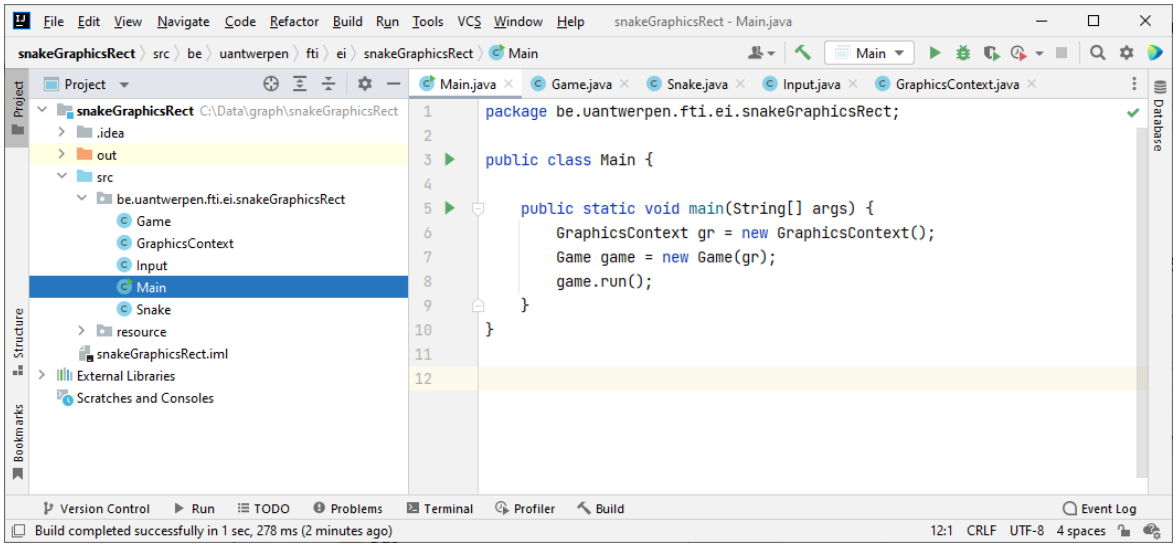
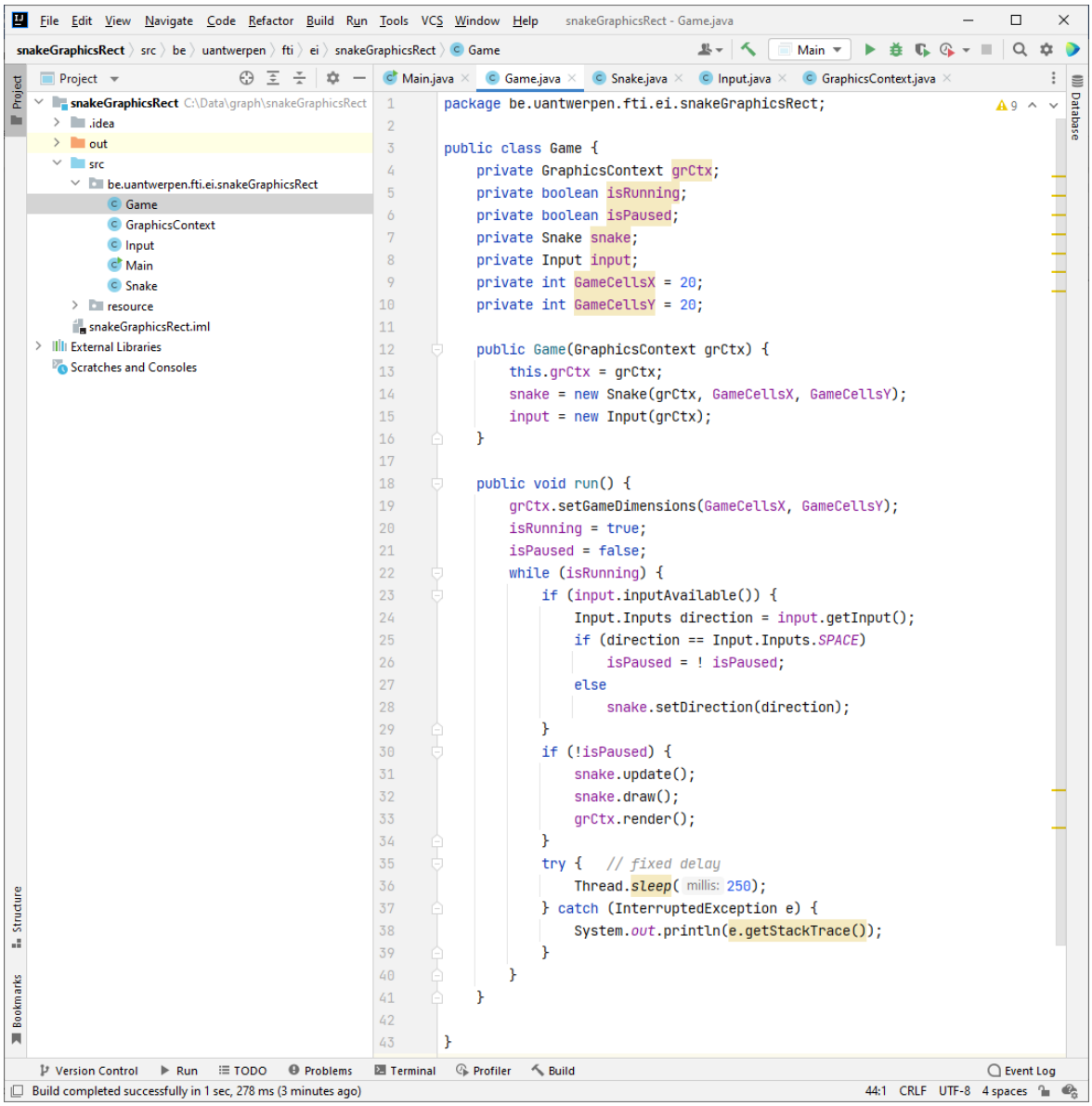
# Oefeningen Java 2d

## Voorbeeld met Java 2d graphics:

(De code kan je terugvinden op Blackboard onder Java2D-snakeRectExample.zip en Java2D-snakeImageExample.zip)



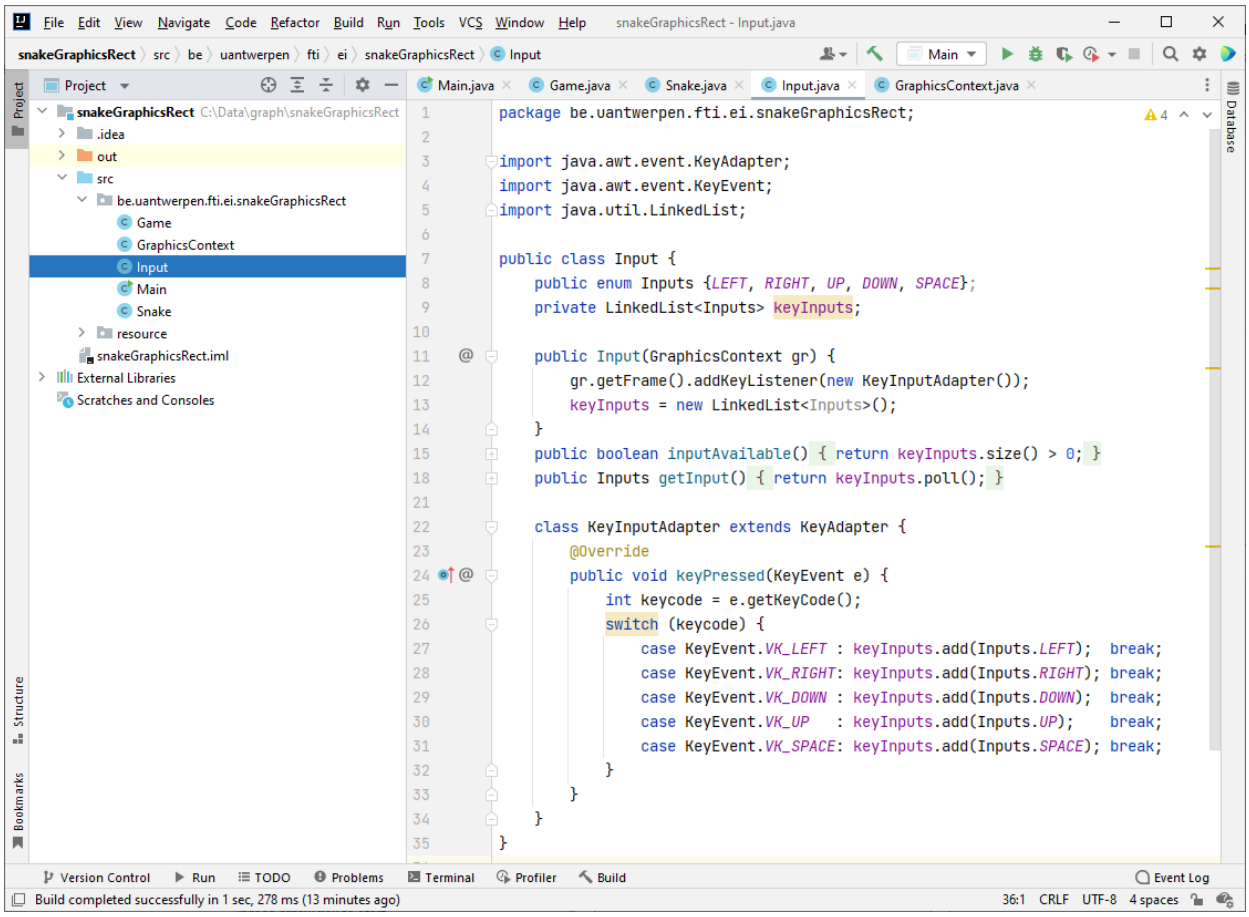



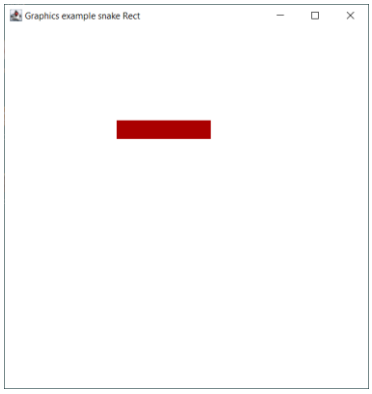
Main.java | Game.java | Snake.java | Input.java | GraphicsContext.java

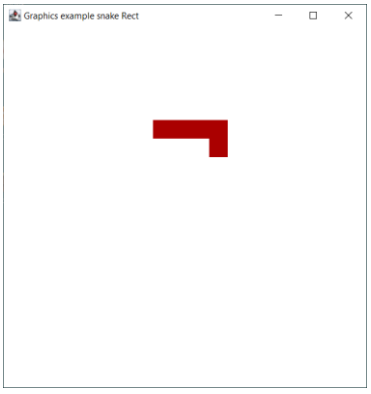
```java
package be.uantwerpen.fti.ei.snakeGraphicsRect;

import javax.swing.*;
import java.awt.*;
import java.awt.image.BufferedImage;

public class GraphicsContext {
    private int ScreenWidth;
    private int ScreenHeight;
    private JFrame frame;
    private JPanel panel;
    private BufferedImage g2dimage;     // used for drawing
    private Graphics2D g2d;             // always draw in this one
    private int size;                   // cel size

    public Graphics2D getG2d() { return g2d; }
    public JFrame getFrame() { return frame; }
    public int getSize() { return size; }

    public GraphicsContext() {
        ScreenWidth = 500;
        ScreenHeight = 520;
        frame = new JFrame();
        panel = new JPanel( isDoubleBuffered: true) {
            @Override
            public void paintComponent(Graphics g) {
                super.paintComponent(g);
                doDrawing(g);
            }
        };
        frame.setFocusable(true);
        frame.add(panel);
        frame.setTitle("Graphics example snake Rect");
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        frame.setSize(ScreenWidth, ScreenHeight);
        frame.setResizable(true);
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
    }

    public void render() { panel.repaint(); }

    private void doDrawing(Graphics g) {
        Graphics2D graph2d = (Graphics2D) g;
        Toolkit.getDefaultToolkit().sync();
        graph2d.drawImage(g2dimage, x: 0, y: 0, observer: null);   // copy buffered image
        graph2d.dispose();
        if (g2d != null)
            g2d.clearRect( x: 0, y: 0, frame.getWidth(), frame.getHeight());
    }

    public void setGameDimensions(int GameCellsX, int GameCellsY) {
        size = Math.min(ScreenWidth/GameCellsX, ScreenHeight/GameCellsY);
        System.out.println("size: "+size);
        frame.setLocation( x: 50, y: 50);
        frame.setSize(ScreenWidth, ScreenHeight);
        g2dimage = new BufferedImage(frame.getWidth(), frame.getHeight(), BufferedImage.TYPE_4BYTE_ABGR_PRE);
        g2d = g2dimage.createGraphics();
        g2d.setBackground(new Color( r: 255, g: 255, b: 255));
        g2d.clearRect( x: 0, y: 0, frame.getWidth(), frame.getHeight());
    }
}
```

Main.java | Game.java | Snake.java | Input.java | GraphicsContext.java

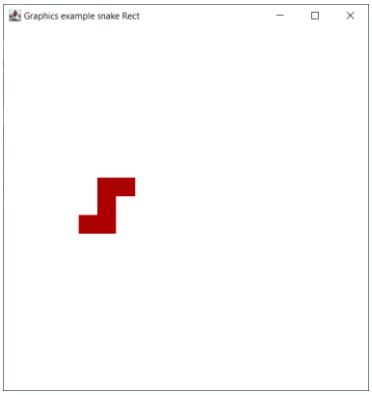
```java
package be.uantwerpen.fti.ei.snakeGraphicsRect;
import java.awt.*;
import java.util.ArrayList;

public class Snake {
    private GraphicsContext grCtx;
    private int dx = 1;
    private int dy = 0;
    private int GameCellsX;
    private int GameCellsY;
    private ArrayList<Point> snake = new ArrayList<>();
    private int curX = 7;
    private int curY = 7;

    public Snake(GraphicsContext grCtx, int gameCellsX, int gameCellsY) {
        this.grCtx = grCtx;
        this.GameCellsX = gameCellsX;
        this.GameCellsY = gameCellsY;
        for (int i=0; i<5; i++) // make snake
            snake.add(new Point( x: i+7,  y: 7));
    }

    public void setDirection(Input.Inputs direction) {
        switch (direction) {
            case LEFT  -> { dx = -1; dy =  0; }
            case RIGHT -> { dx =  1; dy =  0; }
            case DOWN  -> { dx =  0; dy =  1; }
            case UP    -> { dx =  0; dy = -1; }
        }
    }

    public void update() {
        curX = (curX + dx + GameCellsX) % GameCellsX;
        curY = (curY + dy + GameCellsY) % GameCellsY;
        snake.remove( index: 0);            // remove tail
        snake.add(new Point(curX,curY));  // add new head
    }

    public void draw() {
        Graphics2D g2d = grCtx.getG2d();
        int size = grCtx.getSize();
        g2d.setColor(new Color( r: 170, g: 0, b: 0));
        for (int i=0; i<5; i++)
            g2d.fillRect( x: snake.get(i).x*size,  y: snake.get(i).y*size, size, size);
    }
}
```


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
snakeGraphicsRect - Input.java
snakeGraphicsRect > src > be > uantwerpen > fti > ei > snakeGraphicsRect > Input
Main.java
Game.java
Snake.java
Input.java
GraphicsContext.java
snakeGraphicsRect C:\Data\graph\snakeGraphicsRect
.idea
out
src
be.uantwerpen.fti.ei.snakeGraphicsRect
Game
GraphicsContext
Input
Main
Snake
resource
snakeGraphicsRect.iml
External Libraries
Scratches and Consoles
package be.uantwerpen.fti.ei.snakeGraphicsRect;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.util.LinkedList;
public class Input {
public enum Inputs {LEFT, RIGHT, UP, DOWN, SPACE};
private LinkedList<Inputs> keyInputs;
public Input(GraphicsContext gr) {
gr.getFrame().addKeyListener(new KeyInputAdapter());
keyInputs = new LinkedList<Inputs>();
}
public boolean inputAvailable() { return keyInputs.size() > 0; }
public Inputs getInput() { return keyInputs.poll(); }
class KeyInputAdapter extends KeyAdapter {
@Override
public void keyPressed(KeyEvent e) {
int keycode = e.getKeyCode();
switch (keycode) {
case KeyEvent.VK_LEFT : keyInputs.add(Inputs.LEFT); break;
case KeyEvent.VK_RIGHT: keyInputs.add(Inputs.RIGHT); break;
case KeyEvent.VK_DOWN : keyInputs.add(Inputs.DOWN); break;
case KeyEvent.VK_UP : keyInputs.add(Inputs.UP); break;
case KeyEvent.VK_SPACE: keyInputs.add(Inputs.SPACE); break;
}
}
}
}
Version Control
Run
TODO
Problems
Terminal
Profiler
Build
Event Log
Build completed successfully in 1 sec, 278 ms (13 minutes ago)
36:1 CRLF UTF-8 4 spaces


Graphics example snake Rect

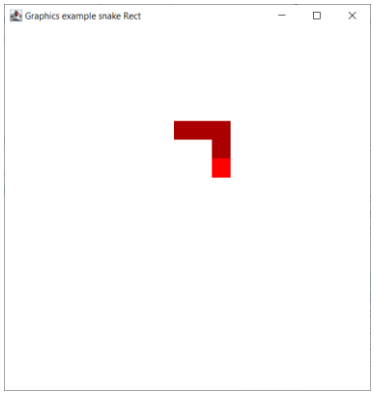

Graphics example snake Rect


Graphics example snake Rect

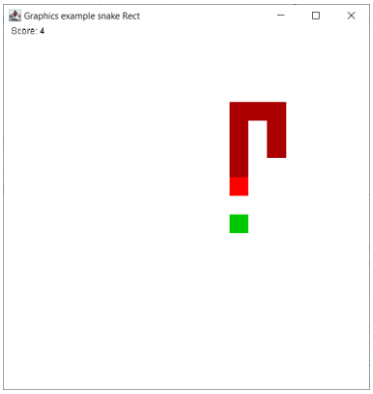

Graphics example snake Rect

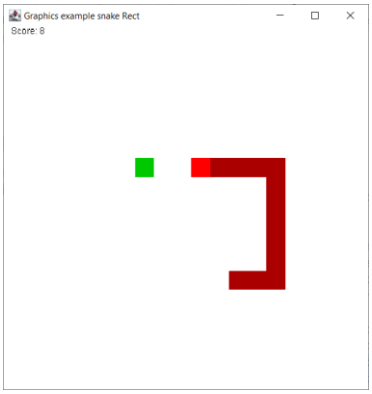
1. Bestudeer de code en zoek de methoden van Java 2d die je nog niet begrijpt op in de Java documentatie. Experimenteer door enkele parameters aan te passen.

2. Teken het hoofd van “snake” in een andere kleur dan de rest.




3. Vervolledig het spel. Laat een voedselpakketje verschijnen op een willekeurige plaats. Als “snake” het voedsel pakt, verhoogt de score en verschijnt er een nieuw voedselpakket op een volgende willekeurige plaats. Nadat het laatste stukje van de staart van “snake” voorbij de vorige voedselplaats komt wordt er aan “snake” een stukje toegevoegd (niet weggelaten…) en de score aangepast.




4. Voeg nu collision detection toe om te controleren of “snake” zichzelf ergens kruist (i.e. raakt het hoofd ergens één van de andere elementen van “snake”). Als dit het geval is, moet de score terug op 0 geplaatst worden maar blijft het spel verder gaan.

5. Optioneel: Bedenk zelf nog enkele originele uitbreidingen en werk ze uit.