Sprint 4 Report

TI2206 - Software Engineering Methods

Karel van Doesburg
Robert Levenbach
Pascal Lubbers
Bas Stinenbosch
Sebastiaan Vermeulen

...presenting Tetris

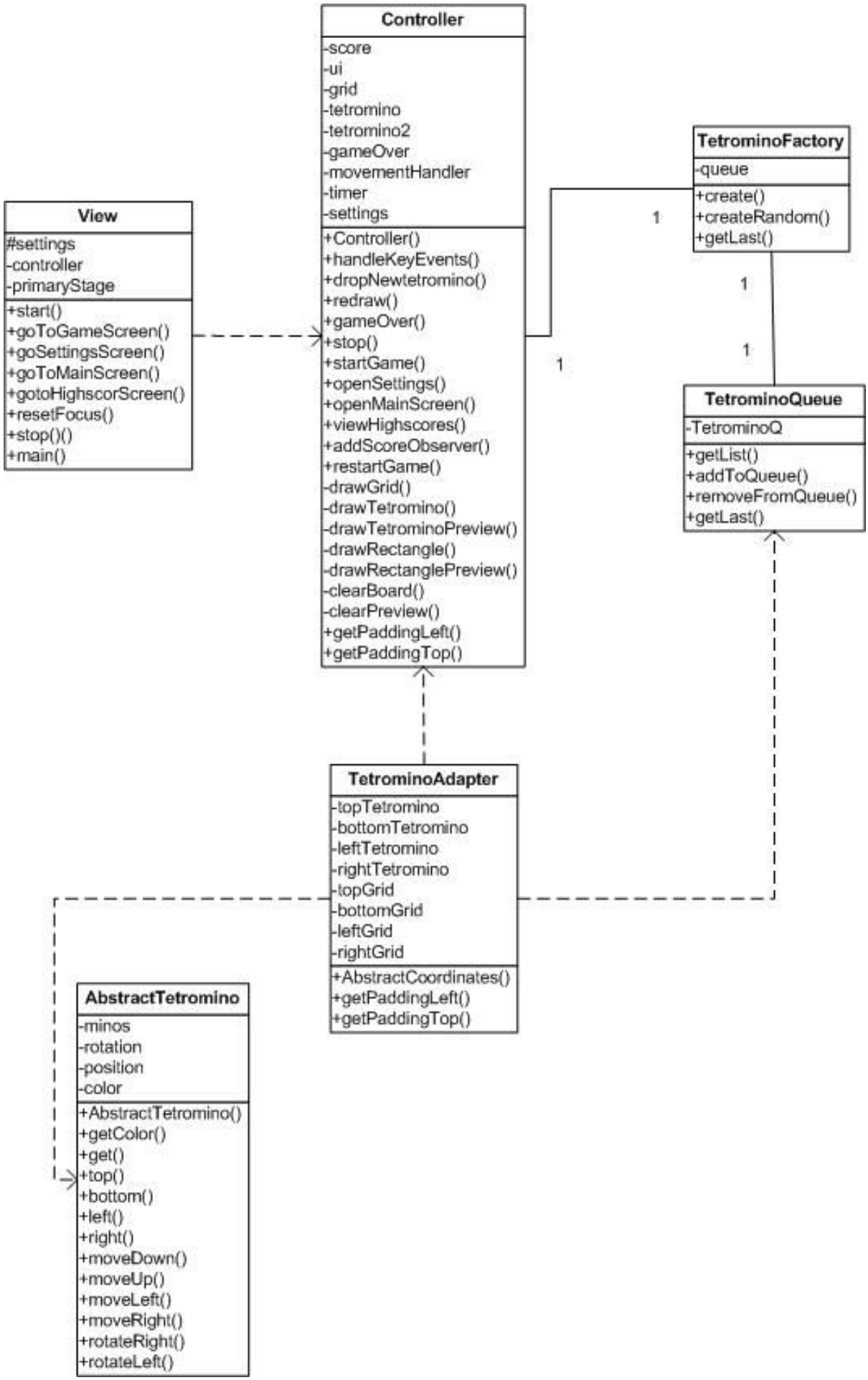
Adapter Design Pattern

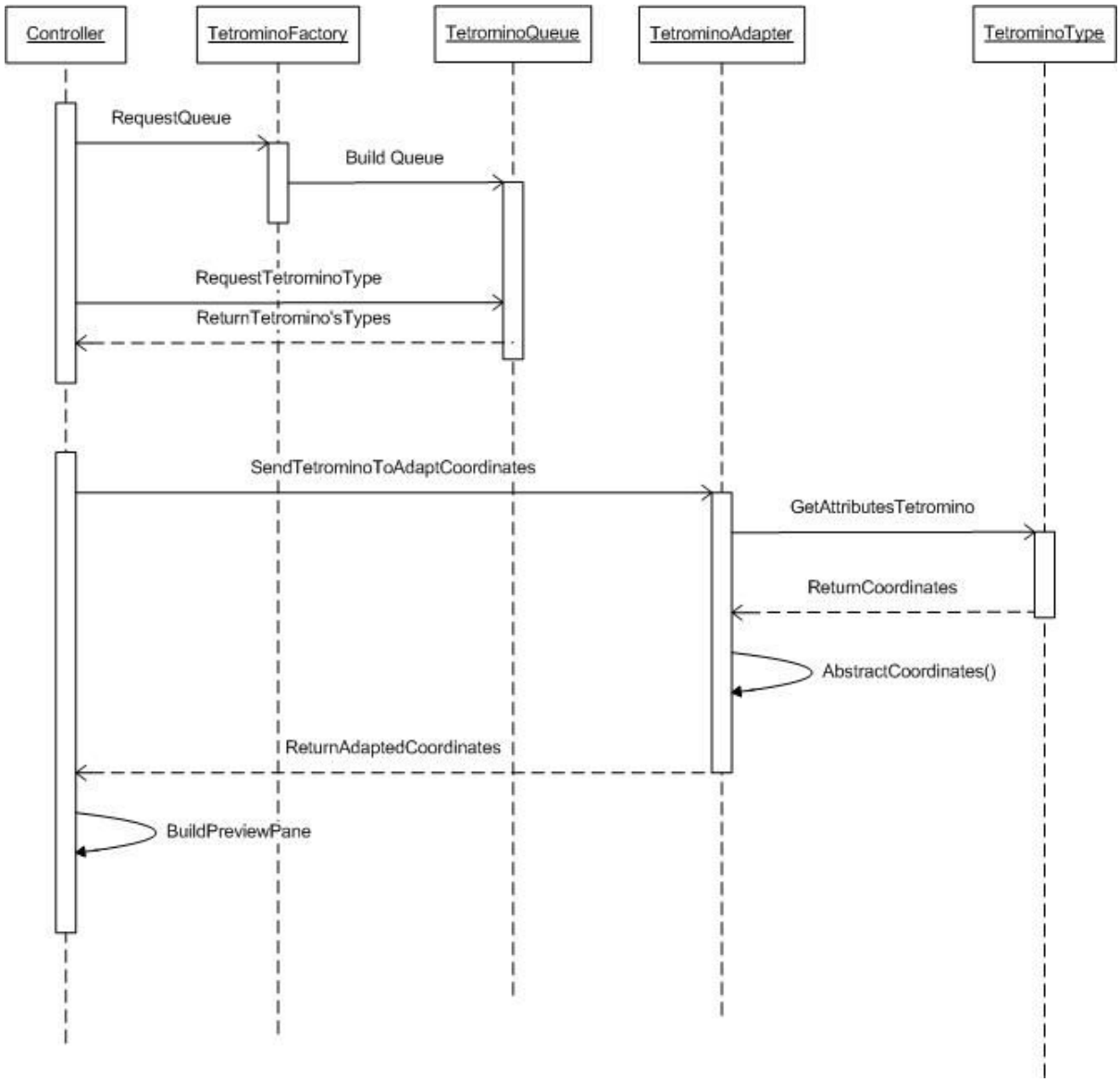
One of this sprint's objective was to implement the adapter pattern. The idea was to create an adapter that would make sure the preview tetromino would be in the center of the preview pane.

Our first idea was to create a adapter that would first get the tetromino type from the queue, the same one that the preview tetromino that was drawn would get. (This later turned out to be a very bad idea, because if an instance would get the tetromino type from the queue, it will actually be removed. Two instances getting the same tetromino type is therefor not going to work). Once it had the Tetromino type, it could then calculate the spacing that should be between the borders of the tetromino and the borders of the canvas. This would then return these values as an integer, where the controller (or view, we didn't know which one was going to work) could use it as padding to get the tetromino in the middle.

The essence was really to get the coordinates of the tetromino type and somehow convert these to an integer that we could use to create the padding. The class diagram and sequence diagram can be seen below.

(*Note the the arrow between the TetrominoAdapter and Controller should be a realization arrow, but Visio didn't have this arrow.)*


Controller
-score
-ui
-grid
-tetromino
-tetromino2
-gameOver
-movementHandler
-timer
-settings
+Controller()
+handleKeyEvents()
+dropNewtetromino()
+redraw()
+gameOver()
+stop()
+startGame()
+openSettings()
+openMainScreen()
+viewHighscores()
+addScoreObserver()
+restartGame()
-drawGrid()
-drawTetromino()
-drawTetrominoPreview()
-drawRectangle()
-drawRectanglePreview()
-clearBoard()
-clearPreview()
+getPaddingLeft()
+getPaddingTop()
TetrominoFactory
-queue
+create()
+createRandom()
+getLast()
1
1
1
1
View
#settings
-controller
-primaryStage
+start()
+goToGameScreen()
+goSettingsScreen()
+goToMainScreen()
+gotoHighscorScreen()
+resetFocus()
+stop()()
+main()
TetrominoQueue
-TetrominoQ
+getList()
+addToQueue()
+removeFromQueue()
+getLast()
TetrominoAdapter
-topTetromino
-bottomTetromino
-leftTetromino
-rightTetromino
-topGrid
-bottomGrid
-leftGrid
-rightGrid
+AbstractCoordinates()
+getPaddingLeft()
+getPaddingTop()
AbstractTetromino
-minos
-rotation
-position
-color
+AbstractTetromino()
+getColor()
+get()
+top()
+bottom()
+left()
+right()
+moveDown()
+moveUp()
+moveLeft()
+moveRight()
+rotateRight()
+rotateLeft()



This turned out to not really work for us, we couldn't find a way to really get the padding. We tried it in the canvas, it the pane itself, but it was hard getting it done. The other big problem was that the position of the where the tetromino was going to be build asked coordinates, build of the Grid class, which are essentially 20x20 pixel blocks. We needed the location of the Tetromino to be in pixels for it to work.

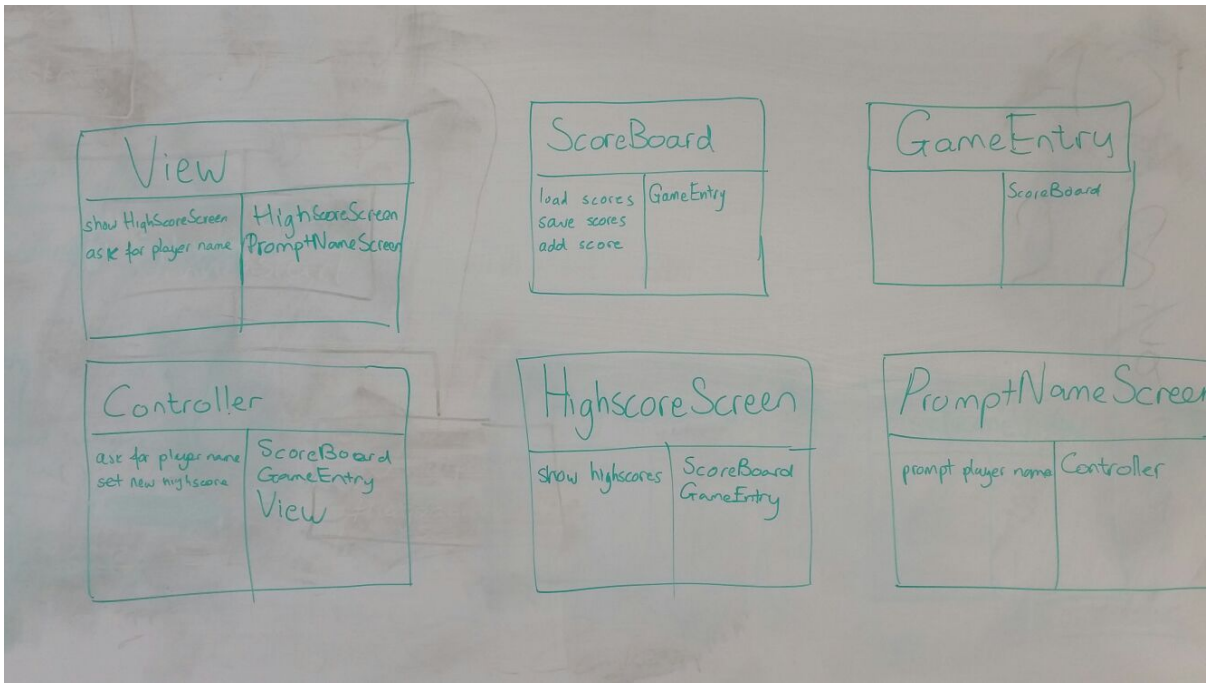
We had two further options: build an adapter, that would make it possible to give a position in pixels to the AbstractTetromino object (which asks for coordinates) or look somewhere else for a solution, instead of in the dropNewTetromino() method in the Controller, where the location is determined.

Finally we found our salvation in the drawRectanglePreview() method, where the location of where it starts to draw in the canvas is given, in pixels. What we had to do now is build an adapter that got the coordinates of the tetromino from the queue, convert these to pixels and then calculate where the method needs to start drawing and this worked. Instead of getting the tetromino from the queue, we build it first in the controller and build the adapter that way, that it could use an AbstractTetromino as input and get the coordinates. This fixed the queue problem.

Essentially our first idea was good, finding the right way to implement it in the right methods was the hard part.

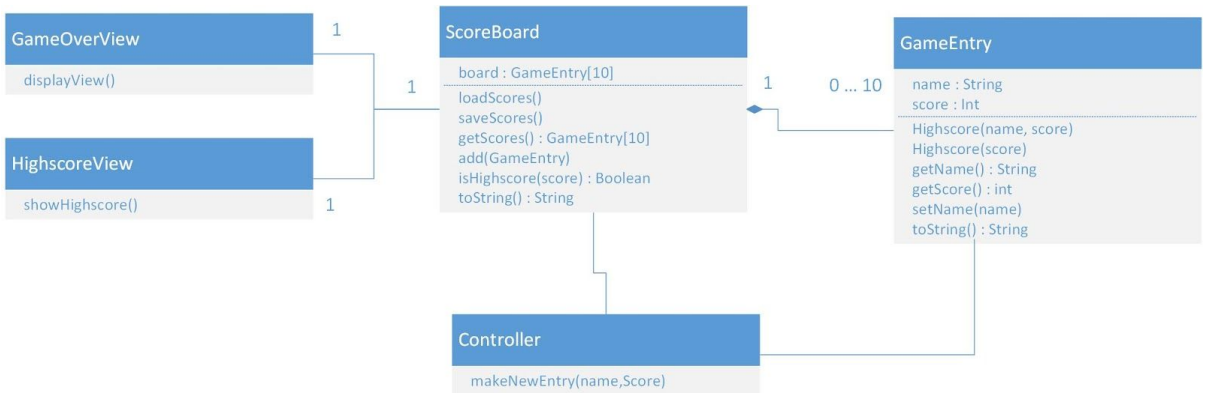
Highscore screen

In this week's assignment the TA wanted us to implement a highscore board. we've used RDD to identify all the classes needed.



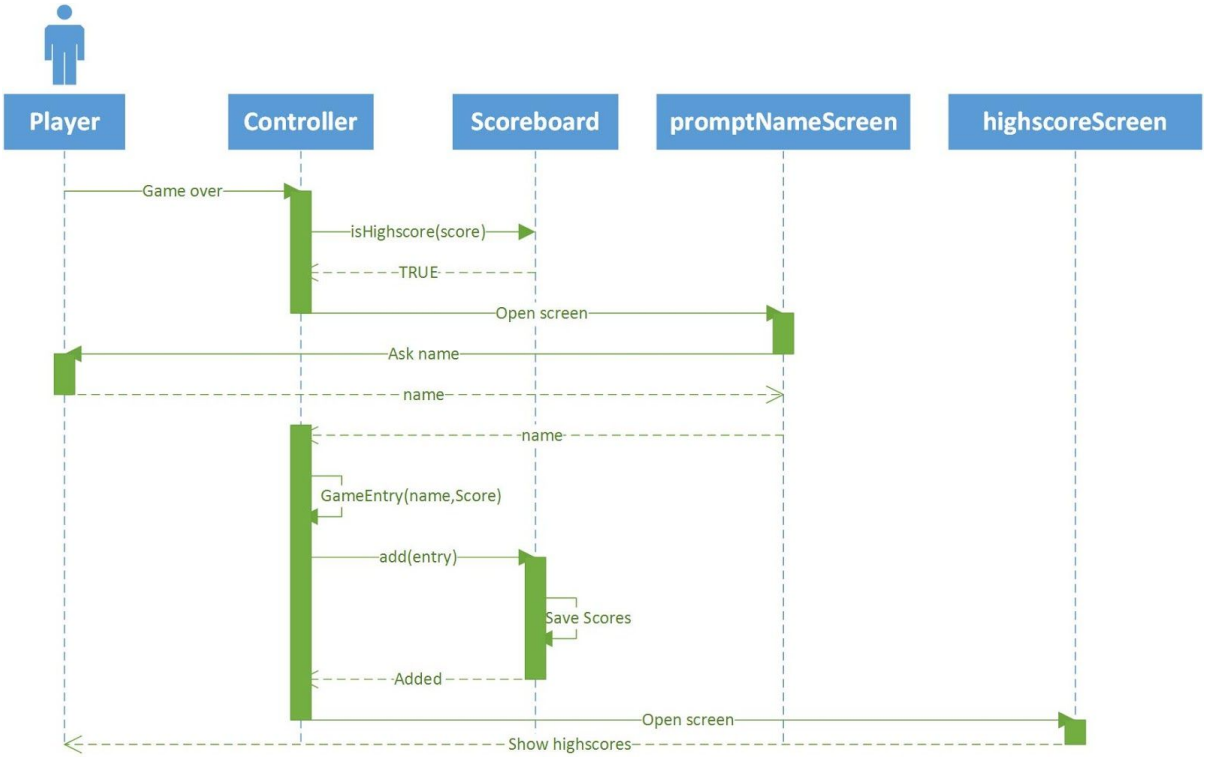
We need to have a scoreboard class that can read and write entries from a xml file. The Score board also must be able to add an entry. Therefore we need to have an entry class. If a player is game over the controller must check if the score is a high score and if so it must prompt the user for his/her name. When the player added his/her name the score will be added to the scoreboard and the high score screen will be shown.

To see how to implement this functionality a class diagram was made:



Here we can see how the ScoreBoard and GameEntry work together. While implementing the classes we came to the conclusion that we don't need the extra constructor and the

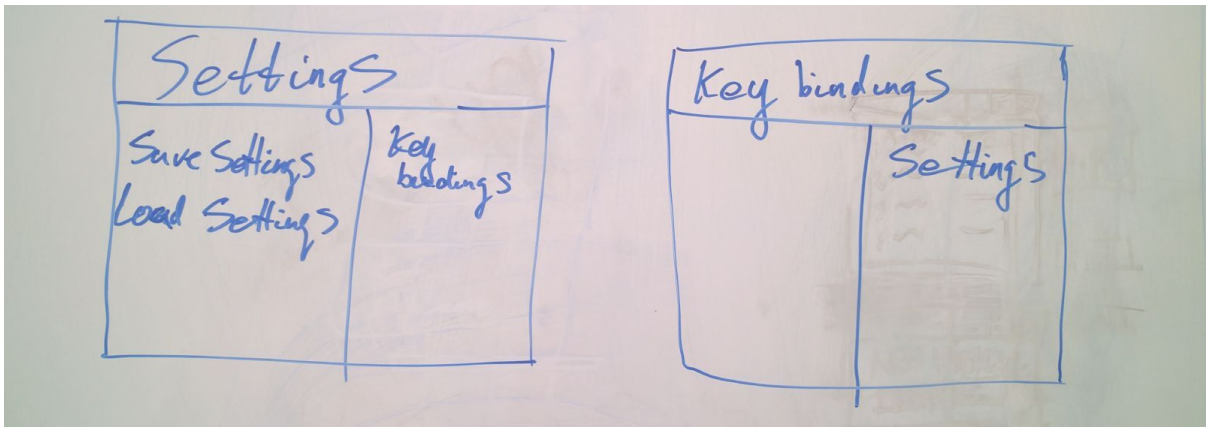
setName function because we can make the GameEntry after the user is asked for their name. To make clear how the controller must use the highscore class, we made a sequence diagram. This diagram shows how everything works together to make the highscore functionality work!



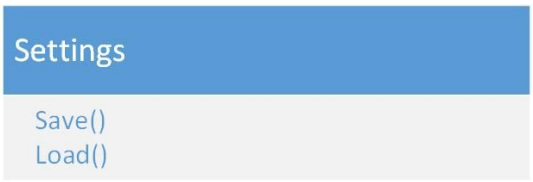
When the user is game over and doesn't have a highscore the user will be directed to the highscore screen. The scores will be loaded when the ScoreBoard is constructed (while constructing the controller).

Save settings

In a part of our free time we wanted to make it possible to save and load the user preferences automatically to/form a file. This way the colors and key bindings will be saved when changed. We user RDD to find out what to do:



In the settings we need to make a save and load functionality. and the class will work together with the key bindings class. For this to work we only need to implement two methods:



That was what we thought… While trying to implement it we had to change the key bindings. at first that class was using KeyCodes to bind the keys but we can't save keyCodes so we had to change the bindings to Strings using the toString method.

We also made a sequence diagram to see how these functions should be called: