

Getting started with oauth2 for Google

Posted At : October 31, 2014 12:59 AM | Posted By : William Eatman

Related Categories: oauth2, coldfusion, Google Calendar

Github Project: <https://github.com/billeatman/oauth2-Examples>

For **CommandBox**: forgebox install oauth2-Examples

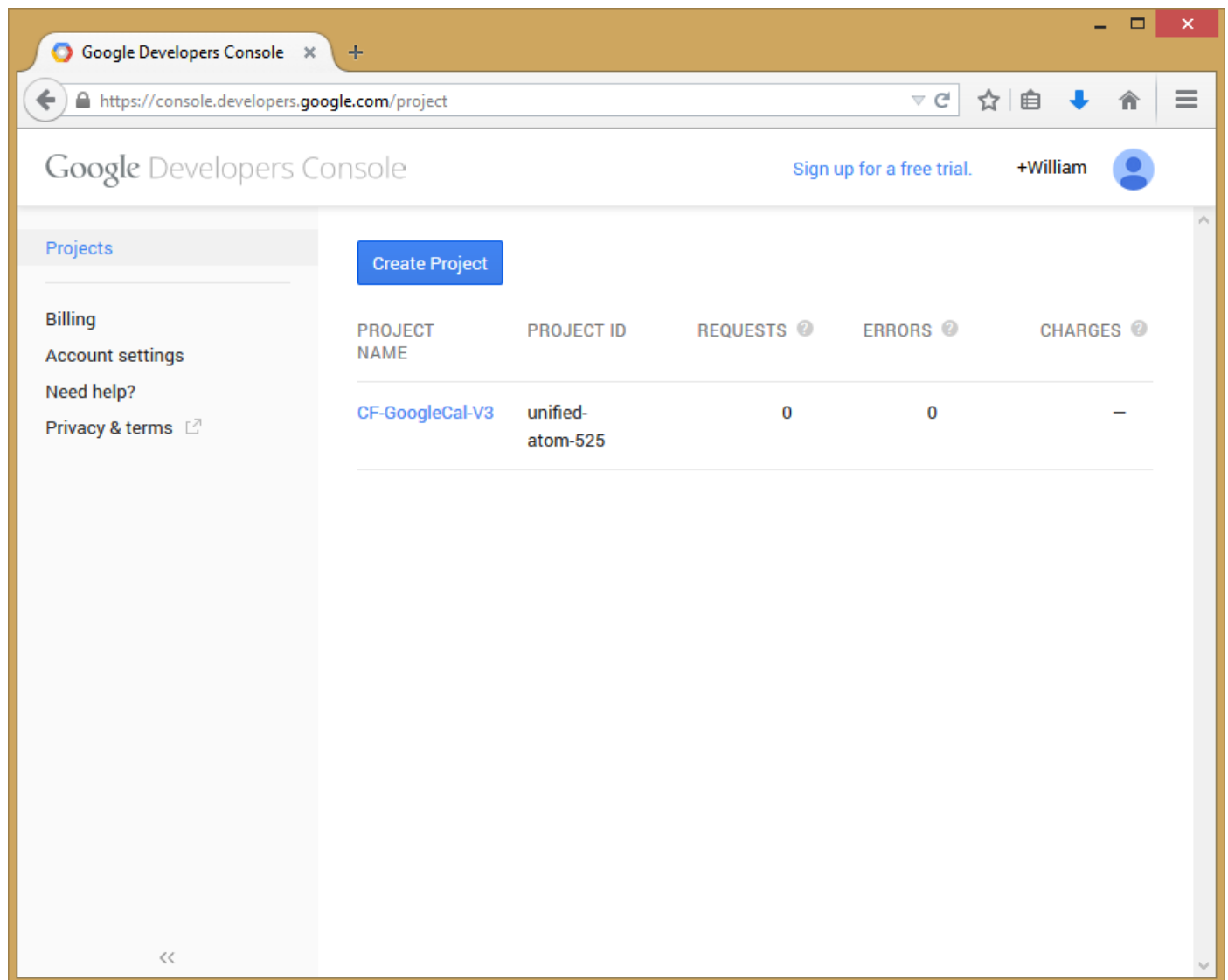
I'm going to focus on how to set up the oauth2 project to work with Google. It's a confusing process if you've never done it before.

If you are new to oauth2, I would first read through the docs on Google.

<https://developers.google.com/accounts/docs/OAuth2>

Let's get started!

Our first step is to create a project using the **Google Developers Console**.




I've created a project for my GoogleCal-V3 project. Now we can enable the APIs need for our application. I've enabled a few things for this project including the Calendar API.

The screenshot shows the Google Developers Console interface. The browser address bar displays the URL: `https://console.developers.google.com/project/unified-atom-525/apiui/api`. The page title is "Google Developers Console" and the user is logged in as "+William".

The left sidebar shows the navigation menu with the following sections:

- Projects
 - CF-GoogleCal-V3
 - Overview
 - Permissions
 - Billing & settings
- APIs & auth
 - APIs (selected)
 - Credentials
 - Consent screen
 - Push
- Monitoring
- Source Code
- Compute
- Storage
- Big Data
- Support

The main content area is titled "Enabled APIs" and includes the following text: "Some APIs are enabled automatically. You can disable them if you're not using their services." Below this is a table of enabled APIs:

NAME	QUOTA	STATUS
Analytics API	0%	ON
BigQuery API	0%	ON
Calendar API	0%	ON
Google Cloud SQL		ON
Google Cloud Storage		ON
Google Cloud Storage JSON API 		ON

Below the "Enabled APIs" section is the "Browse APIs" section, which includes a search filter: "Filter by API name or description". Below the filter is another table with the following columns: NAME, QUOTA, and STATUS.

NAME	QUOTA	STATUS
Ad Exchange Buyer API	1,000 requests/day	OFF

After enabling needed API, select "Credentials" then click the "Create new Client ID" button. You should see the dialog below. I've filled it in assuming that the oauth2 project is in a directory off of the root called oauth2.

Create Client ID

APPLICATION TYPE

Web application
Accessed by web browsers over a network.

Service account
Calls Google APIs on behalf of your application instead of an end-user.
[Learn more](#)

Installed application
Runs on a desktop computer or handheld device (like Android or iPhone).

AUTHORIZED JAVASCRIPT ORIGINS
Cannot contain a wildcard (http://*.example.com) or a path (http://example.com/subdir).

`http://localhost`

AUTHORIZED REDIRECT URIS
One URI per line. Needs to have a protocol, no URL fragments, and no relative paths. Can't be a non-private IP Address.

`http://localhost/oauth2/index.cfm`

[Create Client ID](#) [Cancel](#)

Now you can copy the newly generated `client_id` and `client_secret` from the Google Developer Console to the `application.cfc` in the `oauth2` project. In this example I'm using the scope for Google Calendar, but you can use whatever scope you want.

List of scopes: <https://developers.google.com/oauthplayground/>

```
<cffunction name="onApplicationStart"
    access="public"
    returntype="boolean"
    output="false"
    hint="Fires when the application is first created.">

    <!-- remove previous access and struct -->
    <cftry>
    <cfset application.oauth2.revokeAccess()>
    <cfcatch>
    </cfcatch>
    </cftry>

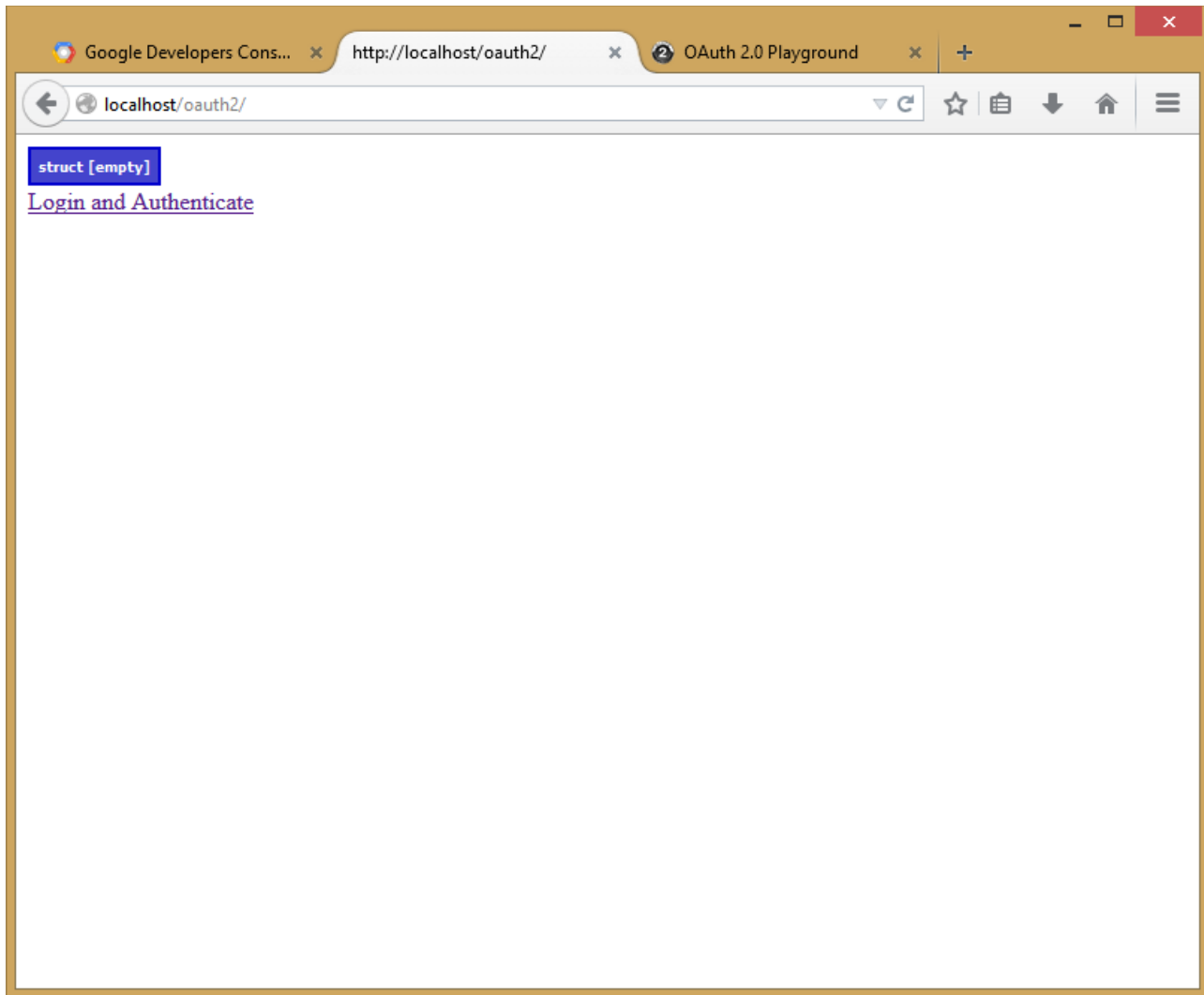
    <!-- CHANGE ME -->
    <!-- Fill in your own client_id, client_secret and scope -->
    <cfset application.oauth2 = new oauth2.oauth2(
        client_id      = '
        client_secret  = '
        redirect_uri   = 'http://localhost/oauth2/index.cfm',
        scope          = 'https://www.googleapis.com/auth/calendar',
        state          = '',
        access_type    = 'offline', <!-- Use offline for refresh tokens
        approval_prompt = 'force'
    ) />

    <cfreturn true />
</cffunction>
```

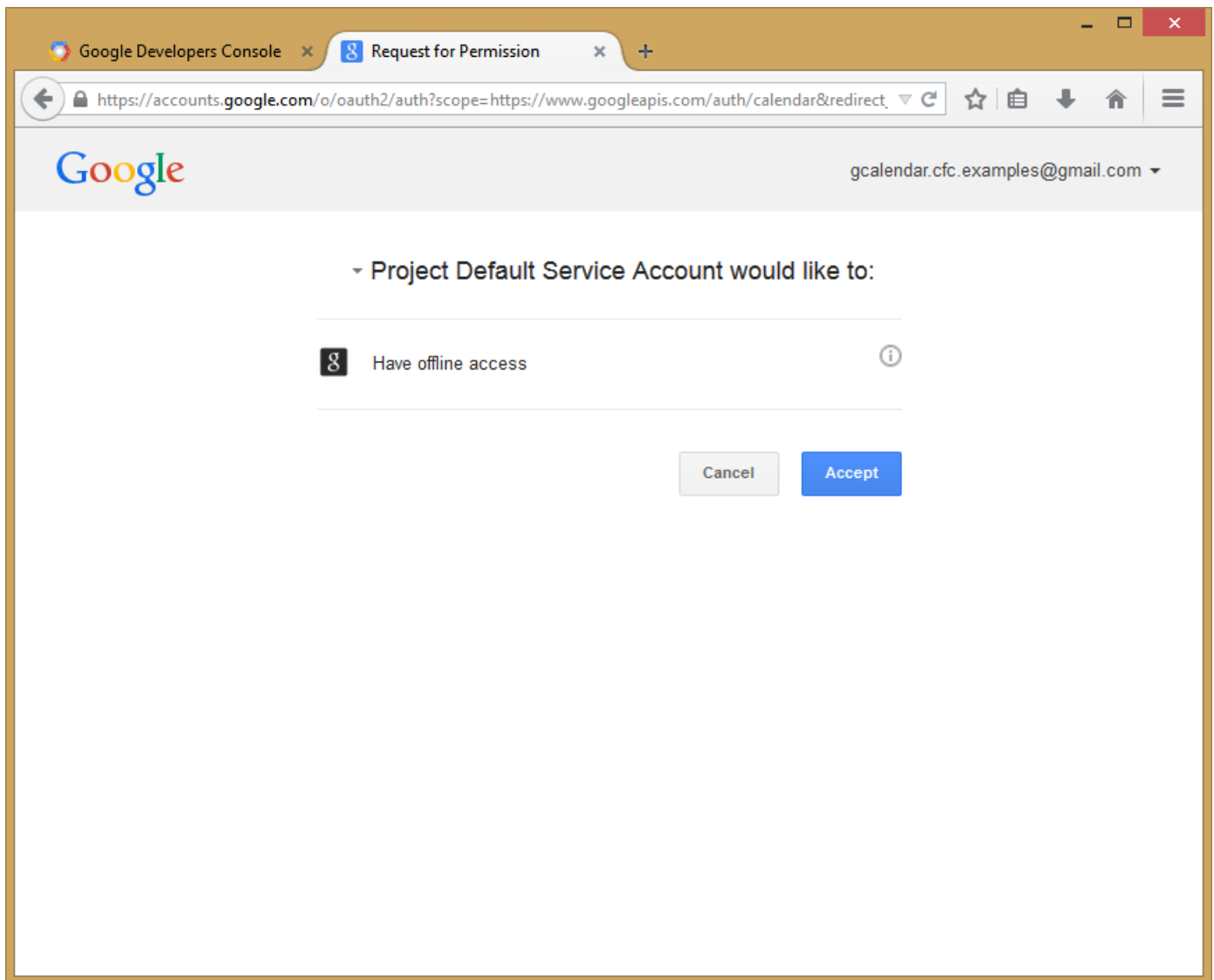
Once this is filled in, let's reinit the application.

<http://localhost/oauth2/?reinit=true>

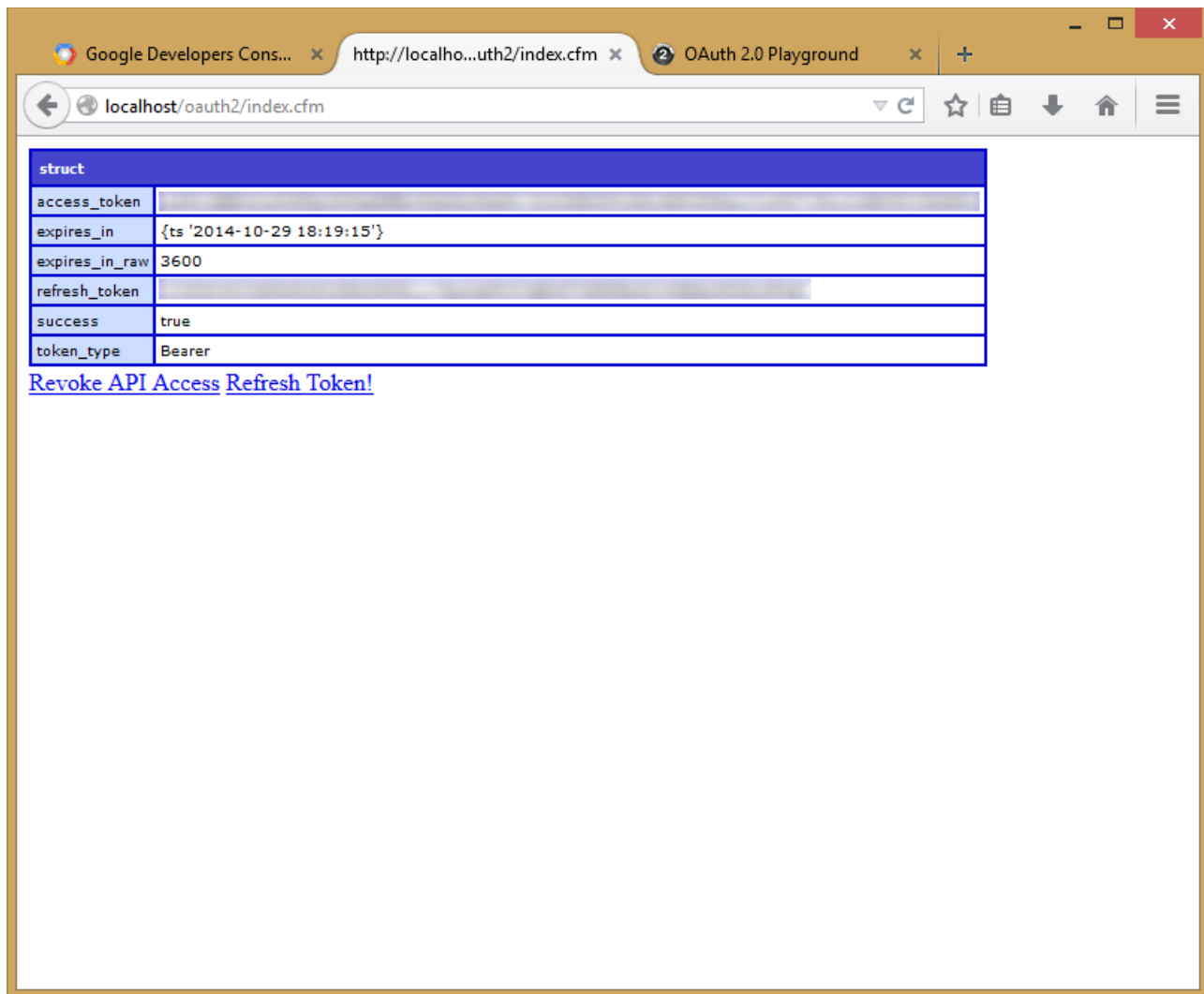
We should now be at the screen shown below.



Make sure at this point that your browser is logged into the Google account After clicking "Login and Authenticate"



After clicking "Accept", you should be at a screen that shows your access. :-)



The screenshot shows a web browser window with the address bar displaying `localhost/oauth2/index.cfm`. The page content is a table representing an OAuth 2.0 response structure. The table has a blue header row labeled "struct" and several data rows. Below the table, there are two blue links: "Revoke API Access" and "Refresh Token!".

struct	
access_token	[REDACTED]
expires_in	{ts '2014-10-29 18:19:15'}
expires_in_raw	3600
refresh_token	[REDACTED]
success	true
token_type	Bearer

[Revoke API Access](#) [Refresh Token!](#)

If anyone is interested in helping me finish the Calendar V3 project before the middle of next month, let me know.