PALLELSHIFT ENGINE

Engine documentation

FrancoDev
T. +54 11 6276 7256 | franco.nic.ch@hotmail.com

Control de Versiones				
Versión	**Descripción**	**Autor**	**Fecha**	**Distribución**
1.0	Inicial version	Chiaravalloti Franco	26/06/2023	Anyone

FrancoDev
T. +54 11 6276 7256 | franco.nic.ch@hotmail.com

Content

1. Introduction

1.1. Purpose

To document and deliver knowledge about my game engine.

FrancoDev
T. +54 11 6276 7256 | franco.nic.ch@hotmail.com

2. Editor State.

2.1 – Class structure:

The editor state is integrated into the main menú, allows the user to create their own game map, save and load it into the main game.
Its composed of:

2.1.1 – Import:

```
class State;
class Gui;
class PauseMenu;
class TileMap;
```

The State class has the window and config information that the Editor will use in order to build the window, allocate the TileMap and all the features.
The Gui class has the button builders and general action, also the rectangle that we will use to generate the editor tile selector.
The PauseMenu class has the menu that overlap in order to pause the editor, inside we have Gui class buttons.
The TileMap class has the logic to create the tile map.

2.1.2 – Constructor & Destructor:

```
EditorState(StateData* state_data);
virtual ~EditorState();
```

The main constructor has a pointer to the StateData class, to resume, this class has the grid sizes, general configuration like the game resolution, etc.
This allow to set the general grid size for the map elements and the editor interface, also know the type of resolution the user chose.
The destructor is generic, destroys the buttons, pause menu, tile map and the texture selector.

2.1.3 – Variables:

```
1- sf::View view;
2- sf::Font font;
3- sf::Text cursorText;
4- PauseMenu* pmenu;

5- std::map<std::string, gui::Button*> buttons;

6- TileMap* tileMap;

7- sf::RectangleShape selectorRect;
8- sf::RectangleShape sidebar;
9- sf::IntRect textureRect;

10-gui::TextureSelector* textureSelector;


11-bool collision;
12-short type;
13-float cameraSpeed;
14-int layer;
```

1* -> The view is a SFML class that allow the user to create a basic view, we use this to build the editor screen.

2* -> font is a SFML class that allow you to create fonts for your elements.

3* -> cursorText is a simple SFML class that allow you to write text on a simple target.

4* -> The pmenu is a pointer to the PauseMenu class, this allow us to create a simple pause menu that can be build and later accessed on the editor.

5* -> “buttons” is a map variable which contains the different buttons (from the gui Button class) that we are going to use on the editor and pause menu.

6* -> the tileMap is the variable which will contains the information of the tile architecture that we will save on the file to use as our map on the main game.

7* -> selectorRect is a variable which represent the rectangle that will contains the different tiles that are able to use for the edition.

8* -> The sidebar is a rectangle which is located on the left side of the screen, has a button to hide and show the tile selector panel.

9* -> The textureRect is the variable which contains the tiles textures that we are going to use.

10* -> The textureSelector is the Gui class variable in which we are going to build the selector panel, its contains all we need for this.

11* -> The collision variable allows us to define a tile with collision.

12* -> The type variable allows us to define different tile types.

13* -> The cameraSpeed variable allow us to define the speed of the camera movement inside the editor.

14* -> The layer variable allow us to define the layer in which the tile is located cause we can have tile over tile.

2.1.5 – Functions:

Initialization functions:

```
void initVariables();
void initView();
void initBackground();
void initfonts();
void initText();
void initKeybinds();
void initPauseMenu();
void initButtons();
void initGui();
void initTileMap();
```

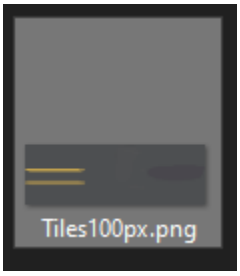
Functions:

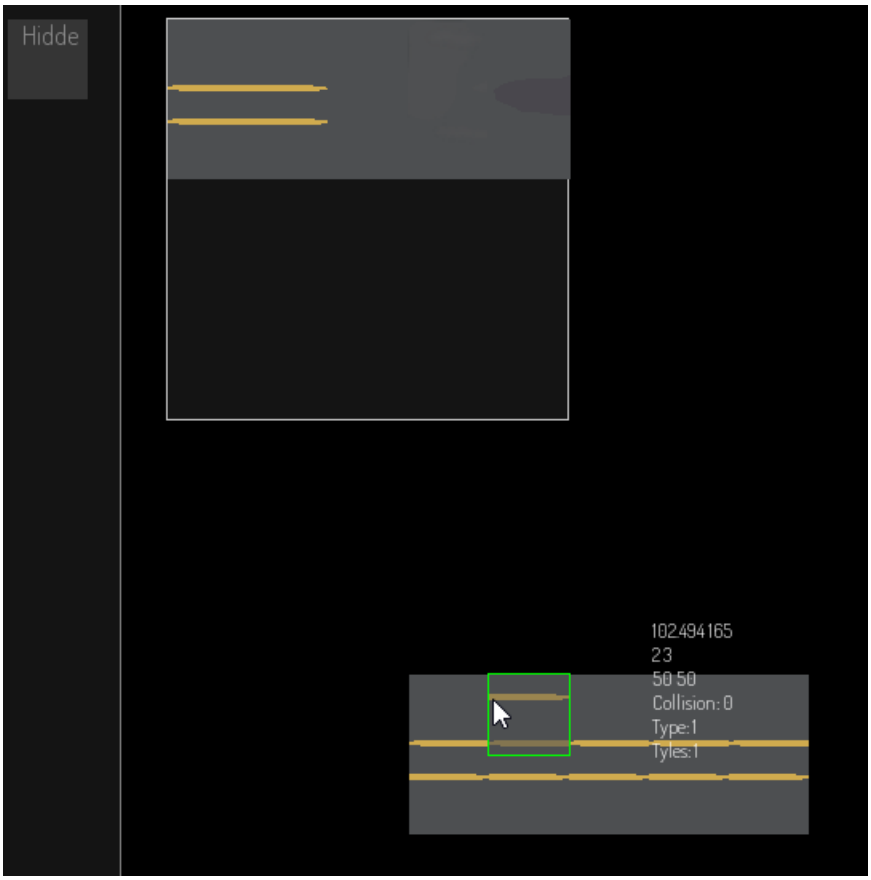
```
void updateInput(const float& dt);
void updateEditorInput(const float& dt);
void updateButtons();
void updateGui(const float& dt);
void updatePauseMenuButtons();
void update(const float& dt);
void renderButtons(sf::RenderTarget& target);
void renderGui(sf::RenderTarget& target);
void render(sf::RenderTarget* target = nullptr);
```

- **initVariables**: This function initializes all the important variables states.
- **initView**: This function initializes the view, set the sizes for this, using the state data information (resolution width and height), and set the center of the screen.
- **initBackground**: This function was designed to manipulate a special background but at the moment has not purpose will be developed later.
- **initFonts**: As the name indicate, initialize the basic fonts we are going to use, that means, load the font file.
- **initText**: Initialize the text characteristics as the Font, the text color, size, etc.
- **initKeybinds**: Load the configuration file for the editor key bind you defined.
- **initPauseMenu**: Init the pause menu class and create de buttons.
- **initButtons**: Actually is empty, designed for adding future functions.
- **InitGui**: Initialize the graphic user interface for the editor that's include the texture selector.
- **InitTileMap**: Initialize the tile map class, here you load the tile file which contains the tile template that you are going to use to build the map.

```
}

void EditorState::initTileMap()
{
    this->tileMap = new TileMap(this->stateData->gridSize, 10, 10, "Resources/Images/Tiles/Tiles100px.png");
}
```





FrancoDev
T. +54 11 6276 7256 | franco.nic.ch@hotmail.com