### **CE3606 :: JavaScript 2: Advanced JavaScript for websites and Web Applications**

# 

# 

# 

# 

# 

# Lecture 8

## **Exercises sheet**

## **Aristos Markogiannakis**

## 

## **Please use these notes together with my lecture presentation.**

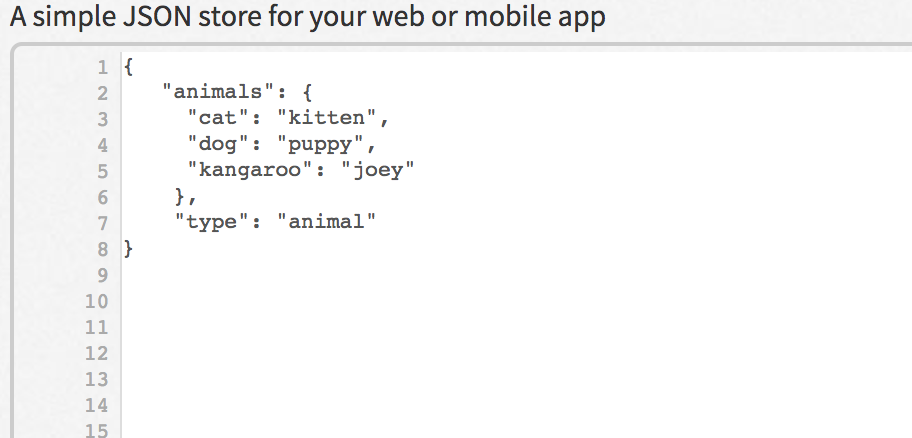
**Exercise 1:**

**Step 1:**

Navigate to the <http://myjson.com/> website

**Step 2:**

Add the following json to the page



**Step 3:**

Press Save

**Once you do this you should be able to get a url with your api**

**For example:**

[**https://api.myjson.com/bins/19h9tm**](https://api.myjson.com/bins/19h9tm)

**Step 4:**

We will now write some code to get that JSON and use it with JavaScript create a javascript file called exercise1.js and copy the following code.

fetch('https://api.myjson.com/bins/8i2vr')

.then(function(response) {

console.log(response.json());

}) // .then do something with the data

.catch(function(error) {

console.log('Fetch failed', error);

});

**Step 5:**

Create an html file called exercise1.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<title></title>

</head>

<body>

<div id='content'>

</div>

<script src="exercise1.js"></script>

</body>

</html>

**Step 6:**

To run your exericise you will need to install the node module – http-server

To do this run **npm install http-server**

Then on your command prompt **run http-server**

**Open your browser in http://localhost:8080**

**Check on your console the output from the fetch.**

**Run the console and see if you get the output.**

**Exercise 2**

Download the code from Moodle

**Step 1**

With the visual studio open the folder

**Step 2**

Open the terminal from the **View Menu → Integrated Terminal**

**Step 3**

Change directory to api directory by typing the command ‘**cd api**’

**Step 4**

Run npm install

**Step 5**

Stay on that directory and type “**node index.js**”

On your browser type <http://localhost:3001/>

You have three apis running

* [users](http://localhost:3001/users)
* [types](http://localhost:3001/types)
* [files](http://localhost:3001/files)

Users

|  |
| --- |
| [  {  "id": 1,  "givenName": "Peter",  "familyName": "Capaldi"  },  {  "id": 2,  "givenName": "Matt",  "familyName": "Smith"  },  {  "id": 3,  "givenName": "David",  "familyName": "Tennant"  },  {  "id": 4,  "givenName": "Christopher",  "familyName": "Eccleston"  },  {  "id": 5,  "givenName": "Jenna",  "familyName": "Coleman"  },  {  "id": 6,  "givenName": "Alex",  "familyName": "Kingston"  },  {  "id": 7,  "givenName": "Karen",  "familyName": "Gillan"  },  {  "id": 8,  "givenName": "Catherine",  "familyName": "Tate"  } ] |

Types

|  |
| --- |
| [  {  "creationDateTime": "2016-08-17T13:07:19.800Z",  "id": "article",  "documentsCount": 5,  "description": "Articles about the programme",  "name": "Article Page",  "colourId": "golden"  },  {  "creationDateTime": "2016-08-13T15:00:44.200Z",  "id": "profile",  "documentsCount": 6,  "description": "Actor/Actress profiles",  "name": "Profile Page",  "colourId": "spray"  } ] |

Files

|  |
| --- |
| [  {  "creationDateTime": "2015-10-07T07:39:27.947Z",  "status": "Approved",  "modifiedBy": 1,  "type": "article",  "uri": "/project/test/content/e9dd9a7d-8343-5faa-871b-d21b618f2f49",  "version": 2,  "id": "e9dd9a7d-8343-5faa-871b-d21b618f2f49",  "fileId": "Pivo-of.",  "scheduled": false,  "title": "Pivo of.",  "createdBy": 4,  "modifiedDateTime": "2015-10-07T15:39:27.947Z",  "live": false,  "popularity": 2  },  {  "creationDateTime": "2015-03-28T13:52:38.348Z",  "status": "Approved",  "modifiedBy": 3,  "type": "profile",  "uri": "/project/test/content/f896b536-ffe1-5863-a6e5-a64d15ee101f",  "version": 2,  "id": "f896b536-ffe1-5863-a6e5-a64d15ee101f",  "fileId": "Kabla-gomut.",  "scheduled": false,  "title": "Kabla gomut.",  "createdBy": 1,  "modifiedDateTime": "2015-03-28T14:52:38.348Z",  "live": false,  "popularity": 4  },  {  "creationDateTime": "2015-11-25T09:25:28.429Z",  "status": "Approved",  "modifiedBy": 7,  "type": "article",  "uri": "/project/test/content/52341adb-513e-5d20-8042-8eddd0196386",  "version": 1,  "id": "52341adb-513e-5d20-8042-8eddd0196386",  "fileId": "Lebhag-nes.",  "scheduled": false,  "title": "Lebhag nes.",  "createdBy": 1,  "modifiedDateTime": "2015-11-25T14:25:28.429Z",  "live": true,  "popularity": false  }  ] |

Some of these calls are very slow to respond so we need to use what we learnt on our previous class.

What we need to do is to read all three endpoints and create a combined output one that is the files with who created them.

And all the types with all files and users.

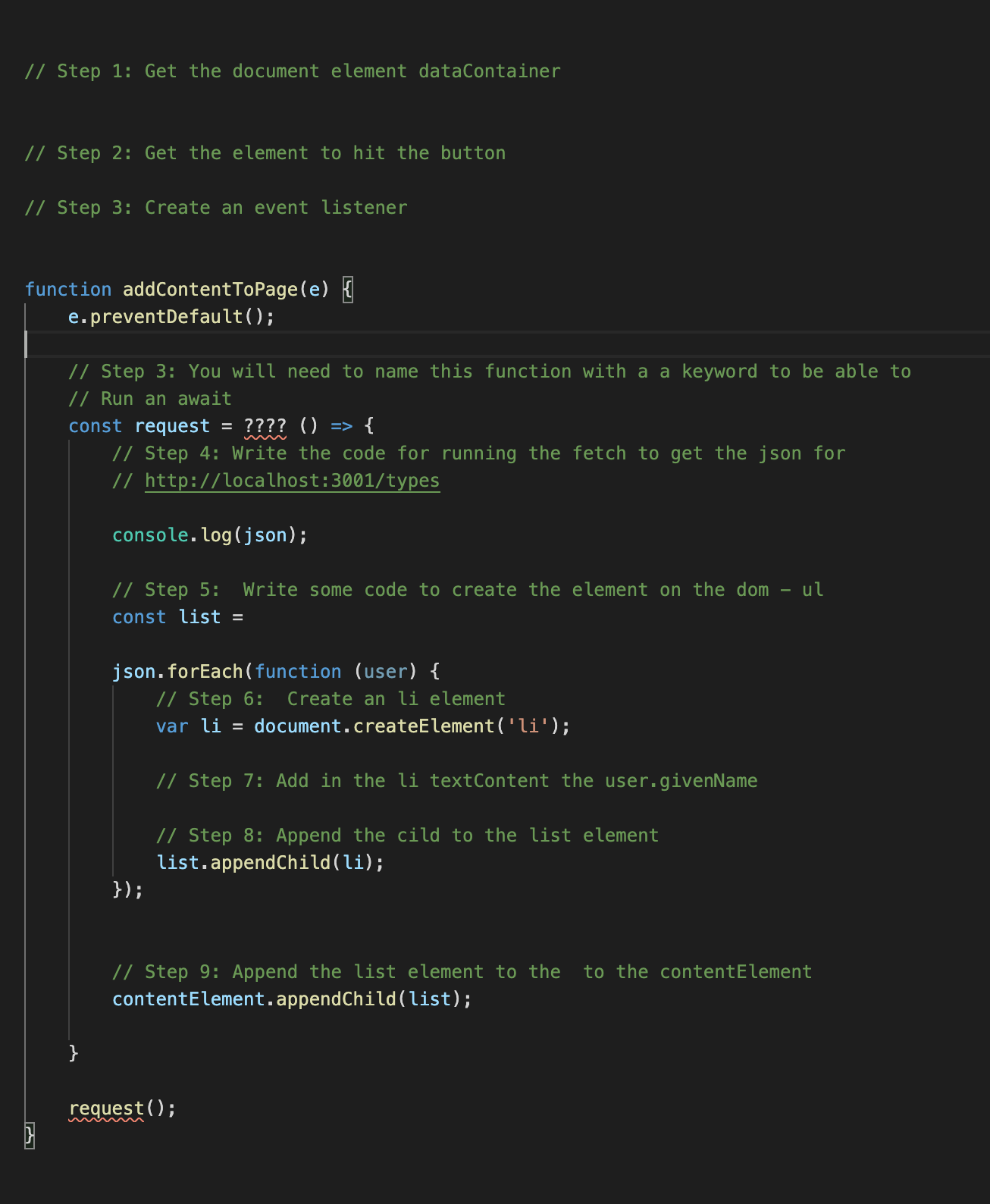
We will create four small code exercises based on the example shown on the board.

**Exercise1 - Print the types**

You will find inside the folder two files

* Exercise1.html (you don’t need to change it, **only open and load it**)
* exercise1.js

Follow the comments on what code you need to write



The end result should look like the following image (when you press the button Load Data you should see the names of the users api). You will have to use the types api for this and call this url **localhost:3001/types**

The end result should be a list of the type’s name

* Article Page
* Profile Page

**Exercise 2 - Print the files**

You will find inside the folder two files

* Exercise2.html (you don’t need to change it, **only open and load it**)
* exercise2.js

You will need to repeat the same exercise but this time load the files from the file api

* Kabla gomut.

The end result should be a list of all the files names

E.g

* Pivo of
* Kabla gomut.
* Kabla gomut.
* …

**Exercise 3**

You will find inside the folder two files

* Exercise3.html (you don’t need to change it, **only open and load it**)
* exercise3.js

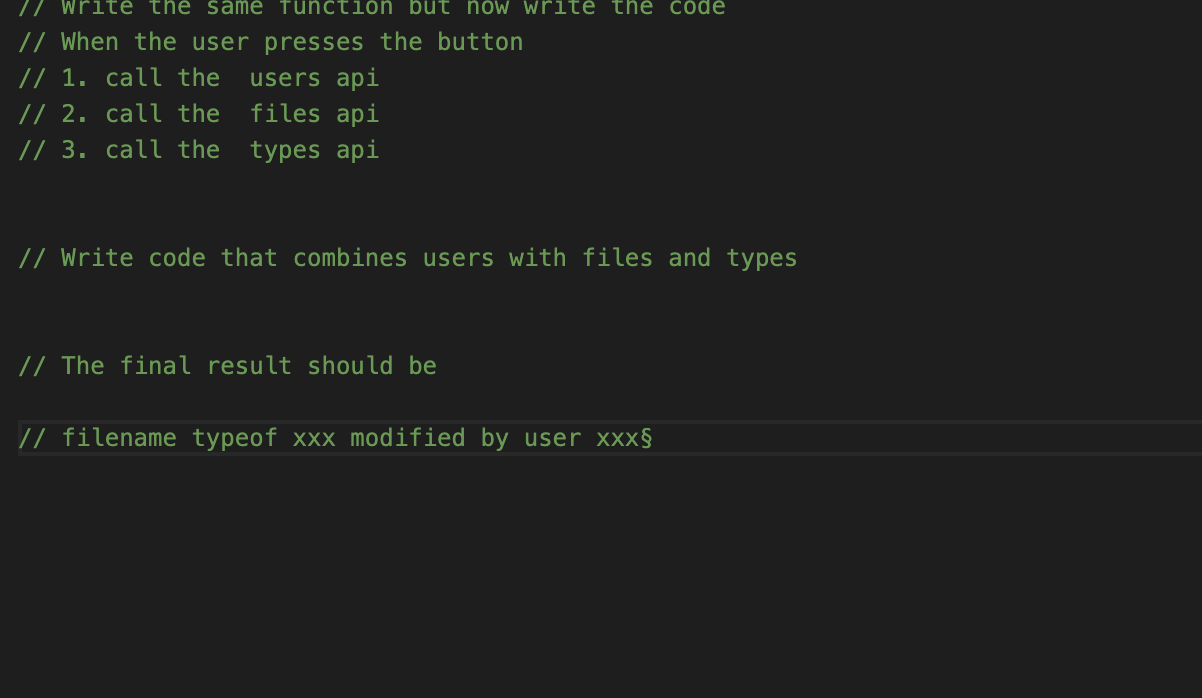
You will need to repeat the same exercise but this time load a combined output for files, users, and types from the file api

**localhost:3001/files**

**localhost:3001/users**

**localhost:3001/types**

Your exercise3.js file with the comments looks like this



The end result should be

* **Pivo of** was modfied by **Peter** and and it is an **Article Page**
* **Kabla gomut** was modfied by **David** and it is **Profile Page**
* **….**

The hightided values will be from each indvididual api.

For this exercise you will need to use the following

* Array map and filter methods for manipulating the data
* Fetch getting the data from the api
* Async and Await for synchronise all the calls and putting them in one thread