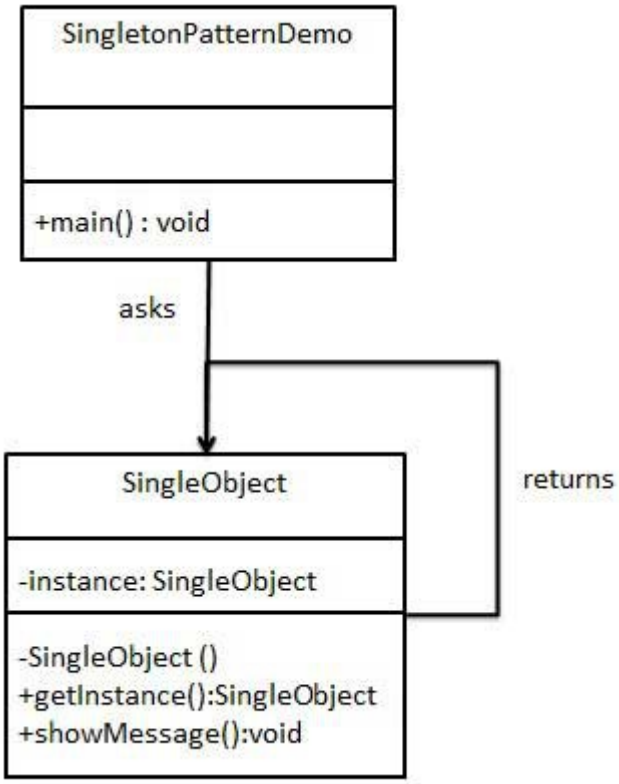
Singleton Pattern

This pattern involves a single class which is responsible to creates own object while making sure that only single object get created. This class provides a way to access its only object which can be accessed directly without need to instantiate the object of the class.

We're going to create a *SingleObject* class. *SingleObject* class have its constructor as private and have a static instance of itself.

SingleObject class provides a static method to get its static instance to outside world. *SingletonPatternDemo*, our demo class will use *SingleObject* class to get a *SingleObject* object.



Create a Singleton Class.

SingleObject.java

```
public class SingleObject {

   //create an object of SingleObject
   private static SingleObject instance = new SingleObject();

   //make the constructor private so that this class cannot be
   //instantiated
   private SingleObject(){}

   //Get the only object available
   public static SingleObject getInstance(){
      return instance;
   }

   public void showMessage(){
      System.out.println("Hello World!");
   }
}
```

Get the only object from the singleton class.

SingletonPatternDemo.java

```
public class SingletonPatternDemo {
   public static void main(String[] args) {

      //illegal construct
      //Compile Time Error: The constructor SingleObject() is not visible
      //SingleObject object = new SingleObject();

      //Get the only object available
      SingleObject object = SingleObject.getInstance();

      //show the message
      object.showMessage();
   }
}
```