

Google Analytics

Google Analytics for Firebase provides free, unlimited reporting on up to 500 distinct events. The SDK automatically captures certain key events and user properties, and you can define your own custom events to measure the things that uniquely matter to your business.

Analytics surfaces data about user behavior in your iOS and Android and Web apps, enabling you to make better decisions about your app or game and marketing optimization. View crash data, notification effectiveness, deep-link performance, in-app purchase data, and more.

Check the [official page](#) for more information.

Setup

Before starting to use any Firebase extensions, you are required to follow some initial configuration steps. However if you've already done these for any of the other modules you can skip this configuration section and go straight to using the API functions.

- [Create Project](#)
- [Platform Setup](#)

Functions

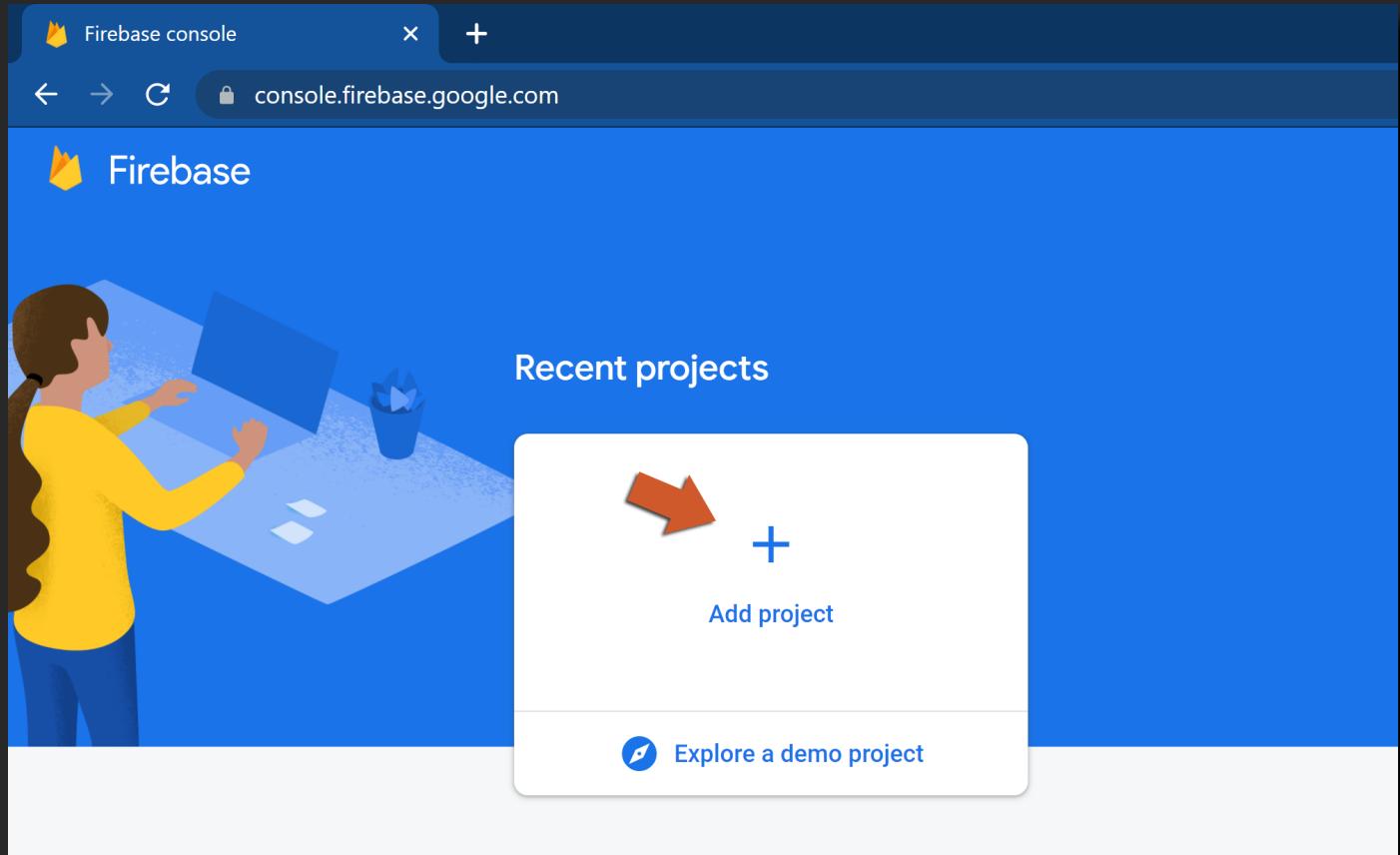
The following functions are given for working with Analytics extension:

- [FirebaseAnalytics_LogEvent](#)

Create Project

Before working with any Firebase functions, you must set up your Firebase project:

1. Go to the [Firebase Console](#) web site.
2. Click on **Add Project** to create your new project.



3. Enter a name for your project and click on the **Continue** button.

- ✗ Create a project (Step 1 of 3)

Let's start with a name for
your project?

Enter your project name

my-awesome-project-id

Select parent resource

Continue

4. On the next page, make sure that **Enable Google Analytics for this project** is enabled and then click the **Continue** button:

- ✗ Create a project (Step 2 of 3)

Google Analytics

for your Firebase project

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, Predictions, and Cloud Functions.

Google Analytics enables:

💡 A/B testing ?

🌀 Crash-free users ?

🌐 User segmentation & targeting across
Firebase products

👉 Event-based Cloud Functions triggers ?

📈 Predicting user behavior ?

📊 Free unlimited reporting ?



Enable Google Analytics for this project
Recommended

Previous



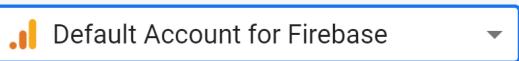
Continue

5. Select your account and click the **Create project** button:

X Create a project (Step 3 of 3)

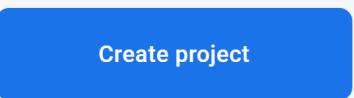
Configure Google Analytics

Choose or create a Google Analytics account [?](#)

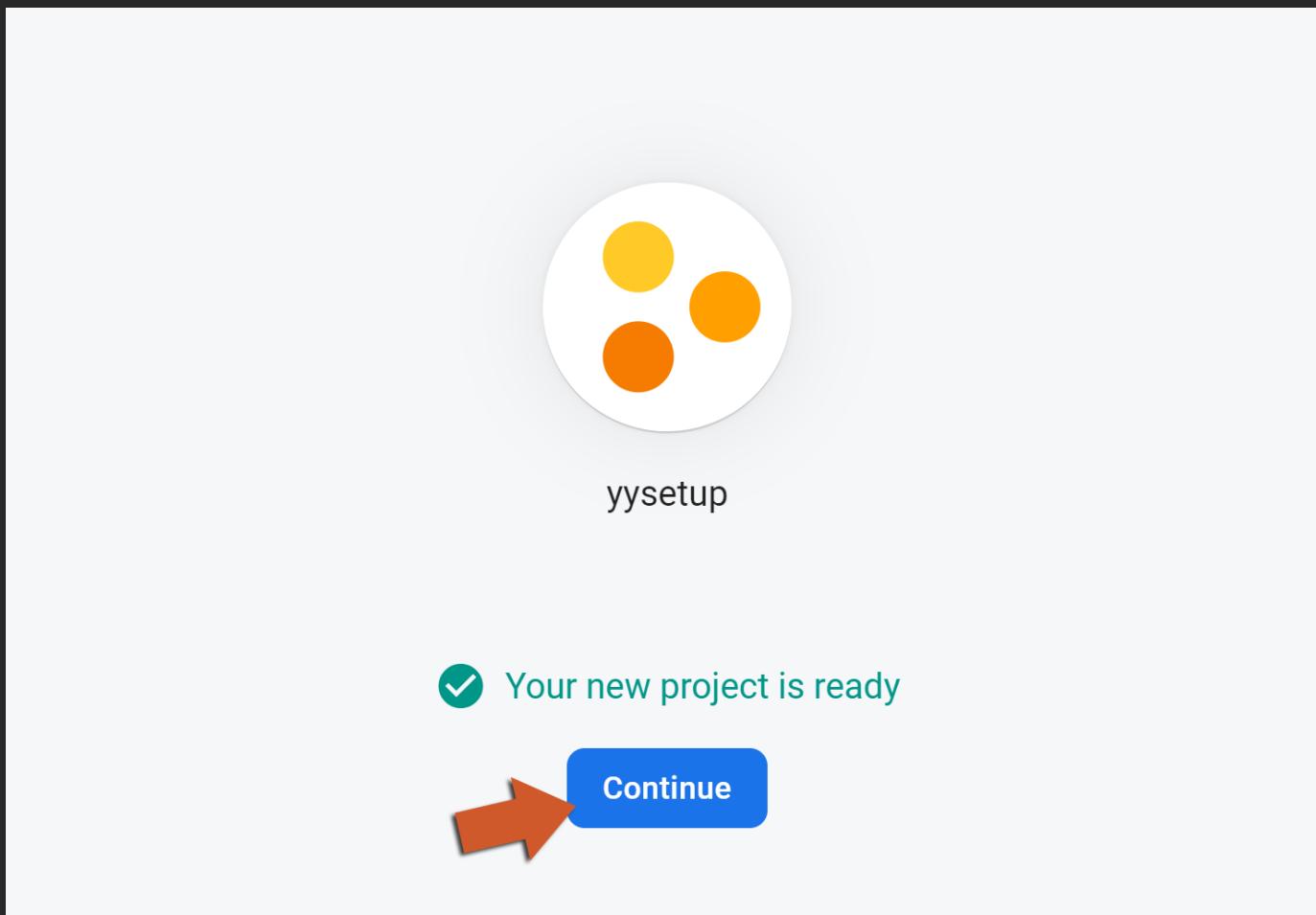
 

Automatically create a new property in this account 

Upon project creation, a new Google Analytics property will be created in your chosen Google Analytics account and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. [Learn more.](#)

[Previous](#)  

6. Wait a moment until your project is created; after a few moments you should see the page shown below:



7. You will now be taken to your new project's home page:

The screenshot shows the Firebase console interface. On the left, a dark sidebar lists various services: Project Overview, Authentication, Firestore Database, Realtime Database, Storage, Hosting, Functions, Machine Learning, Crashlytics, Performance, Test Lab, App Distribution, and Extensions. A 'Spark' plan is selected, indicated by a green border around the 'Free \$0/month' button. At the top right, there are links for 'Go to docs', a bell icon, and a user profile with the letter 'J'. The main content area is titled 'ysetup' and features a blue background with white text. It says 'Get started by adding Firebase to your app' and includes icons for iOS, Android, web development, and a smartphone. Below this is a callout box with the text 'Store and sync app data in milliseconds' and two small images illustrating data storage.

8. Continue your adventure with the Firebase extensions provided for GameMaker!

Platform Setup

Firebase Analytics implementation uses SDK dependencies and therefore is only available on the **Android**, **iOS** and **Web** targets. In this section we will cover the required setup necessary to start using the Analytics extension on your game.

Select your target platform below and follow the simple steps to get your project up and running (you only need follow this topics once per project):

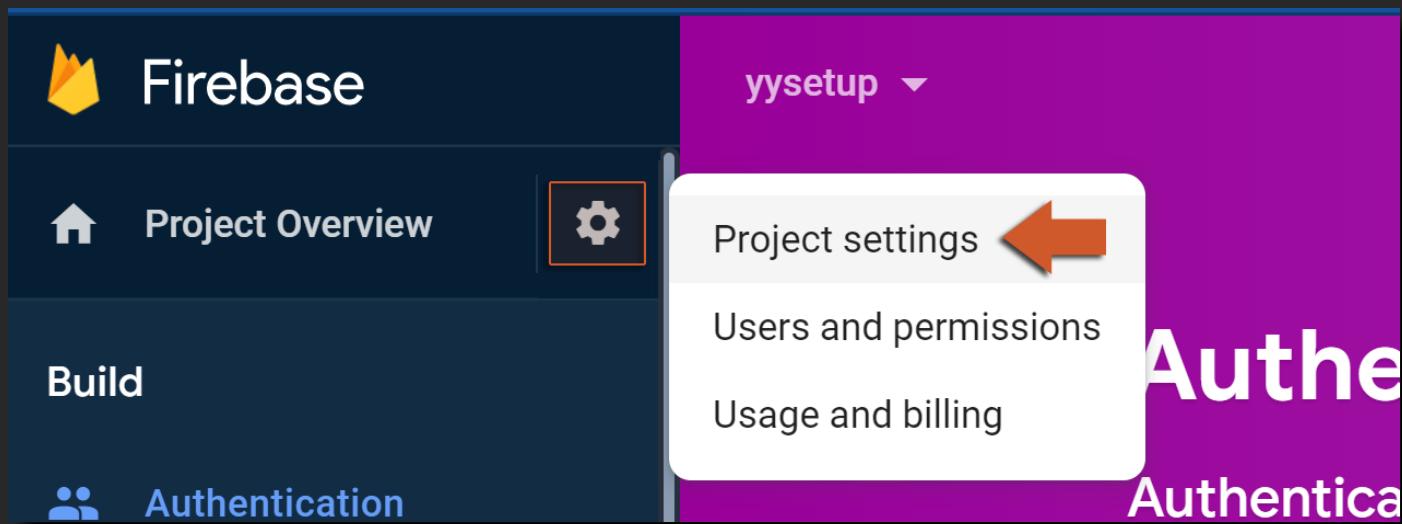
- [Android Setup](#)
- [iOS Setup](#)
- [Web Setup](#)

Android Setup

This setup is necessary for all the Firebase modules using Android and needs to be done once per project, and basically involves importing the `google-services.json` file into your project.

IMPORTANT Please refer to [this Helpdesk article](#) for instructions on setting up an Android project.

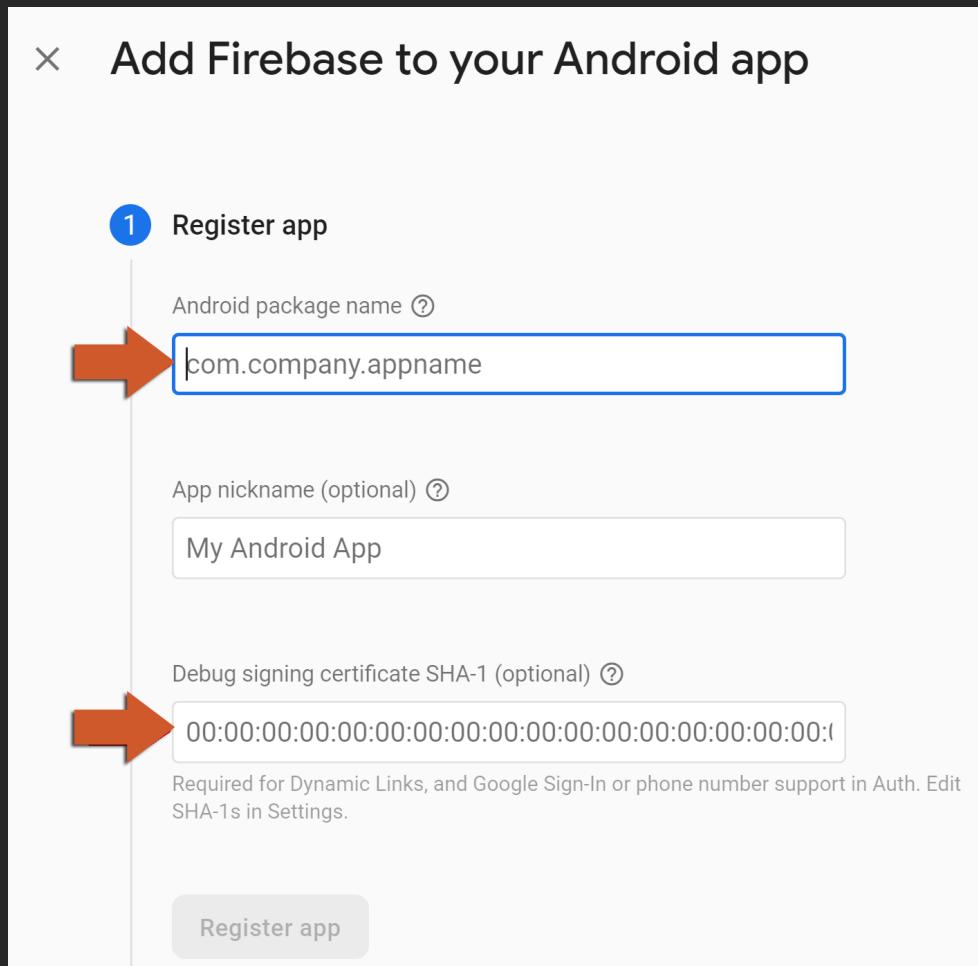
1. Click the **Settings** icon (next to Project Overview) and select **Project settings**:



2. Now go to the **Your apps** section and click on the **Android** button:

The screenshot shows the 'Project settings' page under the 'ysetup' dropdown. The 'Your apps' section is visible, showing a message: 'There are no apps in your project. Select a platform to get started.' Below this message are four circular icons representing different platforms: iOS, TV, Android (highlighted with a red box), and Web. The 'Your apps' button is also highlighted with a red box.

3. On this screen you need enter your **Package name** (required), **App nickname** (optional) and **Debug signing certificate SHA-1** (required if you are using Firebase Authentication).



You can get your package name from the [Android Game Options](#), and your **Debug signing certificate SHA-1** from the [Android Preferences](#) (under Keystore):



4. Click on **Download google-services.json** (make sure to save this file as we will need it in subsequent steps).

× Add Firebase to your Android app

✓ Register app

Android package name: com.yoyogames.YoyoPlayServices2, App nickname: yysetup_android

2 Download config file

Instructions for Android Studio below | [Unity](#) [C++](#)

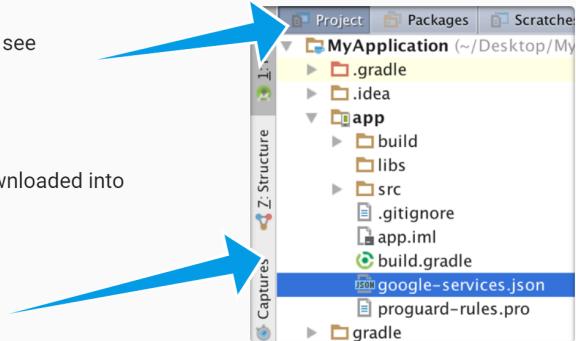
 [Download google-services.json](#)

Switch to the Project view in Android Studio to see your project root directory.

Move the google-services.json file you just downloaded into your Android app module root directory.



google-services.json



[Next](#)

5. Ignore this screen, as this is already done in the extension.

× Add Firebase to your Android app

✓ Register app

Android package name: com.yoyogames.YoyoPlayServices2, App nickname: yysetup_android

Download config file

3 Add Firebase SDK

Instructions for Gradle | [Unity](#) [C++](#)

The Google services plugin for [Gradle](#) loads the google-services.json file you just downloaded. Modify your build.gradle files to use the plugin.

Project-level build.gradle (<project>/build.gradle):

```
buildscript {  
    repositories {  
        // Check that you have the following line (if not, add it):  
        google() // Google's Maven repository  
    }  
    dependencies {  
        ...  
    }  
}
```

6. Click on the Continue to console button.

× Add Firebase to your Android app

Register app

Android package name: com.yoyogames.YoyoPlayServices2, App nickname: yysetup_android

Download config file

Add Firebase SDK

4 Next steps

You're all set!

Make sure to check out the [documentation](#) to learn how to get started with each Firebase product that you want to use in your app.

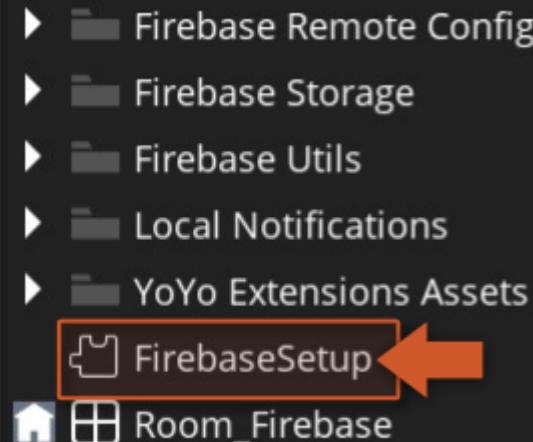
You can also explore [sample Firebase apps](#).

Or, continue to the console to explore Firebase.

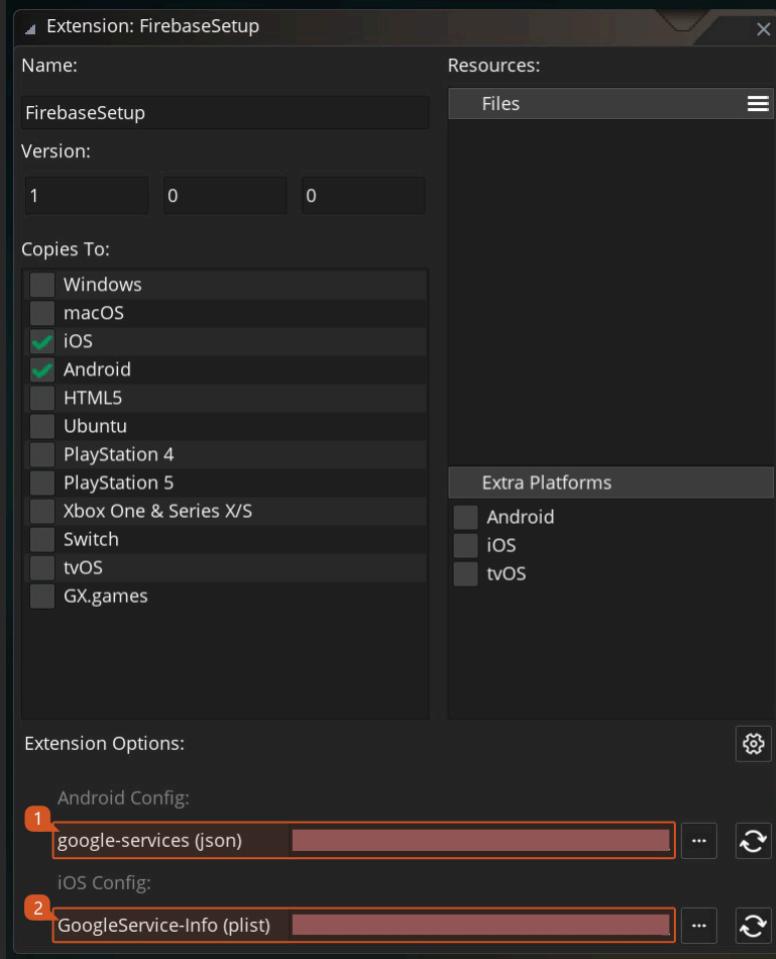
Previous

Continue to console

7. Now go into GameMaker, double click the extension **FirebaseSetup** asset.



8. In the extension panel just fill in the paths for the correct files (Android and/or iOS).



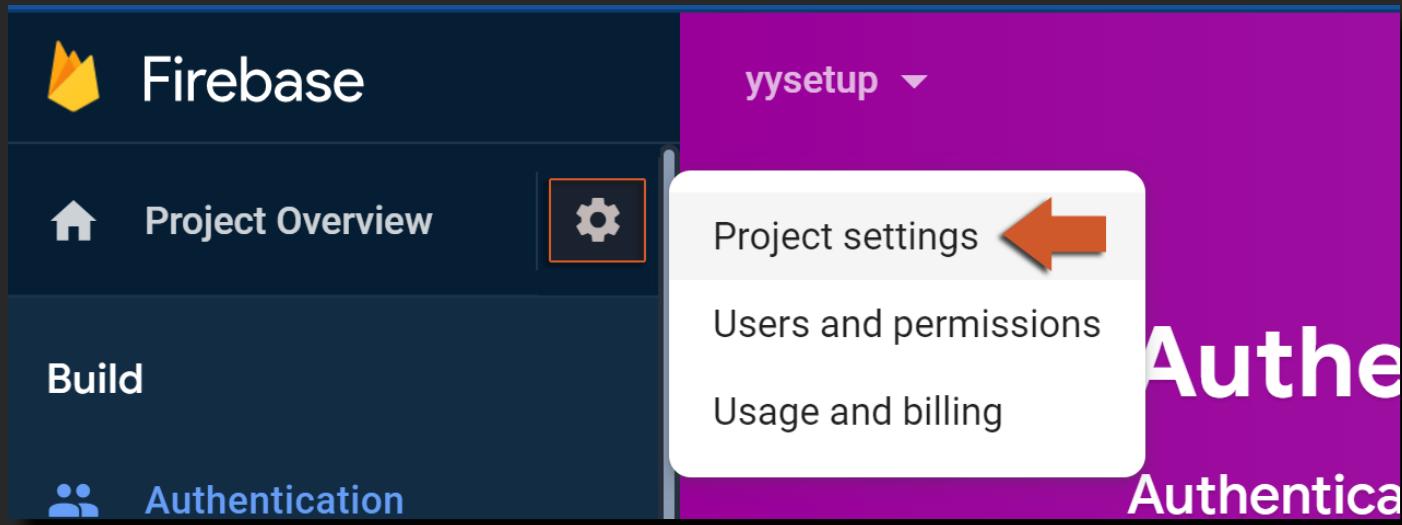
9. You have now finished the main setup for all Firebase Android modules!

iOS Setup

This setup is necessary for all the Firebase modules using iOS and needs to be done once per project, and basically involves importing the `GoogleServices-Info.plist` file into your project.

IMPORTANT Please refer to [this Helpdesk article](#) for instructions on setting up an iOS project.

1. Click the **Settings** icon (next to Project Overview) and select **Project settings**:



2. Now go to the **Your apps** section and click on the **iOS** button:

The screenshot shows the 'Project settings' page under the 'ysetup' tab. The 'Your apps' section is visible, containing a message: 'There are no apps in your project. Select a platform to get started.' Below this message are three circular icons representing different platforms: iOS, TV, and Web. The iOS icon is highlighted with a red box. To the right of the icons is a circular arrow icon.

3. Fill the form with your iOS Bundle ID, App nickname and AppStore ID (last two are optional).

×

Add Firebase to your iOS app

1 Register app

iOS bundle ID ⓘ

An orange arrow points to this input field.

App nickname (optional) ⓘ

App Store ID (optional) ⓘ

4. Click on **Download GoogleService-info.plist** (make sure to save this file as we will need it in subsequent steps).

× Add Firebase to your iOS app

✓ Register app

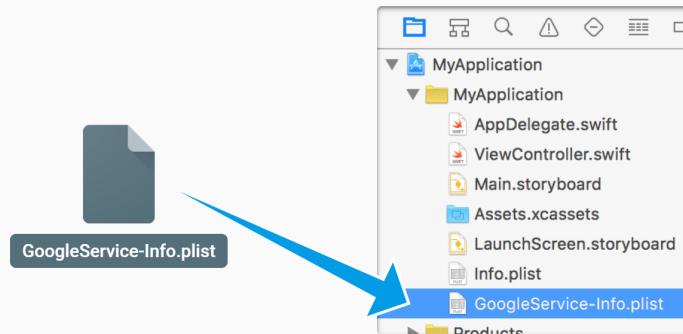
iOS bundle ID: com.yoyogames.yyfirebase

2 Download config file

Instructions for Xcode below | [Unity](#) [C++](#)

→ [Download GoogleService-Info.plist](#)

Move the GoogleService-Info.plist file you just downloaded into the root of your Xcode project and add it to all targets.



[Next](#)

5. Ignore this screen, as this is already done in the extension.

× Add Firebase to your iOS app

✓ Register app

iOS bundle ID: com.yoyogames.yyfirebase

Download config file

3 Add Firebase SDK

Instructions for CocoaPods | [SwiftPM](#) [Download ZIP](#) [Unity](#) [C++](#)

Google services use [CocoaPods](#) to install and manage dependencies. Open a terminal window and navigate to the location of the Xcode project for your app.

Create a Podfile if you don't have one:

\$ pod init



Open your Podfile and add:

```
# add the Firebase pod for Google Analytics
```

6. Ignore this screen as well, as this is also done in the extension.

× Add Firebase to your iOS app

- ✓ Register app
iOS bundle ID: com.yoyogames.yyfirebase

- Download config file

- Add Firebase SDK

4 Add initialization code

To connect Firebase when your app starts up, add the initialization code below to your main `AppDelegate` class.

Swift Objective-C

```
import UIKit
import Firebase

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
```



7. Click on the Continue to console button:

× Add Firebase to your iOS app

- ✓ Register app
iOS bundle ID: com.yoyogames.yyfirebase

- Download config file

- Add Firebase SDK

- Add initialization code

5 Next steps

You're all set!

Make sure to check out the [documentation](#) to learn how to get started with each Firebase product that you want to use in your app.

You can also explore [sample Firebase apps](#).

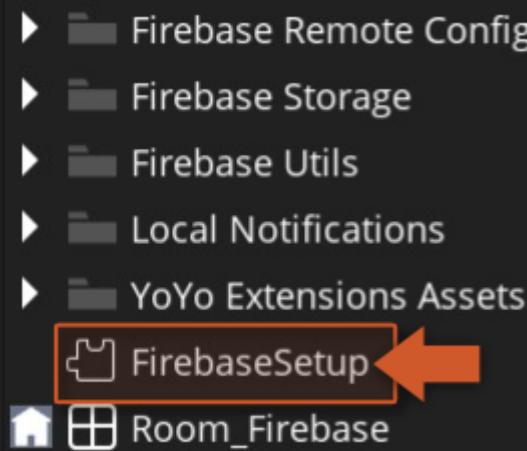
Or, continue to the console to explore Firebase.

Previous

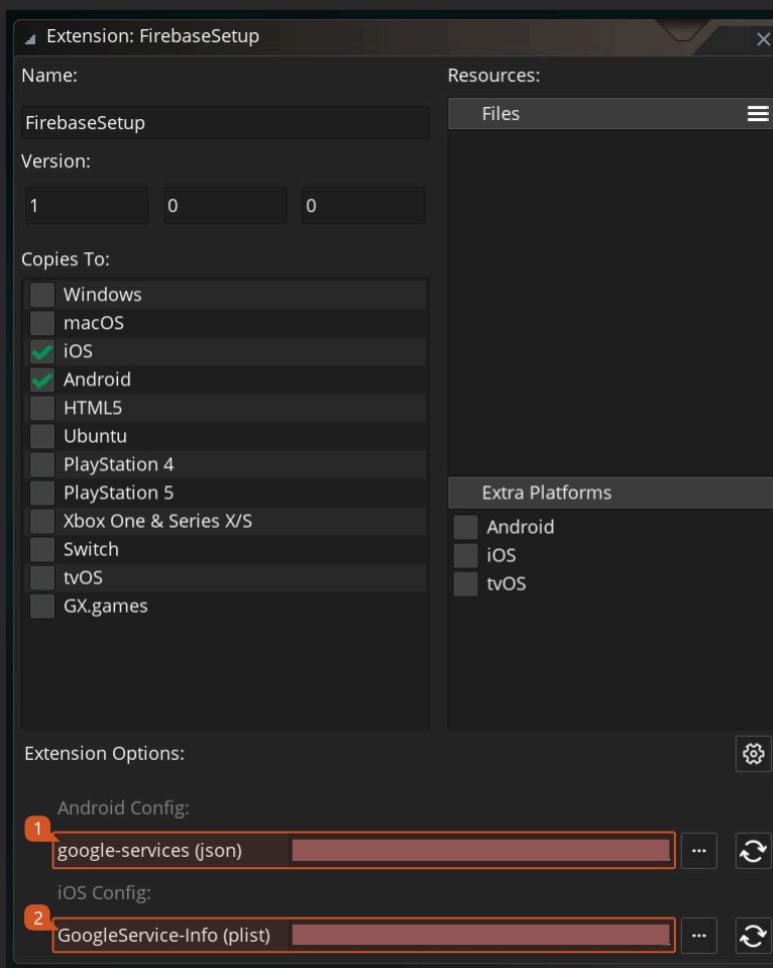
Continue to console



8. Now go into GameMaker, double click the extension **FirebaseSetup** asset.



9. In the extension panel just fill in the paths for the correct files (Android and/or iOS).

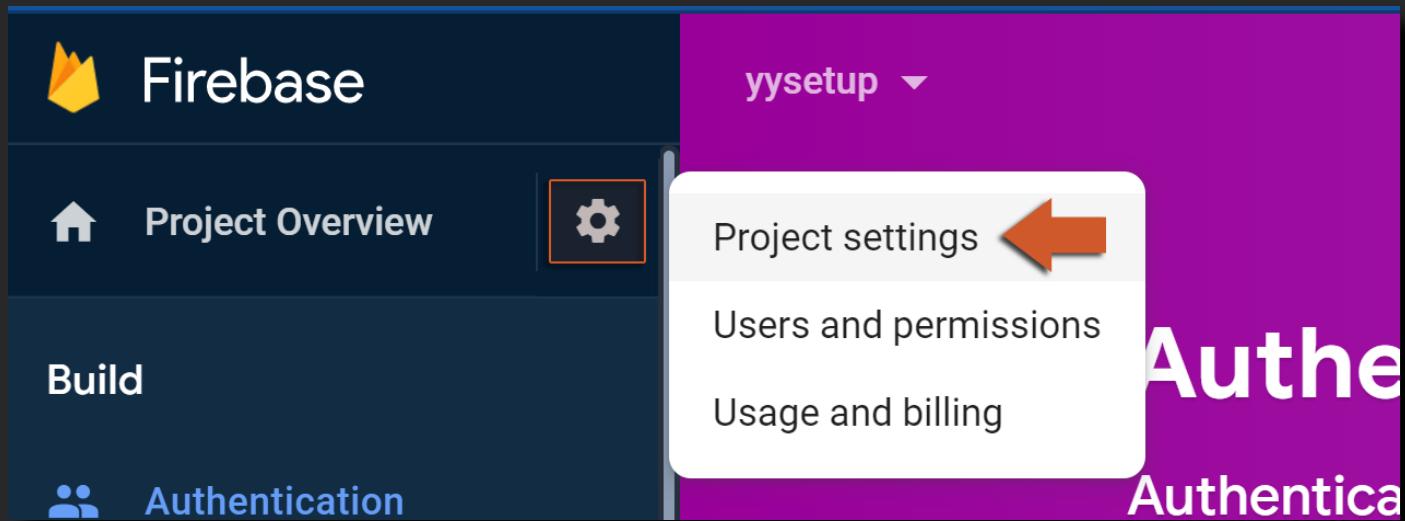


10. Make sure to set up **CocoaPods** for your project *unless* you are only using the REST API in an extension (if one is provided -- not all extensions provide a REST API) or the Firebase Cloud Functions extension (which only uses a REST API).
11. You have now finished the main setup for all Firebase iOS modules!

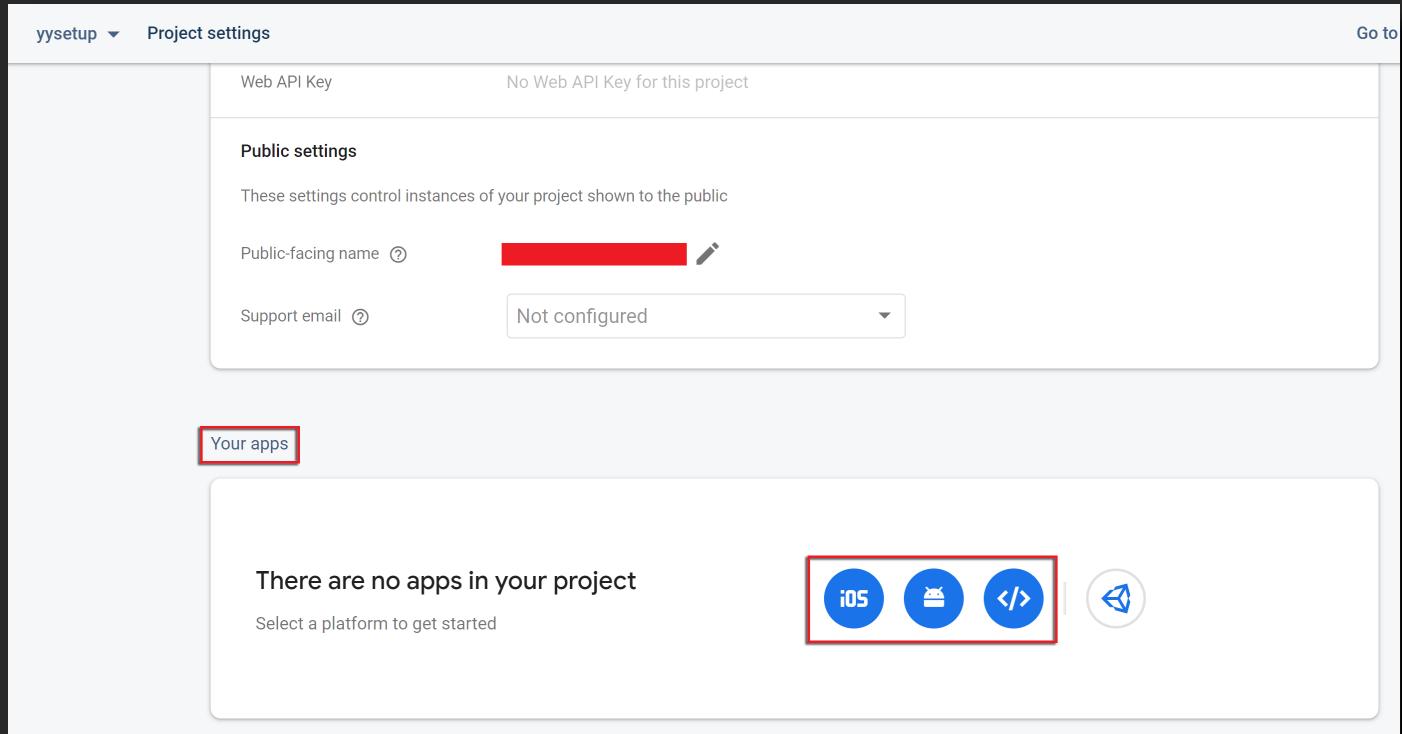
Web Setup

This setup is necessary for all the Firebase modules using Web export and needs to be done once per project, and basically involves adding Firebase libraries and your Firebase values to the `index.html` file in your project.

1. Click the **Settings** icon (next to Project Overview) and then **Project settings**:



2. Now go to the **Your apps** section and click on the **Web (</>)** button:



3. Enter your App nickname (required):

X Add Firebase to your web app

1 Register app

App nickname ⓘ

My web app



! An app nickname is required.

Also set up **Firebase Hosting** for this app. [Learn more ↗](#)

Hosting can also be set up later. It's free to get started anytime.

Register app

4. On this screen, just copy the `firebaseConfig` struct:

2 Add Firebase SDK

Use npm ⓘ Use a <script> tag ⓘ

If you're already using [npm ↗](#) and a module bundler such as [webpack ↗](#) or [Rollup ↗](#), you can run the following command to install the latest SDK:

```
$ npm install firebase
```



Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
import { getAnalytics } from "firebase/analytics";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries
```

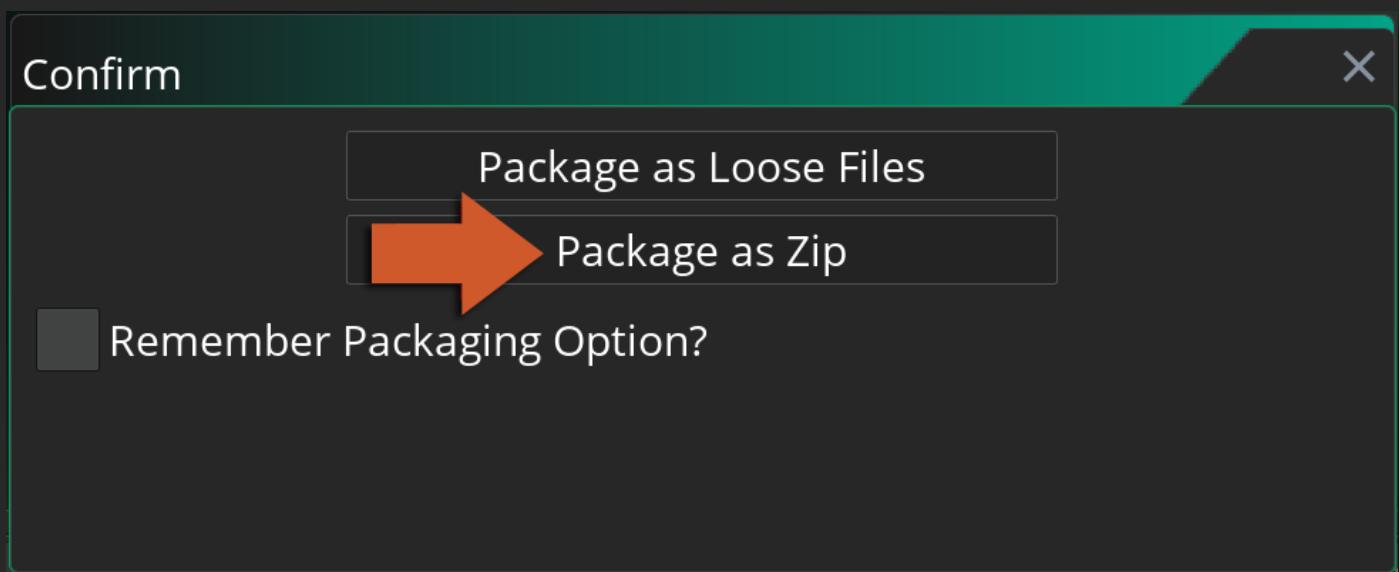
```
// Your web app's Firebase configuration
// For Firebase JS SDK v7.20.0 and later, measurementId is optional
const firebaseConfig = {
  apiKey: [REDACTED],
  authDomain: [REDACTED],
  projectId: [REDACTED],
  storageBucket: [REDACTED],
  messagingSenderId: [REDACTED],
  appId: [REDACTED],
  measurementId: [REDACTED]
};
```

```
// Initialize Firebase
const app = initializeApp(firebaseConfig);
const analytics = getAnalytics(app);
```

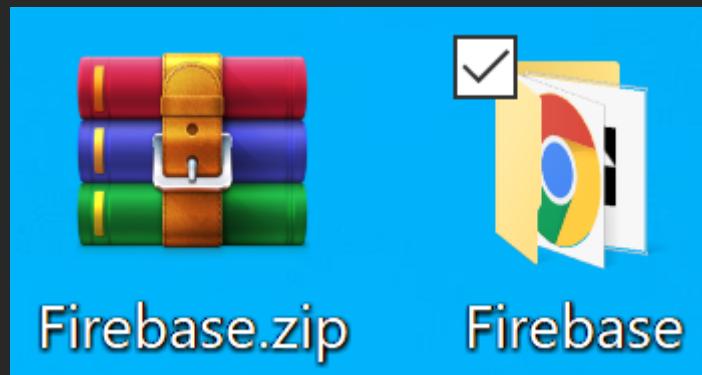


5. Now go back into GameMaker Studio 2 and build your project.

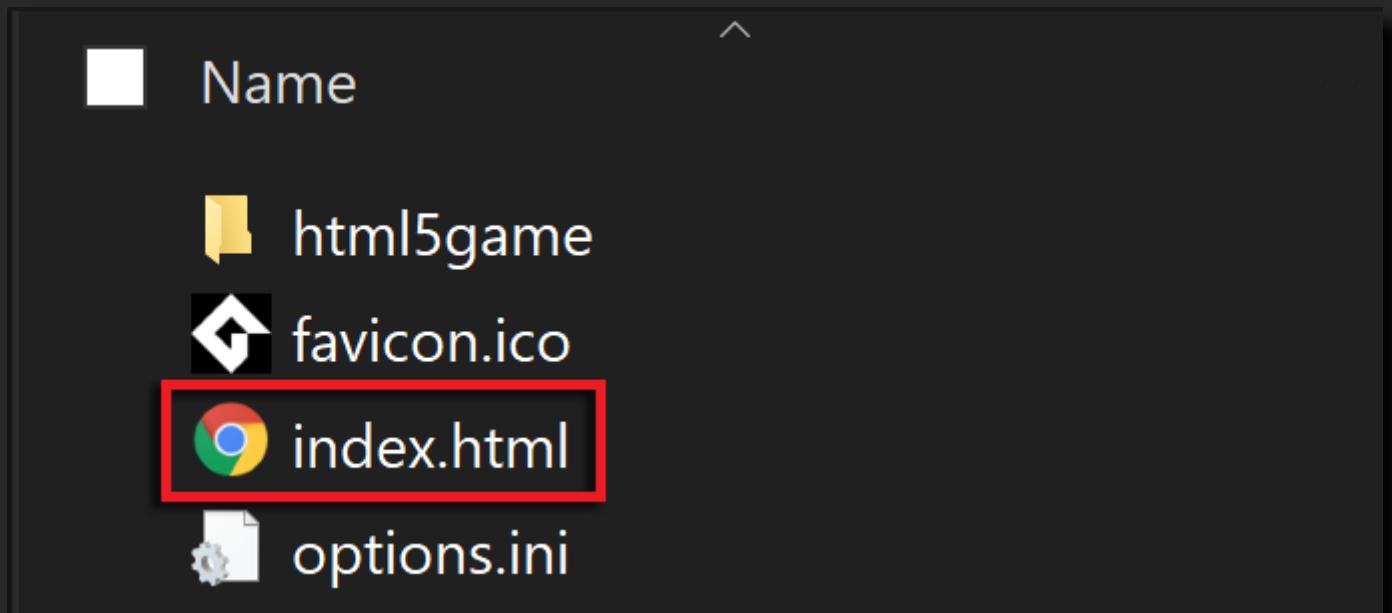
6. Choose the **Package As Zip** option:



7. Locate the created package and **extract it** into a folder.



8. Open the extracted folder and look for an **index.html** file.



9. Open the **index.html** file in Notepad++ or Visual Studio Code (or any other text editor you prefer).

```

65  /* END - Login Dialog Box */
66  :webkit-full-screen {
67      width: 100%;
68      height: 100%;
69  }
70  </style>
71  </head>
72
73 <body>
74     <div class="gm4html5_div_class" id="gm4html5_div_id">
75         <!-- Create the canvas element the game draws to -->
76         <canvas id="canvas" width="1366" height="768" >
77             <p>Your browser doesn't support HTML5 canvas.</p>
78         </canvas>
79     </div>
80
81     <!-- Run the game code -->

```

10. Now we need to add the following code between the `</head>` and `<body>` tags (line 72 in the `html.index` image above):

```

<script src="https://www.gstatic.com/firebasejs/8.9.1.firebaseio.js"></script>
<script src="https://www.gstatic.com/firebasejs/8.9.1/firebaseAnalytics.js"></script>
<script src="https://www.gstatic.com/firebasejs/8.9.1/firebaseAuth.js"></script>
<script src="https://www.gstatic.com/firebasejs/8.9.1.firebaseio.js"></script>
<script src="https://www.gstatic.com/firebasejs/8.9.1/firebaseRestore.js"></script>
<script src="https://www.gstatic.com/firebasejs/8.9.1/firebaseRemoteConfig.js"></script>

<script>
  const firebaseConfig = {
    apiKey: "",
    authDomain: "",
    databaseURL: "",
    projectId: "",
    storageBucket: "",
    messagingSenderId: "",
    appId: "",
    measurementId: ""
  };
  firebase.initializeApp(firebaseConfig);

</script>

```

11. Replace the `const firebaseConfig` part with the one copied in step 4:

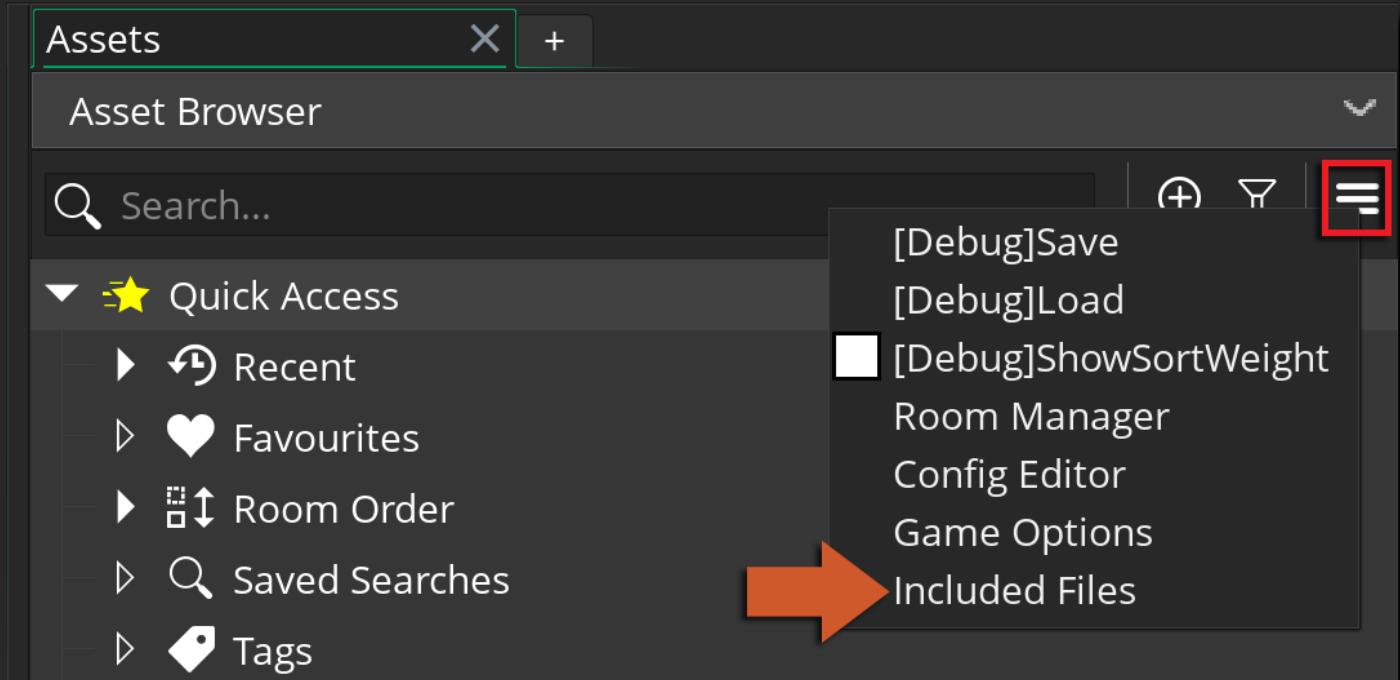
```

68         height: 100%;
69     }
70 
```

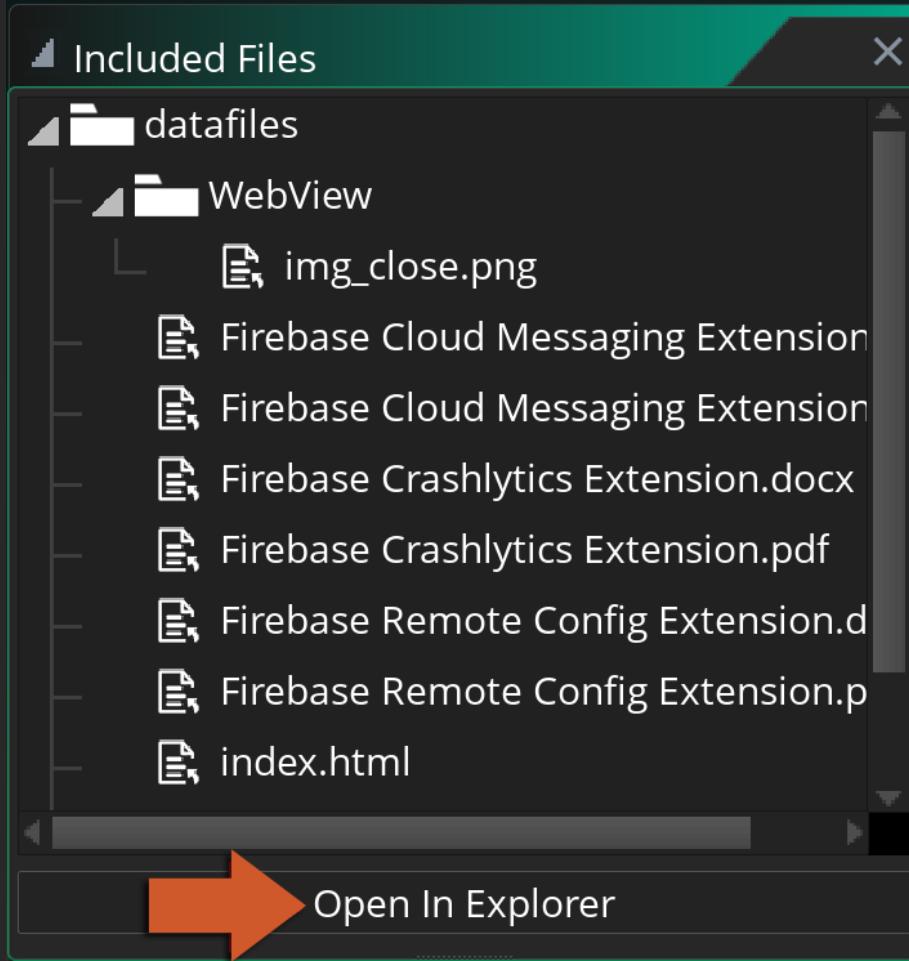
```

71 </style>
72 </head>
73
74 <!-- Firebase -->
75 <script src="https://www.gstatic.com/firebasejs/8.9.1.firebaseio.js"></script>
76 <script src="https://www.gstatic.com/firebasejs/8.9.1/firebase-analytics.js"></script>
77 <script src="https://www.gstatic.com/firebasejs/8.9.1/firebase-auth.js"></script>
78 <script src="https://www.gstatic.com/firebasejs/8.9.1/firebase-database.js"></script>
79 <script src="https://www.gstatic.com/firebasejs/8.9.1/firebase-firebase.js"></script>
80
81
82 <script>
83
84 // Your web app's Firebase configuration
85 // For Firebase JS SDK v7.20.0 and later, measurementId is optional
86 const firebaseConfig = {
87   apiKey: "XXXXXXXXXXXXXXXXXXXXXX",
88   authDomain: "XXXXXXXXXXXXXXXXXXXXXX",
89   databaseURL: "XXXXXXXXXXXXXXXXXXXXXX",
90   projectId: "XXXXXXXXXXXXXXXXXXXXXX",
91   storageBucket: "XXXXXXXXXXXXXXXXXXXXXX",
92   messagingSenderId: "XXXXXXXXXXXXXX",
93   appId: "XXXXXXXXXXXXXXXXXXXXXX",
94   measurementId: "XXXXXXXXXXXXXX"
95 };
96
97 firebase.initializeApp(firebaseConfig);
98
99 </script>
100
101
102
103 <body>
104     <div class="gm4html5_div_class" id="gm4html5_div_id">
```

12. Go back into GameMaker and open your **Included Files** folder.



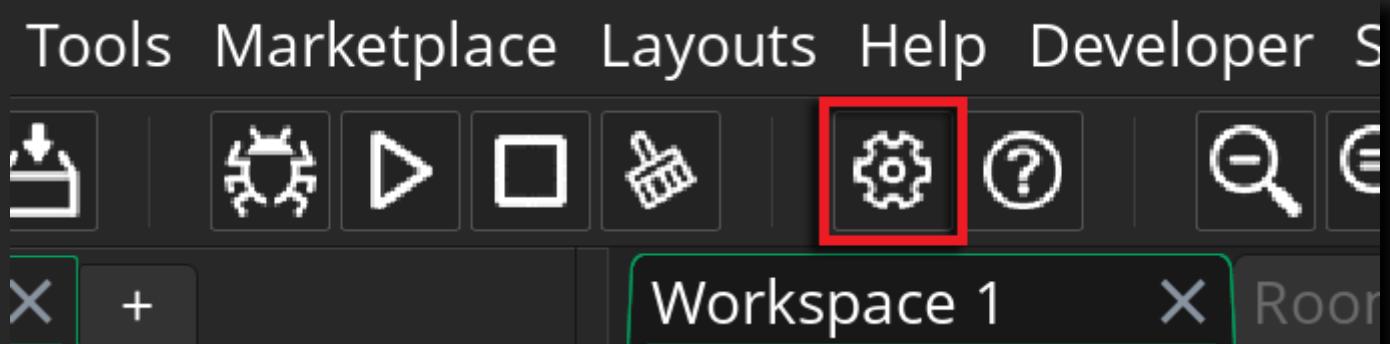
13. Press the Open in Explorer button:



14. Place your **index.html** file inside the folder that opens (/datafiles).

Name	Date modified	Type
WebView	9/14/2021 7:44 AM	File folder
index.html	9/13/2021 9:12 AM	Chrome H

15. Back in GameMaker, click on the **Game Options** button.



16. Go into the **HTML5** platform settings

Game Options - Main

- ▲ Main Options
 - General
- ▲ Platform Settings
 - Opera GX
 - Windows
 - macOS
 - Ubuntu
 - HTML5** ←
 - Android
 - Amazon Fire
 - iOS
 - tvOS

17. In the Advanced section go to the "Include file as index.html" dropdown and select the **index.html** option (this is the file we have just added to the included files).

HTML5 - General

Created with GameMaker Studio 2

Browser Title

1 0 0 0 Version

html5game Folder Name

index.html Output Name

▲ Options

- Output debug to console
- Display cursor
- Display "Running outside server" alert

« ▲ Advanced

Use Default

Included file as index.html

Use Default

index.html

Loading bar extension

18. Press **Apply** and the main setup for all Firebase Web modules is finished!

FirebaseAnalytics_LogEvent

This function logs an app event. The event can have up to 25 parameters. Note that events with the same name must have the same parameters. Up to 500 event names are supported. Using predefined `FirebaseAnalytics.Event` and/or `FirebaseAnalytics.Param` is recommended for optimal reporting.

Syntax:

```
FirebaseAnalytics_LogEvent(event, params)
```

Argument	Type	Description
event	string	The event data (available <code>events</code> reference)
params	string	The data as a JSON formatted string (available <code>params</code> reference)

Returns:

N/A

Example:

```
var _data = { screen_name : room_get_name(room) };
FirebaseAnalytics_LogEvent("screen_view", json.stringify(_data))
```

In the code sample above we will submit a log to the event "screen_view" (see `event` details), so first create a data struct with a parameter named "screen_name" (this is one of the allowed parameters) and convert it to a string using the `json.stringify` function.

FirebaseAnalytics_ResetAnalyticsData

This function clears all analytics data for the app from the current device and resets the app instance ID.

NOTE This will not clear analytics previously collected by the app that have already been pushed to the server.

Syntax:

```
FirebaseAnalytics_ResetAnalyticsData()
```

Returns:

N/A

Example:

```
if (global.stopDataCollection)
{
    FirebaseAnalytics_ResetAnalyticsData();
    FirebaseAnalytics_SetAnalyticsCollectionEnabled(false);
}
```

The code above will check if a global variable is set to true (`global.stopDataCollection`) and if so we will clear the local collected data for this extension and disable the analytics data collection (using the function [FirebaseAnalytics_SetAnalyticsCollectionEnabled](#)).

FirebaseAnalytics_SetAnalyticsCollectionEnabled

This function is used to enable or disable analytics collection for the app on the current device. This setting is persisted across app sessions. By default it is enabled.

Syntax:

```
FirebaseAnalytics_SetAnalyticsCollectionEnabled(enable)
```

Argument	Type	Description
enable	boolean	Whether or not analytics collection should be enabled.

Returns:

N/A

Example:

```
if (global.stopDataCollection)
{
    FirebaseAnalytics_ResetAnalyticsData();
    FirebaseAnalytics_SetAnalyticsCollectionEnabled(false);
}
```

The code above will check if a global variable is set to true and if so we will clear the local collected data (using the function [FirebaseAnalytics_ResetAnalyticsData](#)) for this extension and disable the analytics data collection.

FirebaseAnalytics_SetDefaultEventParameters

This function sets "default" parameters that will be provided with every event logged from the SDK, including automatic ones. The values passed in the parameters bundle will be added to the map of default event parameters. These parameters persist across app runs. They are of lower precedence than event parameters, so if an event parameter and a parameter set using this function have the same name, the value of the event parameter will be used. The same limitations on event parameters apply to default event parameters.

Syntax:

```
FirebaseAnalytics_SetDefaultEventParameters(parameters)
```

Argument	Type	Description
parameters	string	A JSON formatted string of a struct to be added to every event. These parameters will be added to the default event parameters, replacing any existing parameter with the same name. Valid parameter values are string and double . Setting a key's value to <code>pointer_null</code> will clear that parameter.

Returns:

N/A

Example:

```
var parameters = {  
  playerName: "Hero",  
  bossName: pointer_null,  
};
```

```
FirebaseAnalytics_SetDefaultEventParameters(json_stringify(parameters))
```

With the code above we are setting the default parameters to be used on all further calls to **FirebaseAnalytics_LogEvent**. First we create a struct (`parameters`) with a `"playerName"` and

"bossName" keys, the boss name is set to `pointer_null` meaning it will be removed from the set of default event parameters.

FirebaseAnalytics_SetSessionTimeoutDuration

This function sets the duration of inactivity (in milliseconds) that terminates the current session. The default value is 1800000 (30 minutes).

Syntax:

```
Fi rebaseAnal yti cs_SetSessionTimeoutDuration(milliseconds)
```

Argument	Description
milliseconds	Session timeout duration in milliseconds.

Returns:

N/A

Example:

```
Fi rebaseAnal yti cs_SetSessionTimeout(3600000) //1 hour
```

The code above sets the timeout duration to 1 hour, meaning that if no events are logged within this timeout the session will be terminated.

FirebaseAnalytics_SetUserId

This function sets the user ID property. This feature must be used in accordance with [Google's Privacy Policy](#).

Syntax:

```
Fi rebaseAnalytics_SetUserId(userID)
```

Argument	Description
userID	The user ID string to assign to the user of this app on this device, which must be non-empty and no more than 256 characters long. Setting the ID to <code>pointer_null</code> removes the user ID.

Returns:

N/A

Example:

```
Fi rebaseAnalytics_SetUserId("myUser123");
```

The above code will set the user ID to the string `"myUser123"`.

FirebaseAnalytics_SetUserProperty

This function sets a user property to the given value. Up to 25 user property names are supported. Once set, user property values persist throughout the app life cycle and across sessions.

Syntax:

```
FirebaseAnalytics_SetUserProperty(key, value)
```

Argument	Description
key	The name of the user property to set. Should contain 1 to 24 alphanumeric characters or underscores and must start with an alphabetic character. IMPORTANT The "firebase_", "google_" and "ga_" prefixes are reserved and should not be used for user property names.
value	The value of the user property. Values can be up to 36 characters long. Setting the value to <code>pointer_null</code> removes the user property.

Returns:

N/A

Example:

```
FirebaseAnalytics_SetUserProperty("Location", "UK");
```

The code above will set an user property `"Location"` to the value `"UK"`.