ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Πακέτα (Packages), javadoc, Eclipse

Χ. Τζώρτζης

Πακέτα (Packages)

- Τα πακέτα είναι ένας τρόπος να οργανώσουμε συλλογές κλάσεων Java σε βιβλιοθήκες, και να αποφύγουμε τη σύγκρουση ονομάτων
- Ένα πακέτο μπορεί να περιέχει οποιοδήποτε αριθμό κλάσεων
  - που σχετίζονται μεταξύ τους με βάση το σκοπό, την εμβέλεια ή την κληρονομικότητά τους

Πακέτα και δηλώσεις **Import**

- Η Java χρησιμοποιεί *πακέτα* για να σχηματίσει βιβλιοθήκες κλάσεων
- Ένα **πακέτο** είναι μια ομάδα κλάσεων που έχουν τοποθετηθεί σε ένα κατάλογο (directory/folder)
- Μπορεί να χρησιμοποιηθούν από κάθε πρόγραμμα που περιλαμβάνει μια εντολή/πρόταση ***import*** που αναφέρεται στο πακέτο
  - Η import πρέπει να τοποθετείται στην αρχή του αρχείου προγράμματος
    - Μόνο κενές γραμμές, σχόλια, και δηλώσεις πακέτων μπορεί να προηγούνται
  - Το πρόγραμμα μπορεί να βρίσκεται σε διαφορετικό κατάλογο από το πακέτο

Δηλώσεις **import**

- Δίνουν εντολή στον compiler να καταστήσει ορατό ένα πακέτο στο αρχείο κώδικα
- Έχουμε ήδη χρησιμοποιήσει δηλώσεις import για να συμπεριλάβουμε προκαθορισμένα πακέτα της Java
  - π.χ. την κλάση **Scanner** από το πακέτο **java.util**

    import java.util.Scanner;
- Μπορούμε να έχουμε πρόσβαση σε όλες τις διαθέσιμες κλάσεις ένος πακέτου, αντί μόνο σε μία:

    import java.util.*;
  - Δεν υπάρχει επιπλέον επιβάρυνση για να εισάγουμε (import) ολόκληρο το πακέτο

Δημιουργία πακέτου

- Η δημιουργία ενός πακέτου είναι σχετικά απλή και μοιάζει με τη δημιουργία μιας κλάσης

Βήματα:

1. Επιλογή ονόματος για το πακέτο
2. Δημιουργία αντίστοιχου καταλόγου στο σκληρό δίσκο
   - Το βήμα αυτό εκτελείται αυτόματα σε ολοκληρωμένα περιβάλλοντα ανάπτυξης (IDE) όπως είναι ο Jbuilder, το Eclipse κτλ.
3. Προσθήκη της δήλωσης "**πακέτου**" στην αρχή του πηγαίου κώδικα. Π.χ.

   package test.package3.project2;

Δημιουργία και δήλωση **πακέτων**

- Για να δημιουργήσουμε ένα πακέτο, ομαδοποιούμε όλες τις κλάσεις σε ένα κατάλογο (directory), και προσθέτουμε στην αρχή κάθε αρχείου της κλάσης, την ακόλουθη δήλωση :

  package package_name;

  - Μόνο τα αρχεία **.class** πρέπει να βρίσκονται στον κατάλογο
    - τα αρχεία **.java** είναι προαιρετικά
  - Μόνο κενές γραμμές και σχόλια μπορούν να προηγούνται της δήλωσης πακέτου
  - Εάν υπάρχουν δηλώσεις import αλλά και δηλώσεις πακέτων, η δήλωση πακέτων πρέπει να προηγείται όλων των δηλώσεων import που υπάρχουν

Το πακέτο **java.lang**

- Περιέχει τις κλάσεις που είναι θεμελιώδεις για τον προγραμματισμό σε Java
  - Εισάγονται αυτόματα, και έτσι δεν χρειάζεται δήλωση import
  - Οι κλάσεις που γίνονται διαθέσιμες με το **java.lang** περιλαμβάνουν τις **Math, String**, και τις περιβάλλουσες (**wrapper**) κλάσεις

Ονόματα πακέτων και καταλόγων

- Το όνομα ενός πακέτου είναι το όνομα διαδρομής (path name) για τον κατάλογο ή υπο-κατάλογο (subdirectory) που περιέχει τις κλάσεις του πακέτου
  - Μόνο που αντί για "**/**" ή "****" χρησιμοποιείται το "**.**"
- Η Java χρειάζεται δυο πράγματα για να εντοπίσει τον κατάλογο ενός πακέτου:

1. το όνομα του πακέτου
2. και την τιμή της μεταβλητής περιβάλλοντος **CLASSPATH**
   - Η **CLASSPATH** είναι όμοια με την **PATH**, και τίθεται με τον ίδιο τρόπο για ένα δεδομένο λειτουργικό σύστημα
     - Π.χ. σε Windows C:\Java\jdk1.7.0\bin; C:\Java\packages
   - Η **CLASSPATH** τίθεται ίση με τη λίστα των κατάλογων σε έναν υπολογιστή (περιλαμβανόμενου του τρέχοντος, "**.**"), στα οποία η Java θα αναζητήσει πακέτα
   - Η Java ψάχνει αυτή τη λίστα με τη σειρά, και χρησιμοποιεί τον πρώτο κατάλογο στη λίστα, όπου βρέθηκε το πακέτο

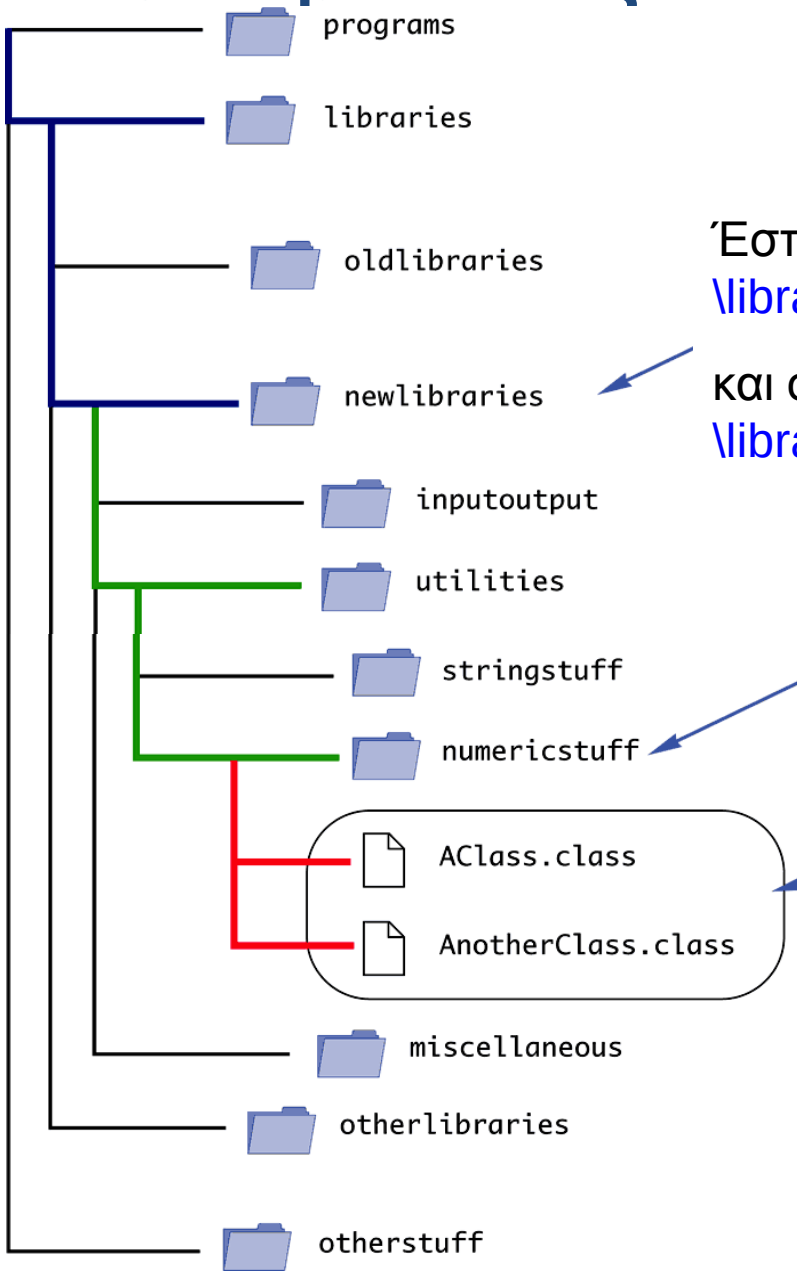
Ονόματα πακέτων

- Γενικά για ονόματα πακέτων χρησιμοποιούνται μόνο πεζά γράμματα και ψηφία, χωρίς underscore.
- Π.χ.:

  java.lang

  java.awt.image

Όνομα ενός πακέτου



Έστω το CLASSPATH περιέχει το \libraries\newlibraries

και οι κλάσεις του πακέτου είναι στο \libraries\newlibraries\utilities\numericstuff

Το πακέτο θα ονομαστεί utilities.numericstuff

και όλες οι κλάσεις στο πακέτο θα ξεκινούν με:

package utilities.numericstuff

Κάθε κλάση που χρησιμοποιεί κλασεις από το πακέτο utilities.numericstuff πρέπει να περιέχει την:

import utilities.numericstuff.*;

Προσοχή: οι υπό-κατάλογοι δεν εισάγονται αυτόματα

- Όταν ένα πακέτο b αποθηκεύεται σε ένα υπό-κατάλογο του κατάλογου που περιέχει κάποιο άλλο πακέτο a, το να εισάγουμε (import) το περιβάλλον πακέτο a, δεν σημαίνει ότι εισάγεται το πακέτο b του υπό-κατάλογου
- Η import δήλωση:

  import utilities.numericstuff.*;

εισάγει μόνο το πακέτο **utilities.numericstuff**

- Οι import δηλώσεις:

  import utilities.numericstuff.*;

  import utilities.numericstuff.statistical.*;

εισάγουν και το **utilities.numericstuff** αλλά

και το **utilities.numericstuff.statistical**

Το εξ' ορισμού (default) πακέτο

- Όλες οι κλάσεις στον τρέχοντα κατάλογο ανήκουν σε ένα προεπιλεγμένο ανώνυμο πακέτο που αποκαλείται το *default package*
- Εφόσον ο τρέχων κατάλογος (.) είναι τμήμα της μεταβλητής **CLASSPATH**, όλες οι κλάσεις στο default package είναι αυτόματα διαθέσιμες στο πρόγραμμα
- Το CLASSPATH τίθεται στα Windows από το Control Panel (system->system properties-> advanced->environment variables), στο Unix με εντολές όπως:
  - set CLASSPATH=.;/libraries/newlibraries;/stuff/specialjava
  - export CLASSPATH

  ή
  - setenv CLASSPATH=.;/libraries/newlibraries;/stuff/specialjava
  - export

Προσοχή: Μη συμπερίληψη του τρέχοντος κατάλογου στο CLASSPATH

- Εάν η μεταβλητή **CLASSPATH** έχει ήδη τεθεί, ο τρέχων κατάλογος πρέπει να συμπεριλαμβάνεται ως μια εναλλακτική
  - Διαφορετικά, η Java ίσως δεν είναι σε θέση να βρει τα αρχεία.**class** για το πρόγραμμα
- Εάν η μεταβλητή **CLASSPATH** δεν έχει τεθεί, τότε όλα τα αρχεία κλάσεων για ένα πρόγραμμα πρέπει να τοποθετηθούν στον τρέχοντα κατάλογο

Καθορισμός ενός CLASSPATH κατά τη μεταγλώττιση

- Το CLASSPATH μπορεί να καθοριστεί “χειρονακτικά” όταν μεταγλωττίζεται μια κλάση
  - Απλά προσθέτουμε το **–classpath** ακολουθούμενο από το επιθυμητό CLASSPATH
    - Π.χ. javac –classpath .;C:libraries\newlibraries YourClass.java
  - Αυτό θα μεταγλωττίσει την κλάση, κάνοντας override προηγούμενες αναθέσεις στο **CLASSPATH**
- Θα πρέπει να χρησιμοποιούμε την επιλογή **–classpath** όταν εκτελείται η κλάση
    - Π.χ. java –classpath .;C:libraries\newlibraries YourClass

Συγκρούσεις ονομάτων

- Επιπρόσθετα του να κρατάμε τις βιβλιοθήκες κλάσεων οργανωμένες, τα πακέτα παρέχουν έναν τρόπο για να χειριστούμε τις συγκρούσεις ονομάτων (*name clashes*): μια κατάσταση όπου δυο κλάσεις έχουν το ίδιο όνομα
  - Πχ. σε μια ένα μεγάλο σύνολο κλάσεων, δύο διαφορετικές κλάσεις μπορεί να έχουν το ίδιο όνομα.
    - Αυτό αντιμετωπίζεται εύκολα εάν οι δύο κλάσεις ανήκουν σε διαφορετικά πακέτα.
  - Διαφορετικοί προγραμματιστές που γράφουν διαφορετικά πακέτα μπορεί να χρησιμοποιούν το ίδιο όνομα για δυο ή περισσότερες κλάσεις
  - Αυτή η αμφιλογία (ambiguity) μπορεί να επιλυθεί με χρήση του πλήρως προσδιορισμένου ονόματος (*fully qualified name* - δηλ., το όνομα του πακέτου προηγείται του ονόματος της κλάσης) για να διαχωρίσουμε τις δυο κλάσεις

    package_name.ClassName
  - Εάν χρησιμοποιηθεί το πλήρες όνομα, δεν είναι πλέον αναγκαίο να εισάγουμε (import) την κλάση
    - επειδή περιλαμβάνει ήδη το όνομα του πακέτου

Χρήση πακέτων

Μπορούμε να χρησιμοποιήσουμε μια κλάση:

- κατευθείαν με το όνομά της, εάν βρίσκεται στο πακέτο java.lang
- με το πλήρες όνομά της, εάν βρίσκεται σε κάποιο άλλο πακέτο, πχ. java.util.random
- με το όνομά της κλάσης, εάν έχουμε κάνει ‘import”
  - είτε την κλάση μόνο

    import java.util.vector;
  - είτε όλα τα περιεχόμενου του αντίστοιχου πακέτου

    import java.util.*;

Πλεονεκτήματα των πακέτων

- Επιτρέπουν την οργάνωση των κλάσεων σε πακέτα-συλλογές
- Μειώνουν τα προβλήματα από τις συγκρούσεις ονομάτων
- Μπορούν να χρησιμοποιηθούν για την ταυτοποίηση των κλάσεων
  - Πχ. οι κλάσεις που υλοποιεί μια ομάδα για μία εργασία μπορούν να οργανωθούν σε ένα πακέτο με το όνομα της ομάδας, της εργασίας κτλ.
  - Στην πράξη, οι εταιρείες παραγωγής λογισμικού δημιουργούν ιεραρχίες πακέτων που έχουν ως αφετηρία το όνομα της εταιρείας

javadoc

- Σε αντίθεση με γλώσσες όπως η C++, η Java τοποθετεί τη διεπαφή (API –Application Programming Interface) και την υλοποίηση μιας κλάσης στο ίδιο αρχείο
- Όμως, η Java έχει ένα πρόγραμμα, το **javadoc**, το οποίο αυτόματα εξάγει τη διεπαφή από τον ορισμό της κλάσης και παράγει τεκμηρίωση (documentation)
  - Αυτή η πληροφορία παρουσιάζεται σε μορφή HTML
  - Εάν μια κλάση σχολιάζεται επαρκώς, ένας προγραμματιστής χρειάζεται να αναφέρεται μόνο στην τεκμηρίωση του API που παράγει το *javadoc* για να χρησιμοποιήσει την κλάση
  - Το **javadoc** μπορεί να εξάγει τεκμηρίωση από οτιδήποτε, μια κλάση ή ένα ολόκληρο πακέτο

Σχόλια σε κλάσεις για το **javadoc**

- Το πρόγραμμα **javadoc** εξάγει τις κεφαλίδες κλάσεων, τις κεφαλίδες για μερικά σχόλια, και τις κεφαλίδες για όλες τις public μεθόδους, instance variables, και static μεταβλητές
  - Στο κανονικό default mode, δεν εξάγονται τα σώματα των μεθόδων ούτε τα private μέλη της κλάσης
- Για να εξαγάγουμε ένα σχόλιο, πρέπει:

1. το σχόλιο να *προηγείται αμέσως* μιας public κλάσης ή ορισμού μεθόδου ή κάποιου άλλου public μέλους
2. **ΚΑΙ** το σχόλιο να είναι ένα block σχόλιο (/* ....*/), και το πρώτο ***/**** πρέπει να περιέχει ένα επιπλέον *

/** . . . */

- Σημείωση: Επιπλέον επιλογές πρέπει να τεθούν για να εξαχθούν τα σχόλια γραμμής (*//*) και τα private μέλη

Σχόλια σε κλάσεις για το **javadoc** (II)

- Το σχόλιο που προηγείται του ορισμού μιας public μεθόδου πρέπει να περιλαμβάνει περιγραφές των παραμέτρων, των επιστρεφόμενων τιμών, και των όποιων εξαιρέσεων (exceptions)
- Αυτός το τύπος πληροφορίας έπεται του συμβόλου @ και αποκαλείται ετικέτα @ ή *@ tag*
- @ tags έπονται των γενικών σχολίων, και κάθε ένα καταλαμβάνει μια γραμμή από μόνο του

```
/**
General Comments about the method . . .
@param aParameter Description of aParameter
@return What is returned
. . .
*/
```

@ Tags

- Τα @ tags πρέπει να τοποθετούνται με τη σειρά που φαίνεται παρακάτω
- Εάν υπάρχουν πολλαπλές παράμετροι, κάθε μια πρέπει να έχει το δικό της **@param** σε χωριστή γραμμή, και κάθε μια να παρατίθεται σύμφωνα με την αριστερά-προς-δεξιά σειρά της στη λίστα των παραμέτρων
- Εάν υπάρχουν πολλαπλοί συγγραφείς, κάθε ένας θα πρέπει να έχει το δικό του **@author** σε χωριστή γραμμή

@param Parameter_Name Parameter_Description
@return Description_Of_Value_Returned
@throws Exception_Type Explanation
@deprecated
@see Package_Name.Class_Name
@author Author
@version Version_Information

Εκτελώντας το **javadoc**

- Για να εκτελέσετε το **javadoc** σε ένα πακέτο, δώστε την ακόλουθη εντολή:

  javadoc –d Documentation_Directory Package_Name

  - Τα HTML έγγραφα που θα παραχθούν θα τοποθετηθούν στο **Documentation_Directory**
  - Εάν τα **–d** και **Documentation_Directory** παραληφθούν, τότε το **javadoc** θα δημιουργήσει κατάλληλα directories για το documentation
- Για να εκτελέσετε το **javadoc** σε μια κλάση, δώστε την ακόλουθη εντολή από τον κατάλογο που περιέχει το αρχείο της κλάσης:

  javadoc ClassName.java
- Για να εκτελέσετε το **javadoc** σε όλες τις κλάσεις ενός κατάλογου, δώστε την ακόλουθη εντολή:

  javadoc *.java

Εξωτερική τεκμηρίωση

- Υπάρχουν εργαλεία όπως το javadoc, που παράγουν αυτόματα εξωτερική τεκμηρίωση, με βάση τα σχόλια του κώδικα
  - Βάζουμε σχόλια που προηγούνται μιας κλάσης, ενός πεδίου μιας μεθόδου
  - περιλαμβάνουν javadoc macros @param, @return, @author, @version …
  - /** begin comment that is included in javadoc */
- Τελικά παράγονται html σελίδες εξωτερικής τεκμηρίωσης

Επιλογές για `javadoc`

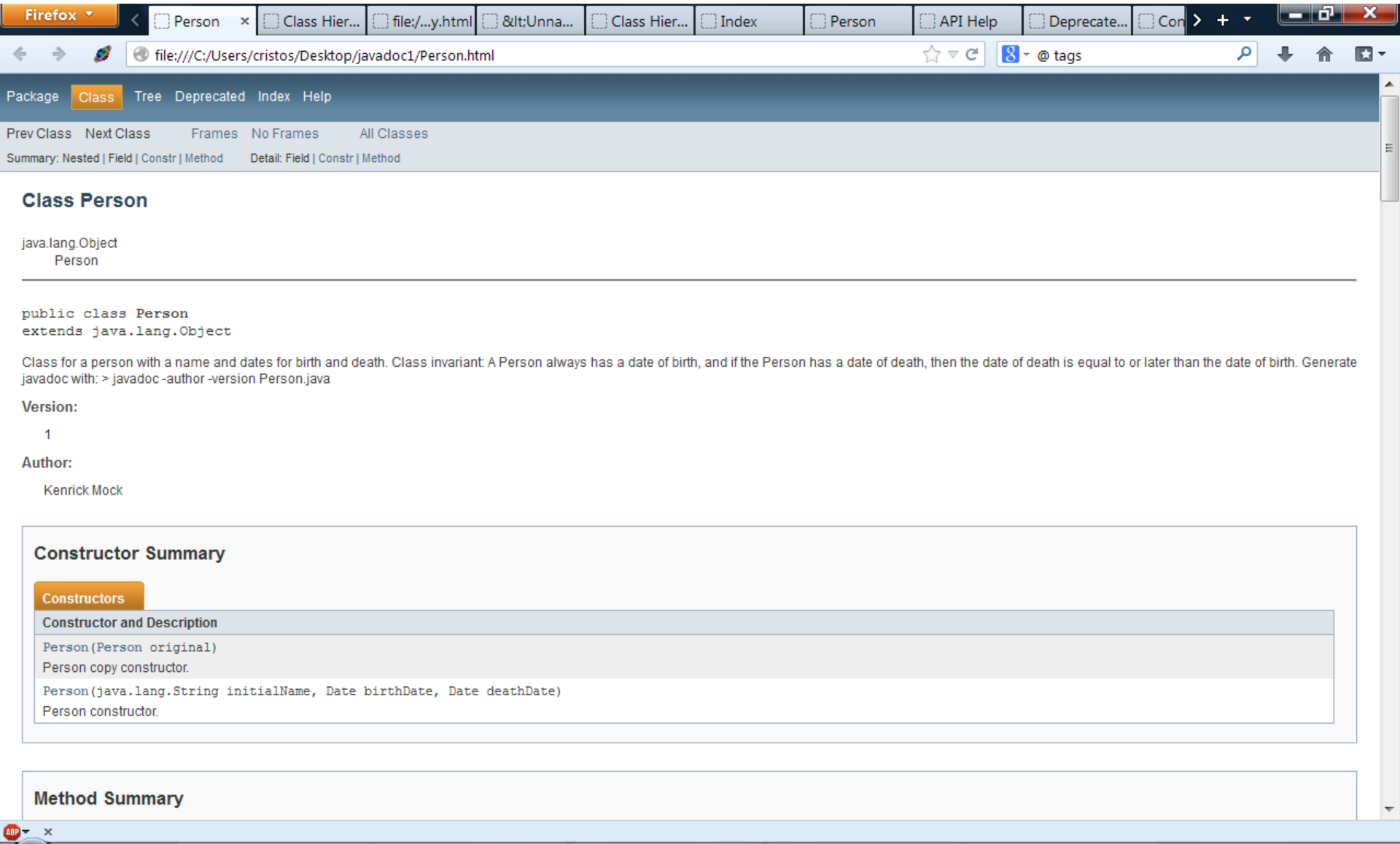
`-link` *Link_To_Other_Docs*	Provides a link to another set of documentation. Normally, this is used with either a path name to a local version of the Java documentation or the URL of the Sun Web site with standard Java documentation.
`-d` *Documentation_Directory*	Specifies a directory to hold the documentation generated. *Documentation_Directory* may be a relative or absolute path name.
`-author`	Includes author information (from `@author` tags). This information is omitted unless this option is set.
`-version`	Includes version information (from `@version` tags). This information is omitted unless this option is set.
`-classpath` *List_of_Directories*	Overrides the CLASSPATH environment variable and makes *List_of_Directories* the CLASSPATH for the execution of this invocation of `javadoc`. Does not permanently change the CLASSPATH variable.
`-private`	Includes private members as well as public members in the documentation.

```
/**
 Class for a person with a name and dates for birth and death.
 Class invariant: A Person always has a date of birth, and if the Person  has a date of death, then
   the date of death is equal to or later than the
 date of birth.
 Generate javadoc with:
   > javadoc -author -version Person.java

 @author Kenrick Mock
 @version 1
*/

public class Person
{
    private String name;
    private Date born;
    private Date died;//null indicates still alive.

   /**
   Person constructor.
   @param initialName name of the person
   @param birthDate date of birth
   @param deathDate date of death
   */
```

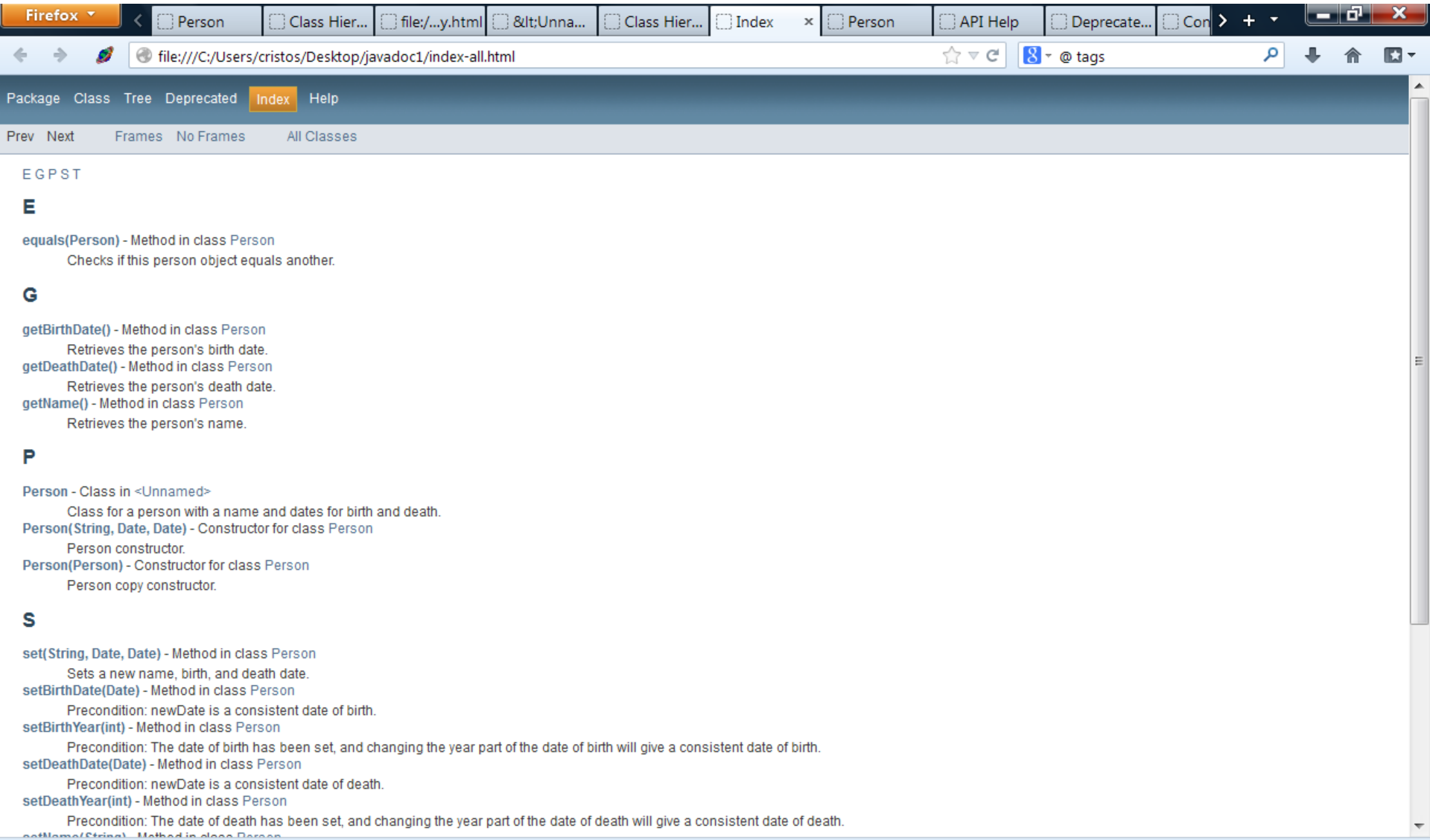
Dry run

```
C:\Java\jdk1.7.0\bin"\javadoc –author -version Person.java
Loading source file Person.java...
Constructing Javadoc information...
Standard Doclet version 1.7.0
Building tree for all the packages and classes...
Generating \Person.html...
Generating \package-frame.html...
Generating \package-summary.html...
Generating \package-tree.html...
Generating \constant-values.html...
Building index for all the packages and classes...
Generating \overview-tree.html...
Generating \index-all.html...
Generating \deprecated-list.html...
Building index for all classes...
Generating \allclasses-frame.html...
Generating \allclasses-noframe.html...
Generating \index.html...
Generating \help-doc.html...
```

Person.html


Firefox
Person
Class Hier...
file:/...y.html
<Unna...
Class Hier...
Index
Person
API Help
Deprecate...
file:///C:/Users/cristos/Desktop/javadoc1/Person.html
@ tags
Package
Class
Tree
Deprecated
Index
Help
Prev Class
Next Class
Frames
No Frames
All Classes
Summary: Nested | Field | Constr | Method
Detail: Field | Constr | Method
Class Person
java.lang.Object
Person
public class Person
extends java.lang.Object
Class for a person with a name and dates for birth and death. Class invariant: A Person always has a date of birth, and if the Person has a date of death, then the date of death is equal to or later than the date of birth. Generate javadoc with: > javadoc -author -version Person.java
Version:
1
Author:
Kenrick Mock
Constructor Summary
Constructors
Constructor and Description
Person(Person original)
Person copy constructor.
Person(java.lang.String initialName, Date birthDate, Date deathDate)
Person constructor.
Method Summary

index-all.html



Eclipse

- Το Eclipse είναι ένα πλήρως επεκτάσιμο, open-source ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment- IDE) για ανάπτυξη όλων των τύπων εφαρμογών της Java
- Έχει πολλές ενσωματωμένες λειτουργίες για την απλοποίηση και εξορθολογισμό ανάπτυξης προγραμμάτων σε Java, και ιδιαίτερα πολύπλοκων συστημάτων λογισμικού
- Είναι η βάση για πολλά εμπορικά IDE όπως τα: Rational XDE, WebSphere Application Developer, κλπ.

Εγκατάσταση

- Μπορείτε να εγκαταστήσετε το Eclipse σε μηχάνημα Windows / Linux
- Θα χρειαστείτε το Eclipse IDE από το: http://www.eclipse.org/downloads/index.php
- και Java Runtime Environment (JRE) από το:

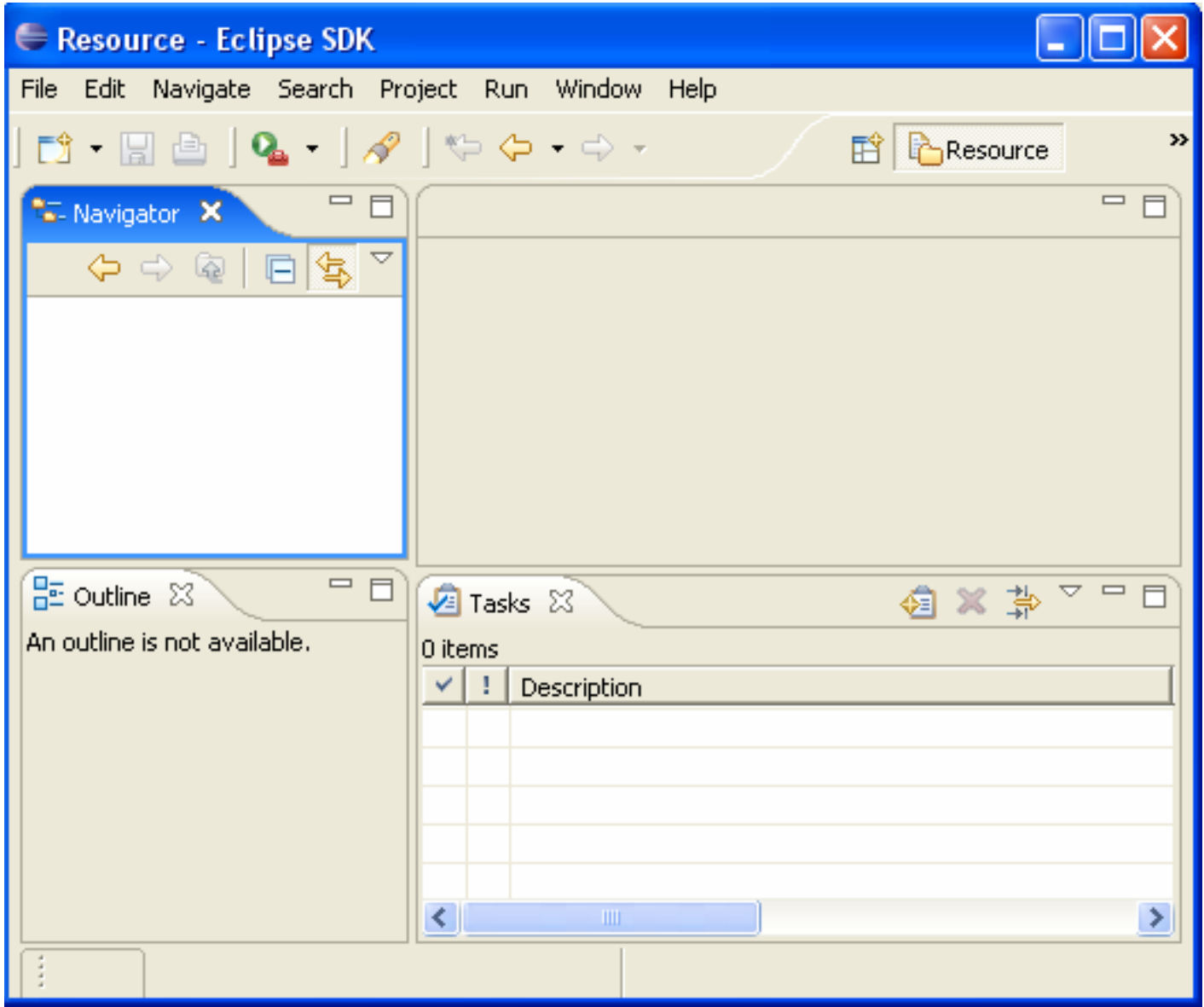
http://www.oracle.com/technetwork/java/javase/downloads/index.html

Eclipse Workspace

- Μόλις ολοκληρωθεί η εγκατάσταση, θα πρέπει να επιλέξετε ένα φάκελο για το workspace όπου θα είναι αποθηκευμένα τα project σας
- Μπορείτε να αποθηκεύσετε πολλά project στο ίδιο workspace αλλά σε διαφορετικές υποφακέλους
- Το Eclipse υποστηρίζει πολλαπλούς χώρους εργασίας (workspace) και μπορείτε να επιλέξετε workspace, όταν ξεκινάτε το Eclipse

Eclipse Setup

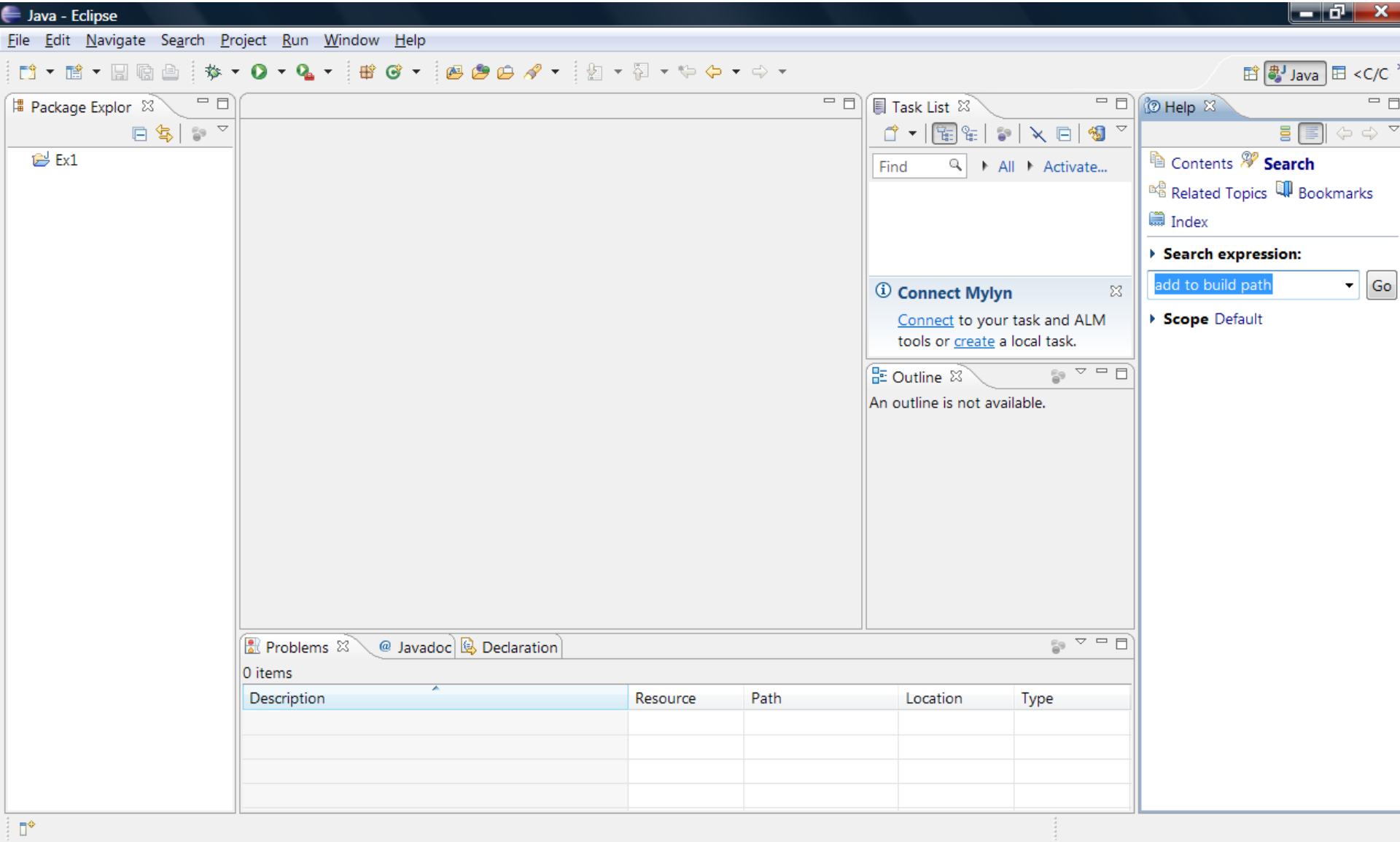
- Μόλις «τρέξει» το Eclipse, θα πρέπει να δείτε το παρακάτω:



Το πρώτο Project με το Eclipse

- Για να δημιουργήσετε ένα Project στο Eclipse, επιλέξτε
  File -> New -> Project
- Για να δημιουργήσετε ένα πρόγραμμα Java, επιλέξτε Java Project -> Next -> Enter Project Name (FirstEclipseProject) -> Select Create Separate Source and Output Folders -> Finish
- Μόλις ολοκληρωθεί το πιο πάνω, το Eclipse θα σας ζητήσει να ενεργοποιήσετε την Java Perspective, επιλέξτε Yes και έπειτα η προοπτική αυτή θα ανοίξει
- Παρατηρήστε τις διαφορετικές απόψεις (Views) και προοπτικές (Perspectives) στο μενού Window, επιλέγοντας Open Perspective ή Show View
- Αναπτύξτε το FirstEclipseProject από το Package Explorer View για να δείτε την άδεια δομή του Project

Java Perspective



Η πρώτη κλάση με το Eclipse

- Για να δημιουργήσετε μια κλάση Java στο Eclipse, επιλέξτε File -> New -> Class-> Πληκτρολογήστε το όνομα πακέτου* (firsteclipsepackage) ->Πληκτρολογήστε το όνομα της κλάσης (FirstEclipseClass) -> Επιλέξτε public static void main -> Αποεπιλέξτε Inherited abstract methods-> Finish

*Χρησιμοποιείστε το προκαθορισμένο πακέτο ή ορίστε δικό σας με File -> New ->Package->firsteclipsepackage

- Παρατηρήστε το αρχείο πηγαίου κώδικα που άνοιξε στην main view
- Κάτω από την μέθοδο main, πληκτρολογήστε: System.out.println(“My First Eclipse Class is Running");
- Αποθηκεύστε το αρχείο μέσω του File -> Save ή χρησιμοποιώντας το συνδυασμό πλήκτρων CTRL + S
- Εκτελέστε την κλάση main, επιλέγοντας Run -> Run as-> Java Application, και δείτε τα αποτελέσματα σε Console view

Εισάγωγη (Import) εξωτερικών πακέτων

- Προσθέστε κάτω από την main στην κλάση FirstEclipseClass, μια νέα μέθοδο public static void printVector(Vector input)
- Αποθηκεύστε το αρχείο, και παρατηρήστε τη λέξη Vector να υπογραμμίζεται με κόκκινο
  - Σύμφωνα με τα προβλήματα που αναφέρονται (Problems listed) "δεν μπορεί να επιλυθεί" ("cannot be resolved to a type")
- Επιλέξτε Source -> Organize Imports ή επιλέξτε το συνδυασμό πλήκτρων CTRL + SHIFT + O
- Αποθηκεύστε το αρχείο και δέστε πως το Vector επιλύθηκε και ότι ένα νέο πακέτο java.util.Vector εισήχθη
- Εναλλακτικά: αντί για Source -> Organize Imports, επιλέξτε τη λέξη Vector και επιλέξτε Source -> Add Import

Πλοήγηση Εξωτερικών Τύπων (External Types)

- Κάτω από την μέθοδο printVector, γράψτε “for" και πιέστε CTRL + Space για να ανοίξει ο type navigator
- Για να αναφερθείτε στο σώμα της printVector από τη μέθοδο main, κρατήστε το CTRL και κάντε κλικ στο printVector.
  - Το ίδιο ισχύει και για τα χαρακτηριστικά (attributes)
- Επιλέξτε Iterate over array και στη συνέχεια αντικαταστήσετε το "array.length" με "input.", για να πάρετε μια λίστα από τα type members για το Vector
- Επιλέξτε το size() και στο σώμα του βρόχου for γράψτε τα ακόλουθα, παρατηρώντας τον type navigator να ανοίγει μετά από κάθε τελεία που πληκτρολογούμε System.out.println (input.get (i));