

## Homework: JSON Exercise

### 1. Objectives

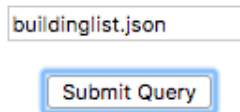
- Become familiar with the navigating JavaScript JSON objects;
- Use of JSON.parse parser and synchronous XMLHttpRequest;
- Transform the content of a JSON document into an HTML page.

### 2. Description

You are required to write an HTML/JavaScript program, which takes the URL of a JSON document containing the list of the tallest buildings in the world, parses the JSON file, and extracts the list of tallest buildings, displaying them in a table. The JavaScript program will be embedded in an HTML file so that it can be executed within a browser.

- Your program should display a text box to enter the JSON file name as shown below in Figure 1. On clicking the “Submit Query” button, your program should display the table as shown below, Figure 2. If the text box is left empty and Submit Query is clicked, an appropriate error message must be shown.

**Enter URL for list of tallest buildings JSON File**




The image shows a web form with a text input field containing the text "buildinglist.json". Below the input field is a button labeled "Submit Query". The button has a blue border and a slight shadow.

**Figure 1: Initial Page**

- Once the JSON file is downloaded, a JavaScript function within the HTML file parses the JSON document that was passed as an input in the edit box.
- After parsing the JSON document, a popup window or tab should display a table consisting of the data for largest manufacturers of cars on 2017 that are contained in the JSON file. For example, given the following JSON document:

<http://csci571.com/hw/hw4/buildinglist.json>

the table below should be displayed:




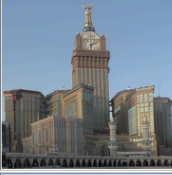


Rank	Name	City / Country	Height (ft)	Wiki Page	Image
1	Burj Khalifa	<ul style="list-style-type: none"> <li>• Dubai</li> <li>• United Arab Emirates</li> <li>• Currently the tallest building and tallest structure in the world from 2010 onwards; tallest building completed in the 2010s.</li> </ul>	2,717	<a href="https://en.wikipedia.org/wiki/Burj_Khalifa">https://en.wikipedia.org/wiki/Burj_Khalifa</a>	
2	Shanghai Tower	<ul style="list-style-type: none"> <li>• Shanghai</li> <li>• China</li> <li>• The tallest building in China and in East Asia and tallest twisted building.</li> </ul>	2,073	<a href="https://en.wikipedia.org/wiki/Shanghai_Tower">https://en.wikipedia.org/wiki/Shanghai_Tower</a>	
3	Abraj Al-Bait Clock Tower	<ul style="list-style-type: none"> <li>• Mecca</li> <li>• Saudi Arabia</li> <li>• Tallest building with a clock face. Tallest in Saudi Arabia.</li> </ul>	1,971	<a href="https://en.wikipedia.org/wiki/Abraj_Al_Bait">https://en.wikipedia.org/wiki/Abraj_Al_Bait</a>	
4	Ping An Finance Center	<ul style="list-style-type: none"> <li>• Shenzhen</li> <li>• China</li> <li>• Tallest building in Shenzhen and Guangdong Province. Shares the record of the highest observation deck with Shanghai Tower (#2) at 562 m.</li> </ul>	1,965	<a href="https://en.wikipedia.org/wiki/Ping_An_Finance_Centre">https://en.wikipedia.org/wiki/Ping_An_Finance_Centre</a>	

Figure 2: Table containing tallest buildings from buildinglist.json

Here is a version of the *buildinglist.json* file containing the data that is displayed above:

```
{
  "Mainline": {
    "Table": {
      "Header": {
        "Data": [
          "Rank",
          "Name",
          "City / Country",
          "Height (ft)",
          "Wiki Page",
          "Image"
        ]
      },
      "Row": [
        {
          "Rank": "1",
```

```

        "Name": "Burj Khalifa",
        "Hubs": {
            "Hub": [
                "Dubai",
                "United Arab Emirates",
                "Currently the tallest building and tallest structure in the world from 2010
onwards; tallest building completed in the 2010s."
            ]
        },
        "Height": "2,717",
        "HomePage": "https://en.wikipedia.org/wiki/Burj_Khalifa",
        "Logo": "https://csci571.com/hw/hw4/Burj_Khalifa.jpg"
    },

    {
        "Rank": "2",
        "Name": "Shanghai Tower",
        "Hubs": {
            "Hub": [
                "Shanghai",
                "China",
                "The tallest building in China and in East Asia and tallest twisted building."
            ]
        },
        "Height": "2,073",
        "HomePage": "https://en.wikipedia.org/wiki/Shanghai_Tower",
        "Logo": "https://csci571.com/hw/hw4/Shanghai_tower_dec_26,_2014.jpg"
    },

    [.....]

    ]
}
}
}

```

### 3. Error Handling

An error condition that should be checked for a JSON file containing NO buildings. An example of a JSON files which does not contain building entries:

```

{
  "Mainline": {
    "Table": {
      "Header": {
        "Data": [
          "Rank",
          "Name",
          "City / Country",
          "Height (ft)",
          "Wiki Page",

```

```

    "Logo"
  ]
}
}
}

```

In addition, your program should handle the case when the JSON file does not exist. A proper message should be displayed.

The “structure” of the input JSON files will not change. We will not test the case where one of the keys is missed. In other words, every *Row* always contains the keys: *Rank*, *Name*, *Hubs*, *Height*, *HomePage*, and *Logo*. The *Hubs* tag contains an array with key *Hub*. Note that The *Hub* array may contain ZERO or more values. The first Hub array element should be **boldfaced**.

If the value of a tag is empty, you should display a blank cell.

You are required to handle the case where the Header data values are different. Please note that the Data tag values might differ but the JSON structure remains the same. For example, the Header can be like this:

```

"Header": {
  "Data": [
    "Rank of building",
    "Name of Building",
    "Location Info",
    "Feet",
    "Wikipedia Page",
    "Logo"
  ]
},

```

No other error conditions need be checked. In all cases if an error is found your program should show an alert box indicating the error was detected.

## 4. Hints

- *Step 1: Writing Your HTML/JavaScript program - Using the DOM Parser*

Here's how you could use the Microsoft DOM API and the Mozilla DOM API (used in Firefox) to load and parse a JSON document into a DOM tree, and then use the DOM API to extract information from that document.

```
<script type="text/javascript">
var jsonDoc;
function loadJSON (url) {
    var xmlhttp=new XMLHttpRequest();
    xmlhttp.open("GET",url,false); //open, send, responseText are
    xmlhttp.send();               //properties of XMLHttpRequest

    jsonDoc=xmlhttp.responseText;
    return jsonDoc;
}
// ..... processing the document goes here
</script>
```

Now you can parse the JSON file with `JSON.parse` and generate the HTML table on the fly by navigating through the JSON object. You can assume that every JSON file will have identical Object, Array and key names.

Your task is to write a program that transforms this type of JSON file into the table as shown above.

- *Step 2: Display the Resulting HTML Document*

You should use the DOM `document.write` method to output the required HTML.

- *Step 3: Use JavaScript control syntax*

The only program control statements that you will need to use for this exercise are the “if” and the “for” statements. The syntax of both

statements is practically identical to the syntax of the corresponding statement in the C, C++ and Java languages, as in:

```
if (container_keys[j]=="Image") {  
    // do stuff  
}  
for (j=0; j<container_keys.length; j++) {  
    // do more stuff  
}
```

Please make a note of the following issue:

### **Cross-Site Scripting (XSS):**

JavaScript cannot call the resources from another domain. This is called cross side scripting which is not allowed in browsers. Therefore, you must put your JSON files and your script in the same domain. Please make sure, when you are testing your implementation, to place both the HTML file and the JSON file on your local machine IN THE SAME FOLDER. A sample file can be copied from here:

<http://csci571.com/hw/hw4/buildinglist.json>

Window.open() method must be used to pop up a new window which would display the final widget.

Image files can be either local or remote, as these files do not exhibit the cross-site scripting issue.

### **Scrollable Window:**

The popup window should be scrollable so the user can read all records listed in the window.

### **Freevar.com BODY tag capture**

Freevar.com web server (that will serve your code) looks for the end-BODY tag and tries to insert a “powered by Freevar.com” logo. If you have embedded in you JavaScript the string “</BODY>”, this will cause a JavaScript runtime error. The solution is to split the string “</BODY>” in two strings, as in:

```
"</bo" + "dy>"
```

## 6. Image Test Files

These image files are provided for testing purpose.

[http://csci571.com/hw/hw4/Burj\\_Khalifa.jpg](http://csci571.com/hw/hw4/Burj_Khalifa.jpg)

[http://csci571.com/hw/hw4/Shanghai\\_tower\\_dec\\_26,\\_2014.jpg](http://csci571.com/hw/hw4/Shanghai_tower_dec_26,_2014.jpg)

<http://csci571.com/hw/hw4/Abraj-al-Bait-Towers.JPG>

[http://csci571.com/hw/hw4/Ping\\_An\\_Finance\\_Center\\_201512.jpg](http://csci571.com/hw/hw4/Ping_An_Finance_Center_201512.jpg)

[http://csci571.com/hw/hw4/Goldin\\_Finance\\_117\\_3.jpg](http://csci571.com/hw/hw4/Goldin_Finance_117_3.jpg)

[http://csci571.com/hw/hw4/Lotte\\_world\\_tower.jpg](http://csci571.com/hw/hw4/Lotte_world_tower.jpg)

[http://csci571.com/hw/hw4/One\\_World\\_Trade\\_Center\\_cropped2.jpg](http://csci571.com/hw/hw4/One_World_Trade_Center_cropped2.jpg)

## 7. Material You Need to Submit

On your course homework page, your link for this homework should go to a page that looks like the one displayed in the Figure on page 1. **This page should include your entire JavaScript/HTML/CSS program in a single file.** Also, you should upload your source code electronically to the csci571 GitHub Classroom account so that it can be compared to all other students' code via the MOSS code comparison tool. If your submission is composed of multiple files, 3 points will be deducted.