Bay Area Video Coalition

Introduction to PHP

Richard Mitchell, Instructor
www.urchard.com/teaching/

Intro to PHP

Setting up Aptana

Go to menu item *Aptana Studio 3 : Preferences :*

1. Set the theme in *Aptana Studio : Themes*—pick one you like.
2. Set these items in *General : Editors: Text Editors*
    a. Set *Displayed tab width:* to 2
    b. Check box *Insert spaces for tabs*

Intro to PHP

Creating a project in Aptana

1. In Finder, create a new directory /Applications/MAMP/htdocs/*php_class_project*
2. From the Aptana Project Explorer view press **command n**.
3. Select the *PHP Project* wizard
4. Type a name for the project, like bavc php
5. Deselect *Use default location*
6. Click *Browse* and navigate to /Applications/MAMP/htdocs/*php_class_project*
7. Click *Open*
8. Click *Finish*

Intro to PHP
Embedding PHP in HTML

"Escaping" PHP:

```
<?xml version="1.0" encoding="UTF-8"?>
<!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>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title>Hello World</title>
  </head>
  <body>
    <p>
      <?php echo 'Hello World!'; ?>
    </p>
  </body>
</html>
```

PHP Comments

Comments are not evaluated—it's like they aren't there.

Single line comment (C++ style)

```
// this is a comment ...
```

Multi-line comment (C style)

```
/* comment starts here
...
and ends here */
```

PHP Documentor

```
/**
 *  a comment for automatic documentation generation
 */
```

Perl style single line

```
# this is a comment too
```

Intro to PHP
Numbers

Decimal	Binary	Hexadecimal
0	0000 0000	00
1	0000 0001	01
2	0000 0010	02
3	0000 0011	03
4	0000 0100	04
5	0000 0101	05
6	0000 0110	06
7	0000 0111	07
8	0000 1000	08
9	0000 1001	09
10	0000 1010	0A
11	0000 1011	0B
12	0000 1100	0C
13	0000 1101	0D
14	0000 1110	0E
15	0000 1111	0F
16	0001 0000	10

Intro to PHP

Data Types—Numbers

Integers

• binary	0b1	0b1101		begins with a zero b
• octal:	01	015	03734	begins with a zero
• decimal:	1	13	2012	
• hexadecimal	0x1	0xD	0x7DC	begins with a zero x

Floating point (floats)

• plain	1200.0	13.5	0.002012
• exponential	1.2e3	1.35e1	2.012e-3

Data Types—Strings

A string is an array of characters (equivalent to bytes). Strings are specified in four ways:

- single quoted—simple, no variable or escape expansion

```
'Hello World\n'
```

- double quoted—variables and escape sequences expanded

```
"Hello World\n"
```

- heredoc—expands like double-quoted string

```
$str = <<<EOD
        Hello World
EOD;
```

- nowdoc—no expansion

```
$str = <<<'EOD'
        Hello World
EOD;
```

Intro to PHP
Data Types—Strings

Escape sequences

• `\n`	newline	0A	10
• `\r`	carriage return	0D	13
• `\t`	tab	09	9
• `\\`	back slash		
• `\$`	dollar sign		
• `\"`	double quote		
• `\[0-7]{1,3}`	octal notation		
• `\x[0-9A-Fa-f]{1,2}`	hexadecimal notation		
• `\v`	vertical tab	0B	11
• `\f`	form feed	0C	12
• `\e`	escape	1B	27

Intro to PHP
Variables

Declaration

- variable names start with dollar sign ($)
- the next character must be a letter or underscore
- following chars may be letters or digits
- variable names are case sensitive

Example assignment of a string to a variable

```
$my_var = "some string";
```

Scope

- the context in which the variable is valid
- global means may be accessed anywhere (must be declared global in functions)

Intro to PHP
Variables—predefined

Super globals

- `$GLOBALS`—all global variables in the script
- `$_SERVER`—info about the server
- `$_GET`—HTML form submissions
- `$_POST`—HTML form submissions
- `$_FILES`—uploaded files
- `$_COOKIE`—cookies on the browser
- `$_SESSION`—server-maintained state between requests
- `$_REQUEST`—combines get, post, and cookie arrays
- `$_ENV`—info about the server environment

Intro to PHP
Data Types—Arrays

Numeric index

- ordered data structure
- numbered index starting at 0
- specify with `array()`

```
$my_array = array('red', 'spaghetti', 'Lost in Space');

echo $my_array[1];  // outputs spaghetti
```

Associative array

- key => value pairs
- specify with `array()`

```
$my_array = array(
  'red' => 'red',
  'food' => 'spaghetti',
  'tv' => 'Lost in Space'
);

echo $my_array['food'];  // outputs spaghetti
```

Intro to PHP

Data Types—Objects and Classes

Objects are a collection of data items, including other objects and functions.

- Properties are variables, which store values.
- Methods are functions, which do things.
- Objects are created as instances of a class.

First create the class with its constructor, properties, and methods.

```
class MyClass {
  // property
  private $prop;

  // constructor
  public function __construct($prop) {
    $this->prop = $prop;
  }

  // method
  public function get_prop()
  {
    return $this->prop;
  }
}
```

Intro to PHP

Data Types—Objects and Classes

Objects are class instances created with the **`new`** keyword and the class constructor

```
$myclass = new MyClass('some value for prop');
```

Object properties and methods are accessed using `->` notation.

```
echo $myclass->get_prop();
```

Intro to PHP
Data Types

Special type `NULL`, variable has no value.

Boolean evaluations

```
TRUE                |       FALSE
1                   |       0
1.0                 |       0.0
'hello'             |       ''
'1'                 |       '0'
array('red')        |       array()
<any true value>    |       NULL
```

Boolean examples

```
var_dump((bool) 1);
var_dump((bool) 0);
var_dump((bool) array('red'));
var_dump((bool) array());
```

Intro to PHP
Constants

Defining a constants:

```
define('LONG_DATE', 'j F Y');
```

Using the constant

```
date(LONG_DATE, strtotime($timestamp));
```

Magic constants, i.e. defined by PHP

```
__LINE__
__FILE__
__FUNCTION__
__CLASS__
```

Intro to PHP
Operators

Arithmetic

```
+ - * / %
```

Assignment

```
= += -= // etc.
```

Increment and decrement—postfix and prefix

```
$i++
--$count_data
```

String concatenation

```
. .=
```

Bitwise

```
& | ^ ~ << >>
```

Intro to PHP
Operators

Comparison

```
== === != !== < <= >= >
```

Logical

```
! && || xor
```

Suppress error message

```
@
```

Intro to PHP
Expressions

An expression has a value.

```
$a = 5;
$b = $a;
echo $b;
$b = $a = 5;
echo $a = 5;
```

Pre- and postfix increment—when the value changes

- prefix: value changes and then is presented (old value discarded)
- postfix: old value is presented then changed

Comparison operators evaluate to `TRUE` or `FALSE`.

The ternary expression—used in assignments.

```
(condition) ? true_value : false_value
```

Intro to PHP

Controlling Program Flow

Statements

- consist of one or more expressions
- end with a semicolon

Blocks (i.e. compound statements)

- one or more statements grouped with curly braces { }

if, else, and elseif.

```
if (condition) {
  statements;
}
elseif (condition) {
  statements;
}
else {
  statements;
}
```

Intro to PHP
Controlling Program Flow

While—the simplest loop

```
while (condition) {
  statements;
}
```

For—more sophisticated looping

```
for (initial_expr; condition; iteration_expr) {
  statements;
}
```

Foreach—two forms for looping through arrays

```
foreach (array_expr as $value) {
  statements;
}

foreach (array_expr as $key => $value) {
  statements;
}
```

Intro to PHP
Controlling Program Flow

Break—getting out early

```
while (condition) {
  if (condition) {
    break;  // stop the loop
  }
  statements;
}
```

Continue—skipping a loop

```
for (initial_expr; condition; iteration_expr) {
  if (condition) {
    continue;  // skip this loop (but do the rest of them)
  }
  statements;
}
```

Intro to PHP
Controlling Program Flow

Switch—a more versatile (and elegant) if-else

```
switch (expression) {
  case value_1:
    statements;
    break;

  case value_2:
    statements;
    break;

  default:
    statements;
    break;
}
```

Including external scripts

```
include('path_to_file');
require('path_to_file');
include_once('path_to_file');
```

Intro to PHP
Functions

A void function

```
function func_name() {
  statements;
}
```

A void function with parameters

```
function func_name(param, param2);
  statements doing something with the parameters;
}
```

A function with parameters returning a value

```
function func_name(param, param2);
  statements doing something with the parameters;
  return TRUE;
}
```

Intro to PHP
Functions

A function parameter may have a default value.

```
function func_name(param = '') {
  statements;
}
```

A function can return any type of value.

```
function func_name(param, param2);
  statements doing something with the parameters;
  return array(TRUE, 3, 5);
}
```

PHP comes with its own functions as well as many libraries of functions and objects. See the documentation.