**Cloning**

>When you want to create the copy of primitive variable then you can use assignment operator(=).

>After creating the copy of primitive variable,if you want to do some modification on copy variable then original value will not be affected.

Ex:- X Y

**10**

**10**

Int x=10;

**//creating Copy**

Int y=x;

**// Modifying using Copy**

**X Y**

**15**

**10**

y=y+5;

>Whenever you are using assignment operator(**=**) to create the copy of reference variable then address of existing object will be copied to the reference variable.

>when you do some modification with Object data using copy reference variable then original object value will be modified because both the reference variable are pointing to same object.

>So we cant create copy of object variable using assignment Operator(**=).**

**Ex:- Stu@100 Stu@100**

S1

name= raushan

Class Student{

String name; **Stu@100**

S2

}

Student s1=new Student();

S1.name=”raushan”;

**//Creating Copy**

Student s2=s1;

**//Modifying data Using Copy Stu@100**

S1

**Stu@100**

Name=Raunak

S2.name=”Raunak”;

**Stu@100**

S2

**Note:-**

>You can use cloning to create the copy of object.

>Cloning is the Process of creating copy of existing Object.

>A process of creating the copy of actual object memory but not its reference variable, is known as cloning mechanism.

>to perform the duplicate copy of any object we need to use the clone() method which is present inside object class.

**Synax:-**

**Protected native Object clone() throws CloneNotSupportedException**

>you can create the copy of existing object by invoking clone() method of object class.

>clone() method is defined as protected in object class.so outside the java.lang package you can access this method in the subclass directly or with that subclass only.

>when you want to clone any class object then you must override clone() method in that class.

>java class objects are not eligible for cloning by default.

>when you want to clone any class object then that class has to implement marker interface called java.lang.Cloneable.

>clone method is throwing checked Exception called CloneNotSupportedException.

>when you try to clone any class object without implementing Cloneable interface then CloneNotSupportedException thrown by the jvm.

**Cloning is classified into Two Types:-**

**1)Shallow Cloning 2)Deep Cloning**

**Shallow cloning:-**

>this is the default implementation of java.lang.Object class clone() method.

>the process of cloning only current object but if any member object is there then it is not eligible for cloning is known as shallow cloning.

>and it will be supported only primitive member is as a instance member of class is there .and it will not supported to create a duplicate copy of object if member object is there and if you modify the member object data then it will affect the original object data.

**Deep Cloning:-**

>The process of cloning current object as well as if any member object is there then it is known as deep cloning.

>Deep Cloning is the best approach if member objects is there.

>There is no default implementation for deep cloning.

>For deep cloning developers must and should have to provide the implementation while overriding the clone() method in the current sub-class.

>if you modify the data of cloned object then it won’t affect the actual object

>if you want to modify the data of the member object then it also won’t affect the other objects.

**Note:-**

>Deep cloning and Shallow cloning can be provided with custom implementation by the developer

>in that custom implementation if you want to force to implement Cloneable interface then we are checking with instanceof operator then in case if you want to ignore Cloneable marker interface then no need to verify with instanceof operator.

>so we can provide custom implementation for both deep and shallow cloning either by forcing to implement Cloneable or by ignoring Cloneable interface implementation but there is a predefined implementation for shallow implementation but not for deep cloning.