Introduction to
Mobile App Development

MODULE 6:
Corona Physics Library

Module 6

1. Why do we need physics?
2. Physics Library Functions
3. Hello Physics App
4. Playing with Gravity App

Thompson Rivers University

Why do we need physics?

3

Physics

Thompson Rivers University

Game Engine

- Physics are commonly used for apps that involve a simulation of objects that move, collide, and interact under various physical forces like gravity.
- Corona makes it very easy to add physics to your apps, even if you've never worked with a physics engine before.

Example: Angry Birds - https://www.youtube.com/watch?v=2sAc0z7S7fI

https://docs.coronalabs.com/api/library/physics/index.html

Physics

Thompson Rivers University

Game Engine

- To work with the Corona physics engine, just begin with familiar Corona display objects.

- Corona treats physical body properties as an extension of its graphics objects, so any standard display object including images, vector objects, or animated sprites can be "made physical" and they will automatically interact with other objects in the simulation.

https://docs.coronalabs.com/api/library/physics/index.html

Thompson Rivers University

Physics Library Functions

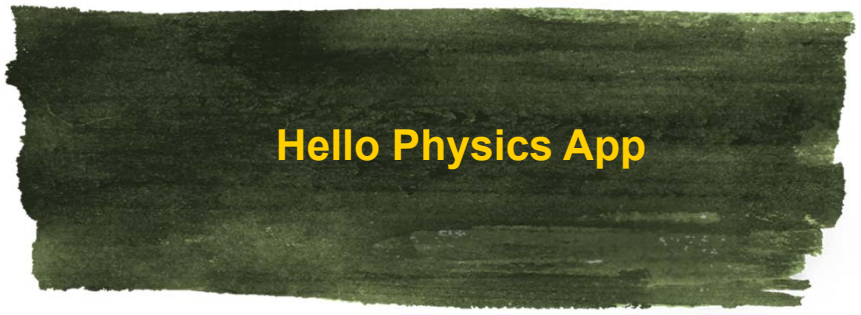
6

Physics Library Functions

List

- physics.addBody()
- physics.removeBody()
- physics.start()
- physics.pause()
- physics.stop()
- physics.newJoint()
- physics.rayCast()
- physics.reflectRay()
- physics.newParticleSystem()
- physics.queryRegion()
- physics.setGravity()
- physics.getGravity()
- physics.setDrawMode()
- physics.setScale()
- physics.setTimeStep()
- physics.setPositionIterations()
- physics.setVelocityIterations()
- physics.setContinuous()
- physics.setAverageCollisionPositions()
- physics.getAverageCollisionPositions()
- physics.setReportCollisionsInContentCoordinates()
- physics.getReportCollisionsInContentCoordinates()
- physics.setDebugErrorsEnabled()
- physics.getDebugErrorsEnabled()
- physics.setMKS()
- physics.getMKS()
- physics.toMKS()
- physics.fromMKS()
- physics.setTimeScale()
- physics.getTimeScale()

https://docs.coronalabs.com/api/library/physics/index.html



8

Hello Physics project

1

```
local physics = require( "physics" )
```

Hello Physics project

2

```
physics.start()
```

Hello Physics project

3

```
local sky = display.newImage
( "bkg_clouds.png", 160, 195 )
```

Hello Physics project

4

```
local ground = display.newImage
( "ground.png", 160, 445 )
```

Hello Physics project

5

```
physics.addBody
( ground, "static",
{ friction=0.5, bounce=0.3 } )
```

Hello Physics project

6

```
local crate = display.newImage
( "crate.png", 180, -50 )
```

Hello Physics project

7

```
crate.rotation = 15
```

Hello Physics project

8

```
physics.addBody( crate,
{ density=3.0, friction=0.5,
bounce=0.3 } )
```

Playing With Gravity App

Playing With Gravity

Steps

- Apply Physics Properties
- Add Listeners to Buttons
- Add a Function to Update Gravity Values onClick

Playing With Gravity

Steps

```
display.setStatusBar(display.HiddenStatusBar)

local physics = require("physics")
physics.start(true)
physics.setDrawMode("hybrid")
local gravityX = display.newText("0.0",  490, 875, native.systemFont, 36  )
local gravityY = display.newText("0.0", 195, 875, native.systemFont, 36 )

-- set inital value for gravity
physics.setGravity(0,-0.1)

-- initialize gx and gy to store gravity changes
gx = 0
gy = -0.1

local xCenter = display.contentCenterX
local yCenter = 768/2
```


Playing With Gravity

Steps

```
-- create border area so object doesn't fall off screen
local ground = display.newRect(xCenter, 768,  768, 10)
ground:setFillColor(1,1,1,1)

local leftSide = display.newRect(5, yCenter,10,768)
leftSide:setFillColor(1,1,1,1)
local rightSide = display.newRect(768,yCenter,10,768)
rightSide:setFillColor(1,1,1,1)

local top= display.newRect(379,5,768,10)
top:setFillColor(1,1,1,1)

 -- add border to physics as a static object (unaffected by gravity)
 physics.addBody(ground, "static")
 physics.addBody(leftSide, "static")
 physics.addBody(rightSide, "static")
 physics.addBody(top, "static")
```

Playing With Gravity

Steps

```
-- load the crate and add it as a body
local crate = display.newImage("crateB.png" )
crate.x = 389
crate.y= 389
physics.addBody(crate, "dynamic",{density=10.0, friction =1, bounce = 1})

-- load arrow buttons and position buttons
local upButton = display.newImage("arrowButton.png", 200, 800)
upButton.rotation = -90

local downButton = display.newImage("arrowButton.png", 200, 950)
downButton.rotation=90

local leftButton = display.newImage("arrowButton.png", 400, 875)
leftButton.rotation=180

local rightButton = display.newImage("arrowButton.png", 600, 875)
```


Playing With Gravity

Steps

```
-- Upadate the displayed value of gravity
local function updateGravity()
gx, gy = physics.getGravity()
gravityX.text = gx
gravityX.text=(gravityX.text:sub(1,4))
gravityY.text = gy
gravityY.text=(gravityY.text:sub(1,4))
end



-- add event listeners for each button
upButton:addEventListener("tap", upButtonEvent)
downButton:addEventListener("tap", downButtonEvent)
leftButton:addEventListener("tap", leftButtonEvent)
rightButton:addEventListener("tap", rightButtonEvent)
```

```
-- adjust the gravity for each button event
local function upButtonEvent (event)
        physics.setGravity(gx,gy-0.1)
        updateGravity()
end

local function downButtonEvent (event)
        physics.setGravity(gx,gy+0.1)
        updateGravity()
end

local function leftButtonEvent (event)
        physics.setGravity(gx-0.1,gy)
        updateGravity()
end

local function rightButtonEvent (event)
        physics.setGravity(gx+0.1,gy)
        updateGravity()
end
```

End of Module 6