**Work in Progress Report**

Major developments/breakthroughs(reference specific code please):

Since this our first time working with LibGDX, we had a major development in the form of getting our project architecture up and running.

We decided that instead of jumping straight into our release schedule, we would try and figure out some of the basics in running a project and trying to get basic code to work, then slowly implement those scratches into our main game.

We all finally got our Android Studio to work successfully, after facing some issues with emulators and project architecture in the past.

In our Animation scratch, one of our major breakthroughs was successfully getting an animation to work. We created a code that simply cycled through an animation for a sprite and then built off of that, creating a program that implemented user input along with those animations. This was our first major development of our game, and we learned a lot about the LibGDX TexturePacker, implementing that file into Android Studio, and manipulating it in a way that allowed us to animate the sprite when the user pressed an arrow key.

Another major development came from the Tiled map editor. We learned how to use Tiled to create a map, and render that map in Android. Along with this we also learned about Orthographic cameras, and how we can use them to follow a player around the map, allowing for the screen to scroll.

Here is the code that rendered the Tiled Map:

tmGameMap= new TmxMapLoader().load("IntoTheWoodsRPGMap.tmx");

OrhtoTmrRenderer = new OrthogonalTiledMapRenderer(tmGameMap);

A fairly big breakthrough we had was with the button scratch. We got the button to appear on screen and clicking it made a line of text appear in the debug window. We got the button to appear on the main screen in the end, but it currently doesn’t switch screens (check below).

Major Challenges/setbacks( reference specific code please):

One of our major setbacks was figuring out how to animate sprites. Initially, we based our Animation scratch off of some example code from a group last semester. We tried to take the architecture of that program and implement into a scratch code, where we simply wanted to test the concept of sprite animation. We discovered that the concept of Scene2D, something we were not quite fully familiar with and the fact that movement was handled in an odd way that ended up confusing us further.

Michelle encountered some problems working with switching between screens with the click of a button. We ran into some problems when it came to implementing the texture pack of the button, first because of issues with Texture Packer and then through the code running into many issues with it not accepting the texture it was given. Then, We were working on implementing multiple screens, a menu and the game screen, with a button on the main screen. Currently on the main screen the button will appear, but right now does nothing when clicked - right now the button currently isn’t calling the action listener because we were having issues calling the button variable - this code is supposed to make the button switch screens, yet it does not:

**if** (**button**.isPressed()) {

**gamIntoTheWoods**.**currentState** = GamIntoTheWoods.GameState.***GAME***;

**gamIntoTheWoods**.updateState();

}

Any modifications to your specifications/release schedule:

We made one modification to our release schedule, which is changing the plan to have to four arrow keys move the player around to using a d-pad to move the player around to later on in the release schedule (within the next 1-2 WIPs).

**Description of your scratch/test program:**

Describe the generic concept you needed to test out:

We created multiple scratch codes to test out generic concepts for our game:

1. We wanted to animate a sprite
2. Button scratch: We needed to figure how to draw buttons to the screen and implement a button listener. This scratch come out of a rising necessity. We were having difficulty combining screen switching with button implementation, so we decided to isolate the two concepts.
3. Tiled Map Scratch: Our goal with this scratch code was to get comfortable with the Tiled map editor, and successfully implement it into Android Studio so that an animated sprite could move around on the map. Along the way, we realized that a natural next step to our Tiled map was to integrate an Orthographic camera

Source any web site/book that helped you with that concept:

1. Sprite Animation: Hello World Tutorials- https://www.youtube.com/user/TheLazyTryhard
2. Button: Hello World Tutorials- https://www.youtube.com/user/TheLazyTryhard
3. Tiled Map + Orthographic Camera: The following sources were used to help us create this scratch: http://gamedevelopment.tutsplus.com/tutorials/introduction-to-tiled-map-editor-a-great-platform-agnostic-tool-for-making-level-maps--gamedev-2838

<https://www.youtube.com/watch?v=ikajOOa4EPI>

https://www.youtube.com/watch?v=MT5YwZsQnF8

Describe the code and the lesson that you learned from it:

1. Sprite Animation. We learned how to set up and work with the texture packer as well as updating the images that are drawn to the screen. In addition we learned how to implement user input so that when the user presses an arrow key, the sprite moves in that direction and cycles through an animation. While we won’t be using the arrow keys in our actual game, learning how to move a Sprite and use animations is definitely valuable information.
2. Button: We learned how draw buttons and apply an action listener to it.
3. Tiled Map + Orthographic Camera: In this scratch code, the user controls a sprite around the game screen using the four directional arrow keys. The screen scrolls with the player, and the player can move around on top of a map drawn in Tiled. From this scratch code, we learned how to use the Tiled map editor to create a 2-Dimensional map, how to render and add that map to a game, and how to allow the screen to scroll using an Orthographic Camera.

Describe any challenges that you enjoyed in integrating this scratch code into your major project:

1. Sprite: A major challenge was splitting up the sprite correctly so the animations play the sprite in the proper order. For example, at one point it played on in each direction so the character spun while moving
2. Button: The hardest thing was setting up the fonts because it is required that you take a font and turn it into an .fnt;
3. Tiled Map + Orthographic Camera: We have yet to integrate this scratch code into the main project, but one of the challenges will will encounter in the near future is how to handle collision detection with the Tiled map.