# API

## A guide for managers & developers

PART 3: **Options**

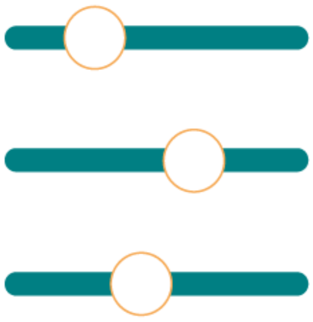
Prepared by OfCourseMe

## OfCourseMe APIs

# 2. Options

**"Options" service is closely related to "courses search". This service is necessary if you want to provide your users with filters in order to refine their searches.**

Your systems can call our "options" service at any given time (e.g. every night) and get in return the full list of filtering options available, with translations in several languages:



- **Platforms:** all the platforms available in your content subset
- **Providers:** the single institutions (e.g. universities)
- **Languages:** all the languages with at least one content available
- **Duration:** duration ranges
- **Level:** beginner - intermediate - advanced

**With this data at hand, you can have filters implemented right below your search bar and be sure that in case new options become available, e.g. you choose to add a new platform to your content subset, they will be readily displayed among your filtering options.**

OfCourseMe APIs

# 2. Options

Please find below the key basic information in order to get started, for developers

## Description

Call endpoint: `https://api.ofcourse.me/v2.0/option`
This endpoint is useful to:
• get values which should be passed as parameters to filter "/courses-search" per platform or language
• get values to understand "/courses-search" or "/course" response field values
To use this endpoint you must pass the `type` parameter (mandatory).

Set `type` value to:
• `platforms` endpoint will return all platforms.
Platforms are ordered alphabetically: option_order field represent platform order, option_value field represent an internal unique id
• `providers`: endpoint will return all providers allowed values (strings)
• `languages:` endpoint will return all the languages (two-character) value for which there is at least one course in the database. Use the Key Values chapter to understand the corresponding language.
• `translations_languages`: endpoint will return all languages available to translate API content.
• `translations`: endpoint will return all key-values (see key-values section)translated in language specified by filter lang (default "en"). See this example

If you don't pass `type` parameter or you pass an invalid value, the API will return an "INVALID-TYPE-PARAMETER" error.

## Examples:

`https://api.ofcourse.me/v2.0/options?type=platforms`
Returns platforms options with those fields:

• option_order: incremental order value (alphabetically sorting)
• option_label: platform name
• option_value: platform internal unique id
• option_icon_extended: platform extended icon
• option_icon: platform icon

`https://api.ofcourse.me/v2.0/options?type=providers`
Returns providers allowed values

`https://api.ofcourse.me/v2.0/options?type=language`
Returns languages key-values

## Translations Examples

`https://api.ofcourse.me/v2.0/options?type=translations_languages`
Returns all languages available to translate API content

`https://api.ofcourse.me/v2.0/options?type=translations`
Returns all key-values translated in english by default

`https://api.ofcourse.me/v2.0/options?type=translations&lang=it`
Returns all key-values translated in italian

## Subset

Learn more about subset and exclusive content

Your calls could be limited by a defined subset on options basis.
This behavior also affects this endpoint.

Assume your subset defines that you can see only courses which are related to platform "3" ("Udacity").
This means you can never retrieve courses associated with other platforms, but only those courses served by Udacity.
Calling /options?type=platform endpoint with no params will result in ANY platform related to courses defined in your subset.
In this case you will get only platform "3" ("Udacity").

Assume your subset defines that you can see only courses which are provided by provider "Google".
This means you can retrieve all courses provided by "Google".
It also means you can never retrieve courses provided ONLY by other providers, but you can always retrieve courses provided by other providers AND "Google".

Assume also that there exists at least a course which is provided by "Google" AND "TED", this course is part of your subset.
For the same reason, calling /options?type=provider endpoint not necessarily will result only in providers which define the subset.
In this case, your subset is limited only to courses provided by "Google".
So, calling this endpoint, you will NOT receive only "Google" but ALL providers which are related to courses provided by "Google".
As we have assumed that there exists at least a course which is provided by "Google" AND "TED", this endpoint response will include both providers as well as other providers which are providing courses with "Google".

## IMPORTANT NOTICE:

You can use every option returned by this endpoint in /courses-search parameters, always not exceeding your subset definition.

In our last assumption, if you ask for courses related to provider "TED", as your subset defines these courses should be provided also by "Google", you'll get only courses provided by both ("Google" and "TED"), even not specifying provider "Google" in your query.

`https://api.ofcourse.me/v2.0/options?type=platforms`
Assume your subset defines that you should see only courses served by platform "3" (Udacity), returns only platform "3" (Udacity)

`https://api.ofcourse.me/v2.0/options?type=providers`
Assume your subset defines that you should see only courses provided by "Google" and exist one course provided by both "Google" and "TED",
returns at least both providers "Google" and "TED" (and all other providers which are providing courses with "Google")

## User tracking parameters

It is possible, if desired, to identify a single user by appending a "user_id" parameter to your API-calls.

You can use a session id or an internal id (integer/string) to set the "user_id" parameter.
Also, if desidered, you could identify a user group, family or entity by appending a "entity_id" parameter to your API-calls.

You can use an internal group, family or entity id (integer/string) to set the "entity_id" parameter.
This will help us computing API-usage statistics and metrics by single user and / or by user group / or by user family / or by user entity.

User tracking parameters are optionals.

Here is a basic example about the "/courses-search" endpoint.

`https://api.ofcourse.me/v2.0/options?type=providers&user_id=USER-ID`
The "user_id" parameter is properly set to identify a single user

`https://api.ofcourse.me/v2.0/options?type=providers&entity_id=GROUP-ID`
The "entity_id" is properly set to identify user group

`https://api.ofcourse.me/v2.0/options?type=providers&user_id=USER-ID&entity_id=GROUP-ID`
The "user_id" parameter is properly set to identify a single user and "entity_id" is properly set to identify user group

`https://api.ofcourse.me/v2.0/options?type=providers&user_id=USER-ID&entity_id=FAMILY-ID`
The "user_id" parameter is properly set to identify a single user and "entity_id" is properly set to identify user family

`https://api.ofcourse.me/v2.0/options?type=providers&user_id=USER-ID&entity_id=ENTITY-ID`
The "user_id" parameter is properly set to identify a single user and "entity_id" is properly set to identify user entity

## Code Examples

JS (jQuery AJAX)
PHP (CURL)
C# (WebRequest)
Java (HttpUrlConnection)

```
                                                  $.ajax({
                                                            url: "https://api.ofcourse.me/v2.0/
options?type=platforms",
                                                            type: GET,
                                                            headers: {"x-api-key":
"***********************************IcAuSy"} // insert your API Key
                                                  })
                                                  .done(function(response) {
                                                            console.log(response);
                                                  })
                                                  .fail(function(err) {
                                                            console.log(err);
                                                  });
```