# Erklärung des MVC-Patterns:

Unser Programm ist (wie gefordert) unter Anwendung des MVC-Schemas geschrieben worden.

Model:

Zum Model gehören die Packages net.lighthouse.levels und net.lighthouse.model . Ihre Visibility ist public, da das Model und alle Modelobjekte in Controller, Collider und View benutzt werden.

Im Model Package sind:

1. alle Objekte, die in unserem Breakout vorhanden sind (BBall, BBlock mit Collection BlockList, BBoss, Blaser, BPaddle und BText, sowie das Grundobjekt BObject.
2. Eine Klasse, die den momentanen Status des Spiels darstellen kann, indem sie Informationen über momentan vorhandene Objekte speichert und Methoden zum manipulieren stellt (MainModel)

Zu 1.:

Das Grundobjekt ist das BObject. Es hat eine Position, Dimensionen (Höhe/Breite) und eine Farbe.

Alle Objekte im Spiel erben von BObject. Daher haben sie auch die oben genannten Eigenschaften. Außerdem hat

* BBall eine interne Repräsentation der Geschwindigkeit des Balls, Getter und Setter dafür, eine move Methode und und eine Methode die die voraussichtliche nächste Position returnt.
* BBoss eine interne Darstellung seiner HP
* BPaddle eine move Methode auf x.
* BText eine interne Darstellung von seinem Text und Getter+Setter dafür.

BBlock ist ein BObject und hat keine zusätzlichen Methoden. Blaser ist ein BBall und hat keine zusätzlichen Methoden.

BlockList extended ArrayList<BBlock> und stellt die Methode getBlockAtXY, mit der man einen Block an einer bestimmten Stelle holen kann.

Zu 2.:

MainModel ist die Klasse die bei uns den momentanen zustand des Spiels darstellt (Model). Sie

* hat Listen für alle Objekte, die im Spiel vorkommen können (außer Paddle, es gibt nur ein Paddle), sowie Getter und Setter für diese Listen.
* merkt sich den userScore.
* stellt mit addObject eine Methode etwas dem Model hinzuzufügen.
* Stellt mit contains eine Methode um zu testen, ob ein bestimmtes Objekt im Model ist.
* Ist iterable. Wenn iterator() aufgerufen wird, werden alle Objekte, die momentan im Model sind mit toArrayList Methode zu einer Arraylist hinzugefügt und der iterator dieser ArrayList zurückgegeben.

View:

Zur View gehört das Package net.lighthouse.view.

In diesem Package ist das Interface „View“. Dieses Interface definiert das Verhalten einer MainView. Eine solche MainView muss

* Eine init() Methode haben, die das GraphicsProgram Fenster rescaled und