The For-Each Loop

In this article from my free Java Course, I will be discussing the for-each loop. The for-each loop is a simplified loop that allows you to iterate on a group of objects like arrays.

Array

The array is an extremely powerful tool that allows you to store multiple objects or primitive data types in one place. You can read more about it in my article dedicated to arrays. For now I'll just give you enough to understand the for-each loop. As an example of an array, imagine an array "persons" that internally holds an entire group of persons. Let's see how we can statically iterate over a `persons` array that can hold ten persons, using a for-loop that I introduced you to in the last article:

```
for(int i=0; i<10; i++ ){
    persons[i].helloWorld();
}
```

Example 1

Each iteration of the loop will increment the variable integer primitive i, that will successively be used to access each element in the persons array.

As you can see, the loop is not easy to read or even to understand. Also, we have to put the size of the array in the loop upfront. As we will see later, there is a better, dynamic way to get the array's size, but that won't make it easier to read.

The For-Each loop

A for-each loop essentially works like a simplified for-loop. A for-each loop uses a simpler, more readable syntax. It does all the dirty work behind the scenes.

```
for(Person person : persons){
    person.helloWorld();
}
```

Example 2

As you can see in Example 2, we don't need to put in size of the `persons` array. The for-each loop will conveniently retrieve it for us. Also, we don't need to index and fiddle around with array cells. For each iteration over the array, it will retrieve the current person object for us, and we can conveniently call the `helloWorld()` method on each `Person` object.

However, keep in mind that the simplicity of the for-each loop comes at a price. There is no such thing as a free lunch! It is not as dynamic as a for-loop, as it always indexes through every spot in the array. With the counter variable "i" missing, you can't look at every other cell, or every third cell, or the first half of the cells. Of course - you *could* add an additional counter variable - but that would render the benefit of the simplified for-each loop useless. Don't even think about it!

The for-each loop is **the perfect loop** to use when you simply need to go through every element of a container. Actually, this is the most common use case for a loop, which makes the for-each loop the most commonly used loop in Java!

Thanks for reading!