Bird Challenge

Objectives

Use all of the concepts in the PHP: Object-oriented Programming series by Kevin Skoglund to write an object-oriented program that displays the contents of a CSV file. It is based on the Bicycle Challenge. I recommend that you complete the Bicycle Challenge first as a way to prepare for this assignment.

Set up

- Watch chapters **6: Magic Methods** and **7: A PHP OOP Project**.
- For this project, we only need to show the table of bird data. You can omit the extra files.
- I've provided an image of a bird in Moodle.
- Create a new folder in your **web250** folder named **asgn04-bird-challenge**
- Choose
    - Copy Mr. Skoglund's solution into your **asgn04-bird-challenge** folder OR
    - Build the files from scratch
    - Combine building from scratch and copying his files. You will want to copy his **parseCSV.class.php** and **functions.php** files.
- Rename **bicycle.php** to **bird.php** and make the necessary changes to the file.
- Read **wnc-birds.csv** or visit webtech.tech/web250/asgn04-bird-challenge to see the data and column headings.
- Use git. I recommend creating a branch called **bird-challenge** but this is up to you. The important thing is to use git. Make sure you checkout to the branch.

Coding requirements

Basically, you are required to create a table of bird data from the **wnc-brids.csv** file. This takes some time and most likely require that you watch the videos more than once.

Private and Public Folders

Use the same file configuration with **private** and **public** folders.

Constructors

The **Bird** class must use a constructor (It actually makes it easier).

Consservation

Create **constant** associative array for the conservation levels. The levels are 1 = Low concern, 2 = moderate concern, 3 = High concern, 4 = Extreme concern. You will need a `conservation_id` that contains the number and a method called `conservation_level` that displays the conservation level. That way, if the name of the level changes, you only need to edit the code in one place.

Fortunately for our current list all of the conservation concerns are **low**.

Initialize and Functions files

The author provides two files, **initialize.php** and **functions.php**. These are files he created in an earlier series called, "PHP with MySQL: Essential Training". You have already learned a lot of the contents from this series. We don't want to get too bogged down with **all** of the details at this point, however we need to see how these two files work together.

Read the contents of **initialize.php**. Notice which files have it `included` at the beginning. It is located at the beginning of each page. It prevents you from coding multiple `include_once()` calls on every page.

Watch these videos to see how to use **initialize.php** and **functions.php** together.

Include and Require Files Make Page Assets Reusable

The author also provides a **functions.php** page. We will examine this more carefully later in the class. For now, use the `h()` function to prevent XSS (Cross-Site Scripting). It is a shortcut he has created for the PHP function `htmlspecialchars()`. We will cover this topic in more detail later as well. Watch this video showing why you want to use it.

Encode for HTML

Parsing

The video on putting together the **parseCSV.class.php* file goes deep but you should be able to get most of the concepts.

Add the **parsecsv.class.php** file. Follow his instructions in the Read from a CSV File Chapter so your code will parse the **wnc-birds.csv** file.

array_combine

This function is new to us. Here is a short desciption describing how it works.

array_combine;

Autoload or not

You will need to load your classes one wat or another. Pick any of the three methods he shows.

Work Together!

I found this exercise more difficult than the other exercises. Use the forum to help each other out.

Finished product

A table of bird data uploaded to your web site and the finished code pushed to your GitHub account.

Submit the addresses for each in Moodle.