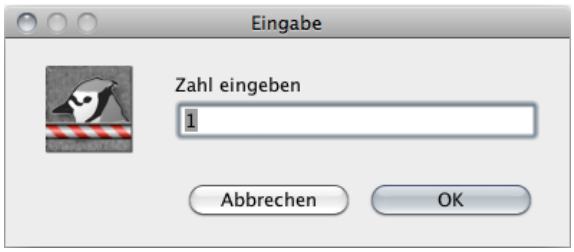
Einlesen von Eingaben

Bei eigentlich allen Programmen muss der Anwender Daten eingeben können. Leider ist die Erstellung von Oberflächen, wie man sie von allen Programmen kennt, allerdings nicht so einfach. Meist gibt es spezielle Programme dazu (JavaEditor) oder komplexe Software Entwicklungsumgebungen (sogenannte IDEs wie NetBeans von SUN), die aber für den Einstieg in Java viel zu komplex sind.

Sollen dennoch Daten eingegeben werden können, so gibt es mehrere Möglichkeiten, die hier nochmals zusammengefasst werden.

Eingabe über ein Dialogfenster

Zunächst muss die Swing-Bibliothek eingebunden werden:




```
import javax.swing.JOptionPane;
```

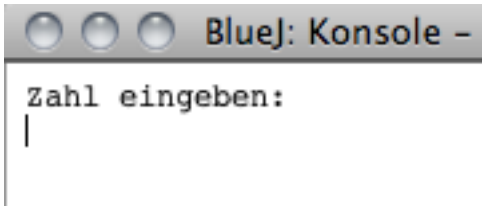
Anschließend lässt sich ein Eingabedialog (`inputDialog`) öffnen. Der erste Parameter ist dabei der beschreibende Text, der zweite Wert eine vorgegebene Zeichenkette, die bereits in der Eingabezeile stehen soll.

```
public void einlesen() {
    String eingabe = JOptionPane.showInputDialog("Zahl eingeben!", "1");
    try {
        int zahl= Integer.parseInt(eingabe);
    } catch (NumberFormatException e) {
        System.out.println(eingabe + " ist keine Zahl");
    }
}
```

Bei Interaktionen mit einem Anwender muss immer bedacht werden, dass der Anwender auch mal fehlerhafte Eingaben machen kann. So wird beispielsweise an Stelle einer Zahl ein Buchstabe eingeben. Darauf muss das Programm aber reagieren ohne das es gleich abstürzt. Hierzu gibt es die dargestellte `try...catch` -Formulierung.

Zunächst wird versucht (engl. try) die Eingabe in eine ganzzahlige Zahl umzuwandeln. Sollte es dabei zu einem Fehler (Ausnahme; engl: exception) kommen, so wird diese aufgefangen (engl. catch) und eine hilfreiche Bemerkung ausgegeben oder weitere Funktionen aufgerufen.

Eingabe über das Konsolenfenster



Im zweiten Fall muss das Ein- und Ausgabe-Paket (engl. InputOutput) von Java zunächst mit eingebunden werden.

```
import java.io.*;
```

Zunächst muss ein sogenannter Leser (engl. Reader) für den Eingabestrom (engl. InputStream) erzeugt werden. Dieser Strom an Zeichen wird in einem Zwischenspeicher abgelegt, man könnte auch sagen der Zeichenstrom wird abgepuffert, daher auch BufferdReader. Abschließend kann dann zeilenweise (engl. readLine) der Datenstrom eingelesen werden.

Auch hier erfolgt dann eine try-catch-Fehlerbehandlung für den Fall, dass der Benutzer aus Versehen etwas Falsches eingibt.

```
public void einlesen() {
      String eingabe="";
      try {
            BufferedReader input = new BufferedReader (
                  new InputStreamReader (System.in));
            System.out.println("Zahl eingeben:");
            eingabe = input.readLine();
      } catch (IOException e) {
            System.err.println(e.toString());
      }
      try {
            sa1 = Integer.parseInt(eingabe);
      } catch (NumberFormatException e) {
            System.out.println(eingabe + " ist keine Zahl");
      }
}
```

Andere Eingaben

Sollen andere Formate eingelesen werden, so kann stehen einem folgende Funktionen zur Verfügung:

Zielformat	Funktion
ganze Zahl	Integer.parseInt(String)
Kommazahl	Double.parseDouble(String)
Boolean (Wahrheitswert)	Boolean.parseBoolean(String)

Aufgaben