Java Scope

In Java, variables are only accessible inside the region they are created. This is called **scope**.

Method Scope

Variables declared directly inside a method are available anywhere in the method following the line of code in which they were declared:

Example

```
public class Main {

  public static void main(String[] args) {

    // Code here CANNOT use x

    int x = 100;

    // Code here can use x

    System.out.println(x);

  }

}
```

Block Scope

A block of code refers to all of the code between curly braces `{}`.

Variables declared inside blocks of code are only accessible by the code between the curly braces, which follows the line in which the variable was declared:

Example

```
public class Main {

  public static void main(String[] args) {

    // Code here CANNOT use x

    { // This is a block

      // Code here CANNOT use x

      int x = 100;

      // Code here CAN use x

      System.out.println(x);

    } // The block ends here

  // Code here CANNOT use x

  }

}
```

Anything which is initialised inside the method can use anywhere in the method we can use it inside the blocks we can update it and the original value get updated. But we cannot initialised inside the blocks

Anything which is initialised inside the blocks is available for that block only we cannot use it outside the block but we can initialised outside the blocks.

```java
public class Blocks {

    public static void main(String[] args) {
        //method
        int a = 10;
        int b = 20;

        //blocks
        {
//            int a = 100;            //  we cannot initialised inside the
blocks
            a = 120;                  // we can update inside the blocks
            System.out.println(a);    // we can use inside the blocks



            int c = 30;
            System.out.println(c);
        }

        System.out.println(a);


        int c = 40;                   //we can initialised outside the
blocks
        System.out.println(c);
    }
}
```