Wrapper Class

Wrapper class provides a way to use primitive data types as objects.

Primitive data types are the most basic data types available within the java language. There are 8: boolean, byte, char, short, int, long, float and double.

NEED OF WRAPPER CLASS

1. They convert primitive data types into objects
2. The classes in java.util.package handles only objects and hence wrapper classes help in this case also.
3. Data structures in the collection framework, such as Arraylist store only objects and not primitive types
4. An object is needed to support synchronization in multithreading.

PRIMITIVE DATA TYPES AND THEIR CORRESPONDING WRAPPER CLASS

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

AUTOBOXING AND UNBOXING

Automatic conversion of primitive data types to the object of their corresponding wrapper class is known as autoboxing. Example - conversion of float to Float, boolean to Boolean.

```
public void AutoboxingSample(){
        char alph = 'a';
//      Autoboxing - primitive to Character object conversion
        Character a = alph;

        ArrayList<Integer> arrayList = new ArrayList<Integer>();
//      We use autoboxing here to store integers because arraylist stores only objects
        arrayList.add(25);

//      printing the values from object
        System.out.println(arrayList.get(0));

}
```

Unboxing is converting an object of a wrapper class to its corresponding primitive type. Example, conversion of Float to float, Character to char and Integer to Int.

```
public void UnboxingSample(){
        Character alph = 'a';
//      Unboxing- Character object to primitive conversion
        char a = alph;

        ArrayList<Integer> arrayList = new ArrayList<Integer>();
        arrayList.add(24);

//      We are unboxing here because the get method returns an integer object
        Int num = arrayList.get(0);

//      printing the values from primitive data type
        System.out.println(num);
```

```
}
```