View Javadoc

```java
package model.element;

import java.util.Observable;

import contract.IElement;
import contract.ILevelMap;
import entity.Level;
import model.element.mobile.*;
import model.element.motionless.*;

/**
 * The Class Model.
 *
 * @author Group 5
 */
public class LevelMap extends Observable implements ILevelMap {

    /** The levelmap; */
    private IElement[][] levelmap = new IElement[40][22];

    /**
     * Instantiates a new LevelMap.
     *
     * @param level
     *                       The level.
     */
    public LevelMap(Level level) {

            for(int x=0; x<40; x++) {
                    for(int y=0; y<22; y++) {

                            String E = level.getElement(x, y);
                            switch(E) {
                            case "B":
                                    this.setElement(x, y, new UnbreakableBlock(x, y, this));
                                    break;
                            case "C":
                                    this.setElement(x, y, new Block(x, y, this));
                                    break;
                            case "R":
                                    this.setElement(x, y, new Rock(x, y, this));
                                    break;
                            case "E":
                                    this.setElement(x, y, new Enemy(x, y, this));
                                    break;
                            case "S":
                                    this.setElement(x, y, new Exit(x, y, this));
                                    break;
                            case "P":
                                    this.setElement(x, y, new Player(x, y, this));
                                    break;
                            case "D":
                                    this.setElement(x, y, new Diamond(x, y, this));
                                    break;
                            default :
                                    this.setElement(x, y, null);

                            }

                    }

            }

    }

    /*
```

```
	 * (non-Javadoc)
	 * 
	 * @see contract.ILevelMap#getPlayer()
	 */
	@Override
	public IElement getPlayer() {
		
		for(int x=0; x<40; x++) {
			for(int y=0; y<22; y++) {
				
				IElement element = this.getElement(x, y);
				
				if(element instanceof Player) {
					return element;
				}
			}
		}
		
		return null;
	}

	/*
	 * (non-Javadoc)
	 * 
	 * @see contract.ILevelMap#getElement()
	 */
	@Override
	public IElement getElement(int x, int y) {
		
		return this.levelmap[x][y];
	}

	/*
	 * (non-Javadoc)
	 * 
	 * @see contract.ILevelMap#setElement()
	 */
	@Override
	public void setElement(int x, int y, IElement element) {
		this.levelmap[x][y] = element;
		this.setChanged();
		this.notifyObservers();
	}

	/*
	 * (non-Javadoc)
	 * 
	 * @see contract.ILevelMap#removeElement()
	 */
	@Override
	public void removeElement(int x, int y) {
		
		this.setElement(x, y, null);
	}

	/*
	 * (non-Javadoc)
	 * 
	 * @see contract.ILevelMap#getObservable()
	 */
	public Observable getObservable() {
		return this;
	}

	/*
	 * (non-Javadoc)
	 * 
	 * @see contract.ILevelMap#popDiamond()
	 */
```

```
    @Override
    public void popDiamond(int a, int b) {

            for(int x=a-1; x<a+2; x++) {
                    for(int y =b-1; y<b+2; y++) {
                            this.setElement(x, y, new Diamond(x, y, this));
                    }
            }
    }

}
```