IN1010 - Seminar 14

- GUI

Laget av: Marlen Jarholt (marleja@ifi.uio.no) og Johanna Haarseth (johannph@ifi.uio.no)

Praktisk

Husk å sjekke emnesiden regelmessig

Undervisningstilbud:

- https://www.uio.no/studier/emner/matnat/ifi/IN1010/v21/undervisningstilbud/
- Jeg har konkrete spørsmål/problemer med min kode -> Labtime!
- Jeg vil ha mer liveprogrammering -> Plenumstime!
- Jeg vil jobbe med andre (og kanskje en kjapp recap av forelesning) -> Gruppetime!
- Jeg vil ha en recap av de vanskeligste konseptene fra forelesning -> Repetisjonsgruppe!

Juks - kopiering av kode

Hovedpoenget med obligene er at dere skal øve. Man kan ikke bli utvikler uten å progge.

Er man stuck burde man finne ut nøyaktig hva som er vanskelig og så googel det/tenke på det/spørre noen om akkurat det. I jobb er det ingen som sitter med en fasit dere kan titte litt på. En av de _viktigste_ skillsene for å bli god utvikler er å kunne google.

Ser dere på koden til andre som har løst obligen mister dere masse læringspotensial.

Kopiering av kode er juks. Og det går bare utover den som kopierer, ikke den som deler kode.

Send Marlen en direktemelding i chatten

Vil du jobbe sammen med noen andre ? (ja /nei)

Hvis du har noen ønsker på hvem du vil jobbe med, så send det i samme melding

Svar gjerne også om svaret skulle være nei

Repetisjon forrige uke

S O L I D

- **Single responsibility principle**
  En klasse bør bare ha ett ansvarsområde, og dette bør være veldefinert
- **Open/closed principle**
  Komponenter bør være åpne for utvidelser, men lukket for modifikasjoner. Eksempel: klasser kontrollerer hva subklasser kan endre (private, protected, final, etc).
- **Liskov substitution principle**
  Objekter i et program bør kunne byttes ut med instanser av subtyper uten at dette endrer programmets korrekthet.
- **Interface segregation principle**
  Mange små grensesnitt er bedre enn store, monolittiske "super-grensesnitt"
- **Dependency inversion principle**
  Komponenter bør avhenge av abstraksjoner, og ikke konkrete implementasjoner

Enhetstesting

Teste komponenter av koden hver for seg, f.eks. en metode eller en klasse.

Etter enhetstesting kan vi sette komponentene sammen og teste hvordan det fungerer sammen (integrasjontest) og etterhvert hele systemet (systemtesting). Men i in1010 lærer vi bare litt enhetstesting.

Poenget er å finne feil så tidlig som mulig. Dere har sikkert selv erfart at det er vanskeligere å finne feil dess mer kode det er.

Repetisjon denne uka

Import + basics

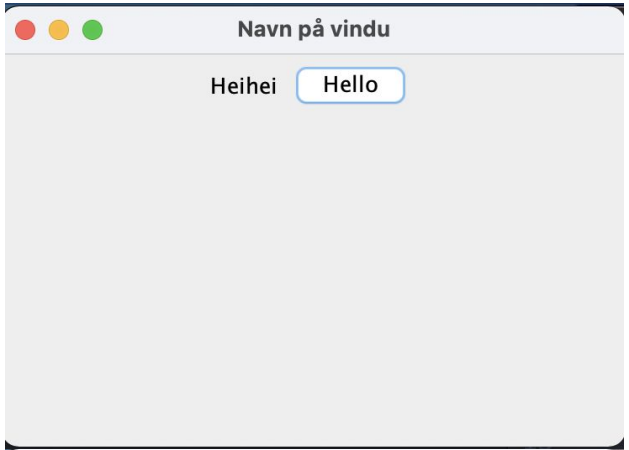
```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class TestGraphic{
  public static void main(String[] args) {
    JFrame vindu = new JFrame("Navn på vindu");
    vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel panel = new JPanel();
    vindu.add(panel);

    vindu.pack();
    vindu.setVisible(true);
  }
}
```

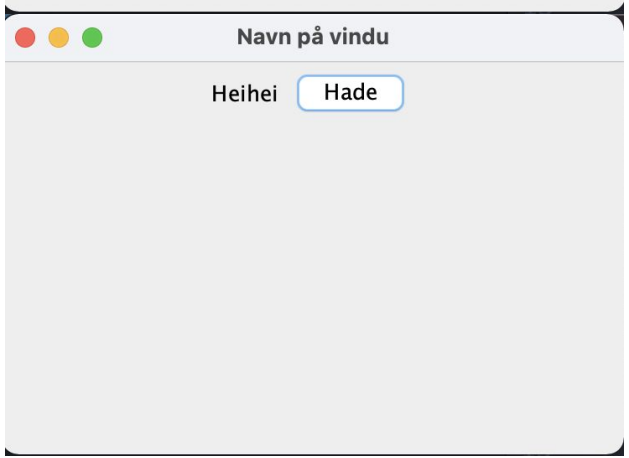
Tekst

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class TestGraphic{
  public static void main(String[] args) {
    JFrame vindu = new JFrame("Navn på vindu");
    vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel panel = new JPanel();
    vindu.add(panel);

    JLabel tekst = new JLabel("Heihei");
    panel.add(tekst);

    vindu.pack();
    vindu.setVisible(true);
  }
}
```

Knapper

```
5  class TestGraphic{
6    public static void main(String[] args) {
7      JFrame vindu = new JFrame("Navn på vindu");
8      vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9
10     JPanel panel = new JPanel();
11     vindu.add(panel);
12
13     JLabel tekst = new JLabel("Heihei");
14     panel.add(tekst);
15
16     JButton knapp = new JButton("Hello");
17     panel.add(knapp);
18
19
20     vindu.pack();
21     vindu.setVisible(true);
22
23     class Knapp implements ActionListener{
24       @Override
25       public void actionPerformed(ActionEvent e){
26         knapp.setText("Hade");
27       }
28     }
29     knapp.addActionListener(new Knapp());
30
31   }
```

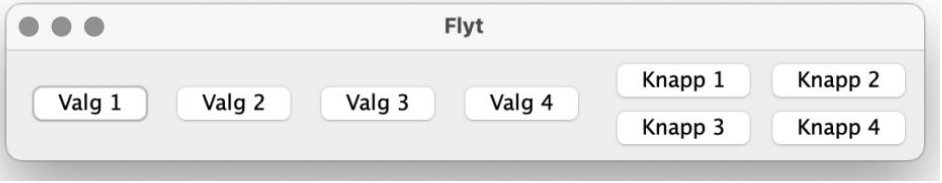




FlowLayout vs GridLayout

FlowLayout er default, layout for et JPanel kan endres med .setLayout().

```
JPanel flyt = new JPanel();
flyt.add(new JButton("Valg 1"));
flyt.add(new JButton("Valg 2"));
flyt.add(new JButton("Valg 3"));
flyt.add(new JButton("Valg 4"));
panel.add(flyt);

JPanel ruter = new JPanel();
ruter.setLayout(new GridLayout(2,2));
ruter.add(new JButton("Knapp 1"));
ruter.add(new JButton("Knapp 2"));
ruter.add(new JButton("Knapp 3"));
ruter.add(new JButton("Knapp 4"));
panel.add(ruter);
```



Ris, ros, forslag ?

https://nettskjema.no/a/180345

Breakoutrooms

1. Slå på kamera og ha en presentasjonsrunde
2. Jobb sammen med ukesoppgavene, de ligger på emnesiden -> grupper
    a. Enten ved at én deler skjerm eller med codecollab.io/
    b. OBS: codecollab er gratistjeneseter som UiO ikke har avtale med, sannsynligvis vil de samle data om dere. Dere kan fint løse oppgavene uten å bruke den tjenesten!
3. Bruk “ask for help”-knappen for å få hjelp 😊
4. Vi møtes her igjen for å gå gjennom oppgavene til slutt (dere bestemmer hvilke)

Jobbe med oppgaver