Java Learning

**General:**

Java code is case sensitive. This includes not only keywords and language syntax, but variable names and data types as well.

**JShell**:

It is used in windows to run Java programs.

Go to command prompt-> Jshell-> write programs.

/list is used to print all the command we executed in current session of jshell.

/var is used to list all the variables created in current session.

In jshell, We can declare a variable again with data type and same name but in a java program, We are not allowed to do that.

**Keywords in java**

A keyword is any one of a number of reserved words, that have a predefined meaning in the java language.

Link to all the keywords in java: <https://docs.oracle.com/javase/specs/jls/se17/html/jls-3.html#jls-3.9>

**Variables in Java:**

Variables are a way to store information in out program. It is stored in RAM.

**Expression in Java:**

Expression is the code segment, that is on the right side of the equals sign in an assignment or declaration statement.

**Primitive Data Types in Java:**

1. Number: byte, short, int, long: Here every type has a min and max limit for type.
2. Real numbers: float, double
3. Single character: char
4. Boolean value: Boolean

**Wrapper Classes for data types:**

Java used the concept of a wrapper class, for all of its eight primitive data types.

A wrapper class provides simple operations, as well as some basic information about the primitive data type, which cannot be stored on the primitive itself.

Here are primitive type and wrapper classes:

1. byte: Byte: 8 bits
2. short: Short: 16 bits
3. char: Character
4. int: Integer: 32 bits
5. long: Long: 64 bits
6. float: Float
7. double: Double
8. boolean: Boolean

If we try to put a value larger than specified range, it will be overflow. Same if we try to put a value lesser than the specified range, it will be underflow.

**Casting in Java:**

Casting means to treat or convert a number, from one type to another. We put the type we want the number to be, in paratheses;

Normal casting returns, an integer so we need to convert in into byte.

byte myMinByteValue = Byte.*MIN\_VALUE*;  
byte myMinByteValueHalf = (byte) (myMinByteValue/2);

**Float and Double primitives:**

We use a floating point number when we need more precision in calculation.

There are two primitive types in java for expressing floating point numbers, the float and double.

Float: width: 32 bits

Double: width: 64 bits

Double is more precise than float.

We must write f at the end of a float number and d at the end of a double number.

Ex: float x = 2.3f; double d = 2.4d; float num = 5.34, this line will give error because we must either specify f at the end of use casting.

Double is default data type for decimal literals.

**Char in Java:**

Holds one and only one character, Literal enclosed in single quotes.

Wrapper for char in Character.

A char occupies 16 bits in memory because it is stored in memory as a 2 byte number.

We can get unicodes for all the letter from this website: <https://symbl.cc/en/>

// There are total 3 ways to assign a value to a variable.  
char myChar = 'D';  
char myUnicodeChar = '\u0044';  
char myIntegerChar = 68;

**Boolean in Java:**

A Boolean value allows for two opposite choices, true or false, yes or no, one or zero.

The wrapper class for boolean is Boolean.

Ex:

boolean myTrueBooleanValue = true;  
boolean myFalseBooleanValue = false;  
boolean isCustomerAgeGreaterThanTwentyOne = true;

**String in Java:**

A String is a class that contains a sequence of characters. String is a class not primitive type.

String are Immuatable, means we can’t change a string after it is created. If we need to change over string frequently, than we should use either StringBuilder or StringBuffer.

**Java String** class provides a lot of methods to perform operations on strings such as compare(), concat(), equals(), split(), length(), replace(), compareTo(), intern(), substring() etc.

The java.lang.String class implements *Serializable*, *Comparable* and *CharSequence* [interfaces](https://www.javatpoint.com/interface-in-java).

we can create strings in Java by using these three classes.

1. String
2. StringBuilder
3. StringBuffer

The Java String is immutable which means it cannot be changed. Whenever we change any string, a new instance is created. For mutable strings, you can use StringBuffer and StringBuilder classes.

Each time you create a string literal, the JVM checks the "string constant pool" first. If the string already exists in the pool, a reference to the pooled instance is returned. If the string doesn't exist in the pool, a new string instance is created and placed in the pool. For example:

1. String s1="Welcome";
2. String s2="Welcome";//It doesn't create a new instance



Note: String objects are stored in a special memory area known as the "string constant pool".

String s="Sachin";

 s.concat(" Tendulkar");//concat() method appends the string at the end

System.out.println(s);//will print Sachin because strings are immutable objects

Here Sachin is not changed but a new object is created with Sachin Tendulkar. That is why String is known as immutable.

**Operators in Java:**

An operand is a term used to describe any object that is manipulated by an operator.

Ex: 2+3, Here 2 and 3 are operand and + is operator.

Combination of operators and operands is called expression.

**If-Else in Java:**

The [Java](https://www.javatpoint.com/java-tutorial) *if statement* is used to test the condition. It checks [boolean](https://www.javatpoint.com/boolean-keyword-in-java) condition: *true* or *false*. There are various types of if statement in Java.

* if statement
* if-else statement
* if-else-if ladder
* nested if statement

if(condition1){

//code to be executed if condition1 is true

}else if(condition2){

//code to be executed if condition2 is true

}

else if(condition3){

//code to be executed if condition3 is true

}

...

else{

//code to be executed if all the conditions are false

}

**Switch Statement in Java:**

The Java *switch statement* executes one statement from multiple conditions. It is like [if-else-if](https://www.javatpoint.com/java-if-else) ladder statement.

int number=20;

//Switch expression

switch(number){

//Case statements

case 10: System.out.println("10");

break;

case 20: System.out.println("20");

break;

case 30: System.out.println("30");

break;

//Default case statement

default:System.out.println("Not in 10, 20 or 30");

}

We can use switch statement inside other switch statement in Java. It is known as nested switch statement.