

Eclipse

Avsikt

Att bekanta dig med Eclipse programmeringsmiljö, dvs att med hjälp av Eclipse

1. skapa ett nytt projekt
2. skriva in källkod (sparas som .java-fil)
3. kompilera (översätta) koden till byte-kod (sparas som .class-fil)
4. köra programmet.

Men innan du börjar med detta måste du göra ett par installationer

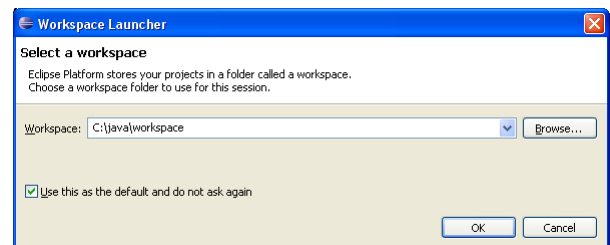
1. Installera **J2SE**. J2SE innehåller nödvändiga verktyg för att kompilera och exekvera program skrivna i programmeringsspråket java. Hämta dokumentet JSE.pdf från kurssidan och följ instruktionerna under rubriken "INSTALLATION AV JSE 6 (Windows)". För att Eclipse ska fungera (nästa punkt) behöver du bara utföra de två översta raderna i instruktionen.
2. Installera **Eclipse**. Nuvarande version heter **Ganymede**. Klicka på länken på kurssidan eller gå direkt till <http://www.eclipse.org/downloads/>, hämta hem Eclipse och packa upp katalogen Eclipse.

När du har gjort detta ska du skriva ett mycket enkelt program och i samband med detta gå igenom ovanstående punkter.

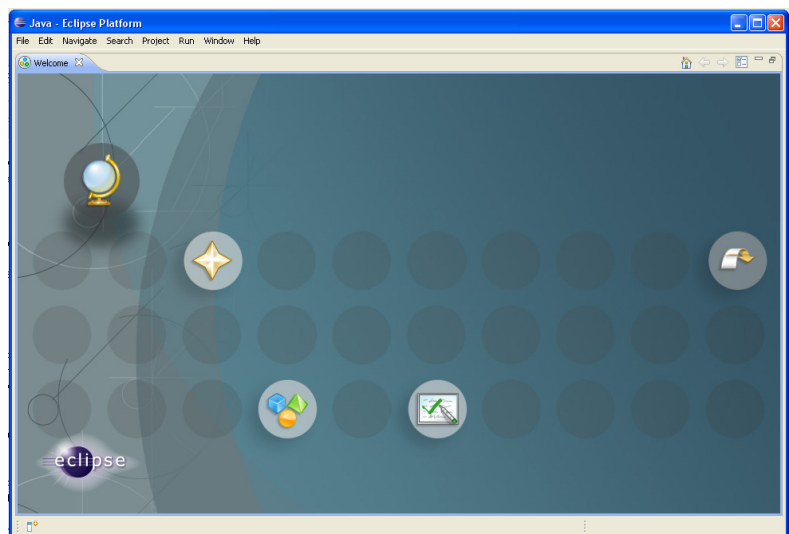
Starta Eclipse

- Dubbelklicka på eclipse.exe i katalogen med namnet Eclipse.

All den kod som du skriver med hjälp av Eclipse sparas någonstans på datorn. Var Eclipse-projekt ska sparas anger du första gången Eclipse startas. Skriv t.ex. in `C:\java\workspace` i dialogen som visar sig. Markera "Use this as the default and do not ask again" och klicka sedan på OK.



Nu ska ett fönster liknande figuren till höger synas.



1. Skapa ett nytt projekt

I Eclipse använder man sig av projekt. Projektet hjälper till att hålla ordning på filer, paket och mycket annat som kan ingå i utvecklingen av program.

- Välj **File – New – Java Project**

Nu visas ett dialog-fönster - **New Java Project**.

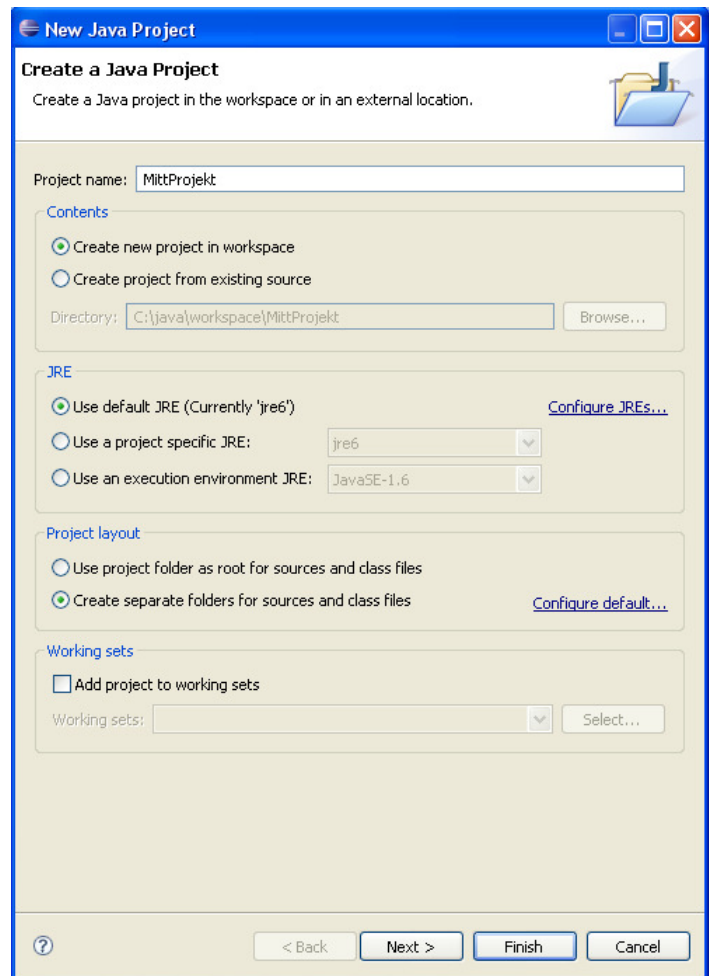
Överst ska du ge projektet ett namn. Kalla det för **MittProjekt**.

Övriga inställningar kan du låta vara oförändrade.

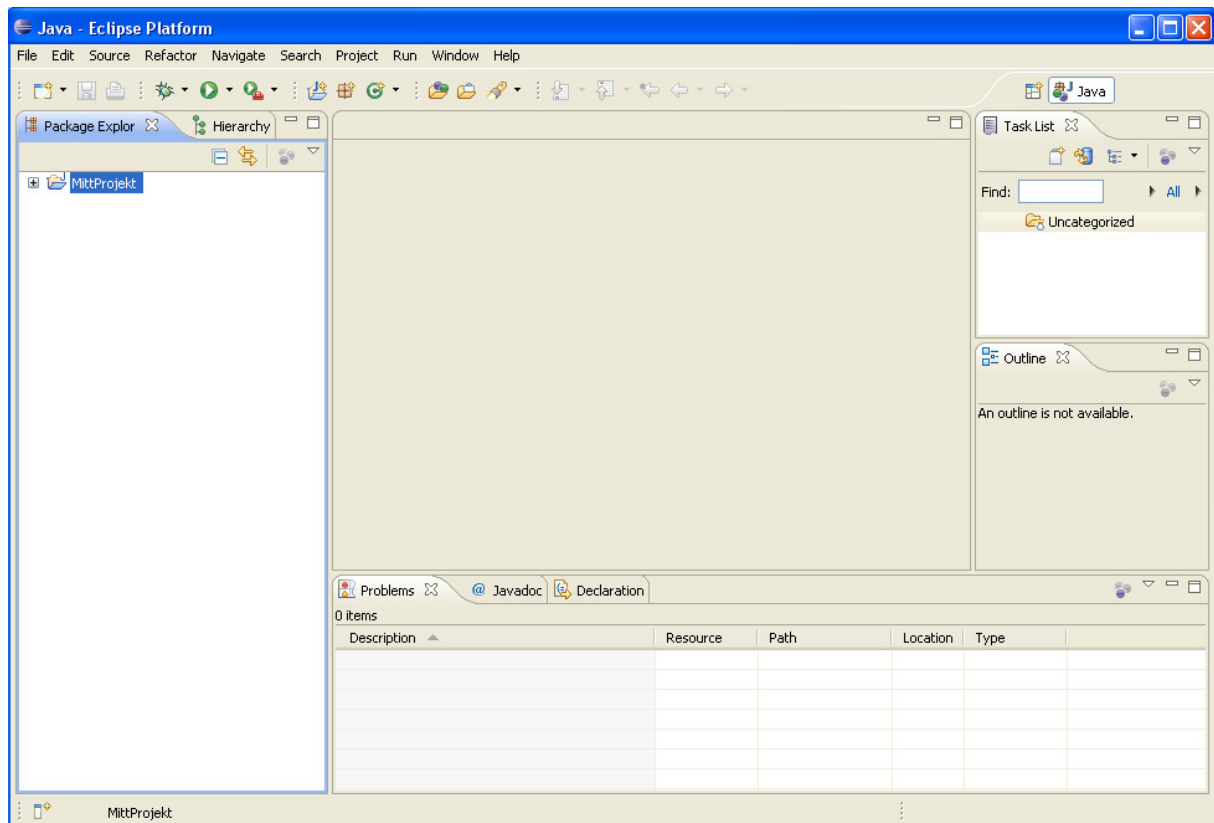
Klicka slutligen på **Next>**.

Nu visar sig ett nytt fönster – **Java Settings**. Du ska inte göra några nya inställningar i detta projektet utan klickar på **Finish** direkt.

Stäng **Welcome**-sidan genom att klicka på **Stäng**-krysset till höger om **Welcome** (om du inte redan gjort det).



Nu kommer fönstret att ungefär se ut så här:



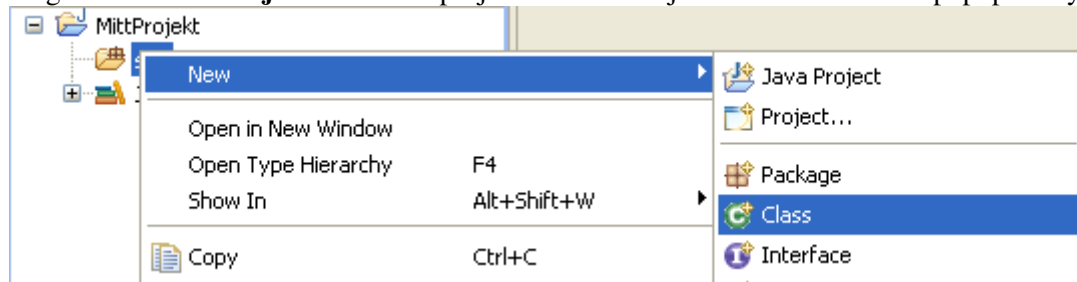
2. Skriva in källkod

Man skriver in källkoden med hjälp av ett speciellt program, en *editor*. En editor är som ett enkelt ordbehandlingsprogram. Men för att underlätta för programmeraren så innehåller ofta en editor speciella hjälpfunktioner.

Skapa källkodsfil i ett projekt

Eclipse innehåller bl.a. en editor. Nu ska du skaffa dig ett tomt dokument att skriva källkod i. Samtidigt ska du passa på att ge dokumentet ett bra namn.

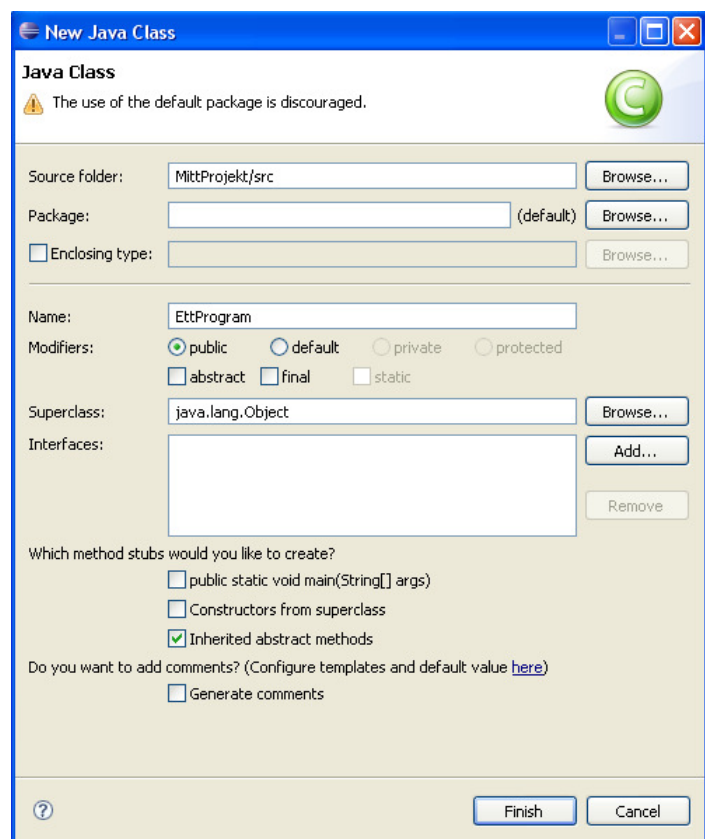
- Högerklicka **MittProjekt** eller **src** i projektfönstret. Välj sedan **New – Class** i popup-menyn.



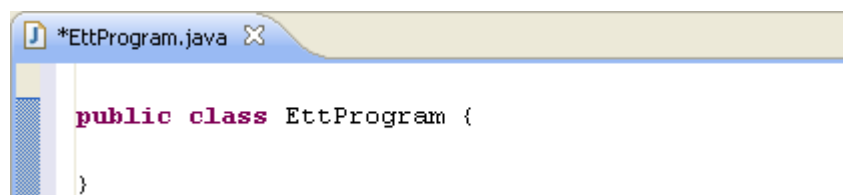
I dialogfönstret **Java Class** ska du ange lite information, bl.a

- om klassen ska vara i något paket. Lämna tomt denna gång.
- klassens namn (ska alltid börja med stor bokstav).

Ange att klassens namn ska vara **EttProgram** och klicka sedan på **Finish**.



Nu skaps en källkodsfil med skalet till klassen EttProgram.

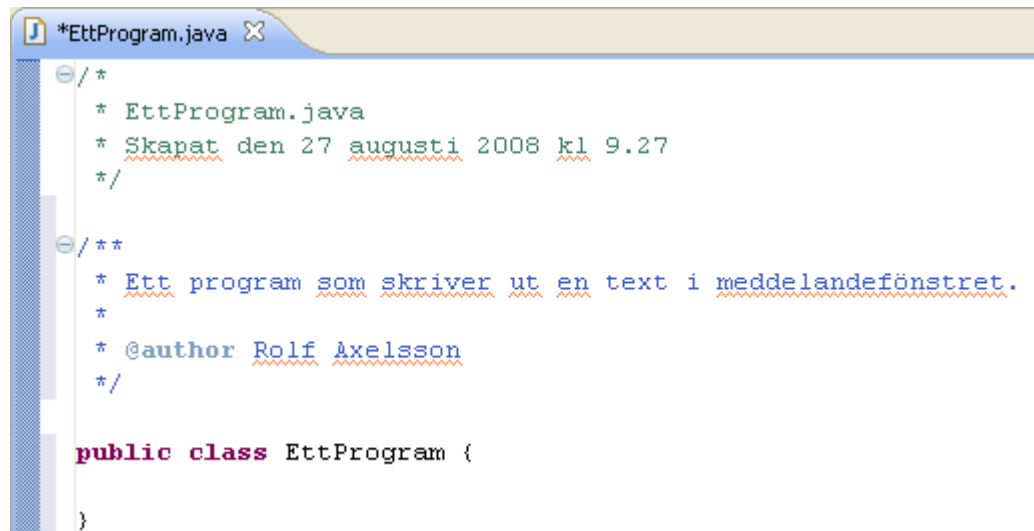


Om du tittar i projektkatalogen (C:\java\workspace\MittProjekt) så finner du bl.a. en src-katalog. I src-katalogen är filen EttProgram.java. EttProgram.java en vanlig textfil i vilken du kommer att lägga till kommentarer och kod.

Skriva in källkoden

Du ska göra vissa tillägg i EttProgram.java:

- Lägg till kommentarer ovanför klassen. Kommentarer är dokumentation av java-programmet till för dig själv och andra som jobbar med programmet. Kommentarer tillhör inte programmet och påverkar inte programkörningen.



```
/*
 * EttProgram.java
 * Skapat den 27 augusti 2008 kl 9.27
 */

/**
 * Ett program som skriver ut en text i meddelandefönstret.
 *
 * @author Rolf Axelsson
 */

public class EttProgram {
}
```

Den översta kommentaren är en s.k. *flerradskommentar*. En sådan kan du göra i Eclipse genom att skriva `/*` på en tom rad och sedan trycka på ENTER. Nu genereras automatiskt raderna

```
/*
 *
 */
```

och du kan börja skriva kommentaren. I Eclipse är flerradskommentarer gröna till färgen.

Flerradskommentaren

- börjar med `/*`
- avslutas med `*/`
- allt mellan startmarkeringen och slutmarkeringen är kommentarer.

Kommentaren direkt ovanför klassen är en *javadoc-kommentar*. En sådan kan du göra i Eclipse genom att skriva `/**` på en tom rad och sedan trycka på ENTER. Nu genereras automatiskt raderna

```
/**
 *
 */
```

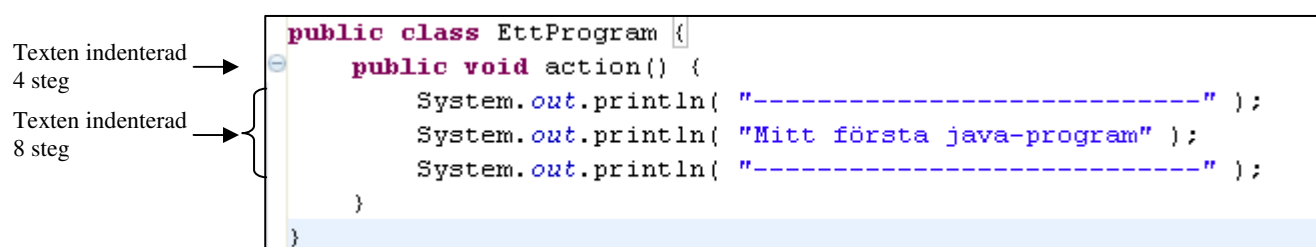
I Eclipse är javadoc-kommentarer blå till färgen. En javadoc-kommentar

- börjar med `/**`
- avslutas med `*/`
- allt mellan startmarkeringen och slutmarkeringen är kommentarer.

På kursen kommer du att stifta bekantskap med ytterligare ett sätt att skriva kommentarer, s.k. *enradskommentar*:

`// Enradskommentar`. Allt till höger om `//` är kommentar

- Lägg till metoden **action** så att klassen EttProgram ser ut så här:



```
public class EttProgram {
    public void action() {
        System.out.println( "-----" );
        System.out.println( "Mitt första java-program" );
        System.out.println( "-----" );
    }
}
```

Det är viktigt att göra indragningar i koden (inleda kodraden med ett antal mellanslag), sk indenteringar. Grundregeln för indentering är:

- Öka indenteringen efter {
- Minska indenteringen efter }

När du skriver in metoden `action` och instruktionerna som tillhör `action` (raderna som börjar med `System.`) så visar sig en röd markering till vänster om raden du just jobbar med. Denna röda markering är kvar så länge Eclipse inte tycker att det som står på raden går att översätta till bytekod.

Spara programmet efter att du ändrat i programmet. Du sparar programmet genom att klicka på disketten eller genom att välja **File – Save**. Om du tittar i `src`-katalogen i Projektmappen ser du filen **EttProgram.java**. Eclipse placerar källkodsfilerna som tillhör projektet i `src`-katalogen. Projekt-mappen hittar du i katalogen `C:\java\workspace`.

Ett par kommentarer till programmet:

- Klassens namn är alltid samma som filens namn (utom suffixet `.java`). I ovanstående exempel heter klassen *EttProgram* och filen *EttProgram.java*. Eftersom klassnamn alltid ska börja med stor bokstav så måste filens namn börja med stor bokstav.
- Raden med klassnamn inleds som regel med **public**.
- I det här programmet är det koden i metoden **action** som ska exekveras lite senare i denna instruktion.

3. Kompilera programmet

För att erhålla ett körbart program måste *källkoden* du skrivit översättas till *maskinkod*. Maskinkod är instruktioner som kan utföras av datorn. Att översätta källkoden till maskinkod kallas för att *kompilera* programmet. Det är ett speciellt program, en s.k. *kompilator*, som utför denna översättning.

I Java sker översättningen till maskinkod i två steg:

1. Först översätts källkoden till något som kallas *Java-bytekod*. Resultatet av denna översättning lagras i en class-fil. Programmet som utför denna översättning heter **javac.exe**. Program i java som är färdiga att exekveras (köras) lagras som en eller flera class-filer.
2. Sedan översätts bytekoden till maskinkod. Samtidigt körs maskinkoden. Översättningen av byte-koden och exekveringen av programmet sköts av programmet **java.exe** (kallas för JVM – Java Virtual Mashine).

När du arbetar med **Eclipse** så kompilerar du källkoden och exekverar Java-bytekoden i ett steg när du väljer att exekvera ett program (kommer du att göra i nästa avsnitt). Men varje gång du sparar en fil så kommer filen att kompileras. Om du tittar i projektkatalogen så hittar du katalogen **bin**. I katalogen bin finner du filen **EttProgram.class**, dvs. filen med bytekod.

För att källkoden skall kunna översättas till bytekod så måste den vara skriven enligt vissa mycket bestämda regler, den måste följa en bestämd *syntax*.

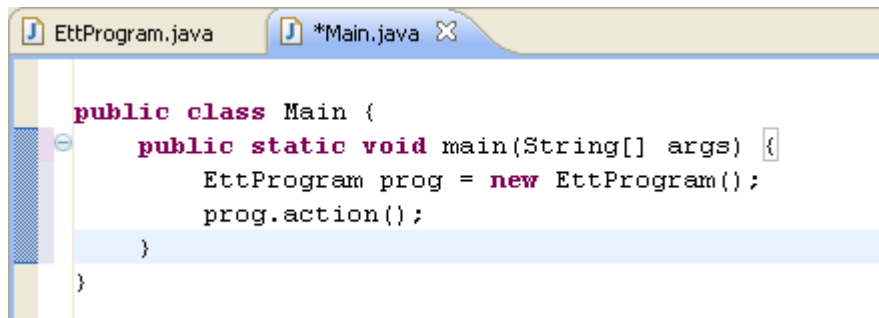
Editorn markerar med en röd symbol i kanten om den finner några felaktigheter i koden. Detta ger dig en möjlighet att rätta till felen innan du exekverar kompileringsprogrammet. *Kompilatorn* hittar de syntaxfel som inte rättats till och ger anvisning om var felet kan finnas. Du ser dessa anvisningar i meddelande-fönstret. Eftersom kompilatorn inte kan veta avsikten med programmet kan den inte heller veta exakt vad som är fel. Kompilatorn anger troligaste felorsaken.

Om du klickar på anvisningen i meddelandefönstret så markeras den delen på raden som innehåller felet. Rätta till felet och eventuella fler felaktigheter. Spara sedan programmet på nytt.

4. Köra programmet

För att köra programmet måste du skriva en **main**-metod. I java startar alltid ett program i en main-metod. Nedanstående main-metod kan placeras i vilken klass som helst (även i klassen EttProgram). Men för att programmet ska fungera måste klassen EttProgram är tillgänglig.

Skapa klassen **Main** (OBS! börjar på stor bokstav!) på samma sätt som du skapade EttProgram. Se till att Main-klassen har följande innehåll:



```
public class Main {  
    public static void main(String[] args) {  
        EttProgram prog = new EttProgram();  
        prog.action();  
    }  
}
```

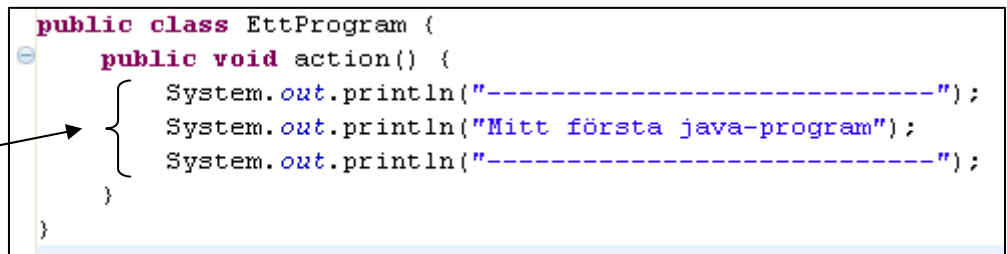
Kodraden

```
EttProgram prog = new EttProgram();
```

gör att koden du skrivit i klassen EttProgram kan användas. Raden

```
prog.action();
```

gör att koden i metoden action exekveras.



```
public class EttProgram {  
    public void action() {  
        System.out.println("-----");  
        System.out.println("Mitt första java-program");  
        System.out.println("-----");  
    }  
}
```

När programmet är utan syntax-fel (röda markeringar) är det dags att köra programmet.

- Högerklicka på **Main.java** i projekt-fönstret och välj **Run As – 1 Java Application**. Om du redan har sparat Main.java respektive EttProgram.java så startar exekveringen direkt, annars får du en fråga om filerna ska sparas. Klicka på OK.

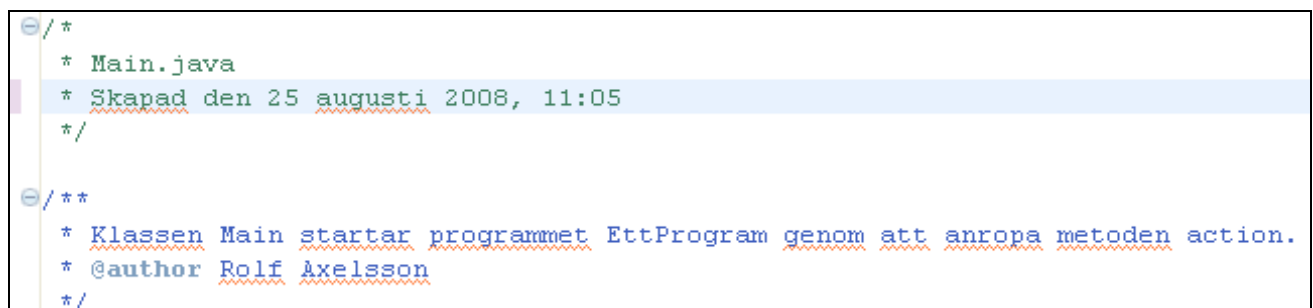
I java körs byte-koden med hjälp av ett program, en s.k. "Java Virtual Mashine". Byte-koden finns i class-filerna **EttProgram.class** och **Main.class**.

Resultatet av körningen är en utskrift i meddelandefönstret.



```
<terminated> Main [Java Application] C:\Program\Java\jre6\bin\javaw.exe (29 aug 2008 10.54.25)  
  
-----  
Mitt första java-program...  
-----
```

Passa på att skriva in vettiga kommentarer överst i Main.java. De kan t.ex. se ut så här:



```
/*  
 * Main.java  
 * Skapad den 25 augusti 2008, 11:05  
 */  
  
/**  
 * Klassen Main startar programmet EttProgram genom att anropa metoden action.  
 * @author Rolf Axelsson  
 */
```

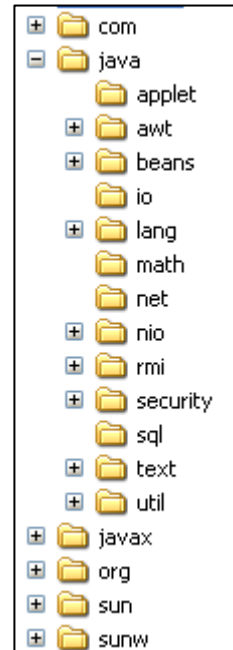
Paket

I programmeringsspråket java finns det ett stort antal färdigskrivna klasser som programmerare kan använda sig av. Dessa klasser lagras i en katalogstruktur där varje katalog kallas för ett paket.

I figuren till höger ser du huvudpaketerna, bl.a *java* och *javax*. Dessa paket innehåller i sin tur paket, s.k. *underpaket*.



- Paketet *java* är öppnat i figuren till höger. Du ser att *java* innehåller ett antal paket. Dessa är: *applet*, *awt*, *beans*, *io*, *lang*, *math*, *net*, *nio*, *rmi*, *security*, *sql*, *text*, *util*. Dessa paket innehåller i sin tur klasser och ofta även underpaket. De kursiverade paketen innehåller klasser som kommer att användas på kursen. Paketerna namn uttrycker hela katalogstrukturen. T.ex. heter ovanstående underpaket: *java.applet*, *java.awt* osv
- Paketet *javax* innehåller också ett antal underpaket. Ett viktigt sådant på kursen är *swing*. Klassen *JOptionPane*, vilken finns i *javax.swing*, ska du använda i exemplet nedan.



Importera paket

Ett speciellt viktigt paket i java är *java.lang*. Alla klasser som finns i paketet *java.lang* kan användas direkt i java-program. Och endast dessa klasser. Vill man använda en klass som finns i ett annat paket (det vill man) så måste man ange detta i sitt program – man måste *importera* klassen.

Med raden

```
import javax.swing.JOptionPane;
```

anger man att klassen *JOptionPane* i paketet *javax.swing* ska användas i programmet (egentligen i den klass man arbetar med).

Nu ska du ändra i programmet för att testa att mata in tecken via dialog-fönster.

Gör de ändringar som du ser i figurerna nedan. Efter figurerna förklaras de kortfattat.

```
*EttProgram.java X
/*
 * EttProgram.java
 * Skapat den 27 augusti 2008 kl 9.27
 */
import javax.swing.JOptionPane;
/**
 * Ett program som skriver ut en text i meddelandefönstret.
 * @author Rolf Axelsson
 */
public class EttProgram {
    public void action() {
        String name = JOptionPane.showInputDialog( "Ånge ditt namn" );
        System.out.println("-----");
        System.out.println("Ett program skrivet av " + name );
        System.out.println("-----");
    }
}
```



```
EttProgram.java Main.java X
/*
 * Main.java
 * Skapat den 25 augusti 2008, 11:05
 */

/**
 * Klassen Main startar programmet EttProgram genom att anropa metoden action.
 * @author Rolf Axelsson
 */
public class Main {
    public static void main(String[] args) {
        EttProgram prog = new EttProgram();
        prog.action();
        javax.swing.JOptionPane.showMessageDialog( null, "Nu är programmet slut!" );
    }
}
```

För att kompilatorn skall hitta metoden **JOptionPane.showInputDialog(String)** så måste du tala om för kompilatorn var klassen finns. Ett sätt att göra detta är att skriva

```
import javax.swing.JOptionPane;
```

Denna rad ska stå ovanför klassen (se EttProgram).

Det går även bra att importera samtliga klasser som finns i ett paket:

```
import javax.swing.*;
```

Om du skriver på detta sätt kan alla klasser i *javax.swing*-paketet användas i programmet. Du kan testa genom att byta `JOptionPane` mot `*` i `import`-raden i `EttProgram`.

Det går bra att använda en klass även om man inte importerat klassen / paketet med klassen. Men man måste i detta fallet ange klassens fullständiga namn, dvs. även det paket som klassen är placerad i. Detta ges exempel på i klassen `Main`:

```
javax.swing.JOptionPane.showMessageDialog( null, "Nu är ..." );
```

Flera main-metoder

Slutligen kan du kopiera `main`-metoden i klassen `Main` och klistra in den sist i klassen `EttProgram`. Sedan kan du högerklicka `EttProgram` och välja `Run File`. `EttProgram` kompileras först och sedan startar programmet i `main`-metoden i `EttProgram`.

Detta visar att `main`-metoden kan placeras i vilken klass som helst.

```
EttProgram.java X Main.java
/*
 * EttProgram.java
 * Skapat den 27 augusti 2008 kl 9.27
 */
import javax.swing.JOptionPane;

/**
 * Ett program som skriver ut en text i meddelandefönstret.
 * @author Rolf Axelsson
 */
public class EttProgram {
    public void action() {
        String name = JOptionPane.showInputDialog( "Ånge ditt namn" );
        System.out.println("-----");
        System.out.println("Ett program skrivet av " + name );
        System.out.println("-----");
    }

    public static void main(String[] args) {
        EttProgram prog = new EttProgram();
        prog.action();
        JOptionPane.showMessageDialog( null, "Nu är programmet slut!" );
    }
}
```