# 17.0 Appletter

Der findes to slags Java-programmer: **Applikationer** og **appletter.** En Java applet er et Java program som er lavet til at blive lagt ind i et HTML-dokument, transporteret over et netværk og kørt/udført af en Web-browser.  
En Java **applikation** er en Java-program, der kan køre selvstændigt vha. en Java-oversætter. Hidtil har vi kun arbejdet med applikationer. Applikationerne kan kendes på at de indeholder en *main*-metode som oversætteren kører/afvikler.

**Appletter** indeholder ikke en *main*-metode idet de køres inde i en browser. Browseren virker her som en main-metode. Når en applet afvikles køres *paint*-metoden. Egentlig køres en initialiserings-metode først: *init()* , - hvis en sådan er skrevet. Eller køres der en nedarvet default-*init*-metode umiddelbart før *paint*-metoden.  
Java appletter var de første udførbare programmer der kunne overføres via World Wide Web. Java-appletter er endnu et medie der kan udveksles over nettet, - ligesom vi er vant til at tekst, billeder og lyd kan udveksles.

Lokal computer

Fjern computer

Web browser

Java-oversætter

Applet

init(), paint()

Java compiler

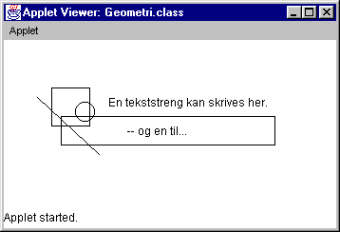
Java oversætter

Bytecode compiler

Internettet / WWW i HTML-dok.

Figuren viser hvordan Java-bytecoden kan køres lokal eller sendes via www til en anden computer.  
Selvom appletter er lavet til at blive overført via nettet og køre i en browser, behøver den det ikke. Med Sun’s Java Software Development Kit følger en A*pplet Viewer”*, der kan oversætte og afvikle appletter. Dette vil vi benytte os af mens vi udforsker og udvikler appletter.  
Lad os se et eksempel (Filer se side 7):

1 import java.applet.Applet;   
2 import java.awt.\*;  
3

4 public class Geometri extends Applet  
5 {  
6 public void paint (Graphics page)  
7 {  
8 page.drawRect (50, 50, 40, 40); // square  
9 page.drawRect (60, 80, 225, 30); // rectangle  
10 page.drawOval (75, 65, 20, 20); // circle  
11 page.drawLine (35, 60, 100, 120); // line  
12 page.drawString ("En tekststreng kan skrives her.", 110, 70);  
13 page.drawString ("-- og en til...", 130, 100);

14 }

15 }

Når du kører[[1]](#footnote-1) programmet vil du se dette:

Linie 1 og 2 er import-erklæringer for at få adgang til pakker med java-klasser: Dels Applet-klasserne og dels awt-klasserne (Abstract Windowing Toolkit).  
 I linie 4 sker der noget særlig nyt foruden navngivning: Klassen Geometri **extends** Applet, dette betyder at den arver fra den oprindelige Appletklasse herunder default-metoder som fx *init().*  
Arv er et vigtigt begreb indenfor Java-programmering. Dette emne vil jeg komme ind på i en senere lektion.  
 Linie 6 er metoden *paint()* som medtager argumentet *page* af typen *Graphics*. Dette er meget populært sagt et objekt der er håndtag til den grafiske flade hvorpå der kan tegnes og skrives.   
 Linie 8-13 er metoder som dette *Graphics*-objekt har: Firkant, oval, linie, og tekststreng. Læg mærke til at der angives både hvor store (lange og brede) figurerne skal være samt hvor på fladen de skal være. Et Java koordinatsystem ser således ud:

**Opgave 17.1:** Prøv at ændre på  
 tallene og iagttag hvad der sker.  
Angiv hvad der er højde, bredde og startpunkt.

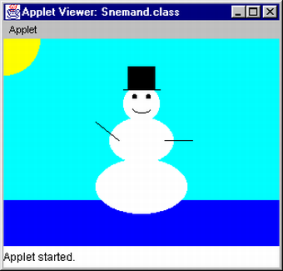
(0,0) x-akse

y-akse

**17.1 Appletter i HTML-dokumenter**

For at kunne sende en applet over nettet så den kan afvikles af en browser, skal appletten stå i et html-dokument. Følgende kode skal stå for at få nedenstående snemand sendt med:

<applet code="applettPakke.Snemand.class" width=300 height=225>  
</applet>

Som før nævnt er en applet ligesom et billede, en tekst eller en lyd. Først skal typen angives, derefter navnet på Java-bytecoden (incl. evt. sti). Tilsidst angives størrelsen på det vindue hvori appletten skal vises.   
I Eclipse skabes der altid et html-dokument med samme navn som appletten, når man beder om at få dannet en ny applet.  
 Her er en Snemands applet:

package applettPakke;

import java.applet.Applet;

import java.awt.\*;

public class Snemand extends Applet

{

public void paint(Graphics page)

{

final int MID = 150;

final int TOP = 50;

setBackground (Color.cyan);

page.setColor (Color.blue);

page.fillRect (0, 175, 300, 50); // jord

page.setColor (Color.yellow);

page.fillOval (-40, -40, 80, 80); // sol

page.setColor (Color.white);

page.fillOval (MID-20, TOP, 40, 40); // hoved

page.fillOval (MID-35, TOP+35, 70, 50); // øvre torso

page.fillOval (MID-50, TOP+80, 100, 60); // nedre torso

page.setColor (Color.black);

page.fillOval (MID-10, TOP+10, 5, 5); // venstre øje

page.fillOval (MID+5, TOP+10, 5, 5); // højre øje

page.drawArc (MID-10, TOP+20, 20, 10, 190, 160); // smil

page.drawLine (MID-25, TOP+60, MID-50, TOP+40); // venstre arm

page.drawLine (MID+25, TOP+60, MID+55, TOP+60); // højre arm

page.drawLine (MID-20, TOP+5, MID+20, TOP+5); // hatteskygge

page.fillRect (MID-15, TOP-20, 30, 25); // hattepul

}

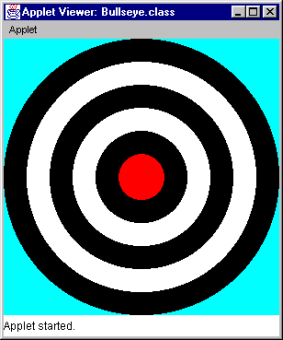
}

Til slut kommer tre eksempler på mere komplicerede appletter, hvor der bruges betingelser og loops til at tegne forskellige figurer og tildele dem farver.

package applettPakke;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Bullseye.java Author: Lewis and Loftus

// Demonstrerer brug af betingelser og loops

// til at styre tegningen

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.applet.Applet;

import java.awt.\*;

public class Bullseye extends Applet

{

private final int MAX\_WIDTH = 300;

private final int NUM\_RINGS = 5;

private final int RING\_WIDTH = 25;

//-------------------------------

// Paints a bullseye target.

//-------------------------------

public void paint (Graphics page)

{

int x = 0, y = 0, diameter;

setBackground (Color.cyan);

diameter = MAX\_WIDTH;

page.setColor (Color.white);

for (int count = 0; count < NUM\_RINGS; count++)

{

if (page.getColor() == Color.black) // alternate colors

page.setColor (Color.white);

else

page.setColor (Color.black);

page.fillOval (x, y, diameter, diameter);

diameter -= (2 \* RING\_WIDTH);

x += RING\_WIDTH;

y += RING\_WIDTH;

}

// Draw the red bullseye in the center

page.setColor (Color.red);

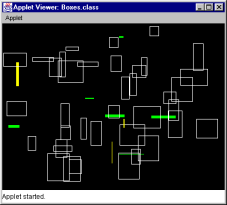
page.fillOval (x, y, diameter, diameter);

}

}

package applettPakke;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Boxes.java Author:Lewis and Loftus

// Demonstrates brug af betingelser

// og loops til at styre tegningen.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.applet.Applet;

import java.awt.\*;

public class Boxes extends Applet

{

private final int NUM\_BOXES = 50;

private final int THICKNESS = 5;

private final int MAX\_SIDE = 50;

private final int MAX\_X = 350;

private final int MAX\_Y = 250;

//-------------------------------

// Paints boxes of random width and

// height in a random location.

// Narrow or short boxes are highlighted with a fill color.

//-----------------------------------------------------------------

public void paint(Graphics page)

{

int x, y, width, height;

setBackground (Color.black);

for (int count = 0; count < NUM\_BOXES; count++)

{

x = (int) (Math.random() \* MAX\_X);

y = (int) (Math.random() \* MAX\_Y);

width = (int) (Math.random() \* MAX\_SIDE);

height = (int) (Math.random() \* MAX\_SIDE);

if (width <= THICKNESS) // check for narrow box

{

page.setColor (Color.yellow);

page.fillRect (x, y, width, height);

}

else

if (height <= THICKNESS) // check for short box

{

page.setColor (Color.green);

page.fillRect (x, y, width, height);

}

else

{

page.setColor (Color.white);

page.drawRect (x, y, width, height);

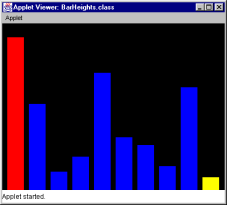
}

} //slut for-sætning

} // slut paint-metode

} // slut klasse

package applettPakke;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// BarHeights.java Author:Lewis & Loftus

// Demonstrerer brug af betingelser og loops

// til at styre tegningen.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.applet.Applet;

import java.awt.\*;

public class BarHeights extends Applet

{

private final int NUM\_BARS = 10;

private final int BAR\_WIDTH = 30;

private final int MAX\_HEIGHT = 300;

private final int GAP = 9;

//----------------------------------

// Tegner søjler med forskellige højder,

// finder højeste og laveste, som gen-

// tegnes i andre farver til sidst.

//-----------------------------------------------------------------

public void paint (Graphics page)

{

int x, height;

int tallX = 0, tallest = 0, shortX = 0, shortest = MAX\_HEIGHT;

setBackground (Color.black);

page.setColor (Color.blue);

x = GAP;

for (int count = 0; count < NUM\_BARS; count++)

{

height = (int) (Math.random() \* MAX\_HEIGHT);

page.fillRect (x, MAX\_HEIGHT-height, BAR\_WIDTH, height);

if (height > tallest) // Keep track of the tallest bars

{

tallX = x; tallest = height;

}

if (height < shortest) // Keep track of the shortest bars

{

shortX = x; shortest = height;

}

x = x + BAR\_WIDTH + GAP;

}

// Redraw the tallest bar in red

page.setColor (Color.red);

page.fillRect (tallX, MAX\_HEIGHT-tallest, BAR\_WIDTH, tallest);

// Redraw the shortest bar in yellow

page.setColor (Color.yellow);

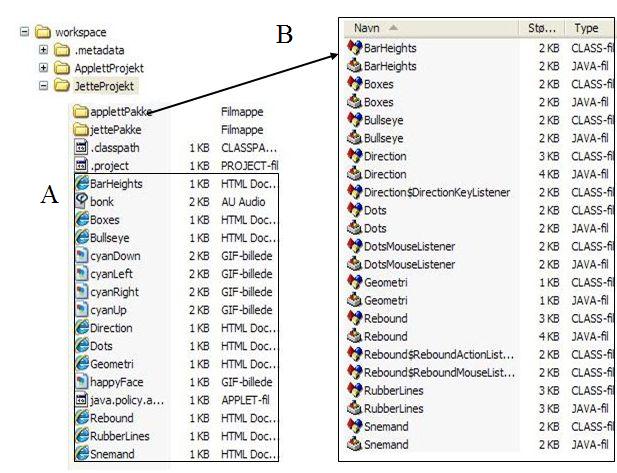
page.fillRect (shortX, MAX\_HEIGHT-shortest, BAR\_WIDTH, shortest);

}

}

Sådan får du programmerne ind i din editor:

1. Åben Eclipse, åben dit Eclipse-projekt og opret en ny pakke med navnet ”applettPakke”. Gør dig selv en tjeneste og stav præcist og husk der er forskel på små og store bogstaver i java.
2. Hent zip-filenApplett_zipfil fra Fronter i program-mappen *PrgMappeLektion13-14.*
3. Åben en sti-finder eller lignende og udpak filerne. Vær opmærksom på at zip filen indeholder:   
    A) en hel række html-filer, gif, lyd m.m.   
   og B) en Eclipse-mappe/pakke ”applettPakke” med alle \*.java og \*.class filer.  
     
   Udpak zip-filen inde i dit Eclipse-projekt.



Filerne skal ligge om illustreret herover:   
 - html, gif og lyd ligger inde i projekt og ved siden af pakker.  
 - \*.java og \*.class ligger i pakke ”applettPakke” under projektet.

4) Opdater din Eclipse-editor med F5.

5) Hvis du ikke har overholdt navnet ”applettPakke”, må du selv ændre samtlige package-navne i java-filerne, samt rette kildeteksten til samtlige html-filer…

### opgave 17.2

Lav en applet der tegner det olympiske logo. Cirklerne i logoet skal være farvet blå, gul, sort, grøn, og rød.

### Opgave 17.3

Lave en applet der viser dit navn i skygge tekst ved at skrive navnet i sort og derefter gentegne navnet en lille smule forskudt i en lysere farve.

God arbejdslyst Just

1. Kør appletten fra fx stifinderen ved at dobbeltklikke på Geometri.html. Du kan også bruge din browser. Men husk det skal være html-filen ( ikke java kildeteksten) [↑](#footnote-ref-1)