Project Report

Jiaying Ye

February 12, 2016

Contents

1 About this project

This is a city-builder game where the player's objective is to build a colonization base on another planet.

All used libraries and theirs purposes are listed below.

Assimp	Load more formats of 3D models
irrKlang	Play sounds in the background
INIReader	Read define.ini file for each object
zlib	Dependence of irrKlang
imgui	Debug panel

2 How to Build/Run

The project is tested on Linux environment. Running on Lab environment is recommended.
*premake*4 is provided to generante $Makefile$.
To build th project:

```
premake4 gmake
make clean
make
```

To run the project:

```
./run
```

"run" is a simple bash script to setup the dynamic library path in order to run the project. If "run" doesn't have permission to execute, please use chmod to add permission.

3 How to Play/Control

(Be patient waiting the game to load.)

- 1: Building grid
- 2: Total resource display (Energy, Water, Ore)
- 3: Building buttons (Upgrade, or choose to build a miner/solar panel/luqid tank)
- 4: Mouse over popup display (shows the resource used/generated by the building)
- 5: Game Menu (Save, Load, Quit)
- 6: Debug Menu (Display FPS, cheat in resource, fast forward time)

Control:

- LMB, click to select a position on the grid
- MMB, scroll to zoom in/out camera
- RMB, hold and drag to rotate camera
- F9, toggle GamePlay overlay

Construct a building:
(Mouse over a building type can display its construction cost)

- Step 1, Left mouse click and choose a type of building from (3)
- Setp 2, Left mouse click and place it on the grid

4 Major features:

Beside all the features from the objectives and the features you can easily spot in the game.
There are many more in the background.

- Separated threads for each components in MVC.

  To achieve better respond time. And the biggest advantage is that the UI and background are independent so when loading, the program is not frozen.

- Modularized game object storage/definition.

  All game objects (terrain, mesh, building, overlay button, texture, 3D model, sound) are stored in *GameData* folder.
  They can be easily replaced/updated.
  Especially for terrain/mesh/building object. A $define.ini$ file is used to define each object, in terms of its mesh count, original transformation, resource cost/gain.
  In such way, by changing the files in *GameData* folder, I can make a completely different map or gameplay experience without ever touching the code! (DLC?)

5 Major bugs:

Although all listed Objectives are completed, but the game is so complicated that some more features should be in the game are not implemented.

- Grid is only a flat surface on a limited area.

  With no pre-processing of the terrain mesh, it takes a long time to determine the height and angle of each cell in grid, if we want to put the grid along side the terrain. So I give this up and only place a limited flat grid.

- Models and textures are not perfectly suitable for a plant colonization scenario.

I underestimated the workload of make custom 3D models and textures. In the end, I realized that the project has become "making a very low-end game engine, and use that to build a game". I am not a good artist and I simply can't make myself some good looking 3D models and textures.

- Game resource gain and cost is not totally balanced yet.

  For the same reason above, the game is not balanced. But this is easy to do since this is just math. And everything about that can be changed in the *define.ini* file, thanks to the object storage I designed.