GUIslice

0.8.8

Generated by Doxygen 1.8.8

Sun Apr 30 2017 11:11:38

Contents

Chapter 1

README

GUIslice library

A lightweight GUI framework suitable for embedded displays

- Website (www.impulseadventure.com)
- Documentation wiki (github)
- Release notes
- Pure C library, no dynamic memory allocation
- Widgets: text, images, buttons, checkboxes, radio buttons, sliders, radial controls, scrolling textboxes, etc. plus extensions and multiple pages.
- Platform-independent GUI core currently supports: SDL1.2, SDL2.0, Adafruit-GFX
- Typical target: Raspberry Pi, Arduino, Cortex M0 (Feather M0), LINUX, Beaglebone Black
- Typical displays: PiTFT, Waveshare, Adafruit TFT 2.2" / 2.8" / 1.44', OLED 0.96", 4D Cape
- Supports touchscreen control
- No GUIslice installation – just add include files and go!
- LINUX Dependencies: sdl, sdl-ttf, optional: tslib
- Arduino Dependencies: Adafruit-GFX plus display (eg. ILI9341) / touch driver library (eg. STMPE610)

Screenshots

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

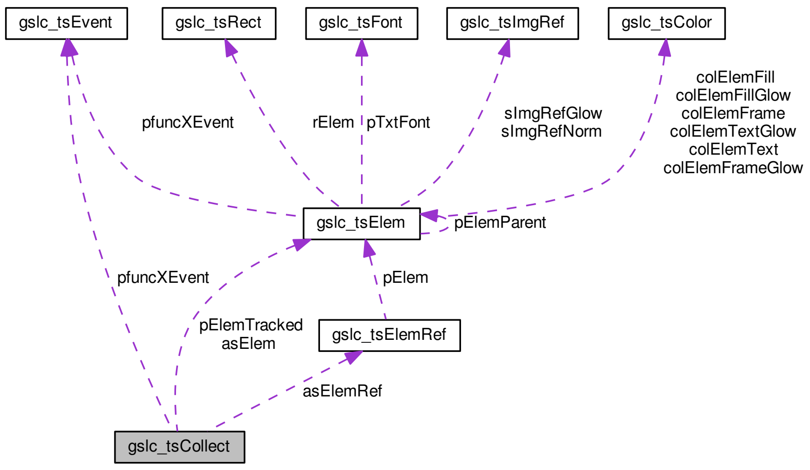
Class Documentation

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 nElemAutoIdNext

    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_tsElem ∗ pElemTracked

  Element 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::nElemAutoIdNext

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_tsElem∗ gslc_tsCollect::pElemTracked

Element 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.
- uint8_t unused

  Unused value to pad structure.

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.

4.2.2.4 uint8_t gslc_tsColor::unused

Unused value to pad structure.

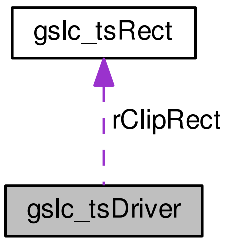
The documentation for this struct was generated from the following file:

- 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.
- SDL_Surface ∗ pSurfScreen

  Surface ptr for screen.
- struct tsdev ∗ pTsDev

  Ptr to touchscreen device.

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 SDL_Surface∗ gslc_tsDriver::pSurfScreen

Surface ptr for screen.

4.3.1.3 struct tsdev∗ gslc_tsDriver::pTsDev

Ptr to touchscreen device.

4.3.1.4 gslc_tsRect gslc_tsDriver::rClipRect

Clipping rectangle.

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

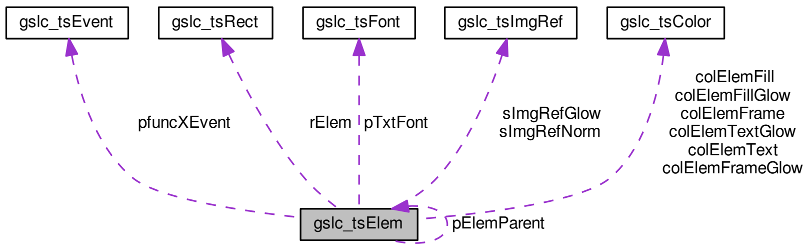
- src/GUIslice_drv_adagfx.h
- src/GUIslice_drv_sdl.h

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.
- bool bValid

    Element was created properly.
- int16_t nType

    Element type enumeration.
- gslc_tsRect rElem

    Rect region containing element.
- int16_t nGroup

    Group ID that the element belongs to.
- bool bGlowEn

    Enable glowing visual state.
- bool bClickEn

    Element accepts touch events.
- bool bFrameEn

    Element is drawn with frame.
- bool bFillEn

    Element is drawn with inner fill.
- 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_tsElem ∗ pElemParent

    Parent element reference.
- char pStrBuf [GSLC_LOCAL_STR_LEN]

    Text string 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.
- gslc_teRedrawType eRedraw

    Type of redraw requested for element.
- bool bGlowing

    Element is currently glowing.

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 bool gslc_tsElem::bClickEn

Element accepts touch events.

4.4.2.2 bool gslc_tsElem::bFillEn

Element is drawn with inner fill.

This is also used during redraw to determine if elements underneath are visible and must be redrawn as well.

4.4.2.3 bool gslc_tsElem::bFrameEn

Element is drawn with frame.

4.4.2.4 bool gslc_tsElem::bGlowEn

Enable glowing visual state.

4.4.2.5 bool gslc_tsElem::bGlowing

Element is currently glowing.

4.4.2.6 bool gslc_tsElem::bValid

Element was created properly.

4.4.2.7 gslc_tsColor gslc_tsElem::colElemFill

Color for background fill.

4.4.2.8 gslc_tsColor gslc_tsElem::colElemFillGlow

Color to use for fill when glowing.

4.4.2.9 gslc_tsColor gslc_tsElem::colElemFrame

Color for frame.

4.4.2.10 gslc_tsColor gslc_tsElem::colElemFrameGlow

Color to use for frame when glowing.

4.4.2.11 gslc_tsColor gslc_tsElem::colElemText

Color of overlay text.

4.4.2.12 gslc_tsColor gslc_tsElem::colElemTextGlow

Color of overlay text when glowing.

4.4.2.13 gslc_teRedrawType gslc_tsElem::eRedraw

Type of redraw requested for element.

4.4.2.14 int8_t gslc_tsElem::eTxtAlign

Alignment of overlay text.

4.4.2.15 gslc_teTxtFlags gslc_tsElem::eTxtFlags

Flags associated with text buffer.

4.4.2.16 int16_t gslc_tsElem::nGroup

Group ID that the element belongs to.

4.4.2.17 int16_t gslc_tsElem::nId

Element ID specified by user.

4.4.2.18 uint8_t gslc_tsElem::nStrBufMax

Size of string buffer.

4.4.2.19 uint8_t gslc_tsElem::nTxtMargin

Margin of overlay text within rect region.

4.4.2.20 int16_t gslc_tsElem::nType

Element type enumeration.

4.4.2.21 gslc_tsElem∗ gslc_tsElem::pElemParent

Parent element reference.

Used during redraw to notify parent elements that they require redraw as well. Primary usage is in compound elements.

4.4.2.22 GSLC_CB_DRAW gslc_tsElem::pfuncXDraw

Callback func ptr for custom drawing.

4.4.2.23 GSLC_CB_EVENT gslc_tsElem::pfuncXEvent

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

4.4.2.24 GSLC_CB_TICK gslc_tsElem::pfuncXTick

Callback func ptr for timer/main loop tick.

4.4.2.25 GSLC_CB_TOUCH gslc_tsElem::pfuncXTouch

Callback func ptr for touch.

4.4.2.26 char gslc_tsElem::pStrBuf[GSLC_LOCAL_STR_LEN]

Text string to overlay.

4.4.2.27 gslc_tsFont∗ gslc_tsElem::pTxtFont

Ptr to Font for overlay text.

4.4.2.28 void∗ gslc_tsElem::pXData

Ptr to extended data structure.

4.4.2.29 gslc_tsRect gslc_tsElem::rElem

Rect region containing element.

4.4.2.30 gslc_tsImgRef gslc_tsElem::sImgRefGlow

Image reference to draw (glowing)

4.4.2.31 gslc_tsImgRef gslc_tsElem::sImgRefNorm

Image reference to draw (normal)

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

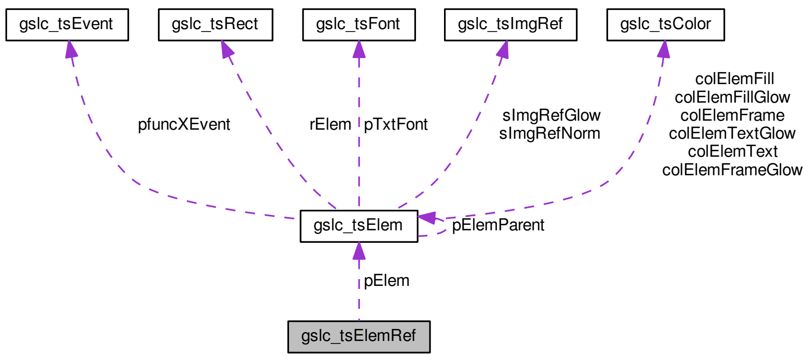
- src/GUIslice.h

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.
- void ∗ pvFont

  Void ptr to the Font (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 int16_t gslc_tsFont::nId

Font ID specified by user.

4.8.2.2 uint16_t gslc_tsFont::nSize

Font size.

4.8.2.3 void∗ gslc_tsFont::pvFont

Void ptr to the Font (type defined by driver)

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

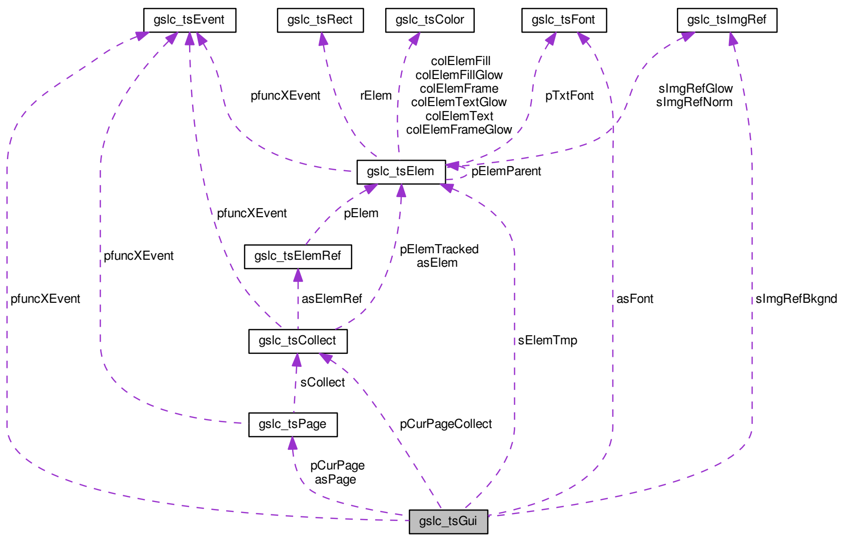
- src/GUIslice.h

4.9 gslc_tsGui Struct Reference

GUI structure.

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

Collaboration diagram for gslc_tsGui:



Public Attributes

- uint16_t nDispW

  Width of the display (pixels)
- uint16_t nDispH

  Height of the display (pixels)
- uint8_t nDispDepth

  Bit depth of display (bits per pixel)
- gslc_tsFont ∗ asFont

  Collection of loaded fonts.
- uint8_t nFontMax

  Maximum number of fonts to allocate.
- uint8_t nFontCnt

  Number of fonts allocated.
- gslc_tsElem sElemTmp

  Temporary element.
- 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 sImgRefBkgnd

  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::sElemTmp

Temporary element.

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_teImgRefFlags eImgFlags

  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_teImgRefFlags gslc_tsImgRef::eImgFlags

Image reference flags.

4.10.2.2 const char∗ gslc_tsImgRef::pFname

Pathname to input image file [FILE,SD].

4.10.2.3 const unsigned char∗ gslc_tsImgRef::pImgBuf

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

4.10.2.4 void∗ gslc_tsImgRef::pvImgRaw

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

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

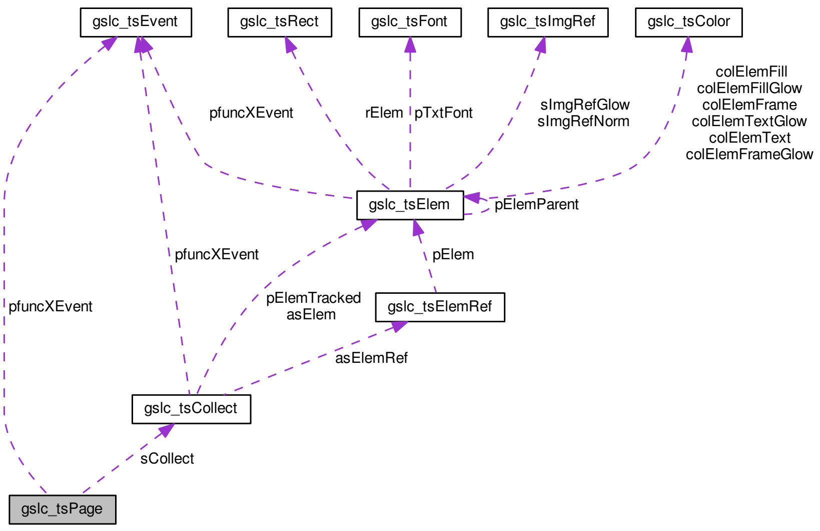
- src/GUIslice.h

4.11 gslc_tsPage Struct Reference

Page structure.

`#include <GUIslice.h>`

Collaboration diagram for gslc_tsPage:



Public Attributes

- 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.

4.11.1 Detailed Description

Page structure.

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

4.11.2 Member Data Documentation

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

- int x

  X coordinate.
- int y

  Y coordinate.

4.12.1 Detailed Description

Define point coordinates.

4.12.2 Member Data Documentation

4.12.2.1 int gslc_tsPt::x

X coordinate.

4.12.2.2 int 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:

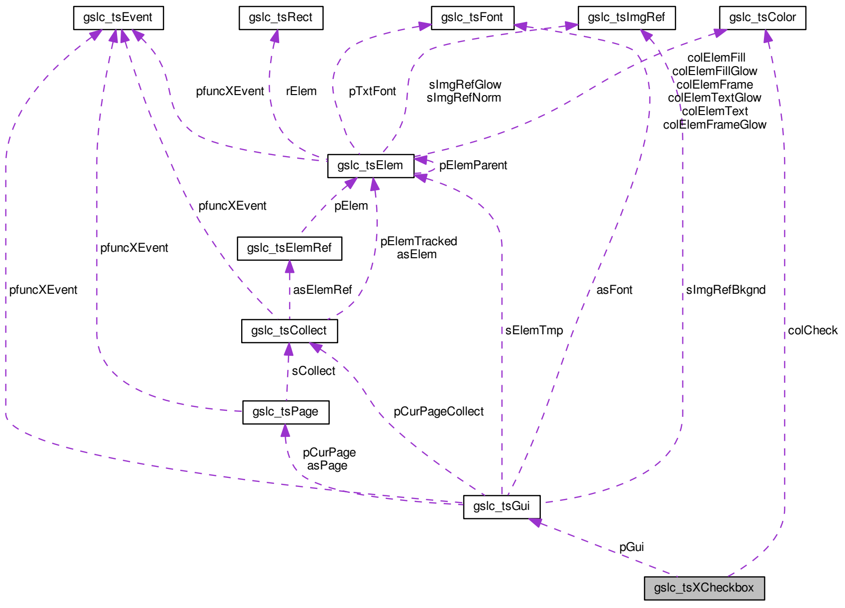
- src/GUIslice.h

4.14 gslc_tsXCheckbox Struct Reference

Extended data for Checkbox element.

`#include <GUIslice_ex.h>`

Collaboration diagram for gslc_tsXCheckbox:



Public Attributes

- gslc_tsGui ∗ pGui

    Ptr to GUI (for radio group control)
- 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.

4.14.2.5 gslc_tsGui* gslc_tsXCheckbox::pGui

Ptr to GUI (for radio group control)

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

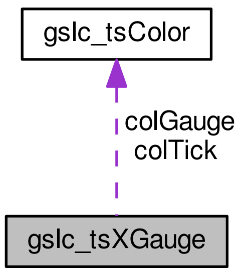
- src/GUIslice_ex.h

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_teXGaugeStyle 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_teXGaugeStyle 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:

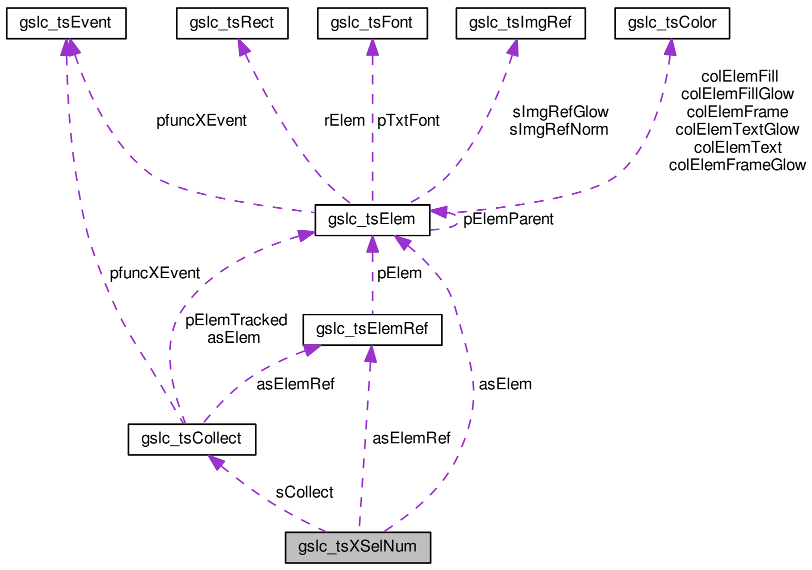
- src/GUIslice_ex.h

4.16 gslc_tsXSelNum Struct Reference

Extended data for SelNum element.

`#include <GUIslice_ex.h>`

Collaboration diagram for gslc_tsXSelNum:



Public Attributes

- int16_t nCounter

    Counter for demo purposes.
- gslc_tsCollect sCollect

    Collection management for sub-elements.
- gslc_tsElemRef asElemRef [4]

    Storage for sub-element references.
- gslc_tsElem asElem [4]

    Storage for sub-elements.

4.16.1 Detailed Description

Extended data for SelNum element.

4.16.2 Member Data Documentation

4.16.2.1 gslc_tsElem gslc_tsXSelNum::asElem[4]

Storage for sub-elements.

4.16.2.2 gslc_tsElemRef gslc_tsXSelNum::asElemRef[4]

Storage for sub-element references.

4.16.2.3 int16_t gslc_tsXSelNum::nCounter

Counter for demo purposes.

4.16.2.4 gslc_tsCollect gslc_tsXSelNum::sCollect

Collection management for sub-elements.

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

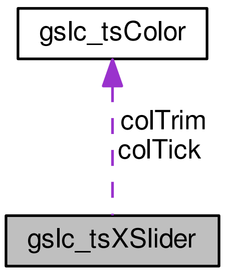
- src/GUIslice_ex.h

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:

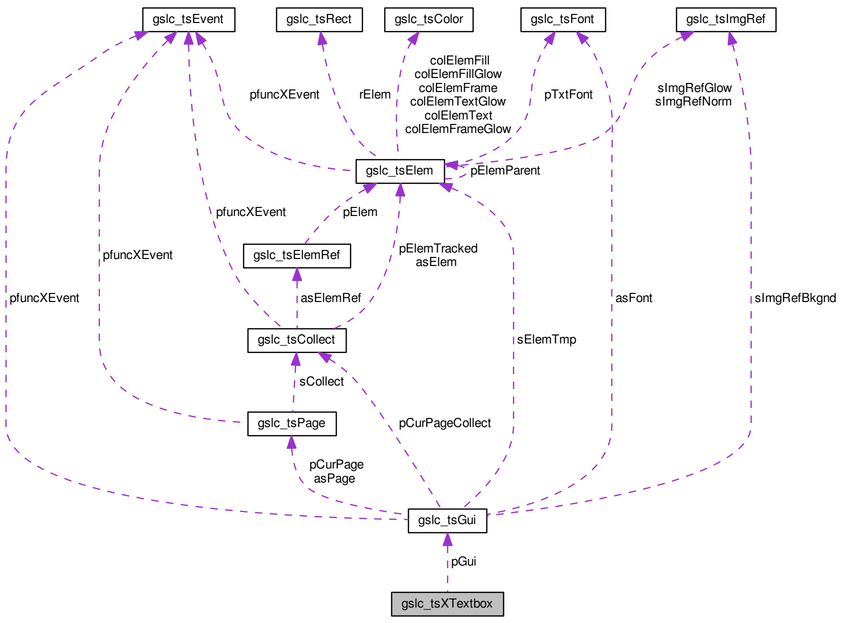
- src/GUIslice_ex.h

4.18 gslc_tsXTextbox Struct Reference

Extended data for Textbox element.

`#include <GUIslice_ex.h>`

Collaboration diagram for gslc_tsXTextbox:



Public Attributes

- gslc_tsGui ∗ pGui

  Ptr to GUI (for radio group control)
- 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))

4.18.2.17 gslc_tsGui* gslc_tsXTextbox::pGui

Ptr to GUI (for radio group control)

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

- src/GUIslice_ex.h

Chapter 5

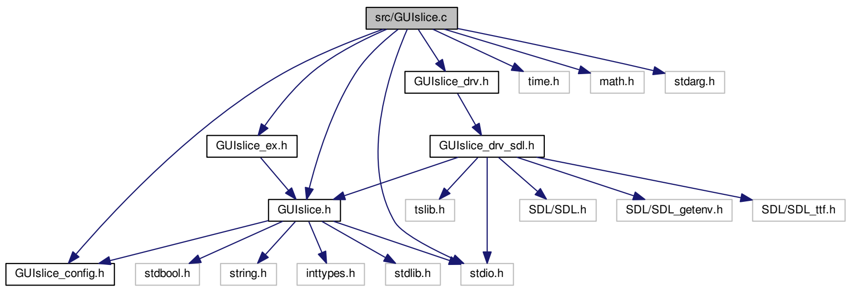
File Documentation

5.1 README.md File Reference

5.2 src/GUIslice.c File Reference

```
#include "GUIslice_config.h"
#include "GUIslice.h"
#include "GUIslice_ex.h"
#include "GUIslice_drv.h"
#include <stdio.h>
#include <time.h>
#include <math.h>
#include <stdarg.h>
```

Include dependency graph for GUIslice.c:



Macros

- #define GUISLICE_VER "0.8.8"

Enumerations

- enum gslc_teDebugPrintState { GSLC_DEBUG_PRINT_NORM, GSLC_DEBUG_PRINT_TOKEN, GSLC_DEBUG_PRINT_UINT16, GSLC_DEBUG_PRINT_STR }

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 ∗as↩ Font, 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_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 (gslc_tsGui ∗pGui, 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_tsImgRef gslc_ResetImage ()

  Create a blank image reference structure.
- gslc_tsImgRef gslc_GetImageFromFile (const char ∗pFname, gslc_teImgRefFlags eFmt)

  Create an image reference to a bitmap file in LINUX filesystem.
- gslc_tsImgRef gslc_GetImageFromSD (const char ∗pFname, gslc_teImgRefFlags eFmt)

  Create an image reference to a bitmap file in SD card.
- gslc_tsImgRef gslc_GetImageFromRam (unsigned char ∗pImgBuf, gslc_teImgRefFlags eFmt)

  Create an image reference to a bitmap in SRAM.
- gslc_tsImgRef gslc_GetImageFromProg (const unsigned char ∗pImgBuf, gslc_teImgRefFlags 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 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.
- 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 n←Col)

  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, const char ∗acFontName, 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_ts←ElemRef ∗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_tsElem ∗ 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_tsPage ∗pPage, GSLC_CB_EVENT funcCb)

  Assign the event callback function for a page.
- int gslc_ElemGetId (gslc_tsElem ∗pElem)

  Get an Element ID from an element structure.
- gslc_tsElem ∗ 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_tsElem ∗ 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_tsElem ∗ gslc_ElemCreateBtnImg (gslc_tsGui ∗pGui, int16_t nElemId, int16_t nPage, gslc_tsRect r↩
Elem, gslc_tsImgRef sImgRef, gslc_tsImgRef sImgRefSel, GSLC_CB_TOUCH cbTouch)

  Create a graphical Button Element.
- gslc_tsElem ∗ gslc_ElemCreateBox (gslc_tsGui ∗pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem)

  Create a Box Element.
- gslc_tsElem ∗ 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_tsElem ∗ 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_tsElem ∗pElem, gslc_teRedrawType eRedraw)

  Draw an element to the active display.
- void gslc_ElemSetFillEn (gslc_tsElem ∗pElem, bool bFillEn)

  Set the fill state for an Element.
- void gslc_ElemSetFrameEn (gslc_tsElem ∗pElem, bool bFrameEn)

  Set the frame state for an Element.
- void gslc_ElemSetCol (gslc_tsElem ∗pElem, gslc_tsColor colFrame, gslc_tsColor colFill, gslc_tsColor col↩
FillGlow)

  Update the common color selection for an Element.
- void gslc_ElemSetGlowCol (gslc_tsElem ∗pElem, gslc_tsColor colFrameGlow, gslc_tsColor colFillGlow, gslc_tsColor colTxtGlow)

  Update the common color selection for glowing state of an Element.
- void gslc_ElemSetGroup (gslc_tsElem ∗pElem, int nGroupId)

  Set the group ID for an element.

- int gslc_ElemGetGroup (gslc_tsElem ∗pElem)

  Get the group ID for an element.
- void gslc_ElemSetTxtAlign (gslc_tsElem ∗pElem, unsigned nAlign)

  Set the alignment of a textual element (horizontal and vertical)
- void gslc_ElemSetTxtMargin (gslc_tsElem ∗pElem, unsigned nMargin)

  Set the margin around of a textual element.
- void gslc_ElemSetTxtStr (gslc_tsElem ∗pElem, const char ∗pStr)

  Update the text string associated with an Element ID.
- void gslc_ElemSetTxtCol (gslc_tsElem ∗pElem, gslc_tsColor colVal)

  Update the text string color associated with an Element ID.
- void gslc_ElemSetTxtMem (gslc_tsElem ∗pElem, gslc_teTxtFlags eFlags)

  Update the text string location in memory.
- void gslc_ElemUpdateFont (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, int nFontId)

  Update the Font selected for an Element's text.
- void gslc_ElemSetRedraw (gslc_tsElem ∗pElem, gslc_teRedrawType eRedraw)

  Update the need-redraw status for an element.
- gslc_teRedrawType gslc_ElemGetRedraw (gslc_tsElem ∗pElem)

  Get the need-redraw status for an element.
- void gslc_ElemSetGlow (gslc_tsElem ∗pElem, bool bGlowing)

  Update the glowing indicator for an element.
- bool gslc_ElemGetGlow (gslc_tsElem ∗pElem)

  Get the glowing indicator for an element.
- void gslc_ElemSetGlowEn (gslc_tsElem ∗pElem, bool bGlowEn)

  Update the glowing enable for an element.
- bool gslc_ElemGetGlowEn (gslc_tsElem ∗pElem)

  Get the glowing enable for an element.
- void gslc_ElemSetStyleFrom (gslc_tsElem ∗pElemSrc, gslc_tsElem ∗pElemDest)

  Copy style settings from one element to another.
- void gslc_ElemSetEventFunc (gslc_tsElem ∗pElem, GSLC_CB_EVENT funcCb)

  Assign the event callback function for a element.
- void gslc_ElemSetDrawFunc (gslc_tsElem ∗pElem, GSLC_CB_DRAW funcCb)

  Assign the drawing callback function for an element.
- void gslc_ElemSetTickFunc (gslc_tsElem ∗pElem, GSLC_CB_TICK funcCb)

  Assign the tick callback function for an element.
- bool gslc_ElemOwnsCoord (gslc_tsElem ∗pElem, int16_t nX, int16_t nY, bool bOnlyClickEn)

  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_tsElem ∗ gslc_CollectElemAdd (gslc_tsCollect ∗pCollect, const gslc_tsElem ∗pElem, gslc_teElemRef←Flags eFlags)

  Add an element to a collection.
- bool gslc_CollectGetRedraw (gslc_tsCollect ∗pCollect)

  Determine if any elements in a collection need redraw.
- gslc_tsElem ∗ gslc_ElemAdd (gslc_tsGui ∗pGui, int16_t nPageId, gslc_tsElem ∗pElem, gslc_teElemRefFlags eFlags)

  Add the Element to the list of generated elements in the GUI environment.
- bool gslc_SetClipRect (gslc_tsGui ∗pGui, gslc_tsRect ∗pRect)

  Set the clipping rectangle for further drawing.
- void gslc_ElemSetImage (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, gslc_tsImgRef sImgRef, gslc_tsImgRef 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_ElemSendEventTouch (gslc_tsGui ∗pGui, gslc_tsElem ∗pElemTracked, gslc_teTouch eTouch, int16_t nX, int16_t nY)

  Trigger an element's touch event.
- void gslc_ResetElem (gslc_tsElem ∗pElem)

  Initialize an Element struct.
- void gslc_ResetFont (gslc_tsFont ∗pFont)

  Initialize a Font struct.
- void gslc_ElemDestruct (gslc_tsElem ∗pElem)

  Free up any members associated with an element.
- void gslc_CollectDestruct (gslc_tsCollect ∗pCollect)

  Free up any members associated with an element collection.
- void gslc_PageDestruct (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_tsElem ∗ gslc_CollectFindElemById (gslc_tsCollect ∗pCollect, int16_t nElemId)

  Find an element in a collection by its Element ID.
- int gslc_CollectGetNextId (gslc_tsCollect ∗pCollect)

  Allocate the next available Element ID in a collection.
- gslc_tsElem ∗ gslc_CollectGetElemTracked (gslc_tsCollect ∗pCollect)

  Get the element within a collection that is currently being tracked.
- void gslc_CollectSetElemTracked (gslc_tsCollect ∗pCollect, gslc_tsElem ∗pElem)

  Set the element within a collection that is currently being tracked.
- gslc_tsElem ∗ gslc_CollectFindElemFromCoord (gslc_tsCollect ∗pCollect, int16_t nX, int16_t nY)

  Find an element in a collection by a coordinate coordinate.
- void gslc_CollectSetParent (gslc_tsCollect ∗pCollect, gslc_tsElem ∗pElemParent)

  Assign the parent element reference to all elements within a collection.
- void gslc_CollectSetEventFunc (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_nLUTSinF0X16 [257]

5.2.1 Macro Definition Documentation

5.2.1.1 #define GUISLICE_VER "0.8.8"

5.2.2 Enumeration Type Documentation

5.2.2.1 enum gslc_teDebugPrintState

Enumerator

GSLC_DEBUG_PRINT_NORM

GSLC_DEBUG_PRINT_TOKEN

GSLC_DEBUG_PRINT_UINT16

GSLC_DEBUG_PRINT_STR

5.2.3 Function Documentation

5.2.3.1 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	*pClipRect*	Pointer to clipping region
in,out	*pnX0*	Ptr to X coordinate of line start
in,out	*pnY0*	Ptr to Y coordinate of line start
in,out	*pnX1*	Ptr to X coordinate of line end
in,out	*pnY1*	Ptr to Y coordinate of line end

Returns

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

5.2.3.2 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	*pClipRect*	Pointer to clipping region

`in`	*nX*	X coordinate of point
`in`	*nY*	Y coordinate of point

Returns

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

5.2.3.3 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`	*pClipRect*	Pointer to clipping region
`in,out`	*pRect*	Ptr to rectangle

Returns

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

5.2.3.4 void gslc_CollectDestruct (gslc_tsCollect ∗ *pCollect*)

Free up any members associated with an element collection.

Parameters

`in`	*pCollect*	Pointer to collection

Returns

none

5.2.3.5 gslc_tsElem∗ gslc_CollectElemAdd (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 are the call is complete.

Parameters

`in`	*pCollect*	Pointer to the collection
`in`	*pElem*	Ptr to the element to add
`in`	*eFlags*	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 in the collection that has been added or NULL if there was an error

5.2.3.6 bool gslc_CollectEvent (void ∗ *pvGui,* gslc_tsEvent *sEvent*)

Common event handler function for an element collection.

Parameters

in	*pvGui*	Void pointer to GUI
in	*sEvent*	Event data structure

Returns

true if success, false if fail

5.2.3.7 gslc_tsElem∗ gslc_CollectFindElemById (gslc_tsCollect ∗ *pCollect,* int16_t *nElemId*)

Find an element in a collection by its Element ID.

Parameters

in	*pCollect*	Pointer to the collection
in	*nElemId*	Element ID to search for

Returns

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

5.2.3.8 gslc_tsElem∗ gslc_CollectFindElemFromCoord (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	*pCollect*	Pointer to the collection
in	*nX*	Absolute X coordinate to use for search
in	*nY*	Absolute Y coordinate to use for search

Returns

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

5.2.3.9 gslc_tsElem∗ gslc_CollectGetElemTracked (gslc_tsCollect ∗ *pCollect*)

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

Parameters

in	*pCollect*	Pointer to the collection

Returns

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

5.2.3.10 int gslc_CollectGetNextId (gslc_tsCollect ∗ *pCollect*)

Allocate the next available Element ID in a collection.

Parameters

`in`	*pCollect*	Pointer to the collection

Returns

Element ID that is reserved for use

5.2.3.11 bool gslc_CollectGetRedraw (gslc_tsCollect ∗ *pCollect*)

Determine if any elements in a collection need redraw.

Parameters

`in`	*pCollect*	Pointer to Element collection

Returns

True if redraw required, false otherwise

5.2.3.12 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`	*pCollect*	Pointer to the collection
`in`	*asElem*	Internal element array storage to associate with the collection
`in`	*nElemMax*	Maximum number of elements that can be added to the internal element array (ie. RAM))
`in`	*asElemRef*	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`	*nElemRefMax*	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

5.2.3.13 void gslc_CollectSetElemTracked (gslc_tsCollect ∗ *pCollect,* gslc_tsElem ∗ *pElem*)

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

Parameters

`in`	*pCollect*	Pointer to the collection
`in`	*pElem*	Ptr to element to mark as being tracked

Returns

none

5.2.3.14 void gslc_CollectSetEventFunc (gslc_tsCollect ∗ *pCollect,* GSLC_CB_EVENT *funcCb*)

Assign the event callback function for an element collection.

Parameters

in	*pCollect*	Pointer to collection
in	*funcCb*	Function pointer to event routine (or NULL for default))

Returns

none

5.2.3.15 void gslc_CollectSetParent (gslc_tsCollect ∗ *pCollect,* gslc_tsElem ∗ *pElemParent*)

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	*pCollect*	Pointer to the collection
in	*pElemParent*	Ptr to element that is the parent

Returns

none

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

Handle touch events within the element collection.

Parameters

in	*pGui*	Pointer to the GUI
in	*pCollect*	Ptr to the element collection
in	*pEventTouch*	Ptr to the touch event structure

Returns

none

5.2.3.17 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	*colStart*	Starting color
in	*colEnd*	Ending color
in	*nMidAmt*	Position (0..1000) between start and end color at which the midpoint between colors should appear. Normally set to 500 (half-way).

in	*nBlendAmt*	The position (0..1000) between start and end at which we want to calculate the resulting blended color.

Returns

Blended color

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

Returns

Blended color

5.2.3.19 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.
- gslc_cosFX(nAngDeg∗64)/32768.0 = cos(nAngDeg∗2pi/360)

Parameters

in	*n64Ang*	Angle (in units of 1/64 degrees)

Returns

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

5.2.3.20 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

in	*pFmt*	Format string to use for printing
in	*...*	Variable parameter list

Returns

none

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

Draw a filled circle.

Parameters

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

Returns

none

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

Draw a filled quadrilateral.

Parameters

in	*pGui*	Pointer to GUI
in	*psPt*	Pointer to array of 4 points
in	*nCol*	Color RGB value for the frame

Returns

true if success, false if error

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

Draw a filled rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to fill
in	*nCol*	Color RGB value to fill

Returns

none

5.2.3.24 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	*pGui*	Pointer to GUI
in	*nX0*	X Coordinate #1
in	*nY0*	Y Coordinate #1
in	*nX1*	X Coordinate #2
in	*nY1*	Y Coordinate #2
in	*nX2*	X Coordinate #3
in	*nY2*	Y Coordinate #3
in	*nCol*	Color RGB value for the fill

Returns

true if success, false if error

5.2.3.25 void gslc_DrawFrameCircle (gslc_tsGui ∗ *pGui,* int16_t *nMidX,* int16_t *nMidY,* uint16_t *nRadius,* gslc_tsColor *nCol*)

Draw a framed circle.

Parameters

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

Returns

none

5.2.3.26 void gslc_DrawFrameQuad (gslc_tsGui ∗ *pGui,* gslc_tsPt ∗ *psPt,* gslc_tsColor *nCol*)

Draw a framed quadrilateral.

Parameters

in	*pGui*	Pointer to GUI
in	*psPt*	Pointer to array of 4 points
in	*nCol*	Color RGB value for the frame

Returns

true if success, false if error

5.2.3.27 void gslc_DrawFrameRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a framed rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to frame
in	*nCol*	Color RGB value for the frame

Returns

none

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

Returns

true if success, false if error

5.2.3.29 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	*pGui*	Pointer to GUI
in	*nX0*	X coordinate of line startpoint
in	*nY0*	Y coordinate of line startpoint
in	*nX1*	X coordinate of line endpoint
in	*nY1*	Y coordinate of line endpoint
in	*nCol*	Color RGB value for the line

Returns

none

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

Returns

none

5.2.3.31 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	*pGui*	Pointer to GUI
in	*nX*	X coordinate of line startpoint
in	*nY*	Y coordinate of line startpoint
in	*nRadStart*	Starting radius of line
in	*nRadEnd*	Ending radius of line
in	*n64Ang*	Angle of ray (degrees ∗ 64). 0 is up, +90∗64 is to right From -180∗64 to +180∗64
in	*nCol*	Color RGB value for the line

Returns

none

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

Returns

none

5.2.3.33 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	*pGui*	Pointer to GUI
in	*nX*	Pixel X coordinate to set
in	*nY*	Pixel Y coordinate to set
in	*nCol*	Color pixel value to assign

Returns

none

5.2.3.34 gslc_tsElem∗ gslc_ElemAdd (gslc_tsGui ∗ *pGui,* int16_t *nPageId,* gslc_tsElem ∗ *pElem,* gslc_teElemRefFlags *eFlags*)

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

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

Parameters

in	*pGui*	Pointer to GUI
in	*nPageId*	Page ID to add element to (GSLC_PAGE_NONE to skip in case of temporary creation for compound elements)
in	*pElem*	Pointer to Element to add
in	*eFlags*	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 or NULL if fail

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

Create a new element with default styling.

Parameters

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

Returns

Initialized structure

5.2.3.36 gslc_tsElem∗ 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`	*pGui*	Pointer to GUI
`in`	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
`in`	*nPage*	Page ID to attach element to
`in`	*rElem*	Rectangle coordinates defining box size

Returns

Pointer to the Element or NULL if failure

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

Returns

Pointer to the Element or NULL if failure

5.2.3.38 gslc_tsElem∗ 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`	*pGui*	Pointer to GUI

in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*rElem*	Rectangle coordinates defining text background size
in	*pStrBuf*	String to copy into element
in	*nStrBufMax*	Maximum length of string buffer (pStrBuf). Only applicable if GSLC_LOCAL↩ _STR=0. Ignored if GSLC_LOCAL_STR=1.)
in	*nFontId*	Font ID to use for text display
in	*cbTouch*	Callback for touch events

Returns

Pointer to the Element or NULL if failure

5.2.3.39 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*rElem*	Rectangle coordinates defining box size
in	*sImgRef*	Image reference to load

Returns

Pointer to the Element or NULL if failure

5.2.3.40 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*nX0*	X coordinate of line startpoint
in	*nY0*	Y coordinate of line startpoint
in	*nX1*	X coordinate of line endpoint
in	*nY1*	Y coordinate of line endpoint

Returns

Pointer to the Element or NULL if failure

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

Returns

Pointer to the Element or NULL if failure

5.2.3.42 void gslc_ElemDestruct (gslc_tsElem ∗ *pElem*)

Free up any members associated with an element.

Parameters

in	*pElem*	Pointer to element

Returns

none

5.2.3.43 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	*pGui*	Pointer to GUI
in	*nPageId*	ID of page containing element
in	*nElemId*	ID of element

Returns

none

5.2.3.44 bool gslc_ElemDrawByRef (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_teRedrawType *eRedraw*)

Draw an element to the active display.

- Element is referenced by an element pointer

Parameters

`in`	*pGui*	Pointer to GUI
`in`	*pElem*	Ptr to Element to draw
`in`	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.2.3.45 bool gslc_ElemEvent (void ∗ *pvGui,* gslc_tsEvent *sEvent*)

Common event handler function for an element.

Parameters

`in`	*pvGui*	Void pointer to GUI
`in`	*sEvent*	Event data structure

Returns

true if success, false if fail

5.2.3.46 bool gslc_ElemGetGlow (gslc_tsElem ∗ *pElem*)

Get the glowing indicator for an element.

Parameters

`in`	*pElem*	Pointer to Element

Returns

True if element is glowing

5.2.3.47 bool gslc_ElemGetGlowEn (gslc_tsElem ∗ *pElem*)

Get the glowing enable for an element.

Parameters

`in`	*pElem*	Pointer to Element

Returns

True if element supports glowing

5.2.3.48 int gslc_ElemGetGroup (gslc_tsElem ∗ *pElem*)

Get the group ID for an element.

Parameters

in	*pElem*	Pointer to Element

Returns

Group ID or GSLC_GROUP_ID_NONE if unassigned

5.2.3.49 int gslc_ElemGetId (gslc_tsElem ∗ *pElem*)

Get an Element ID from an element structure.

Parameters

in	*pElem*	Pointer to element structure

Returns

ID of element or GSLC_ID_NONE if not found

5.2.3.50 gslc_teRedrawType gslc_ElemGetRedraw (gslc_tsElem ∗ *pElem*)

Get the need-redraw status for an element.

Parameters

in	*pElem*	Pointer to Element

Returns

Redraw status

5.2.3.51 bool gslc_ElemOwnsCoord (gslc_tsElem ∗ *pElem,* 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	*pElem*	Element used for boundary test
in	*nX*	X coordinate to test
in	*nY*	Y coordinate to test
in	*bOnlyClickEn*	Only output true if element was also marked as "clickable" (eg. bClickEn=true)

Returns

true if inside element, false otherwise

5.2.3.52 bool gslc_ElemSendEventTouch (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElemTracked,* 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	*pGui*	Pointer to GUI
in	*pElemTracked*	Pointer to tracked Element (or NULL for none))
in	*eTouch*	Touch event type
in	*nX*	X coordinate of event (absolute coordinate)
in	*nY*	Y coordinate of event (absolute coordinate)

Returns

true if success, false if error

5.2.3.53 void gslc_ElemSetCol (gslc_tsElem ∗ *pElem,* gslc_tsColor *colFrame,* gslc_tsColor *colFill,* gslc_tsColor *colFillGlow*)

Update the common color selection for an Element.

Parameters

in	*pElem*	Pointer to Element
in	*colFrame*	Color for the frame
in	*colFill*	Color for the fill
in	*colFillGlow*	Color for the fill when glowing

Returns

none

5.2.3.54 void gslc_ElemSetDrawFunc (gslc_tsElem ∗ *pElem,* 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	*pElem*	Pointer to Element
in	*funcCb*	Function pointer to drawing routine (or NULL for default))

Returns

none

5.2.3.55 void gslc_ElemSetEventFunc (gslc_tsElem ∗ *pElem,* GSLC_CB_EVENT *funcCb*)

Assign the event callback function for a element.

Parameters

in	*pElem*	Pointer to element

in	*funcCb*	Function pointer to event routine (or NULL for default))

Returns

none

5.2.3.56 void gslc_ElemSetFillEn (gslc_tsElem ∗ *pElem*, bool *bFillEn*)

Set the fill state for an Element.

Parameters

in	*pElem*	Pointer to Element
in	*bFillEn*	True if filled, false otherwise

Returns

none

5.2.3.57 void gslc_ElemSetFrameEn (gslc_tsElem ∗ *pElem*, bool *bFrameEn*)

Set the frame state for an Element.

Parameters

in	*pElem*	Pointer to Element
in	*bFrameEn*	True if framed, false otherwise

Returns

none

5.2.3.58 void gslc_ElemSetGlow (gslc_tsElem ∗ *pElem*, bool *bGlowing*)

Update the glowing indicator for an element.

Parameters

in	*pElem*	Pointer to Element
in	*bGlowing*	True if element is glowing

Returns

none

5.2.3.59 void gslc_ElemSetGlowCol (gslc_tsElem ∗ *pElem*, gslc_tsColor *colFrameGlow*, gslc_tsColor *colFillGlow*, gslc_tsColor *colTxtGlow*)

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

Parameters

`in`	*pElem*	Pointer to Element
`in`	*colFrameGlow*	Color for the frame when glowing
`in`	*colFillGlow*	Color for the fill when glowing
`in`	*colTxtGlow*	Color for the text when glowing

Returns

none

5.2.3.60 void gslc_ElemSetGlowEn (gslc_tsElem ∗ *pElem,* bool *bGlowEn*)

Update the glowing enable for an element.

Parameters

`in`	*pElem*	Pointer to Element
`in`	*bGlowEn*	True if element should support glowing

Returns

none

5.2.3.61 void gslc_ElemSetGroup (gslc_tsElem ∗ *pElem,* int *nGroupId*)

Set the group ID for an element.

- Typically used to associate radio button elements together

Parameters

`in`	*pElem*	Pointer to Element
`in`	*nGroupId*	Group ID to assign

Returns

none

5.2.3.62 void gslc_ElemSetImage (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_tsImgRef *sImgRef,* gslc_tsImgRef *sImgRefSel*)

Set an element to use a bitmap image.

Parameters

`in`	*pGui*	Pointer to GUI
`in`	*pElem*	Pointer to Element to update
`in`	*sImgRef*	Image reference (normal state)
`in`	*sImgRefSel*	Image reference (glowing state)

Returns

none

5.2.3.63 void gslc_ElemSetRedraw (gslc_tsElem ∗ *pElem,* gslc_teRedrawType *eRedraw*)

Update the need-redraw status for an element.

Parameters

`in`	*pElem*	Pointer to Element
`in`	*eRedraw*	Redraw state to set

Returns

none

5.2.3.64 void gslc_ElemSetStyleFrom (gslc_tsElem ∗ *pElemSrc*, gslc_tsElem ∗ *pElemDest*)

Copy style settings from one element to another.

Parameters

`in`	*pElemSrc*	Pointer to source Element
`in`	*pElemDest*	Pointer to destination Element

Returns

none

5.2.3.65 void gslc_ElemSetTickFunc (gslc_tsElem ∗ *pElem*, 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`	*pElem*	Pointer to Element
`in`	*funcCb*	Function pointer to tick routine (or NULL for none))

Returns

none

5.2.3.66 void gslc_ElemSetTxtAlign (gslc_tsElem ∗ *pElem,* unsigned *nAlign*)

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

Parameters

in	*pElem*	Pointer to Element
in	*nAlign*	Alignment to specify: • 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

5.2.3.67 void gslc_ElemSetTxtCol (gslc_tsElem ∗ *pElem,* gslc_tsColor *colVal*)

Update the text string color associated with an Element ID.

Parameters

in	*pElem*	Pointer to Element
in	*colVal*	RGB color to change to

Returns

none

5.2.3.68 void gslc_ElemSetTxtMargin (gslc_tsElem ∗ *pElem,* unsigned *nMargin*)

Set the margin around of a textual element.

Parameters

in	*pElem*	Pointer to Element

in	*nMargin*	Number of pixels gap to leave surrounding text

Returns

none

5.2.3.69 void gslc_ElemSetTxtMem (gslc_tsElem ∗ *pElem*, gslc_teTxtFlags *eFlags*)

Update the text string location in memory.

Parameters

in	*pElem*	Pointer to Element
in	*eFlags*	Flags associated with text memory location (GSLC_TXT_MEM_∗)

Returns

none

5.2.3.70 void gslc_ElemSetTxtStr (gslc_tsElem ∗ *pElem*, const char ∗ *pStr*)

Update the text string associated with an Element ID.

Parameters

in	*pElem*	Pointer to Element
in	*pStr*	String to copy into element

Returns

none

5.2.3.71 void gslc_ElemUpdateFont (gslc_tsGui ∗ *pGui*, gslc_tsElem ∗ *pElem*, int *nFontId*)

Update the Font selected for an Element's text.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element
in	*nFontId*	Font ID to select

Returns

none

5.2.3.72 gslc_tsEvent gslc_EventCreate (gslc_teEventType *eType*, uint8_t *nSubType*, void ∗ *pvScope*, void ∗ *pvData*)

Create an event structure.

Parameters

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

Returns

None

5.2.3.73 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	*rRect*	Rectangular region before resizing
in	*nExpandW*	Number of pixels to expand the width (if positive) of contract the width (if negative)
in	*nExpandH*	Number of pixels to expand the height (if positive) of contract the height (if negative)

Returns

gslc_tsRect() with resized dimensions

5.2.3.74 bool gslc_FontAdd (gslc_tsGui ∗ *pGui,* int16_t *nFontId,* const char ∗ *acFontName,* uint16_t *nFontSz*)

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

Parameters

in	*pGui*	Pointer to GUI
in	*nFontId*	ID to use when referencing this font
in	*acFontName*	Filename path to the font
in	*nFontSz*	Typeface size to use

Returns

true if load was successful, false otherwise

5.2.3.75 gslc_tsFont∗ gslc_FontGet (gslc_tsGui ∗ *pGui,* int16_t *nFontId*)

Fetch a font from its ID value.

Parameters

in	*pGui*	Pointer to GUI
in	*nFontId*	ID value used to reference the font (supplied originally to gslc_FontAdd()

Returns

A pointer to the font structure or NULL if error

5.2.3.76 gslc_tsImgRef gslc_GetImageFromFile (const char * *pFname,* gslc_teImgRefFlags *eFmt*)

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

Parameters

in	*pFname*	Pointer to filename string of image in filesystem
in	*eFmt*	Image format

Returns

Loaded image reference

5.2.3.77 gslc_tsImgRef gslc_GetImageFromProg (const unsigned char ∗ *pImgBuf*, gslc_teImgRefFlags *eFmt*)

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

Parameters

in	*pImgBuf*	Pointer to image buffer in memory
in	*eFmt*	Image format

Returns

Loaded image reference

5.2.3.78 gslc_tsImgRef gslc_GetImageFromRam (unsigned char ∗ *pImgBuf*, gslc_teImgRefFlags *eFmt*)

Create an image reference to a bitmap in SRAM.

Parameters

in	*pImgBuf*	Pointer to image buffer in memory
in	*eFmt*	Image format

Returns

Loaded image reference

5.2.3.79 gslc_tsImgRef gslc_GetImageFromSD (const char ∗ *pFname*, gslc_teImgRefFlags *eFmt*)

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

Parameters

in	*pFname*	Pointer to filename string of image in SD card
in	*eFmt*	Image format

Returns

Loaded image reference

5.2.3.80 int gslc_GetPageCur (gslc_tsGui ∗ *pGui*)

Fetch the current page ID.

Parameters

`in`	*pGui*	Pointer to GUI

Returns

Page ID

5.2.3.81 bool gslc_GetTouch (gslc_tsGui ∗ *pGui*, int16_t ∗ *pnX*, int16_t ∗ *pnY*, uint16_t ∗ *pnPress*)

Initialize the touchscreen device driver.

Parameters

`in`	*pGui*	Pointer to GUI
`out`	*pnX*	Ptr to int to contain latest touch X coordinate
`out`	*pnY*	Ptr to int to contain latest touch Y coordinate
`out`	*pnPress*	Ptr to int to contain latest touch pressure value

Returns

true if touch event, false otherwise

5.2.3.82 char∗ gslc_GetVer (gslc_tsGui ∗ *pGui*)

Get the GUIslice version number.

Returns

String containing version number

5.2.3.83 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`	*pGui*	Pointer to GUI

Returns

none

5.2.3.84 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	*pGui*	Pointer to GUI
in	*pvDriver*	Void pointer to Driver struct (gslc_tsDriver∗)
in	*asPage*	Pointer to Page array
in	*nMaxPage*	Size of Page array
in	*asFont*	Pointer to Font array
in	*nMaxFont*	Size of Font array

Returns

true if success, false if fail

5.2.3.85 void gslc_InitDebug (GSLC_CB_DEBUG_OUT *pfunc*)

Initialize debug output.

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

Parameters

in	*pfunc*	Pointer to user character-out function

Returns

none

5.2.3.86 bool gslc_InitTouch (gslc_tsGui ∗ *pGui,* const char ∗ *acDev*)

Initialize the touchscreen device driver.

Parameters

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

Returns

true if successful

5.2.3.87 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`	*nSelX*	X coordinate to test
`in`	*nSelY*	X coordinate to test
`in`	*rRect*	Rectangular region to compare against

Returns

true if inside region, false otherwise

5.2.3.88 bool gslc_IsInWH (gslc_tsGui ∗ *pGui*, 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`	*pGui*	Pointer to GUI
`in`	*nSelX*	X coordinate to test
`in`	*nSelY*	X coordinate to test
`in`	*nWidth*	Width to test against
`in`	*nHeight*	Height to test against

Returns

true if inside region, false otherwise

5.2.3.89 void gslc_OrderCoord (int16_t ∗ *pnX0*, int16_t ∗ *pnY0*, int16_t ∗ *pnX1*, int16_t ∗ *pnY1*)

5.2.3.90 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`	*pGui*	Pointer to GUI
`in`	*nPageId*	Page ID to assign
`in`	*psElem*	Internal element array storage to associate with the page
`in`	*nMaxElem*	Maximum number of elements that can be added to the internal element array (ie. RAM))
`in`	*psElemRef*	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	*nMaxElemRef*	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

5.2.3.91 void gslc_PageDestruct (gslc_tsPage ∗ *pPage*)

Free up any members associated with a page.

Parameters

in	*pPage*	Pointer to Page

Returns

none

5.2.3.92 bool gslc_PageEvent (void ∗ *pvGui,* gslc_tsEvent *sEvent*)

Common event handler function for a page.

Parameters

in	*pvGui*	Void pointer to GUI
in	*sEvent*	Event data structure

Returns

true if success, false if fail

5.2.3.93 gslc_tsPage∗ gslc_PageFindById (gslc_tsGui ∗ *pGui,* int16_t *nPageId*)

Find a page in the GUI by its ID.

Parameters

in	*pGui*	Pointer to GUI
in	*nPageId*	Page ID to search

Returns

Ptr to a page or NULL if none found

5.2.3.94 gslc_tsElem∗ 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`	*pGui*	Pointer to GUI
`in`	*nPageId*	Page ID to search
`in`	*nElemId*	Element ID to search

Returns

Ptr to an element or NULL if none found

5.2.3.95 bool gslc_PageFlipGet (gslc_tsGui ∗ *pGui*)

Get state of pending page flip state.

Parameters

`in`	*pGui*	Pointer to GUI

Returns

True if screen requires page flip

5.2.3.96 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`	*pGui*	Pointer to GUI

Returns

None

5.2.3.97 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`	*pGui*	Pointer to GUI
`in`	*bNeeded*	True if screen requires page flip

Returns

None

5.2.3.98 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	*pGui*	Pointer to GUI

Returns

none

5.2.3.99 bool gslc_PageRedrawGet (gslc_tsGui ∗ *pGui*)

Get the need-redraw status for the current page.

Parameters

in	*pGui*	Pointer to GUI

Returns

True if redraw required, false otherwise

5.2.3.100 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	*pGui*	Pointer to GUI

Returns

none

5.2.3.101 void gslc_PageRedrawSet (gslc_tsGui ∗ *pGui,* bool *bRedraw*)

Update the need-redraw status for the current page.

Parameters

in	*pGui*	Pointer to GUI
in	*bRedraw*	True if redraw required, false otherwise

Returns

none

5.2.3.102 void gslc_PageSetEventFunc (gslc_tsPage ∗ *pPage,* GSLC_CB_EVENT *funcCb*)

Assign the event callback function for a page.

Parameters

in	*pPage*	Pointer to page
in	*funcCb*	Function pointer to event routine (or NULL for default))

Returns

none

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

Convert polar coordinate to cartesian.

Parameters

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

Returns

none

5.2.3.104 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	*pGui*	Pointer to GUI

Returns

None

5.2.3.105 void gslc_ResetElem (gslc_tsElem ∗ *pElem*)

Initialize an Element struct.

Parameters

in	*pElem*	Pointer to Element

Returns

none

5.2.3.106 void gslc_ResetFont (gslc_tsFont ∗ *pFont*)

Initialize a Font struct.

Parameters

in	*pFont*	Pointer to Font

Returns

none

5.2.3.107 gslc_tsImgRef gslc_ResetImage ( )

Create a blank image reference structure.

Returns

Image reference struct

5.2.3.108 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	*pGui*	Pointer to GUI
in	*nCol*	RGB Color to use

Returns

true if success, false if fail

5.2.3.109 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	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

true if success, false if fail

5.2.3.110 bool gslc_SetClipRect (gslc_tsGui ∗ *pGui,* gslc_tsRect ∗ *pRect*)

Set the clipping rectangle for further drawing.

Parameters

in	*pGui*	Pointer to GUI
in	*pRect*	Pointer to Rect for clipping (or NULL for entire screen)

Returns

true if success, false if error

5.2.3.111 void gslc_SetPageCur (gslc_tsGui ∗ *pGui*, int16_t *nPageId*)

Select a new page for display.

Parameters

in	*pGui*	Pointer to GUI
in	*nPageId*	Page ID to select as current

Returns

none

5.2.3.112 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.
- gslc_sinFX(nAngDeg∗64)/32768.0 = sin(nAngDeg∗2pi/360)

Parameters

in	*n64Ang*	Angle (in units of 1/64 degrees)

Returns

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

5.2.3.113 void gslc_SwapCoords (int16_t ∗ *pnXa*, int16_t ∗ *pnYa*, int16_t ∗ *pnXb*, int16_t ∗ *pnYb*)

5.2.3.114 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	*pGui*	Pointer to GUI
in	*pPage*	Pointer to current page
in	*nX*	X coordinate of touch event

`in`	*nY*	Y coordinate of touch event
`in`	*nPress*	Pressure level of touch event (0 for none, else touch)

Returns

none

5.2.3.115 void gslc_Update (gslc_tsGui ∗ *pGui*)

Perform main GUIslice handling functions.

- Handles any touch events
- Performs any necessary screen redraw

Parameters

`in`	*pGui*	Pointer to GUI

Returns

None

5.2.4 Variable Documentation

5.2.4.1 GSLC_CB_DEBUG_OUT g_pfDebugOut = NULL

Global debug output function.

- The user assigns this function via gslc_InitDebug()

5.2.4.2 uint16_t m_nLUTSinF0X16

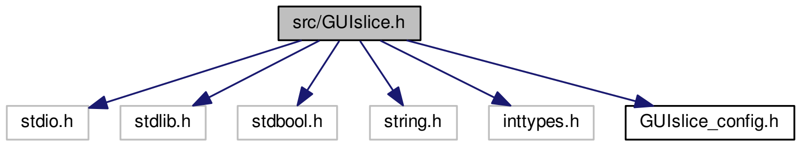
Initial value:

```
= {
  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,0xBFC6,0xC0D0,0xC1D7,0xC2DD,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,
      0xFAC4,
  0xFB13,0xFB61,0xFBAB,0xFBF4,0xFC3A,0xFC7D,0xFCBE,0xFCFD,0xFD39,0xFD73,0xFDAA,0xFDDF,0xFE12,0xFE42,0xFE6F,
      0xFE9A,
  0xFEC3,0xFEE9,0xFF0D,0xFF2E,0xFF4D,0xFF69,0xFF83,0xFF9B,0xFFB0,0xFFC2,0xFFD2,0xFFE0,0xFFEB,0xFFF3,0xFFFA,
      0xFFFD,
  0xFFFF,
}
```

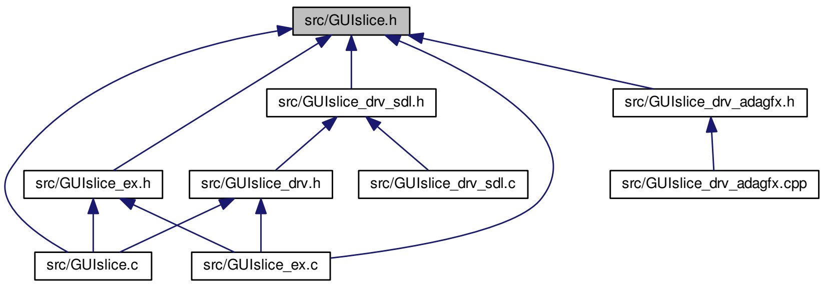
5.3 src/GUIslice.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <string.h>
#include <inttypes.h>
#include "GUIslice_config.h"
```

Include dependency graph for GUIslice.h:



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



Classes

- struct gslc_tsRect

  Rectangular region. Defines X,Y corner coordinates plus dimensions.
- struct gslc_tsPt

  Define point coordinates.
- struct gslc_tsColor

  Color structure. Defines RGB triplet.
- struct gslc_tsEvent

  Event structure.
- struct gslc_tsEventTouch

Structure used to pass touch data through event.
- struct gslc_tsFont

  Font reference structure.
- struct gslc_tsImgRef

  Image reference structure.
- struct gslc_tsElem

  Element Struct.
- struct gslc_tsElemRef

  Element reference structure.
- struct gslc_tsCollect

  Element collection struct.
- struct gslc_tsPage

  Page structure.
- struct gslc_tsGui

  GUI structure.

Macros

- #define GSLC_2PI 6.28318530718
- #define GSLC_ALIGNV_TOP 0x10

  Vertical align to top.
- #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}

Red (dark4)
- #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_MAX_EVT 30
- #define GSLC_DEBUG_PRINT(sFmt,...)

  Macro to enable optional debug output.
- #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_ElemCreateBox_P(pGui, nElemId, nPage, nX, nY, nW, nH, colFrame, colFill, bFrameEn, bFillEn)

  Create a read-only box 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 ∗pvElem, gslc_teRedrawType eRedraw)

    Callback function for element drawing.
- typedef bool(∗ GSLC_CB_TOUCH)(void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nX, int16_t nY)

    Callback function for element touch tracking.
- typedef bool(∗ GSLC_CB_TICK)(void ∗pvGui, void ∗pvElem)

    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_TOUC↩
  H_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_TO↩
  UCH_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_EVTS↩
  UB_DRAW_FORCE }

  Event sub-types.
- enum gslc_teRedrawType { GSLC_REDRAW_NONE, GSLC_REDRAW_FULL, GSLC_REDRAW_INC }

  Redraw types.
- enum gslc_teElemRefFlags { GSLC_ELEMREF_NONE = 0, GSLC_ELEMREF_SRC_RAM = (1<<0), GS↩
  LC_ELEMREF_SRC_PROG = (2<<0), GSLC_ELEMREF_SRC = (7<<0) }

  Element reference flags: Describes characteristics of an element.
- enum gslc_teImgRefFlags {
  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_FM↩
  T_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 ∗as↩
  Font, 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_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.
- 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 (gslc_tsGui ∗pGui, 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_tsImgRef gslc_ResetImage ()

  Create a blank image reference structure.
- gslc_tsImgRef gslc_GetImageFromFile (const char ∗pFname, gslc_teImgRefFlags eFmt)

  Create an image reference to a bitmap file in LINUX filesystem.
- gslc_tsImgRef gslc_GetImageFromSD (const char ∗pFname, gslc_teImgRefFlags eFmt)

  Create an image reference to a bitmap file in SD card.
- gslc_tsImgRef gslc_GetImageFromRam (unsigned char ∗pImgBuf, gslc_teImgRefFlags eFmt)

  Create an image reference to a bitmap in SRAM.
- gslc_tsImgRef gslc_GetImageFromProg (const unsigned char ∗pImgBuf, gslc_teImgRefFlags 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.
- 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 n←
Col)

  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, const char ∗acFontName, 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_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_ts↩ElemRef ∗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_tsElem ∗ 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.
- gslc_tsElem ∗ 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_tsElem ∗ 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_tsElem ∗ gslc_ElemCreateBtnImg (gslc_tsGui ∗pGui, int16_t nElemId, int16_t nPage, gslc_tsRect r↩Elem, gslc_tsImgRef sImgRef, gslc_tsImgRef sImgRefSel, GSLC_CB_TOUCH cbTouch)

    Create a graphical Button Element.
- gslc_tsElem ∗ gslc_ElemCreateBox (gslc_tsGui ∗pGui, int16_t nElemId, int16_t nPage, gslc_tsRect rElem)

    Create a Box Element.

- gslc_tsElem ∗ 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_tsElem ∗ 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_tsElem ∗pElem)

    Get an Element ID from an element structure.
- void gslc_ElemSetFillEn (gslc_tsElem ∗pElem, bool bFillEn)

    Set the fill state for an Element.
- void gslc_ElemSetFrameEn (gslc_tsElem ∗pElem, bool bFrameEn)

    Set the frame state for an Element.
- void gslc_ElemSetCol (gslc_tsElem ∗pElem, gslc_tsColor colFrame, gslc_tsColor colFill, gslc_tsColor col↩ FillGlow)

    Update the common color selection for an Element.
- void gslc_ElemSetGlowCol (gslc_tsElem ∗pElem, gslc_tsColor colFrameGlow, gslc_tsColor colFillGlow, gslc_tsColor colTxtGlow)

    Update the common color selection for glowing state of an Element.
- void gslc_ElemSetGroup (gslc_tsElem ∗pElem, int nGroupId)

    Set the group ID for an element.
- int gslc_ElemGetGroup (gslc_tsElem ∗pElem)

    Get the group ID for an element.
- void gslc_ElemSetTxtAlign (gslc_tsElem ∗pElem, unsigned nAlign)

    Set the alignment of a textual element (horizontal and vertical)
- void gslc_ElemSetTxtMargin (gslc_tsElem ∗pElem, unsigned nMargin)

    Set the margin around of a textual element.
- void gslc_ElemSetTxtStr (gslc_tsElem ∗pElem, const char ∗pStr)

    Update the text string associated with an Element ID.
- void gslc_ElemSetTxtCol (gslc_tsElem ∗pElem, gslc_tsColor colVal)

    Update the text string color associated with an Element ID.
- void gslc_ElemSetTxtMem (gslc_tsElem ∗pElem, gslc_teTxtFlags eFlags)

    Update the text string location in memory.
- void gslc_ElemUpdateFont (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, int nFontId)

    Update the Font selected for an Element's text.
- void gslc_ElemSetRedraw (gslc_tsElem ∗pElem, gslc_teRedrawType eRedraw)

    Update the need-redraw status for an element.
- gslc_teRedrawType gslc_ElemGetRedraw (gslc_tsElem ∗pElem)

    Get the need-redraw status for an element.
- void gslc_ElemSetGlowEn (gslc_tsElem ∗pElem, bool bGlowEn)

    Update the glowing enable for an element.
- void gslc_ElemSetStyleFrom (gslc_tsElem ∗pElemSrc, gslc_tsElem ∗pElemDest)

    Copy style settings from one element to another.
- bool gslc_ElemGetGlowEn (gslc_tsElem ∗pElem)

    Get the glowing enable for an element.
- void gslc_ElemSetGlow (gslc_tsElem ∗pElem, bool bGlowing)

    Update the glowing indicator for an element.
- bool gslc_ElemGetGlow (gslc_tsElem ∗pElem)

    Get the glowing indicator for an element.
- void gslc_ElemSetEventFunc (gslc_tsElem ∗pElem, GSLC_CB_EVENT funcCb)

    Assign the event callback function for a element.
- void gslc_ElemSetDrawFunc (gslc_tsElem ∗pElem, GSLC_CB_DRAW funcCb)

Assign the drawing callback function for an element.
- void gslc_ElemSetTickFunc (gslc_tsElem ∗pElem, GSLC_CB_TICK funcCb)

  Assign the tick callback function for an element.
- bool gslc_ElemOwnsCoord (gslc_tsElem ∗pElem, 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_tsElem ∗ gslc_CollectElemAdd (gslc_tsCollect ∗pCollect, const gslc_tsElem ∗pElem, gslc_teElemRef←Flags eFlags)

  Add an element to a collection.
- bool gslc_CollectGetRedraw (gslc_tsCollect ∗pCollect)

  Determine if any elements in a collection need redraw.
- gslc_tsElem ∗ gslc_CollectFindElemById (gslc_tsCollect ∗pCollect, int16_t nElemId)

  Find an element in a collection by its Element ID.
- gslc_tsElem ∗ gslc_CollectFindElemFromCoord (gslc_tsCollect ∗pCollect, int16_t nX, int16_t nY)

  Find an element in a collection by a coordinate coordinate.
- int gslc_CollectGetNextId (gslc_tsCollect ∗pCollect)

  Allocate the next available Element ID in a collection.
- gslc_tsElem ∗ gslc_CollectGetElemTracked (gslc_tsCollect ∗pCollect)

  Get the element within a collection that is currently being tracked.
- void gslc_CollectSetElemTracked (gslc_tsCollect ∗pCollect, gslc_tsElem ∗pElem)

  Set the element within a collection that is currently being tracked.
- void gslc_CollectSetParent (gslc_tsCollect ∗pCollect, gslc_tsElem ∗pElemParent)

  Assign the parent element reference to all elements within a collection.
- void gslc_CollectSetEventFunc (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_ts←Rect rElem, char ∗pStrBuf, uint8_t nStrBufMax, int16_t nFontId)

  Create a new element with default styling.
- gslc_tsElem ∗ gslc_ElemAdd (gslc_tsGui ∗pGui, int16_t nPageId, gslc_tsElem ∗pElem, gslc_teElemRefFlags eFlags)

  Add the Element to the list of generated elements in the GUI environment.
- bool gslc_SetClipRect (gslc_tsGui ∗pGui, gslc_tsRect ∗pRect)

  Set the clipping rectangle for further drawing.
- void gslc_ElemSetImage (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, gslc_tsImgRef sImgRef, gslc_tsImgRef 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_tsElem ∗pElem, gslc_teRedrawType 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_tsPage ∗pPage)

  Free up any members associated with a page.
- void gslc_CollectDestruct (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_tsElem ∗pElemTracked, gslc_teTouch 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.

5.3.1 Macro Definition Documentation

5.3.1.1 #define GSLC_2PI 6.28318530718

5.3.1.2 #define GSLC_ALIGN_BOT_LEFT GSLC_ALIGNH_LEFT | GSLC_ALIGNV_BOT

Align to bottom-left.

5.3.1.3 #define GSLC_ALIGN_BOT_MID GSLC_ALIGNH_MID | GSLC_ALIGNV_BOT

Align to middle of bottom.

5.3.1.4 #define GSLC_ALIGN_BOT_RIGHT GSLC_ALIGNH_RIGHT | GSLC_ALIGNV_BOT

Align to bottom-right.

5.3.1.5 #define GSLC_ALIGN_MID_LEFT GSLC_ALIGNH_LEFT | GSLC_ALIGNV_MID

Align to middle of left side.

5.3.1.6 #define GSLC_ALIGN_MID_MID GSLC_ALIGNH_MID | GSLC_ALIGNV_MID

Align to center.

5.3.1.7 #define GSLC_ALIGN_MID_RIGHT GSLC_ALIGNH_RIGHT | GSLC_ALIGNV_MID

Align to middle of right side.

5.3.1.8 #define GSLC_ALIGN_TOP_LEFT GSLC_ALIGNH_LEFT | GSLC_ALIGNV_TOP

Align to top-left.

5.3.1.9 #define GSLC_ALIGN_TOP_MID GSLC_ALIGNH_MID | GSLC_ALIGNV_TOP

Align to middle of top.

5.3.1.10 #define GSLC_ALIGN_TOP_RIGHT GSLC_ALIGNH_RIGHT | GSLC_ALIGNV_TOP

Align to top-right.

5.3.1.11 #define GSLC_ALIGNH_LEFT 0x01

Horizontal align to left.

5.3.1.12 #define GSLC_ALIGNH_MID 0x02

Horizontal align to middle.

5.3.1.13 #define GSLC_ALIGNH_RIGHT 0x04

Horizontal align to right.

5.3.1.14 #define GSLC_ALIGNV_BOT 0x40

Vertical align to bottom.

5.3.1.15 #define GSLC_ALIGNV_MID 0x20

Vertical align to middle.

5.3.1.16 #define GSLC_ALIGNV_TOP 0x10

Vertical align to top.

5.3.1.17 #define GSLC_COL_BLACK (gslc_tsColor) { 0, 0, 0}

Black.

5.3.1.18 #define GSLC_COL_BLUE (gslc_tsColor) { 0, 0,255}

Blue.

5.3.1.19 #define GSLC_COL_BLUE_DK1 (gslc_tsColor) { 0, 0,224}

Blue (dark1)

5.3.1.20 #define GSLC_COL_BLUE_DK2 (gslc_tsColor) { 0, 0,192}

Blue (dark2)

5.3.1.21 #define GSLC_COL_BLUE_DK3 (gslc_tsColor) { 0, 0,160}

Blue (dark3)

5.3.1.22 #define GSLC_COL_BLUE_DK4 (gslc_tsColor) { 0, 0,128}

Blue (dark4)

5.3.1.23 #define GSLC_COL_BLUE_LT1 (gslc_tsColor) { 32, 32,255}

Blue (light1)

5.3.1.24 #define GSLC_COL_BLUE_LT2 (gslc_tsColor) { 64, 64,255}

Blue (light2)

5.3.1.25 #define GSLC_COL_BLUE_LT3 (gslc_tsColor) { 96, 96,255}

Blue (light3)

5.3.1.26 #define GSLC_COL_BLUE_LT4 (gslc_tsColor) {128,128,255}

Blue (light4)

5.3.1.27 #define GSLC_COL_BROWN (gslc_tsColor) {165,42,42}

Brown.

5.3.1.28 #define GSLC_COL_CYAN (gslc_tsColor) {0,255,255}

Cyan.

5.3.1.29 #define GSLC_COL_GRAY (gslc_tsColor) {128,128,128}

Gray.

5.3.1.30 #define GSLC_COL_GRAY_DK1 (gslc_tsColor) { 96, 96, 96}

Gray (dark)

5.3.1.31 #define GSLC_COL_GRAY_DK2 (gslc_tsColor) { 64, 64, 64}

Gray (dark)

5.3.1.32 #define GSLC_COL_GRAY_DK3 (gslc_tsColor) { 32, 32, 32}

Gray (dark)

5.3.1.33 #define GSLC_COL_GRAY_LT1 (gslc_tsColor) {160,160,160}

Gray (light1)

5.3.1.34 #define GSLC_COL_GRAY_LT2 (gslc_tsColor) {192,192,192}

Gray (light2)

5.3.1.35 #define GSLC_COL_GRAY_LT3 (gslc_tsColor) {224,224,224}

Gray (light3)

5.3.1.36 #define GSLC_COL_GREEN (gslc_tsColor) { 0,255, 0}

Green.

5.3.1.37 #define GSLC_COL_GREEN_DK1 (gslc_tsColor) { 0,224, 0}

Green (dark1)

5.3.1.38 #define GSLC_COL_GREEN_DK2 (gslc_tsColor) { 0,192, 0}

Green (dark2)

5.3.1.39 #define GSLC_COL_GREEN_DK3 (gslc_tsColor) { 0,160, 0}

Green (dark3)

5.3.1.40 #define GSLC_COL_GREEN_DK4 (gslc_tsColor) { 0,128, 0}

Green (dark4)

5.3.1.41 #define GSLC_COL_GREEN_LT1 (gslc_tsColor) { 32,255, 32}

Green (light1)

5.3.1.42 #define GSLC_COL_GREEN_LT2 (gslc_tsColor) { 64,255, 64}

Green (light2)

5.3.1.43 #define GSLC_COL_GREEN_LT3 (gslc_tsColor) { 96,255, 96}

Green (light3)

5.3.1.44 #define GSLC_COL_GREEN_LT4 (gslc_tsColor) {128,255,128}

Green (light4)

5.3.1.45 #define GSLC_COL_MAGENTA (gslc_tsColor) {255,0,255}

Magenta.

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

Orange.

5.3.1.47 #define GSLC_COL_PURPLE (gslc_tsColor) {128,0,128}

Purple.

5.3.1.48 #define GSLC_COL_RED (gslc_tsColor) {255, 0, 0}

Red.

5.3.1.49 #define GSLC_COL_RED_DK1 (gslc_tsColor) {224, 0, 0}

Red (dark1)

5.3.1.50 #define GSLC_COL_RED_DK2 (gslc_tsColor) {192, 0, 0}

Red (dark2)

5.3.1.51 #define GSLC_COL_RED_DK3 (gslc_tsColor) {160, 0, 0}

Red (dark3)

5.3.1.52 #define GSLC_COL_RED_DK4 (gslc_tsColor) {128, 0, 0}

Red (dark4)

5.3.1.53 #define GSLC_COL_RED_LT1 (gslc_tsColor) {255, 32, 32}

Red (light1)

5.3.1.54 #define GSLC_COL_RED_LT2 (gslc_tsColor) {255, 64, 64}

Red (light2)

5.3.1.55 #define GSLC_COL_RED_LT3 (gslc_tsColor) {255, 96, 96}

Red (light3)

5.3.1.56 #define GSLC_COL_RED_LT4 (gslc_tsColor) {255,128,128}

Red (light4)

5.3.1.57 #define GSLC_COL_TEAL (gslc_tsColor) {0,128,128}

Teal.

5.3.1.58 #define GSLC_COL_WHITE (gslc_tsColor) {255,255,255}

White.

5.3.1.59 #define GSLC_COL_YELLOW (gslc_tsColor) {255,255,0}

Yellow.

5.3.1.60 #define GSLC_COL_YELLOW_DK (gslc_tsColor) {64,64,0}

Yellow (dark)

5.3.1.61 #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_D↩ EBUG_PRINT("Loaded OK",0);

Parameters

in	*sFmt*	Format string for debug message

5.3.1.62 #define gslc_ElemCreateBox_P(*pGui, nElemId, nPage, nX, nY, nW, nH, colFrame, colFill, bFrameEn, bFillEn*)

Value:

```
static const gslc_tsElem sElem##nElemId = {                              \
      nElemId,                                                            \
      true,                                                               \
      GSLC_TYPE_BOX,                                                      \
      (gslc_tsRect){nX,nY,nW,nH},                                         \
      GSLC_GROUP_ID_NONE,false,false,bFrameEn,bFillEn,                    \
      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,                                                               \
      NULL,                                                               \
      NULL,                                                               \
      NULL,                                                               \
      false,                                                              \
      false,                                                              \
  };                                                                      \
  gslc_ElemAdd(pGui,nPage,(gslc_tsElem*)&sElem##nElemId,
      GSLC_ELEMREF_SRC_RAM);
```

Create a read-only box element.

Parameters

in	*pGui*	Pointer to GUI
in	*nElemId*	Unique element ID to assign
in	*nPage*	Page ID to attach element to
in	*nX*	X coordinate of element
in	*nY*	Y coordinate of element
in	*nW*	Width of element
in	*nH*	Height of element
in	*colFrame*	Color for the frame
in	*colFill*	Color for the fill
in	*bFrameEn*	True if framed, false otherwise
in	*bFillEn*	True if filled, false otherwise

5.3.1.63 #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 gslc_tsElem sElem##nElemId = {                             \
      nElemId,                                                            \
      true,                                                               \
      GSLC_TYPE_TXT,                                                      \
      (gslc_tsRect){nX,nY,nW,nH},                                         \
      GSLC_GROUP_ID_NONE,false,false,bFrameEn,bFillEn,                    \
      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,                                                   \
      false,                                                  \
      false,                                                  \
  };                                                          \
  gslc_ElemAdd(pGui,nPage,(gslc_tsElem*)&sElem##nElemId,
      GSLC_ELEMREF_SRC_RAM);
```

Create a read-only text element.

Parameters

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

5.3.1.64 #define GSLC_MAX_EVT 30

5.3.2 Typedef Documentation

5.3.2.1 typedef int16_t(∗ GSLC_CB_DEBUG_OUT)(char ch)

5.3.2.2 typedef bool(∗ GSLC_CB_DRAW)(void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

Callback function for element drawing.

5.3.2.3 typedef bool(∗ GSLC_CB_EVENT)(void ∗pvGui, gslc_tsEvent sEvent)

Callback function for element drawing.

5.3.2.4 typedef bool(∗ GSLC_CB_TICK)(void ∗pvGui, void ∗pvElem)

Callback function for element tick.

5.3.2.5 typedef bool(∗ GSLC_CB_TOUCH)(void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nX, int16_t nY)

Callback function for element touch tracking.

5.3.2.6 typedef struct gslc_tsColor gslc_tsColor

Color structure. Defines RGB triplet.

5.3.2.7 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.

5.3.2.8 typedef struct gslc_tsEvent gslc_tsEvent

Event structure.

5.3.2.9 typedef struct gslc_tsEventTouch gslc_tsEventTouch

Structure used to pass touch data through event.

5.3.2.10 typedef struct gslc_tsPt gslc_tsPt

Define point coordinates.

5.3.2.11 typedef struct gslc_tsRect gslc_tsRect

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

5.3.3 Enumeration Type Documentation

5.3.3.1 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)

5.3.3.2 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.

5.3.3.3 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 stored in RAM (internal element array)

GSLC_ELEMREF_SRC_PROG Element is stored in program memory (PROGMEM, read-only, external to element array)

GSLC_ELEMREF_SRC Mask for Source flags.

5.3.3.4 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.

5.3.3.5 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.

5.3.3.6 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.

5.3.3.7 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.

5.3.3.8 enum gslc_teImgRefFlags

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.

5.3.3.9 enum gslc_tePageId

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.

5.3.3.10 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.

5.3.3.11 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.

5.3.3.12 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

5.3.3.13 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.

5.3.4 Function Documentation

5.3.4.1 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`	*pClipRect*	Pointer to clipping region
`in,out`	*pnX0*	Ptr to X coordinate of line start
`in,out`	*pnY0*	Ptr to Y coordinate of line start
`in,out`	*pnX1*	Ptr to X coordinate of line end
`in,out`	*pnY1*	Ptr to Y coordinate of line end

Returns

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

5.3.4.2 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`	*pClipRect*	Pointer to clipping region
`in`	*nX*	X coordinate of point
`in`	*nY*	Y coordinate of point

Returns

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

5.3.4.3 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	*pClipRect*	Pointer to clipping region
in,out	*pRect*	Ptr to rectangle

Returns

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

5.3.4.4 void gslc_CollectDestruct (gslc_tsCollect ∗ *pCollect*)

Free up any members associated with an element collection.

Parameters

in	*pCollect*	Pointer to collection

Returns

none

5.3.4.5 gslc_tsElem∗ gslc_CollectElemAdd (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 are the call is complete.

Parameters

in	*pCollect*	Pointer to the collection
in	*pElem*	Ptr to the element to add
in	*eFlags*	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 in the collection that has been added or NULL if there was an error

5.3.4.6 bool gslc_CollectEvent (void ∗ *pvGui,* gslc_tsEvent *sEvent*)

Common event handler function for an element collection.

Parameters

in	*pvGui*	Void pointer to GUI
in	*sEvent*	Event data structure

Returns

true if success, false if fail

5.3.4.7 gslc_tsElem∗ gslc_CollectFindElemById (gslc_tsCollect ∗ *pCollect,* int16_t *nElemId*)

Find an element in a collection by its Element ID.

Parameters

`in`	*pCollect*	Pointer to the collection
`in`	*nElemId*	Element ID to search for

Returns

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

5.3.4.8 gslc_tsElem∗ gslc_CollectFindElemFromCoord (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`	*pCollect*	Pointer to the collection
`in`	*nX*	Absolute X coordinate to use for search
`in`	*nY*	Absolute Y coordinate to use for search

Returns

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

5.3.4.9 gslc_tsElem∗ gslc_CollectGetElemTracked (gslc_tsCollect ∗ *pCollect*)

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

Parameters

`in`	*pCollect*	Pointer to the collection

Returns

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

5.3.4.10 int gslc_CollectGetNextId (gslc_tsCollect ∗ *pCollect*)

Allocate the next available Element ID in a collection.

Parameters

`in`	*pCollect*	Pointer to the collection

Returns

Element ID that is reserved for use

5.3.4.11 bool gslc_CollectGetRedraw (gslc_tsCollect ∗ *pCollect*)

Determine if any elements in a collection need redraw.

Parameters

in	*pCollect*	Pointer to Element collection

Returns

True if redraw required, false otherwise

5.3.4.12 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	*pCollect*	Pointer to the collection
in	*asElem*	Internal element array storage to associate with the collection
in	*nElemMax*	Maximum number of elements that can be added to the internal element array (ie. RAM))
in	*asElemRef*	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	*nElemRefMax*	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

5.3.4.13 void gslc_CollectSetElemTracked (gslc_tsCollect ∗ *pCollect*, gslc_tsElem ∗ *pElem*)

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

Parameters

in	*pCollect*	Pointer to the collection
in	*pElem*	Ptr to element to mark as being tracked

Returns

none

5.3.4.14 void gslc_CollectSetEventFunc (gslc_tsCollect ∗ *pCollect*, GSLC_CB_EVENT *funcCb*)

Assign the event callback function for an element collection.

Parameters

in	*pCollect*	Pointer to collection
in	*funcCb*	Function pointer to event routine (or NULL for default))

Returns

none

5.3.4.15 void gslc_CollectSetParent (**gslc_tsCollect** ∗ *pCollect,* **gslc_tsElem** ∗ *pElemParent*)

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`	*pCollect*	Pointer to the collection
`in`	*pElemParent*	Ptr to element that is the parent

Returns

none

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

Handle touch events within the element collection.

Parameters

`in`	*pGui*	Pointer to the GUI
`in`	*pCollect*	Ptr to the element collection
`in`	*pEventTouch*	Ptr to the touch event structure

Returns

none

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

Returns

Blended color

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

Returns

Blended color

5.3.4.19 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.
- gslc_cosFX(nAngDeg∗64)/32768.0 = cos(nAngDeg∗2pi/360)

Parameters

in	*n64Ang*	Angle (in units of 1/64 degrees)

Returns

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

5.3.4.20 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

in	*pFmt*	Format string to use for printing
in	...	Variable parameter list

Returns

none

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

Draw a filled circle.

Parameters

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

Returns

none

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

Draw a filled quadrilateral.

Parameters

in	*pGui*	Pointer to GUI
in	*psPt*	Pointer to array of 4 points
in	*nCol*	Color RGB value for the frame

Returns

true if success, false if error

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

Draw a filled rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to fill
in	*nCol*	Color RGB value to fill

Returns

none

5.3.4.24 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	*pGui*	Pointer to GUI
in	*nX0*	X Coordinate #1
in	*nY0*	Y Coordinate #1
in	*nX1*	X Coordinate #2

in	*nY1*	Y Coordinate #2
in	*nX2*	X Coordinate #3
in	*nY2*	Y Coordinate #3
in	*nCol*	Color RGB value for the fill

Returns

true if success, false if error

5.3.4.25 void gslc_DrawFrameCircle (gslc_tsGui ∗ *pGui,* int16_t *nMidX,* int16_t *nMidY,* uint16_t *nRadius,* gslc_tsColor *nCol*)

Draw a framed circle.

Parameters

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

Returns

none

5.3.4.26 void gslc_DrawFrameQuad (gslc_tsGui ∗ *pGui,* gslc_tsPt ∗ *psPt,* gslc_tsColor *nCol*)

Draw a framed quadrilateral.

Parameters

in	*pGui*	Pointer to GUI
in	*psPt*	Pointer to array of 4 points
in	*nCol*	Color RGB value for the frame

Returns

true if success, false if error

5.3.4.27 void gslc_DrawFrameRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a framed rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to frame
in	*nCol*	Color RGB value for the frame

Returns

none

5.3.4.28 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`	*pGui*	Pointer to GUI
`in`	*nX0*	X Coordinate #1
`in`	*nY0*	Y Coordinate #1
`in`	*nX1*	X Coordinate #2
`in`	*nY1*	Y Coordinate #2
`in`	*nX2*	X Coordinate #3
`in`	*nY2*	Y Coordinate #3
`in`	*nCol*	Color RGB value for the frame

Returns

true if success, false if error

5.3.4.29 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`	*pGui*	Pointer to GUI
`in`	*nX0*	X coordinate of line startpoint
`in`	*nY0*	Y coordinate of line startpoint
`in`	*nX1*	X coordinate of line endpoint
`in`	*nY1*	Y coordinate of line endpoint
`in`	*nCol*	Color RGB value for the line

Returns

none

5.3.4.30 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`	*pGui*	Pointer to GUI
`in`	*nX*	X coordinate of line startpoint
`in`	*nY*	Y coordinate of line startpoint
`in`	*nW*	Width of line (in +X direction)
`in`	*nCol*	Color RGB value for the line

Returns

none

5.3.4.31 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	*pGui*	Pointer to GUI
in	*nX*	X coordinate of line startpoint
in	*nY*	Y coordinate of line startpoint
in	*nRadStart*	Starting radius of line
in	*nRadEnd*	Ending radius of line
in	*n64Ang*	Angle of ray (degrees ∗ 64). 0 is up, +90∗64 is to right From -180∗64 to +180∗64
in	*nCol*	Color RGB value for the line

Returns

none

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

Returns

none

5.3.4.33 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	*pGui*	Pointer to GUI
in	*nX*	Pixel X coordinate to set
in	*nY*	Pixel Y coordinate to set
in	*nCol*	Color pixel value to assign

Returns

none

5.3.4.34 gslc_tsElem∗ gslc_ElemAdd (gslc_tsGui ∗ *pGui,* int16_t *nPageId,* gslc_tsElem ∗ *pElem,* gslc_teElemRefFlags *eFlags*)

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

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

Parameters

in	*pGui*	Pointer to GUI
in	*nPageId*	Page ID to add element to (GSLC_PAGE_NONE to skip in case of temporary creation for compound elements)
in	*pElem*	Pointer to Element to add
in	*eFlags*	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 or NULL if fail

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

Create a new element with default styling.

Parameters

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

Returns

Initialized structure

5.3.4.36 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*rElem*	Rectangle coordinates defining box size

Returns

Pointer to the Element or NULL if failure

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

Returns

Pointer to the Element or NULL if failure

5.3.4.38 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*rElem*	Rectangle coordinates defining text background size
in	*pStrBuf*	String to copy into element
in	*nStrBufMax*	Maximum length of string buffer (pStrBuf). Only applicable if GSLC_LOCAL↩_STR=0. Ignored if GSLC_LOCAL_STR=1.)
in	*nFontId*	Font ID to use for text display
in	*cbTouch*	Callback for touch events

Returns

Pointer to the Element or NULL if failure

5.3.4.39 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*rElem*	Rectangle coordinates defining box size
in	*sImgRef*	Image reference to load

Returns

Pointer to the Element or NULL if failure

5.3.4.40 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*nX0*	X coordinate of line startpoint
in	*nY0*	Y coordinate of line startpoint
in	*nX1*	X coordinate of line endpoint
in	*nY1*	Y coordinate of line endpoint

Returns

Pointer to the Element or NULL if failure

5.3.4.41 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*rElem*	Rectangle coordinates defining text background size
in	*pStrBuf*	String to copy into element
in	*nStrBufMax*	Maximum length of string buffer (pStrBuf). Only applicable if GSLC_LOCAL←_STR=0. Ignored if GSLC_LOCAL_STR=1.)

in	*nFontId*	Font ID to use for text display

Returns

Pointer to the Element or NULL if failure

5.3.4.42 void gslc_ElemDestruct (gslc_tsElem ∗ *pElem*)

Free up any members associated with an element.

Parameters

in	*pElem*	Pointer to element

Returns

none

5.3.4.43 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	*pGui*	Pointer to GUI
in	*nPageId*	ID of page containing element
in	*nElemId*	ID of element

Returns

none

5.3.4.44 bool gslc_ElemDrawByRef (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_teRedrawType *eRedraw*)

Draw an element to the active display.

- Element is referenced by an element pointer

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Ptr to Element to draw
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.3.4.45 bool gslc_ElemEvent (void ∗ *pvGui,* gslc_tsEvent *sEvent*)

Common event handler function for an element.

Parameters

`in`	*pvGui*	Void pointer to GUI
`in`	*sEvent*	Event data structure

Returns

true if success, false if fail

5.3.4.46 bool gslc_ElemGetGlow (**gslc_tsElem** ∗ *pElem*)

Get the glowing indicator for an element.

Parameters

`in`	*pElem*	Pointer to Element

Returns

True if element is glowing

5.3.4.47 bool gslc_ElemGetGlowEn (**gslc_tsElem** ∗ *pElem*)

Get the glowing enable for an element.

Parameters

`in`	*pElem*	Pointer to Element

Returns

True if element supports glowing

5.3.4.48 int gslc_ElemGetGroup (**gslc_tsElem** ∗ *pElem*)

Get the group ID for an element.

Parameters

`in`	*pElem*	Pointer to Element

Returns

Group ID or GSLC_GROUP_ID_NONE if unassigned

5.3.4.49 int gslc_ElemGetId (**gslc_tsElem** ∗ *pElem*)

Get an Element ID from an element structure.

Parameters

`in`	*pElem*	Pointer to element structure

Returns

ID of element or GSLC_ID_NONE if not found

5.3.4.50 gslc_teRedrawType gslc_ElemGetRedraw (gslc_tsElem * *pElem*)

Get the need-redraw status for an element.

Parameters

`in`	*pElem*	Pointer to Element

Returns

Redraw status

5.3.4.51 bool gslc_ElemOwnsCoord (gslc_tsElem ∗ *pElem,* 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`	*pElem*	Element used for boundary test
`in`	*nX*	X coordinate to test
`in`	*nY*	Y coordinate to test
`in`	*bOnlyClickEn*	Only output true if element was also marked as "clickable" (eg. bClickEn=true)

Returns

true if inside element, false otherwise

5.3.4.52 bool gslc_ElemSendEventTouch (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElemTracked,* 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`	*pGui*	Pointer to GUI
`in`	*pElemTracked*	Pointer to tracked Element (or NULL for none))
`in`	*eTouch*	Touch event type
`in`	*nX*	X coordinate of event (absolute coordinate)
`in`	*nY*	Y coordinate of event (absolute coordinate)

Returns

true if success, false if error

5.3.4.53 void gslc_ElemSetCol (gslc_tsElem ∗ *pElem,* gslc_tsColor *colFrame,* gslc_tsColor *colFill,* gslc_tsColor *colFillGlow*)

Update the common color selection for an Element.

Parameters

`in`	*pElem*	Pointer to Element

in	*colFrame*	Color for the frame
in	*colFill*	Color for the fill
in	*colFillGlow*	Color for the fill when glowing

Returns

none

5.3.4.54 void gslc_ElemSetDrawFunc (gslc_tsElem ∗ *pElem,* 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	*pElem*	Pointer to Element
in	*funcCb*	Function pointer to drawing routine (or NULL for default))

Returns

none

5.3.4.55 void gslc_ElemSetEventFunc (gslc_tsElem ∗ *pElem,* GSLC_CB_EVENT *funcCb*)

Assign the event callback function for a element.

Parameters

in	*pElem*	Pointer to element
in	*funcCb*	Function pointer to event routine (or NULL for default))

Returns

none

5.3.4.56 void gslc_ElemSetFillEn (gslc_tsElem ∗ *pElem,* bool *bFillEn*)

Set the fill state for an Element.

Parameters

in	*pElem*	Pointer to Element
in	*bFillEn*	True if filled, false otherwise

Returns

none

5.3.4.57 void gslc_ElemSetFrameEn (gslc_tsElem ∗ *pElem,* bool *bFrameEn*)

Set the frame state for an Element.

Parameters

in	*pElem*	Pointer to Element
in	*bFrameEn*	True if framed, false otherwise

Returns

none

5.3.4.58 void gslc_ElemSetGlow (gslc_tsElem ∗ *pElem*, bool *bGlowing*)

Update the glowing indicator for an element.

Parameters

in	*pElem*	Pointer to Element
in	*bGlowing*	True if element is glowing

Returns

none

5.3.4.59 void gslc_ElemSetGlowCol (gslc_tsElem ∗ *pElem*, gslc_tsColor *colFrameGlow*, gslc_tsColor *colFillGlow*, gslc_tsColor *colTxtGlow*)

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

Parameters

in	*pElem*	Pointer to Element
in	*colFrameGlow*	Color for the frame when glowing
in	*colFillGlow*	Color for the fill when glowing
in	*colTxtGlow*	Color for the text when glowing

Returns

none

5.3.4.60 void gslc_ElemSetGlowEn (gslc_tsElem ∗ *pElem*, bool *bGlowEn*)

Update the glowing enable for an element.

Parameters

in	*pElem*	Pointer to Element
in	*bGlowEn*	True if element should support glowing

Returns

none

5.3.4.61 void gslc_ElemSetGroup (gslc_tsElem ∗ *pElem*, int *nGroupId*)

Set the group ID for an element.

- Typically used to associate radio button elements together

Parameters

in	*pElem*	Pointer to Element
in	*nGroupId*	Group ID to assign

Returns

none

5.3.4.62 void gslc_ElemSetImage (gslc_tsGui ∗ *pGui*, gslc_tsElem ∗ *pElem*, gslc_tsImgRef *sImgRef*, gslc_tsImgRef *sImgRefSel*)

Set an element to use a bitmap image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference (normal state)
in	*sImgRefSel*	Image reference (glowing state)

Returns

none

5.3.4.63 void gslc_ElemSetRedraw (gslc_tsElem ∗ *pElem*, gslc_teRedrawType *eRedraw*)

Update the need-redraw status for an element.

Parameters

in	*pElem*	Pointer to Element
in	*eRedraw*	Redraw state to set

Returns

none

5.3.4.64 void gslc_ElemSetStyleFrom (gslc_tsElem ∗ *pElemSrc*, gslc_tsElem ∗ *pElemDest*)

Copy style settings from one element to another.

Parameters

in	*pElemSrc*	Pointer to source Element
in	*pElemDest*	Pointer to destination Element

Returns

none

5.3.4.65 void gslc_ElemSetTickFunc (gslc_tsElem ∗ *pElem*, 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	*pElem*	Pointer to Element
in	*funcCb*	Function pointer to tick routine (or NULL for none))

Returns

none

5.3.4.66 void gslc_ElemSetTxtAlign (gslc_tsElem ∗ *pElem,* unsigned *nAlign*)

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

Parameters

in	*pElem*	Pointer to Element
in	*nAlign*	Alignment to specify: • 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

5.3.4.67 void gslc_ElemSetTxtCol (gslc_tsElem ∗ *pElem,* gslc_tsColor *colVal*)

Update the text string color associated with an Element ID.

Parameters

in	*pElem*	Pointer to Element
in	*colVal*	RGB color to change to

Returns

none

5.3.4.68 void gslc_ElemSetTxtMargin (gslc_tsElem ∗ *pElem,* unsigned *nMargin*)

Set the margin around of a textual element.

Parameters

in	*pElem*	Pointer to Element
in	*nMargin*	Number of pixels gap to leave surrounding text

Returns

none

5.3.4.69 void gslc_ElemSetTxtMem (gslc_tsElem ∗ *pElem*, gslc_teTxtFlags *eFlags*)

Update the text string location in memory.

Parameters

in	*pElem*	Pointer to Element
in	*eFlags*	Flags associated with text memory location (GSLC_TXT_MEM_∗)

Returns

none

5.3.4.70 void gslc_ElemSetTxtStr (gslc_tsElem ∗ *pElem*, const char ∗ *pStr*)

Update the text string associated with an Element ID.

Parameters

in	*pElem*	Pointer to Element
in	*pStr*	String to copy into element

Returns

none

5.3.4.71 void gslc_ElemUpdateFont (gslc_tsGui ∗ *pGui*, gslc_tsElem ∗ *pElem*, int *nFontId*)

Update the Font selected for an Element's text.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element
in	*nFontId*	Font ID to select

Returns

none

5.3.4.72 gslc_tsEvent gslc_EventCreate (gslc_teEventType *eType*, uint8_t *nSubType*, void ∗ *pvScope*, void ∗ *pvData*)

Create an event structure.

Parameters

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

Returns

None

5.3.4.73 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	*rRect*	Rectangular region before resizing
in	*nExpandW*	Number of pixels to expand the width (if positive) of contract the width (if negative)
in	*nExpandH*	Number of pixels to expand the height (if positive) of contract the height (if negative)

Returns

gslc_tsRect() with resized dimensions

5.3.4.74 bool gslc_FontAdd (gslc_tsGui ∗ *pGui,* int16_t *nFontId,* const char ∗ *acFontName,* uint16_t *nFontSz*)

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

Parameters

in	*pGui*	Pointer to GUI
in	*nFontId*	ID to use when referencing this font
in	*acFontName*	Filename path to the font
in	*nFontSz*	Typeface size to use

Returns

true if load was successful, false otherwise

5.3.4.75 gslc_tsFont∗ gslc_FontGet (gslc_tsGui ∗ *pGui,* int16_t *nFontId*)

Fetch a font from its ID value.

Parameters

in	*pGui*	Pointer to GUI
in	*nFontId*	ID value used to reference the font (supplied originally to gslc_FontAdd()

Returns

A pointer to the font structure or NULL if error

5.3.4.76 gslc_tsImgRef gslc_GetImageFromFile (const char * *pFname*, gslc_teImgRefFlags *eFmt*)

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

Parameters

in	*pFname*	Pointer to filename string of image in filesystem
in	*eFmt*	Image format

Returns

Loaded image reference

5.3.4.77 gslc_tsImgRef gslc_GetImageFromProg (const unsigned char ∗ *pImgBuf*, gslc_teImgRefFlags *eFmt*)

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

Parameters

in	*pImgBuf*	Pointer to image buffer in memory
in	*eFmt*	Image format

Returns

Loaded image reference

5.3.4.78 gslc_tsImgRef gslc_GetImageFromRam (unsigned char ∗ *pImgBuf*, gslc_teImgRefFlags *eFmt*)

Create an image reference to a bitmap in SRAM.

Parameters

in	*pImgBuf*	Pointer to image buffer in memory
in	*eFmt*	Image format

Returns

Loaded image reference

5.3.4.79 gslc_tsImgRef gslc_GetImageFromSD (const char ∗ *pFname*, gslc_teImgRefFlags *eFmt*)

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

Parameters

in	*pFname*	Pointer to filename string of image in SD card
in	*eFmt*	Image format

Returns

Loaded image reference

5.3.4.80 int gslc_GetPageCur (gslc_tsGui ∗ *pGui*)

Fetch the current page ID.

Parameters

in	*pGui*	Pointer to GUI

Returns

Page ID

5.3.4.81 bool gslc_GetTouch (**gslc_tsGui** ∗ *pGui,* int16_t ∗ *pnX,* int16_t ∗ *pnY,* uint16_t ∗ *pnPress*)

Initialize the touchscreen device driver.

Parameters

in	*pGui*	Pointer to GUI
out	*pnX*	Ptr to int to contain latest touch X coordinate
out	*pnY*	Ptr to int to contain latest touch Y coordinate
out	*pnPress*	Ptr to int to contain latest touch pressure value

Returns

true if touch event, false otherwise

5.3.4.82 char∗ gslc_GetVer (**gslc_tsGui** ∗ *pGui*)

Get the GUIslice version number.

Returns

String containing version number

5.3.4.83 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	*pGui*	Pointer to GUI

Returns

none

5.3.4.84 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	*pGui*	Pointer to GUI
in	*pvDriver*	Void pointer to Driver struct (gslc_tsDriver∗)
in	*asPage*	Pointer to Page array
in	*nMaxPage*	Size of Page array
in	*asFont*	Pointer to Font array
in	*nMaxFont*	Size of Font array

Returns

true if success, false if fail

5.3.4.85 void gslc_InitDebug (GSLC_CB_DEBUG_OUT *pfunc*)

Initialize debug output.

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

Parameters

in	*pfunc*	Pointer to user character-out function

Returns

none

5.3.4.86 bool gslc_InitTouch (gslc_tsGui ∗ *pGui,* const char ∗ *acDev*)

Initialize the touchscreen device driver.

Parameters

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

Returns

true if successful

5.3.4.87 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	*nSelX*	X coordinate to test
in	*nSelY*	X coordinate to test
in	*rRect*	Rectangular region to compare against

Returns

true if inside region, false otherwise

5.3.4.88 bool gslc_IsInWH (gslc_tsGui ∗ *pGui,* 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	*pGui*	Pointer to GUI
in	*nSelX*	X coordinate to test
in	*nSelY*	X coordinate to test
in	*nWidth*	Width to test against
in	*nHeight*	Height to test against

Returns

true if inside region, false otherwise

5.3.4.89 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	*pGui*	Pointer to GUI
in	*nPageId*	Page ID to assign
in	*psElem*	Internal element array storage to associate with the page
in	*nMaxElem*	Maximum number of elements that can be added to the internal element array (ie. RAM))
in	*psElemRef*	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	*nMaxElemRef*	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

5.3.4.90 void gslc_PageDestruct (gslc_tsPage ∗ *pPage*)

Free up any members associated with a page.

Parameters

in	*pPage*	Pointer to Page

Returns

none

5.3.4.91 bool gslc_PageEvent (void ∗ *pvGui,* gslc_tsEvent *sEvent*)

Common event handler function for a page.

Parameters

in	*pvGui*	Void pointer to GUI
in	*sEvent*	Event data structure

Returns

true if success, false if fail

5.3.4.92 gslc_tsPage∗ gslc_PageFindById (gslc_tsGui ∗ *pGui,* int16_t *nPageId*)

Find a page in the GUI by its ID.

Parameters

in	*pGui*	Pointer to GUI
in	*nPageId*	Page ID to search

Returns

Ptr to a page or NULL if none found

5.3.4.93 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nPageId*	Page ID to search
in	*nElemId*	Element ID to search

Returns

Ptr to an element or NULL if none found

5.3.4.94 bool gslc_PageFlipGet (gslc_tsGui ∗ *pGui*)

Get state of pending page flip state.

Parameters

in	*pGui*	Pointer to GUI

Returns

True if screen requires page flip

5.3.4.95 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	*pGui*	Pointer to GUI

Returns

None

5.3.4.96 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	*pGui*	Pointer to GUI
in	*bNeeded*	True if screen requires page flip

Returns

None

5.3.4.97 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	*pGui*	Pointer to GUI

Returns

none

5.3.4.98 bool gslc_PageRedrawGet (gslc_tsGui ∗ *pGui*)

Get the need-redraw status for the current page.

Parameters

in	*pGui*	Pointer to GUI

Returns

True if redraw required, false otherwise

5.3.4.99 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	*pGui*	Pointer to GUI

Returns

none

5.3.4.100 void gslc_PageRedrawSet (gslc_tsGui ∗ *pGui,* bool *bRedraw*)

Update the need-redraw status for the current page.

Parameters

in	*pGui*	Pointer to GUI
in	*bRedraw*	True if redraw required, false otherwise

Returns

none

5.3.4.101 void gslc_PageSetEventFunc (gslc_tsPage ∗ *pPage,* GSLC_CB_EVENT *funcCb*)

Assign the event callback function for a page.

Parameters

`in`	*pPage*	Pointer to page
`in`	*funcCb*	Function pointer to event routine (or NULL for default))

Returns

none

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

Convert polar coordinate to cartesian.

Parameters

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

Returns

none

5.3.4.103 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`	*pGui*	Pointer to GUI

Returns

None

5.3.4.104 void gslc_ResetElem (gslc_tsElem ∗ *pElem*)

Initialize an Element struct.

Parameters

`in`	*pElem*	Pointer to Element

Returns

none

5.3.4.105 void gslc_ResetFont (gslc_tsFont ∗ *pFont*)

Initialize a Font struct.

Parameters

`in`	*pFont*	Pointer to Font

Returns

none

5.3.4.106 gslc_tsImgRef gslc_ResetImage ( )

Create a blank image reference structure.

Returns

Image reference struct

5.3.4.107 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`	*pGui*	Pointer to GUI
`in`	*nCol*	RGB Color to use

Returns

true if success, false if fail

5.3.4.108 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`	*pGui*	Pointer to GUI
`in`	*sImgRef*	Image reference

Returns

true if success, false if fail

5.3.4.109 bool gslc_SetClipRect (gslc_tsGui ∗ *pGui,* gslc_tsRect ∗ *pRect*)

Set the clipping rectangle for further drawing.

Parameters

in	*pGui*	Pointer to GUI
in	*pRect*	Pointer to Rect for clipping (or NULL for entire screen)

Returns

true if success, false if error

5.3.4.110 void gslc_SetPageCur (gslc_tsGui ∗ *pGui*, int16_t *nPageId*)

Select a new page for display.

Parameters

in	*pGui*	Pointer to GUI
in	*nPageId*	Page ID to select as current

Returns

none

5.3.4.111 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.
- gslc_sinFX(nAngDeg∗64)/32768.0 = sin(nAngDeg∗2pi/360)

Parameters

in	*n64Ang*	Angle (in units of 1/64 degrees)

Returns

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

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

Returns

none

5.3.4.113 void gslc_Update (gslc_tsGui ∗ *pGui*)

Perform main GUIslice handling functions.

- Handles any touch events
- Performs any necessary screen redraw

Parameters

in	*pGui*	Pointer to GUI

Returns

None

5.3.5 Variable Documentation

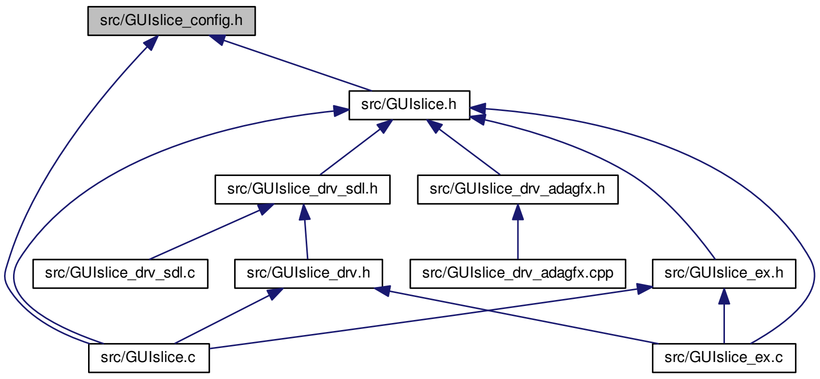
5.3.5.1 GSLC_CB_DEBUG_OUT g_pfDebugOut

Global debug output function.

- The user assigns this function via gslc_InitDebug()

5.4 src/GUIslice_config.h File Reference

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



Macros

- #define DRV_DISP_SDL1
- #define DRV_TOUCH_TSLIB
- #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 DEBUG_ERR 1
- #define ADATOUCH_SWAP_XY 1
- #define ADATOUCH_FLIP_X 0
- #define ADATOUCH_FLIP_Y 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

5.4.1 Macro Definition Documentation

5.4.1.1 #define ADATOUCH_FLIP_X 0

5.4.1.2 #define ADATOUCH_FLIP_Y 1

5.4.1.3 #define ADATOUCH_SWAP_XY 1

5.4.1.4 #define DEBUG_ERR 1

5.4.1.5 #define DRV_DISP_SDL1

5.4.1.6 #define DRV_SDL_FIX_START 1

5.4.1.7 #define DRV_SDL_MOUSE_SHOW 0

5.4.1.8 #define DRV_TOUCH_TSLIB

5.4.1.9 #define GSLC_BMP_TRANS_EN 1

5.4.1.10 #define GSLC_BMP_TRANS_RGB 0xFF,0x00,0xFF

5.4.1.11 #define GSLC_DEV_FB "/dev/fb1"

5.4.1.12 #define GSLC_DEV_TOUCH "/dev/input/touchscreen"

5.4.1.13 #define GSLC_DEV_VID_DRV "fbcon"

5.4.1.14 #define GSLC_LOCAL_STR 1

5.4.1.15 #define GSLC_LOCAL_STR_LEN 30

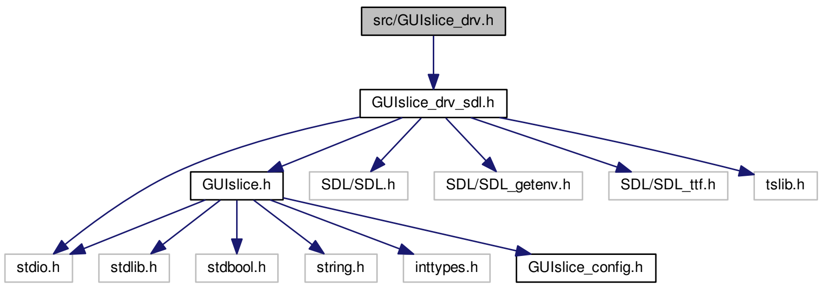
5.4.1.16 #define GSLC_USE_FLOAT 1

5.4.1.17 #define GSLC_USE_PROGMEM 0

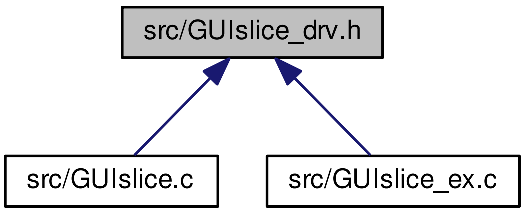
5.5 src/GUIslice_drv.h File Reference

```
#include "GUIslice_drv_sdl.h"
```

Include dependency graph for GUIslice_drv.h:



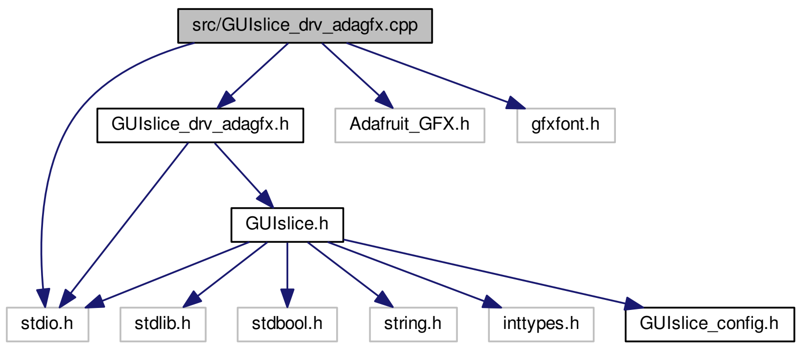
This graph shows which files directly or indirectly include this file:



5.6 src/GUIslice_drv_adagfx.cpp File Reference

```
#include "GUIslice_drv_adagfx.h"
#include <stdio.h>
#include <Adafruit_GFX.h>
#include <gfxfont.h>
```

Include dependency graph for GUIslice_drv_adagfx.cpp:



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.
- void ∗ gslc_DrvFontAdd (const char ∗acFontName, uint16_t nFontSz)

    Load a font from a file and return pointer to it.
- void gslc_DrvFontsDestruct (gslc_tsGui ∗pGui)

    Release all fonts defined in the GUI.
- bool gslc_DrvGetTxtSize (gslc_tsGui ∗pGui, gslc_tsFont ∗pFont, const char ∗pStr, gslc_teTxtFlags eTxt←Flags, 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_DrvDrawFillRect (gslc_tsGui ∗pGui, gslc_tsRect rRect, gslc_tsColor nCol)

  Draw a filled rectangle.
- bool gslc_DrvDrawFrameRect (gslc_tsGui ∗pGui, gslc_tsRect rRect, gslc_tsColor nCol)

  Draw a framed 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 ∗, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)

  Draw a framed circle.
- bool gslc_DrvDrawFillCircle (gslc_tsGui ∗, 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.
- void gslc_DrvDrawMonoFromMem (gslc_tsGui ∗pGui, int16_t x, int16_t y, const unsigned char ∗bitmap, bool bProgMem)
- 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_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 handler.
- uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)

5.6.1 Function Documentation

5.6.1.1 uint16_t gslc_DrvAdaptColorToRaw (**gslc_tsColor** *nCol*)

5.6.1.2 void gslc_DrvDestruct (**gslc_tsGui** ∗ *pGui*)

Free up any members associated with the driver.

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

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.6.1.3 void gslc_DrvDrawBkgnd (**gslc_tsGui** ∗ *pGui*)

Copy the background image to destination screen.

Parameters

in	*pGui*	Pointer to GUI

Returns

true if success, false if fail

5.6.1.4 bool gslc_DrvDrawFillCircle (gslc_tsGui ∗ *pGui,* int16_t *nMidX,* int16_t *nMidY,* uint16_t *nRadius,* gslc_tsColor *nCol*)

Draw a filled circle.

Parameters

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

Returns

true if success, false if error

5.6.1.5 bool gslc_DrvDrawFillRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a filled rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to fill
in	*nCol*	Color RGB value to fill

Returns

true if success, false if error

5.6.1.6 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	*pGui*	Pointer to GUI
in	*nX0*	X Coordinate #1
in	*nY0*	Y Coordinate #1
in	*nX1*	X Coordinate #2
in	*nY1*	Y Coordinate #2

in	*nX2*	X Coordinate #3
in	*nY2*	Y Coordinate #3
in	*nCol*	Color RGB value to fill

Returns

true if success, false if error

5.6.1.7 bool gslc_DrvDrawFrameCircle (gslc_tsGui ∗ *pGui,* int16_t *nMidX,* int16_t *nMidY,* uint16_t *nRadius,* gslc_tsColor *nCol*)

Draw a framed circle.

Parameters

in	*pGui*	Pointer to GUI
in	*nMidX*	Center of circle (X coordinate)
in	*nMidY*	Center of circle (Y coordinate)
in	*nRadius*	Radius of circle
in	*nCol*	Color RGB value to frame

Returns

true if success, false if error

5.6.1.8 bool gslc_DrvDrawFrameRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a framed rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to frame
in	*nCol*	Color RGB value to frame

Returns

true if success, false if error

5.6.1.9 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	*pGui*	Pointer to GUI
in	*nX0*	X Coordinate #1
in	*nY0*	Y Coordinate #1
in	*nX1*	X Coordinate #2
in	*nY1*	Y Coordinate #2

in	*nX2*	X Coordinate #3
in	*nY2*	Y Coordinate #3
in	*nCol*	Color RGB value to frame

Returns

true if success, false if error

5.6.1.10 bool gslc_DrvDrawImage (gslc_tsGui ∗ *pGui,* int16_t *nDstX,* int16_t *nDstY,* gslc_tsImgRef *sImgRef*)

Copy all of source image to destination screen at specified coordinate.

Parameters

in	*pGui*	Pointer to GUI
in	*nDstX*	Destination X coord for copy
in	*nDstY*	Destination Y coord for copy
in	*sImgRef*	Image reference

Returns

true if success, false if fail

5.6.1.11 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	*pGui*	Pointer to GUI
in	*nX0*	Line start (X coordinate)
in	*nY0*	Line start (Y coordinate)
in	*nX1*	Line finish (X coordinate)
in	*nY1*	Line finish (Y coordinate)
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.6.1.12 void gslc_DrvDrawMonoFromMem (gslc_tsGui ∗ *pGui,* int16_t *x,* int16_t *y,* const unsigned char ∗ *bitmap,* bool *bProgMem*)

5.6.1.13 bool gslc_DrvDrawPoint (gslc_tsGui ∗ *pGui,* int16_t *nX,* int16_t *nY,* gslc_tsColor *nCol*)

Draw a point.

Parameters

in	*pGui*	Pointer to GUI

in	*nX*	X coordinate of point
in	*nY*	Y coordinate of point
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.6.1.14 bool gslc_DrvDrawPoints (gslc_tsGui ∗ *pGui,* gslc_tsPt ∗ *asPt,* uint16_t *nNumPt,* gslc_tsColor *nCol*)

Draw a point.

Parameters

in	*pGui*	Pointer to GUI
in	*asPt*	Array of points to draw
in	*nNumPt*	Number of points in array
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.6.1.15 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	*pGui*	Pointer to GUI
in	*nTxtX*	X coordinate of top-left text string
in	*nTxtY*	Y coordinate of top-left text string
in	*pFont*	Ptr to Font
in	*pStr*	String to display
in	*eTxtFlags*	Flags associated with text string
in	*colTxt*	Color to draw text

Returns

true if success, false if failure

5.6.1.16 void∗ gslc_DrvFontAdd (const char ∗ *acFontName,* uint16_t *nFontSz*)

Load a font from a file and return pointer to it.

Parameters

in	*acFontName*	Filename path to the font
in	*nFontSz*	Typeface size to use

Returns

true if load was successful, false otherwise

5.6.1.17 void gslc_DrvFontsDestruct (gslc_tsGui * *pGui*)

Release all fonts defined in the GUI.

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.6.1.18 bool gslc_DrvGetTouch (gslc_tsGui ∗ *pGui,* int16_t ∗ *pnX,* int16_t ∗ *pnY,* uint16_t ∗ *pnPress*)

Get the last touch event from the SDL handler.

Get the last touch event from the SDL_Event handler.

Parameters

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

Returns

true if an event was detected or 0 otherwise

5.6.1.19 bool gslc_DrvGetTxtSize (gslc_tsGui ∗ *pGui,* gslc_tsFont ∗ *pFont,* const char ∗ *pStr,* gslc_teTxtFlags *eTxtFlags,* uint16_t ∗ *pnTxtSzW,* uint16_t ∗ *pnTxtSzH*)

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

Parameters

in	*pGui*	Pointer to GUI
in	*pFont*	Ptr to Font structure
in	*pStr*	String to display
in	*eTxtFlags*	Flags associated with text string
out	*pnTxtSzW*	Ptr to width of text
out	*pnTxtSzH*	Ptr to height of text

Returns

true if success, false if failure

5.6.1.20 void gslc_DrvImageDestruct (void ∗ *pvImg*)

Release an image surface.

Parameters

in	*pvImg*	Void ptr to image

Returns

none

5.6.1.21 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	*pGui*	Pointer to GUI

Returns

true if success, false if fail

5.6.1.22 bool gslc_DrvInitTouch (gslc_tsGui ∗ *pGui,* const char ∗ *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	*pGui*	Pointer to GUI
in	*acDev*	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

5.6.1.23 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	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

Image pointer (surface/texture) or NULL if error

5.6.1.24 void gslc_DrvPageFlipNow (gslc_tsGui ∗ *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.6.1.25 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	*pGui*	Pointer to GUI
in	*nCol*	RGB Color to use

Returns

true if success, false if fail

5.6.1.26 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	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

true if success, false if fail

5.6.1.27 bool gslc_DrvSetClipRect (gslc_tsGui ∗ *pGui*, gslc_tsRect ∗ *pRect*)

Set the clipping rectangle for future drawing updates.

Parameters

in	*pGui*	Pointer to GUI
in	*pRect*	Rectangular region to constrain edits

Returns

none

5.6.1.28 bool gslc_DrvSetElemImageGlow (gslc_tsGui ∗ *pGui*, gslc_tsElem ∗ *pElem*, gslc_tsImgRef *sImgRef*)

Set an element's glow-state image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference

Returns

true if success, false if error

5.6.1.29 bool gslc_DrvSetElemImageNorm (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_tsImgRef *sImgRef*)

Set an element's normal-state image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference

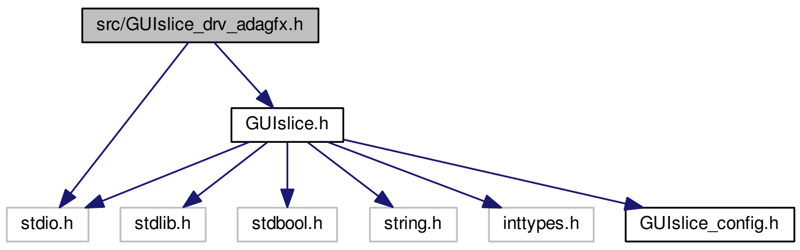
Returns

true if success, false if error

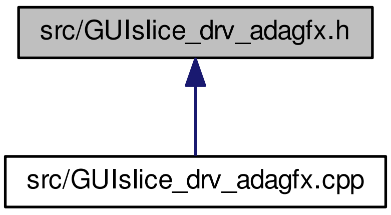
5.7 src/GUIslice_drv_adagfx.h File Reference

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

Include dependency graph for GUIslice_drv_adagfx.h:



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



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()

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.
- void ∗ gslc_DrvFontAdd (const char ∗acFontName, uint16_t nFontSz)

  Load a font from a file and return pointer to it.
- void gslc_DrvFontsDestruct (gslc_tsGui ∗pGui)

  Release all fonts defined in the GUI.
- bool gslc_DrvGetTxtSize (gslc_tsGui ∗pGui, gslc_tsFont ∗pFont, const char ∗pStr, gslc_teTxtFlags eTxt↩ Flags, 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_ts↩ Color nCol)

  Draw a framed circle.
- bool gslc_DrvDrawFillCircle (gslc_tsGui ∗pGui, int16_t nMidX, int16_t nMidY, uint16_t nRadius, gslc_tsColor nCol)

  Draw a filled circle.
- bool gslc_DrvDrawFrameTriangle (gslc_tsGui ∗pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)

  Draw a framed triangle.
- bool gslc_DrvDrawFillTriangle (gslc_tsGui ∗pGui, int16_t nX0, int16_t nY0, int16_t nX1, int16_t nY1, int16_t nX2, int16_t nY2, gslc_tsColor nCol)

  Draw a filled triangle.
- bool gslc_DrvDrawImage (gslc_tsGui ∗pGui, int16_t nDstX, int16_t nDstY, gslc_tsImgRef sImgRef)

  Copy all of source image to destination screen at specified coordinate.
- void gslc_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.
- uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor nCol)

5.7.1 Macro Definition Documentation

5.7.1.1 #define DRV_HAS_DRAW_CIRCLE_FILL 1

Support gslc_DrvDrawFillCircle()

5.7.1.2 #define DRV_HAS_DRAW_CIRCLE_FRAME 1

Support gslc_DrvDrawFrameCircle()

5.7.1.3 #define DRV_HAS_DRAW_LINE 1

Support gslc_DrvDrawLine()

5.7.1.4 #define DRV_HAS_DRAW_POINT 1

Support gslc_DrvDrawPoint()

5.7.1.5 #define DRV_HAS_DRAW_POINTS 0

Support gslc_DrvDrawPoints()

5.7.1.6 #define DRV_HAS_DRAW_RECT_FILL 1

Support gslc_DrvDrawFillRect()

5.7.1.7 #define DRV_HAS_DRAW_RECT_FRAME 1

Support gslc_DrvDrawFrameRect()

5.7.1.8 #define DRV_HAS_DRAW_TEXT 1

Support gslc_DrvDrawTxt()

5.7.1.9 #define DRV_HAS_DRAW_TRI_FILL 1

Support gslc_DrvDrawFillTriangle()

5.7.1.10 #define DRV_HAS_DRAW_TRI_FRAME 1

Support gslc_DrvDrawFrameTriangle()

5.7.2 Function Documentation

5.7.2.1 uint16_t gslc_DrvAdaptColorToRaw (gslc_tsColor *nCol*)

5.7.2.2 void gslc_DrvDestruct (gslc_tsGui ∗ *pGui*)

Free up any members associated with the driver.

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

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.7.2.3 void gslc_DrvDrawBkgnd (gslc_tsGui ∗ *pGui*)

Copy the background image to destination screen.

Parameters

in	*pGui*	Pointer to GUI

Returns

true if success, false if fail

Copy the background image to destination screen.

5.7.2.4 bool gslc_DrvDrawFillCircle (gslc_tsGui ∗ *pGui,* int16_t *nMidX,* int16_t *nMidY,* uint16_t *nRadius,* gslc_tsColor *nCol*)

Draw a filled circle.

Parameters

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

Returns

true if success, false if error

5.7.2.5 bool gslc_DrvDrawFillRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a filled rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to fill
in	*nCol*	Color RGB value to fill

Returns

true if success, false if error

5.7.2.6 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	*pGui*	Pointer to GUI
in	*nX0*	X Coordinate #1
in	*nY0*	Y Coordinate #1
in	*nX1*	X Coordinate #2
in	*nY1*	Y Coordinate #2
in	*nX2*	X Coordinate #3
in	*nY2*	Y Coordinate #3
in	*nCol*	Color RGB value to fill

Returns

true if success, false if error

5.7.2.7 bool gslc_DrvDrawFrameCircle (gslc_tsGui ∗ *pGui,* int16_t *nMidX,* int16_t *nMidY,* uint16_t *nRadius,* gslc_tsColor *nCol*)

Draw a framed circle.

Parameters

in	*pGui*	Pointer to GUI
in	*nMidX*	Center of circle (X coordinate)
in	*nMidY*	Center of circle (Y coordinate)
in	*nRadius*	Radius of circle
in	*nCol*	Color RGB value to frame

Returns

true if success, false if error

5.7.2.8 bool gslc_DrvDrawFrameRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a framed rectangle.

Parameters

`in`	*pGui*	Pointer to GUI
`in`	*rRect*	Rectangular region to frame
`in`	*nCol*	Color RGB value to frame

Returns

true if success, false if error

5.7.2.9 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`	*pGui*	Pointer to GUI
`in`	*nX0*	X Coordinate #1
`in`	*nY0*	Y Coordinate #1
`in`	*nX1*	X Coordinate #2
`in`	*nY1*	Y Coordinate #2
`in`	*nX2*	X Coordinate #3
`in`	*nY2*	Y Coordinate #3
`in`	*nCol*	Color RGB value to frame

Returns

true if success, false if error

5.7.2.10 bool gslc_DrvDrawImage (gslc_tsGui ∗ *pGui,* int16_t *nDstX,* int16_t *nDstY,* gslc_tsImgRef *sImgRef*)

Copy all of source image to destination screen at specified coordinate.

Parameters

`in`	*pGui*	Pointer to GUI
`in`	*nDstX*	Destination X coord for copy
`in`	*nDstY*	Destination Y coord for copy
`in`	*sImgRef*	Image reference

Returns

true if success, false if fail

5.7.2.11 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	*pGui*	Pointer to GUI
in	*nX0*	Line start (X coordinate)
in	*nY0*	Line start (Y coordinate)
in	*nX1*	Line finish (X coordinate)
in	*nY1*	Line finish (Y coordinate)
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.7.2.12 bool gslc_DrvDrawPoint (gslc_tsGui ∗ *pGui*, int16_t *nX*, int16_t *nY*, gslc_tsColor *nCol*)

Draw a point.

Parameters

in	*pGui*	Pointer to GUI
in	*nX*	X coordinate of point
in	*nY*	Y coordinate of point
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.7.2.13 bool gslc_DrvDrawPoints (gslc_tsGui ∗ *pGui*, gslc_tsPt ∗ *asPt*, uint16_t *nNumPt*, gslc_tsColor *nCol*)

Draw a point.

Parameters

in	*pGui*	Pointer to GUI
in	*asPt*	Array of points to draw
in	*nNumPt*	Number of points in array
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.7.2.14 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	*pGui*	Pointer to GUI
in	*nTxtX*	X coordinate of top-left text string
in	*nTxtY*	Y coordinate of top-left text string

in	*pFont*	Ptr to Font
in	*pStr*	String to display
in	*eTxtFlags*	Flags associated with text string
in	*colTxt*	Color to draw text

Returns

true if success, false if failure

5.7.2.15 void∗ gslc_DrvFontAdd (const char ∗ *acFontName,* uint16_t *nFontSz*)

Load a font from a file and return pointer to it.

Parameters

in	*acFontName*	Filename path to the font
in	*nFontSz*	Typeface size to use

Returns

true if load was successful, false otherwise

5.7.2.16 void gslc_DrvFontsDestruct (gslc_tsGui ∗ *pGui*)

Release all fonts defined in the GUI.

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.7.2.17 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	*pGui*	Pointer to GUI
out	*pnX*	Ptr to X coordinate of last touch event
out	*pnY*	Ptr to Y coordinate of last touch event
out	*pnPress*	Ptr to Pressure level of last touch event (0 for none, 1 for touch)

Returns

true if an event was detected or false otherwise

Get the last touch event from the SDL_Event handler.

Parameters

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

Returns

true if an event was detected or 0 otherwise

5.7.2.18 bool gslc_DrvGetTxtSize (gslc_tsGui ∗ *pGui,* gslc_tsFont ∗ *pFont,* const char ∗ *pStr,* gslc_teTxtFlags *eTxtFlags,* uint16_t ∗ *pnTxtSzW,* uint16_t ∗ *pnTxtSzH*)

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

Parameters

in	*pGui*	Pointer to GUI
in	*pFont*	Ptr to Font structure
in	*pStr*	String to display
in	*eTxtFlags*	Flags associated with text string
out	*pnTxtSzW*	Ptr to width of text
out	*pnTxtSzH*	Ptr to height of text

Returns

true if success, false if failure

5.7.2.19 void gslc_DrvImageDestruct (void ∗ *pvImg*)

Release an image surface.

Parameters

in	*pvImg*	Void ptr to image

Returns

none

5.7.2.20 bool gslc_DrvInit (gslc_tsGui ∗ *pGui*)

Initialize the SDL library.

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

PRE:

- The environment variables should be configured before calling gslc_DrvInit(). This can be done with gslc_↩
DrvInitEnv() or manually in user function.

Parameters

in	*pGui*	Pointer to GUI

Returns

true if success, false if fail

5.7.2.21 bool gslc_DrvInitTouch (gslc_tsGui ∗ *pGui,* const char ∗ *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	*pGui*	Pointer to GUI
in	*acDev*	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

5.7.2.22 bool gslc_DrvInitTs (gslc_tsGui ∗ *pGui,* const char ∗ *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	*pGui*	Pointer to GUI
in	*acDev*	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

5.7.2.23 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	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

Image pointer (surface/texture) or NULL if error

5.7.2.24 void gslc_DrvPageFlipNow (gslc_tsGui ∗ *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.7.2.25 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	*pGui*	Pointer to GUI
in	*nCol*	RGB Color to use

Returns

true if success, false if fail

5.7.2.26 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	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

true if success, false if fail

5.7.2.27 bool gslc_DrvSetClipRect (gslc_tsGui ∗ *pGui,* gslc_tsRect ∗ *pRect*)

Set the clipping rectangle for future drawing updates.

Parameters

in	*pGui*	Pointer to GUI
in	*pRect*	Rectangular region to constrain edits

Returns

none

5.7.2.28 bool gslc_DrvSetElemImageGlow (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_tsImgRef *sImgRef*)

Set an element's glow-state image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference

Returns

true if success, false if error

5.7.2.29 bool gslc_DrvSetElemImageNorm (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_tsImgRef *sImgRef*)

Set an element's normal-state image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference

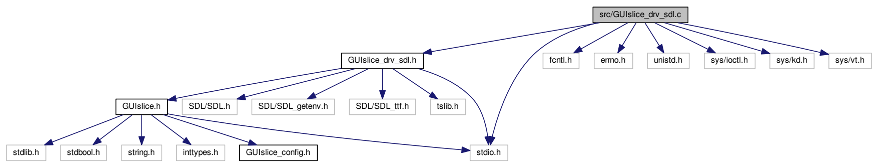
Returns

true if success, false if error

5.8 src/GUIslice_drv_sdl.c File Reference

```
#include "GUIslice_drv_sdl.h"
#include <stdio.h>
#include <fcntl.h>
#include <errno.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <sys/kd.h>
#include <sys/vt.h>
```

Include dependency graph for GUIslice_drv_sdl.c:



Macros

- #define DRV_SDL_FIX_TTY "/dev/tty0"

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.
- void ∗ gslc_DrvFontAdd (const char ∗acFontName, uint16_t nFontSz)

  Load a font from a file and return pointer to it.
- void gslc_DrvFontsDestruct (gslc_tsGui ∗pGui)

  Release all fonts defined in the GUI.
- bool gslc_DrvGetTxtSize (gslc_tsGui ∗pGui, gslc_tsFont ∗pFont, const char ∗pStr, gslc_teTxtFlags eTxt↩ Flags, 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_DrvDrawFillRect (gslc_tsGui ∗pGui, gslc_tsRect rRect, gslc_tsColor nCol)

  Draw a filled rectangle.
- bool gslc_DrvDrawFrameRect (gslc_tsGui ∗pGui, gslc_tsRect rRect, gslc_tsColor nCol)

  Draw a framed 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)

  NOTE: Background image is stored in pGui->sImgRefBkgnd.
- 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_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.
- uint32_t gslc_DrvAdaptColorRaw (gslc_tsGui ∗pGui, gslc_tsColor nCol)

  Convert an RGB color triplet into the surface pixel value.
- bool gslc_DrvScreenLock (gslc_tsGui ∗pGui)

  Lock an SDL surface so that direct pixel manipulation can be done safely.
- void gslc_DrvScreenUnlock (gslc_tsGui ∗pGui)

  Unlock the SDL surface after pixel manipulation is complete.
- uint32_t gslc_DrvDrawGetPixelRaw (gslc_tsGui ∗pGui, int16_t nX, int16_t nY)

  Get the pixel at (X,Y) from the active screen.
- void gslc_DrvDrawSetPixelRaw (gslc_tsGui ∗pGui, int16_t nX, int16_t nY, uint32_t nPixelVal)

  Set a pixel on the active screen to the given color.
- void gslc_DrvPasteSurface (gslc_tsGui ∗pGui, int16_t nX, int16_t nY, void ∗pvSrc, void ∗pvDest)

  Copy one image region to another.
- bool gslc_TDrvInitTouch (gslc_tsGui ∗pGui, const char ∗acDev)

  Perform any touchscreen-specific initialization.
- int gslc_TDrvGetTouch (gslc_tsGui ∗pGui, int16_t ∗pnX, int16_t ∗pnY, uint16_t ∗pnPress)

  Get the last touch event from the tslib handler.

5.8.1 Macro Definition Documentation

5.8.1.1 #define DRV_SDL_FIX_TTY "/dev/tty0"

5.8.2 Function Documentation

5.8.2.1 SDL_Color gslc_DrvAdaptColor (gslc_tsColor *sCol*)

Translate a gslc_tsColor into an SDL_Color.

Parameters

in	*sCol*	gslc_tsColor

Returns

Converted SDL_Color

5.8.2.2 uint32_t gslc_DrvAdaptColorRaw (gslc_tsGui ∗ *pGui,* gslc_tsColor *nCol*)

Convert an RGB color triplet into the surface pixel value.

This is called to produce the native pixel value required by the raw pixel manipulation routines.

Parameters

in	*pGui*	Pointer to GUI
in	*nCol*	RGB value for conversion

Returns

A pixel value for the current screen format

5.8.2.3 SDL_Rect gslc_DrvAdaptRect (gslc_tsRect *rRect*)

Translate a gslc_tsRect into an SDL_Rect.

Parameters

in	*rRect*	gslc_tsRect

Returns

Converted SDL_Rect

5.8.2.4 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	*sTTY*	Terminal device (eg. "/dev/tty0")

Returns

true if success

5.8.2.5 void gslc_DrvDestruct (gslc_tsGui ∗ *pGui*)

Free up any members associated with the driver.

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

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.8.2.6 void gslc_DrvDrawBkgnd (gslc_tsGui ∗ *pGui*)

NOTE: Background image is stored in pGui->sImgRefBkgnd.

Copy the background image to destination screen.

5.8.2.7 bool gslc_DrvDrawFillRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a filled rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to fill
in	*nCol*	Color RGB value to fill

Returns

true if success, false if error

5.8.2.8 bool gslc_DrvDrawFrameRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a framed rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to frame
in	*nCol*	Color RGB value to frame

Returns

true if success, false if error

5.8.2.9 uint32_t gslc_DrvDrawGetPixelRaw (gslc_tsGui ∗ *pGui*, int16_t *nX*, int16_t *nY*)

Get the pixel at (X,Y) from the active screen.

PRE:

- Screen surface must be locked

Parameters

in	*pGui*	Pointer to GUI
in	*nX*	Pixel X coordinate
in	*nY*	Pixel Y coordinate

Returns

Pixel color value from the coordinate or 0 if error

5.8.2.10 bool gslc_DrvDrawImage (gslc_tsGui ∗ *pGui*, int16_t *nDstX*, int16_t *nDstY*, gslc_tsImgRef *sImgRef*)

Copy all of source image to destination screen at specified coordinate.

Parameters

in	*pGui*	Pointer to GUI
in	*nDstX*	Destination X coord for copy
in	*nDstY*	Destination Y coord for copy
in	*sImgRef*	Image reference

Returns

true if success, false if fail

5.8.2.11 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	*pGui*	Pointer to GUI
in	*nX0*	Line start (X coordinate)
in	*nY0*	Line start (Y coordinate)
in	*nX1*	Line finish (X coordinate)
in	*nY1*	Line finish (Y coordinate)
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.8.2.12 bool gslc_DrvDrawPoint (**gslc_tsGui** ∗ *pGui,* int16_t *nX,* int16_t *nY,* **gslc_tsColor** *nCol*)

Draw a point.

Parameters

in	*pGui*	Pointer to GUI
in	*nX*	X coordinate of point
in	*nY*	Y coordinate of point
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.8.2.13 bool gslc_DrvDrawPoints (**gslc_tsGui** ∗ *pGui,* **gslc_tsPt** ∗ *asPt,* uint16_t *nNumPt,* **gslc_tsColor** *nCol*)

Draw a point.

Parameters

in	*pGui*	Pointer to GUI
in	*asPt*	Array of points to draw
in	*nNumPt*	Number of points in array
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.8.2.14 void gslc_DrvDrawSetPixelRaw (**gslc_tsGui** ∗ *pGui,* int16_t *nX,* int16_t *nY,* uint32_t *nPixelCol*)

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

PRE:

- Screen surface must be locked

Parameters

in	*pGui*	Pointer to GUI
in	*nX*	Pixel X coordinate to set
in	*nY*	Pixel Y coordinate to set
in	*nPixelCol*	Raw color pixel value to assign

Returns

none

5.8.2.15 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	*pGui*	Pointer to GUI
in	*nTxtX*	X coordinate of top-left text string
in	*nTxtY*	Y coordinate of top-left text string
in	*pFont*	Ptr to Font
in	*pStr*	String to display
in	*eTxtFlags*	Flags associated with text string
in	*colTxt*	Color to draw text

Returns

true if success, false if failure

5.8.2.16 void∗ gslc_DrvFontAdd (const char ∗ *acFontName*, uint16_t *nFontSz*)

Load a font from a file and return pointer to it.

Parameters

in	*acFontName*	Filename path to the font
in	*nFontSz*	Typeface size to use

Returns

true if load was successful, false otherwise

5.8.2.17 void gslc_DrvFontsDestruct (gslc_tsGui ∗ *pGui*)

Release all fonts defined in the GUI.

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.8.2.18 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.

Parameters

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

Returns

true if an event was detected or false otherwise

Get the last touch event from the SDL_Event handler.

Parameters

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

Returns

true if an event was detected or 0 otherwise

5.8.2.19 bool gslc_DrvGetTxtSize (gslc_tsGui ∗ *pGui,* gslc_tsFont ∗ *pFont,* const char ∗ *pStr,* gslc_teTxtFlags *eTxtFlags,* uint16_t ∗ *pnTxtSzW,* uint16_t ∗ *pnTxtSzH*)

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

Parameters

`in`	*pGui*	Pointer to GUI
`in`	*pFont*	Ptr to Font structure
`in`	*pStr*	String to display
`in`	*eTxtFlags*	Flags associated with text string
`out`	*pnTxtSzW*	Ptr to width of text
`out`	*pnTxtSzH*	Ptr to height of text

Returns

true if success, false if failure

5.8.2.20 void gslc_DrvImageDestruct (void ∗ *pvImg*)

Release an image surface.

Parameters

`in`	*pvImg*	Void ptr to image

Returns

none

5.8.2.21 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	*pGui*	Pointer to GUI

Returns

true if success, false if fail

5.8.2.22 bool gslc_DrvInitTouch (gslc_tsGui ∗ *pGui,* const char ∗ *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	*pGui*	Pointer to GUI
in	*acDev*	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

5.8.2.23 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	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

Image pointer (surface/texture) or NULL if error

5.8.2.24 void gslc_DrvPageFlipNow (gslc_tsGui ∗ *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.8.2.25 void gslc_DrvPasteSurface (gslc_tsGui ∗ *pGui,* int16_t *nX,* int16_t *nY,* void ∗ *pvSrc,* void ∗ *pvDest*)

Copy one image region to another.

- This is typically used to copy an image to the main screen surface

Parameters

in	*pGui*	Pointer to GUI
in	*nX*	Destination X coordinate of copy
in	*nY*	Destination Y coordinate of copy
in	*pvSrc*	Void Ptr to source surface (eg. a loaded image)
in	*pvDest*	Void Ptr to destination surface (typically the screen)

Returns

none

5.8.2.26 void gslc_DrvReportInfoPost ( )

Report driver debug info (after initialization)

Returns

none

5.8.2.27 void gslc_DrvReportInfoPre ( )

Report driver debug info (before initialization)

Returns

none

5.8.2.28 bool gslc_DrvScreenLock (gslc_tsGui $*$ *pGui*)

Lock an SDL surface so that direct pixel manipulation can be done safely.

This function is called before any direct pixel updates.

POST:

- Primary screen surface is locked

Parameters

in	*pGui*	Pointer to GUI

Returns

true if success, false otherwise

5.8.2.29 void gslc_DrvScreenUnlock (gslc_tsGui $*$ *pGui*)

Unlock the SDL surface after pixel manipulation is complete.

This function is called after all pixel updates are done.

POST:

- Primary screen surface is unlocked

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.8.2.30 bool gslc_DrvSetBkgndColor (gslc_tsGui ∗ *pGui*, gslc_tsColor *nCol*)

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

in	*pGui*	Pointer to GUI
in	*nCol*	RGB Color to use

Returns

true if success, false if fail

5.8.2.31 bool gslc_DrvSetBkgndImage (gslc_tsGui ∗ *pGui*, gslc_tsImgRef *sImgRef*)

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

in	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

true if success, false if fail

5.8.2.32 bool gslc_DrvSetClipRect (gslc_tsGui ∗ *pGui*, gslc_tsRect ∗ *pRect*)

Set the clipping rectangle for future drawing updates.

Parameters

in	*pGui*	Pointer to GUI
in	*pRect*	Rectangular region to constrain edits

Returns

none

5.8.2.33 bool gslc_DrvSetElemImageGlow (gslc_tsGui ∗ *pGui*, gslc_tsElem ∗ *pElem*, gslc_tsImgRef *sImgRef*)

Set an element's glow-state image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference

Returns

true if success, false if error

5.8.2.34 bool gslc_DrvSetElemImageNorm (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_tsImgRef *sImgRef*)

Set an element's normal-state image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference

Returns

true if success, false if error

5.8.2.35 int gslc_TDrvGetTouch (gslc_tsGui ∗ *pGui,* int16_t ∗ *pnX,* int16_t ∗ *pnY,* uint16_t ∗ *pnPress*)

Get the last touch event from the tslib handler.

Parameters

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

Returns

non-zero if an event was detected or 0 otherwise

5.8.2.36 bool gslc_TDrvInitTouch (gslc_tsGui ∗ *pGui,* const char ∗ *acDev*)

Perform any touchscreen-specific initialization.

Parameters

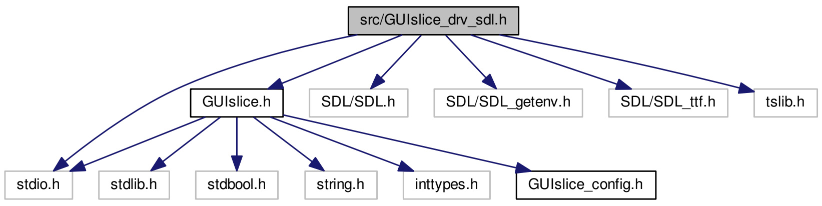
in	*pGui*	Pointer to GUI
in	*acDev*	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

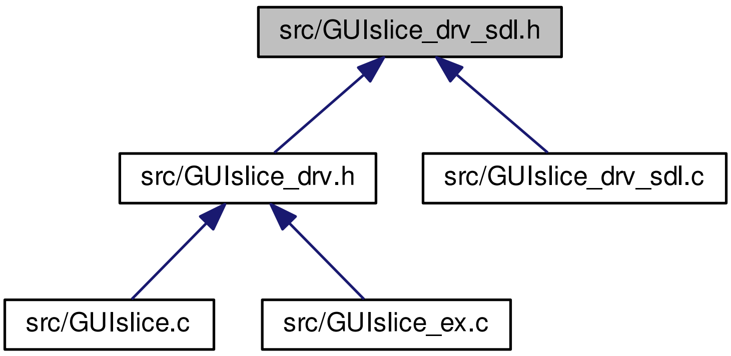
true if successful

5.9 src/GUIslice_drv_sdl.h File Reference

```
#include "GUIslice.h"
#include <stdio.h>
#include <SDL/SDL.h>
#include <SDL/SDL_getenv.h>
#include <SDL/SDL_ttf.h>
#include "tslib.h"
```
Include dependency graph for GUIslice_drv_sdl.h:



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



Classes

- struct gslc_tsDriver

Macros

- #define DRV_HAS_DRAW_POINT 1

  Support gslc_DrvDrawPoint()

- #define DRV_HAS_DRAW_POINTS 1

  Support gslc_DrvDrawPoints()
- #define DRV_HAS_DRAW_LINE 0

  Support gslc_DrvDrawLine()
- #define DRV_HAS_DRAW_RECT_FRAME 0

  Support gslc_DrvDrawFrameRect()
- #define DRV_HAS_DRAW_RECT_FILL 1

  Support gslc_DrvDrawFillRect()
- #define DRV_HAS_DRAW_CIRCLE_FRAME 0

  Support gslc_DrvDrawFrameCircle()
- #define DRV_HAS_DRAW_CIRCLE_FILL 0

  Support gslc_DrvDrawFillCircle()
- #define DRV_HAS_DRAW_TRI_FRAME 0

  Support gslc_DrvDrawFrameTriangle()
- #define DRV_HAS_DRAW_TRI_FILL 0

  Support gslc_DrvDrawFillTriangle()
- #define DRV_HAS_DRAW_TEXT 1

  Support gslc_DrvDrawTxt()

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.
- void ∗ gslc_DrvFontAdd (const char ∗acFontName, uint16_t nFontSz)

  Load a font from a file and return pointer to it.
- void gslc_DrvFontsDestruct (gslc_tsGui ∗pGui)

  Release all fonts defined in the GUI.
- bool gslc_DrvGetTxtSize (gslc_tsGui ∗pGui, gslc_tsFont ∗pFont, const char ∗pStr, gslc_teTxtFlags eTxt↩ Flags, 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_DrvScreenLock (gslc_tsGui ∗pGui)

  Lock an SDL surface so that direct pixel manipulation can be done safely.
- void gslc_DrvScreenUnlock (gslc_tsGui ∗pGui)

  Unlock the SDL surface after pixel manipulation is complete.
- uint32_t gslc_DrvAdaptColorRaw (gslc_tsGui ∗pGui, gslc_tsColor nCol)

  Convert an RGB color triplet into the surface pixel value.
- uint32_t gslc_DrvDrawGetPixelRaw (gslc_tsGui ∗pGui, int16_t nX, int16_t nY)

  Get the pixel at (X,Y) from the active screen.
- void gslc_DrvDrawSetPixelRaw (gslc_tsGui ∗pGui, int16_t nX, int16_t nY, uint32_t nPixelCol)

  Set a pixel on the active screen to the given color.
- void gslc_DrvPasteSurface (gslc_tsGui ∗pGui, int16_t nX, int16_t nY, void ∗pvSrc, void ∗pvDest)

  Copy one image region to another.
- bool gslc_DrvInitTouch (gslc_tsGui ∗pGui, const char ∗acDev)

  Perform any touchscreen-specific initialization.
- bool gslc_TDrvInitTouch (gslc_tsGui ∗pGui, const char ∗acDev)

  Perform any touchscreen-specific initialization.
- int gslc_TDrvGetTouch (gslc_tsGui ∗pGui, int16_t ∗pnX, int16_t ∗pnY, uint16_t ∗pnPress)

  Get the last touch event from the tslib handler.

5.9.1 Macro Definition Documentation

5.9.1.1 #define DRV_HAS_DRAW_CIRCLE_FILL 0

Support gslc_DrvDrawFillCircle()

5.9.1.2 #define DRV_HAS_DRAW_CIRCLE_FRAME 0

Support gslc_DrvDrawFrameCircle()

5.9.1.3 #define DRV_HAS_DRAW_LINE 0

Support gslc_DrvDrawLine()

5.9.1.4 #define DRV_HAS_DRAW_POINT 1

Support gslc_DrvDrawPoint()

5.9.1.5 #define DRV_HAS_DRAW_POINTS 1

Support gslc_DrvDrawPoints()

5.9.1.6 #define DRV_HAS_DRAW_RECT_FILL 1

Support gslc_DrvDrawFillRect()

5.9.1.7 #define DRV_HAS_DRAW_RECT_FRAME 0

Support gslc_DrvDrawFrameRect()

5.9.1.8 #define DRV_HAS_DRAW_TEXT 1

Support gslc_DrvDrawTxt()

5.9.1.9 #define DRV_HAS_DRAW_TRI_FILL 0

Support gslc_DrvDrawFillTriangle()

5.9.1.10 #define DRV_HAS_DRAW_TRI_FRAME 0

Support gslc_DrvDrawFrameTriangle()

5.9.2 Function Documentation

5.9.2.1 SDL_Color gslc_DrvAdaptColor (gslc_tsColor *sCol*)

Translate a gslc_tsColor into an SDL_Color.

Parameters

`in`	*sCol*	gslc_tsColor

Returns

Converted SDL_Color

5.9.2.2 uint32_t gslc_DrvAdaptColorRaw (gslc_tsGui ∗ *pGui,* gslc_tsColor *nCol*)

Convert an RGB color triplet into the surface pixel value.

This is called to produce the native pixel value required by the raw pixel manipulation routines.

Parameters

in	*pGui*	Pointer to GUI
in	*nCol*	RGB value for conversion

Returns

A pixel value for the current screen format

5.9.2.3 SDL_Rect gslc_DrvAdaptRect (gslc_tsRect *rRect*)

Translate a gslc_tsRect into an SDL_Rect.

Parameters

in	*rRect*	gslc_tsRect

Returns

Converted SDL_Rect

5.9.2.4 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	*sTTY*	Terminal device (eg. "/dev/tty0")

Returns

true if success

5.9.2.5 void gslc_DrvDestruct (gslc_tsGui ∗ *pGui*)

Free up any members associated with the driver.

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

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.9.2.6 void gslc_DrvDrawBkgnd (gslc_tsGui ∗ *pGui*)

Copy the background image to destination screen.

Parameters

in	*pGui*	Pointer to GUI

Returns

true if success, false if fail

Copy the background image to destination screen.

5.9.2.7 bool gslc_DrvDrawFillRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a filled rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to fill
in	*nCol*	Color RGB value to fill

Returns

true if success, false if error

5.9.2.8 bool gslc_DrvDrawFrameRect (gslc_tsGui ∗ *pGui,* gslc_tsRect *rRect,* gslc_tsColor *nCol*)

Draw a framed rectangle.

Parameters

in	*pGui*	Pointer to GUI
in	*rRect*	Rectangular region to frame
in	*nCol*	Color RGB value to frame

Returns

true if success, false if error

5.9.2.9 uint32_t gslc_DrvDrawGetPixelRaw (gslc_tsGui ∗ *pGui,* int16_t *nX,* int16_t *nY*)

Get the pixel at (X,Y) from the active screen.

PRE:

- Screen surface must be locked

Parameters

in	*pGui*	Pointer to GUI
in	*nX*	Pixel X coordinate
in	*nY*	Pixel Y coordinate

Returns

Pixel color value from the coordinate or 0 if error

5.9.2.10 bool gslc_DrvDrawImage (gslc_tsGui ∗ *pGui,* int16_t *nDstX,* int16_t *nDstY,* gslc_tsImgRef *sImgRef*)

Copy all of source image to destination screen at specified coordinate.

Parameters

in	*pGui*	Pointer to GUI
in	*nDstX*	Destination X coord for copy
in	*nDstY*	Destination Y coord for copy
in	*sImgRef*	Image reference

Returns

true if success, false if fail

5.9.2.11 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	*pGui*	Pointer to GUI
in	*nX0*	Line start (X coordinate)
in	*nY0*	Line start (Y coordinate)
in	*nX1*	Line finish (X coordinate)
in	*nY1*	Line finish (Y coordinate)
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.9.2.12 bool gslc_DrvDrawPoint (gslc_tsGui ∗ *pGui*, int16_t *nX*, int16_t *nY*, gslc_tsColor *nCol*)

Draw a point.

Parameters

in	*pGui*	Pointer to GUI
in	*nX*	X coordinate of point
in	*nY*	Y coordinate of point
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.9.2.13 bool gslc_DrvDrawPoints (gslc_tsGui ∗ *pGui*, gslc_tsPt ∗ *asPt*, uint16_t *nNumPt*, gslc_tsColor *nCol*)

Draw a point.

Parameters

in	*pGui*	Pointer to GUI
in	*asPt*	Array of points to draw

in	*nNumPt*	Number of points in array
in	*nCol*	Color RGB value to draw

Returns

true if success, false if error

5.9.2.14 void gslc_DrvDrawSetPixelRaw (gslc_tsGui ∗ *pGui,* int16_t *nX,* int16_t *nY,* uint32_t *nPixelCol*)

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

PRE:

- Screen surface must be locked

Parameters

in	*pGui*	Pointer to GUI
in	*nX*	Pixel X coordinate to set
in	*nY*	Pixel Y coordinate to set
in	*nPixelCol*	Raw color pixel value to assign

Returns

none

5.9.2.15 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	*pGui*	Pointer to GUI
in	*nTxtX*	X coordinate of top-left text string
in	*nTxtY*	Y coordinate of top-left text string
in	*pFont*	Ptr to Font
in	*pStr*	String to display
in	*eTxtFlags*	Flags associated with text string
in	*colTxt*	Color to draw text

Returns

true if success, false if failure

5.9.2.16 void∗ gslc_DrvFontAdd (const char ∗ *acFontName,* uint16_t *nFontSz*)

Load a font from a file and return pointer to it.

Parameters

in	*acFontName*	Filename path to the font
in	*nFontSz*	Typeface size to use

Returns

Void ptr to driver-specific font if load was successful, NULL otherwise

Parameters

in	*acFontName*	Filename path to the font
in	*nFontSz*	Typeface size to use

Returns

true if load was successful, false otherwise

5.9.2.17 void gslc_DrvFontsDestruct (gslc_tsGui ∗ *pGui*)

Release all fonts defined in the GUI.

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.9.2.18 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	*pGui*	Pointer to GUI
out	*pnX*	Ptr to X coordinate of last touch event
out	*pnY*	Ptr to Y coordinate of last touch event
out	*pnPress*	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	*pGui*	Pointer to GUI
out	*pnX*	Ptr to X coordinate of last touch event
out	*pnY*	Ptr to Y coordinate of last touch event
out	*pnPress*	Ptr to Pressure level of last touch event (0 for none, >0 for touch)

Returns

true if an event was detected or 0 otherwise

Get the last touch event from the SDL_Event handler.

Parameters

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

Returns

true if an event was detected or 0 otherwise

Parameters

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

Returns

true if an event was detected or false otherwise

Get the last touch event from the SDL_Event handler.

Parameters

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

Returns

true if an event was detected or 0 otherwise

5.9.2.19 bool gslc_DrvGetTxtSize (gslc_tsGui ∗ *pGui,* gslc_tsFont ∗ *pFont,* const char ∗ *pStr,* gslc_teTxtFlags *eTxtFlags,* uint16_t ∗ *pnTxtSzW,* uint16_t ∗ *pnTxtSzH*)

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

Parameters

in	*pGui*	Pointer to GUI
in	*pFont*	Ptr to Font structure
in	*pStr*	String to display
in	*eTxtFlags*	Flags associated with text string
out	*pnTxtSzW*	Ptr to width of text
out	*pnTxtSzH*	Ptr to height of text

Returns

true if success, false if failure

5.9.2.20 void gslc_DrvImageDestruct (void ∗ *pvImg*)

Release an image surface.

Parameters

in	*pvImg*	Void ptr to image

Returns

none

5.9.2.21 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().

Parameters

in	*pGui*	Pointer to GUI

Returns

true if success, false if fail

- 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	*pGui*	Pointer to GUI

Returns

true if success, false if fail

5.9.2.22 bool gslc_DrvInitTouch (gslc_tsGui ∗ *pGui*, const char ∗ *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	*pGui*	Pointer to GUI
in	*acDev*	Device path to touchscreen eg. "/dev/input/touchscreen"

Returns

true if successful

5.9.2.23 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	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

Image pointer (surface/texture/path) or NULL if error

Transparency is enabled by GSLC_BMP_TRANS_EN through use of color (GSLC_BMP_TRANS_RGB).

Parameters

in	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

Image pointer (surface/texture) or NULL if error

5.9.2.24 void gslc_DrvPageFlipNow (gslc_tsGui ∗ *pGui*)

Force a page flip to occur.

This generally copies active screen surface to the display.

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.9.2.25 void gslc_DrvPasteSurface (gslc_tsGui ∗ *pGui*, int16_t *nX*, int16_t *nY*, void ∗ *pvSrc*, void ∗ *pvDest*)

Copy one image region to another.

- This is typically used to copy an image to the main screen surface

Parameters

`in`	*pGui*	Pointer to GUI
`in`	*nX*	Destination X coordinate of copy
`in`	*nY*	Destination Y coordinate of copy
`in`	*pvSrc*	Void Ptr to source surface (eg. a loaded image)
`in`	*pvDest*	Void Ptr to destination surface (typically the screen)

Returns

none

5.9.2.26 void gslc_DrvReportInfoPost ( )

Report driver debug info (after initialization)

Returns

none

5.9.2.27 void gslc_DrvReportInfoPre ( )

Report driver debug info (before initialization)

Returns

none

5.9.2.28 bool gslc_DrvScreenLock (gslc_tsGui ∗ *pGui*)

Lock an SDL surface so that direct pixel manipulation can be done safely.

This function is called before any direct pixel updates.

POST:

- Primary screen surface is locked

Parameters

`in`	*pGui*	Pointer to GUI

Returns

true if success, false otherwise

5.9.2.29 void gslc_DrvScreenUnlock (gslc_tsGui ∗ *pGui*)

Unlock the SDL surface after pixel manipulation is complete.

This function is called after all pixel updates are done.

POST:

- Primary screen surface is unlocked

Parameters

in	*pGui*	Pointer to GUI

Returns

none

5.9.2.30 bool gslc_DrvSetBkgndColor (gslc_tsGui * *pGui*, gslc_tsColor *nCol*)

Configure the background to use a solid color.

- The background is used when redrawing the entire page

Parameters

in	*pGui*	Pointer to GUI
in	*nCol*	RGB Color to use

Returns

true if success, false if fail

5.9.2.31 bool gslc_DrvSetBkgndImage (gslc_tsGui * *pGui*, gslc_tsImgRef *sImgRef*)

Configure the background to use a bitmap image.

- The background is used when redrawing the entire page

Parameters

in	*pGui*	Pointer to GUI
in	*sImgRef*	Image reference

Returns

true if success, false if fail

5.9.2.32 bool gslc_DrvSetClipRect (gslc_tsGui * *pGui*, gslc_tsRect * *pRect*)

Set the clipping rectangle for future drawing updates.

Parameters

in	*pGui*	Pointer to GUI
in	*pRect*	Rectangular region to constrain edits

Returns

true if success, false if error

Parameters

in	*pGui*	Pointer to GUI
in	*pRect*	Rectangular region to constrain edits

Returns

none

5.9.2.33 bool gslc_DrvSetElemImageGlow (gslc_tsGui * *pGui*, gslc_tsElem * *pElem*, gslc_tsImgRef *sImgRef*)

Set an element's glow-state image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference

Returns

true if success, false if error

5.9.2.34 bool gslc_DrvSetElemImageNorm (gslc_tsGui * *pGui*, gslc_tsElem * *pElem*, gslc_tsImgRef *sImgRef*)

Set an element's normal-state image.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element to update
in	*sImgRef*	Image reference

Returns

true if success, false if error

5.9.2.35 int gslc_TDrvGetTouch (gslc_tsGui * *pGui*, int16_t * *pnX*, int16_t * *pnY*, uint16_t * *pnPress*)

Get the last touch event from the tslib handler.

Parameters

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

Returns

non-zero if an event was detected or 0 otherwise

5.9.2.36 bool gslc_TDrvInitTouch (gslc_tsGui * *pGui*, const char * *acDev*)

Perform any touchscreen-specific initialization.

Parameters

in	*pGui*	Pointer to GUI
in	*acDev*	Device path to touchscreen eg. "/dev/input/touchscreen"

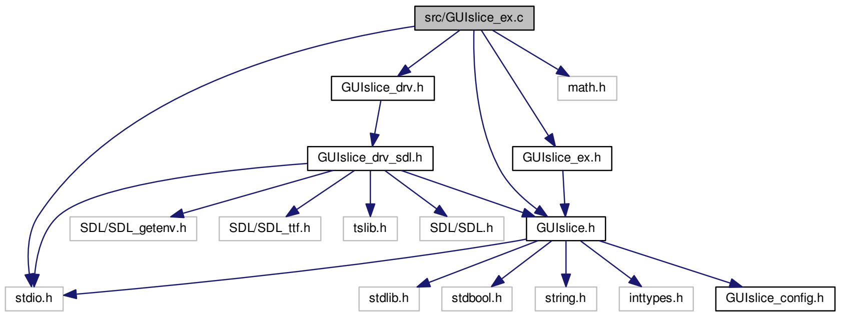
Returns

true if successful

5.10 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_tsElem ∗ 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_tsElem ∗pElem, gslc_teXGaugeStyle nStyle)

  Configure the style of a Gauge element.
- void gslc_ElemXGaugeSetIndicator (gslc_tsElem ∗pElem, gslc_tsColor colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)

  Configure the appearance of the Gauge indicator.
- void gslc_ElemXGaugeSetTicks (gslc_tsElem ∗pElem, gslc_tsColor colTick, uint16_t nTickCnt, uint16_t n←TickLen)

  Configure the appearance of the Gauge ticks.
- void gslc_ElemXGaugeUpdate (gslc_tsElem ∗pElem, int16_t nVal)

  Update a Gauge element's current value.
- void gslc_ElemXGaugeSetFlip (gslc_tsElem ∗pElem, bool bFlip)

  Set a Gauge element's fill direction.
- bool gslc_ElemXGaugeDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a gauge element on the screen.
- bool gslc_ElemXGaugeDrawProgressBar (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, gslc_teRedrawType e←Redraw)

Helper function to draw a gauge with style: progress bar.

- void gslc_ElemXGaugeDrawRadialHelp (gslc_tsGui ∗pGui, int16_t nX, int16_t nY, uint16_t nArrowLen, uint16_t nArrowSz, int16_t n64Ang, bool bFill, gslc_tsColor colFrame)
- bool gslc_ElemXGaugeDrawRadial (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, gslc_teRedrawType eRedraw)

  Helper function to draw a gauge with style: radial.
- bool gslc_ElemXGaugeDrawRamp (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, gslc_teRedrawType eRedraw)

  Helper function to draw a gauge with style: ramp.
- gslc_tsElem ∗ 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 Element.
- bool gslc_ElemXCheckboxGetState (gslc_tsElem ∗pElem)

  Get a Checkbox element's current state.
- gslc_tsElem ∗ gslc_ElemXCheckboxFindChecked (gslc_tsGui ∗pGui, int16_t nGroupId)

  Find the checkbox within a group that has been checked.
- void gslc_ElemXCheckboxSetState (gslc_tsElem ∗pElem, bool bChecked)

  Set a Checkbox element's current state.
- void gslc_ElemXCheckboxToggleState (gslc_tsElem ∗pElem)

  Toggle a Checkbox element's current state.
- bool gslc_ElemXCheckboxDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a Checkbox element on the screen.
- bool gslc_ElemXCheckboxTouch (void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)

  Handle touch events to Checkbox element.
- gslc_tsElem ∗ 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 b←Vert)

  Create a Slider Element.
- void gslc_ElemXSliderSetStyle (gslc_tsElem ∗pElem, 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_tsElem ∗pElem)

  Get a Slider element's current position.
- void gslc_ElemXSliderSetPos (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, int16_t nPos)

  Set a Slider element's current position.
- void gslc_ElemXSliderSetPosFunc (gslc_tsElem ∗pElem, GSLC_CB_XSLIDER_POS funcCb)

  Assign the position callback function for a slider.
- bool gslc_ElemXSliderDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a Slider element on the screen.
- bool gslc_ElemXSliderTouch (void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)

  Handle touch events to Slider element.
- gslc_tsElem ∗ gslc_ElemXSelNumCreate (gslc_tsGui ∗pGui, int16_t nElemId, int16_t nPage, gslc_tsXSel←Num ∗pXData, gslc_tsRect rElem, int8_t nFontId)

  Create a SelNum Element.
- bool gslc_ElemXSelNumDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a SelNum element on the screen.
- int gslc_ElemXSelNumGetCounter (gslc_tsGui ∗pGui, gslc_tsXSelNum ∗pSelNum)

  Get the current counter associated with SelNum.
- void gslc_ElemXSelNumSetCounter (gslc_tsXSelNum ∗pSelNum, int16_t nCount)

  Set the current counter associated with SelNum.
- bool gslc_ElemXSelNumClick (void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nX, int16_t nY)

  Handle a click event within the SelNum.

- bool gslc_ElemXSelNumTouch (void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nRelX, int16_t n←RelY)

  Handle touch (up,down,move) events to SelNum element.
- gslc_tsElem ∗ 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_ElemXTextboxLineWrAdv (gslc_tsXTextbox ∗pBox)
- void gslc_ElemXTextboxScrollSet (gslc_tsElem ∗pElem, uint8_t nScrollPos, uint8_t nScrollMax)

  Set the textbox scroll position (nScrollPos) as a fraction of nScrollMax.
- void gslc_ElemXTextboxBufAdd (gslc_tsXTextbox ∗pBox, char chNew, bool bAdvance)
- void gslc_ElemXTextboxColSet (gslc_tsElem ∗pElem, gslc_tsColor nCol)

  Insert a color set code into the current buffer position.
- void gslc_ElemXTextboxColReset (gslc_tsElem ∗pElem)

  Insert a color reset code into the current buffer position.
- void gslc_ElemXTextboxWrapSet (gslc_tsElem ∗pElem, bool bWrapEn)

  Enable or disable line wrap within textbox.
- void gslc_ElemXTextboxAdd (gslc_tsElem ∗pElem, char ∗pTxt)

  Add a text string to the textbox.
- bool gslc_ElemXTextboxDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a Textbox element on the screen.

Variables

- static const int16_t SELNUM_ID_BTN_INC = 100
- static const int16_t SELNUM_ID_BTN_DEC = 101
- static const int16_t SELNUM_ID_TXT = 102

5.10.1 Function Documentation

5.10.1.1 gslc_tsElem∗ 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 Element.

Parameters

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

Returns

Element pointer or NULL if failure

5.10.1.2 bool gslc_ElemXCheckboxDraw (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teRedrawType *eRedraw*)

Draw a Checkbox element on the screen.

- Called from gslc_ElemDraw()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.10.1.3 gslc_tsElem∗ gslc_ElemXCheckboxFindChecked (gslc_tsGui ∗ *pGui,* int16_t *nGroupId*)

Find the checkbox within a group that has been checked.

Parameters

in	*pGui*	Pointer to GUI
in	*nGroupId*	Group ID to search

Returns

Element Ptr or NULL if none checked

5.10.1.4 bool gslc_ElemXCheckboxGetState (gslc_tsElem ∗ *pElem*)

Get a Checkbox element's current state.

Parameters

in	*pElem*	Pointer to Element

Returns

Current state

5.10.1.5 void gslc_ElemXCheckboxSetState (gslc_tsElem ∗ *pElem,* bool *bChecked*)

Set a Checkbox element's current state.

Parameters

in	*pElem*	Pointer to Element
in	*bChecked*	New state

Returns

none

5.10.1.6 void gslc_ElemXCheckboxToggleState (gslc_tsElem ∗ *pElem*)

Toggle a Checkbox element's current state.

Parameters

in	*pElem*	Pointer to Element

Returns

none

5.10.1.7 bool gslc_ElemXCheckboxTouch (void ∗ *pvGui*, void ∗ *pvElem*, **gslc_teTouch** *eTouch*, int16_t *nRelX*, int16_t *nRelY*)

Handle touch events to Checkbox element.

- Called from gslc_ElemSendEventTouch()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eTouch*	Touch event type
in	*nRelX*	Touch X coord relative to element
in	*nRelY*	Touch Y coord relative to element

Returns

true if success, false otherwise

5.10.1.8 **gslc_tsElem**∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*pXData*	Ptr to extended element data structure
in	*rElem*	Rectangle coordinates defining gauge size
in	*nMin*	Minimum value of gauge for nVal comparison
in	*nMax*	Maximum value of gauge for nVal comparison

in	*nVal*	Starting value of gauge
in	*colGauge*	Color for the gauge indicator
in	*bVert*	Flag to indicate vertical vs horizontal action (true = vertical, false = horizontal)

Returns

Pointer to Element or NULL if failure

5.10.1.9 bool gslc_ElemXGaugeDraw (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teRedrawType *eRedraw*)

Draw a gauge element on the screen.

- Called from gslc_ElemDraw()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.10.1.10 bool gslc_ElemXGaugeDrawProgressBar (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_teRedrawType *eRedraw*)

Helper function to draw a gauge with style: progress bar.

- Called from gslc_ElemXGaugeDraw()

Parameters

in	*pGui*	Ptr to GUI
in	*pElem*	Ptr to Element
in	*eRedraw*	Redraw status

Returns

true if success, false otherwise

5.10.1.11 bool gslc_ElemXGaugeDrawRadial (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_teRedrawType *eRedraw*)

Helper function to draw a gauge with style: radial.

- Called from gslc_ElemXGaugeDraw()

Parameters

in	*pGui*	Ptr to GUI
in	*pElem*	Ptr to Element
in	*eRedraw*	Redraw status

Returns

true if success, false otherwise

5.10.1.12 void gslc_ElemXGaugeDrawRadialHelp (gslc_tsGui ∗ *pGui*, int16_t *nX*, int16_t *nY*, uint16_t *nArrowLen*, uint16_t *nArrowSz*, int16_t *n64Ang*, bool *bFill*, gslc_tsColor *colFrame*)

5.10.1.13 bool gslc_ElemXGaugeDrawRamp (gslc_tsGui ∗ *pGui*, gslc_tsElem ∗ *pElem*, gslc_teRedrawType *eRedraw*)

Helper function to draw a gauge with style: ramp.

- Called from gslc_ElemXGaugeDraw()

Parameters

in	*pGui*	Ptr to GUI
in	*pElem*	Ptr to Element
in	*eRedraw*	Redraw status

Returns

true if success, false otherwise

5.10.1.14 void gslc_ElemXGaugeSetFlip (gslc_tsElem ∗ *pElem*, 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	*pElem*	Pointer to Element
in	*bFlip*	If set, reverse direction of fill from default

Returns

none

5.10.1.15 void gslc_ElemXGaugeSetIndicator (gslc_tsElem ∗ *pElem*, gslc_tsColor *colGauge*, uint16_t *nIndicLen*, uint16_t *nIndicTip*, bool *bIndicFill*)

Configure the appearance of the Gauge indicator.

Parameters

`in`	*pElem*	Pointer to Element
`in`	*colGauge*	Color of the indicator
`in`	*nIndicLen*	Length of the indicator
`in`	*nIndicTip*	Size of the indicator tip
`in`	*bIndicFill*	Fill in the indicator if true

Returns

none

5.10.1.16 void gslc_ElemXGaugeSetStyle (gslc_tsElem ∗ *pElem*, 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`	*pElem*	Pointer to Element
`in`	*nType*	Gauge style enumeration

Returns

none

5.10.1.17 void gslc_ElemXGaugeSetTicks (gslc_tsElem ∗ *pElem*, gslc_tsColor *colTick*, uint16_t *nTickCnt*, uint16_t *nTickLen*)

Configure the appearance of the Gauge ticks.

Parameters

`in`	*pElem*	Pointer to Element
`in`	*colTick*	Color of the gauge ticks
`in`	*nTickCnt*	Number of ticks to draw around / along gauge
`in`	*nTickLen*	Length of the tick marks to draw

Returns

none

5.10.1.18 void gslc_ElemXGaugeUpdate (gslc_tsElem ∗ *pElem*, int16_t *nVal*)

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

in	*pElem*	Pointer to Element
in	*nVal*	New value to show in gauge

Returns

none

5.10.1.19 bool gslc_ElemXSelNumClick (void ∗ *pvGui*, void ∗ *pvElem*, gslc_teTouch *eTouch*, int16_t *nX*, int16_t *nY*)

Handle a click event within the SelNum.

- This is called internally by the SelNum touch handler

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eTouch*	Touch event type
in	*nX*	Touch X coord
in	*nY*	Touch Y coord

Returns

none

5.10.1.20 gslc_tsElem∗ gslc_ElemXSelNumCreate (gslc_tsGui ∗ *pGui*, int16_t *nElemId*, int16_t *nPage*, gslc_tsXSelNum ∗ *pXData*, gslc_tsRect *rElem*, int8_t *nFontId*)

Create a SelNum Element.

Parameters

in	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*pXData*	Ptr to extended element data structure
in	*rElem*	Rectangle coordinates defining element size
in	*nFontId*	Font ID to use for drawing the element

Returns

Pointer to Element or NULL if failure

5.10.1.21 bool gslc_ElemXSelNumDraw (void ∗ *pvGui*, void ∗ *pvElem*, gslc_teRedrawType *eRedraw*)

Draw a SelNum element on the screen.

- Called during redraw

Parameters

`in`	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
`in`	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
`in`	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.10.1.22 int gslc_ElemXSelNumGetCounter (gslc_tsGui ∗ *pGui*, gslc_tsXSelNum ∗ *pSelNum*)

Get the current counter associated with SelNum.

Parameters

`in`	*pGui*	Ptr to GUI
`in`	*pSelNum*	Ptr to Element

Returns

Current counter value

5.10.1.23 void gslc_ElemXSelNumSetCounter (gslc_tsXSelNum ∗ *pSelNum*, int16_t *nCount*)

Set the current counter associated with SelNum.

Parameters

`in`	*pSelNum*	Ptr to Element
`in`	*nCount*	New counter value

Returns

none

5.10.1.24 bool gslc_ElemXSelNumTouch (void ∗ *pvGui*, void ∗ *pvElem*, gslc_teTouch *eTouch*, int16_t *nRelX*, int16_t *nRelY*)

Handle touch (up,down,move) events to SelNum element.

- Called from gslc_ElemSendEventTouch()

Parameters

`in`	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
`in`	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
`in`	*eTouch*	Touch event type
`in`	*nRelX*	Touch X coord relative to element
`in`	*nRelY*	Touch Y coord relative to element

Returns

true if success, false otherwise

5.10.1.25 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*pXData*	Ptr to extended element data structure
in	*rElem*	Rectangle coordinates defining checkbox size
in	*nPosMin*	Minimum position value
in	*nPosMax*	Maximum position value
in	*nPos*	Starting position value
in	*nThumbSz*	Size of the thumb control
in	*bVert*	Orientation (true for vertical)

Returns

Element pointer or NULL if failure

5.10.1.26 bool gslc_ElemXSliderDraw (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teRedrawType *eRedraw*)

Draw a Slider element on the screen.

- Called from gslc_ElemDraw()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.10.1.27 int gslc_ElemXSliderGetPos (gslc_tsElem ∗ *pElem*)

Get a Slider element's current position.

Parameters

in	*pElem*	Pointer to Element

Returns

Current slider position

5.10.1.28 void gslc_ElemXSliderSetPos (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* int16_t *nPos*)

Set a Slider element's current position.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element
in	*nPos*	New position value

Returns

none

5.10.1.29 void gslc_ElemXSliderSetPosFunc (gslc_tsElem ∗ *pElem,* GSLC_CB_XSLIDER_POS *funcCb*)

Assign the position callback function for a slider.

Parameters

in	*pElem*	Pointer to element
in	*funcCb*	Function pointer to position routine (or NULL for none)

Returns

none

5.10.1.30 void gslc_ElemXSliderSetStyle (gslc_tsElem ∗ *pElem,* bool *bTrim,* gslc_tsColor *colTrim,* uint16_t *nTickDiv,* int16_t *nTickLen,* gslc_tsColor *colTick*)

Set a Slider element's current position.

Parameters

in	*pElem*	Pointer to Element
in	*bTrim*	Show a colored trim?
in	*colTrim*	Color of trim
in	*nTickDiv*	Number of tick divisions to show (0 for none)
in	*nTickLen*	Length of tickmarks
in	*colTick*	Color of ticks

Returns

none

5.10.1.31 bool gslc_ElemXSliderTouch (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teTouch *eTouch,* int16_t *nRelX,* int16_t *nRelY*)

Handle touch events to Slider element.

- Called from gslc_ElemSendEventTouch()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eTouch*	Touch event type
in	*nRelX*	Touch X coord relative to element
in	*nRelY*	Touch Y coord relative to element

Returns

true if success, false otherwise

5.10.1.32 void gslc_ElemXTextboxAdd (gslc_tsElem ∗ *pElem*, 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	*pElem*	Pointer to element
in	*pTxt*	Pointer to text string (null-terminated)

Returns

none

5.10.1.33 void gslc_ElemXTextboxBufAdd (gslc_tsXTextbox ∗ *pBox*, char *chNew*, bool *bAdvance*)

5.10.1.34 void gslc_ElemXTextboxColReset (gslc_tsElem ∗ *pElem*)

Insert a color reset code into the current buffer position.

Parameters

in	*pElem*	Pointer to element

Returns

none

5.10.1.35 void gslc_ElemXTextboxColSet (gslc_tsElem ∗ *pElem*, gslc_tsColor *nCol*)

Insert a color set code into the current buffer position.

Parameters

in	*pElem*	Pointer to element
in	*nCol*	Color to assign for next text written to textbox

Returns

none

5.10.1.36 gslc_tsElem∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*pXData*	Ptr to extended element data structure
in	*rElem*	Rectangle coordinates defining checkbox size
in	*nFontId*	Font ID to use for text area
in	*pBuf*	Ptr to text buffer (already allocated) with size (nBufRows∗nBufCols) chars
in	*nBufRows*	Number of rows in buffer
in	*nBufCols*	Number of columns in buffer (incl special codes)

Returns

Element pointer or NULL if failure

5.10.1.37 bool gslc_ElemXTextboxDraw (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teRedrawType *eRedraw*)

Draw a Textbox element on the screen.

- Called from gslc_ElemDraw()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.10.1.38 void gslc_ElemXTextboxLineWrAdv (gslc_tsXTextbox ∗ *pBox*)

5.10.1.39 void gslc_ElemXTextboxScrollSet (gslc_tsElem ∗ *pElem,* uint8_t *nScrollPos,* uint8_t *nScrollMax*)

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

Parameters

in	*pElem*	Pointer to element
in	*nScrollPos*	New scroll position
in	*nScrollMax*	Maximum scroll position

Returns

none

5.10.1.40 void gslc_ElemXTextboxWrapSet (gslc_tsElem ∗ *pElem,* bool *bWrapEn*)

Enable or disable line wrap within textbox.

Parameters

in	*pElem*	Pointer to element
in	*bWrapEn*	Enable line wrap if true

Returns

none

5.10.2 Variable Documentation

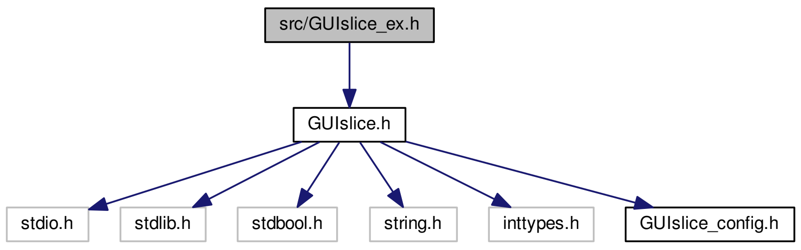
5.10.2.1 const int16_t SELNUM_ID_BTN_DEC = 101 `[static]`

5.10.2.2 const int16_t SELNUM_ID_BTN_INC = 100 `[static]`

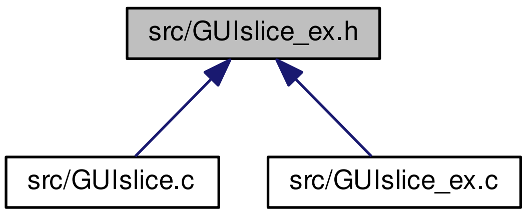
5.10.2.3 const int16_t SELNUM_ID_TXT = 102 `[static]`

5.11 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_tsXSelNum

  Extended data for SelNum element.
- struct gslc_tsXTextbox

  Extended data for Textbox element.

Macros

- #define SELNUM_STR_LEN 6
- #define GSLC_XTEXTBOX_CODE_COL_SET 187

  Definitions for textbox special inline codes.
- #define GSLC_XTEXTBOX_CODE_COL_RESET 188

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_SELNUM,
  GSLC_TYPEX_TEXTBOX }

  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, GS←
  LCX_CHECKBOX_STYLE_ROUND }

  Checkbox drawing style.

Functions

- gslc_tsElem ∗ 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_tsElem ∗pElem, gslc_teXGaugeStyle nType)

  Configure the style of a Gauge element.
- void gslc_ElemXGaugeSetIndicator (gslc_tsElem ∗pElem, gslc_tsColor colGauge, uint16_t nIndicLen, uint16_t nIndicTip, bool bIndicFill)

  Configure the appearance of the Gauge indicator.
- void gslc_ElemXGaugeSetTicks (gslc_tsElem ∗pElem, gslc_tsColor colTick, uint16_t nTickCnt, uint16_t n←
  TickLen)

  Configure the appearance of the Gauge ticks.
- void gslc_ElemXGaugeUpdate (gslc_tsElem ∗pElem, int16_t nVal)

  Update a Gauge element's current value.

- void gslc_ElemXGaugeSetFlip (gslc_tsElem ∗pElem, bool bFlip)

  Set a Gauge element's fill direction.
- bool gslc_ElemXGaugeDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a gauge element on the screen.
- bool gslc_ElemXGaugeDrawProgressBar (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, gslc_teRedrawType e↩ Redraw)

  Helper function to draw a gauge with style: progress bar.
- bool gslc_ElemXGaugeDrawRadial (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, gslc_teRedrawType eRedraw)

  Helper function to draw a gauge with style: radial.
- bool gslc_ElemXGaugeDrawRamp (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, gslc_teRedrawType eRedraw)

  Helper function to draw a gauge with style: ramp.
- gslc_tsElem ∗ 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 Element.
- bool gslc_ElemXCheckboxGetState (gslc_tsElem ∗pElem)

  Get a Checkbox element's current state.
- void gslc_ElemXCheckboxSetState (gslc_tsElem ∗pElem, bool bChecked)

  Set a Checkbox element's current state.
- gslc_tsElem ∗ gslc_ElemXCheckboxFindChecked (gslc_tsGui ∗pGui, int16_t nGroupId)

  Find the checkbox within a group that has been checked.
- void gslc_ElemXCheckboxToggleState (gslc_tsElem ∗pElem)

  Toggle a Checkbox element's current state.
- bool gslc_ElemXCheckboxDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a Checkbox element on the screen.
- bool gslc_ElemXCheckboxTouch (void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)

  Handle touch events to Checkbox element.
- gslc_tsElem ∗ 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 b↩ Vert)

  Create a Slider Element.
- void gslc_ElemXSliderSetStyle (gslc_tsElem ∗pElem, 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_tsElem ∗pElem)

  Get a Slider element's current position.
- void gslc_ElemXSliderSetPos (gslc_tsGui ∗pGui, gslc_tsElem ∗pElem, int16_t nPos)

  Set a Slider element's current position.
- void gslc_ElemXSliderSetPosFunc (gslc_tsElem ∗pElem, GSLC_CB_XSLIDER_POS funcCb)

  Assign the position callback function for a slider.
- bool gslc_ElemXSliderDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a Slider element on the screen.
- bool gslc_ElemXSliderTouch (void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nRelX, int16_t nRelY)

  Handle touch events to Slider element.
- gslc_tsElem ∗ gslc_ElemXSelNumCreate (gslc_tsGui ∗pGui, int16_t nElemId, int16_t nPage, gslc_tsXSel↩ Num ∗pXData, gslc_tsRect rElem, int8_t nFontId)

  Create a SelNum Element.
- bool gslc_ElemXSelNumDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a SelNum element on the screen.
- int gslc_ElemXSelNumGetCounter (gslc_tsGui ∗pGui, gslc_tsXSelNum ∗pSelNum)

  Get the current counter associated with SelNum.

- void gslc_ElemXSelNumSetCounter (gslc_tsXSelNum ∗pSelNum, int16_t nCount)

  Set the current counter associated with SelNum.
- bool gslc_ElemXSelNumClick (void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nX, int16_t nY)

  Handle a click event within the SelNum.
- bool gslc_ElemXSelNumTouch (void ∗pvGui, void ∗pvElem, gslc_teTouch eTouch, int16_t nRelX, int16_t n←RelY)

  Handle touch (up,down,move) events to SelNum element.
- gslc_tsElem ∗ 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.
- bool gslc_ElemXTextboxDraw (void ∗pvGui, void ∗pvElem, gslc_teRedrawType eRedraw)

  Draw a Textbox element on the screen.
- void gslc_ElemXTextboxAdd (gslc_tsElem ∗pElem, char ∗pTxt)

  Add a text string to the textbox.
- void gslc_ElemXTextboxColSet (gslc_tsElem ∗pElem, gslc_tsColor nCol)

  Insert a color set code into the current buffer position.
- void gslc_ElemXTextboxColReset (gslc_tsElem ∗pElem)

  Insert a color reset code into the current buffer position.
- void gslc_ElemXTextboxWrapSet (gslc_tsElem ∗pElem, bool bWrapEn)

  Enable or disable line wrap within textbox.
- void gslc_ElemXTextboxScrollSet (gslc_tsElem ∗pElem, uint8_t nScrollPos, uint8_t nScrollMax)

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

5.11.1 Macro Definition Documentation

5.11.1.1 #define GSLC_XTEXTBOX_CODE_COL_RESET 188

5.11.1.2 #define GSLC_XTEXTBOX_CODE_COL_SET 187

Definitions for textbox special inline codes.

5.11.1.3 #define SELNUM_STR_LEN 6

5.11.2 Typedef Documentation

5.11.2.1 typedef bool(∗ GSLC_CB_XSLIDER_POS)(void ∗pvGui, void ∗pvElem, int16_t nPos)

Callback function for slider feedback.

5.11.3 Enumeration Type Documentation

5.11.3.1 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_SELNUM	SelNum extended element.
GSLC_TYPEX_TEXTBOX	Textbox extended element.

5.11.3.2 enum **gslc_teXCheckboxStyle**

Checkbox drawing style.

Enumerator

GSLCX_CHECKBOX_STYLE_BOX Inner box.

GSLCX_CHECKBOX_STYLE_X Crossed.

GSLCX_CHECKBOX_STYLE_ROUND Circular.

5.11.3.3 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.

5.11.4 Function Documentation

5.11.4.1 **gslc_tsElem**∗ 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 Element.

Parameters

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

Returns

Element pointer or NULL if failure

5.11.4.2 bool gslc_ElemXCheckboxDraw (void ∗ *pvGui,* void ∗ *pvElem,* **gslc_teRedrawType** *eRedraw*)

Draw a Checkbox element on the screen.

- Called from gslc_ElemDraw()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.11.4.3 gslc_tsElem∗ gslc_ElemXCheckboxFindChecked (gslc_tsGui ∗ *pGui*, int16_t *nGroupId*)

Find the checkbox within a group that has been checked.

Parameters

in	*pGui*	Pointer to GUI
in	*nGroupId*	Group ID to search

Returns

Element Ptr or NULL if none checked

5.11.4.4 bool gslc_ElemXCheckboxGetState (gslc_tsElem ∗ *pElem*)

Get a Checkbox element's current state.

Parameters

in	*pElem*	Pointer to Element

Returns

Current state

5.11.4.5 void gslc_ElemXCheckboxSetState (gslc_tsElem ∗ *pElem*, bool *bChecked*)

Set a Checkbox element's current state.

Parameters

in	*pElem*	Pointer to Element
in	*bChecked*	New state

Returns

none

5.11.4.6 void gslc_ElemXCheckboxToggleState (gslc_tsElem ∗ *pElem*)

Toggle a Checkbox element's current state.

Parameters

in	*pElem*	Pointer to Element

Returns

none

5.11.4.7 bool gslc_ElemXCheckboxTouch (void ∗ *pvGui*, void ∗ *pvElem*, gslc_teTouch *eTouch*, int16_t *nRelX*, int16_t *nRelY*)

Handle touch events to Checkbox element.

- Called from gslc_ElemSendEventTouch()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eTouch*	Touch event type
in	*nRelX*	Touch X coord relative to element
in	*nRelY*	Touch Y coord relative to element

Returns

true if success, false otherwise

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

Returns

Pointer to Element or NULL if failure

5.11.4.9 bool gslc_ElemXGaugeDraw (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teRedrawType *eRedraw*)

Draw a gauge element on the screen.

- Called from gslc_ElemDraw()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.11.4.10 bool gslc_ElemXGaugeDrawProgressBar (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_teRedrawType *eRedraw*)

Helper function to draw a gauge with style: progress bar.

- Called from gslc_ElemXGaugeDraw()

Parameters

in	*pGui*	Ptr to GUI
in	*pElem*	Ptr to Element
in	*eRedraw*	Redraw status

Returns

true if success, false otherwise

5.11.4.11 bool gslc_ElemXGaugeDrawRadial (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_teRedrawType *eRedraw*)

Helper function to draw a gauge with style: radial.

- Called from gslc_ElemXGaugeDraw()

Parameters

in	*pGui*	Ptr to GUI
in	*pElem*	Ptr to Element
in	*eRedraw*	Redraw status

Returns

true if success, false otherwise

5.11.4.12 bool gslc_ElemXGaugeDrawRamp (gslc_tsGui ∗ *pGui,* gslc_tsElem ∗ *pElem,* gslc_teRedrawType *eRedraw*)

Helper function to draw a gauge with style: ramp.

- Called from gslc_ElemXGaugeDraw()

Parameters

in	*pGui*	Ptr to GUI
in	*pElem*	Ptr to Element
in	*eRedraw*	Redraw status

Returns

true if success, false otherwise

5.11.4.13 void gslc_ElemXGaugeSetFlip (gslc_tsElem ∗ *pElem,* 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	*pElem*	Pointer to Element
in	*bFlip*	If set, reverse direction of fill from default

Returns

none

5.11.4.14 void gslc_ElemXGaugeSetIndicator (gslc_tsElem ∗ *pElem,* gslc_tsColor *colGauge,* uint16_t *nIndicLen,* uint16_t *nIndicTip,* bool *bIndicFill*)

Configure the appearance of the Gauge indicator.

Parameters

in	*pElem*	Pointer to Element
in	*colGauge*	Color of the indicator
in	*nIndicLen*	Length of the indicator
in	*nIndicTip*	Size of the indicator tip
in	*bIndicFill*	Fill in the indicator if true

Returns

none

5.11.4.15 void gslc_ElemXGaugeSetStyle (gslc_tsElem ∗ *pElem,* 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	*pElem*	Pointer to Element
in	*nType*	Gauge style enumeration

Returns

none

5.11.4.16 void gslc_ElemXGaugeSetTicks (gslc_tsElem ∗ *pElem,* gslc_tsColor *colTick,* uint16_t *nTickCnt,* uint16_t *nTickLen*)

Configure the appearance of the Gauge ticks.

Parameters

in	*pElem*	Pointer to Element
in	*colTick*	Color of the gauge ticks
in	*nTickCnt*	Number of ticks to draw around / along gauge
in	*nTickLen*	Length of the tick marks to draw

Returns

none

5.11.4.17 void gslc_ElemXGaugeUpdate (gslc_tsElem ∗ *pElem,* int16_t *nVal*)

Update a Gauge element's current value.

- Note that min & max values are assigned in create()

Parameters

in	*pElem*	Pointer to Element
in	*nVal*	New value to show in gauge

Returns

none

5.11.4.18 bool gslc_ElemXSelNumClick (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teTouch *eTouch,* int16_t *nX,* int16_t *nY*)

Handle a click event within the SelNum.

- This is called internally by the SelNum touch handler

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)

in	*eTouch*	Touch event type
in	*nX*	Touch X coord
in	*nY*	Touch Y coord

Returns

none

5.11.4.19 gslc_tsElem∗ gslc_ElemXSelNumCreate (gslc_tsGui ∗ *pGui,* int16_t *nElemId,* int16_t *nPage,* gslc_tsXSelNum ∗ *pXData,* gslc_tsRect *rElem,* int8_t *nFontId*)

Create a SelNum Element.

Parameters

in	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*pXData*	Ptr to extended element data structure
in	*rElem*	Rectangle coordinates defining element size
in	*nFontId*	Font ID to use for drawing the element

Returns

Pointer to Element or NULL if failure

5.11.4.20 bool gslc_ElemXSelNumDraw (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teRedrawType *eRedraw*)

Draw a SelNum element on the screen.

- Called during redraw

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.11.4.21 int gslc_ElemXSelNumGetCounter (gslc_tsGui ∗ *pGui,* gslc_tsXSelNum ∗ *pSelNum*)

Get the current counter associated with SelNum.

Parameters

in	*pGui*	Ptr to GUI
in	*pSelNum*	Ptr to Element

Returns

Current counter value

5.11.4.22 void gslc_ElemXSelNumSetCounter (gslc_tsXSelNum ∗ *pSelNum,* int16_t *nCount*)

Set the current counter associated with SelNum.

Parameters

`in`	*pSelNum*	Ptr to Element
`in`	*nCount*	New counter value

Returns

none

5.11.4.23 bool gslc_ElemXSelNumTouch (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teTouch *eTouch,* int16_t *nRelX,* int16_t *nRelY*)

Handle touch (up,down,move) events to SelNum element.

- Called from gslc_ElemSendEventTouch()

Parameters

`in`	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
`in`	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
`in`	*eTouch*	Touch event type
`in`	*nRelX*	Touch X coord relative to element
`in`	*nRelY*	Touch Y coord relative to element

Returns

true if success, false otherwise

5.11.4.24 gslc_tsElem∗ 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`	*pGui*	Pointer to GUI
`in`	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
`in`	*nPage*	Page ID to attach element to
`in`	*pXData*	Ptr to extended element data structure
`in`	*rElem*	Rectangle coordinates defining checkbox size
`in`	*nPosMin*	Minimum position value
`in`	*nPosMax*	Maximum position value
`in`	*nPos*	Starting position value
`in`	*nThumbSz*	Size of the thumb control
`in`	*bVert*	Orientation (true for vertical)

Returns

Element pointer or NULL if failure

5.11.4.25 bool gslc_ElemXSliderDraw (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teRedrawType *eRedraw*)

Draw a Slider element on the screen.

- Called from gslc_ElemDraw()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.11.4.26 int gslc_ElemXSliderGetPos (gslc_tsElem ∗ *pElem*)

Get a Slider element's current position.

Parameters

in	*pElem*	Pointer to Element

Returns

Current slider position

5.11.4.27 void gslc_ElemXSliderSetPos (gslc_tsGui ∗ *pGui*, gslc_tsElem ∗ *pElem*, int16_t *nPos*)

Set a Slider element's current position.

Parameters

in	*pGui*	Pointer to GUI
in	*pElem*	Pointer to Element
in	*nPos*	New position value

Returns

none

5.11.4.28 void gslc_ElemXSliderSetPosFunc (gslc_tsElem ∗ *pElem*, GSLC_CB_XSLIDER_POS *funcCb*)

Assign the position callback function for a slider.

Parameters

in	*pElem*	Pointer to element
in	*funcCb*	Function pointer to position routine (or NULL for none)

Returns

none

5.11.4.29 void gslc_ElemXSliderSetStyle (gslc_tsElem ∗ *pElem*, bool *bTrim*, gslc_tsColor *colTrim*, uint16_t *nTickDiv*, int16_t *nTickLen*, gslc_tsColor *colTick*)

Set a Slider element's current position.

Parameters

`in`	*pElem*	Pointer to Element
`in`	*bTrim*	Show a colored trim?
`in`	*colTrim*	Color of trim
`in`	*nTickDiv*	Number of tick divisions to show (0 for none)
`in`	*nTickLen*	Length of tickmarks
`in`	*colTick*	Color of ticks

Returns

none

5.11.4.30 bool gslc_ElemXSliderTouch (void ∗ *pvGui,* void ∗ *pvElem,* gslc_teTouch *eTouch,* int16_t *nRelX,* int16_t *nRelY*)

Handle touch events to Slider element.

- Called from gslc_ElemSendEventTouch()

Parameters

`in`	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
`in`	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
`in`	*eTouch*	Touch event type
`in`	*nRelX*	Touch X coord relative to element
`in`	*nRelY*	Touch Y coord relative to element

Returns

true if success, false otherwise

5.11.4.31 void gslc_ElemXTextboxAdd (gslc_tsElem ∗ *pElem,* 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`	*pElem*	Pointer to element
`in`	*pTxt*	Pointer to text string (null-terminated)

Returns

none

5.11.4.32 void gslc_ElemXTextboxColReset (gslc_tsElem ∗ *pElem*)

Insert a color reset code into the current buffer position.

Parameters

in	*pElem*	Pointer to element

Returns

none

5.11.4.33 void gslc_ElemXTextboxColSet (**gslc_tsElem** ∗ *pElem*, **gslc_tsColor** *nCol*)

Insert a color set code into the current buffer position.

Parameters

in	*pElem*	Pointer to element
in	*nCol*	Color to assign for next text written to textbox

Returns

none

5.11.4.34 **gslc_tsElem**∗ 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	*pGui*	Pointer to GUI
in	*nElemId*	Element ID to assign (0..16383 or GSLC_ID_AUTO to autogen)
in	*nPage*	Page ID to attach element to
in	*pXData*	Ptr to extended element data structure
in	*rElem*	Rectangle coordinates defining checkbox size
in	*nFontId*	Font ID to use for text area
in	*pBuf*	Ptr to text buffer (already allocated) with size (nBufRows∗nBufCols) chars
in	*nBufRows*	Number of rows in buffer
in	*nBufCols*	Number of columns in buffer (incl special codes)

Returns

Element pointer or NULL if failure

5.11.4.35 bool gslc_ElemXTextboxDraw (void ∗ *pvGui*, void ∗ *pvElem*, **gslc_teRedrawType** *eRedraw*)

Draw a Textbox element on the screen.

- Called from gslc_ElemDraw()

Parameters

in	*pvGui*	Void ptr to GUI (typecast to gslc_tsGui∗)
in	*pvElem*	Void ptr to Element (typecast to gslc_tsElem∗)
in	*eRedraw*	Redraw mode

Returns

true if success, false otherwise

5.11.4.36 void gslc_ElemXTextboxScrollSet (gslc_tsElem ∗ *pElem,* uint8_t *nScrollPos,* uint8_t *nScrollMax*)

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

Parameters

in	*pElem*	Pointer to element
in	*nScrollPos*	New scroll position
in	*nScrollMax*	Maximum scroll position

Returns

none

5.11.4.37 void gslc_ElemXTextboxWrapSet (gslc_tsElem ∗ *pElem,* bool *bWrapEn*)

Enable or disable line wrap within textbox.

Parameters

in	*pElem*	Pointer to element
in	*bWrapEn*	Enable line wrap if true

Returns

none