# JAVA COURSE

# VARIABLES

Imagine variables like a box inside of computer that you can put your changeable values there. This values could be anything like numbers, letters, words, sentences, float numbers, long paragraphs etc. Everything you can imagine within programming could be a variable if it is something which can have a value. And computer allows you to have hundreds of thousands of boxes which contains its own pieces of information.

We use variables because they make a computer programming much more useful and dynamic for end users. For example imagine that we write a calculator, we will need to put some variables so boxes there that can hold the numbers that end user will give us. If we wouldn’t have these boxes which named as variables in computer language, we couldn’t take the values from the end user to make any calculation and the program would not be worth to anything. Therefore variables are one of the most important things in programming.

## Data Types

One of the most popular and important data type is “**int” .** It has a pretty long range that could be enough for most of programs. Just to have an idea about the range : –2,147,483,648 to 2,147,483,647

If we try to put a value bigger than this range, we will get an error because of the fact that it is out of the range than allocated place in the memory. If we would like to use a smaller allocated data type, then we could use **byte.** The range of byte is much smaller like -128 to 127. Other data type is **short.** The range of the short is -32768 to 32767. Our other data type is **long**  which you can understand from name also that it has a long range like –9,223,372,036,854,775,808 to 9,223,372,036,854,775,807.

Briefly what we have learned : **long, int, short, byte** data types. These are for whole numbers always. For decimal numbers what do we use ?

For decimal numbers we have **float**, and **double. Float**  has 7 decimal digit after the point but **double** has 16 digit after the point. After we put our values to variable, we need to write **f** for **float, d** for **double**.

And we have more primitive data types of course ☺ We keep moving with **char** data type which is used for single characters. Also we can use Unicode for characters as well instead of putting the letter inside the variable. For this : <https://unicode-table.com/ru/#control-character> we can take a look at this page.

We should use for copyright character for example ***char mychar = ‘\u009A ’;***

Another type is **boolean** which is true or false.

Until now we covered 8 data types listed below :

* ***byte***
* ***short***
* ***int***
* ***long***
* ***float***
* ***double***
* ***char***
* ***Boolean***

These were all primitive data types. So what is that not primitive data type ?

Well we will learn about **String** data type now which is not a primitive data type. Usage of **String**  is not almost different than other data types. We put the keyword String and name of our variable and use “ character to add your string between.

***String myString = “This is my first String”;***

That simple as it is in the example above. With adding “+” character between strings, we can combine them to together as they were written next to each other which is a power of “+” operator for strings. We can use also Unicode for strings as well. **String** is not a primitive data type because as it is with capital “S”, it shows that it is actually a class. So what we are doing here is actually, we are creating an object for that class and using it. **Integer** is also another un-primitive data type

## Important

*int is a primitive type. Variables of type int store the actual binary value for the integer you want to represent. int.parseInt("1") doesn't make sense because int is not a class and therefore doesn't have any methods.*

*Integer is a class, no different from any other in the Java language. Variables of type Integer store references to Integer objects, just as with any other reference (object) type. Integer.parseInt("1") is a call to the static method parseInt from class Integer (note that this method actually returns an int and not an Integer).*

*To be more specific, Integer is a class with a single field of type int. This class is used where you need an int to be treated like any other object, such as in generic types or situations where you need nullability.*

*Note that every primitive type in Java has an equivalent wrapper class:*

* *byte has Byte*
* *short has Short*
* *int has Integer*
* *long has Long*
* *boolean has Boolean*
* *char has Character*
* *float has Float*
* *double has Double*

*Wrapper classes inherit from Object class, and primitive don't. So it can be used in collections with Object reference or with Generics.*

*Since java 5 we have autoboxing, and the conversion between primitive and wrapper class is done automatically. Beware, however, as this can introduce subtle bugs and performance problems; being explicit about conversions never hurts.*

## Summary

* Primitive and un-primitive data types
* What is the reference and value for a data types ?
* Ranges of data types
* How does “+” operator work with String and char data types?

## Mission

***String manipulations and methods for interviews…***

# Arrays