Projektarbeit Informatik

Softwaredokumentation   
LaserChess V2.0



Website: <http://stocyr.github.com/LaserChess/>

Dozent: Ivo Oesch

Autoren: Marcel Bärtschi, Jascha Haldemann, Nicola Käser, Cyril Stoller

2 Semester 2012

**Zusamenfassung**

**Inhaltsverzeichnis**

1 Aufgabenstellung 1-1

2 Richtlinien 2-2

3 Analyse 3-3

4 Design 4-4

4.1 Grobdesign 4-4

4.2 Detaildesign 4-4

4.3 Modulbeschreibungen 4-4

4.4 Datenstrukturen 4-4

5 Implementation 5-5

6 Test 6-6

7 Anhang 7-7

# Aufgabenstellung

# Richtlinien

Für dieses Projekt gelten folgende Richtlinien:

Jedes Modul besteht grundsätzlich aus zwei Dateien, einer Schnittstellendefinitionsdatei (Headerdatei .h-File) und einer Implementationsdatei (.c-File). Die main()-Fuktion ist in der Haupt-C-Datei LaserChess.C enthalten. Alle projektweite Typendefinitionen sind in der Datei LaserChess.H enthalten. Die Namen der Header und der Implementationsdatei entsprechen jeweils dem Modulnamen, soweit möglich und sinnvoll.

Der Selbstschutz in den Headerdateien erfolgt in der Form

#ifndef NAME

#define NAME

<Schnittstellendefinition>

#endif

Wobei NAME wie folgt aufgebaut ist DATEINAME\_H. Jedes Modul ist mit einem Modulheader ausgestattet, jeweils in der Schnittstellen und der Implementationsdatei. Der Header enthält den Modulnamen, den Dateinamen, einen kurzen Funktionsbeschrieb zum Modul, die Namen aller zur Verfügung gestellten Funktionen und den Namen des ursprünglichen Autors. Die History jedes Files kann auf <https://github.com/stocyr/LaserChess/tree/master/src> angesehen werden. Zuerst klickt man auf den Namen des gewünschten Files, dann auf den Button Blame oben rechts.

Jede Funktion ist mit einem Funktionsheader ausgestattet. Der Header enthält den Funktionsnamen, einen kurzen Funktionsbeschrieb, die Namen und Funktion der Argumente, eine Beschreibung des Rückgabewertes, und den Namen des Autors. Funktionen sollen klar definierte Aufgaben haben. Namen von Bezeichnern sollen Aussagekräftig sein, und die Funktion des entsprechenden Objektes erklären oder andeuten.

Modul- und Funktionsheader werden in englischer Sprache geschrieben. Kommentare im Code drin jedoch sind auf Deutsch verfasst und immer in der Art // <Kommentar>, damit ganze Codeblöcke zu debug-Zwecken mit /\* ... \*/ auskommentiert werden können. Der Code ist grundsätzlich mit sinnvollem und aussagekräftigem Kommentar zu versehen.

Makros (#define) werden grundsätzlich in Grossbuchstaben geschrieben, zu Strukturierung können Underscores verwendet werden. Beispiele: PI, MAXIMAL\_FIELD\_WIDTH

Enums werden grundsätzlich mit einem grossen Anfangsbuchstaben geschrieben. Structs werden grundsätzlich klein geschrieben. Variablen werden grundsätzlich klein geschribenen und nötigenfalls mit Underscore "\_" in Wörter aufgeteilt. Funktionen werden grundsätzlich klein geschribenen und nötigenfalls mit Underscore "\_" in Wörter aufgeteilt.

Für jedes Modul ist grundsätzlich ein Programmierer verantwortlich. Dieser ist für die Sauberkeit und das Nachführen des Modulheaders zuständig. Andere Programmierer dürfen jedoch ebenfalls Änderungen in diesem Modul vornehmen.

# Analyse

# Design

## Grobdesign



## Detaildesign

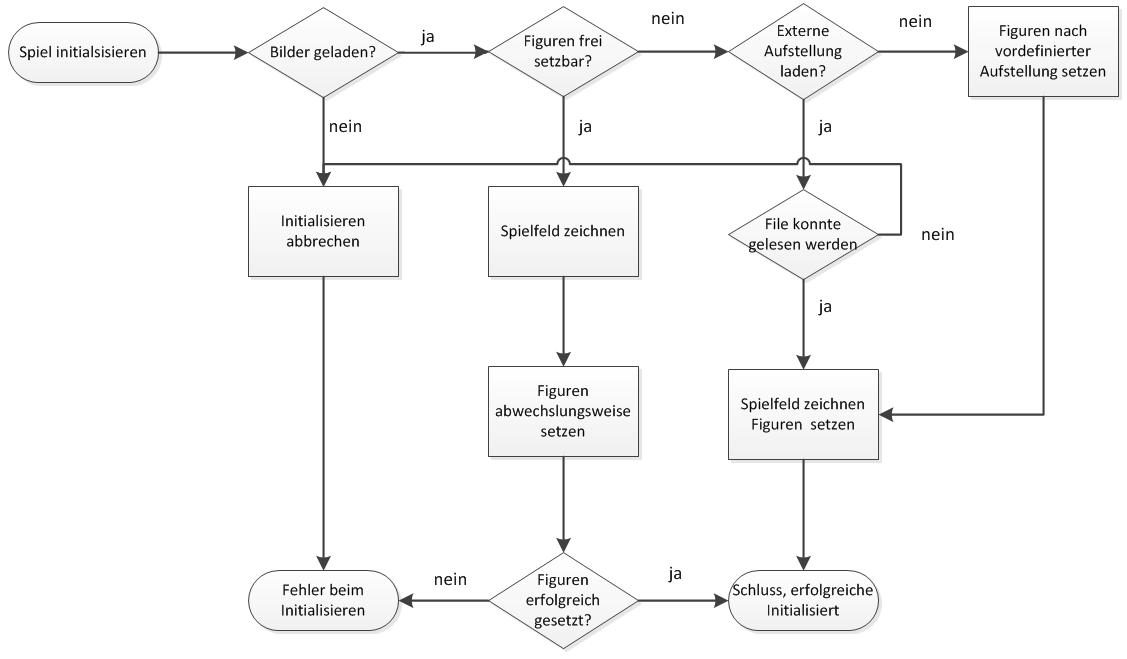
## Modulbeschreibungen

## Struktogramme und Flussdiagramme

### Modul LaserChess.c



### Modul Spiel.c



### Modul Logik.c

#### laser()



#### is\_inside\_map()



#### is\_figure()



#### move\_figure()



#### destroy\_figure()

G:\BFH\Struktogramme Logik.c\destroy_figure.png

#### mouseclick\_to\_map()



#### path\_handler()



#### map\_extension\_handler()



#### play\_sound()



### Modul Grafik.c

## Datenstrukturen

# Implementation

# Test

# Anhang

1. Pflichtenheft
2. Zeitplanung
3. Structured Designe (CRC)
4. Styleguideline