**What are Wrapper Classes in Java?**

Wrapper classes are Java predefined classes that are responsible to convert the given string type the numerical value into equivalent primitive data type and vice-versa. A wrapper class is bundled default with the Java library and it is located in (jre/lib/rt.jar file). When we create an object to a wrapper class, it contains a field and in this field, we can store primitive data types. Some of the wrapper classes equivalent to primitive data types are as follows:



**Need for Wrapper Classes:**

The wrapper objects hold much more memory compared to primitive types.

Wrapper Class will convert primitive data types into objects. The object is needed to support synchronization in multithreading.

Wrapper class objects allow null values while primitive data type doesn’t allow it.

public class WrapperDemo

{

public static void main (String[]args)

{

String i = "10";

int k = Integer.parseInt (i);

double m = Double.parseDouble ("10");

System.out.println (k);

System.out.println (m);

boolean status = Boolean.parseBoolean ("20");

System.out.println (status);

}

}

Note: The Wrapper class accepts only string numeric values and the Boolean wrapper class is added from version1.5 Java.

**How to Implement a Wrapper Class in Java?**

Wrapper class can be implemented in Java in the following two ways:

Autoboxing

Unboxing

In general, autoboxing and unboxing take place whenever a conversion into an object or from an object is required. Thus, autoboxing/ unboxing might occur when an argument is passed to a method, or when a value is returned by a method.

Autoboxing in Java:

“Boxing” refers to converting a primitive value into a corresponding wrapper object. Because this can happen automatically, it’s known as autoboxing. In addition to the simple case of assignments, autoboxing automatically occurs whenever a primitive type must be converted into an object.

With autoboxing, it is no longer necessary to manually construct an object in order to wrap a primitive type. You need only assign that value to a type-wrapper reference. Java automatically constructs the object for you.

public class AutoBoxing

{

public static void main (String args[])

{

int a = 100; // Primitive data type

Integer I = a; // Autoboxing will occur internally.

}

}

**Unboxing in Java:**

Auto-unboxing takes place whenever an object must be converted into a primitive type. Auto-unboxing is the process by which the value of a boxed object is automatically extracted (unboxed) from a type wrapper when its value is needed. There is no need to call a method such as intValue( ) or doubleValue( ).

public class AutoBoxing

{

public static void main (String args[])

{

Integer a = new Integer (15); // Wrapper class object

int I = a; // Unboxing will occur internally.

}

}

Need for Autoboxing and Unboxing in Java:

The addition of autoboxing and auto-unboxing greatly streamlines the coding of several algorithms, removing the tedium of manually boxing and unboxing values.

It also helps prevent errors.

Moreover, it is very important to generics, which operate only on objects.