By Majd Abdulqadir

Robot Rush Game

Game documentation and HowTo guide.

ROBOT RUSH
TAP ON ENEMIES TO SHOOT THEM, BUT DON'T LET THEM GET TO YOU!

This document contains:

Package Description and features

Robot Rush is an action packed game full of challenge and fun. The game is ready to release straight out of the box, and it can also be easily customized to make even more engaging to your players. The game supports PC/Mac, iOS, Android. It can be played with the mouse, keyboard, gamepad, or touch controls!

How to Play?

Tap on enemies to rotate and shoot at them, and collect items by shooting them too. Don't let enemies reach you.

Features:
- Game ready for release straight out of the box, just build and play!
- Works on all platforms, PC, Mac, iOS, Android, etc
- Supports multiple resolutions and aspect ratios, automatically.
- Supports Mouse and Touch controls.
- Easily customizable with lots of options to control game difficulty.
- Great learning resource with commented scripts and documentation.

Current version 1.10

Update history

1.10 (21.09.2018)
- Option to automatically shoot while holding the button, like a machine gun
- Option to shoot immediately without needing to rotate to the target position

1.05 (28.08.2018)
- Added Victory screen when finishing all rounds in a level

1.0 (26.07.2018)
- Initial version

Credits

The font used is Maniac by Vladimir NIkolic

The sounds are courtesy of the free sound project.

Music is **Your Sparking Rubber Soul** from the album **Fractal Planetoid 2.0** (Public Domain)

Credits go to these authors for their great sound samples: **adcbicycle, oddworld, ggctuk, lmr9, nickmorris, peridactyloptrix, cydon**

Please rate my file, I'd appreciate it 🙂

Overview of the game's library contents

Let's take a look inside the game files. Open the main RRGAssets folder using Unity 5.5 or newer. Take a look at the project library, usually placed on the right or bottom side of the screen. Here are the various folders inside:

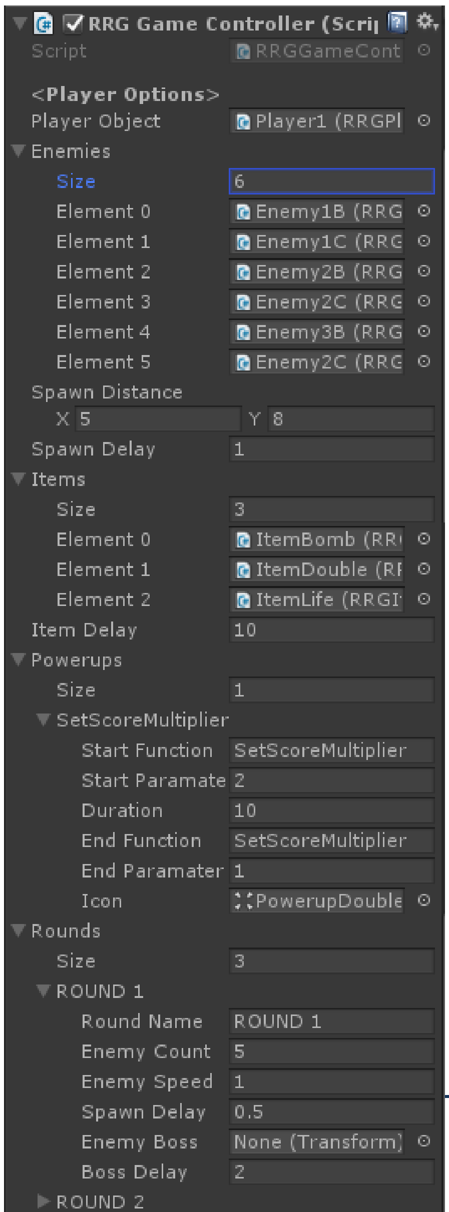
- **Animations:** Holds the animation clips made with Unity's built-in animation system.
- **FLA:** Holds the object graphics made with Flash CS3. These are vector graphics than can be easily scaled without loss of quality and then exported as PNG to be used in Unity.
- **Prefabs:** Holds all the prefabs used in the game. These are distributed to various folders for easier access, Buttons, Enemies, Objects, etc. It also holds all the canvases in the game which are used to hold buttons and other UI elements.
- **Scenes:** The first scene that runs in the game is RRGMenu. From this scene you can get to the Game scene.
- **Scripts:** Holds all the scripts used in the game. Each prefab contains one or more of these scripts.
- **Sounds:** Holds all the sounds used in the game. Shoot,Item, etc
- **Textures:** Holds all the textures used in the game which are used as sprites in Unity.

Customization Guide

Getting started

Robot Rush Game Template (RRG) is considered a complete project, and as such is supposed to work as the starting point of your planned game, rather than an addition to an existing project. That said, you may of course pick and choose some of the scripts/models to import into your existing project, but RRG works best as a starter kit which you can customize any part of to your liking.

The Game Controller



RRG Game Controller (Scri	
Script	RRGGameCont
<Player Options>	
Player Object	Player1 (RRGPl
Enemies	
Size	6
Element 0	Enemy1B (RRG
Element 1	Enemy1C (RRG
Element 2	Enemy2B (RRG
Element 3	Enemy2C (RRG
Element 4	Enemy3B (RRG
Element 5	Enemy2C (RRG
Spawn Distance	
X 5	Y 8
Spawn Delay	1
Items	
Size	3
Element 0	ItemBomb (RR
Element 1	ItemDouble (R
Element 2	ItemLife (RRGI
Item Delay	10
Powerups	
Size	1
SetScoreMultiplier	
Start Function	SetScoreMultiplier
Start Paramate	2
Duration	10
End Function	SetScoreMultiplier
End Paramater	1
Icon	PowerupDouble
Rounds	
Size	3
ROUND 1	
Round Name	ROUND 1
Enemy Count	5
Enemy Speed	1
Spawn Delay	0.5
Enemy Boss	None (Transform)
Boss Delay	2
ROUND 2	

The Game Controller is the main prefab that controls all the progress of the game from start to finish. It controls the UI of the game, creates and throws targets at the player. The Game Controller is also used to calculate the bonus the player gets when hitting a target.

Player Object – The default player object. If we have a shop in the scene, the player is assigned from it instead.

Enemies – A list of enemies that will be spawned randomly in all rounds in the scene.

Spawn Distance – The distance range where enemies are spawned around the player position.

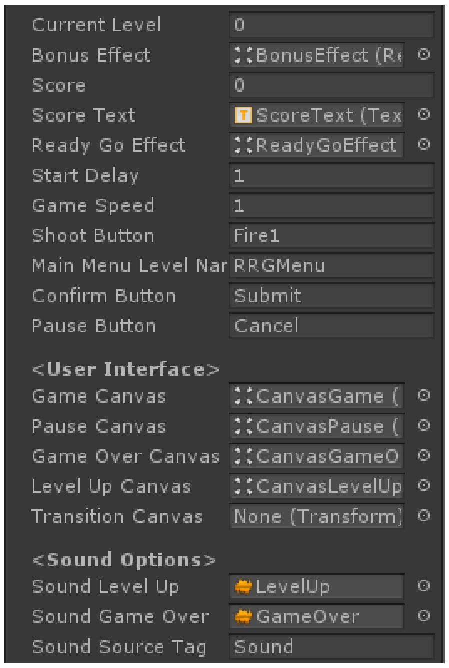
Spawn Delay – The default delay between spawning enemies.

Items – A list of items that will be spawned randomly in all rounds in the scene.

Item Delay – A list of powerups that can be activated in the scene. Each power up triggers a custom function in the gamecontroller, and after a delay it triggers another function (supposedly the opposite of the powerup, to turn it off).

Rounds – A list of levels in the game, including the number of kills needed to win the game, the speed of enemies, and the spawn rate of enemies.

- **Round Name –** The name of the current round.
- **Enemy Count –** The number of kills needed to win this level.
- **Enemy Speed –** The speed of the enemies in this level.
- **Spawn Delay –** How quickly enemies are spawned in this level.
- **Enemy Boss –** The enemy boss that is spawned in this level.
- **Boss Delay –** How long to wait before spawning the boss.



Current Level	0
Bonus Effect	BonusEffect (R
Score	0
Score Text	ScoreText (Tex
Ready Go Effect	ReadyGoEffect
Start Delay	1
Game Speed	1
Shoot Button	Fire1
Main Menu Level Nar	RRGMenu
Confirm Button	Submit
Pause Button	Cancel
<User Interface>	
Game Canvas	CanvasGame (
Pause Canvas	CanvasPause (
Game Over Canvas	CanvasGameO
Level Up Canvas	CanvasLevelUp
Transition Canvas	None (Transform)
<Sound Options>	
Sound Level Up	LevelUp
Sound Game Over	GameOver
Sound Source Tag	Sound

Current Level – The index number of the current level we are on.
Bonus Effect – The bonus effect that shows how much bonus we got when we hit a target.

Score – The score of the game. Score is earned by shooting enemies and getting streaks.

Score Text – The text object that displays the score, assigned from the scene.

ReadyGoEffect – The effect displayed before starting the game.

Game Speed – The overall game speed. This affects the entire game (Time.timeScale).

ShootButton – The button that shoots a bullet.

Main Menu Level Name – The level of the main menu that can be loaded after the game ends.

Confirm Button – The keyboard/gamepad button that will restart the game after game over.

Pause Button – The keyboard/gamepad button that pauses the game.

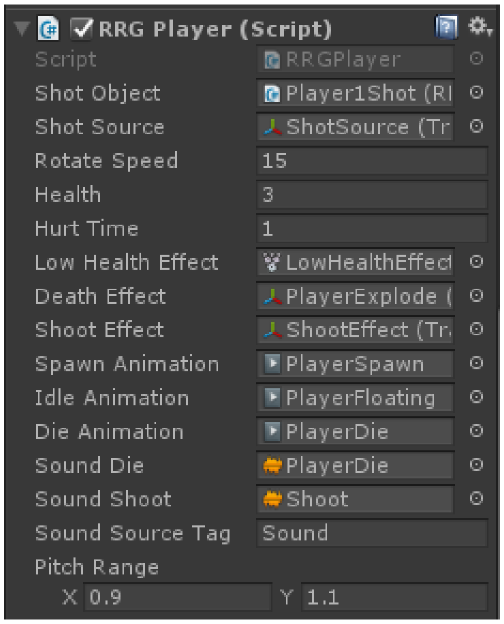
User Interface – Various canvases for the UI, assign them from the scene.

Sounds – Various sounds that play during the game.

Sound Source Tag – The audio source from which the Game Over sound plays.

Editing the Player

The player can rotate and shoot at enemies. Here is what you can change in the player component:



RRG Player (Script)

Script	RRGPlayer
Shot Object	Player1Shot (RI
Shot Source	ShotSource (Tr
Rotate Speed	15
Health	3
Hurt Time	1
Low Health Effect	LowHealthEffect
Death Effect	PlayerExplode (
Shoot Effect	ShootEffect (Tr
Spawn Animation	PlayerSpawn
Idle Animation	PlayerFloating
Die Animation	PlayerDie
Sound Die	PlayerDie
Sound Shoot	Shoot
Sound Source Tag	Sound
Pitch Range	X 0.9 Y 1.1

Shot Object - The shot that this player can shoot and hit enemies with.

Shot Source - The source from which shots are fired. The muzzle of a gun.

Rotate Speed – How fast the player rotates towards the target position.

Health – How much damage we can take before dying.

Hurt Time – A delay before allowing the player to be hit again.

Low Health Effect – The effect that appears when the player is near death.

Death Effect – The effect that is created at the location of this object when it is destroyed.

Shoot Effect - The muzzle effect when shooting.

Animations – These are various animation clips that play during the game.

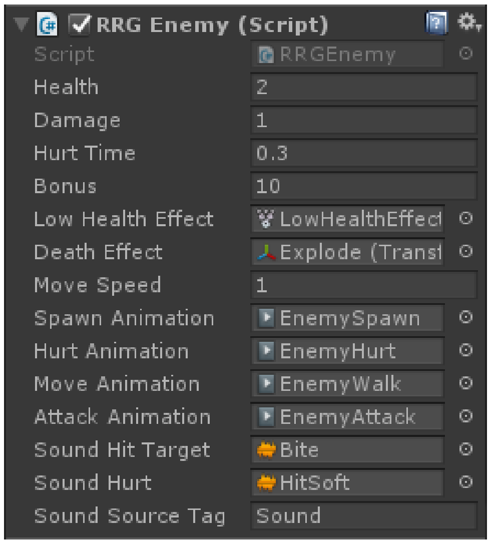
Sounds – These are various sound effects that play during the game.

Sound Source Tag – The source from which sounds are played.

Pitch Range - A random range for the pitch of the audio source, to make the sound more varied.

Editing the Enemy

The enemy is spawned around the player randomly. It has a health value and will kill the player if it touches it. Here is what you can change in the enemy component:



RRG Enemy (Script)

Script	RRGEnemy
Health	2
Damage	1
Hurt Time	0.3
Bonus	10
Low Health Effect	LowHealthEffect
Death Effect	Explode (Transt
Move Speed	1
Spawn Animation	EnemySpawn
Hurt Animation	EnemyHurt
Move Animation	EnemyWalk
Attack Animation	EnemyAttack
Sound Hit Target	Bite
Sound Hurt	HitSoft
Sound Source Tag	Sound

Health – The health value of the enemy. If health reaches 0 the enemy dies.

Damage – The health value of the enemy.

Hurt Time - How long the hurt effect is active when the enemy gets hurt.

Bonus – How many points we get from killing this enemy.

Low Health Effect – The effect that appears when this enemy's health is low.

Death Effect - The effect that is created at the location of this enemy when it dies.

Move Speed - The movement speed of the enemy. This is controlled through the Levels in the Game Controller.

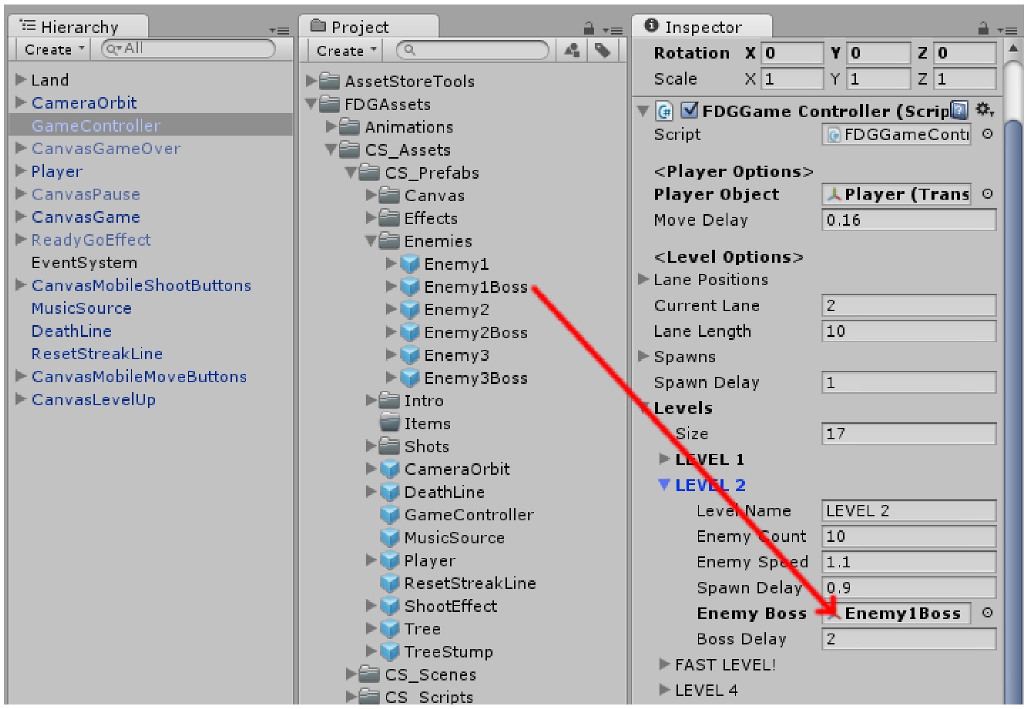
Animations – These are various animation clips that play during the game.

Sounds – These are various sound effects that play during the game.

Sound Source Tag – The source from which sounds are played.

What about bosses?

Bosses are just like normal enemies, except they have higher health and they appear at the end of a level. In each level you can choose to assign a boss or not.



Hierarchy
Create
Land
CameraOrbit
GameController
CanvasGameOver
Player
CanvasPause
CanvasGame
ReadyGoEffect
EventSystem
CanvasMobileShootButtons
MusicSource
DeathLine
ResetStreakLine
CanvasMobileMoveButtons
CanvasLevelUp

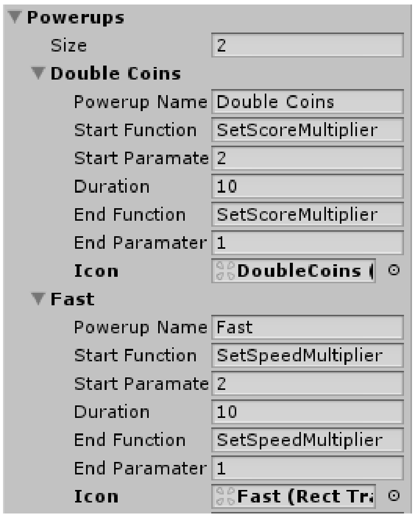
Project
Create
AssetStoreTools
FDGAssets
Animations
CS_Assets
CS_Prefabs
Canvas
Effects
Enemies
Enemy1
Enemy1Boss
Enemy2
Enemy2Boss
Enemy3
Enemy3Boss
Intro
Items
Shots
CameraOrbit
DeathLine
GameController
MusicSource
Player
ResetStreakLine
ShootEffect
Tree
TreeStump
CS_Scenes
CS_Scripts

Inspector
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
FDGGame Controller (Scrip
Script FDGGameContr
<Player Options>
Player Object: Player (Trans
Move Delay: 0.16
<Level Options>
Lane Positions
Current Lane: 2
Lane Length: 10
Spawns
Spawn Delay: 1
Levels
Size: 17
LEVEL 1
LEVEL 2
Level Name: LEVEL 2
Enemy Count: 10
Enemy Speed: 1.1
Spawn Delay: 0.9
Enemy Boss: Enemy1Boss
Boss Delay: 2
FAST LEVEL!
LEVEL 4

Simply drag and drop the boss enemy in the Enemy Boss slot. You can also set how many seconds to wait before spawning the boss after all other enemies have been spawned.

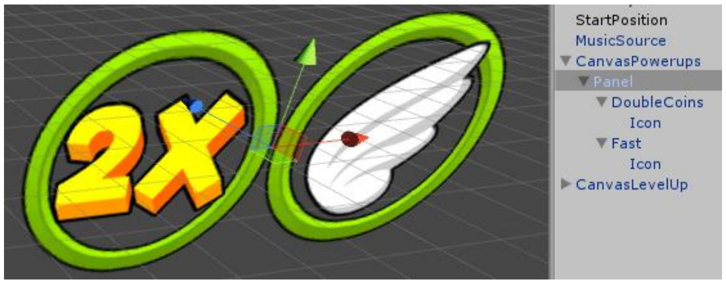
Editing Powerups

Powerups are special items you can pick up in the game. When a powerup is activated it runs a function in the gamecontroller, using SendMessage. The list of powerups you have is defined in the game controller.



```
▼ Powerups
    Size                2
  ▼ Double Coins
      Powerup Name      Double Coins
      Start Function    SetScoreMultiplier
      Start Paramate    2
      Duration          10
      End Function      SetScoreMultiplier
      End Paramater     1
      Icon              DoubleCoins (
  ▼ Fast
      Powerup Name      Fast
      Start Function    SetSpeedMultiplier
      Start Paramate    2
      Duration          10
      End Function      SetSpeedMultiplier
      End Paramater     1
      Icon              Fast (Rect Tr
```

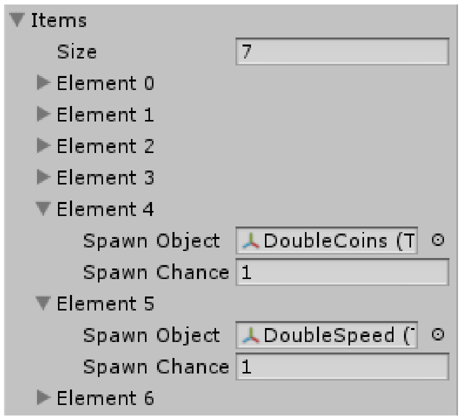
Let's see how the powerups are listed. The first power up doubles the score from collecting coins. It does so by sending a **SetScoreMultiplier** function command with a parameter of 2. Then the game counts 10 seconds before sending a **SetScoreMultiplier** function again with a parameter of 1, making the score multiplier return to normal. Finally we assigned an icon to represent the power up duration while it's activated. You can see what the icon looks like below.



This is the powerup icon, made of a fillamount circle shows the duration of the powerup and an icon.

Another powerup we have speeds up the player to twice its normal speed for 10 seconds. This is done by sending a **SetSpeedMultiplier** function command with a parameter of 2. To end the power up duration we wait 10 seconds and then call the same function but with a parameter of 1, making the player speed go back to normal.

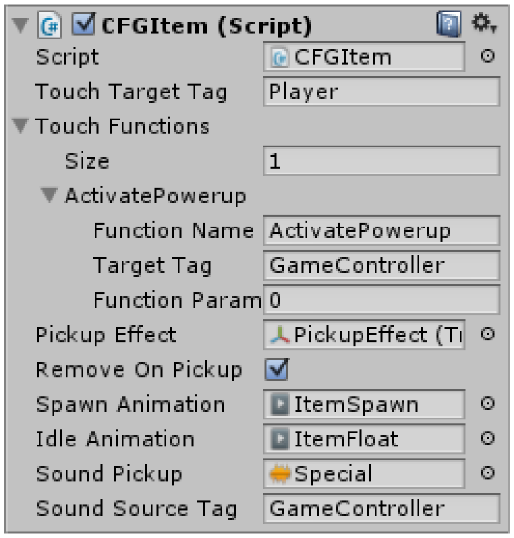
In order for the powerups to work we need to put items in the game which activate those powers when picked up. We also do this in the game controller.



```
▼ Items
    Size              7
  ▶ Element 0
  ▶ Element 1
  ▶ Element 2
  ▶ Element 3
  ▼ Element 4
      Spawn Object    DoubleCoins (T
      Spawn Chance    1
  ▼ Element 5
      Spawn Object    DoubleSpeed (
      Spawn Chance    1
  ▶ Element 6
```

The **Items** list holds all the possible items that can show up in a game. The item is chosen used a weighted randomization determined by the value of **SpawnChance**. The higher the value the more chance this item has to appear.

Let's take a look at the items we pick up in the game and how they activate a powerup. Go to the **Prefabs > Items** folder and drag **DoubleCoins** and **DoubleSpeed** items to the scene. The component in those prefabs is called **GFGItem**.

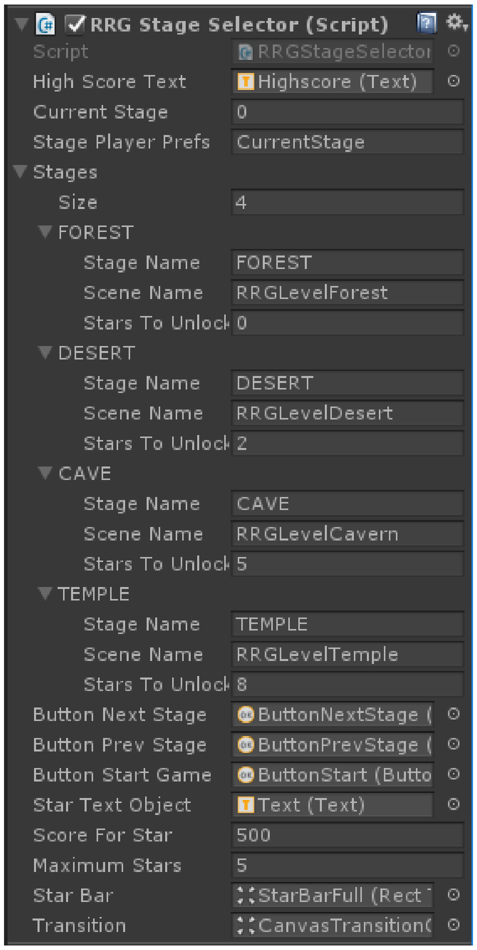


```
▼ CFGItem (Script)
  Script              CFGItem
  Touch Target Tag    Player
▼ Touch Functions
    Size              1
  ▼ ActivatePowerup
      Function Name   ActivatePowerup
      Target Tag      GameController
      Function Param  0
  Pickup Effect       PickupEffect (Ti
  Remove On Pickup    ✓
  Spawn Animation     ItemSpawn
  Idle Animation      ItemFloat
  Sound Pickup        Special
  Sound Source Tag    GameController
```

Click on the DoubleCoins object and you'll see the component, in which we called an **ActivatePowerup** function command from the gamecontroller, with a parameter of 0. This activates the ***first*** powerup in the powerups list in the gamecontroller, which is the **DoubleCoins** powerup. If you take a look at the **DoubleSpeed** object you'll see that the parameter we sent is 1, which activates the ***second*** powerup in the list.

The Stage Selector

The stage selector allows you to have several stages in the game. Each level must be unlocked by earning a certain number of stars. Each star can be earned when getting a certain score in a stage. You can customize all of these settings.



```
RRG Stage Selector (Script)
Script                RRGStageSelector
High Score Text       Highscore (Text)
Current Stage         0
Stage Player Prefs    CurrentStage
Stages
  Size                4
  FOREST
    Stage Name        FOREST
    Scene Name        RRGLevelForest
    Stars To Unlock   0
  DESERT
    Stage Name        DESERT
    Scene Name        RRGLevelDesert
    Stars To Unlock   2
  CAVE
    Stage Name        CAVE
    Scene Name        RRGLevelCavern
    Stars To Unlock   5
  TEMPLE
    Stage Name        TEMPLE
    Scene Name        RRGLevelTemple
    Stars To Unlock   8
Button Next Stage     ButtonNextStage (
Button Prev Stage     ButtonPrevStage (
Button Start Game     ButtonStart (Butto
Star Text Object      Text (Text)
Score For Star        500
Maximum Stars         5
Star Bar              StarBarFull (Rect T
Transition            CanvasTransition(
```

High Score Text – The text object that shows the high score we got in the current stage.

Current Stage – The current stage we are in. 0 is the first stage, 1 the second.

Stage Player Prefs – This is the name of the record that holds the current stage we are in. This is saved locally.

Stages – A list of stages you can unlock and play.

Stage Name – The name that appears when we select a stage. This is not the name of the scene that is loaded.

Scene Name – The scene that is loaded when we start this stage.

Stars To Unlock – The number of stars needed in order to unlock this level.

Button Next/Prev/Start – These buttons assigned from the scene let us switch stages and start the stage.

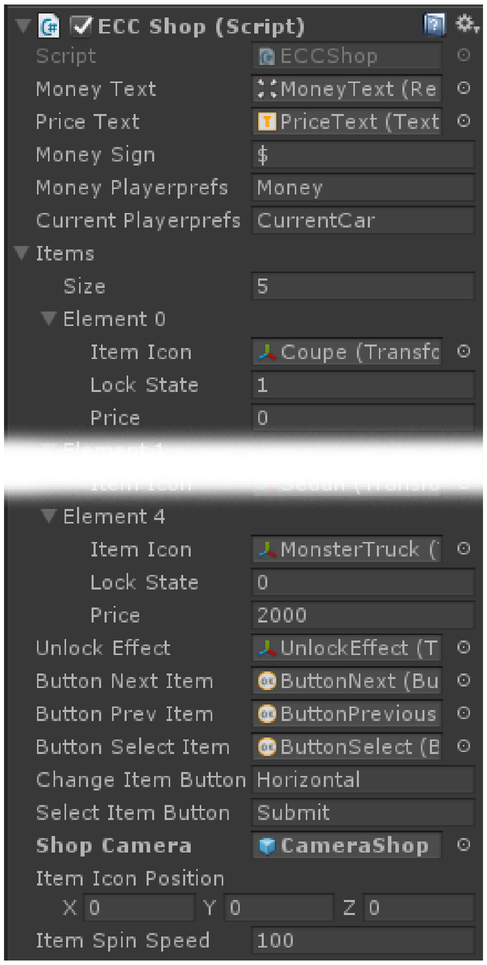
Start Text Object – The text inside the start button which also shows how many stars we need to unlock a level.

Score For Star – The number of points needed in a level in order to get a star. This is calculated based on the high score we got in the level, so for example if we have 100 highscore we get 1 star, and for 200 highscore we get 2 stars.

Maximum Stars – The maximum stars we can get in a stage.

The Shop

This script displays items that can be unlocked with in-game money. Each item has a 3D icon that spins while showing the price of the item if it is still locked. Locked items are also darkened.



ECC Shop (Script)	
Script	ECCShop
Money Text	MoneyText (Re
Price Text	PriceText (Text
Money Sign	$
Money Playerprefs	Money
Current Playerprefs	CurrentCar
Items	
Size	5
Element 0	
Item Icon	Coupe (Transfo
Lock State	1
Price	0
Element 1	
Item Icon	Sedan (Transfo
Element 4	
Item Icon	MonsterTruck (
Lock State	0
Price	2000
Unlock Effect	UnlockEffect (T
Button Next Item	ButtonNext (Bu
Button Prev Item	ButtonPrevious
Button Select Item	ButtonSelect (B
Change Item Button	Horizontal
Select Item Button	Submit
Shop Camera	CameraShop
Item Icon Position	X 0 Y 0 Z 0
Item Spin Speed	100

Money Text – The text object that displays the money we have.

Price Text – The text object that displays the price of the current item.

Money Sign – The sign that appears next to the money text.

Money Player Prefs – The player prefs record of the money we have.

Current Player Prefs – The player prefs record of the current item we selected.

Items – A list of items in the game, their names, the scene they link to, the price to unlock it, and the state of the item (-1 - locked and can't be played, 0 - unlocked no stars, 1-X the star rating we got on the item.

Item Icon - The object representing this item, can be a 2D icon or a 3D object.

Lock State - Is the item locked or not? 0 = locked, 1 = unlocked.

Price - How much money we need to unlock this item.

Unlock Effect - The effect that appears when we unlock this item.

Buttons - Buttons for going to the next/previous items, and selecting items**.**

Shop Camera - The camera object that is turned on when this shop is activated**.**

Item Icon Position - Buttons for going to the next/previous items, and selecting items**.**
Item Spin Speed - The rotation speed of the currently se

UnityAds Integration (Unity 5.2 +)

Since Unity 5.2 UnityAds integration has been simplified, here's how you can have full screen video ads in your game.

This video shows a quick process of integrating UnityAds into your project. In the example we used one of my templates, but it works on all my other templates too.

https://www.youtube.com/watch?v=EQNTgfV35DU

Here is what we did in the process:

1. Sign in to your Unity account in order to allow Unity Services such as UnityAds to be activated.

2. Open Build Settings and switch the platform to one of the supported ones (iOS, Android).

3. Download Puppeteer's UnityAds package from: puppeteerinteractive.com/freebies/PUPUnityAds.unitypackage

4. Drag the downloaded package into your Unity project, and import it. This UnityAds prefab can be used to display ads every several minutes.

5. Drag the prefab into any scene where you want ads to be shown. Make sure to save changes.

6. The time check is shared between all prefabs in all scenes, so you will never show too many ads.

7. The final step is to activate UnityAds services and get your unique project ID.

8. Open the services window and choose your organization, then click create.

9. Choose UnityAds from the list and turn it On.

10. Choose age group for your project (Will affect the nature of ads shown), and save changes.

11. While working on your project keep Test Mode activated. But when you are ready to release the final project, switch Test Mode off.

12. That's it! Now when you start the game, an ad will be shown after 3 minutes. The ad will never appear during gameplay or post-game screen. Instead, it will wait until the next level load (restart, main menu, etc) and then show the ad.

Before releasing a game, make sure you uncheck **Enable Test Mode.**

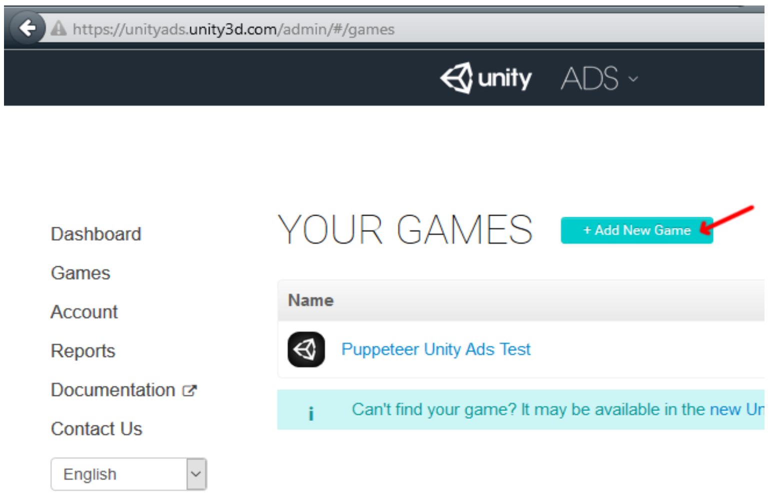
For more info about integrating UnityAds read this:

http://unityads.unity3d.com/help/monetization/integration-guide-unity

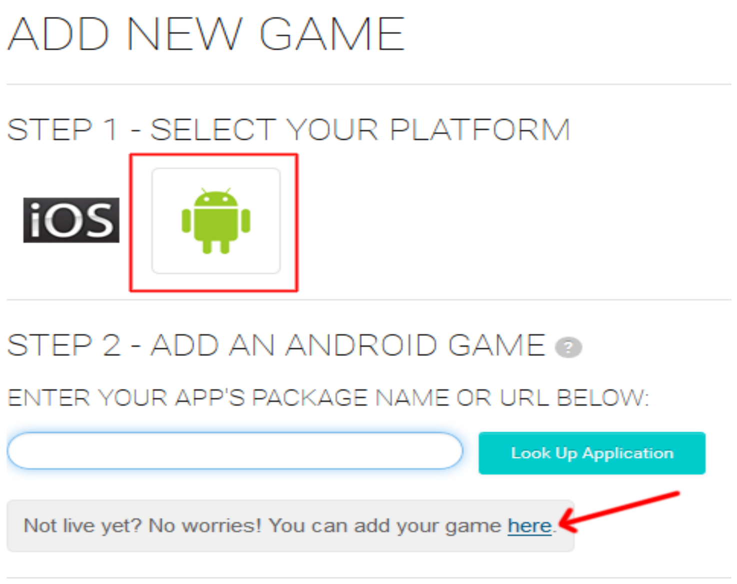
Integrating UnityAds into your project (Unity 4)

Adding support for UnityAds into your current project is simple and shouldn't take you more than 5 minutes. Let's start:

First we need to create our game entry on the UnityAds website. Go to https://unity3d.com/services/ads and create a new game. If you already have your app set and your GameID noted, just skip this part and go straight to importing the UnityAds package into the game.



Now we need to choose the platform. The process is similar for both iOS and Android but for the purpose of this tutorial we'll choose Android. If you have an app on Android, enter its name to find it. If you don't have an app, click below where the red arrow points in order to enter the name of the app that has not been added to the store yet. This way you can test the app before it goes live.


ADD NEW GAME

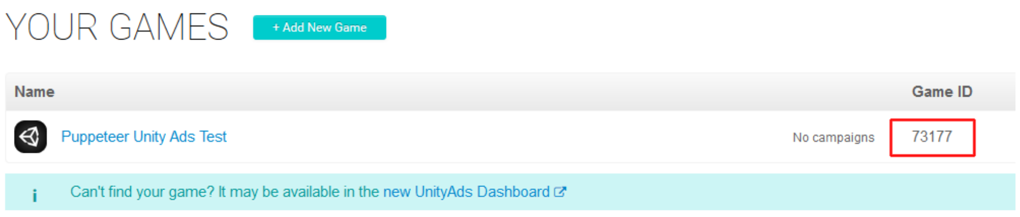
STEP 1 - SELECT YOUR PLATFORM

STEP 2 - ADD AN ANDROID GAME

ENTER YOUR APP'S PACKAGE NAME OR URL BELOW:

Not live yet? No worries! You can add your game here.

After you created your app in the website, make note of the Game ID that appears. This will be used to link the ads to your app.


YOUR GAMES

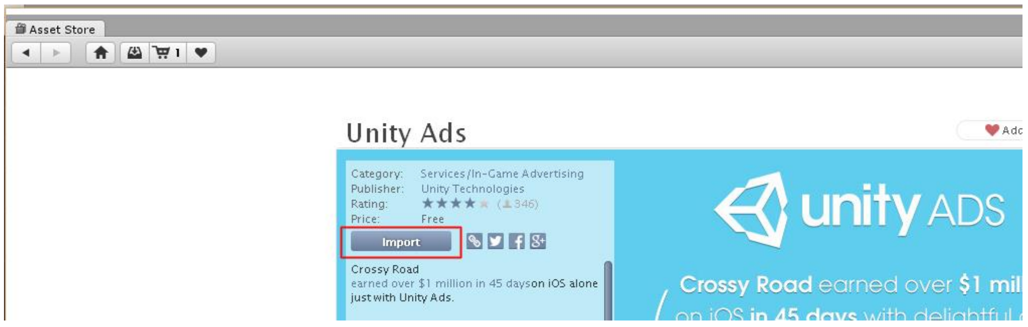
Name		Game ID
Puppeteer Unity Ads Test	No campaigns	73177

Can't find your game? It may be available in the new UnityAds Dashboard

In Unity Editor

Now we need to import the UnityAds package. Open the Unity Asset Store and download the UnityAds package. Import it into your project.

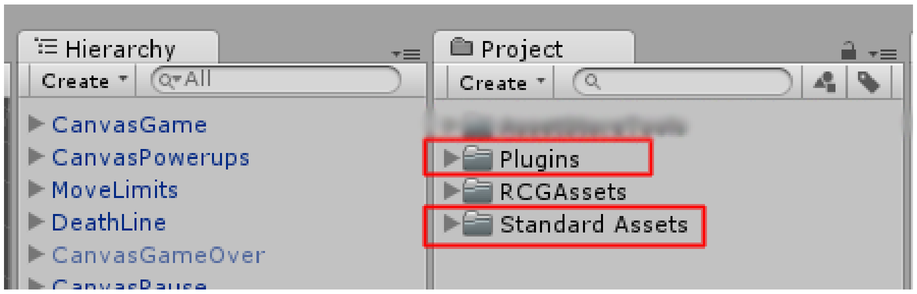
(https://www.assetstore.unity3d.com/en/#!/content/21027)



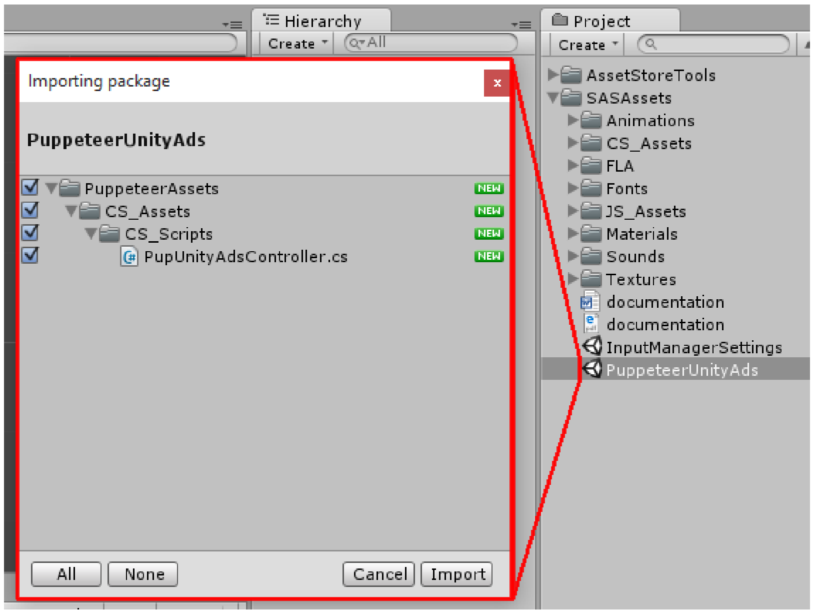
Category: Services/In-Game Advertising
Publisher: Unity Technologies
Rating: ★★★★ (346)
Price: Free

Crossy Road earned over $1 million in 45 days on iOS alone just with Unity Ads.

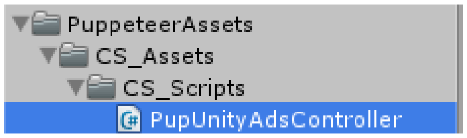
After import you should have two additional folders in your project.

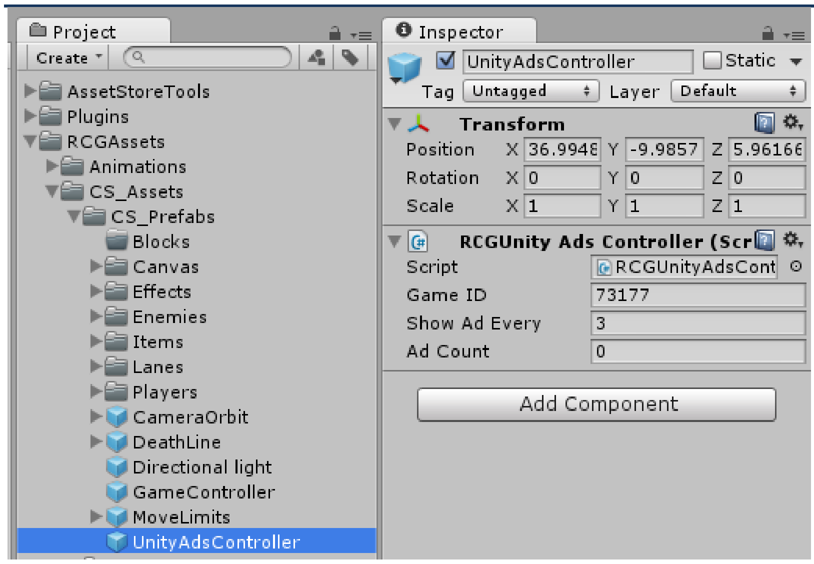


Now we need to bring in the code that integrates the ads into our game. Click on the **PuppeteerUnityAds** package in your project to import it into the game, or choose **Assets > Import Package > Custom Package…** from the top menu and navigate to the **PuppeteerUnityAds** package in your project to import it.



PupUnityAdsController.cs is the main script that links your app to the unityads system. Drag it into your game controller. Now when you look at it you see you can set the GameID of your app, and how often the ads appear. The ad is checked when the level is loaded. "**Show Ad Every**" decides how many times the level needs to be loaded before an ad appears.





UnityAdsController
Tag Untagged Layer Default

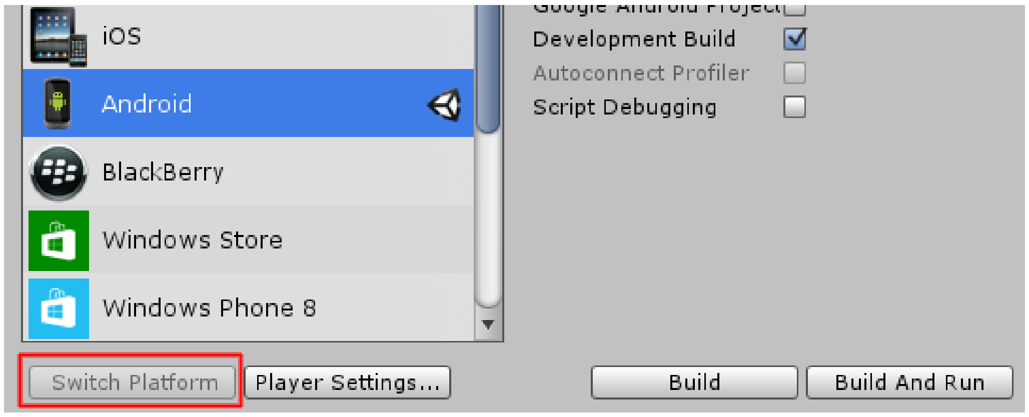
Transform

Position	X 36.9948	Y -9.9857	Z 5.96166
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

RCGUnity Ads Controller (Scr

Script	RCGUnityAdsCont
Game ID	73177
Show Ad Every	3
Ad Count	0

In order to test the ads, we need to switch to the Android platform.



Development Build ✓
Autoconnect Profiler
Script Debugging

That’s it! Now start a level and restart it 3 times, then you should see a blue screen showing the ad system has been activated correctly. If you build to Android you should see an actual video ad appear after 3 level loads.

It is highly advised, whether you are a designer or a developer to look further into the code and customize it to your pleasing. See what can be improved upon or changed to make this file work better and faster. Don't hesitate to send me suggestions and feedback to puppeteerint@gmail.com

Follow me on twitter for updates and freebies!

Good luck with your modifications!