Wrapper Classes

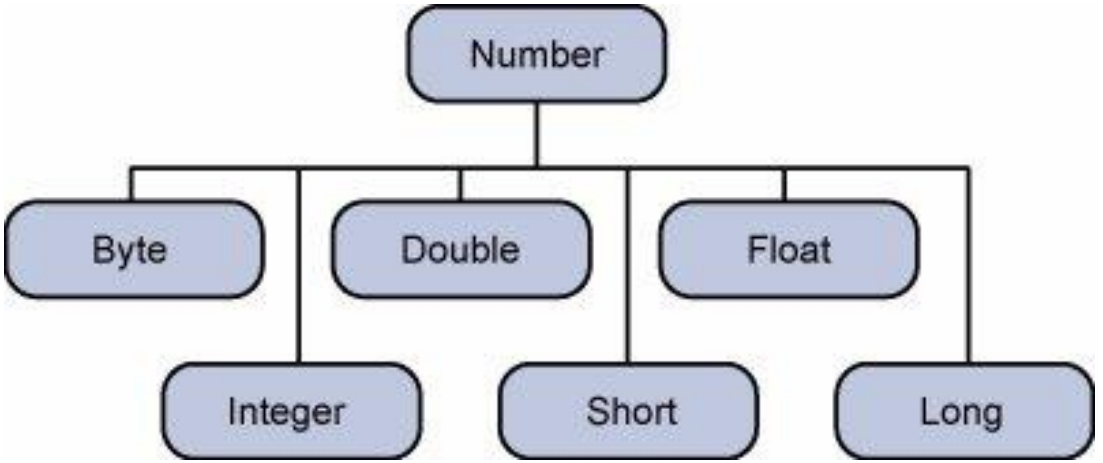
- Java is an object-oriented language and can view everything as an object.

- A simple file can be treated as an object (with *java.io.File*), an address of a system can be seen as an object (with *java.util.URL*), an image can be treated as an object (with *java.awt.Image*)

- And a simple data type can be converted into an object (with wrapper classes).Wrapper classes are used to convert any data type into an object.

- As the name says, a wrapper class wraps (encloses) around a data type and gives it an object appearance.
- Wherever, the data type is required as an object, this object can be used.
- Wrapper classes include methods to unwrap the object and give back the data type.

Primitive Type	**Wrapper class**
boolean	Boolean
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double



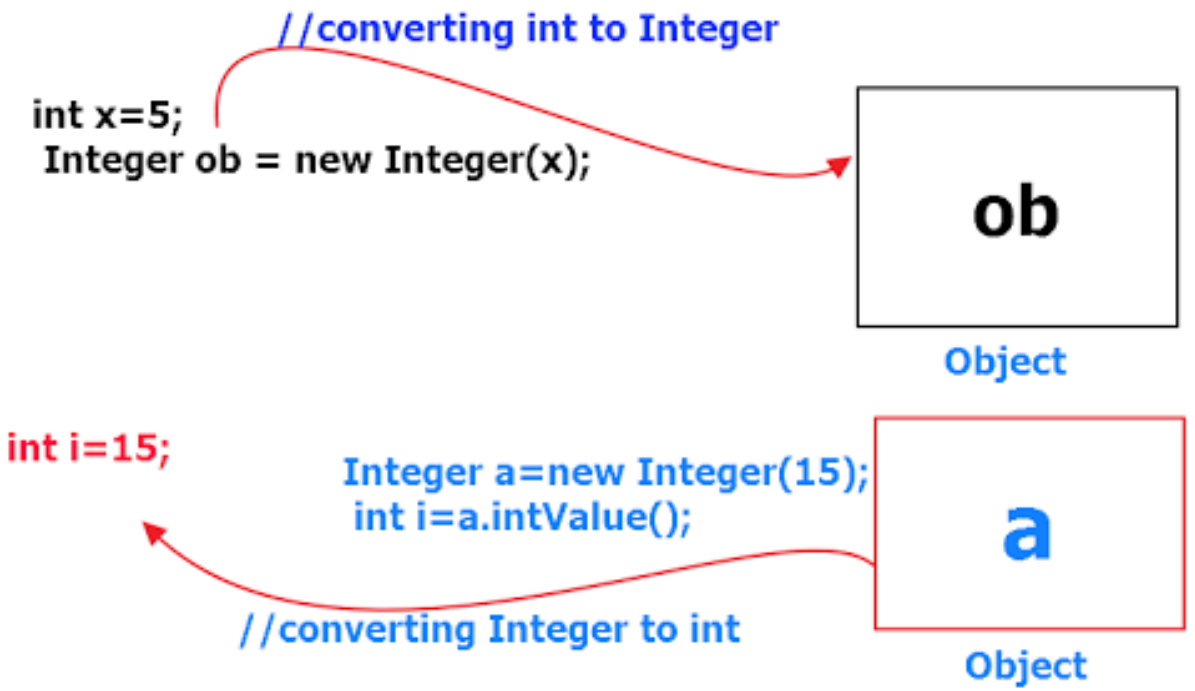
- Converting primitive data types into object is called boxing, and this is taken care by the compiler. Therefore, while using a wrapper class you just need to pass the value of the primitive data type to the constructor of the Wrapper class.

- And the Wrapper object will be converted back to a primitive data type, and this process is called unboxing. The *Number* class is part of the java.lang package.

- Following is an example of boxing and unboxing

```
public class Test {
   public static void main(String
args[]) {

      Integer x = 5;
      x =  x + 10;
      System.out.println(x);
   }
}
```



Autoboxing and Unboxing

Methods Implemented by all Subclasses of Number

Method	Description
byte byteValue() short shortValue() int intValue() long longValue() float floatValue() double doubleValue()	Converts the value of this Number object to the primitive data type returned.
int compareTo(Byte anotherByte) int compareTo(Double anotherDouble) int compareTo(Float anotherFloat) int compareTo(Integer anotherInteger) int compareTo(Long anotherLong) int compareTo(Short anotherShort)	Compares this Number object to the argument.
boolean equals(Object obj)	Determines whether this number object is equal to the argument.

Usage of methods

- Similarly for other classes also

```
Integer obj = new Integer(15);
    int i = obj.intValue()
```

```
public class IntegerDemo {
   public static void main(String[] args) {
   Integer obj1 = new Integer("25");
   Integer obj2 = new Integer("10");
   int retval =  obj1.compareTo(obj2);
   if(retval > 0) { System.out.println("obj1 is greater than obj2"); }

   else if(retval < 0) { System.out.println("obj1 is less than obj2");  }

   else {    System.out.println("obj1 is equal to obj2");  }
   }//end of main
} //end of class
```

boolean equals(Object obj)

```
import java.lang.*;
public class BooleanDemo {
   public static void main(String[] args) {
      Integer b1, b2;
      boolean res;
      b1 = new Integer(10);
      b2 = new Integer(20);
      // assign the result of equals method on b1, b2 to res
      res = b1.equals(b2);
      String str = "b1:" +b1+ " and b2:" +b2+ " are equal is " +
res;

      // print res value
      System.out.println( str );
   }
}
```

Conversion methods

- Each Number class contains other methods that are useful for converting numbers to and from strings and for converting between number systems.
- The following table lists these methods in the Integer class. Methods for the other Number subclasses are similar:

Methods	Description
static int parseInt(String s)	Returns an integer (decimal only)
static int parseInt(String s, int radix)	Returns an integer, given a string representation of decimal, binary, octal, or hexadecimal (radix equals 10, 2, 8, or 16 respectively) numbers as input.
String toString()	Returns a String object representing the value of this Integer.
static String toString(int i)	Returns a String object representing the specified integer.
static Integer valueOf(int i)	Returns an Integer object holding the value of the specified primitive.
static Integer valueOf(String s)	Returns an Integer object holding the value of the specified string representation.
static Integer valueOf(String s, int radix)	Returns an Integer object holding the integer value of the specified string representation, parsed with the value of radix. For example, if s = "333" and radix = 8, the method returns the base-ten integer equivalent of the octal number 333.

```
public class Test{
   public static void main(String
args[]){
      int x =Integer.parseInt("9");
      double c =
Double.parseDouble("5");
       int b =
Integer.parseInt("444",16);
      System.out.println(x);
      System.out.println(c);
      System.out.println(b);
   }
}
```

```
import java.lang.*;
public class IntegerDemo {
   public static void main(String[]
args) {
   Integer i = new Integer(10);

   String retval = i.toString(30);
   System.out.println("Value = " +
retval);
   }
}
```