Game Report

Character Motions:

PixelNinja has two different moving characters, PixelNinja (player) and the monsters. While the ninja is controlled by the player, the monsters have a basic A.I.

Player (Ninja):

Ninja is controlled by the player by the conventional 'A, S, D' keys and instead of using 'W' since it's a 2D game the ninja jumps with the 'Space' key.

Ninja has a special '*NinJump'* jump that defies gravity. With enough power the ninja can jump all the way to the top of the map. The ninja also has speed and acceleration values to make the movement look smoother. The ninja can air strafe (change direction whilst in the air).

Monster:

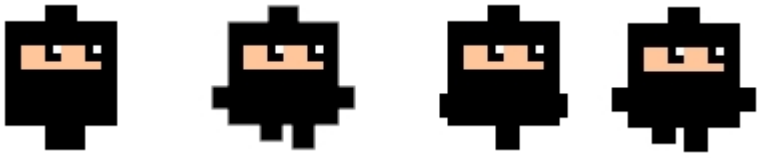
The monsters have a basic A.I. that enables them to follow the player in the map. This is done by moving the monster slowly towards the coordinates of

the ninja. The monsters rotate and translate accordingly. Monsters can only attack in melee form which means they have to get close. When they get close enough (in the range) they stop and attack. They also have an attack cool down to make the game fairer.

Actions:

Ninja can move, jump, and shoot shurikens in any direction. Each 'magazine' has 10 shurikens and player has to reload using the 'R' key. Ninja doesn't have infinite shurikens so be careful! Ninja cannot reload if there are still shurikens left in the magazine which means you cannot just spam 'R' key. Each action has its own texture animation. Walking has four states, and the others like firing, jumping have one.

Monsters have 3 actions, floating around the map, attacking the player and waiting for the cool down time to finish to be able to attack again.



Controls:

You can move your character with the 'A, S, D' keys on your keyboard to move the character, your mouse to aim and the mouse click to throw shurikens. 'R' key is for reloading.

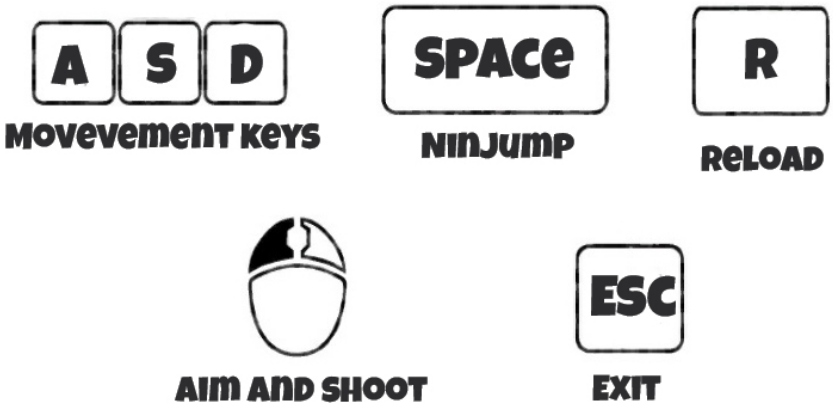
A -> Move Left

D-> Move Right

S -> Move Down (fall faster)

R-> Reload

Left Mouse Button-> Throw Shurikens



Collision Detection:

Most of the collision detection in the game is done by using boundary boxes. For the player movement, the collision detection is done in a way that player only has to check the collision on the direction it is going. For example if the player is going left the code checks the left side of the player and the right side of all the boundary boxes in the map. This is achieved by passing a char value to the method when a key is pressed (L(eft), D(own), R(ight), U(p)) If there is a collision the player isn't allowed to move in that direction and is pushed in the opposite direction.

The only time circle collision is used in the game is between the player and the monsters. Circles that have the same center point as the player and monster are created and checked against each other.

All the static game objects like platforms in the game are created by putting the x and y coordinates and the width, height of the collision box.

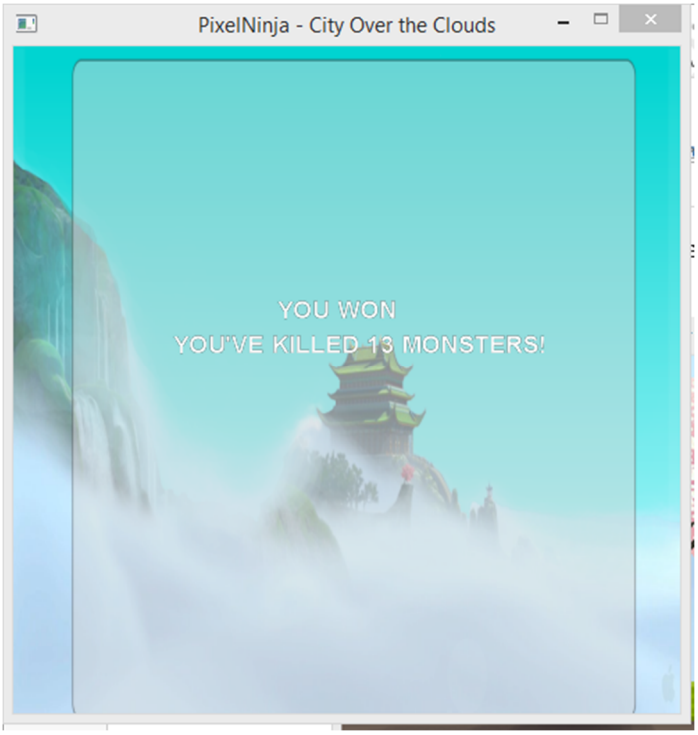
Coding Style:

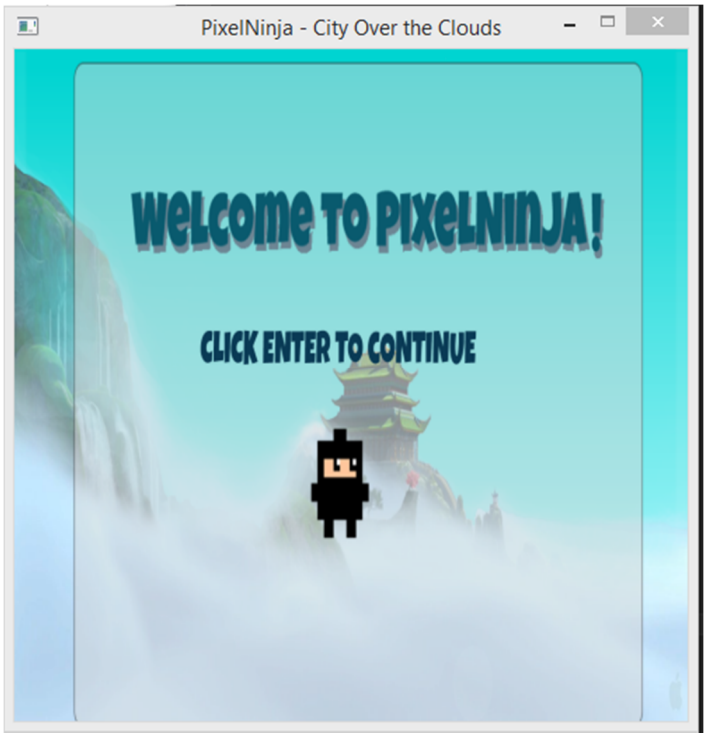
I have some experience in Unity3D engine making 2D games. That's why I wanted to write my code well-structured like the gameObject layout that Unity has. I created a GameObject class with the essential fields such as X, Y coordinates width and height, radius (for circle collision detection) and texture.

With a method to draw as well. Then I realised if I put all the methods and fields inside this class it's going to get complicated really fast. So created a second class called SGameObject (s for special) and put some accessor getter and setter methods inside this class. I've also included methods such as turn, move and update. Then I used these classes to create my Player, monster, CollisionBox and Shuriken classes.

Aim of the Game:

Aim of the game is to reach the finish at the platform on top of the map. When you start the game you will realize that you cannot jump anywhere close to that. You will need to kill the monsters to increase your jumping distance. This works by getting the player objects kill count and multiplying it by the jumpTime variable. You can also get on top of the house and jump from there if you don't want to fight monsters any longer than you have to. Oh and here's a little known fact – ninjas cannot swim-.

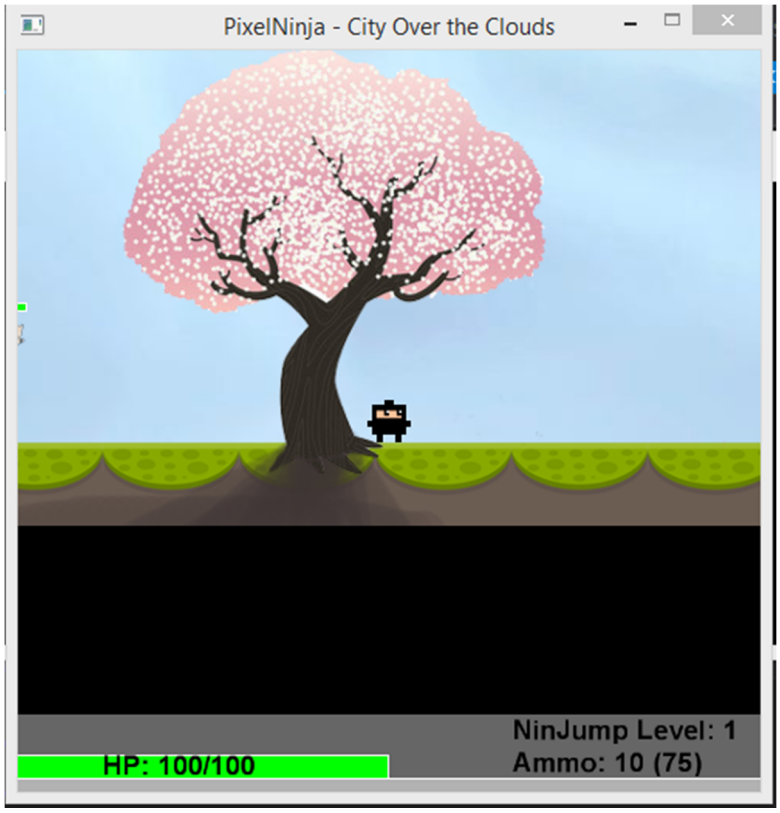




User Interface:

The start screen consists of a welcome message and instructions to continue to the game. When the key 'Enter' is pressed the bool flag start screen becomes false and the game starts.

The end screen consists of how you completed the game (won or lost) with the amount of monsters you've killed. The UI has your HP bar which goes yellow then red when you get hit. Your NinJump level and your clip/magazine is shown on the right side of the UI.



[i]

[i] Beren Kusmenoglu 100047218