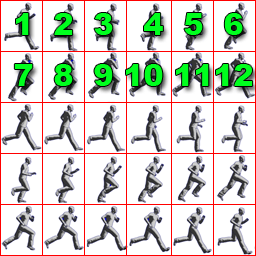
Scratch Program Descriptions

**Animations**

GitHub URL: <https://github.com/JoseRivas/LibgdxAnimation>

Tutorial Link: <https://github.com/libgdx/libgdx/wiki/2D-Animation>

**private static final int FRAME\_COLS = 6;**

**private static final int FRAME\_ROWS = 5;**

**for** **(int** i **=** 0**;** i **<** FRAME\_ROWS**;** i**++)** **{ This for loop, cycles through the columns and rows of image**   
 **for** **(int** j **=** 0**;** j **<** FRAME\_COLS**;** j**++)** **{**  
 walkFrames**[**index**++]** **=** tmp**[**i**][**j**];  
 }**  
 **}**  
 walkAnimation **=** **new** **Animation(**0.025f**,** walkFrames**);** *// #11*  
 spriteBatch **=** **new** **SpriteBatch();** *// #12*  
 stateTime **=** 0f**;** *// #13*

#11:

* The first parameter tells the animation, how much time is allocated for each frame. (in seconds)
* The more frames an animation has, the smaller the time it will be and the smoother it will look. (BUT…. It consumes more memory.)

**Buttons** Creation is an aspect of our game to control the angle of flight of the penguin

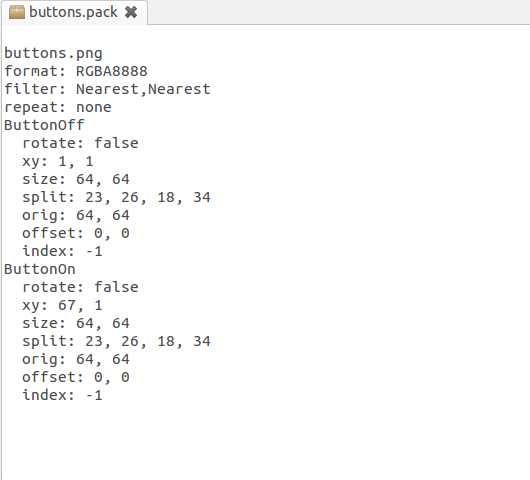
Since we are using LibGDX, the way to create buttons is different from XML. We learned how to create and use buttons from the following site:

Paste URL: <http://stackoverflow.com/questions/21488311/libgdx-how-to-create-a-button>

GitHub URL: <https://github.com/JoseRivas/LibgdxButtons>

buttonAtlas = new TextureAtlas(Gdx.files.internal("buttons/buttons.pack"));

This lines references the pack of images such as button pressed and buttons released. An additional program needed to be installed called “Texture Packer”. It creates a pack file (.txt)



**Orthographic Camera**

**GitHub Branch: OrthoCam**

Source Link: http://www.gamefromscratch.com/post/2013/11/06/LibGDX-Tutorial-7-Camera-basics.aspx

camera = new OrthographicCamera(1280, 720);

Gdx.input.setInputProcessor(new GestureDetector(this));

……

public boolean pan(float x, float y, float deltaX, float deltaY) {

// TODO Auto-generated method stub

camera.translate(deltaX,0); Gesture method for panning camera.update();

return false;

}

What is Orthographic Camera vs Perspective?

Basically an orthographic camera renders what is in the scene exactly the size it is.

Once the image is panned, simultaneously the boolean pan method with do camera.update(); to simulate panning.

**Collision Detection**

GitHub Branch: Collision

Source Link: <http://programmersweb.blogspot.ca/2012/07/simple-libgdx-box2d-bouncing-ball.html>

**//Ground body**

BodyDef groundBodyDef =new BodyDef();

groundBodyDef.position.set(new Vector2(0, 10));

Body groundBody = world.createBody(groundBodyDef);

PolygonShape groundBox = new PolygonShape(); **// imaginary floor, where the ball is bouncing**

groundBox.setAsBox((camera.viewportWidth) \* 2, 10.0f);

groundBody.createFixture(groundBox, 0.0f);

**Using Box2d we implemented created a world based on vectors.**

World world = new World(new Vector2(0, -100), true);\

Variable : static final int velocity = 10; // set the velocity of the ball (Approx -9.8 m/s {down})

// Dynamic Body (Ball)

BodyDef bodyDef = new BodyDef();

bodyDef.type = BodyType.DynamicBody;

bodyDef.position.set(camera.viewportWidth / 2, camera.viewportHeight / 2);

Body body = world.createBody(bodyDef);

CircleShape dynamicCircle = new CircleShape();

dynamicCircle.setRadius(5f); // radius of the circle/shape

FixtureDef fixtureDef = new FixtureDef();

fixtureDef.shape = dynamicCircle;

fixtureDef.density = 10.0f; ←-----real world attributes such as density

fixtureDef.friction = 0.0f;←-----real world attributes such as friction

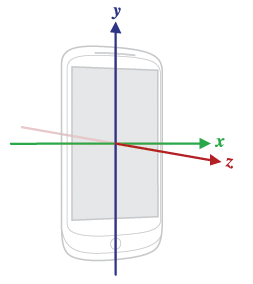
fixtureDef.restitution = 1;

**Accelerometer**

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.