

1		3
4		6
	8	

Abschlusspräsentation Studienarbeit SbS-Sudoku

Matrikelnummern: 3819525, 7750470

www.dhbw-stuttgart.de

Gliederung

Aufgabenstellung
Verwendete Technologien
Datenübertragung
Design
Testkonzept
Herausforderungen
Fazit
Ausblick

2

- Aufgabenstellung
- Verwendete Technologien
- Datenübertragung
- Design
 - Allgemeines Design Frontend
 - Seitenablauf
- Testkonzept
- Herausforderungen
- Fazit
- Ausblick
 - Weitere Entwicklungsmöglichkeiten

Aufgabenstellung

Sudoku Helper
Unterstützung für jedes mögliche Sudoku
Schrittweises Heranführen an die Lösung

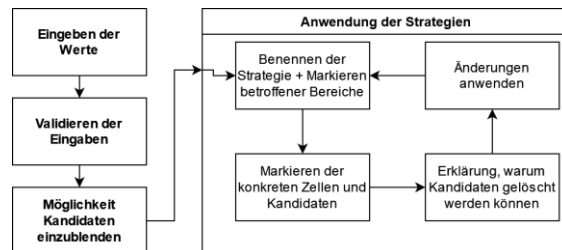


Abbildung 1: Bedienschritte des Programms

- Entwicklung eines Sudoku-Helpers
- Soll Nutzer Schritt-für-Schritt an die Lösung heranführen
- Abfolge
 - Öffnen der Webseite und eingeben der bekannten Werte
 - Validierung, ob es sich um ein echtes Sudoku handelt
 - Möglichkeit Kandidaten einblenden zu lassen
 - Anwenden der Strategien:
 - Hilfe 1: Benennen der anzuwendenden Strategie, sowie hervorheben der Bereiche auf die die Strategie angewendet wird
 - Hilfe 2: Markieren der konkreten Zellen und Kandidaten auf die die Strategie angewandt wird
 - Hilfe 3: Erklärung, warum bestimmte Kandidaten laut der Strategie gestrichen werden können
 - Hilfe 4: Kandidaten löschen bzw. Änderungen anwenden
- Unterstützung für jedes mögliche Sudoku -> ca. 25 Strategien

Verwendete Technologien

Implementierung als Webanwendung

Backend:

- Python
- Django

Frontend:

- Bootstrap
- JQuery

Versionsverwaltung:

- Github



Abbildung 2: Logos der verwendeten Technologien

4

- Implementierung als Webanwendung, nicht als App
 - (mehr Erfahrung)
- Backend:
 - Python als Programmiersprache
 - Interpretiert -> Nicht immer erneutes kompilieren erforderlich
 - Objektorientiert
 - Vergleichsweise viel Erfahrung
 - Django als Framework
 - Quelloffenes Web-Framework
 - Dynamische HTML-Generierung
- Frontend
 - Kein Java-Script Framework (wie vue oder so)
 - Relativ simple Aufgaben
 - Keine Kenntnisse vorhanden -> Einarbeitungszeit hätte Zeitgewinn wahrscheinlich wieder zunichte gemacht
 - Bootstrap
 - CSS Library
 - Stellt diverse UI-Komponenten bereit
 - Elemente sind so konzipiert, dass sie auf allen Bildschirmgrößen

- funktionieren
 - Umfangreiche Tutorials
- JQuery
 - JavaScript Library
 - Vereinfacht die Manipulation der DOM stark
 - Verkürzt den Code gegenüber Standard-JS
- Versionsverwaltung:
 - Github
 - Verteiltes System basierend auf GIT
 - Jeder Nutzer arbeitet mit eigenständiger Kopie des Projektes
 - Synchronisation über die Github-Server

Bildquellen:

- *Python:* <https://de.wikipedia.org/wiki/Datei:Python-logo-notext.svg>
- *Django:* <https://de.wikipedia.org/wiki/Datei:Django-logo.svg>
- *Bootstrap:* https://de.m.wikipedia.org/wiki/Datei:Bootstrap_logo.svg
- *JQuery:* <https://commons.wikimedia.org/wiki/File:jQuery-Logo.svg>
- *GitHub:* <https://github.com/logos>

Datenübertragung

```
<input readonly id="2_1" name="2_1" type="number" value="7" maxlength="1"/>
```

Abbildung 3: vereinfachtes HTML für ein Sudoku-Eingabefeld

```
<input id="candidates_3_4" name="candidates_3_4" value="[7, 8, 9]" readonly/>
```

Abbildung 4: vereinfachtes HTML für ein Eingabefeld,
in dem die Kandidaten für ein Feld zwischengespeichert werden

5

Frontend zu Backend

- Werte des Sudokus
 - Über HTML-Eingabefelder
 - Über Attribut „Name“ und die „ID“ werden die Koordinaten des Feldes in dem Sudoku festgehalten
 - Wenn Werte nicht vom Nutzer eingetragen werden, als readonly Feld und Wert wird über Django Templates dynamisch eingetragen („value“)
- Jeweiligen Kandidaten
 - Wichtig, da Algorithmen ggf. nur Kandidaten ändern (-> „sichern“ über das FE)
 - Auch über input-Feld (Koordinaten auch wieder über Name + ID)
 - Value enthält die Liste der Kandidaten als JSON-Darstellung
 - Wird nicht angezeigt, da irrelevant für Frontend

Datenübertragung

```
return (True, {
    'algorithm': 'hidden_pair',
    'values': [5,6],
    'reason': 'row',
    'removed_candidates': { 34: [1,3], 35: [1,2,3,7]}
})
```

Abbildung 5: Beispiel für die Rückgabewerte eines Algorithmus

Algorithmus Ergebnisse ans Frontend

- Wahrheitswert, ob Algorithmus erfolgreich
- Informationen die für die Algorithmusanzeige wichtig sind werden in Form von einer Menge an Key-Value-Paaren (Dictionary) an das Frontend übertragen
 - Im Backend über JSON serialisiert und in ein Template hereingerendert
 - Im Frontend wieder deserialisiert über JSON
- Beispiel;
 - Welcher Algorithmus
 - Welche Werte verwendet der Algorithmus
 - Was für ein Typ von Einheit ist betroffen
 - Wo werden welche Kandidaten gelöscht

Design - Allgemeines

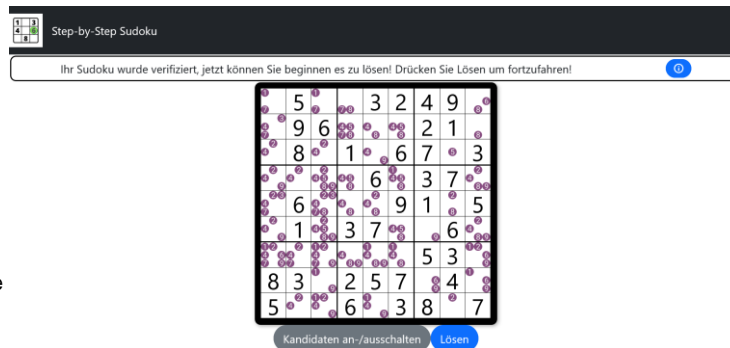


Abbildung 6:
Eine Webseite
der
Anwendung

7

- Allgemeine Seitenaufbau (Grundgerüst, dass bei allen Seiten gleich ist)
 - Leiste mit Logo und Namen der Software
 - Schnellinfo, mit kurzen Informationen, was passiert
 - Lässt sich in Pop-Up aufklappen
 - Sudoku mit Werten und Kandidaten
 - Wächst mit Bildschirm mit

Design - Seitenablauf

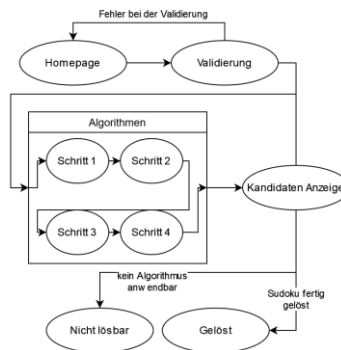


Abbildung 7:
Ablauf der Webseiten

- Start auf Homepage
- Validierung (wenn Fehler zurück zu Homepage)
- Algorithmen (mit den Schritten)
- Anzeige der Kandidaten (fertig gelöst: Gelöst, kein Alg. Anwendbar: Nicht lösbar, sonst: wieder Algorithmen)

Testkonzept (Backend)

Unittests:

- Backtracking
 - Bei dem Testen des Backtracking Algorithmus werden mehrere Sudokus verwendet, bei denen die Anzahl der Lösungen bekannt ist.
- Algorithmen
 - Dies testet, ob im Allgemeinen ein Algorithmus eine Lösung findet, kann aber die vollständige Korrektheit eines Algorithmus nicht gewährleisten.
- Vollständige Tests
 - Versucht, mithilfe der implementierten Algorithmen eine Reihe an vorgegebenen Sudokus zu lösen.

Testkonzept (Frontend)

Integrationstest

Funktionstest

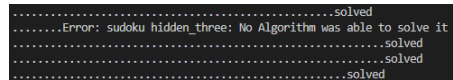
Reaktion auf Eingaben wird getestet

Alle Seiten getestet (außer „Nicht lösbar“ und „Kandidaten Anzeigen“), da keine Interaktion möglich.

Testkonzept (Ergebnisse)

Backend

- X-Kette hat einen Fehler der nicht behoben werden konnte
- Vollständige-Tests
 - Schwer: 0/3 gelöst
 - Mittel: 2/2 gelöst
 - Variierende Schwierigkeit: 20/26 gelöst



```
.....solved
.....Error: sudoku hidden three: No Algorithm was able to solve it
.....solved
.....solved
.....solved
```

Abbildung 8: Vollständige Tests

Frontend

- Von den Frontendtests laufen alle Fehlerfrei durch. Das Frontend und die Interaktion zwischen Frontend und Backend laufen also im Rahmen der Testabdeckung so wie gewünscht.

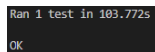
Herausforderungen

Datenübertragung

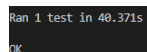
Frontend-Design (mit Kandidaten)

Performerter Backtracking Algorithmus

- 17 Felder gegeben
- 18 Felder gegeben



```
Ran 1 test in 103.772s
OK
```



```
Ran 1 test in 40.371s
OK
```

Abbildung 9: Backtracking Test

Debugging der Algorithmen

- Typfehler

Fazit & Ausblick

Visualisierung Frontend

Algorithmen Testen

X-Kette fehlerhaft

Nur 22 von 31 Sudokus können gelöst werden

Schritt für Schritt Hilfe funktioniert

Weitere Entwicklungsmöglichkeiten

- Weitere Algorithmen implementieren