Foreach Loop

This lesson discusses the foreach loop and its implementation in PHP.

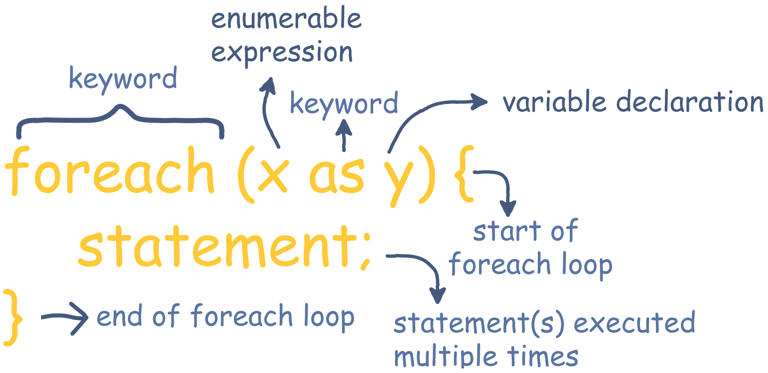
We'll cover the following

Introduction

The `foreach` statement is similar to the `for` statement in that both allow code to *iterate* over the *items* of collections, but the `foreach` *statement* lacks an *iteration* index, so it works even with collections that lack indices altogether.

Syntax

It is written in the following form:



Explanation

- The **enumerable-expression** is the *collection* on which the iteration will happen so it can be an **array** or a **list**
- The **variable-declaration** *declares* a variable that will be set to the successive

elements of the **enumerable-expression** for each pass through the *body*

- The `foreach` *loop* **exits** when there are **no** more elements of the **enumerable-expression** to assign to the variable of the **variable-declaration**

Example

Let’s take a look at an example implementing the `foreach` loop.

```
<?php

$itemsToWrite = array('Alpha', 'Bravo', 'Charlie'); //an array of strings

foreach($itemsToWrite as $item){ //iterating through each element of array itemsToWrite
    echo "$item\n"; //displaying each element of array in console
}

?>
```

Explanation

In the above code:

- the `foreach` statement *iterates* over the elements of the *list* containing *strings* to write **“Alpha”**, **“Bravo”**, and **“Charlie”** to the *console*.

In the next lesson we’ll discuss equivalence of looping structures.