Wrapper Classes in Java

A class that wraps a primitive data type into an object is called **wrapper** class in Java.

In simple words, **wrapper** class provides a mechanism to convert **primitive data type value** into an **object** and vice-versa.

For example:

wrapping **int** into **Integer** class, wrapping **double** into **Double** class, and wrapping **char** into **Character** class.

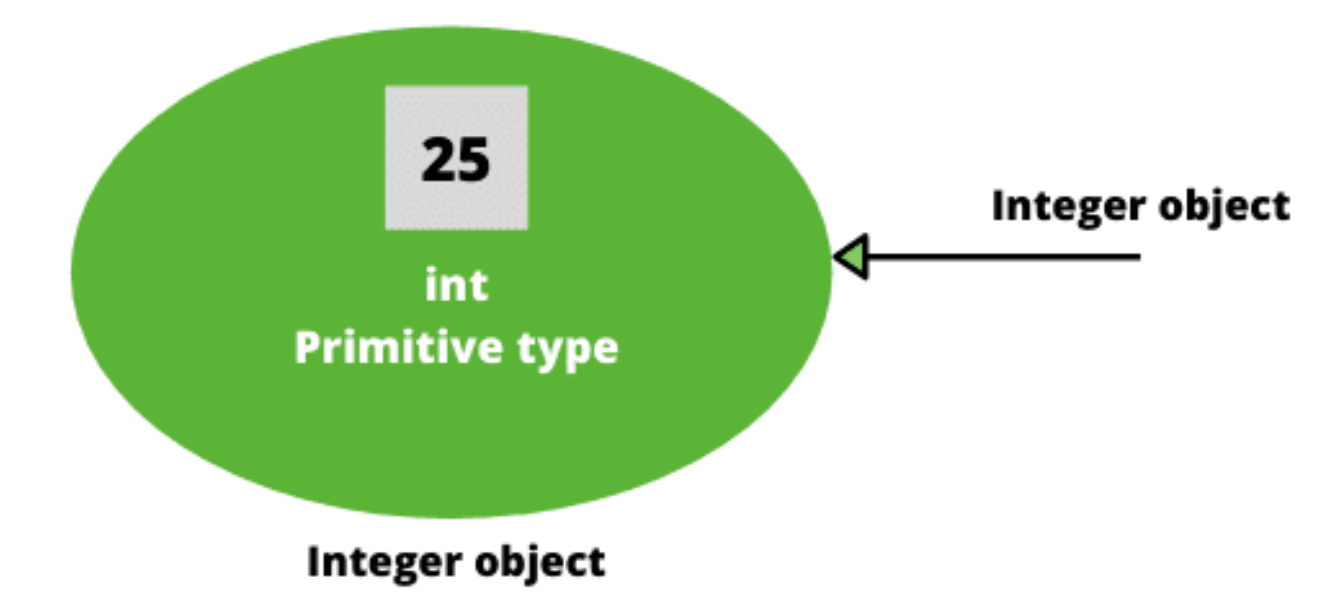
When we create an object of **wrapper** class, it contains a variable where we store the primitive data type value.

We can wrap a **primitive** type value into a **wrapper** class object.

Wrapper Classes in Java

For example, if we create an object of Integer wrapper class, it contains a single variable (or field) that will store an int value like 25, as shown in the below figure.

Thus, Integer is a wrapper class of int data type.



Wrapper Classes in Java

Example:

//Wrapping an int value:

```
int x = 25;
Integer iWrap = new Integer(x); // Pass the primitive type to the wrapper constructor.
```

//Unwrapping an int value:

```
int unWrapped = iWrap.intValue();
```

Why we need or use wrapper classes in Java?

We have seen several applications on the Internet that receive data from the user and send it to the server.

For example:

In a business application like **Amazon**, we enter our details like **name**, **debit** or **credit** card number, **address**, **phone** number, etc at the time of purchasing anything.

All these data are sent to the server. The server expects these data in the form of objects. In our data," '**name**' is a String type object, but debit card number is an **int** type value, that is not an object.

Autoboxing and Unboxing

Java 5.0 version introduced two important new features **autoboxing** and **unboxing** that convert primitive data type values into objects and objects into primitive data type values automatically.

The process of automatic conversion of primitive data type into an object is known as **autoboxing** and vice versa **unboxing**.

Example:

```
// autoboxing
Integer Obj = 100;

// unboxing
int a = Obj;
```

List of Wrapper Classes in Java

1. Java platform provides the list of eight wrapper classes for each of the primitive types.

2. These wrapper classes are defined in **java.lang** package.

3. All the java wrapper classes are immutable and final whose objects each hold a single primitive value. **Immutable** means once objects are created, their internal values cannot be changed.

4. Methods of wrapper classes are implicitly final and may not be overridden.

5. They are useful to convert primitive data types into object forms.

List of Wrapper Classes in Java

Primitive Data type	Corresponding Wrapper class
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean

Points to remember:

1. Wrapper class in Java provides several constructors, constants, and conversion methods for manipulating various primitive data type values.

2. Wrapper classes have no no-arg constructors.

3. Character class does not have a constructor with String as a parameter.

4. The instances of all wrapper classes are immutable and final. That means once the object is created, its internal value cannot be changed.