Table of Contents

[File Structuring 2](#_Toc321085943)

[World File Structure 2](#_Toc321085944)

[Biome File Structure 2](#_Toc321085945)

[Engine Design and Optimisation 3](#_Toc321085946)

[Viewport Culling 3](#_Toc321085947)

[Content Generation 4](#_Toc321085948)

[Biome Generation 4](#_Toc321085949)

[Block Generation 4](#_Toc321085950)

[Tree Generation 5](#_Toc321085951)

[Physics Engine 5](#_Toc321085952)

[Kinematics 5](#_Toc321085953)

[Collision Detection 5](#_Toc321085954)

[Fluid Dynamics 5](#_Toc321085955)

[Afterthoughts 6](#_Toc321085956)

[Possible Improvements 6](#_Toc321085957)

[Content Generation 6](#_Toc321085958)

[Things we did good (IMO) 6](#_Toc321085959)

# File Structuring

## World File Structure

The main idea behind the world file was to list the biomes in the world and also to determine which biome file contained the player’s default spawning position. As with all the save files in Evolution the text is human readable and follows a simple structure, this means that should the player so desire they could edit this file to adjust the properties of the world.

The naming of the file also makes it easy to find, the file name follows the following format: *Worldname*.wld

A sample of a world file with added comments explaining each line is shown below:

|  |  |
| --- | --- |
| *20000,20000* | *(the size in pixels of the world)* |
| *4* | *(the total number of biome segments in the world)* |
| *0,0,1,1,1380,1500,0,0,False* | *(biome nameId, typeId, segment, segment, width, height, positionX positionY, default spawn?)* |
| *0,1,0,0,1380,780,690,605,True* |
| *0,1,1,0,1380,780,2070,605,False* |
| *0,1,1,0,1380,780,2760,605,False* |

## Biome File Structure

The main idea behind the text file was that I wanted the player to be able to edit the worlds biomes manually if they so desired, so the text file was designed with readability in mind. Each block is represented by a two character string so that the type of block that is in a particular position is obvious to the player. I used the ‘|’ (pipe) character as a delimiter to again give the appearance of a block like shape to the reader.

The naming of the file also makes it easy to understand its location in world space, the file name follows the following format: *biomeName\_biomeType(segmentXsegmentY)*.bio

A sample of a biome file with added comments is shown below:

*24,20 (number of blocks along x-axis, number of blocks along y-axis)*

*db|db|db|db|db|mb|mb|db|db|db|db|db|mb|db|wb|mb|db|db|db|db|db|db|db|wb|*

*db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|mb|db|wb|db|*

*db|db|mb|mb|db|db|db|db|wb|db|db|db|wb|db|db|db|db|db|db|db|db|db|db|wb|*

*db|mb|db|mb|db|mb|db|mb|wb|db|db|db|db|db|db|db|db|db|wb|db|wb|db|db|db|*

*db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|db|mb|db|db|db|db|db|*

*(Some rows were omitted since you already get the idea)*

# Engine Design and Optimisation

## Viewport Culling

Early on during the coding of the game, we soon came across a problem. This problem was drawing efficiency and viewport clipping. I came up with the solution of drawing biomes in a “segmented” fashion.

Basically we would write each biome into different segments, the size of these segments would be determined by the width and height of the players monitor. So for a 1920x1080 screen each segment would draw approximately (1920/4) \* (1080/4) pixels... well actually the aim was to have them slightly larger in order to prevent issues when loading new segments as the player moves through the world (basically to give the computer some time to load new segment files and unload ones that no longer needed to be drawn).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  | Monitor Area 4x4 segments (Always Drawn) | | | |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Green area (6x6 segments): represents the segments outside of the monitor that will be drawn but not necessarily visible

Grey Area (>= 8x8 segments): represents the segments that will not be drawn

# Content Generation

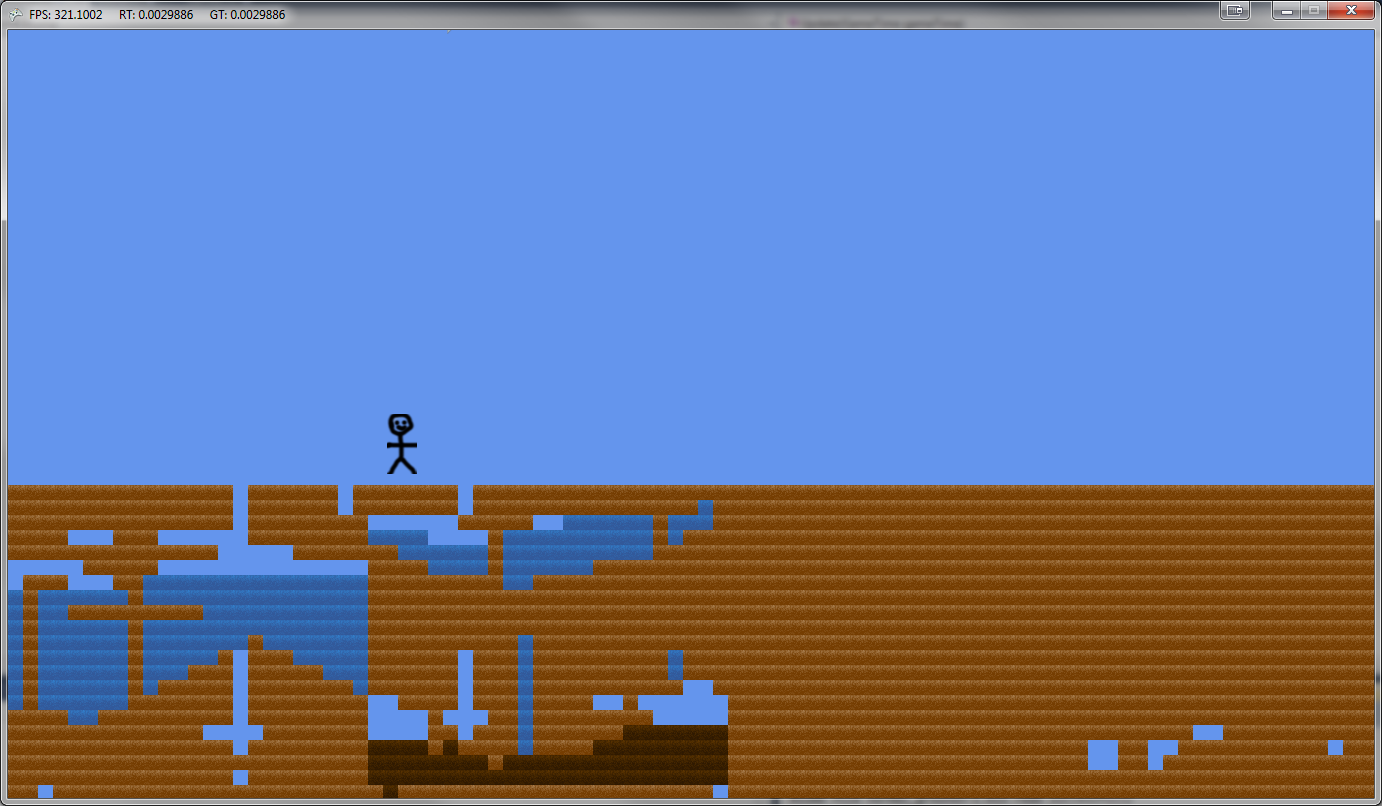
## Biome Generation

### Block Generation

Arguably the most essential part of the game itself, the idea behind block generation started out as a simple idea but soon enough we realised that it would become more complex than first anticipated.

The idea was that each biome nameId (NORMAL, DESERT, OCEAN etc) would have associated with it a list of the differing types of blocks. Each of the block types in this list would then have a percentage value assigned to it in order to represent the likelihood of that type being spawned in the biome. The percentages were out of 100 and the total percentage of all blocks in the biome added together had to equal 100 in order for this to work.

After running the game we found that this method of block generation gave quite a poor aesthetic result as there didn’t seem to be any sense to block positioning. I decided that realistically it would be much better if we instead clustered common blocks together into a diamond-like shape and an inverted triangle-like shape for water blocks.

This was a great success as we found that the world now looked much more realistic as you can see in the image below:

*As you can see this image was taken before we had the official sprites in and before we had implemented a whole host of other features.*

### Tree Generation

The idea I had to get trees to spawn in the world in sensible locations was to only allow them to spawn in a certain range of segments and in particular biomes. Basically I would read in each of these files within the range and starting from the top row check each row until I found a row that has two blocks together and air above each block.

I would then position a tree in the middle of these two blocks. The plan is to have the trees grow as time in the game passes. Each tree will have several tree blocks associated with it and a tree top (so as to make it look nice graphically). Each tree block will drop 1 wood when cut down with an axe.

# Physics Engine

## Kinematics

So far, basic kinematics has been implemented in Evolution. These include static horizontal motion (i.e. Horizontal motion without acceleration) and dynamic vertical motion (i.e. Vertical motion under the influence of gravity). In future iterations, dynamic horizontal motion will be added to allow for steady increases in speed.

The physics engine currently allows for kinematic chains; that is, full jumping arcs can be described by using both static horizontal motion and dynamic vertical motion calculations running in tandem to produce parabolic trajectories without needing to know the launch angle.

## Collision Detection

No collision detection has been implemented in Evolution yet. The current plan is to use a heuristic algorithm to only check the blocks surrounding the player for collisions. If a collision is detected, then the player is moved such that the edge of its hitbox lies flush with the hitbox of the block.

Currently we are using BoundingBox structures for the bounding boxes. These work in 3D space, however, so this must be catered for. When collision detection is fully implemented, continuous gravity will also be implemented, to allow falling from heights under the influence of gravity.

## Fluid Dynamics

No fluid dynamics have been implemented yet. The current plan for simulating fluid dynamics is to move water blocks from the top of a water plane with air gaps into empty spaces below them, and keep moving them from the top of the current plane until they form a completely flat plane wherever there is open air.

# Afterthoughts

## Possible Improvements

### Content Generation

#### Biome Generation

Having coded the biome’s width and height in pixel values rather than the number of blocks wide and high was a pretty stellar mistake on my part. This seemingly simple difference has caused a great amount of heartache and increased the code complexity by an enormous amount, getting it very near the point of being unreadable.

#### Block Generation

Another significant mistake made by me was to decide that each block in game should be 15x15 pixels in size. A number divisible by 2 would have made blocks much easier to work with since and would have enabled us to code the block position using a centre value rather than by adding 15 to each blocks position. In case you were wondering, the textures themselves are in fact not 15x15 but are larger (18x18) to allow for some uneven surfaces in the texture to give a more rugged terrain effect.

## Things we did good (IMO)