Learning Goals

Click to continue

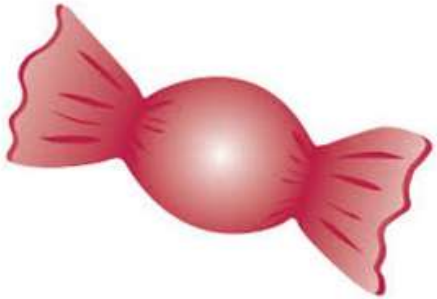
This chapter will help you to understand,

- About Wrapper Classes?
- Types of Wrapper Classes?
- Usage of Wrapper API's?

Wrappers Classes

Click to continue

Wrapper Classes are an object representation of primitive data types



Primitive data types (the actual chocolate) are wrapped or converted into objects using wrapper classes

Primitive data types → changed to → Wrapper Objects

- Primitive data type are not objects. Whenever the data is needed as an object, wrapper classes can be used to convert the primitive data into an object.
- Wrapper classes has many methods for processing/transforming the primitive data types.

  Example: Assume you have a number 100.2345 and you need to round it to 100.23. Wrappers provide API's to perform this easily.

Wrapper Classes

Click to continue

- Each primitive data type has their own corresponding wrapper class.
- The name of the wrapper object is same as the primitive data type except that the first letter is capitalized.
- Eight wrapper classes exist in **java.lang** package that represent 8 primitive data types.

Example: Wrapper class of

- Primitive data type float is Float
- Primitive data type long is Long

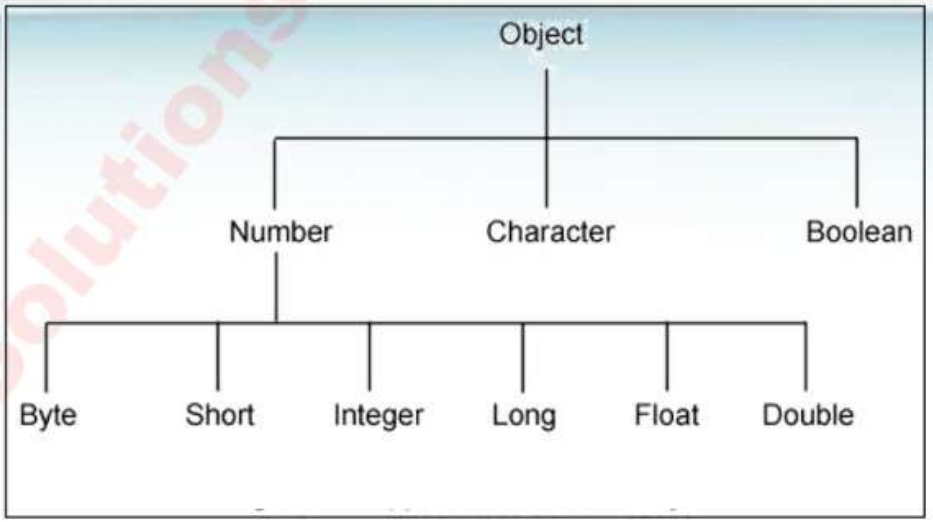
Primitive Wrapper Mapping

Click to continue

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

Hierarchy of Wrapper classes

Click to continue



- Super class of Boolean and Character is Object
- Super class of all Numeric wrapper classes is Number
- Number has six concrete subclasses that hold explicit values of each numeric type Double, Float, Byte, Short, Integer, and Long.

Convert Primitives to Wrappers

Click to continue

Some more examples on other primitive to wrapper conversion is illustrated below.

Float salary = new Float(“5.0f”);

Other illustration

- Boolean boolean1= new Boolean("false");
- Byte byte1= new Byte("2");
- Short short1= new Short("4");
- Integer int1 = new Integer("16");
- Long long1 = new Long("123");
- Float float1 = new Float("12.34f");
- Double double1 = new Double("12.56d");
- Character char1 = new Character('c');

Convert int to String

Click to continue

String to int:

int number = Integer.parseInt("15000");

Converts the String "15000" into an int data type with value as 15000.

Int to string:

String number = Integer.toString(200);

The above statement converts the int 200 into an String data type with value as 100.

NOTE: String object can be converted to any primitive using the corresponding Wrapper Classes

Integer Wrapper

Click to continue

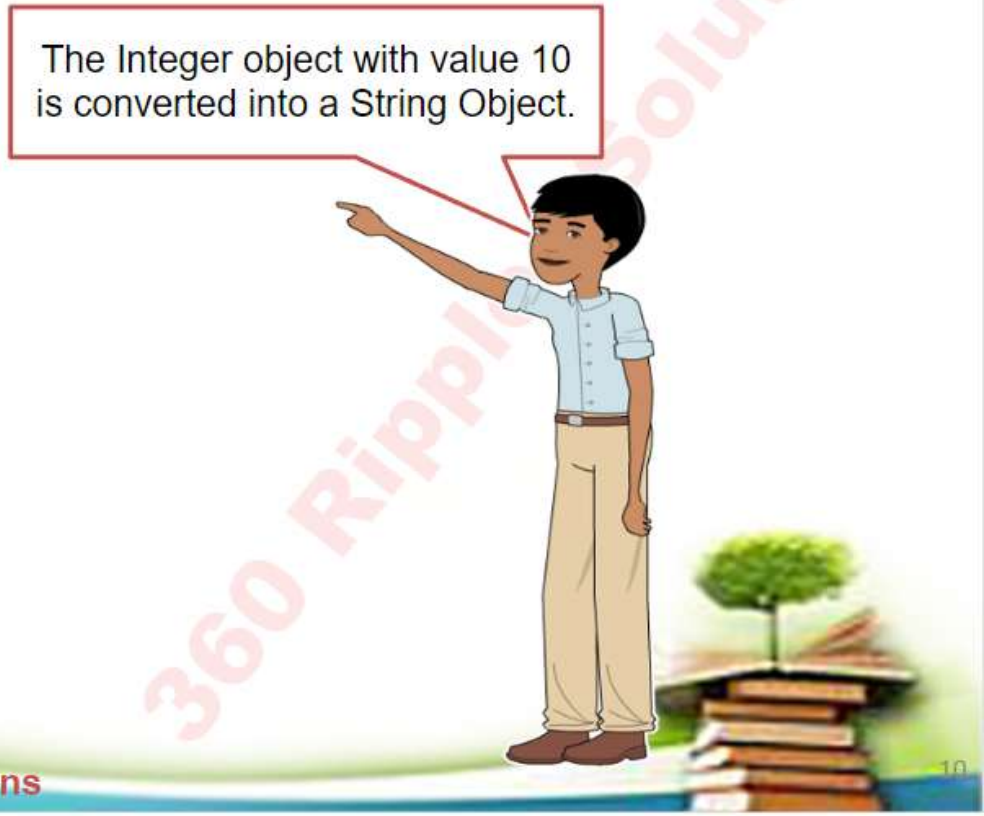
- The Integer class wraps a value of the primitive type int into an object.
- This class also provides several API's for converting int to String and vice versa, as well as other useful methods for processing int value.

Illustration 1: To extract the primitive int value of the Integer object

```
Integer number= new Integer(10);
int intValue = number.intValue();
```

Illustration 2: To get the value of the Integer as a String.

```
Integer number= new Integer(10);
String stringValue = number.toString();
```



1. Create a java class *"IntegerExample"* inside the package com.wrapper.demo.
2. Create a main method and create two Integer Objects"num1" and "num2" with values 1234 and 4321. Implement the following logic,

Problem # 1: Convert num1 as int and num2 as long values and print the values as below.

Output: "The primitive value of the Integer= "<int value> & "The primitive long value of the Integer= "<long value>

Problem # 2: Compares both the Integer variables num1 and num2 using an API and print the bigger value.

Output: " The<value> is bigger than <value>"

Problem # 3: Retrieves and prints the Maximum and minimum value that an int can have.

Output: "Maximum Integer value="<Max Value> "Minimum Integer value="<Min Value>

Problem # 4: Converts a String value of String s = "4321" to an int value and add 200 to it and print it.

Output: The string should be converted into a int incremented to 200 and display it.