Styx and Stones: Level Editing Flow and Architecture

Devin McGinty
dlm348@drexel.edu

February 11, 2016

Contents

1 Overview

Styx and Stones is a 2D platforming game created by the FoodTaste Drexel Senior Project Team consisting of members

- Jinu Jacob
- Brendan Kelley
- Bailey Myers-Morgan
- Devin McGinty
- Le Nguyen
- Nicole Vecere

The purpose of this document is to demonstrate the process of designing levels and loading them into an instance of the Styx and Stones software program for the user to interact with. In the rest of this document the term "the game" refers to the "Styx and Stones" game program. The general process is to design the level in the Tiled 2D map editor, make the level file readable in Unity using the Tiled2Unity command line tool, and read the file using a C# deserializer embedded in the Unity engine.

2 Architecture

2.1 Tiled Map Editor

Tiled is a free and open source 2D tile-based map editor. Tiled allows the person in the role of "level designer" to load in 2D assets and design a level. 2D asset files are formatted as an image file with individual assets formatted as a bitmapped sprite conforming to a fixed-size grid. Tiled uses an intuitive drag and drop layer-based interface for level design. Styx and Stone will implement layers as defined below. Note that these descriptions mention game mechanics that are handled by the game engine in Unity

Background
These are visuals that the player's character can not interact with. Unless specifically defined, background assets will appear behind all other assets.

Surfaces
These are dimensional platforms. Except for certain circumstances, the player can not clip through these platforms on collision.. Common types of surfaces are walls and floors.

Collectibles
Collectible objects are drawn when the level is loaded, and disappear when the player collides with them. Their purpose is logically defined in the game engine.

Switches

Each switch will have a sprite consisting of two graphics. Switches are toggled by the player's character as defined in the game engine. A switch has a boolean value, and the graphic displayed is based on the state of this value. Switches are initialized when the level is loaded.

Other Interactive

There may be other interactive elements in the game, including doors and AI-controlled enemies.

Levels designed using Tiled are exported as `.tmx` files. TMX (Tile Map XML) is a tag-based object format. A level's TMX file defines the tileset file to use, graphical information about the level, and the layout data for the level. It should be noted that Tiled is a design tool rather than a development tool. Therefore, it is not included in the development repository for Styx and Stones.

2.2 Tiled2Unity Deserialization

Tiled2Unity is a community-made open source tool for the Unity game development engine. Tiled2Unity provides deserialization for TMX files to make them readable by Unity. Primitive object interaction is also defined using Tiled2Unity, including sprite transitions and collisions. The mechanics of these behaviors are defined in Unity.

2.3 Unity Game Development Engine

Unity is cross-platform engine and software development kit for game development. All game logic is defined within Unity.

3 Loading Levels

The game engine defines a **Level Select** menu. At this menu the player is presented with levels that they may choose to load. Doing so displays a temporary "Loading" screen. At this point the game engine creates an instance of a `Level` class.

The `Level` constructor is passed a hashed value from the game engine corresponding to the level that the player selected. These values are saved in a hash table with ordered keys. The hash key corresponds to a `.tmx` level file.

The constructer then calls a `loadLevel` method that locates this file. If the file is not found in the data store, the program returns to the **Level Select** screen and displays an error to the user. If the file is found, then it is loaded and deserialized so the level can be stored in memory. The `Level` object is responsible for the state of the level and the graphics in the level.

Once the level is completed (as defined by the mechanics in the game engine) the program displays the "Loading" screen. At this point the destructor method is called on the existing "Level" object, which will free all of the currently-used assets and attributes from memory. The next ordered hash key is loaded, and another `Level` object will be created.

Additionally, user may enter a **Pause** menu while a level is loaded by invoking a corresponding "pause" event. At this point the user may exit the level, in which case the `Level` destructor is called.

The `Level` class may be implemented as a singleton. This decision has not yet been made.

4 Current Challenges

Due to certain dynamic elements within the game, the following behaviors will need to be more tightly defined:

- Surfaces that the player can pass through (ignoring collisions)
- Nonstationary surfaces
- Level completion