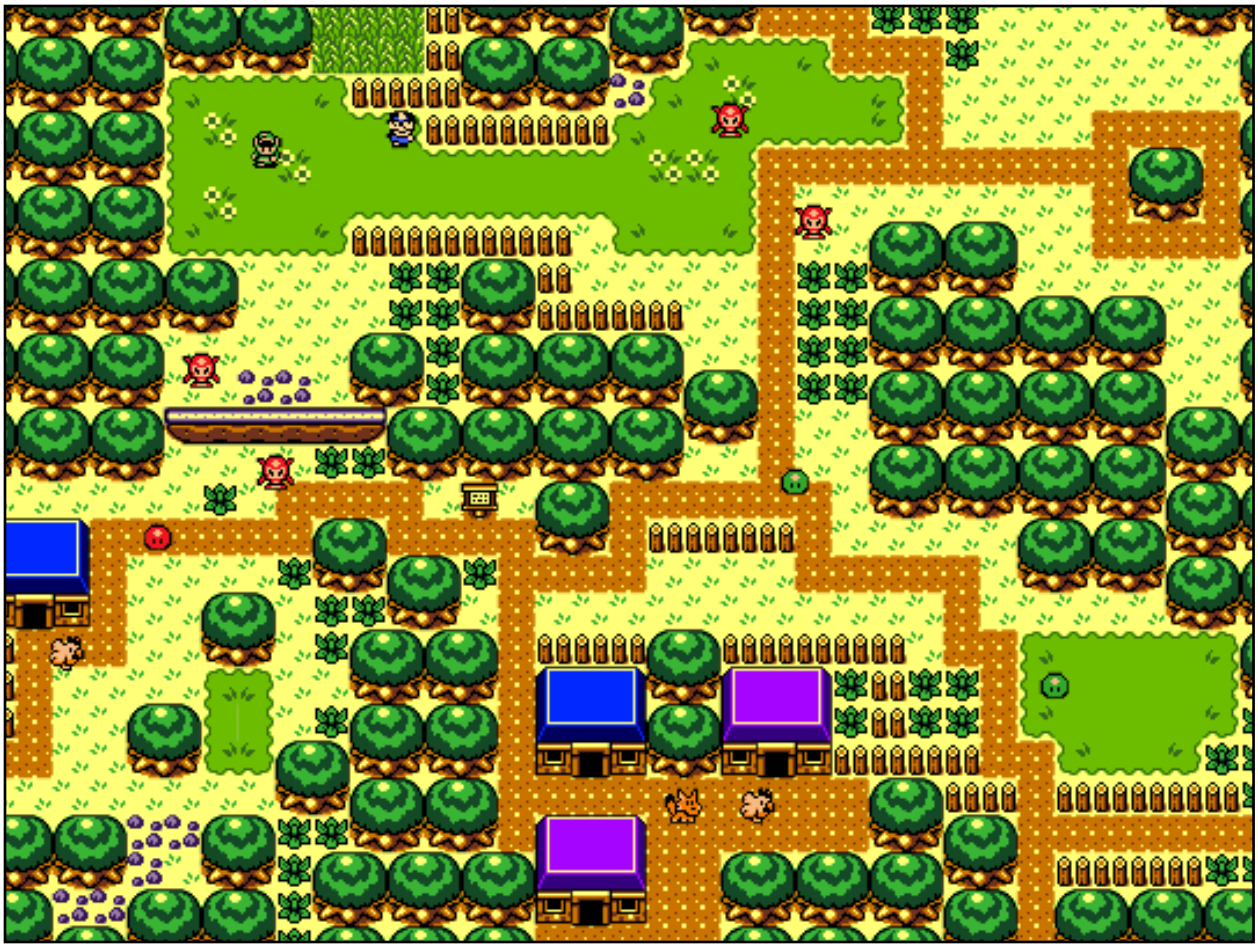
Olivier Goulet
ICS3U
June 13, 2016

ZELDA RIP-OFF: REPORT



PURPOSE

The Zelda Rip-Off's goal was to recreate the video game The Legends of Zelda: Links Awakening from 1993. But my game isn't very much like that… In my game, you can explore the small world that I've created with the sprites of the original game. You will face enemies and collect rupees, a currency from the original Zelda games.

Rules

The rules to the game is simple. To not die. When you die in the game, you will be forced to restart the game. You cannot restart at a checkpoint. A few restrictions in the game are what you'd expect: no walking through walls, trees, homes, etc because of the collision code.

Language

The language used for the game was ActionScript 3, Adobe's programming language which is used in all Flash applications such as flash games and flash players and is used as a standard for the web. It's known for it's flexibility for video processing across many different platforms such as web browsers, mobile applications and many more.

Where I got my code

The game's code was entirely written by me. The only times that I would read other peoples code would be when I'd get a bug that would need fixing or if I wanted to know the existence of a certain method. To find this information, I would read some entries from StackOverFlow and read Adobe's AS3 reference page.

INSTRUCTIONS

My game is located in my drive:
[PATH HERE]

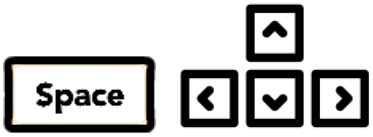
The instructions to my game are simple:
Make sure your volume is not muted so you can hear the sound effects in the game.

To use your sword, press Q. You use your sword to slay animals and enemies in the game.

To use yo ur shield, press E. You use your shield to defend yourself from damage. With the shield drawn, you can still be damaged if the enemy hits your back.

Health is displayed at the bottom of your screen on the black bar. If your health reaches 0, you'll explode and that will cause the game to end.

Control the character with the arrow keys.
Communicate to NPCs and signs by pressing the space bar.



A rupee(image to the left) is the currency used in the Zelda franchise. In my game, it doesn't have a particular use. You cannot buy anything with them. But you can use them to keep track of the score. A blue rupee is worth 1 rupee and a red rupee is worth 5.

LEARNING

I've learned that ActionScript 3 is one of the standard web programming languages, behind many applications that I use which require Flash plugins such as YouTube. It's also behind many Flash games. It's a very simple language, and I prefer it over Python. Programming with ActionScript is a fun experience. Especially when you have a program like Flash to build games on. Infinitely times more convenient than Python and it's Pygame library.

I've also learned that AS3 uses similar syntax as other big programming languages like C, Java and C++. The if statements are surrounded with brackets and the methods are incapsulated within curly brackets.

I've also familiarized myself with AS3's object oriented aspects. I've learned about different types of classes in Flash for AS3 such as MovieClip for game objects like a character or a tree, and the SoundChannel class which allows the programmer to control a sound. Later I've been using my own classes, teaching myself how to use the inheritance features. In my game, I created a class named Mob.as, extending MovieClip. This class is the basic blueprint to any AI in my game. It contains the movement code and all the properties of a creature such as health and specimen type. I then created other classes extending Mob.as such as Chicken.as and Dog.as. It's a very convenient and efficient way to complete a game without the repetition of writing collision code and movement code for every single creature. All I ended up changing in Chicken.as and Dog.as that wouldn't be the same in Mob.as would be the health stat of the creature, specimen type, damage stat, the sprites and animations.

Project management and time management

For project management, I've pretty much learned my lesson with what happened during my last culminating project in grade 10 with pygame, when I submitted a broken game with too much disorganized code. I put a lot more time into this project, working on it every night at home for a minimum of an hour. The only thing that I regret would be changing my mind of which game to make at the 2nd last week before the due date. I worked on a text RPG for the first 1 and a half weeks, and a 2D side scroller Zelda game but quickly got bored of the two and decided to make this game. Although I'm glad I did change the game because this one is a lot better than the other two. This problem wasn't a huge deal though, because I can put many hours into this game because I've got access of Flash at home.

ANALYSIS

I did not accomplish my some goals such as an immersive world to play in. You can't use the rupees and you cannot enter dungeons nor can you enter homes. But I did accomplish to recreate the original Zelda game's basic mechanics such as the collisions, AI, item drops, animations. In my opinion, I've also did a good job in making it look like the original game from 1993.

I didn't modify the goal at all. I would still want to add all those features in but it may of been better to work on this with a partner because all the hours I've spent at home weren't enough.

Bugs

The game is still infested with bugs that I'm still having trouble fixing. The AI's are somehow, rarely, walking beyond collision objects but then get stuck in the middle of the object. The artificial intelligence isn't even intelligent.

I would also fix the frame rate issues. The memory is being all used up from monsters being constantly added to arrays every KEY_DOWN event. I've traced the length of the array in the output. When the length is greater than 1000, it will lag. To fix this I would add a method to the mob class named addToArray() in the constructor. It will push the instance into the array when it spawns, instead of every KEY_DOWN. I could do this right now but I don't want to risk breaking my npc because they are the main feature of my game.

If I had more time

I also couldn't find the problem behind why my character couldn't spawn in a house. The house was the 2nd frame of the world MovieClip (the map). After removing the whole thing, I think the problem was simply just a math problem. The world moves and not the player and so I think I would need to figure out the house position(fixed at the world position which changed throughout the gameplay). I could've made the house a separate movie clip and it would have a fixed position of it's own.

If I had more time, I would definitely add dungeons, a story, shops and refine the programming. If I were to expand the project I think I would need a major redesign. For many things, I wouldn't of ever knew in advance that I would reuse code for other things. For example: I made AI's from scratch all the time with the octoroks(red octopus monsters) and with chickens. When I realized I would need many more AI's and it would take a long time just to make one more, I started making the mob.as class which would have all the other subclasses inherit it's features.

Lastly, I would replace the messy code with simplified code, not needing any major editing and still have a better quality game. The messy code is there because of unexpected errors in my plan causing me to do loops and if statements or because of just simply not planning.