Nick Von Imhof
Nick Gu

CS175 Final Project: Torchlight

The Important Things:

- You can find our windows game executable in the "game executable (windows)" directory of the .zip file in the canvas submission.
- Just in case the canvas submission isn't working, right here is the link to our github repository: https://github.com/F1erCS/CS175-Final-Project
- Please reach out to us at nick_vonimhof@college.harvard.edu or ngu@college.harvard.edu if there are any issues with executing the game file!

When my partner and I think about computer science and specifically about computer graphics, we're immediately drawn to what made us love the field in the first place: computer games.

What We Did:

Using the Unity Engine, we created a 3D endless runner that is similar in nature to many of our first introductions into gaming. I remember in middle school playing Temple Run or Subway Surfers on my Mom's phone and being in awe of how the game was able to be virtually endless and the fun yet primitive visuals. Our game wanted to capture that same excitement with the unlimited nature of the game and the same high-score mechanism, while also providing a unique gameplay element that deals with computer graphics.

When a player loads up the game, they are given the instructions, as well as an option to select three "characters," each with different benefits in the game. Once the player selects a character, the level begins and they must traverse infinitely generated geometry, replenishing their light source, until they inevitably mess up. At that point, the game displays their highest score, and the player may play again. Pressing escape quits the game at any time.

Gameplay

The lighting of the scene tells a lot about the environment. The darker, the more fearful and desperate one's plight is. We have our user carry a radiating light source that gradually diminishes over time

The running of the game was handled through various C# scripts. For our camera and light source following the protagonist, we used linear interpolation to dictate its path following our character to prevent stuttery animation and smooth transitions. We'd have the light degrade in intensity over time and require the player to collect token of some kind to continue having light. When the light was gone, our player is left helpless and is more likely to die as a result of the game.

I think the coolest pieces of our project is that scripting we used to control the various aspects of our game. We have three scenes that are smoothly moved through by our scene manager and various toggles to change our character on the home screen. We also use these scripts to interact with objects in the world that lead to either increases in light or death for our character. It requires a lot of behind the scenes programming in order to actually build a game beyond just the graphics. Various triggers were set in order to put into motion various effects, and using Unity's GUI, we could dynamically fill in the instance variables of various classes as other objects in the scene, which allowed for many of our most interesting interactions.

We also used scripts to automatically generate new terrain and obstacles. By measuring the Eucledian distance between our coordinates and dynamically created terrain, we are able to generate terrain when necessary out of a pool of pre-built components that vary the terrain our character must traverse.

We also are able to keep a high-score through storing it in "PlayerPrefs" and create a score-board by calling update on a text-block throughout the run. This gives our player a higher score the longer they survive, providing a fun objective for the player as they can continue to beat their high-score.

The graphics components aren't necessarily as sophisticated since we have abstracted a lot of it away through using the Unity Engine; however, our project is focusing on utilizing the tools with other programming components in order to create a new experience.

What Didn't Work

Though ambitious, we attempted to create our own 3D models in Blender, however, that was at the last of our list of to-do items and unfortunately, with our lack of late days and strict deadline, we were unable to finish completing our terrain textures and character builds, instead of importing from a source library found on the Unity Store.

Originally, our camera and light would just snap to the location defined by our offset and character; however, we realized that led to really jerky and stuttery movements. We realized that the motion needed to be smoothened, so we of course turned to interpolation, utilizing the built-in interpolation "Vector3.Lerp" function to create a smooth path for our items to follow.

There was also a lot of various debugging issues regarding our scripts. For example, "triggers" and "colliders", which each have different methods to get the individual effects, both have actions when they are touched by other GameObjects. It was then necessary to distinguish between the two and know when to use each of them.

Unfortunately, there were a couple issues with continual forward velocity, so our game isn't a forced, forward runner, but by making light scarce, we force the user to continue walking forward, effectively still capturing the spirit of an endless runner game. This also gives the game a bit more flavor than your standard, run out of the mill runner game.

When creating our character control, we realized that if we used arrow keys to accelerate our object, the character would be sliding rather than having more snappy controls. So, when we call our Update() for our character, we kill all velocity and immediately accelerate it. With jumping and gravity, we also have to create a "grounded" boolean that allows us to jump and not fall through the floor. There are many things that need to be considered, and often it can only be found when testing.

What We Learned

Foremost, we learned the fundamentals of using Unity. This included the more GUI-created assets that we then had to write scripts to interact with. Scripting and understanding how each object interacted in space with each other was super important as we had to consider camera positioning, character placement, asset creation, etc. to build and display the scene that we wanted.

We learned that graphics are truly some of the most complicated components of game design. Having a lot of the creation, drawing, and geometry abstracted away really helped us create just the components related to scripting that allowed our game to function properly.

Playing The Game

Unfortunately our Mac build is a little buggy because we mainly developed on a Windows machine and just created a build on Mac using the same project files. The gameplay still works, but sometimes the death screen doesn't occur when one falls of the map.

What We Used

One of the most useful resources we found was this youtube channel:
https://www.youtube.com/channel/UCYbK_tjZ2OrIZFBvU6CCMiA
He had many useful videos explaining to us how to use unity, make the character models move, and overall teaching us how to write scripts that we could use to influence the world and the scoreboard. The "How To Make A Game In Unity" series was particularly useful to us.
This channel as well, https://www.youtube.com/channel/UCFK6NCbuCIVzA6Yj1G_ZqCg, was invaluable in teaching us how to create the infinitely generated level parts.

We hope you enjoy the game!