UKE 11

Gruppe 9!

raghninw@uio.no

Programmeringsmønstre

- Behavioral - atferdsmessig
- Structural - strukturelle
- Creational - skapende

Iterator - atferdsmessig

- Interface iterable
- hasNext() og next()

Iterator - implementasjon

- Ytre klasse implementerer interfacet *Iterable*
  - *Iterator<T> iterator()*
  - *Iterator()* skal returnere en instans av den egenlagde iterator-klassen

- En indre klasse som implementerer interface *Iterator*
  - *hasNext()*
  - *next()*

Decorator - Strukturelt

- Brukes for å endre egenskaper under kjøring
- Kan også kalles wrapper

Består av:

- Interface for egenskaper
- Basisklasse som implementerer interfacet
- En abstrakt klasse som implementerer interfacet (decorator)
- Konkrete klasser med ekstra egenskaper som utvider decorator

Observatør-mønsteret - atferdsmessig

- Et subjekt observeres
- En observator får beskjed når noe endres
- Subjekt kan også kalles publisher
- Observatør kan også kalles subscriber

MVC - atferdsmessig

- Model
- View
- Controller

- Kan se på View som observatør og Model som subjekt
- Brukes mye i GUI!

Singleton - Skapende

- Ett eneste objekt av en gitt type i programmet
- Kontrollere tilgang til sikkerhetskritiske ressurser, f.eks. en database
- Begrense ressurskrevende objekter til én instans, f.eks. sentralisert logging av ønskede og uønskede hendelser under programkjøring
- Trådsikkerhet er ofte en forutsetning når vi bruker Singleton-mønsteret

```java
public final class LoggerSingleton {
    // Single instance
    private static LoggerSingleton instance = null;
    // Prevent instantiation from other classes by making constructor private
    private LoggerSingleton() {}

    // Method to get the single instance
    public static LoggerSingleton getInstance() {
        if (instance == null) instance = new LoggerSingleton();
        return instance;
    }

    // Method to demonstrate Singleton usage logging messages to the terminal
    public void logMessage(String message) {
        System.out.println("Log Message: " + message);
    }
}
```