|  |
| --- |
| TGM |
| Let’s play Nonogramm |
| 5BHITM |
|  |
| **Krickl | Seckin** |
| **17.01.2015** |

|  |
| --- |
|  |

Let’s play Nonogramm

Inhalt

[Aufgabenstellung 2](#_Toc409429648)

[Zeitaufzeichnung 3](#_Toc409429649)

[Allgemein 4](#_Toc409429650)

[Design 4](#_Toc409429651)

[Probleme 4](#_Toc409429652)

[Logik 4](#_Toc409429653)

[GUI 4](#_Toc409429654)

[Quellen 5](#_Toc409429655)

[Nachschlagen 5](#_Toc409429656)

[Links vom Text 5](#_Toc409429657)

# Aufgabenstellung

Nachdem Sie einige Designpattern in Python betrachtet haben, wollen wir uns einem wichtigen Entwurfmuster spielerisch nähern:

MVC



In einem Team (2) soll das Spiel Nonogramm umgesetzt werden.

* Spielfeld: 15 x 15
* Eine Statusleiste mit Anzeige der noch gesuchten Felder,
* Button zur sofortigen Lösung
* Button zum Neustart
* Auswahlfeld zur Einstellung der Schwierigkeit (EASY/200; MEDIUM/150; HARD/125; EXPERT/90; IMPOSSIBLE/50) auf Basis der gesuchten Felder!

Die Farbe rosa ist natürlich nicht Pflicht und könnte vielleicht vom User variabel eingestellt werden.

Viel Erfolg!

Ressourcen:

Unterlagen zu GUI-Programmierung in Python

https://de.wikipedia.org/wiki/Nonogramm

# Zeitaufzeichnung

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Geschätzte Zeit in h** | **Tatsächliche Zeit in h** | **Verantwortung** |
| Recherchieren und Nachschlagen | 6 |  | Krickl & Seckin |
| Algorithmus zum Generieren eines Nonogramms entwickeln |  |  |  |
| GUI mit pyQt erstellen |  |  |  |
| Import der GUI in PyCharm |  |  |  |
| Zusammenfügen von View und Control |  |  |  |
| Actionhandling implementieren |  |  |  |
| Debuggen |  |  |  |
| Dokumentation des Codes |  |  |  |
| Protokoll erstellen |  |  |  |
| Summe Krickl |  |  |  |
| Summe Seckin |  |  |  |

# Allgemein

Wie funktioniert Nonogramm?

„Das Spiel besteht aus einem Gitter aus beliebig vielen Kästchen. Ziel ist es, die Zellen eines Gitters so einzufärben (bzw. nicht einzufärben), dass die eingefärbten Kästchen in jeder Zeile und Spalte der dafür angegebenen Anzahl und Gliederung entsprechen. Die Zahlenfolge „4 2 1“ vor einer Zeile enthält beispielsweise die Information, dass in dieser Zeile (mit mindestens einem Kästchen Abstand) ein Block von vier zusammenhängenden Zellen, ein Block von zwei zusammenhängenden Zellen sowie eine einzelne Zelle in dieser Reihenfolge einzufärben sind. Aus der Kombination von Zeilen- und Spaltenangaben lässt sich eine (meist eindeutige) Lösung logisch herleiten.“ [1]

# Design

Unser Design haben wir wie folgt aufgebaut:

# Probleme

## Logik

Man findet sehr viele Algorithmen zur Lösung von Nonogrammen, jedoch aber wenige Ansätze zum Generieren. Ein Ansatz war ein 8-Bit Bild zu nehmen und aus dem ein Nonogramm zu generieren. Die von mir gewählte einfachere Methode funktioniert mit einer zufälligen Befüllung der Felder und nachträgliche Auszählung. [2]

Da man bei der Generierung das Spielfeld einfach speichern kann, braucht man keinen Lösungsalgorithmus und vergleicht das aktuelle Spielfeld mit dem Lösungsspielfeld.

## GUI

In dem Spiel gibt es drei Zustände jedes einzelnen Spielfelds: Schwarz, Weiß oder Unbekannt.

Dies wurde grafisch folgendermaßen umgesetzt:

# Quellen

*Kompletter Code siehe GitHub*  
https://github.com/akrickl-tgm/nonogramm.git

## Nachschlagen

Codeacademy – Learning Python  
http://www.codecademy.com/

http://www.entwickler-ecke.de/topic\_NonogrammGriddler++Solver\_60215,0.html

http://sourceforge.net/p/freenono/tickets/milestone/FreeNono%201.0/

## Links vom Text

[1] Nonogramm   
URL: http://de.wikipedia.org/wiki/Nonogramm   
aufgerufen am 19.1.2015

[2] Create a Nonogramm Puzzle  
http://codegolf.stackexchange.com/questions/30081/create-a-nonogram-puzzle  
aufgerufen am 17.1.2015