Grant Types in OAuth 2.0

# Authorization Code

The **Authorization Code** grant type is used when the client wants to request access to protected resources on behalf of another user (i.e. a 3rd party). This is the grant type most often associated with OAuth.

**Use Cases**

* calls on behalf of a third party.

**Allowed Grant Types: response\_type** or **grant\_type**

**Example Request**

First, redirect the user to the following URL:

https://api.mysite.com/authorize?**response\_type=code**&client\_id=TestClient&redirect\_uri=https://myredirecturi.com/cb

A successful authorization will pass the client the authorization code in the URL via the supplied redirect\_uri:

https://myredirecturi.com/cb?**code=SplxlOBeZQQYbYS6WxSbIA&state=xyz**

Once this is done, a token can be requested using the authorization code.

$ curl -u TestClient:TestSecret https://api.mysite.com/token -d **'grant\_type = authorization\_code** &code=xyz'

A successful token request will return a standard access token in JSON format:

{

"access\_token": "03807cb390319329bdf6c777d4dfae9c0d3b3c35",

"expires\_in": 3600,

"token\_type": "bearer",

"scope": null

}

A more detailed view is given below.

|  |  |  |
| --- | --- | --- |
| **Admin Label** | **OAuth2 Parameter** | **Example Value** |
| Client ID | client\_id | ac\_client |
| Client Authentication | client\_secret | 2Federate |
| Allowed Grant Types | response\_type grant\_type | response\_type of "code" (code)  grant\_type of "authorization\_code" (code)  grant\_type of "refresh\_token" (refresh) |
| Redirect URIs | redirect\_uri | sample://oauth2/code/cb |
| Scope Settings | scope | edit |

Example

https://localhost:9031/as/authorization.oauth2?client\_id=ac\_client & response\_type=code  
  &scope=edit & redirect\_uri=sample%3A%2F%2Foauth2%2Fcode%2Fcb

# Implicit

The Implicit grant type is similar to the Authorization Code grant type in that it is used to request access to protected resources on behalf of another user (i.e. a 3rd party). It is optimized for **public** clients, such as those implemented in javascript or on mobile devices, where client credentials cannot be stored.

**Use Cases**

* calls on behalf of a third party
* for browser-based applications (javscript)
* for native applications (desktop and mobile devices)
* for any application where client credentials cannot be safely stored

Allowed Grant Types : **response\_type**

**Example Request**

First, redirect the user to the following URL:

https://api.mysite.com/authorize?**response\_type=token**&client\_id=TestClient&redirect\_uri=https://myredirecturi.com/cb

A successful token request will be returned in the fragment of the URL

https://myredirecturi.com/cb#**access\_token=2YotnFZFEjr1zCsicMWpAA&state=xyz&token\_type=bearer&expires\_in=3600**

A tabular view is given below

|  |  |  |
| --- | --- | --- |
| **Admin Label** | **OAuth2 Parameter** | **Example Value** |
| Client ID | client\_id | im\_client |
| Allowed Grant Types | response\_type | response\_type of "token" |
| Redirect URIs | redirect\_uri | sample://oauth2/code/cb |
| Scope Settings | scope | edit |

Example

https://localhost:9031/as/authorization.oauth2?client\_id=im\_client&response\_type=token&scope=edit&redirect\_uri=sample%3A%2F%2Foauth2%2Fimplicit%2Fcb

Response

This will initiate an authentication process using the browser (user agent). Once the user has authenticated and approved the authorization request they will be redirected to the configured URI with the access token included as a fragment of the URL. A refresh token will NOT be returned to the client.

sample://oauth2/implicit/cb#access\_token=zzz...yyy &token\_type=bearer &expires\_in=14400

# User Credentials (Resource Owner Password Credentials)

The User Credentials grant type (a.k.a. Resource Owner Password Credentials) is used when the user has a trusted relationship with the client, and so can supply credentials directly.

**Use Cases**

* when the client wishes to display a login form
* for applications owned and operated by the resource server (such as a mobile or desktop application)
* for applications migrating away from using direct authentication and stored credentials

**Allowed Grant Types** : **grant\_type**

**Example Request**

Send in the user credentials directly to receive an access token:

$ curl -u TestClient:TestSecret https://api.mysite.com/token -d **'grant\_type = password** & username = bshaffer&password=brent123'

A successful token request will return a standard access token in JSON format:

{

"access\_token": "03807cb390319329bdf6c777d4dfae9c0d3b3c35",

"expires\_in": 3600,

"token\_type": "bearer",

"scope": null

}

Tabular View

|  |  |  |
| --- | --- | --- |
| **Admin Label** | **OAuth2 Parameter** | **Example Value** |
| Client ID | client\_id | ro\_client |
| Client Authentication | client\_secret | 2Federate |
| Allowed Grant Types | response\_type grant\_type | grant\_type of "password" (ropc)  grant\_type of "refresh\_token" (refresh) |
| Scope Settings | scope | edit |

Example

POST https://localhost:9031/as/token.oauth2 HTTP/1.1  
Content-Type: application/x-www-form-urlencoded  
Authorization: Basic cm9fY2xpZW50OjJGZWRlcmF0ZQ==  
  
grant\_type=password & username=joe &password=2Federate &scope=edit

Client Credentials

The Client Credentials grant type is used when the client is requesting access to protected resources under its control (i.e. there is no third party).

**Use Cases**

* service calls
* calls on behalf of the user who created the client.

**Example Request**

# **using HTTP Basic Authentication**

$ curl -u TestClient:TestSecret https://api.mysite.com/token -d '**grant\_type=client\_credentials**'

# **using POST Body**

$ curl https://api.mysite.com/token -d 'grant\_type = client\_credentials & client\_id = TestClient &client\_secret=TestSecret'

A successful token request will return a standard access token in JSON format:

{

"access\_token": "03807cb390319329bdf6c777d4dfae9c0d3b3c35",

"expires\_in": 3600,

"token\_type": "bearer",

"scope": null

}

Tabular View

|  |  |  |
| --- | --- | --- |
| **Admin Label** | **OAuth2 Parameter** | **Example Value** |
| Client ID | client\_id | cc\_client |
| Client Authentication | client\_secret | 2Federate |
| Allowed Grant Types | grant\_type | grant\_type of "client\_credentials" |
| Scope Settings | scope | edit |

Example

The client makes a request (HTTP POST) to the token endpoint with the client credentials presented as HTTP Basic authentication:

POST https://localhost:9031/as/token.oauth2 HTTP/1.1  
Content-Type: **application/x-www-form-urlencoded**  
Authorization: Basic Y2NfY2xpZW50OjJGZWRlcmF0ZQ==  
  
grant\_type=client\_credentials &scope=edit

# Refresh the Token

If a refresh token was requested along with the access token, then the refresh token can be used to request a new access token without having to ask the user to re-authenticate. If the refresh token is still valid, then a new access token and refresh token will be returned to the client.

If the refresh token has been invalidated for any reason, then the client must require the user to re-authenticate to retrieve a new access token. The reasons for refresh tokens becoming invalid are:

* Refresh token has expired;
* Refresh token has been administratively revoked (separation / security reasons);
* User has explicitly revoked the refresh token

To refresh a token, the access token must have been requested with a grant type that supports refresh tokens (authorization code or resource owner password credentials). A request will then be made to the token endpoint with the grant\_type parameter set to "refresh\_token".

|  |  |  |
| --- | --- | --- |
| **Admin Label** | **OAuth2 Parameter** | **Example Value** |
| Client ID | client\_id | ac\_client |
| Client Authentication | client\_secret | 2Federate |
| Allowed Grant Types | response\_type grant\_type | response\_type of "code"  grant\_type of "authorization\_code"  grant\_type of "refresh\_token" |
| Redirect URIs | redirect\_uri | sample://oauth2/code/cb |
| Scope Settings | scope | edit |
| Refresh Token | refresh\_token | 123...789 |

**Example Request**

POST https://localhost:9031/as/token.oauth2 HTTP/1.1  
Content-Type: application/x-www-form-urlencoded  
Authorization: Basic YWNfY2xpZW50OjJGZWRlcmF0ZQ==  
  
grant\_type=refresh\_token &refresh\_token=123…789

**Response**

{  
"access\_token":"aaa…ccc",  
"token\_type":"Bearer",  
"expires\_in":14400,  
"refresh\_token":"456…321"  
}

# In Brief

* **Authorization Code** for apps running on a [web server](https://aaronparecki.com/oauth-2-simplified/#web-server-apps), [browser-based](https://aaronparecki.com/oauth-2-simplified/#browser-based-apps) and [mobile apps](https://aaronparecki.com/oauth-2-simplified/#mobile-apps)
* **Password** for logging in with a [username and password](https://aaronparecki.com/oauth-2-simplified/#other-app-types)
* **Client credentials** for [application access](https://aaronparecki.com/oauth-2-simplified/#other-app-types)
* **Implicit** was previously recommended for clients without a secret, but has been superceded by using the Authorization Code grant with no secret.

# References

<https://developer.pingidentity.com/en/resources/oauth-2-0-developers-guide.html>

<https://aaronparecki.com/oauth-2-simplified/>

<https://bshaffer.github.io/oauth2-server-php-docs/grant-types/client-credentials/>