

# Lab: Asynchronous Programming

Problems for exercises and homework for the ["JavaScript Apps" course @ SoftUni](#).

The following tasks do not have tests in the Judge system. They are for practice.

## Working with Remote Data

For the solution of some of the following tasks, you will need to use an up-to-date version of the **local REST service**, provided in the lesson's resources archive. You can [read the documentation here](#).

## 1. XHR (XmlHttpRequest)

Your task is to **write** a JS function that **loads** a github repository **asynchronously with AJAX**. You should **create** an instance of **XmlHttpRequest** and attach an **onreadystatechange** event to it. (An EventHandler that is called whenever the readyState attribute changes). Obtain the data by making a GET request to the following URL: **`https://api.github.com/users/testnakov/repos`**. In your event handler, when the **readyState** attribute reaches a value of **4** (it is ready), replace the text content of a **div** element with **id "res"** with the value of the **responseText** property of the request. **Do not format** the response in any way.

[More on XmlHttpRequest.open\(\)](#)

## Examples

Load Repos

Load Repos

```
[ { "id": 73502401, "node_id": "MDEwOlJlcG9zaXRvcnk3MzUwMjQwMQ==", "name": "Flappy-Nakov", "full_name": "MDQ6VXNlcjZlZDA2NDY1", "avatar_url": "https://avatars1.githubusercontent.com/u/23406465?v=4", "https://github.com/testnakov", "followers_url": "https://api.github.com/users/testnakov/followers", "following_url": "https://api.github.com/users/testnakov/gists{/gist_id}", "starred_url": "https://api.github.com/users/testnakov/s", "https://api.github.com/users/testnakov/subscriptions", "organizations_url": "https://api.github.com/users/testn", "https://api.github.com/users/testnakov/events{/privacy}", "received_events_url": "https://api.github.com/users/", "https://github.com/testnakov/Flappy-Nakov", "description": "A Javascript Game", "fork": true, "url": "https://api.g
```

## 2. Github Repos

Your task is to **write** a JS function that **executes** an **AJAX** request with **Fetch API** and loads all user **github repositories** by a given username (taken from an input field with **id "username"**) into a **list** (each repository as a **list-item**) with **id "repos"**. Use the properties **full\_name** and **html\_url** of the returned objects to create a link to each repo's GitHub page. If an **error** occurs (like 404 "Not Found"), **append** to the list a list-item with **text** the current instead. Clear the contents of the list before any new content is appended. See the **highlighted lines** of the skeleton for formatting details of each list item.

## Examples

GitHub username:  Load Repos

- [{repo.full\\_name}](#)

GitHub username:  Load Repos

- [k1r1L/Angular-2-Demos](#)
- [k1r1L/Angular-Sli.do](#)
- [k1r1L/awesome-interview-questions](#)
- [k1r1L/CSharp-Web-Development-Basics](#)
- [k1r1L/CSharp-Web-MVC-Frameworks-ASP.NET](#)
- [k1r1L/Databases-Advanced-Entity-Framework](#)
- [k1r1L/Databases-MS-SQL-Server-Exercises](#)
- [k1r1L/Express-Demo-Server](#)
- [k1r1L/express-js-exercises](#)
- [k1r1L/Front-End-Web-FMI-Project](#)
- [k1r1L/Fundamental-Level](#)
- [k1r1L/JavaScript-Advanced](#)
- [k1r1L/JavaScript-Applications](#)
- [k1r1L/JavaScript-Fundamentals](#)
- [k1r1L/React-Project](#)
- [k1r1L/SoftUni-ExpressJS-Fundamentals](#)
- [k1r1L/Softuni-Memes](#)
- [k1r1L/Tetris-JavaFundamentals-Teamwork](#)
- [k1r1L/TicTacToe](#)
- [k1r1L/University-Information-System](#)

## 3. Github Commits

Write a JS program that loads all commit messages and their authors from a github repository using a given 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 following format:  
"**<commit.author.name>**: **<commit.message>**"
- In case of an **error**, add a single **list item** (**<li>**) with text in the following format:  
"**Error: <error.status> (Not Found)**"

## Screenshots:

GitHub username:

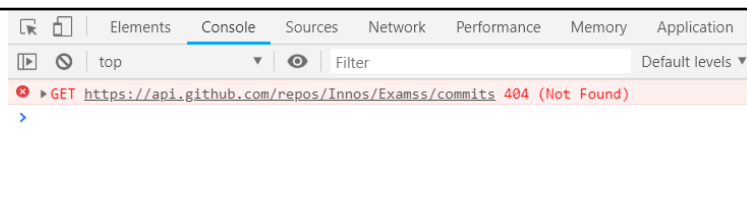
Repo:

- Svetlin Nakov: Delete Console.Cin.v11.suo
- Svetlin Nakov: Create LICENSE
- Svetlin Nakov: Update README.md
- Svetlin Nakov: Added better documentation

GitHub username:

Repo:

- Error: 404 (Not Found)



## 4. Cookbook – Part 1

The resources for this task are available in the following GitHub repository:

<https://github.com/viktorpts/js-apps-workshop>

You may check-out the repository or download the files via the green button labeled "Code" in the upper-right corner. Use the files located in **lesson-02/base** to begin the task. Before starting, make sure you have the most recent version of the repository. To see the solution, check the files inside **lesson-02/finished**.

Write a JS program that loads all recipes from the provided local server. You are **provided with skeleton (HTML & CSS)** for this task, also with **server**, which you will use as localhost. You will be able to load from the server "database" the needed recipes and other details.

### Load all recipes

When the app is started, you need to **load all the recipes** from the server:

# My Cookbook

Recipe 1



Recipe 2



Recipe 3



You have to make **"GET"** request to the server on this **URL**: <http://localhost:3030/jsonstore/cookbook/recipes>

## Load selected recipe

By **clicking on a card** with recipe you need to make a **"GET"** request to the server, and **toggle the information** only for the **selected recipe**.

The **URL** for the details is: <http://localhost:3030/jsonstore/cookbook/details/:id>

Where **":id"** is the id of the selected recipe.

## Recipe 2



### Ingredients:

- 500 g Ingredient 1
- 3 tbsp Ingredient 2
- 2 cups Ingredient 3

### Preparation:

Prepare ingredients

Mix ingredients

Cook until done