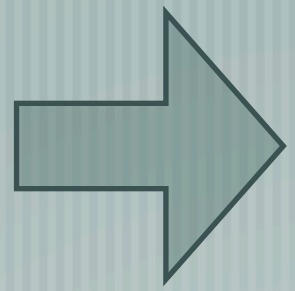
Web Programming with PHP 5

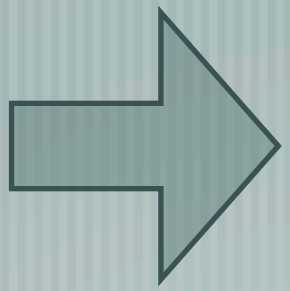
"The right tool for the right job."

PHP as an Acronym

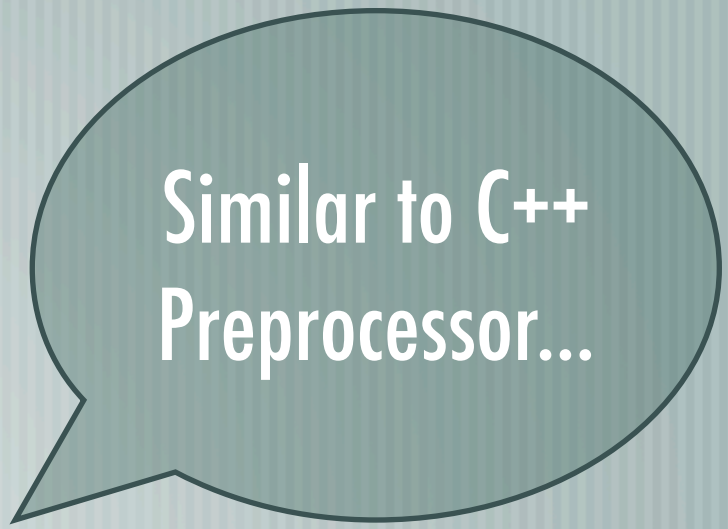
PHP  "PHP: Hypertext Preprocessor"

- This is called a "Recursive Acronym"
  - GNU? "**G**NU's **N**ot **U**nix!"
  - CYGNUS? "**C**YGNUS is **Y**our **GNU** **S**upport"

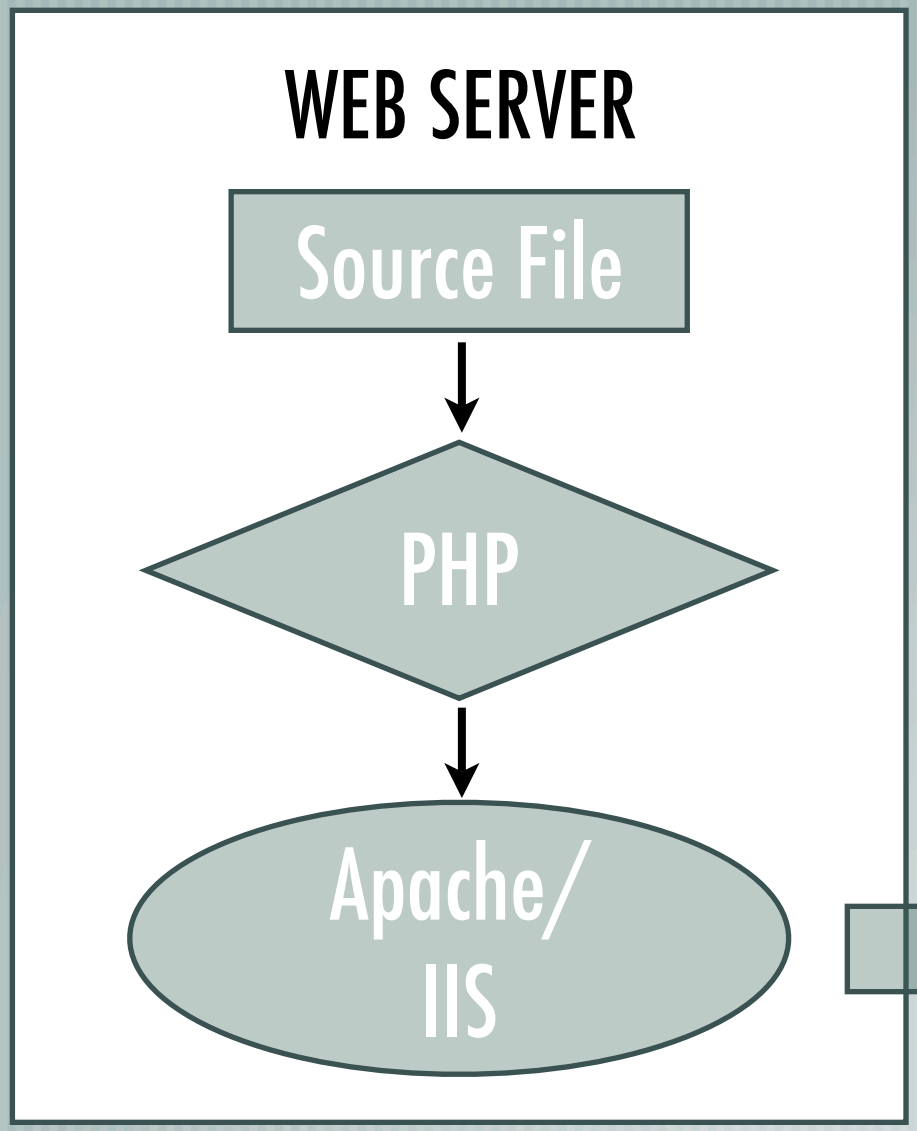
Why the Name Matters

PHP  "PHP: Hypertext Preprocessor"

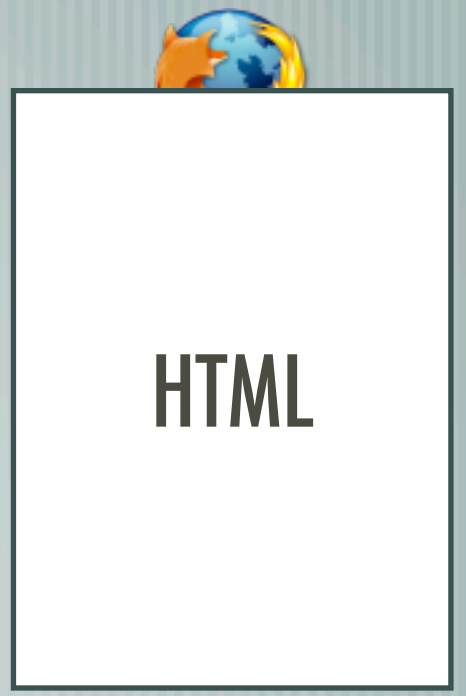
- Hypertext is just HTML
- "Preprocessor" is important for PHP
  - Lexical Substitution
  - Conditions and File Includes



How PHP Works



The purpose of a web server is precisely to send HTML to the browser!



Why do we care?

- 100% Platform-Independent for the user
- No additional style or content-arrangement concerns
- Intelligent application design: How much is done by server?
- Understand: Goofy animations are done with Javascript, Flash, whatever. PHP is for page content.

HTML Syntax: Block Items

- Comments: <!-- … -->
- <head> contains information for the browser
- <body> contains information to display.
- End tags with </tag> or with start tag: <tag />

```
<html>
<head>
    <title>My Awesome Site</title>
</head>

<body>

    <!-- A Picture -->
    <img src="me.jpg" />

    <!-- A Paragraph -->
    <p>My Sexy Photograph</p>

    <!-- A Box -->
    <div style="border-style:
                                dotted">
        <p> Tack in the Box <br />
            Twice!  </p>
    </div>

</body>
</html>
```

Example: html_1.html


```
<html>
<head>
    <title>My Awesome Site</title>
</head>

<body>

    <!-- A Picture -->
    <img src="me.jpg" />

    <!-- A Paragraph -->
    <p>My Sexy Photograph</p>

    <!-- A Box -->
    <div style="border-style:
                                dotted">
        <p> Tack in the Box <br />
            Twice! </p>
    </div>

</body>
</html>
```

HTML Syntax: Tables

- Tables have Rows, Header Cells, and Data Cells
- Good style and closing tags are important for legibility
- In the past, people laid out their web pages with tables: AVOID THIS MIGHTILY!

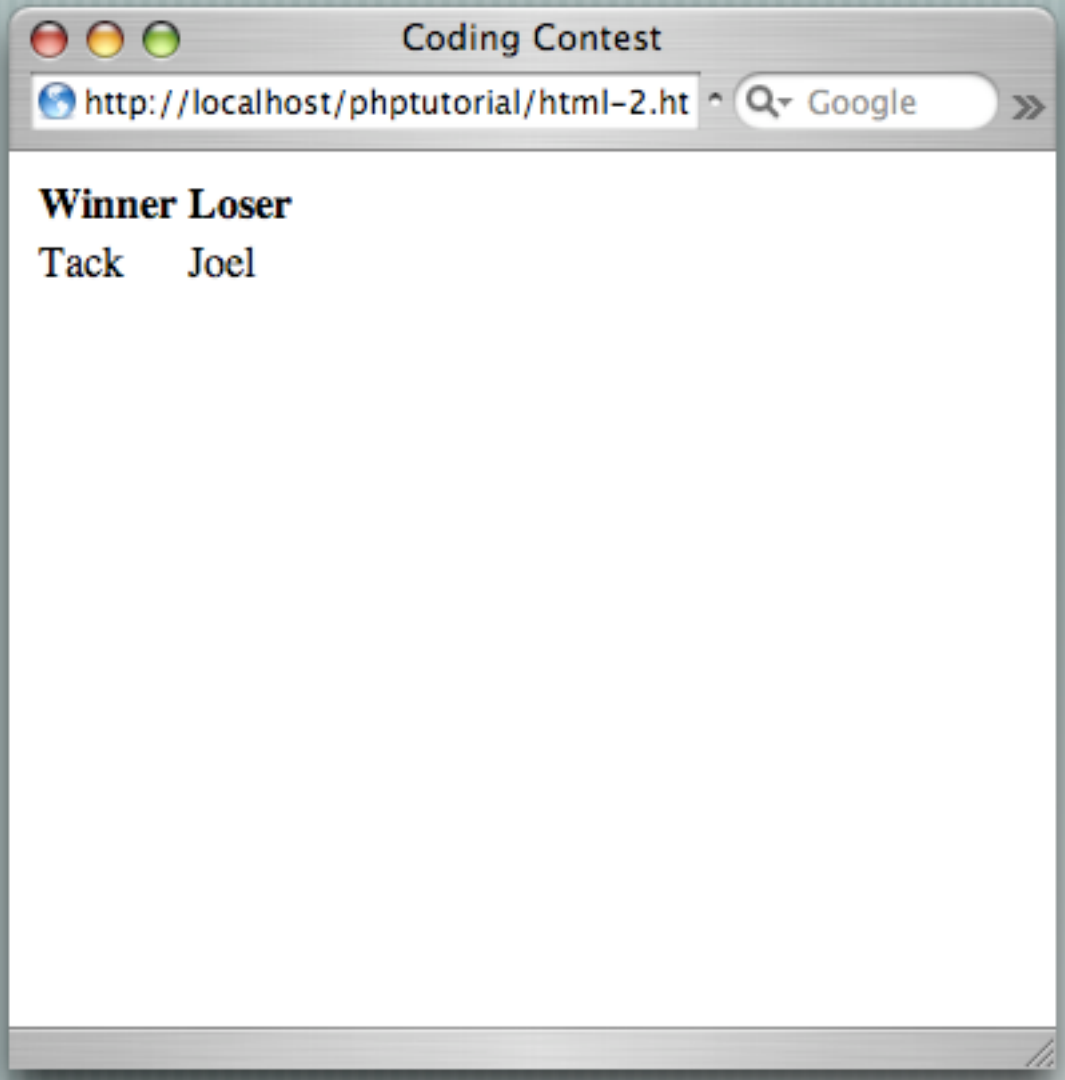
```
<html>
<head>
    <title>Coding Contest</title>
</head>

<body>

    <!-- A Chart -->
    <table>
    <tr>
        <th> Winner </th>
        <th> Loser </th>
    </tr>
    <tr>
        <td> Tack </td>
        <td> Joel </td>
    </tr>
    </table>

</body>
</html>
```

Example: html_2.html




```
<html>
<head>
    <title>Coding Contest</title>
</head>

<body>

    <!-- A Chart -->
    <table>
    <tr>
        <th> Winner </th>
        <th> Loser </th>
    </tr>
    <tr>
        <td> Tack </td>
        <td> Joel </td>
    </tr>
    </table>

</body>
</html>
```

HTML Syntax: Forms

- Forms are used *a lot* in PHP
- Open/close with <form> tags, use <input ... /> tags inbetween.
- Arrange the inputs like regular box items.
- Note the "method" and "action" attributes

```
<html>
<head>
    <title>Forms</title>
</head>

<body>

    <!-- A Form -->
    <form method="post"
          action="forms.html">

    Name: <input name="name"
                 type="text" />
    Good?: <input name="male"
                  type="checkbox" />

    </form>

</body>
</html>
```

Last Word: CSS

- I am not teaching HTML or CSS; it is a pain and it is outside the scope of this talk.
- CSS is not perfect; you will end up blending old stuff with as much CSS as you can.
- Armed with this basic HTML background, we can now introduce some PHP Syntax!

Basic PHP Syntax

- Always start, end with <?php ...stuff... ?>
- Syntax draws heavily from C
- Declare a variable by naming them with a dollar sign.
- All lines end with a semicolon
- Output HTML using "echo"

```
<html>
...
<body>
...

<?php

$name = 'Bob the Fish';
$age = 11;

$dog_age = $age * 7;

echo $name;
echo ' is ';
echo $dog_age;
echo ' in Dog Years.';

?>

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

Variables

```php
<?php

// Create a variable
$myVariable;



// Do things to variables
$augend = 3;
$addend = 5;
$sum = $addend + $augend;
$also_sum = $augend;
$also_sum += $addend;
$zero = 0;
$one = ++$zero;



// Variables can be anything!
$integer = 52;
$floatvalue = 15.62364;
$stringValue = 'Good Night Nurse';
$boolValue = true;

?>
```

- Variables in PHP are Un-typed
- Declare variables by naming them
- All variable names start with a dollar sign
- Standard operations:
  = + - * / % ++ –

```php
<?php

// Example control statements
if ( $meal == 'Steak' )
    echo 'Yum!';
else
    echo 'Blugh!';

switch ( $beverage )
{
    case 'wine':
        $BAC = 0.03;
        break;

    case 'jungle juice':
        $BAC = 0.23;
        echo 'Death!';
        break;
}

for ( $i = 1; $i < 5; ++$i )
    echo $i;


// Special comparison example
if ( 1 === true )
    echo 'Big problems.';

?>
```

Control Statements

- All the regulars reappear:
  if, else if, else, for(), while()
- Comparison operators:
  < > == != >= <= ...
- New comparison: === !==
  (compares value and type)
- Compare strings by dictionary order with strcmp()

Strings

```php
<?php

// A Simple String
echo 'I am a simple string!';



// Concatenation
echo 'A' . ' meets ' . 'B';



// Concatenation and assignment
$string1 = 'dog';
$string2 = 'fleas';
$uglydog = $string1 . $string2;
echo $string1;



// Outputting with tags and
// Interpolating variables
echo '<p> My dog is cute. </p>';
echo 'I said $string1';
echo "I said $string1";
echo "I have two {$string1}s";

?>
```

- Strings are the single most essential core component
- Concatenation Operator: .
- When writing a web script, use PHP to output HTML tags, too.
- Single-Quote Strings are taken 99% Literally
- Double-Quote Strings will interpolate variables automatically

EXAMPLE 1:

Obfuscated PHP Code!

Arrays in PHP

- There are two types of arrays: Numeric and Associative
- Numeric arrays: Indexed by numbers, starting at 0
- Associative arrays: Key-Value pairs
- PHP defines a special for() loop especially for arrays...

Numeric

Index	Value
0	12345
1	'Fish swim good.'
2	[SimpleXML Object]
...	...

Associative

Key	Value
'First Name'	'Hootie'
'Last Name'	'Blowfish'
'Species'	'Salmon'
...	...

Arrays and the foreach() Loop

```php
<?php

// Using Arrays
$meals = array( 'Steak', 'Cat' );
$steak_meal = $meals[0];

$defs = array( 'Steak' => 'Good',
          'Cat' => 'Dry & Bony' );
$review = $defs[$steak_meal];



// The foreach loop
foreach ( $meal as $food )
    echo "I want $food. <br />";

foreach ( $defs as $food => $rev )
    echo "$food is $rev <br />";



// Special Arrays, examples
$_SERVER['PHP_SELF'];
$_GET['firstname'];
$_POST['submit_check'];

?>
```

- Construct arrays with array()
- The foreach() loop iterates through all *defined* members of the array.
- PHP also defines special global arrays, including the ones you see here.
- Remember: Arrays are just a particular type of variable

EXAMPLE 2:

What are $_POST and $_GET

Functions

- Functions in PHP, like in Javascript, are very flexible and easy to write
- Variable scope plays an important role (as always)
- The only tricky thing: default parameters...

```
<?php

function header( $title ) {
    echo <<<HEADER
<html>
<head>
    <title> $title </title>
</head>
HEADER;
}

// Another function
function multiply( $a, $b )   {
    return $a * $b;
}

// Function calls
header('Tack is Dating a Cow');
echo multiply( 127, 23 );

?>
```

That is Basic PHP!

- You know all the core components of syntax.
- To read detail about PHP, http://www.php.net/ is an excellent source; documentation and function reference.
- From now forward, in this talk, we look at some of the cool things PHP can do.
- Something new to PHP 5: Object-Oriented Design

XML: What is it?

- Works like HTML, Syntax-wise
- Purpose: To store information in a predictable fashion, without dealing with databases.
- Pros: Really easy to parse.
- Cons: Size kills speed.

```
<?xml version="1.0" ?>
<staff>

 <person>
  <lastname>Adve</lastname>
  <firstname>Sarita</firstname>
  <title>Associate Professor
    <office>4110 SC</office>
  </title>
 </person>

 <person>
  <lastname>Adve</lastname>
  <firstname>Vikram</firstname>
  <title>Assistant Professor
        </title>
  <office>4235 SC</office>
 </person>

...
</staff>
```

SimpleXML: Really simple.

- These functions come with PHP 5 (One of the big changes)
- Syntax is modeled after C++ pointer-member access
- SimpleXML builds an array of each element; we can iterate!

```php
<?php

// Start by loading the string or
file
$xml = simplexml_load_string
                        ($xmlstring);

// Now we want to echo everybody's
// last name in paragraphs.
foreach ($xml->person as $guy)
{
    echo '<p>';
    echo $guy->lastname;
    echo '</p>\n'
}

// Now say we just want the fifth
// person's last name:
echo '<p> FIFTH: ';
echo $xml->person[4];
echo '</p>\n';

?>
```

MySQL: Really Powerful

- SQL is a language; MySQL is a database.
- SQL instructions are meant to read like sentences.
- Information in “tables”, tables in “databases”

Name	Age	Height
Tack	19	64
Joel	19	34

```php
<?php

// Connect and Select the database
mysql_connect('localhost',
'admin', 'mypassword');

mysql_select_db('awesome_db');


// Now let's grab a name
$name = 'Joel';
$result = mysql_query("SELECT age
FROM 'people' WHERE
name='$name';");

$age = mysql_fetch_assoc($result);

echo $age['age'];


?>
```