Datatypes in Kotlin

In Kotlin, everything is an object. Rather than using wrapper classes, Kotlin promotes primitive datatypes to objects. To get better performance, Kotlin compiler maps basic types to JVM primitives. Following basic types supported by Kotlin are-

- Numbers
- Booleans
- Characters
- Arrays
- String

Numbers in Kotlin

Kotlin supports the following numbers which are mentioned in the table along with their size in byte-

Number Types	Size(in Bytes)	Range
Byte	1	-128 to 127
Short	2	-32768 to 32767
Int	4	-2^{31} to $2^{31} - 1$
Long	8	-2^{63} to $2^{63} - 1$
Float	4	-2^{31} to $2^{31} - 1$
Double	8	-2^{63} to $2^{63} - 1$

To assign a value to a Long variable, a value must be suffix with L and a Foat must be suffix with F. By default, floating point numbers are treated as double and integral numbers are treated as int.

Examples

```
val i = 187 //This is an Integer variable
val l = 98765L //This is Long variable
val f = 9.87F //This is Float variable
val d = 9.87 //This is Double variable
```

Explicit Conversion in Kotlin

Kotlin does not support implicit conversion. If you want to perform conversion between basic types, then it must be done explicitly. Every number is provided with the following functions that enable it to convert between types- toByte(), toShort(), toInt(), toLong(), toFloat(), toDouble(), toChar(). Let's understand it with the help of an example-

```
var b:Byte = 10
var i:Int = b //This will give compile-time error
var j:Int = b.toInt() //This will work
```

Boolean

You can assign `true` or `false` value to any boolean variable. It is mostly used in decision-making statements and loops.

```
fun main(args : Array<String>) {
    val f: Boolean = true
    println("$f")
}
```

Characters

In Kotlin, if you want to represent character then use `Char` data type. Kotlin's `Char` type is different from Java. You cannot use it as a number. In case you assign a number to any `Char` type variable, you will get a compile-time error.

```
fun main(args : Array<String>) {
    val letter: Char = 'K'
    println("$letter")
}
```

When you run the above program, you get the following output-

```
K
```

Java treats character as number and it is valid to write like this-

```
char letter = 75;
```

If you try to run it in Kotlin, you will get error.

```
var letter: Char = 75 //Compile-time error.
```

Arrays in Kotlin

An array is an object that is used to store multiple values of the same type. In other words, it is used to store homogeneous values.

In Kotlin, arrays are represented by the `Array` class. The class has `get` and `set` functions, `size` property, and a few other useful member functions.

String in Kotlin

In Kotlin, `String` class is used to represent strings. When you enclose a sequence of characters in double quotation marks or triple quotes, it becomes string literals.

To learn string in details, visit: String and String Templates in Kotlin