LIS-5364

PHP and OOP Basics

PHP

- First weirdness – it's not really run or compiled like most languages. It lives inside HTML...

  More specifically...

The process

Webserver parses/converts the HTML that it knows

If it sees <?php, it passes that to the PHP "converter"

PHP does its thing, and spits text back into the HTML (as opposed to "directly on the page")

Basic PHP

```
<html>
<head>
<title>My First PHP Page</title>
</head>
<body>
<?php
echo "Hello World!";
?>
</body>
</html>
```

Whitespace?

- Like in HTML, mostly ignored
- Hence, the importance of a semicolon (;) at the end of every line.

Variables

- Start with $
- Case sensitive
- Must begin with alpha or underscore
- Must have only alphanumeric or underscore

PHP Data Types

- 3 classes:

  Scalar – One thing

  Compound – Multiple things

  Special - umm, special

Scalar

String – words and stuff

Integer – whole numbers

Float – Decimals

Boolean – true or false

Compound

- Arrays
- Objects

“echo”

Printing words to the screen.

Single Quotes = literal

Double Quotes = will resolve variable (you'll probably use double waaay more)

More echo:

- But, what if you want to actually print quotes and such? Use Backslash for escaped characters, e.g.

  \' = literal single quote

  \”= literal double quote

  \$ = dollar sign

  \n= newline

  \t = tab

Big chunks of text?

Use “heredoc” to resolve variables, and the new “nowdoc” to do literals

A little on comments

PHP supports C and Shell comments, so:

// .. to the end of the line

# to the end of the line

/* to */

Operators

String operators? Really the period, and even then, you'd probably not use it much.

Math? They basically work as expected. Remember, PHP is weakly typed, so with numbers, they're integers until they're not. Be careful.

Strongly typed languages do not do this, so you get, e.g. 7/2 = 3

(ps, remember modulus and what it's good for?)

Arrays

- Basic arrays:

  0 indexed

  Can declare, but don't need to

  $myarray = array(“these”,”are”,”elements”);

  Add to end with $myarray[] =”yo”;

Associative Arrays

Key-Value relationship

Think like “dictionaries”

Remember, they are one-way functions; A value might be the same for multiple keys, but each key has ONE AND ONLY ONE value.

Assign like: array (“word” => “the definition”)

foreach

Do something on each element of the array.

foreach ($thisarrayelement as $thing)

or

- foreach ($currentelement as $key => $value)

Array functions

array_intersect

array_diff

array_merge

array_unique

sort
shuffle

Other Peoples Code?

- Import / Copy from other peoples code stores, e.g. gitlab and github
- Install via a "php installer" page
- Use a package manager like "composer" (like apt, but for PHP)
- Docker or other container images

Object-Oriented?

- Not quick
- Not efficient
- Not "pure"
- Not designed for "computers" but for people:
  Try to map "objects" to the real world

Basic advantages

- Reusable
- Easy to read, understand, and modify
- Clear and real-life oriented
- Mimics the way humans think about things, not as 01011011, but as “dogs bark, cats meow, but they're all furry”
- More work explaining and planning on the front end → less work for addition/maintenance

Abstraction

- Theoretically seamlessly convert real life concepts into computer data objects.

  “Set and forget”

Encapsulation

- Represent and use essential features without necessarily knowing exactly how they work;
- Only allow interactions with "objects," not the data on the machine. (prevents programmers from tampering with things they shouldn't)

Inheritance

Hierarchically store objects to create "taxonomies."

Classes can have "children" that inherit everything from the parent AND add more

Polymorphism

- The ability to implement and use code that calls for an action or a characteristic, and yet does not require that action/characteristic to be the same every time.

  “I don't care HOW, just get it done”

Classes

- A “general” noun
- A “blueprint”
- An “ideal/theoretical form”
- An “object factory”

(to the computer, an invented data type. Like a string or float or array)

Object (to the human)

- A Proper Noun
- An instance of a class
- One of many possible
- A copy of the “Thing”

(to the computer, a specific piece of data, like a “string” (string) or an array [0,5,19])

Properties (or attributes)

- Adjectives applicable to the class/object
- Characteristics or
- Conditions

(to the computer, variables associated with the class/object)

Methods

- Verbs!
- Anything the object can do
- Anything the object can have done to it

  (to the computer, any FUNCTION associated with a class/object)

Now, to make things way more complicated:

When thinking about class - “noun” may be too narrow. Technically, ANYTHING might make a good object. Even something like an action: