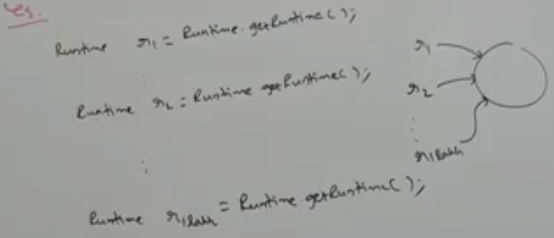
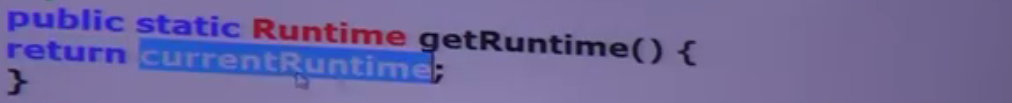
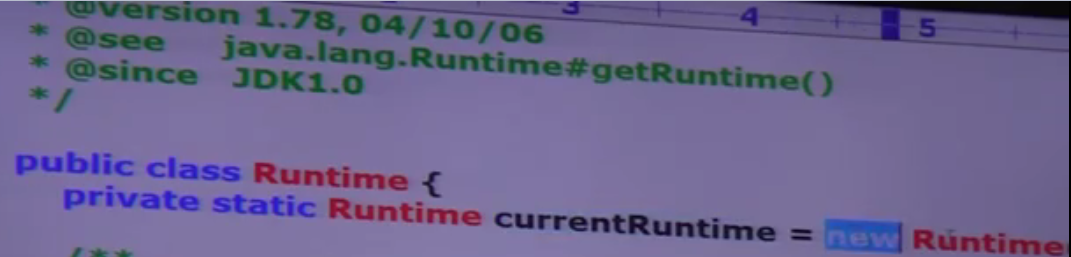
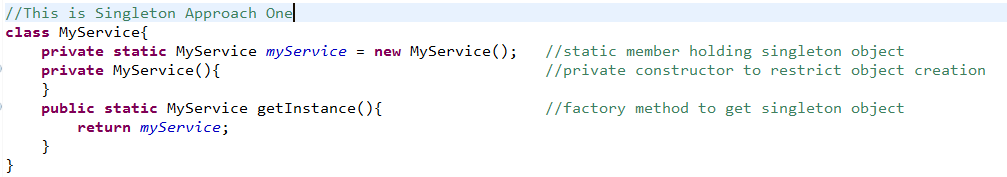
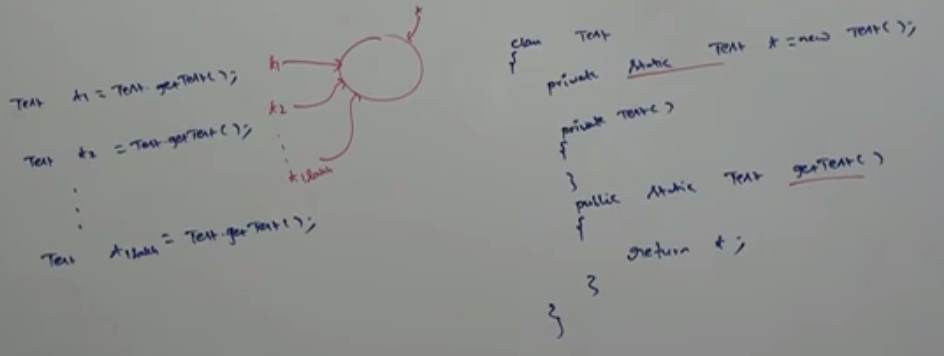
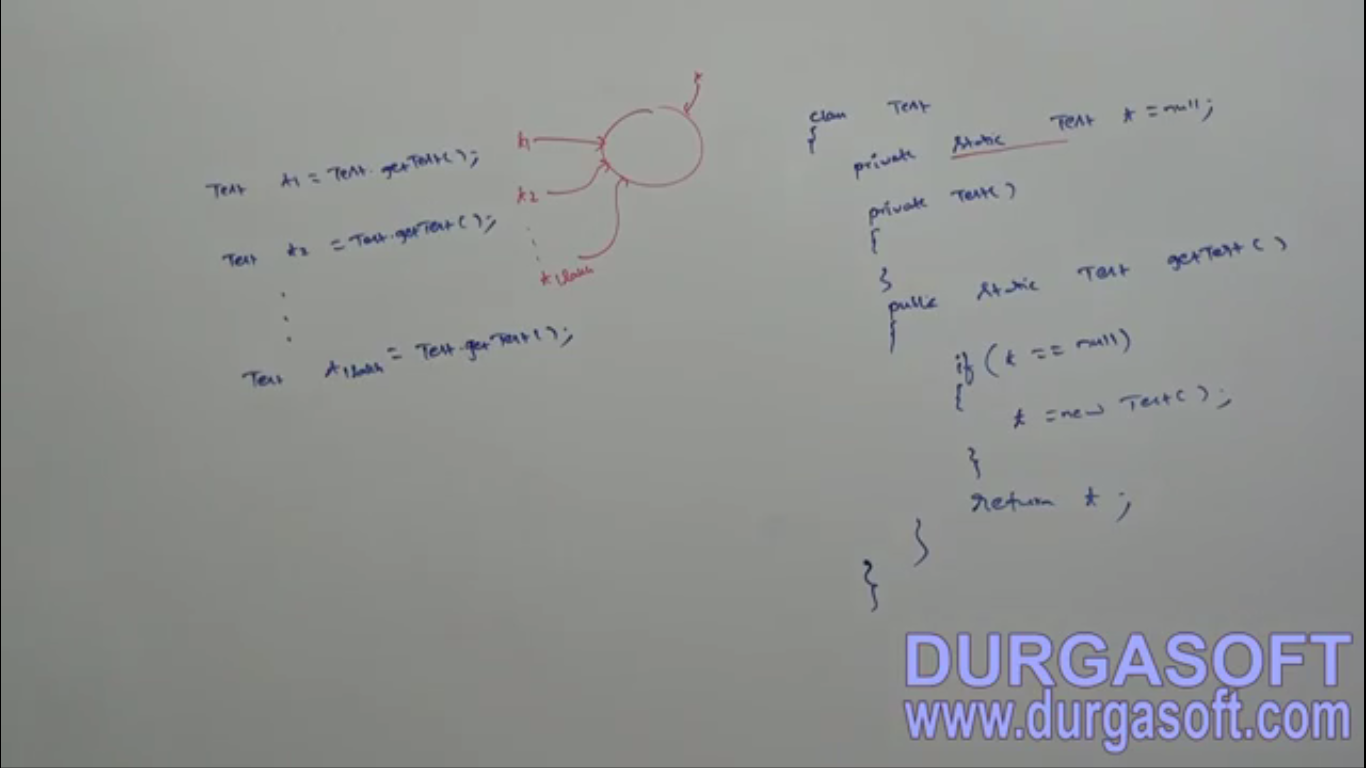
Singleton Classes

1. For any java class, if we are allowed to create only object such type of class is **called Singleton class**.
   1. **Example**:
      1. Runtime
      2. BusinessDelegator
      3. ServiceLocator
   2. **Scenario**:
      1. If several people have same requirement, then it’s not recommended to create a separate object for every requirement. We have to create only one object and we can reuse the same object for every similar requirement so that performance and memory utilization can be improved. This is the central idea of Singleton Class.
2. **Advantages**:
   1. Memory Utilization.
   2. Performance improved.
3. **How to create singleton object?**
   1. Using factory method.
   2. Runtime r1 = Runtime.getRuntime(); // **To communicate with JVM  
      Runtime r2 = Runtime.getRuntime(); // the same above object**
4. **How to create our own singleton class?**
   1. Let’s take a look at Runtime code in JDK  
        
      We can create our own singleton class. For this, we have to use 🡨 This is 1st approach
      1. **private constructor** and
      2. **private static variable** and
      3. **public factory method**.

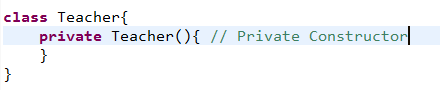


By Durga🡪   
  
In the 1st approach, singleton object is created no matter, it is requested or not as when class is loaded, object is created as the creation statement is an expression assigned to static member.

* 1. **Let’s talk about 2nd approach**

****

## Class is not final but we are not allowed to create child classes, how is it possible? Or Without using final, how we can restrict the creation of child class?

* Declare every constructor of the class private, we can restrict child class creation of this class.   
    
  **NOTE**: for the above class, it’s impossible to create child class.

## 