Name: **ZOHAIB HASSAN SOOMRO**

RollNo#: **19SW42**

Subject: **SDA**



Singleton

* Definition: "The Singleton Pattern ensures a class has only one instance, furthermore, gives a global point of access to it."
* Best Uses
* Logging
* Caches
* Registry Settings
* Access External Resources
* Printer
* Device Driver
* Database

Example: Logger

What is wrong with this code?

public class Logger{

public Logger ( ){ }

public void LogMessage ( ){

// Open Filelog . txt"

/ / Write Message

// Close File

}

}

# Example: Logger

* Since there is an external Shared Resource ("log.txt"), we need to intently control how we communicate with it.
* We shouldn't need to make the Logger class each time we need to get to this Shared Resource. Is there any motivation to?
* We need ONE.

Logger — as a Singleton

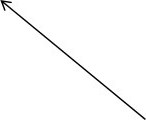
public class Logger{

private Logger { }

private static Logger unique Instance;

public static Logger get Instance ( ){

if (unique Instance null)

unique Instance new Logger ( ) ;

return unique Instance;

}

}

Note the parameterless

Constructor

## Lazy Instantiation

* Instances are only created when it is required.
* Helps in controlling that we've constructed the Singleton only one time.
* If it is asset intensive to set up, we need to do it only one time.

# Singleton Consequences

* Controlled access to sole occurrence works with severe power over when and how the customers access it.
* The singleton patter is improvement over global factors.
* It is not difficult to design an instance of the application that broadens the usefulness of singleton at run-time.
* More adaptable than class operations.

## Singleton Limitations

* The primary limitation of the singleton pattern is that is licenses the formation of just one occasion of the class, while most commonsense applications require various cases to be introduced.
* Moreover, in the event of singleton, the framework strings battle to get to the single example accordingly corrupting the exhibition of the applications.

### Examples

**Cache** — The cache can be used as a singleton object, having a global perspective for all future references to the cache object and the customer application utilizes this in-memory object.

**Configuration File** — Any application has a solitary perspective through its configuration documents. Any progressions to be made to the application should be made by means of this single configuration document.