CLOUD, LEAN AND COMPLETE LMS WITH AN EMPHASIS ON USABILITY AND EASY COURSE CREATION

API Documentation

Version 3.7

www.talentlms.com

Contents

In a nutshell

The TalentLMS API is organized around Representational State Transfer (REST). Our API is designed to use built-in HTTP features, like HTTP authentication and HTTP response codes. Regarding the responses, JSON is returned in all responses from the API, including errors.

If your TalentLMS domain is called "*samples*", then the API endpoint for your domain is: **https://samples.talentlms.com/api**

The current version of the API consumes data – information about users, courses, categories, groups, branches and general details about your domain. Apart from that, via the API you can login/signup a user in your domain, enroll a user to a course and much more.

The summary of resource URL patterns is:

- /v1/users
- /v1/users/id:{userId}
- /v1/users/email:{userEmail}
- /v1/users/username:{userName}
- /v1/userlogin
- /v1/userlogout
- /v1/usersignup
- /v1/deleteuser
- /v1/edituser
- /v1/usersetstatus/user_id:{userId},status:{status}
- /v1/courses
- /v1/courses/id:{courseId}
- /v1/createcourse
- /v1/deletecourse
- /v1/categories
- /v1/categories/id:{categoryId}
- /v1/groups

- /v1/groups/id:{groupId}
- /v1/creategroup
- /v1/deletegroup
- /v1/branches
- /v1/branches/id:{branchId}
- /v1/createbranch
- /v1/deletebranch
- /v1/branchsetstatus/branch_id:{branchId},status:{status}
- /v1/forgotusername/email:{email},domain_url:{domainUrl}
- /v1/forgotpassword/username:{userName},domain_url:{url},redirect_url:{url}
- /v1/addusertocourse
- /v1/removeuserfromcourse/user_id:{userId},course_id:{courseId}
- /v1/getuserstatusincourse/course_id:{courseId},user_id:{userId}
- /v1/resetuserprogress/course_id:{courseId},user_id:{userId}
- /v1/addusertobranch/user_id:{userId},branch_id:{branchId}
- /v1/removeuserfrombranch/user_id:{userId},branch_id:{branchId}
- /v1/addcoursetobranch/course_id:{courseId},branch_id:{branchId}
- /v1/addusertogroup/user_id:{userId},group_key:{groupKey}
- /v1/removeuserfromgroup/user_id:{userId},group_id:{groupId}
- /v1/addcoursetogroup/course_id:{courseId},group_id:{groupId}
- /v1/gotocourse/user_id:{userId},course_id:{courseId}
- /v1/getusersbycustomfield/custom_field_value:{value}
- /v1/getcoursesbycustomfield/custom_field_value:{value}
- /v1/buycourse
- /v1/buycategorycourses
- /v1/getcustomregistrationfields
- /v1/getcustomcoursefields

- /v1/categoryleafsandcourses/id:{categoryId}
- /v1/getusersprogressinunits/unit_id:{unitId},user_id:{userId}
- /v1/gettestanswers/test_id:{testId},user_id:{userId}
- /v1/getsurveyanswers/survey_id:{surveyId},user_id:{userId}
- /v1/getiltsessions/ilt_id:{iltId}
- /v1/gettimeline
- /v1/siteinfo
- /v1/ratelimit

Only super administrators are able to activate API support and get the API key.

Authentication

You authenticate to the TalentLMS API by providing the API key in the request. You can manage your API key from your super–admin account (Account & Settings → Basic settings). Be sure to keep your API key secret, as it is the main resource to authenticate to the API.

To use your API key, a call to *TalentLMS::setApiKey()* is needed only once. The PHP library stores the key and sends it in each request. After setting the API key, you need to specify your domain, by calling *TalentLMS::setDomain()*[1].

Errors

TalentLMS API uses HTTP response codes to indicate success or failure of requests. Specifically, codes in the 2xx range indicate success, codes in the 4xx range indicate an error that resulted from the provided arguments (e.g. instead of an integer a string is supplied) and 500 error code indicate an internal TalentLMS error.

All errors return JSON consisting of a type (invalid_request_error or api_error) and a message describing the error.

[1] See a complete example at the end of this document, describing the basic usage of the PHP library.

Response codes summary

Code	Description
200	Request executed properly
400	A required parameter is missing or an invalid type (e.g. a string) was supplied instead of an integer
401	Invalid API key provided
403	API is not enabled for the specified domain or the domain is currently inactive
404	The requested resource (e.g. user) does not exist
500	Internal server error

Rate limits

Rate limits represent the maximum number of API requests that is permitted to make per hour. These limits depend on your subscription plan and are as follows:

Plan	Limit
Free	50
Small	2.000
Basic	2.000
Plus	10.000
Premium	10.000
Custom	Contact us so we can create a plan that fits your needs

Users

You can retrieve individual users as well as a list of all your users. To retrieve a specific user you need to call *TalentLMS_User::retrieve({userID})* where *{userID}* is the unique identifier describing the requested user. To retrieve all your users you need to call *TalentLMS_User::all()*.

Retrieving a user – Example response


```
{"id":"1",
"login":"dummy",
"first_name":"Dummy",
"last_name":"Dummy",
"email":"dummy@gmail.com",
"user_type":"SuperAdmin",
"timezone":"(GMT +02:00) Athens, Istanbul, Minsk",
"language":"English",
"status":"active",
"deactivation_date":"",
"level":"1",
"points":"0",
"credits":"0",
"created_on":"17/07/2012, 15:29:25",
"last_updated":"27/02/2018, 17:04:31",
"last_updated_timestamp":"1519743871",
"avatar":"http://example.talentlms.com/pages/images/unknown_users.png",
"bio":null,
"login_key":"http://example.talentlms.com/index/autologin/key:x5p5ghehpnv56
w85i3ey",
"courses":[{"id":"1",
      "name":"Social media",
      "role":"instructor",
      "enrolled_on":"06/09/2013, 13:24:52",
              "enrolled_on_timestamp":"1378463092",
      "completed_on":"",
              "completed_on_timestamp":"",
              "completion_status":"not_attempted",
              "completion_percentage":"0",
```

```
                "expired_on":"",
                "expired_on_timestamp":"",
                "total_time":"7m 31s"},
        {"id":"19",
        "name":"Money as debt",
        "role":"instructor",
        "enrolled_on":"06/09/2013, 13:24:52",
                "enrolled_on_timestamp":"1378463092",
        "completed_on":"",
                "completed_on_timestamp":"",
                "completion_status":"incomplete",
                "completion_percentage":"0",
                "expired_on":"",
                "expired_on_timestamp":"",
                "total_time":"8m 41s"}],
"branches":[{"id":"1", "name":"mybranch"}],
"groups":[{"id":"1", "name":"My group"}],
"certifications":[{"course_id":"12", "course_name":" Intro to TalentLMS",
"unique_id":"49471362567302","issued_date":"18/04/2013","expiration_date":"
Never","download_url":"http://example.talentlms.com/certificate/download/id
:1"}],
"badges":[{"name":"Activity Newbie",
                        "type":"activity",
                        "criteria":"4 logins",
                        "issued_on":"05/11/2014, 14:44:23"}]}
```


Note that you can also retrieve a user by his email address or his username. You need to call *TalentLMS_User::retrieve(array('email' => '{userEmail}'))* where *{userEmail}* is the email address describing the requested user or you need to call *TalentLMS_User::retrieve(array('username' => '{userName}'))* where *{userName}* is the username of the requested user.

Retrieving all users – Example response


```
[{
"id":"1",
"login":"dummy",
"first_name":"Dummy",
"last_name":"Dummy",
"email":"dummy@gmail.com",
```

```
"user_type":"SuperAdmin",
"timezone":"(GMT +02:00) Athens, Istanbul, Minsk",
"language":"English",
"status":"active",
"deactivation_date":"",
"level":"1",
"points":"0",
"credits":"0",
"created_on":"17/07/2012, 15:29:25",
"last_updated":"27/02/2018, 17:04:31",
"last_updated_timestamp":"1519743871",
"avatar":"http://example.talentlms.com/pages/images/unknown_users.png",
"bio":null,
"login_key":"http://example.talentlms.com/index/autologin/key:x5p5ghehpnv56w85i3ey"},
{"id":"2",
"login":"dummy2",
"first_name":"Dummy2",
"last_name":"Dummy2",
"email":"dummy2@gmail.com",
"user_type":"Admin-Type",
"timezone":"(GMT 00:00) Greenwich Mean Time: Dublin, Edinburgh, Lisbon, London",
"language":"English",
"status":"active",
"deactivation_date":"03/11/2012, 23:59:59",
"level":"1",
"points":"0",
"credits":"0",
"created_on":"03/08/2012, 15:29:14",
"last_updated":"27/02/2018, 17:04:31",
"last_updated_timestamp":"1519743871",
"avatar":"http://example.talentlms.com/pages/images/unknown_users.png",
"bio":"My short bio",
"login_key":"http://example.talentlms.com/index/autologin/key:mwc9e94f87pdauk14y4n"
}]
```

User single-sign-on

As you can see from the above responses, there is a *login_key* property. Its purpose is to make it possible for your users to access content in your TalentLMS domain without having to login directly via the login page. In order to make it happen you have to redirect the user's browser to the URL provided in login_key property.

The login_key URL changes every time it is used, so don't store it anywhere. You need to make a fresh request for each and every login.

User login

You can login a user in your domain by calling

TalentLMS_User::login(array('login' => '{userName}', 'password' => '{password}', 'logout_redirect' => '{logoutRedirect}')). If the provided credentials are valid and correspond to an active user, then the above call returns a response in the following format.

```
{"user_id":"1",
"login_key":"http://example.talentlms.com/index/autologin/key:12drwrd6d2dx14qi98ya"}
```

All you have to do is to redirect the user's browser to the URL provided in "login_key" key, via a simple redirect function like the one below.

```
function redirect($url){
        header("location:$url");
        exit;
}
```

The 'logoutRedirect' is an optional argument which contains the url to redirect the user when he logs out from your TalentLMS domain. Note that if you don't use the PHP library you have to base64 encode this parameter before you pass it to TalentLMS API.

User logout

You can logout a user from your domain by calling

TalentLMS_User::logout(array('user_id' => '{userId}', 'next' => '{nextUrl}')). The 'nextUrl' is an optional argument which contains the url to redirect the user when he logs out from your TalentLMS domain. The above call returns a response in the following format.

```
{"redirect_url":"http://example.talentlms.com/index/logout/next:"}
```

All you have to do is to redirect the user's browser to the URL provided in "redirect_url" key, via a simple redirect function like the one described above.

User signup

You can signup a user in your domain by calling

TalentLMS_User::signup(array('first_name' => '{firstName}', 'last_name' => '{lastName}', 'email' => '{emailAddress}', 'login' => '{userName}, 'password' => '{password}')). If there are custom registration fields, you can pass them in the array in the following form: '*custom_field_XXX*' => '*{customFieldValue}*', where XXX is the index of the custom field (1 – 20). Checkbox custom fields can be equal to 'on' or 'off'. If the provided arguments are valid, then the above call returns a response in the following format.

```
{"id":"3",
"login":"dummy3",
"first_name":"Dummy3"
"last_name":"Dummy3",
"email":"dummy3@gmail.com",
"user_type":"Learner",
"timezone":"(GMT 00:00) Greenwich Mean Time: Dublin, Edinburgh, Lisbon, London",
"status":"active",
"language":"English",
"level":"1",
"points":"0",
"credits":"0",
"created_on":"29/09/2012, 17:59:06",
"last_updated":"27/02/2018, 17:04:31",
"last_updated_timestamp":"1519743871",
"avatar":"http://example.talentlms.com/pages/images/unknown_users.png",
"bio" null,
"login_key":"http://example.talentlms.com/index/autologin/key:43i14ncsad21xf8s8hcs"}
```

Note that you can pass two additional (optional) arguments, named '*branch_id*' and '*group_id*'. If these are valid, then the new user will become a member of the corresponding branch and/or group. The group assignment will have as a result the new user to become member of the group's courses. Also, you can assign a specific user type to the new user by providing the user type name in the optional argument named '*user_type*' and a specific language by providing the code in the optional argument named '*language*'.

Delete a user

You can delete a user by calling *TalentLMS_User::delete(array('user_id' => '{userId}', 'deleted_by_user_id' => '{deletedByUserId}', 'permanent' => '{permanent}'))*. The field 'deleted_by_user_id' is optional. This field represents the user who is responsible for the deletion of the user. If omitted then the owner of the account (super-administrator) is responsible for the deletion. The field 'permanent' ('yes' or 'no' as possible values) is also optional. If the supplied value equals to 'yes', then the user is deleted permanently. If the provided arguments are valid, then the above call returns a response in the following format.

```
{"message":"Operation completed successfully"}
```

Get custom registration fields

You can retrieve the custom registration fields defined in "→ Custom fields → Custom user fields". To retrieve these fields you need to call *TalentLMS_User::getCustomRegistrationFields()*.

Retrieving custom registration fields – Example response

```
[{"key":"custom_field_1",
"type":"text",
"name":"Text Field",
"mandatory":"yes",
"visible_on_reports":"yes",
"dropdown_values":null,
"checkbox_status":"off",
"selective_availability":"no",
"branch":null,
"main_domain_availability":null,
"previous_key":null,
"order":"1"},
{"key":"custom_field_2",
"type":"dropdown",
"name":"Dropdown Field",
"mandatory":"yes",
"visible_on_reports":"no",
"dropdown_values":"1;2;3;4;5",
"checkbox_status":"off",
```

```
"selective_availability":"yes",
"branch":null,
"main_domain_availability":"yes",
"previous_key":"custom_field_1",
"order":"2"}]
```


Set status

You can change the status (active or inactive) of a user by calling *TalentLMS_User::setStatus(array('user_id' => '{userId}', 'status' => '{status}'))*. The 'status' argument should be equal to 'active' or 'inactive'. If the above call executes successfully, it returns the id of the user and his new status.


```
{
"user_id":"2",
"status":"inactive"
}
```


Forgot username

In case one of your users forgot his username, you can call *TalentLMS_User::forgotUsername(array('email' => '{emailAddress}', 'domain_url' => '{domainUrl}'))*. The 'emailAddress' argument is the email address of the user and upon a successful call to this function an email will be sent, containing his username.

The 'domainUrl' is an optional argument which contains the url you want to appear in the email. Leave it empty to use the url of your learning portal (e.g. example.talentlms.com) or your domain map. Note that if you don't use the PHP library you have to base64 encode this parameter before you pass it to TalentLMS API.

If the above call executes successfully, it returns the id of the user.


```
{"user_id":"1"}
```


Forgot password

In case one of your users forgot his password, you can call *TalentLMS_User::forgotPassword(array('username' => '{userName}', 'domain_url' => '{domainUrl}' , 'redirect_url' => '{redirectUrl}'))*. The 'username' argument is the username of the user and upon a successful call to this function an email will be sent, containing instructions on how to reset his password. The 'domainUrl' argument has the same meaning as in the function explained above.

The 'redirectUrl' is also an optional argument which contains the url to redirect the user after he enters his new password. Leave it empty to redirect him in your domain's login page. Note that if you don't use the PHP library you have to base64 encode this parameter before you pass it to TalentLMS API.

If the above call executes successfully, it returns the id of the user.

```
{"user_id":"1"}
```

Edit user

You can update a user in your domain by calling

TalentLMS_User::edit(array('user_id' => '{userId}', 'first_name' => '{firstName}', 'last_name' => '{lastName}', 'email' => '{emailAddress}', 'login' => '{userName}, 'password' => '{password}', 'bio' => '{bio}', 'timezone' => '{timeZone}', 'credits' => '{credits}')). If there are custom registration fields, you can pass them in the array in the following form: '*custom_field_XXX*' => '*{customFieldValue}*', where XXX is the index of the custom field (1 – 20). Checkbox custom fields can be equal to 'on' or 'off'. If the provided arguments are valid, then the above call returns a response in the following format. Optionally you can set a deactivation date on which the user will become automatically inactive. You can set deactivation date as follows: '*deactivation_date*' => '*{deactivationDate}*' (e.g. '*deactivation_date*' => '28/07/2012'). Note that deactivation date can be set only if the user is already active. Passing an empty value will result on disabling deactivation date.

Edit a user – Example response

```
{
"id":"1",
"login":"dummy",
"first_name":"Dummy",
"last_name":"Dummy",
"email":"dummy@gmail.com",
"user_type":"SuperAdmin",
"timezone":"(GMT +02:00) Athens, Istanbul, Minsk",
"language":"English",
"status":"active",
"deactivation_date":"",
"level":"1",
"points":"0",
"credits":"0",
```

```
"created_on":"17/07/2012, 15:29:25",
"last_updated":"27/02/2018, 17:04:31",
"last_updated_timestamp":"1519743871",
"avatar":"http://example.talentlms.com/pages/images/unknown_users.png",
"bio":null,
"login_key":"http://example.talentlms.com/index/autologin/key:x5p5ghehpnv56w85i3ey"
}
```


Get users by custom field value

You can retrieve users filtered by the value in one of their custom registration fields by calling *TalentLMS_User::getByCustomField(array('custom_field_value' => '{customFieldValue}'))*. This call will return all users who have {customFieldValue} as value in one of their custom registration fields. If the above call executes successfully, it returns a response in the following format.

Get users by custom field – Example response


```
{
"1":{
"id":"1",
"login":"dummy1",
"first_name":"Dummy1",
"last_name":"Dummy1",
"email":"dummy1@gmail.com",
"status":"active",
"language":"English",
"deactivation_date":"",
"created_on":"17/07/2012, 15:29:25",
"last_updated":"27/02/2018, 17:04:31",
"last_updated_timestamp":"1519743871",
"certifications":[{"course_id":"1",
      "course_name":"Course #1",
      "unique_id":"182471441408954752",
              "issued_date":"25/08/2014",
              "expiration_date":"25/09/2014",
      "download_url":"http://example.talentlms.com/certificate/download/id:1"}]}
```

```
}
```

Courses

You can retrieve individual courses as well as a list of all your courses. To retrieve a specific course you need to call *TalentLMS_Course::retrieve({courseID})* where *{courseID}* is the unique identifier describing the requested course. To retrieve all your courses you need to call *TalentLMS_Course::all()*.

Retrieving a course – Example response


```
{"id":"12",
"name":"Intro to TalentLMS",
"code":"",
"category_id":"3",
"description":"A short introduction to TalentLMS",
"price":"$0",
"status":"active",
"creation_date":"15/01/2013, 12:46:30",
"last_update_on":"15/01/2014, 12:46:30",
"creator_id":"1",
"hide_from_catalog":"0",
"time_limit":"0",
"level":"1",
"shared":"0",
"shared_url":"",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"big_avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"certification":"Classic",
"certification_duration":"Forever",
"users":[{"id":"1",
      "name":"D. Dummy",
      "role":"instructor",
              "enrolled_on":"15/04/2013, 12:44:58",
              "enrolled_on_timestamp":"1366019098",
      "completed_on":"", "completed_on_timestamp":"",
              "completion_percentage":"0",
              "expired_on":"",
              "expired_on_timestamp":"",
              "total_time":"3m 33s"},
```

```
      {"id":"2",
      "name":"D. Dummy2",
      "role":"learner",
              "enrolled_on":"15/04/2013, 12:45:02",
              "enrolled_on_timestamp":"1366019102",
      "completed_on":"", "completed_on_timestamp":"",
              "completion_percentage":"0",
              "expired_on":"",
              "expired_on_timestamp":"",
              "total_time":"4m 44s"}],
"units":[{"id":"1",
      "type":"Content",
      "name":"TalentLMS - Introduction",
      "url":"example.talentlms.com/unit/apiview/id:1"},
      {"id":"2",
      "type":"Section",
      "name":"Section 1",
      "delay_time":"60",
      "aggregated_delay_time":"60",
      "formatted_aggregated_delay_time":"1h"}],
"rules":["You must complete all content"],
"prerequisites":[]}
```


Retrieving all courses – Example response


```
[{
"id":"1",
"name":"Social media",
"code":"",
"category_id":"3",
"description":"This course explains what is Social Media, their history, how they can help us communicate in the modern Era, and why they have become so popular.",
"price":"$0",
"status":"active",
"creation_date":"15/02/2013, 12:48:33",
"last_update_on":"15/02/2014, 12:48:33",
"creator_id":"1",
```

```
"hide_from_catalog":"0",
"time_limit":"0",
"level":"1",
"shared":"0",
"shared_url":"",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"big_avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"certification":"Classic",
"certification_duration":"11 days"},
{"id":"12",
"name":"Intro to TalentLMS",
"code":"",
"category_id":"3",
"description":"A short introduction to TalentLMS",
"price":"$0",
"status":"active",
"creation_date":"15/01/2013, 12:46:30",
"last_update_on":"15/01/2014, 12:46:30",
"creator_id":"1",
"hide_from_catalog":"0",
"time_limit":"0",
"level":"1",
"shared":"1",
"shared_url":"http://example.talentlms.com/shared/start/key:GGKMFPJU",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"big_avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"certification":"Classic",
"certification_duration":"Forever"
}]
```


Create a new course

You can create a new course by calling *TalentLMS_Course::create(array('name' => '{name}', 'description' => '{description}', 'code' => '{code}', 'price' => '{price}', 'time_limit' => '{timeLimit}', 'category_id' => '{categoryId}', 'creator_id' => '{creatorId}'))*. All fields, except from 'name', are optional. The field 'creator_id' denotes the id of the user that will be the creator – owner of the course. If omitted then the creator is the owner of the account (super-administrator). If there are

custom course fields, you can pass them in the array in the following form: *'custom_field_XXX' => '{customFieldValue}'*, where XXX is id of the custom field. Checkbox custom fields can be equal to 'on' or 'off'. If the provided arguments are valid, then the above call returns a response in the following format.


```
{"id":"100",
"name":"My new course",
"code":"",
"category_id":"3",
"description":"A new course …",
"price":"$0",
"status":"active",
"creation_date":"17/07/2015, 12:48:30",
"last_update_on":"20/07/2015, 15:41:20",
"creator_id":"1",
"hide_from_catalog":"0",
"time_limit":"0",
"level":"",
"shared":"0",
"shared_url":"",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"big_avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"certification":"",
"certification_duration":""}
```


Delete a course

You can delete a course by calling *TalentLMS_Course::delete(array('course_id' => '{courseId}', 'deleted_by_user_id' => '{deletedByUserId}'))*. The field 'deleted_by_user_id' is optional. This field represents the user who is responsible for the deletion of the course. If omitted then the owner of the account (super-administrator) is responsible for the deletion. If the provided arguments are valid, then the above call returns a response in the following format.


```
{"message":"Operation completed successfully"}
```


Enroll user to course

You can enroll a user to a course by calling *TalentLMS_Course::addUser(array('user_id' => '{userId}', 'course_id' => '{courseId}',*

'role' => '{userRoleInCourse}')), where the argument 'role' is optional and can be equal to 'learner' or 'instructor'. Note that instead of 'user_id' you can use 'user_email' and instead of 'course_id' you can use 'course_name'. If the above call executes successfully, it returns the role assigned to the user for the specific course.

```
[{"user_id":"1", "course_id":"1", "role":"learner"}]
```

Un-enroll user from course

You can un-enroll a user from a course by calling *TalentLMS_Course::removeUser(array('user_id' => '{userId}', 'course_id' => '{courseId}'))*. If the course id corresponds to an existing course and the requested user is enrolled in this course, then the above call executes successfully and it returns a response in the following format.

```
{"user_id":"1", "course_id":"1", "course_name":"my course"}
```

Goto course

You can login a user in your domain and redirect him in a specific course by calling *TalentLMS_Course::gotoCourse(array('user_id' => '{userId}', 'course_id' => '{courseId}', 'logout_redirect' => '{logoutRedirect}', 'course_completed_redirect' => '{courseCompletedRedirect}', 'header_hidden_options' => '{ headerHiddenOptions }'))*. If the provided user/course ids are valid and correspond to a user enrolled in the course, then the above call returns a response in the following format.

```
{"goto_url":"http://example.talentlms.com
/index/gotocourse/key:hpdpq8sg52bxlmdv5kb8,course_id:34"}
```

All you have to do is to redirect the user's browser to the URL provided in "goto_url" key, via a simple redirect function.

The 'logoutRedirect' is an optional argument which contains the url to redirect the user when he logs out from your TalentLMS domain. Note that if you don't use the PHP library you have to base64 encode this parameter before you pass it to TalentLMS API.

The 'courseCompletedRedirect' is an optional argument which contains the url to redirect the user when he clicks on the corresponding button after completing successfully the course. Note that if you don't use the PHP library you have to base64 encode this parameter before you pass it to TalentLMS API.

The 'headerHiddenOptions' is an optional argument which contains a semicolon separated list of header items that will disappear from the header when the learners traversing the units of the course. The possible items of this list are: "courseName;units;sharedFiles;moreOptions;certificationIcon".

Buy course

You can redirect a user to PayPal in order to buy a course by calling *TalentLMS_Course::buyCourse(array('user_id' => '{userId}', 'course_id' => '{courseId}', 'coupon' => '{couponCode}'))*. If the provided user/course ids are valid, then the above call returns a response in the following format.

```
{"redirect_url":"https://www.paypal.com/cgi-bin/webscr?..............."}
```

All you have to do is to redirect the user's browser to the URL provided in "redirect_url" key, via a simple redirect function.

Get custom course fields

You can retrieve the custom course fields defined in "Account & Settings → Custom fields → Custom course fields". To retrieve these fields you need to call *TalentLMS_Course::getCustomCourseFields()*.

Retrieving custom course fields – Example response

```
[{"key":"custom_field_1",
"type":"text",
"name":"Text Field",
"mandatory":"yes",
"visible_on_reports":"yes",
"dropdown_values":"",
"checkbox_status":"off",
"selective_availability":"yes",
"branch":null,
"main_domain_availability":"yes",
"previous_key":null},
{"key":"custom_field_2",
"type":"dropdown",
"name":"Dropdown Field",
"mandatory":"yes",
"visible_on_reports":"no",
"dropdown_values":"1;2;3;4;5",
"checkbox_status":"off",
"selective_availability":"no",
"branch":null,
"main_domain_availability":null,
```

```
"previous_key":"custom_field_1"}]
```


Get courses by custom field value

You can retrieve courses filtered by the value in one of their custom course fields by calling *TalentLMS_Course::getByCustomField(array('custom_field_value' => '{customFieldValue}'))*. This call will return all courses that have {customFieldValue} as value in one of their custom course fields. If the above call executes successfully, it returns a response in the following format.

Get courses by custom field – Example response


```
[{"id":"1",
"name":"Social media",
"code":"",
"category_id":"3",
"description":"This course explains what is Social Media, their history, how they can help us communicate in the modern Era, and why they have become so popular.",
"price":"$0",
"status":"active",
"creation_date":"15/02/2013, 12:48:33",
"last_update_on":"15/02/2014, 12:48:33",
"creator_id":"1",
"hide_from_catalog":"0",
"time_limit":"0",
"level":"1",
"shared":"0",
"shared_url":"",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"big_avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"certification":"Classic",
"certification_duration":"11 days,
"custom_field_1":"social",
"custom_field_2":"5"}]
```


Get user status in course

You can retrieve the status of a user in a course (and its units) by calling *TalentLMS_Course::getUserStatus(array('user_id' => '{userId}', 'course_id' =>*

'{courseId}')). If the provided user/course ids are valid, then the above call returns a response in the following format.


```
{"role":"learner",
"enrolled_on":"21/10/2014, 15:21:29",
"enrolled_on_timestamp":"1413894089",
"completion_status":"Completed",
"completion_percentage":"100",
"completed_on":"22/10/2014, 10:26:15",
"completed_on_timestamp":"1413962775",
"expired_on":"",
"expired_on_timestamp":"",
"total_time":"3m 33s",
"units":[{"id":"1",
      "name":"my test",
      "type":"Test",
      "completion_status":"Completed",
      "completed_on":"22/10/2014, 10:25:29",
      "completed_on_timestamp":"1413962729",
      "score":"100"},
      {"id":"2",
      "name":"my document",
      "type":"Presentation | Document",
      "completion_status":"Completed",
      "completed_on":"22/10/2014, 10:24:55",
      "completed_on_timestamp":"1413962695",
      "score":"100"},
      {"id":"3",
      "name":"my first unit",
      "type":"Content",
      "completion_status":"Completed",
      "completed_on":"22/10/2014, 10:26:02",
      "completed_on_timestamp":"1413962762",
      "score":"100"}]}
```

Reset user progress

You can reset the progress of a user in a course by calling *TalentLMS_Course::resetUserProgress(array('user_id' => '{userId}', 'course_id' => '{courseId}'))*. Note that you can pass an additional (optional) argument, named *'remove_certification'*. If this argument equals to 'yes', then the certification for the user in this course will be removed. If the above call executes successfully, it returns a response in the following format.

```
{"message":"Operation completed successfully"}
```

Categories

You can retrieve individual categories as well as a list of all your categories. To retrieve a specific category you need to call *TalentLMS_Category::retrieve({categoryID})* where *{categoryID}* is the unique identifier describing the requested category. To retrieve all your categories you need to call *TalentLMS_Category::all()*.

Retrieving a category – Example response


```
{"id":"3", "name":"Samples",
"price":"$0",
"parent_category_id":null,
"courses":[{"id":"1",
      "name":"Social media",
      "description":"This course explains what is Social Media, their history, how they can help us communicate in the modern Era, and why they have become so popular.",
              "price":"$0",
              "status":"active",
              "hide_from_catalog":"0",
              "level":"1",
              "shared":"0",
              "shared_url":"",
      "avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
      "big_avatar":"http://example.talentlms.com/pages/images/unknown_small.png"},
      {"id":"12",
      "name":"Intro to TalentLMS",
      "description":"A short introduction to TalentLMS",
              "price":"$0",
              "status":"active",
              "hide_from_catalog":"0",
              "level":"1",
              "shared":"0",
              "shared_url":"",
      "avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
```

```
      "big_avatar":"http://example.talentlms.com/pages/images/unknown_small
.png"},
      {"id":"19",
      "name":"Money as debt",
      "description":"Money as Debt is a short animated documentary film by
Canadian artist and filmmaker Paul Grignon about the monetary systems
practiced through modern banking. The film presents Grignon's view of the
process of money creation by banks and its historical background.",
              "price":"$0",
              "status":"active",
              "hide_from_catalog":"0",
              "level":"1",
              "shared":"0",
              "shared_url":"",
      "avatar":"http://example.talentlms.com/pages/images/unknown_small.png
",
      "big_avatar":"http://example.talentlms.com/pages/images/unknown_small
.png"}]}
```


Retrieving all categories – Example response


```
[{"id":"3",
"name":"Samples",
"price":"$0",
"parent_category_id":null},
{"id":"4",
"name":"Test",
"price":"$0",
"parent_category_id":null}]
```


Get category leafs and courses

You can retrieve the children categories of a specific category and the courses of these categories by calling *TalentLMS_Category::retrieveLeafsAndCourses({categoryID})* where *{categoryID}* is the unique identifier describing the requested parent category.

Retrieving leafs and courses – Example response


```
{
"2":{
"id":"2",
"name":"My 2nd category",
"price":"$0",
"parent_category_id":"1",
"courses":[{"id":"19",
      "name":"Course #1",
      "description":"",
      "shared":"0",
      "shared_url":"",
      "avatar":"http://example.talentlms.com/pages/images/unknown_small.png
",
      "big_avatar":"http://example.talentlms.com/pages/images/unknown_small
.png",
      "units":[{"id":"1",
      "type":"Web content",
      "name":"Unit #1",
      "url":"example.talentlms.com/unit/apiview/id:1"},
      {"id":"2",
      "type":"Section",
      "name":"Section 1",
      "delay_time":"60",
      "aggregated_delay_time":"60",
      "formatted_aggregated_delay_time":"1h"}]}]},
"3":{
"id":"3",
"name":"My 3rd category",
"price":"$0",
"parent_category_id":"1",
"courses":[{"id":"24",
      "name":"Course #3",
      "description":"",
      "shared":"0",
      "shared_url":"",
```

```
      "avatar":"http://example.talentlms.com/pages/images/unknown_small.png
",
      "big_avatar":"http://example.talentlms.com/pages/images/unknown_small
.png",
      "units":[{"id":"2",
      "type":"Web content",
      "name":"Unit #2",
      "url":"example.talentlms.com/unit/apiview/id:2"}]}]}
}
```


Buy category courses

You can redirect a user to PayPal in order to buy all category courses as a bundle by calling *TalentLMS_Category::buyCategoryCourses(array('user_id' => '{userId}', 'category_id' => '{categoryId}', 'coupon' => '{couponCode}'))*. If the provided user/category ids are valid, then the above call returns a response in the following format.


```
{"redirect_url":"https://www.paypal.com/cgi-bin/webscr?..............."}
```


All you have to do is to redirect the user's browser to the URL provided in "redirect_url" key, via a simple redirect function.

Groups

You can retrieve individual groups as well as a list of all your groups. To retrieve a specific group you need to call *TalentLMS_Group::retrieve({groupID})* where *{groupID}* is the unique identifier describing the requested group. To retrieve all your groups you need to call *TalentLMS_Group::all()*.

Retrieving a group – Example response

```
{
"id":"1",
"name":"My group",
"description":"My first group",
"key":"MLdDOnOBn",
"price":"$0",
"owner_id":"1",
"belongs_to_branch":"",
"max_redemptions":"",
"redemptions_sofar":"",
"users":[{"id":"1",
      "name":"D. Dummy"},
      {"id":"2",
      "name":"D. Dummy2"}],
"courses":[{"id":"19",
      "name":"Money as debt"},
      {"id":"1",
      "name":"Social media"}]
}
```

Retrieving all groups – Example response

```
[{
"id":"1",
"name":"My group",
"description":"My first group",
"key":"MLdDOnOBn",
"price":"$0",
"owner_id":"1",
```

```
"belongs_to_branch":"",
"max_redemptions":"",
"redemptions_sofar":""},
{"id":"2",
"name":"Test",
"description":"",
"key":"8t4F2Op98",
"price":"$0",
"owner_id":"1",
"belongs_to_branch":"",
"max_redemptions":"",
"redemptions_sofar":""
}]
```


Create a new group

You can create a new group by calling *TalentLMS_Group::create(array('name' => '{name}', 'description' => '{description}', 'key' => '{key}', 'price' => '{price}', 'creator_id' => '{creatorId}', 'max_redemptions' => '{maxRedemptions}'))*. The fields 'description', 'key', 'price' and 'max_redemptions' are optional. The field 'creator_id' denotes the id of the user that will be the owner of the group. If omitted then the owner – creator is the owner of the account (super-administrator). If the provided arguments are valid, then the above call returns a response in the following format.


```
{"id":"10",
"name":"A new group",
"description":"Group created via API",
"key":"MLdDOnOBm",
"price":"$0",
"owner_id":"1",
"belongs_to_branch":"",
"max_redemptions":"",
"redemptions_sofar":""}
```


Delete a group

You can delete a group by calling *TalentLMS_Group::delete(array('group_id' => '{groupId}', 'deleted_by_user_id' => '{deletedByUserId}'))*. The field 'deleted_by_user_id' is optional. This field represents the user who is responsible for

the deletion of the group. If omitted then the owner of the account (super-administrator) is responsible for the deletion. If the provided arguments are valid, then the above call returns a response in the following format.

```
{"message":"Operation completed successfully"}
```

Add user to group

You can add a user to a group by calling *TalentLMS_Group::addUser(array('user_id' => '{userId}', 'group_key' => '{groupKey}'))*. If the group key corresponds to an existing group and the requested user is not a member of the group, then the above call executes successfully and it returns a response in the following format.

```
{"user_id":"1", "group_id":"1", "group_name":"my group"}
```

The group assignment will have as a result the user to become member of the group's courses.

Remove user from group

You can remove a user from a group by calling *TalentLMS_Group::removeUser(array('user_id' => '{userId}', 'group_id' => '{groupId}'))*. If the group id corresponds to an existing group and the requested user is a member of the group, then the above call executes successfully and it returns a response in the following format.

```
{"user_id":"1", "group_id":"1", "group_name":"my group"}
```

Add course to group

You can add a course to a group by calling *TalentLMS_Group::addCourse(array('course_id' => '{courseId}', 'group_id' => '{groupId}'))*. If the group id corresponds to an existing group and the requested course is not a member of the group, then the above call executes successfully and it returns a response in the following format.

```
{"course_id":"1", "group_id":"1", "group_name":"my group"}
```

Branches

You can retrieve individual branches as well as a list of all your branches. To retrieve a specific branch you need to call *TalentLMS_Branch::retrieve({branchID})* where *{branchID}* is the unique identifier describing the requested branch. To retrieve all your branches you need to call *TalentLMS_Branch::all()*.

Retrieving a branch – Example response


```
{
"id":"1",
"name":"mybranch",
"description":"My first branch",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"theme":"Default",
"timezone":"(GMT +02:00) Athens, Istanbul, Minsk",
"signup_method":"manual",
"internal_announcement":"",
"external_announcement":"",
"language":"en",
"user_type_id":" 4",
"user_type":"Learner-Type",
"group_id":null,
"registration_email_restriction": null,
"users_limit": null,
"disallow_global_login":"0",
"payment_processor":"",
"currency":"US Dollar",
"paypal_email ":"",
"ecommerce_subscription":"0",
"ecommerce_subscription_price":"0",
"ecommerce_subscription_interval":"",
"ecommerce_credits":"0",
"users":[{"id":"1",
      "name":"D. Dummy"},
      {"id":"2",
      "name":"D. Dummy2"}],
"courses":[{"id":"1",
```

```
      "name":"Social media"},
      {"id":"12",
      "name":"Intro to TalentLMS"}]
}
```


Retrieving all branches – Example response


```
[{
"id":"1",
"name":"mybranch",
"description":"My first branch",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"theme":"Default",
"timezone":"(GMT +02:00) Athens, Istanbul, Minsk",
"signup_method":"manual",
"internal_announcement":"",
"external_announcement":"",
"language":"en",
"user_type_id":" 4",
"user_type":"Learner-Type",
"group_id":null,
"registration_email_restriction": null,
"users_limit": null,
"disallow_global_login":"0",
"payment_processor":"",
"currency":"US Dollar",
"paypal_email":"",
"ecommerce_subscription":"0",
"ecommerce_subscription_price":"0",
"ecommerce_subscription_interval":"",
"ecommerce_credits":"0"},
{"id":"2",
"name":"test",
"description":"",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"theme":"Green",
"timezone":"(GMT -03:00) Buenos Aires, Georgetown, Brasilia, Greenland",
```

```
"signup_method":"email",
"internal_announcement":"",
"external_announcement":"",
"language":"en",
"user_type_id":" 4",
"user_type":"Learner-Type",
"group_id":"2",
"registration_email_restriction": null,
"users_limit":"100",
"disallow_global_login":"0",
"payment_processor":"",
"currency":"US Dollar",
"paypal_email":"",
"ecommerce_subscription":"0",
"ecommerce_subscription_price":"0",
"ecommerce_subscription_interval":"",
"ecommerce_credits":"0"
}]
```


Create a new branch

You can create a new branch by calling *TalentLMS_Branch::create(array('name' => '{name}', 'description' => '{description}', 'disallow_global_login' =>*

'{disallowGlobalLogin}', 'group_id' => '{groupId}', 'language' => '{language}', 'timezone' => '{timezone}', ' signup_method' => '{signupMethod}', 'user_type' => '{userType}', 'registration_email_restriction' => '{registrationEmailRestriction}', 'users_limit' => '{usersLimit}', 'ecommerce_processor' => '{ecommerceProcessor}', 'currency' => '{currency}', 'paypal_email' => '{paypalEmail}', 'ecommerce_subscription' => '{ecommerceSubscription}', 'ecommerce_subscription_price' => '{ecommerceSubscriptionPrice}', 'ecommerce_subscription_interval' => '{ecommerceSubscriptionInterval}', 'ecommerce_credits' => '{ecommerceCredits}', 'internal_announcement' => '{internalAnnouncement}', 'external_announcement' => '{externalAnnouncement}', 'creator_id' => '{creatorId}')). All fields, except from 'name', are optional. The field 'creator_id' denotes the id of the user that will be the creator of the branch. If omitted then the creator is the owner of the account (super-administrator). If the provided arguments are valid, then the above call returns a response in the following format.


```
{"id":"10",
```

```
"name":"newbranch",
"description":"A new branch",
"avatar":"http://example.talentlms.com/pages/images/unknown_small.png",
"theme":"Default",
"timezone":"(GMT +02:00) Athens, Istanbul, Minsk",
"signup_method":"manual",
"internal_announcement":"",
"external_announcement":"",
"language":"en",
"user_type_id":" 4",
"user_type":"Learner-Type",
"group_id":null,
"registration_email_restriction":null,
"users_limit": null,
"disallow_global_login":"0",
"payment_processor":"",
"currency":"US Dollar",
"paypal_email":"",
"ecommerce_subscription":"0",
"ecommerce_subscription_price":"0",
"ecommerce_subscription_interval":"",
"ecommerce_credits":"0"}
```


Delete a branch

You can delete a branch by calling *TalentLMS_Branch::delete(array('branch_id' => '{branchId}', 'deleted_by_user_id' => '{deletedByUserId}'))*. The field 'deleted_by_user_id' is optional. This field represents the user who is responsible for the deletion of the branch. If omitted then the owner of the account (super-administrator) is responsible for the deletion. If the provided arguments are valid, then the above call returns a response in the following format.


```
{"message":"Operation completed successfully"}
```


Add user to branch

You can add a user to a branch by calling *TalentLMS_Branch::addUser(array('user_id' => '{userId}', 'branch_id' => '{branchId}'))*. If the requested user is not a member of the

branch, then the above call executes successfully and it returns a response in the following format.

```
{
"user_id":"1",
"branch_id":"1",
"branch_name":"mybranch"
}
```

Remove user from branch

You can remove a user from a branch by calling *TalentLMS_Branch::removeUser(array('user_id' => '{userId}', 'branch_id' => '{branchId}'))*. If the branch id corresponds to an existing branch and the requested user is a member of the branch, then the above call executes successfully and it returns a response in the following format.

```
{"user_id":"1", "branch_id":"1", "branch_name":"mybranch"}
```

Add course to branch

You can add a course to a branch by calling *TalentLMS_Branch::addCourse(array('course_id' => '{courseId}', 'branch_id' => '{branchId}'))*. If the branch id corresponds to an existing branch and the requested course is not a member of the branch, then the above call executes successfully and it returns a response in the following format.

```
{
"course_id":"1",
"branch_id":"1",
"branch_name":"mybranch"
}
```

Set status

You can change the status (active or inactive) of a branch by calling *TalentLMS_Branch::setStatus(array('branch_id' => '{branchId}', 'status' => '{status}'))*. The 'status' argument should be equal to 'active' or 'inactive'. If the above call executes successfully, it returns the id of the branch and its new status.

```
{
"branch_id":"2",
```

```
"status":"inactive"
}
```

Units

You can retrieve the score and completion status for unit – user pairs. To retrieve these details for all the users that tried a specific unit you need to call *TalentLMS_Unit::getUsersProgress(array('unit_id' => '{unitID}'))* where *{unitID}* is the unique identifier describing the requested unit. To retrieve the details for a specific unit – user pair you need to call *TalentLMS_Unit::getUsersProgress(array('unit_id' => '{unitID}', 'user_id' => '{userID}'))*.

Retrieving scores for all users – Example response

```
[{
"user_id":"1",
"status":"Completed",
"score":"100"},
{"user_id":"2",
"status":"Completed",
"score":"100"
},
{"user_id":"3",
"status":"In progress",
"score":"60"
}]
```

Retrieving scores for a specific user – Example response

```
[{
"user_id":"1",
"status":"Completed",
"score":"100"
}]
```

Get test answers

You can retrieve the answers of a user in a test by calling *TalentLMS_Unit::getTestAnswers(array('test_id' => '{testID}', 'user_id' => '{userID}'))*. If the test id corresponds to an existing test and the requested user is enrolled in the course containing the test, then the above call executes successfully and it returns a response in the following format.

Retrieving test answers – Example response


```
{
"test_id":"200",
"test_name":"My Test",
"user_id":"200",
"user_name":"D. Dummy",
"score":"100.00%",
"completion_status":"Passed",
"completed_on":"23/03/2015, 18:18:22",
"completed_on_timestamp":"1427127502",
"total_time":"5m 13s",
"questions":[{
                "id":"10",
      "text":"My question #10",
      "type":"Multiple choice",
      "weight":"1",
      "correct":"1",
      "answers":{"1":"yes","2":"no"},
      "correct_answers":{"1":"yes"},
      "user_answers":{"1":"yes"}},
      {"id":"20",
      "text":"My [option1|option2] question [option3|option4] #20",
      "type":"Fill the gap",
      "weight":"1",
      "correct":"1",
      "answers":{"1":"option1|option2","2":"option3|option4"},
      "correct_answers":{"1":"option1","2":"option3"},
      "user_answers":{"1":"option1","2":"option3"}},
      {"id":"30",
      "text":"My question #30",
      "type":"Ordering",
      "weight":"2",
      "correct":"1",
      "answers":{"1":"option1","2":"option2","3":"option3"},
      "correct_answers":{"1":"option1","2":"option2","3":"option3"},
      "user_answers":{"1":"option1","2":"option2","3":"option3"}},
      {"id":"40",
```

```
      "text":"My question #40",
      "type":"Drag and drop",
      "weight":"1",
      "correct":"1",
      "answers":{"1":" option1a <==> option1b ","2":" option2a <==> option2b"},
      "correct_answers":{"1":" option1a <==> option1b ","2":" option2a <==> option2b"},
      "user_answers":{"1":" option1a <==> option1b ","2":" option2a <==> option2b"}},
      {"id":"50",
      "text":"My question #50",
      "type":"Free text",
      "weight":"1",
      "correct":"1",
      "answers":[],
      "correct_answers":[],
      "user_answers":["my answer #50"]}]
}
```


Get survey answers

You can retrieve the answers of a user in a survey by calling *TalentLMS_Unit::getSurveyAnswers(array('survey_id' => '{surveyID}', 'user_id' => '{userID}'))*. If the survey id corresponds to an existing survey and the requested user is enrolled in the course containing the survey, then the above call executes successfully and it returns a response in the following format.

Retrieving survey answers – Example response


```
{
"survey_id":"100",
"survey_name":"My Survey",
"user_id":"200",
"user_name":"D. Dummy",
"completion_status":"Completed",
"completed_on":"23/03/2015, 14:00:23",
"completed_on_timestamp":"1427112023",
"total_time":"3m 42s",
```

```
"questions":[{
                "id":"1",
      "text":"My question #1",
      "type":"Multiple choice",
      "answers":{"1":"yes","2":"no","3":"maybe"},
      "user_answers":{"1":"yes"}},
      {"id":"2",
      "text":"My question #2",
      "type":"Multiple choice",
      "answers":{"1":"yes","2":"no"},
      "user_answers":{"1":"yes"}},
      {"id":"3",
      "text":"My question #3",
      "type":"Free text",
      "answers":[],
      "user_answers":["my answer #3"]},
      {"id":"4",
      "text":"My question #4",
      "type":"Free text",
      "answers":[],
      "user_answers":["my answer #4"]}]
}
```

Get ILT sessions

You can retrieve the sessions of an ILT unit by calling *TalentLMS_Unit::getIltSessions(array('ilt_id' => '{iltID}'))*. If the ILT id corresponds to an existing ILT unit, then the above call executes successfully and it returns a response in the following format.

Retrieving sessions for an ILT unit – Example response

```
[{
"id":"1",
"name":"Session #1",
"multiname":"",
"linked_to":"",
"type":"webinar",
```

```
"owner_id":"1",
"instructor_id":"1",
"description":"",
"location":"",
"start_timestamp":"1484906400",
"start_date":"20/01/2017, 12:00:00",
"capacity":"unlimited",
"duration_minutes":"60"
},
{"id":"2",
"name":"Session #2",
"multiname":"",
"linked_to":"",
"type":"classroom",
"owner_id":"1",
"instructor_id":"1",
"description":"",
"location":"",
"start_timestamp":"1485870300",
"start_date":"31/01/2017, 15:45:00",
"capacity":"100",
"duration_minutes":"120"
}]
```

Domain details

You can retrieve general details about your domain, including the total number of users, courses, categories, groups and branches in your domain. Also, you can retrieve the signup method (direct, captcha, email, admin or manual), the PayPal email address, the domain map of your domain and the monthly active users if you are subscribed in an unlimited plan. To retrieve these details you need to call *TalentLMS_Siteinfo::get()*.

Retrieving domain details – Example response

```
{
"total_users":"100",
"total_courses":"4",
"total_categories":"2",
"total_groups":"2",
"total_branches":"2",
"monthly_active_users":"30",
"signup_method":"direct",
"paypal_email":"",
"domain_map":"",
"date_format":"DDMMYYYY"
}
```

Possible date formats are: DDMMYYYY, MMDDYYYY and YYYYMMDD.

Rate limiting

You can check your current rate limit status at any time by calling TalentLMS_Siteinfo::getRateLimit().

Retrieving rate limits – Example response

```
{
"limit":"2000",
"remaining":"1999",
"reset":"1374757895",
"formatted_reset":"25/07/2013, 16:11"
}
```

'limit' represents the maximum number of requests that is permitted to make per hour. 'remaining' is the number of requests remaining in the current rate limit window. 'reset' represents the time at which the current rate limit window resets in UTC epoch seconds, while 'formatted_reset' represents this time in a readable format.

Note that calling this method does not count against your rate limit.

Get timeline

You can retrieve timeline entries for certain event types. To retrieve timeline entries you need to call *TalentLMS_Siteinfo::getTimeline(array('event_type' => '{eventType}'))* where *{eventType}* is the identifier describing the event type you want. Please refer to Appendix section for a complete list of available event types. For user/course/branch/group/unit related event types you can pass a second argument ('user_id' or 'course_id' or 'branch_id' or 'group_id' or 'unit_id') to get entries associated with that entity only. For example, you can retrieve all log–in events for a specific user or the courses that have been added to a specific group.

Note that the response contains the latest 200 timeline entries.

Retrieving certification issues for a user – Example response


```
[{
"action":"certification_issue_certification",
"message":"D. Dummy was awarded a certification for the course my course #1",
"timestamp":"15/12/2016, 12:49:02",
"unix_timestamp":"1481798942",
"user_id":"10",
"user_username":"dummy",
"user_email":"dummy@example.com",
"user_fullname":"D. Dummy",
"object_id":"1",
"object_name":"my course #1",
"event_counter":"1"
},
{"action":"certification_issue_certification",
"message":" D. Dummy was awarded a certification for the course my course #2",
"timestamp":"06/05/2015, 15:27:42",
"unix_timestamp":"1430915262",
```

```
"user_id":"10",
"user_username":"dummy",
"user_email":"dummy@example.com",
"user_fullname":"D. Dummy",
"object_id":"2",
"object_name":"my course #2",
"event_counter":"1"
},
{"action":"certification_issue_certification",
"message":" D. Dummy was awarded a certification for the course my course #3",
"timestamp":"30/03/2015, 14:18:17",
"unix_timestamp":"1427714297",
"user_id":"10",
"user_username":"dummy",
"user_email":"dummy@example.com",
"user_fullname":"D. Dummy",
"object_id":"3",
"object_name":"my course #3",
"event_counter":"1"
}]
```

Appendix A. Examples

The following source code shows a complete example of how to access the TalentLMS API, using the PHP library.

```
<?php

header('Content-Type: text/html; charset=utf-8');
require_once(dirname(__FILE__).'/talentlms/lib/TalentLMS.php');

try{
	TalentLMS::setApiKey('m8LnkM55HNyimpeq7usv3TRVlVpKwq');
	TalentLMS::setDomain('example.talentlms.com');

	$owner = TalentLMS_User::retrieve(1);
	$talentlmsIntro = TalentLMS_Course::retrieve(12);
	$samples = TalentLMS_Category::retrieve(3);
	$mybranch = TalentLMS_Branch::retrieve(1);
	$mygroup = TalentLMS_Group::retrieve(1);

	$users = TalentLMS_User::all();
	$courses = TalentLMS_Course::all();
	$categories = TalentLMS_Category::all();
	$branches = TalentLMS_Branch::all();
	$groups = TalentLMS_Group::all();
	$siteInfo = TalentLMS_Siteinfo::get();

	printf("My domain consists of: %d users, %d courses, %d categories, %d groups and %d branches", $siteInfo['total_users'], $siteInfo['total_courses'], $siteInfo['total_categories'], $siteInfo['total_groups'], $siteInfo['total_branches']);
}
catch(Exception $e){
	echo $e->getMessage();
}

?>
```

The following example shows the way to print the users of each branch in your TalentLMS domain.

```
<?php

header('Content-Type: text/html; charset=utf-8');
require_once(dirname(__FILE__).'/talentlms/lib/TalentLMS.php');

try{
	TalentLMS::setApiKey('m8LnkM55HNyimpeq7usv3TRVlVpKwq');
	TalentLMS::setDomain('example.talentlms.com');

	$branches = TalentLMS_Branch::all();

	foreach($branches as $branch){
		$currentBranch = TalentLMS_Branch::retrieve($branch['id']);
		$users = $currentBranch['users'];

		if(count($users) > 0){
			echo 'Users of branch <b>'.$currentBranch['name'].'</b>
are:<br/>';
			echo '<ul>';

			foreach($users as $user){
				echo '<li>'.$user['name'].'</li>';
			}

			echo '</ul>';
		}
	}
}
catch(Exception $e){
	echo $e->getMessage();
}

?>
```

The following example shows the way to print the users who have completed the most courses.

```
<?php

header('Content-Type: text/html; charset=utf-8');
require_once(dirname(__FILE__).'/talentlms/lib/TalentLMS.php');

try{
	TalentLMS::setApiKey('m8LnkM55HNyimpeq7usv3TRVlVpKwq');
	TalentLMS::setDomain('example.talentlms.com');

	$users = TalentLMS_User::all();
	$countCompletedCourses = array();

	foreach($users as $user){
		$currentUser = TalentLMS_User::retrieve($user['id']);
		$courses = $currentUser['courses'];
		$countCompletedCourses[$currentUser['id']] = 0;

		foreach($courses as $course){
			if($course['completed_on'] != ''){
				$countCompletedCourses[$currentUser['id']]++;
			}
		}
	}

	$maxs = array_keys($countCompletedCourses,
max($countCompletedCourses));
	$topUsers = array();

	foreach($maxs as $max){
		$topUser = TalentLMS_User::retrieve($max);
		$topUsers[] = $topUser['first_name'].' '.$topUser['last_name'];
	}

	if(count($topUsers)){
```

```
            echo 'Users who have completed the most courses are:
'.implode(",", $topUsers);
      }
}
catch(Exception $e){
      echo $e->getMessage();
}


?>
```

The following example shows the way to get detailed branch reports, such as the number of assigned/completed users in each course which is member of a branch.

```
<?php

ini_set('display_errors', false);
header('Content-Type: text/html; charset=utf-8');
require_once(dirname(__FILE__).'/talentlms/lib/TalentLMS.php');

echo '<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">';
echo "<html><head><title>Branch Reports</title></head><body>";

try{
      TalentLMS::setApiKey('3Q96EjNMS2eWQBbM2ivIfmgcENMycD');
      TalentLMS::setDomain('amakrid.talentlms.com');

      $branches = TalentLMS_Branch::all();      // get all branches

      foreach($branches as $branch){
                              // get info about this branch
            $branchInfo = TalentLMS_Branch::retrieve($branch['id']);
            $branchUsers = $branchInfo['users'];      // branch members
            $branchUsersIds = array();    // the ids of branch members

            foreach($branchUsers as $user){
                  array_push($branchUsersIds, $user['id']);
            }

            echo "<h3><u>Branch: {$branch['name']}</u></h3>";
                              // number of users in the branch
            echo "Number of users: ".count($branchInfo['users'])."<br/>";
                              // number of courses in the branch
            echo "Number of courses: ".count($branchInfo['courses'])."<br/>";
            echo "<ul>";

                              // iterate over the courses of the branch
```

```
            foreach($branchInfo['courses'] as $course){
                                        // get info about this course
                $courseInfo = TalentLMS_Course::retrieve($course['id']);
                $assignedUsers = 0;     // number of users in this course
                                        //   number   of   users   that
completed the course
                $completedUsers = 0;
                $learners = 0;                // number of learners
                $instructors = 0;       // number of instructors
                $completedUsersList = '';     // list of completed users
                $nonCompletedUsersList = '';  //  list  of  non  completed
users




                                        //  iterate  over  the  users  of
this course
                foreach($courseInfo['users'] as $user){
                                                        // check if user
is member of the branch
                      if(in_array($user['id'], $branchUsersIds)){
                            $assignedUsers++;

                            if($user['role'] == 'learner'){
                                  $learners++;
                            }
                            else if($user['role'] == 'instructor'){
                                  $instructors++;
                            }

                            if(!empty($user['completed_on'])){
                                  $completedUsers++;
                                  $completedUsersList                  .=
"<li><b>{$user['name']}</b>     -     {$user['role']}     (Completed     on:
{$user['completed_on']})</li>";
                            }
                            else{
```

```
                                    $nonCompletedUsersList                .=
"<li>{$user['name']} - {$user['role']}</li>";
                                }
                            }
                }
                                    echo "<li style='padding: 6px 0px;'>";
                echo "Course: <u><b>{$course['name']}</b></u><br/><br/>";
                echo "Number of assigned users: {$assignedUsers}<br/>";
                echo "Number of completed users: {$completedUsers}<br/>";
                echo "Number of learners: {$learners}<br/>";
                echo "Number of instructors: {$instructors}<br/>";

                if(isset($_GET['users']) && $_GET['users'] == '1'){
                      echo
"<br/><ol>{$completedUsersList}{$nonCompletedUsersList}</ol>";
                }

                echo "</li>";
            }

            echo "</ul>";
      }
}
catch(Exception $e){
      echo $e->getMessage();
}
echo "</body></html>";
?>
```

The following example shows how to activate all users in your account

```
<?php

ini_set('display_errors', false);
header('Content-Type: text/html; charset=utf-8');
require_once(dirname(__FILE__).'/talentlms/lib/TalentLMS.php');

try{
      TalentLMS::setApiKey('2ZgwpuGV97xZfFI4ag5agGwfCPgXB5');
      TalentLMS::setDomain('yourdomain.talentlms.com);

      $users = TalentLMS_User::all();

      foreach($users as $user){
          if ($user['status']!='active')
               TalentLMS_User::setStatus(array('user_id'   =>   $user['id'],
'status' => 'active'));
      }

catch(Exception $e){
      echo $e->getMessage();
}

?>
```

Appendix B. Mappings

Time zone mappings

```
-12 => "(GMT -12:00) Eniwetok, Kwajalein"
-11 => "(GMT -11:00) Midway Islands, Samoa"
-10 => "(GMT -10:00) Hawaii"
-9 => "(GMT -09:00) Alaska"
-8 => "(GMT -08:00) Pacific Time (US & Canada), Tijuana"
-7 => "(GMT -07:00) Mountain Time (US & Canada)"
-7b => "(GMT -07:00) Mountain Time, no DST (US & Canada)"
-6 => "(GMT -06:00) Central Time (US & Canada)"
-5 => "(GMT -05:00) Eastern Time (US & Canada)"
-44 => "(GMT -04:00) Chile"
-4.5 => "(GMT -04:30) Caracas"
-4 => "(GMT -03:00) Buenos Aires, Georgetown, Brasilia, Greenland"
-3.5 => "(GMT -03:30) Newfoundland"
-3 => "(GMT -03:00) Buenos Aires, Georgetown, Brasilia, Greenland"
-2 => "(GMT -02:00) Mid-Atlantic"
-1 => "(GMT -01:00) Azores"
0 => "(GMT 00:00) Greenwich Mean Time: Dublin, Edinburgh, Lisbon, London"
1 => "(GMT +01:00) Amsterdam, CopenHagen, Madrid, Paris, Vilnius, West
Central Africa"
2 => "(GMT +02:00) Athens, Istanbul, Minsk"
2b => "(GMT +02:00) Johannesburg"
3 => "(GMT +03:00) Moscow, St. Petersburg, Volgograd"
3.5 => "(GMT +03:30) Tehran +3:30"
4b => "(GMT +04:00) Dubai"
4 => "(GMT +04:30) Kabul"
5 => "(GMT +05:00) Islamabad, Karachi, Tashkent"
5.5 => "(GMT +05:30) Bombay, Calcutta, Madras, New Delhi"
6 => "(GMT +06:00) Columbo"
6.5 => "(GMT +06:30) Rangoon"
7 => "(GMT +07:00) Bangkok, Hanoi, Jakarta"
8 => "(GMT +08:00) Beijing, Chongqing, Hong Kong, Urumqi"
8b => "(GMT +08:00) Perth"
```

```
9 => "(GMT +09:00) Osaka, Sapporo, Tokyo"
9.5 => "(GMT +09:30) Darwin"
10 => "(GMT +10:00) Sydney"
10b => "(GMT +10:00) Brisbane"
11 => "(GMT +11:00) Magadan, Solomon Islands, New Caledonia"
12 => "(GMT +12:00) Fiji, Kamchatka, Marshall Islands, New Zealand"
```

Currency mappings

```
dollar => "US Dollar"
cad => "Canadian Dollar"
aud => "Australian Dollar"
nzd => "New Zealand Dollar"
sgd => "Singapore Dollar"
hkd => "Hong Kong Dollar"
jpy => "Japanese Yen"
brl => "Brazilian Real"
mxn => "Mexican Peso"
euro => "Euro"
pound => "UK Pound"
dkk => "Danish Krone"
nok => "Norwegian Krone"
sek => "Swedish Krona"
chf => "Swiss Franc"
try => "Turkish Lira"
rub => "Russian Ruble"
pln => "Polish Zloty"
czk => "Czech Koruna"
inr => "Indian Rupee"
php => "Philippine Peso"
thb => "Thai Baht"
ils => "Israeli New Shekel"
```

Timeline events mappings

```
User log in => "user_login_user"
User registration => "user_register_user"
User self registration => "user_self_register"
User deletion => "user_delete_user"
Undelete user => "user_undelete_user"
User update => "user_property_change"
User payment => "user_create_payment"
User level => "user_upgrade_level"
User badge => "user_unlock_badge"
Course creation => "course_create_course"
Course deletion => "course_delete_course"
Undelete course => "course_undelete_course"
Course update => "course_property_change"
Added user to course => "course_add_user"
Removed user from course => "course_remove_user"
User completed course => "course_completion"
User did not pass course => "course_failure"
Reset progress => "course_reset_user_progress"
Branch creation => "branch_create_branch"
Branch deletion => "branch_delete_branch"
Branch update => "branch_property_change"
Added user to branch => "branch_add_user"
Removed user from branch => "branch_remove_user"
Added course to branch => "branch_add_course"
Removed course from branch => "branch_remove_course"
Group creation => "group_create_group"
Group deletion => "group_delete_group"
Group update => "group_property_change"
Added user to group => "group_add_user"
Removed user from group => "group_remove_user"
Added course to group => "group_add_course"
Removed course from group => "group_remove_course"
Certification issued to user => "certification_issue_certification"
Certification renewed => "certification_refresh_certification"
```

```
Certification removed => "certification_remove_certification"
Certification expired => "certification_expire_certification"
Test completion => "unitprogress_test_completion"
Test fail => "unitprogress_test_failed"
Survey completion => "unitprogress_survey_completion"
Assignment submission => "unitprogress_assignment_answered"
Assignment grading => "unitprogress_assignment_graded"
ILT grading => "unitprogress_ilt_graded"
```