Project Server API

Overview

This is the documentation for the API you can use to communicate with our server. With it, a client can:

- Create a new user account.
- Sign in to an existing user account.
- Sign out from a user account.
- Add a high score to a user account.
- Get the top 10 high scores for all users.
- Get all his own high scores.

HTTP is used as the communication protocol, and each action is associated with its own URI and method. The data you should pass with each request and the data the responses contain are documented below in the section *API Documentation*. The first two actions use JSON, the two in the middle use XML and the last two use JSONP. This information is summarized in Table 1 below.

Action	Method	URI	Type
Create user	`POST`	`/users`	`JSON`
Login	`POST`	`/users/login`	`JSON`
Logout	`POST`	`/users/logout`	`XML`
Add high score	`POST`	`/scores/<username>`	`XML`
Get top 10 high scores	`GET`	`/scores`	`JSONP`
Get user's high scores	`GET`	`/scores/<username>`	`JSONP`

Table 1, Overview of the REST API.

For JSON and XML, the data is passed in the body of the HTTP requests and responses. It is necessary to set the `Content-Type` header to either `application/xml` or `application/json`. The server supports CORS, so you should be able to connect to it using the XHR object without having to upload your files to the server (if you use a modern browser that supports CORS).

Example:

```
var request = new XMLHttpRequest()
request.open("POST", "http://193.10.30.163/users/login", true)
request.setRequestHeader("Content-Type", "application/json")
request.addEventListener("load", function(){
  console.log("We got response from server!")
  console.log("Status code: ", request.status)
  console.log("Body: ", request.responseText)
})
request.send('{"email": "memberA@members.com ", "password": "memberA"}')
```

For JSONP, the data to the server is passed in the query string. The data from the server will be passed in the call to the specified callback function.

Example:

```
var script = document.createElement("script")
script.src = "http://193.10.30.163/scores?callback=myCallback&session= b4..c3"
function myCallback(response){
  console.log("We got response from server!")
  console.log("Status code:", response.status)
  console.log("Data:", response.data)
}
document.head.appendChild(script)
```

The server has the IP address `193.10.30.163`.

Times are represented as UNIX timestamps (number of *seconds* since Jan 01 1970 (UTC)).

At the end of the document, you can also read about how to upload your files to the server, so anyone with an Internet connection can visit your website and play your game.

API Documentation

Here are detailed instructions on how to use the different actions in the API.

Create new user account

Creates a new user on the server, or sends back error messages with explanations. In the body of the request template below, substitute the values in the object with your own values.

Request template

```
POST /users HTTP/1.1
Host: 193.10.30.163
Content-Type: application/json
Content-Length: 119

{"email": "memberA@members.com", "username": "MemberA",
"password": "memberA", "firstName": "Anna", "lastName": "Bas"}
```

Possible response templates

200 OK

The user was created.

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 4

null
```

400 Bad Request

Something is wrong with the request sent to the server (such as a missing property in the JSON-object).

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Content-Length: 4

null
```

415 Unsupported Media Type

The `Content-Type` header is wrong (the server only supports JSON for this resource).

```
HTTP/1.1 415 Unsupported Media Type
Content-Type: application/json
Content-Length: 4

null
```

422 Unprocessable Entity

The server cannot create the user as requested due to validation errors/conflicts. The errors can be found in the JSON-object in the body of the response. Here is what they mean:

- `emailTaken` – There already exists a user with the given email.
- `usernameTaken` – There already exists a user with the given username.
- `passwordEmpty` – The empty string is not allowed as password.

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: application/json
Content-Length: 68

{"emailTaken": false, "usernameTaken": true, "passwordEmpty": false}
```

Sign in to an account

Signs in to an account, or sends back error messages with explanations. In the body of the request template below, substitute the values in the object with your own values.

Request template

```
POST /users/login HTTP/1.1
Host: 193.10.30.163
Content-Type: application/json
Content-Length: 55

{"email": "memberA@members.com", "password": "memberA"}
```

Possible response templates

200 OK

A user account was found with the given email and password. Information about the user account can be found in the JSON object in the body of the response, as well as a session id. This session id can be passed along with upcoming requests for authentication. It will be valid until the user signs out or signs in again.

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 119

{"email": "memberA@members.com", "username": "MemberA",
"firstName": "Anna", "lastName": "Bas", "session": "b47...cc3"}
```

400 Bad Request

Something is wrong with the request sent to the server (such as a missing property in the JSON-object).

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Content-Length: 4

null
```

401 Unauthorized

No account with the given email and password were found. More information can be found in the JSON object in the body of the response.

```
HTTP/1.1 401 Unauthorized
Content-Type: application/json
Content-Length: 44

{"wrongEmail": true, "wrongPassword": false}
```

415 Unsupported Media Type

The `Content-Type` header is wrong (the server only supports JSON for this resource).

```
HTTP/1.1 415 Unsupported Media Type
Content-Type: application/json
Content-Length: 4

null
```

Sign out from an account

Destroys the user's session so it cannot be used anymore. In the body of the request template below, substitute the values in the object with your own values.

Request template

```
POST /users/logout HTTP/1.1
Host: 193.10.30.163
Content-Type: application/xml
Content-Length: 67

<?xml version="1.0"?>
<data>
  <session>b47...cc3</session>
</data>
```

Possible response templates

200 OK

The session id was successfully destroyed and can no longer be used for authentication.

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: 29

<?xml version="1.0"?>
<data/>
```

400 Bad Request

Something is wrong with the request sent to the server (such as a missing element in the `<data>` element).

```
HTTP/1.1 400 Bad Request
Content-Type: application/xml
Content-Length: 29

<?xml version="1.0"?>
<data/>
```

415 Unsupported Media Type

The `Content-Type` header is wrong (the server only supports XML for this resource).

```
HTTP/1.1 415 Unsupported Media Type
Content-Type: application/xml
Content-Length: 29

<?xml version="1.0"?>
<data/>
```

Add score

Adds a score for the user `<username>` (from the URI, to be replaced by the actual username). In the body of the request template below, substitute the values in the object with your own values.

Request template

```
POST /scores/<username> HTTP/1.1
Host: 193.10.30.163
Content-Type: application/xml
Content-Length: 87

<?xml version="1.0"?>
<data>
  <session>b47...cc3</session>
  <score>10</score>
</data>
```

Possible response templates

200 OK

The score has been added.

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: 29

<?xml version="1.0"?>
<data/>
```

400 Bad Request

Something is wrong with the request sent to the server (such as a missing element in the `<data>` element).

```
HTTP/1.1 400 Bad Request
Content-Type: application/xml
Content-Length: 29

<?xml version="1.0"?>
<data/>
```

401 Unauthorized

The session id passed in the request is no longer valid.

```
HTTP/1.1 401 Unauthorized
Content-Type: application/xml
Content-Length: 29

<?xml version="1.0"?>
<data/>
```

404 Not Found

There does not exist a user with the given `<username>`.

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: 29

<?xml version="1.0"?>
<data/>
```

415 Unsupported Media Type

The `Content-Type` header is wrong (the server only supports XML for this resource).

```
HTTP/1.1 415 Unsupported Media Type
Content-Type: application/xml
Content-Length: 29

<?xml version="1.0"?>
<data/>
```

Get top 10 high scores

Returns the top 10 high scores (sorted). The user must be logged in. In the URI in the request template below, substitute the values in the query string with your own values.

Request template

```
GET /scores?callback=myCallback&session=b47...cc3 HTTP/1.1
Host: 193.10.30.163
```

Possible response templates

Mimicked 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/javascript
Content-Length: XXX

myCallback({status: 200, data: {
  scores: [
    {username: "MemberA", firstName: "Anna", lastName: "Bas",
     score: 10, addedAt: 1463047887},
    ...
  ]
}})
```

Mimicked 400 Bad Request

Something is wrong with the request sent to the server (such as a missing parameter in the URI).

```
HTTP/1.1 200 OK
Content-Type: application/javascript
Content-Length: 37

myCallback({status: 400, data: null})
```

Mimicked 401 Unauthorized

The session id passed in the request is no longer valid.

```
HTTP/1.1 200 OK
Content-Type: application/javascript
Content-Length: 37

myCallback({status: 401, data: null})
```

Get user's high scores

Returns all the scores for the user `<username>` (from the URI) unsorted. In the URI in the request template below, substitute the values in the query string with your own values.

Request template

```
GET /scores/<username>?callback=myCallback&session=b47...cc3 HTTP/1.1
Host: 193.10.30.163
```

Possible response templates

Mimicked 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/javascript
Content-Length: XXX

myCallback({status: 200, data: {
  scores: [
    {score: 10, addedAt: 1463047887},
    ...
  ]
}})
```

Mimicked 400 Bad Request

Something is wrong with the request sent to the server (such as a missing parameter in the URI).

```
HTTP/1.1 200 OK
Content-Type: application/javascript
Content-Length: 37

myCallback({status: 400, data: null})
```

Mimicked 401 Unauthorized

The session id passed in the request is no longer valid.

```
HTTP/1.1 200 OK
Content-Type: application/javascript
Content-Length: 37

myCallback({status: 401, data: null})
```

Mimicked 404 Not Found

There does not exist a user with the given `<username>`.

```
HTTP/1.1 200 OK
Content-Type: application/javascript
Content-Length: 37

myCallback({status: 404, data: null})
```

Uploading your website to the server

In order to pass, you must upload your website to our web server. When you have done that, anyone with an Internet connection and a web browser should be able to play your game.

The File Transfer Protocol (FTP) is the most commonly used protocol to transfer files between clients and servers. There exist many free FTP programs one can use to do this, including:

- FileZilla: `https://filezilla-project.org`
- WinSCP: `https://winscp.net`

Feel free to use whichever you like. To connect to our server, use the settings shown in Table 2 below.

Host	`193.10.30.163`
Protocol	FTP (File Transfer Protocol)
User	`student`
Password	`db2AB4Ht`
Transfer mode	Must be *Passive* if you use the school's computers.

Table 2, Settings for connecting to the server using FTP.

With the settings shown in Table 2 above, you get access to a folder on the server at the URL `http://193.10.30.163/games`. All students in this course share this folder, so start by creating your own folder named after your JU ID in this folder, so we do not get any trouble with name clashing of files between students. When you have uploaded all your files to your own folder, you should be able to play your game by visiting `http://193.10.30.163/games/<your-ju-id>/the-html-file.html`. Why not try out your website in a web browser on a smartphone or a tablet?