Multiplot

0.5.5

Generated by Doxygen 1.8.15

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

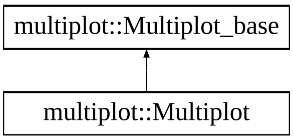
Chapter 3

Class Documentation

3.1 multiplot::Multiplot Class Reference

`#include <multiplot.h>`

Inheritance diagram for multiplot::Multiplot:



Classes

- class Point2d
- class Trace

Public Member Functions

- **Multiplot** (const int x, const int y, const int w, const int h, const std::wstring &ttitle=L"Multiplot - updates on www.andre-krause.net", bool fullscreen=false)
- Trace & operator[ ] (int _trace)
- Trace & operator() (int _trace)
- Trace & trace (int _trace)
- void plot (const float x, const float y)
- template< class T >
  void plot (const std::vector< T > &v)
- void color3f (float r, float g, float b)
- void set_title (const std::wstring &title_)
- void set_linewidth (float width)
- void set_pointsize (float psize)
- void set_scrolling (int max_points_to_plot)
- void set_scaling (enum MP_SCALING sc, float x_min=-10, float x_max=10, float y_min=-10, float y_max=10)
- void set_grid (enum MP_GRIDSTYLE ggridx=MP_LINEAR_GRID, enum MP_GRIDSTYLE ggridy=MP_LINEAR_GRID, float ggridx_step=-1.0, float ggridy_step=-1.0, float w=1.0)
- void set_bg_color (float r, float g, float b)
- void set_grid_color (float r, float g, float b)
- void clear_all ()
- void clear (int trace)

Protected Member Functions

- void **initgl** ()
- Point2d **draw_grid** ()
- virtual void **draw** ()

Protected Attributes

- float **cur_point_size**
- unsigned int **cur_trace**
- std::wstring **title**
- Point2d **bg_color**
- Point2d **grid_color**
- int **scaling_**
- Point2d **range_min**
- Point2d **range_max**
- Point2d **minimum**
- Point2d **maximum**
- Point2d **scale**
- Point2d **offset**
- std::vector< Trace > **traces**
- int **gridx**
- int **gridy**
- float **gridx_step**
- float **gridy_step**
- float **grid_linewidth**
- Point2d **grid_spacing**

Additional Inherited Members

3.1.1 Detailed Description

this class creates a window to wich you can add an arbitrary number of autoscaling traces.

3.1.2 Member Function Documentation

3.1.2.1 clear()

```
void multiplot::Multiplot::clear (
             int trace )  [inline]
```

this function call clears trace number t

3.1.2.2 clear_all()

```
void multiplot::Multiplot::clear_all ( )  [inline]
```

this function call simply clears all traces

3.1.2.3 color3f()

```
void multiplot::Multiplot::color3f (
             float r,
             float g,
             float b )  [inline]
```

change current drawing color for current trace.

3.1.2.4 operator()()

```
Trace& multiplot::Multiplot::operator() (
             int _trace )  [inline]
```

Access function. allows direct access to a trace.

3.1.2.5 operator[]()

```
Trace& multiplot::Multiplot::operator[] (
             int _trace )  [inline]
```

Access function. allows direct access to a trace.

3.1.2.6 plot() [1/2]

```
void multiplot::Multiplot::plot (
             const float x,
             const float y )  [inline]
```

plots a point at x,y to the currently active trace. select a trace with a call to trace(int _tracenumber);

3.1.2.7 plot() [2/2]

```
template<class T >
void multiplot::Multiplot::plot (
             const std::vector< T > & v )  [inline]
```

plots a vector of values to the currently active trace. the x value is running from 0 .. vector.size()-1 select a trace with a call to trace(int _tracenumber);

3.1.2.8 set_bg_color()

```
void multiplot::Multiplot::set_bg_color (
             float r,
             float g,
             float b )  [inline]
```

sets the background color

3.1.2.9 set_grid()

```
void multiplot::Multiplot::set_grid (
            enum MP_GRIDSTYLE ggridx = MP_LINEAR_GRID,
            enum MP_GRIDSTYLE ggridy = MP_LINEAR_GRID,
            float ggridx_step = -1.0,
            float ggridy_step = -1.0,
            float w = 1.0 )  [inline]
```

call this function if you wish a grid to be plotted in your graph. by default, no grids are plotted. call this function with the first two arguments set to either MP_NO_GRID, MP_LINEAR_GRID or MP_LOG_GRID. the next two arguments gridx_step and gridy_step specify the grid spacing. Zero or a negative value like -1 enables auto - spacing. The last parameter w sets the grid-linewidth. the default is 1 pixel.

3.1.2.10 set_grid_color()

```
void multiplot::Multiplot::set_grid_color (
            float r,
            float g,
            float b )  [inline]
```

sets the grid color

3.1.2.11 set_linewidth()

```
void multiplot::Multiplot::set_linewidth (
            float width )  [inline]
```

changes current line width.

3.1.2.12 set_pointsize()

```
void multiplot::Multiplot::set_pointsize (
            float psize )  [inline]
```

changes current point size.

3.1.2.13 set_scaling()

```
void multiplot::Multiplot::set_scaling (
            enum MP_SCALING sc,
            float x_min = -10,
            float x_max = 10,
            float y_min = -10,
            float y_max = 10 )  [inline]
```

changes the (auto-)scaling behaviour of the multiplot window. you can choose between MP_AUTO_SCALE MP_↩ AUTO_SCALE_EQUAL MP_FIXED_SCALE

3.1.2.14 set_scrolling()

```
void multiplot::Multiplot::set_scrolling (
             int max_points_to_plot )  [inline]
```

changes scrolling behaviour for current trace - see class Trace for details.

3.1.2.15 set_title()

```
void multiplot::Multiplot::set_title (
             const std::wstring & title_ )  [inline]
```

sets the window title.

3.1.2.16 trace()

```
Trace& multiplot::Multiplot::trace (
             int _trace )  [inline]
```

sets the current trace. traces are numbered from zero to N. memory for the traces is automatically allocated.

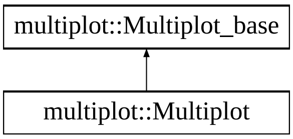
The documentation for this class was generated from the following file:

- multiplot.h

3.2 multiplot::Multiplot_base Class Reference

```
#include <multiplot.h>
```

Inheritance diagram for multiplot::Multiplot_base:



Public Member Functions

- Multiplot_base (int x, int y, int w, int h, const std::wstring &ttitle, bool fullscreen)
- void show ()
- bool **check** ()
- unsigned int **w** ()
- unsigned int **h** ()
- bool **valid** ()
- void **valid** (bool v)
- virtual void **draw** ()
- void **set_caption** (const std::wstring &t)
- void redraw ()

Protected Member Functions

- LRESULT **WndProc** (UINT uMsg, WPARAM wParam, LPARAM lParam)
- bool **CreateGLWindow** (int x, int y, int width, int height, const std::wstring &title, BYTE bits=0, bool fullscreenflag=false)
- void **DestroyGLWindow** ()

Static Protected Member Functions

- static LRESULT CALLBACK **StaticWndProc** (HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
- static LRESULT CALLBACK **window_handler** (HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam)

Protected Attributes

- unsigned int **width**
- unsigned int **height**
- bool **valid_**
- bool **active**
- bool **fullscreen**
- HDC **hDC**
- HGLRC **hRC**
- HWND **hWnd**
- HINSTANCE **hInstance**

3.2.1 Detailed Description

class Multiplot_base is for low level Window handling and creates an OpenGL Context.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Multiplot_base()

```
multiplot::Multiplot_base::Multiplot_base (
             int x,
             int y,
             int w,
             int h,
             const std::wstring & ttitle,
             bool fullscreen )  [inline]
```

this constructor tells multiplot where to put the window on the desktop in pixel-coordinates(x,y) and with wich width and height (w,h)

3.2.3 Member Function Documentation

3.2.3.1 redraw()

```
void multiplot::Multiplot_base::redraw ( )  [inline]
```

call redraw to refresh the window and to redraw all traces.

3.2.3.2 show()

```
void multiplot::Multiplot_base::show ( )  [inline]
```

call show() to make the window visible only needed if using FLTK as window-creation backend.

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

- multiplot.h

3.3 multiplot::Multiplot::Point2d Class Reference

Public Member Functions

- **Point2d** (float xx, float yy, float rr=1, float gg=1, float bb=1, float _lwidth=1.0, float _point_size=0.0)

Public Attributes

- float **x** = 0.0f
- float **y** = 0.0f
- float **r** = 1.0f
- float **g** = 1.0f
- float **b** = 1.0f
- float **point_size** = 0.0f
- float **line_width** = 1.0f

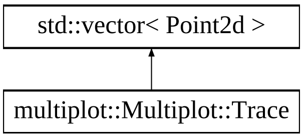
The documentation for this class was generated from the following file:

- multiplot.h

3.4 multiplot::Multiplot::Trace Class Reference

```
#include <multiplot.h>
```

Inheritance diagram for multiplot::Multiplot::Trace:



Public Member Functions

- void **draw** (Point2d &minimum, Point2d &maximum, Point2d &scale, Point2d &offset)
- void plot (const float x, const float y)
- void color3f (float r, float g, float b)
- void set_linewidth (float width)
- void set_pointsize (float psize)
- void set_max_points (int mx)
- void set_scrolling (int max_points_to_plot)
- void clear ()

Public Attributes

- unsigned int **max_points**
- bool **scroll**
- unsigned int **pos**
- float **cur_col** [3]
- float **cur_line_width**
- float **cur_point_size**

3.4.1 Detailed Description

class Trace describes a single Trace. A Multiplot-Window can contain an unlimited number of Traces.

3.4.2 Member Function Documentation

3.4.2.1 clear()

```
void multiplot::Multiplot::Trace::clear ( )  [inline]
```

clear() removes all points from the trace. the trace is empty afterwards and can be filled with plot(x,y) again.

3.4.2.2 color3f()

```
void multiplot::Multiplot::Trace::color3f (
             float r,
             float g,
             float b )  [inline]
```

sets the current drawing color in rgb format. r,g,b are in the range [0..1]

3.4.2.3 plot()

```
void multiplot::Multiplot::Trace::plot (
             const float x,
             const float y )  [inline]
```

plot a point at (x,y) to the currently active trace. you may switch the trace with a call to trace(int _trace)

3.4.2.4 set_linewidth()

```
void multiplot::Multiplot::Trace::set_linewidth (
             float width )  [inline]
```

call set_linewidth to change the thickness of the traces. the default value is 1 pixel, if you set the linewidth to zero, no lines are drawn. this is usefull to create scatter-plots.

3.4.2.5 set_max_points()

```
void multiplot::Multiplot::Trace::set_max_points (
             int mx )  [inline]
```

set the maximum number of points to be plotted. this is useful to avoid slow drawing of your trace. if you have 1000 plot-points and set the number of max_points to 100, then only every tenth point gets plotted.

3.4.2.6 set_pointsize()

```
void multiplot::Multiplot::Trace::set_pointsize (
             float psize )  [inline]
```

this function sets the size of the plot-points. the default value is zero, so no points are drawn at all. if you wish to create a scatter-plot, set the pointsize to a value bigger than zero and the linesize to zero.

3.4.2.7 set_scrolling()

```
void multiplot::Multiplot::Trace::set_scrolling (
             int max_points_to_plot )  [inline]
```

if you call set_scrolling with a positive number of points to be plotted, your graph will scroll left out of the plot-window as you add new plot-points. Zero or a negative number disables scrolling.

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

- multiplot.h

Index