Basavaraj Aili

KodNest CSR June 2023

(19-07-2023)

Assignment: Wrapper Classes in JAVA

Wrapper Classes in JAVA

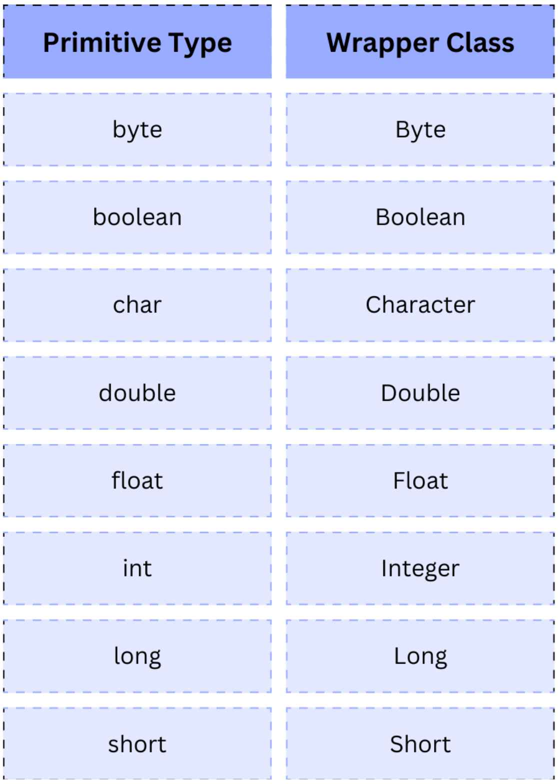
Generic classes are object-oriented and do not allow Primitives. As a result, Wrapper classes are required because they convert primitive data types into objects, and objects are critical if we need to change the arguments passed into a method

The Java programming language includes the java.lang package, which contains classes that are essential to the design, the most significant of which are Object and Class.

As a result, Java wrapper classes are objects that wrap or represent the values of primitive data types. When a wrapper class object is created, it includes a field that can store primitive data types.

An object of one type contains only fields of that type, so a Double type object contains only fields of the Double type, representing that value so that a reference to it can be kept in a variable of reference type.

Wrapper classes are the eight classes that comprise the Java.lang libraries in Java. The eight parent classes are as follows:



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

A wrapper class in the Java programming language is a class that acts as a container or wrapper for primitive data types. This enables them to be used in situations where objects are required and adds additional capability by allowing these primitive data types to be handled as objects.

The Java wrapper classes are:

- **'Boolean'** for the boolean data type.
- **'Byte'** for the byte data type.
- **'Short'** stands for the short data type.
- **'Integer'** for the int data type.
- **'Long'** for the long data type.
- **'Float'** stands for the float data type.
- **"Double"** designating a double data type.
- **'Character'** refers to the char data type.

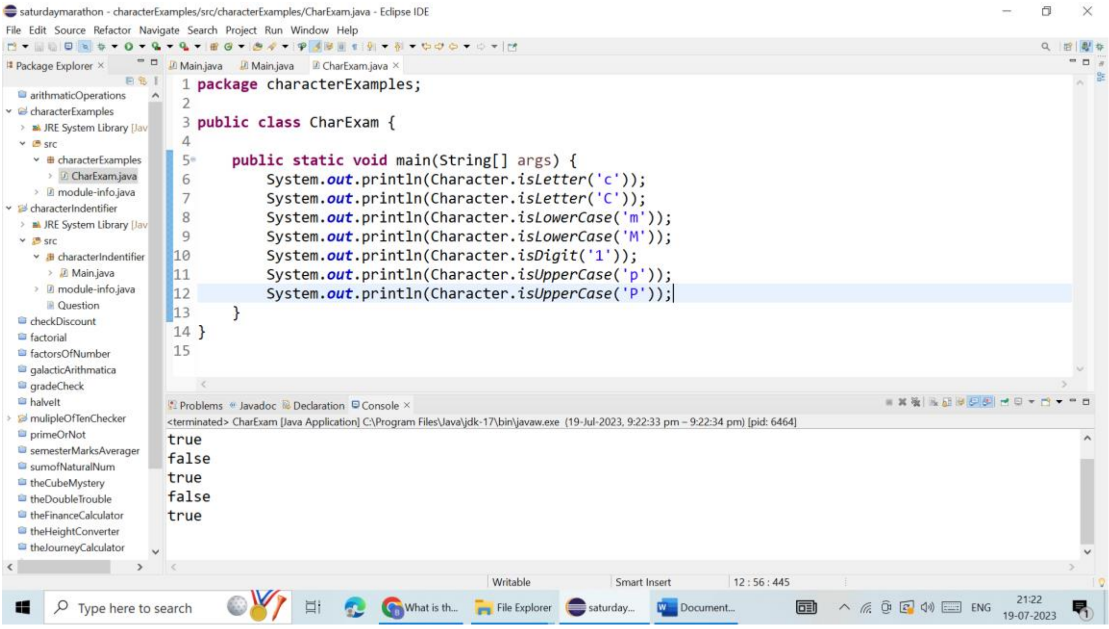
Character wrapper class and its methods in Java

The Character class of the **java.lang** package wraps a value of the primitive datatype char. It offers a number of useful class (i.e., static) methods for manipulating characters. You can create a Character object with the Character constructor.

```
Character ch = new Character('a');
```

Following are the notable methods of the Character class.

1	**isLetter()** **Determines whether the specified char value is a letter.**
2	**isDigit()** Determines whether the specified char value is a digit.
3	**isWhitespace()** Determines whether the specified char value is white space.
4	**isUpperCase()** Determines whether the specified char value is uppercase.
5	**isLowerCase()** Determines whether the specified char value is lowercase.
6	**toUpperCase()** Returns the uppercase form of the specified char value.
7	**toLowerCase()** Returns the lowercase form of the specified char value.
8	**toString()** Returns a String object representing the specified character value that is, a one-character string.



```java
package characterExamples;

public class CharExam {

    public static void main(String[] args) {
        System.out.println(Character.isLetter('c'));
        System.out.println(Character.isLetter('C'));
        System.out.println(Character.isLowerCase('m'));
        System.out.println(Character.isLowerCase('M'));
        System.out.println(Character.isDigit('1'));
        System.out.println(Character.isUpperCase('p'));
        System.out.println(Character.isUpperCase('P'));
    }
}
```

```
<terminated> CharExam [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe  (19-Jul-2023, 9:22:33 pm – 9:22:34 pm) [pid: 6464]
true
false
true
false
true
```

Problem Statement: Character Identifier

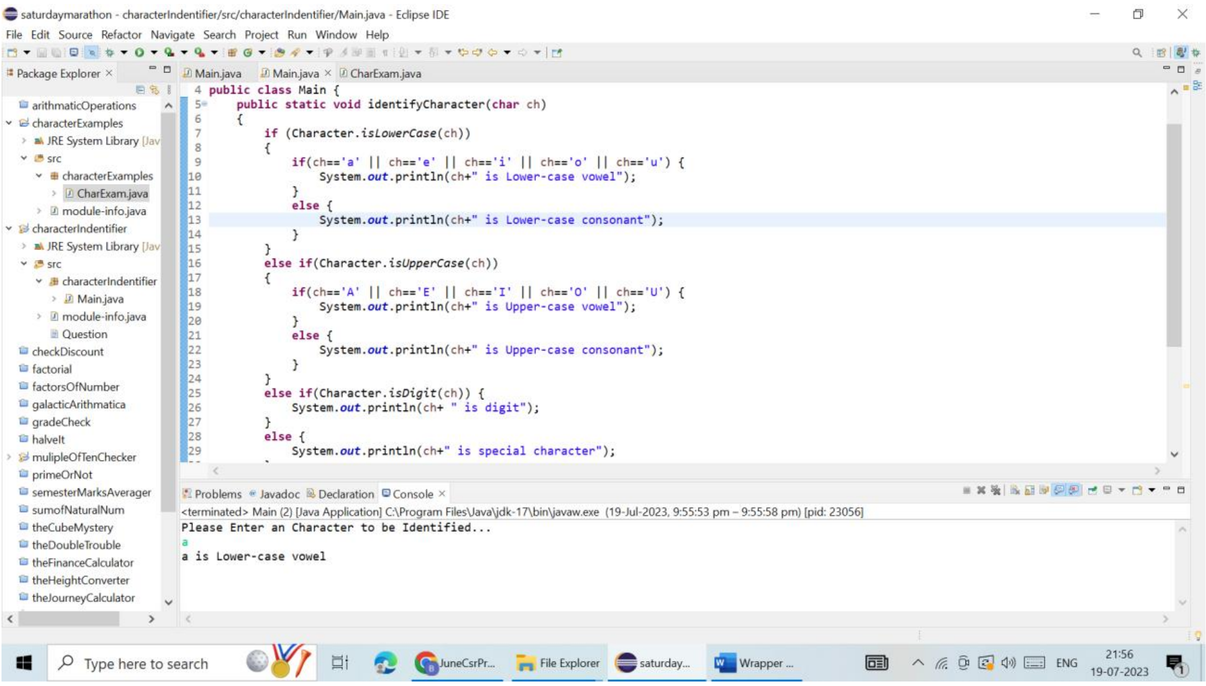
Question Description: Create a program that identifies a character as a lower-case vowel, upper-case vowel, lower-case consonant, upper-case consonant, digit, or special character. The program should accept a character 'ch' as input.

Boiler Plate Code:

```
public class Main

{

public static void identifyCharacter(char ch)

{

// your code here

}

public static void main(String[] args)

{

}

}
```

Sample Input: 'a'

Sample Output: Lower-case vowel



```
public class Main {
	public static void identifyCharacter(char ch)
	{
		if (Character.isLowerCase(ch))
		{
			if(ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch=='u') {
				System.out.println(ch+" is Lower-case vowel");
			}
			else {
				System.out.println(ch+" is Lower-case consonant");
			}
		}
		else if(Character.isUpperCase(ch))
		{
			if(ch=='A' || ch=='E' || ch=='I' || ch=='O' || ch=='U') {
				System.out.println(ch+" is Upper-case vowel");
			}
			else {
				System.out.println(ch+" is Upper-case consonant");
			}
		}
		else if(Character.isDigit(ch)) {
			System.out.println(ch+ " is digit");
		}
		else {
			System.out.println(ch+" is special character");
```

```
<terminated> Main (2) [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (19-Jul-2023, 9:55:53 pm – 9:55:58 pm) [pid: 23056]
Please Enter an Character to be Identified...
a
a is Lower-case vowel
```

Solution:

```
package characterIndentifier;
import java.util.Scanner;

public class Main {
	public static void identifyCharacter(char ch)
	{
		if (Character.isLowerCase(ch))
		{
			if(ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch=='u') {
				System.out.println(ch+" is Lower-case vowel");
			}
			else {
				System.out.println(ch+" is Lower-case consonant");
			}
		}
		else if(Character.isUpperCase(ch))
		{
			if(ch=='A' || ch=='E' || ch=='I' || ch=='O' || ch=='U') {
				System.out.println(ch+" is Upper-case vowel");
			}
			else {
				System.out.println(ch+" is Upper-case consonant");
			}
		}
		else if(Character.isDigit(ch)) {
			System.out.println(ch+ " is digit");
		}
		else {
			System.out.println(ch+" is special character");
		}
	}
	public static void main(String[] args)
	{
		Scanner scan=new Scanner(System.in);
		System.out.println("Please Enter an Character to be Identified...");
		char ch=scan.next().charAt(0);
		identifyCharacter(ch);
	}

}
```

Output:

```
Please Enter an Character to be Identified...
a
a is Lower-case vowel
```