# Laboration 16

Avsikten med laborationen är att du ska träna på att göra några grafiska program, dvs. placera komponenter i ett fönster och hantera vissa händelser. Du ska använda null layout när du gör programmen. Börja med att skapa paketet ***laboration16***.

## Grundläggande uppgifter

### Uppgift 16a

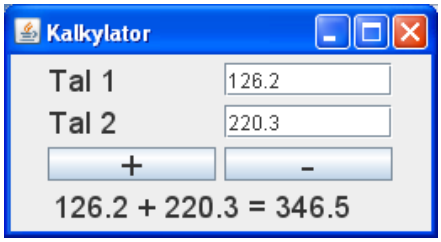
Klassen ***Calculator*** är given (skapa klassen i paketet *laboration16*):

```
package laboration16;
import javax.swing.JFrame;

public class Calculator {
    public void start() {
        JFrame frame = new JFrame( "Kalkylator" );
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        frame.add( new CalcPanel() );
        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        Calculator app = new Calculator();
        app.start();
    }
}
```




Uppgiften går ut på att göra en enkel kalkylator där användaren ska kunna mata in ett par tal och sedan klicka på en knapp och få ett resultat. Instruktionerna i Uppgift 16a leder dig steg för steg till ett färdigt program

#### 1. Skapa en klass som är en panel

Högerklicka paketet *laboration16* och välj ***New*** – ***Class***. Ange ***CalcPanel*** som namn på klassen.

Följande ska du göra:

1. Klassen ska importera paketen *javax.swing*, *java.awt* och *java.awt.event*

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

2. Klassen ska ärva *JPanel*

   ```
   public class CalcPanel extends JPanel {
   ```

3. Ange lämplig storlek på panelen. Detta ska du göra i konstruktorn. Lämplig storlek på fönstret är bredden 250 pixlar och höjden 105 pixlar.

   ```
   public CalcPanel() {
       setPreferredSize( new Dimension(250,105) );
   }
   ```


4. Nu är det dags att testköra programmet ***Calculator***.

## 2. Placera komponenter i fönstret

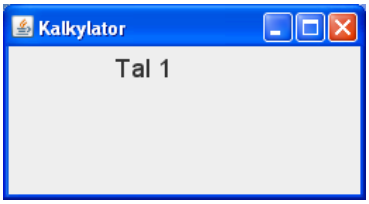
### *Text-komponenten ”Tal 1”*

Nu ska du placera in texten *Tal 1*. Följande ska du göra:

1. Lägg till en instansvariabel av typen *JLabel* i klassen
```
private JLabel lblNbr1 = new JLabel( "Tal 1" );
```
2. Lägg till en instansvariabel av typen *Font* i klassen. Fonten ska användas till texterna i programmet.
```
private Font fontLabels = new Font( "SansSerif", Font.PLAIN, 18 );
```

I punkterna 3 – 5 lägger du till koden i konstruktorn.

3. Ge labeln bredden 100 pixlar och höjden 20 pixlar.
```
lblNbr1.setPreferredSize( new Dimension( 100, 20 ) );
```
4. Sätt labelns font till font-objektet du skapat ovan.
```
lblNbr1.setFont( fontLabels );
```
5. Placera labeln i panelen.
```
add( lblNbr1 );
```




Testkör programmet när du placerat texten.

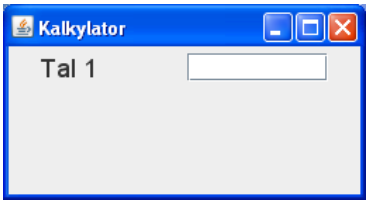
### *Inmatningsfönster för Tal 1*

Nu ska du placera in inmatningsfönstret för Tal 1. Följande ska du göra:

1. Lägg till en instansvariabel av typen *JTextField* i klassen
```
private JTextField tfNbr1 = new JTextField();
```

I punkterna 2 – 4 lägger du till koden i konstruktorn.

2. Ge inmatningsfönstret bredden 100 pixlar och höjden 20 pixlar.
```
tfNbr1.setPreferredSize( new Dimension( 100, 20 ) );
```
3. Placera inmatningsfönstret i containern (huvudfönstret).
```
add( tfNbr1 );
```




Testkör programmet när du placerat inmatningsfönstret.

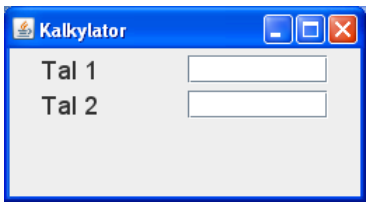
### *Text-komponenten ”Tal 2”*

Nu ska du placera in texten *Tal 2* i panelen. Du ska ge komponenten samma storlek som *lblNbr1* och texten ska visas med samma font som texten i *lblNbr1*. Ge JLabel-komponenten ett lämpligt namn, t.ex. *lblNbr2*.


Testkör programmet när du lagt till komponenen.

### *Inmatningsfönster för Tal 2*

Nu ska du placera in ett inmatningsfönster för Tal 2. Inmatningsfönsteret ska ha samma storlek som *tfNbr1*. Ge JTextField-komponenten ett lämpligt namn, t.ex. *tfNbr2*.




Testkör programmet när du placerat inmatningsfönstret.

### *Plus-knappen*

Följande ska du göra:

1. Lägg till en instansvariabel av typen *JButton* i klassen
```
private JButton btnAdd = new JButton( "+" );
```

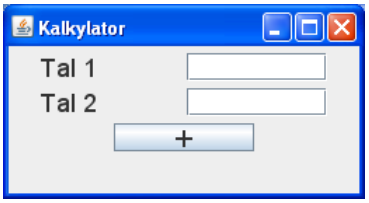
2. Lägg till en instansvariabel av typen *Font* i klassen. Fonten ska användas till knapparna.
```
private Font fontButtons = new Font( "SansSerif", Font.PLAIN, 24 );
```

I punkterna 3 – 5 lägger du till koden i konstruktorn.

3. Ge knappen bredden 100 pixlar och höjden 20 pixlar.
```
btnAdd.setPreferredSize( new Dimension( 100, 20 ) );
```

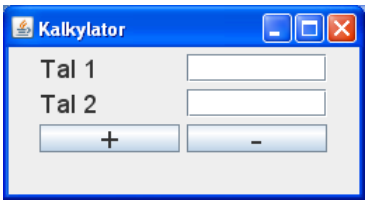
4. Sätt knappens font till font-objektet du skapat ovan.
```
btnAdd.setFont( fontButtons );
```

5. Placera knappen i fönstret.
```
add( btnAdd );
```




Testkör programmet när du placerat knappen.

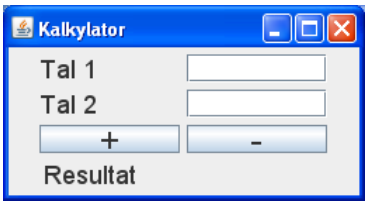
### *Minus-knappen*




Nu ska du placera in minusknappen i fönstret. Knappen ska ha samma storlek som *btnAdd*. Minus-tecknet ska visas med samma font som plus-tecknet på *btnAdd*. Ge knappen ett lämpligt namn, t.ex. *btnSub*.

Testkör programmet när du placerat knappen.

### *Text-komponenten som visar resultatet*




Nu ska du placera in JLabel-komponenten som ska visa resultatet av beräkningen. Komponenenten ska vara 200 pixlar bred och 20 pixlar hög. Texten ska visas med samma font som tidigare text. Ge komponenten ett lämpligt namn, t.ex. *lblResult*. Se till att texten ”Resultat” visas. Senare ska vi visa resultat av beräkningar i denna JLabel-komponent.

Testkör programmet när du placerat texten.

## 3. Beräkna vid klick på knapp

Nu när designen av GUI:et är färdigt så ska du se till att något händer när användaren klickar på plus-knappen eller minus-knappen.

För att lyssna på klick på knapp använder man en *ActionListener*. Klassen finns i paketet *java.awt.event* och import av paketet lades till i när konstruktionen av panelen började.

Klassen ska implementera *ActionListener* och klassen ska innehålla metoden *actionPerformed*.

Följande ska du göra:

1. Lägg till *implements ActionListener* i klassens deklaration. När du gör detta blir det en röd markering i kanten. Denna ska försvinna när du är färdig med punkt 2.
```
public class CalcPanel extends JPanel implements ActionListener {
```

2. Lägg till metoden *actionPerformed* efter konstruktorn.
```
public void actionPerformed(ActionEvent e) {

}
```

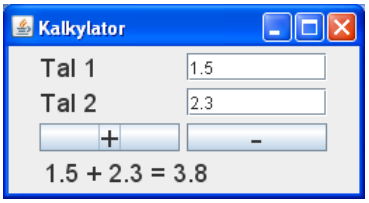
### *Plus-knappen*

Nu ska du se till att programmet lyssnar efter klick på plus-knappen och utför en beräkning om så är fallet. Gör så här:

1. Registrera klassen som lyssnare på plus-knappen. Lägg till följande rad i konstruktorn:
   ```
   btnAdd.addActionListener( this );
   ```
2. Se till att en beräkning utförs då systemet anropar *actionPerformed*. Eftersom fler knappar kommer att använda *actionPerformed*-metoden så måste du kontrollera vilken knapp som klickats på. Så här kan du skriva *actionPerformed*:

   ```
   public void actionPerformed(ActionEvent e) {
       double nbr1, nbr2;
       String nbr1Str, nbr2Str, res;
       nbr1Str = tfNbr1.getText();
       nbr2Str = tfNbr2.getText();
       nbr1 = Double.parseDouble( nbr1Str );
       nbr2 = Double.parseDouble( nbr2Str );
       if( e.getSource() == btnAdd ) {
           res = nbr1Str + " + " + nbr2Str + " = " + (nbr1 + nbr2);
           lblResult.setText( res );
       }
   }
   ```

   De sex första raderna utförs oavsett vilken knapp användaren klickar. Instruktionerna i if-satsen utförs endast om användaren klickat på btnAdd-knappen.




Kör programmet, mata in tal i inmatningsfönsten och klicka på plus-knappen.

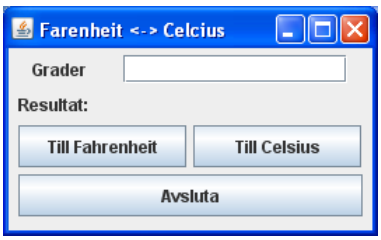
### *Minus-knappen*

Nu ska du aktivera minus-knappen. Det är samma två steg som ovan:

1. Registera klassen som lyssnare på minus-knappen
2. Gör beräkning och visa resultatet. Det innebär endast att du ska lägga till en else-if på if-saten i actionperformed:
   ```
   if( e.getSource() == btnAdd ) {
       // se ovan
   } else if( e.getSource() == btnSub {
       // skapa lämplig sträng att visa
       // visa strängen i resultat-labeln.
   }
   ```

Kör programmet med lite olika värden.

## Uppgift 16b




Gör ett program som ändrar mellan Fahrenheitgrader och Celsiusgrader. Du ska lösa uppgiften i tre steg, samma som i Uppgift 16a. Denna gång har du inte en given main-metod. Under punkt 1 får du ett alternativt sätt att se hur panelen ser ut.

1. *Skapa en klass som är en panel* (extends JPanel). Ge klassen ett lämpligt namn, t.ex. ***FCConverterPanel***. Lämpliga mått på panelen är svårt att veta från början. Utgå från en lite breddare och lite högre panel än i förra uppgiften och ändra sedan när det behövs.
   Lägg till en *main*-metod i klassen. Den ska se ut så här:

```
public static void main(String[] args) {
    FCConverterPanel panel = new FCConverterPanel();
    panel.setBorder( BorderFactory.createLineBorder(Color.BLACK));
    JOptionPane.showMessageDialog(null,panel);
}
```


   Exekvera *main*-metoden. En tom panel visar sig i dialog-fönstret. Border-inställningen gör att panelens storlek syns.
   När du vill se hur en panel, som du jobbar med, ser ut kan du använda en main-metod med samma instruktioner som ovanstående. Panelen kommer dock aldrig bli mindre än 195 pixlar bred och 32 pixlar hög när den visas av *JOptionPane.showMessageDialog*.

2. Nu ska du placera ut komponenterna så panelen ser vettigt ut när ett program visar det.
   Komponenterna är:
   2 st *JLabel*
   1 st *JTextField*
   3 st *JButton*
   Tänk på att ge komponenterna lämlig bredd och höjd genom anrop till setPreferredSize. Både bredden och höjden varierar. Tänk också på att placera in komponenterna i rätt ordning.
   Testkör efter inplacering av varje komponent. Så du har kontroll över det som sker.

3. Slutligen ska du lägga till händelsehantering. Det innebär att du kommer använda följande konstruktion i *actionPerformed*-metoden. Men variabelnamnen är kanske inte samma.

```
if( e.getSource() == btnCelsiusToFarenheit ) {
    // kod om användaren klickat på "Till Fahrenheit"
} else if( e.getSource() == btnFarenheitToCelsius ) {
    // kod om användaren klickat på "Till Celsius"
} else if( e.getSource() == btnExit ) {
    System.exit( 0 );
}
```

**Följande information behöver du när du skriver actionPerformed:**

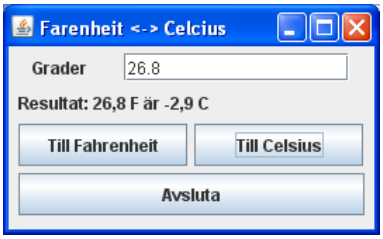
Formel för att ändra från Celsius till Fahrenheit:
F = 32 + 1.8 * C

Formel för att ändra från Fahrenheit till Celsius
C = ( F - 32 ) / 1.8;

För att formatera double-värden i en sträng till ett bestämt antal decimaler kan du använda metoden *String.format*:

```
double nbr1 = 12.7366542;
double nbr2 = 4.8928221;
String res1 = String.format("%1.1f", tal1);     // en decimal
String res2 = String.format("3 decimaler: %1.3f", tal2 );
// När ovanstående rader exekverats så är värdena på res1 och res2:
// res1 = "12.7"
// res2 = "3 decimaler: 4.893"
```

### Skapa en startklass med ett fönster




Nu återstår det att skapa en startklass, ***FCConverter***, vilken ska ha samma struktur som *Calculator* i Uppgift 16a. Det innebär egentligen endast några mindre förändringar (nya klassnamn och ny fönstertitel).

## Uppgift 16c


I denna uppgift ska du använd JRadioButton-komponenter (JF s 278-). Du ska skapa en panel vilken innehåller en JLabel-komponent och 4 JRadioButton-komponenter. När användaren klickar på en radiobutton ska panelens bakgrundsfärg ändras.

1. Skapa klassen ***ColorPanel***, vilken ska ärva *JPanel*. Ge panelen bredd och höjd och skriv en *main*-metod vilken visar panelen.

2. Lägg till en *JLabel*-komponent och 4 *JRadioButton*-komponenter i panelen. *Kör programmet* och testa att klicka i JRadioButton-komponentena. Som du märker avmarkeras aldrig en knapp som en gång markerats. Lägg till en *ButtonGroup* (i konstruktorn):

   ```
   ButtonGroup group = new ButtonGroup();
   ```

   Lägg sedan till vardera JRadioButton-komponent i ButtonGroup-komponenten, t.ex.

   ```
   group.add( rbRed );
   ```

   *Kör programmet* och testa att klicka på olika radiobuttons. Nu kan endast en knapp vara markerad åt gången.

3. Lägg till händelsehantering. Det gör du genom att skriva en inre klass vilken implementerar *ActionListener*. I *actionPerformed* gäller det att ta reda på vilken radiobutton som är markerad. Metoden *isSelected* i klassen *JRadioButton* returnerar *true* om knappen är markerad och annars *false*.

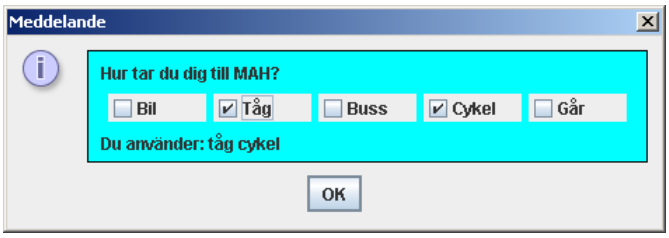
   ```
   public class ColorPanel extends JPanel {
       // diverse instansvariabler och konstruktor
       private class ColorListener implements ActionListener {
           public void actionPerformed(ActionEvent e) {
               if (rbRed.isSelected()==true) {
                   setBackground(Color.RED);
               } else if (rbGreen.isSelected()==true) {
                   setBackground(Color.GREEN);
               } else if (rbBlue.isSelected()==true) {
                   setBackground(Color.BLUE);
               } else if (rbRandom.isSelected()==true) {
                   Color color = new Color(rand.nextInt(256), rand.nextInt(256),
                                           rand.nextInt(256));
                   setBackground(color);
               }
           }

       }
       // main-metod
   }
   ```

   Glöm inte att lägga till den inre klassen som lyssnare på JRadioButton-komponenterna:

   ```
   ColorListener listener = new ColorListener();
   rbRed.addActionListener( listener );
   :
   ```

## Uppgift 16d




I denna uppgift ska du använd JCheckBox-komponenter (JF s 275-). Du ska skapa en panel vilken innehåller 2 JLabel-komponenter och 4 JCheckBox-komponenter. När användaren markerar/avmarkerar checkboxar ska information skrivas ut i en label.

1. Skapa klassen ***TransportPanel***, vilken ska ärva *JPanel*. Ge panelen bredden 400 och höjden 60. Sätt panelens bakgrundsfärg till Color.CYAN. Skriv en *main*-metod vilken visar panelen.
2. Lägg till två *JLabel*-komponenter och 5 *JCheckBox*-komponenter i panelen. *Kör programmet*. Resultatet ska likna panelen ovan.
3. Lägg till händelsehantering. Det gör du genom att skriva en inre klass vilken implementerar ItemListener. För att kontrollera om en checkbox är markerad anropar du *isSelected*-metoden för checkboxen. Algortmen i itemStateChanged-metoden blir:

   deklarera String-objektet *res* och initiera till ”Du använder: ”
   om *cbCar* är vald
   lägg till ”bil ” i slutet av *res*
   om *cbTrain* är vald
   lägg till ”tåg” i slutet av *res*
   osv.
   överför värdet i *res* till *lblResult* (*setText*-metoden)

```
public class TransportPanel extends JPanel {

    private class TransportListener implements ItemListener {
        public void itemStateChanged(ItemEvent e) {
            String res = "Du använder: ";
            // komplettera med kod
            lblResult.setText(res);
        }
    }

}
```

   Glöm inte att lägga till den inre klassen TransportPanel som lyssnare på JCheckBox-komponenterna:

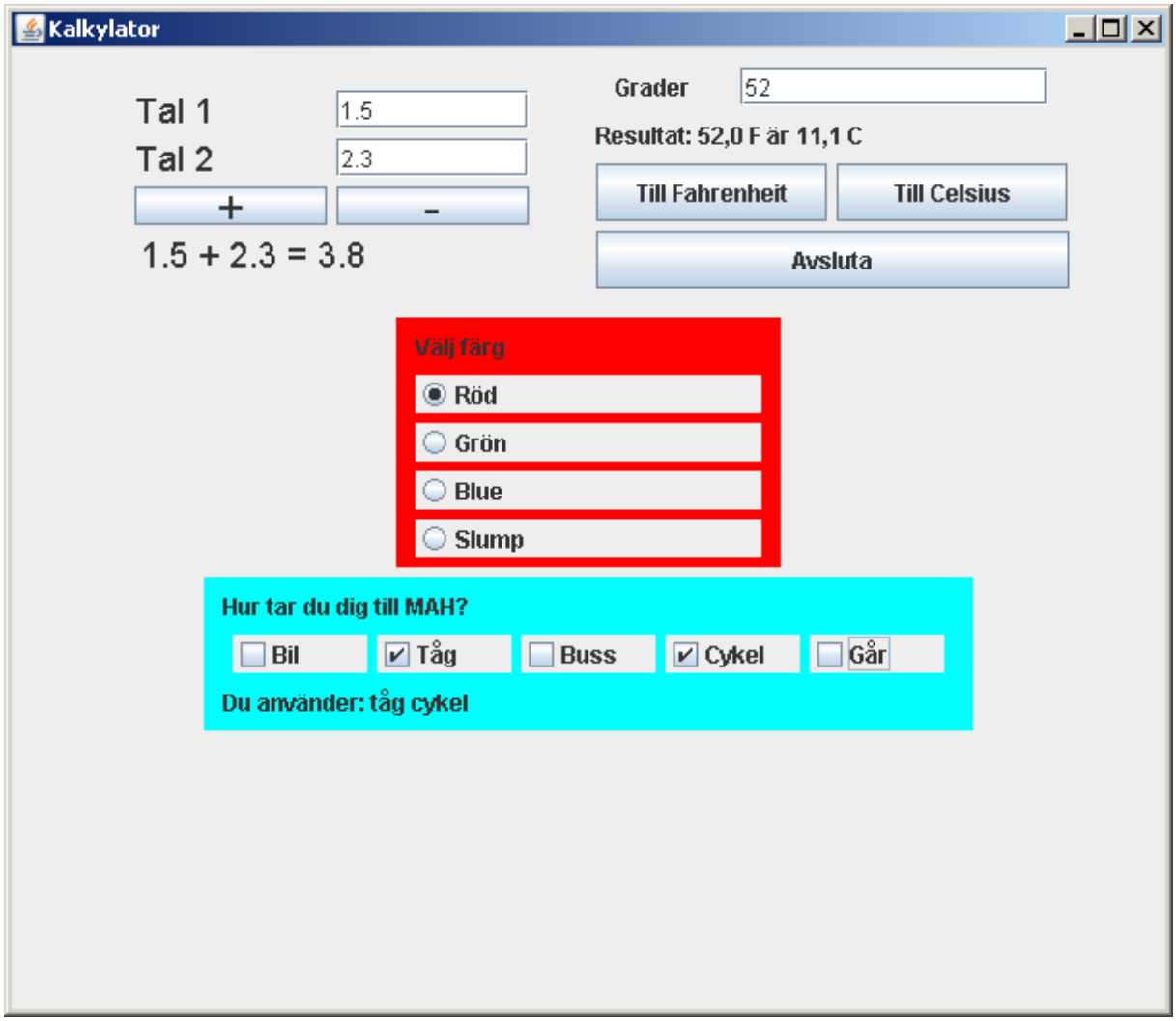
```
TransportListener listener = new TransportListener();
cbCar.addItemListener( listener );
:
```

## Uppgift 16e

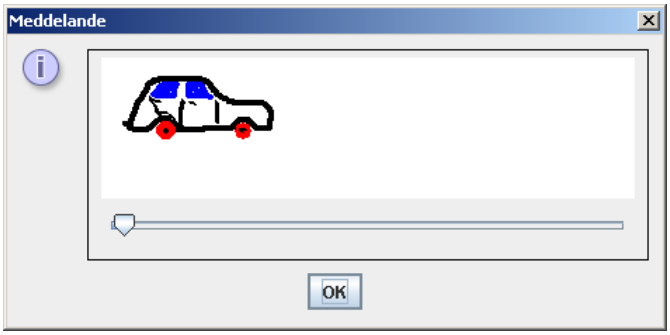
Det går utmärkt att placera en panel i en annan panel. Nu ska du skapa en klass som ärver JPanel. Ge klassen namnet ***AllPanels***.

Ge panelen storleken 600x500.
Placera en *CalcPanel*, en *FCConverterPanel*, en *ColorPanel* och en *TransportPanel* i panelen.

Skapa sedan en startklass, ***StartAll***, i vilken ett objekt av typen *AllPanels* skapas och sedan placeras i fönstret (liknande klassen *Calculator*).



# Fördjupande uppgifter




## Uppgift 16f

Du ska skriva ett program som låter användaren flytta en bild (en bil i detta exemplet). Till din hjälp har du följande resurser:

- Klassen ***PaintPanel*** – En komponent du kan placera i t.ex. en JPanel. I en PaintPanel kan du bl.a. rita och visa bilder.
- Klassen ***Car*** – Är en modell av en bil och har tre viktiga egenskaper: *image*, *x* och *y*. Det är samma klass som du använde på programmeringsuppgift 2
- I programmet används bilden *BilH.gif*.

| **Car** |
| --- |
| - image : ImageIcon<br>- x : int<br>- y : int |
| +Car(ImageIcon)<br><br>+getImage() : ImageIcon<br>+getX() : int<br>+getY() : int<br>+moveTo(int neewX,int newY) |

När du skriver programmet så ska du åter ta dig genom de tre stegen:

1. Skapa en klass som ärver JPanel. Ge panelen storlek så en PaintPanel-komponent och en JSlider-komponent ryms i den. Klassen bör ha följande instansvariabler:
   1 st PaintPanel, 1 st JSlider, 1 st Car
2. PaintPanel – skapa ett panel, ge panelen lämplig storlek och placera panelen på huvudpanelen.
   JSlider – skapa en horisontell slider, med samma bredd som PaintPanel-objektet och höjden 30. Placera slidern på huvudpanelen.
   ImageIcon – skapa ett objekt vilket du använder när Car-objektet skapas.
   Car – skapa ett objekt, tilldela x värdet 10 och y värdet 10.
   När ovanstående är färdigt kan du anropa showImage i PaintPanel. Använd Car-objektet för bild som ska visas och positionen för bilden.

   Placera panelen i ett fönster eller en dialog för att se utseendet. Bör vara ungefär som panelen ovan.
3. Nu ska du lyssna på JSlider-komponenten. Så fort användaren flyttar på pilen i slidern så ska bilen flytta med.
   Skriv en inre klass vilken implementerar ChangeListener.
   Skriv metoden public *void stateChanged(ChangeEvent e)* i den inre klassen. Se till att Car-objektets x-värde uppdateras med värdet i slider-objektet. Anropa därefter showImage på nytt.

   Glöm inte att registrera den inre klassen som lyssnare på slider-objektet.

## Förslag till lösningar

### Uppgift 16a

```
package laboration16;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class CalcPanel extends JPanel implements ActionListener {
    private JLabel lblNbr1 = new JLabel( "Tal 1" );
    private JLabel lblNbr2 = new JLabel( "Tal 2" );
    private JLabel lblResult = new JLabel( "Resultat" );
    private JTextField tfNbr1 = new JTextField();
    private JTextField tfNbr2 = new JTextField();
    private JButton btnAdd = new JButton( "+" );
    private JButton btnSub = new JButton( "-" );
    private Font fontButtons = new Font( "SansSerif", Font.PLAIN, 24 );
    private Font fontLabels = new Font( "SansSerif", Font.PLAIN, 18 );

    public CalcPanel() {
            this.setPreferredSize(new Dimension(250,105));

        lblNbr1.setPreferredSize( new Dimension( 100, 20 ) );
        lblNbr1.setFont( fontLabels );
        tfNbr1.setPreferredSize( new Dimension( 100, 20 ) );
        lblNbr2.setPreferredSize( new Dimension( 100, 20 ) );
        lblNbr2.setFont( fontLabels );
        tfNbr2.setPreferredSize( new Dimension( 100, 20 ) );
        btnAdd.setPreferredSize( new Dimension( 100, 20 ) );
        btnAdd.setFont( fontButtons );
        btnSub.setPreferredSize( new Dimension( 100, 20 ) );
        btnSub.setFont( fontButtons );
        lblResult.setPreferredSize( new Dimension( 200, 20 ) );
        lblResult.setFont( fontLabels );

        btnAdd.addActionListener( this );
        btnSub.addActionListener( this );

        add( lblNbr1 );
        add( tfNbr1 );
        add( lblNbr2 );
        add( tfNbr2 );
        add( btnAdd );
        add( btnSub );
        add( lblResult );
   }

    public void actionPerformed(ActionEvent e) {
        double nbr1, nbr2;
        String nbr1Str, nbr2Str, res;
        nbr1Str = tfNbr1.getText();
        nbr2Str = tfNbr2.getText();
        nbr1 = Double.parseDouble( nbr1Str );
        nbr2 = Double.parseDouble( nbr2Str );
        if( e.getSource() == btnAdd ) {
            res = nbr1Str + " + " + nbr2Str + " = " + (nbr1 + nbr2);
            lblResult.setText( res );}
        } else if( e.getSource() == btnSub ) {
            res = nbr1Str + " - " + nbr2Str + " = " + (nbr1 - nbr2);
            lblResult.setText( res );
        }
    }
}
```

## Uppgift 16b

```
package laboration16;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class FCConverterPanel extends JPanel implements ActionListener {
    private JLabel lblDegrees = new JLabel("Grader ");
    private JLabel lblResult = new JLabel("Resultat: ");
    private JTextField tfDegrees = new JTextField();
    private JButton btnFToC = new JButton("Till Celsius");
    private JButton btnCToF = new JButton("Till Fahrenheit");
    private JButton btnExit = new JButton("Avsluta");

    public FCConverterPanel() {
        setPreferredSize( new Dimension( 260, 130) );

        lblDegrees.setPreferredSize( new Dimension( 60, 20 ) );
        lblResult.setPreferredSize( new Dimension( 246, 20 ) );
        btnCToF.setPreferredSize( new Dimension( 120, 30 ) );
        btnFToC.setPreferredSize( new Dimension( 120, 30 ) );
        btnExit.setPreferredSize( new Dimension( 246, 30 ) );
        tfDegrees.setPreferredSize( new Dimension( 160, 20 ) );

        btnCToF.addActionListener(this);
        btnFToC.addActionListener(this);
        btnExit.addActionListener(this);

        add( lblDegrees );
        add( tfDegrees );
        add( lblResult );
        add( btnCToF );
        add( btnFToC );
        add( btnExit );
    }

    public void actionPerformed(ActionEvent e) {
        double deg, res;
        String txt;
        if(e.getSource()==btnCToF) {
            deg = Double.parseDouble(tfDegrees.getText());
            res = 32+1.8*deg;
            txt = String.format("Resultat: %1.1f C är %1.1f F", deg, res );
            lblResult.setText( txt );
        }
        else if(e.getSource()==btnFToC) {
            deg = Double.parseDouble(tfDegrees.getText());
            res = (deg-32)/1.8;
            txt = String.format("Resultat: %1.1f F är %1.1f C", deg, res );
            lblResult.setText( txt );
        }
        else if(e.getSource()==btnExit) {
            System.exit(0);
        }
    }

    public static void main(String[] args) {
        FCConverterPanel panel = new FCConverterPanel();
        panel.setBorder( BorderFactory.createLineBorder(Color.BLACK));
        JOptionPane.showMessageDialog(null,panel);
    }
}
```

```
package laboration16;
import javax.swing.JFrame;

public class FCConverter {
    public void start() {
        JFrame frame = new JFrame( "Farenheit <-> Celcius" );
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        frame.add( new FCConverterPanel() );
        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        FCConverter app = new FCConverter();
        app.start();
    }
}
```

## Uppgift 16c

```
package laboration16;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;

public class ColorPanel extends JPanel {
    private JLabel lblColor = new JLabel("Välj färg");
    private JRadioButton rbRed = new JRadioButton("Röd");
    private JRadioButton rbGreen = new JRadioButton("Grön");
    private JRadioButton rbBlue = new JRadioButton("Blue");
    private JRadioButton rbRandom = new JRadioButton("Slump");
    private Random rand = new Random();

    public ColorPanel() {
        setPreferredSize(new Dimension(200, 130));

        lblColor.setPreferredSize(new Dimension(180, 20));
        ButtonGroup group = new ButtonGroup();
        rbRed.setPreferredSize(new Dimension(180, 20));
        rbGreen.setPreferredSize(new Dimension(180, 20));
        rbBlue.setPreferredSize(new Dimension(180, 20));
        rbRandom.setPreferredSize(new Dimension(180, 20));
        group.add(rbRed);
        group.add(rbGreen);
        group.add(rbBlue);
        group.add(rbRandom);

        addListeners();

        add(lblColor);
        add(rbRed);
        add(rbGreen);
        add(rbBlue);
        add(rbRandom);
    }

    private void addListeners() {
        ColorListener listener = new ColorListener();
        rbRed.addActionListener(listener);
        rbGreen.addActionListener(listener);
        rbBlue.addActionListener(listener);
        rbRandom.addActionListener(listener);
    }

    private class ColorListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            if (rbRed.isSelected()) {
                setBackground(Color.RED);
            } else if (rbGreen.isSelected()) {
                setBackground(Color.GREEN);
            } else if (rbBlue.isSelected()) {
                setBackground(Color.BLUE);
            } else if (rbRandom.isSelected()) {
                Color color = new Color(rand.nextInt(256), rand.nextInt(256),
                        rand.nextInt(256));
                setBackground(color);
            }
        }
    }

    public static void main(String[] args) {
        ColorPanel panel = new ColorPanel();
        panel.setBorder(BorderFactory.createLineBorder(Color.BLACK));
        JOptionPane.showMessageDialog(null, panel);
    }
}
```

## Uppgift 16d

```
package laboration16;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class TransportPanel extends JPanel {
    private JLabel lblTransport = new JLabel("Hur tar du dig till MAH?");
    private JLabel lblResult = new JLabel();
    private JCheckBox cbCar = new JCheckBox("Bil");
    private JCheckBox cbTrain = new JCheckBox("Tåg");
    private JCheckBox cbBus = new JCheckBox("Buss");
    private JCheckBox cbCycle = new JCheckBox("Cykel");
    private JCheckBox cbWalk = new JCheckBox("Går");

    public TransportPanel() {
        setPreferredSize(new Dimension(400, 80));
        setBackground(Color.CYAN);

        lblTransport.setPreferredSize(new Dimension(380, 20));
        cbCar.setPreferredSize(new Dimension(70, 20));
        cbTrain.setPreferredSize(new Dimension(70, 20));
        cbBus.setPreferredSize(new Dimension(70, 20));
        cbCycle.setPreferredSize(new Dimension(70, 20));
        cbWalk.setPreferredSize(new Dimension(70, 20));
        lblResult.setPreferredSize(new Dimension(380, 20));

        addListeners();

        add(lblTransport);
        add(cbCar);
        add(cbTrain);
        add(cbBus);
        add(cbCycle);
        add(cbWalk);
        add(lblResult);
    }

    private void addListeners() {
        TransportListener listener = new TransportListener();
        cbCar.addItemListener(listener);
        cbTrain.addItemListener(listener);
        cbBus.addItemListener(listener);
        cbCycle.addItemListener(listener);
        cbWalk.addItemListener(listener);
    }

    private class TransportListener implements ItemListener {
        public void itemStateChanged(ItemEvent e) {
            String res = "Du använder: ";
            if (cbCar.isSelected()) {
                res += "bil ";
            }
            if (cbTrain.isSelected()) {
                res += "tåg ";
            }
            if (cbBus.isSelected()) {
                res += "buss ";
            }
            if (cbCycle.isSelected()) {
                res += "cykel ";
            }
```

```
            if (cbWalk.isSelected()) {
                res += "går";
            }
            lblResult.setText(res);
        }

    }

    public static void main(String[] args) {
        TransportPanel panel = new TransportPanel();
        panel.setBorder(BorderFactory.createLineBorder(Color.BLACK));
        JOptionPane.showMessageDialog(null, panel);
    }
}
```

## Uppgift 16e

```
package laboration16;
import javax.swing.*;
import java.awt.*;

public class AllPanel extends JPanel {
    private CalcPanel pnlCalc = new CalcPanel();
    private FCConverterPanel pnlConvert = new FCConverterPanel();
    private ColorPanel pnlColor = new ColorPanel();
    private TransportPanel pnlTransport = new TransportPanel();

    public AllPanel() {
        setPreferredSize( new Dimension(600,500) );
        add( pnlCalc );
        add( pnlConvert );
        add( pnlColor );
        add( pnlTransport );
    }
}
-----------------------------------------------------------------

package laboration16;
import javax.swing.JFrame;

public class StartAll {
    public void start() {
        JFrame frame = new JFrame( "Kalkylator" );
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); n

        frame.add( new AllPanel() );
        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        StartAll app = new StartAll();
        app.start();
    }
}
```

## Uppgift 16f

```
package laboration16;
import javax.swing.*;
import javax.swing.event.*;

import java.awt.*;

public class CarPanel extends JPanel {
    private PaintPanel panel = new PaintPanel();
    private JSlider jsMove = new JSlider(JSlider.HORIZONTAL,0,380,10);
    private Car car;

    public CarPanel() {
        setPreferredSize( new Dimension( 400,150 ) );

        ImageIcon image = new ImageIcon("M:/Bilder/BilH.gif");
        car = new Car(image);

        panel.setPreferredSize(new Dimension(380,100));
        jsMove.setPreferredSize(new Dimension(380,30));
        add(panel);
        add(jsMove);
        car.moveTo(10,10);
        showCar();
        jsMove.addChangeListener( new CL() );
    }

    private void showCar() {
        panel.showImage(car.getImage(), car.getX(), car.getY());
    }

    private class CL implements ChangeListener {
        public void stateChanged(ChangeEvent e) {
            car.moveTo(jsMove.getValue(), car.getY());
            showCar();
        }
    }

    public static void main(String[] args) {
        CarPanel panel = new CarPanel();
        panel.setBorder(BorderFactory.createLineBorder(Color.BLACK));
        JOptionPane.showMessageDialog(null, panel);
    }
}
```