Server-side Web Development 1: PHP contd.

Backend Web Development

by: Salahuddin ElKazak

Program Controls contd.

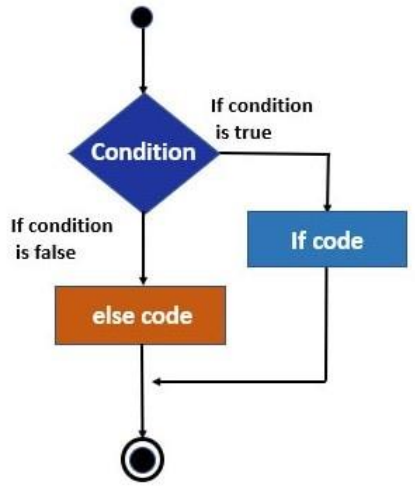
Selections: if...else vs. switch...case

Lecture 2

Selections Comparison

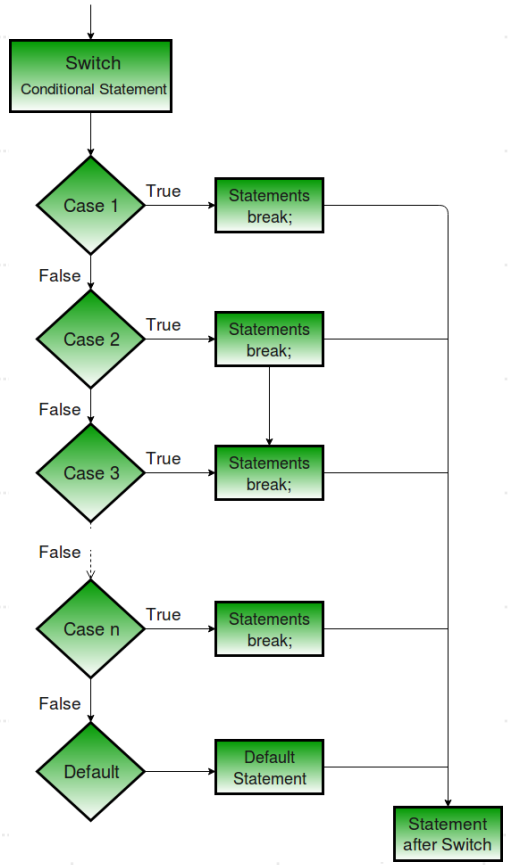
if…else

```
if ($artType == "PT")
  $output = "Painting";
else if ($artType == "SC")
  $output = "Sculpture";
else
  $output = "Other";
```



switch…case

```
switch ($artType) {
  case "PT":
    $output = "Painting";
    break;
  case "SC":
    $output = "Sculpture";
    break;
  default:
    $output = "Other";
}
```



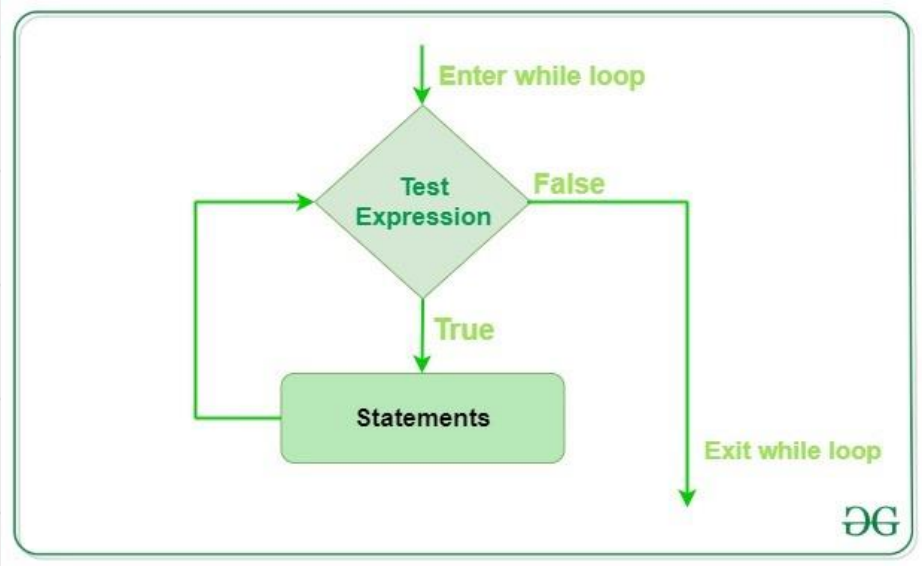
Program Controls contd.

Loops and Iterations

Lecture 2

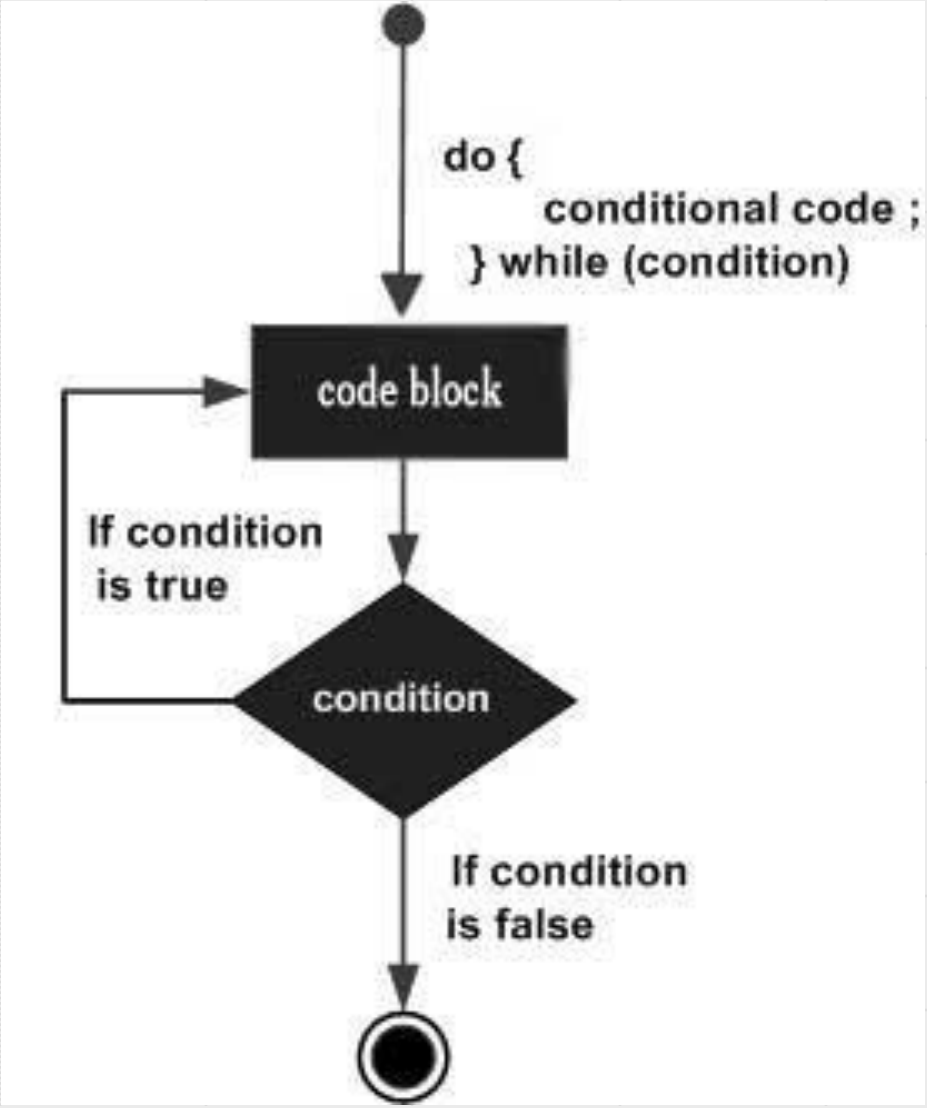
while-loop

```php
// Example: Reading a file line by line
$file = fopen("example.txt", "r");
while ($line = fgets($file)) {
  echo $line; // Process the line
}
fclose($file);
```



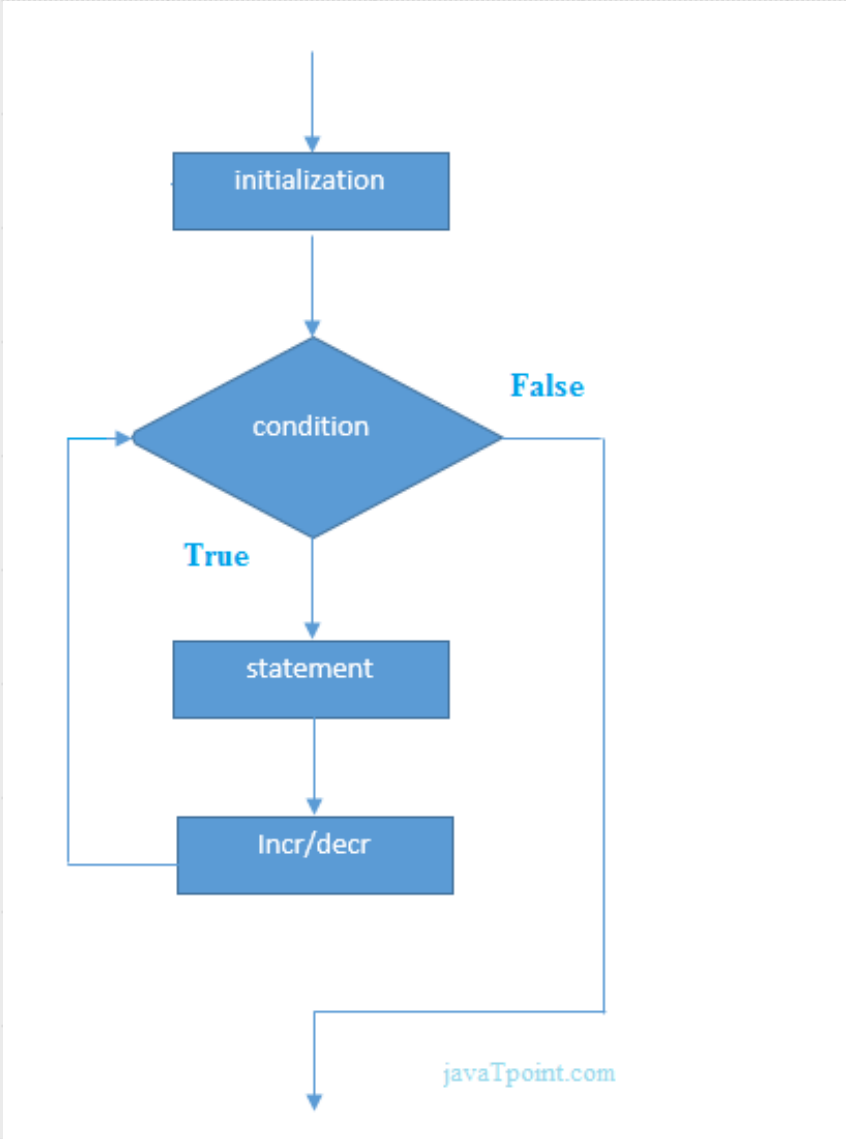
do-while-loop

```
// Example: Asking for input
  at least once
do {
    $input = readline("Enter a
  number greater than 0: ");
} while ($input <= 0);

echo "You entered: $input";
```



for-loop

```
// Example: Looping through an array of
  numbers
$numbers = [1, 2, 3, 4, 5];


for ($i = 0; $i < count($numbers); $i++)
  {
     echo $numbers[$i] . "\n";
}
```

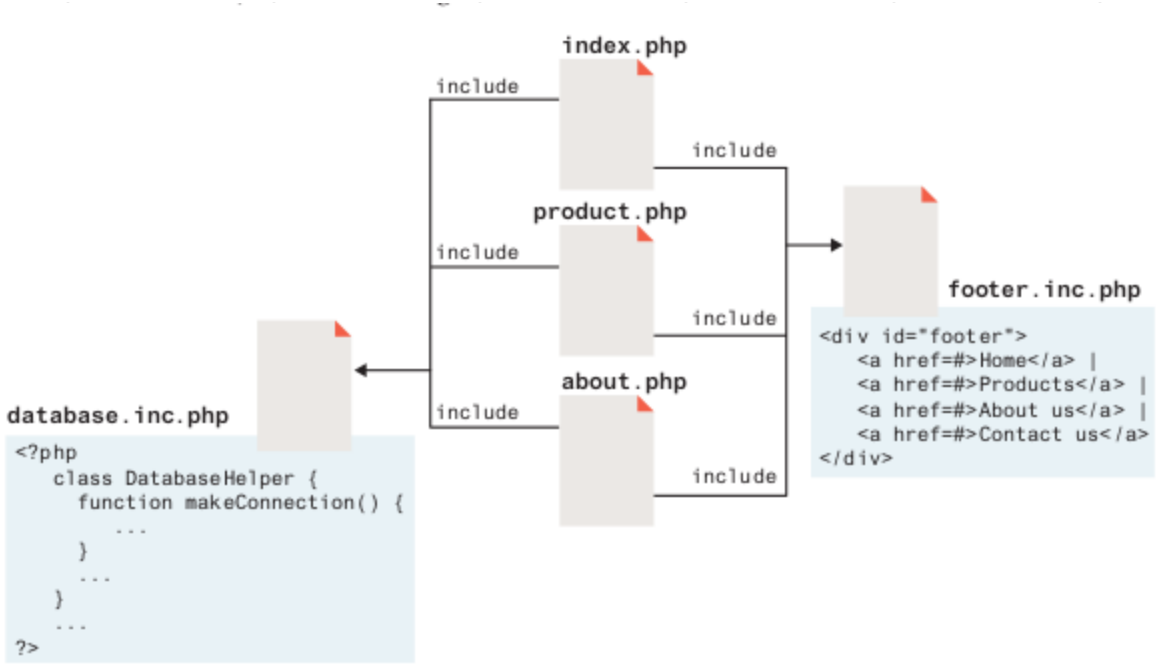


Alternate Syntax

```
<?php if ($userStatus == "loggedin") : ?>
  <a href="account.php">Account</a>
  <a href="logout.php">Logout</a>
<?php else : ?>
  <a href="login.php">Login</a>
  <a href="register.php">Register</a>
<?php endif; ?>
```

PHP's Include Feature for Code Reuse

- PHP allows you to **reuse code** by including content from one file into another.
- Commonly used in **almost every PHP page** to manage both **HTML markup** and **PHP logic**.
- Four key statements for including files:
  - `include`: Adds a file, continues if not found (shows a warning).
  - `require`: Adds a file, but **stops execution** if not found (shows an error).
  - `include_once` & `require_once`: Ensures a file is only included **once** to prevent duplicate code and save memory.
- **Essential for larger projects**, especially when multiple developers work together and files are nested.



Include cntd.: Example

By convention, PHP files have the .php extension.

example.php

```php
<?php
include('exampleData.inc.php');
?>
<!DOCTYPE html>
<html lang="en">
<head>
   ...
</head>
<body>
<form>
  <fieldset>
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" value="<?= $name ?>" >

    <label for="mail">Email:</label>
    <input type="email" id="mail" name="email" value="<?= $email ?>" >

    <label for="interests">Interests:</label>
    <select id="interests" name="interests">
      <?php
      for ($i=0; $i<5; $i++) {
          $count = $i + 1;
          echo "<option>Interest " . $count . "</option>";
      }
      ?>
    </select>
    <button type="submit">
    Contact us
    </button>
  </fieldset>
</form>
</body>
</html>
```

Files that are included can have any extension, though in this example we are using the extension .inc.php to make it clearer later that this is an include file.

exampleData.inc.php

```php
<?php
$name = 'Randy Connolly';
$email = 'someone@example.com';
?>
```

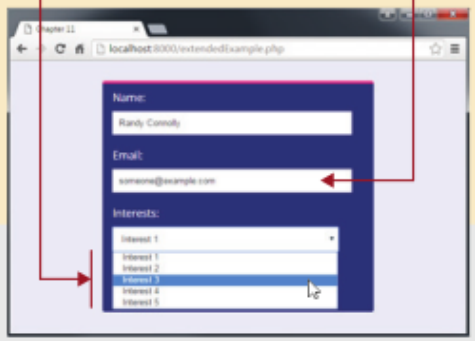
The include function inserts the contents of the specified file.

Common practice is to place include statements (and variables used throughout the page) at the top of the page.

Here we are outputing the contents of the $name variable into the value attribute.

The contents of the $email variable is "injected" into the value attribute.

Use a loop to output five <option> elements.



Result in browser.

Functions

Lecture 2

About Functions

- PHP functions let you organize **small, reusable pieces of code** that accomplish specific tasks.
- Just like in JavaScript, functions can:
  - Be called multiple times from anywhere in your code.
  - Accept **parameters** to work with different data.
  - Optionally **return a value** to the caller.
- PHP offers two types of functions:
  - **User-defined functions**: Created by you for custom tasks.
  - **Built-in functions**: Predefined by PHP, offering a rich library to simplify tasks.
- Using functions makes code **easier to maintain, reuse, and understand.**

Function Syntax

Here are some user-defined functions cover a range of common needs in web development, such as input sanitization, redirection, and file handling.

1. Sanitize Input

This function helps sanitize user inputs to prevent SQL injection and XSS attacks.

```php
function sanitize_input($data) {
    return htmlspecialchars(stripslashes(trim($data)));
}
```

2. Redirect

A simple function to redirect users to a new page.

```php
function redirect($url) {
    header("Location: $url");
    exit();
}
```

3. Generate Random String

This function generates a random string of a specified length, useful for passwords or tokens.

```php
function generate_random_string($length = 10) {
    return
    substr(str_shuffle("0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVW
    XYZ"), 0, $length);
}
```

4. Validate Email Address

This function validates an email address format.

```php
function validate_email($email) {
    return filter_var($email, FILTER_VALIDATE_EMAIL);
}
```

5. Get Client IP Address

A function to retrieve the user's real IP address.

```php
function get_client_ip() {
    $ip = '';
    if (!empty($_SERVER['HTTP_CLIENT_IP'])) {
        $ip = $_SERVER['HTTP_CLIENT_IP'];
    } elseif (!empty($_SERVER['HTTP_X_FORWARDED_FOR'])) {
        $ip = $_SERVER['HTTP_X_FORWARDED_FOR'];
    } else {
        $ip = $_SERVER['REMOTE_ADDR'];
    }
    return $ip;
}
```

6. Convert Array to CSV

Converts a PHP array into CSV format and saves it to a file.

```php
function array_to_csv($array, $filename = "output.csv", $delimiter = ",") {
    $f = fopen($filename, 'w');
    foreach ($array as $line) {
        fputcsv($f, $line, $delimiter);
    }
    fclose($f);
}
```

Function Syntax cntd.

7. Calculate Age from Date

This function calculates a person's age based on their birthdate.

```
function calculate_age($birthdate) {
    $dob = new DateTime($birthdate);
    $now = new DateTime();
    return $now->diff($dob)->y;
}
```

8. Check if String Contains Substring

This function checks if a substring exists within a string.

```
function contains($haystack, $needle) {
    return strpos($haystack, $needle) !== false;
}
```

9. Limit Text Length

This function truncates a string and adds an ellipsis if it exceeds a given length.

```
function limit_text($text, $limit) {
    return (strlen($text) > $limit) ? substr($text, 0, $limit) . "..." : $text;
}
```

10. Format File Size

This function converts bytes into human-readable file sizes (KB, MB, GB, etc.).

```
function format_size($size) {
    $units = ['B', 'KB', 'MB', 'GB', 'TB'];
    $power = $size > 0 ? floor(log($size, 1024)) : 0;
    return number_format($size / pow(1024, $power), 2) . ' ' . $units[$power];
}
```

- How to invoke? ***function_name()***
- *PHP 7 introduced optional return type and parameter type declarations, ensuring functions return or accept specific types.*
- *If the types don't match, a TypeError is thrown when strict typing is enabled.*
- *Syntax?*

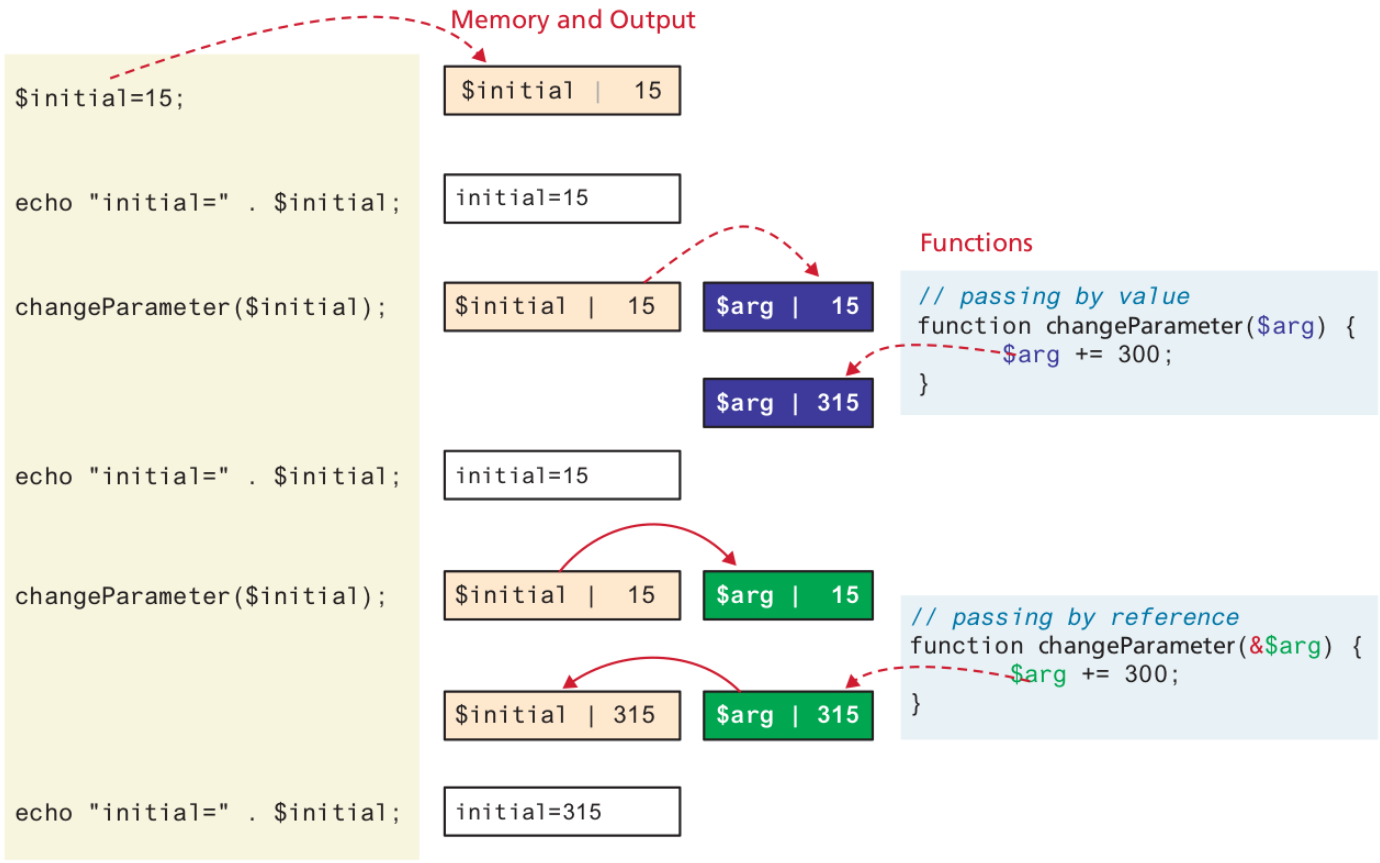
```
function mustReturnString(string $name) : string {
    return "hello ". $name;
}
```

Parameter Default Value

```
/**
* This function returns a nicely formatted string using the current
* system time. The showSeconds parameter controls whether or not
* to show the seconds.
*/
function getNiceTime($showSeconds=true) {
   if ($showSeconds==true)
       return date("H:i:s");
   else
       return date("H:i");
}
```

Pass by Value vs Pass by Reference

- Do you just want to pass in a value?
- Or do you want the function to alter the variable?



Deeper Dive: Anonymous functions & Arrow functions

- Anonymous function:

```
$sum = function($a,$b) {
   echo "here";
   return $a + $b;
};
$foo = $sum(3,4);
```

- Arrow function:

```
$sum = fn($a,$b) => $a + $b;
```

- *More Advanced Arrow function e.g.:*
-

```
<?php

fn(array $x) => $x;
static fn(): int => $x;
fn($x = 42) => $x;
fn(&$x) => $x;
fn&($x) => $x;
fn($x, ...$rest) => $rest;l
```

Arrays

Lecture 2

What are arrays?

- PHP arrays are flexible data structures that function as ordered maps, associating values with keys.
- Unlike other languages where arrays are sequentially indexed by integers, PHP arrays can use both **integers** and **strings** as keys, and *values can be of any type*.
- This flexibility allows PHP arrays to behave like
  - arrays,
  - hash tables,
  - dictionaries, and
  - lists in other languages.

How to define them?

```
// empty array named days
$days = array();

// list of values (default keys are 0, 1, 2, … , n-1)
$days = array("Mon", "Tue", "Wed", "Thu", "Fri");
$days = ["Mon", "Tue", "Wed", "Thu", "Fri"];     // alternate syntax

// can also define them individually
$days = array();
$days[0] = "Mon";
$days[1] = "Tue";
…
// or even
$days = [];
$days[] = "Mon";
$days[] = "Tue";
…
```

0	Mon
1	Tue
2	Wed
3	Thu
4	Fri

Associative & Multidimensional Arrays

```php
$month = array(
  array("Mon","Tue","Wed","Thu","Fri"),
  array("Mon","Tue","Wed","Thu","Fri"),
  array("Mon","Tue","Wed","Thu","Fri"),
  array("Mon","Tue","Wed","Thu","Fri")
);
echo $month[0][3];    // outputs Thu

$cart = [];
$cart[] = array("id" => 37, "title" => "Burial at Ornans",
                "quantity" => 1);
$cart[] = array("id" => 345, "title" => "The Death of Marat",
                "quantity" => 1);
$cart[] = array("id" => 63, "title" => "Starry Night", "quantity" => 1);

echo $cart[2]["title"];   // outputs Starry Night

$stocks = [
  ["AMZN", "Amazon"],
  ["APPL", "Apple"],
  ["MSFT", "Microsoft"]
];
echo $stocks[2][1];    // outputs Microsoft

$aa = [
  "AMZN" => ["Amazon", 234],
  "APPL" => ["Apple", 342],
  "MSFT" => ["Microsoft", 165]
];
```

```php
echo $aa["APPL"][0];   // outputs Apple

$bb = [
  "AMZN" => ["name" =>"Amazon", "price" => 234],
  "APPL" => ["name" => "Apple", "price" => 342],
  "MSFT" => ["name" => "Microsoft", "price" => 165]
];
echo $bb["MSFT"]["price"];  // outputs 165
```

Lab Exercise

Create a PHP script that performs the following tasks:

- Define an array of at least 5 students with their corresponding grades (use student names as keys and grades as values).
- Write a function `calculateAverage($grades)` that takes the array of grades and returns the average grade.
- Write a function `findHighestGrade($grades)` that returns the student with the highest grade.
- Write a function `displayStudents($grades)` that prints a list of all students and their grades.
- Display the following on the webpage:
  - The list of students and their grades.
  - The average grade.
  - The student with the highest grade.
  - Requirements:
- Make sure the functions are well-structured.
- Use HTML for displaying the output clearly on a webpage.
- **Bonus:**
- Add a function that lists students who scored above the average grade.

Iterating through an Array

- **Using `foreach`:**

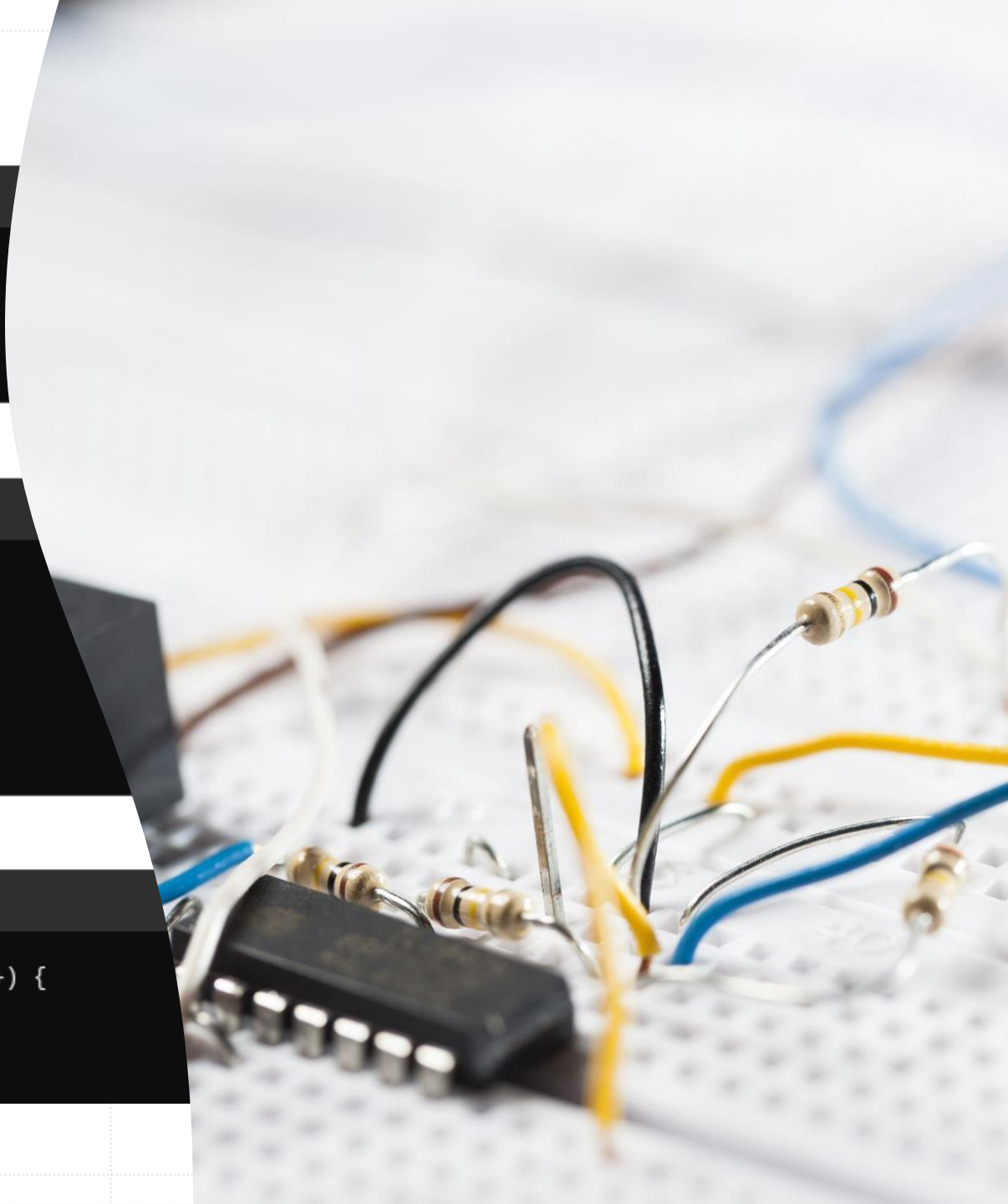
```php
foreach ($array as $key => $value) {
    echo "$key: $value";
}
```

- **Using `while`:**

```php
$i = 0;
while ($i < count($array)) {
    echo $array[$i];
    $i++;
}
```

- **Using `for`:**

```php
for ($i = 0; $i < count($array); $i++) {
    echo $array[$i];
}
```



Add/Update/Delete Elements

- **Adding/Updating:**

```php
$array['key'] = 'value';  // Adds or updates
```

- **Deleting:**

```php
unset($array['key']);  // Removes an element
```

Check if a value exists

- **Check if key exists:**

```php
array_key_exists('key', $array);  // True if key exists
```

- **Check if value exists:**

```php
in_array('value', $array);  // True if value exists
```

- **Check if a key is set and not null:**

```php
isset($array['key']);  // True if key exists and value is not null
```

PROTIP: Rarely go for switch-case

Switch…case

```
switch ($artType) {
  case "PT":
    $output = "Painting";
    break;
  case "SC":
    $output = "Sculpture";
    break;
  default:
    $output = "Other";
}
```

Better Alternative (most of the time): associative map

```
$artTypes = [
  "PT" => "Painting",
  "SC" => "Sculpture"
  ];
$output = $artTypes[$artType] ??
  "Other";
```

Classes & Objects

Lecture 3

Terminology

- **Object-Oriented Programming (OOP)** allows developers to work with items (objects) that have specific properties (attributes) and methods (functions).
- Objects are created from blueprints called **classes**, which define their properties and behavior.
- Once an object (or instance) is created from a class, it holds its own set of values for its properties and operates independently from the class.

Terminology … and why?

- Classes should be defined in their own files and can be included in scripts using **`include`**, **`require`**, etc.
- New objects are created using the ***new*** keyword, allowing as many instances as memory permits.
- ***Why OOP?***
  - Encapsulation
  - Abstraction
  - Inheritance
  - Polymorphism
  - Modularity

Defining Classes

```php
<?php
// Define the Car class
class Car {
    // Properties (Attributes)
    public $make;
    public $model;
    public $year;
    // Constructor method
    public function __construct($make, $model, $year) {
        $this->make = $make;
        $this->model = $model;
        $this->year = $year;
    }
    // Method to display car information
    public function displayInfo() {
        return "This car is a {$this->year} {$this->make} {$this->model}.";
    }
}
?>
```

Instantiating Objects

Properties

Constructor

Method

Visibility

Static

Inheritance

$_GET and $_POST Superglobal Arrays

Lecture 3

Superglobal Arrays

Determining if Any Data was Sent

Accessing Form Array Data

Using Query Strings

Sanitizing Query Strings

Working with the HTTP Header

Lecture 2

Redirect Using Location Header

Setting the Content-Type Header

Setting the Content-Type Header contd.