TDA 545: Objektorienterad programmering

Föreläsning 14:
Grafik & mera händelsehantering

Magnus Myréen

Chalmers, läsperiod 1, 2015-2016

Idag

Idag: grafik — läs kap 17

Viktigt i denna föreläsning:

- att rita i Java
- inner klasser
- händelsehantering (timer, mus, tangentbord)

Nästa gång: exceptions, kap 15 fram t.o.m. sida 655

…men först: tentan närmar sig!

Tentan: granskar om ni kan programmera i Java

Fråga: hur lär man sig programmera?

… genom att läsa kod? Nej.

… genom att läsa en bok? Nej.

… genom att lyssna på föreläsningar? Nej.

Svar: genom att programmera!

Tips: Planera din kod *innan du börjar skriva*.

Kompilera ofta.

Testa halvfärdig kod.

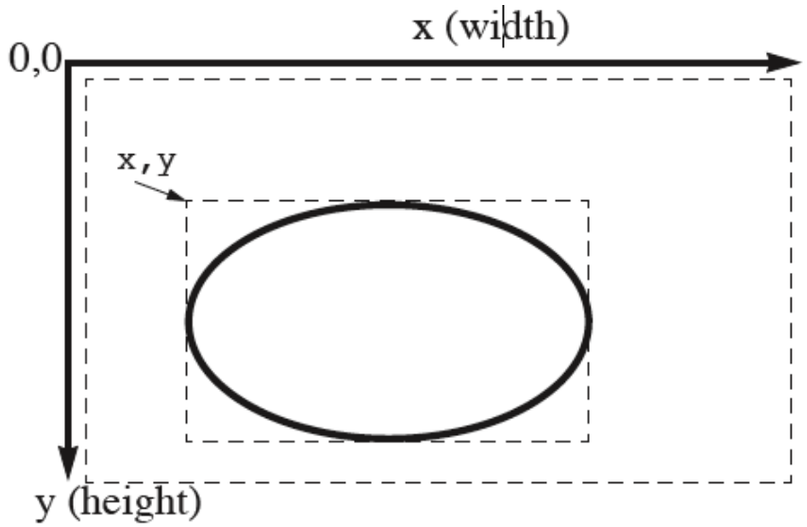
Dela programmeringsproblemet i mindre bitar.

Att rita i Java

Man ritar i tex en JPanel. För att rita definierar man om callback metoden "paintComponent(Graphics g)" Den anropas sedan av systemet. Vill man tvinga fram en omritning så anropar man repaint() som sedan anropar paintComponent.

Det finns många "rit" primitiver i **Graphics** som drawline, drawrect, fillrect, drawString drawPolygon, drawPolyline, ...
och ännu fler i **Graphics2D**

Jag kommer att använda Graphics här.
Kordinatsystem:



Exempel: att rita i Java

```
import java.awt.*;
import javax.swing.*;

class DemoPanel extends JPanel {

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawRect(10, 10, 10, 10);
        g.drawRect(30, 10, 20, 10);
        int x = this.getWidth();
        int y = this.getHeight();
        g.drawRect(x/5, y/5, x-2*x/5, y/3);
    }

}

public class Demo1 extends JFrame {

    public Demo1() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(200,150);
        setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel);
        setVisible(true);
    }

    public static void main(String[] args) {
        Demo1 f = new Demo1();
    }

}
```

Ärver JPanel som vi ska rita i.

Överskuggar paintComponent metoden.

I Java ritar man med Graphics objekt.

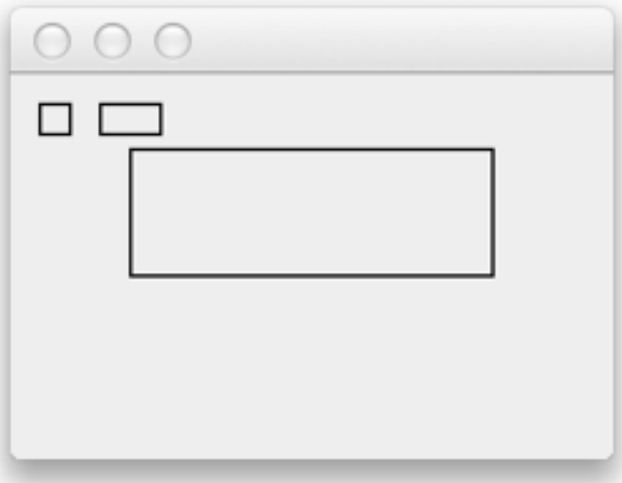
Vi ritar tre rektanglar.

Den sista anpassar sig till storleken av JPanelen, dvs använder x och y.

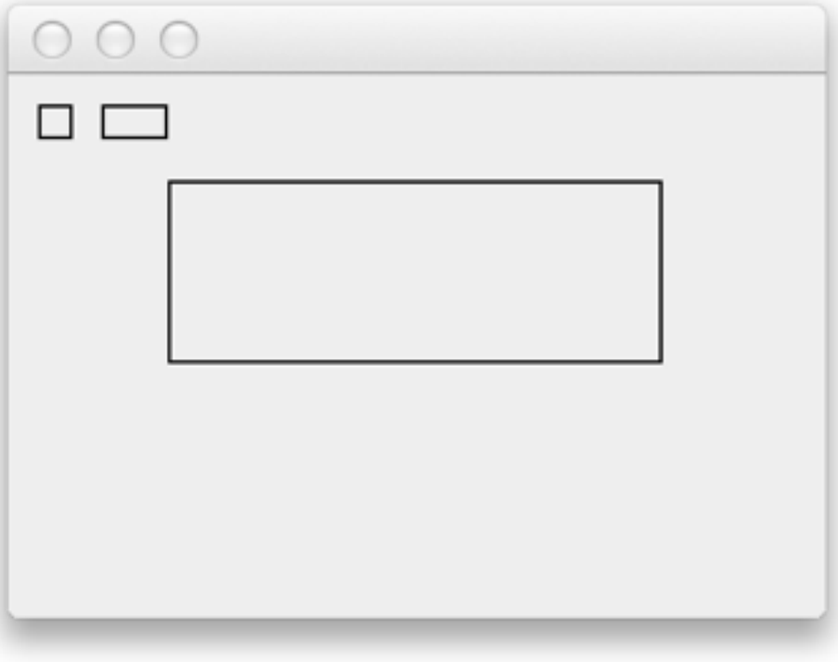
I huvudklassen skapar och använder vi en DemoPanel precis som en vanlig JPanel.

Exempel: att rita i Java

```
import java.awt.*;
import javax.swing.*;

class DemoPanel extends JPanel {

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawRect(10, 10, 10, 10);
        g.drawRect(30, 10, 20, 10);
        int x = this.getWidth();
        int y = this.getHeight();
        g.drawRect(x/5, y/5, x-2*x/5, y/3);
    }

}

public class Demo1 extends JFrame {

    public Demo1() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(200,150);
        setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel);
        setVisible(true);
    }

    public static void main(String[] args) {
        Demo1 f = new Demo1();
    }

}
```



Om ändrar på fönstrets storlek ändras också den sista rektangeln.



Exempel: att rita i Java

Aningen trist att skriva DemoPanel som separat klass. Den *hör ju till* klassen Demo1.

Lösning: skriv DemoPanel inuti den andra klassen! (Nästa sida…)

```
import java.awt.*;
import javax.swing.*;

class DemoPanel extends JPanel {

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawRect(10, 10, 10, 10);
        g.drawRect(30, 10, 20, 10);
        int x = this.getWidth();
        int y = this.getHeight();
        g.drawRect(x/5, y/5, x-2*x/5, y/3);
    }

}

public class Demo1 extends JFrame {

    public Demo1() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(200,150);
        setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel);
        setVisible(true);
    }

    public static void main(String[] args) {
        Demo1 f = new Demo1();
    }

}
```

Inner klass!

```
import java.awt.*;
import javax.swing.*;

public class Demo2 extends JFrame {

    private class DemoPanel extends JPanel {

        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            g.drawRect(10, 10, 10, 10);
            g.drawRect(30, 10, 20, 10);
            int x = this.getWidth();
            int y = this.getHeight();
            g.drawRect(x/5, y/5, x-2*x/5, y/3);
        }

    }

    public Demo2() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(200,150);
        setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel);
        setVisible(true);
    }

    public static void main(String[] args) {
        Demo2 f = new Demo2();
    }

}
```

Man kan skriva klasser inuti andra klasser.

Vanligtvis är dessa **`private`**.

Inner klasser har diverse ***fördelar:***
den inre klassen har tillgång till instans-variablerna i den yttre klassen.
(Exempel: filerna för labb 3)

Att rita (forts.)

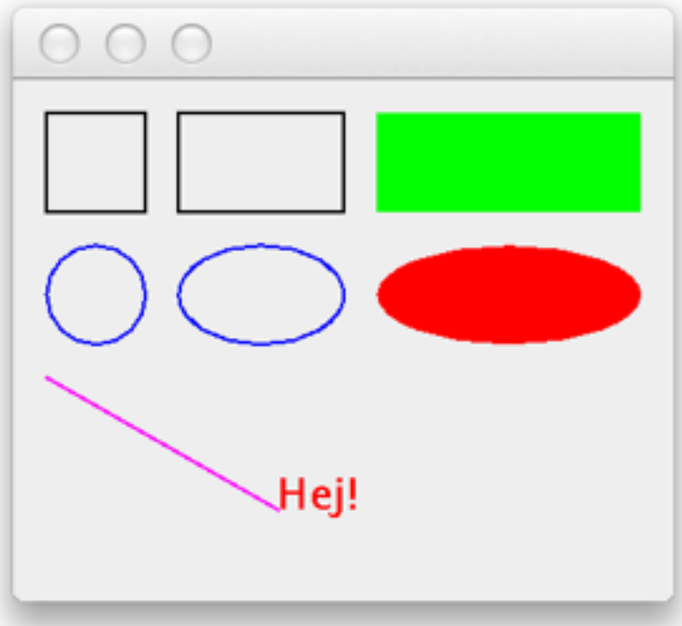
```
import java.awt.*;
import javax.swing.*;

public class Demo3 extends JFrame {
    private class DemoPanel extends JPanel {
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            g.drawRect(10, 10, 30, 30);
            g.drawRect(50, 10, 50, 30);
            g.setColor(Color.GREEN);
            g.fillRect(110, 10, 80, 30);
            g.setColor(Color.BLUE);
            g.drawOval(10, 50, 30, 30);
            g.drawOval(50, 50, 50, 30);
            g.setColor(Color.RED);
            g.fillOval(110, 50, 80, 30);
            g.setColor(Color.MAGENTA);
            g.drawLine(10,90,80,130);
            g.setColor(Color.RED);
            g.drawString("Hej!",80,130);
        }
    }
    public Demo3() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(200,180);
        setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel);
        setVisible(true);
    }
    public static void main(String[] args) {
        Demo3 f = new Demo3();
    }
}
```

drawRect(x,y,width,height)

30 pixels bred, 30 pixels hög

50 pixels bred, 30 pixels hög

Ovaler (och cirklar) på samma sätt!



Man kan skriva text med drawString.
Använder nuvarande font.
Ändra på fonten med `g.setFont(…)`

Övning på ritning

”Riktiga” slumptal är jämnt fördelade och oberoende av varandra. Ett sätt att testa om Javas slumptal är oberoende är att rita många punkter med slumpmässiga koordinater i ett fönster. Om man då kan se ”mönster” i bilden är slumptalen inte oberoende.

(Körningen till höger avslöjar inga mönster, så Javas slumptal är nog bra.)

Skriv ett program som ritar 50 000 slumpmässiga punkter i en JPanel som du lägger i en JFrame med storleken 400x400 pixlar. Man ritar lämpligen en punkt genom att rita en liten cirkel, 2x2 pixel.

(8p)

Slumptal med **Random**

I java.util finns klassen Random för att skapa slumtalsgeneratorer.

Klassen har två konstruktorer:

Random() // sätter slumptalsfröt till systemklockan

Random(long seed) // sätter slumtalsfröt till seed

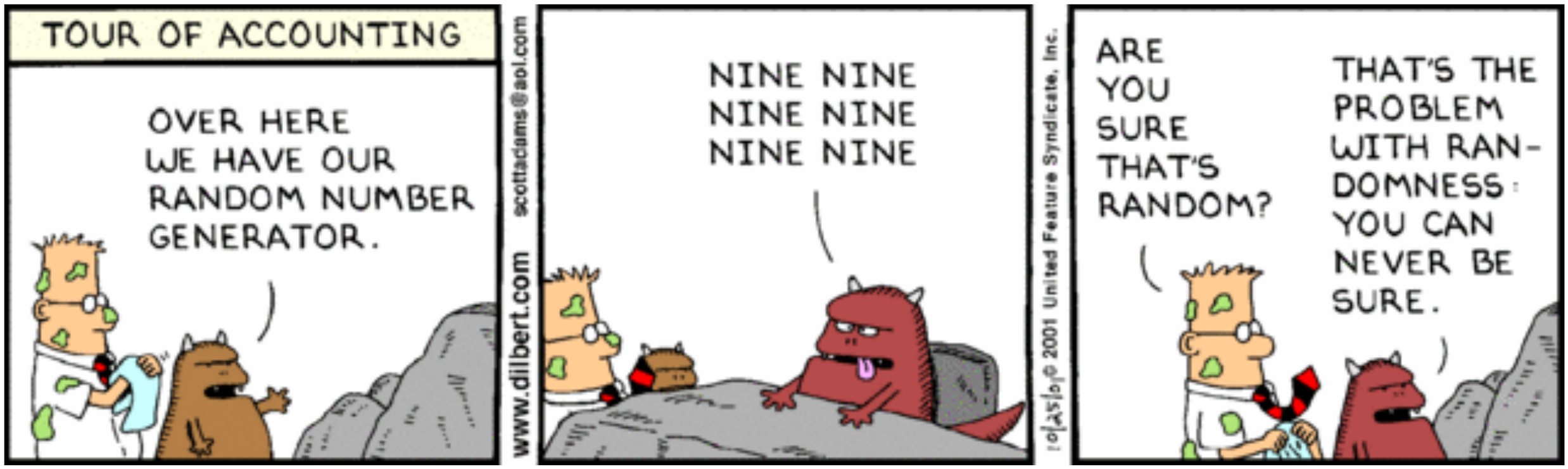
Klassen Random innehåller bl.a en instansmetod `int nextInt(int n)` som ger ett slumtal med likformig fördelning i intervallet [0, n[.

Här finns också

nextBoolean, nextDouble, nextLong m.fl.

Nedanstående program skapar en slumptalsgenerator och använder den för att skriva ut ett slumptal i intervallet [1, 4].

```
import java.util.*;
public class Slump {
    public static void main (String[] args) {
        Random generator = new Random();
        System.out.println(generator.nextInt(4)+1);
    }
}
```



http://dilbert.com/strips/comic/2001-10-25/

Övning på ritning (igen)

”Riktiga” slumptal är jämnt fördelade och oberoende av varandra. Ett sätt att testa om Javas slumptal är oberoende är att rita många punkter med slumpmässiga koordinater i ett fönster. Om man då kan se ”mönster” i bilden är slumptalen inte oberoende.

(Körningen till höger avslöjar inga mönster, så Javas slumptal är nog bra.)

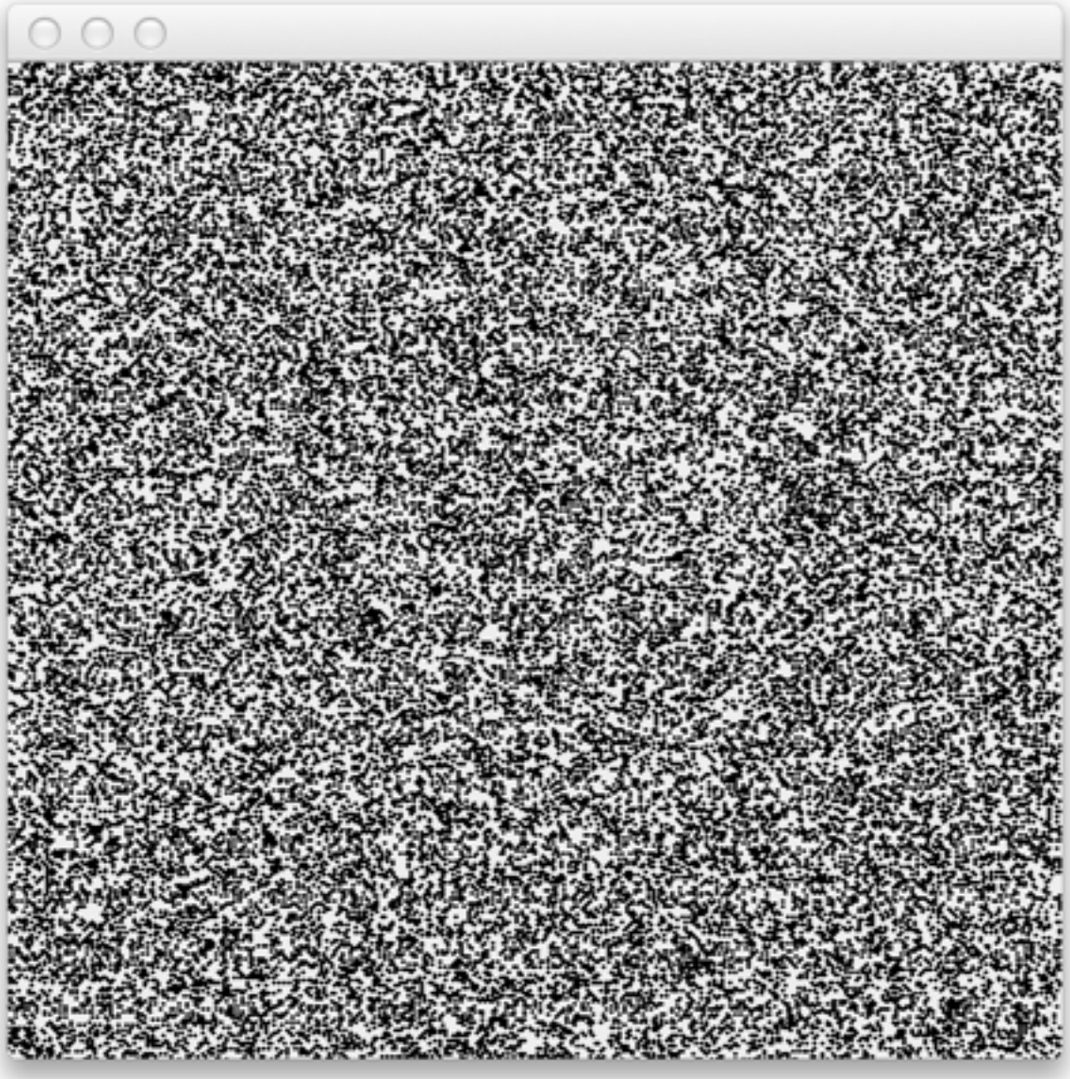
Hur ska vi implementer detta?

Skriv ett program som ritar 50 000 slumpmässiga punkter i en JPanel som du lägger i en JFrame med storleken 400x400 pixlar. Man ritar lämpligen en punkt genom att rita en liten cirkel, 2x2 pixel.

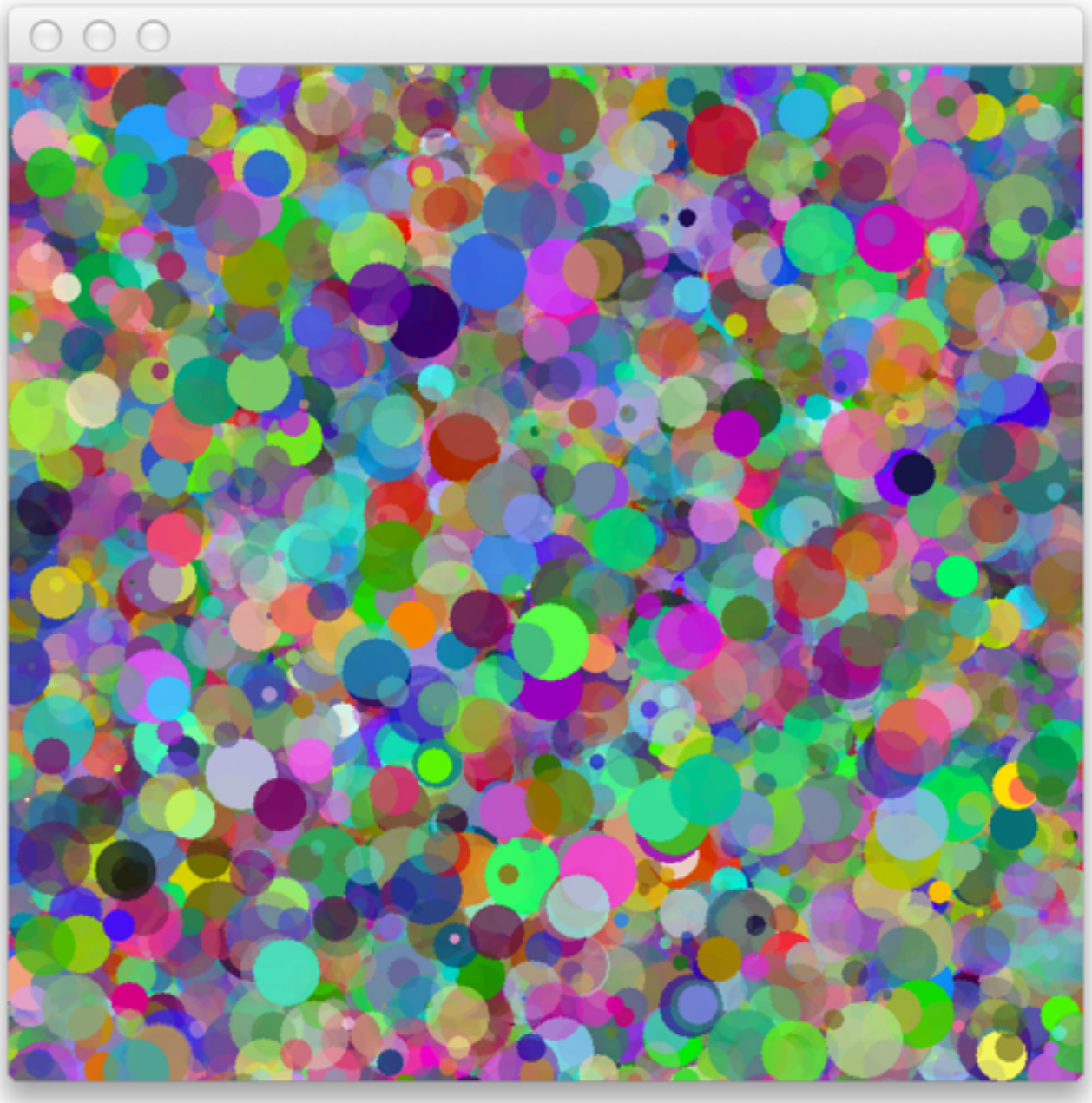
(8p)

Kod som ritar slumptal

```
import java.awt.*;
import javax.swing.*;
import java.util.*;

public class RandDots extends JFrame {
    private class DemoPanel extends JPanel {
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            Random generator = new Random();
            for (int i=0; i<50000; i++) {
                int x = generator.nextInt(this.getWidth()+1);
                int y = generator.nextInt(this.getHeight()+1);
                g.fillOval(x-1, y-1, 2, 2);
            }
        }
    }
    public RandDots() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400,400);
        setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel);
        setVisible(true);
    }
    public static void main(String[] args) {
        RandDots f = new RandDots();
    }
}
```



Uppgift: gör bilden färggrann!



Tips: ... `g.setColor(new Color(...));` ...

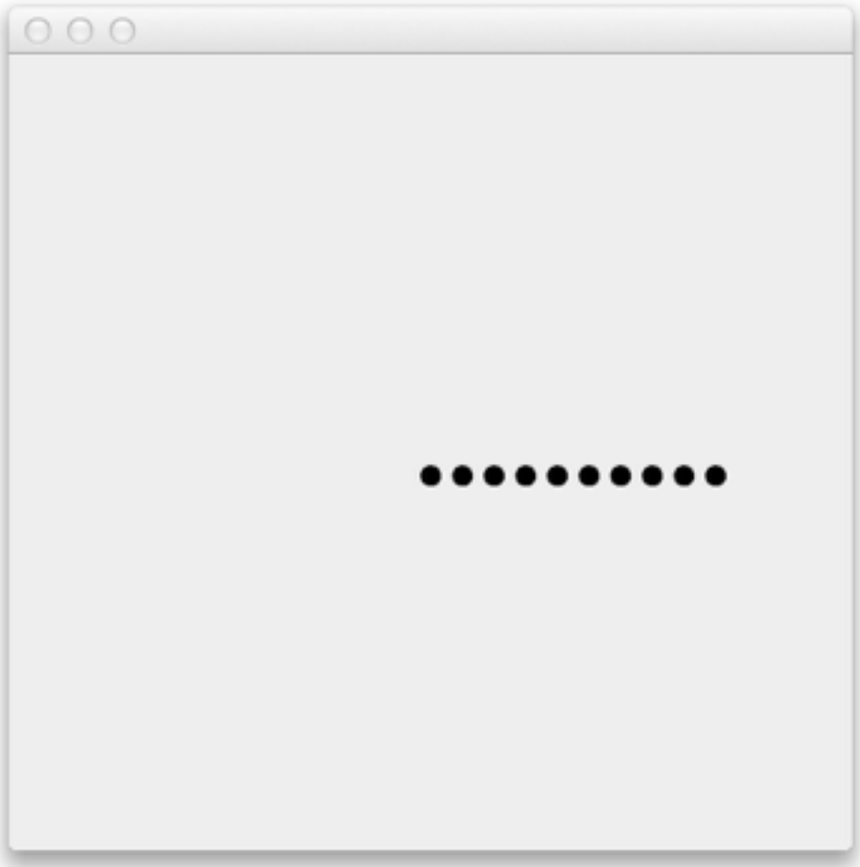
Ritning är roligast
när det
kombineras med händelser (timer, mus, tangentbord)

Man kan skriva spel mm.

Först utan händelser

```
import java.awt.*;
import javax.swing.*;
import java.util.*;

public class FunDots extends JFrame {
    private class DemoPanel extends JPanel {
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            for (int i=0; i<10; i++) {
                int x = 15 * i;
                int y = 0;
                g.fillOval(195+x, 195+y, 10, 10);
            }
        }
    }
    public FunDots() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400,400); setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel); setVisible(true);
    }
    public static void main(String[] args) {
        FunDots f = new FunDots();
    }
}
```

Ritar 10 bollar från mitten till höger.



195 = 400/2 - 10/2

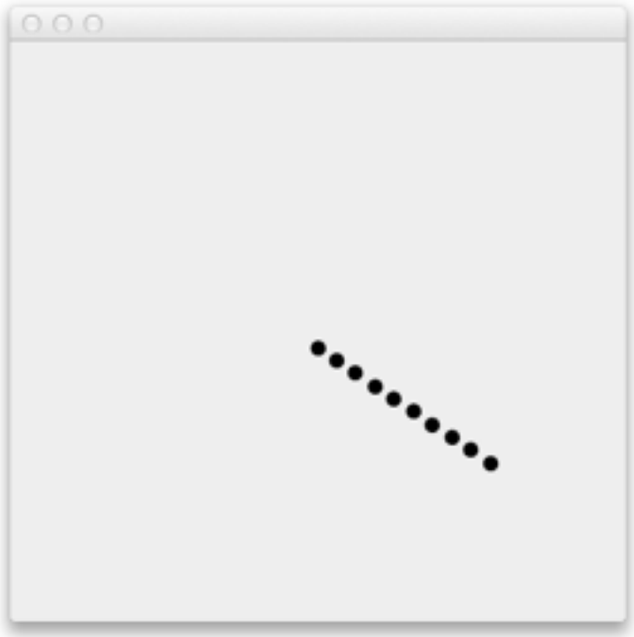
400 bred, 400 hög

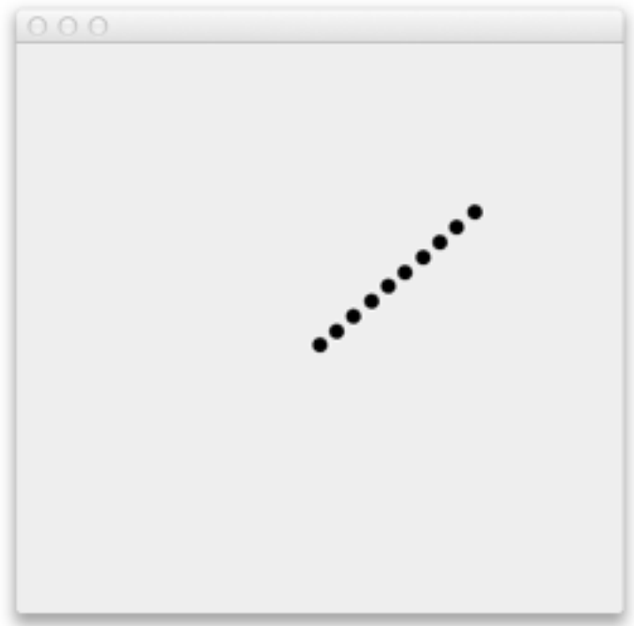
En klass för rotation i 2D plan

```
class Rotate2D {

    // code is based on http://en.wikipedia.org/wiki/Rotation_matrix

    public static double getX(double x, double y, double angle) {
        return x * Math.cos(angle) - y * Math.sin(angle);
    }

    public static double getY(double x, double y, double angle) {
        return x * Math.sin(angle) + y * Math.cos(angle);
    }

}
```

Med timer + rotation

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class RotatingDots extends JFrame {
    private class DemoPanel extends JPanel
                            implements ActionListener {
        private double angle = 0.0;
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            for (int i=0; i<10; i++) {
                int x = 15 * i;
                int y = 0;
                int x1 = (int)Rotate2D.getX(x,y,angle);
                int y1 = (int)Rotate2D.getY(x,y,angle);
                g.fillOval(195+x1,195+y1,10,10);
            }
        }
        public void actionPerformed(ActionEvent e) {
            angle = angle + Math.PI / 128.0;
            this.repaint();
        }
    }
    public RotatingDots() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400,400); setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel); setVisible(true);
        Timer t = new Timer(20,panel);
        t.start();
    }
    public static void main(String[] args) {
        RotatingDots f = new RotatingDots();
    }
}
```

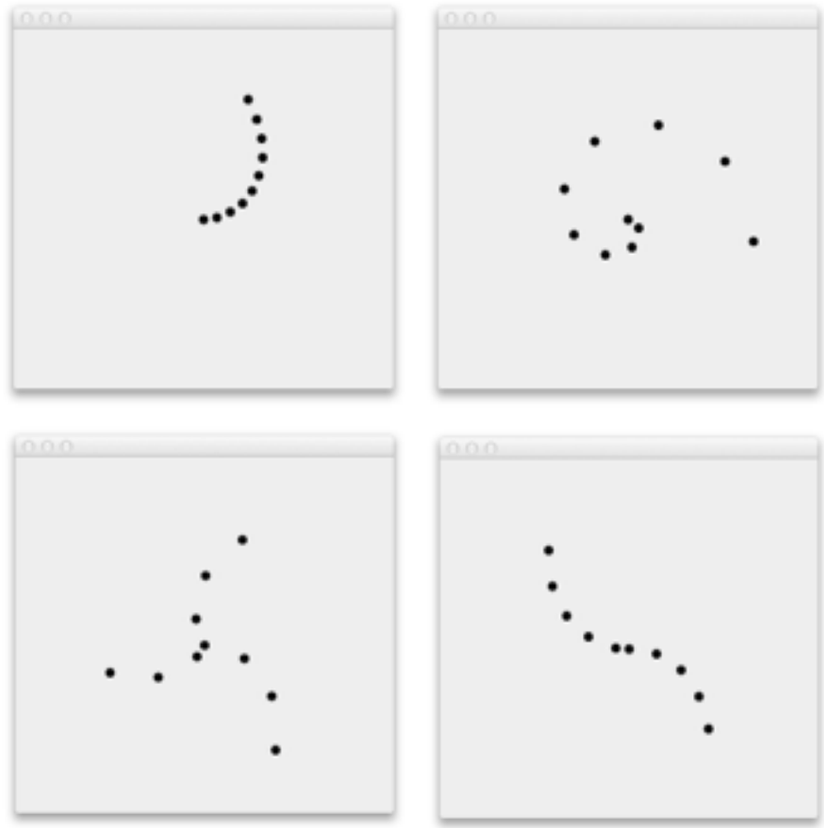




…en annan version

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class RotatingDots2 extends JFrame {
    private class DemoPanel extends JPanel
                            implements ActionListener {
        private double angle = 0.0;
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            for (int i=0; i<10; i++) {
                int x = 15 * i;
                int y = 0;
                int x1 = (int)Rotate2D.getX(x,y,angle * ((double) i / 4));
                int y1 = (int)Rotate2D.getY(x,y,angle * ((double) i / 4));
                g.fillOval(195+x1,195+y1,10,10);
            }
        }
        public void actionPerformed(ActionEvent e) {
            angle = angle + Math.PI / 128.0;
            this.repaint();
        }
    }
    public RotatingDots2() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400,400); setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        add(panel); setVisible(true);
        Timer t = new Timer(20,panel);
        t.start();
    }
    public static void main(String[] args) {
        RotatingDots2 f = new RotatingDots2();
    }
}
```

Bollarna roterar med olika fart.



Att fånga mushändelser

Uppgift: skriv ett program som ritar en boll där musen har tryckts.

Tips: `MouseListener`

Kod som fångar mushändelser

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class MouseDot extends JFrame {
    private class DemoPanel extends JPanel
                            implements MouseListener {
        private int x = -50;
        private int y = -50;
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            g.fillOval(x-5,y-5,10,10);
        }
        public void mouseClicked(MouseEvent e) {
            x = e.getX();
            y = e.getY();
            this.repaint();
        }
        public void mouseEntered(MouseEvent e) {}
        public void mouseExited(MouseEvent e) {}
        public void mousePressed(MouseEvent e) {}
        public void mouseReleased(MouseEvent e) {}

    }
    public MouseDot() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400,400); setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        panel.addMouseListener(panel);
        add(panel); setVisible(true);
    }
    public static void main(String[] args) {
        MouseDot f = new MouseDot();
    }
}
```

Vi lyssnar på mushändelser.

Den körs när musen blivit klickad.

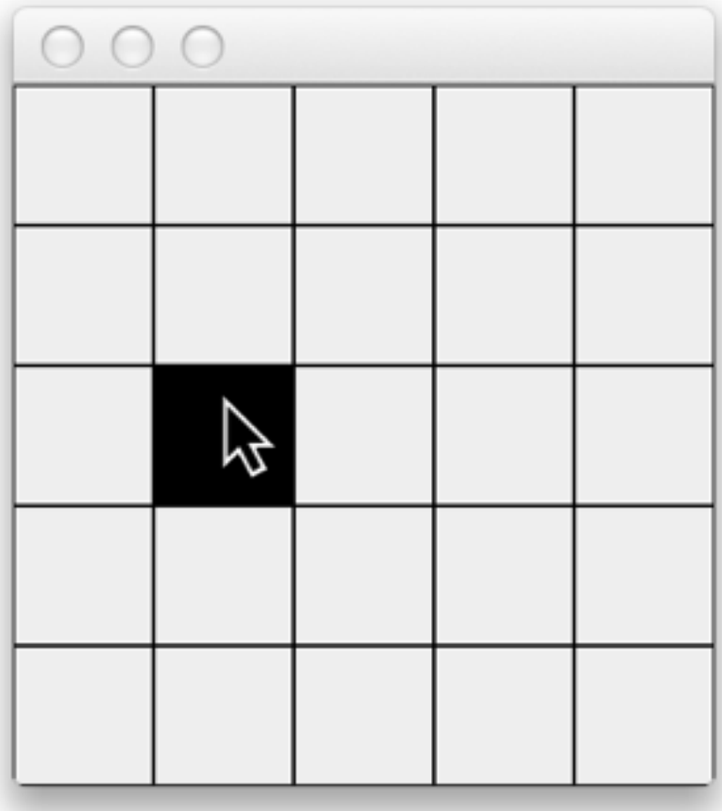
Vi sparar värdena på x och y…

… och så kallar vi på repaint …

… som kör paintComponent.

Början av ett enkelt spel…

Uppgift: skriv ett program som ritar ett rutfält och färgar rutan som tryckts.



Kod som ritar rutfält & händelser

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class MouseBoxes extends JFrame {
    private class DemoPanel extends JPanel
                            implements MouseListener {
        private int BOX_WIDTH = 40;
        private int x = -50;
        private int y = -50;
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            for (int i=0; i<5; i++) {
                for (int j=0; j<5; j++) {
                    if (x == i && y == j) {
                        g.fillRect(i*BOX_WIDTH,j*BOX_WIDTH,
                                   BOX_WIDTH,BOX_WIDTH);
                    } else {
                        g.drawRect(i*BOX_WIDTH,j*BOX_WIDTH,
                                   BOX_WIDTH,BOX_WIDTH);
                    }
                }
            }
        }
        public void mouseClicked(MouseEvent e) {
            x = e.getX() / BOX_WIDTH;
            y = e.getY() / BOX_WIDTH;
            this.repaint();
        }
        public void mouseEntered(MouseEvent e) {}
        public void mouseExited(MouseEvent e) {}
        public void mousePressed(MouseEvent e) {}
        public void mouseReleased(MouseEvent e) {}
    }
    public MouseBoxes() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        panel.addMouseListener(panel);
        panel.setPreferredSize(new java.awt.Dimension(200,200));
        add(panel); pack(); setVisible(true);
    }
    public static void main(String[] args) {
        MouseBoxes f = new MouseBoxes();
    }
}
```

Här ritas en tom ruta.

Här ritas en fylld ruta.

Vi sparar koordinaterna enligt rutornas koordinatsystem.

Att fånga mushändelser

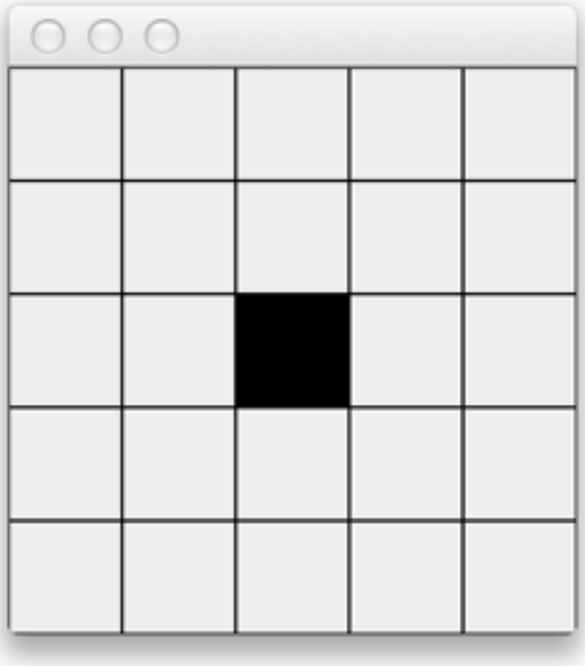
Uppgift: Lös problem 6 från tentan från år 2011 ***utan knappar***.

http://www.cse.chalmers.se/edu/year/2013/course/tda545/courseMtrl/sampleExams/20111021.pdf

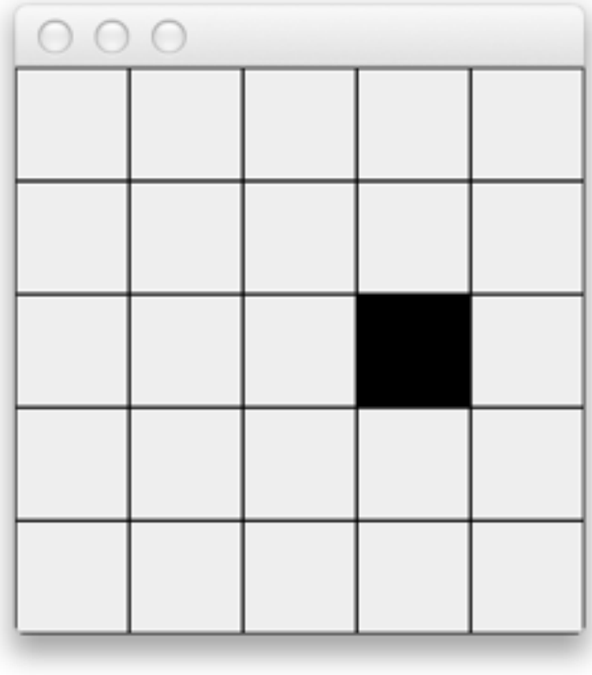
Uppgift: Lös problem 6 från tentan från år 2011 ***med knappar***.

Att fånga tangenthändelser

Uppgift: Skriv ett program där piltangenterna flyttar på den mörka rutan.



Ett tryck till höger
flyttar den svarta rutan:



Kod som fångar tangenthändelser

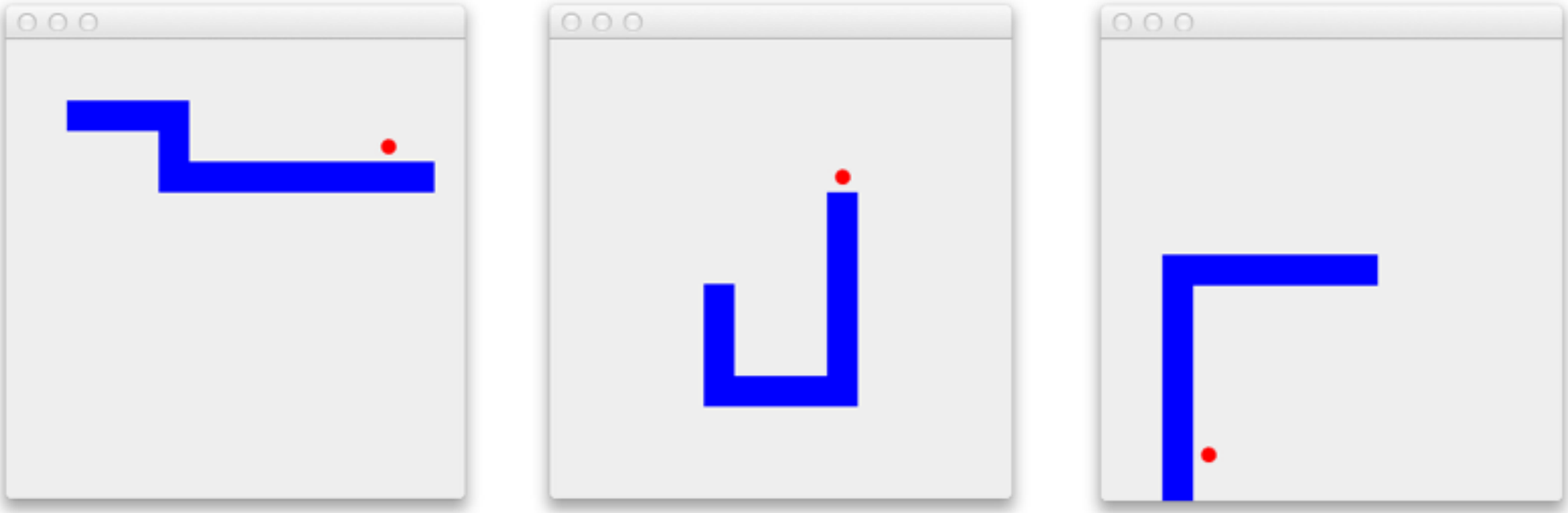
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class KeyDemo extends JFrame {
    private class DemoPanel extends JPanel
                          implements KeyListener {
        private int BOX_WIDTH = 40;
        private int x = 2;
        private int y = 2;
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            for (int i=0; i<5; i++) {
                for (int j=0; j<5; j++) {
                    if (x == i && y == j) {
                        g.fillRect(i*BOX_WIDTH,j*BOX_WIDTH,
                                   BOX_WIDTH,BOX_WIDTH);
                    } else {
                        g.drawRect(i*BOX_WIDTH,j*BOX_WIDTH,
                                   BOX_WIDTH,BOX_WIDTH);
                    }
                }
            }
        }
        public void keyPressed(KeyEvent e) {
            int k = e.getKeyCode();
            if (k == KeyEvent.VK_DOWN && y < 4) {
                y++; repaint();
            } else if (k == KeyEvent.VK_UP && 0 < y) {
                y--; repaint();
            } else if (k == KeyEvent.VK_LEFT && 0 < x) {
                x--; repaint();
            } else if (k == KeyEvent.VK_RIGHT && x < 4) {
                x++; repaint();
            }
        }
        public void keyTyped(KeyEvent e) {}
        public void keyReleased(KeyEvent e) {}
    }
    public KeyDemo() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocation(50,50);
        DemoPanel panel = new DemoPanel();
        addKeyListener(panel);
        panel.setPreferredSize(new java.awt.Dimension(200,200));
        add(panel); pack(); setVisible(true);
    }
    public static void main(String[] args) {
        KeyDemo f = new KeyDemo();
    }
}
```

Denna klass lyssnar på tangentbordet.

När en tangent har pressats så kör vi denna metod, som i vissa fall anropar **`repaint`**.

Ett spel

Uppgift: Skriv ett *snake* spel!



```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.Random;

public class Snake extends JFrame {
    private static int ROWS = 15;
    private static int BOX_WIDTH = 40;
    private class SnakePanel extends JPanel
        implements KeyListener, ActionListener {
        private int k = KeyEvent.VK_DOWN;
        private int x = 0;
        private int y = 0;
        private int[][] board = new int[ROWS][ROWS];
        private Random generator = new Random();
        public SnakePanel() {
            for (int i=0; i<ROWS; i++) {
                for (int j=0; j<ROWS; j++) {
                    board[i][j] = 0;
                }
            }
            board[x][y] = 4;
            placeFood();
        }
        private void placeFood() {
            int x = generator.nextInt(ROWS);
            int y = generator.nextInt(ROWS);
            for (int i=0; i<ROWS; i++) {
                for (int j=0; j<ROWS; j++) {
                    if (board[(i+x) % ROWS][(j+y) % ROWS] == 0) {
                        board[(i+x) % ROWS][(j+y) % ROWS] = -1;
                        return;
                    }
                }
            }
        }
        public void paintComponent(Graphics g) {
            super.paintComponent(g);
            for (int i=0; i<ROWS; i++) {
                for (int j=0; j<ROWS; j++) {
                    if (board[i][j] > 0) {
                        g.fillRect(i*BOX_WIDTH,j*BOX_WIDTH,
                                   BOX_WIDTH,BOX_WIDTH);
                    } else if (board[i][j] < 0) {
                        g.drawOval(i*BOX_WIDTH,j*BOX_WIDTH,
                                   BOX_WIDTH,BOX_WIDTH);
                    }
                }
            }
        }
        public void actionPerformed(ActionEvent e) {
            int old = board[x][y];
            if (k == KeyEvent.VK_DOWN && y < ROWS-1) {
                y++;
            } else if (k == KeyEvent.VK_UP && 0 < y) {
                y--;
            } else if (k == KeyEvent.VK_LEFT && 0 < x) {
                x--;
            } else if (k == KeyEvent.VK_RIGHT && x < ROWS-1) {
                x++;
            }
            board[x][y] = old+1;
            for (int i=0; i<ROWS; i++) {
                for (int j=0; j<ROWS; j++) {
                    if (board[i][j] > 0) {
                        board[i][j] = board[i][j]-1;
                    }
                }
            }
            repaint();
        }
        public void keyPressed(KeyEvent e) {
            k = e.getKeyCode();
        }
        public void keyTyped(KeyEvent e) {}
        public void keyReleased(KeyEvent e) {}
    }
    public Snake() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocation(50,50);
        SnakePanel panel = new SnakePanel();
        addKeyListener(panel);
        panel.setPreferredSize(new java.awt.Dimension(ROWS*BOX_WIDTH,ROWS*BOX_WIDTH));
        add(panel); pack(); setVisible(true);
        Timer t = new Timer(100,panel);
        t.start();
    }
    public static void main(String[] args) {
        Snake f = new Snake();
    }
}
```

Kod för Snake som skrevs på föreläsningen

Obs. Koden blev ju inte färdig.
Kör den så ser du vad som saknas.

Experimentera gärna med koden!

Samma kod finns i zip filen.