Perulangan

Sebagai *Developer* kita dituntut untuk membuat baris kode menjadi lebih efisien. Misalnya kita melakukan instruksi yang sama lebih dari sekali, seperti contoh di bawah ini:

```
1. System.out.print("Halo\n");
2. System.out.print("Halo\n");
3. System.out.print("Halo\n");
```

Output :

```
1. Halo
2. Halo
3. Halo
```

Baris kode di atas kita melakukan perintah yang sama sebanyak 3 kali dengan menulis kode yang sama. Hal ini sama ilustrasinya dengan mengirimkan pesan melalui email dengan alamat yang berbeda-beda. Tidak efisien, bukan? Memang kurang efisien kalau kita membuat baris kode dengan mengirimkan secara satu per satu. Dalam kasus seperti ini kita tidak harus mengulang baris kode yang sama secara terus-menerus. Lebih tepat, gunakan metode perulangan.

Perulangan For

Perulangan memungkinkan kita melakukan instruksi secara berulang-ulang secara sederhana, jelas dan ringkas. Ada berbagai cara melakukan perulangan, namun yang paling populer adalah menggunakan for. Secara umum, bentuk pernyataan for di dalam Java adalah sebagai berikut:

```
1. for (initialization; termination; increment) {
2.     statement(s)
3. }
```

Perhatikan contoh kode berikut:

```
1. System.out.print("Halo\n");
2. System.out.print("Halo\n");
3. System.out.print("Halo\n");
```

Dengan menggunakan perulangan, kita bisa menghasilkan output yang sama dari baris kode di atas dengan menggunakan salah satu metode perulangan:

```java
for(int x = 0; x < 3; x++) {
    System.out.print("Halo\n");
}
```

Output:

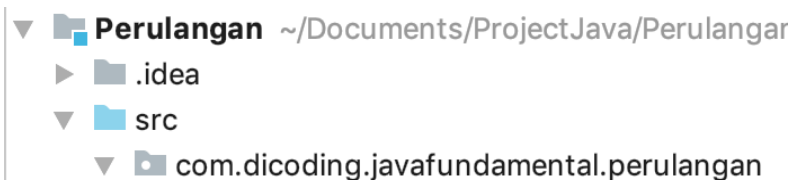
```
Halo
Halo
Halo
```

Sama *output*-nya, namun caranya terlihat lebih ringkas, bukan ?

Codelab Perulangan For

Selanjutnya kita coba melakukan perulangan dengan menuliskan angka dari 1 hingga 10 dengan menggunakan for.

1. Buatlah proyek baru dengan nama Perulangan dengan nama package `com.dicoding.javafundamental.perulangan` di dalamnya:



2. Buatlah sebuah kelas baru dengan nama `For`, kemudian tambahkan kode berikut:

```java
package com.dicoding.javafundamental.perulangan;

public class For {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            System.out.println("Angka : " + i);
        }
    }
}
```

3. Selanjutnya jalankanlah kode di atas pada IDE yang kalian gunakan. Bila sukses, seharusnya Console akan menampilkan output seperti ini.

```
Angka : 1
Angka : 2
Angka : 3
Angka : 4
Angka : 5
Angka : 6
Angka : 7
Angka : 8
Angka : 9
Angka : 10
```

Bedah Code Perulangan For

Kita membuat *variabel* dengan tipe data *integer* yang kita sebut sebagai *initialization*. Dalam hal ini kita menginisialisasi angka dimulai dari 1.

```
1.  int i=1;
```

Pada tahap ini kita menentukan batasan nilai akhir suatu perulangan atau disebut dengan termination. Contoh di atas adalah angka lebih kecil atau sama dengan 10.

```
1.  i<=10;
```

Kita menentukan aksi terhadap perulangan, aksi tersebut bisa menaikkan (*increment*) atau menurunkan (*decrement*). Apabila kita ingin melakukan aksi *increment* maka nilai awal (*initialization*) harus lebih kecil daripada nilai akhir (*termination*). Begitupun sebaliknya, jika melakukan aksi menurun maka nilai awal (*initialization*) harus lebih besar daripada nilai akhir (*termination*). Seperti contoh di atas kita melakukan aksi *increment*.

```
1.  i++;
```

Nah, apabila ketiga kondisi di atas sudah terpenuhi, pengulangan dapat tercipta. Bagaimana dengan varian implementasi dengan memanfaatkan for?

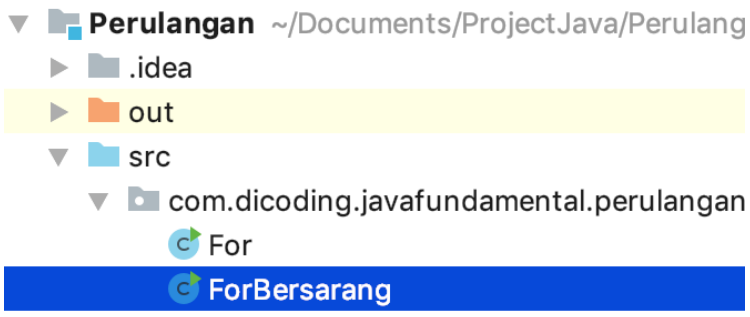
Perulangan For Bersarang

For dapat menerapkan pengulangan bersarang (nested). Dengan kata lain kita membuat melakukan for di dalam for.

Codelab Perulangan For Bersarang

Sebagai contoh kita akan membuat segitiga bintang legendaris dengan memanfaatkan for bersarang.

1. Bukalah kembali proyek Perulangan dan buatlah kelas baru dengan nama `ForBersarang`.

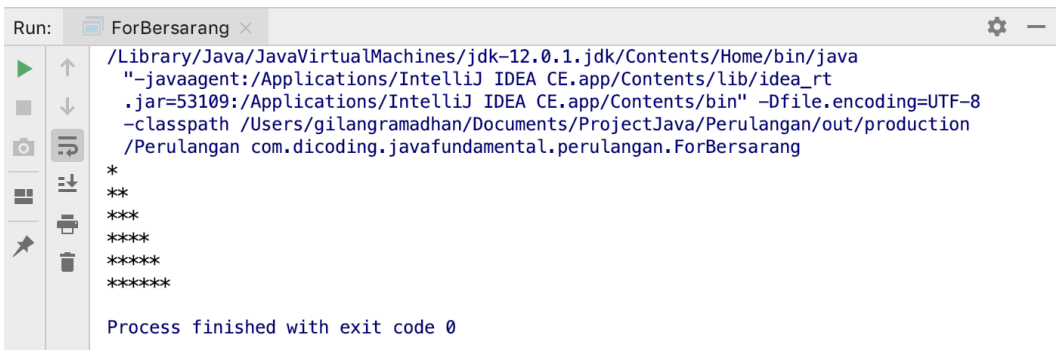


2. Masukkan kode berikut ke dalam kelas `ForBersarang`:

```
1.  public class ForBersarang {
2.      public static void main(String[] args) {
3.          int a = 5;
4.          for (int i = 0; i <= a; i++) {
5.              for (int j = 0; j <= i; j++) {
6.                  System.out.print("*");
7.              }
8.              System.out.println("");
9.          }
10.     }
11. }
```

3. Jalankan kode di atas maka hasilnya akan jadi seperti ini:

```
*

**

***

****

*****

******
```



Bedah Code Perulangan For Bersarang

Perhatikan kode berikut ini:

```
1.  for (int i = 0; i <= a; i++) {
2.      for (int j = 0; j <= i; j++) {
3.          System.out.print("*");
4.      }
5.      System.out.println("");
6.  }
```

Inilah yang disebut dengan perulangan for bersarang. Perulangan ke pertama akan mempengaruhi perulangan kedua, begitu juga ketika ada perulangan ketiga, keempat dan seterusnya, maka akan dipengaruhi oleh perulangan sebelumnya.

Perulangan While

Perulangan dengan menggunakan while digunakan jika proses saat kondisi terpenuhi. While akan mengecek kesesuaian dengan kondisi pra perintah.

Codelab Perulangan While

1. Bukalah kembali proyek Perulangan dan buatlah kelas baru dengan nama `While`.

2. Masukkan kode berikut ke dalam kelas `While`:

```
1.  package com.dicoding.javafundamental.perulangan;
2.
3.  public class While {
4.      public static void main(String[] args) {
5.          int value = 1;
6.          while (value <= 10) {
7.              System.out.print("Angka : " + value);
8.              value++;
9.              System.out.print("\n");
10.         }
11.     }
12. }
```

3. Jalankan kode di atas maka hasilnya akan jadi seperti ini:

```
Angka : 1
Angka : 2
Angka : 3
Angka : 4
Angka : 5
Angka : 6
Angka : 7
Angka : 8
Angka : 9
Angka : 10
```

Bedah Code Perulangan While

Kita mendeklarasikan variabel nilai dengan angka 1.

```
1.  int value = 1;
```

Pengecekan suatu kondisi pada variabel. Apabila nilai lebih kecil atau sama dengan 10.

```
1.  while (value <= 10)
```

Melakukan sebuah perintah apabila kondisi terpenuhi.

```
1.  System.out.print("Angka : " + value);
2.  value++;
3.  System.out.print("\n");
```

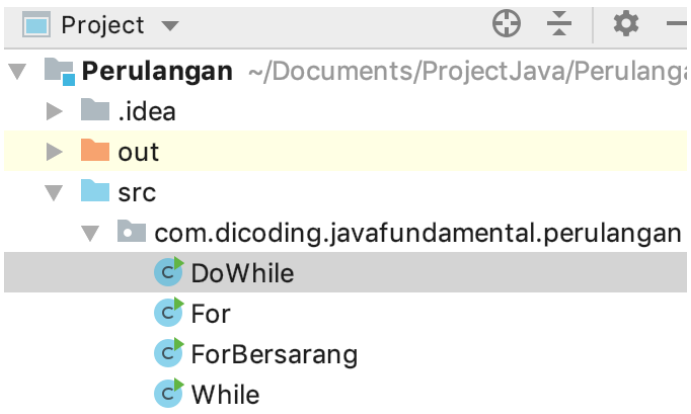
Perulangan Do-While

Perulangan yang mempunyai fungsi yang sama dengan While, tetapi pengecekan kondisinya dilakukan di akhir.

Pada perulangan ini minimal melaksanakan perintah sekali, kemudian mengecek kondisi.

Codelab Perulangan Do-While

1. Bukalah kembali proyek Perulangan dan buatlah kelas baru dengan nama `DoWhile`.



2. Masukkan kode berikut ke dalam kelas `DoWhile`:

```
1.  package com.dicoding.javafundamental.perulangan;
2.
3.  public class DoWhile {
4.      public static void main(String[] args) {
5.          int value = 1;
6.          do {
7.              System.out.println("Angka : " + value);
8.              value++;
9.          } while (value <= 10);
10.     }
11. }
```

3. Jalankan kode di atas maka hasilnya akan jadi seperti ini:

```
Angka : 1
Angka : 2
Angka : 3
Angka : 4
Angka : 5
Angka : 6
Angka : 7
Angka : 8
Angka : 9
Angka : 10
```

Bedah Code Perulangan Do-While

Bagaimana bisa menghasilkan output seperti di atas? Mari kita bahas bersama-sama:

1. Kita mendeklarasikan variabel nilai dengan angka 1.

```
1.  int value = 1;
```

2. Melakukan sebuah perintah terlebih dahulu.

```
1.  System.out.println("Angka : " + value);
2.  value++;
```

3. Pengecekan suatu kondisi terhadap variabel. Apabila terpenuhi maka perintah dilanjutkan.

```
1.  while (value <= 10)
```

Nah, kita sudah mencoba macam-macam perulangan pada Java. Penggunaan metode perulangan tergantung pada kasus dan mengaplikasiannya. Bisa jadi salah satu metode, paling cocok diimplementasikan pada kasus tertentu. Metode mana yang paling kamu suka?

Silakan unduh latihan pada modul ini di Source Code Perulangan.

Sebelumnya

Selanjutnya

© 2021 Dicoding | Dicoding adalah merek milik PT Presentologics, perusahaan induk dari PT Dicoding Akademi Indonesia.

Terms • Privacy