How to setup OAuth2 Authentication with Google

Global Support
08 2016

Prerequisites

You will require the following items to set up OAuth2 with a Google account:

- Your server URL – public or private – is required. Examples https://www.my_server.com or https://MyServerName.

A Google account that will be used to administer the service. This can be done through following link: https://accounts.google.com/SignUp?continue=https%3A%2F%2Faccounts.google.com%2FManageAccount

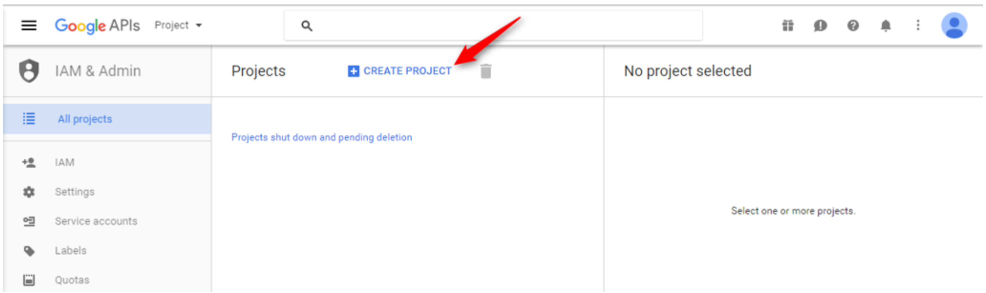
We will use the mock account *my_admin_account@gmail.com* in the following example.

- Select a name for your OAuth2 service. It must start with a letter (A-Z or a-z) followed by any combination of letters (A-Z or a-z), digits or underscores. In the following example, the name *MyAuth2* is used.
- **Oauth2** must be configured as a valid authentication method in your Sage X3 **nodelocal.js** file such as below:

```
session: {
    timeout: 30, // minutes
    checkInterval: 60, // seconds
    auth: ["basic", "oauth2"]
}
```

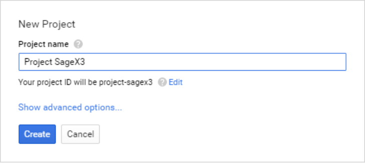
Step 1: Create a client ID

- Create a project through https://console.developers.google.com



No project selected

Select one or more projects.



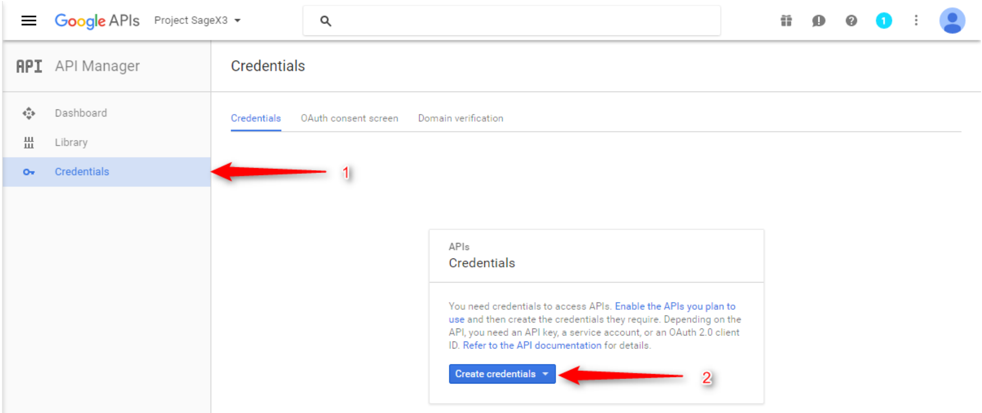
New Project

Project name

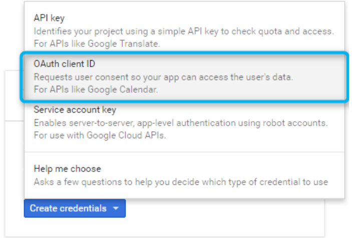
Project SageX3

Your project ID will be project-sagex3 Edit

- Go to **Credentials** and create new Credentials:



You need credentials to access APIs. Enable the APIs you plan to use and then create the credentials they require. Depending on the API, you need an API key, a service account, or an OAuth 2.0 client ID. Refer to the API documentation for details.



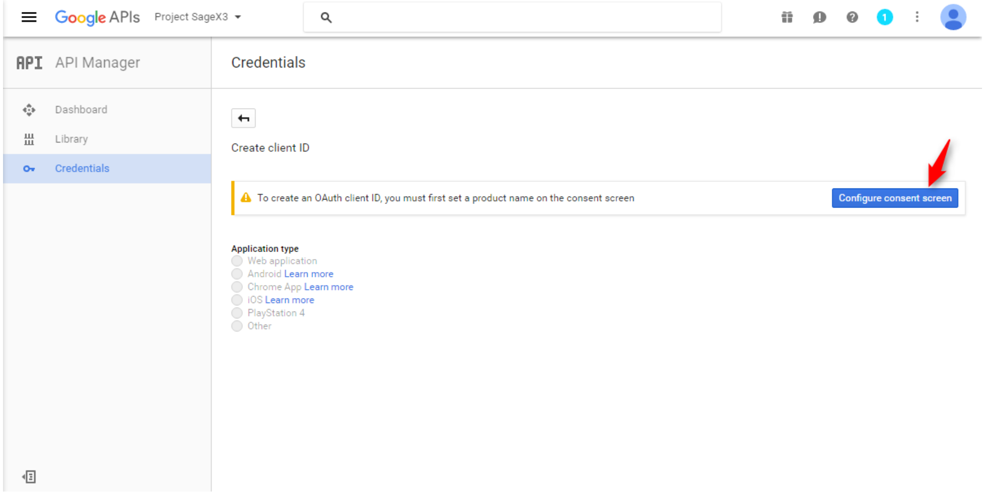
API key
Identifies your project using a simple API key to check quota and access.
For APIs like Google Translate.

OAuth client ID
Requests user consent so your app can access the user's data.
For APIs like Google Calendar.

Service account key
Enables server-to-server, app-level authentication using robot accounts.
For use with Google Cloud APIs.

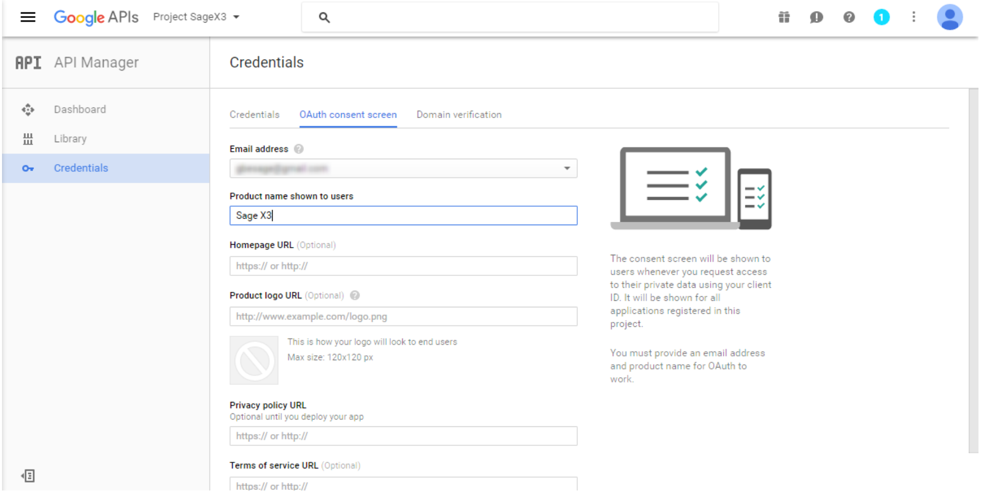
Help me choose
Asks a few questions to help you decide which type of credential to use

- The **Product Name** needs to be supplied the first time you create credentials (and only that time):



To create an OAuth client ID, you must first set a product name on the consent screen

Application type
○ Web application
○ Android Learn more
○ Chrome App Learn more
○ iOS Learn more
○ PlayStation 4
○ Other



Email address: [redacted]

Product name shown to users: Sage X3

Homepage URL (Optional): https:// or http://

Product logo URL (Optional): http://www.example.com/logo.png

This is how your logo will look to end users
Max size: 120x120 px

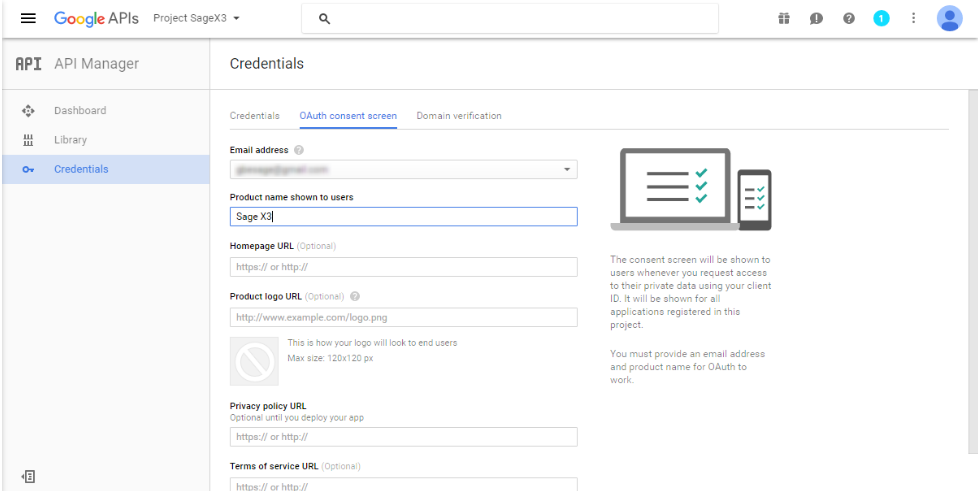
Privacy policy URL
Optional until you deploy your app
https:// or http://

Terms of service URL (Optional): https:// or http://

The consent screen will be shown to users whenever you request access to their private data using your client ID. It will be shown for all applications registered in this project.

You must provide an email address and product name for OAuth to work.

- When you save, you will be redirected to the following screen:



Email address [redacted]

Product name shown to users: Sage X3

Homepage URL (Optional): https:// or http://

Product logo URL (Optional): http://www.example.com/logo.png

This is how your logo will look to end users
Max size: 120x120 px

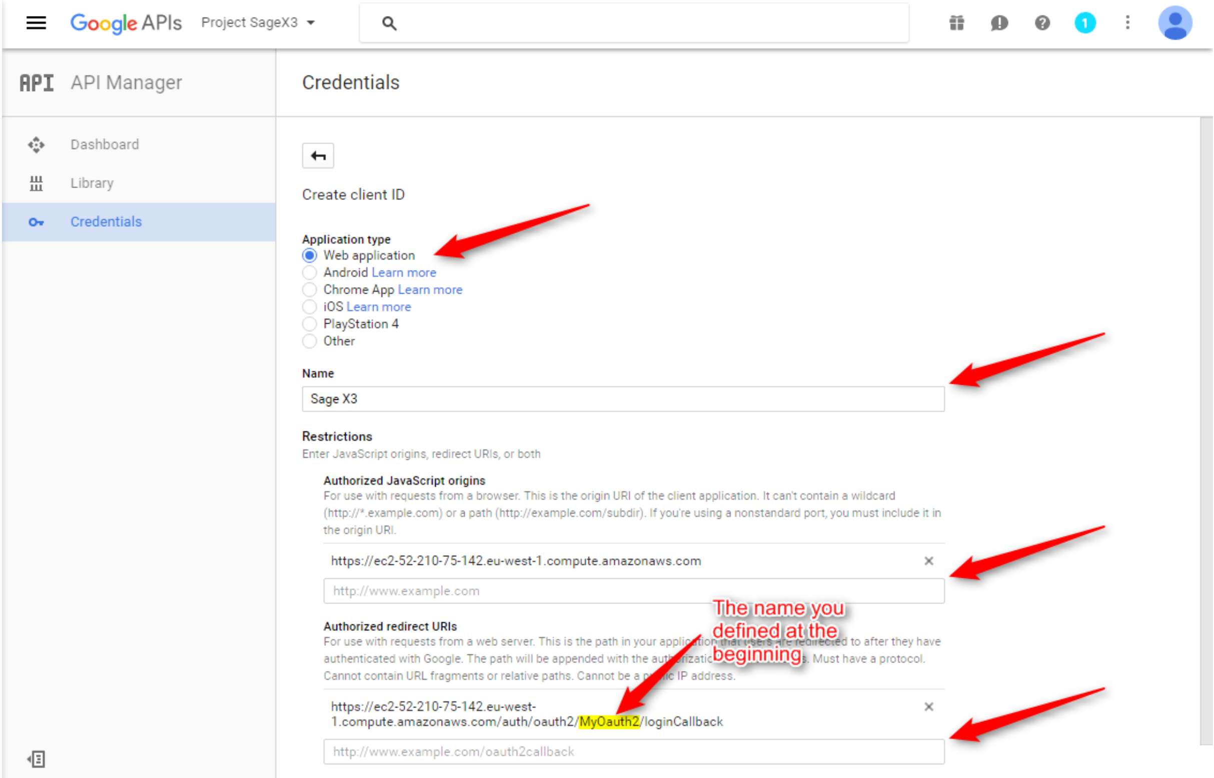
Privacy policy URL
Optional until you deploy your app: https:// or http://

Terms of service URL (Optional): https:// or http://

The consent screen will be shown to users whenever you request access to their private data using your client ID. It will be shown for all applications registered in this project.

You must provide an email address and product name for OAuth to work.

- Select **Web Application** and populate the **Name**, **Authorized JavaScript origins** and **Authorized redirect URLs**:



Create client ID

Application type
● Web application
○ Android Learn more
○ Chrome App Learn more
○ iOS Learn more
○ PlayStation 4
○ Other

Name: Sage X3

Restrictions
Enter JavaScript origins, redirect URIs, or both

Authorized JavaScript origins
For use with requests from a browser. This is the origin URI of the client application. It can't contain a wildcard (http://*.example.com) or a path (http://example.com/subdir). If you're using a nonstandard port, you must include it in the origin URI.

https://ec2-52-210-75-142.eu-west-1.compute.amazonaws.com ×

http://www.example.com

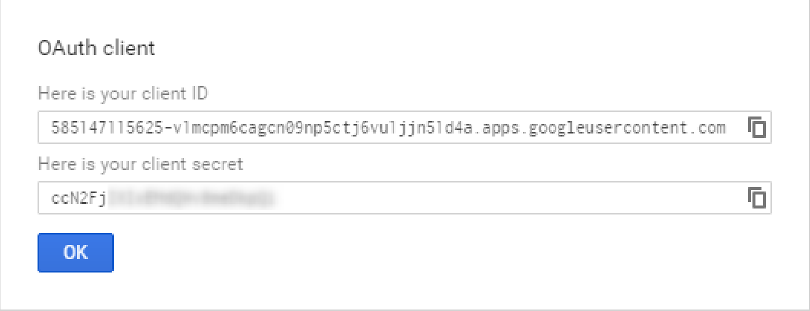
Authorized redirect URIs
For use with requests from a web server. This is the path in your application that users are redirected to after they have authenticated with Google. The path will be appended with the authorization code for access. Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.

https://ec2-52-210-75-142.eu-west-1.compute.amazonaws.com/auth/oauth2/MyOauth2/loginCallback ×

http://www.example.com/oauth2callback

The name you defined at the beginning

- Click **Create** and you will be given a ClientID and Secret, which you should store in a safe place (e.g. KeePass password manager), for reference later:



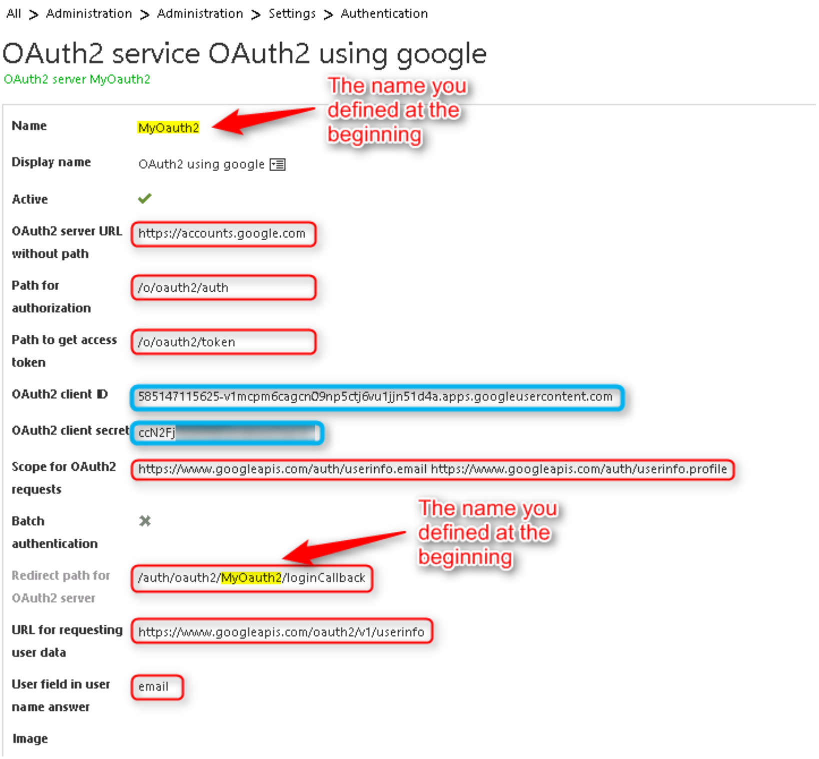
OAuth client

Here is your client ID
585147115625-v1mcpm6cagcn09np5ctj6vuljjn5ld4a.apps.googleusercontent.com

Here is your client secret
ccN2Fj[redacted]

Step 2: Create an Oauth2 service in the Sage X3 Web Server (Product Update 9)

- Go to Administration > Settings > Authentication > OAuth2 – Create a new OAuth2 service:



All > Administration > Administration > Settings > Authentication

OAuth2 service OAuth2 using google

OAuth2 server MyOauth2

Name	MyOauth2
Display name	OAuth2 using google
Active	✓
OAuth2 server URL without path	https://accounts.google.com
Path for authorization	/o/oauth2/auth
Path to get access token	/o/oauth2/token
OAuth2 client ID	585147115625-v1mcpm6cagcn09np5ctj6vu1jjn51d4a.apps.googleusercontent.com
OAuth2 client secret	ccN2Fj[redacted]
Scope for OAuth2 requests	https://www.googleapis.com/auth/userinfo.email https://www.googleapis.com/auth/userinfo.profile
Batch authentication	✖
Redirect path for OAuth2 server	/auth/oauth2/MyOauth2/loginCallback
URL for requesting user data	https://www.googleapis.com/oauth2/v1/userinfo
User field in user name answer	email
Image	

The name you defined at the beginning

The name you defined at the beginning

- The name of the service must match the service name chosen earlier (Prerequisites) exactly. The display name can be chosen freely.
- The frames highlighted in red above hold constant values for Oauth2 **Google** authentication. They must be populated exactly as shown in the picture above for any Google service:
  - **Oauth2 server URL without path**: https://accounts.google.com
  - **Path for authorization**: /o/oauth2/auth
  - **Path to get access token**: /o/oauth2/token
  - **Scope for Oauth2 requests**: https://www.googleapis.com/auth/userinfo.email https://www.googleapis.com/auth/userinfo.profile
  - **URL for requesting user data**: https://www.googleapis.com/oauth2/v1/userinfo
  - **Redirect path for Oauth2 server**: /auth/oauth2/MyOauth2/loginCallback
- The information in blue (**Oauth2 client ID** and **Oauth2 client secret**) corresponds to the client ID shown and to the password defined in the **Client ID for web application** section on the Google service site, on the page that shows the client ID.

Step 3: Link your users to their Google account

At this stage you are ready to enable oAuth2 authentication for all users. Follow the steps below to link users to their Google accounts:

- Ensure that both **oauth2** and **basic** are enabled in your nodelocal.js file.
- Edit the Global settings. Change the default authentication method to **oauth2**.
- Edit the admin user. Set his/her authentication method to **DB** (This is a safety net in case your auth2 configuration does not work - you will change it later).
- Edit a test user (other than admin). Set their email to a Google account for which you have valid credentials, your personal account for example.
- Log in with the test user. If you get an error, log in again as admin to fix the oauth2 configuration and try again.
- When the test is successful, log in again as admin, assign a Google account email to the admin user and change the admin user to use the default authentication method (oauth2).
- Check all user emails and edit them if necessary to match the user's Google account.
- Edit your nodelocal.js and only enable **oauth2**. Restart the Web server. Your server is now safely configured to authenticate all users, including Admin, with their Google accounts.

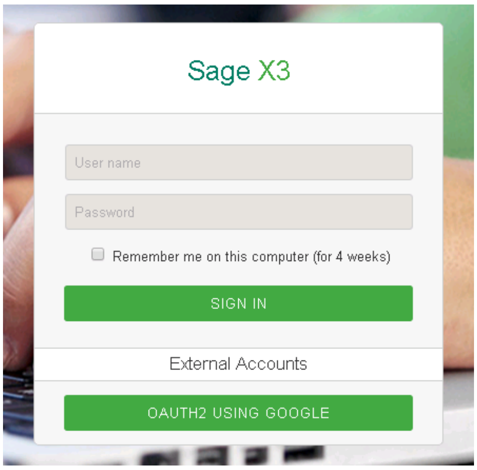
Note

We recommend you use an external identity service such as LDAP or oAuth2 *for all users*, including special users that support web service calls. Nevertheless, you will need to adapt your web service clients to authenticate with oAuth2

If you are using web services published by Sage X3, you can temporarily activate both **basic** and **oauth2** in your nodelocal.js file and configure the special web service users to use basic authentication. This will allow you to keep your web services in operation while you adapt them for oauth2. Once you have upgraded your web service clients you should edit nodelocal.js again and only enable **oauth2** to tighten security.

User connection

The login screen contains buttons for different authentication methods. The user must click on the button that shows the name of the service (as defined in the prerequisites above) in capital letters. There may be several buttons below the **External Accounts** headline, similar to the example below:



Sage X3

User name

Password

□ Remember me on this computer (for 4 weeks)

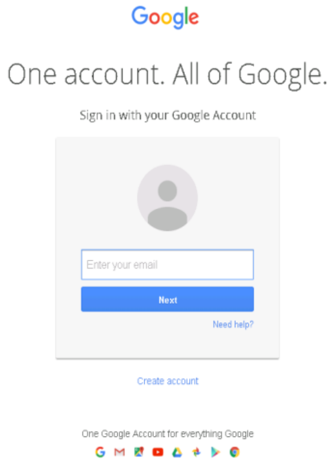
SIGN IN

External Accounts

OAUTH2 USING GOOGLE

A direct link can also be typed and/or set in your browser favorites to access Google authentication directly: http://www.my_server.com/auth/oauth2/MyOauth2/loginStart

When this is done, you are redirected to the Google site to authenticate using your Google account if this has not already been done. The page appears as follows:



Google

One account. All of Google.

Sign in with your Google Account

Enter your email

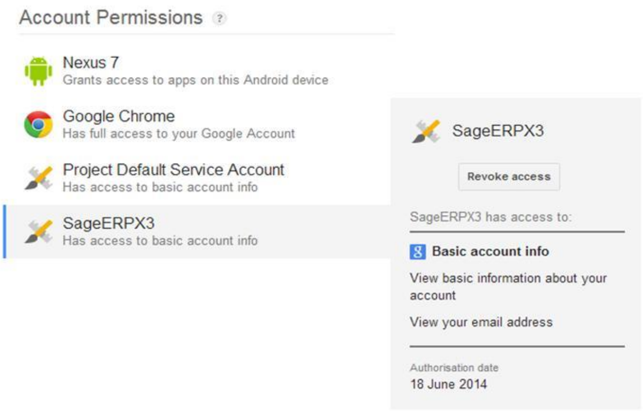
Next

Need help?

Create account

One Google Account for everything Google

You will be prompted by Google to allow Sage X3 to access the email address for that profile when logging in for the first time:



Account Permissions ?

Nexus 7
Grants access to apps on this Android device

Google Chrome
Has full access to your Google Account

Project Default Service Account
Has access to basic account info

SageERPX3
Has access to basic account info

SageERPX3

Revoke access

SageERPX3 has access to:

Basic account info

View basic information about your account

View your email address

Authorisation date
18 June 2014

Note: Your authentication will be valid until you log out of your Google account or clear your browser's cookies As a result Sage X3 may not need to prompt you to authenticate if your Google login is still valid.