Bimaru-Dokumentation

# M:\Desktop\BimaruKlassendiagramm.pngKlassendiagramm

# Struktur

## Überblick

Grundsätzlich wurde versucht die Applikation nach dem MVC-Prinzip zu implementieren.

Dadurch ist die Klasse „GameGridModel“ entstanden, die den gesamten Zustand des aktuellen Spiels speichert. Die einzelnen Views wurden in verschiedenen Klassen unterteilt, welche dann in der Hauptklasse „BimaruGame“ eingebunden und dargestellt wurden.

## GameGridModel

Diese Klasse enthält das aktuell geladene Bimaru-Rätsel, sowie die von Spieler gesetzten Felder.

Die vom Spieler gesetzten Felder werden in einem 2D-Array gespeichert. Die Werte 0, 1, 2 repräsentieren die möglichen Feldzustände (Leer, Wasser und Schiffsteil).  
Das Rätsel selbst hingegen wird als einzelne Schiffs(„Ship“)- und Hinweis(„Hint“)-Objekte gespeichert.   
Die Schiffe bestehen aus der Position der Schiffsspitze, Länge und Orientierung des Schiffes. Die Hinweise bestehen nur aus der Position, welche in der View danach als konstant aufgedeckt dargestellt wird.

Das Model beinhaltet dazu noch verschiedene Funktionen z.B. zum Togglen der Felder oder zum Abfragen des Feldstatus.

Der Konstruktor sowie die Getter- und Setterfunktionen wurden so programmiert, dass wir uns die JAXB-Schnittstelle zu Nutze machen können, um das Model aus XML-Dateien zu laden.

## GameGridView

Diese Klasse stellt das komplette Spielfeld dar, welches aus einzelnen Felder (JButtons) und Schiffszähler (JLabels) am Rand besteht.

Zu Beginn holt es aus dem Model die Spielfeldgrösse, und erstellt entsprechend viele Felder, welche in einem GridLayout dargestellt werden.

Wird ein Feld im Model getogglet, merkt die View das (Observer-Pattern) und aktualisiert die Darstellung der Felder im Umkreis des betreffenden Felds auch. Dies ist nötig um z.B. entstehende Konflikte oder konkrete Schiffsteile richtig darzustellen.  
Auch werden dabei die Zähler aktualisiert und entsprechend farbig dargestellt.

## FieldButton

Diese Klasse stellt ein einzelnes Feld dar, und wurde als modifiziertes JButton implementiert.  
Bei Klick oder Drag auf das Feld wird im Model das Togglen des Felds ausgelöst.  
  
Die Felder erkennen eigenständig, anhand der Felder in ihrer Umgebung, ob ein Schiffsteil konfliktierend oder (nicht) definiert platziert wurde, und stellt diese Schiffteile entsprechend dar.