

1ST ORAL

From

Team



Romain BIESSY

Renaud GAUBERT

Aenora TYE

Erwan VASSEURE

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Communications</b>	<b>3</b>
2.1	Twitter, Facebook and YouTube . . . . .	3
2.2	Web site . . . . .	3
<b>3</b>	<b>Licencing</b>	<b>4</b>
<b>4</b>	<b>Terrain display</b>	<b>5</b>
4.1	What was done . . . . .	5
4.2	Optimizations . . . . .	5
4.3	Results . . . . .	6
4.4	Sky . . . . .	6
<b>5</b>	<b>Player actions</b>	<b>7</b>
5.1	Adding blocks . . . . .	7
5.2	Removing them . . . . .	7
<b>6</b>	<b>Terran generation</b>	<b>8</b>
6.1	A new structure . . . . .	8
6.2	Perlin 3d algorithm . . . . .	8
6.3	Biomes . . . . .	8
<b>7</b>	<b>Buildings</b>	<b>9</b>
7.1	Constructions blocks . . . . .	9
7.2	Saving and loading the structure . . . . .	9
<b>8</b>	<b>Physics</b>	<b>10</b>
8.1	Collisoins . . . . .	10
8.2	Jump . . . . .	10

<b>9 Character</b>	<b>11</b>
<b>10 AI</b>	<b>12</b>
10.1 Pathfinding . . . . .	12
10.2 A star (A*) . . . . .	12
<b>11 Conclusion</b>	<b>13</b>

# **Chapter 1**

## **Introduction**

## **Chapter 2**

# **Communications**

### **2.1 Twitter, Facebook and YouTube**

### **2.2 Web site**

## **Chapter 3**

# **Licencing**

## Chapter 4

# Terrain display

### 4.1 What was done

For the 1st oral presentation, the display system was the biggest problem we had, FPS were very low, around 30, even though we only displayed little Islands (5\*5\*5 chunks, which are 16\*16\*16 blocks tall). Therefore the game was laggy and unpleasant to play

This system was based on the idea that each blocks are composed of 6 faces that are visible or not (Not visible means not added to the scene). When we detected visible faces, we added a unique entity to the scene. However, MOGRE as any other 3d engines had troubles supporting thousands of faces at a time (around 10 000 entities).

### 4.2 Optimizations

So we came up with another idea, which was a bit risky. Basically creating an entity in Mogre is very simple :

```
ManualObject block = new ManualObject("name");
block.Begin("texture here", RenderOperation.OperationTypes.OT_TRIANGLE_LIST);

    block.Position(new Vector3(0, 0, 0));
    block.Position(new Vector3(0, 100, 0));
    block.Position(new Vector3(100, 0, 0));
    block.Position(new Vector3(100, 100, 0));

    block.Quad(0, 1, 2, 3);
block.End();
```

As you can see, we only have to give the object position and to tell to draw a quad (which is actually a shortcut to creating 2 triangles). Basically, the idea we had was to gather all visible blocks of the same material in an instance of an object (which was called multiblock). However this was not simple for 2 reasons :

- We cannot tell the material to delete a face at a certain position neither can we add new faces to it
- We wanted blocks which had more than one texture on it (for example the grass cube doesn't have the same texture on the the top face than on the bottom face)

First of all you need to understand our architecture :

As you can see here each chunk contains an array of block. At first we stored in each cell a new instance of an object (The cube object) but !we made some changes and we now !instanciet! once blocks of

each material and then store the references to the corresponding block in the array.

However with this model, we can only have one unique texture per block. Therefore, using inheritance, we came up with the following idea : in a base class, we create an abstract method called `getComposingFaces` which returns the material. In the derived class, if it needs multiple texture then it'll have to override this method, returning an array of string composed of the name of its faces which would be

This system may seem a bit complicated, and it was, but it was totally worth the cost, we increased the FPS rate by 20, it now turns around 400. And, the Islands size were trippled, getting from 4\*4 to 12\*12 (on the x, z coordinates with y the height)

## 4.3 Results

## 4.4 Sky



## Chapter 5

# Player actions

### 5.1 Adding blocks

One of the downside of gathering all faces in one object is that you can't add faces to the object. Therefore we had to think a bit outside the object. What we did was the following : When the player adds cube to the terrain, we display the face as independent objects each and remember the position he added a block. When he has added 30 blocks, we launch a new thread whose role is to recreate each multiblocks that were modified.

First he must change the list of Vector3 the instance of Multiblock has and add the blocks the player added. Then he must recreate the object and when it's done remove the old one and replace it with a new one.

C# having a thread system which is easy to handle, this part was pretty easy.

### 5.2 Removing them

On the other hand removing blocks from the scene was a tricky part. Because the function we needed was not implemented in C# we had to get it from the C++ librarys. And, what's more, the function needed pointers so we had to use the unsafe option.

The basic idea is that a Manual Object is just an abstract layer to Vertex and Vertices. Thus we had to get the vertex buffer, Lock it, get the position of the first element and from there on tell what was the face number we wanted to access, and multiply this number by the "size" of a Vector3 this would set the pointer to the right position. From there on we only had to set for a face, it's 4 coords to 0 and check how many faces were visibles.

However 2 problems arose : the first block with multiple structures, had to be dealt with in a separate way.

The second was that when removing a block, we had to refresh the surrounding blocks, thus most of the time creating faces which were not in the manual object (that was the cause of a bug). So we had to be carefull not to suppress

## **Chapter 6**

# **Terran generation**

### **6.1 A new structure**

### **6.2 Perlin 3d algorithm**

### **6.3 Biomes**

## **Chapter 7**

# **Buildings**

### **7.1 Constructions blocks**

### **7.2 Saving and loading the structure**

## **Chapter 8**

# **Physics**

### **8.1 Collisoins**

### **8.2 Jump**

## **Chapter 9**

# **Character**

## **Chapter 10**

# **AI**

### **10.1 Pathfinding**

### **10.2 A star (A\*)**

## **Chapter 11**

# **Conclusion**