

# Aufgabe 4

## Sudoku-GUI

P-II SS 2006

Maximilian Störzer, Daniel Wasserrab, Dennis Giffhorn

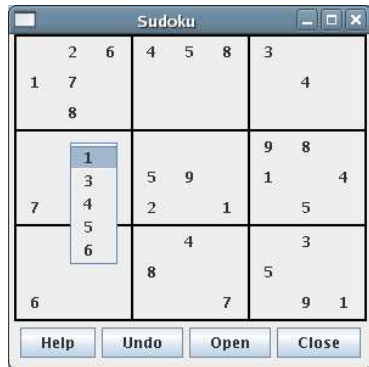
LS Softwaresysteme

30. Juni 2006



# Motivation

- Interaktives Sudoku-Lösungsprogramm
- basierend auf Struktur aus Aufgabe 3, d.h. Änderungen in Klassen (eigentlich) nicht nötig
- Ziele:
  - Arbeiten mit Swing-Komponenten wie Buttons, Menüs, etc.
  - sinnvolle Benutzerführung
  - Wiederverwenden von vorhandener Implementierung



Folgende Funktionen muss Benutzeroberfläche bereitstellen:

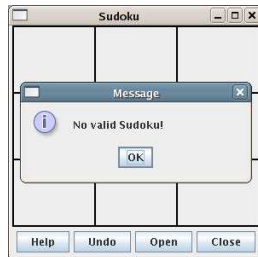
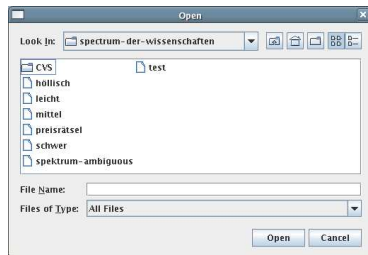
- Öffnen von Sudoku-Dateien
- Popup-Menü zur Auswahl der zu setzenden Zahl
- Hilfe-Funktion
- Undo-Funktion
- Schliessen der GUI



# Funktionalität: Sudoku-Dateien öffnen

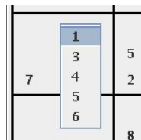


- Dateiformat wie in Aufgabe 3, also  $9 \times 9$  Matrix aus Zahlen bzw. Punkten
- GUI-Klasse für entsprechende Operation: `JFileChooser` stellt GUI für Dateiauswahl bereit
- Nach Einlesen Sudoku sofort darstellen
- falls ungültiges Startbrett
  - Fehlermeldung mittels Dialogbox
  - keine Anzeige des Sudokus
  - kein Zustand wird angelegt



# Funktionalität: Auswahl von Zahlen

- intelligentes Board kennt alle Möglichkeiten für ein Feld
- bei Rechtsklick auf leeres Feld Anzeige dieser Möglichkeiten
- Auswahl einer Möglichkeit setzt diese in das Feld
- Rechtsklick auf Feld ohne Möglichkeiten öffnet Dialogbox mit entsprechender Fehlermeldung
- Setzen der letzten Zahl öffnet Dialogbox mit Benachrichtigung, dass Sudoku gelöst



# Funktionalität: Hilfe

- Auswahl der Hilfe setzt Zahl aus Lösung
  - jedoch nicht beliebig, sondern Zahl, die in der aktuellen Situation weiterhilft
  - Realisierung:
    - Lösen des aktuellen Zustands mittels `solve(State s)`
    - dazu: neue Implementierung des Interfaces `State`
    - merkt sich die jeweils erste gesetzte Zahl beim Lösen
    - verwaltet einen konkreten Zustand wie in Aufgabe 3 und delegiert Methodenaufrufe an diesen weiter
- Beispiel (angenommen, `foo` sei Methode des Interface `State`):

```
public int foo(Structure s, boolean b) {  
    \\  
    // evtl. zusätzliche Berechnungen  
    return state.foo(s, b);  
}
```

- Aufruf Hilfe auf unlösbarem Sudoku:  
Dialogbox mit Fehler



# Funktionalität: Undo/Schliessen

Undo:



- entfernt letzte gesetzte Zahl
- dazu: Stack mit bisherigen Zuständen im Programmablauf
- Setzen einer Zahl: **push**, Undo: **pop**
- Stichwort: Klonen von Zuständen
- Nichtimplementierung  $\Rightarrow$  Funktionalitätsnote max. 'B'

Schliessen:

Klick auf 'x' soll GUI schliessen



- besteht aus 9 Blöcken (getrennt durch Linien)
- jeder Block besteht aus 9 Feldern für eine Zahl
- sinnvoll: Unterklasse von `JPanel` für 9 GUI-Elemente
- Nur Repräsentation des aktuellen Programmzustands (auf Konsistenz achten!)
- jeweils ein Button für Öffnen von Dateien, Hilfe, Undo und Schliessen
- Beispiel: siehe Demo



Fragen?

# Fragen?

