**Java Object Creation types**

We can create object in java using the following types

1. new keyword
2. Reflection API
   1. By using forName
   2. By using Constructor
3. Clone
4. Class loader
5. Serialization

Create a class with the name **Demo** which will implement ***Cloneable, Serializable.***

**Demo.java**

package com.learning.object.creation;

import java.io.Serializable;

public class Demo implements Cloneable,Serializable{

public static Long versionID = 123456789L;

public Demo() {}

protected Object clone() throws CloneNotSupportedException {

return super.clone();

}

}

Create a class with method ***main*** and implement different ways of creating of object.

**DemoMain.java**

package com.learning.object.creation;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.lang.reflect.Constructor;

import java.lang.reflect.InvocationTargetException;

public class DemoMain {

public static void main(String[] args) {

**// 1. Creating object using new key word**

System.out.println("Creating object using new key word...");

Demo objCreatedByNew = new Demo();

System.out.println(objCreatedByNew);

**// 2. Creating object using reflection**

**/\***

**\* While using reflection we can create object in 2 ways**

**\* 1.forname()**

**\* 2.constructor()**

**\*/**

System.out.println("---------------------------------------");

System.out.println("Creating object using Reflection -- forName");

try {

Demo objCreatedByForName = (Demo)

Class.forName("com.learning.object.creation.Demo").newInstance();

System.out.println(objCreatedByForName);

} catch (InstantiationException | IllegalAccessException | ClassNotFoundException e) {

e.printStackTrace();

}

System.out.println("Creating object using Reflection -- Constructor");

Constructor<Demo> constructor = (Constructor<Demo>)

Demo.class.getConstructors()[0];

Demo objCreatedFromConstructor = null;

try {

objCreatedFromConstructor = (Demo) constructor.newInstance();

} catch (InstantiationException | IllegalAccessException | IllegalArgumentException

| InvocationTargetException e) {

e.printStackTrace();

}

System.out.println(objCreatedFromConstructor);

**// 3. Creating object using clone**

System.out.println("---------------------------------------");

System.out.println("Creating object using clone");

Demo objectCreatedByClone = null;

try {

objectCreatedByClone = (Demo) objCreatedByNew.clone();

} catch (CloneNotSupportedException e) {

e.printStackTrace();

}

System.out.println(objectCreatedByClone);

**// 4. Creating object using class loader**

System.out.println("---------------------------------------");

System.out.println("Creating object using class loader");

try {

Demo objectCreatedByClassLoader = (Demo) new DemoMain().getClass().getClassLoader()

.loadClass("com.learning.object.creation.Demo").newInstance();

System.out.println(objectCreatedByClassLoader);

} catch (InstantiationException | IllegalAccessException | ClassNotFoundException e) {

e.printStackTrace();

}

**// 5. Creating object using serialization**.

System.out.println("---------------------------------------");

System.out.println("Creating object using serialization");

try {

ObjectOutputStream oos = new ObjectOutputStream(new

FileOutputStream("serializedObjectFile.txt"));

oos.writeObject(objectCreatedByClone);

oos.close();

ObjectInputStream ois = new ObjectInputStream(new FileInputStream("serializedObjectFile.txt"));

Demo objectCreatedBySerialize = (Demo) ois.readObject();

ois.close();

System.out.println(objectCreatedBySerialize);

} catch (IOException | ClassNotFoundException e) {

e.printStackTrace();

}

}

}