Body body;

Body[] wallBodies = new Body[30];

int wallBodiesCount = 0;

int turnsRemaining;

public void addWall(float xPos, float yPos, float width, float height){

BodyDef wallBodyDef = new BodyDef();

wallBodyDef.position.set(new Vector2(xPos, yPos));

Body wallBody = world.createBody(wallBodyDef);

PolygonShape wallBox = new PolygonShape();

wallBox.setAsBox(width, height);

wallBody.createFixture(wallBox, 1.0f);

wallBodies[wallBodiesCount] = wallBody;

wallBodiesCount++;

}

I contributed to the code used to transition between levels, not create the button

TextButton nextLevelButton = new TextButton("Next", skin, "default");

nextLevelButton.setColor(Color.ORANGE);

nextLevelButton.setWidth(100);

nextLevelButton.setHeight(100);

nextLevelButton.getLabel().setFontScale(2);

nextLevelButton.setPosition(Gdx.graphics.getWidth() /2-1200f, Gdx.graphics.getHeight()/2+250f);

nextLevelButton.addListener(new ClickListener(){

@Override

public void clicked(InputEvent event, float x, float y){

for (int i=0; i<wallBodies.length; i++) {

if (wallBodies[i] != null) {

world.destroyBody(wallBodies[i]);

}

}

world.destroyBody(body);

music.stop();

level ++;

if(level == 21){

level--;

game.setScreen(MyGame.endGameScreen);

}

create();

}

});

@Override

public void create() {

Camera stuff was made as a group

// Use a camera to map from box2d to screen co-ordinates

camera = new OrthographicCamera();

// Screen resolution

camera.viewportHeight = 320;

camera.viewportWidth = 480;

// Select which part of the world you want the camera to see

camera.position.set(camera.viewportWidth \* .5f, camera.viewportHeight \* .5f, 0f);

// Update the changed camera settings!

camera.update();

//side walls

addWall(0,0,(camera.viewportWidth), 1f);

addWall(0,camera.viewportHeight,(camera.viewportWidth), 1f);

addWall(0,0,0, (camera.viewportHeight));

addWall(camera.viewportWidth+1,0,1f, (camera.viewportHeight));

Random ranNum = new Random();

if (level==1) {

turnsRemaining=5;

addWall(270, 250, 40, 60);

addWall(300, 55, 70, 40);

wallBodiesCount=0;

reachedLevel =1;

}

else if(level==2){

turnsRemaining=12;

addWall(150, 120, 10, 110);

addWall(270, 200, 10, 110);

addWall(350, 120, 10, 110);

if(reachedLevel <2) {

reachedLevel = 2;

}

}

else if (level==3) {

turnsRemaining=7;

addWall(60, 110, 10, 30);

addWall(140, 100, 30, 50);

addWall(250, 160, 50, 80);

addWall(400, 250, 20, 30);

wallBodiesCount=0;

if(reachedLevel <3) {

reachedLevel = 3;

}

}

// Create the body for the ball

BodyDef bodyDef = new BodyDef();

// Make the ball a dynamic body so that it can be accelerated

// and can collide with the ground and bounce

bodyDef.type = BodyType.DynamicBody;

// Start the ball positioned near the top of the viewport

bodyDef.position.set(camera.viewportWidth / 5, camera.viewportHeight / 2f);

// add the ball to the world

body = world.createBody(bodyDef);

body.setFixedRotation(true);

//slows the player down

body.setLinearDamping(.8f);

// Give the ball a circular shape

CircleShape dynamicCircle = new CircleShape();

// Set the size of the ball shape

dynamicCircle.setRadius(5f);

// Create a new fixture

FixtureDef fixtureDef = new FixtureDef();

// Give it the circle shape

fixtureDef.shape = dynamicCircle;

// Give it a density so the ball will have mass

fixtureDef.density = 0.5f;

// Ignore friction

fixtureDef.friction = 0.0f;

// Enable bounce

fixtureDef.restitution = 1f;

// add the fixture to the ball body

body.createFixture(fixtureDef);

}

boolean moving = false;

@Override

public void render() {

int x = Gdx.input.getX();

int y = Gdx.graphics.getHeight() - Gdx.input.getY();

//This code sets the direction to the way the circle is facing

// and sets the speed it travels in that direction to moveSpeed

int moveSpeed = 50000;

double xDirectionD = (Math.cos(body.getAngle()));

float xDirection = (float) xDirectionD;

double yDirectionD = (Math.sin(body.getAngle()));

float yDirection = (float) yDirectionD;

Vector2 moveVelocity = new Vector2(xDirection \* moveSpeed, yDirection \* moveSpeed);

int rotationThreshold = 5;

//if the ball is below the rotation threshold set moving to false, isn't at 0 for gameplay

if(Math.abs(body.getLinearVelocity().x)<=rotationThreshold && Math.abs(body.getLinearVelocity().y)<=rotationThreshold) {

moving=false;

}

//If the ball has stopped moving rotate it

if (!moving){

body.setTransform(body.getPosition().x, body.getPosition().y, body.getAngle() + (0.0174533f\*4));

}

I wrote this to work with keyboard input, Harris added the if for touch input, Tung added the if to check it isn’t clicking the menu

if((Gdx.input.isButtonPressed(Input.Buttons.LEFT) && !moving || (Gdx.input.isKeyPressed(Input.Keys.RIGHT) && !moving)) && turnsRemaining>0) {

if(Gdx.input.getX() > Gdx.graphics.getWidth() /2-1100f) {

body.applyLinearImpulse(moveVelocity, body.getMassData().center, true);

moving = true;

turnsRemaining--;

}

}

//Press left to go to next level

if(Gdx.input.isKeyPressed(Input.Keys.LEFT)){

for (int i=0; i<wallBodies.length; i++){

if (wallBodies[i]!=null) {

world.destroyBody(wallBodies[i]);

}

}

world.destroyBody(body);

music.stop();

level++;

if(level == 21){

level--;

game.setScreen(MyGame.endGameScreen);

}

create();

}

//press up to restart or run out of lives mostly for testing

if(Gdx.input.isKeyPressed(Input.Keys.UP)) {//|| (turnsRemaining==0 && !moving)){

for (int i=0; i<wallBodies.length; i++){

if (wallBodies[i]!=null) {

world.destroyBody(wallBodies[i]);

}

}

world.destroyBody(body);

if(level == 21){

level--;

game.setScreen(MyGame.endGameScreen);

}

music.stop();

create();

}

//Check position of body and goal. If body is reach a goal, move to next level.

if((body.getPosition().x < 440 && body.getPosition().x >400) && (body.getPosition().y >14 && body.getPosition().y < 75)){

I wrote the parts in red brackets {

for (int i=0; i<wallBodies.length; i++){

if (wallBodies[i]!=null) {

world.destroyBody(wallBodies[i]);

}

}

world.destroyBody(body);

}

//Increase level by 1

music.stop();

{ level++; }

if(level == 21){

level--;

game.setScreen(MyGame.endGameScreen);

}

{ create(); }

}

I set up the turns remaining and level code but I didn’t create the labels in game for them

String turn;

turn ="Turns Remaining: " +turnsRemaining;

Skin skin = new Skin(Gdx.files.internal("uidata/uiskin.json"));

Label turnRemainingLabel = new Label(turn, skin);

turnRemainingLabel.setPosition(Gdx.graphics.getWidth() /2-1150f, Gdx.graphics.getHeight()/2 +580f);

turnRemainingLabel.setSize(200,200);

turnRemainingLabel.setFontScale(3);

turnRemainingLabel.setColor(Color.ORANGE);

String currentLevel;

currentLevel = "Current Level: "+ level;

Label levelLabel = new Label(currentLevel, skin);

levelLabel.setPosition(Gdx.graphics.getWidth() /2+800, Gdx.graphics.getHeight()/2 +580f);

levelLabel.setSize(200,200);

levelLabel.setFontScale(3);

levelLabel.setColor(Color.ORANGE);

for(int i=0; i < stageForLabel.getActors().size; i++){

stageForLabel.getActors().removeIndex(i);

}

stageForLabel.addActor(turnRemainingLabel);

stageForLabel.addActor(levelLabel);

Gdx.input.setInputProcessor(stage);

stageForLabel.draw();

stage.draw();

}