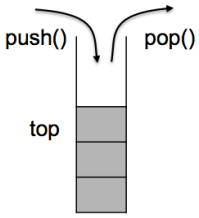
Die Klasse Stack in Java

Deklaration, Veranschaulichung mit GUI +

Anwendung “wohlgeformte Klammerausdrücke”



Die generische Klasse Stack <ContentType>

```
public class Stack <ContentType> {

    private class StackNode {
      private ContentType content = null;
      private StackNode nextNode = null;

      public StackNode(ContentType pContent) {
        content = pContent;
        nextNode = null;
      }

      public void setNext(StackNode pNext) {
        nextNode = pNext;
      }

      public StackNode getNext() {
        return nextNode;
      }

      public ContentType getContent() {
        return content;
      }
    }
```

Objekte der generischen Klasse **Stack** (Keller, Stapel) verwalten beliebige Objekte vom Typ **ContentType** nach dem **Last-In-First-Out**-Prinzip, d.h., das zuletzt abgelegte Objekt wird als erstes wieder entnommen.

*Ein Stackelement wird durch die intern deklarierte Klasse **StackNode** festgelegt – es besitzt einen **Inhalts**-Eintrag vom Typ ContentType sowie einen **Verweis** auf das nachfolgende Element (vom Typ StackNode).*

```
  private StackNode head;

  public Stack () {head = null;}

  public boolean isEmpty() {
    return (head == null);
  }

  public void push(ContentType pContent) {
    if (pContent != null) {
      StackNode node = new StackNode(pContent);
      node.setNext(head); head = node;
    }
  }

  public void pop() {
     if (!isEmpty()) {
       head  = head.getNext();
     }
   }

   public ContentType top() {
     if (!isEmpty()) {
       return head.getContent();
       } else {
         return null;
       }
   }
  }
```

Die Klasse StackGui (mit JavaFX) inkl. Anwendung

```
public class Controller {
   @FXML private TextField tfIsEmpty; … @FXML private Button btKopie;
   private Stack<String> s;
   public void btStack_click() {
      s = new Stack <String> (); gibAus(); btIsEmpty.setDisable(false); …
    }
   public void btIsEmpty_click() {
      if (s.isEmpty()) tfIsEmpty.setText("ja - leer!");
           else tfIsEmpty.setText("nicht leer!");
   }
   public void btPush_click() {
      s.push(tfPush.getText()); gibAus();
   }
   public void btPop_click() {
      s.pop(); gibAus();
   }
   public void btTop_click() {
      tfTop.setText("" + s.top());
   }
   public void gibAus() {
      Stack <String> s2 = new Stack <String> (); lv.getItems().clear();
      while (!s.isEmpty()){ lv.getItems().add(s.top()); s2.push(s.top()); s.pop(); }
      while (!s2.isEmpty()) { s.push(s2.top()); s2.pop(); }
   }
   public void btKopie_click() { String k = tfKlammern.getText(); tfKopie.setText(k); }
   public void btKlammern_click() {
      String k = tfKlammern.getText();
      //offene Klammern kommen auf den Stack
      if (!k.equals("") && (k.charAt(0)=='(' || k.charAt(0)=='[' || k.charAt(0)=='{')) {
         s.push(k.substring(0,1)); gibAus(); k = k.substring(1);mtfKlammern.setText(k);
      }
      //geschlossene Klammern werden mit oberstem Stack-Eintrag verglichen
      else if (!k.equals("") && !s.isEmpty() &&
            (k.charAt(0)==')' && s.top().equals("(") ||
            k.charAt(0)==']' && s.top().equals("[") ||
            k.charAt(0)=='}' && s.top().equals("{"))) {
         s.pop(); gibAus(); k = k.substring(1); tfKlammern.setText(k);
         //Stack leer, Textfeld leer, kein Fehler - also wohlgeformt :)!
         if (s.isEmpty() && k.equals("")) { tfKlammern.setText("wohlgef. :)"); }
      }
      //Fehler
      else { tfKlammern.setText("inakzept.!"); s = new Stack(); gibAus(); }
   }
}
```