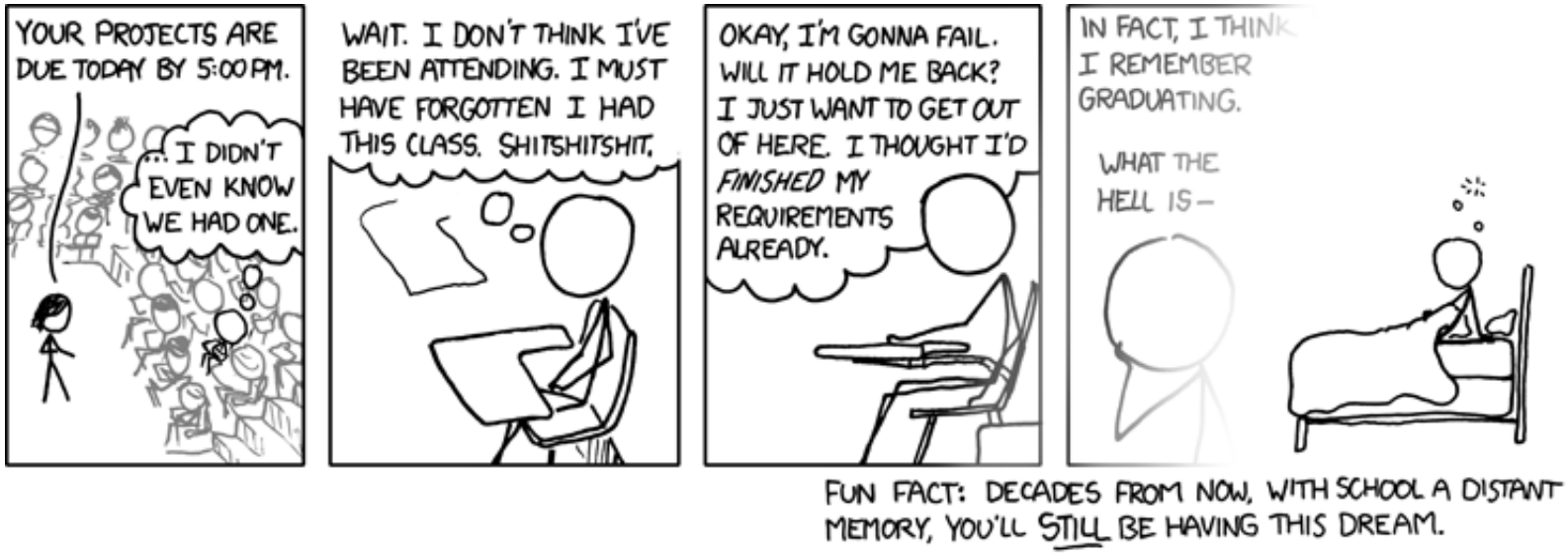


Form checking, classes, and files

INFO/CS 2300:
Intermediate Web Design and Programming

Exercise handout on half wall or on your account on the server

Development Strategy

- Quickest path to success then broaden
- Test on server (environment is different)
  - Operating System differences
  - Path differences

- Keep working copies
  - p1 folder
  - p1_backup
  - p1_backup2

Checking user input

This is a bad idea

```
<?php
   $user_input = $_POST[ 'user_input' ] );
   print( $user_input );
?>
```

Be skeptical

You should *always*, *always*, *always* check user input

Why? Malicious users might be trying to do something

Clueless users might do stuff you don’t expect like enter letters instead of numbers

form.php

```
<?php
    //Try entering these for usernames in different browsers
    //<script>window.open("http://cornell.edu");</script>
    //<p style="font-size: 5em;">steve</p>
    if ( ! empty( $_POST[ "username" ] ) ) {
        $user = $_POST[ 'firstname' ];
        print("<p>Welcome, $user! </p>");
    } else {
?>
    <form method="post">
        <p>What is your name?
            <input type="text" name="username">
        </p>
        <input type="submit" value="Click to submit">
    </form>
<?php
    }
?>
```

HTML Entities
(a partial solution)

```
print( "&lt;p&gt;steve&lt;/p&gt;" );
```

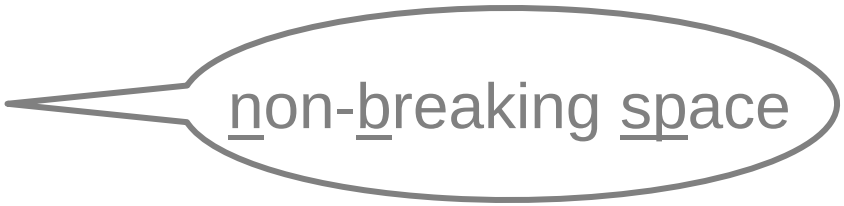
```
<p>steve</p>
```

```
print( "&lt;p&gt;steve&lt;/p&gt;" );
```

```
<p>steve</p>
```

A few HTML Entities

&	&amp;
<	&lt;
>	&gt;
»	&raquo;
¼	&frac14;
©	&copy;



Lots more

http://www.w3schools.com/charsets/ref_html_entities_4.asp

safer-form.php

```
<?php
    //Try entering these for usernames in different browsers
    //<script>window.open("http://cornell.edu");</script>
    //<p style="font-size: 5em;">steve</p>
    if ( ! empty( $_POST[ "firstname" ] ) ) {
        $firstname = $_POST[ 'firstname' ];
        $first_name_html = htmlentities( $firstname );
        print( "<p>Welcome, $first_name_html !</p>" );
    } else {
?>
    <form method="post">
        <p>What is your name?
            <input type="text" name="firstname">
        </p>
        <input type="submit" value="Click to submit">
    </form>
<?php
    }
?>
```

filter_input

```
<?php
        $firstname = $_POST[ 'firstname' ];
        $first_name_html = htmlentities( $firstname );
        print( "<p>Welcome, $first_name_html !</p>" );

    //Alternatively
    $first_name_html = filter_input( INPUT_POST, 'firstname',
                FILTER_SANITIZE_FULL_SPECIAL_CHARS)

?>
```

also INPUT_GET

post variable name

This filter constant is not as thorough as htmlentities

Input Numbers

```
<?php
    $number_input =  filter_input( INPUT_POST, 'number',
                                                FILTER_VALIDATE_INT );

    //Check to see if a number or something else
    if( is_numeric( $number ) ) {
        print( "<p>Your number is $number!</p>" );
    } else {
        print( "<p>You didn't enter a number</p>" );
    }

?>
```

More filter constants

- FILTER_VALIDATE_INT
- FILTER_SANITIZE_NUMBER_INT
- FILTER_VALIDATE_FLOAT
- FILTER_VALIDATE_URL
- FILTER_VALIDATE_EMAIL
- FILTER_SANITIZE_STRING

http://php.net/manual/en/filter.constants.php

Input Dates

Prevent clueless entries by using HTML5 date input, but not for Firefox

<input type="date" name="birthday">

Validating dates is complicated.

This is a good place to start
http://stackoverflow.com/questions/10691949/
http://caniuse.com/#search=date

Being careful

- Use preg_match and / or filter_input to check that input contains only what you want.
- If you want to print out a string the user entered, use htmlentities($input) to make sure all HTML special characters are translated.

Example


```
$new_ride = array( $_POST['coaster'],
    $_POST['ride_count'] );
```


```
$coaster = filter_input( INPUT_POST, 'coaster',
    FILTER_SANITIZE_STRING );
$count = filter_input( INPUT_POST, 'ride_count',
    FILTER_SANITIZE_STRING );
$new_ride = array( $coaster, $count );
```

Click in!

Click in!

The purpose of htmlentities is:

A. To validate user input
B. To safely print text to the screen
C. To format user input as HTML
D. All of the above
E. None of the above

Click in!

The purpose of htmlentities is:

A. To validate user input
B. To safely print text to the screen
C. To format user input as HTML
D. All of the above
E. None of the above

PHP Classes

If we talk about a movie …

We might describe its

- Name
- Year released
- Length
- and many other qualities

Example – Movie class

```
<?php
    class Movie {
        public $title;
        public $year;
        public $length;
    }

    $movie = new Movie();
    $movie->title = 'The Princess Bride';
    $movie->year = '1987';
    $movie->length = 98;

    print( "Have you watched $movie->title?" );
?>
```

Movie class with a function

```
<?php
    class Movie {
        public $title;
        public $year;
        public $length;

        public function the_question() {
            return "Have you watched $this->title?";
        }
    }

    $movie = new Movie();
    $movie->title = 'The Princess Bride';
    $movie->year = '1987';
    $movie->length = 98;

    print( $movie->the_question() );
?>
```

What is ‘$this’?

$this is a special variable used inside method definitions to refer to the current instance of the object.

Using a class more than once

```
<?php
    class Movie {
        public $title;
        public $year;
        public $length;
    }
    $movie_1 = new Movie();
    $movie_1->title = 'The Princess Bride';
    $movie_1->year = '1987';
    $movie_1->length = 98;

    $movie_2 = new Movie();
    $movie_2->title = 'Hidden Figures';
    $movie_2->year = '2016';
    $movie_2->length = 127;

    print( "Have you seen $movie_1->title or $movie_2->title?" );
?>
```

Objects, Instances and Classes

A class is the definition. (Cookie cutter)

Objects are a group of variables and functions that can work easily together. The cookie itself

When we use the object ($movie_1 and $movie_2) we call it an instance of the class (Movie).

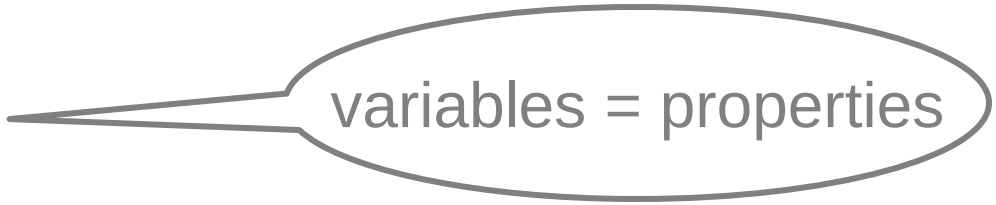
A new kind of variable

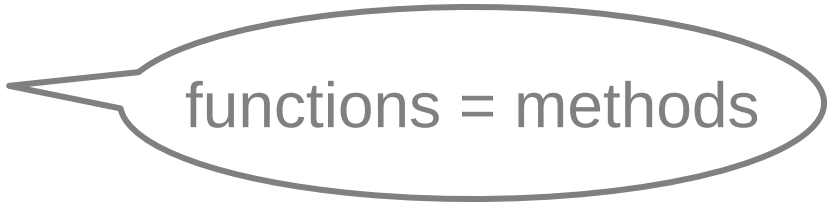
Defining an object defines a new *kind* of variable (e.g. like a string, or integer).

Just like $a = 5 and $b = 10 are both of the integer variable type, $movie_1 and $movie_2 are both of the Movie variable type.

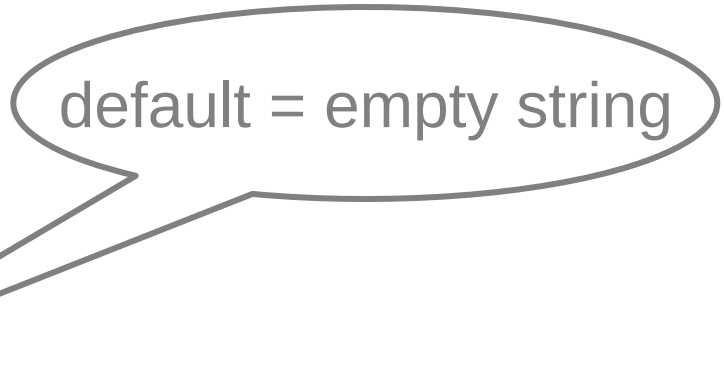
Properties and Methods

```
class ObjectName {
    public variablename1;
    public variablename2;
    private variablename3;
    …

    function function_name() {

    }

    …
}
```





A special method: constructors




```
class Movie {
    public $title;
    public $year;
    public $length;

    function __construct( $title = "", $year = "", $length = null ) {
        $this->title = $title;
        $this->year = $year;
        $this->length = $length;
    }
}

$movie = new Movie( 'The Princess Bride', '1987', 98 );
print( "Have you watched $movie->title?" );
```

OOP

Some programming languages/styles work entirely with objects: said to be *object-oriented* (e.g. Java, C++, Ruby)

OOP = object-oriented programming

Why use objects?

Code organization – it is pretty clear from the file structure where variables and functions for movies belong

Namespace – a generic function name like fix_title won't collide with a function of the same name somewhere else in the code

Easier to pass around groups of variables as parameters

Files

Sometimes you'll want to a website to remember information between visits. One way to do this is to store information in a file.

File functions

file_exists('filename')

Checks whether a file or directory exists

Returns TRUE or FALSE

File functions open/close

fopen(filename, mode)

Opens the file *filename* for reading and/or writing depending on mode

Returns a file pointer if file is opened successfully, or “false” if not

fclose(file pointer)

Clean up when done

fopen modes

$file_pointer = fopen(“rollercoaster.txt”, $mode);

Mode	Read	Write	Overwrite	Create	Pointer
r	X				beginning
r+	X	X			beginning
w		X	X	X	beginning
w+	X	X	X	X	beginning
a		X		X	end
a+	X	X		X	end

more >> http://php.net/manual/en/function.fopen.php

Click In!

Click In!

What mode should be used to log error messages in order as they occur?

A. r+
B. w
C. w+
D. a
E. a+

Mode	Read	Write	Overwrite	Create	Point
r	X				beg
r+	X	X			beg
w		X	X	X	beg
w+	X	X	X	X	beg
a		X		X	end
a+	X	X		X	end

Click In!

What mode should be used to log error messages in order as they occur?

A. r+

B. w

C. w+

D. a

E. a+

Mode	Read	Write	Overwrite	Create	Point
r	X				beg
r+	X	X			beg
w		X	X	X	beg
w+	X	X	X	X	beg
a		X		X	end
a+	X	X		X	end

OK but don't need read

Writing files

fputs($file_pointer, $string)

Writes $string to the file given by filepointer $file_pointer;

returns “false” if there’s an error

Reading files

fgets($file_pointer)

Returns the next line from the file given by filepointer $file_pointer

feof($file_pointer)

Returns true if the end of file has been reached

A quick way to read a file

```
$file_pointer = fopen( 'file.txt', 'r' );
if ( ! $file_pointer ) { print( 'error' ); exit; }
$lines = array();
while( ! feof( $file_pointer ) ) {
    $lines[] = fgets( $file_pointer );
}
fclose( $file_pointer );
```

An even quicker way

```
$myarray = file( “rollercoaster.txt” );
```

Preparing a row for storage

In our PHP program:

$row = array(“Top Thrill”, “Steel”, “Cedar Point”, 8);

In file:

Top Thrill \t Steel \t Cedar Point \t 8 \n

Now you try…

Saving and reading the data

Suppose the data in the table is stored as an array $coasters, where each element is another array, whose first element is the first field for that row (e.g. “El Toro”), second element is the second field (e.g. “Steel”), etc.

Write the code that reads the file “rollercoaster.txt” into an array of coasters.

What about writing the file?

Reading the data file

```
$file_pointer = fopen( 'rollercoaster.txt', 'r' );
$coasters = array();
while( ! feof( $file_pointer ) ) {
    $line = fgets( $file_pointer );
    $coaster = explode( "\t", $line );
    //Strip the "\n" off the end of the value
    $coaster[3] = intval( $coaster[3] );
    $coasters[] = $coaster;
}
```

Writing the data file

```
$coasters = array(
    array( 'Top Thrill Dragster', 'Steel', 'Cedar Point', 8 );
    array( 'El Toro', 'Steel', 'Six Flags Great Adventure', 1);
);
$lines = array();
foreach( $coasters as $coaster ) {
     $line = implode( "\t", $coaster );
     $lines[] = $line;
}
$contents = implode( "\n", $lines );
$file_pointer = fopen( 'rollercoaster.txt', 'w' );
fputs( $file_pointer, $contents );
fclose( $file_pointer );
```

Review

- You must *always, always, always* check your user's input
- Objects are useful for 'packaging' up functions and associated data.
- You can read and write files for storing data between page loads

Reminders…

- Project 1 due Tuesday 5 pm