**Singleton Design Pattern**

Singleton design pattern is used to create one and only one instance of a class.

**Intent**

* Ensure a class only has one instance, and provide a global point of access to it.

The intent is to have only one instance of a class and it should be accessible to multiple clients through well-defined access point.

**Problem**

* Application requires only one instance of the class and that instance of class should be accessible to multiple clients.

There are scenarios where one and only one instance of class is required, for example File System, System Manager etc and it has to be accessed from multiple applications. Introducing of global variable provides access to multiple applications but does not make sure that there will be only one instance.

**Solution**

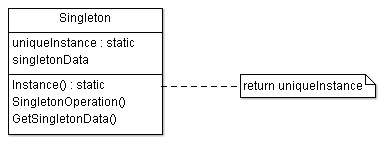
* Make sure only one instance of class is created and the same one is accessible to multiple clients.

The class itself can take the responsibility of creation of not more than one instance and can provide the well-defined access point for multiple applications.

**Where it is applicable?**

* One and only one instance of a class is required and well-defined access point is provided to access it by multiple applications.

**Structure**



**Participant classes**

* **Singleton** class has method Instance() which has responsibility to create single instance and is access point for providing the instance to multiple applications. The class has data member uniqueInstance which keeps the instance information.

**How they work together?**

* Client uses Instance() method of Singleton class to get the instance of this class. The Instance() method checks for the instance, if instance is already created then it returns the same, otherwise it creates the instance and returns it to the client.
* The access of instance of class is controlled and well-defined. It also provides scope to have variable number of instance with small modification, if required in some scenario. The constructor has to be protected so that it cannot be instantiated directly by client. Here Instance uses lazy initialization as instance is created when it is first used. The subclassing can be supported in multiple ways, one way may be to register the instances and provide them when it is required by returning the appropriate one.