GUI («Graphical User Interface»)

- Om GUI
- AWT og Swing
  - Hvordan lage et GUI-vindu
  - Hvordan lage en trykknapp
  - Et quiz-program

Se også på

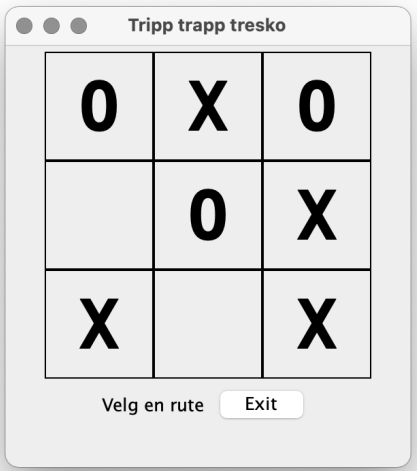
- Big Java kapittel 10–11
- Programkoden til eksemplene ligger i https://www.uio.no/studier/emner/matnat/ifi/IN1010/v21/programmer/GUI/
- Et godt nettkurs fra Oracle: https://docs.oracle.com/javase/tutorial/uiswing/index.html

Tripp-trapp-tresko med og uten GUI

```
+---+---+---+
| 1 | 2 | 3 |
+---+---+---+
| 4 | 5 | 6 |
+---+---+---+
| 7 | 8 | 9 |
+---+---+---+
```

```
$ java TTT1
X spiller 8
Hva spiller O? 5
O spiller 5
X spiller 1
Hva spiller O? 3
O spiller 3
X spiller 2
Hva spiller O? 7
O spiller 7
Vinneren er O!
```

```
$ java TTT2
+---+---+---+
| X |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
Hva spiller O? 5
+---+---+---+
| X |   |   |
+---+---+---+
|   | O |   |
+---+---+---+
|   |   | X |
+---+---+---+
Hva spiller O? 8
      :
Hva spiller O? 2
Vinneren er O!
+---+---+---+
| X | O |   |
+---+---+---+
|   | O |   |
+---+---+---+
| X | O | X |
+---+---+---+
```



Fordeler med GUI

- Mer intuitivt å bruke
- Færre muligheter for brukerfeil
- Visuelt mer tiltalende

Ulemper

- Mer komplisert å programmere
- Mange ulike GUI-biblioteker å velge blant
- Svært få biblioteker fungerer for både Linux, Mac og Windows.

Historien om Qt

- Opprinnelig laget av det norske firmaet *Trolltech* ved Sannerbrua i Oslo i 1994.
- Hovedideen var at programmer skrevet i C++ kunne linkes med et Qt-bibliotek og så kjøre på ulike systemer.
- Sannsynligvis det mest brukte *generelle* vindusbiblioteket i 1990- og 2000-årene.
- Solgt til Nokia i 2008 (og siden til Digia og The Qt Company).
- Finnes i både kommersiell og åpen kildekode-versjoner.
- Brukes i dag i Google Earth, Walt Disney animation studios, Tesla-biler, . . .

Hvorfor GUI i IN1010?

- Allmennkunnskap for programmerere
- Et godt eksempel på bruk av OO
- Viser en ny programmeringstankegang:
  Hendelsesorientert programmering
- Et godt eksempel på bruk av parallellisering og tråder

Java og GUI

Java har alltid hatt som mål at programmene skal kunne kjøres uendret på alle platformer.

AWT («Abstract Window Toolkit») fra 1995:
et ganske enkelt grensesnitt mot OSets vindussystem

Swing fra 2007:
et mer avansert og generelt system for vinduer, trykknapper etc bygget oppå AWT

JavaFX fra 2012:
et enda mer komplett og velstrukturert system

Fra 2018: JavaFX ikke lenger en del av standard Java men et frittstående produkt.

GUI i IN1010

I IN1010 bruker vi **AWT+Swing** selv om JavaFX er mer moderne.

- I læreboken brukes AWT+Swing.
- AWT+Swing er en del av standard Java; JavaFX må installeres separat, og mange studenter hadde problemer med det i fjor.
- Alle GUI-mekanismene vi skal bruke, finnes i AWT+Swing; utvidelsene i JavaFX ville vi ikke benyttet uansett.
- FavaFX er generelt vanskeligere å programmere.

Demo 1: Et minimalt GUI-program

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

class Mini {
    public static void main (String[] args) {
        JFrame vindu = new JFrame("Xxx");
        vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        vindu.add(panel);

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

Importere klasser

Disse klassene dekker vårt behov i IN1010:

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

Deklarere vinduet

Aller først må vi deklarere vinduet vårt; det er en **JFrame**:

```
JFrame vindu = new JFrame("Xxx");
vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

NB!

Vi *må* angi at programmet vårt skal stoppe når vinduet lukkes.

Opprett tegneflaten

Nå kan vi opprette tegneflaten vår (en **JPanel**) og sette den inn i vinduet.

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

Gjør vinduet synlig

Til sist må vi huske på to ting:

1. Pakk alt innholdet i vinduet vårt pent sammen:

   ```
   vindu.pack();
   ```

2. Gjør vinduet med alt innhold synlig:

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

Kjøring

Grafiske programmer trenger mer støtte ved kjøring fra systemet enn våre tidligere tekstorienterte programmer.

Kjøring på privat maskin

Det enkleste er å installere Java på egen maskin (se nettsiden). Når man installerer vanlig JDK, får man alltid med AWT+Swing.

Kjøring på en Ifi-maskin

Det er også mulig å kjøre GUI-programmene på en Ifi-maskin.

- Fra Windows: Åpne **view.uio.no** i en nettleser; velg *Ifi Workstation*. (*UiO Windows Desktop* har ikke Java.)
- Fra Mac: Installer **XQuartz** fra **//www.xquartz.org/** eller bruk view.uio.no.
- Fra Linux: Gi kommandoen **ssh -Y login.ifi.uio.no**

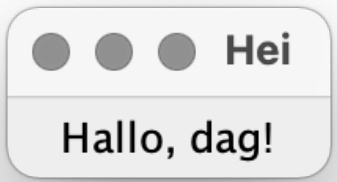
Demo 2: Bokser og slikt

På tegneflaten (JPanel-objektet) kan vi plassere ulike «bokser»:

- tekst (JLabel)
- trykknapper (JButton)
- tekstfelt (JTextField)
- tegneflater (JPanel)
- ...

La oss lage et program som ønsker oss velkommen:

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

class Hallo {
    public static void main (String[] arg) {
        JFrame vindu = new JFrame("Hei");
        vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        vindu.add(panel);

        String bruker = System.getProperty("user.name");
        JLabel hilsen = new JLabel("Hallo, " +  bruker + "!");
        panel.add(hilsen);

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



Det nye her er:

1. Vi ber Java om brukernavnet til den som kjører programmet:

```
String bruker = System.getProperty("user.name");
```

2. Vi lager en JLabel med hilsenen og legger den på vårt JPanel:

```
JLabel hilsen = new JLabel("Hallo, " +  bruker + "!");
panel.add(hilsen);
```

Struktur

Vi har altså en

- en JFrame som inneholder
  - et JPanel som inneholder
    - en JLabel.

Slik bygges et GUI-vindu opp.

Trykknapper

Den vanligste formen for interaksjon med en bruker er **trykknapper**. Når man skal la en slik, må man:

1. opprette trykknappen (en **JButton**)
2. sette den på en tegneflate (en **JPanel**)
3. skrive kode som skal utføres ved et trykk (en **ActionListener**)
4. koble koden til trykknappen (med **addActionListener**)

Et program som stopper seg selv

La oss lage et program som kun stopper seg selv.

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

class Exit {
    public static void main (String[] arg) {
        JFrame vindu = new JFrame("Exit");
        vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        vindu.add(panel);

        JButton exitKnapp = new JButton("Exit");
        class Stopper implements ActionListener {
            @Override
            public void actionPerformed (ActionEvent e) {
                System.exit(0);
            }
        }
        exitKnapp.addActionListener(new Stopper());
        panel.add(exitKnapp);

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

1: opprette knappen

En trykknapp er en JButton med angitt tekst:

```
JButton exitKnapp = new JButton("Exit");
```

2: plassere på tegneflaten

Knappen må settes på en flate:

```
panel.add(exitKnapp);
```

3: skrive kode

Koden skrives ved å lage en klasse som implementerer **ActionListener** og definerer metoden **actionPerfomed**:

```
class Stopper implements ActionListener {
    @Override
    public void actionPerformed (ActionEvent e) {
        System.exit(0);
    }
}
```

4: koble knapp og kode

Til sist kan vi koble koden til knappen:

```
exitKnapp.addActionListener(new Stopper());
```

Resultatet

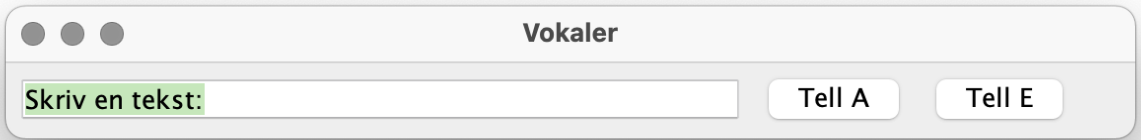
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class Exit {
    public static void main (String[] arg) {
        JFrame vindu = new JFrame("Exit");
        vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        vindu.add(panel);

        JButton exitKnapp = new JButton("Exit");
        class Stopper implements ActionListener {
            @Override
            public void actionPerformed (ActionEvent e) {
                System.exit(0);
            }
        }
        exitKnapp.addActionListener(new Stopper());
        panel.add(exitKnapp);

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

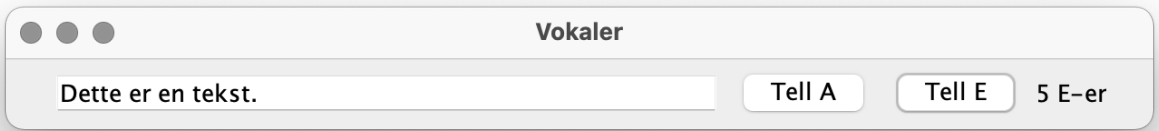
Klassen JTextField

Ofte er det ønskelig å motta tekst fra brukeren; til dette brukes en **JTextField** der man angir initiell tekst og bredde:

```
JTextField tekst = new JTextField("Skriv en tekst:", 30);
```

Brukeren kan redigere denne teksten, og programmet kan lese og skrive med metodene **getText** og **setText**.





Demo 4: Et quizprogram

Vi skal lage et GUI-program med en liten quiz.

1. Programmet ber brukeren om navnet på en oppgavefil.
2. Programmet leser filen og oppretter et GUI-vindu med
   - spørsmålet
   - fire trykknapper med svarternativer plassert i 2×2-mønster.
3. Hvis brukeren velger korrekt alternativ, endres teksten på trykknappen til **OK**.
4. Ved feil valg blir teksten på trykknappen forandret til **Nei**.

Spørsmålsfilen

Filen med spørsmål og svar inneholder:

1. spørsmålet (på første linje)
2. fire svaralternativer; korrekte svar har en * først.

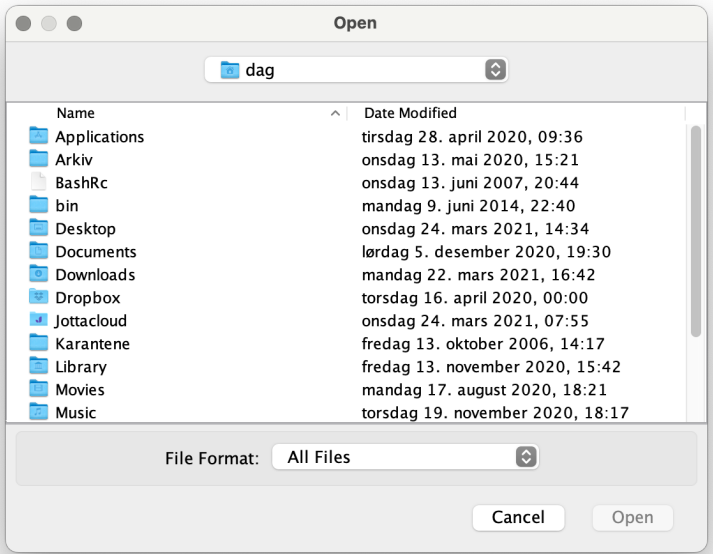
hovedstad-belgia.txt

```
Hovedstaden i Belgia?
Paris
Amsterdam
*Brussel
Luxembourg
```

Å velge en fil

Klassen **JFileChooser** inneholder det vi trenger for å be brukeren om et filnavn.

```
JFileChooser velger =
    new JFileChooser();
int resultat =
    velger.showOpenDialog(null);

if (resultat ==
    JFileChooser.APPROVE_OPTION)
{
    File f =
        velger.getSelectedFile();
} else {
    // Cancel
}
```



Hvordan plasseres elementene?

Brukeren kan angi hvordan elementene (JButton, JLabel etc) plasseres på en tegneflate (dvs JPanel-et). Det finnes mer enn 10 ulike måter.

Standard er **FlowLayout** der elementene kommer etter hverandre som ord i en tekst. Om nødvendig vil det bli en ny «linjeskift».

Noen ganger foretrekker vi et rutenett.

Ulike typer layout

```
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);
```

Del 1: få tak i filen

Det første main-metoden skal gjøre, er å be brukeren om navnet på en fil:

```
public static void main (String[] arg) {
    JFileChooser velger = new JFileChooser();
    int resultat = velger.showOpenDialog(null);
    if (resultat != JFileChooser.APPROVE_OPTION)
        System.exit(1);
    File f = velger.getSelectedFile();
    Scanner leser = null;
    try {
        leser = new Scanner(f);
    } catch (FileNotFoundException e) {
        System.exit(1);
    }
```

Del 2: initiere GUI-vinduet

Dette er helt standard:

```
JFrame vindu = new JFrame("Quiz");
vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JPanel panel = new JPanel();
vindu.add(panel);
```

Del 3: lese spørsmålet

Første linje i filen inneholder spørsmålet:

```
JLabel spoersmaal = new JLabel(leser.nextLine());
panel.add(spoersmaal);
```

Del 4: lese svaralternativene

Så kommer de fire svaralternativene som plasseres på sin egen 2×2-flate:

```
static Svar[] svar = new Svar[4];

JPanel alternativer = new JPanel();
alternativer.setLayout(new GridLayout(2,2));
for (int i = 0;  i < 4;  i++) {
    String s = leser.nextLine();
    if (s.startsWith("*")) {
        svar[i] = new Svar(s.substring(1), true);
    } else {
        svar[i] = new Svar(s, false);
    }
    svar[i].initGUI();
    alternativer.add(svar[i]);
}
panel.add(alternativer);
```

Del 5: klassen Svar med svarhåndtering

```
class Svar extends JButton {
    String svar;
    boolean riktig;

    Svar (String s, boolean r) {
        super(s);
        svar = s;  riktig = r;
    }

    class Velger implements ActionListener {
        @Override
        public void actionPerformed (ActionEvent e) {
            if (riktig) setText("OK");
            else setText("NEI");
        }
    }

    void initGUI () {
        addActionListener(new Velger());
    }
}
```

Hele programmet (del 1)

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

import java.io.*;
import java.util.Scanner;

class Quiz {
    static Svar[] svar = new Svar[4];

    public static void main (String[] arg) {
        JFileChooser velger = new JFileChooser();
        int resultat = velger.showOpenDialog(null);
        if (resultat != JFileChooser.APPROVE_OPTION)
            System.exit(1);
        File f = velger.getSelectedFile();
        Scanner leser = null;
        try {
            leser = new Scanner(f);
        } catch (FileNotFoundException e) {
            System.exit(1);
        }
```

Hele programmet (del 2)

```
        JFrame vindu = new JFrame("Quiz");
        vindu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        vindu.add(panel);

        JLabel spoersmaal = new JLabel(leser.nextLine());
        panel.add(spoersmaal);

        JPanel alternativer = new JPanel();
        alternativer.setLayout(new GridLayout(2,2));
        for (int i = 0;  i < 4;  i++) {
            String s = leser.nextLine();
            if (s.startsWith("*")) {
                svar[i] = new Svar(s.substring(1), true);
            } else {
                svar[i] = new Svar(s, false);
            }
            svar[i].initGUI();
            alternativer.add(svar[i]);
        }
        panel.add(alternativer);

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

Hele programmet (del 3)

```
class Svar extends JButton {
    String svar;
    boolean riktig;

    Svar (String s, boolean r) {
        super(s);
        svar = s;  riktig = r;
    }

    class Velger implements ActionListener {
        @Override
        public void actionPerformed (ActionEvent e) {
            if (riktig) setText("OK");
            else setText("NEI");
        }
    }

    void initGUI () {
        addActionListener(new Velger());
    }
}
```

Oblig 7

Nå kan dere nok til å begynne på oblig 7.

En advarsel

Det er oppdaget innleveringer av tidligere obliger hvor koden er kopiert fra andre.

Universitetet i Oslo ser svært alvorlig på slik kopiering.

Viktig!

Diskuter gjerne ideer, men skriv all koden selv.