17.12.2014

Martin Weber, Jan-Philipp Steinwender, Lukas Sträßler

Lights Out

SEW-Teamprojekt

Inhalt

[Aufgabenstellung 0](#_Toc406609273)

[Kompetenzen 0](#_Toc406609274)

[Grundlegende Aufgabe 0](#_Toc406609275)

[Team 0](#_Toc406609276)

[Aufwandsschätzung 0](#_Toc406609277)

[UML-Design 0](#_Toc406609278)

[Ausführung 0](#_Toc406609279)

[Model 0](#_Toc406609280)

[View 0](#_Toc406609281)

[Controller 0](#_Toc406609282)

[Entwicklungsumgebung 0](#_Toc406609283)

[Installation 0](#_Toc406609284)

[Benötigte Zeit 0](#_Toc406609285)

[GitHub 0](#_Toc406609286)

# Aufgabenstellung

## Kompetenzen

* Folgende Kompetenzen wurden trainiert:
* MVC – Prinzip
* Umgang mit GitHub
* Teamarbeit

Grundlegende Aufgabe

Es soll eine Readme Datei erstellt werden in der das Team und die Arbeitsaufteilung definiert wird.

Programmieren sie das Spiel "Lights Out"

Funktion des Spiels: Ein 5\*5 großes Feld ist mit Buttons gefüllt. Jeder Button kann entweder den Zustand ein oder aus haben. Sobald ein Spiel gestartet wird werden zufällig viele Buttons auf den Zustand „ein“ gesetzt. Wird nun ein Button gedrückt invertieren alle Buttons um diesen herum und er selbst den Zustand. Das Ziel ist alle Buttons auf den Zustand aus zu setzten.

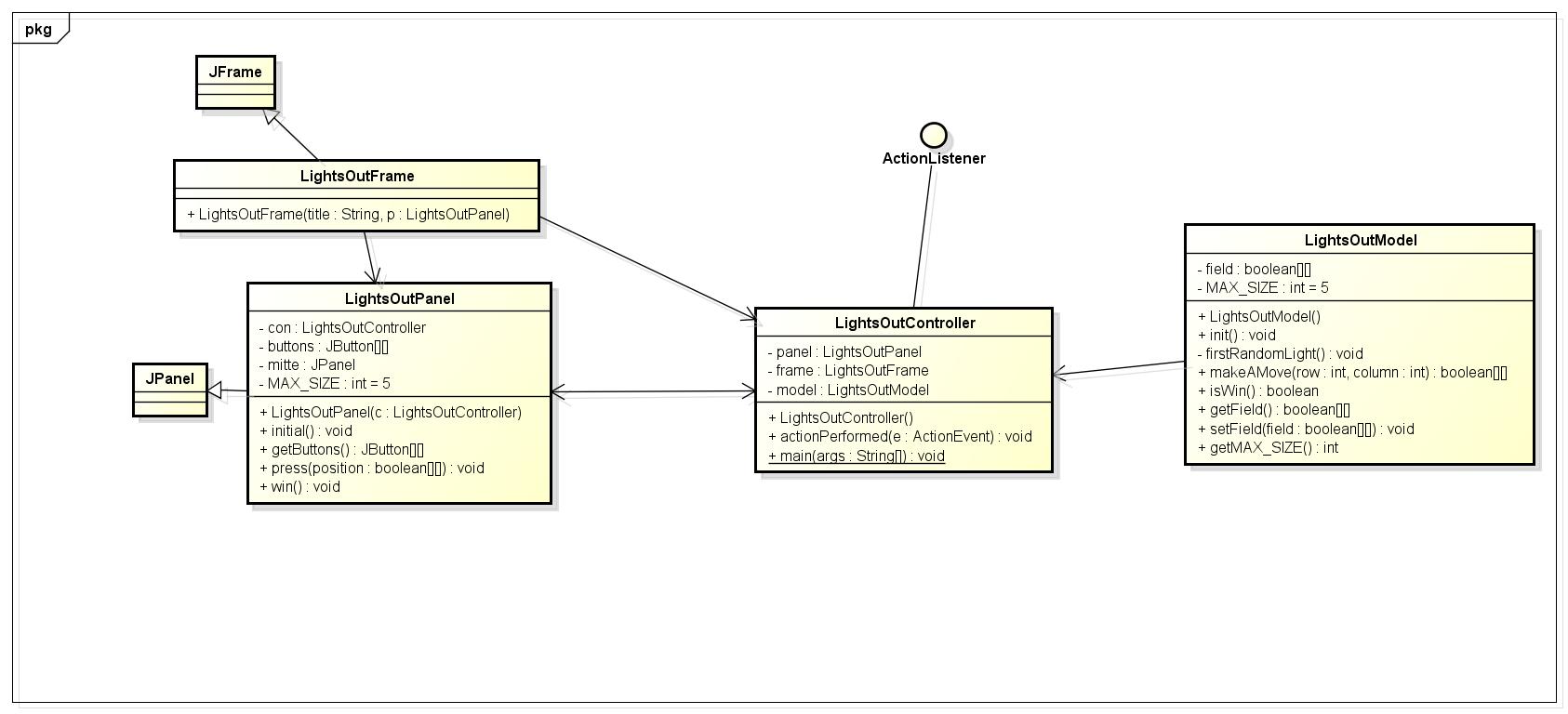
## Team

Das Team besteht aus 3 Personen, wie in der Vorgabe festgelegt wurde. Die Arbeitsaufteilung war frei wählbar und konnte im Team besprochen und festgelegt werden.   
In unserem Fall wurde festgelegt, dass jede Person eine Komponente des MVC - Prinzips übernimmt.

# Aufwandsschätzung

|  |  |  |  |
| --- | --- | --- | --- |
|  | Weber | Sträßler | Steinwender |
| Model |  | 4 | 0 |
| View |  |  | 4 |
| Controller | 1 |  | 0 |
| Management | 2 |  | 0 |
| Stunden pro Person | 3 |  | 4 |
|  |  | Gesamt: | 7+Sträßler |

# UML-Design



# Ausführung

## Model

Das Model enthält die Spiellogik die für die Umsetzung benötigt wird. Das Model wurde so geschrieben, dass später andere Spielfeldgrößen benutzt werden können. Es gibt 2 Attribute. Eine int Konstante und ein boolean Array welches das Spielfeld darstellt. Eingeschaltete Buttons werden mittels **true** und **false** realisiert.   
Die Methode makeAMove schaltet mit Hilfe von der Position des gedrückten Buttons die richtigen Lichter ein und aus.

Beim initialisieren des Spiels werden zufällige Buttons eingeschalten. Dies erfolgt durch mehrmaliges aufrufen (1-50 mal) der makeAMove Methode in der init Methode und simulieren von zufälligen drücken eines Buttons durch den Computer. Somit ist das Spiel auch immer lösbar.

## View

Das View Element beinhaltet die Klasse LightsOutFrame und LightsOutPanel.

Die Frame Klasse ist für die groben Einstellungen des Fensters zuständig, wie zum Beispiel: Größe des Fensters.

Die Panel Klasse ist für den Inhalt des Fensters verantwortlich. In der Klasse gibt es eine Konstante um die Anzahl der Felder zu bestimmen. Die Klasse verfügt über eine initial Methode in der nur die Buttons erstellt und auf schwarz gestellt werden. Weiters gibt es eine Methode (press()) in der die Farbe der Buttons geändert wird. Entweder von schwarz auf weiß oder von weiß auf schwarz. Zusätzlich beinhaltet sie eine Methode für ein Gewinn-Popup.

## Controller

Im Controller wird die graphische Komponente mit der logischen Komponente verknüpft. Er enthält außerdem den ActionListener um auf im Panel auftretende Aktionen reagieren zu können. Falls ein neues Spiel gestartet wird, ruft der Controller die init Methode im Model auf um neue Werte zu berechnen. Diese werden dann an die View Componente übergeben, damit diese das Spiel darstellen kann. Sobald ein Button des Spiels ausgelöst wird liest der Controller aus dem ActionCommand des Buttons seine Position aus und speichert sie in 2 int Variablen, die er dem Model übergibt und aktualisiert danach das Spiel mit den neuen Werten in den er diese an die View übergibt. Am Schluss jedes Zuges wird überprüft ob alle Lichter aus sind und die Person gewonnen hat. Falls ja wird die Gewinnmeldung im View aufgerufen und ein neues Spiel im Model berechnet und an die View übergeben.

# Entwicklungsumgebung

Das Programm wurde in Eclipse unter Windows 8.1 entwickelt.   
Die verschiedenen benutzten Versionen von Eclipse waren: Luna.

# Installation

Um das Programm ausführen zu können wird eine aktuelle JRE vorausgesetzt. Diese kann gratis unter <http://www.java.com/de/> heruntergeladen werden.

Danach muss die aktuelle Version 1.3 unter dem folgenden Link heruntergeladen werden:

<https://github.com/MartinWeber02/weber_straeszler_steinwender_A08/releases/tag/1.3>

Diese JAR-Datei kann dann auf dem Rechner ausgeführt werden.

# Benötigte Zeit

|  |  |  |  |
| --- | --- | --- | --- |
|  | Geplant | Tatsächlich | % |
| Weber | 3 | 4 | 125 |
| Sträßler | 4 | 5 | 125 |
| Steinwender | 4 | 5 | 125 |
| Gesamt | 11 | 14 | 125 |

# GitHub

Die Zusammenarbeit an dem Code im Team wurde durch GitHub deutlich vereinfacht. Dort wurde ein gemeinsames Repository eingerichtet welches unter dem folgenden Link gefunden werden kann:

<https://github.com/MartinWeber02/weber_straeszler_steinwender_A08>