**String class:**

* It’s a class present in java.lang package
* String class is used to store character sequence (Array of characters)
* It provides methods for string manipulation
* In string class toString() method is overridden to print the content of the string
* In string class equals method is also overridden to compare the current string object and the given string object based on character stored in the string

String instance can be created in 2 ways:

1.Using new keyword

E.g.: String s1= new String(“java and selenium”);

2.By assigning string literal

E.g.: String s2=”Java and Selenium”;

* String is immutable that means once a string object is created, the same object can’t be modified. If you try to modify a string, a new string will be created instead of modifying the existing string—see D83\_StringClass program

**Important methods of String class:**

toString()

equals()

equalsIgnoreCase()

length()

charAt()

substring()

replace()

indexOf()

trim()

**Difference b/w String, StringBuffer and StringBuilder classes:**

|  |  |  |
| --- | --- | --- |
| String | StringBuffer | StringBuilder |
| String class is immutable | Mutable | Mutable |
| String is slow and consumes more memory when you concat too many strings because every time it creates new instance. | StringBuffer is faster than String and slower than String builder and consumes less memory when you concat strings. | [StringBuilder is more faster than StringBuffer](https://www.journaldev.com/137/stringbuffer-vs-stringbuilder). |
| String class overrides the equals() method of Object class. So you can compare the contents of two strings by equals() method. | StringBuffer class doesn't override the equals() method of Object class. | StringBuffer class doesn't override the equals() method of Object class. |
| NA | StringBuffer classes are S*ynchronized* | StringBuilder classes are *non-synchronized* |

**Arrays:** its available in java.util package

* In java, array is an object.
* Array is a group of similar types
* Array is fixed in size

Array can be declared using below syntax:

1. dataype[] variable=new datatype[size]
2. datatype[] variable={element1, element2…etc.}

* “length” is a variable which returns the size of an array
* Array elements are always accessed with respect to index

**Wrappers classes:**

For each primitive data type there is a correspondence non-primitive types. These non-primitives are called as “wrapper classes”.

All wrapper classes present in “java.lang” package.

All wrappers classes are “Final” classes.

|  |  |
| --- | --- |
| Primitive data types | Wrapper classes |
| byte | Byte |
| short | Short |
| int | Integer |
| long | Long |
| float | Float |
| double | Double |
| char | Character |
| boolean | Boolean |

**Boxing:** The process of converting primitive type into non-primitive(object) is called as Boxing.

E.g.:

1. byte k=10;

Byte b= new Byte(k);

2. int a=20;

Integer i=new Integer(a);

**Un-Boxing:** the process of converting wrapper classes( non-primitive type or object ) into corresponding primitive type is called as Un-Boxing.

E.g.:

1. Double d= new Double(10.3);

double d1=d.doubleValue();

2.Integer i1=new Integer(10);

Int i2=i1.intValue();