LECTURE03

PHP-1:STRUCTURE AND CONTROL

CS418/518: WEB PROGRAMMING

BY DR. JIAN WU

Courtesy: presentation slides from Dr. Justin Brunelle

WHAT IS PHP?

- PHP: Hypertext Processor
  - Server-side scripting
  - Free alternative to Microsoft ASP.NET
  - Can be embedded in HTML
  - Similar syntax to Perl & C
  - Frequently combined with MySQL

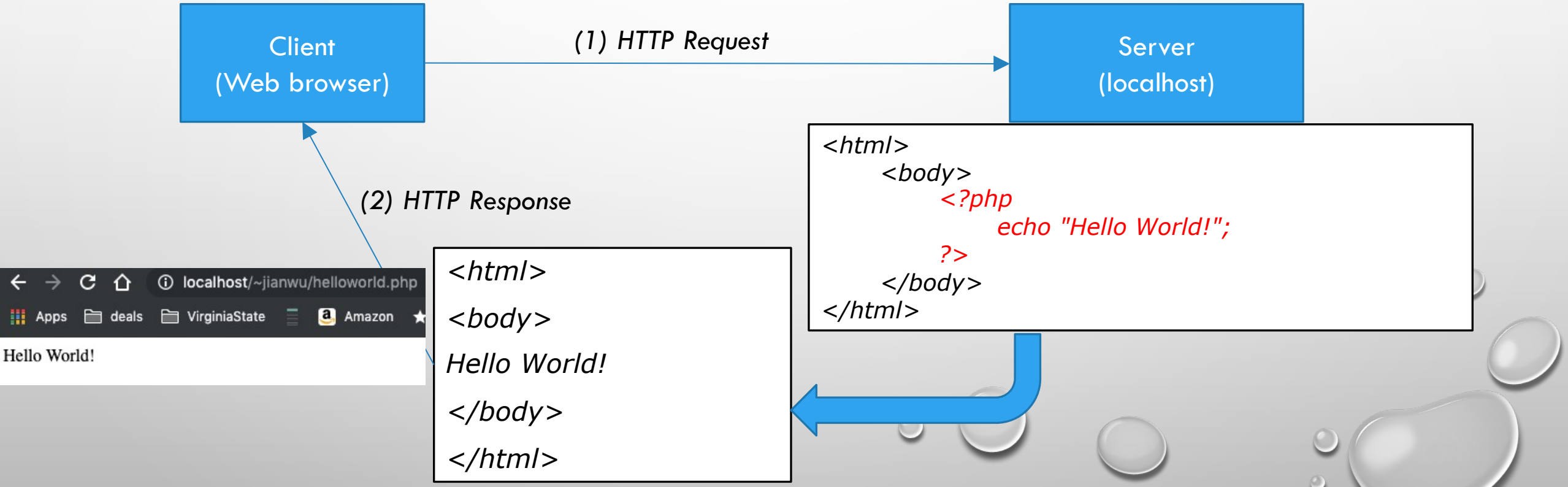
BASICS

- **PHP script is executed on the server side**
- The client has no idea of what was happening behind the webpage he/she sees
- You cannot open a PHP file by double clicking it or by a web browser!
- You MUST save PHP file on a web server and visit it through a web browser!

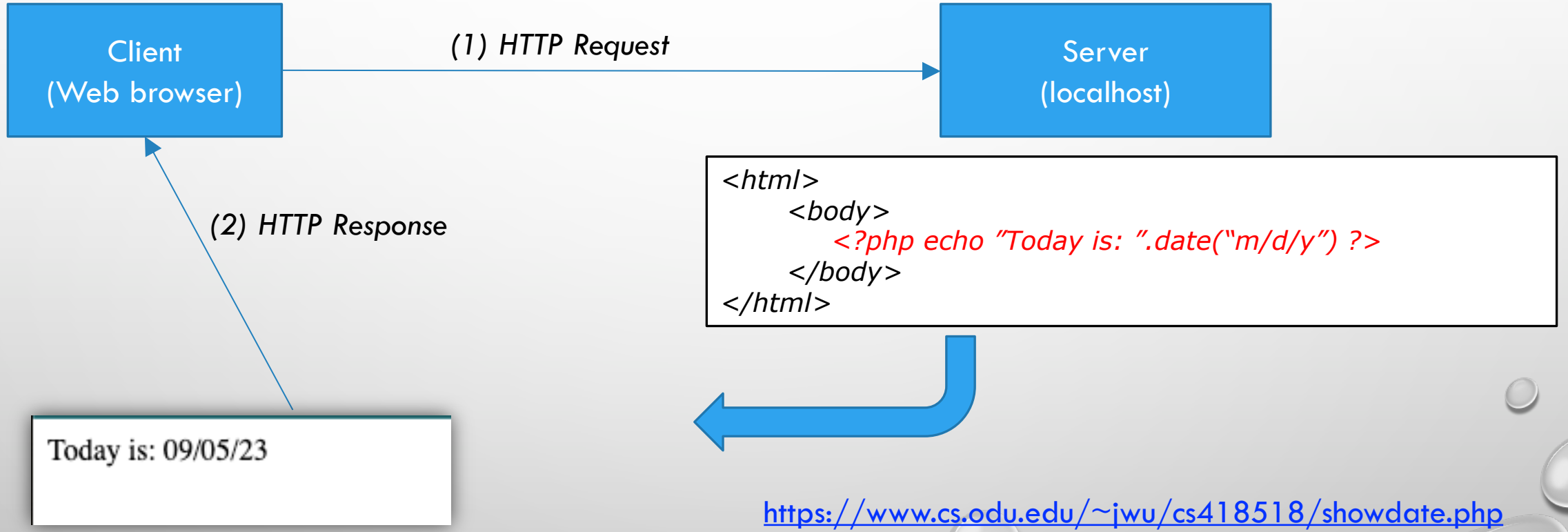
A HelloWorld EXAMPLE IN PHP

file extension MUST be .php

- https://www.cs.odu.edu/~jwu/cs418518/helloworld.php



ANOTHER EXAMPLE: SHOW DATE



https://www.cs.odu.edu/~jwu/cs418518/showdate.php

BASIC SYNTAX

- File extension must be “.php”
  - Start PHP code blocks with `<?php`
  - End PHP code blocks with `?>`
  - End lines with a `;`
- Comments (like C):
  - Single line: `// This is a comment`
  - Multi-line: `/* ... */`
- Write output with `echo "Hello world!";`
- Indicate variables with `$variableName`

Case Type	Example
Original Variable as String	`some awesome var`
Camel Case	`someAwesomeVar`
Snake Case	`some_awesome_var`
Kebab Case	`some-awesome-var`
Pascal Case	`SomeAwesomeVar`
Upper Case Snake Case	`SOME_AWESOME_VAR`

helloworld.php vs. helloworld.html

helloword.php

```
<html>
<head>
<title>PHP Test</title>
</head>
<body>
<?php echo '<p>Hello World!</p>'; ?>
</body>
</html>
```

A .php file can contain HTML elements.
A .html file cannot contain PHP statements.

name it as helloworld.html or helloworld.php

```
<html>
<head>
<title> PHP Test </title>
</head>
<body>
<h1>Hello World!</h1>
</body>
</html>
```

VARIABLE TYPES

- "boolean"
- "integer"
- "double" (for historical reasons "double" is returned in case of a float, and not simply "float")
- "string"
- "array"
- "object"
- "resource"
- “NULL”

```php
<?php

$data = array(1, 1., NULL, new stdClass, 'foo');

foreach ($data as $value) {
    echo gettype($value), "\n";
}

?>
```

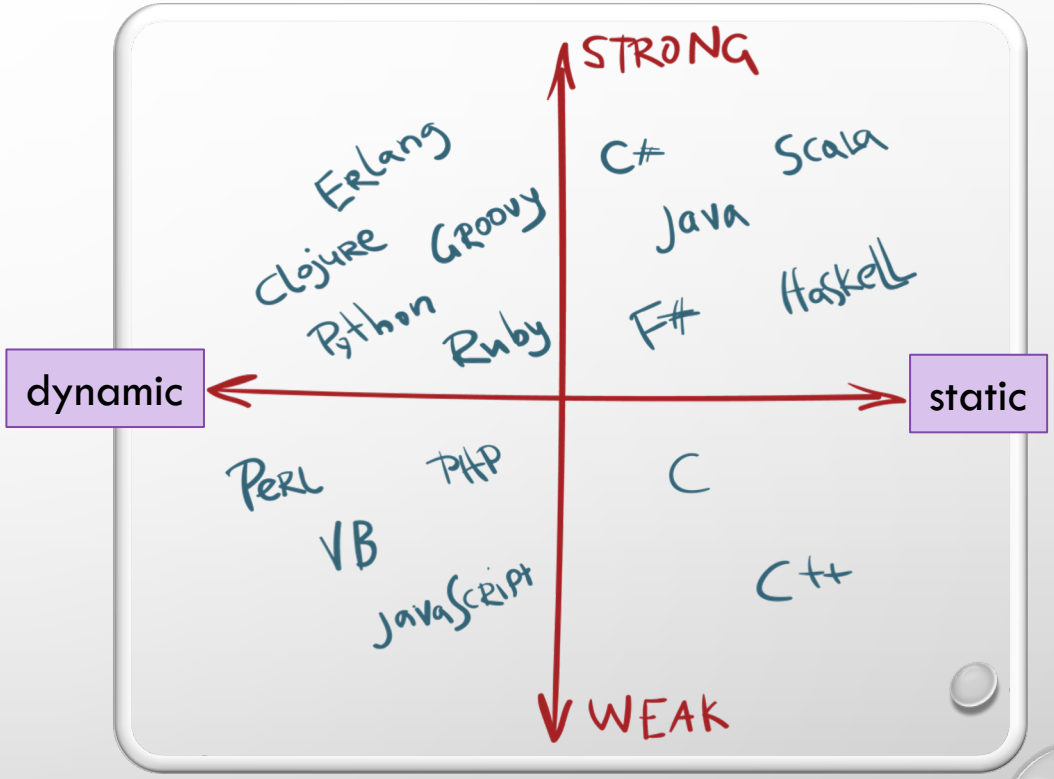
You can mix datatypes in an array.
gettype() can be used for obtaining the data type.

STRONG AND WEAK TYPING

- PHP is considered a weakly (loosely) typed language
- Weakly (Loosely) typed languages refer to those programming scripts that do not require defining the variable. To put it simply, while declaring a variable in these programming scripts, there is no need of defining or classifying the data type stored in them. So it is OK to do

  `10 + "10" + true`

  and it it will run fine (the result is 21).
- A **strongly-typed language** is one in which variables are bound to specific data types, and will result in **type errors** if types do not match up as expected in the expression — regardless of when **type** checking occurs. Examples: Java, Pascal, Go, Ada and C.
- Python is strongly, dynamically typed.



STRINGS

- A string is a sequence of characters, like "Hello world!".
- Commonly used string functions:
  - `echo strlen("Hello world!"); // outputs 12`
  - `echo str_word_count("Hello world!"); // outputs 2`
  - `echo strrev("Hello world!"); // outputs !dlrow olleH`
  - `echo strpos("Hello world!", "world"); // outputs 6: The first character's position is 0!`
  - `echo str_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly`
- String concatenation: string1 . string2 (a space is permitted between . and the string)

NUMBERS

Caution: This automatic conversion can sometimes break your code.

- PHP provides automatic data type conversion.
- If you assign an integer value to a variable, the type of that variable will automatically be an integer. Then, if you assign a string to the same variable, the type will change to a string. Range [-2147483648, 2147483647]. A value greater (or lower) than this, will be stored as double
- Commonly used functions
  - $x = 5985; var_dump(is_int($x)); // Check if the type of a variable is integer. Display bool(true)
  - $x = 10.365; var_dump(is_float($x)); // Check if the type of a variable is float
  - $x = 1.9e411; var_dump($x); // Check if a numeric value is finite or infinite:
  - $x = acos(8); var_dump($x); // Invalid calculation will return a NaN value
  - $x = “5985”; var_dump(is_numeric($x)); // Check if the variable is numeric
  - $x = 23465.768; $int_cast = (int)$x; echo $int_cast; // cast a float to int
  - $x = “23465.768”; $int_cast = (int)$x; echo $int_cast; // cast a string to in

var_dump() is like echo,
but it prints all information about a variable.

MATH

- echo(pi()); // returns 3.1415926535898
- echo(min(0, 150, 30, 20, -8, -200)); // returns -200
  echo(max(0, 150, 30, 20, -8, -200)); // returns 150
- echo(abs(-6.7)); // returns 6.7
- echo(sqrt(64)); // returns 8
- echo(round(0.60)); // returns 1
  echo(round(0.49)); // returns 0
- echo(rand());
- echo(rand(10, 100)); // random integer between 10 and 100 (inclusive)
- Complete math reference: https://www.w3schools.com/php/php_ref_math.asp

ARITHMETIC AND ASSIGNMENT OPERATORS

Arithmetic operators

Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y
**	Exponentiation	$x ** $y	Result of raising $x to the $y'th power

Assignment Operators

Assignment	Same as...	Description
x = y	x = y	The left operand gets set to the value of the expression on the right
x += y	x = x + y	Addition
x -= y	x = x - y	Subtraction
x *= y	x = x * y	Multiplication
x /= y	x = x / y	Division
x %= y	x = x % y	Modulus

COMPARISON OPERATORS

Operator	Name	Example	Result
==	Equal	$x == $y	Returns true if $x is equal to $y
===	Identical	$x === $y	Returns true if $x is equal to $y, and they are of the same type
!=	Not equal	$x != $y	Returns true if $x is not equal to $y
<>	Not equal	$x <> $y	Returns true if $x is not equal to $y
!==	Not identical	$x !== $y	Returns true if $x is not equal to $y, or they are not of the same type
>	Greater than	$x > $y	Returns true if $x is greater than $y
<	Less than	$x < $y	Returns true if $x is less than $y
>=	Greater than or equal to	$x >= $y	Returns true if $x is greater than or equal to $y
<=	Less than or equal to	$x <= $y	Returns true if $x is less than or equal to $y
<=>	Spaceship	$x <=> $y	Returns an integer less than, equal to, or greater than zero, depending on if $x is less than, equal to, or greater than $y. Introduced in PHP 7.

INCREMENT/DECREMENT OPERATORS

Operator	Name	Description
++$x	Pre-increment	Increments $x by one, then returns $x
$x++	Post-increment	Returns $x, then increments $x by one
--$x	Pre-decrement	Decrements $x by one, then returns $x
$x--	Post-decrement	Returns $x, then decrements $x by one

Logical operators

Operator	Name	Example	Result
and	And	$x and $y	True if both $x and $y are true
or	Or	$x or $y	True if either $x or $y is true
xor	Xor	$x xor $y	True if either $x or $y is true, but not both
&&	And	$x && $y	True if both $x and $y are true
||	Or	$x || $y	True if either $x or $y is true
!	Not	!$x	True if $x is not true

ARRAY OPERATORS

Operator	Name	Example	Result
+	Union	$x + $y	Union of $x and $y
==	Equality	$x == $y	Returns true if $x and $y have the same key/value pairs
===	Identity	$x === $y	Returns true if $x and $y have the same key/value pairs in the same order and of the same types
!=	Inequality	$x != $y	Returns true if $x is not equal to $y
<>	Inequality	$x <> $y	Returns true if $x is not equal to $y
!==	Non-identity	$x !== $y	Returns true if $x is not identical to $y

Conditional Assignment Operators

Operator	Name	Example	Result
?:	Ternary	$x = *expr1* ? *expr2* : *expr3*	Returns the value of $x. The value of $x is *expr2* if *expr1* = TRUE. The value of $x is *expr3* if *expr1* = FALSE
??	Null coalescing	$x = *expr1* ?? *expr2*	Returns the value of $x. The value of $x is *expr1* if *expr1* exists, and is not NULL. If *expr1* does not exist, or is NULL, the value of $x is *expr2*. Introduced in PHP 7

[For examples, see https://www.w3schools.com/php/php_operators.asp](https://www.w3schools.com/php/php_operators.asp)

CONTROL STATEMENTS

- if statement - executes some code if one condition is true
- if...else statement - executes some code if a condition is true and another code if that condition is false
- if...elseif...else statement - executes different codes for more than two conditions
- switch statement - selects one of many blocks of code to be executed

```php
<?php
$t = date("H");

if ($t < "20") {
  echo "Have a good day!";
}
?>
```

```php
<?php
$t = date("H");

if ($t < "20") {
  echo "Have a good day!";
} else {
  echo "Have a good night!";
}
?>
```

```php
<?php
$t = date("H");

if ($t < "10") {
  echo "Have a good morning!";
} elseif ($t < "20") {
  echo "Have a good day!";
} else {
  echo "Have a good night!";
}
?>
```

SWITCH STATEMENT

must add "break" to jump out of switch()

```php
<?php
$favcolor = "red";

switch ($favcolor) {
  case "red":
    echo "Your favorite color is red!";
    break;
  case "blue":
    echo "Your favorite color is blue!";
    break;
  case "green":
    echo "Your favorite color is green!";
    break;
  default:
    echo "Your favorite color is neither red, blue, nor green!";
}
?>
```

PHP SUPER GLOBAL VARIABLES

These are all associative arrays, i.e., arrays of (key, value) pairs.

- `$_SERVER` – server parameters
- `$_GET` – variables when the method is GET
  - Variables are in the query string
- `$_POST` – variables when the method is POST (previous example)
  - Form data is not present in the query string
  - Frequently use when passing credentials
- `$_COOKIE` – values stored in the browser's cookies for the page
- `$_REQUEST` – associative array of $_GET, $_POST and $_COOKIE

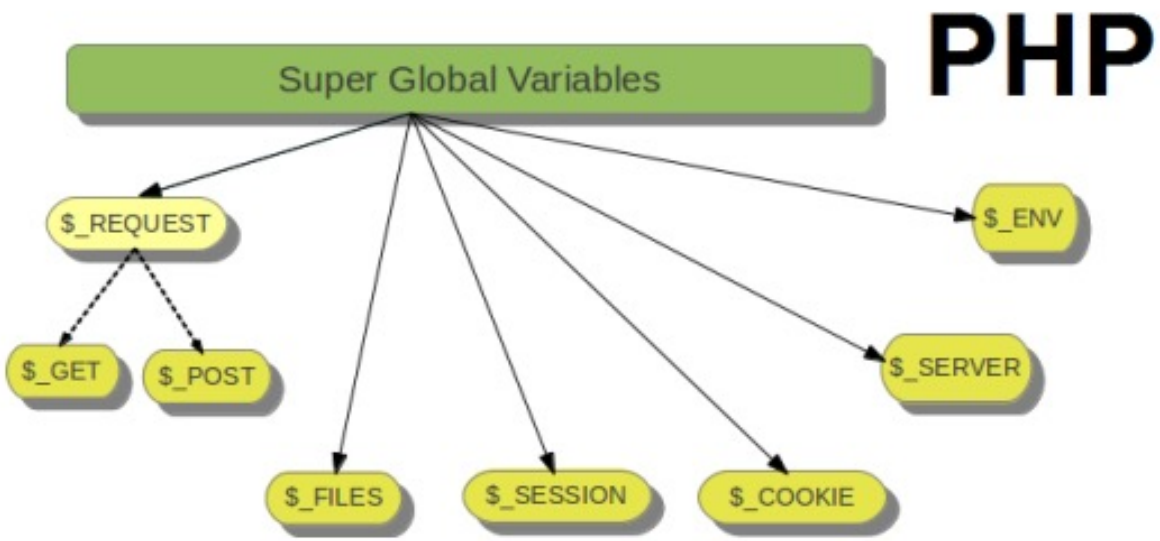
```php
<?php

$_GET['foo'] = 'a';
$_POST['bar'] = 'b';
var_dump($_GET); // Element 'foo' is string(1) "a"
var_dump($_POST); // Element 'bar' is string(1) "b"
var_dump($_REQUEST); // Does not contain elements 'foo' or 'bar'

?>
```

SUPER GLOBAL VARIABLES

http://localhost/superglobalvar.php




```php
<html>
  <body>
    <?php echo "Server name: ".$_SERVER['SERVER_NAME'] ?> </br>
    <?php echo "Http user agent: ".$_SERVER['HTTP_USER_AGENT']?>
  </body>
</html>
```

$_REQUEST is An associative array that by default contains the contents of *$_GET*, *$_POST*.

ARRAYS

- Arrays
  - `$names = array("John", "Mike", "Steve")`
  - `echo $names // Array`
  - `print_r($names); // Array ([0]=>John  [1]=>Mike  [2]=> Steve)`
- Key-value pairs
  - ```
    $scores = array (
    "Old Dominion" => 20,
     "Virginia Tech" => 10);
    ```
    ```

A PHP ARRAY IS ALSO AN ASSOCIATIVE ARRAY

- $employee = array ("Tom","Robertson","21","13.5","male","cashier","2.5");

KEYS	VALUES
0	Tom
1	Robertson
2	21
3	13.5
4	male
5	cashier
6	2.5

LOOPS

```
foreach(array as key => value){
    //statements
}
foreach(array as value){
    //statements
}
```

```php
<?php
$age
= array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43"
);

foreach($age as $x => $val) {
  echo "$x = $val<br>";
}
?>
```

```php
<?php
$colors
= array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
  echo "$value <br>";
}
?>
```

FOR LOOP

```
for (init counter; test counter; increment counter) {
  code to be executed for each iteration;
}
```

```
<?php
for ($x = 0; $x <= 10; $x++) {
  echo "The number is: $x <br>";
}
?>
```

BUILT-IN FUNCTIONS

Use a function:

```
<html>
<body>
    <?php
        echo "A: What is the day today?<br>";
        echo "B: ".date("l, F d Y")."<br>";
        echo "A: <b>Not Friday yet?!</b><br>"
    ?>
</body>
</html>
```

l: a full textual representation of the day after the week: Sunday through Saturday.
F: A full textual representation of a month, such as January or March.
d: Day of the month, 2 digits with leading zeros.
Y: A two-digit representation of a year, e.g., 99, 01.
https://www.w3schools.com/php/php_date.asp

More string functions
http://php.net/manual/en/book.strings.php
More math functions
http://php.net/manual/en/ref.math.php

USER DEFINED FUNCTIONS

```
function functionName(){
//statements
}
```

```php
<?php
function writeMsg() {
  echo "Hello world!";
}

writeMsg(); // call the function
?>
```

LOCAL AND GLOBAL VARIABLES

```php
<?php
$a = 1; /* global scope */

function test()
{
    echo $a; /* reference to local scope variable */
}

test();
?>
```

OBJECTS

```
class Person {
    function __construct($name){
        $this->name = $name;
    }
    function whatDoYouDo(){
        echo $this->name." dances!";
    }
}
```

```
class Programmer extends Person {
    function startCoding($language){
        echo $this->name." is coding in ".$language."!";
    }
    function whatDoYouDo(){
        echo $this->name." codes!";
    }
}
```

```
$mary = new Person("Mary");
$mary->whatDoYouDo();
$john = new Programmer("John");
$john->whatDoYouDo();
$john->startCoding("PHP");
```

OBJECT EXAMPLES

php constructor

```php
<?php
class Fruit {
  public $name;
  public $color;

  function __construct($name) {
    $this->name = $name;
  }
  function get_name() {
    return $this->name;
  }
}

$apple = new Fruit("Apple");
echo $apple->get_name();
?>
```

php object

```php
<?php
class Fruit {
  // Properties
  public $name;
  public $color;

  // Methods
  function set_name($name) {
    $this->name = $name;
  }
  function get_name() {
    return $this->name;
  }
}
?>
```

REGULAR EXPRESSIONS (REGEX)

- A regular expression is a sequence of characters that forms a search pattern.
- When you search for data in a text, you can use this search pattern to describe what you are searching for.

```
$exp = "/w3schools/i”;
```

- / is the **delimiter,** *w3schools* is the **pattern** that is being searched for, and i is a **modifier** that makes the search case-insensitive.
- Regular expression functions:

Function	Description
preg_match()	Returns 1 if the pattern was found in the string and 0 if not
preg_match_all()	Returns the number of times the pattern was found in the string, which may also be 0
preg_replace()	Returns a new string where matched patterns have been replaced with another string

REGULAR EXPRESSION EXAMPLES

- a case-insensitive search for "w3schools" in a string:

```php
<?php
$str = "Visit W3Schools";
$pattern = "/w3schools/i";
echo preg_match($pattern, $str); // Outputs 1
?>
```

- a case-insensitive count of the number of occurrences of "ain" in a string

```php
<?php
$str = "The rain in SPAIN falls mainly on the plains.";
$pattern = "/ain/i";
echo preg_match_all($pattern, $str); // Outputs 4
?>
```

- a case-insensitive regular expression to replace Microsoft with W3Schools in a string

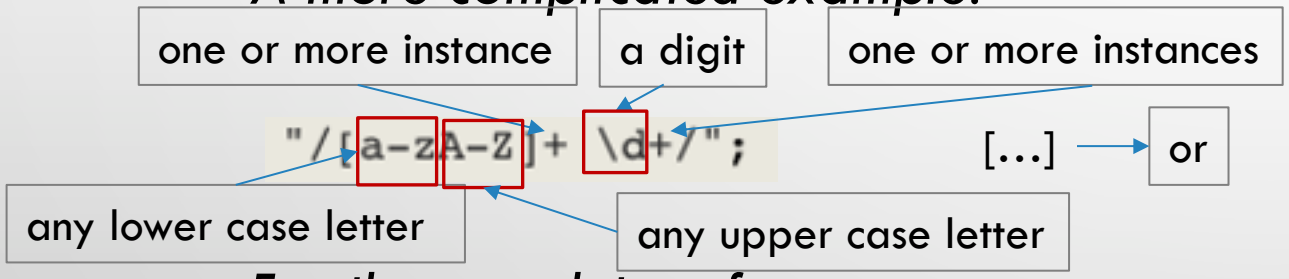
```php
<?php
$str = "Visit Microsoft!";
$pattern = "/microsoft/i";
echo preg_replace($pattern, "W3Schools", $str); // Outputs
"Visit W3Schools!"
?>
```

GROUPING IN REGEX

- Use grouping to search for the word "banana" by looking for *ba* followed by two instances of *na*.

```php
<?php
$str = "Apples and bananas.";
$pattern = "/ba(na){2}/i";
echo preg_match($pattern, $str); // Outputs 1
?>
```

- *A more complicated example:*




```php
// Lets use a regular expression to match a date string. Ignore
// the output since we are just testing if the regex matches.
$regex = "/[a-zA-Z]+ \d+/";
if (preg_match($regex, "June 24")) {
    // Indeed, the expression "[a-zA-Z]+ \d+" matches the date string
    echo "Found a match!";
} else {
    // If preg_match() returns false, then the regex does not
    // match the string
    echo "The regex pattern does not match. :(";
}
```

- *For the complete reference, see* [*https://www.w3schools.com/php/php_ref_regex.asp*](https://www.w3schools.com/php/php_ref_regex.asp)

HOW TO RUN

- PHP script is interpreted on the **server**, NOT the client
- You cannot open a .php file directly by a web browser
- You must
  1. Put the .php file on a web server
  2. Access it from a URL (a.k.a. web address)
  3. One place to test if a php script is working is linux.cs.odu.edu:~/secure_html/

- PHP is case sensitive for
  - variables, e.g., `$names` vs. `$Names`
  - constants, e.g., `"names"` vs. `"Names"`
  - array keys, e.g., `$scores = array ("odu"=>1,"ODU"=>2);`
- PHP is case insensitive for
  - functions, e.g., `date()` and `DATE()` are the same
- HTML is case insensitive for
  - tag names (though most people use lowercase)
  - e.g., `<head>` and `<HEAD>` are the same

LEARNING PHP BY EXAMPLES

- We go straight to problems and explain fundamentals

BACKUP SLIDES BEYOND THIS POINT

JIAN WU