Pragmatisk programmering

GUI

Hva er GUI?

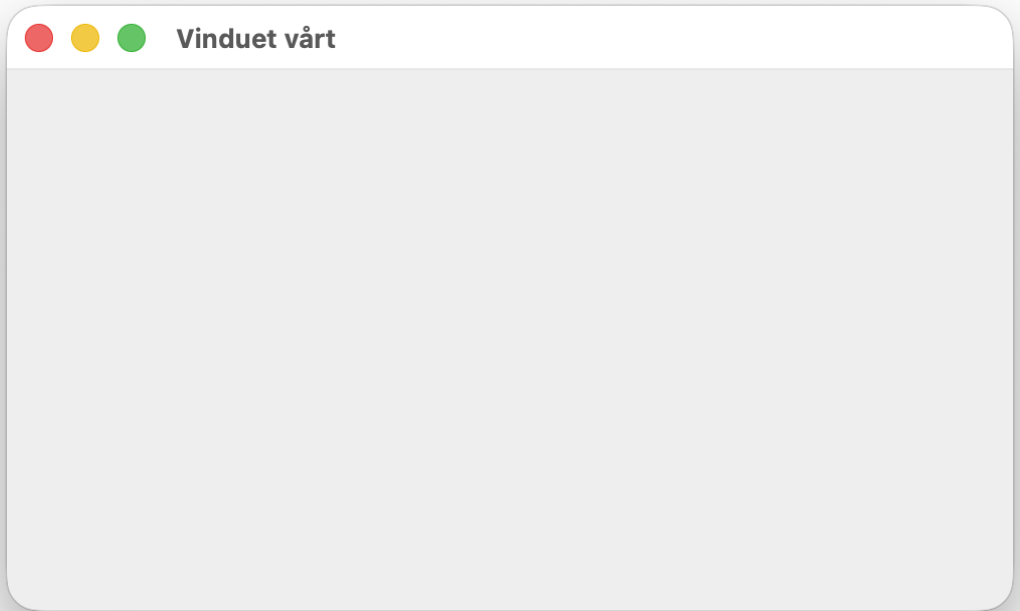
- Står for Graphical User Interface
- Gir et visuelt grensesnitt som bruker kan interagere med
  - Mer brukervennlig enn før, da alt har gått via terminalen
- I IN1010 bruker vi AWT + Swing
  - Mye som må importeres

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

public class GUIEksempel {
    public static void main(String[] args) {
        JFrame vindu = new JFrame( title: "Vinduet vårt");
        vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        vindu.setPreferredSize(new Dimension( width: 500,  height: 300));

        try {
            UIManager.setLookAndFeel(
                    UIManager.getCrossPlatformLookAndFeelClassName()
            );
        } catch (Exception e){
            System.exit( status: 1);
        }

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



Noen objekter

- `JFrame`: Vinduet som opprettes, og som brukeren ser
- `JPanel`: En “tegneflate”, som andre objekter kan legges inni
- `JLabel`: Et objekt med tekst, som vi kan legge inni et panel
- `JButton`: En trykkbar knapp; vi kan spesifisere handlingen som skjer når knappen trykkes på

```java
JFrame vindu = new JFrame( title: "Vindu med tekst og knapp"); // deklarerer vinduet vårt, parameter er navn på vindu
vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // angir at programmet skal stoppe når vinduet lukkes
vindu.setPreferredSize(new Dimension( width: 500,  height: 300)); // angir størrelse på vinduet i piksler

try {
    // angir standard Swing-utseende, ellers forsøker Swing å lage grafikk som ligner OS-ets
    UIManager.setLookAndFeel(
            UIManager.getCrossPlatformLookAndFeelClassName()
    );
} catch (Exception e){
    System.exit( status: 1);
}

JPanel panel = new JPanel(); // oppretter tegneflaten vår
JLabel tekst = new JLabel( text: "Dette er tekst"); // oppretter en tekst
JButton knapp = new JButton( text: "Dette er en knapp"); // oppretter en knapp

// setter panelet inn i vinduet, og alt annet i panelet
vindu.add(panel);
panel.add(tekst);
panel.add(knapp);

vindu.pack(); // pakker alt innholdet i vinduet pent sammen
vindu.setLocationRelativeTo(null); // sørger for at vinduet plasseres midt på skjermen
vindu.setVisible(true); // gjør vinduet med alt innhold synlig (må gjøres ETTER pakking!)
```



Layout for JPanel

- Hvis ingenting annet er spesifisert, så vil elementer i JPanel legges til etter hverandre
- Man kan spesifisere layout til JPanel hvis man ønsker
  - **BorderLayout**: Man kan plassere elementer nord, sør, øst, vest eller i midten
  - **GridLayout**: Spesifiserer hvor mange rader og kolonner man vil ha, elementer legges til i et rutenett
  - **FlowLayout**: Elementer legges til i rad, så lenge det er plass

Hendelser

- Akkurat nå gjør knappene ingenting når de blir trykket på
- For å bestemme handlinger, kan vi definere en klasse som implementerer grensesnittet `ActionListener`
- Denne klassen venter på en hendelse (f.eks. trykk av en knapp), og kjører metoden `actionPerformed` når hendelsen skjer

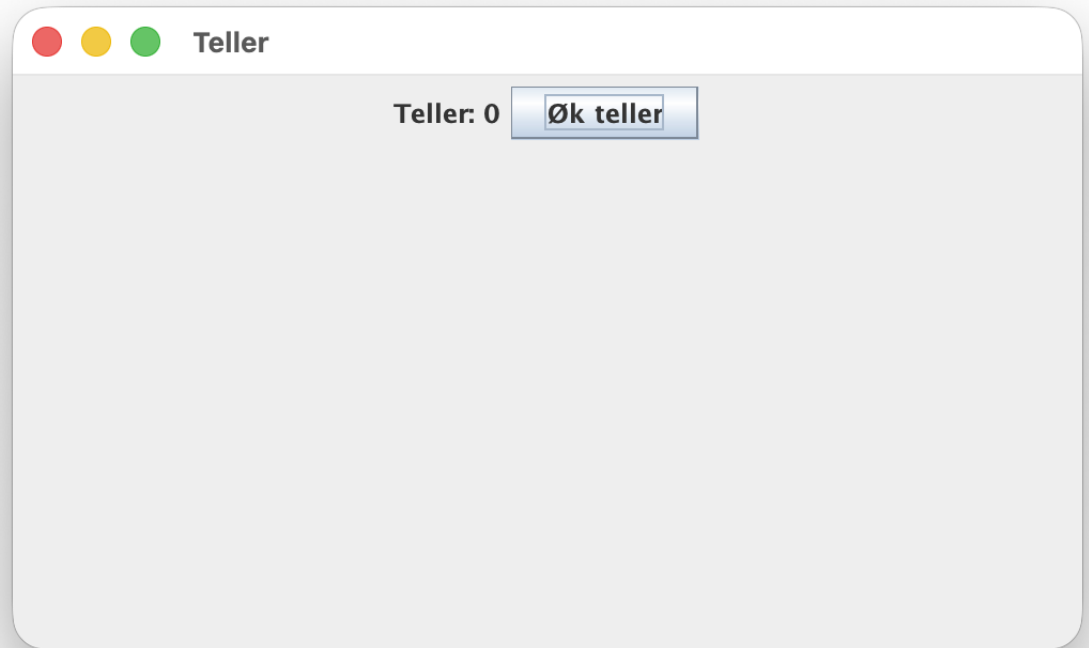
```java
private static int teller = 0;  3 usages

public static void main(String[] args) {
    JFrame vindu = new JFrame( title: "Teller");
    vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    vindu.setPreferredSize(new Dimension( width: 500,  height: 300));

    try {
        UIManager.setLookAndFeel(
                UIManager.getCrossPlatformLookAndFeelClassName()
        );
    } catch (Exception e){
        System.exit( status: 1);
    }

    JPanel panel = new JPanel();
    JLabel tekst = new JLabel( text: "Teller: " + teller);
    JButton knapp = new JButton( text: "Øk teller");

    knapp.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            teller++;
            tekst.setText("Teller: " + teller);
        }
    });
```



Største takeaway: TA MED GODE NOTATER PÅ EKSAMEN