OAuth Client Integration

> This documentation is intended for Clients and Third Party Services integrating with World Manager via OAuth 2.0.

Overview

World Manager supports integration through the OAuth 2.0 protocol based on the current OAuth 2.0 Authorization Framework RFC. For integration with World Manager using OAuth 2.0, it is necessary to have an understanding of the specification and how the protocol works.

The implementation used by World Manager only supports a limited set of Grants, Scopes and Accounts (by Roles) which are covered later in this document.

Terminology

For clarity, the terms related to OAuth and SSO used on this page refer to:

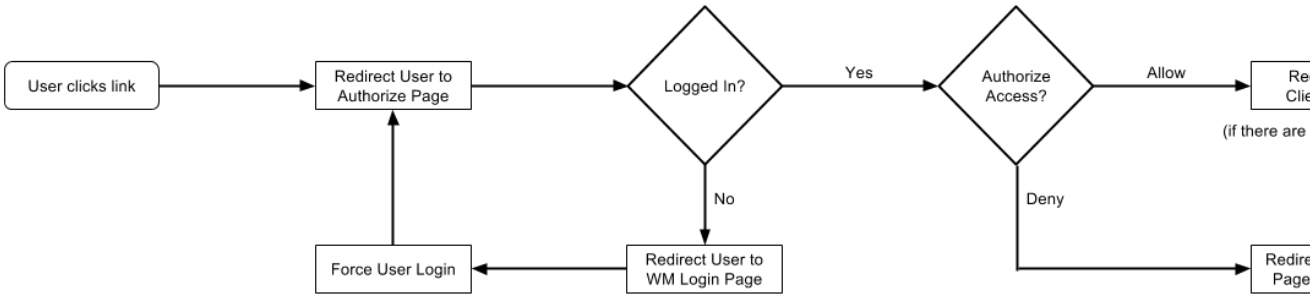
- User — the World Manager platform user (also known as Resource Owner)
- Client — the third-party website or service (e.g. external website wanting to use OAuth and SSO)
- Provider — also known as the Authorisation Server which facilitates access to the Client upon the User allowing it (i.e. World Manager)
- API — the data store for the User details to be accessed (also known as Resource Server)

Basic Flow

The basic flow from a User's perspective for OAuth when either allowing or denying is as follows:

1. User clicks link on World Manager platform or login link on Client
2. User is redirected* to the Authorize page
    a. if not logged in, User is redirected to World Manager login page** and forced to log in first
3. User is presented with the Authorize page
    a. User can allow access — “Authorise”; or
    b. User can deny access — “Not Now”
4. User is redirected back to the third-party website based on allow or deny
   (at this step the Client will use the received `authorizationCode` to generate an `AccessToken`)
    a. if allowed, the User should be logged in
    b. if denied, the User is shown the relevant error

** for outgoing clicks from a platform (e.g. clicking a menu link), the user is first redirect to the Client, then back to the Authorise page with the necessary parameters set*
*** the World Manager login page is for the corresponding platform of the User*



Client Integration

Client Details

Clients can be configured and managed via the Integration Clients page when logged in with a world-level account.

All the required details for OAuth integration with World Manager can be created and accessed through this tool.

For typical integration, the Client will need to utilise the `{client_id}` and `{client_secret}` which are automatically generated alpha-numeric strings to be used on the various requests made to the Provider and API (Resource Server). Additionally, the `{redirect_uri}` is required for Requesting an `AccessToken`, as it is a key part of the authorisation process.

> To avoid compromising the security of the User and the API (Resource Server), the `{client_secret}` should never be publicly exposed.

Details Fields

Within the Client Details page, the fields used for OAuth integration are shown below with a description of their usage.

Field	Is Required	Description
Name	**Required**	The name of the Client integrating. As an example, on Authorization Page, this will be shown like: *"Do you authorize **name** to access your platform ... "*
Authorization Message	Optional	This text will be displayed to users on the Authorization page. If specified, this message is shown under the list of Scopes available.
Scopes	**Required**	The Scopes which this client will have access to upon authorization. For further details, see the Supported Scopes section.
Redirect URL	**Required**	A link (also known as the {redirect_uri} which the User will be redirected to upon either Authorizing or Denying ('Not Now'). This URL is used as part of the security and validation checks for OAuth.
Homepage URL	Optional	A link to the website of the Client's service/website. If specified, this will show as a link with the 'Name' of the Client.
Privacy Policy URL	Optional	A link to the Privacy Policy of the Client's service/website. If specified, this will show as a link underneath the 'Authorize' and 'Not Now' buttons.
Terms Policy URL	Optional	A link to the Terms Policy of the Client's service/website. If specified, this will show as a link underneath the 'Authorize' and 'Not Now' buttons.

Supported Grants

Integration with OAuth can only be done via the Authorization Code Grant.

This means that the User will initially have to authorise with the Provider through a consent form prior to the Client being able to access the API (Resource Server) on the User's behalf.

Supported Scopes

The OAuth implementation currently only supports a single Scope. See the table below for the details of this Scope.

Scope	Resource Server URI	Description
`authentication`	`{platform-url}/api/oauth/authentication`	A value for whether or not the User is authenticated with the respective platform. The result is the User's UUID if successful. For a code sample, see Accessing API (Resource Server).

Supported Accounts

The OAuth implementation has been restricted to just regular accounts on the corresponding platform. This means that only accounts with roles such as World Manager, National Manager, Area Manager, General Manager, Store Manager and Employee accounts are allowed to access the Authorisation Page. Attempts to access the Authorisation Page from any other role will result in an error.

Beyond the initial authorisation and consent by the User, any subsequent requests to generate a new Access Token are restricted to User accounts which are active at the time of the requests. Attempts to generate a new Access Token for an inactive account will result in an unauthorized request error.

Any tokens associated with a User account are removed upon their deletion.

Authorisation Page

Users can consent to the Client by using the authorization page (URL specified below). As only the Authorization Code Grant is supported, this is the only means for a Client to gain access to the API (Resource Server).

Authorisation Page Endpoint

```
{platform-url}/oauth/authorize?response_type=code&client_id={client_id}&redirect_uri={redirect_uri}
```

The `{platform-url}` needs to be specified for the authorisation page to be accessible.

Parameters

The values for the parameters specified can be found on the corresponding Integration Clients page for the Client being integrated.

Name	Is Required	Description
`response_type`	**Required**	Response type for the request on the Authorisation Page. The string “code” is the only acceptable value for this parameter.
`client_id`	**Required**	The unique ID of the corresponding Client that is being integrated.
`redirect_uri`	Optional	Further information about it's usage is covered under the Redirect URI heading below.

Redirect URI

Specifying the `redirect_uri` is optional when providing the URL to the authorisation page.

If the `redirect_uri` is specified, it should match the Redirect URL specified on the Integration Clients page. However, it will fail in such cases where:

- the protocol doesn't match (e.g. https/http)
- the domain is different
- the path is different, for example:

- using `/callback.html` (Redirect URL in Integration Clients) and `/myCallback.html` (Redirect URI on authorisation page)
- however, using `/callback.html` (Redirect URL in Integration Clients) and `/callback.html?key1=value1` (Redirect URI on authorisation page) is fine

Such failures will result in the error page shown instead.

Access Tokens

Token Endpoint

```
{platform-url}/oauth/token
```

An `AccessToken` has a time to live of 12 hours. The expiry date of an `AccessToken` is updated each time it is renewed using a `RefreshToken`.

Upon the expiry of an `AccessToken`, the User will either need to re-authorise for a new `AccessToken` or the Client will need to request a new `AccessToken` using the `RefreshToken`. Requests to the API (Resource Server) with an expired token will yield an error.

Refresh Tokens

Token Endpoint

```
{platform-url}/oauth/token
```

A `RefreshToken` does not expire and can be used for any period beyond it's creation to generate a new `AccessToken` if the corresponding User account is still active.

Sample OAuth Requests

Request Parameters

```
platform_url="https://platform.worldmanager.com"
client_id="…"
client_secret="…"
```

Requesting an Access Token

Prior to being able to access the API (Resource Server), an `AccessToken` needs to be generated. This token is generated using the `authorizationCode` which would have been received upon the User authorising/consenting for the Client to be able to access their details based on the requested scopes.

This request can be done through CURL, with the parameters specified below the example. The `{platform-url}` needs to be specified for the token endpoint page to be accessible.

Request Parameters

```
request_uri="https://example.org/callbackUrl"
auth_code="…"
```

cURL Request

```
curl -X POST "$platform_url/oauth/token" \
     -u "$client_id:$client_secret"      \
     -d "grant_type=authorization_code"  \
     -d "redirect_uri=$redirect_uri"     \
     -d "code=$auth_code"
```

HTTPie Request

```
http -p b -a $client_id:$client_secret -f POST $platform_url/oauth/token
grant_type=authorization_code client_id=$client_id redirect_uri=$redirect_uri
code=$auth_code
```

JSON Response

```
{
   "access_token" : "023450f8-52ba-4fa2-8fb3-fec9f8d88465",
   "token_type"   :"bearer",
   "refresh_token : "cb27b2d9-76ca-41d9-b42a-56c6c6f57fc8",
   "expires_in"   : 43199,
   "scope"        : "authentication"
}
```

Parameters

The values for the parameters specified can be found on the corresponding Integration Clients page for the Client being integrated.

Name	Is Required	Description
`client_id`	**Required**	The Client ID, this is part of the access credentials.
`client_secret`	**Required**	The Client Secret, this is part of the access credentials.
`grant_type`	**Required**	Grant type for the token request. Value *must* be set to “authorization_code”.
`redirect_uri`	**Required**	Must be identical to the Redirect URI specified on the Authorisation Page if specified, otherwise identical to the Redirect URI specified for the Client.
`code`	**Required**	The authorization code received from the authorization server.

Possible Errors for Response

There are two different formats used for presenting errors resulting from an OAuth request.

The first set of responses are triggered by an invalid credentials (e.g. `{client_id}` or `{client_secret}`), whilst the latter is from other OAuth requests (e.g. invalid token).

Using Invalid Credentials

Request with an incorrect $client_id specified

```
{
   "timestamp" : "2015-08-06T07:11:06.546+0000",
   "status"    : 401,
   "error"     : "Unauthorized",
   "message"   : "Client with given clientId not found: s3c8a4b61812e6fe5",
   "path"      : "/oauth/token"
}
```

Request with an incorrect $client_secret

```
{
   "timestamp" : "2015-08-06T07:11:27.651+0000",
   "status"    : 401,
   "error"     : "Unauthorized",
   "message"   : "Bad credentials",
   "path"      : "/oauth/token"
}
```

See OAuth Error Codes for further details about the response format, codes and descriptions.

Requesting an Access Token using a Refresh Token

When an `AccessToken` has expired, a new token can be generated using the `RefreshToken` which would have been received upon creation of the `AccessToken`.

This request can be done through CURL using the parameters specified in the example below. The `{platform-url}` needs to be specified for the token endpoint page to be accessible.

Request Parameters

```
refresh_token="…"
```

URL Request

```
curl -X POST "$platform_url/oauth/token" \
     -u "$client_id:$client_secret"      \
     -d "grant_type=refresh_token"       \
     -d "refresh_token=$refresh_token"
```

JSON Response

```
{
   "access_token"  : "56cd3982-7133-4449-afcd-e1bb8c48f59f",
   "token_type"    : "bearer",
   "refresh_token" : "e9611989-29ad-4417-9db0-9743f409166c",
   "expires_in"    : 43199,
   "scope"         : "authentication"
}
```

Parameters

The values for the parameters specified can be found on the corresponding Integration Clients page for the Client being integrated.

Name	Is Required	Description
`client_id`	**Required**	The Client ID, this is part of the access credentials.
`client_secret`	**Required**	The Client Secret, this is part of the access credentials.
`grant_type`	**Required**	Grant type for the token request. Value *must* be set to “refresh_token”.
`refresh_token`	**Required**	The UUID of the Access Token token given by the Authorisation Server.

Typical Responses for Errors

There are two different formats used for presenting errors resulting from an OAuth request.

The first set of responses are triggered by an invalid credentials (e.g. `{client_id}` or `{client_secret}`), whilst the latter is from other OAuth requests (e.g. invalid token).

Using Invalid Credentials

Request with an incorrect $client_id specified

```
{
   "timestamp" : "2015-08-06T07:01:33.723+0000",
   "status"    : 401,
   "error"     : "Unauthorized",
   "message"   : "Client with given clientId not found: 3c8a4b61812e6fe5",
   "path"      : "/oauth/token"
}
```

Request with an incorrect $client_secret specified

```
{
   "timestamp" : "2015-08-06T07:01:43.866+0000",
   "status"    : 401,
   "error"     : "Unauthorized",
   "message"   : "Bad credentials",
   "path"      : "/oauth/token"
}
```

See OAuth Error Codes for further details about the response format, codes and descriptions.

Accessing API

Resource Server (API) Endpoint

```
$platform_url/api/oauth/authentication
```

This request can be done through CURL, with the parameters specified below the example. The `{platform-url}` needs to be specified for the API endpoint page to be accessible.

With the `AccessToken`, requests can be made to the API (Resource Server) which contains the details of the User based on the requested scopes on the consent page. A valid and non-expired `AccessToken` is required for such requests.

Request Parameters

```
access_token="…"
```

cURL Request

```
curl -X GET "$platform_url/api/oauth/authentication" -H "Authorization: Bearer
$access_token"
```

JSON Response

```
{
   "status" : "SUCCESS",
   "data"   : {
      "uuid" : "f3e7a670-e079-48fa-bf52-f267ad620695"
   },
   "messages"  : { },
   "timestamp" : "2015-08-06T05:03:38.080+0000",
   "url"       : "https://resource.example.com/api/oauth/authentication"
}
```

Parameters

The values for the parameters specified can be found on the corresponding Integration Clients page for the Client being integrated.

Name	Is Required	Description
`access_token`	**Required**	Response type for the request on the Authorisation Page. The string “code” is the only acceptable value for this parameter.

Possible Errors for Response

See OAuth Error Codes for further details about the response format, codes and descriptions.

OAuth Error Codes

Response Structure

OAuth requests which have resulted in an error will generally provide a response in the format indicated below.

```
{
   "error" : "error type …",
   "error_description" : "error description …"
}
```

Possible Responses

The status, error, description and notes of the errors which may be encountered are defined below.

HTTP Status	OAuth Error	Error Description	Notes
`400`	`unsupported_grant_type`	Unsupported grant type: `{grant_type}`	*An invalid Grant Type was specified:* • *if requesting `AccessToken`, only "authorization_code" is supported for this particular request.* • *if request `RefreshToken`, only “refresh_token” is supported for this particular request.*
`400`	`invalid_grant`	Invalid authorization code: `{authorization_code}`	*The specified Authorization Code is not valid or has already been used.*
`400`	`invalid_grant`	Redirect URI mismatch	*The specified Redirect URI does not match the Redirect URI associated with the Client.*
`400`	`invalid_grant`	Invalid refresh token: `{token.uuid}`	*A Refresh Token with the given UUID cannot be found.*
`400`	`invalid_request`	An authorization code must be supplied	*The Authorization Code parameter was not specified.*
`401`	`invalid_client`	Given client ID does not match authenticated client	*The Client associated with the request doesn't match the Authenticated Client.*
`401`	`invalid_account`	User with the given UUID was not found: `{user.uuid}`	*The User associated with the RefreshToken cannot be found or was deleted.*
`401`	`inactive_account`	User with the given UUID is not active: `{user.uuid}`	*The User associated with the RefreshToken is no longer active.*
`401`	`inactive_account`	User with the given UUID is not active (password): `{user.uuid}`	*The User associated with the RefreshToken has an “expired” password.*
`401`	`invalid_token`	Invalid access token: `{token.uuid}`	*An Access Token with the given UUID cannot be found.*

`401`	`invalid_token`	Access token expired: `{token.uuid}`	*The Access Token with the given UUID has expired.*
`401`	`unauthorized`	Full authentication is required to access this resource	*No valid bearer (access token) was specified.*