Wednesday, November 10, 2021 10:32 AM

Primitive Data Type ---- in built types in the language

byte	Integer type	1 byte
short	Integer type	2 bytes
int	Integer type	4 bytes
long	Integer type	8 bytes
float	Decimal point	4 bytes
double	Decimal point	8 bytes
char	Character data type	2 bytes - Unicode encoding
boolean	true and false	Undefined (different for different JVMs)

User Defined Types / Custom Types

1. Class
2. Interface
3. Enum
4. annotations

String is a class ---------- object of the class is created!!!
String s = "ggg" ; //convenience syntax -- but internally it is creating object
Constant pool = ARRAY ---------- String s3 = "hello" --- check if the string is in the constant pool or not, if yes do not create object - **Reuse** same object !! Reuse is not causing problem because string is immutable !!!
Advantage of REUSE = memory space is saved!!!

0	hello
1	hi
2	bye

String s1= new String("ggg") ; ---- always a new object is created

COMPILE TIME evaluated Strings are always maintained in the constant pool
String s1 = "hello";
String s2 = " world" ;

Run time evaluated strings are NOT maintained in the constant pool
s1.concat(s2)
String s1 = new String("ee")

---

package

import = shortcut - message to compiler to use the full package name wherever the class name is

used .

If we don't want to give import --- we give fully qualified class name every time we use the class!!!

DEFAULT package in JAVA = java.lang package = COMPILER searches the classes without full package name or import in the java.lang package

java
----lang
----- String
----- System
----- StringBuffer
----- StringBuilder

This package comes along with JDK installation --- it is in **lib** folder of JDK
Our classes are in the **bin** folder of our project in our WORKSPACE !!

Wrapper Classes in Java ----------

The class that wraps a primitive type variable !!!

Every primitive type has a corresponding wrapper class in the java.lang package !!!

Java Naming Convention ----- camel case

Keywords	small case
primitive data types	small case
class name	**M**yClass
property/variable name	firstwordSecondword
method names	firstwordSecondword
constants	MAX_VALUE

```
class MyWrapperClass
{
    private int value;

    MyWrapperClass(int x)
    {
        Value =x
    }


    Public int intValue()
    {
        Return value;
    }

    ……
```

}

```
Main
{
        Int  x = 99;

        MyWrapperClass mwc = new MyWrapperClass(x) ;  // we are wrapping primitive in the
        object
        Int y=    mwc.intValue();
}
```

---

java.lang package has all the following wrapper classes

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

byte data type gets 1 byte space in JVM
short gets 2 bytes
int gets 4
long 8

_ _ _ _ _ _ _ _ } 2 raised to 8 = 256 /2 = 128

Range of byte is -128 to 127

WRAPPER classes have useful STATIC methods, STATIC constants !!!

1. **MAX_VALUE, MIN_VALUE**

Short range } 2 raised to 16 = 256*256=65,536 65536 /2 = 32768
-32768 to + 32767

Int range } 2 raised to 32 = 65536 * 65536 = 4E9 /2 = 2,000,000,000

Long range } 2 raised to 64 =

2. **Parse a data type from String**

        12 + 13 = 25 } + added

        "12"+"13" = "1213" } + concatenated

Extracting the data type from a string

Int x = Integer.parseInt("12") } x will be 12
Long.parseLong("8882324234023840")

Double.parseDouble()

Boolean.parseBoolean()

3. Character Wrapper class has useful methods like
   isLetter , isDigit.

```
boolean ans = Character.isDigit('L');
              ans = Character.isLetter('a');
              ans = Character.isUpperCase('a');
        System.out.println(ans);
```

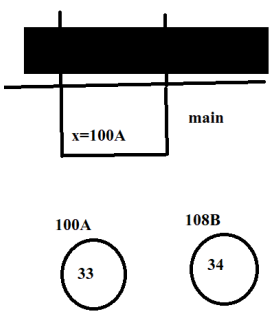
Assignment

1. Write a class WrapperExample
   a. Write a main method ---- accept a new password from user
      pass that password to a **verify**(pwd) , this method returns a boolean
      If password is valid then returned value is true , else false
      check if password is valid , if yes then say congratulations your password is set
      Else ask the user to reenter ---Loop for 5 times in case of invalid pwd

   b. Write a method boolean verify(String p) (use **charAt, length API of String class , use Character.isDigit, isLetter,isUpperCase,isLowerCase,……..)**
      Return true only if password satisfies the following condition , else return false
      1. The password length must be between 8 and 20
      2. The password must begin with a letter
      3. The password must have at least one capital character
      4. The password must have at least one digit
      5. The password must have at least one small character
      6. The password must have at least one special char ====>>> **_ - # $**

Wrapping a value in the object = BOXING
UnWrapping a value from the object = UNBOXING

All wrapper classes are IMMUTABLE



Assignment

```
package study;

public class TestWrapperClass4 {


	public static void main(String[] args) {

		//Integer x = 33;
		MyWrapper y = new MyWrapper();
		y.value = 33;
		System.out.println(y.value);
		increment(y);
		System.out.println(y.value);
	}

	public static void increment(Integer x)
	{
		x=x+1; //boxing + autoboxing
	}

	public static void increment(MyWrapper y)
	{
		y.value = y.value +1;
	}
}


class MyWrapper
{
	int value;
}
```

--- write a swap function that will accept 2 numbers and swap their values
Print before and after in main

Public static void swap(…….. V1, ……..v2)

Draw memory diagrams !!!!

Command Line Arguments in Java ----------

Assignment

Accept a few numbers from the command line
Show the sum of those numbers.
(concept = command line arguments , Integer.parseInt(string)

Accept a few strings from command line
Show the length of each string and show each string in uppercase .

Accept one command line argument that is an option 1 or 2 or 3
If the option is 1 show table of 2 upto 100
If the option is 2 show a poem
If the option is 3 show a message to the user
If the option is not given or it is other than 1,2,3---- show incorrect option

---

Int x = 33 y=33

If(x==y) } value is compared because x and y are primitive types

Integer x = 33 , y =33

If(x==y) address is compared because x and y are objects