Introduction to Server-Side and Client-Side Scripting

by Sophia

WHAT'S COVERED

In this lesson, you will learn more about the process and application of server-side scripting for web development. You will be introduced to the common server-side scripting languages and their uses and strengths. Additionally, you will be introduced to the popular server-side scripting language, PHP.

Specifically, this lesson will cover the following:

1. Introduction to Server-Side Scripting
2. Server-Side Scripting Concepts
3. Introduction to PHP

1. Introduction to Server-Side Scripting

Recall that there are two sides to scripting for websites and web applications: client side and server side. Furthermore, scripting is the ability to write programming code that will be executed either by a webserver or a client's browser in order to manipulate content within the Document Object Model (DOM) and provide functionality to the user interface.

As a brief recap, remember that JavaScript is the primary scripting language used on the client side. JavaScript is executed by the client's web browser and is able to manipulate content on the webpage in real time. However, once the HTML, CSS, and JavaScript code reaches the web browser, JavaScript is pretty much on its own and cannot access resources on the webserver, with the exception of issuing asynchronous AJAX calls to the resources on the server.

Let us explore the concepts and applications of server-side scripting. We will briefly discuss some of the options for server-side scripting languages, but we will focus on PHP.

2. Server-Side Scripting Concepts

Server-side scripting refers to code that resides on and is executed by the webserver. The script files on the server can be used in a number of ways:

1. To generate entire webpages with real-time data
2. To connect to and request data from a database
3. To generate a fragment of HTML code
4. To transmit data to be stored in a database
5. To call other scripts
6. To generate a response or confirmation email
7. And a lot more

In general, scripting languages on a webserver can have access to a wide range of resources and other systems. Furthermore, there are additional frameworks and add-ons that can be included to add new features, increase productivity, reduce development time, and so on.
Using server-side scripting begins by ensuring the **scripting language's processing engine** is installed and configured on the webserver. When using the Microsoft webserver **Internet Information Service (IIS)**, support for **ASP.NET** scripts comes pre-installed. However, if you want to use PHP, Ruby, **Java**, or server-side JavaScript using NodeJS, then these will need to be installed onto the webserver and configured. The configuration of the scripting engine is beyond the scope of this course but typically involves associating file extensions like .js, .php, or .cs with the correct scripting engine. This way, when an HTTP request comes into the server and requests a file such as "loginHandler.php," as the requested file contains the .php file extension, the server will first pass the file to the PHP scripting engine and then return the resulting content back to the client.

Below is a table of commonly used server-side scripting languages:

Title	Description
ASP.NET C# / VB	ASP (Active Server Page) runs on Microsoft's powerful .NET framework and is a general-purpose scripting language. Additionally, ASP.NET supports both C# (C-Sharp) as well as Visual Basic (VB) and can be developed using Microsoft's Visual Studio IDE applications.
PHP	PHP is an open-source, general-purpose, server-side scripting language. Being open source, PHP provides an incredible amount of flexibility and compatibility with other technologies.
Python	**Python** is a general-purpose, server-side scripting language that has gained popularity due to its powerful yet simplified coding syntax. Furthermore, lots of frameworks and extensions have been developed for and around Python.
NodeJS	NodeJS is not just a server-side scripting language but is also a runtime environment that can run and host its own applications and webserver. NodeJS is often used to program webservers themselves, as well as provide scripting support for existing webservers through server-side JavaScript.

Ruby	Ruby is a general-purpose, server-side scripting language designed with the goal of making programming fun. Ruby removes some of the tedious and problematic burdens of memory management and concurrency.
Java	Java is a powerful server-side and general-purpose programming language that supports object-oriented coding and concurrency and is designed to be executable on most digital computer systems. Java introduced the term “write once, run anywhere.”
GoLang	GoLang is one of the newer programming languages from Google that was designed for lightweight web services that run on mobile devices. The main benefit of the lightweight web service architecture is that it enables integration with a wide range of other programming languages.

So, how does one go about choosing the ideal language for one's project?

The first consideration would be the project requirements and the type of application being developed. For example, PHP tends to be ideal for content management systems and e-commerce platforms, while Python is more popular for data analysis and machine learning.

The second consideration is the current or desired computing environment. The current computing environment (i.e., Microsoft vs. Apple vs. Linux, etc.) may also introduce compatibility or performance concerns with different scripting languages. One example is the use of **Ruby on Rails**, which is a framework that runs on top of Ruby and requires the **Apache** or **Nginx** webserver. These two webservers run better on Linux-based operating systems.

Additional considerations include the following:

- Scripting language experience: Developers often have a personal preference based on familiarity with a specific scripting language(s).
- Cost: Budget and resources as some scripting languages are not free, and some require extensive hardware support.
- Debugging and testing options: Some scripting frameworks are supported by different IDEs, which will provide external tools for testing and debugging. However, there are some scripting frameworks that include their own testing and debugging tools.
- Simplification: In some cases, it may be possible to reduce the complexity of the development process by using a scripting framework that relies on a single language for both the front as well as the back end. For example, server-side JavaScript can simplify a project by relying solely on JavaScript as the front- and back-end scripting language.

Scripting Language Processing Engine
Server-based software that executes language-specific code contained within a script file, replacing the script code with the text-based output that results from the code’s execution.

Internet Information Service (IIS)

Server-based webserver software developed by Microsoft that hosts files and resources online and responses to HTTP requests.

ASP.NET
Server-side framework designed for web development and server-side scripting used for creating dynamic webpage content.

Java
A common, high-level object-oriented programming language that can be used as a server-side scripting language, as well as platform-independent applications.

Python
A popular, high-level general-purpose programming language that works well as a server-side scripting language.

Ruby on Rails
A server-side web application framework designed for rapid development of model-view-controller-based web services, pages, and databases.

Apache
A popular, free, open-source HTTP server.

Nginx
A free, flexible, open-source HTTP server.

---

3. Introduction to PHP

PHP (PHP: Hypertext Preprocessor) is a powerful, general-purpose, open-source scripting language. PHP can be used to generate entire webpages, including HTML, CSS, and JavaScript, as well as create small or complex web services. PHP is also a great language for handling web form submissions, generating emails, connecting to databases, and more.

PHP files are plain-text files that have the .php file extension. Any and all PHP code must be surrounded in the following syntax:

EXAMPLE PHP script container:

```
<?php //code goes here ?>
```

The PHP script container can be placed anywhere within a .php file and will be executed and removed from the code file, leaving behind any resulting output of the code. Once the file has been executed, the remaining code and data get transmitted back to the system that made the request. This allows us to use PHP in two key ways:

1. Build an entire HTML and CSS webpage with PHP containers and code strategically placed throughout the document and saved as a .php file.

a. When the page, such as about.php, is requested by a client,
b. the server passes about.php to the PHP engine for processing;
c. any HTML, CSS, or JavaScript code is left alone;
d. any <?php ?> containers are processed, removed, and their results left in their place;
e. and, finally, the resulting DOM is returned to the requesting client.

This option gives us the ability to dynamically create webpages each time they are requested by a user. This can include looking up user preferences and information by comparing the cookies on the user's system, matching them to the user in the database, and applying them to the page's configuration and welcome message. On the other hand, we can also use PHP scripts to process simple requests, retrieve data values from the server or database, and a lot more.
2. Process an operational request, such as a form submission, login, registration, and so on.

a. A form's action attribute can be used to call a script, such as registerUserHandler.php, or an AJAX request can be used to call a script, such as checkExistingUser.php.
b. When the user enters their desired username in the registration form, an onChanged event can trigger the AJAX call to checkExistingUser.php, which compares what the user entered as their username against the database of existing users for a match. The script can echo back a response in text to indicate a success (no match found) or an error (username already exists).
c. Then, when the user submits the form, the data in the fields get packaged and sent to the targeted script.
d. The script receives the form data, establishes a connection to the database, and transmits the data to the database in order to register the new user.

These types of PHP scripts are stand-alone scripts that can receive data (such as a form submission, JSON data structures, or URL parameters), perform any necessary operations on the server (such as connecting to a database or loading a particular file), and then use the echo command to return the data to the requesting system. PHP output commands will be covered more later in this tutorial.
The great thing about server-side scripts is that they can be triggered by just about any event by simply tying the event to a JavaScript function that issues an AJAX request.

SUMMARY

In this lesson, you learned more about the process of how **server-side scripting** is utilized to generate dynamic content prior to the webpage being sent to the client. Additionally, you learned about some of the different **server-side scripting concepts** and languages. Lastly, you were formally **introduced to the PHP** server-side scripting language.

Source: This Tutorial has been adapted from "The Missing Link: An Introduction to Web Development and Programming " by Michael Mendez. Access for free at https://open.umn.edu/opentextbooks/textbooks/the-missing-link-an-introduction-to-web-development-and-programming. License: Creative Commons attribution: CC BY-NC-SA.

TERMS TO KNOW

ASP.NET
Server-side framework designed for web development and server-side scripting used for creating dynamic webpage content.

Apache
A popular, free, open-source HTTP server.

Internet Information Service (IIS)
Server-based webserver software developed by Microsoft that hosts files and resources online and responses to HTTP requests.

Java
A common, high-level object-oriented programming language that can be used as a server-side scripting language, as well as platform independent applications.

Nginx
A free, flexible, open-source HTTP server.

Python
A popular, high-level general-purpose programming language that works well as a server-side scripting language.

Ruby on Rails
A server-side web application framework designed for rapid development of model-view-controller-based web services, pages, and databases.

Scripting Language Processing Engine
Server-based software that executes language-specific code contained within a script file, replacing the script code with the text-based output that results from the code's execution.