SENG 401: Lab 2

Web Programming

Server-Side Programming: PHP, XML, and AJAX

Winter 2019

Due: 2 Weeks from the lab (February 14th)

Server-Side Programming: PHP, XML, and AJAX

PHP

PHP is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team. PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Preprocessor. PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter. (Ref: Wikipedia)

PHP is a powerful language to create dynamic and interactive websites that can be customized based on users' preferences. According to W3Techs more than 82% of servers in the whole world use PHP as server-side programming language.

In order to run webpages based on PHP, you need to have a web server (e.g. Apache) equipped with PHP. You can download and install ready-to-use packages that contain all Web Serving tools (e.g. Z-WAMP downloadable from http://zwamp.sourceforge.net/). All the webpages developed by PHP should be served from root folder of the server. Usually a folder is assigned for this purpose inside the major folder where Web Server (e.g. Apache or Z-WAMP) is installed. This unique folder is usually named either **htdocs** or **www**. In order to run a PHP file (e.g. test.php) you need to copy the file in the root (e.g. htdocs/test.php or www/test.php) and call it on your web browser. Since the root is usually served as **localhost**, in order to call your PHP file you need to type the URL as: http://localhost/test.php. If you are using a port other than 80 for your server you need to provide the port number in the URL. For example: http://localhost:81/test.php.

XML

In computing, Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures, such as those used in web services. Several schema systems exist to aid in the definition of XML-based languages, while programmers have developed many application programming interfaces (APIs) to aid the processing of XML data. (Ref: Wikipedia)

AJAX

Ajax (asynchronous JavaScript and XML) is a set of web development techniques using many web technologies on the client-side to create asynchronous Web applications. With Ajax, web applications can send data to and retrieve data from a server asynchronously (in the background) without interfering with the display and behavior of the existing page. By decoupling the data interchange layer from the presentation layer, Ajax allows for web pages, and by extension web applications, to change content dynamically without the need to reload the entire page. In practice, modern implementations commonly substitute JSON for XML due to the advantages of being native to JavaScript. (Ref: Wikipedia)

Practice

In order to learn PHP, XML and AJAX, refer to tutorials from W3SCHOOLS website at:

1. http://www.w3schools.com/php/default.asp and cover the following topics: **PHP Tutorial, PHP Forms, PHP-XML,** and **PHP-AJAX.**

2. http://www.w3schools.com/xml/default.asp and cover the following topics: **XML Tutorial, XML AJAX,** and **XML DOM.**

3. http://www.w3schools.com/js/js_ajax_intro.asp and cover the following topic: **JS AJAX.**

4. http://www.w3schools.com/jquery/default.asp and cover the following topic: **jQuery AJAX.**

Another useful source for PHP is its documentation at: http://php.net/. Through this course you may need to refer to PHP documentation several times to find appropriate functions and commands to accomplish a task.

Questions:

After you finished the PHP tutorial, answer the following questions:

1. What is the difference between **echo** and **print** commands?
2. What data types does PHP support?
3. What is the syntax to define **constants** in PHP?
4. List Arithmetic Operators in PHP and compare them with JavaScript's?
5. List Assignment Operators in PHP and compare them with JavaScript's?
6. List Logical Operators in PHP and compare them with JavaScript's?
7. List String Operators in PHP.
8. List Array Operators in PHP.
9. List all PHP Conditional Statements.
10. Define the syntax for different **for loops** in PHP.
11. What are the **array** types in PHP?
12. What is the difference between Tree-Based XML Parsers and Event-Based XML Parsers?

Tasks

PHP programs need to be run with a PHP interpreter. In our labs, we will use Apache to run the interpreter. For Tasks 1 and 5 you should create .php files (e.g. in the C:\Apache24\htdocs directory) to be accessed via a browser with support of PHP functionality. In the browser you should call each file using "http://localhost/phpexample1.php" URL template.

1. Scrutinize **json_decode** and **json_encode** functions from php.net and write a program in PHP to convert a GeoJSON file (e.g. CalgarySchools.geojson) into an array and vice versa?
2. Convert the program above into AJAX so the conversion request is triggered on client-side, process done on the server side and the result is sent from the server to the client.
3. Run the code from http://www.codexworld.com/convert-array-to-xml-in-php/ to see how to convert an array into XML.
4. Scrutinize **explode, trim, push_array, empty, is_real, is_numeric, is_string,** and **is_null** functions from php.net and develop a program that:
    a. Accepts the coordinates of two point from a form
    b. Tests that values are not null (empty values). If a value is not provided it should generate an appropriate message.
    c. Tests that the values are in the range (latitudes between -90 and +90 and longitudes between -180 and +180). If the values are not in the range the program should generate an appropriate message.
    d. Determine the quadrant of each point.
    e. Determine the bearing between two points.
    f. Determine the great circle distance (geodesic distance) between two points. (use Haversine formula. http://www.movable-type.co.uk/scripts/gis-faq-5.1.html)
5. Convert the program above into AJAX so the form collects coordinates on the client side and sends them to the server side for calculations. The results are sent back to the client and shown on the page without refreshing the page.

What/When/Who to submit?

There is 1 set of Questions. Answer them thoroughly in a document and keep them for your midterm and final exams.

There is 1 sets of Tasks. Each task is worth 4 points which makes the total of: (10 *4) = 40.

Total points of this lab is 40.

Tasks should be presented to your TA on the session of February 14th.