Catani WIP 2

# Breakthroughs:

Obstacles and hit detection: In the game I wanted a variety of different types of movement for obstacles. At first I attempted to have one main obstacle class with different movement functions being different but I realized that it was easier to have one parent obstacle class subclasses for each different type of obstacle movement. I further detailed this in the Scratch Description below.

Animations: After many failed attempts. I have managed to create a function that acts as a “cookie cutter” for different animations. I can still work on making the function cleaner

# Challenges:

Animations: Tim was working on creating an animation class. He was attempting to create a function that would be the “cookie cutter” for any animation we wanted to implement. I ended up creating something that worked, however, it was REALLY inefficient. The problem was that I could not grab the Animation array that held the different “frames” from the function. After restarting twice. I managed to find a work around. It’s not the cleanest work around. However, it cleans up the amount of code I need to write in order to create a new animation.

***\*\*And car crash\*\****

# Updated Release Schedule:

|  |  |
| --- | --- |
| **Release Name** | **New incremental features of this release** |
| **1.0** | **Scrolling background and ability to flip gravity** |
| **2.0** | **Scrolling obstacles with hit detection** |
| **3.0** | **Add larger variety of obstacles such as pitfalls and spawn obstacles with increasing difficulty for the player as score progresses** |
| **3.5** | **Add coin with random spawn locations** |
| **4.0** | **Develop Main Menu and Gameover screen that transition into one another as well as Game screen** |
| **4.5** | **Add dynamic score in meters and speed up scrolling background as game progresses** |
| **5.0** | **Change terrain, remove flat ground. Replace with different sections of level (Example: Rolling hills, stairs going up/down etc.** |
| **6.0** | **Add shop with variety of skins for main character** |
| **7.0** | **Add sounds (flipping sound, death sound etc.)** |
| **8.0** | **Attempt firebase multiplayer on android devices** |

# Scratch Obstacles:

The objective of this scratch was to create distinct types of obstacles. I started by making a SprObstacle class which is a subclass of the libgdx sprite class. The obstacle's render function updates its position according to a velocity vector and draws it to the ScrGame sprite batch.

public void render(SpriteBatch batch) {

setPosition(getX() + vVel.x, getY() + vVel.y);

draw(batch);

}

It also has an isHIt method using the libgdx built in hit detection

public boolean isHit(Rectangle player) {

if (player.overlaps(getBoundingRectangle())) {

return true;

}

return false;

}

After this I decided to create an obstacle that moved in a sinusoidal motion. This obstacle is a subclass of the SprObstacle class. I named the class ObsFlying. In this class the only new method I added was an update function, this uses the libgdx sin function and my knowledge from advanced functions to create smooth sinusoidal movement across the screen

public void update(){

setY((MathUtils.sin(getX()/1920\*MathUtils.PI2)\*500)+540);

}

Now with the framework of making subclasses from the SprObstacle class I decided to make another type of obstacle one that rotates

ObsSpinning. The obstacle is a sublcass of SprObstacle which is a subclass of Sprite meaning that this class can use any of the sprite methods. SO to make the object spin all I did was use the rotate function in degrees.

public void update(){

rotate(1);

}

After this I wanted the method of rendering obstacles to be as efficient as possible so i used a libgdx array and because all the obstacles originate from SprObstacle they are all the same type in an array. Additionally each obstacle with unique movement uses the same a function with the same name, update which allows for easier implementation in the array. The obstacles are rendered with a for each loop.

for (SprObstacle obstacle : obstacles) {

obstacle.render(batch);

if (obstacle.getX() < 0 - obstacle.getWidth()) {

obstacle.setX(viewport.getWorldWidth());

obstacle.setY(MathUtils.random(FLOOR, CEILING - obstacle.getHeight()));

}

if(obstacle.isHit(chrMain.getBoundingRectangle())){

System.out.println("HIt " + obstacle);

}

}

In this for loop it renders the obstacle as well as sets its coordinates so it continually loops on the screen and checks for hit detection.

# 

# Scratch Animations:

This class was to have a function that takes a spritesheet, splits it up into however many columns and rows there are, and have a specific animation array that takes all of the “Frames” created.

void readyFadeAnimation(String SprSheetName, int nColumns, int nRows, int index) {

Texture SpriteSheet = assets.manager.get(SprSheetName, Texture.class);

TextureRegion[][] tmp = TextureRegion.split(SpriteSheet,

SpriteSheet.getWidth() / nColumns, SpriteSheet.getHeight() / nRows);

txtRegAnimationFrames = new TextureRegion[nColumns \* nRows];

for (int i = 0; i < nRows; i++) {

for (int j = 0; j < nColumns; j++) {

txtRegAnimationFrames[index++] = tmp[i][j];

}

}

AnmCreateAnimation = new Animation<TextureRegion>(0.10f, txtRegAnimationFrames);

}

**This part was the easy part. The problem was that I needed a different animation array for every “txtRegAnimationFrames”. So I tried throwing in my own creates animation array**

void readyFadeAnimation(String SprSheetName, int nColumns, int nRows, int index, Animation<TextureRegion> AnmCreateAnimation) {

... code above

AnmCreateAnimation = new Animation<TextureRegion>(0.10f, txtRegAnimationFrames); }

**The problem was that if I already created Animation<TextureRegion> AnmCreateAnimation, then I can’t create a new instance of it within the function. I did some research to see if there was anyway to set the texture region of Animation<TextureRegion> as txtRegAnimationFrames. But I couldn’t find anything. So I had to do a work around, It’s not the greatest work around, but it works.**

void readyFadeAnimation(String SprSheetName, int nColumns, int nRows, int index, Animation<TextureRegion> AnmCreateAnimation) {

... code above

if (SprSheetName.contains("AniBtnPlaySprite.png")) {  
 nImgNum = 1; }

if (SprSheetName.contains("shopbutton.png")) {  
 nImgNum = 2; }

}

**The nImgNum integer Controls what Array holds what Spirtesheet Like so:**

public Animator(Assets \_assets) {

... code

readyFadeAnimation("AniBtnPlaySprite.png", 2, 6, 0);

if (nImgNum == 1) {

AniBtnFade = new Animation(0.10f, txtRegAnimationFrames);

}

readyFadeAnimation("shopbutton.png", 2, 6, 0);

if (nImgNum == 2) {

AniTest = new Animation(0.10f, txtRegAnimationFrames);

}

}

**Now that each Animation array holds its own frames, I can load them into different functions that runs different animations. These are the functions used is the different screen classes as example: animation.draw**

void drawAni(SpriteBatch batch, int x, int y) { // nImgNum = 1

ElapsedTime += Gdx.graphics.getDeltaTime();

batch.draw((TextureRegion) AniBtnFade.getKeyFrame(ElapsedTime, true), x, y);

}

void drawtest(SpriteBatch batch, int x, int y) { // nImgNum = 2

ElapsedTime += Gdx.graphics.getDeltaTime();

batch.draw((TextureRegion) AniTest.getKeyFrame(ElapsedTime, true), x, y);

}