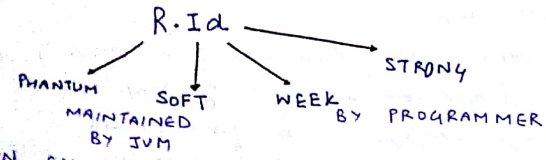
23/09/2017

## → RUN-TIME CLASS

To represent Java's Run-Time instance, object of Run-Time class is created.
[created by JVM.]
It works as a mediator b/w JVM and our java program
To fetch information from JVM at the Run-Time of a Program, we use Run-Time Class

```
class Temp()          STATIC METHOD
                      OBJECT MADE BY JVM
{ public static void main()
{ Runtime rt = Runtime.GetRuntime();
  sop(rt.totalmemory()).
  sop(rt.freememory()).
}}
```

TOTAL HEAP SIZE
MEMORY FREE IN HEAP

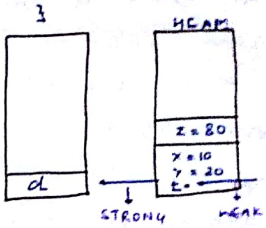



RUN BY JVM [↓
[LIKE FINALIZE METHOD WHICH HAS TO BE RUN RId not ON OBJECT]



HAS A RELATIONSHIP

```
class Demo                    class Temp
{ int x = 10;                 { int z = 80;
  int y = 20;                 }
  Temp t = new Temp();
    WEAK
  PSVM()
  { Demo d = new Demo();
      STRONG
  }
}
```




EXPLANATION

x, y shows has a relationship - ASSOCIATION, [one object has another object]
When we create object of demo class, d goes into heap [object] Data Member get memory; x, y and t [Reference Variable 4 byte] Then object of class Temp gets memory in heap as 'z'. Then Demo class Object gets created and [d is having all reference of variables in heap]
Strong Reference gets direct address and Weak References gets in-direct address

There are 4 types of references to objects

1. Direct References like we normally use, as in integers (= new integer(13)) are called strong References and have no special class. The remaining three are soft references, Weak References (W.R) and Phantum References (PoR)