# Lab: Asynchronous Programming

Problems for exercises and homework for the ["JavaScript Applications" course @ SoftUni.](https://softuni.bg/trainings/2249/js-applications-march-2019) Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1570>.

## Github Commits

Write a JS program that loads all commit messages and their authors from a github repository using a given HTML.

### HTML Template

You are given the following HTML:

|  |
| --- |
| commits.html |
| <!DOCTYPE **html**> <**html lang="en"**> <**head**>  <**meta charset="UTF-8"**>  <**title**>Github Commits</**title**>  <**script src="https://code.jquery.com/jquery-3.1.1.min.js"**></**script**>  <**style**>  **@import url(https://fonts.googleapis.com/css?family=Open+Sans)**;  **body** {  **font-family**: "Open Sans", serif;  }  **input[type=text**] {  **padding**: **5px 10px**;  **margin**: **8px 0**;  **display**: **inline-block**;  **border**: **1px solid #ccc**;  **border-radius**: **4px**;  }  **button** {  **background-color**: **#4caf50**;  **color**: **white**;  **padding**: **10px 14px**;  **margin**: **8px 0**;  **border**: **none**;  **border-radius**: **4px**;  **cursor**: **pointer**;  }  </**style**> </**head**> <**body**>  GitHub username:  <**input type="text" id="username" value="nakov"** /> <**br**>  Repo: <**input type="text" id="repo" value="nakov.io.cin"**/>  <**button onclick="***loadCommits*()**"**>Load Commits</**button**>  <**ul id="commits"**></**ul**>  <**script**>  **function** *loadCommits*() {  *//* **AJAX****call** *…* }  </**script**> </**body**> </**html**> |

The loadCommits() function should get the username and repository from the **HTML** textboxes with ids "username" and "repo" and make a GET request to the **Github API**:  
**"https://api.github.com/repos/<username>/<repository>/commits"**

Swap <username> and <repository> with the ones from the HTML:

* In case of **success**, for **each** entry add a list item (li) in the unordered list (ul) with id="commits" with text in the format:

"<commit.author.name>: <commit.message>"

* In case of an **error**, add a single list item (li) with text in the format:  
  "Error: <error.status> (<error.statusText>)"

### Screenshots:





Submit only the loadCommits() function in [Judge](https://judge.softuni.bg/Contests/1570) System.

## Blog

Write a program for reading blog content. It needs to make **requests** to the **server** and display **all blog posts** and their **comments**. Use the following HTML to test your solution:

|  |
| --- |
| blog.html |
| <!DOCTYPE **html**>  <**html**>  <**head**>  <**meta charset="UTF-8"**>  <**title**>Blog</**title**>  <**script src="https://code.jquery.com/jquery-3.1.1.min.js"**></**script**>  <**style**>  **@import url(https://fonts.googleapis.com/css?family=Open+Sans)**;  **body** {  **font-family**: **'Open Sans'**, **serif**;  }  **select** {  **padding**: **10px 15px**;  **margin**: **8px 0**;  **display**: **inline-block**;  **border**: **1px solid #ccc**;  **border-radius**: **4px**;  }  **button** {  **background-color**: **#4CAF50**;  **color**: **white**;  **padding**: **10px 15px**;  **margin**: **8px 0**;  **border**: **none**;  **border-radius**: **4px**;  **cursor**: **pointer**;  }  </**style**>  </**head**>  <**body**>  <**h1**>All Posts</**h1**>  <**button id="btnLoadPosts"**>Load Posts</**button**>  <**select id="posts"**></**select**>  <**button id="btnViewPost"**>View</**button**>  <**h1 id="post-title"**>Post Details</**h1**>  <**ul id="post-body"**></**ul**>  <**h2**>Comments</**h2**>  <**ul id="post-comments"**></**ul**>  <**script src="solution.js"**></**script**>  <**script**>  *attachEvents*();  </**script**>  </**body**>  </**html**> |

Submit only the attachEvents() function that attaches events to the buttons and contains all program logic. You will need to create a **Kinvey** **database** to test your code (instructions below).

The button with id="btnLoadPosts" should make a GET request to "**/posts**". The **response** from the **server** will be an **array of objects** in the following format:

{

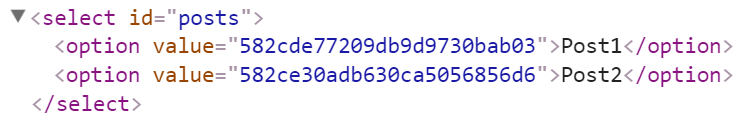
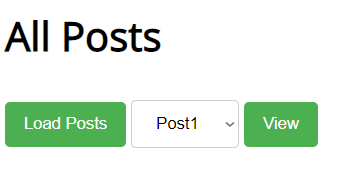
\_id: "postId",

title: "postTitle",

body: "postContent"

}

Create an <option> for each post using its \_id as value and title as text inside the node with id="posts".



When the button with id="btnViewPost" is clicked, a GET request should be made to "/posts/{postId}" to obtain the selected post (from the dropdown menu with id="posts") and another **request** to "/comments/?query={"post\_id":"{postId}"}" to obtain all comments. The **first request** will return **a single object** as described above, while the **second** will return an **array of objects** in the format:

{

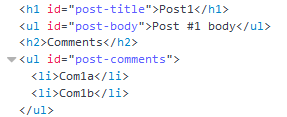
\_id: "commentId",

text: "commentContent",

post\_id: "postId"

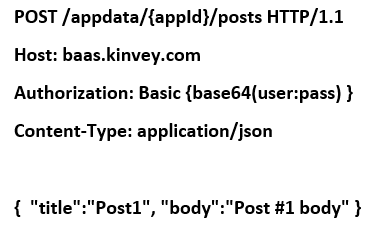
}

Display the post title inside "#post-title" and the post content inside "#post-body". Display **each comment** as a **<**li**>** inside "#post-comments" and don’t forget to clear its content beforehand.

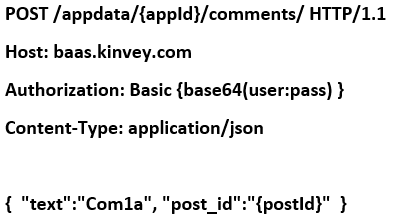


### Hints

* Create a **Kinvey database** with the required content.
* Then create a **user** and a **password**. You will need these, along with your **app ID** for authentication.
* Use the following **POST** request to **create** blog posts through **Postman**:



Note the **empty line** between the **header** and the **content** - the **request** **won’t work** without it. The authorization string consists of the **username** and **password** appended together with a **colon** between them, hashed with the btoa() function (built into the browser). The resulting post will have an \_id automatically assigned by Kinvey. You will then use this **ID** when creating comments for each blog post.



After the posts and comments are created, your database should look like this:

