A strong foundation in PHP fundamentals is crucial for any aspiring PHP developer. Interviewers will likely start by assessing your understanding of these core concepts.

PHP Basics

PHP, short for Hypertext Preprocessor, is a server-side scripting language primarily used for web development to create dynamic web pages. This means that PHP code is executed on the web server, and the result is typically sent to the user's web browser in the form of HTML. PHP code is often embedded directly within HTML files using special tags, <?php to start a PHP code block and ?> to end it. This allows for the seamless integration of dynamic content within static web pages. Understanding PHP's role in the server-side execution of web applications is a fundamental concept for junior developers.

- **How is PHP commonly used?**
    - PHP is primarily used for web development to create dynamic web pages.
- **How does PHP differ from languages like JavaScript?**
    - PHP is a server-side scripting language, meaning the code is executed on the web server, and the result is sent to the user's browser as HTML. JavaScript, on the other hand, is primarily a client-side scripting language, executed in the user's web browser.

PHP exhibits partial case sensitivity. Variable names in PHP are case-sensitive, meaning that $myVariable and $MyVariable would be treated as two distinct variables. However, function names, whether they are built-in PHP functions or functions defined by the user, are not case-sensitive. For example, a function named myFunction() can be called as myfunction(), MyFunction(), or even MYFUNCTION(). It is important to remember this distinction when writing and debugging PHP code.

- **Is PHP a case-sensitive language?**
    - PHP exhibits partial case sensitivity. Variable names are case-sensitive, while function names are not.

To enable the PHP parsing engine to differentiate PHP code from other elements on a web page, such as HTML, a mechanism known as "escaping to PHP" is used. This involves using the opening tag <?php to signal the beginning of PHP code and the closing tag ?> to indicate its end. Everything between these tags is interpreted as PHP code by the server.

- **What does "escaping to PHP" mean?**
    - "Escaping to PHP" refers to the mechanism of using the opening tag <?php to signal the beginning of PHP code and the closing tag ?> to indicate its end, allowing the PHP parsing engine to differentiate PHP code from other elements like HTML.

Variables and Constants

Variables in PHP are used to store data that can be manipulated during the execution of a script. When naming variables in PHP, certain rules must be followed. A variable name must always begin with a dollar sign ($), followed by a letter or an underscore (_). Subsequent characters in the variable name can be letters, numbers, or underscores. Variable names cannot contain spaces or special characters such as +, -, or %. It is also crucial to remember that PHP variable names are case-sensitive.

- **What are the characteristics of PHP variables?**

    - PHP variables are used to store data that can be manipulated during script execution. They are dynamically typed, and their values can change.
- **What are the rules for naming a PHP variable?**
    - A PHP variable name must begin with a dollar sign ($), followed by a letter or an underscore (_). Subsequent characters can be letters, numbers, or underscores. Variable names cannot contain spaces or special characters. They are also case-sensitive.

PHP is a dynamically typed language, which means that you do not need to explicitly declare the data type of a variable before assigning a value to it. The data type of a variable is determined by the value that is currently stored in it, and this type can change during the script's execution if a new value of a different type is assigned. The value of a variable at any given time is the value of its most recent assignment. If a variable is used before it has been assigned a value, it will have a default value depending on the context.
The scope of a variable defines the region within a script where the variable can be accessed. PHP has several different scope levels, including local scope (within a function), global scope (outside functions), and static scope (for static variables within functions). Understanding variable scope is essential for managing data and avoiding naming conflicts in your code.
Constants in PHP are similar to variables in that they store values, but once a constant is defined, its value cannot be changed during the script's execution. Constants are defined using the define() function, which takes the constant's name (without a dollar sign prefix) and its value as arguments. Unlike variables, constants have global scope and can be accessed from anywhere in the script. Constants are often used to store values that should remain the same throughout the application, such as configuration settings or mathematical constants.

- **Differentiate between variables and constants in PHP.**
    - Variables store data that can be changed during script execution, while constants store data that cannot be changed after definition.
- **How do variables and constants differ in PHP?**
    - Variables are declared with a '' and their values can be changed. Constants are declared with define(), do not use a '', and their values cannot be changed after they are defined. Constants have global scope.
- **How do you define a constant in PHP?**
    - Constants are defined using the define() function, which takes the constant's name (without a dollar sign) and its value as arguments.
- **What is the purpose of the constant() function?**
    - The constant() function is not mentioned in the text. The define() function is used to define a constant. Constants are used to store values that should remain the same throughout the application.

Data Types
PHP supports eight primary data types that are used to classify the kind of data that a variable can hold. These include scalar types, compound types, and special types.
Scalar Types: These represent single values.

- Integers: These are whole numbers without any decimal point, such as 4195 or -123.
- Doubles (Floats): These represent numbers with a decimal point or in exponential

notation, like 3.14159 or 49.1.
- Booleans: These have only two possible values: true or false, representing logical states.
- Strings: These are sequences of characters, for example, "Hello world!" or 'PHP supports string operations'.

Compound Types: These can hold multiple values.

- Arrays: These are named and indexed collections of other values. Arrays can hold a mix of different data types and can be indexed numerically or using strings as keys (associative arrays). PHP supports indexed arrays, associative arrays, and multidimensional arrays.
- Objects: These are instances of programmer-defined classes that can encapsulate both data (properties) and functions (methods).

Special Types:

- NULL: This is a special type that has only one possible value: NULL. It indicates that a variable has no value assigned to it.
- Resources: These are special variables that hold references to external resources, such as database connections, file handles, or network streams.

Understanding the different data types in PHP is essential for working with data effectively. For instance, knowing the difference between single-quoted and double-quoted strings is important because double-quoted strings allow for variable interpolation and the use of more escape sequences than single-quoted strings.

- **What are the types of PHP variables?**
  - PHP variables can hold data of eight primary types: scalar types (integers, doubles/floats, booleans, strings), compound types (arrays, objects), and special types (NULL, resources).
- **Name the different types of variables in PHP.**
  - The different types of variables in PHP are: integers, doubles (floats), booleans, strings, arrays, objects, NULL, and resources.
- **How many types of an array are there in PHP?**
  - PHP supports indexed arrays, associative arrays, and multidimensional arrays.
- **What are PHP arrays and how are they different from other data types?**
  - PHP arrays are named and indexed collections of other values. They can hold a mix of different data types and can be indexed numerically or using strings (associative arrays). They differ from scalar types, which hold single values.
- **Explain the difference between single-quoted string and double-quoted string?**
  - Double-quoted strings allow for variable interpolation and the use of more escape sequences than single-quoted strings.
- **What is NULL?**
  - NULL is a special data type that has only one possible value: NULL. It indicates that a variable has no value assigned to it.

Operators

PHP provides a wide range of operators that allow you to perform operations on variables and values. These include:

- Arithmetic Operators: Used for performing mathematical calculations (e.g., +, -, *, /, %).

- Assignment Operators: Used for assigning values to variables (e.g., =, +=, -=).
- Comparison Operators: Used for comparing two values (e.g., ==, ===, !=, >, <). The difference between == (equality) and === (identity) is important; === checks if both the value and the data type are the same.
- Logical Operators: Used for combining conditional statements (e.g., && or and, || or or, ! or not). The difference between && and and lies in their precedence.
- Increment/Decrement Operators: Used for increasing or decreasing the value of a variable (e.g., ++, --).
- String Operators: Used for concatenating strings (e.g., .).
- Array Operators: Used for performing operations on arrays (e.g., +, ==, ===).
- Conditional (Ternary) Operator: A shorthand for if-else statements (condition? value_if_true : value_if_false).

While you don't need to have an exhaustive knowledge of every operator for a junior role, understanding the basic arithmetic, comparison, logical, and assignment operators, as well as the ternary operator, is crucial.

- **What is the "ternary operator" in PHP?**
    - The ternary operator is a shorthand for if-else statements, written as (condition ? value_if_true : value_if_false).
- **What is the difference between === and == in PHP?**
    - == (equality) checks if two values are the same. === (identity) checks if both the value and the data type are the same.
- **What is the difference between AND and &&?**
    - The difference between && and and lies in their precedence.

Control Structures

Control structures in PHP allow you to control the flow of execution of your scripts based on certain conditions or to repeat blocks of code.

Conditional Statements:

- if statement: Executes a block of code if a specified condition is true.
- if...else statement: Executes one block of code if a condition is true, and another block of code if the condition is false.
- if...elseif...else statement: Executes different blocks of code for more than two possible conditions.
- switch statement: Selects one block of code to execute from multiple possible cases based on the value of an expression.

Loop Statements:

- while loop: Executes a block of code repeatedly as long as a specified condition is true.
- do...while loop: Similar to the while loop, but it executes the code block at least once before checking the condition.
- for loop: Executes a block of code a specific number of times, often used when you know in advance how many iterations are needed.
- foreach loop: Provides an easy way to iterate over arrays and objects.

Jump Statements:

- break statement: Immediately terminates the execution of a loop (for, while, do-while,

foreach) or a switch statement.
- continue statement: Skips the rest of the current iteration of a loop and proceeds to the next iteration.

Understanding how to use these control structures is fundamental to writing PHP code that can make decisions and perform repetitive tasks.

- **What is the purpose of the break and continue statement?**
    - The break statement immediately terminates the execution of a loop or a switch statement. The continue statement skips the rest of the current iteration of a loop and proceeds to the next iteration.
- **Explain the syntax for the foreach loop with an example.**
    - The foreach loop provides an easy way to iterate over arrays and objects.
        - Syntax: foreach ($array as $key => $value) { // code to be executed; }

Functions

Functions in PHP are reusable blocks of code that perform specific tasks. They help in organizing code, making it more readable, and allowing for code reuse. PHP has a vast number of built-in functions that perform various operations. Additionally, you can define your own functions, known as user-defined functions, to encapsulate specific logic.

To define a user-defined function, you use the function keyword, followed by the function name, an optional list of parameters enclosed in parentheses, and the code to be executed within curly braces. Functions can accept arguments (parameters) that provide input to the function. These arguments can be passed by value (the default) or by reference (allowing the function to modify the original variable). You can also define default values for function parameters. Functions can return a value back to the part of the script that called them using the return statement.

Variables defined inside a function have local scope, meaning they are only accessible within that function. To use global variables within a function, you need to explicitly declare them using the global keyword.

Autoloading is a feature in PHP that allows for the automatic loading of class files when a class is first used, without the need to manually include or require the file. This can be implemented using the spl_autoload_register() function or by following autoloading standards like PSR-4.

The header() function in PHP is used to send raw HTTP headers to the client's browser. This can be used for various purposes, such as setting cookies, redirecting the user to a different page, or specifying the content type of the response.

- **What is PEAR in PHP?**
    - PEAR is not defined in the provided text.
- **What are session variables, and why are they useful?**
    - Session variables are not explicitly defined in the provided text.
- **Describe the difference between "echo" and "print" in PHP.**
    - The difference between echo and print is not explicitly described in the provided text.
- **What does include and require do?**
    - The provided text mentions autoloading with spl_autoload_register(), but doesn't

detail include and require.

- **How is include() different from require()?**
  - The difference between include() and require() is not in the provided text.
- **What is the purpose of autoloading in PHP, and how can it be implemented?**
  - Autoloading allows for the automatic loading of class files when a class is first used. It can be implemented using the spl_autoload_register() function or by following autoloading standards like PSR-4.
- **Explain the significance of the header() function in PHP.**
  - The header() function is used to send raw HTTP headers to the client's browser, for purposes like setting cookies, redirecting users, or specifying content types.