PHP Cheat Sheet

1. Basic Syntax

PHP Tags

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

Comments

```
// Single-line comment

# Single-line comment (alternative)

/*
Multi-line comment
*/
```

Echo and Print

```
echo "Hello, World!";
print "Hello, World!";
```

____

2. Variables and Data Types

Variables

```
$variableName = value;
```

Data Types

- **String**: `"Hello"`
- **Integer**: `42`
- **Float**: `3.14`
- **Boolean**: `true` or `false`
- **Array**: `array("apple", "banana")`

- **Null**: `null`

- **Object**: `new ClassName()`

Type Casting

```
$intVar = (int)"42";
$floatVar = (float)"3.14";
$stringVar = (string)42;
```

---

3. Operators

Arithmetic Operators

- `+` (Addition)
- `-` (Subtraction)
- `*` (Multiplication)
- `/` (Division)
- `%` (Modulus)

- `**` (Exponentiation)

Assignment Operators

- `=` (Assignment)
- `+=` (Add and assign)
- `-=` (Subtract and assign)
- `*=` (Multiply and assign)
- `/=` (Divide and assign)
- `%=` (Modulus and assign)

Comparison Operators

- `==` (Equal)
- `===` (Identical)
- `!=` (Not equal)
- `!==` (Not identical)
- `>` (Greater than)
- `<` (Less than)
- `>=` (Greater than or equal to)
- `<=` (Less than or equal to)

Logical Operators

- `&&` (And)
- `||` (Or)

- `!` (Not)

Increment/Decrement Operators

- `++$var` (Pre-increment)
- `$var++` (Post-increment)
- `--$var` (Pre-decrement)
- `$var--` (Post-decrement)

---

4. Control Structures

If Statement

```
if (condition) {
    // code to execute if condition is true
} elseif (anotherCondition) {
    // code to execute if anotherCondition is true
} else {
    // code to execute if all conditions are false
}
```

Switch Statement

```
switch ($variable) {
    case value1:
        // code to execute if $variable == value1
        break;
    case value2:
        // code to execute if $variable == value2
        break;
    default:
        // code to execute if $variable does not match any case
}
```

Loops

- **For Loop**:

  ```
  for ($i = 0; $i < 10; $i++) {
      // code to execute
  }
  ```

- **While Loop**:

```
while (condition) {
    // code to execute
}
```

- **Do-While Loop**:

```
do {
    // code to execute
} while (condition);
```

- **Foreach Loop**:

```
foreach ($array as $value) {
    // code to execute
}
```

---

5. Functions

Defining a Function

```
function functionName($parameter1, $parameter2) {
    // code to execute
    return $result;
}
```

Calling a Function

```
$result = functionName($arg1, $arg2);
```

Default Parameter Values

```
function greet($name = "Guest") {
    echo "Hello, $name!";
}
```

Variable Functions

```
$functionName = "greet";
$functionName("John");
```

---

6. Arrays

Creating Arrays

```
$array = array("apple", "banana", "cherry");
$array = ["apple", "banana", "cherry"]; // Shorthand
```

Accessing Array Elements

```
echo $array[0]; // Outputs: apple
```

Associative Arrays

```
$assocArray = ["name" => "John", "age" => 30];
echo $assocArray["name"]; // Outputs: John
```

Multidimensional Arrays

```
$multiArray = [
    ["apple", "banana"],
    ["cherry", "date"]
];
echo $multiArray[1][0]; // Outputs: cherry
```

Array Functions

- **`count($array)`**: Returns the number of elements in an array.

- **`array_push($array, $value)`**: Adds one or more elements to the end of an array.

- **`array_pop($array)`**: Removes the last element from an array.

- **`array_merge($array1, $array2)`**: Merges two or more arrays.

- **`sort($array)`**: Sorts an array in ascending order.

- **`rsort($array)`**: Sorts an array in descending order.

- **`array_map($callback, $array)`**: Applies a callback function to each element of an array.

---

7. Strings

String Concatenation

```
$firstName = "John";
$lastName = "Doe";
$fullName = $firstName . " " . $lastName;
```

String Functions

- **`strlen($string)`**: Returns the length of a string.

- **`str_replace($search, $replace, $string)`**: Replaces all occurrences of a search string with a replacement.

- **`strpos($string, $search)`**: Finds the position of the first occurrence of a substring in a string.

- **`substr($string, $start, $length)`**: Returns a part of a string.

- **`trim($string)`**: Removes whitespace from the beginning and end of a string.

- **`explode($delimiter, $string)`**: Splits a string by a delimiter into an array.

- **`implode($glue, $array)`**: Joins array elements into a string.

---

8. File Handling

Reading a File

```
$file = fopen("file.txt", "r");
while (!feof($file)) {
    echo fgets($file);
}
fclose($file);
```

Writing to a File

```
$file = fopen("file.txt", "w");
fwrite($file, "Hello, World!");
fclose($file);
```

File Functions

- **`file_get_contents($filename)`**: Reads a file into a string.

- **`file_put_contents($filename, $data)`**: Writes a string to a file.

- **`unlink($filename)`**: Deletes a file.

- **`is_file($filename)`**: Checks if a file exists.

---

9. Object-Oriented Programming (OOP)

Defining a Class

```
class MyClass {
    public $property;

    public function __construct($value) {
        $this->property = $value;
    }

    public function myMethod() {
        echo $this->property;
    }
}
```

Creating an Object

```
$obj = new MyClass("Hello");
$obj->myMethod(); // Outputs: Hello
```

Inheritance

```
class ChildClass extends MyClass {
    public function newMethod() {
        echo "New Method";
    }
}
```

Access Modifiers

- **`public`**: Accessible from anywhere.

- **`private`**: Accessible only within the class.

- **`protected`**: Accessible within the class and its subclasses.

Static Methods and Properties

```
class MyClass {
    public static $staticProperty = "Static Property";

    public static function staticMethod() {
        echo self::$staticProperty;
    }
}

MyClass::staticMethod(); // Outputs: Static Property
```

____

10. Error Handling

Basic Error Handling

```
try {
    // code that may throw an exception
} catch (Exception $e) {
    echo "Error: " . $e->getMessage();
}
```

Custom Exceptions

```
class MyException extends Exception {
    public function errorMessage() {
        return "Custom Error: " . $this->getMessage();
    }
}

try {
    throw new MyException("Something went wrong");
} catch (MyException $e) {
    echo $e->errorMessage();
}
```

Error Reporting

```
error_reporting(E_ALL);
ini_set('display_errors', 1);
```

____

11. Superglobals

$_GET

```
echo $_GET['param']; // Access query string parameters
```

$_POST

```
echo $_POST['field']; // Access form data submitted via POST
```

$_SESSION

```
session_start();
$_SESSION['username'] = "John";
echo $_SESSION['username'];
```

$_COOKIE

```
setcookie("name", "value", time() + 3600);
echo $_COOKIE['name'];
```

$_SERVER

```
echo $_SERVER['HTTP_USER_AGENT']; // User's browser information
```

---

12. Cookies and Sessions

Setting a Cookie

```
setcookie("username", "John", time() + 3600, "/");
```

Accessing a Cookie

```
echo $_COOKIE['username'];
```

Starting a Session

```
session_start();
$_SESSION['username'] = "John";
```

Accessing Session Data

```
echo $_SESSION['username'];
```

Destroying a Session

```
session_unset();
session_destroy();
```

---

13. Database Interaction

Connecting to a Database

```
$conn = new mysqli("localhost", "username", "password", "database");
```

Executing a Query

```
$result = $conn->query("SELECT * FROM users");
while ($row = $result->fetch_assoc()) {
    echo $row['username'];
}
```

Prepared Statements

```
$stmt = $conn->prepare("INSERT INTO users (username, email) VALUES (?,
?)");
$stmt->bind_param("ss", $username, $email);
$username = "John";
$email = "john@example.com";
$stmt->execute();
```

Closing the Connection

```
$conn->close();
```

---

14. Advanced Topics

Namespaces

```
namespace MyNamespace;
```

```
class MyClass {
    public function myMethod() {
        echo "Hello from MyNamespace";
    }
}
```

Autoloading

```
spl_autoload_register(function ($class) {
    include $class . '.php';
});
```

Traits

```
trait MyTrait {
    public function myMethod() {
        echo "Trait Method";
    }
}

class MyClass {
    use MyTrait;
}

$obj = new MyClass();
$obj->myMethod(); // Outputs: Trait Method
```

Closures

```
$greet = function($name) {
    return "Hello, $name";
};

echo $greet("John"); // Outputs: Hello, John
```

Generators

```
function myGenerator() {
    yield "Apple";
    yield "Banana";
    yield "Cherry";
}

foreach (myGenerator() as $fruit) {
    echo $fruit;
}
```

This cheat sheet provides a comprehensive overview of PHP, covering essential syntax, data types, control structures, functions, arrays, strings, file handling, OOP, error handling, superglobals, cookies and sessions, database interaction, and advanced topics. Use this as a quick reference guide for your PHP development needs.

By Ahmed Baheeg Khorshid

ver 1.0