

Lava Jump!

Modding Guide

Last Updated: 09 JUNE 2018



Product Overview

'Lava Jump!' is a clone of the game 'The Floor Is Lava'.

It is designed to give you a starting point for Modelling and Improving the original game mechanic.

Features Summary

- **Six Art Themes** - Selected by changing one variable.
- **Full Interfaces:** Splash, Home, Play, Help, Gameover.
- **Ads:** Applovin or Appodeal: banner+interstitial.
- **Social:** Twitter, Rating, Sharing (Activity Popup).
- **Additional Feature Hooks:** Facebook, Achievements,

Leaderboards, and Store.

This is a complete game, ready for skinning and improving. It is perfect for learning and for making money.

License

Unless otherwise explicitly stated in your purchase agreement,

- **You may** use this template to learn from on your own.
- **You may** use this template to make a single compiled game and distribute that game on as many app stores as you wish, including but not limited to:
 - Android Binary on Google Play, ...
 - iOS Binary on - Apple App Store
 - Windows, OSX, or HTML5 on Itch.io, Game Jolt, ...
 - HTML5 on Armor Games, Kongregate, ...
 - etc.
- **You may not** (unless your purchase license exclusively states otherwise) make multiple games without buying additional copies of the template.
- **You may not** distribute any part of this template and its content to a third party, except in a compiled game as stated above.
- **You may not** use this template in a classroom setting without express permission. (See 'Contact Info & Help' Below)

Features

This template contains these features:

- **Detailed Guide** - You're reading it now. It provides steps on how to modify the game, framework, as well as how to
 - How To Get Started with Corona
 - Create Twitter App
 - Get IDs and License Keys
 - How to build for release on Google Play and the Apple Store
 - ...
- **Game Code** - Clone of "The Floor Is Lava".
- **Free Art** - 6 FREE example art themes to make it easy to re-skin and modify the game.
- **Free Sounds** - Basic sounds made with sfxr/bfxr.
- **Full Interfaces** -
 - Splash Screen
 - Home (Main Menu)
 - Play Screen
 - About Screen
- **Ads** - Banners and Interstitials via AppLovin or Appodeal.
- **NoAds In-App-Purchases (IAP)** - Pre-coded NoAds IAP item.
- **Twitter** - Pre-coded twitter support.
- **(Activity Popup) Sharing** - Pre-coded Activity Popup code for iOS builds.
- **Rating** - Pre-coded rating code for Android, old iOS and iOS > 10.3.
- **Extras Hooks** - Additional hooks to launch Facebook, Achievements, Leaderboards, and an IAP Shop.
 - **Note:** These are just launchers and include little to no functional code for these features.
- **GDPR Ready** - This template has code to help you request permission from the player and to then enable/disable GDPR consent settings for both Ad providers.

Getting Started With Corona

While I would like to assume you already have Corona installed on your machine and know the basics, I am also (secretly) hoping you are new to Corona and part of the reason you are using it is because of this product.

Note: The following pages contain a summary of the most important steps, but for a much more detailed walk through please visit Corona's site here: <https://coronalabs.com/learn/>.

First steps

If you are new or just need a little help getting started, please follow these steps:

1. Create an FREE account at Corona Labs: <https://coronalabs.com/>
2. Download the latest DAILY build, not the public build:
<https://developer.coronalabs.com/downloads/daily-builds/>
 - **Tip:** The DAILY build is the most recent version of Corona today, while public builds are stable, but can be much older.
 - To get the build, go to the link above then find the top row of releases and download the *.msi file for Windows or *.dmg file for OS X
3. Install Corona on your machine.
4. Run it for the first time, and log in with your account details.
 - You will only need to do this once.

Testing The Installation

1. (If you quit) Start Corona again.
2. Examine the Launcher dialog and find the 'Samples' link. Click it.
3. Choose any sample and run it.
4. Try a few of these to verify your install is working.

If you cannot run the samples, please visit the Corona site here for additional help:

<https://docs.coronalabs.com/guide/programming/intro/index.html#systemreqs>

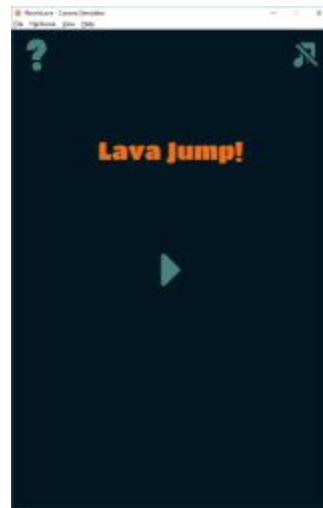
Run This Template

Now that you have Corona installed and running, please run Corona once more and open the template by going to the **template/** folder and opening **main.lua**.

The game will launch and you should see these scenes:



Splash Scene



Home Scene



Play Scene



About Scene

Now, you are ready to modify the template. Simply follow the directions in this guide and enjoy!

Tip: There is an index at the end of this guide to help you find specific topics.

Modifying The Game

While it is expected that you have some understanding of Corona and Lua, this guide is meant to help you along as you make this game your own.

The following is a terse guide on how to modify the various parts of template.

Selecting Different Themes

Changing themes is as simple as changing one line of code.

Using your favorite code editor, open the file **scripts/common.lua** and locate this line:

```
common.theme = "ansimuz"  
-- casual flat medieval prototype ansimuz kenney
```

Then replace the string with any of these:

- casual
- flat
- medieval
- prototype
- ansimuz
- kenney

Reload the game (in the simulator) and viola, a new theme is selected!

Modifying Game Art

Before you modify the artwork, you should take some time studying the code and the art assets found in the **images/** folder and the theme-named subfolders.

For example, if we look in the **images/ansimuz/** folder we will find these images (some not listed):

- **background.png** - The image used as a background in the game and game interfaces. See the prototype/ version of this for safe areas (always visible) and bleed space (shown only on some devices.)

- **ground.png** - This is the ground image that shows when the game is started. It is designed to fill the width of any screen.
- **lava.png** - This is the lava and was made using base art from the Ansimuz packs. As with ground.png, it is designed to fit horizontally on any screen as well as to be tall enough for most settings you may set in **scripts/game.lua**. Be warned that you may need to adjust the height of this image if you make the jump height significantly greater.
- **plat.png, plat_lava.png** - There are two platform images for the game. The first is used for normal platforms and the second is used when lava gets close to a platform.
 - **Note:** You will see that the game code (**scripts/game.lua**) uses a simple texture wrapping trick to make it seem like we have four variations of each image file, when only two are ever in use.
- **player.png** - This is a simple two-frame sprite-sheet used as a fill for the player 'block'. You can elaborate on this further if you want to add more jumping and other animations.

Modifying Button Art and Code

The button artwork and button definition code is kept in the **presets/** folder under the various theme-named subfolders.

For example, if we look in the **presets/ansimuz/** folder we will find:

- **images/** - This is the folder with all of the artwork used when this theme is turned on.
- **images2x/** - These are the @2x images. If you choose to the Dynamic Image Selection (<https://docs.coronalabs.com/guide/basics/configSettings/index.html#dynamicimages>) feature of Corona:
 - Open **config.lua** and uncomment the 'Dynamic Image Selection' section.
 - Copy all of the **presets/images2x/<theme name>/*** files into:
presets/images2x/<theme name>/
- **presets.lua** - This file contains the button preset definitions. Please see the SSK2 docs for how to modify them:
 - https://roaminggamer.github.io/RGDocs/pages/SSK2/libraries/easy_interfaces/#making-button-presets
 - https://roaminggamer.github.io/RGDocs/pages/SSK2/libraries/easy_interfaces/#button-instances

Modifying Game Code

All of the core game code is located in the **scripts/** folder. These are the files you need to modify:

- **common.lua** - Go here to select a theme, and to set the `'releaseMode'`, as well as some common. flags & visual settings.
 - Theme settings - See 'Selecting Different Themes' above.
 - `'releaseMode'` - This is set to `'development'` by default. **YOU MUST** set it to `'production'` before you build for the store.
 - (Game) Flags - Do not change these. They are maintained by the code in **scripts/game.lua**.
 - Visual Settings - Go all the way to the end of the file to adjust corner offsets if you need to. You should only need to do this if you tweak the size of buttons and then only for iPhone X and other cases with a 'notch'.
 - The remaining bulk of this file is dedicated to framework 'extra' settings. See 'Modifying Framework Extra Features' below.
- **game.lua** - This is the CORE game file. It contains the entire implementations of the gameplay code. The code is thoroughly commented.
- **helpers.lua** - This is a module containing 'utility' style code.
- **init.lua** - This file is called from main.lua and handles the initialization of your games:
 - Button presets - You should not make any changes here. If you do want to change the button art and button settings, please see 'Modifying Button Art and Code' above.
 - Persistent Settings - score, best score, sound enable, etc.

- Sound - See the SSK2 docs for how to modify this code:
<https://roaminggamer.github.io/RGDocs/pages/SSK2/libraries/soundMgr/>
- Ads and IAP Extras - See 'Modifying Framework Extra Features' below.
- **easyAds.lua, easyIAP.lua, iap_badger.lua, and utils.lua** - These are all related to framework extras and covered in a separate guide. See 'Modifying Framework Extra Features' below.

Modifying composer.* Scenes

This game users composer.* to handle scene management. The scenes provided with this game are:

- **scenes/splash.lua** - The Splash screen showing the game title and your game company info.
 - Besides showing this information, the purpose of s
- **scenes/home.lua** - This is the main menu/home screen. It provides these interface elements:
 - **Play button** - Launches play scene.
 - **Sound toggle** - Toggles sound effects on/off.
 - **Info button** - Launches info scene.
 - **Game Over Summary** - When you die in the game, **scripts/game.lua** sends us back to the home scene and requests that the 'game over' summary be shown.
 - **Extras** - There are many additional buttons available, but by default they are all turned off. See 'Modifying Framework Extra Features' below for more information on how to enable and configure these features.
 -
- **scenes/play.lua** - This is thin scene that uses the code in **scripts/game.lua** to create and run the gameplay code.
- **scenes/about.lua** - The credits screen and where the user goes to 'restore' IAP if that feature is enabled.
- **scenes/_template.lua** - This is a blank starter scene in case you want to add more interfaces. Just copy it, rename the file copy, and start coding.

The About Scene

The about scene is set up with some default content and layout rules. To change the 'about' images and text All you have to do is edit the about table at the top of the file.

As you will find the table has this form:

```
local about = {
  { text = "some text... \n\n" },
  { text = "more text" },
  { image = "an image path", width = <a width>, height = <a height>
},
}
```

You could specify your own about with this text:

These are some notes about the game.

Some credits go here

More credits go here

... and so on ...

, by writing this code:

```
local about = {
  { text = "These are some notes about the game.\n\n" },
  { text = "Some credits go here\n" },
  { text = "more credits go here\n" },
  { text = "... and so on ..." },
}
```

You can add an image like this:

```
{ image = "images/rg256.png" },
```

The resulting image will be the full size image. Alternately you can force a size like this:

```
{ image = "images/rg256.png", width = 32, height = 32 },
```

Beyond that, you can easily modify the code in this file to make the parser more robust and versatile.

Modifying The Framework

In addition to the game content, this product comes with a complete framework, that includes these **extra** features (in alphabetical order):

- **Ads** - Banner and/or Interstitial and/or Rewarded video ads via Appodeal or Applovin.
- **In-App-Purchases (IAP)** - The framework uses the (most excellent) FREE IAP Badger and a thin custom layer to make adding and supporting IAP just a few lines of easy to read code.
- **Rating** - The framework supports rating on Android, and iOS. As well, for iOS 10.3 it uses the latest FREE rating popup plugin.
- **Sharing (iOS ONLY)** - On iOS you can give your users access to the activity pop-up so then can share their thoughts and feelings about your game with whatever social media apps they may have installed.
- **Twitter** - Lastly, the framework uses Jason Schroeder's PAID Twitter plugin to enable direct tweets via a 'Twitter App'.

The template also provides placeholder hooks if you want to add these features:

- Achievements & Leaderboards
- Facebook
- In-App-Purchase Shop

Important Note: By default, all **extra** features are DISABLED. See the following pages for instructions on how to enable and configure these features.

Ads

Make the following changes to enable and configure ads.

scripts/common.lua - Ad Settings

Enable Feature

```
common.extras.ads_enabled = true
```

Select Provider

```
common.extras.ads_provider= "applovin"
```

OR

```
common.extras.ads_provider= "appodeal"
```

Enable Interstitials

You may optionally enable interstitial ads.

```
common.extras.ads_show_interstitials = true
```

Set Interstitial Frequency

An interstitial ad is shown every Nth player death. In the code below, and ad is shown every 5th death. (*You can ignore this setting if you chose not to enable interstitials.*)

```
common.extras.ads_interstitial_frequency = 5
```

Choose Whether To Request GDPR Permission

By default, both ad networks are not given GDPR permission. However, if you want explicitly request permission you may enable this feature, by setting the following flag to `true`.

```
common.extras.ads_request_gdpr_permission = true
```

Edit Provider Helper

Depending on the provider you have chosen to use, edit either of these files:

- `scripts/ads/applovinAds.lua`
- `scripts/ads/appodealAds.lua`

AppLovin Helper

Locate this marker:

```
-- =====
-- BEGIN EDITING HERE * BEGIN EDITING HERE * BEGIN EDITING HERE
-- =====
```

, then read the comments in the code to configure the ad helper. The following is a summary of the changes you will make in this section:

- **fakeAds** - If you want to use the 'fake ads' feature in the simulator leave this line as it is, otherwise, comment it out.
- **testMode** - Leave this as `true` until you are ready to publish, then set it to `false`.
- **verboseLogging** - Set this to `true` if you want extra output from the AppLovin plugin.
- **androidID** - If you are publishing to Android, get a Android Ad ID from your AppLovin console and put it here.
- **iosID** - If you are publishing to Android, get a Android Ad ID from your AppLovin console and put it here.
- **initDelay** - You should leave this alone, unless you have dug into the code and understand what it does.
- **verbose** - Set this to `true` to enable detailed output from the ad listener.

Use Default GDPR Behavior

Please note, if you want the AppLovin plugin use its default behavior for GDPR, please find this code:

```
-- DO NOT EDIT THIS UNLESS YOU ARE AN EXPERT:
if( ssk.persist.get( "settings.json", "has_gdpr_consent" ) ) then
```

And modify it to read :

```
-- DO NOT EDIT THIS UNLESS YOU ARE AN EXPERT:
if( false and ssk.persist.get( "settings.json", "has_gdpr_consent" ) ) then
```

Appodeal Helper

Locate this marker:

```
-- =====
-- BEGIN EDITING HERE * BEGIN EDITING HERE * BEGIN EDITING HERE
-- =====
```

, then read the comments in the code to configure the ad helper. The following is a summary of the changes you will make in this section:

- **fakeAds** - If you want to use the 'fake ads' feature in the simulator leave this line as it is, otherwise, comment it out.
- **testMode** - Leave this as `true` until you are ready to publish, then set it to `false`.
- **supporteAdTypes** - This table lets you limit the ads Appodeal will serve. Right now it is correctly set for the way the template is configured to work.
- **disableAutoCacheForAdTypes** - This feature is commented out, but if you want you can uncomment it and modify it to disable caching for specific ad types. This is not a good idea unless you want to take direct charge of when ads are loaded.
- **isCOPPACompliant** - This is set to `false` by default, but if your game is COPPA compliant, set this to `true` to get ads more suitable for children.
- **androidID** - If you are publishing to Android, get a Android Ad ID from your Appodeal console and put it here.
- **iosID** - If you are publishing to Android, get a Android Ad ID from your Appodeal console and put it here.
- **initDelay** - You should leave this alone, unless you have dug into the code and understand what it does.
- **verbose** - Set this to `true` to enable detailed output from the ad listener.

Use Default GDPR Behavior

Please note, if you want the Appodeal plugin to do all the work regarding GDPR for you, please locate this line of code in the helper and comment it out:

```
--lparams.hasUserConsent = ssk.persist.get( "settings.json",
"has_gdpr_consent" )
```

build.settings - Ad Settings

Lastly, you should edit the build.settings file based on your choice of provider.

AppLovin

Uncomment this plugin:

```
["plugin.applovin"] = { publisherId = "com.coronalabs" },
```

Uncomment this iOS line:

```
NSAppTransportSecurity = { NSAllowsArbitraryLoads = true },
```

Uncomment these Android lines:

```
"android.permission.INTERNET",
"android.permission.ACCESS_NETWORK_STATE",
"android.permission.WRITE_EXTERNAL_STORAGE",
```

Appodeal

Uncomment this plugin:

```
['plugin.appodeal.beta.base'] = { publisherId = 'com.coronalabs' },
```

Optionally, uncomment these additional plugins:

```
----[[
['plugin.appodeal.beta.AdColony'] = { publisherId = 'com.coronalabs'
},
['plugin.appodeal.beta.AmazonAds'] = { publisherId = 'com.coronalabs'
},
...

```

Uncomment this iOS line:

```
NSAppTransportSecurity = { NSAllowsArbitraryLoads = true },
```


Uncomment these Android lines:

```
"android.permission.INTERNET",  
"android.permission.ACCESS_NETWORK_STATE",  
"android.permission.WRITE_EXTERNAL_STORAGE",
```

Optionally, uncomment these Android lines:

```
"android.permission.GET_ACCOUNTS",  
"android.permission.ACCESS_COARSE_LOCATION",  
"android.permission.ACCESS_FINE_LOCATION",
```

In-App-Purchases

Make the following changes to enable and configure In-App-Purchases.

scripts/common.lua - IAP Settings

Enable Feature

```
common.extras.iap_enabled = true
```

Set Test Mode

While testing in the simulator and on device it is helpful to set this variable to `true`, but remember to set it to `false` when you release

```
common.extras.iap_test_mode = true
```

Disable Inventory Loading

During testing, you may want to disable inventory loading. With this feature set to `true` every restart acts as if none of the IAP items was purchased yet.

Note: If you use this feature and the framework behaves as if an item was purchased you may need to delete the contents of the DocumentsDirectory to reset the state of the app.

```
common.extras.iap_do_not_load_inventory = true
```

Edit IAP Helper

While the framework uses IAP Badger to do the heavy lifting, it includes module that adds a layer of code on top of the badger. This module can be found in the file:

scripts/ads/easyIAP.lua

Edit the IAP helper script and locate this marker:

```
-- =====
-- BEGIN EDITING HERE * BEGIN EDITING HERE * BEGIN EDITING HERE
-- =====
```

, then read the comments in the code to configure the IAPhelper. The following is a summary of the changes you will make in this section:

- **salt**- Type in some random string to replace the default value. This is used to seed the encrypted IAP purchase tracking file on the device.
- **androidID** - If you are publishing to Android, go to your Google Play app page and add a 'NoAds' IAP item. Then grab the ID and place it here.
- **iosID** - If you are publishing to Android, go to your iTunes Connect app page and add a 'NoAds' IAP item. Then grab the ID and place it here.

You may dig into the helper further, but you don't need to.

build.settings - IAP Settings

Next, you should edit the build.settings file.

If you are publishing to Google, then uncomment this plugin:

```
["plugin.google.iap.v3"] = { publisherId = "com.coronalabs",  
supportedPlatforms = { android = true, }, },
```

If you are publishing to Apple, uncomment this iOS line:

```
NSAppTransportSecurity = { NSAllowsArbitraryLoads = true },
```

If you are publishing to Google Play, then uncomment these Android lines:

```
"android.permission.INTERNET",  
"com.android.vending.BILLING",
```

config.lua - IAP Settings

Lastly, if you are publishing on the Google Play store, you should acquire a license key and place it in the **config.lua** file.

If you are publishing to Google, then uncomment this plugin:

```
license = {  
    google = {  
        key = "YOUR_KEY",  
    },  
},
```

Replace `YOUR_KEY` with the (long) license key string.

Tip: If you do not know how to get a license key, please see the chapter 'Getting A Google Play IAP License Key' below.

Rating

Make the following changes to enable and configure rating.

scripts/common.lua - Ratings Settings

Enable Feature

```
common.extras.rating_enabled = true
```

Edit Rating Helper

Next, edit the script: **scripts/extras/rating.lua**

Locate the beginning of the module

```
-- =====
-- Module Begins
-- =====
local public = {}
```

, then examine the code closely. Tweak it as desired:

- **packageID** - If you are publishing to Android, get your App's packageID and replace the generic one provided in the code.
 - **Tip:** If you don't know how to do this, please see 'Package ID, Bundle ID, and Apple ID ...' below.
- **appleID**- If you are publishing to iOS, get your App's numeric appleID and replace the generic one provided in the code.
 - **Tip:** If you don't know how to do this, please see 'Package ID, Bundle ID, and Apple ID ...' below.
- **public.rate()** - This function does all of the rating work. You should NOT need to edit it.

build.settings - Rating Settings

Lastly, you should edit the build.settings file to enable rating.

If you are publishing to the Apple store, uncomment this plugin:

```
["plugin.reviewPopUp"] = { publisherId = "tech.scotth", },
```

Tip: Be sure you have activated this **FREE** plugin or your code will not build:

<https://marketplace.coronalabs.com/corona-plugins/review-popup>

If you are publishing to the Apple store, uncomment this iOS line:

```
NSAppTransportSecurity = { NSAllowsArbitraryLoads = true },
```

If you are publishing to Google Play, uncomment this Android line:

```
"android.permission.INTERNET",
```

Sharing

Make the following changes to enable and configure sharing.

scripts/common.lua - Sharing Settings

Enable Feature

```
common.extras.sharing_enabled = true
```

Edit Sharing Helper

Next, edit the script: **scripts/extras/sharing.lua**

Locate the beginning of the module:

```
-- =====
-- Module Begins
-- =====
```

, then examine the code closely. Tweak it as desired:

- **appleID** - Get your App's numeric appleID and replace the generic one provided in the code.
 - **Tip:** If you don't know how to do this, please see 'Package ID, Bundle ID, and Apple ID ...' below.
- **public.showActivityDialog()** - This function does all of the sharing work. It comes configured with a generic message, but you may want to tweak it a little to make it fit your game better.

build.settings - Sharing Settings

If you are publishing on the Apple store, uncomment this plugin:

```
["CoronaProvider.native.popup.activity"] = { publisherId =
"com.coronalabs" },
```

If you are publishing on the Apple store, uncomment this iOS line:

```
NSAppTransportSecurity = { NSAllowsArbitraryLoads = true },
```

Twitter

Make the following changes to enable and configure twitter.

scripts/common.lua - Twitter Settings

Enable Feature

```
common.extras.twitter_enabled = true
```

Edit Twitter Helper

Next, edit the script: **scripts/extras/twitter.lua**

Locate the beginning of the module:

```
-- =====
-- Module Begins
-- =====
```

, then examine the code closely. Tweak it as desired:

- **apiKey / apiSecret** - Get your Twitter App's API Key and Secret and replace the generic ones provided.
 - **Tip:** If you don't know how to do this, please see 'Creating A Twitter App' below.
- **hashtag** - Replace this with a cool hash tag for your game.
- **androidURL** - If you are publishing on Google Play, get the package ID for your game and update this address string.
 - **Tip:** If you don't know how to do this, please see 'Package ID, Bundle ID, and Apple ID ...' below.
- **iosURL** - If you are publishing on the Apple store, get the appleID for your game and update this address string.
 - **Tip:** If you don't know how to do this, please see 'Package ID, Bundle ID, and Apple ID ...' below.
- **public.showActivityDialog()** - This function does all of the tweeting work. It comes configured with a generic message, but you may want to tweak it a little to make it fit your game better.

build.settings - Twitter Settings

Lastly, you should edit the build.settings file.

Uncomment this plugin:

```
["plugin.twitter"] = { publisherId = "com.jasonschroeder", },
```

Tip: Be sure you have purchased and activated this **PAID** plugin or your code will not build:

<https://marketplace.coronalabs.com/corona-plugins/twitter>

If you are building for the Apple store, uncomment this iOS line:

```
NSAppTransportSecurity = { NSAllowsArbitraryLoads = true },
```

If you are building for Google Play, uncomment these Android lines:

```
"android.permission.INTERNET",  
"android.permission.ACCESS_NETWORK_STATE",
```

Modifying Extras' Button Positions

When you enable the various extra features you will start to see buttons on the home scene.

If you do not like the order of these buttons, you can do the following:

1. Edit the file **scenes/play.lua** and locate statements like this:

```
buttons[#buttons+1] = ...
```

2. Now, just re-order these statements to suit your preferences.

Note: Some of the button creation statements will be in a if-then-else statement.

```
if( ... ) then
    buttons[#buttons+1] = ...
end
```

In such cases, be sure to move the entire statement.

Getting A Google Play IAP License Key

In order to get a license key, you must have set up your Google Play app store page and you must upload binary.

Note: This section assumes you know how to build a release copy of your game for Google Play. If you do not, then take a moment to read 'Building Releases' below.

Note 2: This section also assume you have not set up your Google Play app description yet. If you have, you can skip over some of the following steps.

Google requires you to upload a binary before they will provide you with a key. The easiest way to do this it to upload an alpha or beta build of your app.

Just follow these steps:

Build Game

Build your game using your production certificate and the correct package ID.

Android Build Setup

Application Name:

Version Code:

Version Name:

Package:

A unique Java-style package identifier for your app
(e.g. com.acme.games.myfarmgame)

Project Path:

Target App Store:

Keystore:

Key Alias:

Save to Folder:

☐ Create Live Build

The screenshot displays the Google Play Console interface. On the left is a sidebar with navigation options: 'All applications' (selected), 'Game services', 'Order management', 'Download reports', 'Alerts', and 'Settings'. The main area is titled 'All applications' and includes a search bar, notification bell, and help icon. Below the title, there's a '+1 MORE' link and a 'Filter' dropdown. A 'CREATE APPLICATION' button is located in the top right. A table lists applications with columns: 'App name', 'Active / Total installs', 'Avg. rating / Total #', 'Last update', and 'Status'. The first row shows a blurred application name, a status of 'Active', a rating of 4.5, and a 'Last update' date. The second row is partially visible below it.

Create application

Default language *
English (United States) – en-US ▼

Title *
Lava Is Hot!

CANCEL

CREATE

All applications
Product details
ENGLISH (UNITED STATES) en-US
Manage translations ▾

- Dashboard
- App releases
- Android Instant Apps
- Artifact library
- Device catalog
- App signing
- Store listing
- Content rating
- Pricing & distribution
- In-app products
- Translation service
- Services & APIs
- Optimization tips

Facts marked with * need to be filled before publishing.

Title *	Lava to Hot!	12/18
English (United States) - en-US		

Short description *

English (United States) - en-US	>Lorem ipsum Lorem ipsum Lorem ipsum Lorem ipsum	46/10

Full description *

English (United States) - en-US	Lorem ipsum Lorem ipsum	244/1020

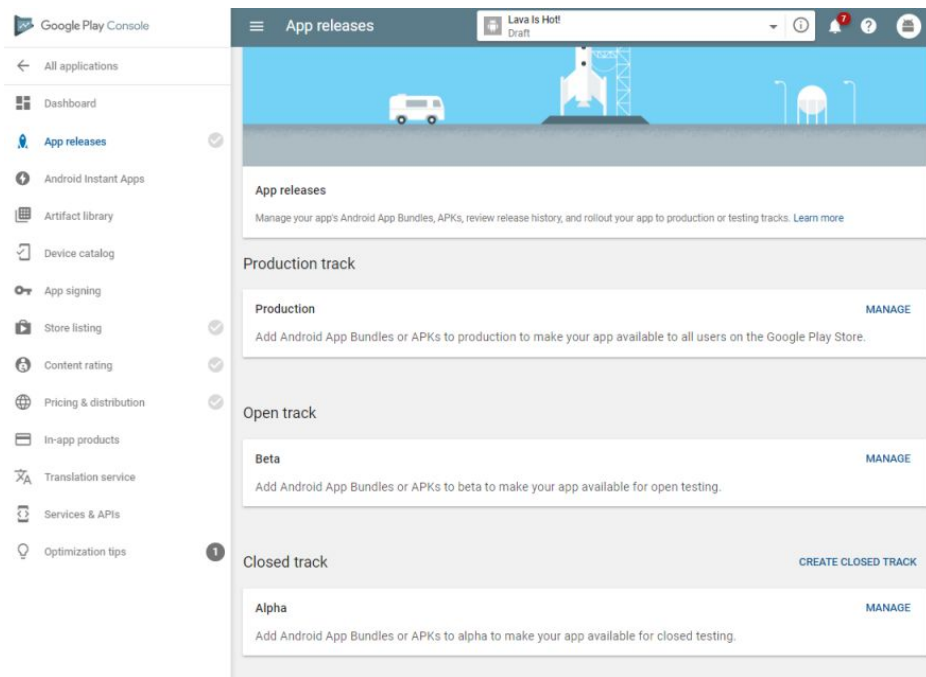
Please check out our [Metadata policy](#) to avoid some common violations related to app metadata. Also, please make sure to review all the other [program policies](#) before you submit App for review.

If your app or store listing is [eligible for advance notice](#) to the Google Play App Review team, contact us prior to publishing.

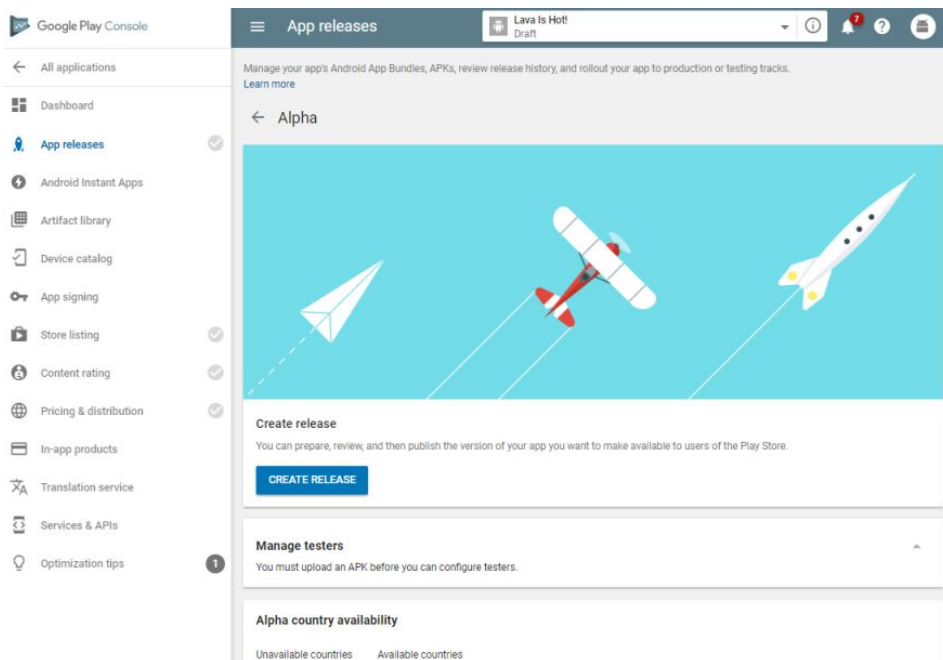
Graphic Assets

[SAVE DATA](#)

On the left-side of the page, click 'App release' then choose the 'manage' option for either Beta or Alpha:



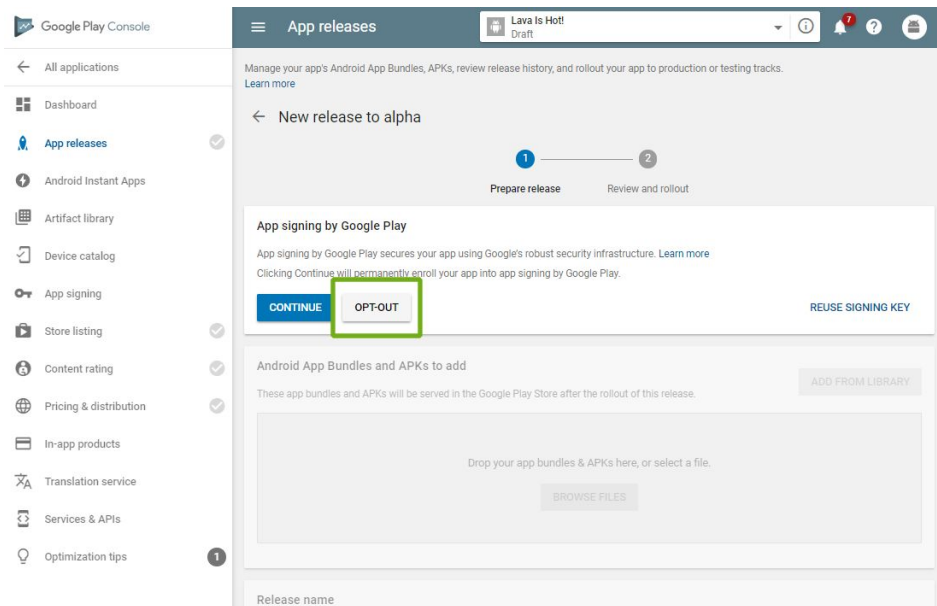
Next, click 'Create Release'



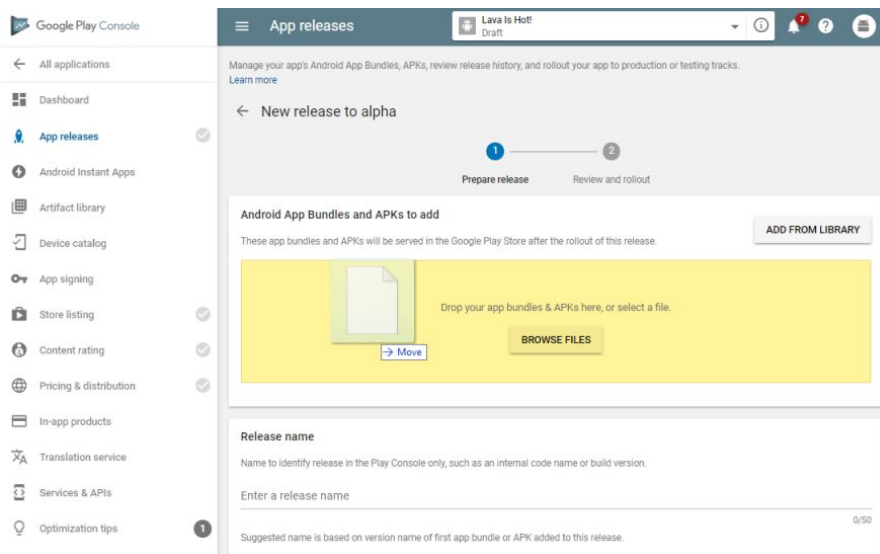
IMPORTANT! IMPORTANT! IMPORTANT! IMPORTANT! IMPORTANT!

Now, be absolutely sure to click the 'Opt-Out' button and to confirm you are opting out.

You want to use your own signing certificate not one supplied by Google.

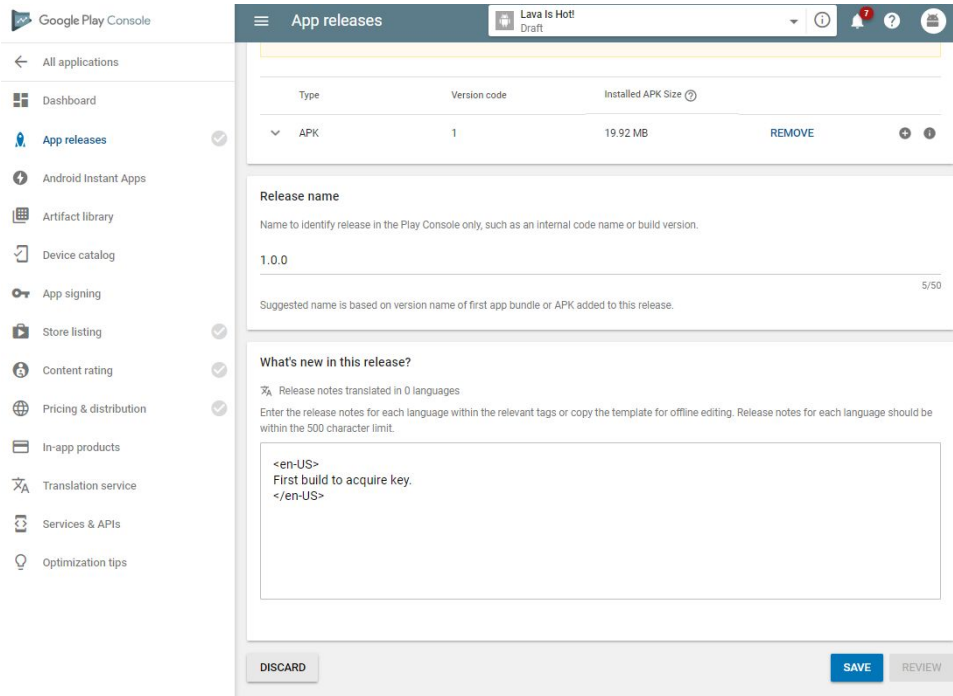


Now, locate your APK and drag-and-drop it to the space shown in yellow below:

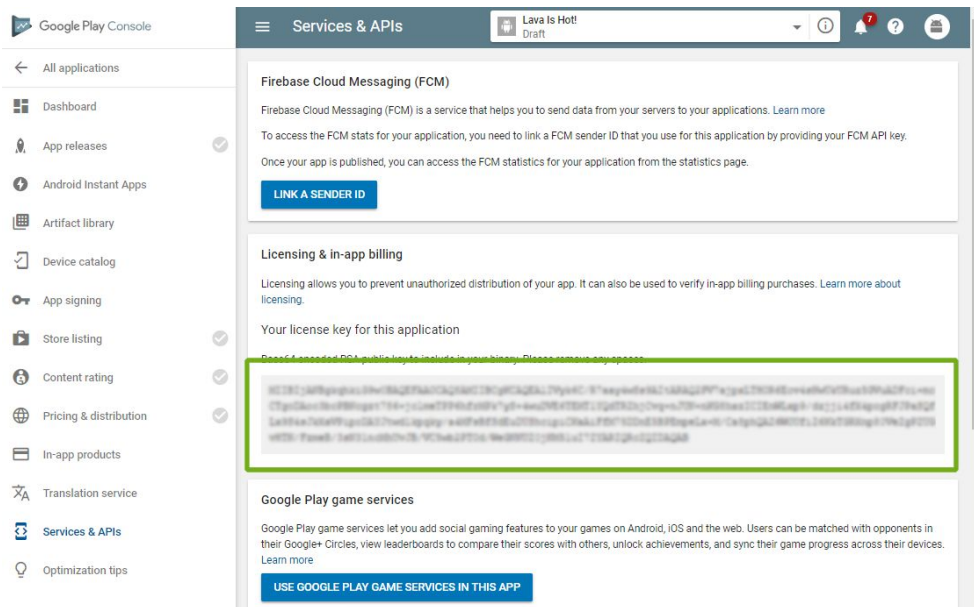


You can also click the 'Browse Files' button to manually upload it if you like.

When the file is done uploading, scroll to the bottom of the page, (optionally) add a release note, then click 'Save':



Finally, you can click on "Services & APIs" on the left-side of the page and you will see an page like the following. Grab the big string outlined in the green box in my screenshot:



As the last set of steps, open config.lua, uncomment the license section, and replace "YOUR_KEY" with the long string you just copied.

This:

```
-- Google Play In-App-Purchase
-- https://docs.coronalabs.com/plugin/google-iap-v3/index.html
--[[
license = {
    google = {
        key = "YOUR_KEY",
    },
},
--]]
```

Becomes something like this:

```
-- Google Play In-App-Purchase
-- https://docs.coronalabs.com/plugin/google-iap-v3/index.html
license = {
    google = {
        key =
"ABNADA$TgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAlIVyk6C/R7aay4wSs9AItARAQ2PV7ajpxLT
HOR6Eov4sNwUkURuz5GVuADFCi+mcCTgoDAoc3bcPBNcgzt786+jclmeTPP6hfrMPk7yS+4wuDVE6TE
MTlYQdTRZhjCvq+nJUN+nKG8hxzICIEEnWLxp9/dzjji4fX4pogRFJPaSQfLx9S4sJkXaVPipoDAYJtw
dlkpqky/a4MFsBfSdEuDUSHcippiCKaAiFfM75DDnE5BPEmpeLx+M/Ca8phQAZ6WOUfiZ6KkTGRXnp80
VeZgPZUGvHTN/FzmeB/3sN31ncMhUvJB/VC5wb2PTOd/WeGNNUZ0jNM5luI7ZYAADFOJWESGASDH",
    },
},
```


Creating A Twitter App

This framework uses the PAID Twitter plugin by Jason Schroeder to allow users to Tweet about your game.

Buy and enable plugin here: <https://marketplace.coronalabs.com/corona-plugins/twitter>

Plugin docs: <http://www.jasonschroeder.com/2015/07/15/twitter-plugin-for-corona-sdk/>

Jason has provided docs for this here:

<http://www.jasonschroeder.com/2015/07/15/twitter-plugin-for-corona-sdk/#registration>

Important!: Your app may not be available immediately after creation. Give it an hour or two and it should be ready for use.


Twitter App Registration

For completeness, here is a summary of how to set up a Twitter App:

1. Go to <https://apps.twitter.com> and click the “Create New App” button.
2. Fill in all the fields with your app’s name, description, website and a callback URL. Note that while it doesn’t matter what your callback URL is, you **MUST** include one for the plugin to work correctly, even though Twitter does not consider it a required field.
3. On your new app’s developer page, click the “Keys and Access Tokens” button. The API Key and API Secret will be shown, for easy copying and pasting into your Corona app.

When you are done, go to the **Details** tab and compare your settings to these:

Details
Settings
Keys and Access Tokens
Permissions



A casual jumper. How high and fast can you jump before you burn baby burn in the hot hot Lava?
<https://www.roaminggamer.com/>

Organization

Information about the organization or company associated with your application. This information is optional.

Organization	None
Organization website	None

Application Settings

Your application's Consumer Key and Secret are used to [authenticate](#) requests to the Twitter Platform.

Access level	Read and write (modify app permissions)
Consumer Key (API Key)	<div></div> (manage keys and access tokens)
Callback URL	<div></div>
Callback URL Locked	No
Sign in with Twitter	Yes
App-only authentication	https://api.twitter.com/oauth2/token
Request token URL	https://api.twitter.com/oauth/request_token
Authorize URL	https://api.twitter.com/oauth/authorize
Access token URL	https://api.twitter.com/oauth/access_token

If you see any differences, go to the correct tab, update the settings, and save them.

Package ID, Bundle ID, and Apple ID ...

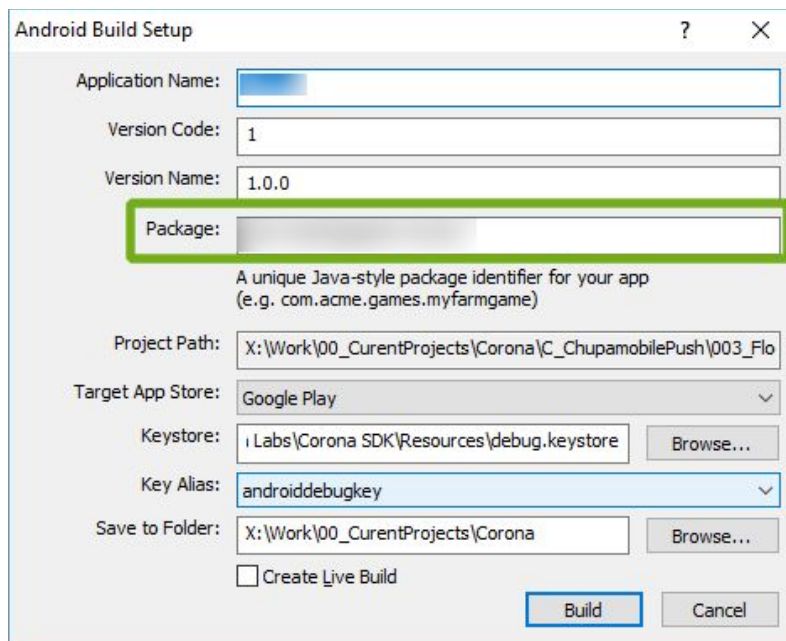
In the prior pages, the instructions may call for any of the following:

- **Package ID** - This is the app ID of your Google Play game.
- **Bundle ID** - This is the app ID of your Apple game.
- **Apple ID** - This is a numeric ID used by the Apple store to look up your game page.

Getting The Package ID

You specify the package ID when building your app and it is typically something like:

com.yourcompanyname.gamename



Android Build Setup

Application Name:

Version Code:

Version Name:

Package:

A unique Java-style package identifier for your app
(e.g. com.acme.games.myfarmgame)

Project Path:

Target App Store:

Keystore:

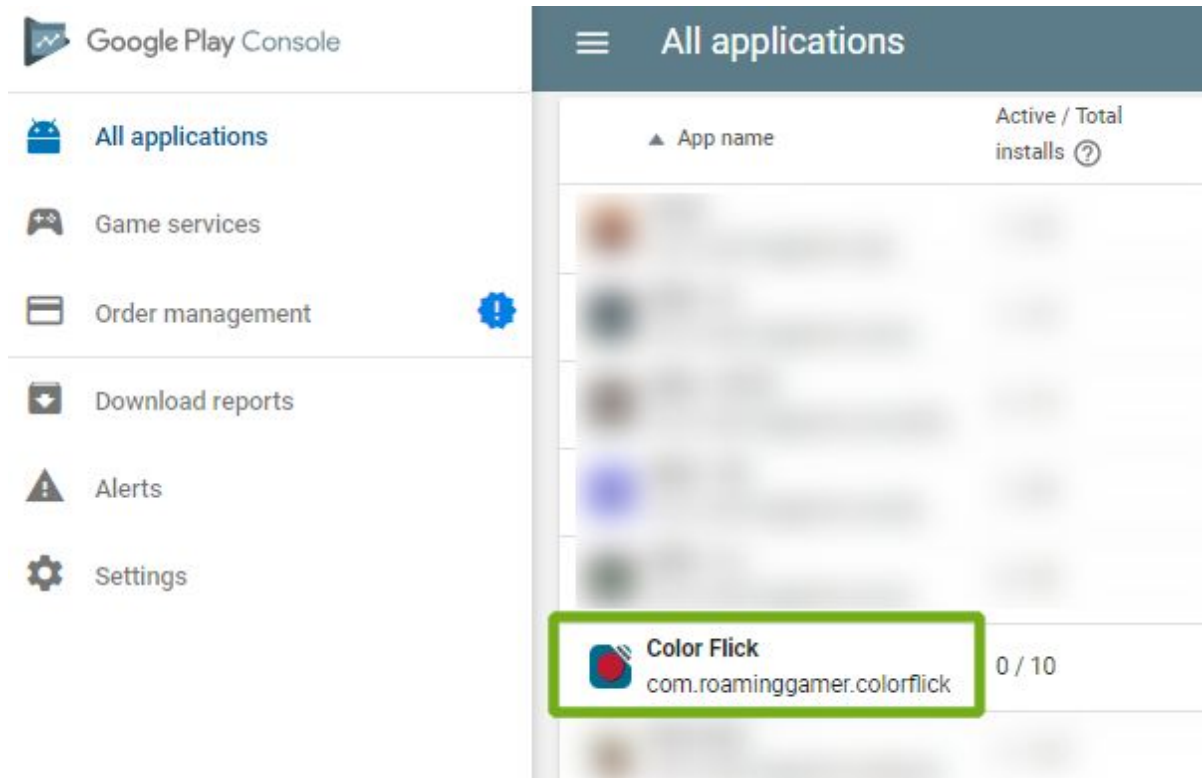
Key Alias:

Save to Folder:

☐ Create Live Build

Later, if you forget, just log into your Google Play console, then find your game and it will be listed below the game name.

For example, my game 'Color Flick' has the package ID: com.roaminggamer.colorflick



Getting The Bundle ID and Apple ID

You specify the bundle ID when you create your game ID on the Apple certificates page. It usually takes the form:

com.yourcompanyname.gamename

Later, you can find this by logging into your iTunes connect account and looking on the 'App Store' page for your app. The numeric Apple ID will be found on the same page.

The screenshot shows the iTunes Connect 'App Information' page. The left sidebar contains navigation links: 'App Store' (selected), 'Features', 'TestFlight', and 'Activity'. Below these are 'APP STORE INFORMATION' links: 'App Information' (selected), 'Pricing and Availability', and 'iOS APP' (showing '1.0 Ready for Sale'). A 'VERSION OR PLATFORM' link is also present. The main content area is titled 'App Information' and includes a 'Save' button. It contains a text input field with 'Optional' and a character count of '30'. Below this is the 'General Information' section, which includes 'Bundle ID' (highlighted with a green box), 'SKU' (16001), and 'Apple ID' (highlighted with a green box). To the right of these are dropdown menus for 'Primary Language' (English (U.S.)), 'Category' (Games), 'Board', and 'Subcategory (optional)'. At the bottom right, there are links for 'License Agreement' (with an 'Edit' link), 'Apple's Standard License Agreement', 'Rating' (Ages 4+), and 'Additional Ratings'.

Building Releases

Android

<https://docs.coronalabs.com/guide/distribution/androidBuild/index.html>

iOS

<https://docs.coronalabs.com/guide/distribution/iOSBuild/index.html>

FAQ

I made a change and it didn't work.

Sometimes during testing and development you will change some setting in scripts/common.lua then re-run and it will seem as if the change did not take effect. The reason for this is that some settings are stored in the persistent storage for the game.

To ensure your changes take effect, do the following:

In Simulator

Using the simulator file menu, open the "Project Sandbox" folder, then open the "Documents" directory.

You will see one or two files in this folder.

Delete them, then restart the simulator.

On Device

If you're testing on the device, simply delete the application (not replace it) then re-install it. Installing over and old copy keeps the persistent storage, but deleting the app removes it.

I set up a Twitter App but I can't tweet. Why not?

A number of things may be wrong, including:

- **Your settings may be wrong.** Please see 'Creating A Twitter App' above and compare your settings to mine.
- **Your IDs may be wrong.** Please be sure you supplied your API Key and Secret as described in 'Edit Twitter Helper' above.
- **You may need to wait.** It take some time for the App to be ready for use. Give it an hour or two and it should definitely be ready.

Why Doesn't the IAP Dialog Pop up when I press the 'NoAds' or 'Restore' button on my device?

Android

This may happen on Android if you have:

- **Failed to supply a license key.** Please see 'Getting A Google Play IAP License Key' above.
- **Failed to build with a production certificate.** You must build your game with a production certificate when testing IAP.
- **Incorrectly set `common.extras.iap_test_mode` to `false`.** If you want to use the IAP Badger testing features in the simulator and on device, you must set this flag to `true`.

ios

This may happen on iOS if you have:

- **Failed to build with a production certificate.** You must build your game with a production certificate when testing IAP.

Why is my production app showing test ads?

In all likelihood you left the `testMode` flag set to `true`.

Please double check your settings by visiting the 'Edit Provider Helper' section above for either Appodeal or AppLovin, depending on the provider you are using.

Important Steps - Before You Publish Your Game

- Ensure you are using valid IDs for all of the extra features you have enabled.
- Be sure you are building a production copy and not using a debug certificate.
- If you are using Ads, be sure to edit the appropriate ad helper and set the `testMode` variable to `false`.
- If you are using IAP, be sure to set the `common.iap_test_mode` variable to `false`.

Credits

All of the assets and libraries in this template are free and covered by very liberal licenses:

Images

Ansimuz - <http://pixelgameart.org/web/>

Grotto Escape (CC0): <http://pixelgameart.org/web/portfolio/grotto-escape-game-art-pack/>

Grotto Escape II (CC3): <http://pixelgameart.org/web/portfolio/grotto-escape-ii-art-pack/>

Note: Ansimuz has some really beautiful paid and free art packs that you should definitely check out.

Kenney - <http://kenney.nl/>

Various (CC0):

- <https://kenney.itch.io/kenney-donation>
- <https://kenney.itch.io/kenney-game-assets-2>
- <https://kenney.itch.io/kenney-game-assets-3>

Note: I can't say enough positive things about Kenney's art packs. While you can get them for free, you should go out and paid the low-low price of \$10 each.

pzuH - <http://kenney.nl/>

This template uses two different free interface packs by this author. There are many more free and paid packs to be found pzuH and you should check them out.

- <https://opengameart.org/users/pzuh>
 - Casual: <https://opengameart.org/content/casual-game-button-pack>
 - Medieval: <https://opengameart.org/content/medieval-game-button-pack>
- Other sites that have pzuH assets:
 - Corona Marketplace: <https://marketplace.coronalabs.com/vendor/pzuh>
 - OpenGameArt.Org: <https://opengameart.org/users/pzuh>
 - Game Art 2D: <https://www.gameart2d.com>

Roaming Gamer - Flat & Prototype

Lastly, there are two basic sets of art made by 'Roaming Gamer':

- Flat - A flat art pack typical of today's casual styled games.
- Prototype - A prototyping set showing safe spaces and bleed areas.

Note: Both of these art sets use Kenney's flat buttons.

Fonts

All fonts used in this template are free and downloaded from: <https://fonts.google.com/>

Sounds

All sounds are free and made using bfxr and sfxr by Roaming Gamer, LLC.

- <https://www.bfxr.net/>
- http://www.drpetter.se/project_sfxr.html

SSK2

This template uses the completely FREE "Super Starter Kit 2" by Roaming Gamer, LLC.

Super Starter Kit 2 (aka SSK2) is a collection of libraries and utilities designed to take your Corona SDK development experience to a whole new level of efficiency and speed.

This library is entirely free to use in any game or app. You can learn more about it here:

<https://roaminggamer.github.io/RGDocs/pages/SSK2/>

Index

Product Overview	1
Features Summary	1
License	2
Features	2
Getting Started With Corona	4
First steps	4
Testing The Installation	4
Run This Template	5
Modifying The Game	6
Selecting Different Themes	6
Modifying Game Art	6
Modifying Button Art and Code	7
Modifying Game Code	8
Modifying composer.* Scenes	9
The About Scene	10
Modifying The Framework	11
Ads	12
scripts/common.lua - Ad Settings	12
Enable Feature	12
Select Provider	12
Enable Interstitials	12
Set Interstitial Frequency	12
Choose Whether To Request GDPR Permission	12
Edit Provider Helper	13
AppLovin Helper	13
Use Default GDPR Behavior	13
Appodeal Helper	14
Use Default GDPR Behavior	14
build.settings - Ad Settings	15
AppLovin	15
Appodeal	15
In-App-Purchases	17

scripts/common.lua - IAP Settings	17
Enable Feature	17
Set Test Mode	17
While testing in the simulator and on device it is helpful to set this variable to true, but remember to set it to false when you release	17
Disable Inventory Loading	17
Edit IAP Helper	18
build.settings - IAP Settings	19
config.lua - IAP Settings	19
Rating	20
scripts/common.lua - Ratings Settings	20
Enable Feature	20
Edit Rating Helper	20
build.settings - Rating Settings	21
Sharing	22
scripts/common.lua - Sharing Settings	22
Enable Feature	22
Edit Sharing Helper	22
build.settings - Sharing Settings	22
Twitter	23
scripts/common.lua - Twitter Settings	23
Enable Feature	23
Edit Twitter Helper	23
build.settings - Twitter Settings	24
Modifying Extras' Button Positions	25
Getting A Google Play IAP License Key	26
Build Game	26
Create App Description On Google Play	27
Creating A Twitter App	32
Twitter App Registration	32
Package ID, Bundle ID, and Apple ID ...	34
Getting The Package ID	34
Getting The Bundle ID and Apple ID	36
Building Releases	37
Android	37
iOS	37

FAQ	37
I made a change and it didn't work.	38
In Simulator	38
On Device	38
I set up a Twitter App but I can't tweet. Why not?	38
Why Doesn't the IAP Dialog Pop up when I press the 'NoAds' or 'Restore' button on my device?	38
Android	38
iOS	39
Why is my production app showing test ads?	40
Important Steps - Before You Publish Your Game	41
Credits	42
Images	42
Ansimuz - http://pixelgameart.org/web/	42
Kenney - http://kenney.nl/	42
pzuH- http://kenney.nl/	42
Roaming Gamer - Flat & Prototype	43
Fonts	43
Sounds	43
SSK2	43
Index	44