

28/04/2018

CLASSTIME	PAGE NO.
Date	/ /

2. CLONING

Rule

Only the objects of those classes can be cloned which are implementing a Cloneable Interface and it is a marker interface.

```
class Demo
{
    implement Cloneable
    int x = 10;
    int y = 20;
```

```
Temp t = new Temp();
```

```
public Demo Clone()
```

```
{
    Demo z = (Demo) super.clone();
    Refer z;
}
```

```
class Temp implements Cloneable
{
    int z = 90;
}
public Temp Clone()
{
    Temp t1 = new Temp()
    super.clone();
    return t1;
}
```

```
psvm()
```

```
{
    Demo d1 = new Demo();
```

Shallow

```
↑
Demo d2 = d1.clone();
```

x = 10
y = 20
t = 6

z = 90

x = 10
y = 20
t = 6

```
d2.t = d1.t.clone();
↓
Deep;
```

z = 90

Program (MyClone.java) → Shallow Cloning.

```
class Temp
```

```
{
```

```
int g;
```

```
Temp (int g)
```

```
{
```

```
this.g = g;
```

```
}
```

```
}
```

```
class MyClone implements Cloneable
```

```
{
```

```
int x = 10;
```

```
Temp t;
```

```
MyClone (int a)
```

```
{
```

```
t = new Temp(a)
```

```
}
```

```
public MyClone clone ()
```

```
{
```

```
//shallow copy
```

```
try {
```

```
return (MyClone).super.clone();
```

```
} catch (CloneNotSupportedException e) {
```

```
return null
```

```
}
```

```
}
```

psvm()

{

MyClone c = null;

MyClone m = new MyClone(100);

System.out.println("m.t.g" + m.t.g);

System.out.println("m.x" + m.x);

try

{

c = m.clone();

> catch (Exception e) {}

Sop("c" + c.t.g);

Sop("c.x" + c.x);

c.t.g = 3000;

c.x = 30;

System.out.println("m after changes" + m.t.g);

System.out.p ln ("m after " + m.x);

if (m.t == c.t)

{

Sop("shallow cloning");

>

>

>

Program (DeepMyClone.java) → Deep Cloning

class Temp implements Cloneable

```
{
    int g;
    Temp(int g)
```

```
{
    this.g = g;
```

```
}
    public Temp clone()
```

```
{
    try
```

```
{
    return (Temp) super.clone();
```

```
}
    catch (Exception e) { return null; }
}
```

class DeepMyClone implements Cloneable

```
{
    int x = 10;
```

```
    Temp t;
```

```
    DeepMyClone(int a)
```

```
{
```

```
    t = new Temp(a);
```

```
}
```

```
    public DeepMyClone() {
        // deep copy
        try {
```

```

DeepMyClone d = (DeepMyClone) super.clone();
do d.t = t.clone();
return d;
} catch (Exception e) { return null; }

```

}

psvm()

```

DeepMyClone e = null;
DeepMyClone m = new DeepMyClone(100);
Sop("m" + m.t.g);
try

```

```

can c = m.clone();
// c.t = m.t.clone();

```

```

} catch (Exception e) {
Sop("c" + c.t.g);
c.t.g = 300;
Sop("m after changes" + m.t.g);
if (m.t != c.t)

```

```

{
Sop("deep cloning");
}
}
}

```