Jun 05, 18 14:07 **GirderSize.cpp** Page 1/1

```cpp
#include "GirderSize.h"
#include "Math.h"
#include "ObjectSizes.h"

float GirderSize::getGirderWidthMeters(int size, int angle){
    angle = GirderSize::normalizeAngle(angle);
    return Math::cosDegrees(angle) * size + Math::sinDegrees(angle) * girder_height;
}

int GirderSize::getGirderWidthPixels(int size, int angle){
    return SCALE_FACTOR * GirderSize::getGirderWidthMeters(size, angle);
}

float GirderSize::getGirderHeightMeters(int size, int angle){
    angle = GirderSize::normalizeAngle(angle);
    return Math::sinDegrees(angle) * size + Math::cosDegrees(angle) * girder_height;
}

int GirderSize::getGirderHeightPixels(int size, int angle){
    return SCALE_FACTOR * GirderSize::getGirderHeightMeters(size, angle);
}

int GirderSize::normalizeAngle(int angle){
    return angle > 90 ? 180 - angle : angle;
}
```

Jun 05, 18 14:07 **GirderSize.h** Page 1/1

```cpp
#ifndef __GIRDERSIZE_H__
#define __GIRDERSIZE_H__

class GirderSize{
    private:
        //Normaliza el angulo entre 0 y 90
        static int normalizeAngle(int angle);
    public:
        //Devuelve el ancho de una viga en metros
        static float getGirderWidthMeters(int size, int angle);

        //Devuelve el ancho de una viga en pixeles
        static int getGirderWidthPixels(int size, int angle);

        //Devuelve el alto de una viga en metros
        static float getGirderHeightMeters(int size, int angle);

        //Devuelve el alto de una viga en pixeles
        static int getGirderHeightPixels(int size, int angle);
};

#endif
```

Jun 03, 18 21:22 **Position.cpp** Page 1/1

```cpp
#include "Position.h"
#include <cmath>

#define FACTOR 100

Position::Position(float x, float y): x(x), y(y){}

Position::~Position(){}

bool Position::operator==(const Position& other){
    return (int)(this->x * FACTOR) == (int)(other.x * FACTOR) && (int)(this->y * FACTOR) == (int)(other.y * FACTOR);
}

float Position::getX() const{
    return this->x;
}

float Position::getY() const{
    return this->y;
}
```

May 28, 18 18:21 **Position.h** Page 1/1

```cpp
#ifndef __POSITION_H__
#define __POSITION_H__

/* Clase que se encarga de representar posiciones en el plano */
class Position{
    private:
        float x;
        float y;

    public:
        /* Constructor */
        Position(float x, float y);

        /* Destructor */
        ~Position();

        /* Devuelve true si las dos posiciones son iguales */
        bool operator==(const Position& other);

        /* Devuelve el valor en X de la posicion */
        float getX() const;

        /* Devuelve el valor en Y de la posicion */
        float getY() const;

};

#endif
```

```
1  #include "ScrollHandler.h"
2  #include <gtkmm/adjustment.h>
3  #include <glibmm/main.h>
4
5  #define SPACE_TO_SCROLL 20
6  #define SCROLL_INCREMENT 25
7
8  ScrollHandler::ScrollHandler(Gtk::ScrolledWindow& window): window(window), last_
   mouse_position(SPACE_TO_SCROLL * 2, SPACE_TO_SCROLL * 2), mouse_in_window(false)
   {
9      this->window.add_events(Gdk::POINTER_MOTION_MASK);
10     this->window.add_events(Gdk::ENTER_NOTIFY_MASK);
11     this->window.add_events(Gdk::ENTER_NOTIFY_MASK);
12     this->window.signal_motion_notify_event().connect(sigc::mem_fun(*this, &Scro
   llHandler::mouseMotionEvent));
13     this->window.set_policy(Gtk::POLICY_NEVER, Gtk::POLICY_NEVER);
14
15     this->window.signal_enter_notify_event().connect(sigc::mem_fun(*this, &Scrol
   lHandler::mouseEntered));
16     this->window.signal_leave_notify_event().connect(sigc::mem_fun(*this, &Scrol
   lHandler::mouseLeft));
17     Glib::signal_timeout().connect(sigc::mem_fun(*this, &ScrollHandler::scroll),
    50);
18 }
19
20 ScrollHandler::~ScrollHandler(){}
21
22 bool ScrollHandler::mouseMotionEvent(GdkEventMotion* motion_event){
23     this->last_mouse_position = Position(motion_event->x, motion_event->y);
24     this->mouse_in_window = true;
25     return true;
26 }
27
28 bool ScrollHandler::mouseEntered(GdkEventCrossing* crossing_event){
29     this->mouse_in_window = true;
30     return true;
31 }
32
33 bool ScrollHandler::mouseLeft(GdkEventCrossing* crossing_event){
34     this->mouse_in_window = false;
35     return true;
36 }
37
38 bool ScrollHandler::scroll(){
39     int window_width = window.get_hadjustment()->get_page_size();
40     int window_height = window.get_vadjustment()->get_page_size();
41
42     if (!this->mouse_in_window){
43         //El mouse esta fuera de la pantalla
44         return true;
45     }
46
47     if (last_mouse_position.getX() < SPACE_TO_SCROLL){
48         //Scroll a la izquierda
49         this->window.get_hadjustment()->set_value(this->window.get_hadjustment()
   ->get_value() - SCROLL_INCREMENT);
50     }
51
52     if (last_mouse_position.getX() > window_width - SPACE_TO_SCROLL){
53         //Scroll a la derecha
54         this->window.get_hadjustment()->set_value(this->window.get_hadjustment()
   ->get_value() + SCROLL_INCREMENT);
55     }
56
57     if (last_mouse_position.getY() < SPACE_TO_SCROLL){
58         //Scroll arriba
```



```
59         this->window.get_vadjustment()->set_value(this->window.get_vadjustment()
   ->get_value() - SCROLL_INCREMENT);
60     }
61
62     if (last_mouse_position.getY() > window_height - SPACE_TO_SCROLL){
63         //Scroll abajo
64         this->window.get_vadjustment()->set_value(this->window.get_vadjustment()
   ->get_value() + SCROLL_INCREMENT);
65     }
66
67     return true;
68 }
```

Jun 02, 18 12:22 **ScrollHandler.h** Page 1/1

```
1  #ifndef __SCROLLHADNLER_H__
2  #define __SCROLLHADNLER_H__
3
4  #include <gtkmm/scrolledwindow.h>
5  #include <gdk/gdk.h>
6  #include "Position.h"
7
8  class ScrollHandler{
9      private:
10         Gtk::ScrolledWindow& window;
11         Position last_mouse_position;
12         bool mouse_in_window;
13
14         bool mouseMotionEvent(GdkEventMotion* motion_event);
15         bool mouseEntered(GdkEventCrossing* crossing_event);
16         bool mouseLeft(GdkEventCrossing* crossing_event);
17
18         bool scroll();
19
20     public:
21         ScrollHandler(Gtk::ScrolledWindow& window);
22
23         ~ScrollHandler();
24
25 };
26
27 #endif
```

May 30, 18 20:03 **ViewPositionTransformer.cpp** Page 1/1

```
1  #include "ViewPositionTransformer.h"
2  #include "ObjectSizes.h"
3
4  ViewPositionTransformer::ViewPositionTransformer(Gtk::Layout& layout): layout(la
   yout){}
5
6  ViewPositionTransformer::~ViewPositionTransformer() {}
7
8  Position ViewPositionTransformer::transformToScreen(const Position& position) {
9      guint width, height;
10     this->layout.get_size(width, height);
11     float x = SCALE_FACTOR * position.getX();
12     float y = height - SCALE_FACTOR * position.getY();
13     return Position(x, y);
14 }
15
16 Position ViewPositionTransformer::transformToScreenAndMove(const Position& posit
   ion, float width, float height){
17     Position pos = this->transformToScreen(position);
18     Position moved(pos.getX() - SCALE_FACTOR * width / 2, pos.getY() - SCALE_FAC
   TOR * height / 2);
19     return moved;
20 }
21
22 Position ViewPositionTransformer::transformToPosition(const Position& position)
   {
23     guint width, height;
24     this->layout.get_size(width, height);
25     float x = position.getX() / SCALE_FACTOR;
26     float y = (height - position.getY()) / (SCALE_FACTOR);
27     return Position(x, y);
28 }
```

May 30, 18 20:03 **ViewPositionTransformer.h** Page 1/1

```h
#ifndef __VIEWTRANSFORMER_H__
#define __VIEWTRANSFORMER_H__

#include <gtkmm/layout.h>
#include "Position.h"

/* Clase que se encarga de transformar posiciones de la pantalla
 * en posiciones en metros */
class ViewPositionTransformer{
    private:
        Gtk::Layout& layout;

    public:
        /* Constructor */
        ViewPositionTransformer(Gtk::Layout& layout);

        /* Destructor */
        ~ViewPositionTransformer();


        /* Dada una posicion en metros, devuelve una posicion en
         * pixeles que representa una posicion de la pantalla*/
        Position transformToScreen(const Position& position);

        /* Dada una posicion en metros, la transforma en una posicion
         * para la pantalla y la desplaza */
        Position transformToScreenAndMove(const Position& position, float width,
 float height);

        /* Dada una posicion en pixeles, devuelve una posicion en metros */
        Position transformToPosition(const Position& position);
};

#endif
```

May 31, 18 12:23 **Water.cpp** Page 1/1

```cpp
#include "Water.h"
#include "Path.h"
#include "ObjectSizes.h"

Water::Water(){}

Water::~Water(){}

void Water::show(Gtk::Layout& layout){
    this->images.clear();

    size_t pos = 0;
    guint width, height;
    layout.get_size(width, height);

    while(pos < width){
        Gtk::Image image;
        image.set(IMAGES_PATH + "Water.png");
        this->images.push_back(std::move(image));
        layout.put(this->images.back(), pos, height - water_height);
        this->images.back().show();
        pos += water_length;
    }
}
```

May 30, 18 20:03 **Water.h** Page 1/1

```
1  #ifndef __WATER_H__
2  #define __WATER_H__
3
4  #include <gtkmm/image.h>
5  #include <gtkmm/layout.h>
6  #include <vector>
7
8  class Water{
9      private:
10         std::vector<Gtk::Image> images;
11
12     public:
13         Water();
14
15         ~Water();
16
17         void show(Gtk::Layout& layout);
18 };
19
20 #endif
```

Jun 06, 18 22:31 **Table of Content** Page 1/1

```
1  Table of Contents
2    1 GirderSize.cpp..... sheets   1 to   1 ( 1) pages   1-  1   26 lines
3    2 GirderSize.h....... sheets   1 to   1 ( 1) pages   2-  2   24 lines
4    3 Position.cpp....... sheets   2 to   2 ( 1) pages   3-  3   21 lines
5    4 Position.h......... sheets   2 to   2 ( 1) pages   4-  4   29 lines
6    5 ScrollHandler.cpp... sheets   3 to   3 ( 1) pages   5-  6   69 lines
7    6 ScrollHandler.h..... sheets   4 to   4 ( 1) pages   7-  7   28 lines
8    7 ViewPositionTransformer.cpp sheets   4 to   4 ( 1) pages   8-  8   29 lines
9    8 ViewPositionTransformer.h sheets   5 to   5 ( 1) pages   9-  9   34 lines
10   9 Water.cpp.......... sheets   5 to   5 ( 1) pages  10- 10   25 lines
11  10 Water.h............ sheets   6 to   6 ( 1) pages  11- 11   21 lines
```