Επισκόπηση των Τεχνολογιών της Java

Κακαρόντζας Γεώργιος
Καθηγητής Εφαρμογών
Τμήμα Τεχνολογίας Πληροφορικής & Τηλεπικοινωνιών
ΤΕΙ Λάρισας
gkakaron@teilar.gr

Γιατί γεννήθηκε η Java

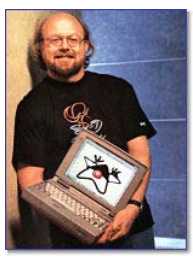
- Αρχικά ο στόχος ήταν ενσωματωμένο λογισμικό συσκευών (embedded software). Αφού διαπιστώθηκε πως δεν υπήρχε αγορά, ο στόχος άλλαξε: το Internet.
- Το Internet έχει ιδιάζουσες απαιτήσεις ασφάλειας και ετερογένειας.
- Η λύση της Java ήταν η Εικονική Μηχανή Java (Java Virtual Machine – JVM).
- Η JVM παρέχει:
  - Ασφάλεια: μπορεί να εφαρμόζει πολιτικές ασφαλείας για τον κώδικα που εκτελείται.
  - Ετερογένεια: Αυτό που αλλάζει είναι η JVM και όχι η φυσική μηχανή.

Σκοπός της σημερινής διάλεξης

- Δεν είναι η εκμάθηση της Java!
- Αφού δούμε κάποια εισαγωγικά στοιχεία θα προχωρήσουμε στην επισκόπηση των βασικών τεχνολογιών που παρέχονται από την Java, για διάφορα είδη εφαρμογών:
  - Απλές εφαρμογές Java
  - Μικρο-εφαρμογές (applets)
  - Εφαρμογές Διαδικτύου (web applications)
  - Πρόσβαση σε Βάσεις Δεδομένων μέσω JDBC

Κάποια ιστορικά στοιχεία

- Στις 23 Μαΐου 1995 ο John Gage διευθυντής του Γραφείου Επιστημών της εταιρίας Sun Microsystems και ο Marc Andreessen, συν-ιδρυτής και αντιπρόεδρος της Netscape, ανακοίνωσαν στο SunWorld της διαθεσιμότητα της Java και την ενσωμάτωσή της στον Netscape Navigator.
- Τότε η ομάδα της Java στην Sun αριθμούσε λιγότερους από 30 ανθρώπους.Δεξιά ο άνθρωπος που θεωρείται ο «πατέρας» της Java, ο κ. James Gosling.



1ο Μέρος: Απλές Εφαρμογές Java

HelloWorld στην Java

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

- Θα πρέπει να αποθηκεύσουμε το αρχείο με το όνομα HelloWorld.java να το μεταγλωττίσουμε σε bytecodes δίνοντας javac HelloWorld.java και αν δεν έχει λάθη να το εκτελέσουμε πληκτρολογώντας java HelloWorld

Τι είναι μία εφαρμογή Java

- Μία Java εφαρμογή είναι ένα πρόγραμμα σχεδιασμένο για να εκτελεστεί από μία JVM στη μηχανή του χρήστη και όχι από ένα πρόγραμμα πλοήγησης.
- Μία Java εφαρμογή θα πρέπει να έχει και μία συνάρτηση main (κύρια) που είναι και το σημείο εισόδου στην εφαρμογή (δηλαδή από εκεί ξεκινά το πρόγραμμα).
- Φυσικά μια συνάρτηση main θα πρέπει να βρίσκεται μέσα σε μία κλάση (class).
  - Η Java είναι μία αντικειμενοστραφής γλώσσα προγραμματισμού και επομένως δεν μπορούμε να έχουμε καμία συνάρτηση που δεν είναι τμήμα μιας κλάσης.

Παραθυρικές Εφαρμογές Java

- Πλούσια βιβλιοθήκη κλάσεων για παραθυρικές εφαρμογές.
  - Για την ακρίβεια δύο βιβλιοθήκες: AWT και Swing
- Η εφαρμογή έχει ένα βασικό παράθυρο το οποίο περιέχει συστατικά. Ο χρήστης αλληλεπιδρά με το περιβάλλον με οικείους τρόπους αλληλεπίδρασης με γραφικά περιβάλλοντα.
- Τυπικά σχεδιάζουμε την γραφική διασύνδεση βοηθούμενοι από κατάλληλα προγράμματα, όπως το NetBeans (www.netbeans.org).

Παραθυρικό HelloWorld (1)

```
public class MainFrame extends
   javax.swing.JFrame {
  public MainFrame() { initComponents(); }
  private void initComponents() { …
        sayHelloButton.addActionListener(new
  java.awt.event.ActionListener() {
          public void
  actionPerformed(java.awt.event.ActionEvent
  evt) {
            sayHelloButtonActionPerformed(evt);
          }
      });
  …}
```

Παραθυρικό HelloWorld (2)

```
private void clearButtonActionPerformed(
          java.awt.event.ActionEvent evt) {
      helloLabel.setText("");
   }
   private void
   sayHelloButtonActionPerformed(
          java.awt.event.ActionEvent evt) {
      helloLabel.setText("Γειά σου Κόσμε!");
   }
   private javax.swing.JButton clearButton;
   private javax.swing.JLabel helloLabel;
   private javax.swing.JButton sayHelloButton;
}
```

Παραθυρικό HelloWorld (3)

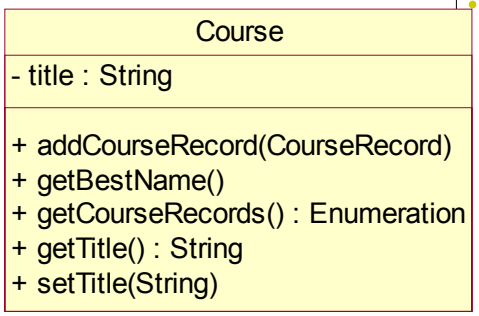
```
public class Main {
public static void main(
   String[] args) {
      MainFrame frame =
          new MainFrame();
      frame.setTitle(
          "Παραθυρικό Hello World");
      frame.setVisible(true);
   }
}
```

Το αποτέλεσμα της εκτέλεσης του παραθυρικού προγράμματος

Η Java είναι αντικειμενοστραφής

- Κάθε πρόγραμμα Java αποτελείται από αντικείμενα που υλοποιούν την λογική της εφαρμογής.
- Το πρόγραμμα αποτελείται από ένα σύνολο αρχείων κλάσεων που τυπικά παρέχουν μία δημόσια διασύνδεση και μια ιδιωτική υλοποίηση. Οι κλάσεις αποτελούν το «καλούπι» από το οποίο δημιουργούνται αντικείμενα κατά την εκτέλεση του προγράμματος.
- Οι κλάσεις οργανώνονται σε πακέτα τα οποία παρέχουν:
  - Καλύτερο έλεγχο εξαρτήσεων σε μεγάλα συστήματα.
  - Ένα χώρο ονομάτων.

Η κλάση Course σε UML

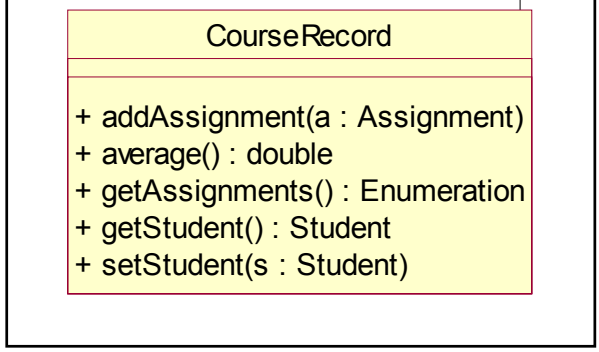


Παράδειγμα με υλοποίηση UML διαγραμμάτων στη Java

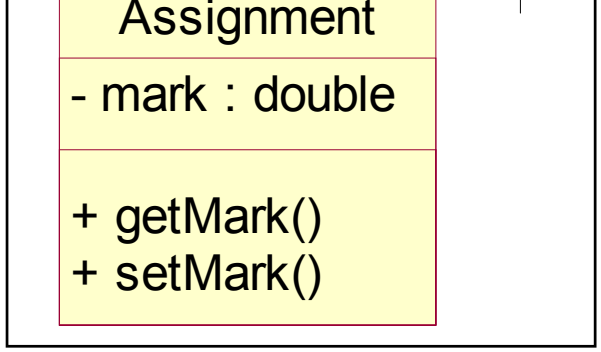
- Έστω ένα μάθημα (Course) το οποίο συσχετίζεται με έναν αριθμό εγγραφών (CourseRecord).
- Κάθε εγγραφή συσχετίζεται με έναν φοιτητή (Student) και ένα πλήθος εργασιών (Assignment).
- Θέλουμε να βρούμε το όνομα του καλύτερου φοιτητή με βάση τον μέσο βαθμό των εργασιών του.

```
import java.util.*;
public class Course {
   private String title;
    private Vector courseRecords;
   Course() {courseRecords=new Vector(); }
   public String getTitle() {return title;}
   public void setTitle( String t) {title=t;}
   public void addCourseRecord(
                              CourseRecord cr) {
      courseRecords.addElement(cr);     }
   public Enumeration getCourseRecords() {
      return courseRecords.elements();     }
   public void printBestStudent() { … }
}
```

Η κλάση CourseRecord σε UML



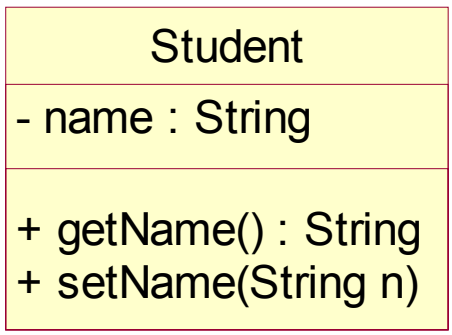
Η κλάση Assignment σε UML



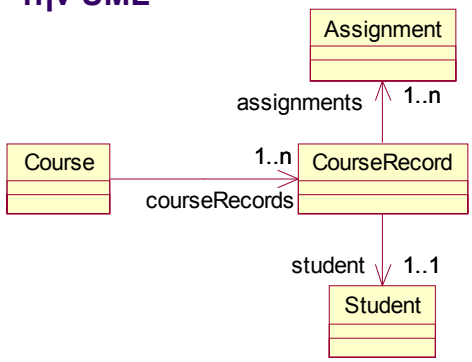

```
import java.util.*;
public class CourseRecord {
   private Student student;
   private Vector assignments;
   CourseRecord() {assignments=new Vector();}
   public Student getStudent() {return student;}
   public void setStudent(Student s) {student = s;}
  public void addAssignment(Assignment a)
  {assignments.addElement(a);}
   public Enumeration getAssignments()
  {return assignments.elements();}
   public double average() { … }
}
```

```
public class Assignment {
   private double mark=-1;
   Assignment() {}
   public double getMark() throws
  NotYetSetException {
      if (mark==-1) {
        throw new NotYetSetException(
                        "Δεν έχει τεθεί βαθμός");
     }
     return mark;
   }
  public void setMark( int m)  {   mark=m;   }
}
```

Η κλάση Student σε UML

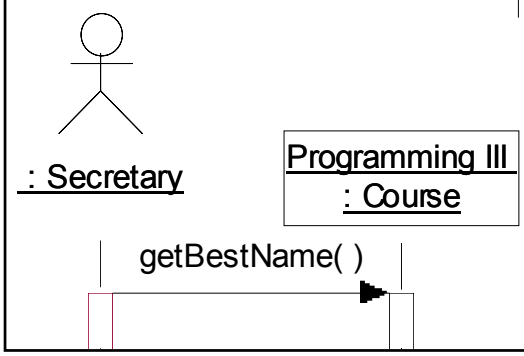


Αποτύπωση συσχετίσεων με την UML

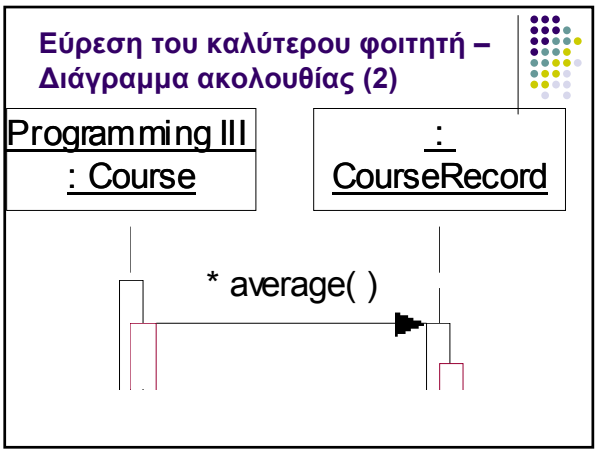



```
public class Student {
   private String name;
   Student() { }
   public String getName() {
      return name;
   }
  public void setName( String n) {
      name=n;
   }
}
```

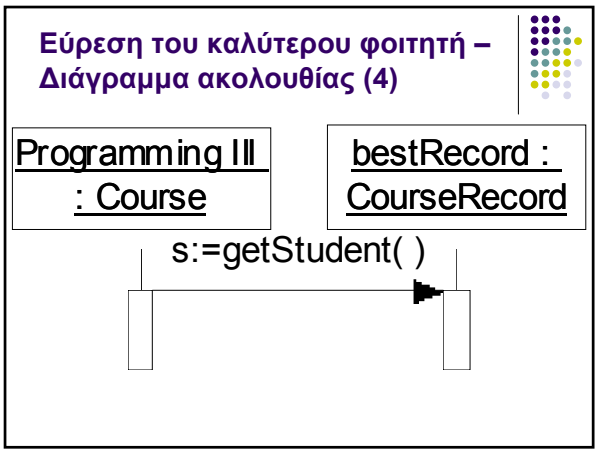
Εύρεση του καλύτερου φοιτητή – Διάγραμμα ακολουθίας (1)



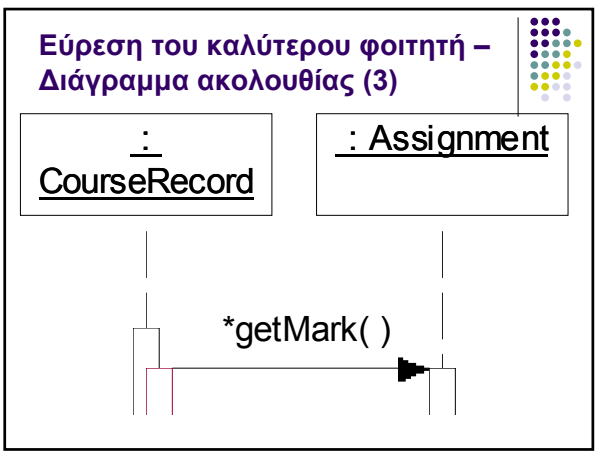
Εύρεση του καλύτερου φοιτητή – Διάγραμμα ακολουθίας (2)



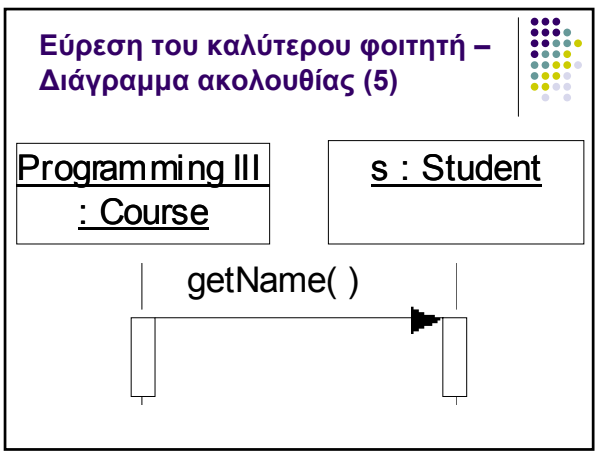
Εύρεση του καλύτερου φοιτητή – Διάγραμμα ακολουθίας (4)



Εύρεση του καλύτερου φοιτητή – Διάγραμμα ακολουθίας (3)



Εύρεση του καλύτερου φοιτητή – Διάγραμμα ακολουθίας (5)



```
public void printBestStudent() {
      Student bestStudent=null;
      Enumeration e = this.getCourseRecords();
      double bestMark=0.0;
      while (e.hasMoreElements()) {
         CourseRecord cr = (CourseRecord)
                                  e.nextElement();
         double avg = cr.average();
         if (avg>bestMark) {
            bestMark=avg;
            bestStudent=cr.getStudent();
         }
      }
      System.out.println("Best Student: "
                              +bestStudent.getName());
  }
```

Τι άλλο παρέχεται με την Java

- Αρχεία και Ρεύματα εισόδου/εξόδου (java.io)
- Συλλογές Αντικειμένων (Java Collection Framework)
- Δικτυακός προγραμματισμός χαμηλού επιπέδου (java.net)
- Επικοινωνία με βάσεις δεδομένων μέσω JDBC (java.sql)
- Απομακρυσμένες κλήσεις αντικειμένων μέσω RMI (java.rmi)
- Εφαρμογές γραφικών (java.awt, javax.media.j3d, javax.vecmath)
- και πολλά άλλα!

```
public double average() {
      double  sum=0.0;
       int count = 0;
       Enumeration e = this.getAssignments();
       while (e.hasMoreElements()) {
          Assignment a =
                (Assignment) e.nextElement();
          try {
             sum += a.getMark();
             count++;
          }
          catch (NotYetSetException ignore) {}
       }
        return (sum/count);
    }
```


2ο Μέρος: Μικροεφαρμογές (Java Applets)

Εισαγωγή

- Ένα Applet είναι ένα πρόγραμμα Java το οποίο εκτελείται σε μία ιστοσελίδα. Τα βήματα που πρέπει να κάνετε για να δημιουργήσετε ένα Java Applet είναι τα ακόλουθα:
  - Πρώτα γράφετε το Applet ακριβώς όπως γράφετε και κάθε άλλο πρόγραμμα Java.
  - Στη συνέχεια μεταφράζετε το Applet για να παράγετε το αρχείο με επέκταση *class* με τα bytecodes
  - Τέλος, ενσωματώνετε το Applet σε μία ιστοσελίδα χρησιμοποιώντας (κατ' ελάχιστο) την ετικέτα <Applet> η οποία έχει την εξής (ελάχιστη) μορφή:

```
<applet code = όνομα-αρχείου.class
width=πλάτος height=ύψος></applet>
```

Οι παράμετροι της ετικέτας applet

- Η ετικέτα applet έχει τις υποχρεωτικές φράσεις *code*, *width* και *height*.
  - Το *code* προσδιορίζει το αρχείο που θα εκτελεστεί όταν θα φορτωθεί η ιστοσελίδα που περιέχει την ετικέτα σε ένα browser.
  - Το *width* προσδιορίζει το πλάτος που θα καταλάβει το applet σε εικονοστοιχεία (pixels) στην περιοχή του browser και το *height* είναι το ύψος που θα καταλάβει το applet σε pixels στην περιοχή του browser.
  - Το </applet> τερματίζει την ετικέτα Applet.
- Το Applet θα καταλάβει το χώρο αυτό στο σημείο που βρίσκεται στην ιστοσελίδα και εκτελείται όταν φορτώνεται η ιστοσελίδα που το περιέχει σε κάποιο browser.

Πως γράφουμε μία μικροεφαρμογή Java

- Ένα πρόγραμμα Java το οποίο θα χρησιμοποιηθεί σαν Applet θα πρέπει να επεκτείνει (extend) την τάξη java.applet.Applet. Έτσι, συνήθως, κάνουμε import το πακέτο java.applet όπως και το πακέτο java.awt σε κάθε Applet.
- Ένα Applet έχει λοιπόν, συνήθως, την ακόλουθη γενική μορφή:

```
import java.awt.*;
import java.applet.*;
public class όνομα-τάξης-Applet
  extends Applet { ...}
```

Παράδειγμα μικροεφαρμογής

```
import java.applet.*;
import java.awt.*;
public class Program1 extends Applet
{
public void paint(Graphics g)
  {
    g.drawString(
    "Ένα πρώτο παράδειγμα χρήσης Applet",
       10, 20);
  }
}
```

HTML σελίδα

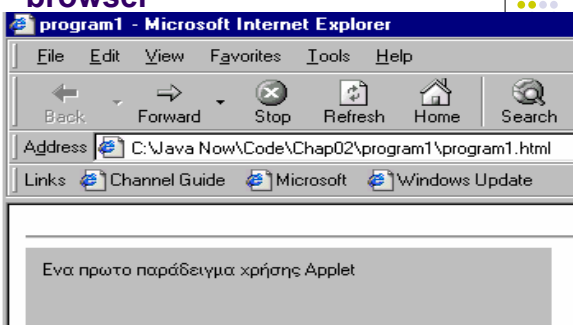
- Το HTML αρχείο που περιέχει το Applet είναι το ακόλουθο:

```
<html>
<head><title>program1</title></head>
<body>
<applet code=Program1.class width=320
  height=240 >
</applet>
</body>
</html>
```

Μέθοδοι-σταθμοί για μία μικροεφαρμογή

- `init`: Αρχικοποιήσεις όταν φορτωθεί το Applet για πρώτη φορά.
- `destroy`: Ενδεχόμενες τελικές ενέργειες που μπορεί να θέλουμε να συμβούν λίγο πριν σταματήσει το Applet να εκτελείται.
- `start`: Δυνατότητα να ξεκινήσουμε κάποιες επεξεργασίες αφού φορτωθεί το Applet.
- `stop`: Καλείται από το σύστημα εκτέλεσης κάθε φορά που ο χρήστης φεύγει από τη σελίδα που περιέχει το Applet είτε για να πάει σε κάποια άλλη σελίδα.

Εκτέλεση του Applet στον browser



Περιορισμοί στις μικροεφαρμογές

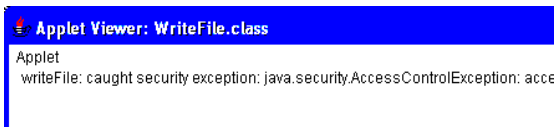
- Ένα applet θεωρείται εξ ορισμού αναξιόπιστο και δεν έχει δικαίωμα:
  - Να γράψει ή να διαβάσει το σύστημα αρχείων του υπολογιστή στον οποίο εκτελείται.
  - Να ανοίξει συνδέσεις δικτύου με άλλους υπολογιστές πέρα απ' αυτόν από τον οποίο κατέβηκε.
  - Να ξεκινήσει την εκτέλεση άλλων προγραμμάτων.
- Οι περιορισμοί αυτοί μπορούν να αρθούν με την σαφή τροποποίηση της εξ ορισμού πολιτικής από τον χρήστη του υπολογιστή στον οποίο εκτελείται το applet.

Παράδειγμα Applet που δημιουργεί αρχείο στο δίσκο (1)

```
import java.awt.*; import java.io.*;
 import java.lang.*; import java.applet.*;
public class WriteFile extends Applet {
  String myFile = "writetest";
  File f = new File(myFile);
  DataOutputStream dos;
public void paint(Graphics g) {
  try {
    dos = new DataOutputStream(new
     BufferedOutputStream(
           new FileOutputStream(myFile),128));
    dos.writeChars("Στοιχεία στον δίσκο\n");
  …
```

Πρόκληση εξαίρεσης από την εκτέλεση της μικροεφαρμογής

- Η εκτέλεση του applet με την εξ ορισμού πολιτική ασφαλείας θα προκαλέσει εξαίρεση ασφάλειας (java.security.AccessControlException) όταν το applet θα επιχειρήσει να δημιουργήσει το αρχείο στον δίσκο.



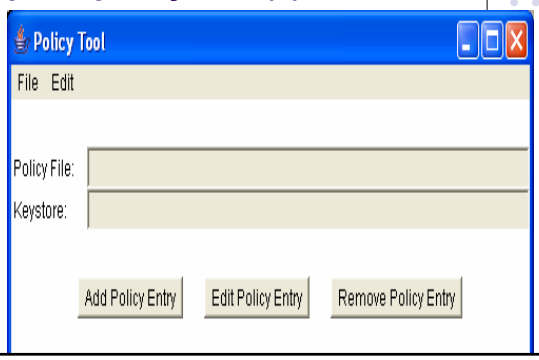
Παράδειγμα Applet που δημιουργεί αρχείο στο δίσκο ()

```
  dos.flush();
    g.drawString(
  "Successfully wrote to the file named " +
  myFile +" -- go take a look at it!", 10, 10);
  }
  catch (SecurityException e) {
    g.drawString(
  "writeFile: caught security exception: " + e,
  10, 10); }
  catch (IOException ioe) {
       g.drawString(
       "writeFile: caught i/o exception", 10, 10); }
  }
}
```

Ρύθμιση πολιτικής ασφαλείας με το policy tool (1)

- Ένα αρχείο ασφάλειας μπορεί να γραφεί με έναν εκδότη κειμένου ή (προτιμότερο) να κατασκευαστεί με το policy tool.
- Για να ξεκινήσετε το policy tool, δώστε την εντολή policytool από την γραμμή εντολών.

Ρύθμιση πολιτικής ασφαλείας με το policy tool (2)



Επιτυχής εκτέλεση της μικροεφαρμογής

- Με την πολιτική που αναπτύξαμε η εκτέλεση του applet επιτρέπεται πλέον να γράψει στο συγκεκριμένο αρχείο στο δίσκο, όπως φαίνεται και στην εικόνα.

Δυνατότητες του policy tool

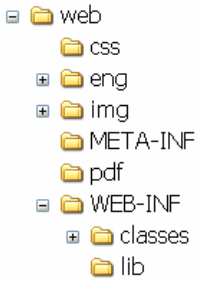
- Μπορείτε να δημιουργήσετε διάφορες πολιτικές ασφαλείας για τις κλάσεις που «κατεβαίνουν» από έναν δικτυακό τόπο και να καθορίσετε τι επιτρέπεται να κάνουν και τι όχι.
- Από την Java 1.2 και μετά αυτό ισχύει και για τις εφαρμογές Java με την διαφορά πως οι εφαρμογές θεωρούνται εξ ορισμού αξιόπιστες.
- Με την ακόλουθη πολιτική ασφάλειας το προηγούμενο applet έχει το δικαίωμα εγγραφής στο αρχείο writetest.

```
grant codeBase "file:///c:/javasec/" {
  permission java.io.FilePermission "writetest",
    "write";
};
```


3ο Μέρος: Εφαρμογές Διαδικτύου (Web Applications)

Δομή εφαρμογών διαδικτύου με Java

- Ο βασικός κατάλογος περιέχει:
  - Ιστοσελίδες, εικόνες, Java Server Pages (JSP) κλπ. Αυτά μπορεί να είναι οργανωμένα σε καταλόγους.
  - Ο κατάλογος WEB-INF περιέχει δύο υποκαταλόγοι & ένα σημαντικό αρχείο
    - classes: servlets
    - lib: βιβλιοθήκες κλάσεων
    - Το αρχείο περιγραφής της εφαρμογής: web.xml



HelloWorld servlet

```
import java.io.*;import javax.servlet.*;
  import javax.servlet.http.*;
public class HelloWorld extends HttpServlet {
   public void doGet(
     HttpServletRequest request,
              HttpServletResponse response)
       throws IOException, ServletException    {
          response.setContentType("text/html");
          PrintWriter out = response.getWriter();
        out.println(
  "<html><head><title>HelloWorld</title></head>");
            out.println("<body>");
            out.println("<h1>Hello World</h1>");
            out.println("</body></html>");
} }
```

Τι είναι τα Java Servlets

- Τα Java Servlets είναι μία web τεχνολογία βασισμένη στη Java με την οποία μπορούμε να αναπτύξουμε εφαρμογές εξυπηρετητή για το Internet.
- Ένα servlet είναι μία κλάση της Java που περιέχεται σε ένα container. Το container διαχειρίζεται το servlet. Το container μπορεί να είναι ταυτόχρονα και web server ή να είναι ένα πρόσθετο συστατικό για ένα web server χωρίς δυνατότητες εκτέλεσης servlet.
  - Για παράδειγμα ο Tomcat, είναι container και ταυτόχρονα web server. Παρόλα αυτά, υπάρχει η δυνατότητα να χρησιμοποιηθεί ο Tomcat μαζί με τον Apache ή τον IIS οι οποίοι δεν μπορούν να εκτελέσουν servlets.

Εκτέλεση του servlet

Πέρασμα παραμέτρων σε servlet

- Για το πέρασμα παραμέτρων συνήθως χρησιμοποιείται μία HTML σελίδα με μία φόρμα στην οποία το πεδίο action προσδιορίζεται σε κάποιο συγκεκριμένο servlet ως χειριστή της αίτησης.
- Στα διάφορα πεδία της φόρμας αποδίδονται ονόματα. Τα ονόματα χρησιμοποιούνται από το servlet για την ανάκτηση των τιμών που έδωσε ο χρήστης.
- Το πέρασμα των παραμέτρων γίνεται είτε με την μέθοδο get (προσαρτημένα στο URL αίτησης), είτε με την μέθοδο post (προσαρτημένα στην HTTP αίτηση).

Ανάκτηση παραμέτρων και σχηματισμός της απόκρισης

```
public void doPost(HttpServletRequest
  request, HttpServletResponse response)
  throws ServletException, IOException    {
      request.setCharacterEncoding(
            "ISO-8859-7");
      response.setContentType(
         "text/html; charset=ISO-8859-7");
      String s = request.getParameter("name");
      PrintWriter out = response.getWriter();
      out.println("Hello to you "+s);
}
```

HTML σελίδα με φόρμα που περνά στοιχεία σε servlet

```
<html><head>
<title>Hello To You</title>
</head>
<body>
  <p>Hello με φόρμα</p>
  <form method="post"  action="HelloToYou">
     <p>Πως σε λένε;
        <input type="text" name="name">
        <input type="submit">
</form>
</body>
</html>
```

Java Server Pages (JSP)

- Μία συμπληρωματική τεχνολογία για την κατασκευή διαδικτυακών εφαρμογών με την Java είναι οι Σελίδες Εξυπηρετητή Java (Java Server Pages ή JSP). Οι JSP επιτρέπουν την εύκολη μίξη κώδικα HTML και κώδικα Java στην ίδια σελίδα.
- Όπως είδαμε με τα Java Servlets ο χρήστης θα πρέπει να γράψει αρκετές εντολές *out.print* για να σχηματίσει την απόκριση του servlet στον χρήστη (δηλ. το αποτέλεσμα). Αυτό είναι κουραστικό όταν η απόκριση είναι πολύπλοκη (π.χ. έχει πίνακες ή άλλα σύνθετα HTML στοιχεία). Είναι πιο βολικό η απόκριση να σχηματίζεται απευθείας γράφοντας HTML και όπου χρειάζεται να χρησιμοποιείται κώδικας Java. Αυτή ακριβώς την ανάγκη έρχονται να καλύψουν οι JSP σελίδες.

Ένα παράδειγμα JSP (1)

```
<%@ page import="java.util.Date,
   java.text.DateFormat" %>
<%@ page contentType=
        "text/html; charset=iso-8859-7" %>
<HTML>
<HEAD>
<TITLE>Η πρώτη μου JSP σελίδα</TITLE>
</HEAD>
<BODY>
…
```

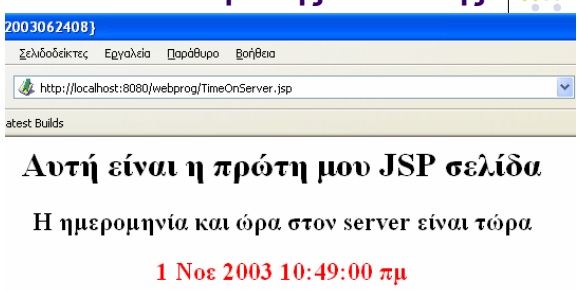
Ένα παράδειγμα JSP (2)

```
<%
   Date today; String dateOut;
   DateFormat dateFormatter;
   dateFormatter =
   DateFormat.getDateTimeInstance(
      DateFormat.DEFAULT,
         DateFormat.DEFAULT);
   today = new Date();
   dateOut = dateFormatter.format(today);
%>
```

Ένα παράδειγμα JSP (3)

```
<H1 ALIGN="CENTER">
   Αυτή είναι η πρώτη μου JSP
   σελίδα</H1>
<H2 ALIGN="CENTER">Η
   ημερομηνία και ώρα στον server
   είναι τώρα</H2>
<H2 ALIGN="CENTER">
   <FONT COLOR="RED">
   <%= dateOut %></FONT></H2>
</BODY></HTML>
```

Το αποτέλεσμα της εκτέλεσης

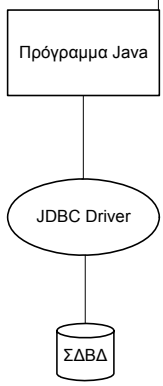


Παραγωγή servlet από JSP σελίδα

- Τη πρώτη φορά που οι χρήστες θα ζητήσουν την JSP σελίδα ο εξυπηρετητής θα παράγει το αντίστοιχο servlet το οποίο και θα εκτελείται κάθε φορά που οι χρήστες ζητάνε τη σελίδα.
- Το servlet θα μεταγλωττιστεί την πρώτη φορά για να παραχθεί το στιγμιότυπο που θα εξυπηρετήσει τις αιτήσεις των πελατών.

Τι είναι το JDBC

- Πρόκειται για ένα API το οποίο μπορεί να χρησιμοποιηθεί από εφαρμογές Java για πρόσβαση σε σχεσιακές Βάσεις Δεδομένων.
- Η πρόσβαση επιτυγχάνεται μέσω ενός JDBC Driver οποίος είναι εξειδικευμένος για κάθε Σύστημα Διαχείρισης Βάσεων Δεδομένων.



4ο Μέρος: Πρόσβαση σε ΒΔ μέσω JDBC

Τύποι JDBC Driver (1)

- Υπάρχουν διάφοροι τύποι JDBC Drivers (1, 2, 3 ή 4).
  - Ο τύπος 1 είναι μία γέφυρα (bridge) για την επικοινωνία του JDBC με το ODBC. Για παράδειγμα το λεγόμενο JDBC-ODBC bridge, είναι ένας Driver που παρέχεται με την Java και παρέχει την δυνατότητα πρόσβασης σε πηγές δεδομένων ODBC.
  - Ο τύπος 2 είναι ένας driver γραμμένος σε μία άλλη γλώσσα προγραμματισμού (native code) συγκεκριμένη για κάθε ΣΔΒΔ που παρέχει ένα εξωτερικό στρώμα Java για τον προγραμματισμό εφαρμογών Java.

Τύποι JDBC Driver (2)

- Ο τύπος 3 είναι ένας driver γραμμένος σε Java που επικοινωνεί χρησιμοποιώντας ένα γενικό πρωτόκολλο δικτύου, με ένα ενδιάμεσο συστατικό (middleware component) το οποίο επικοινωνεί με το ΣΔΒΔ. Το ενδιάμεσο αυτό συστατικό μπορεί να χρησιμοποιήσει οποιοδήποτε τύπο Driver για να επικοινωνήσει με το ΣΔΒΔ.
- Τέλος, ο τύπος 4, είναι ένας driver γραμμένος εξ ολοκλήρου σε Java που χρησιμοποιεί απευθείας το πρωτόκολλο επικοινωνίας ενός συγκεκριμένου ΣΔΒΔ και έτσι επικοινωνεί απευθείας με το ΣΔΒΔ.

Αρχικοποίηση της σύνδεσης

```
import java.io.*; import java.sql.*;
import javax.servlet.*; import javax.servlet.http.*;
public class ShowStudentTable
   extends HttpServlet {
   private Connection con;
   public void init() {
      try {
         Class.forName("com.mysql.jdbc.Driver");
         con=DriverManager.getConnection(
"jdbc:mysql://localhost/school?user=user&pass
word=test123");
      }
      catch (Exception e) { log(e.getMessage()); }
```

Παράδειγμα servlet JDBC

- Έστω ο ακόλουθος πίνακας της ΒΔ school:
  - CREATE TABLE `school`.`students` (`sid` INTEGER UNSIGNED NOT NULL AUTO_INCREMENT, `sfname` VARCHAR(15) NOT NULL, `slname` VARCHAR(30) NOT NULL, PRIMARY KEY(`sid`)) TYPE = InnoDB;
- Έστω ακόμη πως στο ΣΔΒΔ έχουμε δημιουργήσει τον λογαριασμό χρήστη με όνομα χρήστη 'user' και κωδικό 'test123' ο οποίος έχει δικαιώματα στον πίνακα students της ΒΔ school.
- Θα δούμε το παράδειγμα ενός servlet που εμφανίζει τα περιεχόμενα του πίνακα αυτού μέσω JDBC.

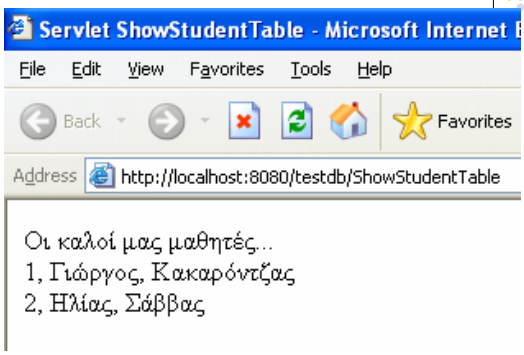
Ανάκτηση Δεδομένων (1)

```
protected void doGet(…) …{
…
if (con!=null) {
        try {
           Statement s = con.createStatement();
           ResultSet r = s.executeQuery(
                "Select * from students");
           out.println(
              "Οι καλοί μας μαθητές...<br>");
           while (r.next()) {
              int id = r.getInt("sid");
              String fname =
                 r.getString("sfname");
              Strng lname =
                 r.getString("slname");    …
```

Ανάκτηση Δεδομένων (2)

```
    out.println(""+id+", "+fname+",
            "+lname+"<br>");
   } //end of while
 }   //end of try
catch (SQLException e) {
   out.println("Error: "+e.getMessage()); }
 }
 out.println("</body>");
 out.println("</html>");
 out.close();
} //end of doGet
```

Το αποτέλεσμα εκτέλεσης του servlet



Κλείσιμο της σύνδεσης

```
public void destroy() {
    if (con!=null) {
       try {
          con.close();
       }
       catch (SQLException e) {
            log(e.getMessage()); }
    }
  }
```

Βιβλιογραφία

- Ο δικτυακός τόπος της java στο Internet: http://java.sun.com/
- Bruce Eckel: “Thinking in Java” online από την διεύθυνση http://www.mindview.net/Books/TIJ/
- Marty Hall: “Core Servlets and Java Server Pages, 1st edition”, online από την διεύθυνση http://pdf.coreservlets.com/
- Το Java Tutorial της SUN από την διεύθυνση http://java.sun.com/docs/books/tutorial/
- Κακαρόντζας Γιώργος σημειώσεις για Java, Java Servlets, JSP, UML και το NetBeans IDE online από την διεύθυνση http://www.cs.teilar.gr/gkakaron/
- Joshua Bloch: “Effective Java”, Addison Wesley, 2001

Ερωτήσεις;