rEST API CaLL Samples

[Document Extract]



|  |
| --- |
| **REST API CALL SAMPLES** [Document Extract] |

**CONTENTS:**

Introduction. 2

Scope of Document. 2

Sample API Calls 2

Get Articles 2

Details 2

Parameters……………………………………………………………………………………………….3

<data> 3

<content> 4

Sample Resonse in JSON………………………………………………………………………………5

Articles Comments 5

Details 5

Parameters……………………………………………………………………………………………….6

<data> 6

<content> 6

Sample Resonse in JSON………………………………………………………………………………7

Sample Listing of API Calls 8

|  |
| --- |
| **INTRODUCTION** |

API calls are made via the [HTTP](http://www.w3.org/Protocols/rfc2616/rfc2616.html) methods [GET](http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.3), [POST](http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.5), [PUT](http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.6), [DELETE](http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.7), and [HEAD](http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.4). The responses return [status codes](http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html) indicating success or failure, along with any applicable [headers](http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html), and [JSON](http://www.json.org/) representing the affected fields (or nothing) in the [message-body](http://www.w3.org/Protocols/rfc2616/rfc2616-sec4.html#sec4.3). The following examples demonstrate common operations using these methods.

**Note:** Not all methods can be used on every data set. For a list of which methods you can use on which data set, please refer to the “Supported Operations” section on the respective Documentation page of the full document.

|  |
| --- |
| **SCOPE OF DOCUMENT** |

# The purpose of this document is to provide a sample of Synchronistic Wellness REST API implementation. This document is to provide an example of an API call and is for reference only. Your specific integration methodology will be customized and configured once the requirement gathering process has been completed.

Our user authentication and authorization system is following the industry’s OAuth (<http://oauth.net>). All API requests should include the app’s API ID number, so that our platform can verify the validity of the client making the request and track the respective API’s activity.

|  |
| --- |
| **SAMPLE API CALLS** |

GET is used to retrieve data without directly modifying it.

**GET Articles**

**Details:**

* Method: GET
* Action: action\_index
* Description: Get latest articles
* Example: <http://api.synchronisticwellness.com/articles/>

**Parameters:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Require** | **Description** | **Remarks** |
| api\_key | String | true | API Application Key |  |
| id | int | false | Article ID (When selecting a specific article) |  |
| title | string | false | title for search | Use this to search by `title` |
| page | int | false | Page Number |  |
| length | int | false | Number of items in one page (max length at 500) |  |
| lang | string | false | Language iso code | Default en |

**Response: <data>**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| page | int | Current page number |
| count | int | Number of items in current page |
| total\_count | int | Total number of all items |
| content | String | Content |

**Response: <content>**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Remarks** |
| id | int | Article ID |  |
| title | String | Article Title |  |
| content | String | Article Title Content |  |
| author\_id | String | Publisher's User id |  |
| author\_name | String | Publisher's User full\_name |  |
| date\_added | String | Time of publication | YYYY-MM-DD HH:MM:SS format |
| thumb | String | URL path of the thumbnail |  |

#### **Sample Response in JSON**

{

"info": {   
 "id": "users",   
 "version": 1   
 },  
 "status": {  
 "id": 0,  
 "message": "Ok"  
 },  
 "data": {  
 "page":1,  
 "count":20,  
 "total\_count":20  
 "content":[  
 {  
 "id":23,  
 "title":"RESTful Webservice",  
 "content":"abcdefghijklmnop..."  
 "author\_id":84988860,  
 "author\_name":"Markg",  
 "date\_added":"2010-07-09 16:29:47",  
 "thumb":"",  
 },  
 {....}  
 ]  
 }

}

**Articles Comments**

**Details:**

* Method: GET
* Action: action\_index
* Description: Get comments for specific article
* Example: <http://api.synchronisticwellness.com/articles/>comments/<version>/<client \_id>/<output>

**Parameters:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Require** | **Description** | **Remarks** |
| api\_key | String | true | API Application Key |  |
| article\_id | int | true | Article ID |  |
| page | int | false | Page Number |  |
| length | int | false | Number of items in one page (max length at 500) |  |

**Response: <data>**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| page | int | Current page number |
| count | int | Number of items in current page |
| total\_count | int | Total number of all items |
| content | String | Content |

**Response: <content>**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Remarks** |
| id | int | Comment ID |  |
| message | String | Comment message |  |
| author\_name | String | Comment's author name |  |
| author\_id | String | Comment's author it |  |
| author\_thumbnail | String | Comment's author thumbnail |  |
| date | String | Date of the comment | YYYY-MM-DD HH:MM:SS format |

#### **Sample Response in JSON**

{

"info": {  
 "id": "users",  
 "version": 1  
 },  
 "status": {  
 "id": 0,  
 "message": "Ok"  
 },  
 "data": {  
 "page":1,  
 "count":2,  
 "total\_count":"5",  
 "content":[{  
 "id":"24103373",  
 "message":"hah, lol",  
 "date":"9 minutes ago",  
 "author\_link":"\/profile\/feeds\/6606752",  
 "author\_thumbnail":"https:\/\/abc.synchronisticwellness.com\/user-images2\/user-images\/thumb\_1234\_411.png",  
 "author\_name":"Markg"  
 },  
 {...}]  
 }

|  |
| --- |
| **SAMPLE LIST OF API CALLS** |

# The following is a listing of many of the most important calls that you would expect for the API to perform:

**GET user/verify\_credentials**

Returns a representation of a user if the supplied credentials are valid; an error message if not.

**GET user/privacy**

Returns a list of the user’s privacy settings as booleans.

**POST user/notify**

Sends a notification to the user’s account with a string message.

**GET user/selfassessments/list**

Returns a list of the user’s self-assessment results.

**POST user/selfassessments/add**

Adds a new assessment to the the user’s assessment list.

**POST goals/complete**

Informs the server that the authenticated user has completed a goal for the given (current) day.

**GET goals/list**

Returns a list of all goals available in the database.

**GET goals/programs/list**

Returns a list of the goals created by users.

**GET goals/user/list**

Returns a list of the user’s past and present health goals. For each goal, contains an ID number, whether the goal has been completed, and, if the user’s privacy settings allow, a text representation of what the goal actually is. Active goals also contain days left until completion, whether the goal has been checked in to on the current day, and a list history of whether the goal has been checked in to on previous days.

**POST goals/user/add**

Adds a goal to the user’s list of goals.

**POST goals/user/edit**

Edit an indicated goal in the user’s list of goals, by index

**GET coaching/messages/list**

Returns a list of objects representing messages sent to or from a coach, each of which contains the sender, recipient, date and time, and message text. Returns an error message if user’s privacy settings forbid this.

**POST coaching/messages/create**

Informs the server that the authenticated user is sending a message to a coach. Includes sender, recipient, date, and time, and message text. Returns an error message if user’s privacy settings forbid this.

**GET coach/messages/list**

Returns a list of the coach’s clients, each one containing a list of the messages sent to and from that client. Each message contains the sender, recipient, date and time, and message text.

**POST coach/messages/create**

Informs the server that the authenticated coach is sending a message to a client. Includes sender, recipient, date, and time, and message text.

**GET coach/materials/list**

Returns a list of the coach’s materials.

**POST coach/materials/addblogpost**

Adds a new blog post to the coach’s materials list.

**POST coach/materials/addquestionnaire**

Adds a new questionnaire to the coach’s materials list.

**GET journal/list**

Returns a list of all the authenticated user’s journal entries, each containing the date, time, and text of the entry. Returns an error message if user’s privacy settings forbid this.

**POST journal/create**

Informs the server that the authenticated user is posting a new journal entry, containing the date, time, text of the entry. Returns an error message if user’s privacy settings forbid this.

**GET points\_balance**

Returns the authenticated user’s balance of Reward Points.

**GET library/list**

Returns a list of all materials in the library.