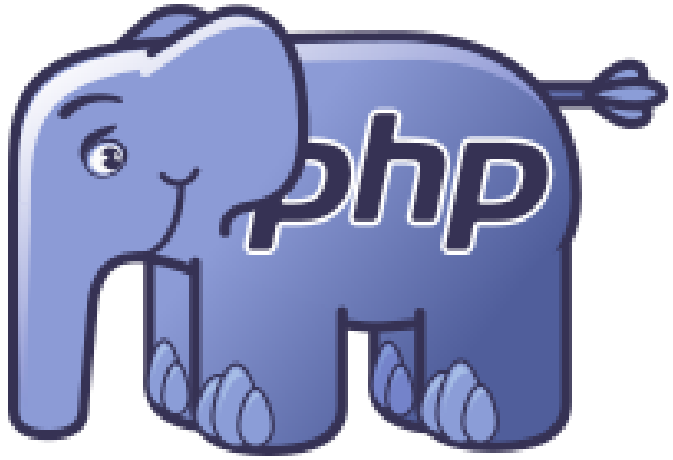
PHP Basics



COMMON STUFF

DATA TYPES AND VARIABLES

EASTERN WASHINGTON UNIVERSITY

How to Print

The `echo` construct is the easiest way to print

Can print quoted strings or variables

```
echo “This is a string\n”;

echo $foo, $bar, $baz;
echo “\n”;
```

Case Sensitivity

Case-insensitive

class names, function names, built-in constructs

(`echo`, `while`, `MyFuNcTiOn()`, `myCLASS`, `mYClaSS`)

Case Sensitive - variables

`$MyName` ≠ `$myname` ≠ `$MyNamE`

statements end with `;`

`{}` contain multiple statements, must have `;` at end of last

Comments

Comments - 3 ways

1. shell-style comments: # - to end of line or PHP section

```
##############################
# This is a comment
##############################
```

2. C++-style comments: // - to end of line or PHP section
3. C-style comments: /* */ - multiple lines of comments

PHP Code

Code surrounded by <?php ?>

```
<?php
  echo "Hello, World"; # hello world example
?>
```

Identifiers

First character must be
- ASCII letter (upper or lower)
- _ underscore
- any character between ASCII 0x7F and 0xFF

after, any of the above plus 0-9

Variables: starts with $, case-sensitive

Function names, Class names - case-Insensitive
- stdClass - reserved class names

Identifiers

Constants
- scalars (Boolean, integer, double, string), arrays can be constants
- once defined, can’t be changed
- use `define()` to set

```
define('TEACHER', "Clay Breshears");
echo TEACHER;
```

Keywords
- reserved for use in PHP
- can't be user-defined identifier

Data Type

8 data types in PHP

(PHP is not a strongly typed language)

4 scalar: integer, floating-point, string, Boolean

2 compound types: array, object

2 special types: resource, NULL

Integers

Range is equivalent to `long` in C

Literals

- Decimal - all digits
- Octal - leading 0, digits 0-7
- Hexadecimal - leading `0x`, digits, A-F
- Binary - `0b`, digits 0,1

All can have `+ -` (example, `-0xFF`)

Use `is_int()` or `is_integer()` to test

EASTERN WASHINGTON UNIVERSITY

Floating-point

Range is equivalent to C `double`

regular : <digits> `.` <digits>

scientific : <digits> `.` <digits>`E` <integer>

Use `is_float()` or `is_real()` to test

String

Arbitrary sequence of characters of arbitrary length

Delimited by single quotes or double quotes

Variables are expanded (interpolated) in DOUBLE QUOTES

```
$name = "Clay";
echo "My name is $name \n";
echo 'My name is $name';
```

Operator to test for equality `==`

Use `is_string()` to test type

String Escape Characters

Double quotes support many string escapes

- Precede with backslash \
- \” \$ \n \t \{ \} \[\]

Single quotes support \\ \’

Boolean

Truth value (keywords: `true` `false`)

ALL the following will be interpreted as `false`:

`false`	`0`	`0.0`	`""` (empty string)
`"0"`	Array with zero elements	`NULL` value	

Use `is_bool()` to test type

Arrays

Holds a group of values

Identified by position (index) or key string

```
$spy[0] = "Tinker";
$spy[1] = "Tailor";
$spy[2] = "Soldier";
$spy[3] = "Poor Man";
$spy[4] = "Beggarman";
```

Associative Arrays

Indexed by string

```
$name["Tinker"] = "Percy Alleline";
$name["Tailor"] = "Bill Haydon";
$name["Soldier"] = "Tony Bland";
$name["Poor Man"] = "Toby Esterhase";
$name["Beggarman"] = "George Smiley";

echo $name["Soldier"];
echo “\n”;
```

Creating Arrays

Assign to specific index (see previous slides)

- Must be done for associative arrays

Append using empty []

```
$spy[] = "Thief";
```

`array()` construct creates an array

```
$spies = array("Tinker", "Butcher", "Baker");
$new_names = array('Tinker' => "Percy",
    'Baker' => "Tom");
```

Array Access

`for`-loop can access indexed arrays

```
for ($i = 0; $i < count($spy); ++$i) {
    echo “$spy[$i]\n”;
}
```

Array Access

`foreach`-loop to iterate over array

```
foreach ($spy as $code_name) {
    echo “Who is $code_name ?\n”;
}
```

key value

```
foreach ($name as $code => $person) {
    echo “$code is designated to $person \n”;
}
```

Some Useful Functions

`print_r()`	see array/object structure
`sort()`	sorts indexed array (inplace)
`asort()`	sorts associative array (inplace)
`is_array()`	tests whether object is array

Objects

Classes are building blocks of OOP design

```
class Person
{
    public $name = '';

    function set_name ($newname = NULL)
    {
        if (!is_null($newname)) {
            $this->name = $newname;
        }
        return $this->name;
    }
}
```

Objects

Once class is defined, objects are created with `new`

Object properties and methods can be accessed with `->` construct

```
$ed = new Person;
$ed->set_name('Edison');
echo "Hello, $ed->name \n";
```

Use `is_object()` to determine if argument is object

Resource

Used to uniquely identify and interact with external resources (e.g., databases)

Resources will have connection/open interface defined

- When there are no more references made to resource, it is automatically shut down
- Thus, no need for `close()` function

Use `is_resource()` to test type

NULL

Keyword (case-insensitive) `NULL`

Represents a variable with no value

Use `is_null()` to test

Variables

Prefixed with a $

Can hold value of ANY type

- Can replace value with another type

No declaration

First time used, allocated in memory

Variable Variables

Reference the value of a variable whose name is stored in another variable by prefacing with $$

```
$bar = 12;
$foo = “bar”;
$$foo = “baz”;
echo "$bar \n";
```

Variable Scope

Scope	Details
Local	Declared in function; local to function only
Global	Declared outside function are global --to access in function, use global keyword
Static	Visible in function, but retains value
Parameter	Local to function where used