**Java Tutorial:**

**Naming in java:**

1. Project name - upper camel case ex: MyProject
2. Class name - upper camel case
3. Method name - lower camel case ex: getMyName()
4. Variable name - lower camel case

**Java has 6 relational operators.**

**==** is the equality operator. This returns true if **both the operands** are referring to the **same object**, otherwise false.

**!=** is for non-equality operator. It returns true **if both the operands** are referring to the **different objects**, otherwise false.

**<** is less than operator.

**>** is greater than operator.

**<=** is less than or equal to operator.

**>=** is greater than or equal to operator.

**Code Block in Java:**

a) if-then:

if (expression) {

----block of code to be executed

}

If you not specify code block then 2nd statement will be executed in java

**Logical Operator in java:**

&& - both should be true

**Note**:

Assignment operator – are used to assign a value to a variable

Relational operator – are used to check if a given variable is equals or not

**Ternary Operator:**

**Syntax:** condition? True: False

Ex:

int ageOfClient = 18;  
String ageTxt = (ageOfClient == 18) ? "Well Matured":"Still Kid";  
System.*out*.println("He is a : " +ageTxt);  
  
String myCar = "BMW";  
boolean isDomestic =(myCar == "BMW")? true : false;  
System.*out*.println("my car name is :" +myCar);

**Note:**

True or false are not keywords in java .These are Boolean literals.

Null is also not a keyword in java.

**Keywords in Java:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| abstract | continue | for | new | switch |
| assert | Do | if | package | try |
| Boolean | default | int | private | this |
| break | double | import | protected | throw |
| byte | else | interface | public | throws |
| case | enum | instanceof | return | transient |
| catch | extends | implements | Goto | while |
| char | final | super | static | void |
| class | finally | short | strictfp | volatile |
| const | float | long | synchronized | native |

**Note**: \_ is also a keyword in java 9

**Expression in Java:** arethe values in java, where values are inside the block **(;, if** these are not expression in java**)**

**Whitespaces**

**Indentation**

**Code – reformat code**

**Code Blocks:**

**If- else**

**If- else if -else**

**If- then-else**

**Methods in Java:**

A method is a way of avoiding code duplication.

A method can be executed multiple times by passing a parameter multiple times.

Methods in Java give us a kind of way to write code once, and then reuse that code anywhere in our program

>**Parameters** - are the data type and variable passed to the methods without declaring the variables.

>**Arguments** – are the values passed to the method call.

In a method if we use void that doesn't return anything, in other words a method declared with void. As the return type, a return statement is not required. But in methods that do return data, a return statement with a value is required. So we've defined a method, and said it was going to return an int

**Parameters** are declared as a list of comma-separated specifiers, each of which has a parameter type and a parameter name (or identifier).

>The calling code must pass **arguments** to the method, with the same or comparable type, and in the same order, as the declaration. The calling code must pass the same number of arguments, as the number of parameters declared.