For-each loop in Java

Difficulty Level : Easy Last Updated : 03 May, 2021

Prerequisite: Decision making in Java

For-each is another array traversing technique like for loop, while loop, do-while loop introduced in Java5.

- It starts with the keyword **for** like a normal for-loop.
- Instead of declaring and initializing a loop counter variable, you declare a variable that is the same type as the base type of the array, followed by a colon, which is then followed by the array name.
- In the loop body, you can use the loop variable you created rather than using an indexed array element.

- It's commonly used to iterate over an array or a Collections class (eg, ArrayList)

Syntax:

```
for (type var : array)
{
    statements using var;
}
```

is equivalent to:

```
for (int i=0; i<arr.length; i++)
{
    type var = arr[i];
    statements using var;
}
```

```
// Java program to illustrate
// for-each loop
class For_Each
{
    public static void main(String[] arg)
    {
        {
            int[] marks = { 125, 132, 95, 116, 110 };

            int highest_marks = maximum(marks);
            System.out.println("The highest score is " + highest_marks);
        }
    }
    public static int maximum(int[] numbers)
    {
        int maxSoFar = numbers[0];

        // for each loop
        for (int num : numbers)
        {
            if (num > maxSoFar)
            {
                maxSoFar = num;
            }
        }
    return maxSoFar;
    }
}
```

Output:

```
The highest score is 132
```

Limitations of for-each loop

decision-making

1. For-each loops are **not appropriate when you want to modify the array**:

```
for (int num : marks)
{
    // only changes num, not the array element
    num = num*2;
}
```

2. For-each loops **do not keep track of index**. So we can not obtain array index using For-Each loop

```
for (int num : numbers)
{
    if (num == target)
    {
        return ???;   // do not know the index of num
    }
}
```

3. For-each **only iterates forward over the array in single steps**

```
// cannot be converted to a for-each loop
for (int i=numbers.length-1; i>0; i--)
{
      System.out.println(numbers[i]);
}
```

4. For-each **cannot process two decision making statements** at once

```
// cannot be easily converted to a for-each loop
for (int i=0; i<numbers.length; i++)
{
    if (numbers[i] == arr[i])
    { ...
    }
}
```

5. For-each also has some **performance overhead** over simple iteration:

```
/*package whatever //do not write package name here */

import java.io.*;
import java.util.*;

class GFG {
    public static void main (String[] args) {
        List<Integer> list = new ArrayList<>();
        long startTime;
        long endTime;
        for (int i = 0; i < 1000000; i++) {
            list.add(i);
        }
        // Type 1
        startTime = Calendar.getInstance().getTimeInMillis();
        for (int i : list) {
            int a = i;
        }
        endTime = Calendar.getInstance().getTimeInMillis();
        System.out.println("For each loop :: " + (endTime - startTime) + " ms"

        // Type 2
        startTime = Calendar.getInstance().getTimeInMillis();
        for (int j = 0; j < list.size(); j++) {
            int a = list.get(j);
        }
        endTime = Calendar.getInstance().getTimeInMillis();
        System.out.println("Using collection.size() :: " + (endTime - startTim

        // Type 3
        startTime = Calendar.getInstance().getTimeInMillis();
        int size = list.size();
        for (int j = 0; j < size; j++) {
            int a = list.get(j);
        }
        endTime = Calendar.getInstance().getTimeInMillis();
        System.out.println("By calculating collection.size() first :: " + (end

        // Type 4
        startTime = Calendar.getInstance().getTimeInMillis();
        for(int j = list.size()-1; j >= 0; j--) {
            int a = list.get(j);
        }
        endTime = Calendar.getInstance().getTimeInMillis();
        System.out.println("Using [int j = list.size(); j > size ; j--] :: " +
    }
}

// This code is contributed by Ayush Choudhary @gfg(code_ayush)
```

?list=PLqM7alHXFySF5ErEHA1BXgibGg7uqmA4_

Related Articles:

For-each in C++ vs Java

Iterator vs For-each in Java

This article is contributed by **Abhishek Verma**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything i