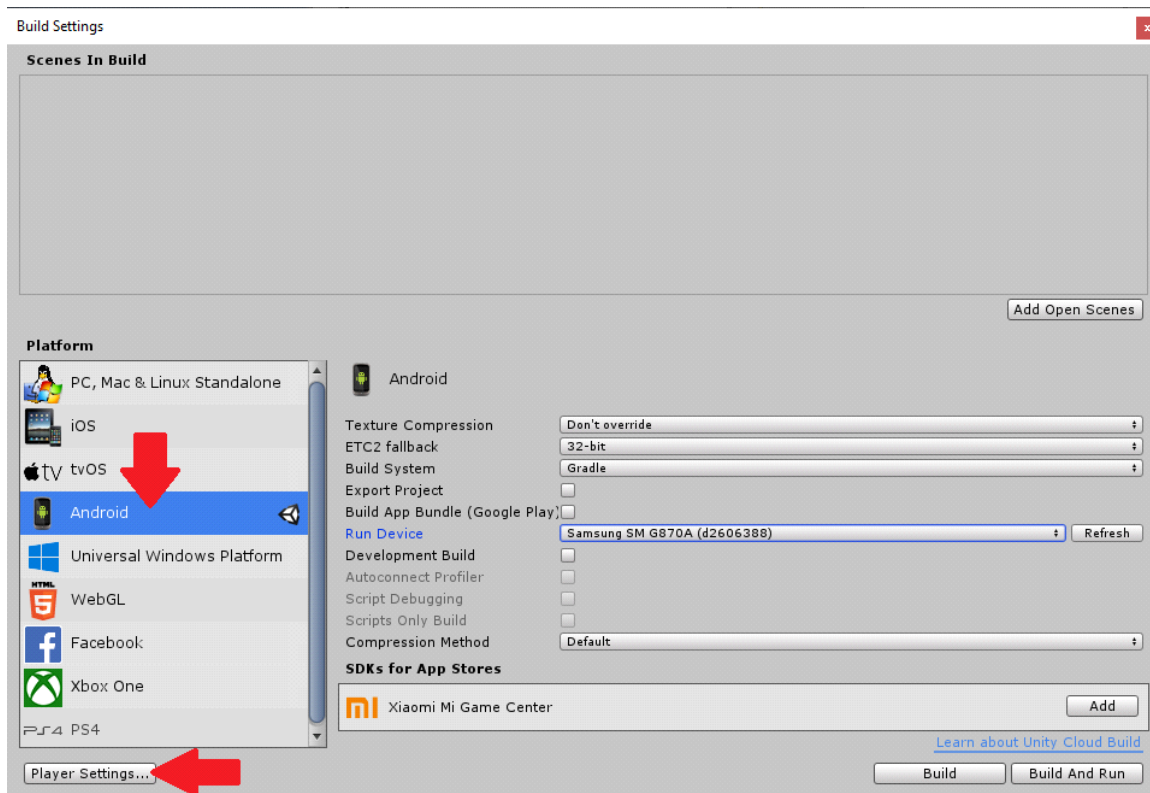


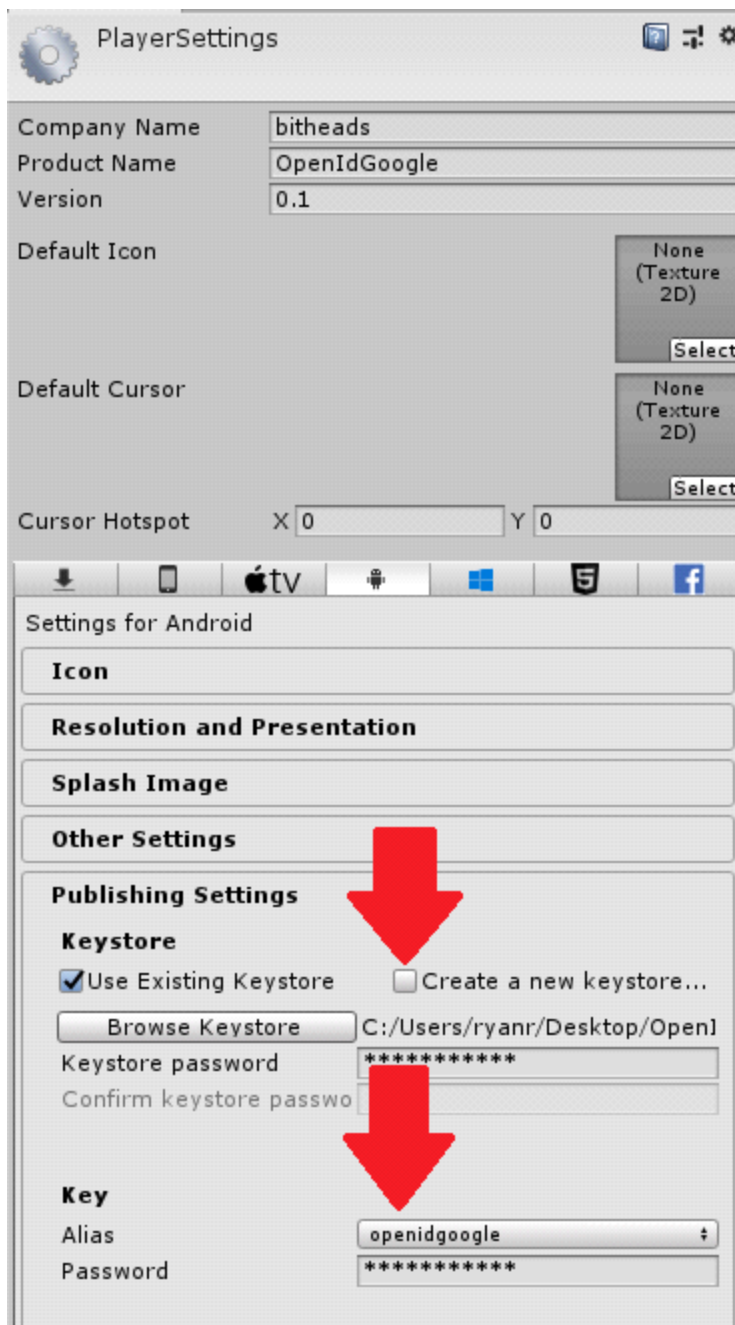
Google Authentication with BrainCloud using

Google OpenId

ANDROID - Unity

1. create unity project, go to File > Build Settings, and change the platform to Android. Then on the bottom left of Build Settings, go into Player Settings and under Publishing Settings, make a new signing keystore and get the SHA-1. Make sure to also add an alias to your key.



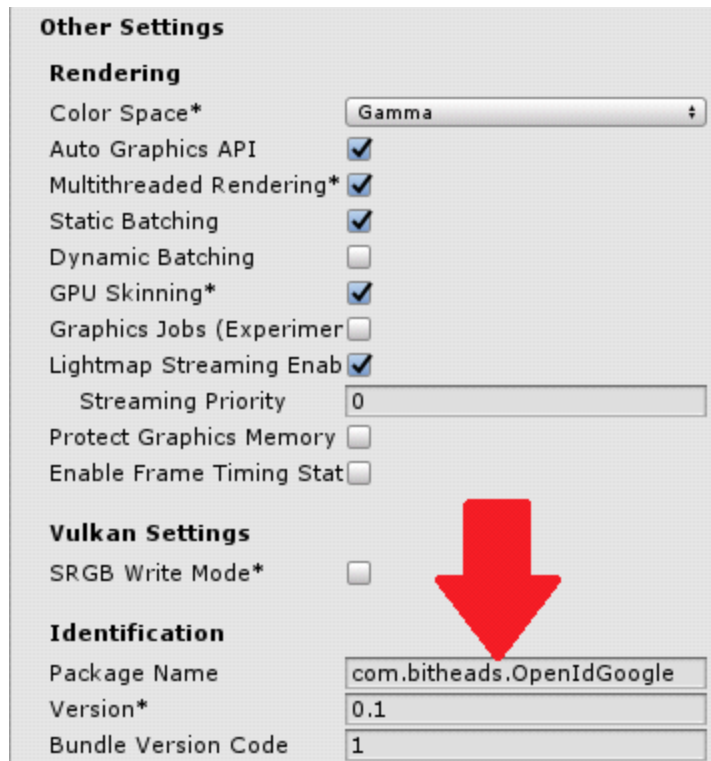


To get the SHA-1 follow google's tutorial here:

<https://support.google.com/cloud/answer/6158849?hl=en#installedapplications&android>

If you're having trouble finding the keytool.exe, try looking in your jdk.

You will also need to make a package name for your project which can also be done in the Player Settings. Make sure it matches the Company name you chose and Product name in the Player Settings.



2. Add Braincloud to your Unity Project. Get the latest release here:

<https://github.com/getbraincloud/braincloud-csharp/releases> and import the package into your project.

Make sure to create a braincloud app here:

<https://portal.braincloudservers.com/>

Make sure to enable the platform Google Android which can be found under Design > Core App Info > Platforms

Core App Info - Platforms



brainCloud performs two sets of version checks when your client connects specify a minimum version of their client application itself (ensuring that u

Minimum Client App Versions

Supported	Platform
<input checked="" type="checkbox"/>	Apple iOS
<input checked="" type="checkbox"/>	Apple tvOS
<input checked="" type="checkbox"/>	Apple watchOS
	BlackBerry
	Facebook
<input checked="" type="checkbox"/>	Google Android

3. Create a New Project on the Google Cloud Console otherwise known as Google Cloud Platform here: <https://console.cloud.google.com/>

Select a project



NEW PROJECT

Search projects and folders

4. Under APIs & Services in your new project, go to the OAuth consent screen and make it an external app unless you have reason to make an internal app. NOTE: DO NOT add an application logo picture to the consent screen unless you have everything you need for all the other input fields. If you add one, you will not be able to reverse it and your app will permanently be in need of verification, and you will need to have links and authorized domains to fix this.



Home



Pins appear here ?



Marketplace



Billing



APIs & Services



Support



IAM & admin



Getting started



Security



Dashboard

Library

Credentials

OAuth consent screen


Domain verification

Page usage agreements


OAuth consent screen

Choose how you want to configure and register your app, including your target users. You can only associate one app with your project.

User Type

☐ Internal 

Only available to users within your organization. You will not need to submit your app for verification.

☒ External 

Available to any user with a Google Account.



5. Go to the Google Sign In plugin page on github and download the latest release of the plugin.

<https://github.com/googlesamples/google-signin-unity>

You will also need to take the steps to Configure a Google API project here:

<https://developers.google.com/identity/sign-in/android/start> which is mentioned under "Configuring the application on the API Console". Step 2 of this tutorial is the most important as it will generate the web client we need on the google cloud platform. During the process it will ask what type of app is calling it, choose ANDROID and put in the info of your app, this will generate both android and a web client credential.

2

Configure a Google API project


To use the sample, you need to provide some additional information to finish setting up your project. Click the button below, and specify the package name `com.google.samples.quickstart.signin` when prompted. You will also need to provide the SHA-1 hash of your signing certificate. See [Authenticating Your Client](#) for information.

Configure a project






6. Go back to your Google Cloud Platform project, and go under APIs and Services, and Credentials. You will notice there is a WebApp that has been auto generated by Google. The web app Id and secret should match those that you were given after step 5.

Credentials

Create credentials  Delete

Create credentials to access your enabled APIs. For more information, see the [authentication documentation](#).

OAuth 2.0 client IDs

<input type="checkbox"/> Name	Creation date 	Type	Client ID
<input type="checkbox"/> OAuth client	Jan 10, 2020	Android	780423637529-dd18o27qo6pafj0l217uak8hpsiq9ntp.apps.googleusercontent.com 
<input type="checkbox"/> Web client (Auto-created for Google Sign-in)	Jan 10, 2020	Web application	780423637529-hcg9gdb04egbbh7h6k9pkukd8a9j9f8u.apps.googleusercontent.com 



7. Go back to your braincloud app, and under Design > Core App Info > Application IDs, and Configure Platforms, select the Google Button and fill in the bottom 3 fields, with the Web App Id and secret of your Google Cloud Platform App. Then save changes.

Core App Info - Application IDs

✓ Save Changes

Configure Platforms

Apple

Facebook

Google

Steam

Twitter

Google Service Account Email

Google Service Account p12 Certificate

Select a certificate file

✗ Not Configured

Google Package Name

Google App ID

780423637529

Google Client ID

780423637529-hcg9gdbo4egbbh7h6k9pkukd8a9j9f8u.apps.googleu:

Google Client Secret

8. Import the google signin unity package into your Unity Project. NOTE: you may run into a few errors upon importing the package. Research has mentioned that there are conflicting declarations in the Parse folder of the plugin. Navigate to it and delete the PARSE folder and its contents and the errors should go away.

```
[14:12:24] Assets\GoogleSignIn\Future.cs(72,40): error CS0433: The type 'TaskCompletionSource<T>' exists in both 'Unity.Tasks, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null' and 'netstandard, Version=2.0.0.0, Culture=neutral, PublicKeyToken=cc7b13ffcd2dd451'
[14:12:24] Assets\GoogleSignIn\GoogleSignIn.cs(117,12): error CS0433: The type 'Task<T>' exists in both 'Unity.Tasks, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null' and 'netstandard, Version=2.0.0.0, Culture=neutral, PublicKeyToken=cc7b13ffcd2dd451'
[14:12:24] Assets\GoogleSignIn\GoogleSignIn.cs(130,12): error CS0433: The type 'Task<T>' exists in both 'Unity.Tasks, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null' and 'netstandard, Version=2.0.0.0, Culture=neutral, PublicKeyToken=cc7b13ffcd2dd451'
```

9. Add Google code to your unity app

Key snippets of code:

```
using Google;
```

This gives you access to Google's services. You may also want "using System.Threading.Tasks;" because we will use a thread for the callback.

```
GoogleSignInConfiguration configuration;
```

This is the configuration of your authentication. You can customize fields to obtain important codes and information about the account that is authenticated to google. The email and the IdToken are what is

needed in order to authenticate a google account with Braincloud. You can initialize it and set the values.

```
configuration = new GoogleSignInConfiguration
{
    WebClientId = webClientId,
    RequestEmail = true,
    RequestIdToken = true,
    RequestAuthCode = true
};
```

Note that this can also be set on the fly instead of on Start or Awake. You just need to access the Configuration of your google instance like this:

```
GoogleSignIn.Configuration.RequestEmail = true;
GoogleSignIn.Configuration.RequestIdToken = true;
GoogleSignIn.Configuration.RequestAuthCode = true;
```

As an example you may have a function and callback like this:

```
public void OnGoogleSignIn()
{
    //set the google configuration to the configuration object you set up
    GoogleSignIn.Configuration = configuration;

    //Can define this if you're using the game signin
    GoogleSignIn.Configuration.UseGameSignIn = false;

    //Can also define these tags here like this.
    //GoogleSignIn.Configuration.RequestEmail = true;
    //GoogleSignIn.Configuration.RequestIdToken = true;
    //GoogleSignIn.Configuration.RequestAuthCode = true;
```

```

        // With the configuration set, its now time to start trying to sign in.
        Pass in a callback to wait for success.

        GoogleSignIn.DefaultInstance.SignIn().ContinueWith(OnGoogleAuthSignIn);
    }

    //use a callback with a task to easily get results from the callback in order to get
    the values you need.

    public void OnGoogleAuthSignIn(Task<GoogleSignInUser> task)
    {
        if (task.IsFaulted)
        {
        }
        else
        {
            authCode = task.Result.AuthCode;

            idToken = task.Result.IdToken;

            email = task.Result.Email;
        }
    }
}

```

10. Take the email result, and the IdToken result, and pass it into the braincloud
AuthenticateGoogleOpenId

```
AuthenticateGoogleOpenId(email, idToken, true, OnSuccess_Authenticate,
OnError_Authenticate);
```

iOS - Unity

1. create unity project, go to File > Build Settings, and change the platform to IOS. Then on the bottom left of Build Settings, go into Player Settings and under Other Settings set your Bundle Identifier. It will be com.CompanyName.ProductName being the values you set up at the top of the Player settings.

2. Add Braincloud to your Unity Project. Get the latest release here:


<https://github.com/getbraincloud/braincloud-csharp/releases> and import the package into your project.

Make sure to create a braincloud app here:

<https://portal.braincloudservers.com/>

Make sure to enable the platform Google Android which can be found under Design > Core App Info > Platforms

Core App Info - Platforms

 brainCloud performs two sets of version checks when your client connects specify a minimum version of their client application itself (ensuring that u

Minimum Client App Versions

Supported	Platform
<input checked="" type="checkbox"/>	Apple iOS
<input checked="" type="checkbox"/>	Apple tvOS
<input checked="" type="checkbox"/>	Apple watchOS
<input checked="" type="checkbox"/>	BlackBerry
<input checked="" type="checkbox"/>	Facebook
<input checked="" type="checkbox"/>	Google Android

3. Create a New Project on the Google Cloud Console otherwise known as Google Cloud Platform here:
<https://console.cloud.google.com/>

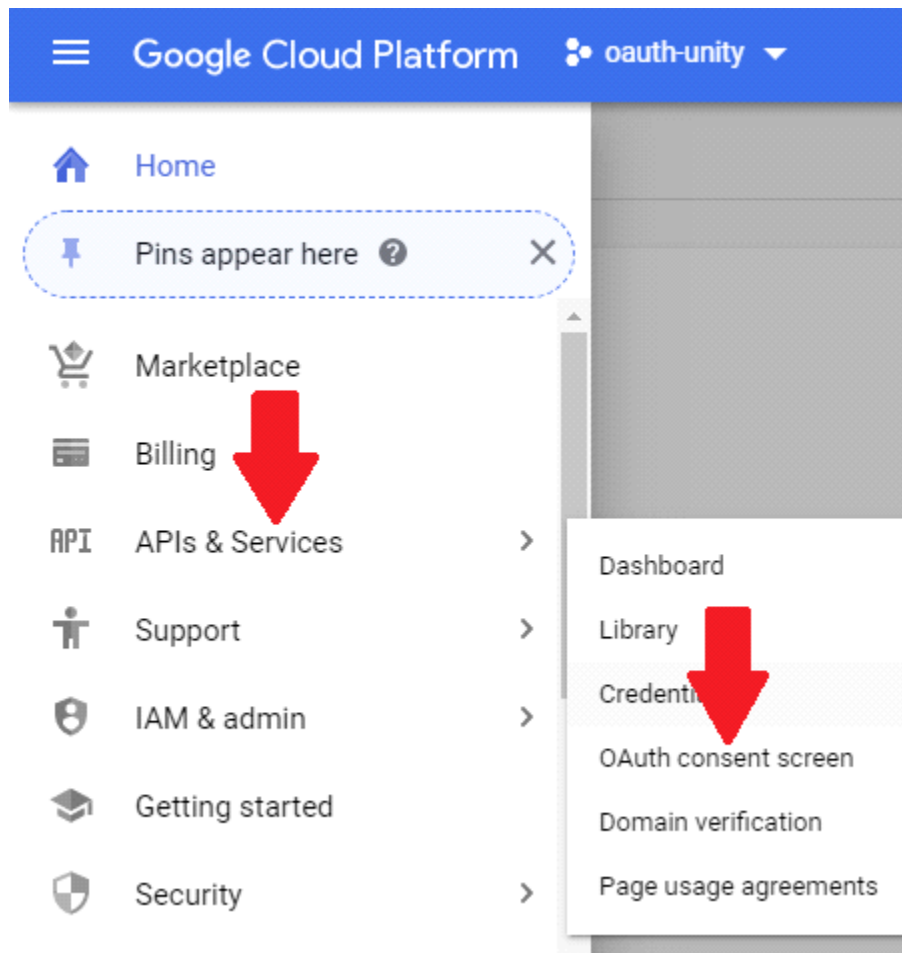
Select a project



 Search projects and folders

4. Under APIs & Services in your new project, go to the OAuth consent screen and make it an external


app unless you have reason to make an internal app. NOTE: DO NOT add an application logo picture to the consent screen unless you have everything you need for all the other input fields. If you add one, you will not be able to reverse it and your app will permanently be in need of verification, and you will need to have links and authorized domains to fix this.




OAuth consent screen

Choose how you want to configure and register your app, including your target users. You can only associate one app with your project.

User Type

☐ Internal 

Only available to users within your organization. You will not need to submit your app for verification.

☒ External 

Available to any user with a Google Account.



5. Go to the Google Sign In plugin page on github and download the latest release of the plugin.

<https://github.com/googlesamples/google-signin-unity>

You will also need to take the steps to Configure a Google API project:

Go to the section Building for iOS and follow the link...

<https://developers.google.com/identity/sign-in/ios/start-integrating>. Follow the instructions, making sure to have xcode, and pods if necessary, then Create an OAuth client ID for your google app.

Get an OAuth client ID

If you haven't already created an OAuth client ID, click the button below to do so.

Create an OAuth client ID




After you create the OAuth client ID, take note of the client ID string, which you will need to configure Google Sign-in in your app. You can optionally download the configuration file, which contains your client ID and other configuration data, for future reference.







If you already created an OAuth client ID, you can find your existing OAuth information by clicking the button below.

Get an existing OAuth client ID

6. Go back to your Google Cloud Platform project, and go under APIs and Services, and Credentials. You will notice there are two ids that have been auto generated by Google. Since this is IOS and Google doesn't accept custom URIs from an iOS device that is of type "WEB", the Client Id and secret you are looking to use is the first of the two, of type "IOS".

OAuth 2.0 Client IDs



<input type="checkbox"/>	Name	Creation date ↓	Type	Client ID	Usage with all services (last 30 days) ?	
<input type="checkbox"/>	OAuth client	Jan 28, 2020	iOS	1015055125254-p6j...	0	  
<input type="checkbox"/>	Web client (Auto-created for Google Sign-in)	Jan 28, 2020	Web application	1015055125254-51e...	0	  

Service Accounts

[Manage service accounts](#)

<input type="checkbox"/>	Email	Name ↑	Usage with all services (last 30 days) ?
No service accounts to display			






7. Go back to your braincloud app, and under Design > Core App Info > Application IDs, and Configure Platforms, select the Google Button and fill in the Google App Id and Google Client Id. You do not need a client secret for IOS google Authentication. Then save changes.

Core App Info - Application IDs

Show Game Design Features

✓ Save Changes

Configure Platforms

 Apple	 Facebook	 Google	 Steam	 Twitter
--	---	---	--	--

Google Service Account Email <input type="text"/>	Google Service Account p12 Certificate <div>Select a certificate file</div> <div>✗ Not Configured</div>
Google Package Name <input type="text"/>	
Google App ID <input type="text" value="1015055125254"/>	
Google Client ID <input type="text" value="1015055125254-p6j1sngbe2ui5mgneaakrkulo6cfemd4.apps.googleu"/>	
Google Client Secret <input type="text"/>	

8. Import the google signin unity package into your Unity Project. NOTE: you may run into a few errors upon importing the package. Research has mentioned that there are conflicting declarations in the Parse folder of the plugin. Navigate to it and delete the PARSE folder and its contents and the errors should go away.

```
[14:12:24] Assets/GoogleSignIn/Future.cs(72,40): error CS0433: The type 'TaskCompletionSource<T>' exists in both 'Unity.Tasks, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null' and 'netstandard, Version=2.0.0.0, Culture=neutral, PublicKeyToken=cc7b13ffcd2dd51'
[14:12:24] Assets/GoogleSignIn/GoogleSignIn.cs(117,12): error CS0433: The type 'Task<T>' exists in both 'Unity.Tasks, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null' and 'netstandard, Version=2.0.0.0, Culture=neutral, PublicKeyToken=cc7b13ffcd2dd51'
[14:12:24] Assets/GoogleSignIn/GoogleSignIn.cs(130,12): error CS0433: The type 'Task<T>' exists in both 'Unity.Tasks, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null' and 'netstandard, Version=2.0.0.0, Culture=neutral, PublicKeyToken=cc7b13ffcd2dd51'
```

9. Add Google code to your unity app

Key snippets of code:

```
using Google;
```

This gives you access to Google's services. You may also want "using System.Threading.Tasks;" because we will use a thread for the callback.

```
GoogleSignInConfiguration configuration;
```

This is the configuration of your authentication. You can customize fields to obtain important codes and information about the account that is authenticated to google. The email and the IdToken are what is needed in order to authenticate a google account with Braincloud. You can initialize it and set the

values.

```
configuration = new GoogleSignInConfiguration
{
    WebClientId = webClientId,

    RequestEmail = true,

    RequestIdToken = true,

    RequestAuthCode = true
};
```

Note that this can also be set on the fly instead of on Start or Awake. You just need to access the Configuration of your google instance like this:

```
GoogleSignIn.Configuration.RequestEmail = true;

GoogleSignIn.Configuration.RequestIdToken = true;

GoogleSignIn.Configuration.RequestAuthCode = true;
```

As an example you may have a function and callback like this:

```
public void OnGoogleSignIn()
{
    //set the google configuration to the configuration object you set up
    GoogleSignIn.Configuration = configuration;

    //Can define this if you're using the game signin
    GoogleSignIn.Configuration.UseGameSignIn = false;

    //Can also define these tags here like this.
    //GoogleSignIn.Configuration.RequestEmail = true;
    //GoogleSignIn.Configuration.RequestIdToken = true;
    //GoogleSignIn.Configuration.RequestAuthCode = true;
```



```

        // With the configuration set, its now time to start trying to sign in.
        Pass in a callback to wait for success.

        GoogleSignIn.DefaultInstance.SignIn().ContinueWith(OnGoogleAuthSignIn);
    }

    //use a callback with a task to easily get results from the callback in order to get
    the values you need.

    public void OnGoogleAuthSignIn(Task<GoogleSignInUser> task)
    {
        if (task.IsFaulted)
        {
        }
        else
        {
            authCode = task.Result.AuthCode;

            idToken = task.Result.IdToken;

            email = task.Result.Email;
        }
    }
}

```

10. Take the email result, and the IdToken result, and pass it into the braincloud
AuthenticateGoogleOpenId

```
AuthenticateGoogleOpenId(email, idToken, true, OnSuccess_Authenticate,
OnError_Authenticate);
```

11. When you build and run your app through xcode you may be required to generate a new signing certificate for your project find a tutorial online for doing this. It may also be easier to check the box Automatically manage signing under the signing section of your build in xcode.

12. You will now want to add your GoogleService-Info.plist file into your project. You can make a custom one or retrieve it from Firebase if you're linking the google app with it. If you do not want to use a GoogleService-Info.plist in the project you will need to edit the GoogleSignInAppController.mm file and pass in your web app id into the *clientId variable, make sure to comment out the *path and *dict variables as well.

13. Dont forget to add the URL Scheme into your xcode project as mentioned in this setup

<https://developers.google.com/identity/sign-in/ios/start-integrating>

14. You'll likely run into a problem with the google sign in plugin. It is due to a lack of support between google and xcode versions. The solution we found for our own tests was in this thread :

<https://github.com/googlesamples/google-signin-unity/issues/108>

The fix for us was to force the download of google sign in 4.4.0 in the xml file of the generated xcode project.

15. After all this, You should have no trouble signing into google and into braincloud with a google account.

*****For other languages and platforms, refer to google's documentation to properly setup your app to get google authentication working. With the new Google Open Id, BrainCloud only needs the google user's account email and the idToken that can be generated through google APIs in order to properly authenticate with a google account.*****