

PHP: Functions, regular expressions

Exercise handout
on half wall

INFO/CS 2300:
Intermediate Web Design and
Programming

Enrollment

One round of enrollment pins went out to people changing sections.

Another round today or tomorrow.

I'm hoping to get a bunch of people enrolled in time for Friday.

More next week

Assignments

HW0 graded – will be released soon. Its broken into 2 assignments on CMS. One for clicker registrations and one for everything else.

Project 1 due Tuesday 5 PM

No Frameworks – (No Bootstrap) – Your code

Mini Crash Courses

localhost (setup server on your computer)

- Thursday 2/2 6 PM Gates G01

CSS

Debugging

(See Piazza for more details)

Assignment inside if ()

Sometimes
used when
assigning
function result
to variable then
checking

```
$a = 'cat';
```

```
if ( $a = 'dog' ) {  
    print 'dog';  
} elseif ( $a = 'cat' ) {  
    print 'cat';  
} else {  
    print 'fish';  
}
```

A: dog

B: cat

C: fish

Why is dog the correct answer?

Arrays (continued)

Associative arrays

Can instead use *strings* as the index to arrays

```
$url["Course Information"] = "info.php";
```

```
$url["Forums"] = "forums.php";
```

```
$url["My Account"] = "account.php";
```

The assignment operator is different

Alternative syntax

```
$url = array(
```

```
    "Course Information" => "info.php",
```

```
    "Forums" => "forums.php",
```

```
    "My Account" => "account.php",
```

```
);
```

The trailing comma after the last item is optional but helpful when copying and pasting new rows

Enumerating associative arrays

```
$menu_items = array(  
    'Science' => 'science.php',  
    'Arts' => 'arts.php',  
    'Business' => 'business.php',  
);  
foreach ( $menu_items as $title => $link ) {  
    print( "<li><a href='$link' >$title</a></li>" );  
}
```



A PHP-aware editor will display PHP variables in a different color even inside " "

A closer look

Both \$title and \$link are assigned each time through the loop.

```
foreach ( $menu_items as $title => $link ) {  
    print( "<li><a href='$link' >$title</a></li>" );  
}
```

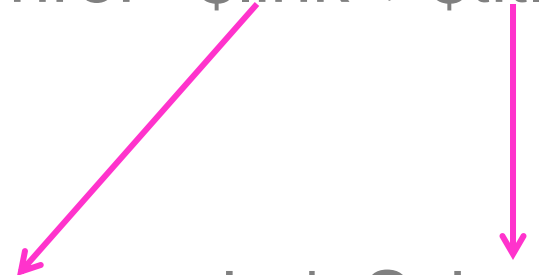
What the PHP processor 'sees' inside the " "

```
print( "<li><a href='$link' >$title</a></li>" );
```

PHP sees the single quote as part of the string to output. HTML doesn't care if it is a single or double quote around the href value

The HTML

```
foreach ($menu_items as $title => $link) {  
    print( "<li><a href='$link' >$title</a></li>" );  
}
```



The diagram consists of two magenta arrows. The first arrow originates from the `$link` variable in the `href` attribute of the PHP code above and points to the string `'science.php'` in the first line of the HTML output below. The second arrow originates from the `$title` variable in the same PHP code and points to the word `Science` in the same line of the HTML output.

```
<li><a href='science.php'>Science</a></li>  
<li><a href='arts.php'>Arts</a></li>  
<li><a href='business.php'>Business</a></li>
```

Why make a menu / list this way?

- Simplifies updates to HTML in the loop
- Easier to find / read / edit the values
- Separates content from HTML markup
- Prepares us for getting content from a database
- Division of responsibility in larger projects

Value increases with complexity

COMPUTING & INFORMATION SCIENCES



Using Facebook when you can't see the faces

Jan. 28, 2016 - Bill Steele - [Bookmark](#)

Visual content on social media sites present challenges to blind users. Cornell researchers suggest that the technology used on Facebook and other social media sites should be adapted to improve accessibility.



Cornell-led team creates gallium nitride power diode

Jan. 19, 2016 - Tom Fleischman - [Bookmark](#)

A team led by Cornell professor Grace Xing has created gallium nitride power diodes capable of serving as building blocks for GaN switches, with many possible power and electronics applications.



Cornell boasts leading cybersecurity research group

Jan. 13, 2016 - - [Bookmark](#)

Four Cornell computer scientists - Ari Juels, Rafael Pass, Thomas Ristenpart and Vitaly Shmatikov - are members of a new cybersecurity, privacy and cryptography reach group at Cornell Tech.



Computing and Information Science receives \$10M grant

Jan. 7, 2016 - - [Bookmark](#)

The National Science Foundation Jan. 7 announced a new \$10 million award to Computer Science Professor Carla Gomes to support transformative computing and technology research.



Robots learn by watching how-to videos

Dec. 18, 2015 - Bill Steele - [Bookmark](#)

Cornell researchers are teaching robots to watch instructional videos and derive a series of step-by-step instructions to perform a task.

A complex form

These arrays aren't actually used in this context. It's the index that is important

```
<?php
    $lypc_pages = array(
        'home' => array(),
        'channel-and-species' => array(),
        'base-pricing' => array(),
        ...,
    ?>
<form method="post" action="#conclusion">
    <?php
        foreach( $lypc_pages as $page_slug => $page_array ) {
            include( "sub-pages/$page_slug.php" );
        }
        include( "sub-pages/dressing-helper.php" );
        include( "sub-pages/conclusion.php" );
    ?>
</form>
```



The Base Pricing page

Base Pricing

Base price:

Pricing Beef Cuts for farmer's market


/ lb HCW Base Price Helper


Base price should be entered in \$/lb. HCW and reflect the price your farm's production operation would need to get if selling that animal outright. [Learn more about base price](#)


Add a premium to the base price:

/ lb HCW

Why add a premium? You are marketing your product in the local foods market and deserve a premium price for your product. Qualities that bring premiums include: breed, diet, certifications (organic, humane, etc.), handling & care, and most importantly, "local." [Learn more about premium pricing](#)

 Previous



Next 

Functions

Function basics

PHP has lots of functions that do lots of things. The basic form:

```
function(arg1, arg2, ... ) {  
}
```

We've already seen some functions:

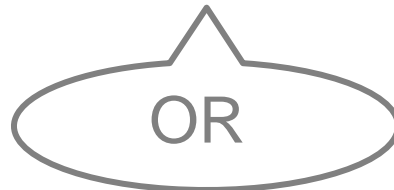
```
print( $name );
```

```
$count = count( $arrayname );
```

Some other useful functions

`isset($variable)`: returns true if the variable has been assigned a value, false otherwise

`empty($variable)`: broader than `isset`.
equivalent to
`! isset($variable) || $variable == false`



form.php

<?php

```
$username = $_POST[ "username" ] ;
```

Beware: Could
cause "undefined
index" error

```
if ( ! empty( $_POST[ "username" ] ) ) {  
    //We'll start doing this more securely next week  
    print( "<p>Welcome, " . $_POST['username'] . "! </p>" );  
} else {  
    ?>
```

```
    <form method="post" action="form.php">  
        <p>What is your name?  
        <input type="text" name="username">  
    </p>  
    <input type="submit" value="Click to submit">  
    </form>
```

<?php

}

?>



Example functions

`mail($to, $subject, $message)`

sends email to email address \$to with
subject \$subject and body \$message

trim

`trim($string)` – returns a string with whitespaces removed from beginning and end

Example

```
$name = ' Spongebob Squarepants ';  
$newname = trim( $name );  
print("$newname$newname");
```

Spongebob SquarepantsSpongebob Squarepants

Strings and arrays

`explode($separator, $string)` – returns an array containing parts of `$string` that were joined by `$separator`.

`implode($glue, $array)` – returns a string containing parts of `$array` joined by `$glue`.

Example: explode / implode

```
$date = "1/28/2015";  
$myarray = explode('/', $date);  
print("Month is $myarray[0],  
      Day is $myarray[1],  
      Year is $myarray[2]");
```



Array('1', '28', '2015')



Month is 1,
Day is 28,
Year is 2015

```
$newdate = implode('-', $myarray);  
print( $newdate );
```



1-28-2015

...and many, many more

Search for what you want such as:
PHP function send mail

Note: the INFO 2300 server is running PHP 5.6.5.



Nearly identical to PHP 7

Click In

Click In

`explode("a", "blah blah blah")`

A. `array("bl", "h bl", "h")`

B. `"bl"`

C. `array("bla", "h bla", "h")`

D. None of the above

Click In

`explode("a", "blah blah blah")`

A. `array("bl", "h bl", "h")`

B. `"bl"`

C. `array("bla", "h bla", "h")`

D. None of the above

Click In

```
preg_replace('/ah /', 'ow ', 'blah blah blah')
```

- A. array("blah", "blah", "blah")
- B. "blow blow blow"
- C. "blow blow blah"
- D. "blow blah blah"
- E. None of the above

Click In

```
preg_replace('/ah /', 'ow ', 'blah blah blah')
```

A. array("blah", "blah", "blah")

B. "blow blow blow"

C. "blow blow blah"

D. "blow blah blah"

E. None of the above

Defining your own functions

Example

Define your own functions for reuse and legibility.

Function name

Argument(s)

```
function makeSubmitButton($name){  
    print( "<input type='submit' value='$name'>" );
```


```
}
```

```
makeSubmitButton("Send message");
```

Returning values

Your functions can return values as well.

```
function increment($x) {  
    $x++;  
    return $x;  
}
```



This is a wrapper.
It adds nothing to
++

```
$y = 0;
```

```
$z = increment($y);
```

```
$z = 1
```


Variable Scope

```
function create_username($netid) {  
    //scope is from outside this function  
    global $course_id;  
  
    //Local variable scope is only inside this function  
    $user_name = $netid . $course_id;  
}
```

Why create your own?

- Simplify your code
- Keep from repeating the same code – simplifies updates

Regular expressions

<http://www.phpro.org/tutorials/Introduction-to-PHP-Regex.html>

Regular Expressions

With `preg_match`, `preg_replace`, and `preg_split`, can actually look for **more complicated patterns** via *regular expressions*.

Regular expressions are patterns expressed via special symbols.

Pattern matching

`preg_match($pattern, $string)` – returns true if the `$pattern` appears in the `$string`

`$pattern` needs to have ‘delimiters’, usually ‘/’.

`preg_match('/geb/', 'Spongebob')`
returns true



Pattern replacing

`preg_replace($pattern, $replacement, $subject)`

returns a string in which all occurrences of
\$pattern in \$subject are replaced by
\$replacement

E.g.

```
preg_replace("/o/", "aw", "Spongebob")
```

returns Spawngebawb

Repeating and grouping

- * -- means zero or more of the preceding "character"
- + -- means one or more of the preceding "character"
- () – treat a group of characters as a unit

Examples

| | |
|---|--------------------|
| <code>preg_match('/a*/', 'SpongeBob')</code> | <code>true</code> |
| <code>preg_match('/ab*/', 'SpongeBob')</code> | <code>false</code> |
| <code>preg_match('/(ab)+/', 'Krusty Krab')</code> | <code>true</code> |
| <code>preg_match('/(ab)*/', 'The Chum Bucket')</code> | <code>true</code> |

Start and end

`^` -- matches when the following
"character" starts the string

`$` -- matches when the preceding
"character" ends the string

| | |
|--|-------|
| <code>preg_match('/^b/', 'SpongeBob')</code> | false |
| <code>preg_match('/b\$/', 'SpongeBob')</code> | true |
| <code>preg_match('/(eb)\$/', 'SpongeBob')</code> | false |

Or

| -- matches if either the preceding or the following "character" matches

```
preg_match('/(on)|(an)/', 'SpongeBob')
```

true

Any and character classes

. – matches any single character

[] – matches any single character inside the brackets (a *character class*)

```
preg_match( 'B.b', 'Bob' )
```

true

```
preg_match( '^[Sp]', 'SpongeBob' )
```

true

```
preg_match( '^[Sp]$', 'SpongeBob' )
```

false

Character class ranges

Character classes are often given by
ranges

[0-9] is shorthand for [0123456789]

[A-Z] matches any uppercase letter

Exercise: Netlingo translator

'brb' => 'be right back'

'cul8r' => 'see you later'

'imho' => 'in my humble opinion'

imho im aatk in 2300



<http://www.phpliveregex.com/>

Click In

If an exercise like this is on the server I

- A. Would still like a paper copy
- B. Don't need a paper copy
- C. Not sure

What about server usernames?

What would we need to test usernames
for the following pattern?
netidsp17

| Tests | result | Comment |
|----------|--------|--------------------|
| sm68sp17 | Yes | |
| smohlke | No | no number, no sp16 |
| sm68SP17 | No | SP is in CAPS |
| | | |
| | | |



| Tests | result | Comment |
|--------------------|--------|------------------------------|
| sm68sp17 | Yes | |
| smohlke | No | no number, no sp16 |
| sm68SP17 | No | Capital SP |
| smohlkesp17 | No | No # in netid, text too long |
| 68sp17 | No | No letters in netid |
| smsp17 | No | No # in netid |
| asasas12121212sp17 | No | Too long netid |
| as12121212sp17 | No | Too many digits netID |
| sm68sp16 | No | last year |
| SM68sp17 | No | Capital Netid |
| asas12sp17 | No | too many letters netid |
| sm68sp17a | No | Letter after sp17 |

$^{\wedge}[a-z]\{2,3\}[0-9]\{1,5\}sp17\$$

Review

- PHP has many useful functions; search "PHP what I want to do"
- Define your own functions.
- Regular Expressions give you powerful pattern matching