The Google Services Gradle Plugin

Introduction

As part of enabling Google APIs or Firebase services in your Android application you may have to add the **`google-services`** plugin to your **`build.gradle`** file:

```
dependencies {
    classpath 'com.google.gms:google-services:4.2.0'
    // ...
}
```

The **`google-services`** plugin has two main functions:

1. Process the **`google-services.json`** file and produce Android resources that can be used in your application's code. See Adding the JSON File (#adding_the_json_file) more information.
2. Add dependencies for basic libraries required for the services you have enabled. This step requires that the **`apply plugin: 'com.google.gms.google-services'`** line be at the **bottom** of your **`app/build.gradle`** file so that no dependency collisions are introduced. You can see the result of this step by running **`./gradlew :app:dependencies`**.

Adding the JSON File

The **`google-services.json`** file is generally placed in the **`app/`** directory (at the root of the Android Studio app module). As of version **`2.2.0`** the plugin supports build type and product flavor specific JSON files. All of the following directory structures are valid:

```
// dogfood and release are build types.
app/
    google-services.json
    src/dogfood/google-services.json
    src/release/google-services.json
    ...
```

Note: Providing a google-services.json file in the release directory allows you to maintain a separate Firebase project for your production APKs.

When product flavors are in use these more complicated directory structures are also valid.

```
// free and paid are product flavors.
app/
    google-services.json
    src/dogfood/paid/google-services.json
    src/release/free/google-services.json
    ...
```

Processing the JSON File

The `google-services.json` file has the following basic structure:

```
{
  "project_info": {...},
  "client": [...],
}
```

The `project_info` object contains general information about your project, while each member of the `client` array contains information about the clients (Android apps) that you have added to the project.

When processing the JSON file for your Android app, the plugin only uses the `client` object that matches your package name (for the current build type) based on the following logic:

- For each member of the `client` array:
  - Check the value of `client_info/android_client_info/package_name`
  - If the package name matches this value, return the member object.
- If none of the members of `client` match the package name, an exception is thrown.

For the rest of this document we will use `{YOUR_CLIENT}` to refer to the member of the `client` array determined by the procedure above.

The main result of the JSON processing is to produce two XML files which you can reference as Android resources in your Java code. Below is an example of each file:

app/build/generated/res/google-services/{build_type}/values/values.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <! -- Present in all applications -->
    <string name="google_app_id" translatable="false">1:1035469437089:android:73a4fb8297b2cd4f</string>

    <! -- Present in applications with the appropriate services configured -->
    <string name="gcm_defaultSenderId" translatable="false">1035469437089</string>
    <string name="default_web_client_id" translatable="false">337894902146-e4uksm38sne0bqrj6uvkbo4oiu4hvigl.apps.goo
    <string name="ga_trackingId" translatable="false">UA-65557217-3</string>
    <string name="firebase_database_url" translatable="false">https://example-url.firebaseio.com</string>
    <string name="google_api_key" translatable="false">AIzbSyCILMsOuUKwN3qhtxrPq7FFemDJUAXTyZ8</string>
    <string name="google_crash_reporting_api_key" translatable="false">AIzbSyCILMsOuUKwN3qhtxrPq7FFemDJUAXTyZ8</strin
    <string name="project_id" translatable="false">mydemoapp</string>

</resources>
```

app/build/generated/res/google-services/{flavor}/{build_type}/xml/global_tracker.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="ga_trackingId" translatable="false">UA-65557218-3</string>
</resources>
```

Every value in the XML files is present in the `google-services.json` file at the locations below. If your Android project has some configuration that prevents you from using the `google-services` plugin, you can safely recreate the XML files manually using these values:

google_app_id:

```
{YOUR_CLIENT}/client_info/mobilesdk_app_id
```

gcm_defaultSenderId:

```
project_info/project_number
```

default_web_client_id:

```
{YOUR_CLIENT}/oauth_client/client_id (client_type == 3)
```

ga_trackingId:

```
{YOUR_CLIENT}/services/analytics-service/analytics_property/tracking_id
```

firebase_database_url:

```
project_info/firebase_url
```

google_api_key:

```
{YOUR_CLIENT}/api_key/current_key
```

google_crash_reporting_api_key:

```
{YOUR_CLIENT}/api_key/current_key
```

project_id:

```
project_info/project_id
```

Troubleshooting

Q: When building I get the following error message: "File google-services.json is missing from module root folder. The Google Services Plugin cannot function without it".

A: The Firebase console (https://console.firebase.google.com) will help you download the `google-services.json`. In addition, the Quickstart guides for most APIs have instructions for generating this file. Once you have downloaded the `google-services.json` file, copy it into the `app/` folder of your Android Studio project, or into the `app/src/{build_type}` folder if you are using multiple build types.

Q: I can't find the symbol "R.string.gcm_defaultSenderId", "R.xml.global_tracker", etc.

A: Make sure the package name in your `build.gradle` file matches the package name you entered when creating the `google-services.json` file. If you are not sure, run through the getting started flow again and get a new json file.

Обновлено Январь 10, 2019