**Role Specific Training Assignment 1: Endless Runner – Ruben Geerts**

**CMV2D - 359131**This document will contain a brief description of the game I created, how I implemented certain aspects of the game, and at the end there will be a reflection on someone else his/her game implementation.   
  
My game  
  
The game is an endless runner, which means you can play indefinitely (unless you die, or the level ends). In my case, the level ends at 1000 platforms. I thought that would be enough to last for a visual representation of my infinite runner concept. In my endless runner you are a red cube. The cube is running with increasingly more speed, and needs to dodge the rock cubes in order to stay alive. There is music in the background, and you can collect coins which also make a coin sound when picked up.   
  
In my levelGeneration script I create 1000 floors by placing them in the air, and have a variable go up by 1 every time a floor is placed. This variable changes the position of the next floor, until there are 1000 floors on the ground.  
  
Further in the levelGeneration script I create random numbers, and with these random numbers I create obstacles and coins. The chance of an obstacle to appear is 1/75 (note that the numbers are **per frame** and not per second). If the random number is 1, then an obstacle will be placed. But I drew another random number, which indicates in what lane this obstacle should be placed. Chance of that is 1/3.   
The coin generation is the same thing, but with other numbers, and the coins being able to spawn in the air (I thought that would be nicer than to have everything on the floor). With the Instantiate command I created the floors, the coins and the obstacles.  
  
In the PlayerController script I placed the coin pick up sound, and the check if the player connects to a coin. If a player collides with a coin, the coin will be turned off, and a sound will be played. This is done by a OnTriggerEnter (the coin is a trigger).  
  
In the PlayerMovement script I created the movement for the player.   
In this script, the keys that are pressed are checked in code. If the player presses the left key, the player will be moved 0.11 units to the left, and vice versa for the other button. For the jumping however, the jump is done by a mathematical formula. If the Up button is pressed, the yspeed is changed to 25, which sets the formula in action. The formula is as follows:   
  
 yspeed = yspeed - (50 \* Time.deltaTime);

ypos = ypos + (yspeed \* Time.deltaTime);

if (ypos < 0.0f)

{

ypos = 0.0f;

yspeed = 0.0f;

}

transform.position = new Vector3(transform.position.x, ypos, transform.position.z);  
  
The yspeed changes to a positive number (instead of 0), and is lowered every frame. The ypos is the position of the player, and is calculated with the speed. The player can only jump when he or she is on the floor (that is the last if statement).   
In PlayerMovement I also added the collider with the obstacles. If I put the obstacles in the playercontroller script, it would not work, because then there would be two triggers instead of one. If a player collides with an obstacle, the timeScale is set to 0 so the game is paused, the music is off, and the player cannot see nor move, the Restart button scene is loaded, and the player is destroyed.  
  
In the rotator script I rotate the coins, and this rotation is based on the time passed since the last frame, and this rotation is done every frame, so the rotation looks smooth.  
  
In the StartGameOnClick script the game is launched (I click the start button), and the exact values of this script, and start button are copied to the restart button at the end of the game when you die. The Time.timeScale is set to 1 again so the game starts again, and you can play another round.