Tutorial Link https://course.testpad.chitkara.edu.in/tutorials/Blocks in Java Class/5b1e44d37becc0459deadac6

TUTORIAL

Blocks in Java Class

Topics

Initializer blocks

Initializer block contains the code that is always executed whenever an instance is created. These blocks are executed before constructors. For example,

Java

```java
class Main
{
  // Initialization Block
  {
    System.out.println("Code Quotient");
  }

  // Constructor of Main class
  Main()
  {
    System.out.println("Constructor Called");
  }

  public static void main(String[] args)
  {
    Main ob1 = new Main();
  }
}
```

```
19
```

You can have as much number of blocks as you want in your class. If compiler finds multiple initializer blocks, then they all are executed from top to bottom i.e. the initializer blocks which is written at top will be executed first. For example,

Java

```
class Main
{
  // Initialization Block  1
  {
    System.out.println("Code Quotient-1");
  }

  // Initialization Block  2
  {
    System.out.println("Code Quotient-2");
  }

  // Constructor of Main class
  Main()
  {
    System.out.println("Constructor Called");
  }

  // Initialization Block  3
  {
    System.out.println("Code Quotient-3");
  }

  public static void main(String[] args)
  {
    Main ob1 = new Main();
  }

  // Initialization Block  4
  {
    System.out.println("Code Quotient-4");
  }
```

```
}

```

Initializer block with parent class

In case of inheritance, these blocks are executed in same manner as other parts i.e. first preference will be given to parent class then derived class. For example,

Java

```
class BaseClass
{
    // Initialization Block
    {
        System.out.println("Code Quotient-Base");
    }

        // Constructor of BaseClass class
    BaseClass ()
    {
        System.out.println("Constructor of BaseClass Called");
    }
}

class Main extends BaseClass
{
    // Initialization Block
    {
        System.out.println("Code Quotient-Derived");
    }

        // Constructor of Main class
    Main()
    {
        System.out.println("Constructor Called-Main Class");
    }

        public static void main(String[] args)
    {
```

```
        Main ob1 = new Main();
    }
}

```

Static blocks

Java supports a special block, called static block (also called static clause) which can be used for static initializations of a class. Unlike other blocks, the code inside static block is executed only once i.e. the first time you make an object of that class or the first time you access a static member of that class (even if you never make an object of that class). For example,

Java

```
class Main
{
  // Initialization Block
  {
    System.out.println("Code Quotient-Non-Static");
  }

  // Constructor of Main class
  Main()
  {
    System.out.println("Constructor Called");
  }

  // static Initialization Block
  static {
    System.out.println("Code Quotient- Static");
  }

  public static void main(String[] args)
  {
    Main ob1 = new Main();
    Main ob2 = new Main();
    Main ob3 = new Main();
  }
}
```

```
26
```

You can notice that static blocks are executed before class constructors. They are generally used to initialize the static members of a class.

Java

```
class Main
{
  static int i;

  // Initialization Block
  {
    System.out.println("Code Quotient-Non-Static");
  }

  // Constructor of Main class
  Main()
  {
    System.out.println("Constructor Called");
  }

  // static Initialization Block
  static {
    i=10;
    System.out.println("Code Quotient- Static");
  }

  public static void main(String[] args)
  {
    Main ob1 = new Main();
    Main ob2 = new Main();
    Main ob3 = new Main();
    System.out.println("i in main = " + Main.i);
  }
}

```