

Objektorienterad programmering i Java

Föreläsning 5
Kort om Java-Applets

Läsanvisningar

Den här föreläsningen syftar till att ge en bild av vad en Java-Applet är och är ganska fristående från föregående föreläsningar. I boken finns det ett litet avsnitt om Applets och HTML, avsnitt 2.6(2.9), samt ett Appendix K (J) som innehåller en HTML-tutorial. I övrigt finns det en mängd Applet-exempel som är vitt utspridda i boken.

Java och Internet

Stenålderns Internet

Internet var på 70-talet helt text-baserat utan bilder och ljud och kördes nästan helt uteslutet på UNIX-datorer inom universitets- militär- och forskningsvärlden. Största delen utgjordes av E-MAIL samt FTP, NEWS och TELNET tjänster och var till största delen endast utbyggt i USA. Till sverige kom internet först i början på 80-talet.

När WWW kom till världen

År 1980 utvecklade en datakonsult vid namn Tim Berners-Lee en HTML-linkande standard åt forskare på CERN i Schweiz för att hantera gemensamma dokument som möjliggjorde länkning mellan noder i ett nätverk. Senare under åren 1989-1991 vidareutvecklade Tim och några andra dessa idéer till ett program som visade hypertext (HTML) och ett protokoll som kunde skicka sådan text över internet (HTTP). Helt plötsligt blev internet mer lättillgängligt för vanliga människor, World Wide Web hade skapats.

Java och Internet, forts.

Från helt okänt, till del av vardag

I slutet av 1993 kom det program från NCSA (Mosaic) som gjorde det möjligt att “surfa” på WWW bland text och bilder. Kort därefter kom Navigator från Netscape och på ett par år växte det fram en mängd internetleverantörer som gav vanliga människor möjligheter att via ett modem få tillgång till nätet och det dröjde inte länge förrän internet blev ett normalt inslag i vår vardag. Idag är det svårt att föreställa sig att WWW funnits i knappt 10 år!

HTML räcker inte till!

HTML är det format som information lagras i på Internet. HTML har utvecklats sedan den första, något enkla versionen, till ett idag ganska omfattande språk. Problemet är att HTML är ganska statiskt och har mycket begränsade möjligheter till interaktion med den person som surfar. Sådan interaktion kräver något mer avancerat än HTML.

Java och Internet, forts.

Hur JAVA kom till

För att göra WWW mer interaktivt krävdes det att man skulle kunna skicka program över nätet. Vidare så krävdes det av ett sådant program att det skulle fungera på olika typer av datorer och operativsystem som finns på Internet. Problemet med att skicka ett program över nätet var att de datorer som är anslutna till Internet är av varierande fabrikat och kör olika operativsystem. Den som skriver ett sådant program skulle inte i förväg kunna veta vilken typ av dator och operativsystem personerna som laddar ner programmet sitter och surfar med. Sådana program som fungerar på olika typer av datorer och operativsystem hade man inte hört talas om.

Men det visade sig att Sun Microsystems redan hade utvecklat ett sådan maskinoberoende programspråk som de kallade för Oak, och de insåg att det skulle passa perfekt för Internet. Sun döpte om språket till Java och lanserade det på Internet. Netscape och Microsoft byggde snabbt in stöd för Java i sina webbläsare.

Applet vs. Application

Det finns två typer av Java-program

- Applications
- Applets

En application är ett sådant program som gjordes i lab1 t.o.m. lab3 där man startar programmet med “java klassnamn” och är inte avsett att användas över Internet. En Applet skiljer sig från en application på ett antal sätt:

- Applets startas från en HTML-sida och körs i en Webb-läsare.
- De har vissa säkerhetsbegränsningar. T ex så kan de inte öppna filer på den dator de körs på eller starta andra program.

Enkelt Applet-exempel

Låt oss ta ett enkelt exempel på en Application och se hur den skulle kunna se ut som en Applet. Så här kan det se ut som en Application:

```
class ForelasApp1
{
    public static void main(String args[])
    {
        System.out.println("Hello world!");
    }
}
```

En Application kompilerar man ju med “javac filnamn.java” och startar sedan med “java klassnamn”. Låt oss nu se hur samma program skulle kunna se ut som en Applet och hur man skapar och startar den.

Applet-Exempel forts.

Skapa .java fil

Precis som med Applications skriver man in programmet i en text-editor (t ex notepad) och sparar som en .java fil. Vi skriver in nedanstående program och sparar som “**ForelasApp1.java**”.

```
import java.applet.*;
import java.awt.*;
public class ForelasApp1 extends Applet
{
    public void paint(Graphics page)
    {
        page.drawString("Hello world!", 50, 25);
    } // method paint
} // class ForelasApp1
```

Kompilera .java filen

Sedan kompilerar man programmet med “**javac ForelasApp1.java**” så att en “**ForelasApp1.class**” fil skapas.

Applet-Exempel forts.

(577-)

Skapa .html fil och inkludera din Applet

För att kunna köra sin Applet måste man skapa en .html fil som inkluderar den .class fil som skapades när man kompilerade sin Java-Applet. Skapa följande HTML-kod i Notepad och spara som “**ForelasApp1.html**” i samma katalog som .java och .class filen ligger.

```
<HTML>
<HEAD>
<TITLE> En enkel Applet </TITLE>
</HEAD>
<BODY>
```

Here is the output of my program:

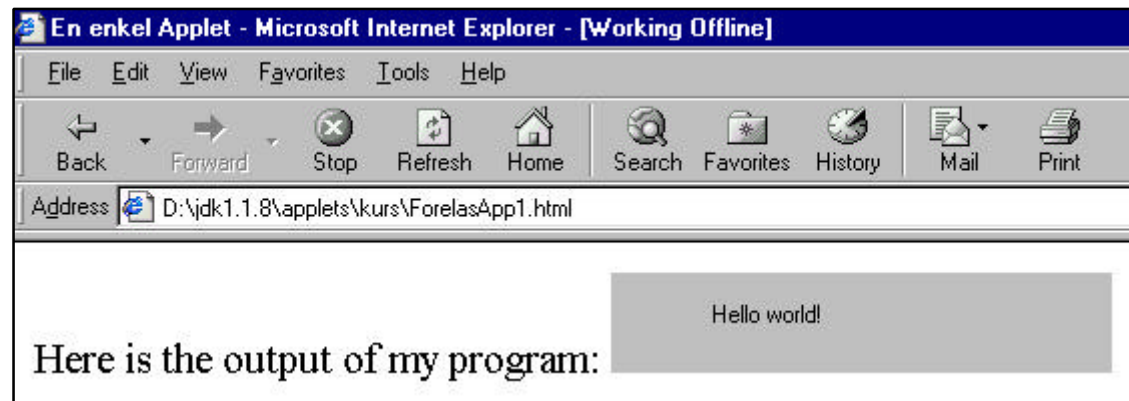
```
<APPLET CODE="ForelasApp1.class" WIDTH=150 HEIGHT=50>
</APPLET>
</BODY>
</HTML>
```

Applet-Exempel forts.

Starta din Applet

För att köra sin Applet så öppnar man helt enkelt sin “**ForelasApp1.html**” med ett program som kan visa Applets. Det kan tex var en WWW-läsare som klarar att visa Applets (så som Netscape och Internet Explorer) eller en Applet-viewer så som den som följer med JDK (den startas med `appletviewer` **ForelasApp1.html** men den visar endast appleten, inte HTML-texten)

Resultatet kommer att bli något i stil med följande: Den vänstra bilden visar exekvering via Applet viewer, den högra bilden via Internet explorer):



Applets gör det möjligt att blanda text och bilder i HTML med grafik genererad av en Applet såsom t ex enkel text (som Hello World exemplet), linjer, cirklar och bilder, eller hela grafiska användargränssnitt med knappar och inmatningsfält o s v.

Viktiga funktioner hos en Applet

se bl.a sid 673-675

(619-)

- En Applet har ingen main-metod som en Application har.
- En Applet har tre stycken speciella metoder varav minst en måste implementeras:
 - `init(...)` - anropas automatiskt när en Applet första gången har lästs in i web läsaren. När appleten är inladdad helt enkelt.
 - `start(...)` - anropas automatiskt efter `init()` när programmet skall starta, eller när man tryckt reload eller refresh.
 - `paint(...)` - anropas automatiskt varje gång en applet ritas om. Kan även startas via metoden `repaint()`

Graphics-klassen

- Till `paint()`-metoden skickas automatiskt in ett objekt som parameter som är av typen **Graphics**. (se HelloWorld exemplet, `paint(Graphics page)`).
- Genom detta Graphics-objekt kan man använda dess metoder rita ut saker på skärmen. Tex så använde vi `drawString()` i HelloWorld exemplet för att rita ut en textsträng. Man kan tänka sig ett graphics-objekt som en slags rityta där man kan presentera grafiska figurer av olika slag.
- Denna rityta som Graphics-objektet utgör har ett koordinatsystem. Varje punkt eller pixel på denna rityta har sin egen koordinat. När man skall rita ut något på ritytan, tex en text med `drawString()` så anger man alltid en position som består av ett X- och ett Y-värde (X, Y). Koordinatsystemet är utformat så att översta vänstra hörnet har koordinaterna 0, 0 (X=0 och Y=0) och nedersta högra hörnet `<width-1, height-1>`. I vårt HelloWorld-exempel angav vi koordinaterna `<150,50>`. Detta innebär

0,0



149,49

Applet och HTML

En Applet inkluderas i en HTML-sida m h a `<APPLET>`-taggen. Den enklaste formen på denna tag ser ut på följande sätt:

```
<APPLET CODE=AppletSubclass.class WIDTH=anInt HEIGHT=anInt></APPLET>
```

- **CODE** Anger vilken .class fil som Appleten består av.
- **WIDTH** Anger bredden på appleten.
- **HEIGHT** Anger höjden på appleten

Parametrar till Applets

(593-)

HTML-tag

Om man vill skicka med parametrar till sin applet, gör man det m h a en <PARAM> HTML-tag på följande sätt:

```
<APPLET CODE=AppletSubclass.class WIDTH=anInt HEIGHT=anInt>  
<PARAM name="paramName" value="paramValue"></APPLET>
```

Varje parameter har ett namn som anges med `name=""` och ett värde som anges med `value=""`. Exempel på en parameter med namnet “meddelande” och värdet “Hej jag heter Kalle!”:

```
<APPLET CODE="ForelasApp2.class" WIDTH=200 HEIGHT=200>  
<PARAM name="meddelande" value="Hej jag heter Kalle!"></APPLET>
```

Det går bra att ha flera parametrar så länge de har olika namn.

Parametrar till Applets, forts.

Läsa av en parameter från applet

För att ta reda på värdet på en parameter inne i sin applet använder man sig av `getParameter(String paramName)`-metoden som returnerar en sträng som utgör värdet (value) på parametern med det angivna namnet. Exempel som gör att Strängen message får värdet *"Hej jag heter Kalle! "*:

```
String message = getParameter("meddelande");
```

Om man anger ett namn på en parameter som inte finns så får man tillbaka en tom sträng.

Vi ändrar vår Applet `ForelasApp1` så att den hämtar en text från html-filen, vi kallar den nya appleten för `ForelasApp2`

ForelasApp2.java

Applet som hämtar parameter från html-fil

```
import java.applet.*;
import java.awt.*;
public class ForelasApp2 extends Applet
{
    private String medText; // här läggs meddelande texten från html vid getParameter
    public void init()
    {
        //meddelande är namn på parametern i html
        medText=getParameter("meddelande");
    }// method init
    public void paint(Graphics page)
    {
        page.drawString(medText, 50, 25); //här skrivs meddelandet ut
    }// method paint
}// class ForelasApp2
```


ForelasApp2.html

```
<HTML>
```

```
<HEAD>
```

```
<TITLE> En enkel Applet </TITLE>
```

```
</HEAD>
```

```
<BODY>
```

Here is the output of my program:

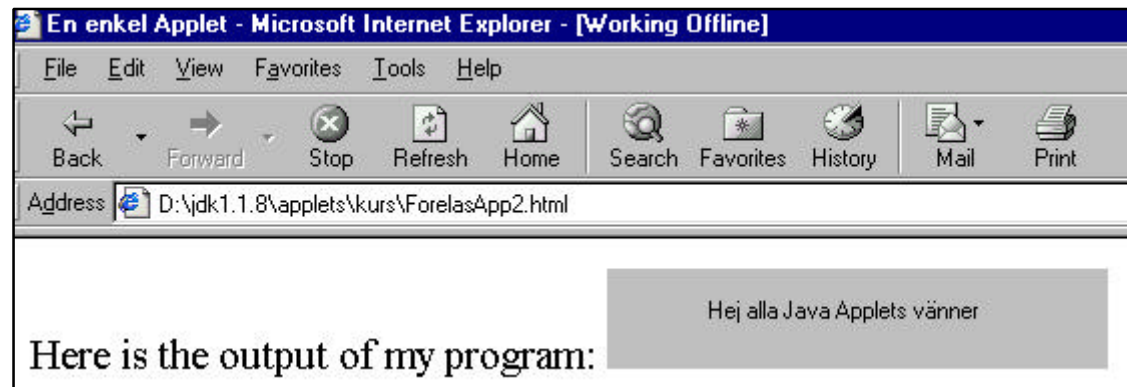
```
<APPLET CODE="ForelasApp2.class" WIDTH=250 HEIGHT=50>
```

```
<Param name="meddelande" value="Hej alla Java Applets vänner" >
```

```
</APPLET>
```

```
</BODY>
```

```
</HTML>
```



Fler Funktioner

Några exempel på fler metoder ur Applet-klassens interface:

- `resize` Ändrar storleken på appleten, sid 675 (620).
- `stop` Anropas automatiskt när appleten skall sluta exekvera.

**Några exempel på fler metoder ur Graphics-klassens interface:
sid 252- (95-)**

- `drawLine` Ritar en linje
- `setColor` För att ändra färg, sid 255-
- `setFont` För att byta font, sid 275-

Fler användbara metoder i dessa klasser hittar man om man tittar på klassernas hela interface där man kan se en lista på alla publika metoder. Java-klassernas interface kan man t ex hitta på denna [www-sida](http://java.sun.com/products/jdk/1.1/docs/api/packages.html):

<http://java.sun.com/products/jdk/1.1/docs/api/packages.html>

En applet med font, och färg

```
//Något om färger se sid 255- (98- ) ,   Något om fonter se sid 275-
import java.awt.*;
import java.applet.*;
public class AppFont extends Applet
{
    Font f;
    public void init ()
    {
        f=new Font("Arial", Font.ITALIC,30); //sätter fonten Arial , kursiv(italic) och storlek 30
        setBackground(Color.blue); //sätt bakgrundsfärgen till blå
        setSize(400,200);           //bredd 400, höjd 200 pixels
    }// metoden init

    public void paint (Graphics g)
    {
        Color farg;
        g.setFont(f);
        farg=new Color(255,255,0); // gul
        g.setColor(farg);
        g.drawString("Detta visar en applet",45,100);
    } //metoden paint
} //class AppFont
```

Här är html -filen och resultat

```
<HTML>
  <HEAD>
    <TITLE> Applet App1</TITLE>
  </HEAD>
  <BODY>
    En strunt applet
    <br>
    <APPLET CODE="AppFont.class" WIDTH=600 height=200> </APPLET>
  </BODY>
</HTML>
```



Här är en applet med Checkbox

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*; // För händelser
public class CheckboxTest extends Applet implements ItemListener{
    private Checkbox check = new Checkbox("CheckboxTest"); // skapar en checkbox
    public void init(){
        add(check); // lägger check till vår applet, gör att den ritas ut
        check.addItemListener(this); // addera lyssnare, this gör att den lyssnar på just denna applet
    } //metoden init
    public void itemStateChanged(ItemEvent e){
        if(check.getState()) //om status on ändra till röd , returnerar true vid on
            setBackground(Color.red);
        else // om status off, ändra till vit
            setBackground(Color.white);
        repaint();
    } //metod itemStateChanged
    public void paint(Graphics g)
    {
        g.drawString("Applet med checkbox, vid selection blir bakgrunden röd",45,100);
    } // metod paint
} // klass CheckboxTest
```

Här är html filen för CheckboxTest

```
<HTML>
  <HEAD>
    <TITLE> Applet App1</TITLE>
  </HEAD>
  <BODY>
    En strunt applet
    <br>
    <APPLET CODE="CheckboxTest.class" WIDTH=600
height=200></APPLET>
  </BODY>
</HTML>
```

Flera applets

Flera applets hittar du bland kursmaterialet
klicka på denna bild

