DroidScript Gameview

Tutorial - Flappy Dino Part 2

Lesson 13 – Physics - Gravity

To take our game to the next level, we can use physics to change the way the game works to make it smoother and more professional.

In the previous version of the game, the dino's y-position was gradually reduced to make it drift down the screen. If we use physics, we do not need to do it in this way and instead we can use Gravity. This will make a much more realistic and natural looking movement.

At this stage you may want to create a new version of the game as we will need to delete some of the old functions and change the way the game works.

The first step to this is adding Physics to the gameview in the OnLoad() function like this. We also add a 'gameOver' that will be used later on. The rest of the OnLoad() function remains the same as before.

```
//Handle game loading.
function OnLoad()
{
    //Add physics
    gfx.AddPhysics()

    //Create background (and stretch it to full screen).
    sky = gfx.CreateBackground( "Img/flappy_back.jpg", "stretch" )

    //Create graphical objects.
    dino = gfx.CreateSprite( "Img/dino_fly_783x136x3x1", "dinos" )
    croc = gfx.CreateSprite( "Img/crocs_149x686x1x2", "enemies" )
    crash = gfx.CreateSpriteSheet( "Img/dino_crash_920x138x4" )
    txt = gfx.CreateText( score.toString(), 0.1, "Img/Desyrel.xml" );

    gameover = gfx.CreateSprite( "Img/gameover.jpg", "gameover" )

    //Create sounds.
    jungle = gfx.CreateSound( "Snd/jungle.mp3" )
    crunch = gfx.CreateSound( "Snd/crunch.mp3" )
    end = gfx.CreateSound( "Snd/game_over.mp3" )
}
```

Next we will will need to edit the OnReady() and OnAnimate() functions to tell the program to use physics and set the characteristics. First change the OnReady() function to look like this:

```
//Called when game has loaded.
function OnReady()
{
    //Set background.
    gfx.AddBackground( sky )

    //Add dino image.
    gfx.AddSprite( dino, 0.1, 0.5, 0.15 )
    dino.SetPhysics( 1, "dynamic" )

    //Set dino physics shape to be a rectangle slightly smaller than
    //the source image boundary so that collisions look more realistic.
    dino.SetShape( "rect", 0.7, 0.7 )


    //Add enemy sprites.
    gfx.AddSprite( croc, 0.8, 0.55, 0.12 )
    croc.SetPhysics( 2, "fixed", 1 )

    //Start animations
    dino.Play( 0, 0.25 )
    croc.Play( 0, 0.05 )

    //Add score text
    gfx.AddText( txt, 0, 0 );

    //Play looping background sound.
    jungle.Play(true)

    //Start the game.
    gfx.Play()
}
```

Finally, rewrite the OnAnimate function to look like this:

```
//Update game objects.
//(called for every frame)
function OnAnimate( time, timeDiff )
{
	//Do nothing if game is over.
	if( gameOver ) return

	//Make background slide to the left a small amount.
	sky.Scroll( -0.002, 0 )

	//Slide croc to left.
	croc.x -= 0.004

	//If croc has gone off left side of screen.
	if( croc.x < -croc.width )
	{
		//Move the croc back to the right hand side.
		croc.x = 1

		//Scale croc with a random value.
		var height = 0.4 + Math.random()*0.4
		croc.SetSize( null, height )
		croc.y = 1 - croc.height
		score++
		txt.SetText( score.toString() )
	}

	//We need to update the physics state when we manually
	//move objects that use physics.
	croc.UpdatePhysics()
}
```

If everything is correct, at this point you can run the game and you will now see the dino fall down the screen much faster than before. We removed the “dino.y += 0.001” line as it is no longer needed because the game is now using gravity to pull the dino downwards.

The “if(gameOver) return” line stops the animation if the game has ended.

The touch control no longer works however and to correct this, adjust the OnControl() function to look like this:

```
//Handle screen touches and key presses.
function OnControl( touchState, touchX, touchY, keyState, key )
{
	//Increase dino's upward velocity when screen touched.
	if( touchState=="Up" && !gameOver ) dino.AddVelocity( 0, -0.3 )
}
```

Lesson 14 – Physics – Enclosures and Collisions

If you play with the game as it is now, you will find that the dino quickly goes off the edge of the screen. This can be stopped by adding a line that encloses the scene:

```
//Handle game loading.
function OnLoad()
{
    //Add physics and enclose scene with roof and floor.
    gfx.AddPhysics()
    gfx.Enclose( -1, "top,bottom" )

    //Create background (and stretch it to full screen).
    sky = gfx.CreateBackground( "Img/flappy_back.jpg", "stretch" )
```

We now need a different way of detecting collisions otherwise the crocs will push our dino off the edge of the screen. For this we will need two new functions. One called OnCollide() to detect the collisions between objects. The “Crash()” function is now called from here. This function should be added to the end of the code and look like this:

```
//Handle sprite collisions.
function OnCollide( a, b )
{
    console.log( "a:" + a.group + " b:"+b.group )

    if( a.group=="dinos" && b.group=="enemies" )
      {
            //If dino hits croc, then play the crash
            //animation and disable physics for croc, so
            //that way is clear for dino to fall to floor.
            Crash()
            croc.EnablePhysics( false )
      }
}
```

We can also add the following line to the OnAnimate() function to make the game harder by allowing the dino to crash into the roof or floor of the scene.

```
      //We need to update the physics state when we manually
      //move objects that use physics.
      croc.UpdatePhysics()

      //Check for dino collisions with spiky grass.
      if( dino.y > 0.8 && !gameOver ) Crash()
}
```

The complete code should now look like this:

```
//Global variables
var gameOver = false
var score = 0
//Handle game loading.
function OnLoad()
{
    //Add physics
    gfx.AddPhysics()
    gfx.Enclose( -1, "top,bottom" )

    //Create background (and stretch it to full screen).
    sky = gfx.CreateBackground( "Img/flappy_back.jpg", "stretch" )

    //Create graphical objects.
    dino = gfx.CreateSprite( "Img/dino_fly_783x136x3x1", "dinos" )
    croc = gfx.CreateSprite( "Img/crocs_149x686x1x2", "enemies" )
    crash = gfx.CreateSpriteSheet( "Img/dino_crash_920x138x4" )
    txt = gfx.CreateText( score.toString(), 0.1, "Img/Desyrel.xml" );

    gameover = gfx.CreateSprite( "Img/gameover.jpg", "gameover" )

    //Create sounds.
    jungle = gfx.CreateSound( "Snd/jungle.mp3" )
    crunch = gfx.CreateSound( "Snd/crunch.mp3" )
    end = gfx.CreateSound( "Snd/game_over.mp3" )
}

//Called when game has loaded.
function OnReady()
{
    //Set background.
    gfx.AddBackground( sky )

    //Add dino image
    gfx.AddSprite( dino, 0.1, 0.5, 0.15 )
    dino.SetPhysics( 1, "dynamic" )

    //Set dino physics shape to be a rectangle slighly smaller than
    //the source image boundary so that collions look more realistic.
    dino.SetShape( "rect", 0.7, 0.7 )


    //Add enemy sprites.
    gfx.AddSprite( croc, 0.8, 0.55, 0.12 )
    croc.SetPhysics( 2, "fixed", 1 )

    //Start animations
    dino.Play( 0, 0.25 )
    croc.Play( 0, 0.05 )

    //Add score text
    gfx.AddText( txt, 0, 0 );

    //Play looping background sound.
    jungle.Play(true)

    //Start the game.
    gfx.Play()
}
```

```
//Update game objects.
//(called for every frame)
function OnAnimate( time, timeDiff )
{
	//Do nothing if game is over.
	if( gameOver ) return

	//Make background slide to the left a small amount.
	sky.Scroll( -0.002, 0 )

	//Slide croc to left.
	croc.x -= 0.004

	//If croc has gone off left side of screen.
	if( croc.x < -croc.width )
	{
		//Move the croc back to the right hand side.
		croc.x = 1

		//Scale croc with a random value.
		var height = 0.4 + Math.random()*0.4
		croc.SetSize( null, height )
		croc.y = 1 - croc.height
		score++
		txt.SetText( score.toString() )
	}

	//We need to update the physics state when we manually
	//move objects that use physics.
	croc.UpdatePhysics()

	//Check for dino collisions with spiky grass.
	if( dino.y > 0.8 && !gameOver ) Crash()
}


//Handle screen touches and key presses.
function OnControl( touchState, touchX, touchY, keyState, key )
{
	//Increase dino's upward velocity when screen touched.
	if( touchState=="Up" && !gameOver ) dino.AddVelocity( 0, -0.3 )
}

//Crash our character and end the game.
function Crash()
{
	//Set gameover state.
	gameOver = true

	//Switch sprite sheets and change play speed.
	dino.SetSpriteSheet( crash )
	dino.Play( 0, 0.1 )

	//Play crunch sound and prevent scroll.
	crunch.Play()
	//Play 'game over' sound after half a second.
	 end.Play( false, 500 )
	setTimeout(GameOver, 500)
}
```

```
function GameOver()
{
    gfx.AddSprite( gameover, 0, 0, 1, 1 )
}

//Handle sprite collisions.
function OnCollide( a, b )
{
    console.log( "a:" + a.group + " b:"+b.group )

    if( a.group=="dinos" && b.group=="enemies" )
      {
            //If dino hits croc, then play the crash
            //animation and disable physics for croc, so
            //that way is clear for dino to fall to floor.
            Crash()
            croc.EnablePhysics( false )
      }
}
```

Lesson 15 – Game Over (Advanced)

The final thing we should add to this game is a way to retry once the game is over. To do this we will add another object we can click on when the “Game Over” screen appears.

First, under the Image assets, make sure you have added the file “replay.png”. Like before, we will then need to load the image in “OnLoad()” and also add an area that is used to detect touch within. Add this code as follows:

```
function OnLoad()
{
    //Add physics
    gfx.AddPhysics()
    gfx.Enclose( -1, "top,bottom" )

    //Create background (and stretch it to full screen).
    sky = gfx.CreateBackground( "Img/flappy_back.jpg", "stretch" )
    //Create graphical objects.
    dino = gfx.CreateSprite( "Img/dino_fly_783x136x3x1", "dinos" )
    croc = gfx.CreateSprite( "Img/crocs_149x686x1x2", "enemies" )
    crash = gfx.CreateSpriteSheet( "Img/dino_crash_920x138x4" )
    txt = gfx.CreateText( score.toString(), 0.1, "Img/Desyrel.xml" );

    gameover = gfx.CreateSprite( "Img/gameover.jpg", "gameover" )
    replay = gfx.CreateSprite( "Img/replay.png", "replay" )

    //Create sounds.
       jungle = gfx.CreateSound( "Snd/jungle.mp3" )
       crunch = gfx.CreateSound( "Snd/crunch.mp3" )
       end = gfx.CreateSound( "Snd/game_over.mp3" )
}
```

Finally we then need to add the code which adds and detects this touch of the replay button.

```
//Handle screen touches and key presses.
function OnControl( touchState, touchX, touchY, keyState, key )
{
       //Increase dino's upward velocity when screen touched.
       if( touchState=="Up" && !gameOver ) dino.AddVelocity( 0, -0.3 )

       else if( gameOver && replay.Contains(touchX,touchY) && touchState=="Down" )
       {
           gfx.Reload();
       }
}

function GameOver()
{
    gfx.AddSprite( gameover, 0, 0, 1, 1 )
    gfx.AddSprite( replay, 0.4, 0.7, 0.2)
}
```

The final complete code is as follows:

```
//Global variables
var gameOver = false
var score = 0

//Handle game loading.
//Handle game loading.
function OnLoad()
{
    //Add physics
    gfx.AddPhysics()
    gfx.Enclose( -1, "top,bottom" )

    //Create background (and stretch it to full screen).
    sky = gfx.CreateBackground( "Img/flappy_back.jpg", "stretch" )

    //Create graphical objects.
    dino = gfx.CreateSprite( "Img/dino_fly_783x136x3x1", "dinos" )
    croc = gfx.CreateSprite( "Img/crocs_149x686x1x2", "enemies" )
    crash = gfx.CreateSpriteSheet( "Img/dino_crash_920x138x4" )
    txt = gfx.CreateText( score.toString(), 0.1, "Img/Desyrel.xml" );

    gameover = gfx.CreateSprite( "Img/gameover.jpg", "gameover" )
    replay = gfx.CreateSprite( "Img/replay.png", "replay" )

    replay.Contains = function( x, y )
    {
        if( x > replay.x && x < replay.x + replay.width
        && y > replay.y && y < replay.y + replay.height ) return true
        else return false
    }

    //Create sounds.
       jungle = gfx.CreateSound( "Snd/jungle.mp3" )
       crunch = gfx.CreateSound( "Snd/crunch.mp3" )
       end = gfx.CreateSound( "Snd/game_over.mp3" )
}
```

Continued on the following pages:

```
//Called when game has loaded.
function OnReady()
{
    //Set background.
    gfx.AddBackground( sky )

    //Add dino image
    gfx.AddSprite( dino, 0.1, 0.5, 0.15 )
    dino.SetPhysics( 1, "dynamic" )

    //Set dino physics shape to be a rectangle slighly smaller than
    //the source image boundary so that collions look more realistic.
    dino.SetShape( "rect", 0.7, 0.7 )

    //Add enemy sprites.
    gfx.AddSprite( croc, 0.8, 0.55, 0.12 )
    croc.SetPhysics( 2, "fixed", 1 )

    //Start animations
    dino.Play( 0, 0.25 )
    croc.Play( 0, 0.05 )

    //Add score text
    gfx.AddText( txt, 0, 0 );

    //Play looping background sound.
    jungle.Play(true)

    //Start the game.
    gfx.Play()
}

//Update game objects.
//(called for every frame)
function OnAnimate( time, timeDiff )
{
       //Do nothing if game is over.
       if( gameOver ) return

       //Make background slide to the left a small amount.
       sky.Scroll( -0.002, 0 )

       //Slide croc to left.
       croc.x -= 0.004

       //If croc has gone off left side of screen.
       if( croc.x < -croc.width )
       {
              //Move the croc back to the right hand side.
              croc.x = 1

              //Scale croc with a random value.
              var height = 0.4 + Math.random()*0.4
              croc.SetSize( null, height )
              croc.y = 1 - croc.height
              score++
              txt.SetText( score.toString() )
       }

       //We need to update the physics state when we manually
       //move objects that use physics.
       croc.UpdatePhysics()

       //Check for dino collisions with spiky grass.
       if( dino.y > 0.8 && !gameOver ) Crash()
}
```

```
//Handle screen touches and key presses.
function OnControl( touchState, touchX, touchY, keyState, key )
{
	//Increase dino's upward velocity when screen touched.
	if( touchState=="Up" && !gameOver ) dino.AddVelocity( 0, -0.3 )

	else if( gameOver && replay.Contains(touchX,touchY) && touchState=="Down" )
	{
	    gfx.Reload();
	}
}


//Crash our character and end the game.
function Crash()
{
	//Set gameover state.
	gameOver = true

	//Switch sprite sheets and change play speed.
	dino.SetSpriteSheet( crash )
	dino.Play( 0, 0.1 )

	//Play crunch sound and prevent scroll.
	crunch.Play()
	//Play 'game over' sound after half a second.
	 end.Play( false, 500 )
	setTimeout(GameOver, 500)
}

function GameOver()
{
    gfx.AddSprite( gameover, 0, 0, 1, 1 )
    gfx.AddSprite( replay, 0.4, 0.7, 0.2)
}

//Handle sprite collisions.
function OnCollide( a, b )
{
    console.log( "a:" + a.group + " b:"+b.group )

    if( a.group=="dinos" && b.group=="enemies" )
	{
		//If dino hits croc, then play the crash
		//animation and disable physics for croc, so
		//that way is clear for dino to fall to floor.
		Crash()
		croc.EnablePhysics( false )
	}
}
```

The full tutorial project can be downloaded here:- http://androidscript.org/tutorials/

//End of code.

//End of tutorial.