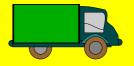
Der Behälter



Der Behälter

Der Behälter ist eine Container-Klasse.

Die Klasse Behälter kapselt die GUI-Darstellung der Komponenten.

Sie ist in einer "normalen" Java-Klasse die Ergänzung, in der die **Graphische Darstellung der Objekte** gekapselt ist.

Er nimmt **graphische-Elemente** auf.

Der Behälter bestimmt einen rechteckigen Bereich, in dem die Objekte des Behälters dargestellt werden. Diese werden **relativ zum Behälter positioniert**.

Dies vereinfacht vor allem die Positionierung der im Behälter zusammengefassten Elemente. Verschiebt man den Behälter, so werden alle Komponenten im Behälter mit verschoben.

Wie die Klasse Zeichnung kann auch der Behälter ein **Raster** einschalten.

Behaelter
© Behaelter()
© Behaelter(int, int)
© Behaelter(int, int, int)
© Behaelter(lContainer)
© Behaelter(IContainer, int, int, int)
⊕ setzeZoomfaktor(double): void
⊕ getBehaelterZoom(): double
⊕hinzufuegen(IComponente): void
⊕ hinzufuegenUndAnpassen(IComponente): void
⊕ getBasisComponente(): BasisComponente
⊕ setzeGroesse(int, int): void
⊛ setzePosition(int, int): void
⊛ setzeDimensionen(int, int, int, int): ∨oid
sichtbarMachen(): void
⊕unsichtbarMachen(): void
⊕ horizontalBewegen(int): ∨oid
⊕ vertikalBewegen(int): void
⊕ nachRechtsBewegen(): void
⊕ nachLinksBewegen(): void
⊕ nachObenBewegen(): void ⊕ nachUntenBewegen(): void
⊕ langsamVertikalBewegen(int): void
⊕ langsamHorizontalBewegen(int): void
setzeMitRaster(boolean): void
⊕ setzeDeltaX(int): void
⊕ setzeDeltaY(int): void
⊕add(Component, int): Component
setzeKomponentenKoordinaten(JComponent int int int int): void
® setzeKomponentenGroesse(JComponent, int, int): void
@setzeKomponentenPosition(JComponent, int, int): void
⊕validate(): void
⊕ getPaneĬ(): JPanel

Die Standard-Konstruktoren fügen Elemente der Toolbox automatisch dem "Behälter" des Programms hinzu.

Zusätzliche Konstruktoren erzeugen die Komponenten gleich im Ziel-Behälter. Setzt man diesen Behälter unsichtbar, so sieht man den Behälterinhalt erst nach dem Sichtbar-Mache. Der Bildaufbau wird ruhiger.

Mit der Methode **hinzufuegen**(IComponente obj) löst man die Komponente obj aus dem bisherigen Behälter und fügt ihn dem neuen Behälter hinzu.

Die Methode **setzeZoomfaktor**(double zf) zoomt alle Komponenten innerhalb des Behälters.

Einsatz des Behälters am Beispiel Auto:

```
* Auto als Behälter
 public class AutoImBehaelter {
       private Behaelter auto;
                                              Deklaration des Behälters Auto
       private Rechteck reUnten;
       private Rechteck reOben;
       private Rechteck tuer;
       private Rechteck fenster;
       private Kreis radLi;
private Kreis radRe;
                                                   Erzeugen des Behälters,
       private Taktgeber takt;
                                                   gleichzeitiges Positionieren
       public AutoImBehaelter() {
             auto = new Behaelter(0, 150, 30^{\circ}0, 350);
             reUnten = new Rechteck(auto, 0, 100, 300, 100);
             reUnten.setzeFarbe("orange");
             reOben = new Rechteck (auto, 750, 0, 150, 110B) chälter setzen
             reOben.setzeFarbe("orange");
             tuer = new Rechteck(auto, 90, 10, 120, 180);
             tuer.setzeFarbe("blau");
             fenster = new Rechteck(auto, 100, 20, 100, 80);
             fenster.setzeFarbe("gelb");
             radLi = new Kreis(auto, 20, 170, 30);
             radLi.setzeFarbe("schwarz");
             radRe = new Kreis(auto, 220, 170, 30);
             radRe.setzeFarbe("schwarz");
       } // Ende Konstruktor
                                                  Bewegen des Behälters Auto
       public void bewege() {
                                                     Alle Objekte im Behälter
             auto.nachRechtsBewegen();
                                                             werden mitbewegt
       } // Ende bewege
} // Ende Klasse
```

Positioniert man den Behälter, so werden alle Objekte des Behälters mitbewegt.

Das Behälter-Objekt kann auf die gewünschte Größe gezoomt werden. Das funktioniert auch bei animierten Objekten!

Die Klasse Behälter besitzt ähnliche Methoden zum Bewegen wie die Klasse Rechteck, Dreieck,