|  |
| --- |
| is::Level Editor |
| User guide |
|  |

Contents

[***Introduction*** 2](#_Toc37109419)

[***I.*** ***What is is::Level Editor?*** 2](#_Toc37109420)

[***II.*** ***How is::Level Editor manages objects ?*** 2](#_Toc37109421)

[***III.*** ***How to use is::Level Editor?*** 2](#_Toc37109422)

[***1.*** ***How to Create a New Level?*** 3](#_Toc37109423)

[***2.*** ***How to load a level?*** 3](#_Toc37109424)

[***3.*** ***What's the auto save?*** 3](#_Toc37109425)

[***4.*** ***What is the Load Texture option?*** 3](#_Toc37109426)

[***5.*** ***How to use the level editor?*** 3](#_Toc37109427)

[***6.*** ***Example of level created*** 5](#_Toc37109428)

Introduction

The purpose of this document is to explain how the level editor is::Level Editor works, how to create levels and integrate them into the game engine.

1. What is is::Level Editor?

is::Level Editor is the level editor of the is::Engine, it classifies the level objects into **5 types**:

* **Blocks :** They serve as block mask, elevator (movable platform) anything that is bound to the solid platforms
* **Bonus**: Used to add health, player items, bonus life, etc.
* **Enemies** : Represents the enemies of a level
* **Player**: Allows to position the player in the level
* **Gameplay Obj** Use to add the objects that manage a level mechanism (goal object, switch, controller objects)
* **Tiles** : Used to add decoration objects that will later turned into tile map by the game engine

1. How is::Level Editor manages objects?

* Each object in the editor is associated with a **unique number** that **identifies** it.
* Each **object type** is an **interval** in which are **grouped objects**.

The intervals of each object type:

* Block objects starts from 0 to 99
* Bonus objects starts from 100 to 199
* Enemy objects starts from 200 to 299
* The Player Object starts from 300 to 399
* Game Play objects starts from 400 to 499
* Tiles objects starts from 500 to ∞
* Each **type of object** is linked to **a texture**. The textures allow to define **the number of object** that a **type** **of** **object** will have according to **the number of sub image** that composes it.

Here's an example to better understand:

 This image is the **texture** of the **Bonus type object**; it is made up of **3 sub-images** so that makes **three (3) objects** for the **Bonus type object** that can be used to create a level. The textures are composed of **32x32** size image. The level editor takes care of cutting it out for you when you use it. ***(You can load your own image texture but see it later in the document)***

So here the number of the first sub picture will be 100, the second 101 and so on. Note that thanks to these numbers, the game engine can interpret the level data and determine how the levels will be constructed with the objects.

1. How to use is::Level Editor?

This part shows you how to use the different options of the level editor.

***(Please see next page)***

|  |  |  |
| --- | --- | --- |
| figure 1 | figure 2 | figure 3 |

1. How to Create a New Level?

* Click **New Level** (see **Figure 1**)
* Click on **the console** (see **Figure 2**) to specify the **size of the level** (the size is expressed in square, 1 square = 32 pixels) and the **level name** (please avoid spaces)

1. How to load a level?

* Click **Load Level** (see **Figure 1**)
* Select the **backup file** **(.SAV)** through the dialog box and click **Open**

1. What's the auto save?

Auto save allows you to automatically save the data of your level during editing without your intervention. By default if you do not select the location of the backup file for your level, it will be stored in the execution directory of the level editor.

* ***How to enable or disable it?***
* Click on **Enable Auto Save** **to turn it off and made the same gesture to activate** (see **Figure 1**)

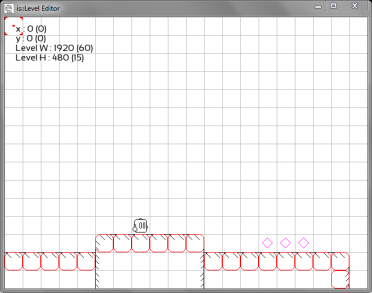
1. What is the Load Texture option?

This option (see **figure** 1) allows you to use your own textures instead of those pre-registered.

Once it's activated a proposal will be made when editing a new level (see **Figure 3**) in order to load your own images to create objects in the level editor.

1. How to use the level editor?

This section shows you how to use the tools of the editor (see **Figure 4**) to create your level.



**figure 4**

|  |  |
| --- | --- |
| 1. ***How choose an object to add to the level?***  * Press on the **right button** of the mouse to display the **choice of option** (see **Figure 5**) * Then choose an **object type** (within the first 6 lines) * Press on the **left button** of the mouse to **insert the object** * To **change the object of a type** use the **vertical wheel** of the mouse or the **keys “Up” and “Down”**  1. ***How to delete an object from the level?***  * Press on the **right button** of the mouse to display the **choice of option** (see **Figure 5**) * Click **Clear Object** to display the **delete cursor** * Click on an object to delete it   You can also use the **Delete key** to display the **delete cursor**   1. ***How to scroll through the pages of level?***  * Move the **cursor** of the mouse at the **edge of the window** (see **Figure 4**)  1. ***How to save the level?***  * Press on the **right button** of the mouse to display the **choice of option** (see **Figure 5**) * Click **Save Level** * Enter the **name of the file**, choose the following **backup location** and click **Save**  1. ***Exporting the level to the is::Engine?***  * Press on the **right button** of the mouse to display the **choice of option** (see **Figure 5**) * Click **Export Level** * Enter **the name of the file**, choose the following **backup location** and click **Save** * ***How to integrate level in game engine?*** * Put the exported file in the **levels** folder (Path: **Project/app\_src /levels**) * Open the **Level.h** file located in **levels** * Add this line to include the level in the project:   **#include "../levels/name\_of\_the\_generated\_level.h "**   * Add the level index **LEVEL\_n** in **enum LevelId** (here **n** represents the umpteenth level) * Add this line in the **getLevelMap()** function:   **case LEVEL\_n**  **: return NAME\_OF\_THE\_GENERATED\_LEVEL; break;**   1. ***How to reload a level?*** (Please note this will delete your current data)  * Press on the **right button** of the mouse to display the **choice of option** (see **Figure 5**) * Click **Reload Level**  1. ***How to hide the grids of the level editor***  * **Press F3 to hide or show the grids**  1. ***Keyboard shortcuts***   Use these keyboard shortcuts to move faster when you edit a level:   * **Key B** to select a Block object * **Key I** to select a Bonus object * **key E** to select an Enemy object * **key P** to choose the player * **Key G** tochoose a Gameplay object * **Key T** to select a Tile object * **Delete key** to display the delete cursor * **Ctrl + S button** to save the level * **Ctrl + R key** to reload level * **Ctrl + H button** to export level | figure 5 |

1. ***Example of level created***

You can find examples levels (which were used for the demo of the game engine) in the **level\_backup** directory.