

# Personal DB/PHP Overview

September 11, 2014

# Outline

1. What is PHP ?
2. PHP Language Basics
3. Demo

# What is PHP?

PHP is a programming language that can do all sorts of things:

- Evaluate form data sent from a browser
- Build custom web content to serve the browser
- Talk to a database
- Send and receive cookies

Reference - <http://www.codecademy.com/courses/web-beginner-en-StaFQ/0/2>

# HTML vs. PHP Request

<http://www.berkeley.edu/academics/index.html>



Server will simply return the HTML file

---

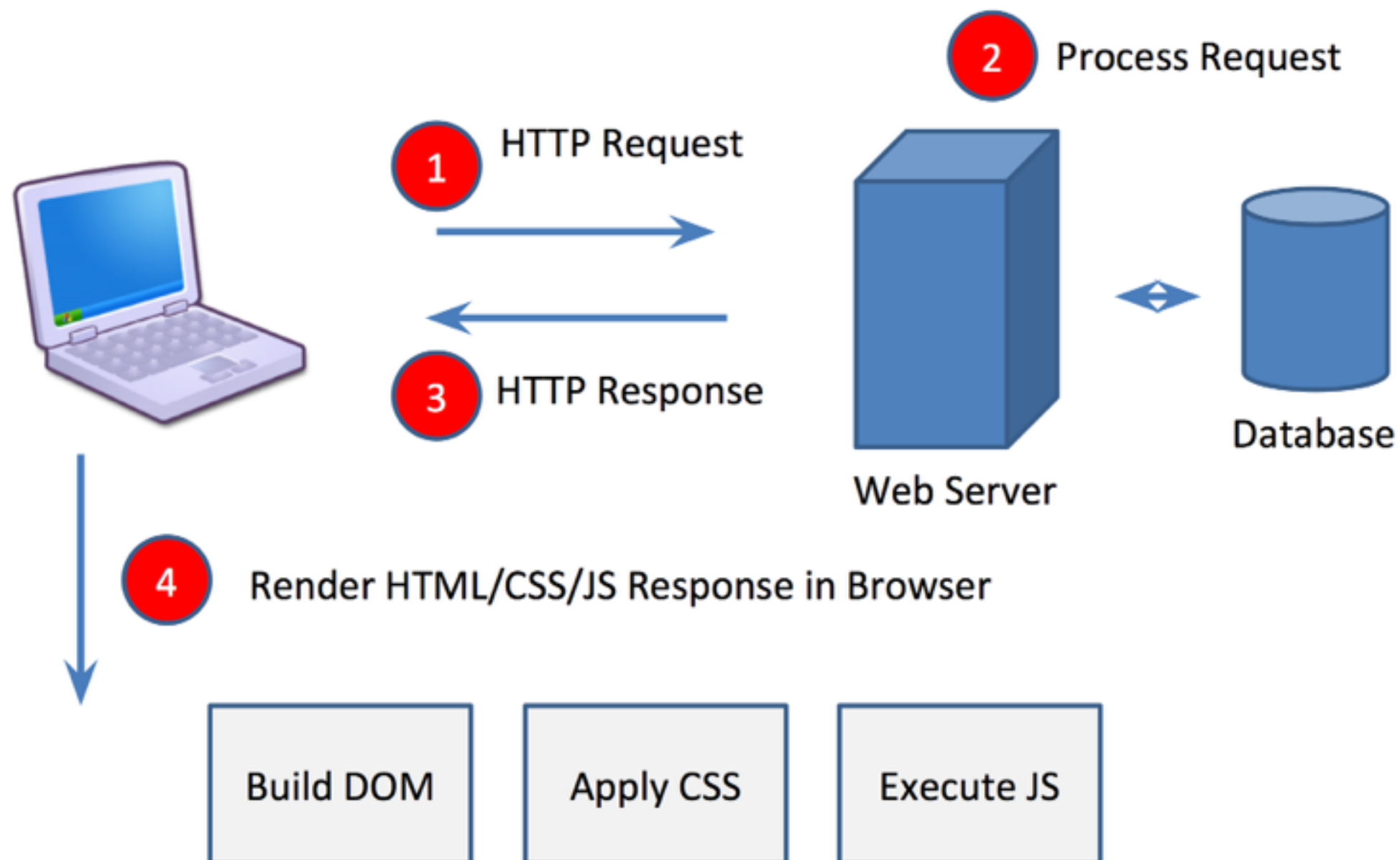
<http://www.berkeley.edu/academics/index.php>



Server will

1. Parse the php file,
2. Execute any php code within the file,
3. Then return a dynamically generated html file

# PHP Process



**Demo**

# PHP Block

```
<?php  
//PHP Code...  
?>
```

- Can put this anywhere and any number of times within the html of your PHP file.
- The code in PHP Blocks merge together to form a single PHP execution script. So if you define a variable in an earlier block, it will be available in a latter block as well.

# PHP statements

`variable_name = value;`

## Data Types

String                      `$a = "some string of characters";`

Integer (whole #'s)      `$a = 1;`

Float (fractional #'s)   `$a = 1.1;`

Boolean                    `$a = true;`

Arrays                     `$a = array(1, 2, 3);`

Objects

*Resources*

*null*                        `$a = null;`

*Special type that basically means "nothing".*

<http://www.codecademy.com/courses/web-beginner-en-StaFQ/1/4>

<http://www.codecademy.com/courses/web-beginner-en-StaFQ/2/1>



# PHP statements

## If... elseif... else

```
if (conditional expression 1){  
    //code to execute if conditional expression 1 = true  
}  
elseif (conditional expression 2){  
    //code to execute if conditional expression 2 = true  
}  
else{  
}
```

The conditional expression needs to be of a boolean (true/false) type. If you provide a variable or function that is not of a boolean type, it will either try to convert it or it will give you an error.

# PHP statements

## echo

When you want to spit out some content into your html, use echo <?php

```
echo "Some Text";
```

```
echo $variable_name;
```

```
echo $someArray[1] . " " . $someArray[2];
```

```
?>
```

Note, there is also a “print” function that is very similar to *echo* but *echo* is more useful since it can print out multiple variables/objects with a single call

<http://www.codecademy.com/courses/web-beginner-en-StaFQ/1/1>

# PHP statements

## Functions

```
function my_function($param1, $param2){  
    //code to execute...  
    //possibly “return” a value/object  
}
```

*Call the Function*

```
$a = 1;  
$b = 2; my_function($a, $b);
```

# PHP statements

## Conditionals

Example	Name	Result
<code>\$a == \$b</code>	Equal	<b>TRUE</b> if <code>\$a</code> is equal to <code>\$b</code> after type juggling.
<code>\$a === \$b</code>	Identical	<b>TRUE</b> if <code>\$a</code> is equal to <code>\$b</code> , and they are of the same type.
<code>\$a != \$b</code>	Not equal	<b>TRUE</b> if <code>\$a</code> is not equal to <code>\$b</code> after type juggling.
<code>\$a &lt;&gt; \$b</code>	Not equal	<b>TRUE</b> if <code>\$a</code> is not equal to <code>\$b</code> after type juggling.
<code>\$a !== \$b</code>	Not identical	<b>TRUE</b> if <code>\$a</code> is not equal to <code>\$b</code> , or they are not of the same type.
<code>\$a &lt; \$b</code>	Less than	<b>TRUE</b> if <code>\$a</code> is strictly less than <code>\$b</code> .
<code>\$a &gt; \$b</code>	Greater than	<b>TRUE</b> if <code>\$a</code> is strictly greater than <code>\$b</code> .
<code>\$a &lt;= \$b</code>	Less than or equal to	<b>TRUE</b> if <code>\$a</code> is less than or equal to <code>\$b</code> .
<code>\$a &gt;= \$b</code>	Greater than or equal to	<b>TRUE</b> if <code>\$a</code> is greater than or equal to <code>\$b</code> .

If we compare numbers to strings (i.e. `1 == "1"`), then string is converted to number and then compared. Note, this does not happen when you do the `"==="` or `"!=="` comparison

<http://www.php.net/manual/en/language.operators.comparison.php>

# PHP statements

## Logical operators

AND && OR ||

Examples (\$a = 1, \$b = 2, \$c = 2)

(\$a > \$b && \$b == \$c) → FALSE

(\$a < \$b || \$a == \$c) → TRUE

(\$a == \$c || \$b == \$c || somefunc(\$a)) → TRUE

# PHP statements

## For Loops

1 2 4

```
for($i = 0; $i < 10; $i++){  
    //execute code  
    3  
    //You can use $i in your code  
}
```

**continue;** Stop execution and jump to “4”

**break;** Break out of loop

Note, you can use “continue;” and “break” in other loops as well (like the “while” loop)

<http://www.codecademy.com/courses/web-beginner-en-L83Do/0/2> <http://www.codecademy.com/courses/web-beginner-en-L83Do/0/3>

1

2

3

4

5

2

3

4

# PHP statements

## While Loops

```
while (conditional expression){
```

```
//code to execute while conditional expression evaluates to true //  
commonly used to read row results from SQL query
```

```
} do{
```

```
//code executes at least once, then continues to execute while conditional  
expression evaluates to true
```

```
} while(conditional expression)
```

<http://www.codecademy.com/courses/web-beginner-en-5YvPF/0/2>

<http://www.codecademy.com/courses/web-beginner-en-5YvPF/0/3>

# PHP statements

## Working with Arrays

**Arrays can contain mixed data types**

```
$myarray = array(1, "string", true);
```

**Associative Arrays** (kind of like Dictionaries, except that the array maintains order)

```
$myarray = array("somekey" => 1, "anotherkey" => "value")
```

```
$myarray["somekey"] → 1
```

Note, can also access by index: `$myarray[0] → 1`

Can add new arrays elements after array has been instantiated by

```
$myarray["someotherkey"] = "string";
```

**Multidimensional Arrays**

```
$locations = array();
```

```
$locations["sanfran"] = array("lat" => 100, "long" => 100); $locations["la"] = array("lat" => 150, "long" => 150); $locations["sanfran"]["lat"] → 100
```

<http://www.codecademy.com/courses/web-beginner-en-8a35h/0/2> (6 exercises)



# PHP statements

## Traversing Arrays

```
foreach ($array as $value){
```

```
//execute code for each value in the array ($value)
```

```
}
```

```
foreach ($array as $key => $value){
```

```
//execute code for each key/value in the array ($value, $key)
```

```
}
```

There are MANY functions that operate on arrays, including sorting, merging, popping, pushing, shifting, splicing, etc. Check out php.net for a full list...

<http://www.codecademy.com/courses/web-beginner-en-L83Do/0/5>

<http://www.codecademy.com/courses/web-beginner-en-L83Do/0/6>

# PHP statements

## isset() vs. empty()

isset() checks to see if a variable is “set,” meaning it is not null. Returns false if not set and true otherwise.

empty() checks to see whether a variables value is empty, meaning whether or not it evaluates to false

```
$a = 0;  
$b = “”;  
$c = null;  
$d = “string”;
```

isset(\$a) → True	isset(\$d) → True
isset(\$b) → True	empty(\$b) → True
isset(\$c) → False	empty(\$c) → True
empty(\$a) → True	empty(\$d) → False

These are used a lot to check whether form inputs sent empty values or whether database results contain null values...

# Appendix

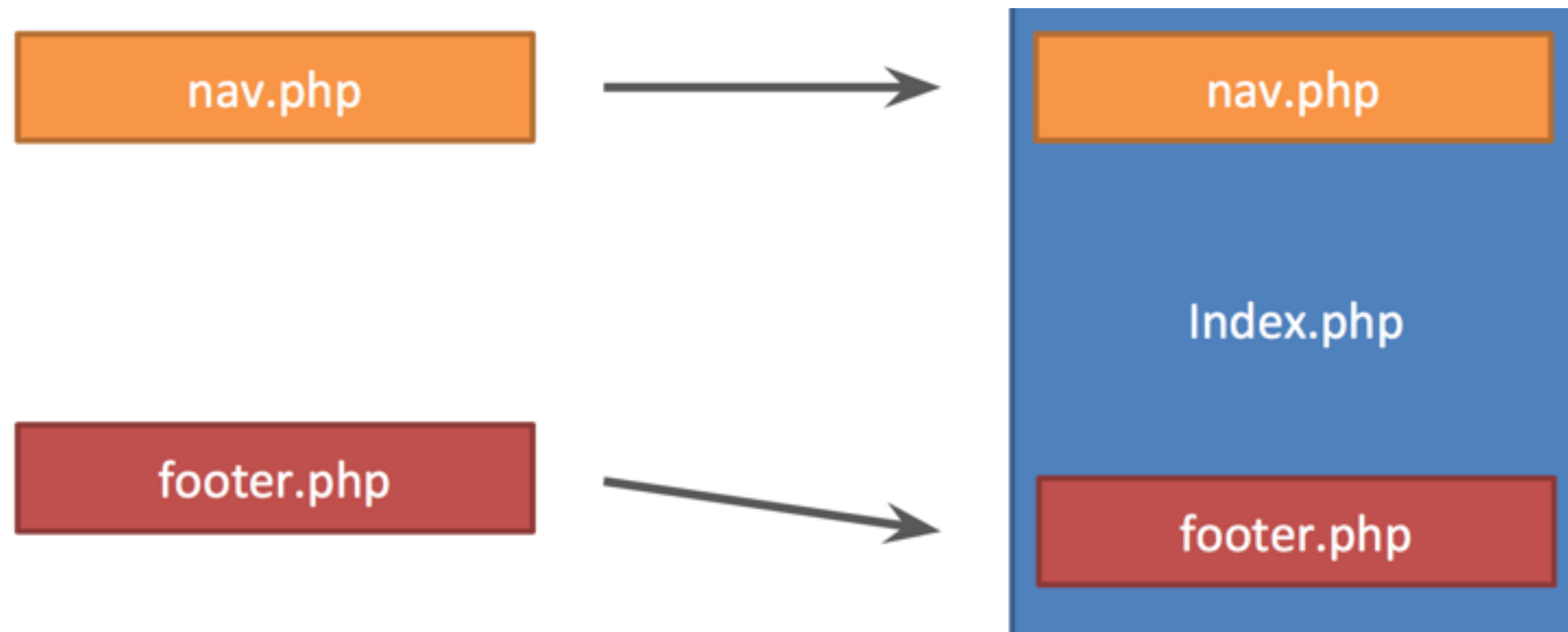
# Super Globals

There are predefined “Super Global” variables that are made available to you through the PHP runtime that you can use within your PHP code.

Super Global	Content
<code>\$_SERVER</code>	Contains info about the web server environment such as Server Name, Request Method, Document Root, etc.
<code>\$_GET</code>	Contains any GET variables in the URL query string
<code>\$_POST</code>	Contains any POST variables submitted via a form post submission
<code>\$_COOKIE</code>	Contains any HTTP Cookie Info
<code>\$_FILES</code>	Contains information on POST file uploads
<code>\$_SESSION</code>	Contains information about any variables registered in a session (if created)

There are some other super globals as well but these are the main ones

# Include



```
<?php include('C:/Path/To/Your/nav.php'); ?>
```

Note, in your folder structure it is a good idea to create a specific folder that contains all your “include files”

# Popular use of Include

Including a set of PHP functions and/or classes

Including any variables that are defined in other PHP files (like `$document_root`, `$your_variable`, etc.)

Including HTML that appears on more than 1 of your pages (i.e. menu, header, footer, sidebars, widgets, etc.)

Including objects that get instantiated in other PHP files (i.e. including a connection handle to your MySQL data