Introduction to PHP

Abdulla Ebrahim Subah

University of Bahrain

October 1, 2024

Static Sites

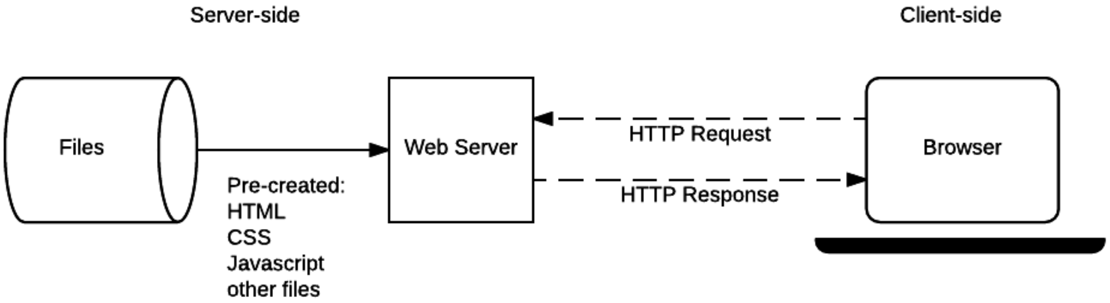

Figure 1: Basic Static App Server

Image Source: MDN Web Docs

Dynamic Sites

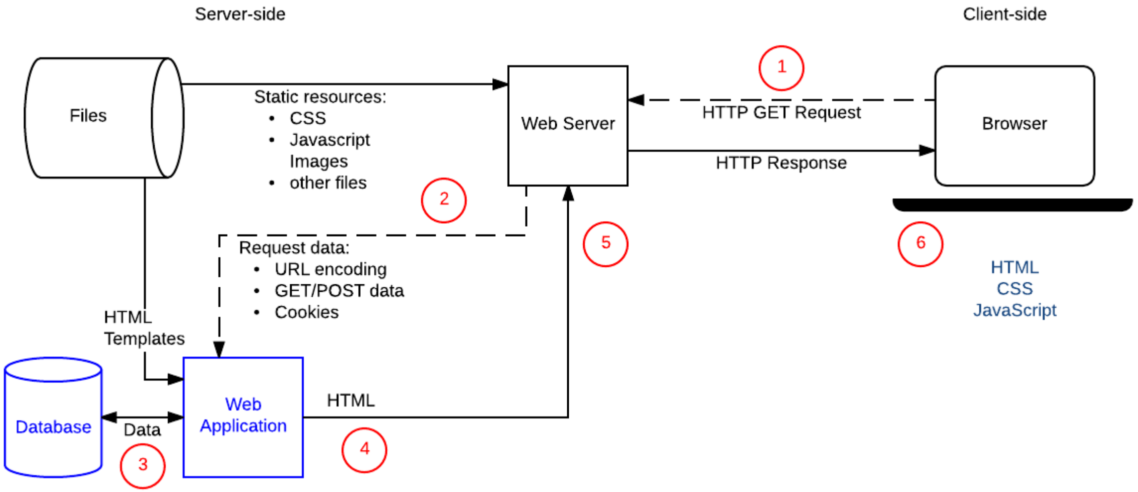

Image Source: MDN Web Docs)

Server-Side Languages

Server-side languages are used to create dynamic web pages. They run on the server and generate HTML that is sent to the client's browser. Examples include:

- **PHP**: Widely used for web development and can be embedded into HTML.
- **Python**: Known for its readability, often used with frameworks like Django and Flask.
- **Ruby**: Popular for its simplicity and productivity, commonly used with Ruby on Rails.
- **JavaScript (Node.js)**: Allows JavaScript to be used for server-side scripting, enabling full-stack development with a single language.

Server-Side Databases

Server-side databases are used to store and manage data for web applications. They run on the server and can be accessed by server-side languages to perform CRUD (Create, Read, Update, Delete) operations. Examples include:

- **MySQL**: An open-source relational database management system.
- **PostgreSQL**: A powerful, open-source object-relational database system.
- **SQLite**: A self-contained, serverless, and zero-configuration database engine.
- **Microsoft SQL Server**: A relational database management system developed by Microsoft.
- **Oracle Database**: A multi-model database management system produced by Oracle Corporation.

NoSQL Databases

NoSQL databases are designed to handle large volumes of data and are optimized for specific data models. Examples include:

- **MongoDB**: A document-oriented NoSQL database known for its flexibility and scalability.
- **Cassandra**: A wide-column store NoSQL database designed for high availability and scalability.
- **Redis**: An in-memory key-value store known for its speed and performance.
- **CouchDB**: A document-oriented NoSQL database that uses JSON to store data.
- **Neo4j**: A graph database that uses graph structures for queries with nodes, edges, and properties.

Web Servers

Web servers are software applications that serve web pages to users upon request. They handle HTTP requests from clients (browsers) and deliver the requested resources. Examples include:

- **Apache HTTP Server**: A widely-used open-source web server known for its flexibility and power.
- **Nginx**: A high-performance web server and reverse proxy server known for its speed and scalability.
- **Microsoft Internet Information Services (IIS)**: A web server created by Microsoft for use with Windows Server.

Selected Stack for This Course

In this course, we will be using the following stack for our server-side development:

- **Apache HTTP Server**: A widely-used open-source web server known for its flexibility and power. It will handle HTTP requests and serve web pages to users.
- **MySQL/MariaDB**: Both are open-source relational database management systems. MySQL is known for its reliability and performance, while MariaDB is a fork of MySQL with additional features and improvements.
- **PHP**: A popular server-side scripting language that is widely used for web development. It can be embedded into HTML and is known for its ease of use and flexibility.

This stack is commonly referred to as the AMP stack (Apache, MySQL/MariaDB, PHP) and is a powerful combination for building dynamic web applications.

Introduction to PHP

Wikipedia:

> ***PHP*** *is a general-purpose scripting language geared towards web development. It was originally created by Danish-Canadian programmer Rasmus Lerdorf in 1993 and released in 1995. The PHP reference implementation is now produced by the PHP Group. PHP was originally an abbreviation of* ***Personal Home Page****, but it now stands for the recursive acronym* ***PHP: Hypertext Preprocessor****.*

PHP Language Manual: https://www.php.net/manual/en/langref.php

PHP Variables

```php
<?php
$var = 'Bob';
$Var = 'Joe';

// outputs "Bob, Joe"
echo "$var, $Var";

// invalid; starts with a number
$4site = 'not yet';

// valid; starts with an underscore
$_4site = 'not yet';

// valid; 'ä' is (Extended) ASCII 228.
$täyte = 'mansikka';
?>
```

```php
<?php
// Assign the value 'Bob' to $foo
$foo = 'Bob';

// Reference $foo via $bar.
$bar = &$foo;

// Alter $bar...
$bar = "My name is $bar";
echo $bar;

// $foo is altered too.
echo $foo;
?>
```

PHP Data Types

PHP supports various data types, which can be categorized as follows:

- **null**: Represents a variable with no value.
- **bool**: Represents a boolean value, either `true` or `false`.
- **int**: Represents an integer value.
- **float**: Represents a floating-point number (also known as double).
- **string**: Represents a sequence of characters.
- **array**: Represents a collection of values, which can be indexed by integers or strings.
- **object**: Represents an instance of a class.
- **callable**: Represents a function that can be called.
- **resource**: Represents a reference to an external resource, such as a file or a database connection.

PHP Variables Cont.

```php
<?php
// Unset AND unreferenced (no use context) variable;
// outputs NULL
var_dump($unset_var);

// Boolean usage; outputs 'false'
// (See ternary operators for more on this syntax)
echo $unset_bool ? "true\n" : "false\n";

// String usage; outputs 'string(3) "abc"'
$unset_str .= 'abc';
var_dump($unset_str);
```

PHP Variables Cont.

```php
// Integer usage; outputs 'int(25)'
$unset_int += 25; // 0 + 25 => 25
var_dump($unset_int);
// Float usage; outputs 'float(1.25)'
$unset_float += 1.25;
var_dump($unset_float);
// Array usage; outputs array(1) {  [3]=>  string(3) "def" }
// array() + array(3 => "def") => array(3 => "def")
$unset_arr[3] = "def";
var_dump($unset_arr);
// Object usage; creates new stdClass object
// (see http://www.php.net/manual/en/reserved.classes.php)
// Outputs: object(stdClass)#1 (1) {  ["foo"]=>  string(3) "bar" }
$unset_obj->foo = 'bar';
var_dump($unset_obj);
?>
```

Predefined PHP Variables

PHP provides a range of predefined variables that are accessible in all scopes. These variables are known as superglobals and include:

- **$GLOBALS**: References all variables available in the global scope.
- **$_SERVER**: Contains information about the server and execution environment.
- **$_GET**: Contains HTTP GET variables.
- **$_POST**: Contains HTTP POST variables.
- **$_FILES**: Contains HTTP File Upload variables.
- **$_REQUEST**: Contains HTTP Request variables.
- **$_SESSION**: Contains session variables.
- **$_ENV**: Contains environment variables.
- **$_COOKIE**: Contains HTTP Cookies.
- **$php_errormsg**: Contains the previous error message.
- **$http_response_header**: Contains HTTP response headers.
- **$argc**: Contains the number of arguments passed to the script.
- **$argv**: Contains an array of arguments passed to the script.

if else

```php
<?php
if ($a > $b) {
    echo "a is bigger than b";
} elseif ($a == $b) {
    echo "a is equal to b";
} else {
    echo "a is smaller than b";
}
?>
```

for loops

```
<?php
/*
 * This is an array with some data we want to modify
 * when running through the for loop.
 */
$people = array(
    array('name' => 'Kalle', 'salt' => 856412),
    array('name' => 'Pierre', 'salt' => 215863)
);

for($i = 0; $i < count($people); ++$i) {
    $people[$i]['salt'] = mt_rand(000000, 999999);
}
```

```
$arr = array(1, 2, 3, 4);
foreach ($arr as &$value) {
    $value = $value * 2;
}
// $arr is now array(2, 4, 6, 8)
unset($value); // break the reference with the last e
?>
```