My Solution on Assignment 1 - Problem 3

This is my Solution of Assignment 1 - Problem 3 from the Computer Science Course CS106A of Prof. Mehran Sahami at the STANFORD University.

CheckerboardKarel_v2.java

```
/*
 * File: CheckerboardKarel_v2.java
 * ---------------------------
 * When you finish writing it, the CheckerboardKarel class should draw
 * a checkerboard using beepers, as described in Assignment 1.  You
 * should make sure that your program works for all of the sample
 * worlds supplied in the starter folder.
 */

import stanford.karel.*;

public class CheckerboardKarel_v2 extends SuperKarel {


    public void run() {
        /* fills a checkerboard of ANY SIZE.
         *
         * Pre-condition:    Karel starts at 1,1 facing EAST
         */

        while( leftIsClear() ) {
            fillLine();
            goLineUp();
            ReturnToStartOfLine();
        }
        fillLine(); // dot forget to fill the last line too.
    }


    private void fillLine() {
        /* Fills a whole Line with either Pattern1 or Pattern2
         * depending on what kind of Pattern has been set on the
         * last line (1 row down)
         *
         * PRE-CONDITION:    Karel @ Field1 of a Row facing EAST
         *
         * POST-CONDITION:    Karel filled the whole line with a
Beeper-Pattern
         *                     sitting on the last Field of a Row,
facing East.
         */
```

```
        // check if last line starts with a beeper
        if( LastLineStartsWithBeeper() ) {
            fillLineWithPattern2();
        } else {
            fillLineWithPattern1();
        }
    }

    private boolean LastLineStartsWithBeeper() {
        /* Returns TRUE or FALSE if the Last Line (one Line UNDER
actual position
         * starts with a Beeper or not.
         *
         * PRE-CONDITION:    Karel @ Field1 on ANY ROW facing EAST
         *
         * POST-CONDITION:    No possition change.
         */
        if (rightIsBlocked() ) { // special Test while on FIRST LINE
            return false;
        } else {
            turnRight();
            move();
            if ( beepersPresent() ) {
                turnAround();
                move();
                turnRight();
                return true;
            } else {
                turnAround();
                move();
                turnRight();
                return false;
            }
        }
    }


    private void fillLineWithPattern1() {
        /* Fills a whole Line with Pattern1
         *
         * PRE-CONDITION:    Karel looks either EAST or WEST at the
first
         *                     Field of a Row.
         *
         * POST-CONDITION:    Karel filled the whole line with a
Beeper-Pattern 1
         *                     which means, starting with a Beeper at
Field1,
         *                     then skipping the next field, and so on.
         */
```

```
        while( frontIsClear() ) {
            putBeeper();
            move();
            // Front STILL clear?? Then do another Step.
            if( frontIsClear() ) {
                move();
            }
        }
    }

    private void fillLineWithPattern2() {
        /* Fills a whole Line with Pattern2
         *
         * PRE-CONDITION:    Karel looks either EAST or WEST at the
first
         *                     Field of a Row.
         *
         * POST-CONDITION:    Karel filled the whole line with a
Beeper-Pattern 2
         *                     which means, starting with an empty
Field1,
         *                     then placing a beeper on the next field,
and so on.
         */
        while( frontIsClear() ) {
            move();
            putBeeper();
            // Front STILL clear?? Then do another Step.
            if( frontIsClear() ) {
                move();
            }
        }
    }


    private void ReturnToStartOfLine() {
        /* Returns Karel to FIRST FIELD of a Row, facing EAST
         *
         * PRE-CONDITION:    Karel facing EAST on any Row on ANY FIELD.
         *
         * POST-CONDITION:    Karel facing East at FIRST FIELD of the
Row.
         */
        turnAround();
        while( frontIsClear() ) {
            move();
        }
        turnAround();
    }
```

```
    private void goLineUp() {
        /* Karel climbs one rown UP
         *
         * PRE-CONDITION:    Karel facing EAST
         *
         * POST-CONDITION:    Karel facing East, climbed one Row up.
         */
        turnLeft();
        move();
        turnRight();
    }
}
```

— *Axel Werner* *2011-02-05 17:40*

java, karel, stanford, university, cs106, computer, science, learning, programming