



420-311-VA Internet Programming

Assignment #6

 Vanier College 
Computer Science Technology Department
Teacher: Sleiman Rabah (Fall 2020)

 **Due date:** as scheduled on LEA 

Revision History

Revision	Date	Author(s)	Description
1.0	November 12, 2020	S.R.	Initial handout.
1.1	November 17, 2020	S.R.	Added an XML document structure to follow and expanded the implementation details.


Contents

1	A Special Note	2
2	Learning Objectives	2
3	Online Resource and Documentation	2
4	Problem Statement	3
5	Implementation Details	3
5.1	Requirements and Constraints	3
5.2	Files and Folder to Create and Use	4
5.3	Steps to Follow	4
5.4	Structure of the XML File	5
5.5	Functions to Be Implemented	6
5.6	The Expected Web Interface	6
6	Evaluation criteria and Instructions	7
7	What to Submit	7

1 A Special Note

The instructions presented in this assignment require special attention, so please take the time to read the present document multiple times. Again, you must read the entire document multiple times before you start writing code.

You should write down the steps to be performed (that is, what needs to be done) before you start coding. Note that subsequent assignments might be based on this one.

NOTE: remember that this is an individual assignment. You are asked to write your own `</>` (code). Any similarity detected in any assignment(s) submitted by another student(s) will result in serious repercussions. Again, you have to write your own code. You are not allowed to share code with, or write code for, or look  at code of, another student. If you do so, you will be seriously penalized. If you have any questions, ask me for help/clarifications.

2 Learning Objectives

The objective of this assignment is to introduce you to the fundamental concepts of client-side programming with AJAX and server-side programming. The main learning objectives include: using AJAX (to create HTTP requests & and process HTTP responses), DOM API, XML, JSON, and Server-Side Programming with PHP, as well as deploying Web resources on a Web server.

In this assignment, we are going to use [XAMPP \[1\]](#) as a web server. For more details, please refer to the **XAMPP guide that was posted on LEA**.

3 Online Resource and Documentation

- * [Bootstrap Components](#) : where all Bootstrap's components are documented.
- * [Working with objects](#)
- * [Working with indexed collections](#)
- * [Defining functions](#)
- * [Getting started with AJAX](#)
- * [Parsing JSON data in JS](#)
- * [Introducing JSON](#)

4 Problem Statement

In this assignment, you are required to modify what you already implemented in the previous assignment and use AJAX to:

- * Fetch the list of items from the XML file and display their related information using Bootstrap cards. The list of items should be fetched when the HTML document has been completely loaded.
- * Implement a live and interactive search: the user of your application will be able to search for items whose information are stored in an ***XML*** file. Upon entering a character in the search input box, a list of matched items should be displayed without refreshing the Web document (see Section 5 for more details).




5 Implementation Details

This section provides you with details about what needs to be implemented and the steps to be followed. Please respect the requirements that are stated in the following sections.

5.1 Requirements and Constraints

Your implementation must **absolutely** respect the following requirements:

- * Use HTML/HTML5, [Bootstrap \[2\]](#) , and [Font Awesome \[3\]](#) to design and implement the web interface.
- * Use JavaScript only: unauthorized use of external code or libraries (such as jQuery) is strictly forbidden. You are required, and expected, to write your own code.
- * You are not allowed to use the *document.write()* and *document.createElement()* functions in this assignment.
- * You must use the *getElementById()* function and *innerHTML* property to dynamically change/update the content or the attribute's value of any HTML element.
- * **Proper error and exception handling.** Please refer to the following online resources [JavaScript error handling \[4\]](#) , and [JavaScript Control flow and error handling \[5\]](#) .
- * Use the [Camel Case \[4\]](#) naming convention in your implementation.
- * Choose consistent variable names. For instance, naming an array of numbers *myArray* will result in a grade deduction.
- * You must document your code (this include the documentation of each and every function) see [JSDoc \[6\]](#) for more details.

- * Your script should alter the content of an HTML document in order to display different kinds of content (the list of items, result of user searches, shopping cart's content, information about an item). However, *console.log()* can be also used for debugging purposes.
- * As a web browser, use  (Google Chrome) or  (Mozzila Firefox). Please do not use /e (Microsoft IE/Edge).
- * You can use use global variables in this assignment.

5.2 Files and Folder to Create and Use

The following are the main files to be created or used:

- * Use the same files that you submitted in the previous assignment.
- * Use the provided PHP file named *search.php*.
- * Create an *XML* file named *products.xml*.

5.3 Steps to Follow

The following are the steps to be done in order (one after the other):

1. Copy the files you used in Assignment #5 to a folder called **assignment6**
2. Deploy your **assignment6** folder on your local Web server.
3. In the **assignment6**, create an *XML* file and populate it with item definitions corresponding to your Item objects you used in Assignment #5 (see Section 5.4).
NOTE: your XML file must be stored in the same folder as *search.php*.
4. After populating your *XML* file with data, test the *search.php* file in the browser to make sure this file is properly parsing your *XML* elements and producing its corresponding *JSON* representation.
5. Implement the required functions (see Section 5.5)

5.4 Structure of the XML File

Your XML file must be structured as what follows. Please note that the names of the XML elements are **case sensitive**.

```

1      <?xml version="1.0" encoding="UTF-8"?>
2      <products>
3          <category>
4              <id></id>
5              <name></name>
6              <description></description>
7              <items>
8                  <item>
9                      <id></id>
10                     <title></title>
11                     <description></description>
12                     <price></price>
13                     ...
14                 </item>
15                 <item>
16                     <id></id>
17                     <title></title>
18                     <description></description>
19                     <price></price>
20                     ...
21                 </item>
22             </items>
23         </category>
24         <category>
25             <id></id>
26             <name></name>
27             <description></description>
28             <items>
29                 <item>
30                     <id></id>
31                     <title></title>
32                     <description></description>
33                     <price></price>
34                     ...
35                 </item>
36             </items>
37         </category>
38     </products>

```

Listing 1: The structure of the XML document to create.

5.5 Functions to Be Implemented

The following **new** functions are to be implemented. However, you are encouraged to implement other function(s) if you judge necessary.

1. ***makeAjaxCall()***: This function does not receive any parameters. It creates and configures an AJAX request and sends it to the provided *search.php*. It must be called to:
 - Fetch the list of items from the Web server. The following URI must be used: **search.php**
 - Perform a live search (that is, upon entering any character in the search box). The search keyword must be passed to the *search.php* script as parameter. The following URI must be used:
search.php?query=your_search_keyword**Hint:** for the interactive search feature, you need to use the proper JS event that triggers the call of this function.
2. ***processResponse()***: This function does not receive any parameters and should be called upon receiving the HTTP response of the HTTP request that was sent by the *makeAjaxCall()* function. It performs the following tasks:
 - It retrieves the obtained **JSON** document (that is, the list of matched products' details) embedded in the HTTP response and passes it as a parameter to ***parseJsonData()***.
 - After successfully populating the catalog array in ***parseJsonData()***, this function calls the ***showListOfItems()*** you implemented in the previous assignment.
3. ***parseJsonData(strData)***: It parses the received JSON data and populates both the *categories* and *catalog* arrays that you used in Assignment #5.
NOTE: You must use the ***JSON.parse()*** JS public function. You are not allowed to use any JS library such as jQuery in this assignment.

5.6 The Expected Web Interface

The Web interface must be designed and implemented using [Bootstrap \[2\]](#) and [Font Awesome \[3\]](#). It consists of a single HTML document whose main content will be dynamically updated according to the user's actions.

All what you have to do is:

1. Add the following Bootstrap components combined with [Font Awesome Icons](#) :
 - a) ***view items*** button: to show the list of all items.

- b) **view cart** button: to show the content of the shopping cart.
- c) **search box**: an HTML *input* element for entering a search keyword.
- d) **search** button: to search in the catalog by the entered search keyword.

6 Evaluation criteria and Instructions

Please read carefully, failing to respect the provided instructions will result in a major grade deduction.

- * The use of file names and functions exactly as provided.
- * Relevance and accuracy of the source code documentation as instructed in Section 5.1.
- * Completeness, correctness, and overall functionality of the implementation.
- * Compliance of the implementation with the stated problem as well as simplicity and appropriateness of your implementation.
- * Programming style, code readability, naming convention, and clarity see Section 5.1.
- * The user-friendliness of the web interface (presentation of the output).
- * Relevance of the test scenarios.
- * Proper validation and error handling.
- * Correctness of data validation.

7 What to Submit

- Compress all your files (.html, .js, .css, .php) and upload them to LÉA before the submission deadline.

References

- [1] Apache Friends, “XAMPP.” <https://www.apachefriends.org/index.html>, 2017. [Online; accessed 04-Apr-2017].
- [2] Bootstrap Project, “Bootstrap CSS Framework.” <http://getbootstrap.com/>, 2020. [Online; accessed 1-Oct-2020].
- [3] Font Awesome Project, “The iconic font and CSS toolkit.” <https://fontawesome.com/>, 2020. [Online; accessed 18-Oct-2020].
- [4] W3 Schools, “JavaScript Errors Handling.” https://www.w3schools.com/js/js_errors.asp, 2020. [Online; accessed 18-Oct-2020].
- [5] Mozilla Web Docs, “Control Flow and Error Handling.” https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Control_flow_and_error_handling, 2020. [Online; accessed 03-Sep-2020].

- [6] The JSDocs Documentation Porject, “JSDoc.” <http://usejsdoc.org/>, 2020. [Online; accessed 03-Sep-2020].
- [7] W3 Schools, “DOM API.” http://www.w3schools.com/js/js_htmlDOM_document.asp, 2016. [Online; accessed 18-Oct-2016].