**Work in Progress Report**

**By: Deep Raithatha, Jash Pandya, Jose Rivas**

**GitHub URL:** [**https://github.com/JoseRivas/FlightlessFury**](https://github.com/JoseRivas/FlightlessFury)

**Major developments/breakthroughs(reference specific code please):**

**Note:** The focus of this WIP and our past two weeks was to make our project fit the standards of this industry and Mr. Gordin to add classes for ease of debugging and great architecture.

**Main game**

Branch: Master

Aid: <https://github.com/libgdx/libgdx/wiki/Accelerometer>

**Using accelerometer to control the rotation of the penguin**

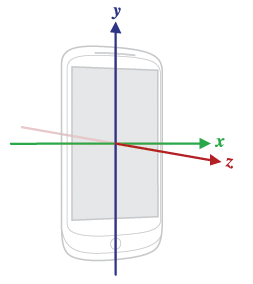
The program uses libgdx to input the accelerometer’s value

public static float accelY(){

return Gdx.input.getAccelerometerY();

}

penguin.body.setAngularVelocity(-accelerometer.accelY()/3);//sets the body’s angle to the accelerometer y-value



Gdx.input.getAccelerometerY();

This prefabricated libgdx method returns a value between -10 and 10. This value is the acceleration due to gravity. This means that at 10 and -10 the phone is turned 90 degrees from the original position. We tweaked the value to a reasonable size for our penguin.

**Split the program into multiple files: penguin file, buttons file etc.**

In the penguin file we have methods that set box2d variables

This updates the penguin’s position for every box2d world step

public void UpdatePos() {

sprite.setPosition((body.getPosition().x \* 100f) - sprite.

getWidth() / 2,

(body.getPosition().y \* 100f) - sprite.getHeight() / 2)

;

sprite.setRotation((float) Math.toDegrees(body.getAngle()));

}

This resets the penguin’s position and velocity to the initial values

public void ResetPos(){

body.setLinearVelocity(0, 0);

body.setAngularVelocity(0);

body.setTransform((sprite.getWidth() + sprite.getWidth() / 2) / 100f,

(Gdx.graphics.getHeight() / 7/3 + sprite.getHeight() / 2) / 100f, 0);

}

This draws the penguin’s sprite using the main file’s batch

public void draw(SpriteBatch batch) {

batch.draw(sprite, sprite.getX(), sprite.getY(), sprite.getOriginX(),

sprite.getOriginY(),

sprite.getWidth(), sprite.getHeight(), sprite.getScaleX(), sprite.

getScaleY(), sprite.getRotation());

}

**Major Challenges/setbacks( reference specific code please):**

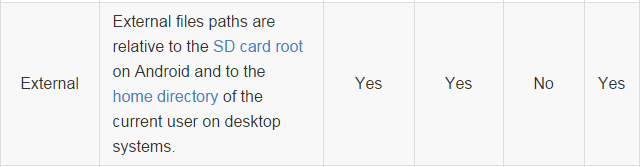
File I/O: (Scratch Test Program)

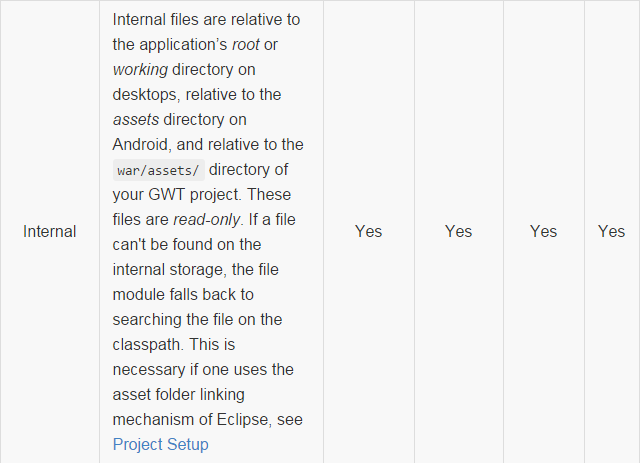
Branch: HandlingInput

Aid: <https://github.com/libgdx/libgdx/wiki/File-handling>

The I/O of libgdx contain OutputStream and InputStream type addons which complicate the concept.

Two types of file import and output:





I have looked at Aaron and Grimes code on how they completed used File I/O in their games. I will look forward to fixing and integrating this aspect.

**Any modifications to your specifications/release schedule:**

We plan to make stages using tiled maps to add variety to the game.

Our next step is to incorporate titled maps and object to object collision to incorporate the aspect of point collection.