API

A guide for managers & developers

Prepared by OfCourseMe

INTRODUCTION

What is an **API?**

API is an acronym that stands for Application Programming Interface, which might sound complicated but it's simply a software that allows two applications to talk to each other, just like a translator.

You use APIs every day without even knowing that,
for very common purposes such as:

- Checking the weather forecasts on Google
- Paying online with Paypal
- Booking a flight with an online travel service such as Expedia or Skyscanner

Well, the last one is less common nowadays, but you get the point. **Someone described APIs as the waiter in a restaurant. The kitchen prepares the food, you enjoy the service, while the API-waiter communicates your order to the kitchen and brings the food to the table.**

On top of being very simple, APIs are:

• **Standardized:** REST-APIs like ours are APIs that adhere to the REST standard. You don't need to know the details, only to be aware that **all REST-APIs in the world work in the same way**. Your developers already had to deal with REST-APIs before, hence this time will be no different.
• **Documented:** as you already know if you are reading this document, **there is always a guide that explains how the APIs work and what their purposes are.** Hence, there are no uncertainties about what can be done and how.
• **Secure:** if there's an API layer in the middle, your device or browser is never directly exposed to the service you are using and vice versa. Also, **the development lifecycle of APIs is tested, secure and robust.**

OfCourseMe APIs

Why using OfCourseMe's APIs

All OfCourseMe features and functionalities are delivered via APIs. Our standalone webapp uses our APIs too, which means that in principle all the magic you can see happen on our demo website can be reproduced exactly within your own environment.

You might decide to integrate our services within an LMS, or a custom platform you built yourself, or an enterprise social network.

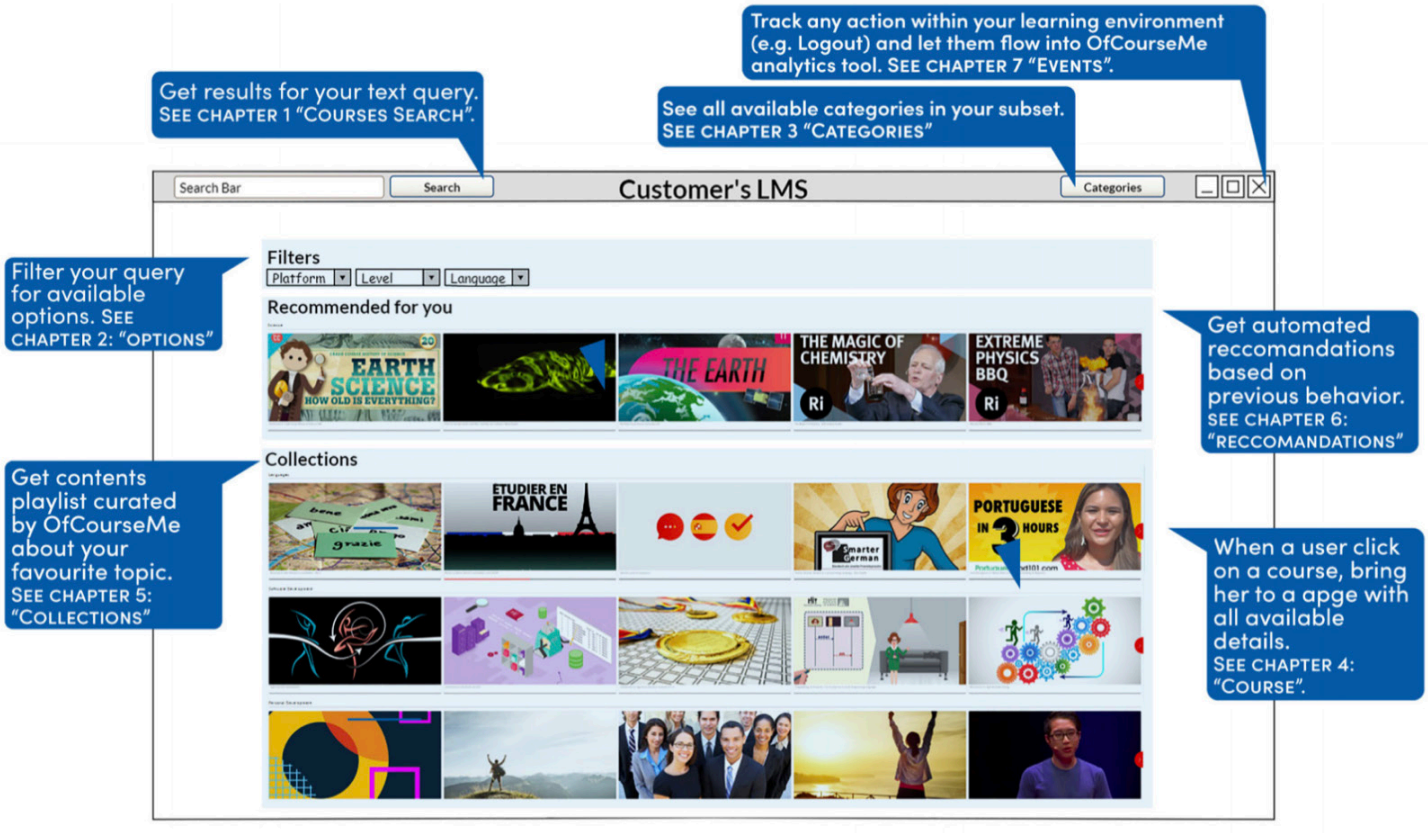
This is - for instance - how Vodafone, Enel, Eni are using OfCourseMe to beef up and complement their existing learning catalogues. If you want to know more about how they use OfCourseMe, you can read some selected case studies here.

OfCourseMe APIs

Our Service in a Nutshell

In the picture below we summarize all the functionalities available on our API service. **It represents an imaginary LMS / learning environment, where for each button / area you have a callout explaining the functionality it triggers and sends you to the corresponding chapter of the guide.** As we'll see, in principle you can call and use each functionality within any learning environment or LMS.

NB: regardless of our API services, you can always decide to limit the third party online contents made available in your systems, defining a subset of platforms, categories, prices. E.g. you might want to exclude paid courses, YouTube videos and leisure categories. We will do this for you, with no development required on your end.



OfCourseMe APIs

Technical Overview

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

Endpoint

OfCourseMe APIs are available calling the endpoint `https://api.ofcourse.me/v2.0`. In order to use our API you need to make a server-side call on `HTTPS` protocol with a `GET` method.

Authentication

The APIs always need a valid API Key to authenticate the calls: you must set the `x-api-key` header and pass a valid value. **If you don't own a valid API Key please contact** info@ofcourse.me.

For production purposes API must be called via server, from IP address pool which should be provided, enabled and trusted by OfCourseMe.

For testing purposes you can call our API from any IP and the call can be performed client-side or server-side. Switching to production stage, your API Key will be associated with one or more provided IP addresses: any API call coming from a not whitelisted IP will return this error:

```
{ "message" : "User is not authorized to access this resource with an explicit deny" }
```

You will always be able to perform client-side API calls using our API Configurator with your API Key, even in production stage.

Subset and exclusive content

Customers might define a subset of our database to be accessible to their employees, i.e. allow their employees to access a portion of courses in our database only.
Customers might also define exclusive content to be accessible to their employees only, i.e. proprietary content which is accessible and searchable together with or on top of public content.

IMPORTANT NOTICE:

- Any action performed through this API will always respect **clients subset/exclusive content agreement with us.**
- Subset and exclusive content a**re computed and secured by authentication with api-keys and IPs** (no action is required by API-user).
- Customers might request us to modify their subsets and exclusive content at any time. **This will be handled by OfCourseMe by means of updating customers API-keys subset and exclusive content settings.**
- **Calls to /courses-search endpoint will always respect client subset, returning only courses included in their subset.**
Other endpoints will always return attributes related to courses included in their subset.
See endpoints documentation for more information on single endpoint behavior.

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 "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/courses-search?user_id=USER-ID`
The "user_id" parameter is properly set to identify a single user

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

`https://api.ofcourse.me/v2.0/courses-search?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/courses-search?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/courses-search?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

OfCourseMe APIs

1. Courses Search

"Courses Search" is the service that allows you to make OfCourseMe's search engine for 3rd parties online learning contents available to your employees, wherever you want.

The most common use cases for this service are:

- **I want a brand new section dedicated to learning contents from the Web.** In this section I want a prominent search bar. **When the user types a text search in the bar, I want to show her all the results found by OfCourseMe that are relevant to that query.**

- **I already have a search bar in our website/LMS, when a user types a search we give her our internal results relevant for that search. We do not have relevant content for all possible topics, hence we often show few or no results. I want to beef them up with OfCourseMe results, displaying them after our internal courses if there are any, but making sure we drastically increase our coverage.**

"Courses Search" service comes with filters available. Via our APIs all searches can be filtered by category, language, platform, level, duration etc. which allows you to put those filters on page and let your users filter their queries as they wish.

An example of a very specific filtered search might be:

"All search results for the keyword "Artificial Intelligence", in English only, from Udacity only, for advanced users, with a minimum duration of 2 hours". A pretty powerful tool, indeed.

When the user clicks on the "search" button, your systems send a request to our APIs which specifies what text search was submitted, with which filters. **Our APIs respond with a list of contents in milliseconds.** Those contents come with elements such as title, image, syllabus, level, duration, so that your developers get everything they need in order to build the course cards in the search results page, in real time. **You choose which data you want to show for each card, depending on how much space is available, which devices and screens your colleagues usually adopt, and your personal taste.**

OfCourseMe APIs

1. Courses Search

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

Description

Call endpoint: `https://api.ofcourse.me/v2.0/courses-search`

Filters

You can filter down your calls to get a text-search or courses in a category. Here are listed all the parameters you can use to filter your calls, they are all optional:

- `query`
set this parameter to make a text-search. Text-search is performed searching for the desired term in course title, course subtitle and course associated categories.
- `category` (multiple values, comma separated)
set this parameter to filter your query by category. Pass one or more ID_CATEGORY divided by comma. You can get all the categories id by calling the /categories endpoint
- `language` (multiple values, comma separated)
set this parameter to filter your query by language. Pass one or more language-code divided by comma. You can get all the language-codes (two-character codes) value by calling the /options endpoint
- `platform` (multiple values, comma separated)
set this parameter to filter your query by platform. Pass one or more platforms divided by comma. You can get all the platforms value by calling the /options endpoint
- `level` (multiple values, comma separated)
set this parameter to filter your query by level. Pass one or more levels divided by comma. Level values are documented in the Key Values chapter.
- `effort` (multiple values, comma separated)
set this parameter to filter your query by effort. Pass one or more effort ranges divided by comma. Effort range values are documented in the Key Values chapter.
- `price_type` (multiple values, comma separated)
set this parameter to filter your query by price type. Price type values are documented in the Key Values chapter.
- `price` (multiple values, comma separated)
set this parameter to filter your query by price ranges. Pass one or more price ranges divided by comma. Price range values are documented in the Key Values chapter. Price value will be ignored if price_type is set only to free ("F").
- `provider` (multiple values contained in double quotes, comma separated)
set this parameter to filter your query by providers. You can get all providers names by calling the /options endpoint.
Pass one or more provider name contained in double quotes and comma separated (syntax

example, **provider:"Amazon","Edx"**).
Be aware you must encapsulate each provider name in double quotes and separate them by comma or you will get an "INVALID-PROVIDER-VALUE" error.
See this examples to understand how to use this filter.

- `course_id` (multiple values, comma separated)
set this parameter to one or more course_id (comma separated) to get only courses specified.
- `edited_after` (UTC timestamp in millisecond. Example 1537793541000)
set this parameter to filter your query by course edited time. Pass a UTC timestamp in milliseconds to get courses edited in our database after this timestamp (including specified millisecond). This parameter is useful to obtain only those courses edited in our database since your last request with the same parameters.
- `lang` (two digit language code)
set this parameter to get course categories in the language needed. You can obtain available languages querying the "option" endpoint with type "translations_languages" (/options?type=translations_languages)
Invalid parameters will raise an `INVALID-PARAMETER` error.
Invalid parameters values will raise an `INVALID-<PARAMETER_NAME>-VALUE` error.

Pagination

Result set is limited to maximum 30 elements per call.

You need to use `start` and `size` parameters to paginate your results:

- `start` is the pointer to the starting record in the result set
- `size` is the length of the result set.

If you don't pass the `start` parameter it will be 0 by default.

If you don't pass the `size` parameter it will be 20 by default.
The maximum value for the `size` parameter is 30.
If you pass a value over 30, it will be changed to 30 as the result set is limited to a maximum of 30 elements per call.

Sorting

You can pass an `order` parameter to sort your result set.
`Order` values are documented in Key Values chapter.
If you don't pass any `order`, the result set will be sorted by default (relevance based on what you searched)

Response Fields

Here are listed the response fields for the course:

- `course_id`: unique ID of the course
- `course_title`: title of the course (string)
- `course_syllabus`: syllabus of the course (string). Syllabus is truncated at 140 characters and sanitized removing html tags inside it to give you a preview of the entire text. To obtain the entire syllabus you must call a single /course endpoint.

- `course_image`: small image of the course (url) for listing purpose. In case no image is provided for a given course, we will serve a fallback image related to that course's category
- `course_hosting_platform`: hosting platform of the course. Call endpoint /options to understand returned values
- `course_owner`: owner of the course (string array)
- `course_provider`: provider of the course (string array)
- `course_level`: level of the course. See Key Values chapter to understand returned values.
- `course_duration`: duration of the course (minutes)
- `course_certificate_available`: defines if the course has a certificate available: "Y" (available) / "N" (unavailable)
- `course_pricing_value`: cost of the course (dollars)
- `course_pricing_type`: type of course pricing. See Key Values chapter to understand returned values.
- `course_active`: defines if the course is still available on the hosting platform: "Y" (active), "N" (inactive).
- `course_release_date`: course_release_date: release date of the course (date)
- `course_categories`: an array of course categories (object array). Each category object will have these attributes:
  - `course_category_id`: category id
  - `course_category_name`: category name
  - `course_category_level`: category level (0 for parent categories, 1 for subcategories)
  - `course_category_parent_id`: category parent id (0 for parent categories)
  - `course_category_parent_name`: category parent name (null for parent categories)
- `search_count`: total count of courses matching search's criteria.

Examples

`https://api.ofcourse.me/v2.0/courses-search?query=Javascript`
Returns first 20 (default values for start=0, size=20, ordered by relevance) courses matching the "Javascript" term in their title, subtitle or categories.

`https://api.ofcourse.me/v2.0/courses-search?query=Javascript&lang=it`
Returns first 20 (default values for start=0, size=20, ordered by relevance) courses matching the "Javascript" term in their title, subtitle or categories. Course categories are translated in italian

`https://api.ofcourse.me/v2.0/courses-search?query=Javascript&level=1`
Returns first 20 courses matching "Javascript" term AND which level is Beginner

`https://api.ofcourse.me/v2.0/courses-search?query=Javascript&level=1,3`
Returns first 20 courses matching "Javascript" term, AND which level is Beginner OR Advanced

`https://api.ofcourse.me/v2.0/courses-search?query=Javascript&order=ED&start=100&size=10`
Returns 10 (size) courses starting from record 100 (start), ordered by effort (descending), matching "Javascript" term

`https://api.ofcourse.me/v2.0/courses-search?query=Javascript&level=1&platform=1`
Returns first 20 courses matching "Javascript" term AND which level is Beginner AND which platform is Coursera

`https://api.ofcourse.me/v2.0/courses-search?query=Javascript&level=1,3&platform=1,3`
Returns first 20 courses matching "Javascript" term, AND which level is Beginner OR Advanced, AND which platform is Coursera OR Udacity

`https://api.ofcourse.me/v2.0/courses-search?query=Javascript&category=22`
Returns first 20 courses matching "Javascript" term which category is Web Development

`https://api.ofcourse.me/v2.0/courses-search?category=22`
Returns first 20 courses which category is "Web Development"

`https://api.ofcourse.me/v2.0/courses-search?category=22,23`
Returns first 20 courses which category is "Web Development" or "Programming Languages"

`https://api.ofcourse.me/v2.0/courses-search?category=22&effort=1&start=100&size=30`
Returns first 30 (size) courses, starting from record 100 (start), which category is "Web Development" and which effort is less than between 20 minutes and 2 hours

`https://api.ofcourse.me/v2.0/courses-search?level=1`
Returns first 20 courses which level is "Beginner"

`https://api.ofcourse.me/v2.0/courses-search?level=1&effort=2,3`
Returns first 20 courses which level is "Beginner" AND which effort is between 2 hours and 10 hours OR between 10 hours and 100 hours

`https://api.ofcourse.me/v2.0/courses-search?price=1&order=PD`
Returns first 20 courses which price is less than 50 dollars, ordered by price descending

`https://api.ofcourse.me/v2.0/courses-search?price=1,3&order=PD`
Returns first 20 courses which price is less than 50 dollars or between 100 and 300 dollars, ordered by price descending

`https://api.ofcourse.me/v2.0/courses-search?course_id=286226,293406`
Returns course which course_id is 286226 or 293406

Provider Filter Examples

`https://api.ofcourse.me/v2.0/courses-search?provider="Amazon"`
Returns first 20 courses which provider is Amazon

`https://api.ofcourse.me/v2.0/courses-search?provider="Amazon","Edx"`
Returns first 20 courses which provider is Amazon OR Edx

`https://api.ofcourse.me/v2.0/courses-search?provider=Amazon`
This call is invalid as provider name is not encapsulated in double quotes

`https://api.ofcourse.me/v2.0/courses-search?provider="Amazon";"Edx"`
This call is invalid as providers names are not comma separated

Subset

Learn more about subset and exclusive content

Your calls could be limited by a defined subset.
Calling this endpoint without any parameter will return you all courses included in your subset.

Assume your subset defines that you can see only courses which are related to category "22" ("Web Development").
This means you can never retrieve courses associated ONLY to other categories, but you can always retrieve courses related to other categories AND category "22".
Assume also that there exists at least a course which is associated with "Web Development" ("22") AND "Programming Languages" ("23"), this course is part of your subset.
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.

Subset could be defined by a combination of multiple parameters.
Assume your subset defines that you can see only courses which are related to category "22" ("Web Development") and platform "3" ("Udacity").
This means you can never retrieve courses associated ONLY to other categories, but you can always retrieve courses related to other categories AND category "22".
It also means that those courses should be served by "Udacity".
Assume that there's at least a course which is associated with "Web Development" ("22") AND "Programming Languages" ("23") served by platform "1" ("Coursera"), this course is not part of your subset.
Assume also that there's at least a course which is associated with "Web Development" ("22") AND "Programming Languages" ("23") served by platform "3", this course is part of your subset.
You should not do anything to limit your subset, as it is automatically computed on your API-key or IP address. See below example for more information

`https://api.ofcourse.me/v2.0/courses-search`
Assume your subset defines that you should see only courses related to category "22" and exist at least one course which is related to category "22" AND another category,
returns all courses directly associated to category "22", courses associated to category "22" AND other categories are included

`https://api.ofcourse.me/v2.0/courses-search?category=23`
Assume your subset defines that you should see only courses related to category "22" and exist only one course which is related to category "22" AND category "23",
returns only the course associated to category "22" AND category "23"

`https://api.ofcourse.me/v2.0/courses-search`
Assume your subset defines that you should see only courses served by platform "3",
returns all courses served by platform "3"

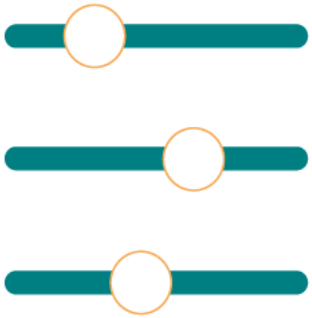
`https://api.ofcourse.me/v2.0/courses-search?platform=4`
Assume your subset defines that you should see only courses served by platform "3",
returns no courses.

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);
});
```

OfCourseMe APIs

3. Categories

“Categories” is the service that allows you to implement a menu with icons and links to OfCourseMe’s categories.

When you click on a link you are redirected to a search results page (SERP) with all the contents available for that category. Key features of the categories menu are:

- The menu is structured on 2 levels:
  - 20 macro categories,
  - >150 sub-categories

- If you have decided to limit the categories available for your employees to a given subset, you will retrieve only categories belonging to that subset.

- When the user clicks on a link to a given category, the search engine requests our API endpoint courses-search all the results available for that category in your subset.

- The same content might be tagged in different categories: i.e “History of philosophy” will be both in History and Philosophy.

OfCourseMe APIs

3.Categories

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

Goal

Endpoint "/categories" is meant to retrieve more details about ALL the categories (topics) associated with at least one course in your subset.
If your subset includes "web development" category but does not include "game design" category, and there is one course about "web developmen" applied to "game design" which is in your subset and is associated to both, endpoint "/categories" will retrieve details about both.

Endpoint "/categories" IS NOT meant to retrieve by default details about the categories in your subset ONLY.
If you want to retrieve details about your selected categories only, please use the list of categories in your subset as a filter in your call to the endpoint "/categories". See details in section "Subset" below.

Description

Call endpoint `https://api.ofcourse.me/v2.0/categories`.

No parameters are needed for this method, but if you want to receive details for a selected categories only then you are allowed to pass the category parameter.

This parameter should be filled with categories ids divided by comma or you'll get the "INVALID-CATEGORIES-IDS-PARAMETER" error.

If you pass a category id which does not correspond to any categories in our database, you'll get a "NO-CATEGORIES-FOUND" error.

You can pass a parameter "lang" (two digit lang code) to get categories in the language needed. You can obtain available languages quering the "option" endpoint with type "translations_languages" (/options?type=translations_languages)

Response fields

Here are listed all response fields for a single category
- `course_category_id`: category id
- `course_category_name`: category name
- `course_category_level`: category level (0 for parent categories, 1 for subcategories)
- `course_category_parent_id`: category parent id (0 for parent categories)
- `course_category_parent_name`: category parent name (null for parent categories)
- `course_category_image`:OfCourseMe image for this category (subcategories will have same image of their parent).

Examples

`https://api.ofcourse.me/v2.0/categories`
Returns all categories related to your subset

`https://api.ofcourse.me/v2.0/categories?lang=it`
Returns all categories related to your subset, translated in italian

`https://api.ofcourse.me/v2.0/categories?category=119,130`
Returns categories which id is 119 and 130

Subset

Learn more about subset and exclusive content

Your calls could be limited by a defined subset of courses based on the categories they are associated with.
Assume your subset defines that you can access only to contents which are related to category "22" ("Web Development").
This means you can retrieve all courses associated to category "22"; It also means you can never retrieve courses associated ONLY to other categories, but you can always retrieve courses related to other categories AND category "22".
Assume also that there exists at least a course which is associated with "Web Development" ("22") AND "Programming Languages" ("23"), this course is part of your subset.

This behavior also affects this endpoint.
Calling /categories endpoint with no params will result in ANY category related to courses defined in your subset.
For the same reason, calling /categories endpoint will not necessarily result only in retrieving the categories selected in order to determine your subset.
In this case, your subset is limited only to courses related category "22".
So, calling this endpoint, you will NOT receive only category "22" but ALL categories which are related to courses directly related to category "22".
As we have assumed that there exists at least a course which is associated to "Web Development" ("22") AND "Programming Languages" ("23"), this endpoint response will also include category "23" as well as other categories related to all courses in category "22".

Note also that parent categories are always returned by this endpoint.
If your subset is defined on a parent category all child categories will be returned by this endpoint.
If your subset is defined on a child category, its parent category will be returned by this endpoint.

Should you need to ONLY show category "22" in your platform using this endpoint, you should **filter this endpoint using a category parameter.**

Important notice:
You can use every category returned by this endpoint in /courses-search category parameter, always not exceeding your subset definition.
In this case, if you ask for courses related to category "23" ("Programming Languages"), as your

subset defines these courses being related also to category ("22"), you'll get only courses related to category "22" and "23", even not specifying category "22" in your query.
See below examples for more detail.

`https://api.ofcourse.me/v2.0/categories`
As your subset defines that you should see only courses related to category "22" and exist at least one course which is related to category "22" AND category "23",
returns category "22", category "23" and their parent categories

`https://api.ofcourse.me/v2.0/categories?category=22`
Returns category 22
You can prevent applying your subset to get all available OfCourseMe categories passing a parameter filter_by_subset with value 0. See example below.

`https://api.ofcourse.me/v2.0/categories?filter_by_subset=0`
Get all OfCourseMe categories not limited by your subset

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.

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

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

`https://api.ofcourse.me/v2.0/categories?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/categories?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/categories?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)

For testing purpose only

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

OfCourseMe APIs

4. Course

“Course” is the service that allows you to create a full page course card, with all the details and metadata related to the specific course.

The user journey looks like follows:

- The user types a text search or picks up a category (see “courses search” and “categories” above).
- You show her a search results page with all the contents available for that query.
- The user chooses a course and clicks on it.
- This triggers a call to our “course” endpoint.
- We return all the details of the course (syllabus, author, destination URL etc.), which you can use to fill in your course detail page, i.e. a full page course card.

The difference between “courses search” and “course” is:

- “Courses search” returns a small set of metadata for many courses: it allows you to create a search results page with many courses and preliminary information for each course
- “Course” returns full metadata for a single course: it allows you to create a single course card, with all the relevant information for that course

It’s pointless to retrieve all the information for all courses: you’d better call the “course” endpoint only when the user picks up a specific course.

OfCourseMe APIs

4.Course

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

Description

Call endpoint `https://api.ofcourse.me/v2.0/course`

You must pass the `course_id` parameter.

This parameter is mandatory to use this endpoint or you'll get the "NO-COURSE-ID-PARAMETER" error.

This parameter should be an integer or you'll get the "INVALID-COURSE-ID-PARAMETER" error.

If you pass a "course_id" which does not correspond to a course in our database, you'll get an "NO-COURSE-FOUND" error.

You can pass a parameter "lang" (two digit language code) to get course categories in the language needed.

You can obtain available languages querying the "option" endpoint with type "translations_languages" (/options?type=translations_languages)

Response Fields

Here are listed the response fields for the course:

- `course_id`: OfCourseMe unique ID of the course
- `course_url`: url of the course (string)
- `course_title`: title of the course (string)
- `course_syllabus`: syllabus of the course (string)
- `course_image`: high resolution image of the course (url).
  In case no image is provided for a given course, we will serve a fallback image related to that course's category
- `course_hosting_platform`: hosting platform of the course.
  Call endpoint /options to understand returned values
- `course_owner`: owner of the course (string array)
- `course_level`: level of the course. See Key Values chapter to understand returned values.
- `course_provider:` provider of the course (string array)
- `course_duration`: duration of the course (minutes)
- `course_languages`: languages of the course.
  See Key Values chapter to understand returned values.

- `course_certificate_available`: defines if the course has a certificate available: "Y" (available) / "N" (unavailable)
- `course_certificate_pricing_value`: cost of the certificate if available
- `course_pricing_value`: cost of the course (dollars)
- `course_pricing_type`: type of course pricing.
  See Key Values chapter to understand returned values.
- `course_release_date`: release date of the course (date)
- `course_active`: defines if the course is still available on the hosting platform: "Y" (active), "N" (inactive).
- `course_last_import`: OfCourseMe last course data import (date)
- `course_categories`: an array of course categories (object array). Each category object will have these attributes:

    - `course_category_id`: category id
    - `course_category_name`: category name
    - `course_category_level`: category level (0 for parent categories, 1 for subcategories)
    - `course_category_parent_id`: category parent id (0 for parent categories)
    - `course_category_parent_name`: category parent name (null for parent categories)

Examples

`https://api.ofcourse.me/v2.0/course?course_id=183559`
Returns course which id is 183559
`https://api.ofcourse.me/v2.0/course?course_id=183559&lang=it`
Returns course which id is 183559. Course categories are translated in Italian

Subset

Learn more about subset and exclusive content

Your calls could be limited by a defined subset.
This endpoint returns you the course only if it is part of your subset.

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.

https://api.ofcourse.me/v2.0/course?course_id=183559&user_id=USER-ID
The "user_id" parameter is properly set to identify a single user
https://api.ofcourse.me/v2.0/course?course_id=183559&entity_id=GROUP-ID
The "entity_id" is properly set to identify user group
https://api.ofcourse.me/v2.0/course?course_id=183559&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/course?course_id=183559&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/course?course_id=183559&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)
For testing purpose only

```
        $.ajax({
             url: "https://api.ofcourse.me/v2.0/course?course_id=183559",
             type: GET,
             headers: {"x-api-key": "**********************************IcAuSy"}
// insert your API Key
        })
        .done(function(response) {
             console.log(response);
        })
        .fail(function(err) {
             console.log(err);
        });
```

OfCourseMe APIs

5. Collections

"Collections" service is the one that represents OfCourseMe the most and delivers the highest value to the final users.

The business need/use case goes like follows:

- When the user logs into the LMS / learning environment, I want to show her some custom playlists with free contents from the web, centered around our companies values / strategic plan / key challenges in the industry

This is the core of our "dynamic curation" service, which we explained in detail here. Here you can find a summary of the main steps:

1. The customer provides OfCourseMe with a detailed brief, including names of the collections, topics they want to cover, preferred format, providers, duration, languages

2. OfCourseMe prepares the playlists and iterates together with the customer until they are satisfied with the outcome
OfCourseMe expose the playlists via APIs at the endpoint "Collections"

3. The customer's IT / system integrator uploads the collections onto the LMS / learning environment by calling the endpoint "Collections"

4. After one quarter we look at the data – what users liked most, what else they looked for – and modify the playlists accordingly

When the user logs into the LMS / learning environment, she now sees the curated playlists immediately, wherever you decided to display them.

OfCourseMe APIs

5. Collections

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

Description

Call endpoint **https://api.ofcourse.me/v2.0/collections**
This endpoint is useful to:
• get an array of pre-configured collections of courses (list of course_id)
To use this endpoint you don't need to pass any parameter.

You can pass a parameter "language" (two digit language code) to get only collections in the language needed.

You can pass a parameter "collection" (collection_id separated by comma) to get only collections needed.

Response Fields

Here are listed the response fields for a single collections:
• `collection_id`: OfCourseMe unique ID of the collection
• `collection_key`: OfCourseMe unique key of the collection (string)
• `collection_title`: title of the collection (string)
• `collection_audience_language`: audience language of the collection (string)
• `courses_ids`: a list of course_id (ordered by relevance) contained by collection (string)

Examples

`https://api.ofcourse.me/v2.0/collections`
Returns pre-configured collections of courses
`https://api.ofcourse.me/v2.0/collections?collection=1,2`
Returns pre-configured collections with collection_id 1 and 2
`https://api.ofcourse.me/v2.0/collections?language=en`
Returns pre-configured english collections of courses
`https://api.ofcourse.me/v2.0/collections?language=it`
Returns pre-configured italian collections of courses

Subset

Learn more about subset and exclusive content
Your calls could be limited by a defined subset on options basis.

This behavior affects also this endpoint: you will get only courses in your subset in returned collections.

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/collections?&user_id=USER-ID
The "user_id" parameter is properly set to identify a single user

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

https://api.ofcourse.me/v2.0/collections?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/collections?&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/collections?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

PHP (CURL)
C# (WebRequest)
Java (HttpUrlConnection)
For testing purpose only

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

OfCourseMe APIs

6. Recommendations

“Recommendation” service is a powerful tool to deliver automatically a custom learning experience to each single user.

Just like on Spotify or Netflix, the use case is the following:

- I want to provide the user with a section “courses you might also like” which displays contents in line with that user’s preferences and behaviours, in order to maximize her likelihood to embrace new contents and ultimately her engagement

In short, OfCourseMe tracks user interactions with the contents (searches, clicks etc.), then our algorithms weigh those interactions and proactively suggest new contents.

Easier done than said!

OfCourseMe APIs

6. Recommendations

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

Description

Call endpoint **https://api.ofcourse.me/v2.0/recommendations**
This endpoint is useful to get recommended courses for a specific user based on previous performed and tracked searches.

To use this endpoint you should before have tracked your /courses-search API-calls appending **user_id** parameter (see User tracking parameters

To use this endpoint you must pass the same **user_id** parameter (mandatory) which you use to track the user /courses-search

If you don't pass a **user_id** parameter or you pass an invalid value, the API will return an "INVALID-USER-ID" error.

Examples

`https://api.ofcourse.me/v2.0/recommendations?user_id=1`
Returns recommended courses for user identified by id 1 based on previous performed and tracked searches

Response Fields

Here are listed the response fields for the course:

- `course_id`: unique ID of the course
- `course_title`: title of the course (string)
- `course_syllabus`: syllabus of the course (string). Syllabus is truncated at 140 characters and sanitized removing html tags inside it to give you a preview of the entire text. To obtain the entire syllabus you must call a single /course endpoint.
- `course_image`: small image of the course (url) for listing purpose. In case no image is provided for a given course, we will serve a fallback image related to that course's category
- `course_hosting_platform`: hosting platform of the course. Call endpoint /options to understand returned values
- `course_owner:` owner of the course (string array)
- `course_provider`: provider of the course (string array)
- `course_level`: level of the course. See Key Values chapter to understand returned values.
- `course_duration`: duration of the course (minutes)
- `course_certificate_available`: defines if the course has a certificate available: "Y" (available) / "N" (unavailable)

• **course_pricing_value**: cost of the course (dollars)
• **course_pricing_type**: type of course pricing. See Key Values chapter to understand returned values.
• **course_active**: defines if the course is still available on the hosting platform: "Y" (active), "N" (inactive).
• **course_release_date**: course_release_date: release date of the course (date)
• **course_categories**: an array of course categories (object array). Each category object will have these attributes:
    • course_category_id: category id
    • course_category_name: category name
    • course_category_level: category level (0 for parent categories, 1 for subcategories)
    • course_category_parent_id: category parent id (0 for parent categories)
    • course_category_parent_name: category parent name (null for parent categories)
• **search_count**: total count of courses recommended.

User tracking parameters

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

To use this endpoint you should before have tracked your /courses-search API-calls appending **user_id** parameter.

You can use a session id or an internal id (integer/string) to set **user_id** *parameter for /courses-search API-calls.*
Having tracked your /courses-search API-calls with a **user_id**, we are able to retrieve recommendations for your user if you pass the same **user_id** to this endpoint.

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, but "user_id" is needed to track search and get recommendations.
Here is a basic example about the "/courses-search" endpoint.

https://api.ofcourse.me/v2.0/courses-search?user_id=USER-ID
The "user_id" parameter is properly set to identify a single user

Code Examples

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

For testing purpose only

```
$.ajax({
    url: "https://api.ofcourse.me/v2.0/recommendations?user_id=1",
    type: GET,
    headers: {"x-api-key": "**********************************IcAuSy"} // insert your API Key
})
.done(function(response) {
    console.log(response);
})
.fail(function(err) {
    console.log(err);
});
```

OfCourseMe APIs

7. Events

This service is meant for reporting and analytics purposes. By default, our analytics tool will provide you with insights related to all those actions listed above that require a call to our APIs, i.e. searches typed, filters

There are other actions related to external courses exposed by OfCourseMe that happen within your LMS / learning environment you might want to track as well.
Examples are:

- Saved courses: courses that the user bookmarked or added to her plan
- Suggested courses: courses that the user suggested to a colleague
- Completed courses: courses that the user marked as completed

You might also want to track actions that aren't directly related to courses, such as:

- Logout: click on the logout button
- Access to compulsory training: click on the link that sends the user to compulsory training section
- Access to online coaching: click on the link that sends the user to online coaching section
- Access to profile: click on the link that sends the user to her profile section
- Etc.

All of these actions happen on your end, with no need to involve OfCourseMe, but if you want they can flow into our unified reporting so you can answer questions such as: how many people completed at least one course? Out of 100 searches, how many courses get completed? What is the total duration of completed courses, i.e. the hours of training actually supplied?

In order to track these actions, it's enough to send a message to our APIs when they happen, as explained below.

OfCourseMe APIs

7.Events

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

Description

Call endpoint **https://api.ofcourse.me/v2.0/events**
This endpoint is useful to log important events performed by your users.

To use this endpoint you must pass **type** parameter (mandatory) which defines your event type: general events ("menu-click") or course related events (e.g "course-click","course-save",..) are admitted

If you don't pass **type** parameter or you pass an invalid value, the API will return an "NO-TYPE-PARAMETER" error.

If you want to log a course related event you should pass the **course_id** parameter

Examples

https://api.ofcourse.me/v2.0/events?type=menu-click
Logs menu-click action
https://api.ofcourse.me/v2.0/events?type=menu-click&user_id=1
Logs menu-click action for user 1
https://api.ofcourse.me/v2.0/events?type=course-click&course_id=1
Logs course-click action for course 1
https://api.ofcourse.me/v2.0/events?type=course-save&course_id=1
Logs course-save action for course 1

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/events?type=course-save&course_id=1&user_id=1
Logs course-save action for course 1 and user 1

Code ExamplesJS

PHP (CURL)
C# (WebRequest)
Java (HttpUrlConnection)

For testing purpose only

```
$.ajax({
    url: "https://api.ofcourse.me/v2.0/events?type=course-save&course_id=1&user_id=1",
    type: GET,
    headers: {"x-api-key": "**********************************IcAuSy"} // insert your API Key
})
.done(function(response) {
    console.log(response);
})
.fail(function(err) {
    console.log(err);
});
```