Multimedia and Serious Games in Education

Carlos Oliveira
Department of Informatics
IFRJ
Rio de Janeiro, Brazil
carlos.roberto@ifrj.edu.br

***Abstract*—According to recently studies, people spend many hours a day using entertainment applications on the Internet. It poses a great opportunity for serious games that can be used on education. This is a recent research area that needs to be addressed. This paper presents an educational game and the tools that are being used to developed this game. The idea is to move away from the traditional textbook approach and make learning pleasurable. In a near future, this game will be evaluated by history and geography teachers and students.**

Keywords-serious game; history and geography learning; online game; development tools.

I. Introduction

We live in an era in which we are surrounded by information. Each day more information is produced and disseminated. This information is from different subjects and come from different sources, such as TV, newspapers and Internet. Nevertheless, Internet is the growing media in terms of user's interesting and access. If well used, information can aggregate value to products and also to people who has specific knowledge. Despite of the large amount of information received, people are quite poor at understanding and remembering information they have received out of context or too long before they can make use of it [1][2][3]. Thus, it poses a problem. How to assure the information will be remembered later?

Different from what happens in schools, games give information on demand, just in time and not out of the context of the game purposes. According to Gee [4], good games find ways to put information inside the worlds the players move through, and make clear the meaning of such information and how it applies to the world.

It is possible to use games to enhance learning at schools. Young people stay plugged on computers and other devices many hours a day. This public is very enthusiastic with technologies. They are also heavy information consumers. It is also important to consider that, in general, players read about a given subject not only inside the game, but also outside the game on websites, books, etc.

The goal of this paper is to present a game that is being developed to be used on schools. The game aim to help teachers on teaching subjects concerned to geography and history. Thus, it is presented the game's story and the tools used on the game development.

The rest of the paper is organized as follows. The game is presented in Section II. In Section III, it is presented the tools used to develop the game. Aiming to give a general idea of the game codification, it is also presented part of the Java script that contains the code to control the character of the game. Section IV concludes the paper and outlines future directions.

II. Game

The game's story takes place on Earth after an alien has an accident with his ship and fall down on our planet. In their search for the key parts of his vehicle, the character passes through well-known touristic points of our planet and knows a bit of our culture. Therefore, the aim of the game is to find the parts of the ship. During this quest, the player will go through different cities on the planet. When the character goes through Rio de Janeiro, for example, the player gets information about the city's history, climate, traditional festivals, among other information that helps the player to know the city. In this game the player also listen to popular songs on the cities where the character is.

It would be impossible to show graphics and information on all touristic cities on Earth. Thus, the game focus on major world cities, such as New York, Rio de Janeiro, London, Paris, Berlin, Rome, Jerusalem, Moscow, Tokyo, Beijing and Sydney.

We believe that the game may be used to facilitate the approach of some subjects in disciplines such as history and geography. Furthermore, the development of the game back interesting with regard to computer programming, graphs for the development of the game and the game logic creation of challenges.

III. Tools

The game is being developed as presented in Figure 1. The game has a component that stores several songs. The second component stores scenarios and their objects. The third component gives intelligence to the game. Because of this component, songs are played according to the visited city. Also, it gives intelligence to the characters.

To develop the game songs, we used the program Fruity Loops [5]. The program Blender [6] was used to create the character and objects of the game. The Unity3D [7] was used to render the scenarios, to put the objects on the scenario, and to make the logic of the game.

In this section, these tools and screenshots of them are presented.

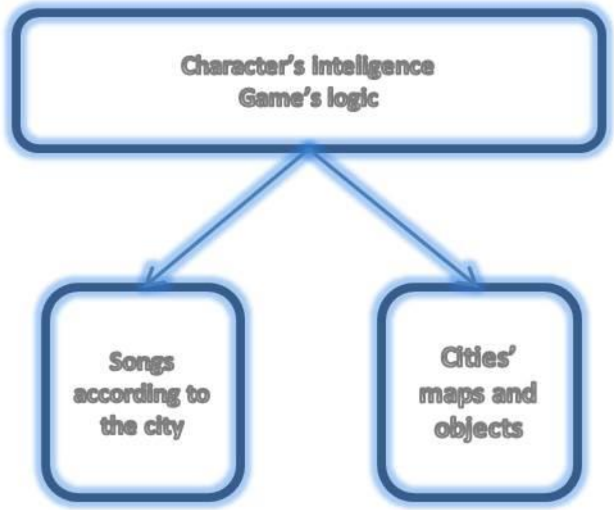


Figure 1. Game architecture.

A. *Fruity Loops*

Fruity Loops Studio resembles a complete recording studio for multimedia projects which is ideal for music demos, songs and any means of audio production. This virtual studio processes audio using an internal 32-bit floating-point engine. It can support sampling rates up to 192 khz either WDM or ASIO enabled drivers. The Mixer interface allows users for channel configurations and mixing 2.5, 5.1, or 7.1 surround sound possible. In addition, Fruity Loops studio also comes with a variety of plug-ins and generators (synthesizers) written in the program's own native plug-in architecture. However, first time users might be overwhelmed with its unlabeled icons and confusing file browser which makes the learning curve steeper. Once the users are already familiar with the program, they can make music in no time. A Fruity Loops screenshot is presented in Figure 2.

Figure 2. Fruity Loops Screenshot.

B. *Blender*

Blender is a professional free and open-source 3D computer graphics software product used for creating animated films, visual effects, art, 3D printed models, interactive 3D applications and video games. Blender's features include 3D modeling, texturing, raster graphics editing, rigging and skinning, fluid and smoke simulation, particle simulation, soft body simulation, sculpting, animating, match moving, camera tracking, rendering, video editing and composing. Alongside the modeling features it also has an integrated game engine. A Blender screenshot is presented in Figure 3.

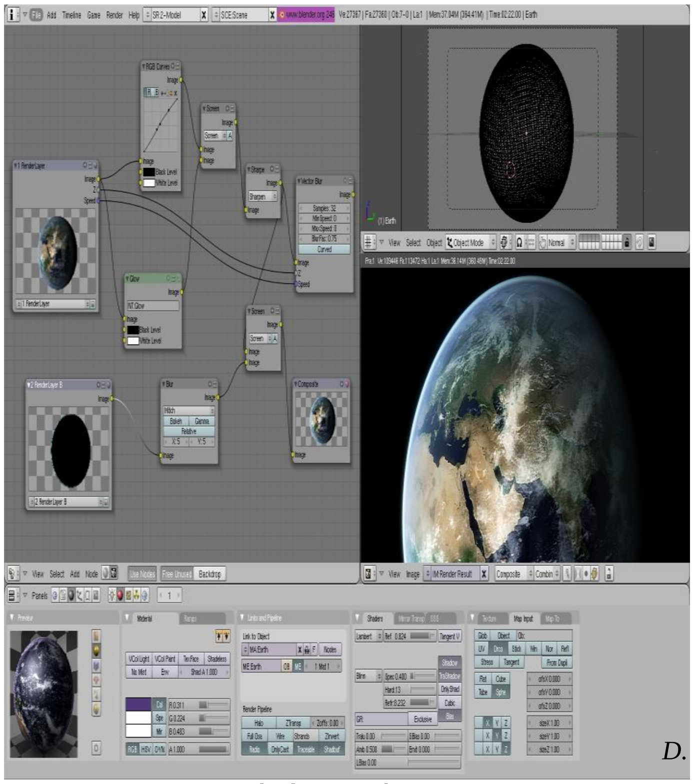
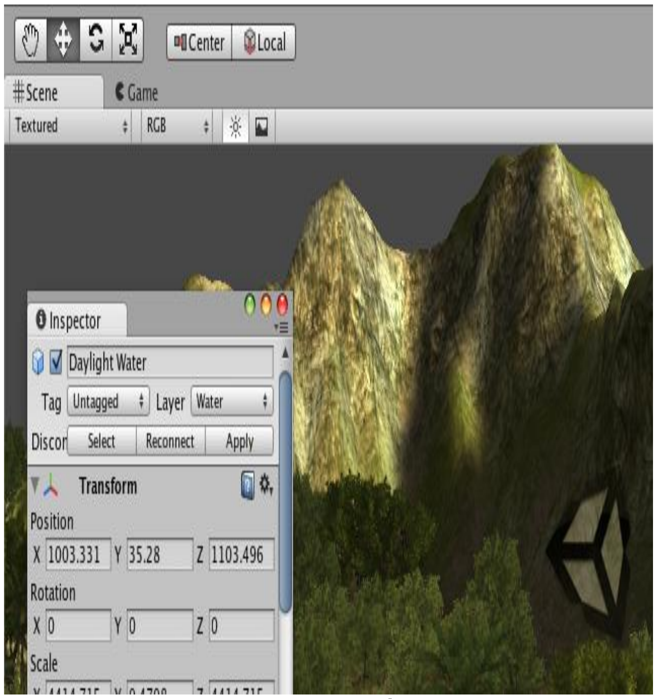


Figure 3. Blender Screenshot.

C. Unity3D

Unity is a cross-platform game creation system including a game engine and integrated development environment. It is used to develop video games for web sites, desktop platforms, consoles, and mobile devices. With an emphasis on portability, the graphics engine targets the following APIs: Direct3D on Windows and Xbox 360; OpenGL on Mac, Windows, and Linux; OpenGL ES on Android and iOS; and proprietary APIs on video game consoles. The game engine's scripting is built on MonoDevelop \cite{mono}, the open-source implementation of the .NET Framework. Programmers can use UnityScript (a custom language with ECMAScript-inspired syntax, referred to as JavaScript by the software).

Uniy3D is used to deal with the codification of the game. Unity combines the character, songs, objects and the codification of the game. A Unity3D screenshot is presented in Figure 4. The variables declaration that permits all character movements on the game are shown below.



Daylight Water
Tag Untagged Layer Water
Discon Select Reconnect Apply
Position
X 1003.331 Y 35.28 Z 1103.496
Rotation
X 0 Y 0 Z 0
Scale

Figure 4. Unity3D Screenshot.

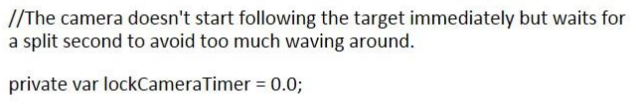
D. Scripts

In this subsection, parts of the game code are presented. Figure 5 presents initial values that defines the speed of the character when walking, running or jumping. The variable presented in Figure 6 is configured to follow the character during the game. Thus, the player controllers the character as a third-person game.

```
@script RequireComponent
(CharacterController)

public var idleAnimation: AnimationClip;
public var walkAnimation: AnimationClip;
public var runAnimation: AnimationClip;
public var jumpPoseAnimation: AnimationClip;

public var walkMaxAnimationSpeed: float=0.75;
public var trotMaxAnimationSpeed: float=1.0;
public var runMaxAnimationSpeed: float=1.0;
public var jumpAnimationSpeed: float=1.15;
public var landAnimationSpeed: float=1.0;

private var _animation:Animation;

enum CharacterState {
        Idle = 0,
        Walking = 1,
        Trotting = 2,
        Running = 3,
        Jumping = 4,}

private var _characterState:
CharacterState;

var walkSpeed = 2.0;
var trotSpeed = 4.0;
var runSpeed = 6.0;

var inAirControlAcceleration=3.0;

var jumpHeight = 0.5;

var gravity = 20.0;
var speedSmoothing = 10.0;
var rotateSpeed = 500.0;
var trotAfterSeconds = 3.0;

var canJump = true;

private var jumpRepeatTime = 0.05;
private var jumpTimeout = 0.15;
private var groundedTimeout = 0.25;
```

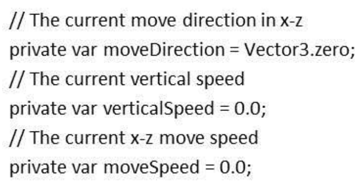
Figure 5. Variables.

Figure 7 presents the variables that allow the character to make two axis movements on the maps. All maps have collision objects that the player needs to detour. Collisions are allowed because of the variable presented in Figure 8.



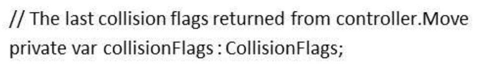
```
//The camera doesn't start following the target immediately but waits for
a split second to avoid too much waving around.

private var lockCameraTimer = 0.0;
```

Figure 6. Camera's code.



```
// The current move direction in x-z
private var moveDirection = Vector3.zero;
// The current vertical speed
private var verticalSpeed = 0.0;
// The current x-z move speed
private var moveSpeed = 0.0;
```

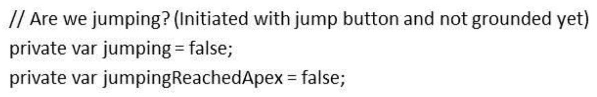
Figure 7. Character's movement.



```
// The last collision flags returned from controller.Move
private var collisionFlags : CollisionFlags;
```

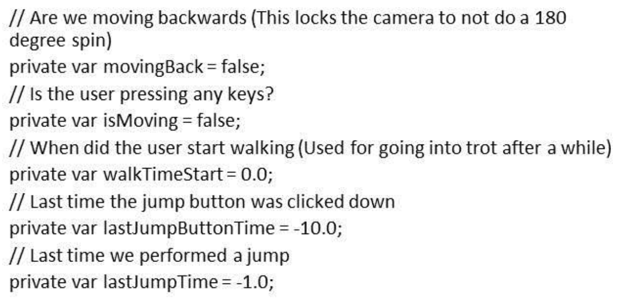
Figure 8. Collision flags.

Jumping makes possible to detour from collision objects. Initially the character is not jumping. Jumping is allowed by the variables presented in Figure 9. The character can jump on different heights. The height of jumping is defined on the variable presented in Figure 11. The height of jumping is calculated taking into account the last jump. Thus, the value of the last jump is stored on a variable presented in Figure 10. Figure 10 also presents variables that permits to show the character when jumping or making other movements.



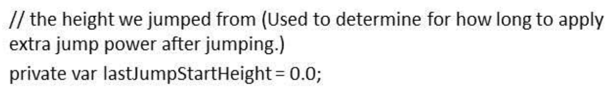
```
// Are we jumping? (Initiated with jump button and not grounded yet)
private var jumping = false;
private var jumpingReachedApex = false;
```

Figure 9. Jump button.



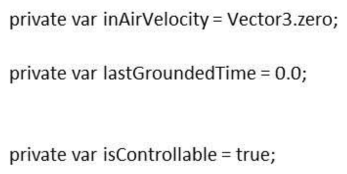
```
// Are we moving backwards (This locks the camera to not do a 180
degree spin)
private var movingBack = false;
// Is the user pressing any keys?
private var isMoving = false;
// When did the user start walking (Used for going into trot after a while)
private var walkTimeStart = 0.0;
// Last time the jump button was clicked down
private var lastJumpButtonTime = -10.0;
// Last time we performed a jump
private var lastJumpTime = -1.0;
```

Figure 10. Camera's movement.



```
// the height we jumped from (Used to determine for how long to apply
extra jump power after jumping.)
private var lastJumpStartHeight = 0.0;
```

Figure 11. Jumping.



```
private var inAirVelocity = Vector3.zero;

private var lastGroundedTime = 0.0;


private var isControllable = true;
```

Figure 12: Speed

IV. Conclusion and Future Work

We are now finishing the game development. The game aims to facilitate the teaching of different subjects, such as geography and history. We argue that it is possible to use games to enhance learning at schools because young people stay plugged on computers and other devices that run games. In this game, we also aim to allow users to modify existing maps and create others. This will permit players to produce information (knowledge) and not only consume. Our goal is to allow users to change the game with no programming knowledge.

Acknowledgment

The author acknowledges financial support from Federal Institute of Rio de Janeiro during the work presented in this paper.

References

[1] L. Barsalou: Language Comprehension: Archival Memory or Preparation for Situated Action? In: Discourse Processes, 1999, pp. 61–81.

[2] A. L. Brown: The advancement of learning. Educational researcher, 1994, pp. 4–12.

[3] A. M. Glenberg, D. A. Robertson: Indexical understanding of instructions. Discourse Processes, vol. 28, no. 1, 1999, pp. 1–26.

[4] J. P. Gee: What video games have to teach us about learning and literacy. Computers in Entertainment (CIE), v. 1, n.1, 2003, p.20-20.

[5] Introducing FL Studio 11, http://www.image-line.com/flstudio/ (Accessed 03/05/2015)

[6] Blender, http://www.blender.org/ (Accessed 03/05/2015)

[7] Unity3D, https://unity3d.com/ (Accessed 03/05/2015)

[8] MonoDevelop, http://monodevelop.com/ (Accessed 03/05/2015)