Цикл for-each в Java

Артем Divertitto

Что такое foreach в Java?

For-each — это разновидность цикла for, которая используется, когда нужно обработать все элементы массива или коллекции. “For each” с английского так и переводится — “для каждого”. Собственно, само словосочетание foreach в этом цикле не используется. Его синтаксис следующий:

```
for (type itVar : array)
{
    Блок операторов;
}
```

Где `type` — тип итерационной переменной (совпадает с типом данных в массиве!), `itVar` — её имя, `array` — массив (тут также может быть другая структура данных, какая-нибудь коллекция, например, `ArrayList`), то есть объект, по которому выполняется цикл. Как вы видите, счётчик в такой конструкции не применяется, итерационная переменная перебирает элементы массива или коллекции, а не значения индекса. При выполнении такого цикла, итерационной переменной последовательно присваивается значение каждого элемента массива или коллекции, после чего выполняется указанный блок операторов (или оператор).

Внимание: цикл for-each можно применить к массивам и любым классам, которые реализуют интерфейс java.lang.Iterable. Эквивалентом коду выше будет следующий цикл **for**:

```
for (int i=0; i<array.length; i++)
{

    Блок операторов;
}
```

Пример использования Java foreach

Создадим массив из оценок ученика. Затем с помощью for-each распечатаем все оценки, выведем среднюю оценку и найдём максимальную из них.

```
public class ForEachTest {

//метод, который распечатывает все оценки
public static void printAllGrades(int[] grades) {
        System.out.print("|");
        for (int num : grades) {

            System.out.print(num + "|");
        }
        System.out.println();
    }

//метод, в котором выводится средняя оценка

public static double midGrade(int[] numbers) {
```

```
        int grade = 0;

        for (int num : numbers) {
            grade = num + grade;
        }
        return ((double) grade / numbers.length);

    }
//метод в котором вычисляется лучшая (максимальная) оценка
    public static int bestGrade(int[] numbers) {
        int maxGrade = numbers[0];

        for (int num : numbers) {
            if (num > maxGrade) {
                maxGrade = num;
            }
        }
        return maxGrade;
    }

public static void main(String[] args) {

//массив оценок
int[] grades = {5, 10, 7, 8, 9, 9, 10, 12};


  int highest_marks = bestGrade(grades);
        System.out.print("All the grades: ");
        printAllGrades(grades);
        System.out.println("The highest grade is " + highest_marks);
        System.out.println("The average grade is " + midGrade(grades));
    }

}
```

Вывод программы:

```
All the grades: |5|10|7|8|9|9|10|12|
The highest grade is 12
The average grade is 8.75
```

А теперь давайте посмотрим, как бы выглядел метод распечатывания всех оценок, выполненный с помощью обычного цикла for:

```
public static void printAllGrades(int[] grades) {
        System.out.print("|");
        for (int i = 0; i < grades.length; i++) {

            System.out.print(grades[i] + "|");
        }
        System.out.println();
    }
```

Если вызвать этот метод из метода `main`, мы получим результат:

```
All the grades: |5|10|7|8|9|9|10|12|
```

Пример использования цикла for-each с коллекциями

Создадим коллекцию из имён и выведем все имена на экран.

```
List<String> names = new ArrayList<>();
```

```
names.add("Snoopy");
names.add("Charlie");
names.add("Linus");
names.add("Shroeder");
names.add("Woodstock");

for(String name : names){
    System.out.println(name);
}
```

Ограничения цикла for-each

Компактная форма for-each считается более читаемой, чем for, и правилом хорошего тона считается использовать именно for-each там, где это можно сделать. Однако for-each — менее универсальная конструкция, чем обычный for. Приведём несколько простых случаев, где воспользоваться for-each не получится вовсе или получится, но с трудом.

1. Если вы хотите пройти по циклу с конца в начало. То есть прямого for-each аналога следующему коду нет:

   ```
   for (int i = array.length - 1; i >= 0; i--)
   {
         System.out.println(array[i]);
   }
   ```

2. For-each не подходит, если вы хотите внести изменения в массив. Например, не получится организовать сортировку массива, не меняя его элементы местами. Или вот, в следующем коде изменится не элемент массива, а только итерационная переменная:

   ```
   for (int itVar : array)
   {
       itVar = itVar++;
   }
   ```

3. Если вы ищете элемент в массиве и вам нужно вернуть (или передать дальше) индекс искомого элемента, лучше воспользоваться обычным циклом for.

Полезное видео о for-each, подготовленное учеником JavaRush

Циклы в курсе JavaRush

На JavaRush к использованию циклов на практике приступают на 4 уровне квеста Java Syntax. Там им посвящено несколько лекций, а также множество задач разного уровня для закрепления навыков работы с ними. Вообще, никуда вы от них не денетесь, циклы — одна из важнейших конструкций в программировании.

Подробнее о for-each и циклах:

1. For и For-Each Loop: сказ о том, как я итерировался, итерировался, да не выитерировался — хорошая подробная статья о циклах for и for-each в Java. Рассчитана на подготовленного читателя (примерно от 10 уровня JavaRush и выше).
2. Оператор While. Статья посвящена самому простому циклу `while`, с которого начинается знакомство с циклами на JavaRush.
3. Хватит писать циклы! Топ-10 лучших методов для работы с коллекциями в Java 8.

Из этой статьи студент JavaRush, который уже прошёл половину курса или больше, узнает много интересного о работе с коллекциями.

Артем Divertitto

Senior Android-разработчик в United Tech

Артем — программист-свитчер, его первая профессия — реабилитолог. Раньше он помогал людям восстанавливать здоровье, даже получил м ... [Читать полную биографию]