Mothertong REST API Information

API Authentication

API Authentication will implement the Resource Owner Password Credentials grant. It is used directly as an authorization grant to obtain an access token. The resource owner credentials are used for a single request and are exchanged for an access token. This grant type can eliminate the need for the client to store the resource owner credentials for future use, by exchanging the credentials with a long-lived access token or refresh token.

The client makes a request to the token endpoint by adding the following parameters:

* grant\_type: Required. Value must be set to “password”
* username: Required. The resource owner (user) username
* password: Required. The resource owner (user) password
* client\_id: Required. A client id provided for the mobile app

The request should look like this:

<https://url/api/oauth/token?grant_type=password&username=USERNAME&password=PASSWORD&client_id=CLIENT_ID>

The response will send an access token and optional refresh token, and constructs the response by adding the following parameters to the entity-body of the HTTP response with a 200 (OK) status code:

- access\_token: Required. The access token issued by the authorization server. Length TBD

- token\_type: Required. The type of the token. Value is case insensitive and is “bearer”.

* expires\_in: Required. The lifetime in seconds of the access token.
* refresh\_token: OPTIONAL. The refresh token, which can be used to obtain new access tokens. Length TBD

The parameters are included in the entity-body of the HTTP response using the "application/json" media type. The parameters are serialized into a JavaScript Object Notation (JSON) structure by adding each parameter at the highest structure level.

Parameter names and string values are included as JSON strings.

Numerical values are included as JSON numbers. The order of parameters does not matter and can vary.

The authorization server will include the HTTP "Cache-Control" response header field with a value of "no-store" in any response containing tokens, credentials, or other sensitive information, as well as the "Pragma" response header field with a value of "no-cache".

For example:

HTTP/1.1 200 OK

Content-Type: application/json;charset=UTF-8

Cache-Control: no-store

Pragma: no-cache

{

"access\_token":"2YotnFZFEjr1zCsicMWpAA",

"token\_type":"bearer",

"expires\_in":86400,

"refresh\_token":"tGzv3JOkF0XG5Qx2TlKWIA"

}

Error response:

The authorization server responds with an HTTP 400 (Bad Request)

status code (unless specified otherwise) and includes the following

parameters with the response:

error

REQUIRED. A single ASCII [USASCII] error code from the

following:

invalid\_request

The request is missing a required parameter, includes an

unsupported parameter value (other than grant type),

repeats a parameter, includes multiple credentials,

utilizes more than one mechanism for authenticating the

client, or is otherwise malformed.

invalid\_client

Client authentication failed.

invalid\_grant

The provided authorization grant or refresh token is

invalid, expired, revoked, or was issued to another client.

unauthorized\_client

The authenticated client is not authorized to use this

authorization grant type.

unsupported\_grant\_type

The authorization grant type is not supported by the

authorization server.

The parameters are included in the entity-body of the HTTP response using the "application/json" media type. The parameters are serialized into a JSON structure by adding each parameter at the highest structure level. Parameter names and string values are included as JSON strings. Numerical values are included as JSON numbers. The order of parameters does not matter and can vary.

For example:

HTTP/1.1 400 Bad Request

Content-Type: application/json;charset=UTF-8

Cache-Control: no-store

Pragma: no-cache

{

"error":"invalid\_request"

}

API request calls

All API requests must include the access\_token provided as a parameter or in the headers.

Example:

<http://url/api/user/id?access_token=ACCESS_TOKEN>

or

GET /api/user/id/ HTTP/1.1

Host: www.url.com

Content-length: 614

Content-type: application/json

Authorization: Bearer ya29.wAG8ZdIjWS62yhYr1KhVrvReXY7oN--xnfWNtFU9tqkJlwoOa5\_SuzHjcQo1vllqxFlm

Service Providers

Service provider: User

Possible types: GET, POST, PUT

Return a specific user profile information

TYPE: GET

URL: <http://url.com/api/user/id>

JSON to send: No json required

Example response:

{

"success": true,

"message": "Record Found",

"data": {

"totalCount": 1,

"user": {

"id": "33",

"email": "enconte@colppy.com",

"first\_name": "Eduardo",

"last\_name": "Conte",

"hybridauth\_provider\_name": "mothertong",

"hybridauth\_provider\_uid": "enconte@colppy.com",

"created": "2015-08-06 20:07:11",

"updated": "0000-00-00 00:00:00",

"gender": "m",

"birthdate": "1977-12-22",

"timezone": "America/Argentina/Buenos\_Aires",

"user\_type\_id": "3",

"contact\_email": "enconte@colppy.com",

"gtalk\_skype": "enconte@gmail.com",

"telephone": "",

"description": null,

"country": "AR",

"city": "42581",

"avatar": null,

"autorenewal": "0",

"welcome\_message": "1",

"userType": {

"id": "3",

"type": "Double"

},

"languages": [

{

"id": "1",

"name": "English"

},

{

"id": "2",

"name": "Spanish"

}

]

}

}

}

Example “not found” response:

{

"success": false,

"message": "RESOURCE '39' NOT FOUND",

"data": {

"errorCode": 404,

"message": "RESOURCE '39' NOT FOUND"

}

}

Create a new user

TYPE: POST

URL: <http://url.com/api/user/>

Parameters required in the json:

|  |  |
| --- | --- |
| email | Valid email address. Max Length: 50 |
| password | Min Length: 5. Max Length: 15 |

Example of JSON:

{

“email”:”enconte@gmail.com”,

”password”:”12345”

}

Possible errors:

* Password cannot be blank.
* Username is not a valid email address.
* Password is too long (maximum is 15 characters).
* Username cannot be blank.
* Passwords must be the same.
* Username is not a valid email address.

Example json response:

{

"success": "true",

"message": "Record Created",

"data": {

"totalCount": "1",

"user": {

"id": "38",

"email": "Juan@sknf.com",

"first\_name": "",

"last\_name": "",

"hybridauth\_provider\_name": "",

"hybridauth\_provider\_uid": "",

"created": "2015-08-07 14:44:16",

"updated": "0000-00-00 00:00:00",

"gender": "",

"birthdate": "0000-00-00",

"timezone": "",

"user\_type\_id": null,

"contact\_email": "",

"gtalk\_skype": "",

"telephone": "",

"description": null,

"country": "",

"city": "0",

"avatar": null,

"autorenewal": "0",

"welcome\_message": "1",

"userType": null,

"languages": []

}

}

}

Example error responses:

{

"success": false,

"message": "Parameter 'username' is not allowed for model (User)",

"data": {

"errorCode": 406,

"message": "Parameter 'username' is not allowed for model (User)"

}

}

Another possible error response:

{

"success": false,

"message": "{\"password\":[\"Password is too long (maximum is 15 characters).\"]}",

"data": {

"errorCode": "400",

"message": "{\"password\":[\"Password is too long (maximum is 15 characters).\"]}"

}

}

Another possible error response:

{

"success": false,

"message": "{\"email\":[\"Email is not a valid email address.\"],\"password\":[\"Password is too long (maximum is 15 characters).\"]}",

"data": {

"errorCode": "400",

"message": "{\"email\":[\"Email is not a valid email address.\"],\"password\":[\"Password is too long (maximum is 15 characters).\"]}"

}

}

Update a specific user profile information

TYPE: PUT

URL: <http://url.com/api/user/id>

Parameters required in the json:

|  |  |
| --- | --- |
| firstName |  |
| lastName |  |
| gender |  |
| birthdate |  |
| country |  |
| city |  |
| timezone |  |
| contactEmail |  |
| gtalkSkype |  |
| telephone |  |

Example of JSON:

{

”first\_name”:”Eduardo”,

”last\_name”:”Conte”,

” gender”:”Eduardo”,

”birthdate”:”Conte”,

”country”:”Eduardo”,

”city”:”Conte”,

”timezone”:”Eduardo”,

”contactEmail”:”Conte”,

”gtalkSkype”:”Eduardo”,

”telephone”:”Conte”

}

Example response:

{

"success": "true",

"message": "Record Updated",

"data": {

"totalCount": "1",

"user": {

"id": "52",

"email": "qweqwe@gmail.com",

"first\_name": "",

"last\_name": "",

"hybridauth\_provider\_name": "",

"hybridauth\_provider\_uid": "",

"created": "2015-08-09 22:39:36",

"updated": "0000-00-00 00:00:00",

"gender": "",

"birthdate": "0000-00-00",

"timezone": "",

"user\_type\_id": null,

"contact\_email": "",

"gtalk\_skype": "",

"telephone": "",

"description": null,

"country": "",

"city": "0",

"avatar": null,

"autorenewal": "0",

"welcome\_message": "1",

"userType": null,

"languages": []

}

}

}

Service provider: VirtualClass

Possible types: GET, POST

Return all classes from a user

TYPE: GET

URL: [http://url.com/api/VirtualClass/?filter = [{“property”:”teacher\_id”, “value”:24,”operator”:”=”}]](http://url.com/api/VirtualClass/?filter =  [{\“property\”:\”teacher_id\”, \“value\”:24,\”operator\”:\”=\”}])

Results can be filtered by any valid param or multiple params as well as an operator.

Available filter operators: \* in \* not in \* = \* != \* > \* >= \* < \* <= \* No operator is "LIKE".

JSON to send: No json required

Example response:

{

"success": true,

"message": "Record(s) Found",

"data": {

"totalCount": "3",

"virtualClass": [

{

"id": "60",

"class\_status\_id": "8",

"teacher\_id": "20",

"student\_id": "19",

"title": "asdfasdfa",

"language\_id": "6",

"comment": "afasdfasdf",

"time": "14",

"communication\_method\_id": "1",

"created": "2015-04-08 11:17:42",

"level": "0",

"date": "2015-07-16",

"last\_action\_id": "2",

"timestamp\_time": "07:00:00",

"reminders": "1",

"teacher\_subscription\_id": "0",

"student\_subscription\_id": "0",

"teacher\_settle": null,

"student\_settle": null

},

{

"id": "61",

"class\_status\_id": "8",

"teacher\_id": "20",

"student\_id": "23",

"title": "Titulo de la clase !!!",

"language\_id": "3",

"comment": "dsfasdfasf",

"time": "40",

"communication\_method\_id": "1",

"created": "2015-04-14 19:24:28",

"level": "0",

"date": "2015-04-16",

"last\_action\_id": "2",

"timestamp\_time": "20:00:00",

"reminders": null,

"teacher\_subscription\_id": "0",

"student\_subscription\_id": "0",

"teacher\_settle": null,

"student\_settle": null

}

]

}

}