

2D-grafik. Något om FrameWorks. Java en kort introduktion och något exempel med AWT, Swing och Graphics2D
Föreläsning 2

Innehåll

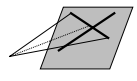
- Något om utmatning
 - hårdvara, tekniker och modeller
 - rastergrafik, primitiver och attribut
- Java mini-introduktion
- Vi tittar också på Tutorialen om 2D-grafik från Sun (<http://java.sun.com/docs/books/tutorial/2d/index.html>)

◀ previous next ▶

GRIP-F2 2D, FW och Java intro

Utmatningstekniker

- Vektoriserade (linjeritande) skärmar
 - var ganska vanliga fram till en bit in på 80-talet
 - uträttning sker genom att linje för linje i en slumpmässig ordning ritas ut på skärmen, därför den engelska termen *random scan*
 - kunde rita linjer mellan godtyckliga punkter på skärmen
- Refresh rate
 - Skärmens fosforyta efterlyser endast en viss tid
 - måste uppdateras flera gånger i sekunden (typiskt 30-70) för att undvika flimmer.


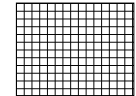
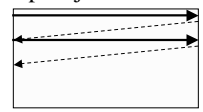


◀ previous next ▶ 2

GRIP-F2 2D, FW och Java intro

...

- Rasterskärm
 - skärmen uppbyggd som en bitkarta
 - bilderna lagras i en speciell refresh buffer
- Raster
 - hela bilden skapas från ett raster, som är en uppsättning horisontella svep-linjer

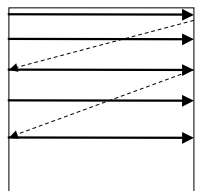




◀ previous next ▶ 3

GRIP-F2 2D, FW och Java intro

...

- Interlacing (sve. radsprång)
 - varannan sveplinje uppdateras i varje svep



- bra för skärmar med låg uppdateringshastighet (med typiskt 30 ggr per sekund). Reducerar flimmer.

◀ previous next ▶ 4

GRIP-F2 2D, FW och Java intro

...

- Monokrom skärm
 - ritar med två färger
- Pixmap (pixel map), bitkarta
 - en representation av en rektangulär area med en vektor av punkter bestående av heltal
 - Namnet pixmap kan betyda både innehållet i refreshbufferten och buffertens minne.
 - För att inte skapa förvirring brukar det senare kallas för *frame buffer*.

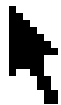
◀ previous next ▶ 5

GRIP-F2 2D, FW och Java intro

...


- Exempel bitkarta till bild (från VisualWorks)

```
0000000000000000
0100000000000000
0110000000000000
0111000000000000
0111100000000000
0111110000000000
0111111000000000
0111110000000000
0111100000000000
0100110000000000
0000011000000000
0000011000000000
0000011000000000
0000011000000000
0000000000000000
```



Cursor normal

```
0000000110000000
0001101001110000
0010011001001000
0010011001001010
0010011001001010
0001001001001101
0001001001001001
0110100000001001
1001100000000001
1000100000000010
0100000000000010
0010000000000010
0001000000000100
0001000000000100
0000100000001000
0000011000000000
0000010000001000
```



Cursor hand

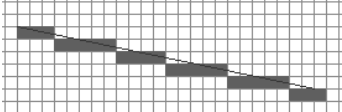
◀ previous next ▶ 6

GRIP-F2 2D, FW och Java intro

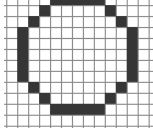
...

- Svepkonvertering/rastrering
 - (matematiska) (linje-)beskrivningar översätts till rasterpunkter

$y = mx + b$



$R^2 = x^2 + y^2$

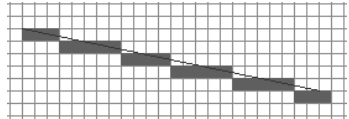
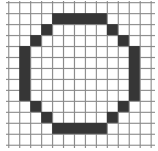


◀ previous next ▶ 7

GRIP-F2 2D, FW och Java intro

Aliasing

- en term från signalprocessing för fel som uppstår då en kontinuerlig signal diskretiseras
 - I grafiska sammanhang syns detta genom att räta linjer blir trappor.

◀ previous next ▶ 8

GRIP-F2 2D, FW och Java intro

Antialiasing

- tekniker för att reducera aliasing-effekter
- vanlig teknik är att låta intensiteten vara proportionell mot hur mycket en viss rasterpunkt befinner sig inom den ideala linjen

Utan antialiasing Antialiasing

◀ previous next ▶

9

GRIP-F2 2D, FW och Java intro

Frameworks (FW)

- Ett FW är centralt för konstruktion av interaktiva applikationer
- Varför ett framework?
 - Svårt att tränga in i toolkits (bibliotek av komponenter), eller följa givna designregler
 - Ett FW ger återanvändbar applikationsdesign eller delsystem
 - Ett FW representeras av ett antal abstrakta klasser och definitioner av hur dessa klasser samarbetar, vilket vägleder programutvecklaren och gör det enklare att konstruera enhetliga gränssnitt och applikationer
 - snabbare
 - mer uniformt

◀ previous next ▶

10

GRIP-F2 2D, FW och Java intro

Vad är ett Framework?

- Ett framework beskriver hur ett problem skall brytas ner
 - Inte bara klasserna utan också hur deras instanser samverkar
 - definierar invarianter som objekten måste dela och anger också hur dom skall användas
 - Ett framework “påtvingar” en modell som programmeraren måste anpassa sig till

◀ previous next ▶

11

GRIP-F2 2D, FW och Java intro

Hollywood

- Omvänd kontroll (Hollywood-principen)
 - Ring inte oss vi ringer er!

Koda med klassbibliotek

Koda med framework

◀ previous next ▶

12

GRIP-F2 2D, FW och Java intro

Grafiskt "Framework"

- Ett grafiskt system fungerar ofta som ett framework i vilket man infogar egna applikationer
- Man anropar
 - initieringsrutiner
 - sätter upp anrop till egna rutiner (call-backs)
 - huvudlooprutin (som sedan anropar dom tidigare definierade call-back-rutinerna)

◀ previous next ▶ 13

GRIP-F2 2D, FW och Java intro

Penplottermodell (alt sköldpaddsgrafik)

- En vanlig modell för 2D-grafik är penplottermodellen
 - Baseras i huvudsak på man kan flyttar en penna
 - som antingen ritar om vidrör pappret
 - eller bara byter position om den är lyft
 - Används bla av
 - LOGO
 - GKS
 - PostScript
- Passar ej bra för 3D eller högre dimensioner

◀ previous next ▶ 14

GRIP-F2 2D, FW och Java intro

Koordinatsystem och koordinater

- Vanligen jobbar vi i Cartesiska koordinater
 - Vissa system har origo nere till vänster
 - andra uppe till vänster
- Världskoordinater
 - Är dom koordinater vi använder i våra applikationer
 - Här bryr vi inte oss om vilka pixlar på skärmen som kommer representera dem
- Skärmkoordinater
 - De koordinater som verkligen representerar punkter på skärmen

◀ previous next ▶ 15

GRIP-F2 2D, FW och Java intro

Vanliga ritrutiner och attribut

- Dom flesta grafiska paket innehåller åtminstone stöd för att rita (2D)
 - Punkter
 - Av olika storlek
 - Linjer
 - Heldragna, streckade, prickade osv
 - Polygoner
 - Fyllda eller icke fyllda
 - Text
 - Med olika storlek och typsnitt
 - Kurvor
 - Som splines och Bezierkurvor

◀ previous next ▶ 16

GRIP-F2 2D, FW och Java intro

Attribut

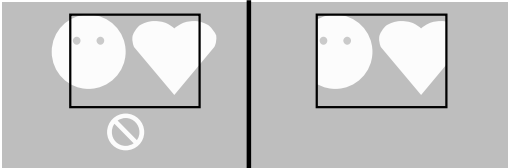
- Det finns också stöd för att ange
 - Färg eller mönster
 - Linjebredd
 - Prickad streckad
- I OpenGL kommer vi senare se prov på fler attribut, tex
 - för att ange olika typer av polygoner
 - och (självklart) i tre dimensioner

17

GRIP-F2 2D, FW och Java intro

Klippning

- Då man konstruerar ett fönster att rita i brukar (eller kan) man också ange klipprektanglar
 - Dvs ange dom områden där objekten skall synas
 - Objekt utanför, eller delvis utanför, "klippes" mot området

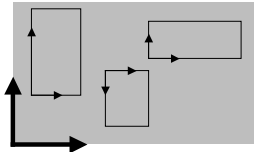


18

GRIP-F2 2D, FW och Java intro

Fönster och vyer

- Ett fönster brukar ofta delas in i delvyer (viewports)



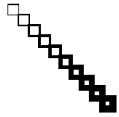
- I varje viewport jobbar man med ett lokalt koordinatsystem (fönstret har också ett koordinatsystem)

19

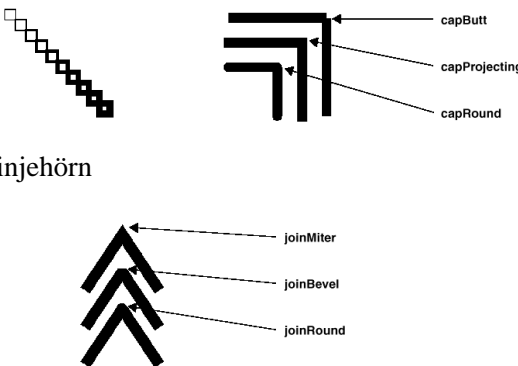
GRIP-F2 2D, FW och Java intro

Attribut

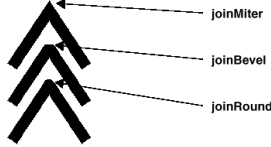
- Linjetjocklek



- Linjeändor



- Linjehörn



20

GRIP-F2 2D, FW och Java intro

Javas API (Application Programming Interface)

- Java:s API innehåller bla awt (abstract windowing toolkit) som delvis kan användas som ett Framework som bla stödjer konstruktion av *applikationer* med fönster
- Konstruktion av egna fönster
 - Fundamental fönsterklass
 - AWT: `Frame` Swing: `JFrame`
 - Ett sätt att konstruera fönster med egna egenskaper (tex speciell utritning) är att subclassa fönsterklassen
 - Beteendet ändras, dvs anpassas till dom egna önskemålen, genom att lämpliga metoder skrivs om i den konkreta (egna) subclassen
 - Tex
 - `paint(...)` eller `paintComponent(...)` för att beskriva utritning
 - Händelsehanterare definieras att ta hand om inmatningshändelser
 - I Swing görs subclassar man ofta `JComponent`

◀ previous next ▶ 21

GRIP-F2 2D, FW och Java intro

Frame och Applet

```

classDiagram
    class java_lang {
        Object
    }
    class java_awt {
        Component
        Container
        Window
        Frame
    }
    class java_applet {
        Applet
    }
    java_lang --> java_awt : Object
    java_awt --> java_applet : Component
    java_awt --> java_applet : Container
    java_awt --> java_applet : Window
    java_awt --> java_applet : Frame
    java_applet --> java_applet : Applet
  
```

- Ofta är det tillräckligt att (Java 1.1)
 - subclassa `Frame` och implementera tex `WindowListener`, `MouseListener` och `MouseMotionListener`
 - senare skall vi titta på hur vi istället kan använda adaptorer
 - skriva `public static void main(String [] args)`
 - instansiera den egna klassen, ange fönsterstorlek och ange att fönster och mushändelser skall tas emot (`addWindowListener(this)` osv), öppna det hela
 - skriva `public void paint(Graphics g)`
 - rita på skärmen
 - konstruera metoder för dom gränssnitt för dom "lyssnare" vi angett att vi skall implementera

◀ previous next ▶ 22

GRIP-F2 2D, FW och Java intro

Java: grunder och syntax

- Java konstruerat på SUN
 - Första versionen släpptes 1995
- Statiskt typat
- Virtuellt maskin som gör det maskinoberoende
- Klassbibliotek
 - språkklasser, stränghantering, grafik, gränssnitt, fönster, nät, händelsehantering, processer, collections, mm
- Bindning till WWW, med applets och liknande
- Har fått sin huvudsakliga spridning via Internet
- Syntax som C, semantik och klasser mer som Smalltalk

◀ previous next ▶ 23

GRIP-F2 2D, FW och Java intro

Java språkets grundbyggsstenar

- Java uppbyggt kring dom objektorienterade fundamenten
 - klass
 - attribut
 - metod, meddelande
 - polymorfi
 - inkapsling
 - arv
 - konstruktör
- Inför också
 - interface
 - package

◀ previous next ▶ 24

GRIP-F2 2D, FW och Java intro

Konstruera klass

- Klassnamn = filnamn
 - En klass skall ha samma namn som den fil den placeras i
 - Klassen Circle i filen Circle.java
 - En fil kan innehålla flera klasser men bara en av dem kan vara deklarerad public,
 - Det är den publika klassens namn som motsvarar filens
- Definition av klass, mall


```
public class KlassNamn extends SuperKlassNamn{
    attribut (instans- och klassvariabler)
    metoder (instans- och klassmetoder)
    konstruktörer
}
```

◀ previous | next ▶ 25

GRIP-F2 2D, FW och Java intro

Deklarera och instansiera

- Deklaration av instans


```
KlassNamn variabelNamn;
```
- Instansiering


```
variabelNamn = new KlassNamn();
```

alternativt, både deklaration och instansiering på en gång

```
KlassNamn variabelNamn = new
    KlassNamn();
```

◀ previous | next ▶ 26

GRIP-F2 2D, FW och Java intro

Sätt upp omgivning

- Environmentvariabel
 - Gör först (NADA)


```
module add java
```
 - Idag (000116) ger detta senaste versionen, JDK1.3

◀ previous | next ▶ 27

GRIP-F2 2D, FW och Java intro

Kompilera och köra

- Kompilera med javac filnamn


```
javac Klassnamn.java
```

 - om allt går bra skapas då en fil med namnet


```
Klassnamn.class
```
- Kör med java klassnamn


```
java Klassnamn
```

◀ previous | next ▶ 28

GRIP-F2 2D, FW och Java intro

Exempel 1 (AWT)

```

package mypackage;
import java.awt.*;

// Klassdefinition (utan extends medför subklass till
// Object)
public class MyApplication {

    public static void main(String [] s) {
        Frame f = new Frame("Mitt första fönster");
        f.setSize(200, 300);
        f.setVisible(true);
    }
}

```

Ofta importerar alla klasser i java.awt

*Instansiera Frame

*Sätt storlek

*Och öppna

◀ previous next ▶

29

GRIP-F2 2D, FW och Java intro

... och så sparar vi, kompilerar och kör

- Spara
 - Spara filen med samma namn som klassen med extension .java, dvs här MyApplication.java
- Kompilera


```
javac MyApplication.java
```

 - Skapar en fil MyApplication.class
- Kör


```
java MyApplication
```

◀ previous next ▶

30

GRIP-F2 2D, FW och Java intro

Exempel 2 (AWT)

```

package mypackage;
import java.awt.*;
public class MyFrame extends Frame {

    public MyFrame() {
    }
    public MyFrame(String s) {
        super(s);
    }

    public void paint(Graphics g) {
        int x = 100, y = 200;
        for (int i = 0; i < 100; i++) {
            g.drawOval(x, y, i, i);
        }
    }

    public static void main(String [] s) {
        MyFrame f = new MyFrame("Mitt andra fönster");
        f.setSize(200, 300);
        f.setVisible(true);
    }
}

```

Vi definierar hur om ritning av fönstret ska gå till

◀ previous next ▶

31

GRIP-F2 2D, FW och Java intro

Exempel 3, Swing, Graphics2D och JFrame

```

package mypackage;
import java.awt.*;
import javax.swing.*;
import javax.swing.*;
import java.awt.geom.*;

public class MyJFrame extends JFrame {

    public void paint(Graphics g) {
        Graphics2D g2 = (Graphics2D)g;
        g2.setStroke(new BasicStroke(8.0f));
        g2.setColor(Color.magenta);
        g2.draw(new Arc2D.Double(40, 70, 200, 100,
            120, 135, Arc2D.OPEN));
    }

    public static void main(String [] s) {
        MyJFrame f = new MyJFrame();
        f.setSize(200, 300);
        f.setVisible(true);
    }
}

```

För JFrame importerar vi javax.swing

För mer avancerad grafik importerar vi java.awt.geom

Med Graphics2D kan vi göra mer avancerade 2D-saker

◀ previous next ▶

32