Day 07 – Wrapper Class & Type Conversion in Java

This chapter explains Wrapper classes in Java and how to convert primitive data types into wrapper objects and wrapper objects back into primitive types. All topics are explained clearly with examples.

1. What is a Wrapper Class?

A Wrapper class converts a primitive data type into an object. Java provides wrapper classes for all primitive data types.

Primitive Types and Wrapper Classes

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

2. Why Wrapper Classes are Needed?

Wrapper classes are used in collections, object-oriented APIs, and utility methods where objects are required.

3. Primitive to Wrapper Conversion (Boxing)

The process of converting a primitive type into a wrapper object is known as Boxing.

Example:

```
int a = 10;
Integer obj = Integer.valueOf(a);
```

Auto Boxing Example

```
int b = 20;
Integer obj2 = b;
```

4. Wrapper to Primitive Conversion (Unboxing)

The process of converting a wrapper object into a primitive type is known as Unboxing.

Example:

```
Integer obj = Integer.valueOf(50);
int x = obj.intValue();
```

Auto Unboxing Example

```
Integer obj2 = 60;
int y = obj2;
```

5. Wrapper Class Utility Methods

- parseInt() – Converts String to int
- valueOf() – Converts primitive or String to wrapper object
- toString() – Converts value to String

6. Common Mistakes

• Storing null in primitive types
• Confusing parseInt() with valueOf()
• Not understanding auto-boxing

Conclusion

Wrapper classes connect primitive data types with objects. Understanding boxing and unboxing is essential for Java development.