Foreach Loop

This lesson discusses the foreach loop and its implementation in C#

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:

```
foreach("variable declaration "in "enumerable-expression"){
 statements;
}
```

ForEach Syntax

Remarks

- 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.

```
class ForEachExample
{
    static void Main()
    {
      string[] itemsToWrite = {"Alpha", "Bravo", "Charlie"}; //an array of strings

        foreach (string item in itemsToWrite) //iterating through each element of array items
        System.Console.WriteLine(item); //displaying each element of array in console
    }
}
```

ForEach Example

In the above code:

- the `foreach` statement *iterates* over the elements of the *string array* to write **"Alpha"**, **"Bravo"**, and **"Charlie"** to the *console*.

In the next lesson we'll discuss the **jump statements** in **C#**.