Graphing Calculator 0

Generated by Doxygen 1.10.0

1	Class Index	1
	1.1 Class List	1
2	File Index	3
	2.1 File List	3
3	Class Documentation	5
	3.1 Application Class Reference	5
	3.1.1 Detailed Description	6
	3.1.2 Constructor & Destructor Documentation	6
	3.1.2.1 Application()	6
	3.1.2.2 ~ Application()	6
	3.1.3 Member Function Documentation	6
	3.1.3.1 initializeVariables()	6
	3.1.3.2 initializeWindow()	7
	3.1.3.3 pollEvents()	7
	3.1.3.4 render()	8
	3.1.3.5 update()	8
	3.1.3.6 windowlsOpen()	8
	3.1.4 Member Data Documentation	8
	3.1.4.1 event	8
	3.1.4.2 graph	8
	3.1.4.3 gui	9
	3.1.4.4 videoMode	9
	3.1.4.5 window	9
	3.2 Graph Class Reference	9
	3.2.1 Detailed Description	10
	3.2.2 Constructor & Destructor Documentation	11
	3.2.2.1 Graph()	11
	3.2.3 Member Function Documentation	11
	3.2.3.1 decXRange()	11
	3.2.3.2 decYRange()	11
	3.2.3.3 drawToWindow()	12
	3.2.3.4 functionVertices()	12
	3.2.3.5 graphToScreen()	13
	3.2.3.6 gridToScreen()	13
	3.2.3.7 incXRange()	13
	3.2.3.8 incYRange()	13
	3.2.3.9 screenToGraph()	14
	3.2.3.10 setGraphMode()	14
	3.2.3.11 updateGraph()	14
	3.2.4 Member Data Documentation	15
	3.2.4.1 background	15
	o.z odolground	

3.2.4.2 font	15
3.2.4.3 graphMode	15
3.2.4.4 gridCols	15
3.2.4.5 gridRows	16
3.2.4.6 gridVector	16
3.2.4.7 hMax	16
3.2.4.8 hMin	16
3.2.4.9 text	16
3.2.4.10 textVector	16
3.2.4.11 wMax	17
3.2.4.12 wMin	17
3.2.4.13 xMax	17
3.2.4.14 xMin	17
3.2.4.15 yMax	17
3.2.4.16 yMin	17
3.3 Gui Class Reference	18
3.3.1 Detailed Description	18
3.3.2 Constructor & Destructor Documentation	18
3.3.2.1 Gui()	18
3.3.3 Member Function Documentation	19
3.3.3.1 drawToWindow()	19
3.3.3.2 updateGui()	19
3.3.4 Member Data Documentation	19
3.3.4.1 ibhMax	19
3.3.4.2 ibhMin	19
3.3.4.3 ibwMax	19
3.3.4.4 ibwMin	19
3.3.4.5 inputBackground	19
Ella Da como adation	0.4
File Documentation 4.1 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/application.cpp File Reference	21
4.2 application.cpp	
4.3 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/application.h File Reference 4.4 application.h	
4.5 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/graph.cpp File Reference 4.6 graph.cpp	
4.7 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/graph.h File Reference	
4.9 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/gui.cpp File Reference	
4.10 gui.cpp	
4.11 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/gui.h File Reference	
4.12 gui.h	27

37

4.13 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/header.h File Reference	28
4.13.1 Variable Documentation	28
4.13.1.1 DEFAULT_H_MAX	28
4.13.1.2 DEFAULT_H_MIN	29
4.13.1.3 DEFAULT_W_MAX	29
4.13.1.4 DEFAULT_W_MIN	29
4.13.1.5 DEFAULT_X_MAX	29
4.13.1.6 DEFAULT_X_MIN	29
4.13.1.7 DEFAULT_Y_MAX	29
4.13.1.8 DEFAULT_Y_MIN	30
4.13.1.9 DEFAULT_ZERO_THRESH	30
4.13.1.10 WINDOW_HEIGHT	30
4.13.1.11 WINDOW_WIDTH	30
4.14 header.h	30
4.15 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/main.cpp File Reference	31
4.15.1 Function Documentation	31
4.15.1.1 main()	31
4.16 main.cpp	31
4.17 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/test_funcs.cpp File Reference	31
4.17.1 Function Documentation	32
4.17.1.1 funcA()	32
4.17.1.2 funcB()	32
4.17.1.3 funcC()	32
4.17.1.4 funcD()	32
4.17.1.5 funcE()	33
4.17.1.6 funcF()	33
4.17.1.7 funcG()	33
4.18 test_funcs.cpp	33
4.19 /home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/test_funcs.h File Reference	34
4.19.1 Function Documentation	34
4.19.1.1 funcA()	34
4.19.1.2 funcB()	34
4.19.1.3 funcC()	34
4.19.1.4 funcD()	35
4.19.1.5 funcE()	35
4.19.1.6 funcF()	35
4.19.1.7 funcG()	35
4.20 test funcs.h	35

Index

Chapter 1

Class Index

1.1 Class List

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

Applicati	on	
	Manages the application window	5
Graph		
	Manages the graph area on screen	9
Gui		
	Manages the gui for the application	8

2 Class Index

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/application.cpp	 21
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/application.h	 22
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/graph.cpp	 23
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/graph.h	 26
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/gui.cpp	 27
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/gui.h	 27
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/header.h	 28
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/main.cpp	
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/test_funcs.cpp	 31
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/test_funcs.h	

File Index

Chapter 3

Class Documentation

3.1 Application Class Reference

Manages the application window.

```
#include <application.h>
```

Public Member Functions

• Application ()

Default Constructor.

virtual ∼Application ()

Destructor.

• const bool windowlsOpen () const

Get whether window is open.

• void pollEvents ()

Poll for events.

• void update ()

Update application objects.

• void render ()

Draw renderable objects to the window and display.

Private Member Functions

• void initializeVariables ()

Initialzie member variables.

• void initializeWindow ()

Initialize window member variable.

Private Attributes

- sf::RenderWindow * window
- sf::VideoMode videoMode
- sf::Event event
- Graph * graph
- Gui * gui

6 Class Documentation

3.1.1 Detailed Description

Manages the application window.

Definition at line 10 of file application.h.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Application()

```
Application::Application ( )
```

Default Constructor.

Definition at line 22 of file application.cpp.

```
00022 this->initializeVariables();
00024 this->initializeWindow();
00025 }
```

3.1.2.2 \sim Application()

```
Application::~Application ( ) [virtual]
```

Destructor.

Definition at line 27 of file application.cpp.

```
00027 {
00028 delete this->window;
00029 delete this->graph;
00030 delete this->gui;
00031 }
```

3.1.3 Member Function Documentation

3.1.3.1 initializeVariables()

```
void Application::initializeVariables ( ) [private]
```

Initialzie member variables.

Definition at line 3 of file application.cpp.

```
00003

00004 this->window = nullptr;

00005 sf::Vector2f xRange(DEFAULT_X_MIN, DEFAULT_X_MAX);

00006 sf::Vector2f yRange(DEFAULT_Y_MIN, DEFAULT_Y_MAX);

00007 sf::Vector2f wRange(DEFAULT_W_MIN, DEFAULT_W_MAX);

00008 sf::Vector2f hRange(DEFAULT_H_MIN, DEFAULT_H_MAX);

00009 this->graph = new Graph(xRange, yRange, wRange, hRange);

00010 this->gui = new Gui();
```

3.1.3.2 initializeWindow()

```
void Application::initializeWindow ( ) [private]
```

Initialize window member variable.

```
Definition at line 13 of file application.cpp.
```

3.1.3.3 pollEvents()

```
void Application::pollEvents ( )
```

Poll for events.

```
Definition at line 37 of file application.cpp.
```

```
00038
          while (this->window->pollEvent(this->event)) {
00039
             switch (this->event.type) {
              case sf::Event::Closed:
00040
00041
                 this->window->close();
00042
00043
              case sf::Event::KeyPressed:
00044
                 switch (this->event.key.code) {
00045
                      case sf::Keyboard::Escape:
00046
                         this->window->close();
00047
                          break;
00048
                      case sf::Keyboard::A:
00049
                         graph->setGraphMode(1);
00050
00051
                      case sf::Keyboard::B:
00052
                         graph->setGraphMode(2);
00053
                          break:
00054
                      case sf::Keyboard::C:
00055
                         graph->setGraphMode(3);
00056
00057
                      case sf::Keyboard::D:
00058
                          graph->setGraphMode(4);
00059
                         break:
00060
                      case sf::Kevboard::E:
00061
                         graph->setGraphMode(5);
00062
00063
                      case sf::Keyboard::F:
00064
                         graph->setGraphMode(6);
00065
                          break:
00066
                      case sf::Keyboard::G:
00067
                         graph->setGraphMode(7);
00068
00069
                      case sf::Keyboard::Right:
00070
                         graph->incXRange(0.1);
00071
                          break:
00072
                      case sf::Keyboard::Left:
00073
                         graph->decXRange(0.1);
00074
                          break;
00075
                      case sf::Keyboard::Up:
00076
                          graph->incYRange(0.1);
00077
                          break:
00078
                      case sf::Keyboard::Down:
00079
                          graph->decYRange(0.1);
00080
                          break;
00081
00082
                  break;
00083
             }
00084
          }
00085 }
```

8 Class Documentation

3.1.3.4 render()

```
void Application::render ( )
```

Draw renderable objects to the window and display.

Definition at line 93 of file application.cpp.

3.1.3.5 update()

```
void Application::update ( )
```

Update application objects.

Definition at line 87 of file application.cpp.

```
00087
00088    pollEvents();
00089    graph->updateGraph();
00090    gui->updateGui();
```

3.1.3.6 windowlsOpen()

```
const bool Application::windowIsOpen ( ) const
```

Get whether window is open.

```
Definition at line 33 of file application.cpp.

00033

00034    return this->window->isOpen();
```

3.1.4 Member Data Documentation

3.1.4.1 event

00035 }

```
sf::Event Application::event [private]
```

Definition at line 15 of file application.h.

3.1.4.2 graph

```
Graph* Application::graph [private]
```

Definition at line 16 of file application.h.

3.1.4.3 gui

```
Gui* Application::gui [private]
```

Definition at line 17 of file application.h.

3.1.4.4 videoMode

```
sf::VideoMode Application::videoMode [private]
```

Definition at line 14 of file application.h.

3.1.4.5 window

```
sf::RenderWindow* Application::window [private]
```

Definition at line 13 of file application.h.

3.2 Graph Class Reference

Manages the graph area on screen.

```
#include <graph.h>
```

Public Member Functions

• Graph (sf::Vector2f xRange, sf::Vector2f yRange, sf::Vector2f wRange, sf::Vector2f hRange)

Default Constructor.

• void setGraphMode (int i)

Set graphMode.

• void updateGraph ()

Update gridVector based on graphMode.

void drawToWindow (sf::RenderWindow *window)

Draw gridVector to a window.

void decXRange (float xDec)

Decrement xMin and xMax by xDec.

void decYRange (float yDec)

Decrement yMin and yMax by yDec.

void incXRange (float xInc)

Increment xMin and xMax by xInc.

void incYRange (float yInc)

Increment yMin and yMax by yInc.

10 Class Documentation

Private Member Functions

• sf::Vector2f gridToScreen (sf::Vector2u grid_pos)

Convert grid (i,j) coordinates to screen (w,h) coordinates.

sf::Vector2f screenToGraph (sf::Vector2f screen_pos)

Convert screen (w,h) coordinates to graph (x,y) coordinates.

sf::Vector2f graphToScreen (sf::Vector2f graph_pos)

Convert graph (x,y) coordinates to screen (w,h) coordinates.

• sf::VertexArray functionVertices ()

Create verticies for a function.

Private Attributes

· float xMin

Minimum x value (graph)

float xMax

Maximum x value (graph)

float yMin

Minimum y value (graph)

float yMax

Maximum y value (graph)

• float wMin

Minimum x value (screen)

float wMax

Maximum x value (screen)

float hMin

Minimum y value (screen)

float hMax

Maximum y value (screen)

• int gridRows

Rows in rectangle grid.

· int gridCols

Columns in rectangle grid.

• sf::VertexArray gridVector

Collection of points to be drawn.

sf::RectangleShape background

Background rectangle.

· int graphMode

Temp variable for declaring graph type.

sf::Text text

Text for drawing on graph.

• std::vector< sf::Text > textVector

Collect of texts to be drawn.

sf::Font font

Font for drawing text.

3.2.1 Detailed Description

Manages the graph area on screen.

Definition at line 9 of file graph.h.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Graph()

Default Constructor.

Definition at line 77 of file graph.cpp.

```
00078 {
00079
            xMin = xRange.x;
08000
            xMax = xRange.y;
            yMin = yRange.x;
00081
00082
            yMax = yRange.y;
00083
            wMin = wRange.x;
            wMax = wRange.y;
00084
            hMin = hRange.x;
00085
            hMax = hRange.y;
00086
            gridRows = hMax-hMin;
gridCols = wMax-wMin;
00087
88000
00089
            graphMode = 1;
00090
            background.setSize(sf::Vector2f(wMax-wMin, hMax-hMin));
background.setPosition(sf::Vector2f(wMin, hMin));
00091
00092
            background.setFillColor(sf::Color::White);
00093
00094
00095
            font.loadFromFile("res/arial.ttf");
00096
            text.setFont(font);
            // pixels, not points
text.setCharacterSize(24);
00097
00098
00099
            text.setFillColor(sf::Color::Black);
00100 }
```

3.2.3 Member Function Documentation

3.2.3.1 decXRange()

Decrement xMin and xMax by xDec.

Definition at line 185 of file graph.cpp.

3.2.3.2 decYRange()

Decrement yMin and yMax by yDec.

Definition at line 192 of file graph.cpp.

12 Class Documentation

3.2.3.3 drawToWindow()

```
void Graph::drawToWindow (
    sf::RenderWindow * window )
```

Draw gridVector to a window.

```
Definition at line 166 of file graph.cpp.
```

```
00167 {
00168
          window->draw(background);
00169
00170
          for (auto r : gridVector)
00172
00173
              window->draw(r);
00174
00175
00176
         window->draw(gridVector);
00177
00178
          for (auto t: textVector)
00179
00180
              window->draw(t);
00181
00182
          return;
00183 }
```

3.2.3.4 functionVertices()

sf::VertexArray Graph::functionVertices () [private]

Create verticies for a function.

Definition at line 27 of file graph.cpp.

```
00028 {
00029
           sf::VertexArray vertices;
00030
           for (int i=0; i<gridRows; i++)</pre>
00031
00032
               for (int j=0; j<gridCols; j++)</pre>
00033
                    sf::Vector2f screenC = gridToScreen(sf::Vector2u(i, j));
sf::Vector2f graphC = screenToGraph(screenC);
00034
00035
00036
                    float x = graphC.x;
                    float y = graphC.y;
00038
                    float fx = 0;
00039
                    \ensuremath{//} determine the function
00040
                    switch (graphMode)
00041
00042
                        case 1:
                           fx = funcA(x);
00043
00044
                             break;
                        case 2:
    fx = funcB(x);
00045
00046
00047
                            break;
00048
                        case 3:
00049
                            fx = funcC(x);
00050
                            break;
00051
                        case 4:
                            fx = funcD(x);
00052
00053
                            break;
00054
                        case 5:
                            fx = funcE(x);
00055
00056
                            break;
                        case 6:
  fx = funcF(x);
00057
00058
                        break; case 7:
00059
00060
                            fx = funcG(x);
00061
00062
                             break;
00063
00064
                    // graph vertices
                       (std::abs(y-fx) < DEFAULT_ZERO_THRESH)
00065
00066
00067
                        sf::Vertex vertex;
00068
                        vertex.position = screenC;
00069
                        vertex.color = sf::Color::Red;
00070
                        vertices.append(vertex);
00071
00072
00073
00074
           return vertices;
```

3.2.3.5 graphToScreen()

Convert graph (x,y) coordinates to screen (w,h) coordinates.

Definition at line 19 of file graph.cpp.

```
00020 {
00021     sf::Vector2f screen_pos;
00022     screen_pos.x = wMin + (graph_pos.x-xMin) * (wMax-wMin) / (xMax-xMin);
00023     screen_pos.y = hMin + (yMax-graph_pos.y) * (hMax-hMin) / (yMax-yMin);
00024     return screen_pos;
00025 }
```

3.2.3.6 gridToScreen()

Convert grid (i,j) coordinates to screen (w,h) coordinates.

Definition at line 3 of file graph.cpp.

```
00004 {
00005     sf::Vector2f screen_pos;
00006     screen_pos.x = wMin + grid_pos.x * (wMax-wMin) / gridCols;
00007     screen_pos.y = hMin + grid_pos.y * (hMax-hMin) / gridRows;
00008     return screen_pos;
00009 }
```

3.2.3.7 incXRange()

Increment xMin and xMax by xInc.

Definition at line 199 of file graph.cpp.

3.2.3.8 incYRange()

Increment yMin and yMax by yInc.

Definition at line 206 of file graph.cpp.

14 Class Documentation

3.2.3.9 screenToGraph()

Convert screen (w,h) coordinates to graph (x,y) coordinates.

Definition at line 11 of file graph.cpp.

```
00012 {
00013     sf::Vector2f graph_pos;
00014     graph_pos.x = xMin + (screen_pos.x-wMin) * (xMax-xMin) / (wMax-wMin);
00015     graph_pos.y = yMax - (screen_pos.y-hMin) * (yMax-yMin) / (hMax-hMin);
00016     return graph_pos;
00017 }
```

3.2.3.10 setGraphMode()

```
void Graph::setGraphMode ( int i )
```

Set graphMode.

Definition at line 102 of file graph.cpp.

3.2.3.11 updateGraph()

```
void Graph::updateGraph ( )
```

Update gridVector based on graphMode.

Definition at line 108 of file graph.cpp.

```
00109 {
00110
           gridVector.clear();
00111
           textVector.clear();
00112
00113
           sf::VertexArray graphVertices = functionVertices();
00114
           for (int i=0; i<graphVertices.getVertexCount(); i++)</pre>
00115
00116
               gridVector.append(graphVertices[i]);
00117
00118
00119
           // draw x axis
00120
           if (yMax > 0 && yMin < 0)
00121
00122
               float screenY = graphToScreen(sf::Vector2f(xMin, 0)).y;
00123
               for (int i=0; i<gridCols; i++)</pre>
00124
00125
                   sf::Vertex vertex;
                   float screenX = gridToScreen(sf::Vector2u(i, 0)).x;
vertex.position = sf::Vector2f(screenX, screenY);
00126
00127
                   vertex.color = sf::Color::Black;
00128
00129
                   gridVector.append(vertex);
00130
00131
          }
00132
00133
           // draw {\bf x} axis tick numbers
00134
           for (int i=static_cast<int>(ceil(xMin)); i<static_cast<int>(ceil(xMax)); i++)
00135
00136
               text.setString(std::to_string(i));
00137
               text.setPosition(graphToScreen(sf::Vector2f(i, 0)));
00138
               textVector.push_back(text);
00139
          }
00140
00141
          // draw y axis
00142
          if (xMax > 0 && xMin < 0)
```

```
00143
          {
00144
               float screenX = graphToScreen(sf::Vector2f(0, yMin)).x;
00145
               for (int i=0; i<gridRows; i++)</pre>
00146
00147
                   sf::Vertex vertex;
                   float screeny = gridToScreen(sf::Vector2u(0, i)).y;
vertex.position = sf::Vector2f(screenX, screenY);
00148
00149
00150
                   vertex.color = sf::Color::Black;
00151
                   gridVector.append(vertex);
00152
              }
          }
00153
00154
00155
          // draw y axis tick numbers
00156
          for (int i=static_cast<int>(ceil(yMin)); i<static_cast<int>(ceil(yMax)); i++)
00157
00158
               text.setString(std::to_string(i));
              text.setPosition(graphToScreen(sf::Vector2f(0, i)));
00159
00160
              textVector.push_back(text);
00161
          }
00162
00163
          return;
00164 }
```

3.2.4 Member Data Documentation

3.2.4.1 background

```
sf::RectangleShape Graph::background [private]
```

Background rectangle.

Definition at line 36 of file graph.h.

3.2.4.2 font

```
sf::Font Graph::font [private]
```

Font for drawing text.

Definition at line 44 of file graph.h.

3.2.4.3 graphMode

```
int Graph::graphMode [private]
```

Temp variable for declaring graph type.

Definition at line 38 of file graph.h.

3.2.4.4 gridCols

```
int Graph::gridCols [private]
```

Columns in rectangle grid.

Definition at line 32 of file graph.h.

16 Class Documentation

3.2.4.5 gridRows

```
int Graph::gridRows [private]
```

Rows in rectangle grid.

Definition at line 30 of file graph.h.

3.2.4.6 gridVector

```
sf::VertexArray Graph::gridVector [private]
```

Collection of points to be drawn.

Definition at line 34 of file graph.h.

3.2.4.7 hMax

```
float Graph::hMax [private]
```

Maximum y value (screen)

Definition at line 28 of file graph.h.

3.2.4.8 hMin

```
float Graph::hMin [private]
```

Minimum y value (screen)

Definition at line 26 of file graph.h.

3.2.4.9 text

```
sf::Text Graph::text [private]
```

Text for drawing on graph.

Definition at line 40 of file graph.h.

3.2.4.10 textVector

```
std::vector<sf::Text> Graph::textVector [private]
```

Collect of texts to be drawn.

Definition at line 42 of file graph.h.

3.2.4.11 wMax

float Graph::wMax [private]

Maximum x value (screen)

Definition at line 24 of file graph.h.

3.2.4.12 wMin

```
float Graph::wMin [private]
```

Minimum x value (screen)

Definition at line 22 of file graph.h.

3.2.4.13 xMax

```
float Graph::xMax [private]
```

Maximum x value (graph)

Definition at line 16 of file graph.h.

3.2.4.14 xMin

```
float Graph::xMin [private]
```

Minimum x value (graph)

Definition at line 14 of file graph.h.

3.2.4.15 yMax

```
float Graph::yMax [private]
```

Maximum y value (graph)

Definition at line 20 of file graph.h.

3.2.4.16 yMin

```
float Graph::yMin [private]
```

Minimum y value (graph)

Definition at line 18 of file graph.h.

18 Class Documentation

3.3 Gui Class Reference

Manages the gui for the application.

```
#include <gui.h>
```

Public Member Functions

• Gui ()

Default Constructor.

• void updateGui ()

Update Gui.

void drawToWindow (sf::RenderWindow *window)

Draw Gui to a window.

Private Attributes

· float ibwMin

Input Region Background Min Width.

· float ibwMax

Input Region Background Max Width.

float ibhMin

Input Region Background Min Height.

· float ibhMax

Input Region Background Max Height.

• sf::RectangleShape inputBackground

Input Region Background.

3.3.1 Detailed Description

Manages the gui for the application.

Definition at line 8 of file gui.h.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 Gui()

```
Gui::Gui ( )
```

Default Constructor.

Definition at line 3 of file gui.cpp.

3.3 Gui Class Reference 19

3.3.3 Member Function Documentation

3.3.3.1 drawToWindow()

```
void Gui::drawToWindow (
              sf::RenderWindow * window )
Draw Gui to a window.
Definition at line 17 of file gui.cpp.
00017
00018
          window->draw(inputBackground);
00019
          return;
00020 }
3.3.3.2 updateGui()
void Gui::updateGui ( )
Update Gui.
Definition at line 13 of file gui.cpp.
00013
00014
          return;
00015 }
```

3.3.4 Member Data Documentation

3.3.4.1 ibhMax

```
float Gui::ibhMax [private]
Input Region Background Max Height.
Definition at line 19 of file gui.h.
```

3.3.4.2 ibhMin

```
float Gui::ibhMin [private]
Input Region Background Min Height.
Definition at line 17 of file gui.h.
```

3.3.4.3 ibwMax

```
float Gui::ibwMax [private]
Input Region Background Max Width.
Definition at line 15 of file gui.h.
```

3.3.4.4 ibwMin

```
float Gui::ibwMin [private]
Input Region Background Min Width.
Definition at line 13 of file gui.h.
```

3.3.4.5 inputBackground

```
sf::RectangleShape Gui::inputBackground [private]
Input Region Background.

Definition at line 21 of file gui.h.
```

20 Class Documentation

Chapter 4

File Documentation

4.1 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/application.cpp File Reference

```
#include "application.h"
```

4.2 application.cpp

Go to the documentation of this file.

```
00001 #include "application.h"
00002
00003 void Application::initializeVariables() {
00004 this->window = nullptr;
00005
          sf::Vector2f xRange(DEFAULT_X_MIN, DEFAULT_X_MAX);
          sf::Vector2f yRange(DEFAULT_Y_MIN, DEFAULT_Y_MAX);
sf::Vector2f wRange(DEFAULT_W_MIN, DEFAULT_W_MAX);
00006
00007
80000
          sf::Vector2f hRange(DEFAULT_H_MIN, DEFAULT_H_MAX);
          this->graph = new Graph(xRange, yRange, wRange, hRange);
00009
00010
          this->gui = new Gui();
00011 }
00012
00013 void Application::initializeWindow() {
00014 this->videoMode.width = WINDOW_WIDTH;
00015 this->videoMode.height = WINDOW_HEIGHT;
00016
          this->window = new sf::RenderWindow(this->videoMode, "Application", sf::Style::None);
00017
                                                     sf::Style::Titlebar | sf::Style::Close);
          this->window->setPosition(sf::Vector2i(0, 0));
00018
          this->window->setFramerateLimit(60);
00019
00020 }
00021
00022 Application::Application() {
00023 this->initializeVariables();
00024
          this->initializeWindow();
00025 }
00026
00027 Application::~Application() {
        delete this->window;
00028
          delete this->graph;
00030
          delete this->gui;
00031 }
00032
00033 const bool Application::windowIsOpen() const {
          return this->window->isOpen();
00035 }
00036
00037 void Application::pollEvents() {
00038 while (this->window->pollEvent(this->event)) {
00039 switch (this->event.type) {
            switch (this->event.type) {
  case sf::Event::Closed:
                   this->window->close();
```

22 File Documentation

```
00042
                 break;
00043
             case sf::Event::KeyPressed:
00044
                 switch (this->event.key.code) {
                     case sf::Keyboard::Escape:
00045
                        this->window->close();
00046
00047
                         break:
                     case sf::Keyboard::A:
                       graph->setGraphMode(1);
break;
00049
00050
00051
                      case sf::Keyboard::B:
00052
                        graph->setGraphMode(2);
00053
                         break:
00054
                     case sf::Keyboard::C:
00055
                        graph->setGraphMode(3);
00056
00057
                      case sf::Keyboard::D:
00058
                         graph->setGraphMode(4);
00059
                         break;
00060
                      case sf::Keyboard::E:
                        graph->setGraphMode(5);
break;
00061
00062
00063
                      case sf::Keyboard::F:
00064
                        graph->setGraphMode(6);
00065
                         break:
00066
                      case sf::Keyboard::G:
00067
                        graph->setGraphMode(7);
00068
00069
                      case sf::Keyboard::Right:
                        graph->incXRange(0.1);
00070
00071
                         break:
00072
                      case sf::Kevboard::Left:
                        graph->decXRange(0.1);
00074
00075
                      case sf::Keyboard::Up:
00076
                        graph->incYRange(0.1);
00077
                         break:
00078
                      case sf::Keyboard::Down:
                         graph->decYRange(0.1);
08000
00081
00082
                 break:
             }
00083
00084
00085 }
00087 void Application::update() {
00088 pollEvents();
00089
         graph->updateGraph();
00090
         gui->updateGui();
00091 }
00092
00093 void Application::render()
00094 {
00095
         window->clear();
00096
         graph->drawToWindow(window);
00097
         gui->drawToWindow(window);
         window->display();
00099 }
00100
00101
```

4.3 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/application.h File Reference

```
#include "header.h"
#include "graph.h"
#include "gui.h"
```

Classes

· class Application

Manages the application window.

4.4 application.h

4.4 application.h

Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "header.h"
00004 #include "graph.h"
00005 #include "gui.h"
00006
00010 class Application {
00011 private:
          // Variables
00013
          sf::RenderWindow* window;
00014
          sf::VideoMode videoMode;
00015
          sf::Event event;
00016
          Graph* graph;
00017
          Gui* gui;
00018
00019
          // Private Member Functions
00021
          void initializeVariables();
00023
          void initializeWindow();
00024 public:
00025
          // Constructors / Destructors
00027
          Application();
          virtual ~Application();
          // Accessors
00030
00032
          const bool windowIsOpen() const;
00033
          // Functions
          void pollEvents();
void update();
00035
00037
00039
          void render();
00040 };
```

4.5 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/graph.cpp File Reference

```
#include "graph.h"
```

4.6 graph.cpp

Go to the documentation of this file.

```
00001 #include "graph.h"
00002
00003 sf::Vector2f Graph::gridToScreen(sf::Vector2u grid_pos)
00004 {
00005
            sf::Vector2f screen_pos;
            screen_pos.x = wMin + grid_pos.x * (wMax-wMin) / gridCols;
screen_pos.y = hMin + grid_pos.y * (hMax-hMin) / gridRows;
00006
00007
80000
            return screen_pos;
00009 }
00010
00011 sf::Vector2f Graph::screenToGraph(sf::Vector2f screen_pos)
00012 {
00013
            sf::Vector2f graph_pos;
            graph_pos.x = xMin + (screen_pos.x-wMin) * (xMax-xMin) / (wMax-wMin);
graph_pos.y = yMax - (screen_pos.y-hMin) * (yMax-yMin) / (hMax-hMin);
00014
00015
00016
            return graph pos;
00017 }
00018
00019 sf::Vector2f Graph::graphToScreen(sf::Vector2f graph_pos)
00020 {
00021
            sf::Vector2f screen pos;
            screen_pos.x = wMin + (graph_pos.x-xMin) * (wMax-wMin) / (xMax-xMin);
screen_pos.y = hMin + (yMax-graph_pos.y) * (hMax-hMin) / (yMax-yMin);
00022
00023
            return screen_pos;
00024
00025 }
00026
00027 sf::VertexArray Graph::functionVertices()
00028 {
            sf::VertexArray vertices;
```

24 File Documentation

```
00030
          for (int i=0; i<gridRows; i++)</pre>
00031
00032
               for (int j=0; j<gridCols; j++)</pre>
00033
                   sf::Vector2f screenC = gridToScreen(sf::Vector2u(i, j));
sf::Vector2f graphC = screenToGraph(screenC);
00034
00035
                   float x = graphC.x;
00037
                   float y = graphC.y;
00038
                   float fx = 0;
00039
                   // determine the function
00040
                   switch (graphMode)
00041
00042
                       case 1:
00043
                           fx = funcA(x);
00044
                           break;
00045
                       case 2:
                           fx = funcB(x);
00046
00047
                           break;
00048
                       case 3:
                          fx = funcC(x);
break;
00049
00050
00051
                       case 4:
                          fx = funcD(x);
00052
00053
                           break;
00054
                       case 5:
00055
                          fx = funcE(x);
                           break;
00056
                       case 6:
  fx = funcF(x);
00057
00058
00059
                           break;
00060
                       case 7:
00061
                           fx = funcG(x);
00062
                            break;
00063
00064
                   // graph vertices
                   if (std::abs(y-fx) < DEFAULT_ZERO_THRESH)</pre>
00065
00066
                   {
00067
                       sf::Vertex vertex;
00068
                       vertex.position = screenC;
00069
                       vertex.color = sf::Color::Red;
00070
                       vertices.append(vertex);
00071
                   }
00072
              }
00073
00074
          return vertices;
00075 }
00076
00077 Graph::Graph(sf::Vector2f xRange, sf::Vector2f yRange, sf::Vector2f wRange, sf::Vector2f hRange)
00078 {
00079
          xMin = xRange.x:
00080
          xMax = xRange.y;
00081
          yMin = yRange.x;
00082
           yMax = yRange.y;
          wMin = wRange.x;
00083
          wMax = wRange.y;
00084
00085
          hMin = hRange.x;
00086
          hMax = hRange.y;
          gridRows = hMax-hMin;
gridCols = wMax-wMin;
00087
88000
          graphMode = 1;
00089
00090
          background.setSize(sf::Vector2f(wMax-wMin, hMax-hMin));
00091
00092
          background.setPosition(sf::Vector2f(wMin, hMin));
00093
          background.setFillColor(sf::Color::White);
00094
00095
          font.loadFromFile("res/arial.ttf");
00096
          text.setFont(font);
00097
          // pixels, not points
          text.setCharacterSize(24);
00098
00099
          text.setFillColor(sf::Color::Black);
00100 }
00101
00102 void Graph::setGraphMode(int i)
00103 {
          graphMode = i;
00104
00105
          return;
00106 }
00107
00108 void Graph::updateGraph()
00109 {
00110
          gridVector.clear();
00111
          textVector.clear();
00112
00113
          sf::VertexArray graphVertices = functionVertices();
00114
          for (int i=0; i<graphVertices.getVertexCount(); i++)</pre>
00115
           {
00116
               gridVector.append(graphVertices[i]);
```

4.6 graph.cpp 25

```
00117
          }
00118
00119
          // draw x axis
00120
          if (yMax > 0 && yMin < 0)
00121
              float screenY = graphToScreen(sf::Vector2f(xMin, 0)).y;
00122
00123
              for (int i=0; i<gridCols; i++)</pre>
00124
00125
                  sf::Vertex vertex;
00126
                  float screenX = gridToScreen(sf::Vector2u(i, 0)).x;
                  vertex.position = sf::Vector2f(screenX, screenY);
00127
                  vertex.color = sf::Color::Black;
00128
00129
                  gridVector.append(vertex);
00130
00131
          }
00132
          // draw x axis tick numbers
00133
00134
          for (int i=static_cast<int>(ceil(xMin)); i<static_cast<int>(ceil(xMax)); i++)
00135
00136
              text.setString(std::to_string(i));
00137
              text.setPosition(graphToScreen(sf::Vector2f(i, 0)));
00138
              textVector.push_back(text);
00139
          }
00140
00141
          // draw y axis
          if (xMax > 0 && xMin < 0)
00142
00143
00144
              float screenX = graphToScreen(sf::Vector2f(0, yMin)).x;
00145
              for (int i=0; i<gridRows; i++)</pre>
00146
00147
                  sf::Vertex vertex;
00148
                  float screenY = gridToScreen(sf::Vector2u(0, i)).y;
00149
                  vertex.position = sf::Vector2f(screenX, screenY);
00150
                  vertex.color = sf::Color::Black;
00151
                  gridVector.append(vertex);
              }
00152
00153
          }
00154
00155
          // draw y axis tick numbers
00156
          for (int i=static_cast<int>(ceil(yMin)); i<static_cast<int>(ceil(yMax)); i++)
00157
00158
              text.setString(std::to_string(i));
              text.setPosition(graphToScreen(sf::Vector2f(0, i)));
00159
00160
              textVector.push_back(text);
00161
00162
00163
          return:
00164 }
00165
00166 void Graph::drawToWindow(sf::RenderWindow* window)
00167 {
00168
          window->draw(background);
00169
00170
00171
          for (auto r : gridVector)
00172
          {
              window->draw(r);
00174
00175
00176
         window->draw(gridVector);
00177
00178
          for (auto t: textVector)
00179
00180
              window->draw(t);
00181
          return;
00182
00183 }
00184
00185 void Graph::decXRange(float xDec)
00186 {
00187
          xMin -= xDec;
00188
          xMax -= xDec;
00189
          return;
00190 }
00191
00192 void Graph::decYRange(float yDec)
00193 {
00194
          yMin -= yDec;
          yMax -= yDec;
00195
00196
          return:
00197 }
00198
00199 void Graph::incXRange(float xInc)
00200 {
00201
          xMin += xInc;
          xMax += xInc;
00202
00203
          return:
```

26 File Documentation

```
00204 }
00205
00206 void Graph::incYRange(float yInc)
00207 {
00208     yMin += yInc;
00209     yMax += yInc;
00210     return;
00211 }
```

4.7 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/graph.h File Reference

```
#include "header.h"
#include "test_funcs.h"
```

Classes

· class Graph

Manages the graph area on screen.

4.8 graph.h

Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "header.h"
00004 #include "test_funcs.h"
00005
00009 class Graph
00010 {
00011 private:
00012
          // Variables
00014
          float xMin;
00016
          float xMax;
         float yMin;
float yMax;
00018
00020
00022
          float wMin;
00024
          float wMax;
00026
          float hMin;
00028
          float hMax;
00030
          int gridRows;
00032
          int gridCols;
          sf::VertexArray gridVector;
00036
          sf::RectangleShape background;
00038
          int graphMode;
00040
          sf::Text text;
00042
          std::vector<sf::Text> textVector;
00044
          sf::Font font;
00045
00046
          // Private Member Functions
00048
          sf::Vector2f gridToScreen(sf::Vector2u grid_pos);
00050
          sf::Vector2f screenToGraph(sf::Vector2f screen_pos);
00052
          sf::Vector2f graphToScreen(sf::Vector2f graph_pos);
          sf::VertexArray functionVertices();
00054
00055 public:
00057
          Graph(sf::Vector2f xRange, sf::Vector2f yRange, sf::Vector2f wRange, sf::Vector2f hRange);
00059
          void setGraphMode(int i);
00061
          void updateGraph();
          void drawToWindow(sf::RenderWindow* window);
00063
          void decXRange(float xDec);
00065
          void decYRange(float yDec);
void incXRange(float xInc);
00067
00071
          void incYRange(float yInc);
00072 };
```

4.9 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/gui.cpp File Reference

```
#include "gui.h"
```

4.10 gui.cpp

Go to the documentation of this file.

```
00001 #include "gui.h"
00002
00003 Gui::Gui() {
00004 ibwMin = 100;
         ibwMax = 500;
        ibhMin = 100;
ibhMax = 980;
00006
00007
         inputBackground.setSize(sf::Vector2f(ibwMax-ibwMin, ibhMax-ibhMin));
80000
         inputBackground.setPosition(sf::Vector2f(ibwMin, ibhMin));
00009
00010
         inputBackground.setFillColor(sf::Color::White);
00011 }
00012
00013 void Gui::updateGui() {
00014
         return;
00015 }
00016
00017 void Gui::drawToWindow(sf::RenderWindow* window) {
00018 window->draw(inputBackground);
00019
00020 }
```

4.11 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/gui.h File Reference

```
#include "header.h"
```

Classes

· class Gui

Manages the gui for the application.

4.12 gui.h

Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "header.h"
00004
00008 class Gui
00010 private:
00011 // Variables
00013 float ibwMin;
        float ibwMax;
float ibhMin;
float ibhMax;
00015
00017
00021
          sf::RectangleShape inputBackground;
00022 public:
00024 Gui();
           void updateGui();
00026
           void drawToWindow(sf::RenderWindow* window);
00028
00029
00030 };
```

28 File Documentation

4.13 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/header.h File Reference

```
#include <iostream>
#include <vector>
#include <ctime>
#include <SFML/Graphics.hpp>
#include <SFML/System.hpp>
#include <SFML/Window.hpp>
#include <SFML/Audio.hpp>
#include <SFML/Network.hpp>
```

Variables

```
• const float WINDOW_WIDTH = 1920
```

Necessary Includes.

const float WINDOW HEIGHT = 1080

Default Window Height.

const float DEFAULT X MIN = -2

Default Min X Value for Graphing.

const float DEFAULT_X_MAX = 2

Default Max X Value for Graphing.

• const float DEFAULT_Y_MIN = -2

Default Min Y Value for Graphing.

• const float DEFAULT_Y_MAX = 2

Default Max Y Value for Graphing.

• const float DEFAULT_W_MIN = 760

Default Min Width Position for Graphing.

• const float DEFAULT_W_MAX = 1560

Default Max Width Position for Graphing.

• const float DEFAULT_H_MIN = 140

Default Min Height Position for Graphing.

• const float DEFAULT_H_MAX = 940

Default Max Height Position for Graphing.

• const float DEFAULT_ZERO_THRESH = 0.01f

Default Zero Threshold, values below are zero.

4.13.1 Variable Documentation

4.13.1.1 DEFAULT_H_MAX

```
const float DEFAULT_H_MAX = 940
```

Default Max Height Position for Graphing.

Definition at line 38 of file header.h.

4.13.1.2 DEFAULT_H_MIN

```
const float DEFAULT_H_MIN = 140
```

Default Min Height Position for Graphing.

Definition at line 36 of file header.h.

4.13.1.3 DEFAULT_W_MAX

```
const float DEFAULT_W_MAX = 1560
```

Default Max Width Position for Graphing.

Definition at line 34 of file header.h.

4.13.1.4 **DEFAULT_W_MIN**

```
const float DEFAULT_W_MIN = 760
```

Default Min Width Position for Graphing.

Definition at line 32 of file header.h.

4.13.1.5 DEFAULT_X_MAX

```
const float DEFAULT_X_MAX = 2
```

Default Max X Value for Graphing.

Definition at line 25 of file header.h.

4.13.1.6 **DEFAULT X MIN**

```
const float DEFAULT_X_MIN = -2
```

Default Min X Value for Graphing.

Definition at line 23 of file header.h.

4.13.1.7 DEFAULT_Y_MAX

```
const float DEFAULT_Y_MAX = 2
```

Default Max Y Value for Graphing.

Definition at line 29 of file header.h.

30 File Documentation

4.13.1.8 DEFAULT_Y_MIN

```
const float DEFAULT_Y_MIN = -2
```

Default Min Y Value for Graphing.

Definition at line 27 of file header.h.

4.13.1.9 DEFAULT ZERO THRESH

```
const float DEFAULT_ZERO_THRESH = 0.01f
```

Default Zero Threshold, values below are zero.

Definition at line 41 of file header.h.

4.13.1.10 WINDOW_HEIGHT

```
const float WINDOW_HEIGHT = 1080
```

Default Window Height.

Definition at line 20 of file header.h.

4.13.1.11 WINDOW WIDTH

```
const float WINDOW_WIDTH = 1920
```

Necessary Includes.

Default Window Width

Definition at line 18 of file header.h.

4.14 header.h

Go to the documentation of this file.

```
00001 #pragma once
00002
00006
00007 #include <iostream>
00008 #include <vector>
00009 #include <ctime>
00010
00011 #include <SFML/Graphics.hpp>
00012 #include <SFML/System.hpp>
00013 #include <SFML/Window.hpp>
00014 #include <SFML/Audio.hpp>
00015 #include <SFML/Network.hpp>
00016
00018 const float WINDOW_WIDTH = 1920;
00020 const float WINDOW_HEIGHT = 1080;
00021
00023 const float DEFAULT_X_MIN = -2;
00025 const float DEFAULT_X_MAX = 2;
00027 const float DEFAULT_Y_MIN = -2;
00029 const float DEFAULT_Y_MAX = 2;
00030
00032 const float DEFAULT_W_MIN = 760;
00034 const float DEFAULT_W_MAX = 1560;
00036 const float DEFAULT_H_MIN = 140;
00038 const float DEFAULT_H_MAX = 940;
00041 const float DEFAULT_ZERO_THRESH = 0.01f;
```

4.15 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/main.cpp File Reference

```
#include "application.h"
```

Functions

• int main ()

Program entry point.

4.15.1 Function Documentation

4.15.1.1 main()

```
int main ( )
```

Program entry point.

Seed random. Create an instance of Application. Update app and render to the screen while the window is open.

Definition at line 9 of file main.cpp.

4.16 main.cpp

Go to the documentation of this file.

```
00001 #include "application.h'
00002
80000
00009 int main()
00010 {
00011
         std::srand(std::time(nullptr));
         Application app;
         while (app.windowIsOpen()) {
         app.update();
app.render();
00014
00015
00016
00017
          return 0:
00018 }
```

4.17 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/test_funcs.cpp File Reference

```
#include "test_funcs.h"
```

32 File Documentation

Functions

- float funcA (float x)
- float funcB (float x)
- float funcC (float x)
- float funcD (float x)
- float funcE (float x)
- float funcF (float x)
- float funcG (float x)

4.17.1 Function Documentation

4.17.1.1 funcA()

```
float funcA ( \label{float x } \mbox{float } x \mbox{ )}
```

Definition at line 3 of file test_funcs.cpp.

```
00003
00004 return x;
00005 }
```

4.17.1.2 funcB()

```
float funcB ( \label{float x } \mbox{float } \mbox{$x$ )}
```

Definition at line 7 of file test_funcs.cpp.

```
00007
00008 return x * x;
00009 }
```

4.17.1.3 funcC()

```
float funcC ( float x )
```

Definition at line 11 of file test_funcs.cpp.

```
00011 {
00012 return x * x * x;
00013 }
```

4.17.1.4 funcD()

```
float funcD ( float x )
```

Definition at line 15 of file test funcs.cpp.

4.18 test_funcs.cpp 33

4.17.1.5 funcE()

```
float funcE (
              float x )
Definition at line 19 of file test_funcs.cpp.
00019
00020
          return std::sin(x);
00021 }
4.17.1.6 funcF()
float funcF (
              float x )
Definition at line 23 of file test_funcs.cpp.
00023
00024
          return std::cos(x);
00025 }
4.17.1.7 funcG()
```

float x)

```
Definition at line 27 of file test_funcs.cpp.

00027

00028
return std::tan(x);

00029 }
```

float funcG (

4.18 test_funcs.cpp

Go to the documentation of this file.

```
00001 #include "test_funcs.h"
00002
00003 float funcA(float x) {
00004    return x;
coud4 return x;
00005 }
00007
00006
00007 float funcB(float x) {
00008 return x * x;
00009 }
00010
00011 float funcC(float x) {
00012     return x * x * x;
         return x * x * x;
00013 }
00014
00015 float funcD(float x) {
00016 return std::abs(x);
00017 }
00018
00020 return std::sin(x);
00021 }
00019 float funcE(float x) {
00022
00023 float funcF(float x) {
00024 return std::cos(x);
00025 }
00026
00027 float funcG(float x) {
00028 return std::tan(x)
         return std::tan(x);
00029 }
```

34 File Documentation

4.19 /home/jakemath/Desktop/code/SFML/Graphing/Graphing App/src/test_funcs.h File Reference

```
#include "math.h"
```

Functions

- float funcA (float)
- float funcB (float)
- float funcC (float)
- float funcD (float)
- float funcE (float)
- float funcF (float)
- · float funcG (float)

4.19.1 Function Documentation

4.19.1.1 funcA()

```
float funcA ( float x )
```

Definition at line 3 of file test_funcs.cpp.

```
00003
00004 return x;
00005 }
```

4.19.1.2 funcB()

```
float funcB ( \label{float x } \mbox{float } x \mbox{ )}
```

Definition at line 7 of file test_funcs.cpp.

```
00007
00008 return x * x;
00009 }
```

4.19.1.3 funcC()

```
float funcC ( \label{float x } \mbox{float } x \mbox{ )}
```

Definition at line 11 of file test_funcs.cpp.

```
00011 {
00012 return x * x * x;
00013 }
```

4.20 test_funcs.h 35

4.19.1.4 funcD()

```
float funcD (
              float x )
Definition at line 15 of file test_funcs.cpp.
00015
00016
          return std::abs(x);
00017 }
4.19.1.5 funcE()
float funcE (
       float x )
Definition at line 19 of file test_funcs.cpp.
00019
00020
          return std::sin(x);
00021 }
```

4.19.1.6 funcF()

```
float funcF (
            float x )
```

Definition at line 23 of file test_funcs.cpp.

```
00023
         return std::cos(x);
00024
00025 }
```

4.19.1.7 funcG()

```
float funcG (
            float x )
```

Definition at line 27 of file test funcs.cpp.

```
00028
          return std::tan(x);
00029 }
```

4.20 test_funcs.h

Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include "math.h"
00004
00005 float funcA(float);
00006 float funcB(float);
00007 float funcC(float);
00008 float funcD(float);
00009 float funcE(float);
00010 float funcF(float);
00011 float funcG(float);
```

36 File Documentation

Index

```
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/srb/eaptbication.cpp,
                                                                                                                                                                                              DEFAULT W MIN
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/srb/æappteicatiaa.h,
                                                                                                                                                                                              DEFAULT X MAX
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb/egraphingApp/srb
                                                                                                                                                                                              DEFAULT X MIN
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/srb/egrdphlh,29
                                                                                                                                                                                              DEFAULT Y MAX
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/srb/egadlexplp,, 29
                                                                                                                                                                                              DEFAULT Y MIN
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/srb/epadler, h, 29
                                                                                                                                                                                              DEFAULT ZERO THRESH
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/srb#eedetr,h30
                                 28, 30
                                                                                                                                                                                              drawToWindow
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/sr@rapim.dpp,
                                                                                                                                                                                                               Gui, 19
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/src/test_funcs.cpp,
                                                                                                                                                                                              event
/home/jakemath/Desktop/code/SFML/Graphing/GraphingApp/srentables. Applying App/srentables. Applying Ap
                                 34, 35
\simApplication
                                                                                                                                                                                                              Graph, 15
                Application, 6
                                                                                                                                                                                              funcA
                                                                                                                                                                                                              test funcs.cpp, 32
Application, 5
                                                                                                                                                                                                              test funcs.h, 34
                \simApplication, 6
                                                                                                                                                                                              funcB
                Application, 6
                                                                                                                                                                                                              test_funcs.cpp, 32
                event, 8
                                                                                                                                                                                                              test funcs.h, 34
                graph, 8
                                                                                                                                                                                              funcC
                gui, 8
                                                                                                                                                                                                              test_funcs.cpp, 32
                initializeVariables, 6
               initializeWindow, 6
                                                                                                                                                                                                              test_funcs.h, 34
                                                                                                                                                                                              funcD
                pollEvents, 7
                                                                                                                                                                                                              test_funcs.cpp, 32
                render, 7
                                                                                                                                                                                                              test_funcs.h, 34
                update, 8
                                                                                                                                                                                              funcE
                videoMode, 9
                                                                                                                                                                                                              test funcs.cpp, 32
                window, 9
                                                                                                                                                                                                              test funcs.h, 35
                windowlsOpen, 8
                                                                                                                                                                                              funcF
background
                                                                                                                                                                                                              test funcs.cpp, 33
                Graph, 15
                                                                                                                                                                                                              test funcs.h, 35
                                                                                                                                                                                              funcG
decXRange
                                                                                                                                                                                                              test_funcs.cpp, 33
                Graph, 11
                                                                                                                                                                                                              test funcs.h, 35
decYRange
                                                                                                                                                                                              functionVertices
                Graph, 11
                                                                                                                                                                                                              Graph, 12
DEFAULT H MAX
                header.h, 28
                                                                                                                                                                                              Graph, 9
DEFAULT H MIN
                                                                                                                                                                                                              background, 15
                header.h, 28
                                                                                                                                                                                                              decXRange, 11
DEFAULT W MAX
                                                                                                                                                                                                              decYRange, 11
```

38 INDEX

drawToWindow, 11	DEFAULT_Y_MIN, 29
font, 15	DEFAULT_ZERO_THRESH, 30
functionVertices, 12	WINDOW_HEIGHT, 30
Graph, 11	WINDOW_WIDTH, 30
graphMode, 15	hMax
graphToScreen, 12	Graph, 16
gridCols, 15	hMin
gridRows, 15	Graph, 16
gridToScreen, 13	
gridVector, 16	ibhMax
hMax, 16	Gui, 19
hMin, 16	ibhMin
incXRange, 13	Gui, 19
incYRange, 13	ibwMax
screenToGraph, 13	Gui, 19
setGraphMode, 14	ibwMin
text, 16	Gui, 19
textVector, 16	incXRange
updateGraph, 14	Graph, 13
wMax, 16	incYRange
wMin, 17	Graph, 13
xMax, 17	initializeVariables
xMin, 17	Application, 6
yMax, 17	initializeWindow
yMin, 17	Application, 6
graph	inputBackground
Application, 8	Gui, 19
graphMode	
Graph, 15	main
graphToScreen	main.cpp, 31
Graph, 12	main.cpp
gridCols	main, 31
Graph, 15	
gridRows	pollEvents
Graph, 15	Application, 7
gridToScreen	
Graph, 13	render
gridVector	Application, 7
Graph, 16	screenToGraph
Gui, 18	Graph, 13
drawToWindow, 19	setGraphMode
Gui, 18	Graph, 14
ibhMax, 19	αιαρίί, 14
ibhMin, 19	test_funcs.cpp
ibwMax, 19	funcA, 32
ibwMin, 19	funcB, 32
inputBackground, 19	funcC, 32
	funcD, 32
updateGui, 19	funcE, 32
gui	funcF, 33
Application, 8	funcG, 33
header.h	test funcs.h
DEFAULT_H_MAX, 28	funcA, 34
DEFAULT_H_MIN, 28	funcB, 34
DEFAULT_W_MAX, 29	funcC, 34
DEFAULT_W_MIN, 29	funcD, 34
DEFAULT X MAX, 29	funcE, 35
DEFAULT X MIN, 29	funcF, 35
DEFAULT_Y_MAX, 29	funcG, 35
DEIMOLI_I_WAX, 20	iuriou, oo

INDEX 39

text Graph, 16 textVectorGraph, 16 update Application, 8 updateGraph Graph, 14 updateGui Gui, 19 videoMode Application, 9 window Application, 9 WINDOW_HEIGHT header.h, 30 WINDOW_WIDTH header.h, 30 windowIsOpen Application, 8 wMax Graph, 16 wMin Graph, 17 xMax Graph, 17 xMin Graph, 17 yMax Graph, 17 yMin Graph, 17