Arduino Gyroscope Driver

Generated by Doxygen 1.8.9.1

Tue Aug 18 2015 22:52:07

Contents

1 Hierarchical Index

1.1 Class Hierarchy

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

2 Class Index

2.1 Class List

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

3 File Index

3.1 File List

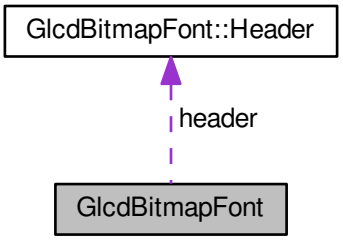
Here is a list of all files with brief descriptions:

4 Class Documentation

4.1 GlcdBitmapFont Class Reference

`#include <GlcdBitmapFont.h>`

Collaboration diagram for GlcdBitmapFont:



Classes

- struct Header

Public Member Functions

- GlcdBitmapFont (SeekableInputStream *inputStream)
- unsigned char getInfo ()
- unsigned char getCharacterWidth ()
- unsigned char getCharacterHeight ()
- unsigned char getSequenceCount ()
- unsigned char getGlyphLength ()
- virtual unsigned char readGlyphData (unsigned char *buf, char c)

Protected Member Functions

- virtual unsigned int getGlyphOffset (char c)

Protected Attributes

- Header header
- unsigned char glyphLength
- SeekableInputStream * inputStream
- unsigned int dataOffset

4.1.1 Detailed Description

Arduino Graphic LCD Library.

GlcdBitmapFont.h

The representation of a glcd font.

Glcd bitmap font is an array which represents the font glyph as a bitmap.

This font has fixed glyph size.

The first bytes specify the font's information and glyph sequences;

Header example:

```
unsigned char info;
unsigned char characterWidth;
unsigned char characterHeight;
unsigned char sequenceCount;
```

```
0x00, 0xww, 0xhh, 0xnn
 |     |     |     |___ Sequence count
 |     |     |_________ Character height (in bits)
 |     |_______________ Character width (in bits)
 |_____________________ Into
```

The next bytes, after the head, specify the sequences information, each sequence have 3 information:

```
unsigned char first;
unsigned char last;
unsigned char offset[msb];
unsigned char offset[lsb];
```

The sequence informations are followed one by another. Example: Considering a font with 2 sequences, this could be the sequence bytes:

```
0x20, 0x22, 0x00, 0xff,
0x40, 0x43, 0x0d, 0xff,
```

which means we have a sequence that starts with the 0x20 char and goes to the 0x22, and the glyph data are stored at the 0x00ff offset. Another sequence starts with the 0x40 char and goes to the 0x43, and the glyph data are stored at the 0x0dff offset.

File structure

Header organization:

```
+-------------------------+
|      Font info          | \
+-------------------------+  \
|   Character width       |   \
+-------------------------+    } Header
|   Character height      |   /
+-------------------------+  /
|   Sequence count        | /
+-------------------------+
|   First character       | \
+-------------------------+  \
|   Last character        |   \
+-------------------------+    } Sequence #0
|   Offset MSB            |   /
+-------------------------+  /
```

```
|   Offset LSB            | /
+-------------------------+
|   First character       | \
+-------------------------+  \
|   Last character        |   \
+-------------------------+    } Sequence #n
|   Offset MSB            |   /
+-------------------------+  /
|   Offset LSB            | /
+-------------------------+
|                         | 0b00000000
|                         | 0b00000000
|                         | 0b00000000
|        FONT DATA        | 0b00000000
|     ARRAY OF BYTES      | 0b00000000
|                         | 0b00000000
|                         | 0b00000000
|                         | 0b00000000
+-------------------------+
```

Font data

Each character could have any multiple by 8 height. For characters which is 8 bits height, they are just made by bytes in sequence, as follows:

```
  L    L    L    L    L
  i    i    i    i    i
  n    n    n    n    n
  e    e    e    e    e
  0    1    2    3    4    ...
+-------------------------...
| b7 | b7 | b7 | b7 | b7 |
| b6 | b6 | b6 | b6 | b6 |
| b5 | b5 | b5 | b5 | b5 |
| b4 | b4 | b4 | b4 | b4 |
| b3 | b3 | b3 | b3 | b3 |
| b2 | b2 | b2 | b2 | b2 |
| b1 | b1 | b1 | b1 | b1 |
| b0 | b0 | b0 | b0 | b0 |
+-----------------------...
```

Or, this is a character with width equals 5 and height equals 8: (can be the "T" letter)

```
+-+-+-+-+-+
|1|1|1|1|1| -> the MSB
|1|1|1|1|1|
|0|0|1|0|0|
|0|0|1|0|0|
|0|0|1|0|0|
|0|0|1|0|0|
|0|0|1|0|0|
|0|0|0|0|0| -> the LSB
+-+-+-+-+-+
```

Just when a character which has the height bigger than 8 that the things comes more difficult. For example, imagine if the above letter with w: 10 and h: 16.

```
+-+-+-+-+-+-+-+-+-+-+       +-+-+-+-+-+-+-+-+-+-+
|1|1|1|1|1|1|1|1|1|1|       |0|0|0|0|1|1|1|1|0|0| -> the MSB
|1|1|1|1|1|1|1|1|1|1|       |0|0|0|0|1|1|1|1|0|0|
|1|1|1|1|1|1|1|1|1|1|       |0|0|0|0|1|1|1|1|0|0|
|1|1|1|1|1|1|1|1|1|1|       |0|0|0|0|1|1|1|1|0|0|
```

```
|0|0|0|0|1|1|1|1|0|0|0|0|       |0|0|0|0|1|1|1|1|0|0|0|0|
|0|0|0|0|1|1|1|1|0|0|0|0|       |0|0|0|0|0|0|0|0|0|0|0|0|
|0|0|0|0|1|1|1|1|0|0|0|0|       |0|0|0|0|0|0|0|0|0|0|0|0|
|0|0|0|0|1|1|1|1|0|0|0|0|       |0|0|0|0|0|0|0|0|0|0|0|0| -> the LSB
+-+-+-+-+-+-+-+-+-+-+-+-+-+       +-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The first 10 bytes are the top part of the character, and the las 10 bytes are the bottom part of the character.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 161 of file GlcdBitmapFont.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 GlcdBitmapFont::GlcdBitmapFont (SeekableInputStream * *inputStream*)

Public constructor.

Parameters

inputStream	The associated input stream.

Definition at line 16 of file GlcdBitmapFont.cpp.

4.1.3 Member Function Documentation

4.1.3.1 unsigned char GlcdBitmapFont::getCharacterHeight ( )

Gets the character height.

Returns

The heigh of a char.

Definition at line 34 of file GlcdBitmapFont.cpp.

4.1.3.2 unsigned char GlcdBitmapFont::getCharacterWidth ( )

Gets the character width.

Returns

The width of a char.

Definition at line 30 of file GlcdBitmapFont.cpp.

4.1.3.3 unsigned char GlcdBitmapFont::getGlyphLength ( )

Gets the glyph length.

Returns

The length of the glyph.

Definition at line 42 of file GlcdBitmapFont.cpp.

4.1.3.4 unsigned int GlcdBitmapFont::getGlyphOffset (char *c*) [protected], [virtual]

Gets the offset to the given character.

Parameters

c	The character to be used.

Returns

The offset.

Definition at line 55 of file GlcdBitmapFont.cpp.

4.1.3.5 unsigned char GlcdBitmapFont::getInfo ( )

Gets the font info.

Returns

Font info entry.

Definition at line 26 of file GlcdBitmapFont.cpp.

4.1.3.6 unsigned char GlcdBitmapFont::getSequenceCount ( )

Gets the sequence count.

Returns

The number of sequences.

Definition at line 38 of file GlcdBitmapFont.cpp.

4.1.3.7 unsigned char GlcdBitmapFont::readGlyphData (unsigned char * *buf*, char *c*) [virtual]

Gets the array of bytes representing the given character.

Parameters

buf	The buffer.
c	The character.

Returns

The number of bytes read.

Definition at line 46 of file GlcdBitmapFont.cpp.

4.1.4 Member Data Documentation

4.1.4.1 unsigned int GlcdBitmapFont::dataOffset [protected]

Data offset.

It is the point when the header ends.

Definition at line 188 of file GlcdBitmapFont.h.

4.1.4.2 unsigned char GlcdBitmapFont::glyphLength [protected]

Glyph length.

Definition at line 178 of file GlcdBitmapFont.h.

4.1.4.3 Header GlcdBitmapFont::header [protected]

Definition at line 173 of file GlcdBitmapFont.h.

4.1.4.4 SeekableInputStream* GlcdBitmapFont::inputStream [protected]

Input stream which font data comes from.

Definition at line 183 of file GlcdBitmapFont.h.

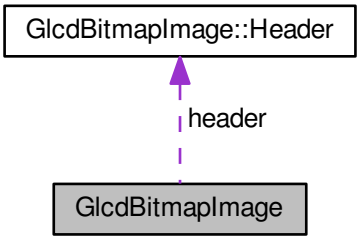
The documentation for this class was generated from the following files:

- GlcdBitmapFont.h
- GlcdBitmapFont.cpp

4.2 GlcdBitmapImage Class Reference

`#include <GlcdBitmapImage.h>`

Collaboration diagram for GlcdBitmapImage:



Classes

- struct Header

Public Member Functions

- GlcdBitmapImage (SeekableInputStream *inputStream)
- unsigned char getInfo ()
- unsigned char getWidth ()
- unsigned char getHeight ()
- virtual bool getPixel (unsigned char x, unsigned char y)
- virtual void readColumn (unsigned char *buf, unsigned char col)
- virtual void readRow (unsigned char *buf, unsigned char row)

Protected Attributes

- Header header
- SeekableInputStream * inputStream
- unsigned int dataOffset

4.2.1 Detailed Description

Arduino Graphic LCD Library.

GlcdBitmapImage.h

The representation of a glcd image.

The first three bytes specify the image's information and dimensions; 'info', 'width' and 'height' respectively;

The rest of the data have the content of the image;

The minimum size of an image is 1x8; this hypothetical image has 4 bytes, as follow:

```
0x00, 0x01, 0x08, 0xXX
 |     |     |     |___ Image content
 |     |     |_________ Height
 |     |_______________ Width
 |_____________________ Info
```

The contents of the image is organized to take advantage of the structure of the display, the display is divided into pages, each page has a number of bytes, these bytes are arranged horizontally, forming a chain of 8-bit high.

File structure

Header organization:

```
+-------------------------+
|      Image info         | 0b00000000
+-------------------------+
|  Image width (in bits)  | 0b00000000
+-------------------------+
|  Image height (in bits) | 0b00000000
+-------------------------+
|                         | 0b00000000
|       IMAGE DATA        | 0b00000000
|     ARRAY OF BYTES      | 0b00000000
|                         | 0b00000000
+-------------------------+
```

Image info organization:

```
+-------------------------+
|      Image info         | 0b00000000
+-------------------------+   |||||||__ Image type (magic number)
                            ||||||___ Image type (magic number)
                            |||||_____
                            ||||______
                            |||_______
                            ||________
                            |_________

|  Image width (in bits)  | 0b00000000 _ 8 bits to represent the width
+-------------------------+
|  Image height (in bits) | 0b00000000 _ 8 bits to represent the height
```

Image data organization:

```
          L    L
          i    i
          n    n
          e    e
```

```
          0    1    ...
        +-------------------------------------------------------------+
        | b7 | b7 |
        | b6 | b6 |
        | b5 | b5 |
page 0  | b4 | b4 |
        | b3 | b3 |
        | b2 | b2 |
        | b1 | b1 |
        | b0 | b0 |
        +-------------------------------------------------------------+
        |
        |
page 1  |

...
x
```

Image example

Here a example os a image with 8x16 (smile face):

```
0x00, 0x08, 0x10,
0x00, 0x30, 0x30, 0x06, 0x06, 0x30, 0x30, 0x00,
0xc0, 0x20, 0x10, 0x08, 0x08, 0x20, 0x20, 0xc0
```

Here is the same image organized as will be printed into the glcd,

```
+-+-+-+-+-+-+-+-+
|0|0|0|0|0|0|0|0| -> the MSB
|0|0|0|0|0|0|0|0|
|0|1|1|0|0|1|1|0|
|0|1|1|0|0|1|1|0|
|0|0|0|0|0|0|0|0|
|0|0|0|1|1|0|0|0|
|0|0|0|1|1|0|0|0|
|0|0|0|0|0|0|0|0| -> the LSB
+-+-+-+-+-+-+-+-+
|1|0|0|0|0|0|0|1| -> the MSB
|1|0|0|0|0|0|0|1|
|0|1|0|0|0|0|1|0|
|0|0|1|0|0|1|0|0|
|0|0|0|1|1|0|0|0|
|0|0|0|0|0|0|0|0|
|0|0|0|0|0|0|0|0|
|0|0|0|0|0|0|0|0| -> the LSB
+-+-+-+-+-+-+-+-+
```

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 136 of file GlcdBitmapImage.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 GlcdBitmapImage::GlcdBitmapImage (SeekableInputStream * *inputStream*)

Public constructor.

Parameters

inputStream	The associated input stream.

Definition at line 16 of file GlcdBitmapImage.cpp.

4.2.3 Member Function Documentation

4.2.3.1 unsigned char GlcdBitmapImage::getHeight ( )

Gets the height of an image.

Returns

The image height.

Definition at line 32 of file GlcdBitmapImage.cpp.

4.2.3.2 unsigned char GlcdBitmapImage::getInfo ( )

Gets the info of an image.

Returns

The image info entry.

Definition at line 24 of file GlcdBitmapImage.cpp.

4.2.3.3 bool GlcdBitmapImage::getPixel (unsigned char *x*, unsigned char *y*) [virtual]

Reads a pixel from the image.

Parameters

x	The x position.
y	The y position.

Returns

The pixel.

Definition at line 36 of file GlcdBitmapImage.cpp.

4.2.3.4 unsigned char GlcdBitmapImage::getWidth ( )

Gets the width of an image.

Returns

The image width.

Definition at line 28 of file GlcdBitmapImage.cpp.

4.2.3.5 void GlcdBitmapImage::readColumn (unsigned char * *buf*, unsigned char *col*) [virtual]

Reads a column from the image.

Parameters

buf	The buffer to be filled with the column data.
col	The column.

Definition at line 47 of file GlcdBitmapImage.cpp.

4.2.3.6 void GlcdBitmapImage::readRow (unsigned char * *buf*, unsigned char *row*) [virtual]

Reads a row from the image.

This will reads a row with 8 bits of height.

Parameters

buf	The buffer to be filled with the column data.
row	The row.

Definition at line 56 of file GlcdBitmapImage.cpp.

4.2.4 Member Data Documentation

4.2.4.1 unsigned int GlcdBitmapImage::dataOffset [protected]

Data offset.

It is the position when the header ends.

Definition at line 157 of file GlcdBitmapImage.h.

4.2.4.2 Header GlcdBitmapImage::header [protected]

Definition at line 147 of file GlcdBitmapImage.h.

4.2.4.3 SeekableInputStream* GlcdBitmapImage::inputStream [protected]

Input stream which font data comes from.

Definition at line 152 of file GlcdBitmapImage.h.

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

- GlcdBitmapImage.h
- GlcdBitmapImage.cpp

4.3 GlcdBitmapRender Class Reference

`#include <GlcdBitmapRender.h>`

Public Member Functions

- GlcdBitmapRender (Glcd *glcd)
- void drawImage (GlcdBitmapImage *image, unsigned char x, unsigned char y)
- void drawImage (GlcdBitmapImage *image, GlcdPoint *p)
- void drawImageAtRow (GlcdBitmapImage *image, unsigned char x, unsigned char row)

Protected Attributes

- Glcd * glcd

4.3.1 Detailed Description

Arduino Graphic LCD Library.

GlcdBitmapRender.h

The header functions to draw bitmaps in a glcd plane.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 18 of file GlcdBitmapRender.h.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 GlcdBitmapRender::GlcdBitmapRender (Glcd * *glcd*)

Public constructor.

Parameters

glcd	The glcd driver instance.

Definition at line 16 of file GlcdBitmapRender.cpp.

4.3.3 Member Function Documentation

4.3.3.1 void GlcdBitmapRender::drawImage (GlcdBitmapImage * *image*, unsigned char *x*, unsigned char *y*)

Draws an image at given x, y position.

Parameters

image	The image to be drawn.
x	The x position.
y	The y position.

Definition at line 19 of file GlcdBitmapRender.cpp.

4.3.3.2 void GlcdBitmapRender::drawImage (GlcdBitmapImage * *image*, GlcdPoint * *p*) [inline]

Draws an image at given point.

Parameters

image	The image to be drawn.
p	The point.

Definition at line 50 of file GlcdBitmapRender.h.

4.3.3.3 void GlcdBitmapRender::drawImageAtRow (GlcdBitmapImage * *image*, unsigned char *x*, unsigned char *row*)

Draws an image at the row.

Parameters

image	The image to be drawn.
x	The x position.

row	The row.

Definition at line 34 of file GlcdBitmapRender.cpp.

4.3.4 Member Data Documentation

4.3.4.1 Glcd* GlcdBitmapRender::glcd [protected]

Glcd driver to render.

Definition at line 24 of file GlcdBitmapRender.h.

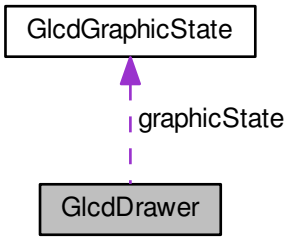
The documentation for this class was generated from the following files:

- GlcdBitmapRender.h
- GlcdBitmapRender.cpp

4.4 GlcdDrawer Class Reference

`#include <GlcdDrawer.h>`

Collaboration diagram for GlcdDrawer:



Public Member Functions

- GlcdDrawer (Glcd *glcd, GlcdGraphicState *graphicState)
- void setGlcd (Glcd *glcd)
- void setGraphicState (GlcdGraphicState *graphicState)
- Glcd * getGlcd ()
- GlcdGraphicState * getGraphicState ()
- void drawLine (unsigned char x1, unsigned char y1, unsigned char x2, unsigned char y2)
- void drawLine (GlcdPoint *p1, GlcdPoint *p2)
- void drawRectangle (unsigned char x1, unsigned char y1, unsigned char x2, unsigned char y2)
- void drawRectangle (GlcdRectangle *r)
- void drawCircle (unsigned char x, unsigned char y, unsigned char radius)
- void drawCircle (GlcdPoint *p, unsigned char radius)

Private Attributes

- Glcd * glcd
- GlcdGraphicState * graphicState

4.4.1 Detailed Description

Arduino Graphic LCD Library.

GlcdDrawer.h

The header functions to draw in a glcd plane.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 19 of file GlcdDrawer.h.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 GlcdDrawer::GlcdDrawer (Glcd * *glcd*, GlcdGraphicState * *graphicState*)

Public constructor.

Parameters

glcd	The driver instance.
graphicState	The graphic state instance.

Definition at line 16 of file GlcdDrawer.cpp.

4.4.3 Member Function Documentation

4.4.3.1 void GlcdDrawer::drawCircle (unsigned char *x*, unsigned char *y*, unsigned char *radius*)

Drawers a circle in the glcd plane.

Parameters

x	The coordinate x.
y	The coordinate y.
radius	The radius.

Definition at line 110 of file GlcdDrawer.cpp.

4.4.3.2 void GlcdDrawer::drawCircle (GlcdPoint * *p*, unsigned char *radius*) [inline]

Drawers a circle in the glcd plane.

Parameters

p	
radius	

Definition at line 124 of file GlcdDrawer.h.

4.4.3.3 void GlcdDrawer::drawLine (unsigned char *x1*, unsigned char *y1*, unsigned char *x2*, unsigned char *y2*)

Drawers a line in the glcd plane.

Parameters

x1	The x axis for the left of the rectangle.

y1	The x axis for the right of the rectangle.
x2	The y axis for the bottom of the rectangle.
y2	The y axis for the top of the rectangle.

Definition at line 35 of file GlcdDrawer.cpp.

4.4.3.4 void GlcdDrawer::drawLine (GlcdPoint * *p1*, GlcdPoint * *p2*) [inline]

Drawers a line in the glcd plane.

Parameters

p1	The point 1.
p2	The point 2.

Definition at line 86 of file GlcdDrawer.h.

4.4.3.5 void GlcdDrawer::drawRectangle (unsigned char *x1*, unsigned char *y1*, unsigned char *x2*, unsigned char *y2*)

Drawers a rectangle on the glcd plane.

Parameters

x1	The x axis for the left of the rectangle.
y1	The x axis for the right of the rectangle.
x2	The y axis for the bottom of the rectangle.
y2	The y axis for the top of the rectangle.

Definition at line 81 of file GlcdDrawer.cpp.

4.4.3.6 void GlcdDrawer::drawRectangle (GlcdRectangle * *r*) [inline]

Drawers a rectangle on the glcd plane.

Parameters

r	The rectangle.

Definition at line 105 of file GlcdDrawer.h.

4.4.3.7 Glcd * GlcdDrawer::getGlcd ( )

Gets the driver instance.

Returns

The driver instance.

Definition at line 27 of file GlcdDrawer.cpp.

4.4.3.8 GlcdGraphicState * GlcdDrawer::getGraphicState ( )

Gets the graphic state instance.

Returns

The graphic state instance.

Definition at line 31 of file GlcdDrawer.cpp.

4.4.3.9 void GlcdDrawer::setGlcd (Glcd * *glcd*)

Sets the driver instance.

Parameters

glcd	The driver instance.

Definition at line 19 of file GlcdDrawer.cpp.

4.4.3.10 void GlcdDrawer::setGraphicState (GlcdGraphicState * *graphicState*)

Sets the graphic state instance.

Parameters

graphicState	The graphic state instance.

Definition at line 23 of file GlcdDrawer.cpp.

4.4.4 Member Data Documentation

4.4.4.1 Glcd* GlcdDrawer::glcd [private]

The driver instance.

Definition at line 25 of file GlcdDrawer.h.

4.4.4.2 GlcdGraphicState* GlcdDrawer::graphicState [private]

The graphic state instance.

Definition at line 30 of file GlcdDrawer.h.

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

- GlcdDrawer.h
- GlcdDrawer.cpp

4.5 GlcdGraphicState Class Reference

`#include <GlcdGraphicState.h>`

Public Types

- enum LinePattern { SOLID_LINE = 0, DOTED_LINE = 1 }
- enum FillPattern { SOLID_FILL = 0, DOTED_FILL = 1 }

Public Member Functions

- GlcdGraphicState ()
- void setFillPattern (FillPattern fillPattern)
- FillPattern getFillPattern ()
- void setLinePattern (LinePattern linePattern)
- LinePattern getLinePattern ()
- void setLeading (unsigned char leading)
- unsigned char getLeading ()
- void setColor (Glcd::Color color)
- Glcd::Color getColor ()
- void invertColor ()
- void setSpace (unsigned char space)
- unsigned char getSpace ()
- void setFill (bool fill)
- bool getFill ()

Public Attributes

- LinePattern linePattern
- FillPattern fillPattern
- Glcd::Color color
- unsigned char leading
- unsigned char space
- bool fill

4.5.1 Detailed Description

Arduino Graphic LCD Library.

GlcdGraphicState.h

The glcd graphic state.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 16 of file GlcdGraphicState.h.

4.5.2 Member Enumeration Documentation

4.5.2.1 enum GlcdGraphicState::FillPattern

Enumerator

SOLID_FILL

DOTED_FILL

Definition at line 24 of file GlcdGraphicState.h.

4.5.2.2 enum GlcdGraphicState::LinePattern

Enumerator

SOLID_LINE

DOTED_LINE

Definition at line 19 of file GlcdGraphicState.h.

4.5.3 Constructor & Destructor Documentation

4.5.3.1 GlcdGraphicState::GlcdGraphicState ( ) [inline]

Public constructor.

Definition at line 62 of file GlcdGraphicState.h.

4.5.4 Member Function Documentation

4.5.4.1 Glcd::Color GlcdGraphicState::getColor ( )

Gets the used color.

Returns

The current used color.

Definition at line 44 of file GlcdGraphicState.cpp.

4.5.4.2 bool GlcdGraphicState::getFill ( )

Gets the fill flag.

Returns

The current fill flag.

Definition at line 74 of file GlcdGraphicState.cpp.

4.5.4.3 GlcdGraphicState::FillPattern GlcdGraphicState::getFillPattern ( )

Gets the fill pattern.

Returns

The fill pattern instance.

Definition at line 20 of file GlcdGraphicState.cpp.

4.5.4.4 unsigned char GlcdGraphicState::getLeading ( )

Gets the leading value.

Returns

The current leading value.

Definition at line 36 of file GlcdGraphicState.cpp.

4.5.4.5 GlcdGraphicState::LinePattern GlcdGraphicState::getLinePattern ( )

Gets the line pattern.

Returns

The line pattern.

Definition at line 28 of file GlcdGraphicState.cpp.

4.5.4.6 unsigned char GlcdGraphicState::getSpace ( )

Gets the space value.

Returns

The current space value.

Definition at line 66 of file GlcdGraphicState.cpp.

4.5.4.7 void GlcdGraphicState::invertColor ( )

Inverts the current color.

Definition at line 48 of file GlcdGraphicState.cpp.

4.5.4.8 void GlcdGraphicState::setColor (Glcd::Color *color*)

Sets the color.

Parameters

color	The next used color.

Definition at line 40 of file GlcdGraphicState.cpp.

4.5.4.9 void GlcdGraphicState::setFill (bool *fill*)

Sets the fill flag.

Parameters

fill	The fill flag.

Definition at line 70 of file GlcdGraphicState.cpp.

4.5.4.10 void GlcdGraphicState::setFillPattern (FillPattern *fillPattern*)

Sets fill pattern.

Parameters

fillPattern	The fill pattern instance.

Definition at line 16 of file GlcdGraphicState.cpp.

4.5.4.11 void GlcdGraphicState::setLeading (unsigned char *leading*)

Sets the leading value.

Parameters

leading	The next leading value.

Definition at line 32 of file GlcdGraphicState.cpp.

4.5.4.12 void GlcdGraphicState::setLinePattern (LinePattern *linePattern*)

Sets the lone pattern.

Parameters

linePattern	The line pattern instance.

Definition at line 24 of file GlcdGraphicState.cpp.

4.5.4.13 void GlcdGraphicState::setSpace (unsigned char *space*)

Sets the space value.

Parameters

space	The next space value.

Definition at line 62 of file GlcdGraphicState.cpp.

4.5.5 Member Data Documentation

4.5.5.1 Glcd::Color GlcdGraphicState::color

The color instance.

Definition at line 42 of file GlcdGraphicState.h.

4.5.5.2 bool GlcdGraphicState::fill

If the shapes need to be filled.

Definition at line 57 of file GlcdGraphicState.h.

4.5.5.3 FillPattern GlcdGraphicState::fillPattern

The fill pattern.

Definition at line 37 of file GlcdGraphicState.h.

4.5.5.4 unsigned char GlcdGraphicState::leading

The number of leading pixels (space between lines).

Definition at line 47 of file GlcdGraphicState.h.

4.5.5.5 LinePattern GlcdGraphicState::linePattern

The line pattern.

Definition at line 32 of file GlcdGraphicState.h.

4.5.5.6 unsigned char GlcdGraphicState::space

The number of space pixels (space between words).

Definition at line 52 of file GlcdGraphicState.h.

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

- GlcdGraphicState.h
- GlcdGraphicState.cpp

4.6 GlcdPoint Class Reference

```
#include <GlcdPoint.h>
```

Public Member Functions

- GlcdPoint ()
- GlcdPoint (unsigned char x, unsigned char y)
- void setX (unsigned char top)
- unsigned char getX ()
- void setY (unsigned char left)
- unsigned char getY ()

Private Attributes

- unsigned char x
- unsigned char y

4.6.1 Detailed Description

Arduino Graphic LCD Library.

GlcdPoint.h

A point is pixel in the glcd screen. Point has a x and y positions.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 14 of file GlcdPoint.h.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 GlcdPoint::GlcdPoint ( )

Public constructor.

Definition at line 16 of file GlcdPoint.cpp.

4.6.2.2 GlcdPoint::GlcdPoint (unsigned char *x*, unsigned char *y*)

Public constructor.

Parameters

x	The x position
y	The y position

Definition at line 20 of file GlcdPoint.cpp.

4.6.3 Member Function Documentation

4.6.3.1 unsigned char GlcdPoint::getX ( )

Gets x position.

Returns

The x position

Definition at line 29 of file GlcdPoint.cpp.

4.6.3.2 unsigned char GlcdPoint::getY ( )

Gets position y.

Returns

The y position

Definition at line 37 of file GlcdPoint.cpp.

4.6.3.3 void GlcdPoint::setX (unsigned char *top*)

Sets x position.

Parameters

top	The x position

Definition at line 25 of file GlcdPoint.cpp.

4.6.3.4 void GlcdPoint::setY (unsigned char *left*)

Sets the y position.

Parameters

left	The y position

Definition at line 33 of file GlcdPoint.cpp.

4.6.4 Member Data Documentation

4.6.4.1 unsigned char GlcdPoint::x [private]

The x position.

Definition at line 20 of file GlcdPoint.h.

4.6.4.2 unsigned char GlcdPoint::y [private]

The y position.

Definition at line 25 of file GlcdPoint.h.

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

- GlcdPoint.h
- GlcdPoint.cpp

4.7 GlcdRectangle Class Reference

`#include <GlcdRectangle.h>`

Public Member Functions

- GlcdRectangle ()
- GlcdRectangle (unsigned char left, unsigned char top, unsigned char right, unsigned char bottom)
- void setTop (unsigned char top)
- unsigned char getTop ()
- void setLeft (unsigned char left)
- unsigned char getLeft ()
- void setRight (unsigned char right)
- unsigned char getRight ()
- void setBottom (unsigned char bottom)
- unsigned char getBottom ()
- unsigned char getWidth ()
- unsigned char getHeight ()
- unsigned int getArea ()

Private Attributes

- unsigned char left
- unsigned char top
- unsigned char right
- unsigned char bottom

4.7.1 Detailed Description

Arduino Graphic LCD Library.

GlcdRectangle.h

A GlcdRectangle is rectangle in the glcd screen.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 14 of file GlcdRectangle.h.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 GlcdRectangle::GlcdRectangle ( )

Public constructor.

Definition at line 16 of file GlcdRectangle.cpp.

4.7.2.2 GlcdRectangle::GlcdRectangle (unsigned char *left*, unsigned char *top*, unsigned char *right*, unsigned char *bottom*)

Public constructor.

Parameters

left	The left position
top	The top position
right	The right position
bottom	The bottom position

Definition at line 20 of file GlcdRectangle.cpp.

4.7.3 Member Function Documentation

4.7.3.1 unsigned int GlcdRectangle::getArea ( )

Gets the area of the image.

Returns

The area

Definition at line 63 of file GlcdRectangle.cpp.

4.7.3.2 unsigned char GlcdRectangle::getBottom ( )

Gets the top position.

Returns

The bottom position

Definition at line 51 of file GlcdRectangle.cpp.

4.7.3.3 unsigned char GlcdRectangle::getHeight ( )

Gets the height of the image.

Returns

The height

Definition at line 59 of file GlcdRectangle.cpp.

4.7.3.4 unsigned char GlcdRectangle::getLeft ( )

Gets the top position.

Returns

The top position

Definition at line 35 of file GlcdRectangle.cpp.

4.7.3.5 unsigned char GlcdRectangle::getRight ( )

Gets the right position.

Returns

The right position

Definition at line 43 of file GlcdRectangle.cpp.

4.7.3.6 unsigned char GlcdRectangle::getTop ( )

Gets the top position.

Returns

The top position

Definition at line 27 of file GlcdRectangle.cpp.

4.7.3.7 unsigned char GlcdRectangle::getWidth ( )

Gets the width of the image.

Returns

The width

Definition at line 55 of file GlcdRectangle.cpp.

4.7.3.8 void GlcdRectangle::setBottom (unsigned char *bottom*)

Sets the bottom position.

Parameters

bottom	The bottom position

Definition at line 47 of file GlcdRectangle.cpp.

4.7.3.9 void GlcdRectangle::setLeft (unsigned char *left*)

Sets the left position.

Parameters

left	The left position

Definition at line 31 of file GlcdRectangle.cpp.

4.7.3.10 void GlcdRectangle::setRight (unsigned char *right*)

Sets the right position.

Parameters

right	The right position

Definition at line 39 of file GlcdRectangle.cpp.

4.7.3.11 void GlcdRectangle::setTop (unsigned char *top*)

Sets the top position.

Parameters

top	The top position

Definition at line 23 of file GlcdRectangle.cpp.

4.7.4 Member Data Documentation

4.7.4.1 unsigned char GlcdRectangle::bottom [private]

The bottom position.

Definition at line 35 of file GlcdRectangle.h.

4.7.4.2 unsigned char GlcdRectangle::left [private]

The left position.

Definition at line 20 of file GlcdRectangle.h.

4.7.4.3 unsigned char GlcdRectangle::right [private]

The right position.

Definition at line 30 of file GlcdRectangle.h.

4.7.4.4 unsigned char GlcdRectangle::top [private]

The top position.

Definition at line 25 of file GlcdRectangle.h.

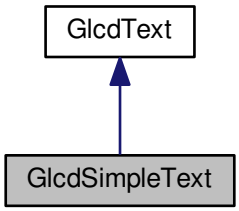
The documentation for this class was generated from the following files:

- GlcdRectangle.h
- GlcdRectangle.cpp

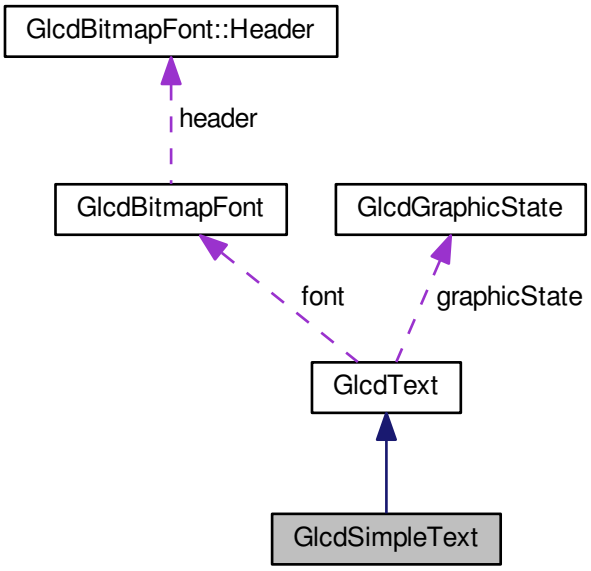
4.8 GlcdSimpleText Class Reference

```
#include <GlcdSimpleText.h>
```

Inheritance diagram for GlcdSimpleText:



Collaboration diagram for GlcdSimpleText:



Public Member Functions

- GlcdSimpleText (Glcd ∗glcd, GlcdBitmapFont ∗font, GlcdGraphicState ∗graphicState)
- virtual void printChar (unsigned char x, unsigned char y, const unsigned char c, unsigned char size)
- void printChar (GlcdPoint ∗p, const unsigned char c, unsigned char size)

Additional Inherited Members

4.8.1 Detailed Description

Arduino Graphic LCD Library.

GlcdSimpleText.h

The functions to draw text in a glcd.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 19 of file GlcdSimpleText.h.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 GlcdSimpleText::GlcdSimpleText (Glcd * *glcd,* GlcdBitmapFont * *font,* GlcdGraphicState * *graphicState*)

Public constructor.

Parameters

The	glcd driver.
The	bitmap font.
The	graphic state.

Definition at line 16 of file GlcdSimpleText.cpp.

4.8.3 Member Function Documentation

4.8.3.1 void GlcdSimpleText::printChar (unsigned char *x,* unsigned char *y,* const unsigned char *c,* unsigned char *size*) [virtual]

Prints a char at given position.

Parameters

The	x position.
The	y position.
The	char.
The	size.

Returns

void

Reimplemented from GlcdText.

Definition at line 19 of file GlcdSimpleText.cpp.

4.8.3.2 void GlcdSimpleText::printChar (GlcdPoint * *p,* const unsigned char *c,* unsigned char *size*) [inline]

Prints a char at given position.

Parameters

p	The point where the char will be printed.
c	The char.
size	The size of the char.

Definition at line 49 of file GlcdSimpleText.h.

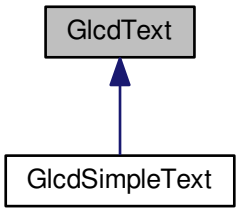
The documentation for this class was generated from the following files:

- GlcdSimpleText.h
- GlcdSimpleText.cpp

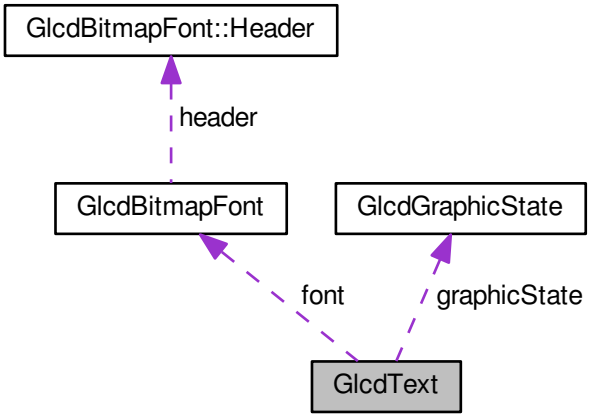
4.9 GlcdText Class Reference

`#include <GlcdText.h>`

Inheritance diagram for GlcdText:



Collaboration diagram for GlcdText:



Public Member Functions

- GlcdText (Glcd *glcd, GlcdBitmapFont *font, GlcdGraphicState *graphicState)
- void setGlcd (Glcd *glcd)
- void setFont (GlcdBitmapFont *font)
- void setGraphicState (GlcdGraphicState *graphicState)
- Glcd * getGlcd ()
- GlcdBitmapFont * getFont ()
- GlcdGraphicState * getGraphicState ()
- virtual void printChar (unsigned char x, unsigned char y, const unsigned char c, unsigned char size)
- void printChar (unsigned char x, unsigned char y, const unsigned char c)
- void printChar (GlcdPoint *p, const unsigned char c, unsigned char size)

- void printChar (GlcdPoint *p, const unsigned char c)
- virtual unsigned char printString (unsigned char left, unsigned char top, unsigned char right, unsigned char bottom, const unsigned char *text, unsigned char count, unsigned char size)
- unsigned char printString (unsigned char left, unsigned char top, unsigned char right, unsigned char bottom, const unsigned char *text, unsigned char count)
- unsigned char printString (GlcdRectangle *area, const unsigned char *text, unsigned char count, unsigned char size)
- unsigned char printString (GlcdRectangle *area, const unsigned char *text, unsigned char count)

Protected Attributes

- Glcd * glcd
- GlcdBitmapFont * font
- GlcdGraphicState * graphicState

4.9.1 Detailed Description

Arduino Graphic LCD Library.

GlcdText.h

The functions to draw text in a glcd plane.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 20 of file GlcdText.h.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 GlcdText::GlcdText (Glcd * *glcd,* GlcdBitmapFont * *font,* GlcdGraphicState * *graphicState*)

Public constructor.

Parameters

glcd	The glcd driver.
font	The font to be used.
graphicState	The graphic state instance.

Definition at line 16 of file GlcdText.cpp.

4.9.3 Member Function Documentation

4.9.3.1 GlcdBitmapFont * GlcdText::getFont ( )

Gets the font.

Returns

The font.

Definition at line 35 of file GlcdText.cpp.

4.9.3.2 Glcd * GlcdText::getGlcd ( )

Gets the glcd driver.

Returns

The glcd driver.

Definition at line 31 of file GlcdText.cpp.

4.9.3.3 GlcdGraphicState * GlcdText::getGraphicState ( )

Gets the graphic state.

Returns

The graphic state instance.

Definition at line 39 of file GlcdText.cpp.

4.9.3.4 void GlcdText::printChar (unsigned char *x*, unsigned char *y*, const unsigned char *c*, unsigned char *size*) [virtual]

Write a char on a graphic lcd.

NOTE: (x,y) is the upper left coordinate of the first letter

Parameters

x	The X position.
y	The Y position.
c	The char.
size	The size.

Reimplemented in GlcdSimpleText.

Definition at line 43 of file GlcdText.cpp.

4.9.3.5 void GlcdText::printChar (unsigned char *x*, unsigned char *y*, const unsigned char *c*) [inline]

Write a char on a graphic lcd.

Parameters

x	The X position.
y	The Y position.
c	The char.

Definition at line 110 of file GlcdText.h.

4.9.3.6 void GlcdText::printChar (GlcdPoint * *p*, const unsigned char *c*, unsigned char *size*) [inline]

Write a char on a graphic lcd.

Parameters

p	The point where the text char will be printed.
c	The char.
size	The size.

Definition at line 121 of file GlcdText.h.

4.9.3.7 void GlcdText::printChar (GlcdPoint * *p*, const unsigned char *c*) [inline]

Write a char on a graphic lcd.

Parameters

p	The point where the text char will be printed.
c	The char.

Definition at line 131 of file GlcdText.h.

4.9.3.8 unsigned char GlcdText::printString (unsigned char *left*, unsigned char *top*, unsigned char *right*, unsigned char *bottom*, const unsigned char * *text*, unsigned char *count*, unsigned char *size*) [virtual]

Write a text on a graphic lcd.

NOTE: (x,y) is the upper left coordinate of the first letter

Parameters

left	The rectangle left position.
top	The rectangle top position.
right	The rectangle right position.
bottom	The rectangle bottom position.
text	The text to be printed.
count	The maximum number of chars to print.
size	The text size.

Returns

The number of printed chars.

Definition at line 85 of file GlcdText.cpp.

4.9.3.9 unsigned char GlcdText::printString (unsigned char *left*, unsigned char *top*, unsigned char *right*, unsigned char *bottom*, const unsigned char * *text*, unsigned char *count*) [inline]

Write a text on a graphic lcd.

NOTE: (x,y) is the upper left coordinate of the first letter

Parameters

left	The rectangle left position.
top	The rectangle top position.
right	The rectangle right position.
bottom	The rectangle bottom position.
text	The text to be printed.
count	The maximum number of chars to print.

Returns

The number of printed chars.

Definition at line 164 of file GlcdText.h.

4.9.3.10 unsigned char GlcdText::printString (GlcdRectangle * *area*, const unsigned char * *text*, unsigned char *count*, unsigned char *size*) [inline]

Write a text on a graphic lcd.

NOTE: (x,y) is the upper left coordinate of the first letter

Parameters

area	The rectangle rectangle.
text	The text, must has the '\0' terminator.
count	The count.
size	The size.

Returns

The number of printed characters.

Definition at line 179 of file GlcdText.h.

4.9.3.11 unsigned char GlcdText::printString (GlcdRectangle * *area*, const unsigned char * *text*, unsigned char *count*) [inline]

Write a text on a graphic lcd.

NOTE: (x,y) is the upper left coordinate of the first letter

Parameters

area	The rectangle rectangle.
text	The text, must has the '\0' terminator.
count	The count.

Returns

The number of printed characters.

Definition at line 193 of file GlcdText.h.

4.9.3.12 void GlcdText::setFont (GlcdBitmapFont * *font*)

Sets the font.

Parameters

font	The font to be used.

Definition at line 23 of file GlcdText.cpp.

4.9.3.13 void GlcdText::setGlcd (Glcd * *glcd*)

Sets the driver.

Parameters

glcd	The glcd driver.

Definition at line 19 of file GlcdText.cpp.

4.9.3.14 void GlcdText::setGraphicState (GlcdGraphicState * *graphicState*)

Sets the graphic state.

Parameters

graphicState	The graphic state instance.

Definition at line 27 of file GlcdText.cpp.

4.9.4 Member Data Documentation

4.9.4.1 GlcdBitmapFont* GlcdText::font [protected]

The used font.

Definition at line 31 of file GlcdText.h.

4.9.4.2 Glcd* GlcdText::glcd [protected]

The Glcd driver.

Definition at line 26 of file GlcdText.h.

4.9.4.3 GlcdGraphicState* GlcdText::graphicState [protected]

The used graphic state.

Definition at line 36 of file GlcdText.h.

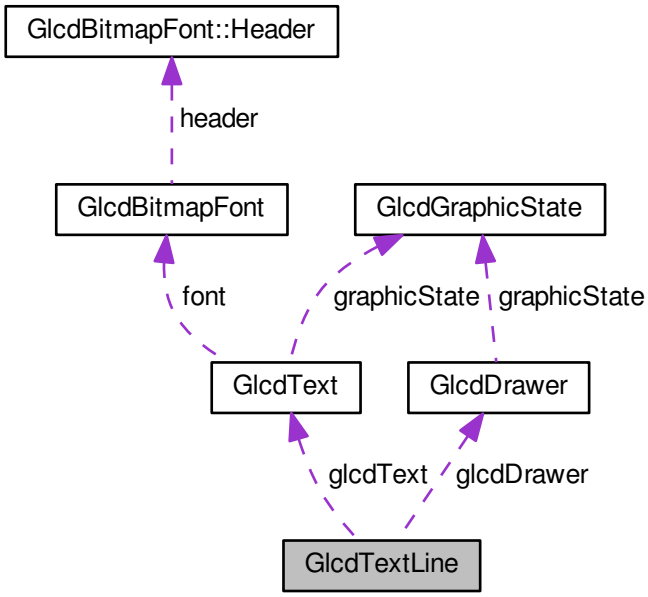
The documentation for this class was generated from the following files:

- GlcdText.h
- GlcdText.cpp

4.10 GlcdTextLine Class Reference

`#include <GlcdTextLine.h>`

Collaboration diagram for GlcdTextLine:



Public Member Functions

- GlcdTextLine (GlcdText *glcdText, GlcdDrawer *glcdDrawer, unsigned char y)
- void printLines (const unsigned char *text, unsigned char count)

Protected Attributes

- GlcdText * glcdText
- GlcdDrawer * glcdDrawer
- unsigned char y

4.10.1 Detailed Description

Arduino Graphic LCD Library.

GlcdTextLine.h

The header functions to draw text in a glcd always at the same line.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 19 of file GlcdTextLine.h.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 GlcdTextLine::GlcdTextLine (GlcdText * *glcdText*, GlcdDrawer * *glcdDrawer*, unsigned char *y*)

Public constructor.

Parameters

glcdText	The glcd text instance.
glcdDrawer	The glcd drawer instance.
y	The y position.

Definition at line 16 of file GlcdTextLine.cpp.

4.10.3 Member Function Documentation

4.10.3.1 void GlcdTextLine::printLines (const unsigned char * *text*, unsigned char *count*)

Prints lines.

Parameters

text	The text to be printed.
count	void

Definition at line 22 of file GlcdTextLine.cpp.

4.10.4 Member Data Documentation

4.10.4.1 GlcdDrawer* GlcdTextLine::glcdDrawer [protected]

The glcd drawer.

Definition at line 30 of file GlcdTextLine.h.

4.10.4.2 GlcdText* GlcdTextLine::glcdText [protected]

The glcd text.

Definition at line 25 of file GlcdTextLine.h.

4.10.4.3 unsigned char GlcdTextLine::y [protected]

The y position.

Definition at line 35 of file GlcdTextLine.h.

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

- GlcdTextLine.h
- GlcdTextLine.cpp

4.11 GlcdBitmapFont::Header Struct Reference

`#include <GlcdBitmapFont.h>`

Public Attributes

- unsigned char info
- unsigned char characterWidth
- unsigned char characterHeight
- unsigned char sequenceCount

4.11.1 Detailed Description

Font header.

Definition at line 167 of file GlcdBitmapFont.h.

4.11.2 Member Data Documentation

4.11.2.1 unsigned char GlcdBitmapFont::Header::characterHeight

Definition at line 170 of file GlcdBitmapFont.h.

4.11.2.2 unsigned char GlcdBitmapFont::Header::characterWidth

Definition at line 169 of file GlcdBitmapFont.h.

4.11.2.3 unsigned char GlcdBitmapFont::Header::info

Definition at line 168 of file GlcdBitmapFont.h.

4.11.2.4 unsigned char GlcdBitmapFont::Header::sequenceCount

Definition at line 171 of file GlcdBitmapFont.h.

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

- GlcdBitmapFont.h

4.12 GlcdBitmapImage::Header Struct Reference

`#include <GlcdBitmapImage.h>`

Public Attributes

- unsigned char info
- unsigned char width
- unsigned char height

4.12.1 Detailed Description

Image header.

Definition at line 142 of file GlcdBitmapImage.h.

4.12.2 Member Data Documentation

4.12.2.1 unsigned char GlcdBitmapImage::Header::height

Definition at line 145 of file GlcdBitmapImage.h.

4.12.2.2 unsigned char GlcdBitmapImage::Header::info

Definition at line 143 of file GlcdBitmapImage.h.

4.12.2.3 unsigned char GlcdBitmapImage::Header::width

Definition at line 144 of file GlcdBitmapImage.h.

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

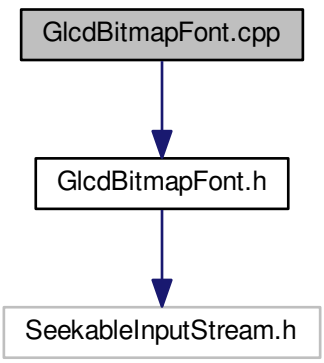
- GlcdBitmapImage.h

5 File Documentation

5.1 GlcdBitmapFont.cpp File Reference

```
#include "GlcdBitmapFont.h"
```

Include dependency graph for GlcdBitmapFont.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_BITMAP_FONT_CPP__ 1

5.1.1 Macro Definition Documentation

5.1.1.1 #define __ARDUINO_LIBRARY_GLCD_BITMAP_FONT_CPP__ 1

Arduino Graphic LCD Library.

GlcdBitmapFont.cpp

The representation of a glcd font.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdBitmapFont.cpp.

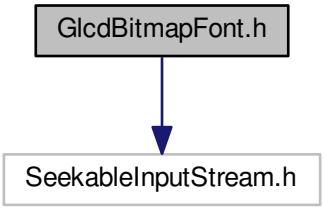
5.2 GlcdBitmapFont.cpp

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_BITMAP_FONT_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_BITMAP_FONT_CPP__ 1
00013
00014 #include "GlcdBitmapFont.h"
00015
00016 GlcdBitmapFont::GlcdBitmapFont(SeekableInputStream* inputStream) :
      inputStream(inputStream) {
00017     dataOffset = sizeof (Header);
00018     inputStream->seek(0);
00019     header.info = inputStream->read();
00020     header.characterWidth = inputStream->read();
00021     header.characterHeight = inputStream->read();
00022     header.sequenceCount = inputStream->read();
00023     glyphLength = header.characterWidth * (header.
      characterHeight / 8);
00024 }
00025
00026 unsigned char GlcdBitmapFont::getInfo() {
00027     return header.info;
00028 }
00029
00030 unsigned char GlcdBitmapFont::getCharacterWidth() {
00031     return header.characterWidth;
00032 }
00033
00034 unsigned char GlcdBitmapFont::getCharacterHeight() {
00035     return header.characterHeight;
00036 }
00037
00038 unsigned char GlcdBitmapFont::getSequenceCount() {
00039     return header.sequenceCount;
00040 }
00041
00042 unsigned char GlcdBitmapFont::getGlyphLength() {
00043     return glyphLength;
00044 }
00045
00046 unsigned char GlcdBitmapFont::readGlyphData(unsigned char *buf, char c) {
00047     unsigned int offset = getGlyphOffset(c);
00048     if (offset == 0) {
00049         return 0;
00050     }
00051     inputStream->seek(offset);
00052     return (unsigned char) inputStream->read(buf, 0, getGlyphLength());
00053 }
00054
00055 unsigned int GlcdBitmapFont::getGlyphOffset(char c) {
00056     unsigned char i, first, last;
00057     unsigned int offset = 0;
00058     inputStream->seek(dataOffset);
00059     for (i = 0; i < getSequenceCount(); i++) {
00060         first = inputStream->read();
00061         last = inputStream->read();
```

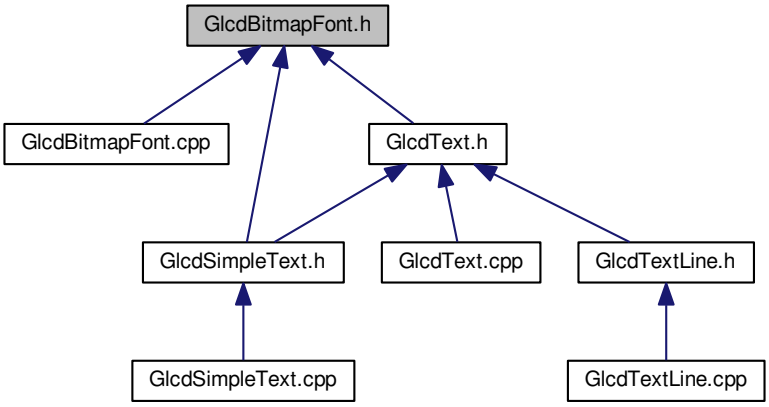
```
00062         if (c >= first && c <= last) {
00063             offset = inputStream->read();
00064             offset <<= 8;
00065             offset |= inputStream->read();
00066             offset += (c - first) * getGlyphLength();
00067             break;
00068         } else {
00069             inputStream->skip(2);
00070         }
00071     }
00072     return offset;
00073 }
00074
00075 #endif /* __ARDUINO_LIBRARY_GLCD_BITMAP_FONT_CPP__ */
```

5.3 GlcdBitmapFont.h File Reference

`#include <SeekableInputStream.h>`
Include dependency graph for GlcdBitmapFont.h:



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



Classes

- class GlcdBitmapFont
- struct GlcdBitmapFont::Header

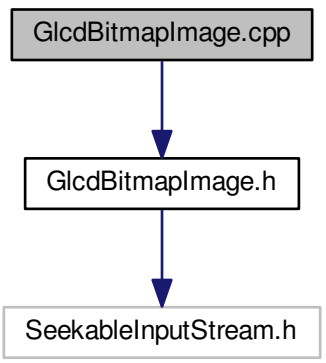
5.4 GlcdBitmapFont.h

```
00001
00156 #ifndef __ARDUINO_LIBRARY_GLCD_BITMAP_FONT_H__
00157 #define __ARDUINO_LIBRARY_GLCD_BITMAP_FONT_H__ 1
00158
00159 #include <SeekableInputStream.h>
00160
00161 class GlcdBitmapFont {
00162 protected:
00163
00167     struct Header {
00168         unsigned char info;
00169         unsigned char characterWidth;
00170         unsigned char characterHeight;
00171         unsigned char sequenceCount;
00172     };
00173     Header header;
00174
00178     unsigned char glyphLength;
00179
00183     SeekableInputStream* inputStream;
00184
00188     unsigned int dataOffset;
00189
00190 public:
00191
00197     GlcdBitmapFont(SeekableInputStream* inputStream);
00198
00204     unsigned char getInfo();
00205
00211     unsigned char getCharacterWidth();
00212
00218     unsigned char getCharacterHeight();
00219
00225     unsigned char getSequenceCount();
00226
00232     unsigned char getGlyphLength();
00233
00241     virtual unsigned char readGlyphData(unsigned char *buf, char c);
00242
00243 protected:
00244
00251     virtual unsigned int getGlyphOffset(char c);
00252 };
00253
00254 #endif /* __ARDUINO_LIBRARY_GLCD_BITMAP_FONT_H__ */
```

5.5 GlcdBitmapImage.cpp File Reference

`#include "GlcdBitmapImage.h"`

Include dependency graph for GlcdBitmapImage.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_BITMAP_IMAGE_CPP__ 1

5.5.1 Macro Definition Documentation

5.5.1.1 #define __ARDUINO_LIBRARY_GLCD_BITMAP_IMAGE_CPP__ 1

Arduino Graphic LCD Library.

GlcdBitmapImage.cpp

The representation of a glcd image.

Author

Dalmir da Silva dalmirdasilva@gmail.com

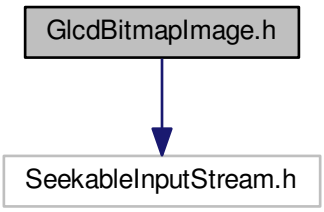
Definition at line 12 of file GlcdBitmapImage.cpp.

5.6 GlcdBitmapImage.cpp

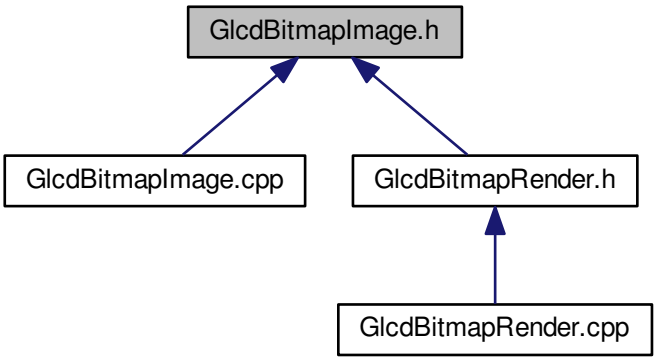
```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_BITMAP_IMAGE_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_BITMAP_IMAGE_CPP__ 1
00013
00014 #include "GlcdBitmapImage.h"
00015
00016 GlcdBitmapImage::GlcdBitmapImage(SeekableInputStream* inputStream) :
      inputStream(inputStream) {
00017     dataOffset = sizeof (Header);
00018     inputStream->seek(0);
00019     header.info = inputStream->read();
00020     header.width = inputStream->read();
00021     header.height = inputStream->read();
00022 }
00023
00024 unsigned char GlcdBitmapImage::getInfo() {
00025     return header.info;
00026 }
00027
00028 unsigned char GlcdBitmapImage::getWidth() {
00029     return header.width;
00030 }
00031
00032 unsigned char GlcdBitmapImage::getHeight() {
00033     return header.height;
00034 }
00035
00036 bool GlcdBitmapImage::getPixel(unsigned char x, unsigned char y) {
00037     unsigned int position;
00038     unsigned char v;
00039     position = (y / 8) * getWidth();
00040     y %= 8;
00041     position += x;
00042     inputStream->seek(dataOffset + position);
00043     v = inputStream->read();
00044     return v & (0x80 >> y);
00045 }
00046
00047 void GlcdBitmapImage::readColumn(unsigned char* buf, unsigned char col) {
00048     unsigned char i, rows = getHeight() / 8;
00049     inputStream->seek(dataOffset + col);
00050     for (i = 0; i < rows; i++) {
00051         buf[i] = inputStream->read();
00052         inputStream->skip(getWidth() - 1);
00053     }
00054 }
00055
00056 void GlcdBitmapImage::readRow(unsigned char* buf, unsigned char row) {
00057     inputStream->seek(dataOffset + row * getWidth());
00058     inputStream->read(buf, 0, getWidth());
00059 }
00060
00061 #endif /* __ARDUINO_LIBRARY_GLCD_BITMAP_IMAGE_CPP__ */
```

5.7 GlcdBitmapImage.h File Reference

`#include <SeekableInputStream.h>`
Include dependency graph for GlcdBitmapImage.h:



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



Classes

- class GlcdBitmapImage
- struct GlcdBitmapImage::Header

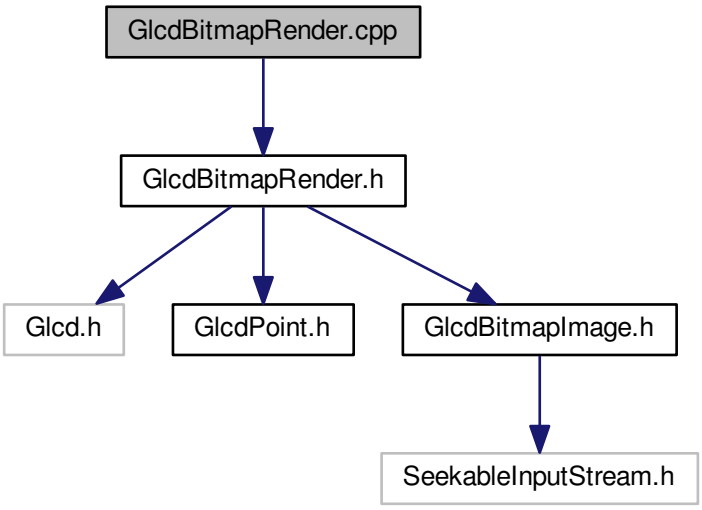
5.8 GlcdBitmapImage.h

```
00001
00131 #ifndef __ARDUINO_LIBRARY_GLCD_BITMAP_IMAGE_H__
00132 #define __ARDUINO_LIBRARY_GLCD_BITMAP_IMAGE_H__ 1
00133
00134 #include <SeekableInputStream.h>
00135
00136 class GlcdBitmapImage {
00137 protected:
00138
00142     struct Header {
00143         unsigned char info;
```

```
00144         unsigned char width;
00145         unsigned char height;
00146     };
00147     Header header;
00148
00152     SeekableInputStream* inputStream;
00153
00157     unsigned int dataOffset;
00158
00159 public:
00160
00166     GlcdBitmapImage(SeekableInputStream* inputStream);
00167
00173     unsigned char getInfo();
00174
00180     unsigned char getWidth();
00181
00187     unsigned char getHeight();
00188
00196     virtual bool getPixel(unsigned char x, unsigned char y);
00197
00204     virtual void readColumn(unsigned char* buf, unsigned char col);
00205
00214     virtual void readRow(unsigned char* buf, unsigned char row);
00215 };
00216
00217 #endif /* __ARDUINO_LIBRARY_GLCD_BITMAP_IMAGE_H__ */
```

5.9 GlcdBitmapRender.cpp File Reference

`#include "GlcdBitmapRender.h"`
Include dependency graph for GlcdBitmapRender.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_BITMAP_DRAWER_CPP__ 1

5.9.1 Macro Definition Documentation

5.9.1.1 #define __ARDUINO_LIBRARY_GLCD_BITMAP_DRAWER_CPP__ 1

Arduino Graphic LCD Library.

GlcdBitmapRender.cpp

The implementation of functions to draw bitmaps in a glcd plane.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdBitmapRender.cpp.

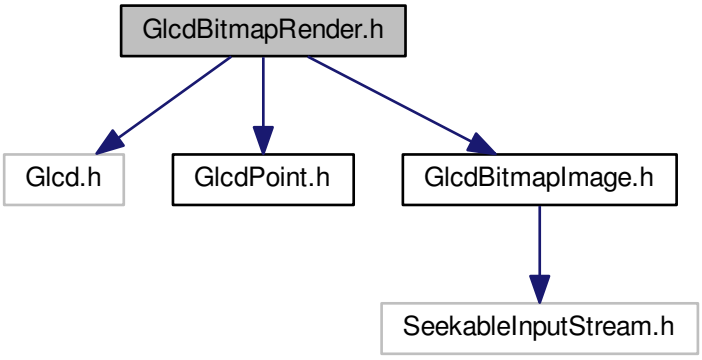
5.10 GlcdBitmapRender.cpp

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_BITMAP_DRAWER_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_BITMAP_DRAWER_CPP__ 1
00013
00014 #include "GlcdBitmapRender.h"
00015
00016 GlcdBitmapRender::GlcdBitmapRender(Glcd *glcd) : glcd(glcd) {
00017 }
00018
00019 void GlcdBitmapRender::drawImage(GlcdBitmapImage* image, unsigned
       char x, unsigned char y) {
00020     unsigned char width = image->getWidth();
00021     unsigned char rows, streak, i, j, k, buf[width];
00022     rows = image->getHeight() / 8;
00023     for (i = 0; i < rows; i++) {
00024         image->readRow(buf, i);
00025         for (j = 0; j < width; j++) {
00026             streak = buf[j];
00027             for (k = 0; k < 8; k++) {
00028                 glcd->plot(x + j, (i * 8) + k + y, (Glcd::Color)(streak & (0x01 << k)));
00029             }
00030         }
00031     }
00032 }
00033
00034 void GlcdBitmapRender::drawImageAtRow(
      GlcdBitmapImage* image, unsigned char x, unsigned char row) {
00035     unsigned char width = image->getWidth();
00036     unsigned char rows, i, j, buf[width];
00037     rows = image->getHeight() / 8;
00038     for (i = 0; i < rows; i++) {
00039         image->readRow(buf, i);
00040         for (j = 0; j < width; j++) {
00041             glcd->streak(x + j, row + i, buf[j]);
00042         }
00043     }
00044 }
00045
00046 #endif /* __ARDUINO_LIBRARY_GLCD_BITMAP_DRAWER_CPP__ */
```

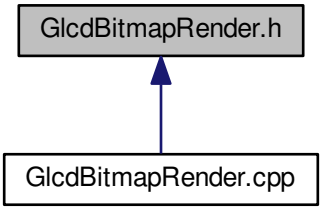
5.11 GlcdBitmapRender.h File Reference

```
#include <Glcd.h>
#include <GlcdPoint.h>
#include <GlcdBitmapImage.h>
```

Include dependency graph for GlcdBitmapRender.h:



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



Classes

- class GlcdBitmapRender

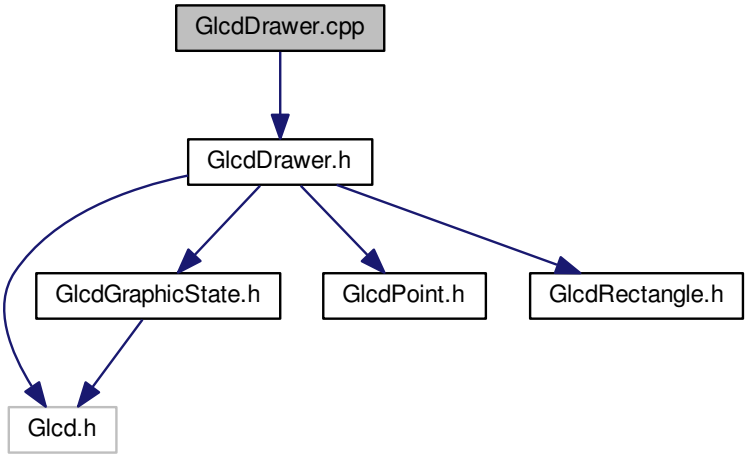
5.12 GlcdBitmapRender.h

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_BITMAP_DRAWER_H__
00012 #define __ARDUINO_LIBRARY_GLCD_BITMAP_DRAWER_H__ 1
00013
00014 #include <Glcd.h>
00015 #include <GlcdPoint.h>
00016 #include <GlcdBitmapImage.h>
00017
00018 class GlcdBitmapRender {
00019 protected:
00020
00024     Glcd *glcd;
00025
00026 public:
00027
00033     GlcdBitmapRender(Glcd *glcd);
00034
```

```
00042     void drawImage(GlcdBitmapImage *image, unsigned char x, unsigned char y);
00043
00050     inline void drawImage(GlcdBitmapImage *image,
      GlcdPoint *p) {
00051         drawImage(image, p->getX(), p->getY());
00052     }
00053
00061     void drawImageAtRow(GlcdBitmapImage *image, unsigned char x, unsigned char
       row);
00062 };
00063
00064 #endif /* __ARDUINO_LIBRARY_GLCD_BITMAP_DRAWER_H__ */
```

5.13 GlcdDrawer.cpp File Reference

`#include "GlcdDrawer.h"`
Include dependency graph for GlcdDrawer.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_DRAWER_CPP__ 1

5.13.1 Macro Definition Documentation

5.13.1.1 #define __ARDUINO_LIBRARY_GLCD_DRAWER_CPP__ 1

Arduino Graphic LCD Library.

GlcdDrawer.cpp

The header functions to draw in a glcd plane.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdDrawer.cpp.

5.14 GlcdDrawer.cpp

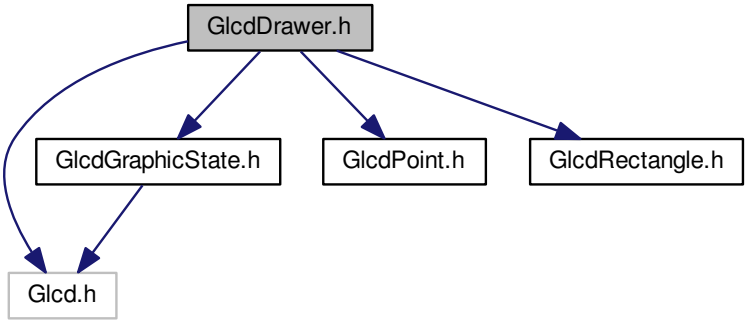
```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_DRAWER_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_DRAWER_CPP__ 1
00013
00014 #include "GlcdDrawer.h"
00015
00016 GlcdDrawer::GlcdDrawer(Glcd *glcd, GlcdGraphicState *graphicState) :
      glcd(glcd), graphicState(graphicState) {
00017 }
00018
00019 void GlcdDrawer::setGlcd(Glcd *glcd) {
00020     this->glcd = glcd;
00021 }
00022
00023 void GlcdDrawer::setGraphicState(GlcdGraphicState *graphicState)
       {
00024     this->graphicState = graphicState;
00025 }
00026
00027 Glcd *GlcdDrawer::getGlcd() {
00028     return glcd;
00029 }
00030
00031 GlcdGraphicState *GlcdDrawer::getGraphicState() {
00032     return graphicState;
00033 }
00034
00035 void GlcdDrawer::drawLine(unsigned char x1, unsigned char y1, unsigned char x2,
      unsigned char y2) {
00036
00037     unsigned char dx, dy;
00038     char sx, sy;
00039     int P, P0;
00040
00041     if (x2 > x1) {
00042         dx = x2 - x1;
00043         sx = 1;
00044     } else {
00045         dx = x1 - x2;
00046         sx = -1;
00047     }
00048
00049     if (y2 > y1) {
00050         dy = y2 - y1;
00051         sy = 1;
00052     } else {
00053         dy = y1 - y2;
00054         sy = -1;
00055     }
00056
00057     P = (dx > dy ? dx : -dy) / 2;
00058
00059     while (1) {
00060
00061         glcd->plot(x1, y1, graphicState->getColor());
00062
00063         if (x1 == x2 && y1 == y2) {
00064             break;
00065         }
00066
00067         P0 = P;
00068
00069         if (P0 > -dx) {
00070             P -= dy;
00071             x1 += sx;
00072         }
00073
00074         if (P0 < dy) {
00075             P += dx;
00076             y1 += sy;
00077         }
00078     }
00079 }
00080
00081 void GlcdDrawer::drawRectangle(unsigned char x1, unsigned char y1, unsigned char
      x2, unsigned char y2) {
00082
00083     if (graphicState->getFill()) {
00084
00085         // Find the y min and max
00086         unsigned char yMin, yMax;
00087
00088         if (y1 < y2) {
00089             yMin = y1;
00090             yMax = y2;
```

```
00091         } else {
00092             yMin = y2;
00093             yMax = y1;
00094         }
00095 
00096         // Drawer lines to fill the rectangle
00097         for (; yMin <= yMax; yMin++) {
00098             drawLine(x1, yMin, x2, yMin);
00099         }
00100     } else {
00101 
00102         // Drawer the 4 sides
00103         drawLine(x1, y1, x2, y1);
00104         drawLine(x1, y2, x2, y2);
00105         drawLine(x1, y1, x1, y2);
00106         drawLine(x2, y1, x2, y2);
00107     }
00108 }
00109 
00110 void GlcdDrawer::drawCircle(unsigned char x, unsigned char y, unsigned char radius) {
00111 
00112     int P = 1 - radius;
00113     unsigned char a = 0;
00114     unsigned char b = radius;
00115 
00116     // To fit the glcd screen
00117     if (x < radius) {
00118         x = radius;
00119     } else if ((GLCD_WIDTH - 1 - x) < radius) {
00120         x = GLCD_WIDTH - 1 - radius;
00121     }
00122 
00123     if (y < radius) {
00124         y = radius;
00125     } else if ((GLCD_HEIGHT - 1 - y) < radius) {
00126         y = GLCD_HEIGHT - 1 - radius;
00127     }
00128 
00129     do {
00130 
00131         // Fill
00132         if (graphicState->getFill()) {
00133 
00134             drawLine(x - a, y + b, x + a, y + b);
00135             drawLine(x - a, y - b, x + a, y - b);
00136             drawLine(x - b, y + a, x + b, y + a);
00137             drawLine(x - b, y - a, x + b, y - a);
00138 
00139         // Stroke
00140         } else {
00141 
00142             glcd->plot(a + x, b + y, graphicState->getColor());
00143             glcd->plot(b + x, a + y, graphicState->getColor());
00144             glcd->plot(x - a, b + y, graphicState->getColor());
00145             glcd->plot(x - b, a + y, graphicState->getColor());
00146             glcd->plot(b + x, y - a, graphicState->getColor());
00147             glcd->plot(a + x, y - b, graphicState->getColor());
00148             glcd->plot(x - a, y - b, graphicState->getColor());
00149             glcd->plot(x - b, y - a, graphicState->getColor());
00150         }
00151 
00152         if (P < 0) {
00153             P += 3 + 2 * a++;
00154         } else {
00155             P += 5 + 2 * (a++ - b--);
00156         }
00157     } while (a <= b);
00158 }
00159 
00160 #endif /* __ARDUINO_LIBRARY_GLCD_DRAWER_CPP__ */
```

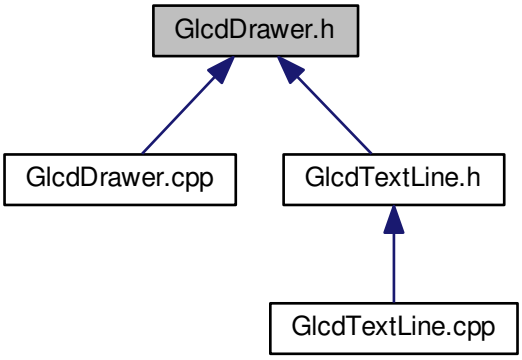
5.15 GlcdDrawer.h File Reference

```
#include <Glcd.h>
#include <GlcdGraphicState.h>
#include <GlcdPoint.h>
#include <GlcdRectangle.h>
```

Include dependency graph for GlcdDrawer.h:



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



Classes

- class GlcdDrawer

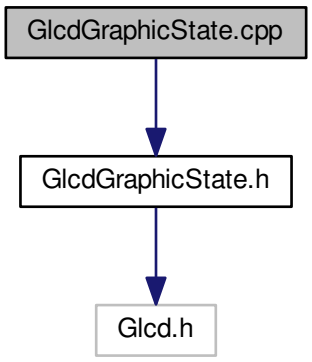
5.16 GlcdDrawer.h

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_DRAWER_H__
00012 #define __ARDUINO_LIBRARY_GLCD_DRAWER_H__ 1
00013
00014 #include <Glcd.h>
00015 #include <GlcdGraphicState.h>
00016 #include <GlcdPoint.h>
00017 #include <GlcdRectangle.h>
00018
00019 class GlcdDrawer {
00020 private:
00021
00025     Glcd *glcd;
```

```
00026
00030     GlcdGraphicState *graphicState;
00031
00032 public:
00033
00040     GlcdDrawer(Glcd *glcd, GlcdGraphicState *graphicState);
00041
00047     void setGlcd(Glcd *glcd);
00048
00054     void setGraphicState(GlcdGraphicState *graphicState);
00055
00061     Glcd *getGlcd();
00062
00068     GlcdGraphicState *getGraphicState();
00069
00078     void drawLine(unsigned char x1, unsigned char y1, unsigned char x2, unsigned char y2);
00079
00086     inline void drawLine(GlcdPoint *p1, GlcdPoint *p2) {
00087         drawLine(p1->getX(), p1->getY(), p2->getX(), p2->
      getY());
00088     }
00089
00098     void drawRectangle(unsigned char x1, unsigned char y1, unsigned char x2, unsigned char y2)
      ;
00099
00105     inline void drawRectangle(GlcdRectangle *r) {
00106         drawRectangle(r->getLeft(), r->getTop(), r->
      getRight(), r->getBottom());
00107     }
00108
00116     void drawCircle(unsigned char x, unsigned char y, unsigned char radius);
00117
00124     inline void drawCircle(GlcdPoint *p, unsigned char radius) {
00125         drawCircle(p->getX(), p->getY(), radius);
00126     }
00127 };
00128
00129 #endif /* __ARDUINO_LIBRARY_GLCD_DRAWER_H__ */
```

5.17 GlcdGraphicState.cpp File Reference

`#include "GlcdGraphicState.h"`
Include dependency graph for GlcdGraphicState.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_GRAPHIC_STATE_CPP__ 1

5.17.1 Macro Definition Documentation

5.17.1.1 #define __ARDUINO_LIBRARY_GLCD_GRAPHIC_STATE_CPP__ 1

Arduino Graphic LCD Library.

GlcdGraphicState.cpp

The glcd graphic state.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdGraphicState.cpp.

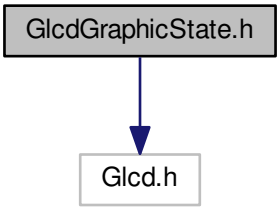
5.18 GlcdGraphicState.cpp

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_GRAPHIC_STATE_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_GRAPHIC_STATE_CPP__ 1
00013
00014 #include "GlcdGraphicState.h"
00015
00016 void GlcdGraphicState::setFillPattern(
      FillPattern fillPattern) {
00017     this->fillPattern = fillPattern;
00018 }
00019
00020 GlcdGraphicState::FillPattern
      GlcdGraphicState::getFillPattern() {
00021     return fillPattern;
00022 }
00023
00024 void GlcdGraphicState::setLinePattern(
      LinePattern linePattern) {
00025     this->linePattern = linePattern;
00026 }
00027
00028 GlcdGraphicState::LinePattern
      GlcdGraphicState::getLinePattern() {
00029     return linePattern;
00030 }
00031
00032 void GlcdGraphicState::setLeading(unsigned char leading) {
00033     this->leading = leading;
00034 }
00035
00036 unsigned char GlcdGraphicState::getLeading() {
00037     return leading;
00038 }
00039
00040 void GlcdGraphicState::setColor(Glcd::Color color) {
00041     this->color = color;
00042 }
00043
00044 Glcd::Color GlcdGraphicState::getColor() {
00045     return color;
00046 }
00047
00048 void GlcdGraphicState::invertColor() {
00049     color = (Glcd::Color) ~color;
00050     /*
00051     switch(color) {
00052         case Glcd::COLOR_BLACK:
00053             color = Glcd::COLOR_WHITE;
00054         break;
00055         case Glcd::COLOR_WHITE:
00056             color = Glcd::COLOR_BLACK;
00057         break;
00058     }
00059     */
00060 }
00061
00062 void GlcdGraphicState::setSpace(unsigned char space) {
00063     this->space = space;
00064 }
00065
00066 unsigned char GlcdGraphicState::getSpace() {
```

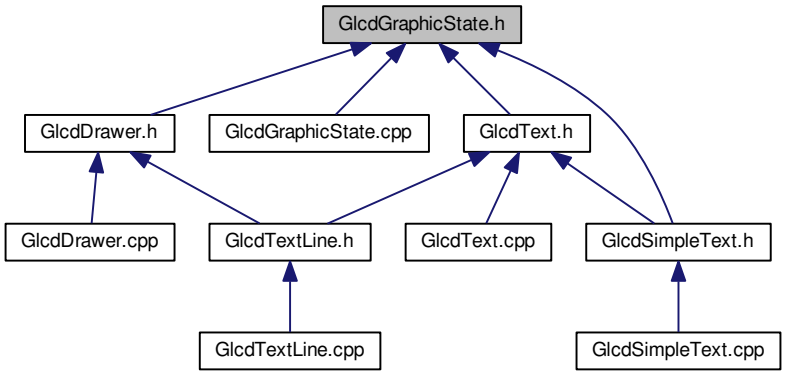
```
00067     return space;
00068 }
00069
00070 void GlcdGraphicState::setFill(bool fill) {
00071     this->fill = fill;
00072 }
00073
00074 bool GlcdGraphicState::getFill() {
00075     return fill;
00076 }
00077
00078 #endif /* __ARDUINO_LIBRARY_GLCD_GRAPHIC_STATE_CPP__ */
```

5.19 GlcdGraphicState.h File Reference

`#include <Glcd.h>`
Include dependency graph for GlcdGraphicState.h:



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



Classes

- class GlcdGraphicState

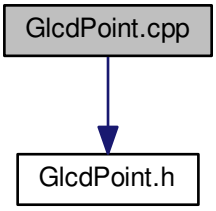
5.20 GlcdGraphicState.h

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_GRAPHIC_STATE_H__
00012 #define __ARDUINO_LIBRARY_GLCD_GRAPHIC_STATE_H__ 1
00013
00014 #include <Glcd.h>
00015
00016 class GlcdGraphicState {
00017 public:
00018
00019     enum LinePattern {
00020         SOLID_LINE = 0,
00021         DOTED_LINE = 1
00022     };
00023
00024     enum FillPattern {
00025         SOLID_FILL = 0,
00026         DOTED_FILL = 1
00027     };
00028
00032     LinePattern linePattern;
00033
00037     FillPattern fillPattern;
00038
00042     Glcd::Color color;
00043
00047     unsigned char leading;
00048
00052     unsigned char space;
00053
00057     bool fill;
00058
00062     GlcdGraphicState() {
00063         fillPattern = SOLID_FILL;
00064         linePattern = SOLID_LINE;
00065         color = Glcd::COLOR_BLACK;
00066         leading = 1;
00067         space = 1;
00068         fill = true;
00069     }
00070
00076     void setFillPattern(FillPattern fillPattern);
00077
00083     FillPattern getFillPattern();
00084
00090     void setLinePattern(LinePattern linePattern);
00091
00097     LinePattern getLinePattern();
00098
00104     void setLeading(unsigned char leading);
00105
00111     unsigned char getLeading();
00112
00118     void setColor(Glcd::Color color);
00119
00125     Glcd::Color getColor();
00126
00130     void invertColor();
00131
00137     void setSpace(unsigned char space);
00138
00144     unsigned char getSpace();
00145
00151     void setFill(bool fill);
00152
00158     bool getFill();
00159 };
00160
00161 #endif /* __ARDUINO_LIBRARY_GLCD_GRAPHIC_STATE_H__ */
```

5.21 GlcdPoint.cpp File Reference

```
#include "GlcdPoint.h"
```

Include dependency graph for GlcdPoint.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_POINT_CPP__ 1

5.21.1 Macro Definition Documentation

5.21.1.1 #define __ARDUINO_LIBRARY_GLCD_POINT_CPP__ 1

Arduino Graphic LCD Library.

GlcdPoint.cpp

A point is pixel in the glcd screen. Point has a x and y positions.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdPoint.cpp.

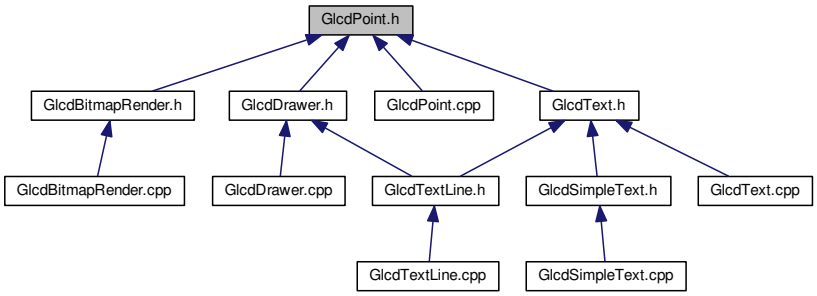
5.22 GlcdPoint.cpp

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_POINT_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_POINT_CPP__ 1
00013
00014 #include "GlcdPoint.h"
00015
00016 GlcdPoint::GlcdPoint() {
00017     GlcdPoint(0, 0);
00018 }
00019
00020 GlcdPoint::GlcdPoint(unsigned char x, unsigned char y) {
00021     this->x = x;
00022     this->y = y;
00023 }
00024
00025 void GlcdPoint::setX(unsigned char x) {
00026     this->x = x;
00027 }
00028
00029 unsigned char GlcdPoint::getX() {
00030     return x;
00031 }
00032
00033 void GlcdPoint::setY(unsigned char y) {
00034     this->y = y;
00035 }
00036
00037 unsigned char GlcdPoint::getY() {
```

```
00038     return y;
00039 }
00040
00041 #endif /* __ARDUINO_LIBRARY_GLCD_POINT_CPP__ */
```

5.23 GlcdPoint.h File Reference

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



Classes

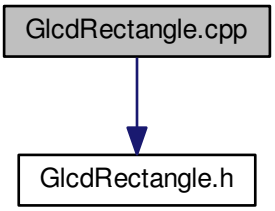
- class GlcdPoint

5.24 GlcdPoint.h

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_POINT_H__
00012 #define __ARDUINO_LIBRARY_GLCD_POINT_H__ 1
00013
00014 class GlcdPoint {
00015 private:
00016
00020     unsigned char x;
00021
00025     unsigned char y;
00026 public:
00027
00031     GlcdPoint();
00032
00039     GlcdPoint(unsigned char x, unsigned char y);
00040
00046     void setX(unsigned char top);
00047
00053     unsigned char getX();
00054
00060     void setY(unsigned char left);
00061
00067     unsigned char getY();
00068 };
00069
00070 #endif /* __ARDUINO_LIBRARY_GLCD_POINT_H__ */
```

5.25 GlcdRectangle.cpp File Reference

`#include "GlcdRectangle.h"`
Include dependency graph for GlcdRectangle.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_RECTANGLE_CPP__ 1

5.25.1 Macro Definition Documentation

5.25.1.1 #define __ARDUINO_LIBRARY_GLCD_RECTANGLE_CPP__ 1

Arduino Graphic LCD Library.

GlcdRectangle.cpp

A GlcdRectangle is rectangle in the glcd screen.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdRectangle.cpp.

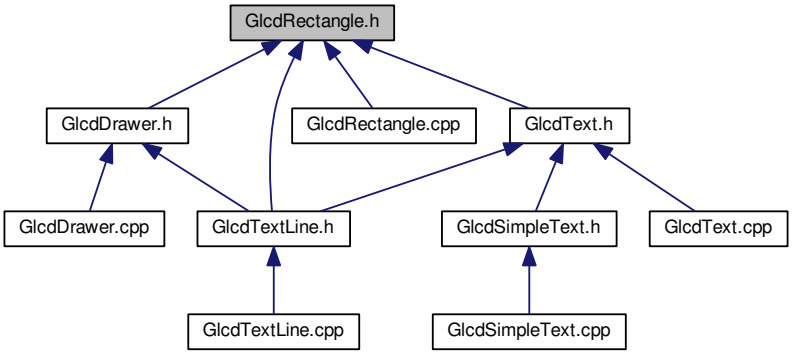
5.26 GlcdRectangle.cpp

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_RECTANGLE_CPP_
00012 #define __ARDUINO_LIBRARY_GLCD_RECTANGLE_CPP__ 1
00013
00014 #include "GlcdRectangle.h"
00015
00016 GlcdRectangle::GlcdRectangle() {
00017     GlcdRectangle(0, 0, 0, 0);
00018 }
00019
00020 GlcdRectangle::GlcdRectangle(unsigned char left, unsigned char top, unsigned
      char right, unsigned char bottom) : top(top), left(left), right(right), bottom(bottom) {
00021 }
00022
00023 void GlcdRectangle::setTop(unsigned char top) {
00024     this->top = top;
00025 }
00026
00027 unsigned char GlcdRectangle::getTop() {
00028     return top;
00029 }
00030
00031 void GlcdRectangle::setLeft(unsigned char left) {
```

```
00032     this->left = left;
00033 }
00034
00035 unsigned char GlcdRectangle::getLeft() {
00036     return left;
00037 }
00038
00039 void GlcdRectangle::setRight(unsigned char right) {
00040     this->right = right;
00041 }
00042
00043 unsigned char GlcdRectangle::getRight() {
00044     return right;
00045 }
00046
00047 void GlcdRectangle::setBottom(unsigned char bottom) {
00048     this->bottom = bottom;
00049 }
00050
00051 unsigned char GlcdRectangle::getBottom() {
00052     return bottom;
00053 }
00054
00055 unsigned char GlcdRectangle::getWidth() {
00056     return (bottom - top);
00057 }
00058
00059 unsigned char GlcdRectangle::getHeight() {
00060     return (right - left);
00061 }
00062
00063 unsigned int GlcdRectangle::getArea() {
00064     return (unsigned int) (getWidth() * getHeight());
00065 }
00066
00067 #endif /* __ARDUINO_LIBRARY_GLCD_RECTANGLE_CPP__ */
```

5.27 GlcdRectangle.h File Reference

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



Classes

- class GlcdRectangle

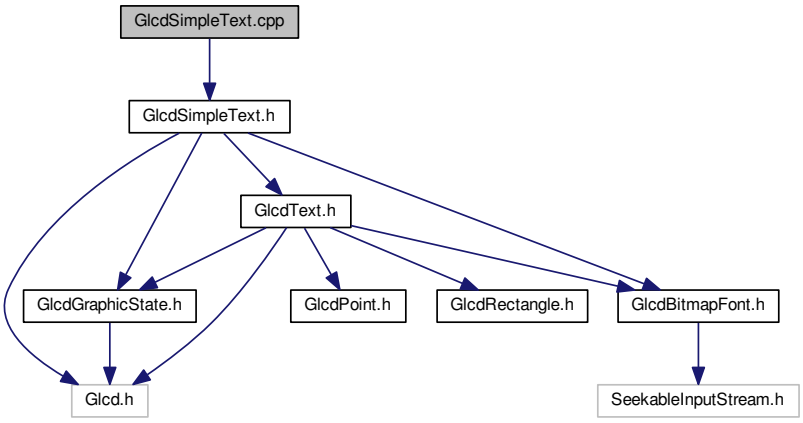
5.28 GlcdRectangle.h

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_RECTANGLE_H__
00012 #define __ARDUINO_LIBRARY_GLCD_RECTANGLE_H__ 1
00013
00014 class GlcdRectangle {
```

```
00015 private:
00016
00020     unsigned char left;
00021
00025     unsigned char top;
00026
00030     unsigned char right;
00031
00035     unsigned char bottom;
00036 public:
00037
00041     GlcdRectangle();
00042
00051     GlcdRectangle(unsigned char left, unsigned char top, unsigned char right, unsigned char
      bottom);
00052
00058     void setTop(unsigned char top);
00059
00065     unsigned char getTop();
00066
00072     void setLeft(unsigned char left);
00073
00079     unsigned char getLeft();
00080
00086     void setRight(unsigned char right);
00087
00093     unsigned char getRight();
00094
00100     void setBottom(unsigned char bottom);
00101
00107     unsigned char getBottom();
00108
00114     unsigned char getWidth();
00115
00121     unsigned char getHeight();
00122
00128     unsigned int getArea();
00129 };
00130
00131 #endif /* __ARDUINO_LIBRARY_GLCD_RECTANGLE_H__ */
```

5.29 GlcdSimpleText.cpp File Reference

`#include "GlcdSimpleText.h"`
Include dependency graph for GlcdSimpleText.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_SIMPLE_TEXT_CPP__ 1

5.29.1 Macro Definition Documentation

5.29.1.1 #define __ARDUINO_LIBRARY_GLCD_SIMPLE_TEXT_CPP__ 1

Arduino - Glcd library.

GlcdSimpleText.cpp

The function bodies to draw text in a glcd plane

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdSimpleText.cpp.

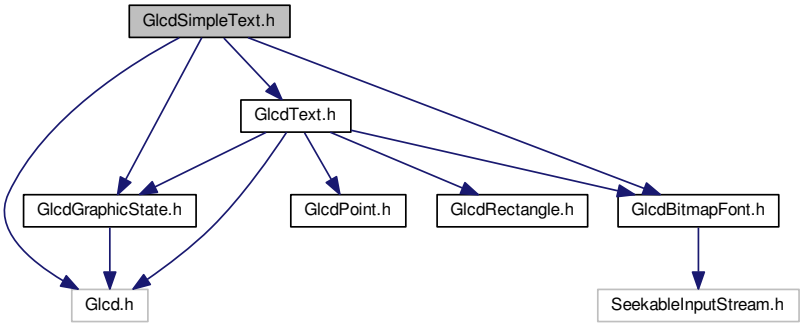
5.30 GlcdSimpleText.cpp

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_SIMPLE_TEXT_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_SIMPLE_TEXT_CPP__ 1
00013
00014 #include "GlcdSimpleText.h"
00015
00016 GlcdSimpleText::GlcdSimpleText(Glcd *glcd,
      GlcdBitmapFont *font, GlcdGraphicState *graphicState) :
      GlcdText(glcd, font, graphicState) {
00017 }
00018
00019 void GlcdSimpleText::printChar(unsigned char x, unsigned char y, const unsigned
      char c, unsigned char size) {
00020
00021     unsigned char rows, column, glyphBuf[font->getGlyphLength()];
00022
00023     // Convert pixel to row.
00024     y /= (GLCD_HEIGHT / 8);
00025
00026     // Read char data
00027     font->readGlyphData(glyphBuf, c);
00028
00029     // Loop through character byte data
00030     for (unsigned char i = 0; i < font->getCharacterWidth(); i++) {
00031
00032         // Rows
00033         rows = font->getCharacterHeight() / 8;
00034
00035         // Loop for all columns.
00036         for (unsigned char j = 0; j < rows; j++) {
00037
00038             // Column
00039             column = glyphBuf[j * font->getCharacterWidth() + i];
00040
00041             if (graphicState->getColor() == Glcd::COLOR_BLACK) {
00042                 column = ~column;
00043             }
00044
00045             // Streaks the column.
00046             glcd->streak(x + i, y, column);
00047         }
00048     }
00049 }
00050
00051 #endif /* __ARDUINO_LIBRARY_GLCD_SIMPLE_TEXT_CPP__ */
```

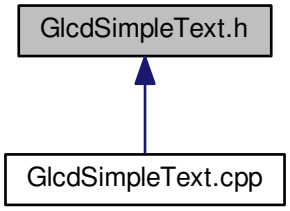
5.31 GlcdSimpleText.h File Reference

```
#include <Glcd.h>
#include <GlcdText.h>
#include <GlcdBitmapFont.h>
#include <GlcdGraphicState.h>
```

Include dependency graph for GlcdSimpleText.h:



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



Classes

- class GlcdSimpleText

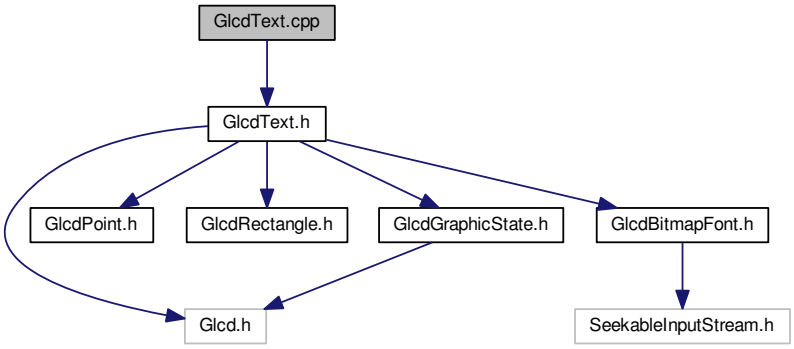
5.32 GlcdSimpleText.h

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_SIMPLE_TEXT_H__
00012 #define __ARDUINO_LIBRARY_GLCD_SIMPLE_TEXT_H__ 1
00013
00014 #include <Glcd.h>
00015 #include <GlcdText.h>
00016 #include <GlcdBitmapFont.h>
00017 #include <GlcdGraphicState.h>
00018
00019 class GlcdSimpleText : public GlcdText {
00020 public:
00021
00029     GlcdSimpleText(Glcd *glcd, GlcdBitmapFont *
      font, GlcdGraphicState *graphicState);
00030
00040     virtual void printChar(unsigned char x, unsigned char y, const unsigned char c, unsigned char
      size);
00041
00049     inline void printChar(GlcdPoint *p, const unsigned char c, unsigned char size) {
00050         printChar(p->getX(), p->getY(), c, size);
00051     }
00052 };
00053
```

```
00054 #endif /* __SDCC_LIBRARY_GLCD_SIMPLE_TEXT_H__ */
```

5.33 GlcdText.cpp File Reference

`#include "GlcdText.h"`
Include dependency graph for GlcdText.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_TEXT_CPP__ 1

5.33.1 Macro Definition Documentation

5.33.1.1 #define __ARDUINO_LIBRARY_GLCD_TEXT_CPP__ 1

Arduino Graphic LCD Library.

GlcdText.cpp

The functions to draw text in a glcd plane.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdText.cpp.

5.34 GlcdText.cpp

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_TEXT_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_TEXT_CPP__ 1
00013
00014 #include "GlcdText.h"
00015
00016 GlcdText::GlcdText(Glcd *glcd, GlcdBitmapFont *font,
      GlcdGraphicState *graphicState) : glcd(glcd), font(font), graphicState(graphicState) {
00017 }
00018
00019 void GlcdText::setGlcd(Glcd *glcd) {
00020     this->glcd = glcd;
00021 }
00022
00023 void GlcdText::setFont(GlcdBitmapFont *font) {
```

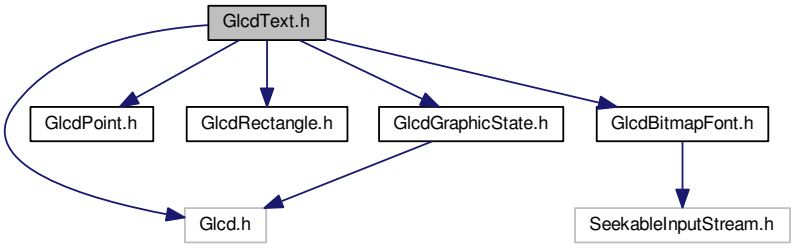
```
00024     this->font = font;
00025 }
00026
00027 void GlcdText::setGraphicState(GlcdGraphicState *graphicState) {
00028     this->graphicState = graphicState;
00029 }
00030
00031 Glcd *GlcdText::getGlcd() {
00032     return this->glcd;
00033 }
00034
00035 GlcdBitmapFont *GlcdText::getFont() {
00036     return this->font;
00037 }
00038
00039 GlcdGraphicState *GlcdText::getGraphicState() {
00040     return this->graphicState;
00041 }
00042
00043 void GlcdText::printChar(unsigned char x, unsigned char y, const unsigned char c,
      unsigned char size) {
00044
00045     unsigned char glyphBuf[font->getGlyphLength()];
00046     unsigned char rows, column = 0;
00047
00048     // Read char data
00049     font->readGlyphData(glyphBuf, c);
00050
00051     // Loop through character byte data
00052     for (unsigned char i = 0; i < font->getCharacterWidth(); i++, x += size) {
00053
00054         rows = font->getCharacterHeight() / 8;
00055
00056         // Loop for all rows.
00057         for (unsigned char r = 0; r < rows; r++) {
00058
00059             // Column
00060             column = glyphBuf[r * font->getCharacterWidth() + i];
00061
00062             // Loop through the vertical pixels
00063             for (unsigned char j = 0; j < font->getCharacterHeight() * size; j++) {
00064
00065                 unsigned char newY = y + (j * size) + (8 * r * size);
00066
00067                 // Check if the pixel should be plotted
00068                 if ((column & (1 << j)) != 0) {
00069
00070                     // The next two loops change the character's size
00071                     for (unsigned char k = 0; k < size; k++) {
00072
00073                         for (unsigned char z = 0; z < size; z++) {
00074
00075                             // Draws the pixel
00076                             glcd->plot(x + z, newY + k, graphicState->
      getColor());
00077                         }
00078                     }
00079                 }
00080             }
00081         }
00082     }
00083 }
00084
00085 unsigned char GlcdText::printString(unsigned char left, unsigned char top, unsigned
      char right, unsigned char bottom, const unsigned char *text, unsigned char count, unsigned char size) {
00086
00087     unsigned char x = left, y = top;
00088     unsigned char i;
00089
00090     // Loop through the passed string
00091     for (i = 0; (i < count) && (text[i] != '\0'); i++) {
00092
00093         // Performs character wrapping
00094         if (x + (font->getCharacterWidth() * size) > right || text[i] == '\n') {
00095
00096             // Set x at left position
00097             x = left;
00098
00099             // Set y at next position down
00100             y += (font->getCharacterHeight() * size) +
      graphicState->getLeading();
00101
00102             // Do not print a space as first character of the line or new line.
00103             if (text[i] == ' ' || text[i] == '\n') {
00104                 continue;
00105             }
00106         }
```

```
00107
00108         // Out of printable area
00109         if (y + (font->getCharacterHeight() * size) >= bottom) {
00110             break;
00111         }
00112
00113         // Print he char
00114         printChar(x, y, text[i], size);
00115
00116         // Move the x position to the next char
00117         x += (font->getCharacterWidth() * size) +
      graphicState->getSpace();
00118     }
00119     return i;
00120 }
00121
00122 #endif /* __ARDUINO_LIBRARY_GLCD_TEXT_CPP__ */
```

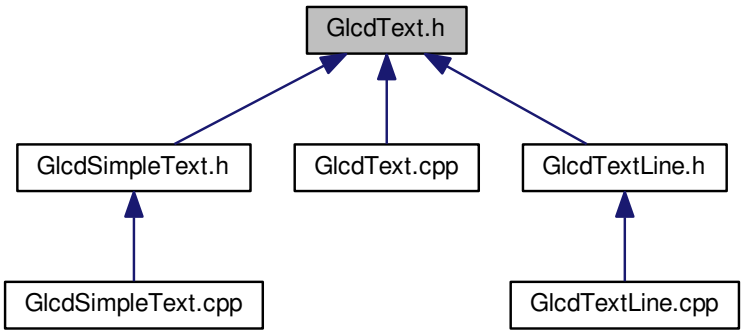
5.35 GlcdText.h File Reference

```
#include <Glcd.h>
#include <GlcdPoint.h>
#include <GlcdRectangle.h>
#include <GlcdBitmapFont.h>
#include <GlcdGraphicState.h>
```

Include dependency graph for GlcdText.h:



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



Classes

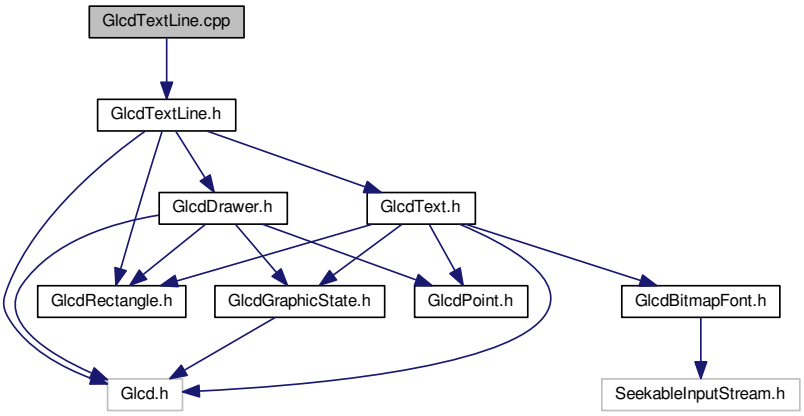
- class GlcdText

5.36 GlcdText.h

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_TEXT_H__
00012 #define __ARDUINO_LIBRARY_GLCD_TEXT_H__ 1
00013
00014 #include <Glcd.h>
00015 #include <GlcdPoint.h>
00016 #include <GlcdRectangle.h>
00017 #include <GlcdBitmapFont.h>
00018 #include <GlcdGraphicState.h>
00019
00020 class GlcdText {
00021 protected:
00022
00026     Glcd *glcd;
00027
00031     GlcdBitmapFont *font;
00032
00036     GlcdGraphicState *graphicState;
00037
00038 public:
00039
00047     GlcdText(Glcd *glcd, GlcdBitmapFont *font,
      GlcdGraphicState *graphicState);
00048
00054     void setGlcd(Glcd *glcd);
00055
00061     void setFont(GlcdBitmapFont *font);
00062
00068     void setGraphicState(GlcdGraphicState *graphicState);
00069
00075     Glcd *getGlcd();
00076
00082     GlcdBitmapFont *getFont();
00083
00089     GlcdGraphicState *getGraphicState();
00090
00101     virtual void printChar(unsigned char x, unsigned char y, const unsigned char c, unsigned char
      size);
00102
00110     inline void printChar(unsigned char x, unsigned char y, const unsigned char c) {
00111         printChar(x, y, c, 1);
00112     }
00113
00121     inline void printChar(GlcdPoint *p, const unsigned char c, unsigned char size) {
00122         printChar(p->getX(), p->getY(), c, size);
00123     }
00124
00131     inline void printChar(GlcdPoint *p, const unsigned char c) {
00132         printChar(p->getX(), p->getY(), c, 1);
00133     }
00134
00149     virtual unsigned char printString(unsigned char left, unsigned char top, unsigned char right
      , unsigned char bottom, const unsigned char *text, unsigned char count, unsigned char size);
00150
00164     inline unsigned char printString(unsigned char left, unsigned char top, unsigned char right,
       unsigned char bottom, const unsigned char *text, unsigned char count) {
00165         return printString(left, top, right, bottom, text, count, 1);
00166     }
00167
00179     inline unsigned char printString(GlcdRectangle *area, const unsigned char *text
      , unsigned char count, unsigned char size) {
00180         return printString(area->getLeft(), area->getTop(), area->
      getRight(), area->getBottom(), text, count, size);
00181     }
00182
00193     inline unsigned char printString(GlcdRectangle *area, const unsigned char *text
      , unsigned char count) {
00194         return printString(area, text, count, 1);
00195     }
00196 };
00197
00198 #endif /* __SDCC_LIBRARY_GLCD_TEXT_H__ */
```

5.37 GlcdTextLine.cpp File Reference

`#include "GlcdTextLine.h"`
Include dependency graph for GlcdTextLine.cpp:



Macros

- #define __ARDUINO_LIBRARY_GLCD_TEXT_LINE_CPP__ 1

5.37.1 Macro Definition Documentation

5.37.1.1 #define __ARDUINO_LIBRARY_GLCD_TEXT_LINE_CPP__ 1

Arduino Graphic LCD Library.

GlcdTextLine.cpp

The functions to draw text in a glcd always at the same line.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file GlcdTextLine.cpp.

5.38 GlcdTextLine.cpp

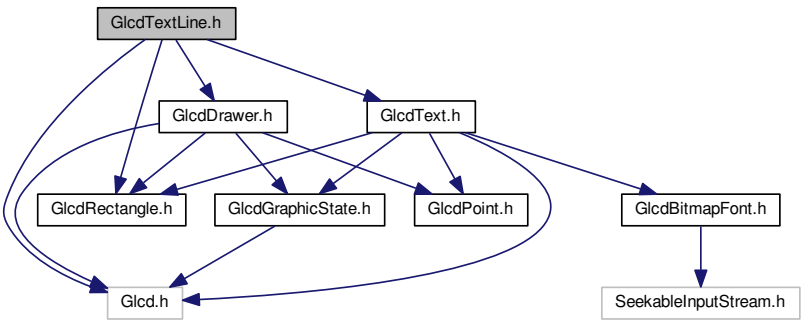
```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_TEXT_LINE_CPP__
00012 #define __ARDUINO_LIBRARY_GLCD_TEXT_LINE_CPP__ 1
00013
00014 #include "GlcdTextLine.h"
00015
00016 GlcdTextLine::GlcdTextLine(GlcdText *glcdText,
      GlcdDrawer *glcdDrawer,
00017         unsigned char y) :
00018         glcdText(glcdText), glcdDrawer(glcdDrawer) {
00019     this->y = y;
00020 }
00021
00022 void GlcdTextLine::printLines(const unsigned char *text, unsigned char count) {
00023     unsigned char realCharacterHeight, index = 0;
00024     realCharacterHeight = glcdText->getFont()->getCharacterHeight()
00025             + glcdText->getGraphicState()->getLeading();
```

```
00026     Serial.print("realCharacterHeight: ");
00027     Serial.println(realCharacterHeight);
00028     Serial.print("y: ");
00029     Serial.println(y);
00030     GlcdRectangle area(0, (y & (GLCD_HEIGHT - 1)), GLCD_WIDTH - 1, ((
      y
00031             + realCharacterHeight) & (GLCD_HEIGHT - 1)));
00032     Serial.print("L: ");
00033     Serial.println(area.getLeft());
00034     Serial.print("R: ");
00035     Serial.println(area.getRight());
00036     Serial.print("T: ");
00037     Serial.println(area.getTop());
00038     Serial.print("B: ");
00039     Serial.println(area.getBottom());
00040     while (1) {
00041         unsigned char printed;
00042         /*
00043         glcdDrawer->getGraphicState()->invertColor();
00044         glcdDrawer->drawRectangle(&area);
00045         glcdDrawer->getGraphicState()->invertColor();
00046         */
00047         printed = glcdText->printString(&area, &text[index], count);
00048         Serial.print("printed:");
00049         Serial.println(printed);
00050         if (printed == 0) {
00051             break;
00052         }
00053         area.setBottom((area.getBottom() + realCharacterHeight) & (GLCD_HEIGHT - 1));
00054         area.setTop((area.getTop() + realCharacterHeight) & (GLCD_HEIGHT - 1));
00055 
00056 
00057         Serial.print("sL");
00058         Serial.println(area.getLeft());
00059         Serial.print("sR: ");
00060         Serial.println(area.getRight());
00061         Serial.print("sT: ");
00062         Serial.println(area.getTop());
00063         Serial.print("sB: ");
00064         Serial.println(area.getBottom());
00065 
00066 
00067         y += realCharacterHeight;
00068         glcdText->getGlcd()->scroll(Glcd::CHIP_ALL, Glcd::SCROLL_UP, realCharacterHeight);
00069         index += printed;
00070     }
00071 }
00072 
00073 #endif /* __ARDUINO_LIBRARY_GLCD_TEXT_LINE_CPP__ */
```

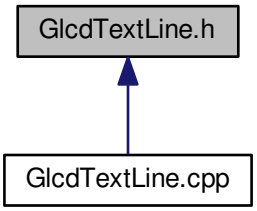
5.39 GlcdTextLine.h File Reference

```
#include <Glcd.h>
#include <GlcdText.h>
#include <GlcdRectangle.h>
#include <GlcdDrawer.h>
```

Include dependency graph for GlcdTextLine.h:



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



Classes

- class GlcdTextLine

5.40 GlcdTextLine.h

```
00001
00011 #ifndef __ARDUINO_LIBRARY_GLCD_TEXT_LINE_H__
00012 #define __ARDUINO_LIBRARY_GLCD_TEXT_LINE_H__ 1
00013
00014 #include <Glcd.h>
00015 #include <GlcdText.h>
00016 #include <GlcdRectangle.h>
00017 #include <GlcdDrawer.h>
00018
00019 class GlcdTextLine {
00020 protected:
00021
00025     GlcdText *glcdText;
00026
00030     GlcdDrawer *glcdDrawer;
00031
00035     unsigned char y;
00036
00037 public:
00038
00045     GlcdTextLine(GlcdText *glcdText, GlcdDrawer *glcdDrawer, unsigned char y)
      ;
00046
00053     void printLines(const unsigned char *text, unsigned char count);
00054 };
00055
00056 #endif /* __ARDUINO_LIBRARY_GLCD_TEXT_LINE_H__ */
```

Index