

Άδειες open source Εργαλεία Ανάπτυξης Κώδικα ΕΛΛΑΚ

Απόστολος Βόγκλης
Βασίλειος Καραβασίλης
Μονάδα Αριστείας ΕΛΛΑΚ | ΕΤΕΠΗ | 9/3/2015

```
PaintView.java x ControlView.java
+ * Copyright (c) 2000, 2005 IBM Corporation and others.
package org.eclipse.swt.examples.paint;

+ import org.eclipse.jface.action.*;

/**
 * The view for the paint application.
 * All rendering happens inside the area created by createPartControl().
 * @see ViewPart
 */
public class PaintView extends ViewPart {
    PaintExample instance = null;

    /**
     * Constructs a Paint view.
     */
    public PaintView() {
    }

    /**
     * Creates the example.
     * @see ViewPart#createPartControl
     */
    public void createPartControl(Composite parent) {
        instance = new PaintExample(parent);
        instance.createGUI(parent);
    }
}
```

Άδειες open source

Εισαγωγή

- Υπάρχουν πολλές άδειες open source
 - περίπου 70
 - <http://opensource.org/licenses/alphabetical>
- Για πληροφορίες
 - Google: open source licenses explained
- Θα παρουσιάσουμε τα πιο διαδεδομένα
 - Γενικά, υπάρχουν πολλές εκδόσεις, θα παρουσιάσουμε την κεντρική ιδέα στο καθένα

Άδειες open source

- GNU General Public Licence (GPL)
- GNU Lesser General Public Licence (LGPL)
- BSD License
- MIT License
- Apache License
- Creative Commons (CC)

Γιατί είναι χρήσιμες

- Είναι ήδη έτοιμες, οπότε μπορούμε να ασχοληθούμε με την δημιουργία του λογισμικού πιο πολύ χρόνο
 - Πρέπει να καταλάβουμε όμως τι λένε
- Γενικά, μας προστατεύουν:
 - Σαν συγγραφείς, δεν μπορεί κάποιος να πει ότι έγραψε αυτός το πρόγραμμα
 - Από κακή χρήση του λογισμικού (πχ αν κάποιος κάψει τον σκληρό χρησιμοποιώντας το πρόγραμμά μας)
 - Ως χρήστες, δεν είμαστε δεσμευμένοι σε έναν συγκεκριμένο πάροχο

GNU General Public Licence (GPL)

- Ίσως η πιο διαδεδομένη
- Ο οποιοσδήποτε μπορεί να:
 - αντιγράψει το λογισμικό
 - να το αναδιανείμει με οποιονδήποτε τρόπο
 - να το πουλήσει
 - Πρέπει όμως να πει ότι είναι GPL και ότι ο κώδικας είναι διαθέσιμος κάπου (δωρεάν)
 - να το τροποποιήσει
 - Πρέπει να δώσει τον κώδικα των αλλαγών
 - να το συμπεριλάβει σε κάποιο άλλο λογισμικό
 - Πρέπει όμως όλο το λογισμικό να είναι GPL
- Παραδείγματα: Linux, Gnome, GCC, MySQL

GNU Lesser General Public Licence (LGPL)

- Παρόμοια με την GPL
- Διαφορά:
- Μπορούμε να συμπεριλάβουμε το λογισμικό σε ένα άλλο που δεν είναι LGPL
 - Μπορούμε να το χρησιμοποιήσουμε και να μην δώσουμε τον πηγαίο κώδικα του νέου λογισμικού
 - Συνήθως σε αυτή την περίπτωση υπάρχει κάποια χρέωση (χρηματική)
- Παραδείγματα: Hibernate, ALSA driver

BSD License

- Πρέπει να δώσουμε ένα αντίγραφο του κειμένου της άδειας
- Πρέπει να πούμε ποιος το έγραψε και ποιό λογισμικό είναι
- Μπορούμε να χρησιμοποιήσουμε ένα λογισμικό με MIT license σε ένα μεγαλύτερο λογισμικό, να δώσουμε το εκτελέσιμο χωρίς τον κώδικα
 - Είτε δωρεάν είτε όχι
- Παραδείγματα: Free BSD, WebKit, Nginx

MIT License

- Έχει το πιο μικρό κείμενο
- Ένα λογισμικό σε MIT License μπορούμε να
 - το αντιγράψουμε
 - το τροποποιήσουμε
 - το διανείμουμε (με ή χωρίς πληρωμή)
 - να μην δώσουμε τον κώδικα
- Πρέπει να συμπεριλάβουμε αντίγραφο του κειμένου της άδειας
- Παραδείγματα: AngularJS, jQuery

Apache License

- Ένα λογισμικό με Apache license μπορούμε να:
 - το αντιγράψουμε
 - το τροποποιήσουμε
 - το διανείμουμε (με ή χωρίς πληρωμή)
 - να μην δώσουμε τον κώδικα
- Πρέπει να συμπεριλάβουμε:
 - τι λογισμικό χρησιμοποιήσαμε
 - τι αλλαγές έχουμε κάνει
 - αντίγραφο του κειμένου της άδειας
- Παραδείγματα: Android, Apache httpd server, puppet

Creative Commons (CC)

- Χρησιμοποιείται κυρίως σε καλλιτεχνικά έργα (φωτογραφίες κλπ)
- Ένα έργο σε CC για να το χρησιμοποιήσουμε πρέπει:
 - να πούμε ποιο έργο από ποιόν δημιουργό χρησιμοποιούμε
 - μπορούμε να το τροποποιήσουμε, αλλά το νέο έργο πρέπει να είναι CC
 - Δεν μπορούμε να το χρησιμοποιήσουμε εμπορικά
- Παραδείγματα: μερικά περιεχόμενα στο flickr

Συχνές Ερωτήσεις

- Πως μπορούμε να έχουμε έσοδα από ελεύθερο λογισμικό;
 - Οι περισσότερες άδειες επιτρέπουν να πουλήσουμε το λογισμικό (είτε τον κώδικα είτε το εκτελέσιμο).
 - Η διαφορά μεταξύ τους είναι αν μας υποχρεώνουν να δώσουμε και τον πηγαίο κώδικα.
 - Μπορούμε να δίνουμε και εκτελέσιμο και κώδικα δωρεάν, αλλά να χρεώνουμε τις υπηρεσίες υποστήριξης (πχ Ubuntu).
 - Μπορούμε να δίνουμε τον πηγαίο κώδικα δωρεάν, αλλά να πουλάμε το εκτελέσιμο (πχ. Red Hat Linux).
 - Μπορούμε να δίνουμε το κυρίως μέρος του λογισμικού δωρεάν, αλλά ένα συγκεκριμένο πρόσθετο να το χρεώνουμε (πχ Mysql Enterprise).
 - Μπορούμε να δίνουμε δωρεάν για open source λογισμικά, αλλά να μας πληρώνουν αν το χρησιμοποιήσουν σε κλειστό (LGPL).
 - Να δεχόμαστε donations (πχ Mozilla firefox).

Συχνές ερωτήσεις

- Μπορεί μια εταιρία να κλέψει το λογισμικό μας;
 - Εξαρτάται από την άδεια.
 - Γενικά, αν κάποιος χρησιμοποιήσει το λογισμικό εσωτερικά, χωρίς να το δώσει σε άλλους, μπορεί να το κάνει (και οι περισσότερες άδειες το επιτρέπουν).
 - Μια εταιρία μπορεί να χρησιμοποιήσει το λογισμικό μας και να παρέχει μια υπηρεσία (όχι να πουλά το λογισμικό).
 - Κάποιες άδειες (πχ GPL) υποχρεώνουν να μια εταιρία χρησιμοποιήσει το λογισμικό ως μέρος άλλου λογισμικού ή τροποποιημένο, να διαθέσει και το υπόλοιπο λογισμικό ή τις τροποποιήσεις με GPL (πχ Hyper-V Linux Drivers). Οπότε θα μπορούμε να δούμε τι αλλαγές έκανε.
 - Αν υπάρχει πρόβλημα αρμόδια είναι τα δικαστήρια (Oracle v Google Android).

Συχνές ερωτήσεις

- Μπορώ να σταματήσω να δίνω τον κώδικα σε ένα δικό μου λογισμικό;
 - Ανάλογα την άδεια.
 - Η GPL δεν το επιτρέπει. Ουσιαστικά ο κώδικας δεν είναι δικός μας πια (αν και είμαστε οι συγγραφείς). Είναι της κοινότητας
 - Κάποιες άδειες επιτρέπουν να μην δίνουμε τις αλλαγές, οπότε μπορούμε να σταματήσουμε να δίνουμε τον κώδικα.

Συχνές ερωτήσεις

- Μπορώ να αλλάξω την άδεια σε ένα λογισμικό; Πως γίνεται να υπάρχουν λογισμικά με 2 άδειες (dual license, πχ GPL μαζί με LGPL)
 - Αν μας ανήκει ο κώδικας (είμαστε οι ιδιοκτήτες) μπορούμε να τον διανείμουμε με όσες άδειες θέλουμε.
 - Μπορούμε επίσης να αλλάξουμε τελείως την άδεια
 - πχ μέχρι την έκδοση 1.9 να έχουμε GPL και από την 2.0 και μετά να έχουμε Apache.
 - Αν οι ιδιοκτήτες είναι παραπάνω από ένας, πρέπει να συμφωνήσουν όλοι.
 - Για ήδη υπάρχοντα project ανοιχτού λογισμικού με πολλούς συμμετέχοντες στην ανάπτυξη είναι δύσκολο να αλλάξει γιατί πρέπει όλοι να συμφωνήσουν, ακόμη και αν κάποιος έγραψε 1 γραμμή κώδικα.
 - Μερικά έργα απαιτούν να τους μεταφερθεί η ιδιοκτησία του κώδικα όταν κάποιος συνεισφέρει κώδικα. Με αυτό τον τρόπο οι υπεύθυνοι του project μπορούν να πάρουν πιο εύκολα απόφαση για σημαντικές αλλαγές (πχ αλλαγή άδειας).

Μονάδες αριστείας

- Δεν υπάρχει υποχρεωτική άδεια.
- Ωστόσο προτείνεται η European Union Public License
 - Παρόμοια με την Apache

Άλλες Ερωτήσεις;

Εργαλεία ανάπτυξης λογισμικού

Ιστορικό

- 40'-60' : Το λογισμικό εξαρτάται από το υλικό.
 - Τεράστια Computer Rooms
 - Το υλικό άλλαζε κάθε 2-3 χρόνια και ήταν ασύμβατο με το προηγούμενο
 - Δεν υπήρχε ο προσωπικός υπολογιστής για κάθε προγραμματιστή
 - Ανάπτυξη σε διάτρητες κάρτες
 - Διαχείριση : Δεν υπήρχε ασφαλής πρόβλεψη για την ολοκλήρωση ενός έργου.

Ιστορικό

- 40'-60' : Εξέλιξη.
 - Εισαγωγή υψηλού επιπέδου γλωσσών προγραμματισμού (FORTRAN, COBOL)
 - Λίγες εταιρίες παρείχαν λογισμικό μαζί όμως με το υλικό
 - Εμφανίστηκε η έννοια της επαναχρησιμοποίησης λογισμικού
 - Αρθρωτός προγραμματισμός και data abstraction

Ιστορικό

- 60'-80' : Τεχνολογία Λογισμικού.
 - Ο όρος εισήχθη το 1968 : Συνέδριο του NATO
 - Software Crisis
 - Κακή ποιότητα λογισμικού
 - Υπέρβαση χρόνου και κόστους παραγωγής
 - Τα προβλήματα αυτά υπήρχαν και σε άλλους κλάδους της μηχανικής
 - Software Engineering : Εφαρμογή λύσεων από άλλους κλάδους στην ανάπτυξη λογισμικού.

Ιστορικό

- 60'-80' : Τεχνολογία Λογισμικού.
 - Απάντηση στο πρόβλημα παραγωγικότητας
 - Εξελίχθηκε όπως στην επίλυση του προβλήματος της ποιότητας
 - Έλλειψη εξειδίκευσης

Ιστορικό

- 80'-90' : Τα Προβλήματα Παραμένουν
 - Σημαντικό πεδίο έρευνας και ανάπτυξης
 - (80') Το κόστος απόκτησης και συντήρησης ήταν διπλάσιο από την ανάπτυξή του
 - (90') Το κόστος απόκτησης αυξήθηκε κατά 30%
 - (1995) Τα μισά λειτουργικά έργα ήταν ανεπιτυχή.
 - Υπέρβαση χρόνου κατά +50% (Μ.Ο.)
 - Το 75% των μεγάλων έργων ήταν αποτυχία ή κατώτερο των προδιαγραφών.

Ιστορικό

- 80'-90' : Τα Προβλήματα Παραμένουν
 - Τι θα μας λύσει τα προβλήματα;
 - Εργαλεία;
 - Διαδικασία υλοποίησης;
 - Επαγγελματισμός / Πειθαρχία;
 - Μεθοδολογία;
 - Διαχείριση έργου;

Ιστορικό

- 90'-Σήμερα : Τα Προβλήματα Παραμένουν(;)
 - Όλα μαζί
 - Εργαλεία
 - Διαδικασία υλοποίησης
 - Επαγγελματισμός / Πειθαρχία
 - Μεθοδολογία
 - Διαχείριση έργου
 - Internet
 - Μικρότερος και πιο ευέλικτος καταμερισμός εργασιών

Δομή Παρουσίασης

- Ιστορικό Ανάπτυξης Λογισμικού
- **Εργαλεία Ανάπτυξης**
- Integrated Development Environments (IDE)
- Eclipse IDE
- Git

Εργαλεία Ανάπτυξης Λογισμικού

- Γλώσσες Προγραμματισμού
 - C/C++
 - Java
 - Visual Basic (-)
- Scripting Γλώσσες
 - Perl
 - PHP
 - Bash

By # of repositories created on Github so far this year:

Rank	Language	# Repositories Created
1	JavaScript	264131
2	Ruby	218812
3	Java	157618
4	PHP	114384
5	Python	95002
6	C++	78327
7	C	67706
8	Objective-C	36344
9	C#	32170
10	Shell	28561
11	CSS	17813
12	Perl	15412
13	CoffeeScript	11133
14	VimL	7857
15	Scala	6918
16	Go	6884
17	Prolog	5829
18	Clojure	4904
19	Haskell	4681
20	Lua	4048

Εργαλεία Ανάπτυξης Λογισμικού

- Linker
 - Συνδέει σύνολα βιβλιοθηκών, αρχείων αντικειμενικού κώδικα (object files) σε εκτελέσιμα προγράμματα.
- Debugger
 - Έλεγχος και αποσφαλμάτωση προγραμμάτων (gdb, Valgrind, Parasoftware Insure++, κτλ)
- Memory Debugger
 - Διαχείριση μνήμης, εντοπισμός memory leaks

Εργαλεία Ανάπτυξης Λογισμικού

- Editors
 - Text Editor με κάποια χρήσιμα στοιχεία. (UltraEdit, Notepad++, vim, emacs)
- Source Code Generator
 - Wizards, μετατροπή UML σε κλάσεις ή ΒΔ
- Refactoring Tools
 - Επεκτασιμότητα και Συντηρησιμότητα
 - Αλλάζουμε τη δομή μιας κλάσης

Εργαλεία Ανάπτυξης Λογισμικού

- Code Coverage
 - Έλεγχος κάλυψης χρήσης κώδικα για ένα συγκεκριμένο test
- Bug Tracking Systems
 - Αναφορά σφαλμάτων και διαχείρισή τους
- Documentation Generator
 - Δημιουργία τεκμηρίωσης βασισμένη στον πηγαίο κώδικα

Εργαλεία Ανάπτυξης Λογισμικού

- Revision Control – Version Control
 - Διαχείριση εκδόσεων (git, svn, cvs, clearcase)
- Build Tools - Integration Tools
 - Αυτοματοποίηση build (make, ant, autotools)
- Unit Testing
 - Αυτοτελής έλεγχος μονάδων κώδικα (κλάση, πακέτο ή μέρος προγράμματος).

Εργαλεία Ανάπτυξης Λογισμικού

- GUI Generator
 - Εισαγωγή στοιχείων γραφικού περιβάλλοντος στο πρόγραμμα με γραφικό τρόπο (WYSIWYG). : QT, Java μηχανές κτλ).

Δομή Παρουσίασης

- Ιστορικό Ανάπτυξης Λογισμικού
- Εργαλεία Ανάπτυξης
- **Integrated Development Environments (IDE)**
- Eclipse IDE
- Git

Integrated Development Environment

- Πολλά εργαλεία
- Αυξάνεται η πολυπλοκότητα
- Γιατί να μην είναι όλα ενσωματωμένα;
- Εγένετω : **IDE**

Integrated Development Environment

- Dartmouth BASIC : Πρώτη εφαρμογή σαν έννοια.
- Maestro I (1975) : Πρώτο σύστημα
 - 22.000 χρήστες
- Borland Turbo Pascal
- Borland Delphi
- Visual Studio

Integrated Development Environment

- Visual Programming : Δημιουργία GUI, κώδικα μέσω UML ή αυτόματο formatting.
- Υποστήριξη γλωσσών : Τα σύγχρονα IDE υποστηρίζουν πολλαπλές γλώσσες προγραμματισμού.
- Πολλές διαθέσιμες πλατφόρμες : Διαθέσιμο σε Windows, Linux, MacOS
- Cloud : online IDE

Δομή Παρουσίασης

- Ιστορικό Ανάπτυξης Λογισμικού
- Εργαλεία Ανάπτυξης
- Integrated Development Environments (IDE)
- **Eclipse IDE**
- Git

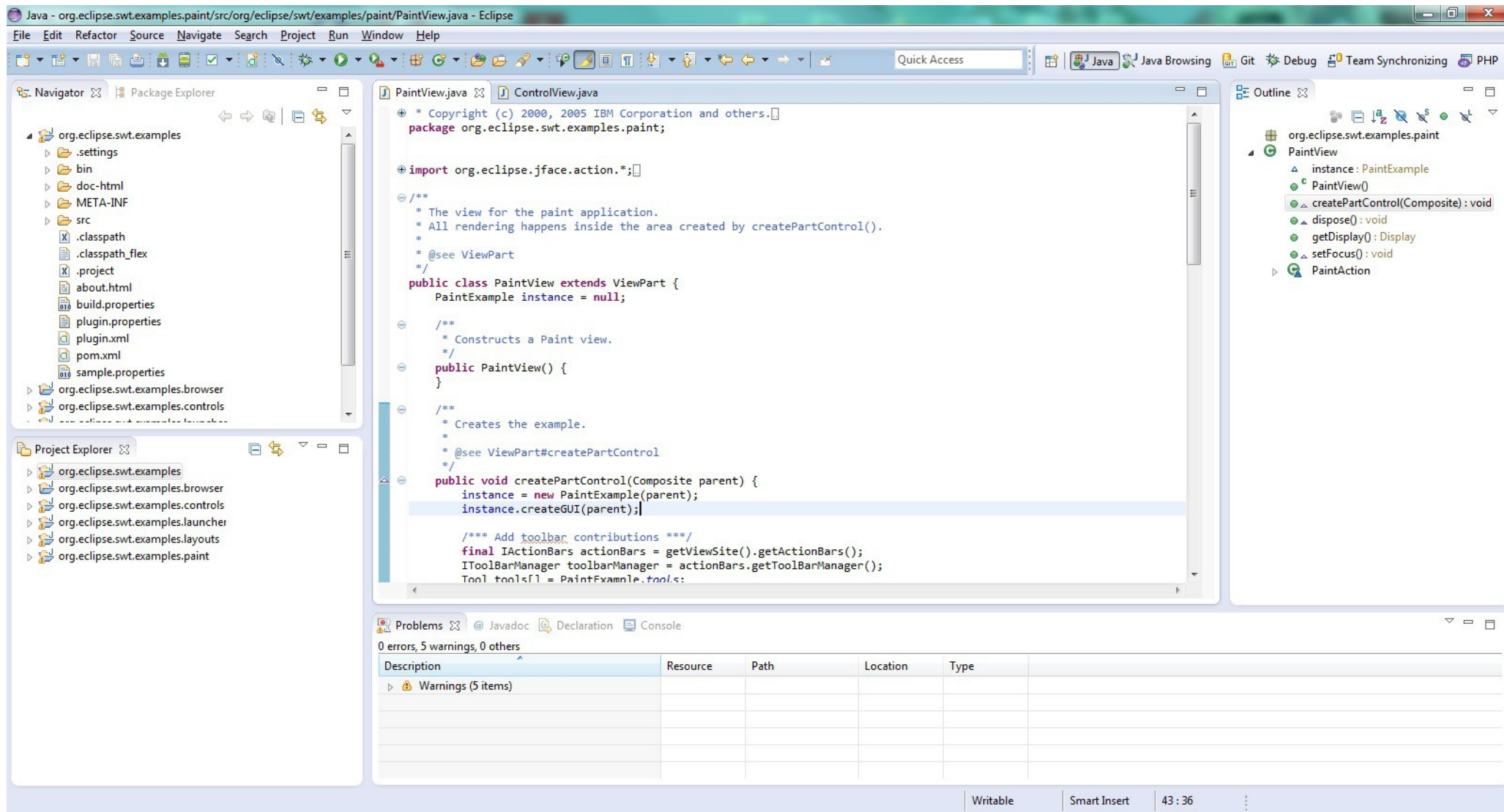
Eclipse

- Προήλθε από project της IBM
- Νοέμβριος 2001 : Δημιουργήθηκε η κοινοπραξία (Borland, IBM, Red Hat, SuSe)
- Ιανουάριος 2004 : Eclipse Foundation
- Ιούνιος 2004 : Έκδοση 3.0
- Ιούνιος 2013 : Έκδοση 4.3 (Kepler)

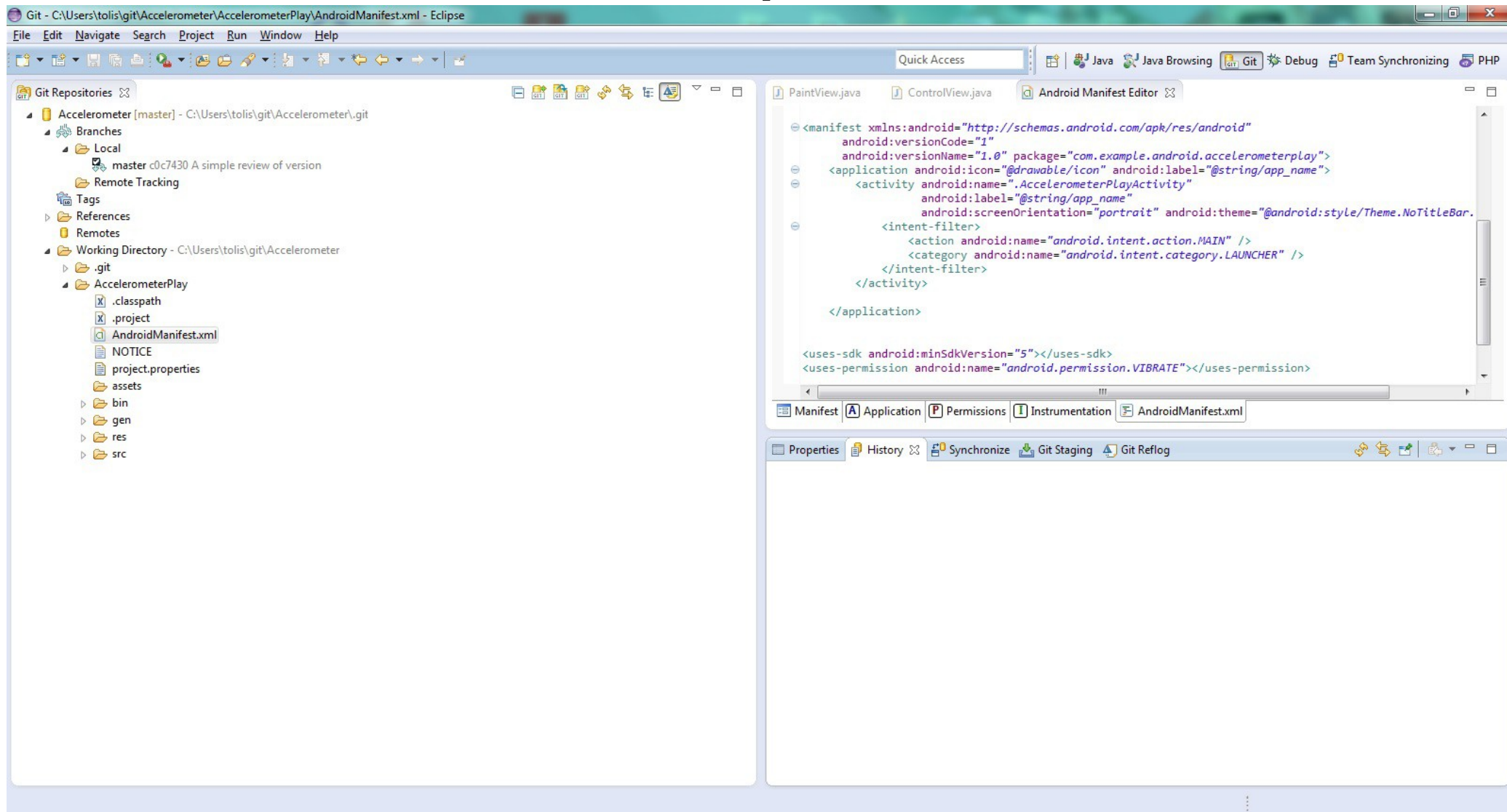
Eclipse

- Βασικό Περιβάλλον Εργασίας
- Επεκτάσιμο μέσω plug-in συστήματος
- Υποστηρίζει πολλές γλώσσες
- Built-in debugger
- Version Control
- Refactoring
- GUI development

Eclipse



Eclipse

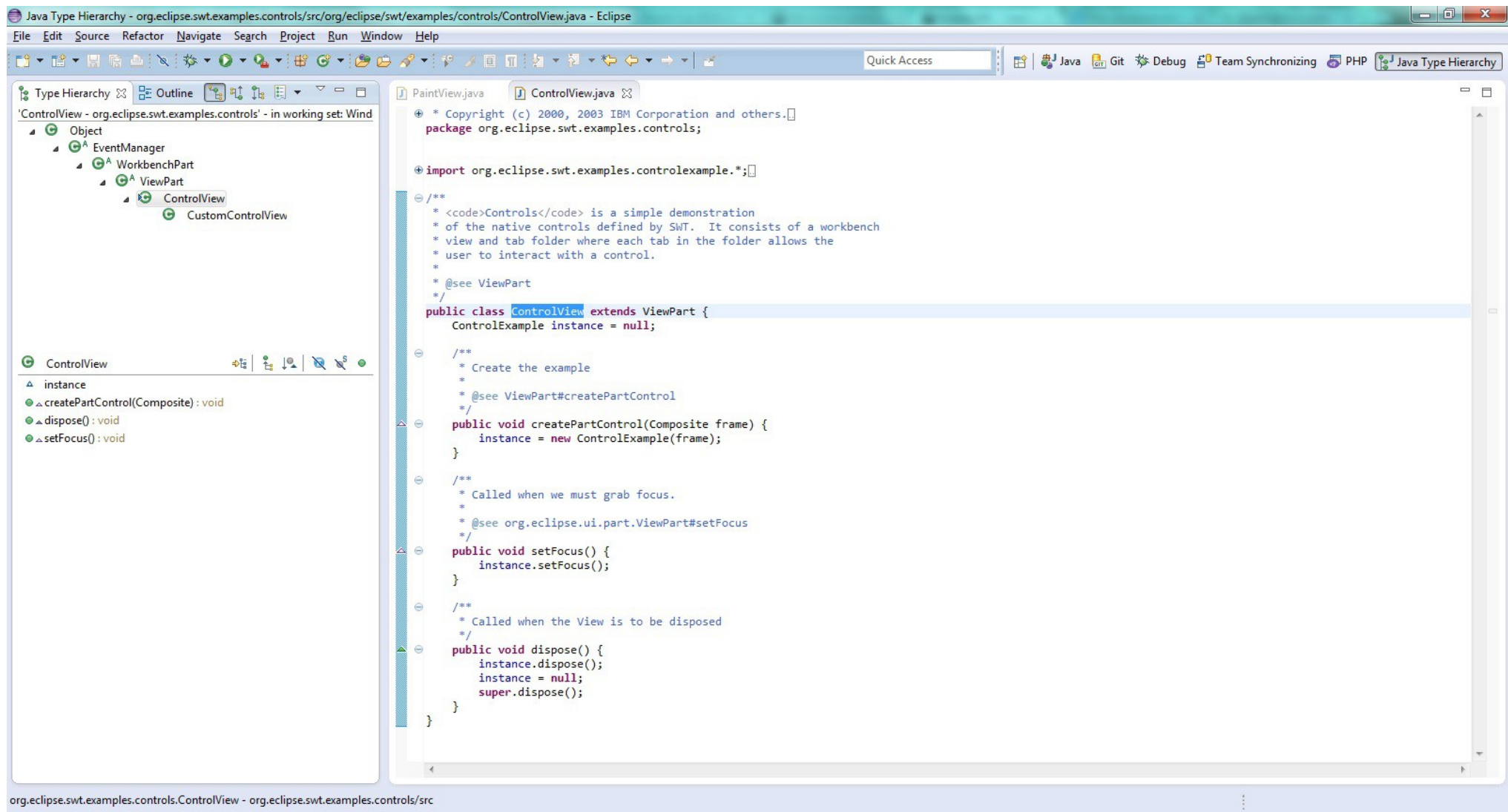


Eclipse

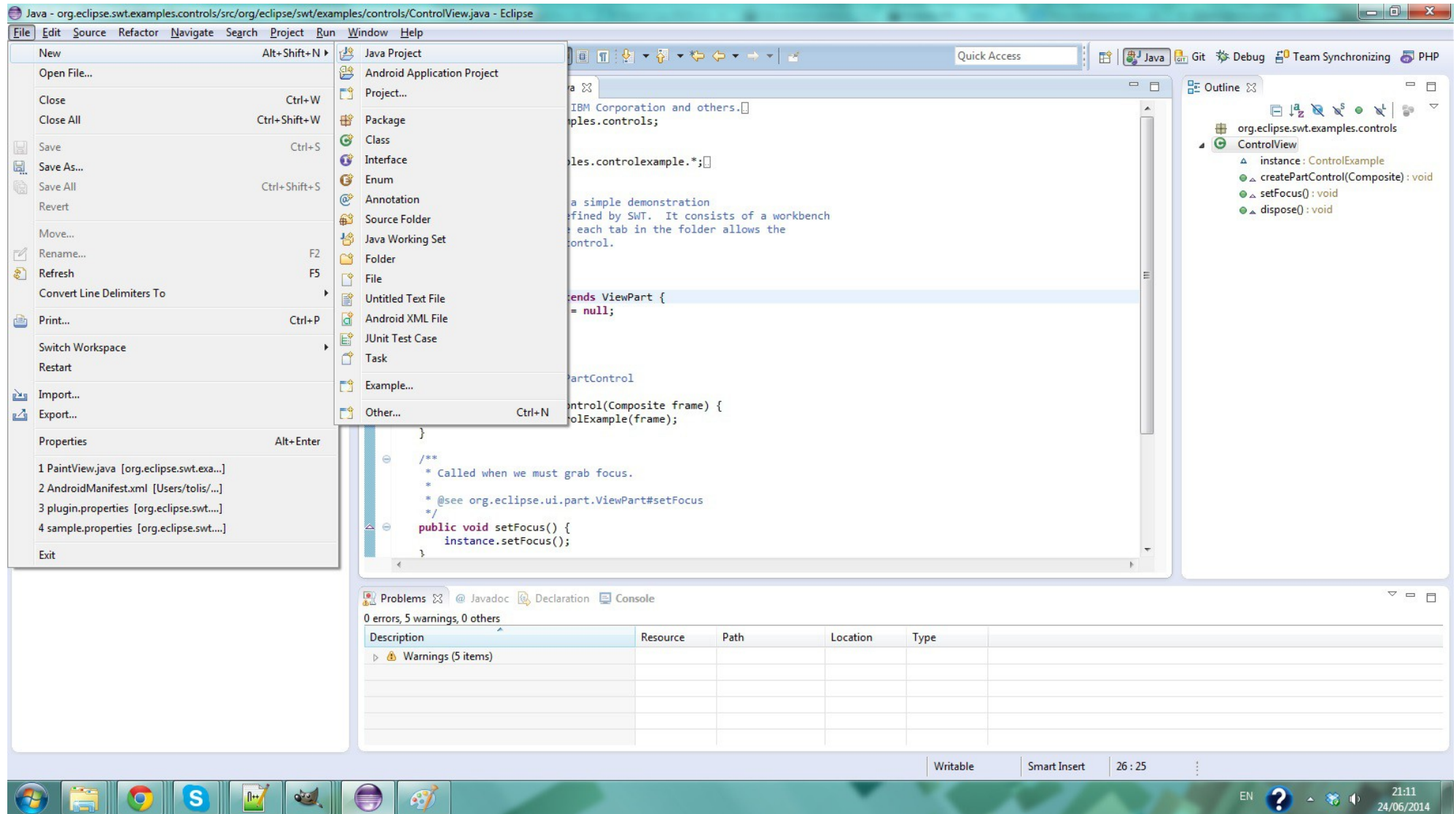
The screenshot displays the Eclipse IDE interface with the following components:

- Top Bar:** Shows the current project and file: `Debug - org.eclipse.swt.examples.paint/src/org/eclipse/swt/examples/paint/PaintView.java - Eclipse`. The menu bar includes `File`, `Edit`, `Source`, `Refactor`, `Navigate`, `Search`, `Project`, `Run`, `Window`, and `Help`.
- Toolbar:** Contains various icons for file operations, editing, and running. A `Quick Access` search bar is also present.
- Left Sidebar:** Includes tabs for `Variables`, `Breakpoints`, `Expressions`, and `Interactive Console`. The `Breakpoints` tab is active, showing a breakpoint at `PaintView [line: 50] - createPartControl(Composite)`.
- Editor Area:** Displays the source code of `PaintView.java`. The code includes comments and a `createPartControl` method. The line `toolbarManager.add(new GroupMarker(group));` is highlighted.
- Right Sidebar:** Contains the `Outline` view, showing the project structure and the `PaintView` class with its methods: `instance: PaintExample`, `PaintView()`, `createPartControl(Composite): void`, `dispose(): void`, `getDisplay(): Display`, `setFocus(): void`, and `PaintAction`.
- Bottom Left:** The `LogCat` view shows a list of messages. The current message is: `D 06-24 21:07:4... 1464 1490 BatterySer... update start`.
- Bottom Right:** The `Debug Output` view is active, showing the output of the debug session.

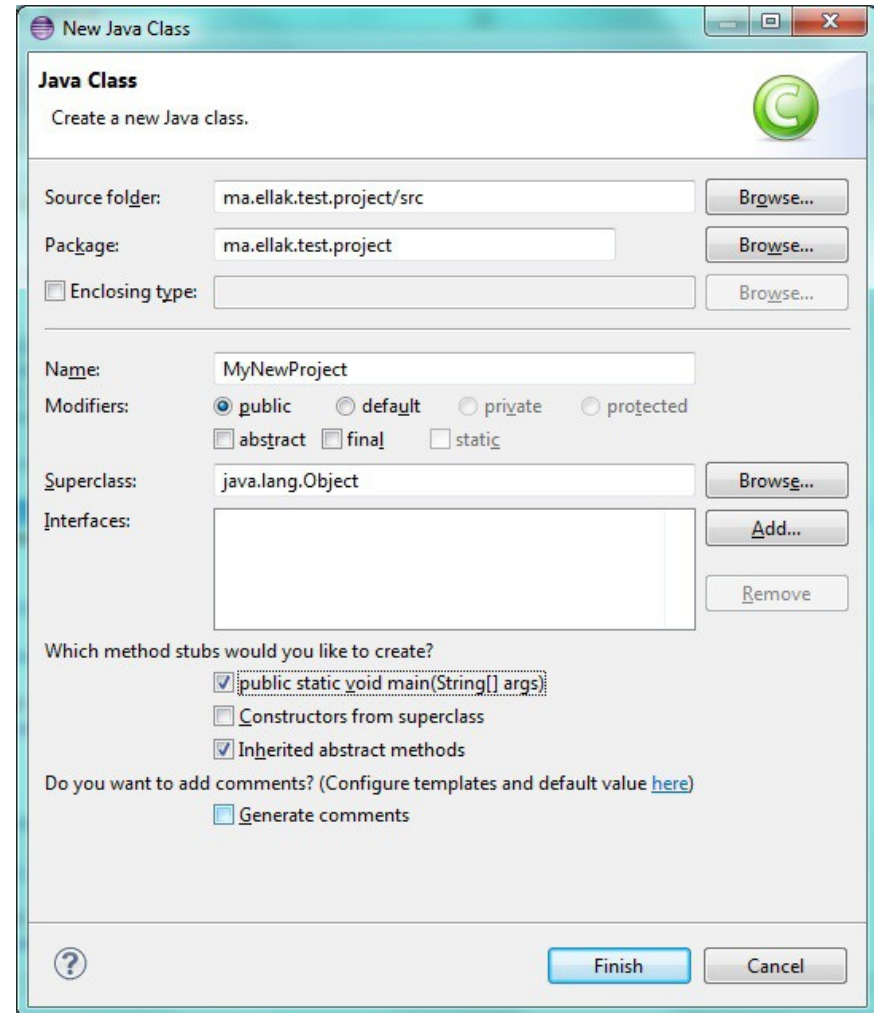
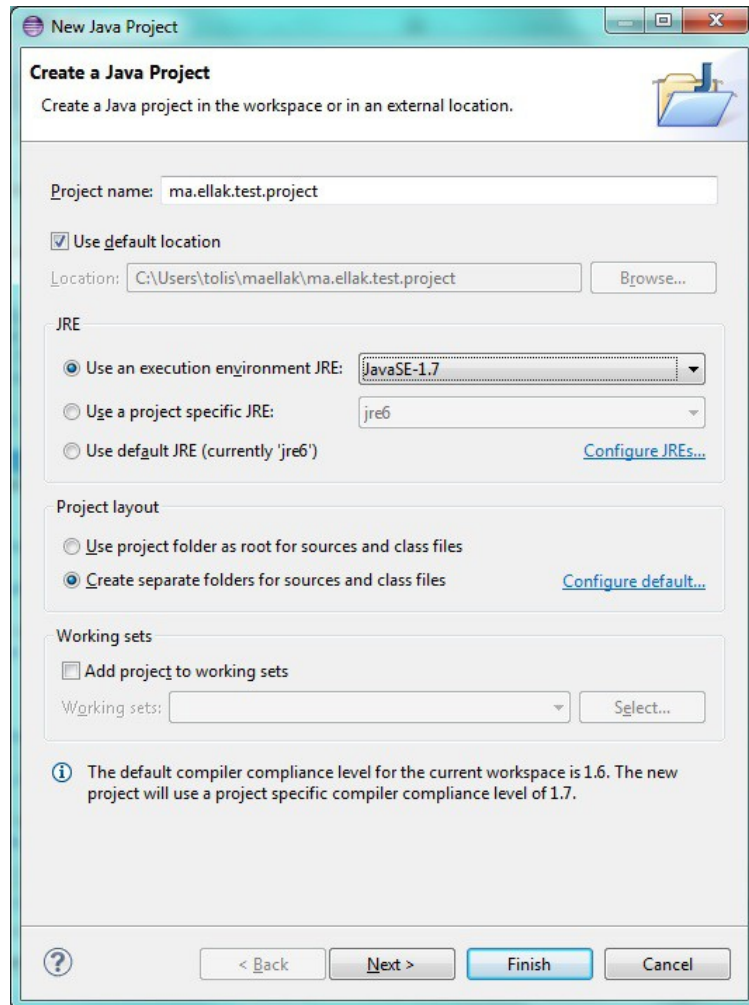
Eclipse



Eclipse



Eclipse



Eclipse

Java - ma.ellak.test.project/src/ma/ellak/test/project/MyJFrame.java - Eclipse

File Edit Refactor Source Navigate Search Project Run Window Help

Quick Access

Java Git Debug Team Synchronizing PHP

Navigator Package Explorer

ma.ellak.test.project

- .settings
- bin
- src
 - ma
 - ellak
 - test
 - project
 - MyJFrame.java
 - Mytestproject.java
- wbp-meta
 - org.eclipse.wb.swing.wbp-palette.xml
 - .classpath
 - .project
- org.eclipse.swt.examples
 - org.eclipse.swt.examples.browser
 - org.eclipse.swt.examples.controls
 - org.eclipse.swt.examples.layouts
 - org.eclipse.swt.examples.paint

Project Explorer

ma.ellak.test.project

- src
 - ma.ellak.test.project
 - MyJFrame.java
 - Mytestproject.java
 - JRE System Library [JavaSE-1.6]
 - wbp-meta
 - org.eclipse.swt.examples
 - org.eclipse.swt.examples.browser
 - org.eclipse.swt.examples.controls
 - org.eclipse.swt.examples.layouts
 - org.eclipse.swt.examples.paint

Structure

Components

- (javax.swing.JFrame)
 - contentPane

Properties

Variable	Value
Layout	(java.awt.BorderLayout)
Class	javax.swing.JPanel
background	240,240,240
border	EmptyBorder
foreground	0,0,0
tab order	
toolTipText	

Palette

System

- Selection
- Marquee
- Choose co...
- Tab Order

Containers

- JPanel
- JScrollPane
- JSplitPane
- JTabbedPane
- JToolBar
- JLayeredPa...
- JDesktopPa...
- JInternalFra...

Layouts

- Absolute la...
- FlowLayout
- BorderLayo...
- GridLayout
- GridBagLay...
- CardLayout
- BoxLayout
- SpringLayout
- FormLayout
- MigLayout

Struts & Springs

Components

- JLabel
- JTextField
- JComboBox
- JButton
- JCheckBox
- JRadioButton
- JToggleBut...
- JTextArea

Outline

ma.ellak.test.prc

- MyJFrame
 - contentPan
 - main(String
 - new Ri
 - MyJFrameC

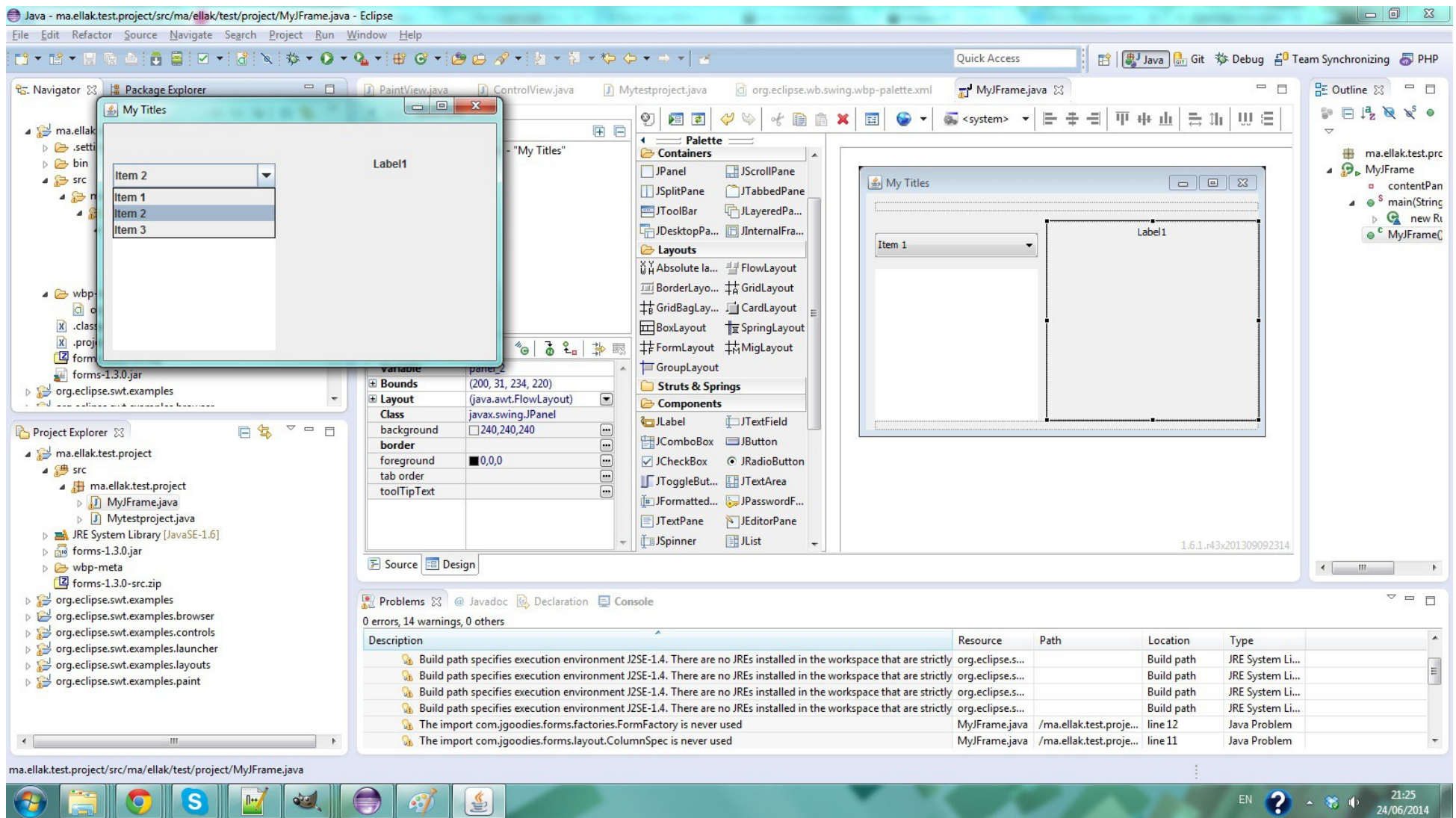
Problems

0 errors, 6 warnings, 0 others

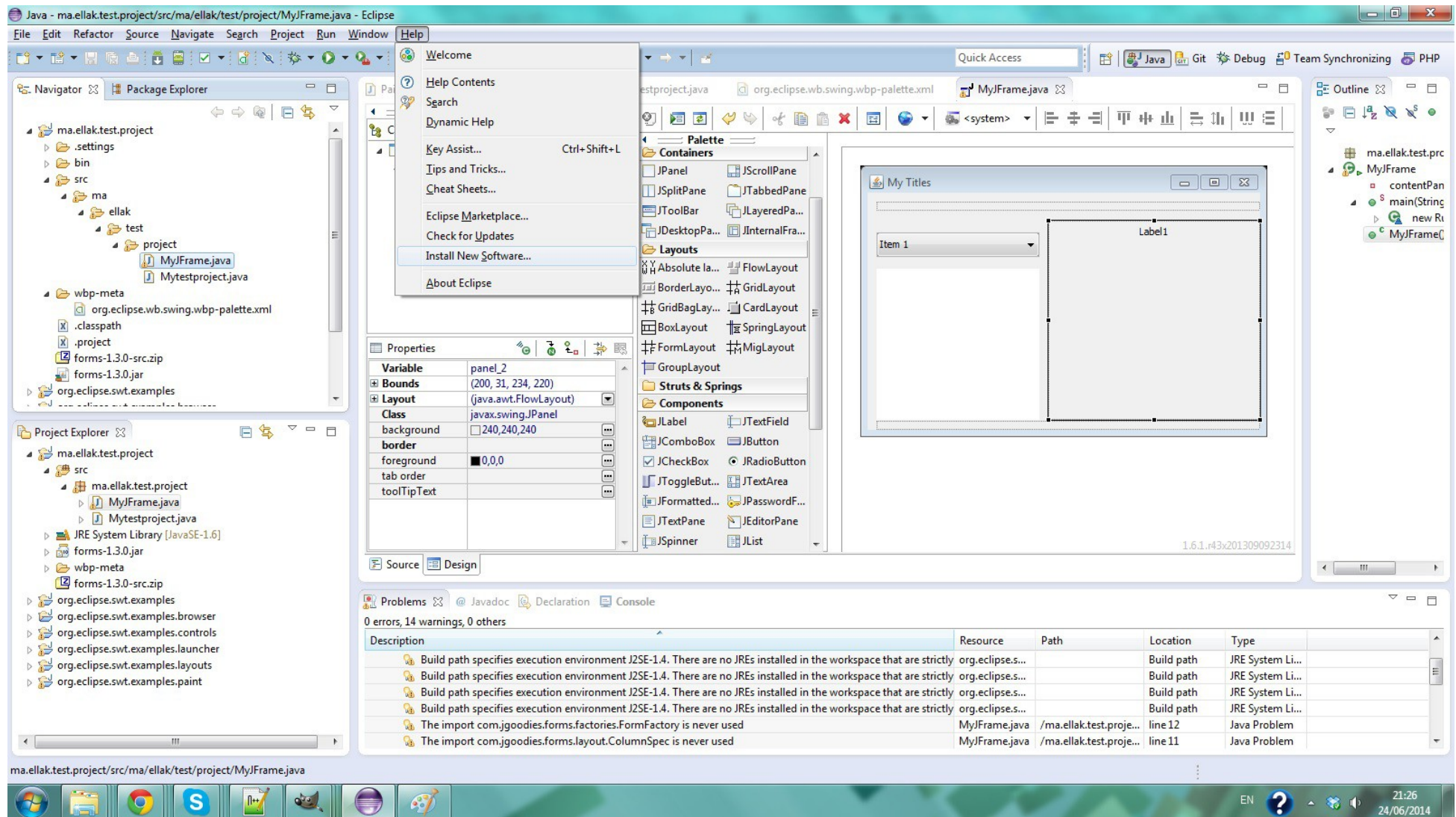
Description	Resource	Path	Location	Type
Build path specifies execution environment J2SE-1.4. There are no JREs installed in the workspace that are strictly	org.eclipse.s...		Build path	JRE System Li...
Build path specifies execution environment J2SE-1.4. There are no JREs installed in the workspace that are strictly	org.eclipse.s...		Build path	JRE System Li...
Build path specifies execution environment J2SE-1.4. There are no JREs installed in the workspace that are strictly	org.eclipse.s...		Build path	JRE System Li...
Build path specifies execution environment J2SE-1.4. There are no JREs installed in the workspace that are strictly	org.eclipse.s...		Build path	JRE System Li...
The serializable class MyJFrame does not declare a static final serialVersionUID field of type long	MyJFrame.java	/ma.ellak.test.proje...	line 10	Java Problem

Writable Smart Insert 36 : 23

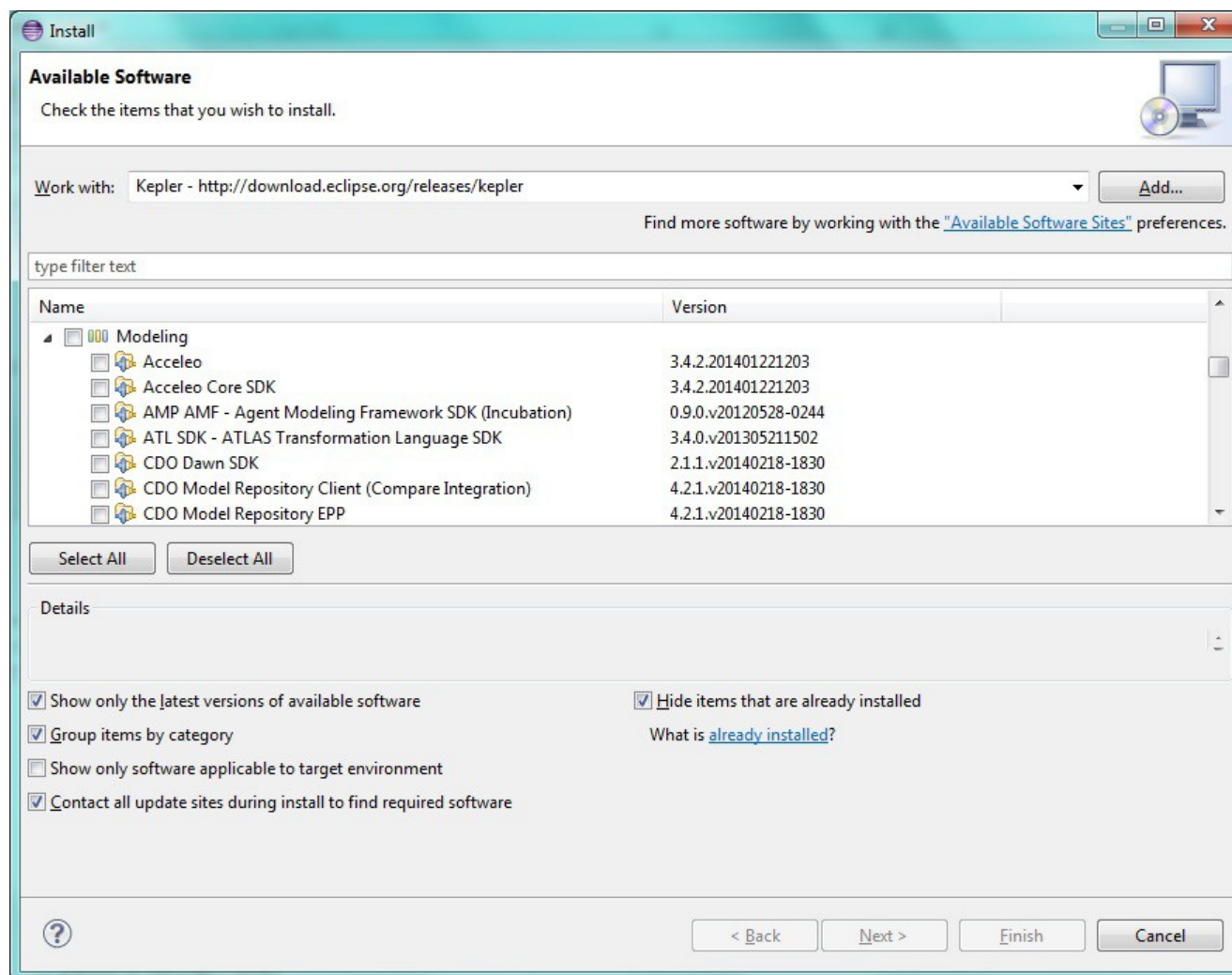
Eclipse



Eclipse



Eclipse



Δομή Παρουσίασης

- Ιστορικό Ανάπτυξης Λογισμικού
- Εργαλεία Ανάπτυξης
- Integrated Development Environments (IDE)
- Eclipse IDE
- **Git**

GIT

version control system

- Ένα σύστημα που καταγράφει τις αλλαγές που γίνονται σε κάποια αρχεία (τι άλλαξε και από ποιόν) στο πέρασμα του χρόνου

Διάφορα VCS

- CVS
- SVN
- Mercurian
- **Git**

Git

- Πρώτη έκδοση 2007
 - από Linus Torvalds (Linux kernel)
- Χρησιμοποιείται για ανάπτυξη:
 - Linux Kernel
 - Android
 - VLC
 - Gcc
 - List: <https://git.wiki.kernel.org/index.php/GitProjects>

Git features

- Atomicity (ατομικότητα):
 - Μια αλλαγή είτε θα επιτύχει είτε θα αποτύχει
 - πχ, δεν θα αποθηκευτούν οι μισές από τις αλλαγές που έγιναν σε ένα αρχείο
- Security (ασφάλεια):
 - Δεν μπορεί κάποιος έξω από το git να αλλάξει κάποια παλιά έκδοση κάποιου αρχείου
 - πχ, δεν μπορούν να γίνουν αλλαγές που δεν φαίνεται ποιος τις έκανε.

Εγκατάσταση Git

- Διαθέσιμο για Windows, Linux, Mac OS X
 - <http://git-scm.com/downloads>
- Γραφικό περιβάλλον: git-gui, gitk
 - Επιπλέον: <http://git-scm.com/downloads/guis>

Χρήση Git

Ορολογία

- **repository**: ένας φάκελος που τα περιεχόμενά του διαχειρίζονται από το git
- **unstaged changes**: αλλαγές που έχουν γίνει, αλλά δεν είναι ενημερωμένο το git για αυτές.
- **staged changes**: αλλαγές που έχουν γίνει και έχει ενημερωθεί το git ώστε να τις συμπεριλάβει στο επόμενο commit.
- **commit**: προσθήκη των αλλαγών στο git.
- **check out**: έλεγχος μιας παλαιότερης έκδοσης που είναι αποθηκευμένη στο git
- **reset**: ακύρωση όλων των αλλαγών και επιστροφή σε προηγούμενη έκδοση

Ορολογία

- **clone**: αντιγραφή τω περιεχομένων ενός repository
- **fetch**: μεταφορά αλλαγών από κάποιο repository
- **merge**: συνένωση των αλλαγών από δύο repository
- **pull**: πρώτα fetch, μετά merge
- **push**: μεταφορά των αλλαγών σε κάποιο προορισμό
- **remote**: καθορισμός του απομακρυσμένου repository
- **resolve conflicts**: πως επιλύονται οι διαφορές, όταν δύο χρήστες έχουν κάνει αλλαγές στο ίδιο αρχείο

Αρχικοποίηση

- mkdir test_git
- cd test_git/
- **git init**
 - Μήνυμα: Initialized empty Git repository in /.../test_git/.git/
 - Δημιουργία του φακέλου .git μέσα στον φάκελο test_git

Παραμετροποίηση

- `git config --global user.name "Vasilhs Karavasilis"`
- `git config --local user.name "Vasilhs Karavasilis"`
- `git config --global user.email "vkaravas@cs.uoi.gr"`
- `git config --local user.email "vkaravas@cs.uoi.gr"`
- `git config -l`

Προσθήκη νέου αρχείου - βήμα 1

- `echo "Line 1" >> file.txt`
- **git status**
 - Untracked files:
 - (use "`git add <file>...`" to include in what will be committed)
 - file.txt
- **git add file.txt**
- **git status**
 - Changes to be committed:
 - (use "`git rm --cached <file>...`" to unstage)
 - new file: file.txt
- **Το αρχείο δεν έχει προστεθεί ακόμη (είναι στην κατάσταση staged changes). Θα προστεθεί στο επόμενο commit.**

Προσθήκη πολλών νέων αρχείων

- **git add .**
- **git add *.txt**
- Αν δεν θέλουμε κάποια αρχεία να τα διαχειρίζεται το git:
 - Δημιουργία του αρχείου **.gitignore** με περιεχόμενα:
 - ~*.doc
 - Το git θα αγνοήσει αρχεία που υπάρχουν μέσα στον φάκελο και αρχίζουμε με ~ και η κατάληξή τους είναι doc.

Αναίρεση της προσθήκης αρχείου

- git status
 - Changes to be committed:
 - (use "git rm --cached <file>..." to unstage)
 - new file: file.txt
- **git reset file.txt**
- git status
 - Untracked files:
 - (use "git add <file>..." to include in what will be committed)
 - file.txt

Προσθήκη νέου αρχείου – βήμα 2

- **git status**
 - Changes to be committed:
 - (use "git rm --cached <file>..." to unstage)
 - new file: file.txt
- **git commit -m "initial commit"**
 - [master (root-commit) cb30931] initial commit
 - 1 file changed, 1 insertion(+)
 - create mode 100644 file.txt
- **git status**
 - On branch master
 - nothing to commit, working directory clean

Αλλαγή στο αρχείο

- **echo "Line 2" >> file.txt**
- **git status**
 - Changes not staged for commit:
 - (use "git add <file>..." to update what will be committed)
 - (use "git checkout -- <file>..." to discard changes in working directory)
 - modified: file.txt
- **git add *.txt**
- **git commit -m "second commit"**
 - [master f3c1f47] second commit
 - 1 file changed, 1 insertion(+)
- **git status**
 - On branch master
 - nothing to commit, working directory clean

Έλεγχος παλαιότερης έκδοσης

- **git log**

- commit **f3c1f474e197c603edb4806f26e2b2308b8b51cf**
- Author: Vasilhs Karavasilis <vkaravas@cs.uoi.gr>
- Date: Mon Jun 23 16:28:00 2014 +0300
- second commit
- commit **cb30931ce86e69f207a057b0445c8eba2fb85656**
- Author: Vasilhs Karavasilis <vkaravas@cs.uoi.gr>
- Date: Mon Jun 23 16:05:55 2014 +0300
- initial commit

- **git checkout <commit_id>**

- Πάμε σε παλαιότερη έκδοση

- **git checkout master**

- Πάμε στην πιο πρόσφατη έκδοση

Παράδειγμα

- `cat file.txt`
 - Line 1
 - Line 2
- **`git checkout`**
`cb30931ce86e69f207a057b0445c8eba2fb85656`
- `cat file.txt`
 - Line 1
- **`git checkout master`**
- `cat file.txt`
 - Line 1
 - Line 2

Ακύρωση όλων των αλλαγών (reset)

- git log

- commit **f3c1f474e197c603edb4806f26e2b2308b8b51cf**
- Author: Vasilhs Karavasilis <vkaravas@cs.uoi.gr>
- Date: Mon Jun 23 16:28:00 2014 +0300
- second commit
- commit **cb30931ce86e69f207a057b0445c8eba2fb85656**
- Author: Vasilhs Karavasilis <vkaravas@cs.uoi.gr>
- Date: Mon Jun 23 16:05:55 2014 +0300
- initial commit

- git reset --hard

cb30931ce86e69f207a057b0445c8eba2fb85656

- HEAD is now at cb30931 initial commit

- git log

- commit **cb30931ce86e69f207a057b0445c8eba2fb85656**
- Author: Vasilhs Karavasilis <vkaravas@cs.uoi.gr>
- Date: Mon Jun 23 16:05:55 2014 +0300
- initial commit

Remote Repository

- GitHub
- Bitbucket
- Codaset

Δημιουργία remote repository από ΤΟΠΙΚΟ

- **git remote add origin https://github.com/ioa-ma-ellak/test1.git**
 - Δεν κάνει κάποια μεταφορά, απλώς καταχωρεί ότι το remote repository είναι το συγκεκριμένο
 - Χρήση του https και όχι του ssh (θέλει κλειδιά)
- **git push -u origin master**
 - Μεταφέρει τα δεδομένα από το τοπικό repository στο github
 - Θα ζητήσει username / password

Δημιουργία ενός άλλου τοπικού repository από κάποιο remote

- **git clone https://github.com/ioa-ma-ellak/test1.git test_git2**
 - Θα δημιουργήσει έναν άλλο φάκελο με το όνομα test3 και μέσα θα βάλει:
 - 1) Τα αρχεία του remote repository
 - 2) Το .git φάκελο (δηλαδή τις προηγούμενες αλλαγές)

Αλλαγές και προώθηση στο remote repository

- `echo "Line 2" >> file.txt`
- `git add *.txt`
- `git commit -m "Commit line 2"`
- **git push**
 - Δεν χρειάζεται επιπλέον ορίσματα η push
 - **Μπορεί να αποτύχει**, αν και κάποιος άλλος έχει αλλάξει το ίδιο αρχείο με εμάς και έχει προλάβει να κάνει πρώτος push.
 - Πρέπει πρώτα να κάνουμε git pull

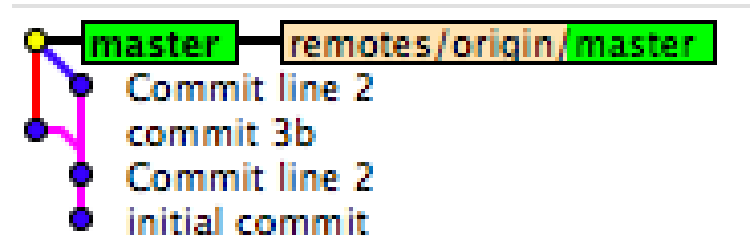
Ανανέωση των τοπικών αλλαγών από το remote repository

- **git pull**

- Φέρνει τα δεδομένα από το remote repository στο τοπικό repository
- Μπορεί να υπάρχει πρόβλημα (πχ και εμείς και κάποιος άλλος να έχει κάνει αλλαγές στο ίδιο αρχείο).
 - Line 1
 - Line 2
 - <<<<<< HEAD
 - **Line 3b**
 - =====
 - **Line 3α**
 - >>>>>> 92c5263199367f217a2e8ec6a2bffa02c1c625bb
- Η πρώτη γραμμή είναι η δική μας, η δεύτερη γραμμή είναι αυτή που υπάρχει στο remote repository

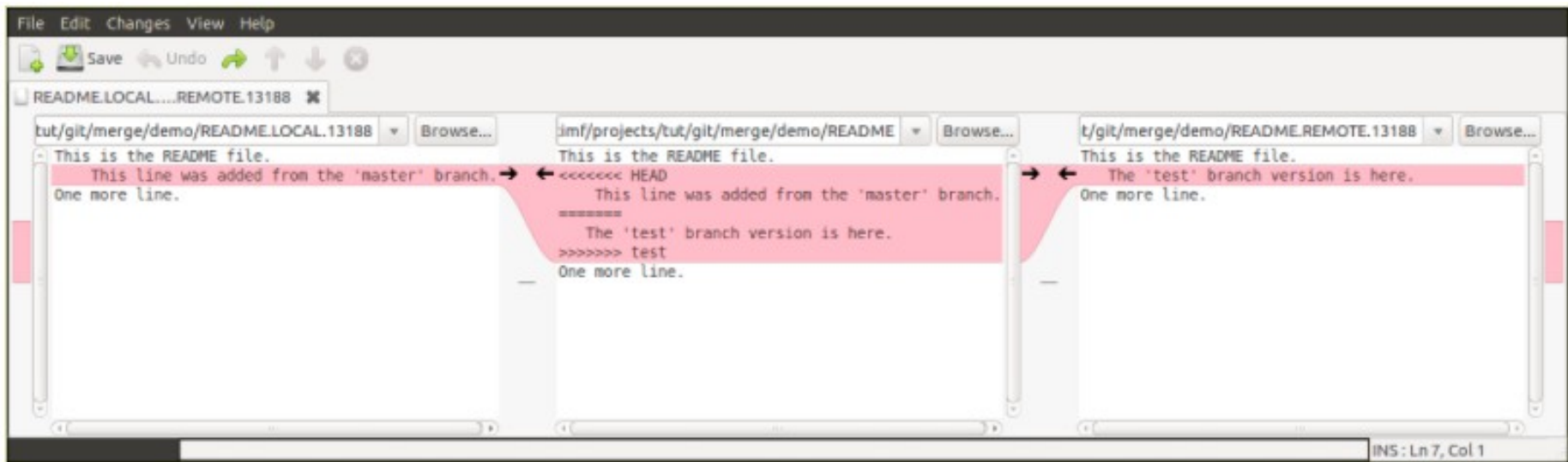
Resolve conflicts

- 1)Κρατάμε όλες τις αλλαγές
 - 2)Κρατάμε μόνο τις δικές μας αλλαγές
 - 3)Κρατάμε μόνο τις αλλαγές του remote repository
 - 4)Κρατάμε κάποιες από το ένα και κάποιες από το άλλο
 - 5)Δεν κρατάμε τίποτα
- Στην συνέχεια κάνουμε **git add ***, **git commit -m “...”**, **git push** ξανά
 - **git log –graph**
 - Βλέπουμε τι έγινε στην συνένωση



Resolve conflicts

- Υπάρχουν και γραφικές διεπαφές (πχ kdiff3, meld)
- Μπορούμε να κάνουμε το git να ανοίγει κατευθείαν πχ το meld:
 - `git mergetool -t meld`



Διαγραφή αρχείου

- **git rm file.txt**
- **git commit -m "remove file.txt"**
- **git push**

Άλλα θέματα

- branch
- hooks
- tags
- bare repository

Ανακεφαλαίωση

- git add *
- git commit -m “new commit”
- git pull
- git push

Ερωτήσεις