CSS450 – Computer Graphics

UFOh No Level Editor

User Manual and Final Report

Ryan Hoaglan, Cameron Rahman, Sid Maxwell

Table of Contents

Introduction

The UFOh No! Level Editor will be designed to allow you to create your own levels to play in with the UFOh No video game. You will be able to create, modify, and save your maps in xml format that can be loaded by the video game application. These levels can then be reloaded into the Level Editor to be modified and saved again.

User Manual

All aspects of the UFOh No! Level Editor are controlled through mouse input. The Level Editor consists of two controllable windows, allowing both editing and room selection.

Mouse Button Configuration

Left Click: Places a new object in the level/ select the object under the pointer
Right Click: Remove the current object from a grid space
Left Click/Move: When selector is the current tool, move an object from one compatible grid space to another

Menu

The menu provided in the main Editor window to allow you to save, load, exit, and clear.

File

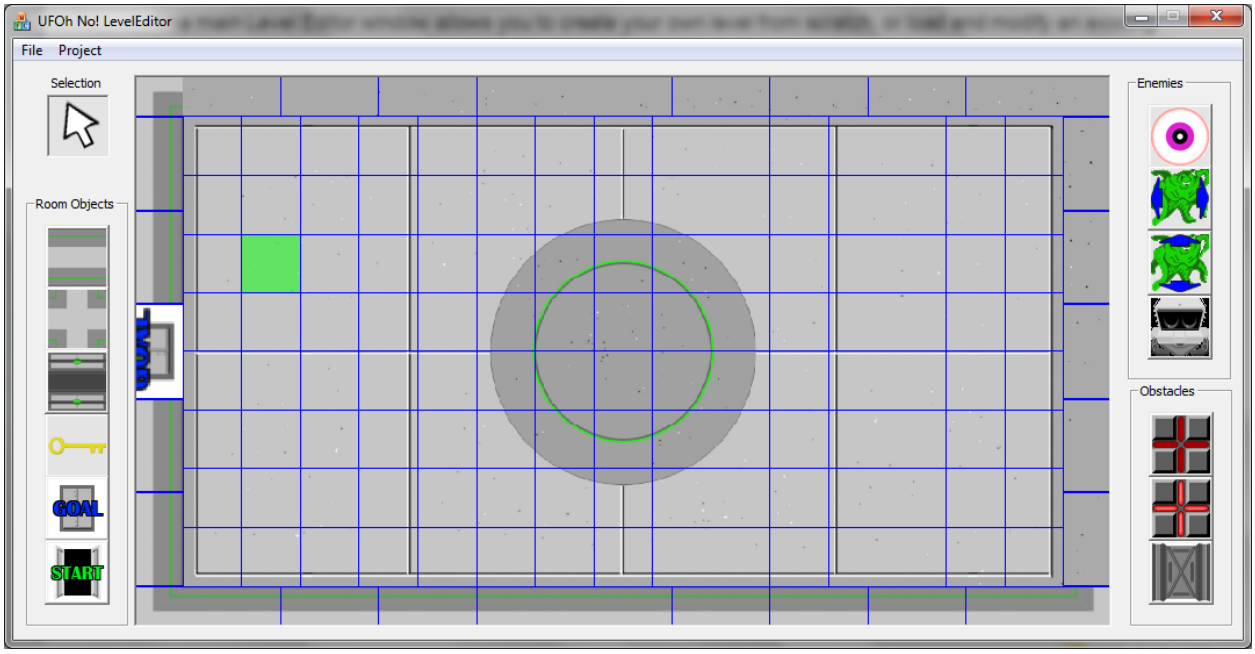
Save/Save As: Saves your current level configuration into an XML file to be read in UFOh No! the video game
New: Clears the level to start from a new template
Open: Opens a created level XML file into the editor to modify its contents
Exit: Exits the editor

Project

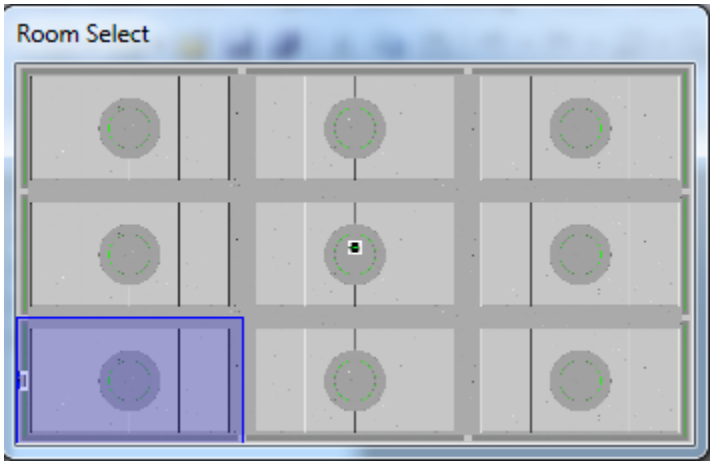
Reset Room: Resets the currently selected room, clearing all objects placed inside the main room grid area
Reset Level: Resets the entire level, including all walls and doors that are not present by default

Window Layout

The main Level Editor window allows you to create your own level from scratch, or load and modify an existing XML level file.



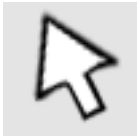
The Room Select floating window allows you to change the current room to any within the level through a left mouse button click. To change rooms, either select the blue room selector with your left mouse button and drag it to a new room location, or simply click on the new room.



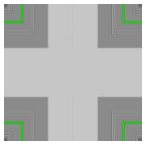
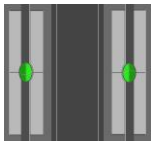
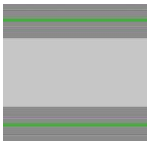
Creating Your Level

When you first open up UFOh No! Level Editor, you are presented with an empty level. However, some basic, required level objects are preplaced within the level. There is a level start, which is the placement of your hero at the start of the level, as well as a level goal. In addition, the outer walls are already supplied, keeping the hero in bounds.

While preplaced, the start and goal are able to be moved to a new location. To do this, simply select the white selector arrow icon with your left mouse button, and then left click and drag on the start or goal and move it to a new location. You can also move the start or goal by selecting either the start or goal icon to the left of the Editing window, and then left clicking on a new location within the map. There are restrictions on where a start and goal can be placed. A start can be placed on either an outer wall, or on an inner room grid space. A goal can only be placed on an outer wall.



Between each room, you can add walls, doors, corner pieces, and even open doorways. These make up the boundaries between rooms.

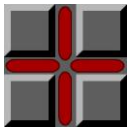




Within each room, you can place a variety of objects, from aliens to blocks, as well as keys. Each different object can be selected by left clicking on an object icon to either side of the Editor window. To place an object inside a room, left click on an available grid spot. Each left click operation places one of the currently selected objects onto the level.







When attempting to place a new object in the level, you are presented with either a green or a red grid space. All green grid spaces are available to place the selected object. When presented with a red grid space, either the space is currently taken by another object, or it is not a compatible object for that grid space.

Design of Objects

The primitives in our world consist of either a background texture or a room object, and they all consist of standard rectangles. Each rectangle primitive is placed on the world coordinate system that is defined as a 3x3 grid of 9 rooms, with each room being 100 x (100 x 9 / 16) to keep the 16:9 ratio of the game. Whenever a primitive object, such as an alien object, is moved, the center is changed in our model. Drawing is based on the

center, determining the four draw points based on this. For each vertical object, like a door or a wall piece along a vertical wall, rotation is handled through rotation matrix transformation. This allows the same object as a horizontal wall piece to appear as a vertical wall.

Each world object, with the exception of the backgrounds, is placed in an object array data structure. There are two array structures in the model, an object array and a wall/corner array. Each one stores primitive information needed when outputting the level to html format, such as room number, x, and y position.

Evaluation

Due to the nature of the project, there are no known bugs at this time. All project operations are per design.

However, while testing the Level Editor, we were able to find limitations in the Editor that will need to be addressed for a future release. While creating levels and testing the editor, we realized that some aspects that were possible in the hard coded game levels were not possible in the editor. Originally made as design decisions to help with user error checking, we only allowed the start and goal to be added to the outer walls (unless the user places the start in a middle grid space). Placing either of these two objects on an inner wall is not possible. We realized that our game had a goal on an inside wall. Our implementation made sense to us, not allowing two paths to the same goal doorway. This implementation does limit the user if they decided to not use all of the rooms, and would like to end the level in a non-outside room. In addition, our game contained some doors that were added to the inner grid spaces. In the Level Editor, doors were limited to wall grid spaces. And finally, we realized that we hid some keys underneath blocks. In the Level Editor, we made a design decision to restrict each grid space to contain only one object. This will need to be adjusted for hidden keys.

We would like to address all of these issues in a future release, to bring back all of the functionality that was present when we hardcoded the initial three game levels.