Foundations to Get You Started

Beth Tucker Long

Who am I?

- Elizabeth Tucker Long (Beth - @e3betht)
- Editor-in-Chief of php|architect magazine

Want to write?
See me after.

- PHP Essentials Instructor
- Freelance consultant

How this talk will work:

- I'll be presenting the concepts on the left of the slide and the code on the right
- Code will build from slide to slide, but due to space constraints, we can't show it all on one slide
- The code being shown is for teaching use only, it is not optimized in any way. It should not be used as is for a live system.

Getting Started

- Start with your standard HTML
- **Opening PHP tag**
- **PHP code**
- **Closing PHP tag**
- Close out your HTML page

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
    transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xml:lang="en" lang="en">
<head><title>Order Some Pizza</title></head>
<body>

<h1>Welcome to Joe's Pizza!</h1>

<?php

    $name = "Beth";
    echo "<p>$name's order:</p>";

?>

</body></html>
```

Start with your standard HTML

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
  Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
  transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
  xml:lang="en" lang="en">
<head><title>Order Some Pizza</title></head>
<body>

<h1>Welcome to Joe's Pizza!</h1>
```

Opening PHP tag

```
<?php
```

Short Opening PHP tag

```
<?
```

PHP code

```
$name = "Beth";
```

Variables:

- Preceded by a dollar sign - $
- May contain letters, numbers, and underscores
- Must start with a letter or underscore
- Loosely-typed

PHP code

```
echo "<p>$name's order:</p>";
```

Other ways to quote a string

```
echo '<p>$name\'s order:</p>';

echo <<<MYSTRING
<p>$name's order:</p>
MYSTRING;
```

Closing PHP tag

```
?>
```

Close out your HTML page

```
</body></html>
```

Getting Started

- Start with your standard HTML
- Opening PHP tag
- PHP code
- Closing PHP tag
- Close out your HTML page

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
    transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xml:lang="en" lang="en">
<head><title>Order Some Pizza</title></head>
<body>

<h1>Welcome to Joe's Pizza!</h1>

<?php

    $name = "Beth";
    echo "<p>$name's order:</p>";

?>

</body></html>
```

Making Comments

```
// This is a comment

# This is a comment

/* This is also a comment
and one you are very likely
to see in code
*/
```

The Order Form

- We'll create a form that looks like this:

Name:

Choose a Size:
o Small
o Medium
o Large

Add Toppings:
☐ Mushrooms ☐ Extra Cheese
☐ Green Peppers ☐ Pepperoni
☐ Black Olives ☐ Sausage

Place Order

```
<form action="./orderPizza.php" method="POST">

<p>Name: <input type="text" name="custName" maxlength="200"/></p>

<p>Choose a Size:<br />
<input type="radio" name="pizzaSize" value="Small" /> Small<br />
<input type="radio" name="pizzaSize" value="Medium" /> Medium<br
    />
<input type="radio" name="pizzaSize" value="Large" /> Large</p>

<p>Add Additional Toppings:<br />
<input type="checkbox" name="Mushrooms" value="Yes" />
    Mushrooms<br />
<input type="checkbox" name="GreenPeppers" value="Yes" /> Green
    Peppers<br />
<input type="checkbox" name="BlackOlives" value="Yes" /> Black
    Olives <br />
<input type="checkbox" name="ExtraCheese" value="Yes" /> Extra
    Cheese<br />
<input type="checkbox" name="Pepperoni" value="Yes" />
    Pepperoni<br />
<input type="checkbox" name="Sausage" value="Yes" /> Sausage</p>

<input type="submit" name="pizzaStatus" value="Place Order" />

</form>
```

Starting the Form

```
<form action="./orderPizza.php" method="POST">
```

Text Field

Name:

```
<p>Name: <input type="text" name="custName"
  maxlength="200"/></p>
```

Radio Buttons

Choose a Size:

o Small

o Medium

o Large

```
<p>Choose a Size:<br />
<input type="radio" name="pizzaSize"
  value="Small" /> Small<br />
<input type="radio" name="pizzaSize"
  value="Medium" /> Medium<br />
<input type="radio" name="pizzaSize"
  value="Large" /> Large</p>
```

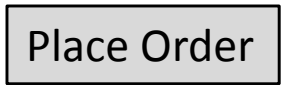
Checkboxes

Add Toppings:

☐ Mushrooms ☐ Extra Cheese
☐ Green Peppers ☐ Pepperoni
☐ Black Olives ☐ Sausage

```
<input type="checkbox" name="Mushrooms"
  value="Yes" /> Mushrooms<br />
<input type="checkbox" name="GreenPeppers"
  value="Yes" /> Green Peppers<br />
<input type="checkbox" name="BlackOlives"
  value="Yes" /> Black Olives <br />
<input type="checkbox" name="ExtraCheese"
  value="Yes" /> Extra Cheese<br />
<input type="checkbox" name="Pepperoni"
  value="Yes" /> Pepperoni<br />
```

Input Field



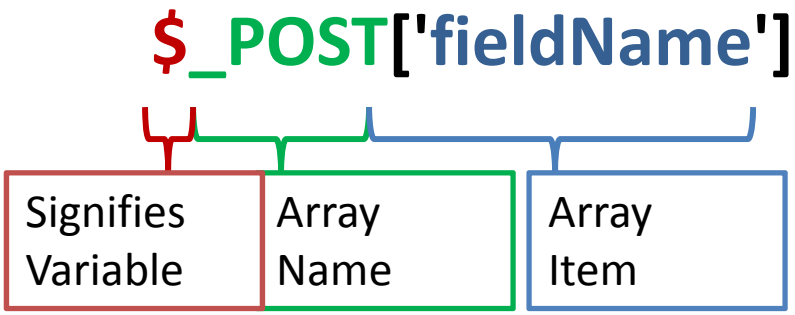

```
<input type="submit" name="pizzaStatus"
  value="Place Order" />
```

Closing the form

```
</form>
```

What we get

- Data from the form comes through as a superglobal, automatically created by PHP for us in an array (a variable that is a container for other variables):



Two Types of Arrays

- Enumerated: $arrayName[]
  - $arrayName[0]
  - $arrayName[1]

- Associative: $arrayName[stringName]
  - $arrayName['firstName']
  - $arrayName['lastName']

What we get

So the variables from our form are:

- $_POST['custName']
- $_POST['pizzaSize']
- $_POST['Mushrooms']
- $_POST['ExtraCheese']
- $_POST['GreenPeppers']
- $_POST['Pepperoni']
- $_POST['Black Olives']
- $_POST['Sausage']
- $_POST['pizzaStatus']

Revised: Adding an Input Array

- We'll create a form that looks like this:

Name:

Choose a Size:
o Small
o Medium
o Large

Add Toppings:
☐ Mushrooms ☐ Extra Cheese
☐ Green Peppers ☐ Pepperoni
☐ Black Olives ☐ Sausage

Place Order

```
<form action="./orderPizza.php" method="POST">

<p>Name: <input type="text" name="custName" maxlength="200"
    /></p>

<p>Choose a Size:<br />
<input type="radio" name="pizzaSize" value="Small" /> Small<br />
<input type="radio" name="pizzaSize" value="Medium" /> Medium<br
    />
<input type="radio" name="pizzaSize" value="Large" /> Large</p>

<p>Add Additional Toppings:<br />
<input type="checkbox" name="pizzaToppings[]" value="Mushrooms"
    /> Mushrooms<br />
<input type="checkbox" name="pizzaToppings[]" value="Green
    Peppers" /> Green Peppers<br />
<input type="checkbox" name="pizzaToppings[]" value="Black
    Olives" /> Black Olives<br />
<input type="checkbox" name="pizzaToppings[]" value="Extra
    Cheese" /> Extra Cheese<br />
<input type="checkbox" name="pizzaToppings[]" value="Pepperoni"
    /> Pepperoni<br />
<input type="checkbox" name="pizzaToppings[]" value="Sausage" />
    Sausage</p>

<input type="submit" name="pizzaStatus" value="Place Order" />

</form>
```

Revised: Adding an Input Array

```
<p>Add Additional Toppings:<br />
<input type="checkbox" name="pizzaToppings[]"
  value="Mushrooms" /> Mushrooms<br />
<input type="checkbox" name="pizzaToppings[]"
  value="Green Peppers" /> Green Peppers<br />
<input type="checkbox" name="pizzaToppings[]"
  value="Black Olives" /> Black Olives<br />
<input type="checkbox" name="pizzaToppings[]"
  value="Extra Cheese" /> Extra Cheese<br />
<input type="checkbox" name="pizzaToppings[]"
  value="Pepperoni" /> Pepperoni<br />
<input type="checkbox" name="pizzaToppings[]"
  value="Sausage" /> Sausage</p>
```

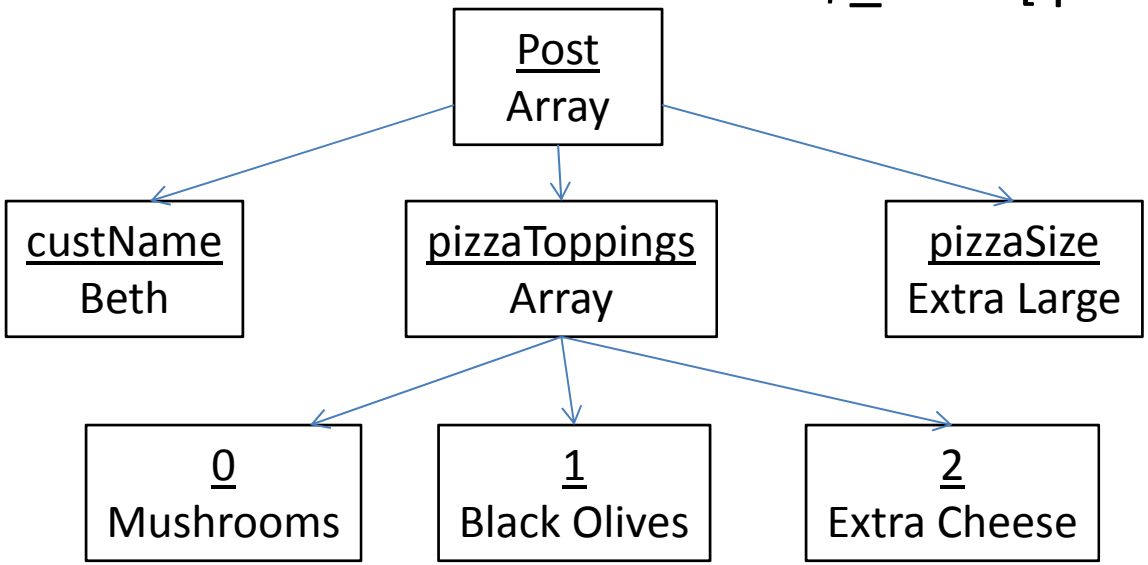
Revised: What we get

- A superglobal that is also an array is called a nested array (an array within an array):

  $_POST['fieldName']['fieldname']

So the variables from our form are:

- $_POST['custName']
- $_POST['pizzaSize']
- $_POST['pizzaToppings']
- $_POST['pizzaStatus']



Quoting Arrays

```
{$_POST['custName']}

.$_POST['pizzaSize'].


echo "<p>This order is for
  {$_POST['custName']}</p>
<p>Size: ".$_POST['pizzaSize']."</p>
<p>Toppings:</p><ul>";
```

Processing the Pizza Specs

- We want to sort through the toppings submitted. We'll use a loop.

```
echo "<p>This order is for
   {$_POST['custName']}</p>
<p>Size: ".$_POST['pizzaSize']."</p>
<p>Toppings:</p><ul>";
foreach($_POST['pizzaToppings'] as
   $topping)  {
   echo "<li>$topping</li>";
}
echo "</ul>";
```

foreach Loop

```
foreach($_POST['pizzaToppings'] as
  $topping)  {

  echo "<li>$topping</li>";

}
```

Making Decisions

```
<?php
if ($_POST['pizzaStatus'] == "Place Order") {
    echo "<p>This order is for {$_POST['custName']}</p>
    <p>Size: ".$_POST['pizzaSize']."</p>
    <p>Toppings:</p><ul>";
    foreach($_POST['pizzaToppings'] as $topping)  {
           echo "<li>$topping</li>";
    }
    echo "</ul>";
}
else {
?>
    <form action="./orderPizza.php" method="POST">
    <p>Name: <input type="text" name="custName" maxlength="200" /></p>
    <p>Choose a Size:<br />
    <input type="radio" name="pizzaSize" value="Small" /> Small<br />
    <input type="radio" name="pizzaSize" value="Medium" /> Medium<br />
    <input type="radio" name="pizzaSize" value="Large" /> Large</p>
    <p>Add Additional Toppings:<br />
    <input type="checkbox" name="pizzaToppings[]" value="Mushrooms" /> Mushrooms<br />
    <input type="checkbox" name="pizzaToppings[]" value="Green Peppers" /> Green Peppers<br />
    <input type="checkbox" name="pizzaToppings[]" value="Black Olives" /> Black Olives<br />
    <input type="checkbox" name="pizzaToppings[]" value="Extra Cheese" /> Extra Cheese<br />
    <input type="checkbox" name="pizzaToppings[]" value="Pepperoni" /> Pepperoni<br />
    <input type="checkbox" name="pizzaToppings[]" value="Sausage" /> Sausage</p>
    <input type="submit" name="pizzaStatus" value="Place Order" />
    </form>
<?php
}
?>
```

if

```
if ($_POST['pizzaStatus'] == "Place Order")
  {
  // Code to be executed
}
```

if-else

```
if ($_POST['pizzaStatus'] == "Place Order")
  {
  // Code to be executed
} else {
  // Code to be executed
}
```

if-elseif-else

```
if ($_POST['pizzaStatus'] == "Place Order")
  {
  // Code to be executed
} elseif ($_POST['pizzaStatus'] ==
  "Continue") {
  // Code to be executed
} else {
  // Code to be executed
}
```

Validation

Besides incomplete submissions, we also always want to avoid malicious submissions.

```
$_POST['custName'] =
  htmlentities($_POST['custName']);

if (strlen($_POST['custName']) < 1) {
  $errorMessages[] = "Please enter your Name.";
}

if(!ctype_alpha($_POST['pizzaSize'])) {
   $errorMessages[] = "Please choose a Size.";
}
```

Validation

```
if (is_array($_POST['pizzaToppings'])) {
  $checkToppings = implode("a",$_POST['pizzaToppings']);

  $checkToppings = str_replace(" ","a", $checkToppings);

  if(!ctype_alpha($checkToppings)) {
       $errorMessages[] = "Please choose some
  Toppings.";
  }
}

if(!ctype_alpha(str_replace(" ","a",
  implode("a",$_POST['pizzaToppings'])))) {
```

Failure Happens

When a validation test fails, make it easy for your user to fix it (Check for malicious submissions, but always treat your users as though it were an accident).

```
if ($_POST['pizzaStatus'] == "Place Order")
   {
   //All our validation tests here
   if(is_array($errorMessages)) {
        echo "<ul>";
        foreach($errorMessages as $message)
   {
                echo "<li>$message</li>";
        }
        echo "</ul>";
        //Form Code Goes Here
   }
   else {
        //Confirmation Code Goes Here
   }
}
else {
   //Form Code Goes Here
}
```

Returning the Form with Data

```
<p>Name: <input type=\"text\" name=\"custName\"
  maxlength=\"200\" value=\"$custName\" /></p>
```

Aside: textarea

```
 <textarea>$data</textarea>
```

Returning the Form with Data

```
<p>Choose a Size:<br />
<input type=\"radio\" name=\"pizzaSize\"
  value=\"Small\" ";
if($pizzaSize == "Small") { echo "checked "; }
echo "/> Small<br />";
```

Returning the Form with Data

```
<p>Add Additional Toppings:<br />
<input type=\"checkbox\"
  name=\"pizzaToppings[]\" value=\"Mushrooms\"
  ";
if(in_array("Mushrooms",$pizzaToppings)) { echo
  "checked "; }
echo "/> Mushrooms<br />";
```

Reducing Redundancy

Create the function:

```
function displayForm($custName, $pizzaSize,
  $pizzaToppings) {
      //Echo Form code Here
}
```

And then use it whenever you need it:

```
else {
  displayForm($_POST['custName'], $_POST['pizzaSize'],
  $_POST['pizzaToppings']);
}
```

Printable receipt

In our script, add this below the confirmation code:

```
$_SESSION['custName'] = $_POST['custName'];
$_SESSION['pizzaSize'] = $_POST['pizzaSize'];
$_SESSION['pizzaToppings'] =
  serialize($_POST['pizzaToppings']);
```

Place this where you want the print link to display:

```
echo "<a href=\"printReceipt.php\">Printable
  Receipt</a>";
```

And place this at the very top of your page (even before the HTML declarations):

```
<?php
  session_start();
?>
```

Script for Printing

```
<?php
   session_start();

   echo "<p>This order is for {$_SESSION['custName']}</p>
   <p>Size: ".$_SESSION['pizzaSize']."</p>
   <p>Toppings:</p><ul>";
   $pizzaToppings =
   unserialize(stripslashes($_SESSION['pizzaToppings']));
   if(is_array($pizzaToppings)) {
       foreach($pizzaToppings as $topping)  {
              echo "<li>$topping</li>";
       }
   }
   echo "</ul>";
?>
```

Got cookies?

Order form code needs to be reorganized so that the validation occurs before any HTML is outputted to the browser. Then add:

```
setcookie("custName",
  $_POST['custName']);
setcookie("pizzaSize",
  $_POST['pizzaSize']);
setcookie("pizzaToppings",
  serialize($_POST['pizzaToppings']);
```

Got cookies?

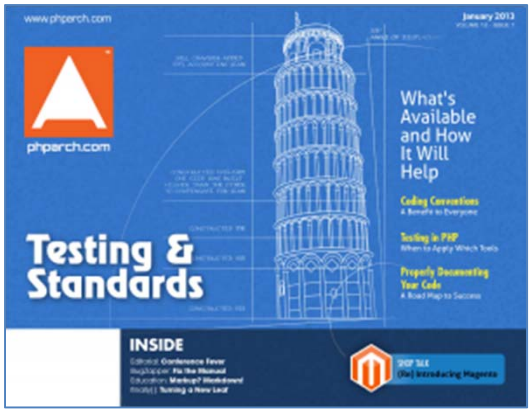
Your print script is updated to:

```
<?php
   echo "<p>This order is for {$_COOKIE['custName']}</p>
   <p>Size: ".$_COOKIE['pizzaSize']."</p>
   <p>Toppings:</p><ul>";
   $pizzaToppings =
   unserialize(stripslashes($_COOKIE['pizzaToppings']));
   if(is_array($pizzaToppings)) {
      foreach($pizzaToppings as $topping)  {
         echo "<li>$topping</li>";
      }
   }
   echo "</ul>";
?>
```

Find Me

- Twitter: e3betht
- Madison PHP User Group (Meetup)
  http://www.madisonphp.com
- Slides Available:
  http://www.TreelineDesign.com/slides

Want more? Take a PHP course! Visit:
www.phparch.com
and click on "TRAINING" for registration info.



php|architect

FREE ISSUES!

Ask me about writing articles for the magazine!

http://www.phparch.com

Feedback or Questions

Joind.in:

https://joind.in/8730

E-mail:

Beth@Musketeers.me