OM
Objektmanagement

OSC OAuth 2.0
API

OM domizil+

Benutzerhandbuch

Inhaltsverzeichnis

1. Allgemeines

Das d+ OSC verwendet das **OAuth 2.0 Protokoll** für Authentifizierung und Autorisierung. Client-Anwendungen müssen registriert werden und können dann einen Access-Token anfordern. Mit diesem Access-Token kann auf APIs zugegriffen werden.

Das d+ OSC unterstützt den **Authorization Code Flow** für Interaktive zur Anforderung von Access-Tokens und den **Client Credentials Flow** für nicht interaktive, Maschine zu Maschine, Anforderung.

2. Client-Registrierung

Für die Registrierung kontaktieren Sie bitte die CP Solutions GmbH. Zu diesem Zeitpunkt werden folgende Daten ausgetauscht.

client_id

Dies ist eine 36-stellige UUID zur Identifikation des Clients.

RFC 6749 2.3.1

client_secret

Dies ist ein zufällig generierter String zur Authentifizierung des Clients.

Redirection URI

(Für Authorization Code Flow)

Dies ist ein URI, an welchen die Benutzer nach der Autorisierung weitergeleitet werden.

scope

Welche scopes der Client Anfordern darf.

Grant Type

Welcher Flow erlaubt wird, entweder authorization_code oder client_credentials.

Name

Anzeigename für den OAuth Client

Tenants

Das d+ OSC verwendet einen eigenen Hostnamen für jeden Tenant. Clients müssen für jeden Tenant registriert werden, auf welchen zugegriffen werden soll. Eine Clientregistrierung kann bei mehreren Tenants verwendet werden.

Beim Zugriff auf die API muss der Hostname des Tenants verwendet werden.

Der Platzhalter <hostname>, der in diesem Dokument verwendet wird, muss durch den Hostnamen des Tenants ersetzt werden, wenn der **OAuth 2.0 Flow** durchgeführt wird.

3. OAuth Access Tokens anfordern über Authorization Code Flow

RFC 6748 4.1

Schritt 1: Autorisierungsanfrage

Der Client generiert eine Autorisierungsanfrage und schickt den User-Agenten des Benutzers zum „Authorization Endpoint“:

```
https://<hostname>/login/oauth/authorize
```

Für die Autorisierung werden folgende Query-Parameter unterstützt:

client_id	Required Die Client-ID der Anwendung
redirect_uri	Optional Die Redirection-URI, zu welcher der Benutzer nach der Autorisierung weitergeleitet wird. Wenn eine Redirection-URI angegeben wird, müssen *Scheme*, *Hostname*, *Port* und *Path* mit jener Redirection-URI übereinstimmen, welche bei der Client-Registrierung vereinbart wurde. Wird keine Redirection-URI angegeben, wird in weiterer Folge jene Redirection-URI verwendet, welche bei der Client Registrierung vereinbart wurde.

response_type	Required Es muss der Wert `code` verwendet werden
scope	Required Dies ist eine Space-delimeted-Liste von scopes, auf welcher der Zugriff angefordert wird (z. B. `profile email`)
access_type	Optional `online` (default): Es wird nur ein Access-Token erzeugt `offline`: Es werden ein Access-Token und ein Refresh-Token erzeugt. Mit dem Refresh-Token kann ein neuer Access-Token angefordert werden, ohne dass der Benutzer die Autorisierung wiederholen muss.
state	Optional Dies ist ein String, mit dem die Anwendung den Zusammenhang zwischen Autorisierungsrequest und Autorisierungsantwort beibehalten kann. Der state-Wert wird beim Weiterleiten des Benutzers nach der Autorisierung als Query-Parameter wieder mit exakt dem gleichen Wert an die Redirection-URI angehängt.

Schritt 2: Weiterleiten des Benutzers zum Authorization Endpoint

Der Benutzer wird an die URI, welchen in **Schritt 1: Autorisierungsanfrage** generiert wurde, weitergeleitet.

Beispiel:

```
https://<hostname>/login/oauth/authorize?client_id=client_id&scope=profile%20email&response_type=code&redirect_uri=https%3A//oauth2.example.com/code&state=state&access_type=offline
```

Schritt 3: Benutzer autorisiert die Anwendung

Der Benutzer muss die Anwendung im d+ OSC autorisieren.

Fehler

Falls der Client nicht identifiziert werden kann, wird dem Benutzer eine Fehlermeldung im d+ OSC angezeigt. Es erfolgt keine Weiterleitung an die Redirection-URL. Damit der Client identifiziert werden kann, muss eine gültige `client_id` übergeben werden. Wird der optionale `redirect_uri` -Parameter verwendet, muss dieser ebenfalls gültig sein.

Schritt 4: Benutzer wird zum Client zurückgeschickt

Nachdem der Benutzer die Anwendung autorisiert hat, wird dieser zur Redirection-URI weitergeleitet. Die Redirection-URI wird mit den folgenden Query-Parametern erweitert:

code	Dies ist ein Autorisierungscode, der vom Client eingelöst werden muss, um einen Access-Token und optional einen Refresh-Token zu erhalten. De Autorisierungscode kann nur einmal verwendet werden und hat eine maximale Gültigkeit von 10 Minuten.
state	Optional Der State-Wert, der in Schritt 1 übergeben wurde)

Fehler

Wenn die Autorisierungsanfrage fehlerhaft ist oder vom Benutzer abgebrochen wird, wird dieser mit einem Fehlercode laut RFC 6749 4.1.2.1 zur Redirection-URI weitergeleitet.

Schritt 5: Autorisierungscode gegen Access-Token tauschen

RFC 6749 4.1.3

Der Client tauscht den Autorisierungscode aus **Schritt 4: Benutzer wird zum Client zurückgeschickt** gegen einen Access-Token und optional einen Refresh-Token.

Token-Endpoint

`https://<hostname>/DPlus.OSCCloud.Rest.Server/api/v1/oauth/token`

Der Client schickt einen HTTP-Post-Request an den Token-Endpoint im Format „`application/x-www-form-urlencoded`“. Der Request muss die folgenden Parameter im Body enthalten:

client_id	die Client-ID der Anwendung
client_secret	das Client-Secret, welches bei der **Client-Registrierung** vereinbart wurde
code	der Autorisierungscode aus **Schritt 4: Benutzer wird zum Client zurückgeschickt**
grant_type	muss den Wert `authorization_code` enthalten
redirect_uri	die exakte Redirection-URI, welche in **Schritt 1: Autorisierungsanfrage** angegeben wurde (Wurde in **Schritt 1: Autorisierungsanfrage** keine Redirection-URI angegeben, muss die exakte Redirection-URI verwendet werden, die bei der **Client-Registrierung** vereinbart wurde.)

Antwort

Es wird ein JSON-Objekt mit folgenden Feldern zurückgegeben:

access_token	Access-Token, mit dem auf die d+ OSC APIs zugegriffen werden kann
expires_in	die Gültigkeitsdauer des Access-Tokens in Sekunden (Access-Tokens haben eine Gültigkeit von zwei Stunden (7200))
refresh_token	Wenn in **Schritt 1: Autorisierungsanfrage** der Parameter `access_type=offline` verwendet wurde, wird ein Refresh-Token zurückgegeben, mit dem ein neuer Access-Token angefordert werden kann.
scope	eine Space-delimeted-Liste von scopes, auf die der Zugriff gewährt wurde
token_type	der Typ des Access-Tokens (ist immer der Wert `Bearer`)

Fehler

Im Fehlerfall wird ein HTTP 400 Fehler (Bad Request) generiert mit einem JSON-Objekt im Body der Antwort mit folgendem Feld:

error	Error-Code laut RFC 6749 5.2

Access-Token aktualisieren

Wenn ein Refresh-Token über den `access_type=offline` in **Schritt 1: Autorisierungsanfrage** angefordert wurde, kann dieser Refresh-Token verwendet werden, um einen neuen Access-Token anzufordern.

Refresh-Tokens haben eine unbegrenzte Gültigkeit und können einmalig verwendet werden. Benutzer können die Autorisierung der Anwendung löschen. In diesem Fall verlieren alle Refresh-Tokens des Benutzers ihre Gültigkeit und die Autorisierung muss erneut durchgeführt werden.

Wenn ein Refresh-Token verwendet wird, um einen neuen Access-Token anzufordern, wird auch ein neuer Refresh-Token erstellt.

Token-Endpoint

`https://<hostname>/DPlus.OSCCloud.Rest.Server/api/v1/oauth/token`

Der Client schickt einen HTTP-Post-Request an den Token-Endpoint im Format „`application/x-www-form-urlencoded`“. Der Request muss die folgenden Parameter im Body enthalten:

client_id	die Client-ID der Anwendung
client_secret	das Client-Secret, das bei der **Client-Registrierung** vereinbart wurde
refresh_token	der Refresh-Token aus **Schritt 4: Benutzer wird zum Client zurückgeschickt**
grant_type	muss den Wert `refresh_token` enthalten

Antwort

Es wird ein JSON-Objekt mit folgenden Feldern zurückgegeben:

access_token	Access-Token, mit dem auf die d+ OSC APIs zugegriffen werden kann
expires_in	die Gültigkeitsdauer des Access-Tokens in Sekunden (Access-Tokens haben eine Gültigkeit von zwei Stunden (7200))

refresh_token	Wenn in **Schritt 1: Autorisierungsanfrage** der Parameter `access_type=offline` verwendet wurde, wird ein Refresh-Token zurückgegeben, mit dem ein neuer Access-Token angefordert werden kann.
scope	eine Space-delimeted-Liste von scopes, auf die der Zugriff gewährt wurde
token_type	der Typ des Access-Tokens (ist immer der Wert `Bearer`)

Fehler

Im Fehlerfall wird ein HTTP 400 Fehler (Bad Request) generiert mit einem JSON-Objekt im Body der Antwort mit folgendem Feld:

error	Error-Code laut RFC 6749 5.2

4. OAuth Access Tokens anfordern über Client Credentials Flow

RFC 6748 4.4

Access-Token Anfrage und Antwort

RFC 6749 4.4.2

Der Client fordert vom OSC Sever einen Access-Token an

Token-Endpoint

`https://<hostname>/DPlus.OSCCloud.Rest.Server/api/v1/oauth/token`

Der Client schickt einen HTTP-Post-Request an den Token-Endpoint im Format „`application/x-www-form-urlencoded`“. Der Request muss die folgenden Parameter im Body enthalten:

client_id	die Client-ID der Anwendung
client_secret	das Client-Secret, welches bei der **Client-Registrierung** vereinbart wurde
grant_type	muss den Wert `client_credentials` enthalten
scope	Optional Dies ist eine Space-delimeted-Liste von scopes, auf welcher der Zugriff angefordert wird (z. B. `profile email`). Wenn nichts angegeben wird, werden alle bei der Client-Registrierung vereinbarten scopes verwendet.

Antwort

Es wird ein JSON-Objekt mit folgenden Feldern zurückgegeben:

access_token	Access-Token, mit dem auf die d+ OSC APIs zugegriffen werden kann
expires_in	die Gültigkeitsdauer des Access-Tokens in Sekunden (Access-Tokens haben eine Gültigkeit von zwei Stunden (7200))
scope	eine Space-delimeted-Liste von scopes, auf die der Zugriff gewährt wurde
token_type	der Typ des Access-Tokens (ist immer der Wert `Bearer`)

Fehler

Im Fehlerfall wird ein HTTP 400 Fehler (Bad Request) generiert mit einem JSON-Objekt im Body der Antwort mit folgendem Feld:

error Error-Code laut RFC 6749 5.2

5. API-Aufrufe

Wenn die Anwendung d+ OSC APIs aufruft, muss der Access-Token im `Authorization` Header übergeben werden:

```
GET /DPlus.OSCCloud.Rest.Server/api/v1/user/info
Host: <hostname>
Authorization: Bearer access_token
```

User Information Endpoint

`https://<hostname>/DPlus.OSCCloud.Rest.Server/api/v1/user/info`

Scopes: `profile email`

Hier können Informationen über den Benutzer abgefragt werden. Die Antwort orientiert sich an den OpenID Connect Standard Claims.

Beispiel:

```
GET /DPlus.OSCCloud.Rest.Server/api/v1/user/info
Host: <hostname>
Authorization: Bearer access_token
```

Die d+ OSC API antwortet mit einem JSON-Objekt, welches folgende Felder enthält:

sub	36-stellige UUID des Benutzers (diese ID ist fix und identifiziert den Benutzer eindeutig)
name	Anzeigename des Benutzers
given_name	Vorname
family_name	Nachname
email	E-Mail-Adresse des Benutzers
email_verified	Boolean, True (wenn der Benutzer die E-Mail-Adresse im d+ OSC verifiziert hat)