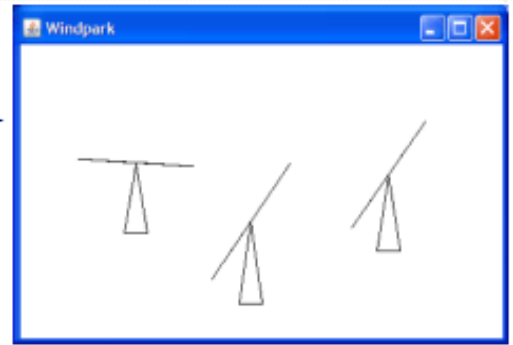
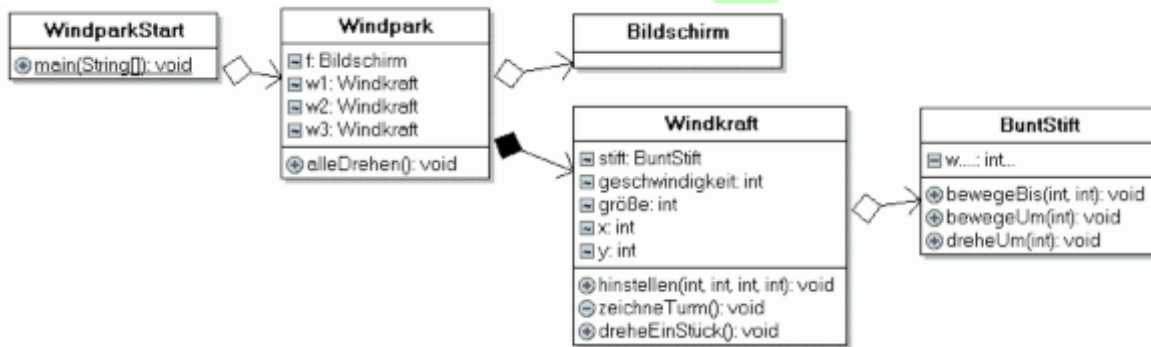


**Windpark**Name: [www.r-krell.de](http://www.r-krell.de)

In einem Windpark stehen mehrere Windmühlen unterschiedlicher Größe, deren Rotoren sich mit verschiedenen Geschwindigkeiten drehen.

Wir überlegten, dass alle Windmühlen (Objekte) im Prinzip nach einem einzigen Bauplan (Klasse Windkraft) gebaut werden können, wenn dieser Bauplan so angelegt wird, dass unterschiedliche Orte, Größen und Geschwindigkeiten möglich sind. Es wurden drei eigene Dateien (Klassen) geplant; die Klassen Bildschirm und BuntStift werden aus der Bibliothek stiftUndCo.\* übernommen:



- ① Klasse Windpark: Von dieser Klasse wird (in der Startdatei WindparkStart) nur ein einziges Objekt *wp* erzeugt, das den einzigen Bildschirm *f* besitzt, auf dem dann alle Windmühlen *w1*, *w2* und *w3* stehen. Die Windmühlen werden hier definiert und erzeugt. Die einzige Methode *alleDrehen* sorgt für die grafische Darstellung der drei drehenden Windmühlen, indem die Methode *dreheEinStück* jedes Windkraft-Objekts *w1*, *w2*, *w3* aufgerufen wird. Jedes *dreheEinStück* benutzt die in der jeweiligen Windmühle gespeicherten Daten und den zu jeder Windmühle gehörenden BuntStift, sodass bei drei Windmühlen tatsächlich drei verschiedene BuntStifte auf den Bildschirm zeichnen.

```

import stiftUndCo.*;

public class Windpark // Bauplan für Windpark, ..
{ // .. der mehrere Windmühlen enthält
    Bildschirm f = new Bildschirm("Windpark",420,250); // 1 Schirm für alle Mühlen
    Windkraft w1 = new Windkraft(); // enthaltene Mühlen
    Windkraft w2 = new Windkraft();
    Windkraft w3 = new Windkraft();

    public void alleDrehen()
    { // Mühlen werden aufgestellt an verschiedenen Stellen,
      // mit versch. Flügellängen und mit zufälliger Geschwindigkeit (zwischen 2 und 6)
      w1.hinstellen(100,100, 50, Hilfe.zufall(2,6));
      w2.hinstellen(200,150, 60, Hilfe.zufall(2,6));
      w3.hinstellen(320,110, 55, Hilfe.zufall(2,6));
      while(true) // Endlosschleife,
      { // .. dreht reihum an jeder Mühle
          w1.dreheEinStück();
          w2.dreheEinStück();
          w3.dreheEinStück();
          Hilfe.warte(10);
      }
    }
}
  
```

- ② Klasse Windkraft (Bauplan für eine bzw. für jede Windmühle). Nach diesem allgemeinen Plan werden tatsächlich drei Objekte  $w1$ ,  $w2$  und  $w3$  erzeugt (in der Klasse Windpark bzw. im Objekt  $wp$  der Startdatei). Der Rotor wird mit relativen Befehlen gezeichnet ( $bewegeUm$ ,  $dreheUm$ ), während für den Turm absolute Koordinaten bevorzugt werden, die von der Turmspitze ausgehend berechnet werden.

```
import stiftUndCo.*;
public class Windkraft // Bauplan für eine Windmühle
{
    BuntStift stift = new BuntStift(); // für Turm und Rotor
    int geschwindigkeit; // Eigenschaften = Datenfelder = ..
    int gröÙe; // .. = globale Variablen
    int x; // für Gradzahl pro Drehstück, Flügelradius,
    int y; // und Bildschirm-Koordinaten der Rotorachse (=Turmspitze)

    public void hinstellen (int xPos, int yPos, int flügelänge, int gradzahl)
    {
        x = xPos; // Speichern der übergebenen..
        y = yPos; // .. Parameter in den Datenfeldern
        gröÙe = flügelänge;
        geschwindigkeit = gradzahl;
        dreheEinStück(); // Zeichnen von Flügel und Turm
    }

    private void zeichneTurm() // Zeichnen des Turms (wird aus dreheEinStück
    { // .. heraus aufgerufen)
        stift.bewegeBis (x,y); // Turmspitze (=Rotormittelpunkt)
        stift.runter();
        stift.bewegeBis (x-10,y+gröÙe+10); // Turm ist gleichschenkliges Dreieck mit
        stift.bewegeBis (x+10,y+gröÙe+10); // Basis 20 Pixel (von x-10 bis x+10) und der
        stift.bewegeBis (x,y); // Höhe Flügelradius+10
    }

    public void dreheEinStück() // Rotor um ein Stück drehen
    {
        stift.bewegeBis (x,y); // Stift zum Rotormittelpunkt (=Turmspitze)
        stift.radiere(); // alten Rotor wegradieren
        stift.bewegeUm (gröÙe); // mit relativer Länge ab Turmspitze
        stift.bewegeUm (-2*gröÙe); // egal in welcher Richtung

        stift.dreheUm (geschwindigkeit); // Stift (und damit Rotor) etwas drehen
        stift.bewegeBis (x,y); // Stift zum Rotormittelpunkt (=Turmspitze)
        stift.normal(); // neuen Rotor zeichnen:
        stift.bewegeUm (gröÙe); // Flügel nach einer Seite
        stift.bewegeUm (-2*gröÙe); // und doppelt so weit zurück zur anderen Seite

        zeichneTurm(); // evtl. auch Stück Turm wegradiert, daher neu zeichnen
    }
}
```

- ③ Startdatei: Nur der Windpark  $wp$  wird erzeugt (weil der seinerseits die Windmühlen erzeugt).

```
public class WindparkStart // Startdatei für den Windpark
{
    public static void main (String[] s)
    {
        Windpark wp = new Windpark(); // Windpark-Objekt wp definieren und erzeugen
        wp.alleDrehen(); // Methode von wp aufrufen, die alles macht
    }
}
```