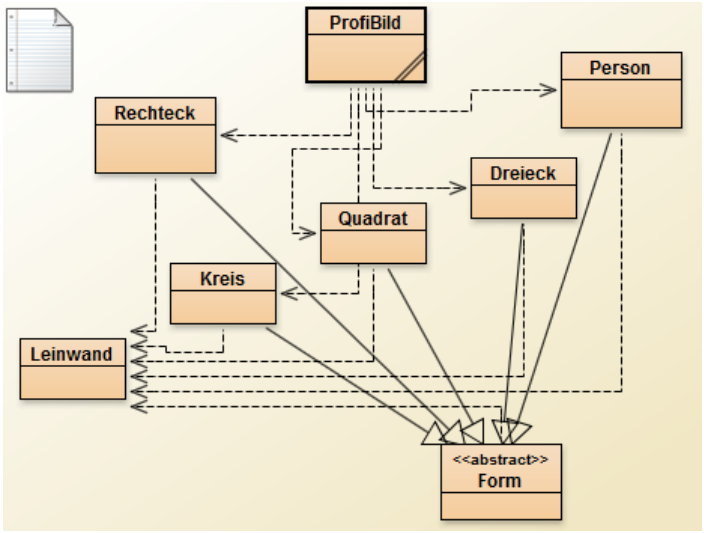
Objektorientierung in

Projekt Figuren

Erweiterung um Prinzip der Vererbung

neue Oberklasse „Form“

und zugehörige Unterklassen




```
public abstract class Form {
//abstrakte Klasse - Gerüst für Unterklassen!

//Attribute sind “protected”, d.h. in Klasse und
//allen Unterklassen sichtbar!
   protected int xPosition; protected int yPosition;
   protected String farbe; protected boolean istSichtbar;

//Methoden, die allen Unterklassen übergeben werden (sollen)
   public Form(int x, int y, String f) {
      xPosition = x;
      yPosition = y;
      farbe = f;
      istSichtbar = false;
   }

   public void sichtbarMachen() {
      istSichtbar = true;
      zeichnen();
   }

   public void unsichtbarMachen() {
      loeschen();
      istSichtbar = false;
   }

   public void nachRechtsBewegen() {
      horizontalBewegen(20);
   }

   public void nachLinksBewegen() {
      horizontalBewegen(-20);
   }

   public void nachObenBewegen() {
      vertikalBewegen(-20);
   }

   public void nachUntenBewegen() {
      vertikalBewegen(20);
   }

   public void horizontalBewegen(int distance) {
      loeschen();
      xPosition += distance;
      zeichnen();
   }

   public void vertikalBewegen(int entfernung) {
      loeschen();
      yPosition += entfernung;
      zeichnen();
   }

   public void langsamHorizontalBewegen(int entfernung) {
      int delta;
      if (entfernung < 0) {
         delta = -1;
         entfernung = -entfernung;
      } else {
         delta = 1;
      }
      for (int i = 0; i < entfernung; i++) {
         xPosition += delta;
         zeichnen();
      }
   }
```

```
public void langsamVertikalBewegen (…) {
   int delta;
   if (entfernung < 0) {
      delta = -1;
      entfernung = -entfernung;
   } else {
      delta = 1;
   }
   for (int i = 0; i < entfernung; i++) {
      yPosition += delta;
      zeichnen();
   }
}
```

```
public void farbeAendern(String neueFarbe) {
    farbe = neueFarbe;
    zeichnen();
}

protected abstract void zeichnen();
 //komplette Auslagerung in Unterklasse

protected void loeschen() {
    if (istSichtbar) {
       Leinwand leinwand = Leinwand.gibLeinwand();
       leinwand.entferne(this);
    }
}
```

```
import java.awt.geom.Ellipse2D;

public class Kreis extends Form { //Kreis wird als Unterklasse der Oberklasse Form definiert
   private int durchmesser; //zusätzliches Attribut wird deklariert

   public Kreis(int d, int x, int y, String f) {
      super(x,y,f); //Konstruktor der Oberklasse (Form) wird aufgerufen
      durchmesser = d; //zusätzliches Attribut wird initialisiert
   }

   public void groesseAendern(int neuerDurchmesser) { //zusätzliche Methode
      loeschen();
      durchmesser = neuerDurchmesser;
      zeichnen();
   }

   public void zeichnen() { //abstrakte Methode der Oberklasse (Form) wird implementiert
      if (istSichtbar) {
         Leinwand leinwand = Leinwand.gibLeinwand();
         leinwand.zeichne(this, farbe, new Ellipse2D.Double(xPosition,
                              yPosition, durchmesser, durchmesser));
         leinwand.warte(10);
      }
   }
```

```
import java.awt.Rectangle;

public class Rechteck extends Form {
   private int laenge;
   private int breite;

   public Rechteck(int l, int b, int x, int y, String f) {
      super(x,y,f); laenge = l; breite = b;
   }

   public void laengeAendern(int neueLaenge) {
      loeschen(); laenge = neueLaenge;
      zeichnen();
   }

   public void breiteAendern(int neueBreite) {
      loeschen(); breite = neueBreite;
      zeichnen();
   }

   public void zeichnen() {
      if (istSichtbar) {
         Leinwand leinwand =
         Leinwand.gibLeinwand();
         leinwand.zeichne(this, farbe, new
         Rectangle(xPosition, yPosition,
                  laenge, breite));
         leinwand.warte(10);
      }
   }
}
```

```
import java.awt.Polygon;

public class Dreieck extends Form {
   private int hoehe;
   private int breite;

   public Dreieck(int h, int b, int x, int y, String f) {
      super(x,y,f);  hoehe = h; breite = b;
   }

   public void groesseAendern(int neueHoehe,
                              int neueBreite) {
      loeschen();
      hoehe = neueHoehe;
      breite = neueBreite;
      zeichnen();
   }

   public void zeichnen() {
      Leinwand leinwand = Leinwand.gibLeinwand();
      int[] xpoints = { xPosition, xPosition + (breite / 2),
                  xPosition - (breite / 2) };
      int[] ypoints = { yPosition, yPosition + hoehe,
                     yPosition + hoehe };
      leinwand.zeichne(this, farbe,
                     new Polygon(xpoints, ypoints, 3));
      leinwand.warte(10);
   }
}
```

… //Klassen Quadrat und Person werden analog verändert!

```
public class ProfiBild {
  private Rechteck Boden; private Quadrat Rumpf; private Dreieck Kopf;
  private Kreis Spitze; private Person Mensch;

   public ProfiBild() {
      Boden = new Rechteck(50,20,50,200,"blau"); Rumpf = new Quadrat(30,60,170,"gelb");
      Kopf = new Dreieck(25,25,75,145,"blau"); Spitze = new Kreis(10,70,135,"rot");
      Mensch = new Person(60,30,130,170,"schwarz");
   }

   public void Zeichne() { Boden.sichtbarMachen(); Rumpf.sichtbarMachen(); … }

   public void Loesche() { Boden.unsichtbarMachen(); …; Mensch.unsichtbarMachen(); }
```