5.16.4.11 `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

5.16.4.12 `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

5.16.4.13 `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

5.16.4.14 void gslc_CollectSetEventFunc (gslc_tsGui * *pGui*, gslc_tsCollect * *pCollect*, GSLC_CB_EVENT *funcCb*)

Assign the event callback function for an element collection.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pCollect</i>	Pointer to collection
in	<i>funcCb</i>	Function pointer to event routine (or NULL for default)

Returns

none

5.16.4.15 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

5.16.4.16 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

5.16.4.17 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

5.16.4.18 `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

5.16.4.19 `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.

5.16.4.20 `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.
- $\text{gslc_cosFX}(\text{nAngDeg} \times 64) / 32768.0 = \cos(\text{nAngDeg} \times 2\pi / 360)$

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.

5.16.4.21 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

5.16.4.22 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

5.16.4.23 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

5.16.4.24 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

5.16.4.25 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

5.16.4.26 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

5.16.4.27 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

5.16.4.28 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

5.16.4.29 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

5.16.4.30 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

5.16.4.31 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

5.16.4.32 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

5.16.4.33 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

5.16.4.34 `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

5.16.4.35 `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>nPageld</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

5.16.4.36 `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.

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

5.16.4.37 `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

5.16.4.38 `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

5.16.4.39 `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)
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
in	<i>cbTouch</i>	Callback for touch events

Returns

Pointer to the Element reference or NULL if failure

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

Create an image Element.

- Draws an image

Parameters

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

Returns

Pointer to the Element reference or NULL if failure

5.16.4.41 `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>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>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

5.16.4.42 `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

5.16.4.43 `void gslc_ElemDestruct (gslc_tsElem * pElem)`

Free up any members associated with an element.

Parameters

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

Returns

none

5.16.4.44 `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>nPageId</i>	ID of page containing element
in	<i>nElemId</i>	ID of element

Returns

none

5.16.4.45 `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

5.16.4.46 `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

5.16.4.47 `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

5.16.4.48 `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

5.16.4.49 `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

5.16.4.50 `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

5.16.4.51 `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

5.16.4.52 `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

5.16.4.53 `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

5.16.4.54 `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
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

5.16.4.55 `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

5.16.4.56 void gslc_ElemSetEventFunc (gslc_tsGui * *pGui*, gslc_tsElemRef * *pElemRef*, GSLC_CB_EVENT *funcCb*)

Assign the event callback function for a element.

Parameters

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

Returns

none

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

Set the fill state for an Element.

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

5.16.4.58 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

5.16.4.59 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

5.16.4.60 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

5.16.4.61 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

5.16.4.62 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

5.16.4.63 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

5.16.4.64 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

5.16.4.65 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

5.16.4.66 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

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

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

Parameters

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

Returns

none

5.16.4.68 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

5.16.4.69 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

5.16.4.70 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

5.16.4.71 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

5.16.4.72 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

5.16.4.73 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

5.16.4.74 `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

5.16.4.75 `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

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

Fetch a font from its ID value.

Parameters

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

Returns

A pointer to the font structure or NULL if error

5.16.4.77 `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

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

5.16.4.79 `gslc_tsImgRef 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

5.16.4.80 `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

5.16.4.81 `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

5.16.4.82 `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

5.16.4.83 `int gslc_GetPageCur (gslc_tsGui * pGui)`

Fetch the current page ID.

Parameters

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

Returns

Page ID

5.16.4.84 `bool gslc_GetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress)`

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

Returns

true if touch event, false otherwise

5.16.4.85 `char* gslc_GetVer (gslc_tsGui * pGui)`

Get the GUIslice version number.

Parameters

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

Returns

String containing version number

5.16.4.86 `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

5.16.4.87 `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

5.16.4.88 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

5.16.4.89 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

5.16.4.90 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

5.16.4.91 `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

5.16.4.92 `void gslc_PageAdd (gslc_tsGui * pGui, int16_t nPagId, 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>nPagId</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

5.16.4.93 `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

5.16.4.94 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

5.16.4.95 gslc_tsPage* gslc_PageFindById (gslc_tsGui * *pGui*, int16_t *nPageId*)

Find a page in the GUI by its ID.

Parameters

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

Returns

Ptr to a page or NULL if none found

5.16.4.96 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.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nPageId</i>	Page ID to search
in	<i>nElemId</i>	Element ID to search

Returns

Ptr to an element or NULL if none found

5.16.4.97 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

5.16.4.98 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

5.16.4.99 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

5.16.4.100 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

5.16.4.101 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

5.16.4.102 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

5.16.4.103 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

5.16.4.104 void gslc_PageSetEventFunc (gslc_tsGui * *pGui*, gslc_tsPage * *pPage*, GSLC_CB_EVENT *funcCb*)

Assign the event callback function for a page.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pPage</i>	Pointer to page
in	<i>funcCb</i>	Function pointer to event routine (or NULL for default)

Returns

none

5.16.4.105 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

5.16.4.106 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

5.16.4.107 void gslc_ResetElem (gslc_tsElem * pElem)

Initialize an Element struct.

Parameters

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

Returns

none

5.16.4.108 void gslc_ResetFont (gslc_tsFont * pFont)

Initialize a Font struct.

Parameters

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

Returns

none

5.16.4.109 gslc_tsImgRef gslc_ResetImage ()

Create a blank image reference structure.

Returns

Image reference struct

5.16.4.110 `bool gslc_SetBkgndColor (gslc_tsGui * pGui, gslc_tsColor nCol)`

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

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

Returns

true if success, false if fail

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

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

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

Returns

true if success, false if fail

5.16.4.112 `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

5.16.4.113 `void gslc_SetElemRefFlag (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, uint8_t nFlagMask, uint8_t nFlagVal)`5.16.4.114 `void gslc_SetPageCur (gslc_tsGui * pGui, int16_t nPageId)`

Select a new page for display.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nPageId</i>	Page ID to select as current

Returns

none

5.16.4.115 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.

5.16.4.116 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

5.16.4.117 void gslc_Update (gslc_tsGui * pGui)

Perform main GUIslice handling functions.

- Handles any touch events
- Performs any necessary screen redraw

- `#define ADATOUCH_X_MAX 3800`
- `#define ADATOUCH_Y_MAX 3700`
- `#define ADATOUCH_SWAP_XY 1`
- `#define ADATOUCH_FLIP_X 0`
- `#define ADATOUCH_FLIP_Y 1`
- `#define GSLC_TOUCH_MAX_EVT 1`
- `#define DEBUG_ERR 1`
- `#define GSLC_FEATURE_COMPOUND 0`
- `#define GSLC_FEATURE_XGAUGE_RADIAL 0`
- `#define GSLC_FEATURE_XGAUGE_RAMP 0`
- `#define GSLC_SD_EN 0`
- `#define GSLC_SD_BUFFPIXEL 50`
- `#define GSLC_CLIP_EN 1`
- `#define GSLC_BMP_TRANS_EN 1`
- `#define GSLC_BMP_TRANS_RGB 0xFF,0x00,0xFF`
- `#define GSLC_LOCAL_STR 0`
- `#define GSLC_LOCAL_STR_LEN 30`
- `#define GSLC_USE_FLOAT 0`
- `#define GSLC_DEV_TOUCH ""`
- `#define GSLC_USE_PROGMEM 0`

5.18.1 Macro Definition Documentation

5.18.1.1 `#define ADAGFX_PIN_CLK`

5.18.1.2 `#define ADAGFX_PIN_CS 10`

5.18.1.3 `#define ADAGFX_PIN_DC 9`

5.18.1.4 `#define ADAGFX_PIN_MISO`

5.18.1.5 `#define ADAGFX_PIN_MOSI`

5.18.1.6 `#define ADAGFX_PIN_RD A0`

5.18.1.7 `#define ADAGFX_PIN_RST 0`

5.18.1.8 `#define ADAGFX_PIN_SDCS 4`

5.18.1.9 `#define ADAGFX_PIN_WR A1`

5.18.1.10 `#define ADAGFX_SPI_HW 1`

5.18.1.11 `#define ADATOUCH_FLIP_X 0`

5.18.1.12 `#define ADATOUCH_FLIP_Y 1`

5.18.1.13 `#define ADATOUCH_I2C_ADDR 0x41`

5.18.1.14 `#define ADATOUCH_I2C_HW 0`

5.18.1.15 `#define ADATOUCH_PIN_CS 8`

5.18.1.16 `#define ADATOUCH_SPI_HW 1`

5.18.1.17 `#define ADATOUCH_SPI_SW 0`

5.18.1.18 `#define ADATOUCH_SWAP_XY 1`

5.18.1.19 `#define ADATOUCH_X_MAX 3800`

5.18.1.20 `#define ADATOUCH_X_MIN 230`

5.18.1.21 `#define ADATOUCH_Y_MAX 3700`

5.18.1.22 `#define ADATOUCH_Y_MIN 260`

5.18.1.23 `#define DEBUG_ERR 1`

5.18.1.24 `#define DRV_DISP_ADAGFX`

5.18.1.25 `#define DRV_DISP_ADAGFX_ILI9341`

5.18.1.26 `#define DRV_TOUCH_ADA_STMPE610`

5.18.1.27 `#define GSLC_BMP_TRANS_EN 1`

5.18.1.28 `#define GSLC_BMP_TRANS_RGB 0xFF,0x00,0xFF`

5.18.1.29 `#define GSLC_CLIP_EN 1`

5.18.1.30 `#define GSLC_DEV_TOUCH ""`

5.18.1.31 `#define GSLC_FEATURE_COMPOUND 0`

5.18.1.32 `#define GSLC_FEATURE_XGAUGE_RADIAL 0`

5.18.1.33 `#define GSLC_FEATURE_XGAUGE_RAMP 0`

5.18.1.34 `#define GSLC_LOCAL_STR 0`

5.18.1.35 `#define GSLC_LOCAL_STR_LEN 30`

5.18.1.36 `#define GSLC_ROTATE 1`

5.18.1.37 `#define GSLC_SD_BUFFPIXEL 50`

5.18.1.38 `#define GSLC_SD_EN 0`

5.18.1.39 `#define GSLC_TOUCH_MAX_EVT 1`

5.18.1.40 `#define GSLC_USE_FLOAT 0`

5.18.1.41 `#define GSLC_USE_PROGMEM 0`

5.19 src/GUIslice_config_esp.h File Reference

Macros

- `#define DRV_DISP_TFT_ESPI`
- `#define DRV_TOUCH_ADA_STMPE610`

- `#define GSLC_FEATURE_COMPOUND 1`
- `#define GSLC_FEATURE_XGAUGE_RADIAL 1`
- `#define GSLC_FEATURE_XGAUGE_RAMP 1`
- `#define DEBUG_ERR 1`
- `#define GSLC_DEV_TOUCH ""`
- `#define GSLC_LOCAL_STR 0`
- `#define GSLC_USE_FLOAT 0`
- `#define GSLC_SD_EN 0`
- `#define GSLC_CLIP_EN 1`
- `#define GSLC_ROTATE 1`
- `#define ADATOUCH_I2C_HW 0`
- `#define ADATOUCH_SPI_HW 1`
- `#define ADATOUCH_SPI_SW 0`
- `#define ADATOUCH_I2C_ADDR 0x41`
- `#define ADATOUCH_PIN_CS PIN_D0`
- `#define ADATOUCH_X_MIN 230`
- `#define ADATOUCH_Y_MIN 260`
- `#define ADATOUCH_X_MAX 3800`
- `#define ADATOUCH_Y_MAX 3700`
- `#define ADATOUCH_SWAP_XY 1`
- `#define ADATOUCH_FLIP_X 0`
- `#define ADATOUCH_FLIP_Y 1`
- `#define GSLC_TOUCH_MAX_EVT 1`
- `#define GSLC_LOCAL_STR_LEN 30`
- `#define GSLC_BMP_TRANS_EN 1`
- `#define GSLC_BMP_TRANS_RGB 0xFF,0x00,0xFF`
- `#define GSLC_USE_PROGMEM 0`

5.19.1 Macro Definition Documentation

5.19.1.1 `#define ADATOUCH_FLIP_X 0`

5.19.1.2 `#define ADATOUCH_FLIP_Y 1`

5.19.1.3 `#define ADATOUCH_I2C_ADDR 0x41`

5.19.1.4 `#define ADATOUCH_I2C_HW 0`

5.19.1.5 `#define ADATOUCH_PIN_CS PIN_D0`

5.19.1.6 `#define ADATOUCH_SPI_HW 1`

5.19.1.7 `#define ADATOUCH_SPI_SW 0`

5.19.1.8 `#define ADATOUCH_SWAP_XY 1`

5.19.1.9 `#define ADATOUCH_X_MAX 3800`

5.19.1.10 `#define ADATOUCH_X_MIN 230`

5.19.1.11 `#define ADATOUCH_Y_MAX 3700`

5.19.1.12 `#define ADATOUCH_Y_MIN 260`

5.19.1.13 `#define DEBUG_ERR 1`

5.19.1.14 `#define DRV_DISP_TFT_ESPI`

5.19.1.15 `#define DRV_TOUCH_ADA_STMPE610`

5.19.1.16 `#define GSLC_BMP_TRANS_EN 1`

5.19.1.17 `#define GSLC_BMP_TRANS_RGB 0xFF,0x00,0xFF`

5.19.1.18 `#define GSLC_CLIP_EN 1`

5.19.1.19 `#define GSLC_DEV_TOUCH ""`

5.19.1.20 `#define GSLC_FEATURE_COMPOUND 1`

5.19.1.21 `#define GSLC_FEATURE_XGAUGE_RADIAL 1`

5.19.1.22 `#define GSLC_FEATURE_XGAUGE_RAMP 1`

5.19.1.23 `#define GSLC_LOCAL_STR 0`

5.19.1.24 `#define GSLC_LOCAL_STR_LEN 30`

5.19.1.25 `#define GSLC_ROTATE 1`

5.19.1.26 `#define GSLC_SD_EN 0`

5.19.1.27 `#define GSLC_TOUCH_MAX_EVT 1`

5.19.1.28 `#define GSLC_USE_FLOAT 0`

5.19.1.29 `#define GSLC_USE_PROGMEM 0`

5.20 src/GUISlice_config_linux.h File Reference

Macros

- `#define DRV_DISP_SDL 1`
- `#define DRV_TOUCH_TSLIB`
- `#define GSLC_FEATURE_COMPOUND 1`
- `#define GSLC_FEATURE_XGAUGE_RADIAL 1`
- `#define GSLC_FEATURE_XGAUGE_RAMP 1`
- `#define DEBUG_ERR 1`
- `#define GSLC_DEV_FB "/dev/fb1"`
- `#define GSLC_DEV_TOUCH "/dev/input/touchscreen"`
- `#define GSLC_DEV_VID_DRV "fbcon"`
- `#define DRV_SDL_FIX_START 1`
- `#define DRV_SDL_MOUSE_SHOW 0`
- `#define GSLC_LOCAL_STR 1`
- `#define GSLC_USE_FLOAT 1`
- `#define ADATOUCH_SWAP_XY 1`
- `#define ADATOUCH_FLIP_X 0`
- `#define ADATOUCH_FLIP_Y 1`
- `#define GSLC_TOUCH_MAX_EVT 1`

- `#define GSLC_LOCAL_STR_LEN 30`
- `#define GSLC_BMP_TRANS_EN 1`
- `#define GSLC_BMP_TRANS_RGB 0xFF,0x00,0xFF`
- `#define GSLC_USE_PROGMEM 0`

5.20.1 Macro Definition Documentation

5.20.1.1 `#define ADATOUCH_FLIP_X 0`

5.20.1.2 `#define ADATOUCH_FLIP_Y 1`

5.20.1.3 `#define ADATOUCH_SWAP_XY 1`

5.20.1.4 `#define DEBUG_ERR 1`

5.20.1.5 `#define DRV_DISP_SDL 1`

5.20.1.6 `#define DRV_SDL_FIX_START 1`

5.20.1.7 `#define DRV_SDL_MOUSE_SHOW 0`

5.20.1.8 `#define DRV_TOUCH_TSLIB`

5.20.1.9 `#define GSLC_BMP_TRANS_EN 1`

5.20.1.10 `#define GSLC_BMP_TRANS_RGB 0xFF,0x00,0xFF`

5.20.1.11 `#define GSLC_DEV_FB "/dev/fb1"`

5.20.1.12 `#define GSLC_DEV_TOUCH "/dev/input/touchscreen"`

5.20.1.13 `#define GSLC_DEV_VID_DRV "fbcon"`

5.20.1.14 `#define GSLC_FEATURE_COMPOUND 1`

5.20.1.15 `#define GSLC_FEATURE_XGAUGE_RADIAL 1`

5.20.1.16 `#define GSLC_FEATURE_XGAUGE_RAMP 1`

5.20.1.17 `#define GSLC_LOCAL_STR 1`

5.20.1.18 `#define GSLC_LOCAL_STR_LEN 30`

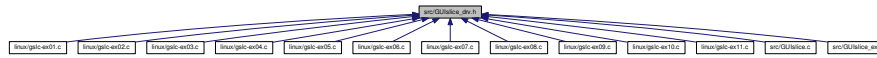
5.20.1.19 `#define GSLC_TOUCH_MAX_EVT 1`

5.20.1.20 `#define GSLC_USE_FLOAT 1`

5.20.1.21 `#define GSLC_USE_PROGMEM 0`

5.21 src/GUISlice_drv.h File Reference

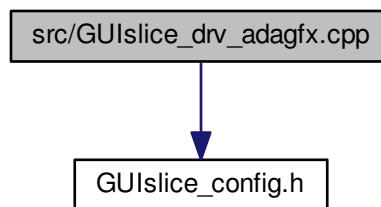
This graph shows which files directly or indirectly include this file:



5.22 src/GUISlice_drv_adagfx.cpp File Reference

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

Include dependency graph for GUISlice_drv_adagfx.cpp:

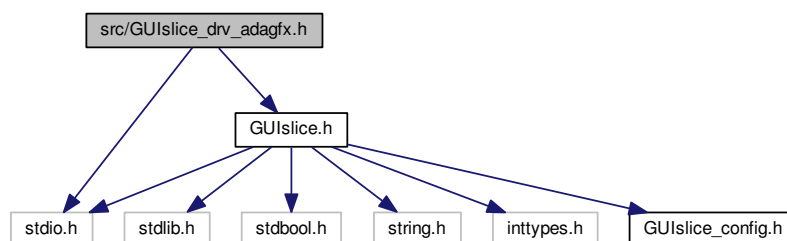


5.23 src/GUISlice_drv_adagfx.h File Reference

```
#include "GUISlice.h"
```

```
#include <stdio.h>
```

Include dependency graph for GUISlice_drv_adagfx.h:



Classes

- struct [gslc_tsDriver](#)

Macros

- `#define DRV_HAS_DRAW_POINT 1`
Support `gslc_DrvDrawPoint()`
- `#define DRV_HAS_DRAW_POINTS 0`
Support `gslc_DrvDrawPoints()`
- `#define DRV_HAS_DRAW_LINE 1`
Support `gslc_DrvDrawLine()`
- `#define DRV_HAS_DRAW_RECT_FRAME 1`
Support `gslc_DrvDrawFrameRect()`
- `#define DRV_HAS_DRAW_RECT_FILL 1`
Support `gslc_DrvDrawFillRect()`
- `#define DRV_HAS_DRAW_CIRCLE_FRAME 1`
Support `gslc_DrvDrawFrameCircle()`
- `#define DRV_HAS_DRAW_CIRCLE_FILL 1`
Support `gslc_DrvDrawFillCircle()`
- `#define DRV_HAS_DRAW_TRI_FRAME 1`
Support `gslc_DrvDrawFrameTriangle()`
- `#define DRV_HAS_DRAW_TRI_FILL 1`
Support `gslc_DrvDrawFillTriangle()`
- `#define DRV_HAS_DRAW_TEXT 1`
Support `gslc_DrvDrawTxt()`
- `#define DRV_OVERRIDE_TXT_ALIGN 0`
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.
- `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)
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_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)
Get the last touch event from the SDL_Event handler.
- bool [gslc_DrvRotateSwapFlip](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation, uint8_t nSwapXY, uint8_t nFlipX, uint8_t nFlipY)
Change rotation and axes swap/flip.
- uint16_t [gslc_DrvAdaptColorToRaw](#) ([gslc_tsColor](#) nCol)

5.23.1 Macro Definition Documentation

5.23.1.1 #define DRV_HAS_DRAW_CIRCLE_FILL 1

Support [gslc_DrvDrawFillCircle\(\)](#)

5.23.1.2 `#define DRV_HAS_DRAW_CIRCLE_FRAME 1`

Support [gslc_DrvDrawFrameCircle\(\)](#)

5.23.1.3 `#define DRV_HAS_DRAW_LINE 1`

Support [gslc_DrvDrawLine\(\)](#)

5.23.1.4 `#define DRV_HAS_DRAW_POINT 1`

Support [gslc_DrvDrawPoint\(\)](#)

5.23.1.5 `#define DRV_HAS_DRAW_POINTS 0`

Support [gslc_DrvDrawPoints\(\)](#)

5.23.1.6 `#define DRV_HAS_DRAW_RECT_FILL 1`

Support [gslc_DrvDrawFillRect\(\)](#)

5.23.1.7 `#define DRV_HAS_DRAW_RECT_FRAME 1`

Support [gslc_DrvDrawFrameRect\(\)](#)

5.23.1.8 `#define DRV_HAS_DRAW_TEXT 1`

Support [gslc_DrvDrawTxt\(\)](#)

5.23.1.9 `#define DRV_HAS_DRAW_TRI_FILL 1`

Support [gslc_DrvDrawFillTriangle\(\)](#)

5.23.1.10 `#define DRV_HAS_DRAW_TRI_FRAME 1`

Support [gslc_DrvDrawFrameTriangle\(\)](#)

5.23.1.11 `#define DRV_OVERRIDE_TXT_ALIGN 0`

Driver provides text alignment.

5.23.2 Function Documentation

5.23.2.1 `uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)`

5.23.2.2 `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

5.23.2.3 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

5.23.2.4 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

5.23.2.5 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

5.23.2.6 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

5.23.2.7 `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

5.23.2.8 `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

5.23.2.9 `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

5.23.2.10 `bool gslc_DrvDrawImage (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, gslc_tslmgRef 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

5.23.2.11 `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

5.23.2.12 `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

5.23.2.13 `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

5.23.2.14 `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

5.23.2.15 `bool gslc_DrvDrawTxt (gslc_tsGui * pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt)`

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

Returns

true if success, false if failure

5.23.2.16 `const void* gslc_DrvFontAdd (gslc_tFontRefType 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

5.23.2.17 `void gslc_DrvFontsDestruct (gslc_tsGui * pGui)`

Release all fonts defined in the GUI.

Parameters

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

Returns

none

5.23.2.18 `bool gslc_DrvGetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress)`

Get the last touch event from the SDL_Event handler.

Get the last touch event from the SDL 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)

Returns

true if an event was detected or false otherwise

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, >0 for touch)

Returns

true if an event was detected or 0 otherwise

5.23.2.19 `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

5.23.2.20 void gslc_DrvImageDestruct (void * *pvlmg*)

Release an image surface.

Parameters

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

Returns

none

5.23.2.21 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

5.23.2.22 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

5.23.2.23 `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

5.23.2.24 `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

5.23.2.25 `void gslc_DrvPageFlipNow (gslc_tsGui * pGui)`

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

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

Returns

none

5.23.2.26 `bool gslc_DrvRotateSwapFlip (gslc_tsGui * pGui, uint8_t nRotation, uint8_t nSwapXY, uint8_t nFlipX, uint8_t nFlipY)`

Change rotation and axes swap/flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)
in	<i>nSwapXY</i>	Touchscreen Swap X/Y axes
in	<i>nFlipX</i>	Touchscreen Flip X axis
in	<i>nFlipY</i>	Touchscreen Flip Y axis

Returns

true if successful

5.23.2.27 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

5.23.2.28 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

5.23.2.29 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

none

5.23.2.30 `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>slmgRef</i>	Image reference

Returns

true if success, false if error

5.23.2.31 `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

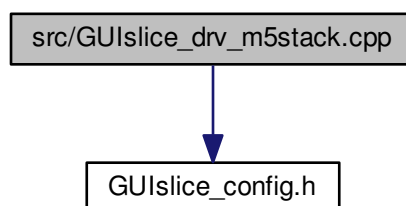
Returns

true if success, false if error

5.24 src/GUIslice_drv_m5stack.cpp File Reference

```
#include "GUIslice_config.h"
```

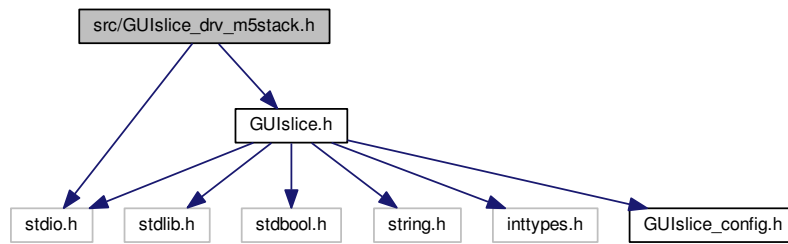
Include dependency graph for GUIslice_drv_m5stack.cpp:



5.25 src/GUIslice_drv_m5stack.h File Reference

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

Include dependency graph for `GUIslice_drv_m5stack.h`:



Classes

- struct `gslc_tsDriver`

Macros

- `#define DRV_HAS_DRAW_POINT 1`
Support `gslc_DrvDrawPoint()`
- `#define DRV_HAS_DRAW_POINTS 0`
Support `gslc_DrvDrawPoints()`
- `#define DRV_HAS_DRAW_LINE 1`
Support `gslc_DrvDrawLine()`
- `#define DRV_HAS_DRAW_RECT_FRAME 1`
Support `gslc_DrvDrawFrameRect()`
- `#define DRV_HAS_DRAW_RECT_FILL 1`
Support `gslc_DrvDrawFillRect()`
- `#define DRV_HAS_DRAW_CIRCLE_FRAME 1`
Support `gslc_DrvDrawFrameCircle()`
- `#define DRV_HAS_DRAW_CIRCLE_FILL 1`
Support `gslc_DrvDrawFillCircle()`
- `#define DRV_HAS_DRAW_TRI_FRAME 1`
Support `gslc_DrvDrawFrameTriangle()`
- `#define DRV_HAS_DRAW_TRI_FILL 1`
Support `gslc_DrvDrawFillTriangle()`
- `#define DRV_HAS_DRAW_TEXT 1`
Support `gslc_DrvDrawTxt()`
- `#define DRV_OVERRIDE_TXT_ALIGN 1`
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.*

 - 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)

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)

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_DrvDrawBkgnd](#) ([gslc_tsGui](#) *pGui)

Copy the background image to destination screen.

- bool [gslc_DrvRotateSwapFlip](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation, uint8_t nSwapXY, uint8_t nFlipX, uint8_t nFlipY)

Change rotation and axes swap/flip.

- uint16_t [gslc_DrvAdaptColorToRaw](#) ([gslc_tsColor](#) nCol)

5.25.1 Macro Definition Documentation

5.25.1.1 #define DRV_HAS_DRAW_CIRCLE_FILL 1

Support [gslc_DrvDrawFillCircle\(\)](#)

5.25.1.2 #define DRV_HAS_DRAW_CIRCLE_FRAME 1

Support [gslc_DrvDrawFrameCircle\(\)](#)

5.25.1.3 #define DRV_HAS_DRAW_LINE 1

Support [gslc_DrvDrawLine\(\)](#)

5.25.1.4 #define DRV_HAS_DRAW_POINT 1

Support [gslc_DrvDrawPoint\(\)](#)

5.25.1.5 #define DRV_HAS_DRAW_POINTS 0

Support [gslc_DrvDrawPoints\(\)](#)

5.25.1.6 #define DRV_HAS_DRAW_RECT_FILL 1

Support [gslc_DrvDrawFillRect\(\)](#)

5.25.1.7 #define DRV_HAS_DRAW_RECT_FRAME 1

Support [gslc_DrvDrawFrameRect\(\)](#)

5.25.1.8 #define DRV_HAS_DRAW_TEXT 1

Support [gslc_DrvDrawTxt\(\)](#)

5.25.1.9 #define DRV_HAS_DRAW_TRI_FILL 1

Support [gslc_DrvDrawFillTriangle\(\)](#)

5.25.1.10 `#define DRV_HAS_DRAW_TRI_FRAME 1`

Support [gslc_DrvDrawFrameTriangle\(\)](#)

5.25.1.11 `#define DRV_OVERRIDE_TXT_ALIGN 1`

Driver provides text alignment.

5.25.2 Function Documentation

5.25.2.1 `uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)`

5.25.2.2 `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

5.25.2.3 `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

5.25.2.4 `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

5.25.2.5 `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

5.25.2.6 `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

5.25.2.7 `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

5.25.2.8 `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

5.25.2.9 `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

5.25.2.10 `bool gslc_DrvDrawImage (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, gslc_tslmgRef 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

5.25.2.11 `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

5.25.2.12 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

5.25.2.13 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

5.25.2.14 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

5.25.2.15 bool gslc_DrvDrawTxt (gslc_tsGui * *pGui*, int16_t *nTxtX*, int16_t *nTxtY*, gslc_tsFont * *pFont*, const char * *pStr*, gslc_teTxtFlags *eTxtFlags*, gslc_tsColor *colTxt*)

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

Returns

true if success, false if failure

5.25.2.16 `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)`

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

Returns

true if success, false if failure

5.25.2.17 `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

5.25.2.18 `void gslc_DrvFontsDestruct (gslc_tsGui * pGui)`

Release all fonts defined in the GUI.

Parameters

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

Returns

none

5.25.2.19 `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

5.25.2.20 `void gslc_DrvImageDestruct (void * pVImg)`

Release an image surface.

Parameters

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

Returns

none

5.25.2.21 `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

5.25.2.22 `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

5.25.2.23 `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

5.25.2.24 `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

5.25.2.25 `bool gslc_DrvRotateSwapFlip (gslc_tsGui * pGui, uint8_t nRotation, uint8_t nSwapXY, uint8_t nFlipX, uint8_t nFlipY)`

Change rotation and axes swap/flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)
in	<i>nSwapXY</i>	Touchscreen Swap X/Y axes
in	<i>nFlipX</i>	Touchscreen Flip X axis
in	<i>nFlipY</i>	Touchscreen Flip Y axis

Returns

true if successful

5.25.2.26 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

5.25.2.27 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

5.25.2.28 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

none

5.25.2.29 `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

5.25.2.30 `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

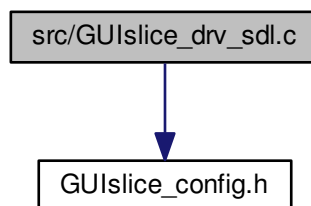
Returns

true if success, false if error

5.26 src/GUISlice_drv_sdl.c File Reference

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

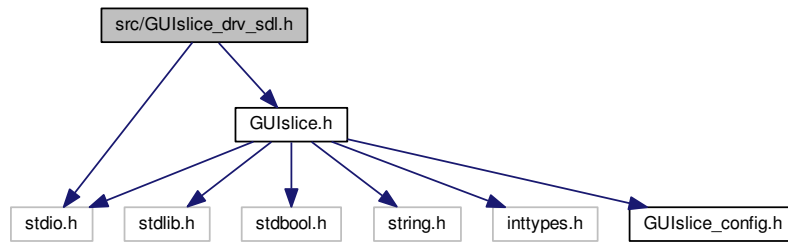
Include dependency graph for GUISlice_drv_sdl.c:



5.27 src/GUISlice_drv_sdl.h File Reference

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

Include dependency graph for `GUIslice_drv_sdl.h`:



Classes

- struct [gslc_tsDriver](#)

Macros

- `#define DRV_HAS_DRAW_POINT 1`
Support [gslc_DrvDrawPoint\(\)](#)
- `#define DRV_OVERRIDE_TXT_ALIGN 0`
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.
- 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)
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)
Get the last touch event from the SDL_Event handler.
- 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.

5.27.1 Macro Definition Documentation

5.27.1.1 #define DRV_HAS_DRAW_POINT 1

Support [gslc_DrvDrawPoint\(\)](#)

5.27.1.2 #define DRV_OVERRIDE_TXT_ALIGN 0

Driver provides text alignment.

5.27.2 Function Documentation

5.27.2.1 `SDL_Color` `gslc_DrvAdaptColor (gslc_tsColor sCol)`

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

Parameters

in	<i>sCol</i>	gslc_tsColor
----	-------------	------------------------------

Returns

Converted SDL_Color

5.27.2.2 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

5.27.2.3 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

5.27.2.4 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

5.27.2.5 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

5.27.2.6 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

5.27.2.7 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

5.27.2.8 bool gslc_DrvDrawImage (gslc_tsGui * *pGui*, int16_t *nDstX*, int16_t *nDstY*, gslc_tslmgRef *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

5.27.2.9 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

5.27.2.10 `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

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

Returns

true if success, false if error

5.27.2.12 `bool gslc_DrvDrawTxt (gslc_tsGui * pGui, int16_t nTxtX, int16_t nTxtY, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, gslc_tsColor colTxt)`

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

Returns

true if success, false if failure

5.27.2.13 `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

5.27.2.14 `void gslc_DrvFontsDestruct (gslc_tsGui * pGui)`

Release all fonts defined in the GUI.

Parameters

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

Returns

none

5.27.2.15 `bool gslc_DrvGetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress)`

Get the last touch event from the SDL_Event handler.

Get the last touch event from the SDL 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)

Returns

true if an event was detected or false otherwise

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, >0 for touch)

Returns

true if an event was detected or 0 otherwise

5.27.2.16 `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

5.27.2.17 void gslc_DrvImageDestruct (void * *pvlmg*)

Release an image surface.

Parameters

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

Returns

none

5.27.2.18 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

5.27.2.19 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

5.27.2.20 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

5.27.2.21 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

5.27.2.22 void gslc_DrvReportInfoPost ()

Report driver debug info (after initialization)

Returns

none

5.27.2.23 void gslc_DrvReportInfoPre ()

Report driver debug info (before initialization)

Returns

none

5.27.2.24 `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

5.27.2.25 `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

5.27.2.26 `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

5.27.2.27 `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

5.27.2.28 `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>slmgRef</i>	Image reference

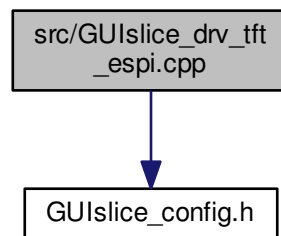
Returns

true if success, false if error

5.28 src/GUIslice_drv_tft_espi.cpp File Reference

```
#include "GUIslice_config.h"
```

Include dependency graph for GUIslice_drv_tft_espi.cpp:

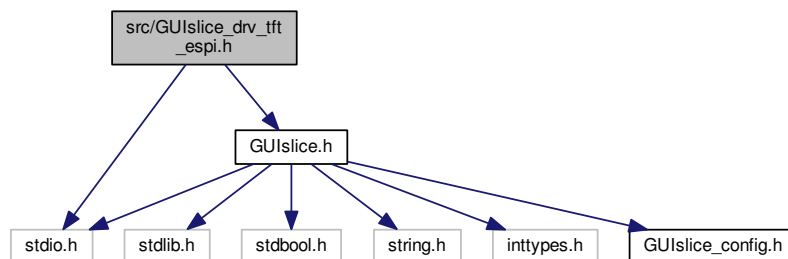


5.29 src/GUIslice_drv_tft_espi.h File Reference

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

```
#include <stdio.h>
```

Include dependency graph for GUIslice_drv_tft_espi.h:



Classes

- struct [gslc_tsDriver](#)

Macros

- #define `DRV_HAS_DRAW_POINT` 1
Support `gslc_DrvDrawPoint()`
- #define `DRV_HAS_DRAW_POINTS` 0
Support `gslc_DrvDrawPoints()`
- #define `DRV_HAS_DRAW_LINE` 1
Support `gslc_DrvDrawLine()`
- #define `DRV_HAS_DRAW_RECT_FRAME` 1
Support `gslc_DrvDrawFrameRect()`
- #define `DRV_HAS_DRAW_RECT_FILL` 1
Support `gslc_DrvDrawFillRect()`
- #define `DRV_HAS_DRAW_CIRCLE_FRAME` 1
Support `gslc_DrvDrawFrameCircle()`
- #define `DRV_HAS_DRAW_CIRCLE_FILL` 1
Support `gslc_DrvDrawFillCircle()`
- #define `DRV_HAS_DRAW_TRI_FRAME` 1
Support `gslc_DrvDrawFrameTriangle()`
- #define `DRV_HAS_DRAW_TRI_FILL` 1
Support `gslc_DrvDrawFillTriangle()`
- #define `DRV_HAS_DRAW_TEXT` 1
Support `gslc_DrvDrawTxt()`
- #define `DRV_OVERRIDE_TXT_ALIGN` 1
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.
- 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)
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)
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_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)
Get the last touch event from the SDL_Event handler.
- bool [gslc_DrvRotateSwapFlip](#) ([gslc_tsGui](#) *pGui, uint8_t nRotation, uint8_t nSwapXY, uint8_t nFlipX, uint8_t nFlipY)
Change rotation and axes swap/flip.
- uint16_t [gslc_DrvAdaptColorToRaw](#) ([gslc_tsColor](#) nCol)

5.29.1 Macro Definition Documentation

5.29.1.1 `#define DRV_HAS_DRAW_CIRCLE_FILL 1`

Support [gslc_DrvDrawFillCircle\(\)](#)

5.29.1.2 `#define DRV_HAS_DRAW_CIRCLE_FRAME 1`

Support [gslc_DrvDrawFrameCircle\(\)](#)

5.29.1.3 `#define DRV_HAS_DRAW_LINE 1`

Support [gslc_DrvDrawLine\(\)](#)

5.29.1.4 `#define DRV_HAS_DRAW_POINT 1`

Support [gslc_DrvDrawPoint\(\)](#)

5.29.1.5 `#define DRV_HAS_DRAW_POINTS 0`

Support [gslc_DrvDrawPoints\(\)](#)

5.29.1.6 `#define DRV_HAS_DRAW_RECT_FILL 1`

Support [gslc_DrvDrawFillRect\(\)](#)

5.29.1.7 `#define DRV_HAS_DRAW_RECT_FRAME 1`

Support [gslc_DrvDrawFrameRect\(\)](#)

5.29.1.8 `#define DRV_HAS_DRAW_TEXT 1`

Support [gslc_DrvDrawTxt\(\)](#)

5.29.1.9 `#define DRV_HAS_DRAW_TRI_FILL 1`

Support [gslc_DrvDrawFillTriangle\(\)](#)

5.29.1.10 `#define DRV_HAS_DRAW_TRI_FRAME 1`

Support [gslc_DrvDrawFrameTriangle\(\)](#)

5.29.1.11 `#define DRV_OVERRIDE_TXT_ALIGN 1`

Driver provides text alignment.

5.29.2 Function Documentation

5.29.2.1 `uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)`

5.29.2.2 `void gslc_DrvDestruct (gslc_tsGui * pGui)`

Free up any members associated with the driver.

- Eg. renderers, windows, background surfaces, etc.

Parameters

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

Returns

none

5.29.2.3 `void gslc_DrvDrawBkgnd (gslc_tsGui * pGui)`

Copy the background image to destination screen.

Parameters

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

Returns

true if success, false if fail

5.29.2.4 `bool gslc_DrvDrawFillCircle (gslc_tsGui * pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)`

Draw a filled circle.

Parameters

<code>in</code>	<code>pGui</code>	Pointer to GUI
<code>in</code>	<code>nMidX</code>	Center of circle (X coordinate)
<code>in</code>	<code>nMidY</code>	Center of circle (Y coordinate)
<code>in</code>	<code>nRadius</code>	Radius of circle
<code>in</code>	<code>nCol</code>	Color RGB value to fill

Returns

true if success, false if error

5.29.2.5 `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

5.29.2.6 `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

5.29.2.7 `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

5.29.2.8 `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

5.29.2.9 `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

5.29.2.10 `bool gslc_DrvDrawImage (gslc_tsGui * pGui, int16_t nDstX, int16_t nDstY, gslc_tslmgRef 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

5.29.2.11 `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

5.29.2.12 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

5.29.2.13 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

5.29.2.14 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

5.29.2.15 bool gslc_DrvDrawTxt (gslc_tsGui * *pGui*, int16_t *nTxtX*, int16_t *nTxtY*, gslc_tsFont * *pFont*, const char * *pStr*, gslc_teTxtFlags *eTxtFlags*, gslc_tsColor *colTxt*)

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

Returns

true if success, false if failure

5.29.2.16 `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)`

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

Returns

true if success, false if failure

5.29.2.17 `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

5.29.2.18 `void gslc_DrvFontsDestruct (gslc_tsGui * pGui)`

Release all fonts defined in the GUI.

Parameters

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

Returns

none

5.29.2.19 `bool gslc_DrvGetTouch (gslc_tsGui * pGui, int16_t * pnX, int16_t * pnY, uint16_t * pnPress)`

Get the last touch event from the SDL_Event handler.

Parameters

in	<i>pGui</i>	Pointer to GUI
out	<i>pnX</i>	Ptr to X coordinate of last touch event
out	<i>pnY</i>	Ptr to Y coordinate of last touch event
out	<i>pnPress</i>	Ptr to Pressure level of last touch event (0 for none, 1 for touch)

Returns

true if an event was detected or false otherwise

5.29.2.20 `bool gslc_DrvGetTxtSize (gslc_tsGui * pGui, gslc_tsFont * pFont, const char * pStr, gslc_teTxtFlags eTxtFlags, int16_t * pnTxtX, int16_t * pnTxtY, uint16_t * pnTxtSzW, uint16_t * pnTxtSzH)`

Get the extent (width and height) of a text string.

Parameters

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

5.29.2.21 `void gslc_DrvImageDestruct (void * pVImg)`

Release an image surface.

Parameters

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

Returns

none

5.29.2.22 `bool gslc_DrvInit (gslc_tsGui * pGui)`

Initialize the SDL library.

- Performs clean startup workaround (if enabled)
- Configures video mode
- Initializes font support

PRE:

- The environment variables should be configured before calling `gslc_DrvInit()`. 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

5.29.2.23 `bool gslc_DrvInitTouch (gslc_tsGui * pGui, const char * acDev)`

Perform any touchscreen-specific initialization.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>acDev</i>	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

5.29.2.24 `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

5.29.2.25 `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

5.29.2.26 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

5.29.2.27 bool gslc_DrvRotateSwapFlip (gslc_tsGui * *pGui*, uint8_t *nRotation*, uint8_t *nSwapXY*, uint8_t *nFlipX*, uint8_t *nFlipY*)

Change rotation and axes swap/flip.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nRotation</i>	Screen Rotation value (0, 1, 2 or 3)
in	<i>nSwapXY</i>	Touchscreen Swap X/Y axes
in	<i>nFlipX</i>	Touchscreen Flip X axis
in	<i>nFlipY</i>	Touchscreen Flip Y axis

Returns

true if successful

5.29.2.28 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

5.29.2.29 `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

5.29.2.30 `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

none

5.29.2.31 `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

5.29.2.32 `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

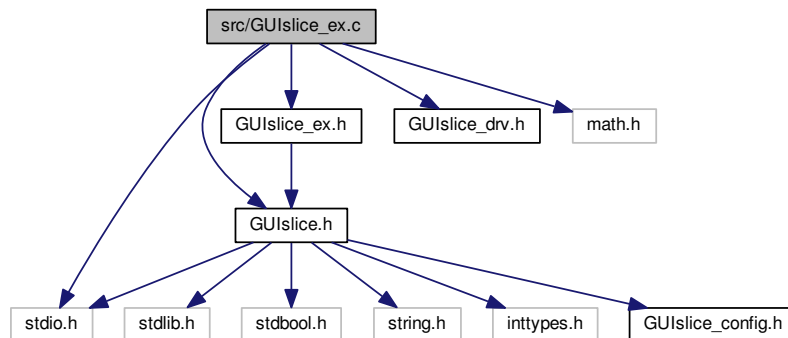
Returns

true if success, false if error

5.30 src/GUISlice_ex.c File Reference

```
#include "GUISlice.h"
#include "GUISlice_ex.h"
#include "GUISlice_drv.h"
#include <stdio.h>
#include <math.h>
```

Include dependency graph for GUISlice_ex.c:



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 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_teRedraw↔Type eRedraw)`
Helper function to draw a gauge with style: progress bar.
- `gslc_tsElemRef * gslc_ElemXCheckboxCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsX↔Checkbox *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_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.

- `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.

- `gslc_tsElemRef * gslc_ElemXTextboxCreate (gslc_tsGui *pGui, int16_t nElemId, int16_t nPage, gslc_tsX↔Textbox *pXData, gslc_tsRect rElem, int16_t nFontId, char *pBuf, uint16_t nBufRows, uint16_t nBufCols)`

Create a Textbox Element.

- void `gslc_ElemXTextboxReset (gslc_tsGui *pGui, gslc_tsElemRef *pElemRef)`

Reset the contents of the textbox.

- void `gslc_ElemXTextboxLineWrAdv (gslc_tsGui *pGui, gslc_tsXTextbox *pBox)`
- 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_tsXTextbox *pBox, unsigned char chNew, bool b↔Advance)`
- 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.

- `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 n↵
YMax)`

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 []`

5.30.1 Function Documentation

5.30.1.1 `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

5.30.1.2 `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

5.30.1.3 `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>nGroupId</i>	Group ID to search

Returns

Element Ptr or NULL if none checked

5.30.1.4 `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

5.30.1.5 `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

5.30.1.6 `void gslc_ElemXCheckboxSetStateHelp (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, bool bChecked)`

5.30.1.7 `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

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

Returns

true if success, false otherwise

5.30.1.9 `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 <i>nVal</i> comparison
in	<i>nMax</i>	Maximum value of gauge for <i>nVal</i> comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator
in	<i>bVert</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element reference or NULL if failure

5.30.1.10 `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

5.30.1.11 `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

5.30.1.12 `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

5.30.1.13 `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

5.30.1.14 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

5.30.1.15 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

5.30.1.16 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

5.30.1.17 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

5.30.1.18 `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

5.30.1.19 bool `gslc_ElemXGraphDraw (void * pvGui, void * pvElemRef, gslc_teRedrawType eRedraw)`

Draw a Graph element on the screen.

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

Parameters

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

Returns

true if success, false otherwise

5.30.1.20 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

5.30.1.21 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

5.30.1.22 void `gslc_ElemXGraphSetStyle (gslc_tsGui * pGui, gslc_tsElemRef * pElemRef, gslc_tXGraphStyle eStyle, uint8_t nMargin)`

Set the graph's additional drawing characteristics.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>pElemRef</i>	Pointer to Element reference
in	<i>eStyle</i>	Drawing style for the graph
in	<i>nMargin</i>	Margin to provide around graph area inside frame

Returns

none

5.30.1.23 **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

5.30.1.24 `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

5.30.1.25 `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

5.30.1.26 `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

5.30.1.27 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

5.30.1.28 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

5.30.1.29 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

5.30.1.30 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

5.30.1.31 void `gslc_ElemXTextboxBufAdd (gslc_tsGui * pGui, gslc_tsXTextbox * pBox, unsigned char chNew, bool bAdvance)`

5.30.1.32 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

5.30.1.33 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

5.30.1.34 `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.

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 text area
in	<i>pBuf</i>	Ptr to text buffer (already allocated) with size (nBufRows*nBufCols) chars
in	<i>nBufRows</i>	Number of rows in buffer
in	<i>nBufCols</i>	Number of columns in buffer (incl special codes)

Returns

Pointer to Element reference or NULL if failure

5.30.1.35 `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

5.30.1.36 `void gslc_ElemXTextboxLineWrAdv (gslc_tsGui * pGui, gslc_tsXTextbox * pBox)`

5.30.1.37 `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

5.30.1.38 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

5.30.1.39 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

5.30.2 Variable Documentation

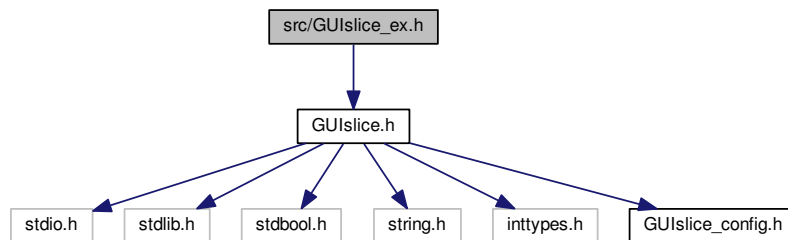
5.30.2.1 `const char GSLC_PMEM ERRSTR_NULL[]`

5.30.2.2 `const char GSLC_PMEM ERRSTR_PXD_NULL[]`

5.31 src/GUISlice_ex.h File Reference

```
#include "GUISlice.h"
```

Include dependency graph for GUISlice_ex.h:



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



Classes

- struct [gslc_tsXGauge](#)
Extended data for Gauge element.
- struct [gslc_tsXCheckbox](#)
Extended data for Checkbox element.
- struct [gslc_tsXSlider](#)
Extended data for Slider element.
- struct [gslc_tsXTextbox](#)
Extended data for Textbox element.
- struct [gslc_tsXGraph](#)
Extended data for Graph element.

Macros

- `#define` [GSLC_XTEXTBOX_CODE_COL_SET](#) 187
Definitions for textbox special inline codes.

- #define `GSLC_XTEXTBOX_CODE_COL_RESET` 188
- #define `gslc_ElemXCheckboxCreate_P`(pGui, nElemId, nPage, nX, nY, nW, nH, nGroup, bRadio_, nStyle_, colCheck_, bChecked_)
Create a Checkbox or Radio button Element in Flash.
- #define `gslc_ElemXSliderCreate_P`(pGui, nElemId, nPage, nX, nY, nW, nH, nPosMin_, nPosMax_, nPos_, nThumbSz_, bVert_, colFrame_, colFill_)
Create a Slider Element in Flash.
- #define `gslc_ElemXGaugeCreate_P`(pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, colFrame_, colFill_, colGauge_, bVert_)
Create a Gauge Element in Flash.

Typedefs

- typedef bool(* `GSLC_CB_XSLIDER_POS`)(void *pvGui, void *pvElem, int16_t nPos)
Callback function for slider feedback.

Enumerations

- enum `gslc_teTypeExtend` {
 `GSLC_TYPEX_GAUGE` = `GSLC_TYPE_BASE_EXTEND`, `GSLC_TYPEX_CHECKBOX`, `GSLC_TYPEX_SLIDER`, `GSLC_TYPEX_TEXTBOX`,
 `GSLC_TYPEX_GRAPH` }
Extended Element types.
- enum `gslc_teXGaugeStyle` { `GSLCX_GAUGE_STYLE_PROG_BAR`, `GSLCX_GAUGE_STYLE_RADIAL`, `GSLCX_GAUGE_STYLE_RAMP` }
Gauge drawing style.
- enum `gslc_teXCheckboxStyle` { `GSLCX_CHECKBOX_STYLE_BOX`, `GSLCX_CHECKBOX_STYLE_X`, `GSLCX_CHECKBOX_STYLE_ROUND` }
Checkbox drawing style.
- enum `gslc_teXGraphStyle` { `GSLCX_GRAPH_STYLE_DOT`, `GSLCX_GRAPH_STYLE_LINE`, `GSLCX_GRAPH_STYLE_FILL` }
Graph drawing style.

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_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.

- [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.

- 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.

- [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.

- [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.

- [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.

- void [gslc_ElemXGraphSetStyle](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, [gslc_teXGraphStyle](#) eStyle, uint8_t nMargin)

Set the graph's additional drawing characteristics.

- void [gslc_ElemXGraphSetRange](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nYMin, int16_t nYMax)

Set the graph's drawing range.

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

Draw a Graph element on the screen.

- void [gslc_ElemXGraphAdd](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, int16_t nVal)

Add a value to the graph at the latest position.

- void [gslc_ElemXGraphScrollSet](#) ([gslc_tsGui](#) *pGui, [gslc_tsElemRef](#) *pElemRef, uint8_t nScrollPos, uint8_t nScrollMax)

Set the graph scroll position (nScrollPos) as a fraction of nScrollMax.

5.31.1 Macro Definition Documentation

5.31.1.1 `#define gslc_ElemXCheckboxCreate_P(pGui, nElemId, nPage, nX, nY, nW, nH, 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>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>colCheck_</i>	Color for inner fill when checked
in	<i>bChecked_</i>	Default state

Returns

none

5.31.1.2 `#define gslc_ElemXGaugeCreate_P(pGui, nElemId, nPage, nX, nY, nW, nH, nMin_, nMax_, nVal_, colFrame_, colFill_, colGauge_, bVert_)`

Create a Gauge Element in Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>nMin_</i>	Minimum value of gauge for nVal comparison
in	<i>nMax_</i>	Maximum value of gauge for nVal comparison
in	<i>nVal_</i>	Starting value of gauge
in	<i>colFrame_</i>	Color for the gauge frame
in	<i>colFill_</i>	Color for the gauge background fill
in	<i>colGauge_</i>	Color for the gauge indicator
in	<i>bVert_</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

none

```
5.31.1.3 #define gslc_ElemXSliderCreate_P( pGui, nElemId, nPage, nX, nY, nW, nH, nPosMin_, nPosMax_, nPos_,
      nThumbSz_, bVert_, colFrame_, colFill_ )
```

Create a Slider Element in Flash.

Parameters

in	<i>pGui</i>	Pointer to GUI
in	<i>nElemId</i>	Unique element ID to assign
in	<i>nPage</i>	Page ID to attach element to
in	<i>nX</i>	X coordinate of element
in	<i>nY</i>	Y coordinate of element
in	<i>nW</i>	Width of element
in	<i>nH</i>	Height of element
in	<i>nPosMin_</i>	Minimum position value
in	<i>nPosMax_</i>	Maximum position value
in	<i>nPos_</i>	Starting position value
in	<i>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

```
5.31.1.4 #define GSLC_XTEXTBOX_CODE_COL_RESET 188
```

```
5.31.1.5 #define GSLC_XTEXTBOX_CODE_COL_SET 187
```

Definitions for textbox special inline codes.

5.31.2 Typedef Documentation

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

Callback function for slider feedback.

5.31.3 Enumeration Type Documentation

5.31.3.1 enum gslc_teTypeExtend

Extended Element types.

Enumerator

- GSLC_TYPEX_GAUGE** Gauge extended element.
- GSLC_TYPEX_CHECKBOX** Checkbox extended element.
- GSLC_TYPEX_SLIDER** Slider extended element.
- GSLC_TYPEX_TEXTBOX** Textbox extended element.
- GSLC_TYPEX_GRAPH** Graph extended element.

5.31.3.2 enum gslc_teXCheckboxStyle

Checkbox drawing style.

Enumerator

- GSLCX_CHECKBOX_STYLE_BOX** Inner box.
- GSLCX_CHECKBOX_STYLE_X** Crossed.
- GSLCX_CHECKBOX_STYLE_ROUND** Circular.

5.31.3.3 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.

5.31.3.4 enum gslc_teXGraphStyle

Graph drawing style.

Enumerator

- GSLCX_GRAPH_STYLE_DOT** Dot.
- GSLCX_GRAPH_STYLE_LINE** Line.
- GSLCX_GRAPH_STYLE_FILL** Filled.

5.31.4 Function Documentation

5.31.4.1 `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

5.31.4.2 `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 <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

5.31.4.3 `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>nGroupId</i>	Group ID to search

Returns

Element Ptr or NULL if none checked

5.31.4.4 `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

5.31.4.5 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

5.31.4.6 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

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

Returns

true if success, false otherwise

5.31.4.8 **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 GSLC_ID_AUTO to autogen)
in	<i>nPage</i>	Page ID to attach element to
in	<i>pXData</i>	Ptr to extended element data structure
in	<i>rElem</i>	Rectangle coordinates defining gauge size
in	<i>nMin</i>	Minimum value of gauge for <i>nVal</i> comparison
in	<i>nMax</i>	Maximum value of gauge for <i>nVal</i> comparison
in	<i>nVal</i>	Starting value of gauge
in	<i>colGauge</i>	Color for the gauge indicator
in	<i>bVert</i>	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element reference or NULL if failure

5.31.4.9 **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 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

5.31.4.10 **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

5.31.4.11 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

5.31.4.12 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.

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

5.31.4.13 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

5.31.4.14 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

5.31.4.15 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

5.31.4.16 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

<i>in</i>	<i>nVal</i>	Data value to add
-----------	-------------	-------------------

Returns

none

5.31.4.17 **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

<i>in</i>	<i>pGui</i>	Pointer to GUI
<i>in</i>	<i>nElemId</i>	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
<i>in</i>	<i>nPage</i>	Page ID to attach element to
<i>in</i>	<i>pXData</i>	Ptr to extended element data structure
<i>in</i>	<i>rElem</i>	Rectangle coordinates defining checkbox size
<i>in</i>	<i>nFontId</i>	Font ID to use for graph area
<i>in</i>	<i>pBuf</i>	Ptr to data buffer (already allocated) with size (nBufMax) int16_t
<i>in</i>	<i>nBufRows</i>	Maximum number of points in buffer
<i>in</i>	<i>colGraph</i>	Color of the graph

Returns

Pointer to Element reference or NULL if failure

5.31.4.18 **bool** **gslc_ElemXGraphDraw** (**void** * *pvGui*, **void** * *pvElemRef*, **gslc_teRedrawType** *eRedraw*)

Draw a Graph element on the screen.

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

Parameters

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

Returns

true if success, false otherwise

5.31.4.19 **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

5.31.4.20 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

5.31.4.21 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

5.31.4.22 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

5.31.4.23 `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>pvElemRef</i>	Void ptr to Element (typecast to <code>gslc_tsElemRef*</code>)
in	<i>eRedraw</i>	Redraw mode

Returns

true if success, false otherwise

5.31.4.24 `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

5.31.4.25 `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

5.31.4.26 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

5.31.4.27 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

5.31.4.28 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

5.31.4.29 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

5.31.4.30 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

5.31.4.31 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

5.31.4.32 `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.

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 text area
in	<i>pBuf</i>	Ptr to text buffer (already allocated) with size (nBufRows*nBufCols) chars
in	<i>nBufRows</i>	Number of rows in buffer
in	<i>nBufCols</i>	Number of columns in buffer (incl special codes)

Returns

Pointer to Element reference or NULL if failure

5.31.4.33 `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

5.31.4.34 `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

5.31.4.35 `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

5.31.4.36 `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