Version 9.7

OAuth Guide

FLOWRIGHT

Contents

1 Welcome 3
1.1 What is oAuth? 3
1.2 How does OAuth work? 3
2 OAuth tokens 4
2.1 How to Get a Token 4
2.2 Making a REST API Call using the Token 5
2.3 Requesting a New Token using the Refresh Token 6
3 Configuration 8
4 Integrations 9
4.1 Using tokens for User Interface Authentication 9
4.2 Using Tokens for Microservice Authentication 10

1 Welcome

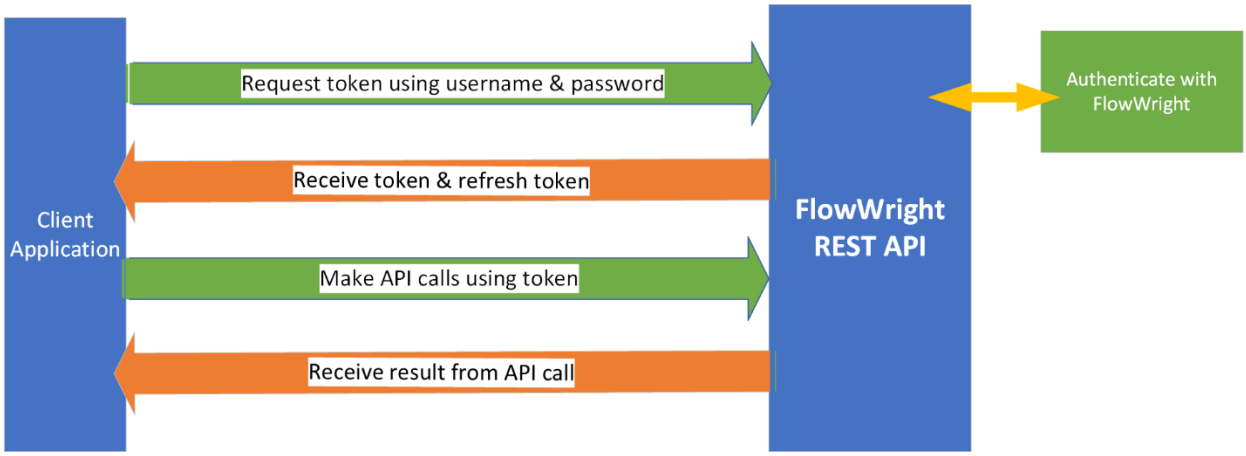
This is FlowWright's oAuth guide for v9.7. This guide describes how to configure and use FlowWright's OAuth.

1.1 What is oAuth?

OAuth is an open standard for access delegation, commonly used as a way for Internet users to grant websites or applications access to information without giving passwords.

1.2 How does OAuth work?

The diagram below illustrates the oAuth process:



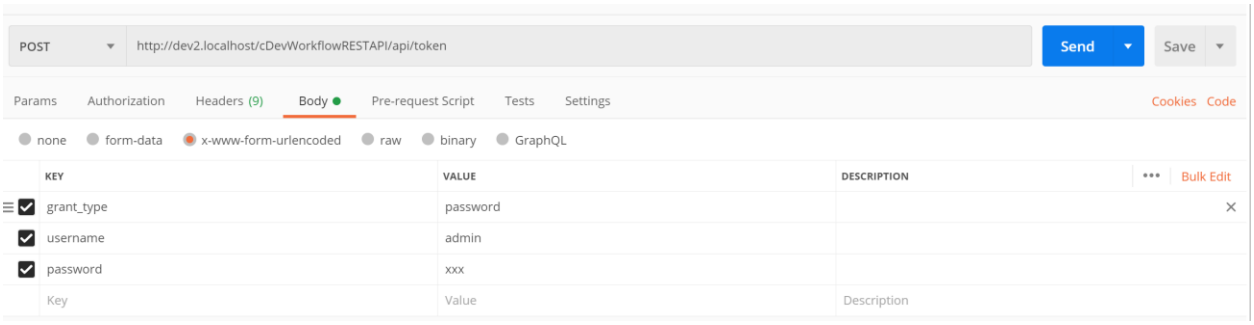
1. A client application authenticates with the FlowWright REST API using a user name and password.
2. If authentication is successful, FlowWright issues a token and a refresh token.
3. Using the token, calls can be made to the FlowWright REST API.
4. FlowWright's REST API will validate the token, process the API call and return a result.

2 OAuth tokens

2.1 How to Get a Token

The first step is to authenticate and receive a token from the REST API. Throughout this documentation, we will be using POSTMAN to demonstrate this. You may use another familiar tool if you wish. If you don't have POSTMAN, you can download it for free using the following link: https://www.postman.com/

Let's get started! In order receive a token, you must first perform a POST with user authentication information.



As shown in the diagram above, an HTTP POST request is sent to the REST API URL:

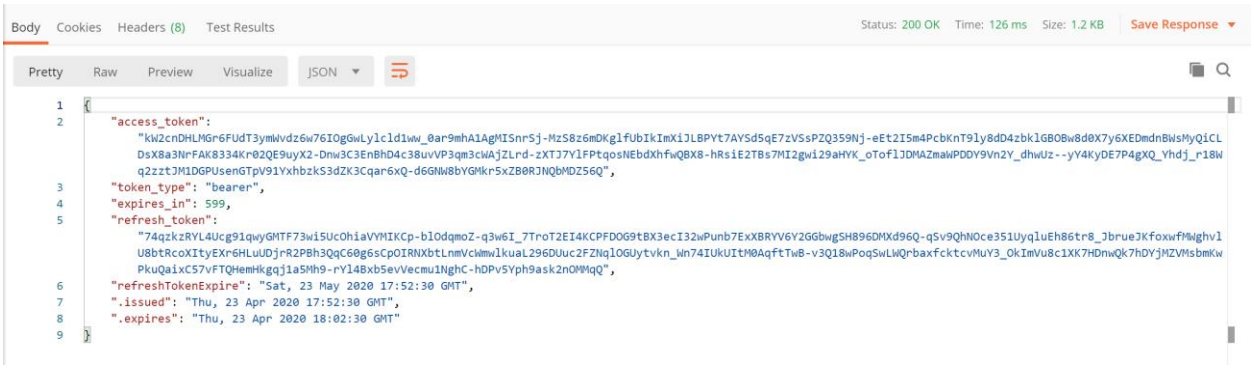
http://localhost/cDevWorkflowRESTAPI/api/token

Within the body of the request, 3 form elements are passed with their respective values:

- username – FlowWright user name
- password – FlowWright user password
- grant_type – password

Form body should be sent as "x-www-form-urlencoded" and then, after the request is sent, the REST API will authenticate against FlowWright security. If authentication is not successful for some reason, you will see a response such as that shown (below):

If the authentication is successful, then you will see a request with the information shown like (below):



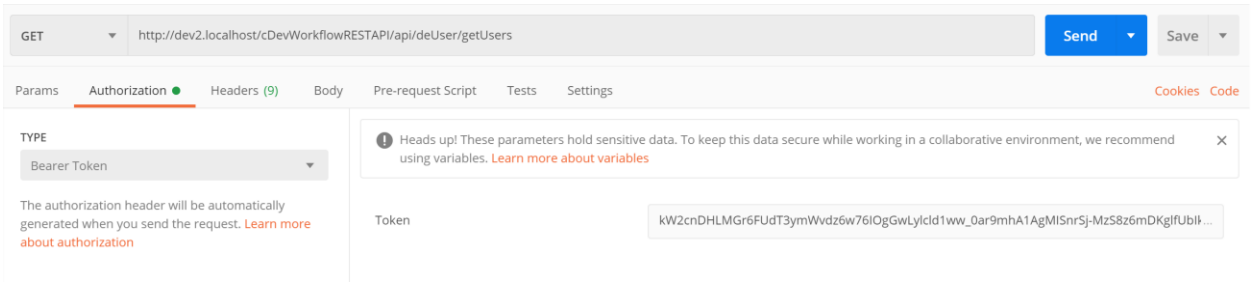
The response sent back is in JSON format and it contains the following information:

- access_token – a token for making REST API calls
- token_type – bearer type token
- expires_in – expiration in seconds
- refresh_token – refresh token for requesting tokens in the future
- refreshTokenExpire – expiration date/time of refresh token in UTC
- .issued – issued date/time of token in UTC
- .expires – expiration date/time of token in UTC

Once received, a token can be used to make any REST API calls during the period it is valid. In the above request, the token is valid for 599 seconds or 10 minutes, and the refresh token is valid for 1 month.

2.2 Making a REST API Call using the Token

Now that we have authenticated successfully, and have an OAuth token, we can use the token to make a REST API call. We will get the list of users using the REST API call "getUsers". Below is the request configuration within the POSTMAN application:



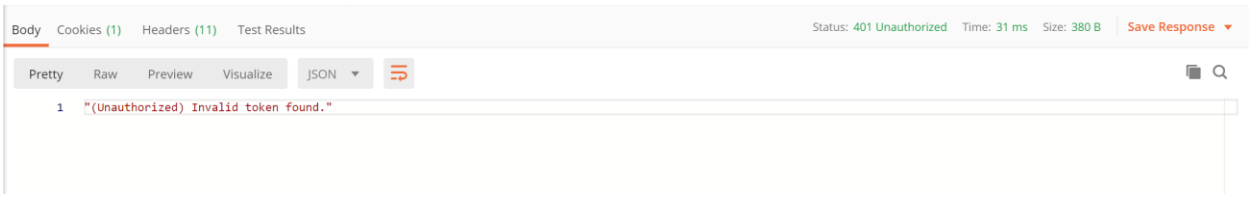
As show above, the HTTP method is set to GET and the REST API call URL is set to:

http://localhost/cDevWorkflowRESTAPI/api/deUser/getUsers

Under the "Authorization" section, select "Bearer Token" as the authentication type and set the token received from the previous call. Click the "Send" button to send the request to the server. FlowWright authentication will validate the token, and, if successful, will process the "getUsers" API call and return the response from the call.

As you can see above, the REST API successfully authenticated using the token and returned the response. The "getUsers" call returned a list of users in JSON format.

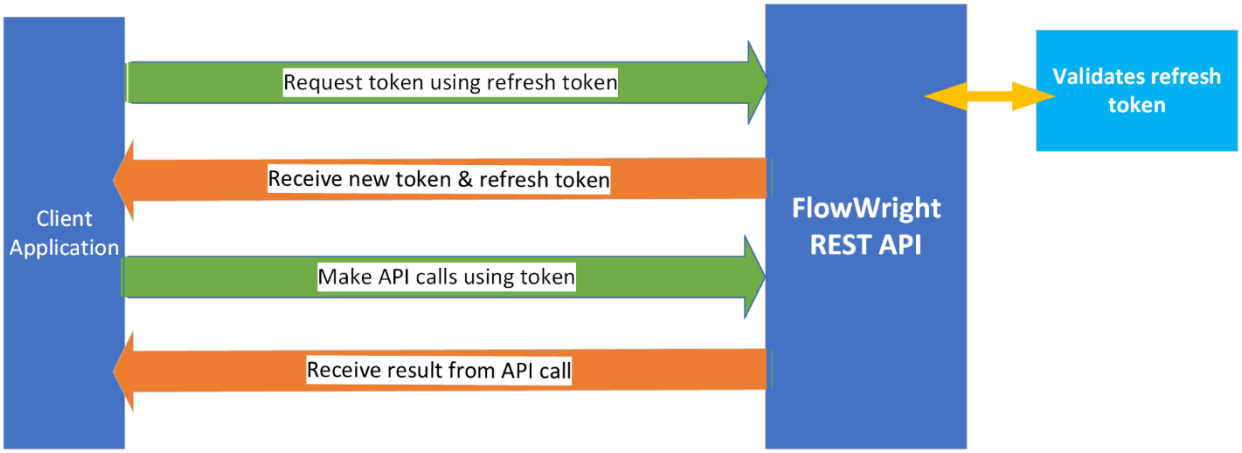
On the other hand, if the token was invalid, you would see the following message:



And, if an expired token was sent, you would see the following message:

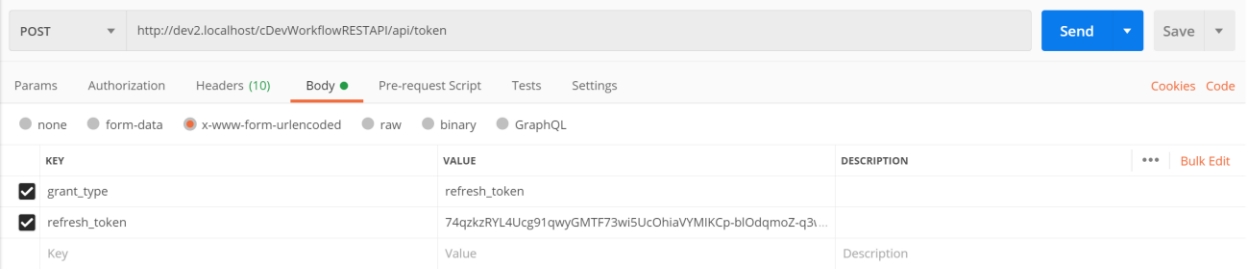
2.3 Requesting a New Token using the Refresh Token

In the case of a token having expired, a new token can be requested using "refresh token" functionality. The process for requesting a refreshed token is similar to what you have already see when requesting a new one. Below is the process for requesting a refreshed token:



As before, use the same token-requesting URL:

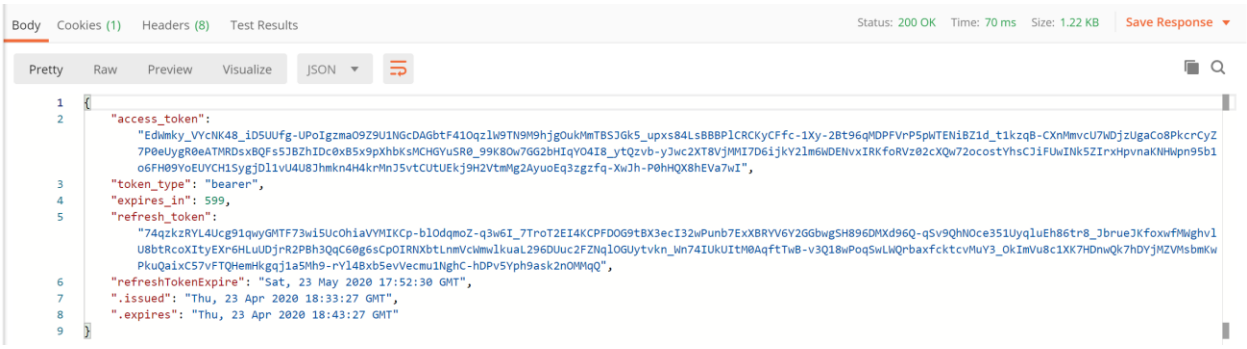
http://localhost/cDevWorkflowRESTAPI/api/token



As per the above diagram, an HTTP POST request is sent to get a new token. The form body has the following fields and values configured:

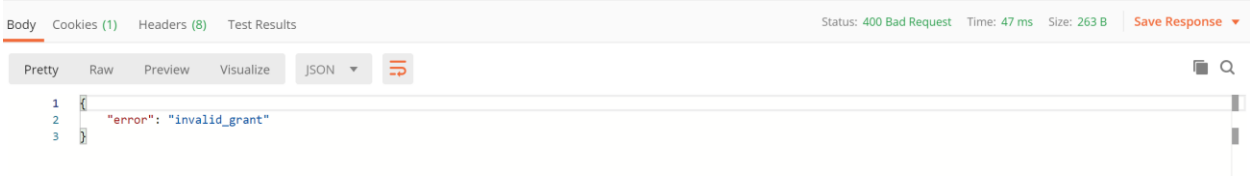
- grant_type – set the grant type to “refresh_token”
- refresh_token – pass the refreshed token received from the first token call

If the request for a refreshed token is successful, then the following response will come back:



The JSON response received has a new OAuth token with a new expiration date/time. The JSON response payload is in the same format as token request format. A successful response is only returned if the ***refreshed token is valid and not expired***.

The new access token can be now used for making REST API calls. If the “refresh token” request was invalid or expired, you would see the following response message:



Note: When requesting a new token, if the refreshed token is valid, the identical refreshed token will be provided again.

3 Configuration

Token request calls to the OAuth API return a JSON response. This JSON response contains 2 tokens and expirations in the UTC date/time format. Expiration timings for both tokens have the following default expiration values:

- token – 10 minutes or 600 seconds
- refresh token – 1 month

These are configuration values within the REST API's application settings file. Navigate to the following REST API directory:

C:\inetpub\wwwroot\cDevWorkflowRESTAPI

Open the file "web.config", you will see the following app settings keys and values:

```
<appSettings>
  <add key="webpages:Version" value="3.0.0.0" />
  <add key="webpages:Enabled" value="false" />
  <add key="ClientValidationEnabled" value="true" />
  <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  <add key="isMultiTenant" value="true" />
  <add key="tokenExpireMinutes" value="10" />
  <add key="refreshTokenExpireDays" value="30" />
</appSettings>
```

Use the key "**tokenExpireMinutes**" to set your own standard # of minutes to expire the token. Use the "**refershTokenExpireDays**" to set your own standard # of days to expire the refresh token. Once the above key values are changed, you will see token expirations have also changed.

4 Integrations

Now that you can request OAuth tokens through the REST API, you can use them to make API calls. And, importantly, these tokens can be also used to take better advantage of FlowWright capabilities.

4.1 Using tokens for User Interface Authentication

The FlowWright configuration manager user interface (UI) can be configured with many kinds of authentication, including, starting with v9.7, OAuth token authentication. To accomplish this, simply pass the token through the URL to the login page.

http://localhost/cDevWorkflow/LoginPage.aspx?token=XXXXXXXXXXXXXXX

Replace the "XXXXXXXXXXXXXX" with the name of your OAuth token. The FlowWright Configuration Manager UI will validate the token, and, if successful, authenticate the user and open the application UI.

For example, if you want to automatically login to the UI and display the "List of Tasks" page, then you can use the following URL:

http://localhost/cDevWorkflow/LoginPage.aspx?ReturnUrl=ConfigTasks.aspx&token=XXXXXXXXXXXXXXX

And, if you want to open the Workflow designer to a certain workflow definition and then display it, then you can use the following URL format:

http://localhost/cDevWorkflow/LoginPage.aspx?ReturnUrl=deDesignWorkflow.aspx?defID=PUT_YOUR_WF_DEFINITON_ID_HERE&token=XXXXXXXXXXXXXX

And, if you want to open the Forms designer to a specific form definition and display that definition, then you can use the following URL format:

http://localhost/cDevWorkflow/LoginPage.aspx?ReturnUrl=deFormDesignWorkflow.aspx?formDefID=PUT_YOUR_FORM_DEFINITION_ID_HERE&token=XXXXXXXXXXXXXX

And, further, if you want to render a Form instance, then you can use the following URL format:

http://localhost/cDevWorkflow/LoginPage.aspx?ReturnUrl=RenderForm.aspx?mode=Render&formInstID=PUT_YOUR_FORM_INSTANCE_ID_HERE&token=XXXXXXXXXXXXXX

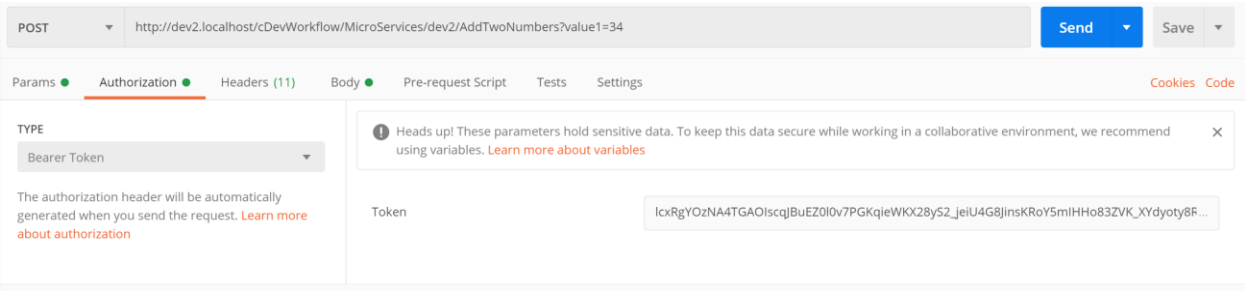
4.2 Using Tokens for Microservice Authentication

FlowWright microservices can be configured with or without authentication. If authentication is turned on, each microservice call made will require authentication. Microservices support two kinds of authentication: 1.) Basic authentication, and 2.) Token authentication.

Here's an example URL for a FlowWright microservice:

http://localhost/cDevWorkflow/MicroServices/dev2/AddTwoNumbers?value1=34

Below, is the request configuration show on POSTMAN:



As shown, an HTTP POST call is sent to the URL along with the authentication information: the authentication type is set to "Bearer Token" and the token value is set to a valid OAuth token.

Then, after the request is sent to the microservice, that FlowWright microservice reponds as shown: