



Agent and Object Technology Lab  
Dipartimento di Ingegneria dell'Informazione  
Università degli Studi di Parma



# Ingegneria del software A

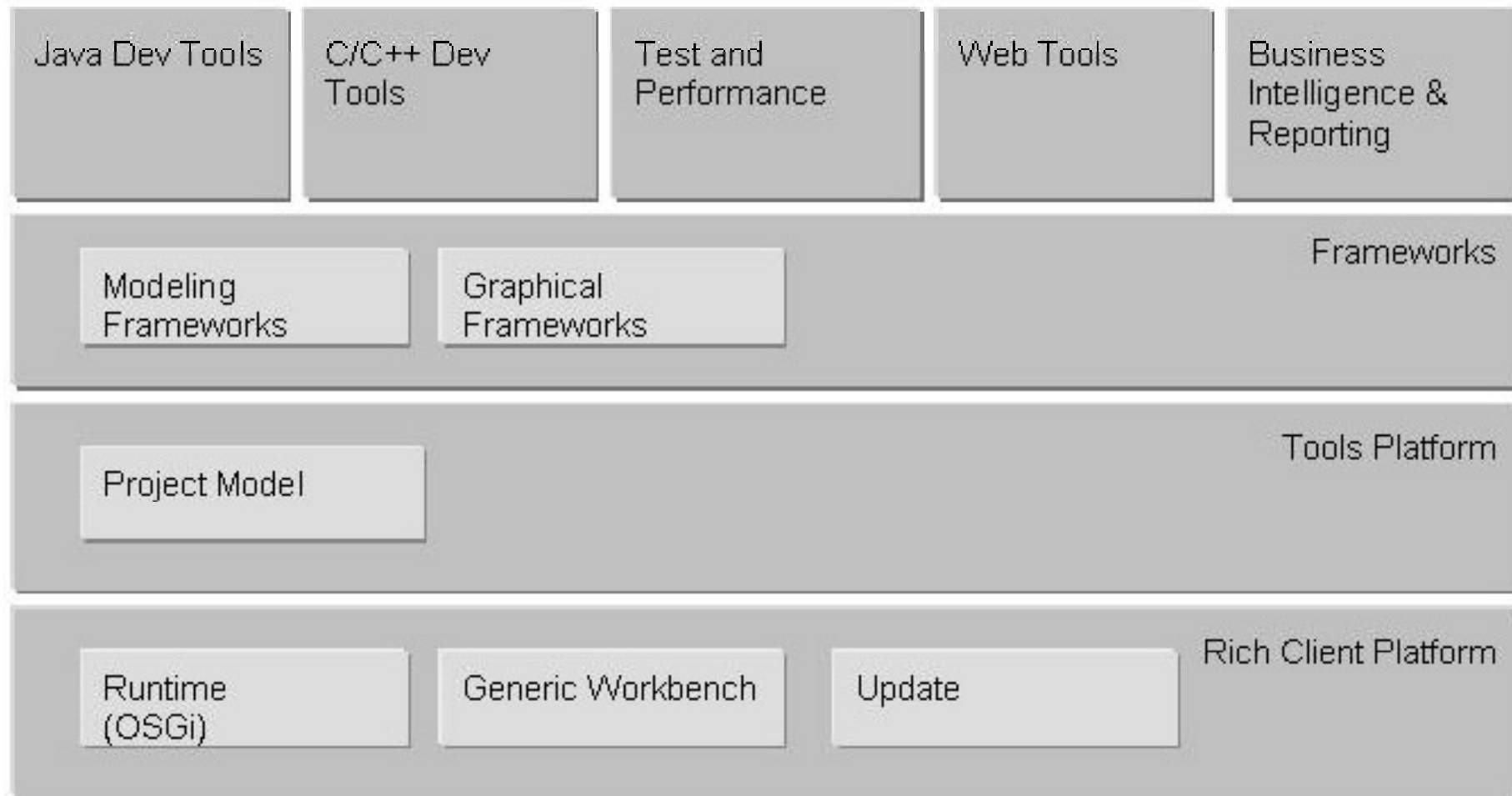


**Prof. Agostino Poggi**

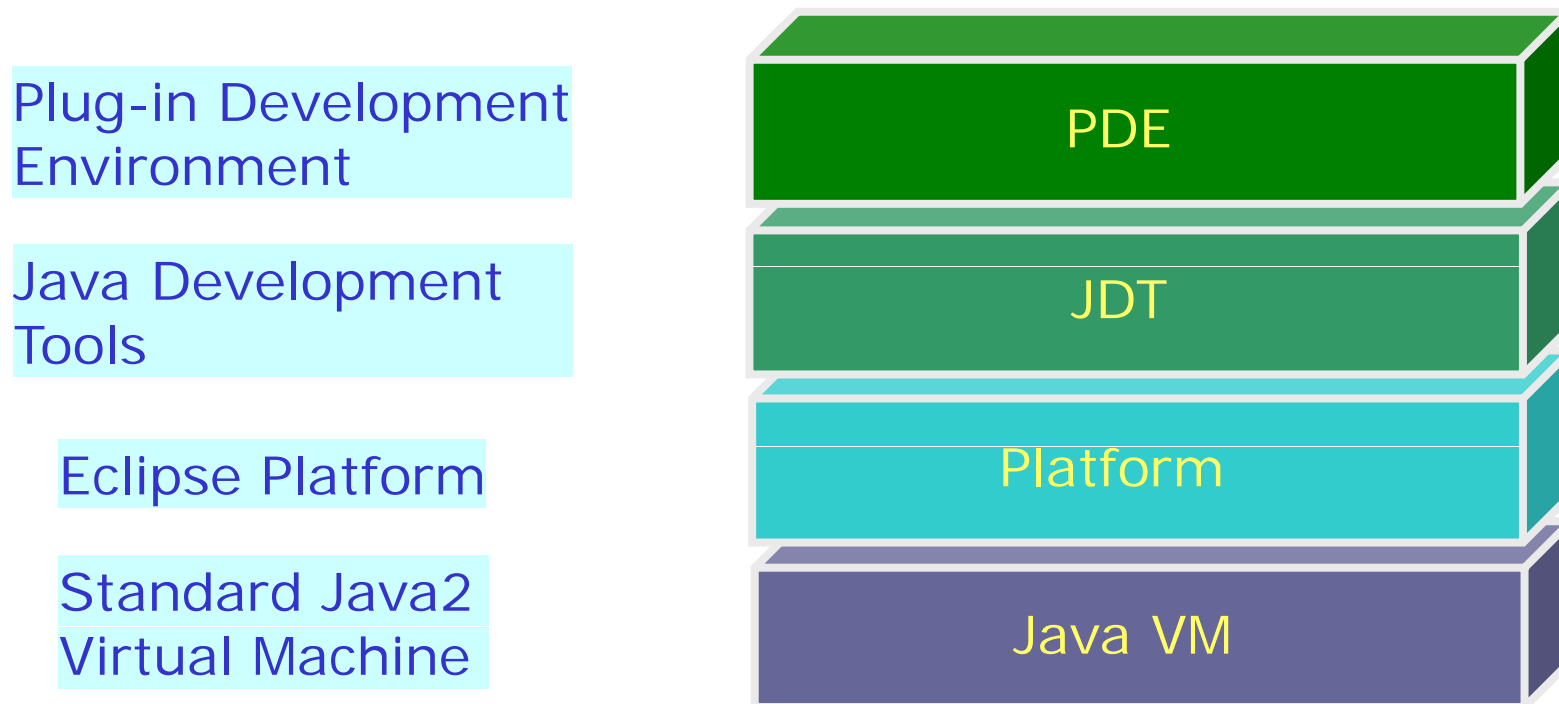
- ◆ Eclipse è un ambiente di sviluppo aperto che è stato creato da IBM partendo in parte dai prodotti VisualAge (Smalltalk e Java IDE)
- ◆ La versione attuale è chiamata Galileo (3.5.1, 25 Settembre 2009) ed è scaricabile dal sito ufficiale: [www.eclipse.org](http://www.eclipse.org)
- ◆ Eclipse è un prodotto open source rilasciato con la licenza EPL (Eclipse Public License)

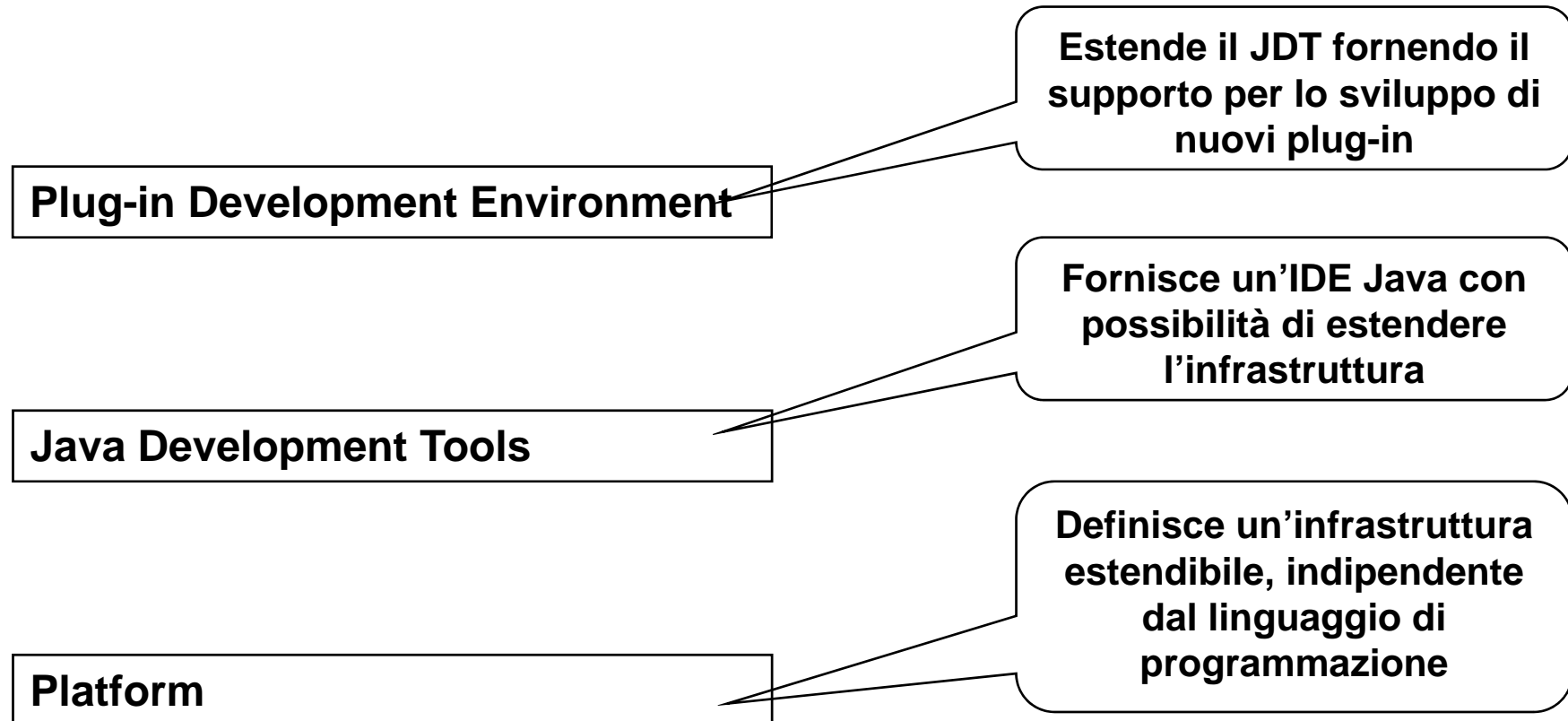
*In many ways Eclipse is the Emacs for the 21st century*

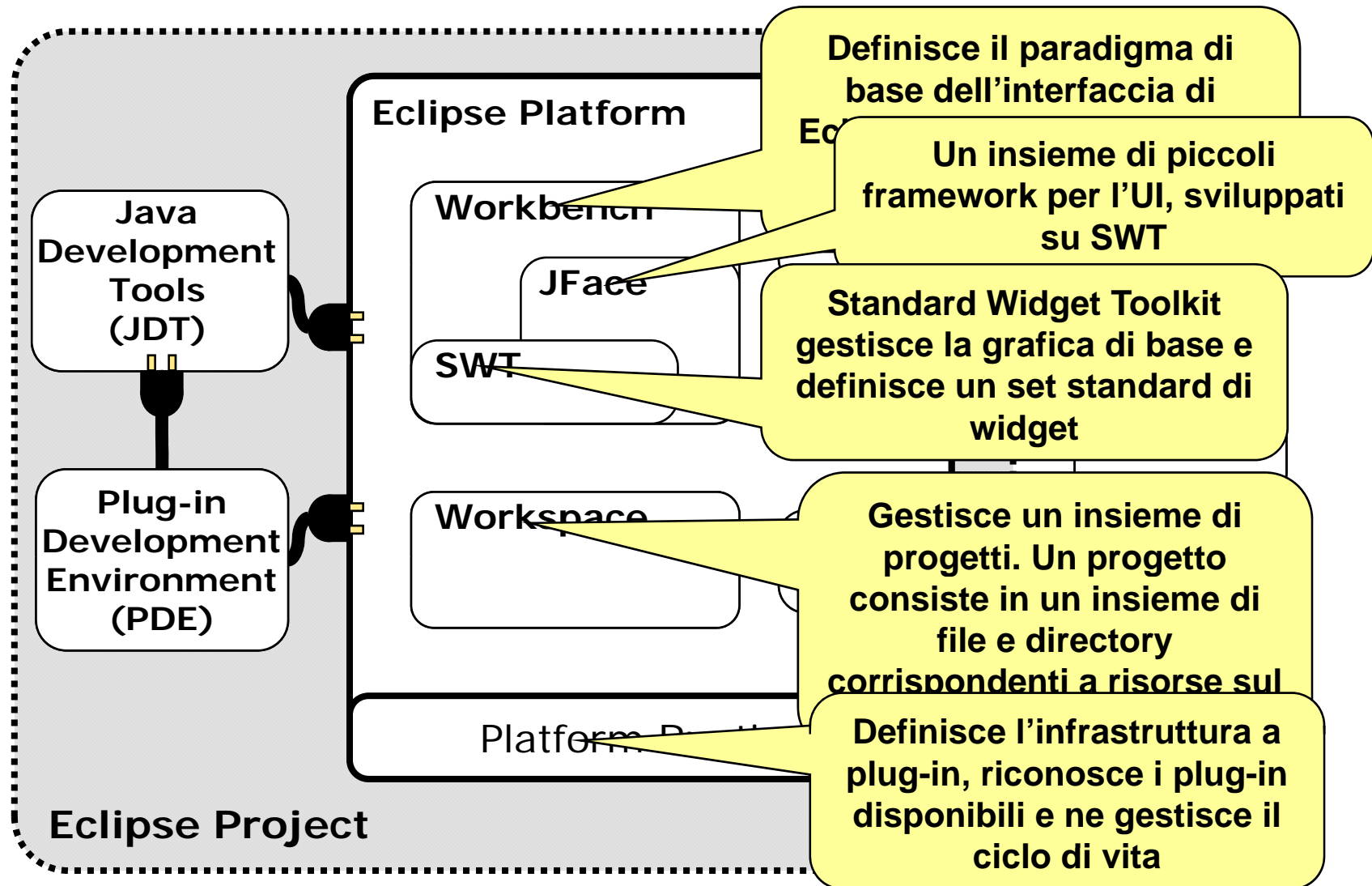
- ◆ Piattaforma aperta per lo sviluppo di applicativi
  - Utilizzabile sul più alto numero di sistemi operativi
  - Utilizzabile con un'interfaccia grafica o senza (GUI o non-GUI)
- ◆ Indipendente dal linguaggio di programmazione
  - Gestire in modo completo vari tipi di linguaggi: HTML, Java, C/C++, XML, JSP, EJB ...
- ◆ Facilitare l'integrazione di tool specifici
  - Possibilità di inserire nuovi tool al prodotto di base per venire incontro alle esigenze del singolo sviluppatore



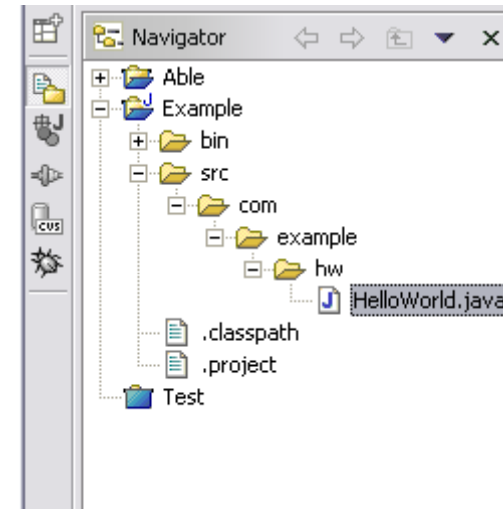
- ♦ Eclipse è una piattaforma universale per integrare tool di sviluppo
- ♦ Ha un'architettura aperta, estendibile, basata su plug-in





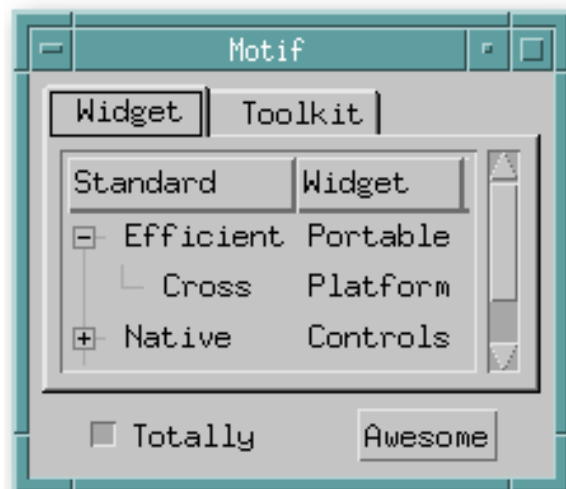
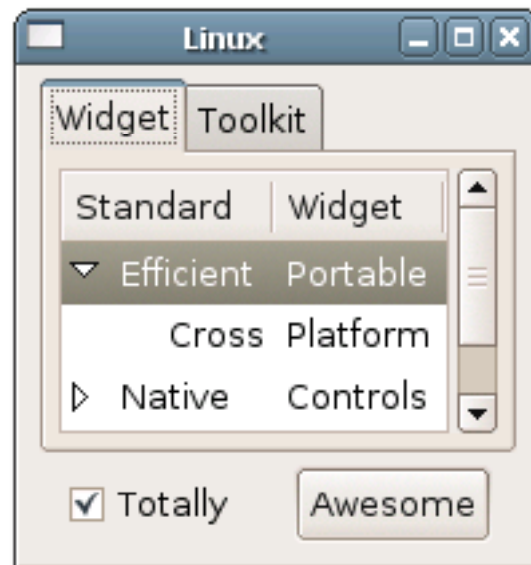


- ◆ Permette di accedere a risorse
  - Files, directory o progetti
- ◆ Il workspace è in grado di gestire 1 o più progetti
  - I progetti corrispondono a directory nel file system
- ◆ Vista navigator
  - Albero di file e directory
- ◆ Vari tool permettono di leggere, creare, modificare, cancellare risorse nel workspace



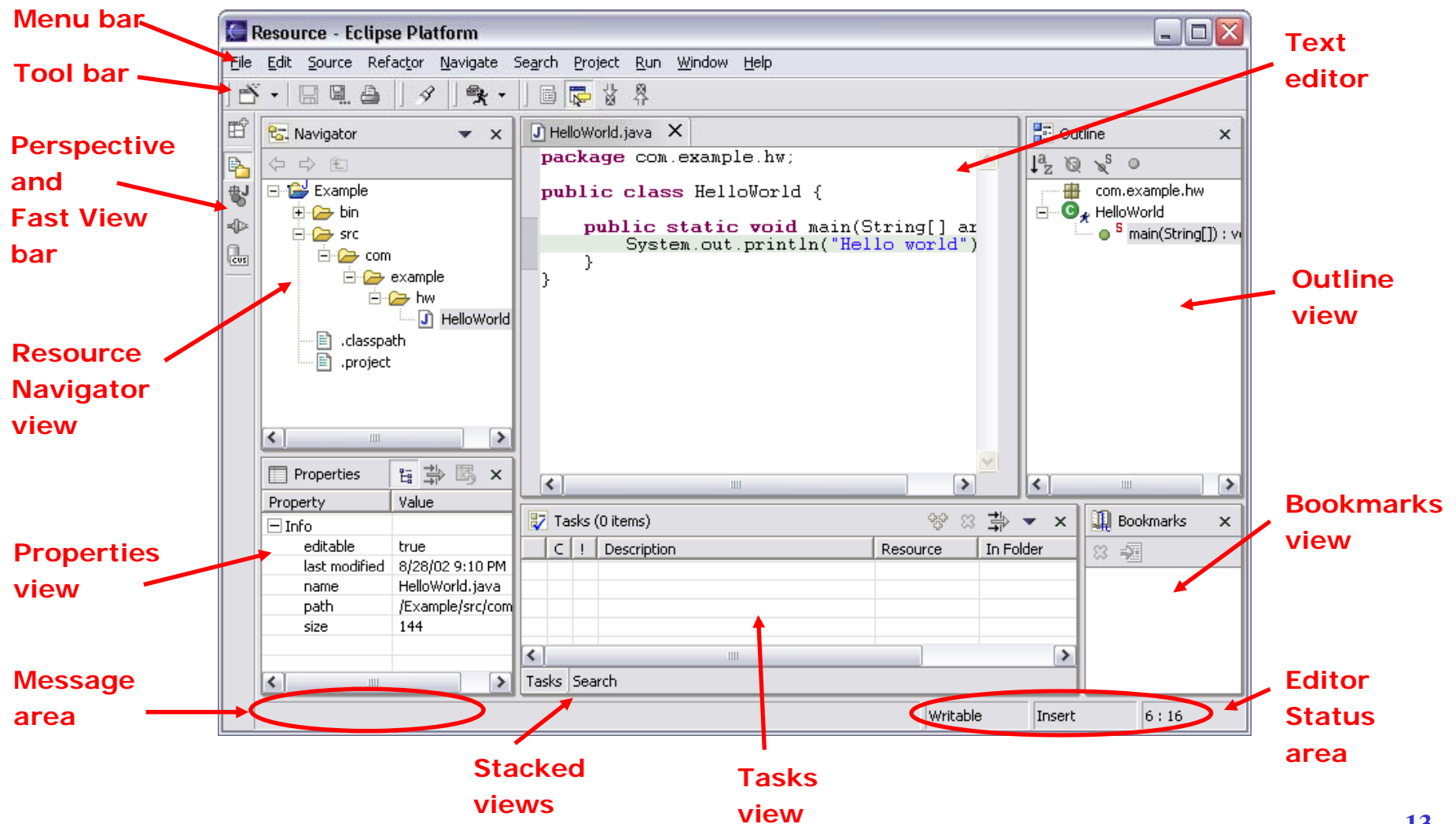


- ◆ Libreria contenente un insieme di strumenti grafici generici per la costruzione di GUI
  - Bottoni, liste, testo, menu, alberi, testi formattati, ...
- ◆ Fornisce delle API indipendenti dal sistema operativo
  - Ma richiede componenti nativi per ogni sistema operativo
  - Ogni piattaforma operativa deve mettere a disposizione il proprio file swt.jar
  - In Windows usa una .dll, in Linux un .so
- ◆ Ogni progetto grafico in Eclipse deve utilizzare SWT
  - Ma SWT può essere utilizzato anche indipendentemente da Eclipse
  - Ha spesso dimostrato prestazioni migliori rispetto a Swing



- ♦ JFace è un framework grafico di più alto livello costruito su SWT
- ♦ Mette a disposizione una serie di classi per gestire le operazioni standard di un'interfaccia grafica
- ♦ Le API e l'implementazione sono indipendenti dal sistema operativo
- ♦ Permette di costruire in modo rapido:
  - Finestre di dialogo, preferenze, wizard, alberi, tabelle, liste, ecc...

- ♦ Il workbench rappresenta l'interfaccia utente della piattaforma Eclipse
- ♦ Il workbench usa JFace e vi aggiunge alcune funzionalità più complesse
- ♦ Componenti fondamentali
  - Editors
  - Views
  - Perspectives



- 
- ◆ Compaiono nell'area centrale del workbench
  - ◆ Aggiungono azioni ai menu del workbench e alle toolbar
    - Open, edit, save, close lifecycle
  - ◆ Il workbench fornisce
    - Extension point per creare nuovi tipi di editor
      - Esempio: JDT crea un editor per gestire i file Java
    - API per editor di testo estendibili ed un framework

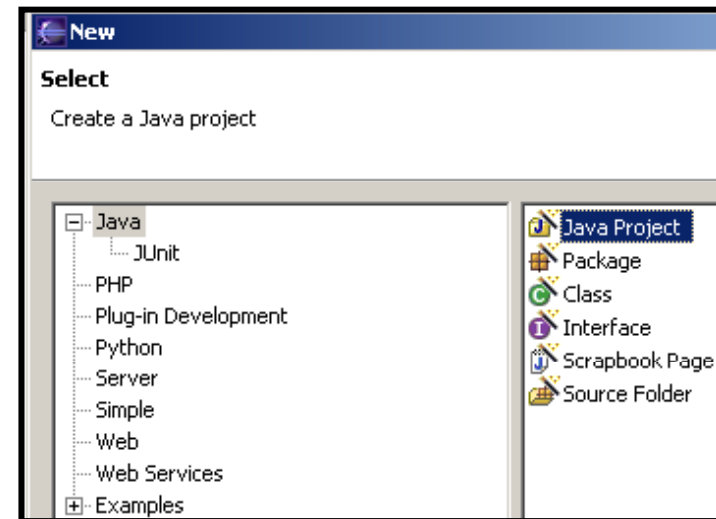
- ◆ Forniscono informazioni su di un oggetto
- ◆ Completano le funzionalità degli editors
  - Esempio: la “Outline view” riassume il contenuto dell’editor
- ◆ Completano altre views
  - Esempio: la “Properties view” caratterizza una selezione
- ◆ Il workbench fornisce:
  - Extension point per definire nuovi tipi di view
- ◆ La piattaforma Eclipse include un certo numero di view standard
  - Esempio: Resource navigator, Outline, Properties, Tasks, Bookmarks, Search, ...

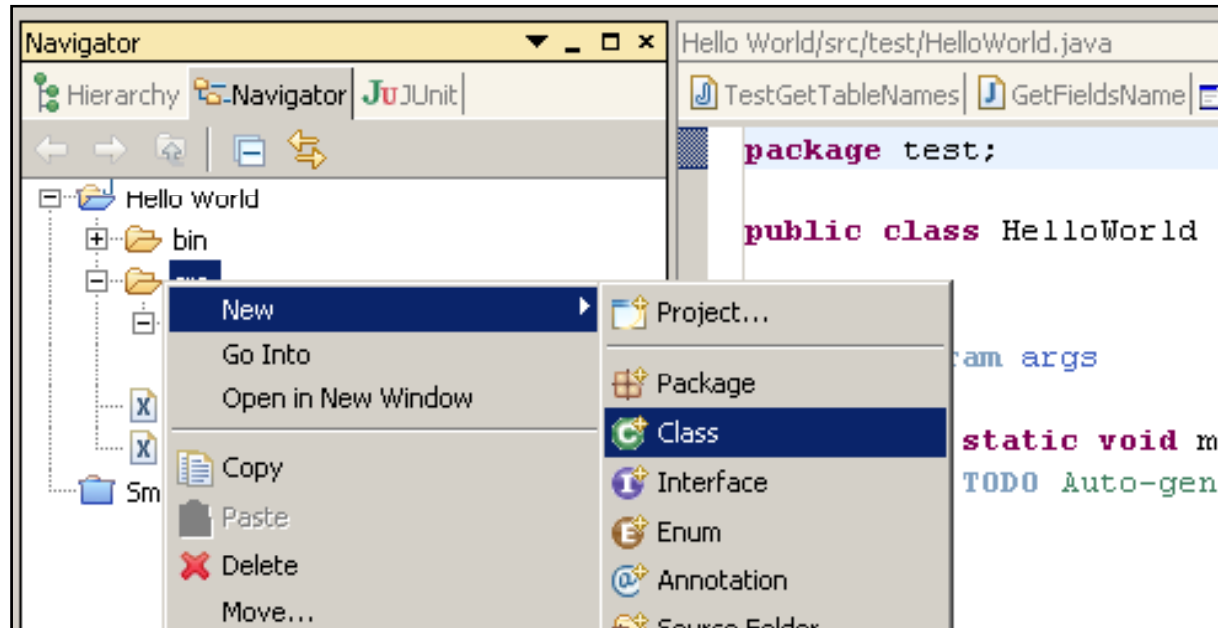
- ◆ Sono disposizioni di views ed editors
- ◆ Diverse perspectives possono essere utilizzate per gestire diverse operazioni
- ◆ Il workbench fornisce:
  - Extension Point per definire nuove perspectives
- ◆ La piattaforma Eclipse include un certo numero di perspectives standard
  - Esempio: Resource, Debug, ...



- ◆ Composto da una serie di plug-in per la creazione di un'IDE per la gestione di progetti Java
- ◆ Fornisce API per essere esteso a piacere
- ◆ Componenti principali:
  - **Core**: definisce l'infrastruttura non grafica
    - Java builder, search, refactoring, tree navigation
  - **UI**: workbench specifico per Java
    - Packages View, Type Hierarchy View, Java Outline View, Wizards per creare elementi Java, Java Editor
  - **Debug**: fornisce strumenti per il debugging
    - JVM introspection
  - **APT**: supporto per le annotazioni introdotte nella versione Java 5.0

- ◆ Cliccare su **File→New→Project...**
- ◆ Scegliere Java Project, dargli un nome e cliccare su 'Finish'





- ◆ Per creare una nuova classe fare click con il tasto destro su una package nella vista “Package Explorer”, poi si scelga dal menù “**New→Class**”

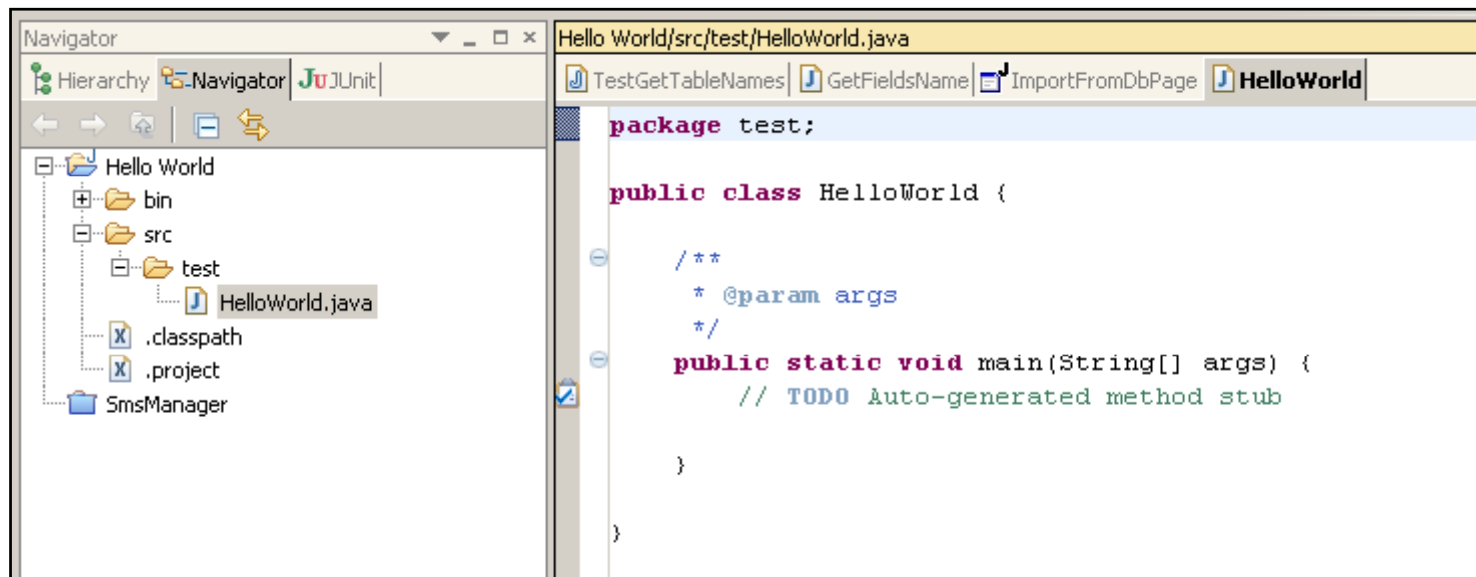
## Creare una nuova classe (2/3)

- Inserire un nome per la classe
- Eclipse controlla mentre digitiamo se il nome è sintatticamente corretto
- Si possono impostare diversi attributi della classe che stiamo creando, come la visibilità, la superclasse, eventuali interfacce implementate, ecc....

The screenshot shows the 'New Java Class' dialog box in the Eclipse IDE. The dialog has a title bar 'New Java Class' and a subtitle 'Java Class' with the instruction 'Create a new Java class.' Below this, there are several fields and options:

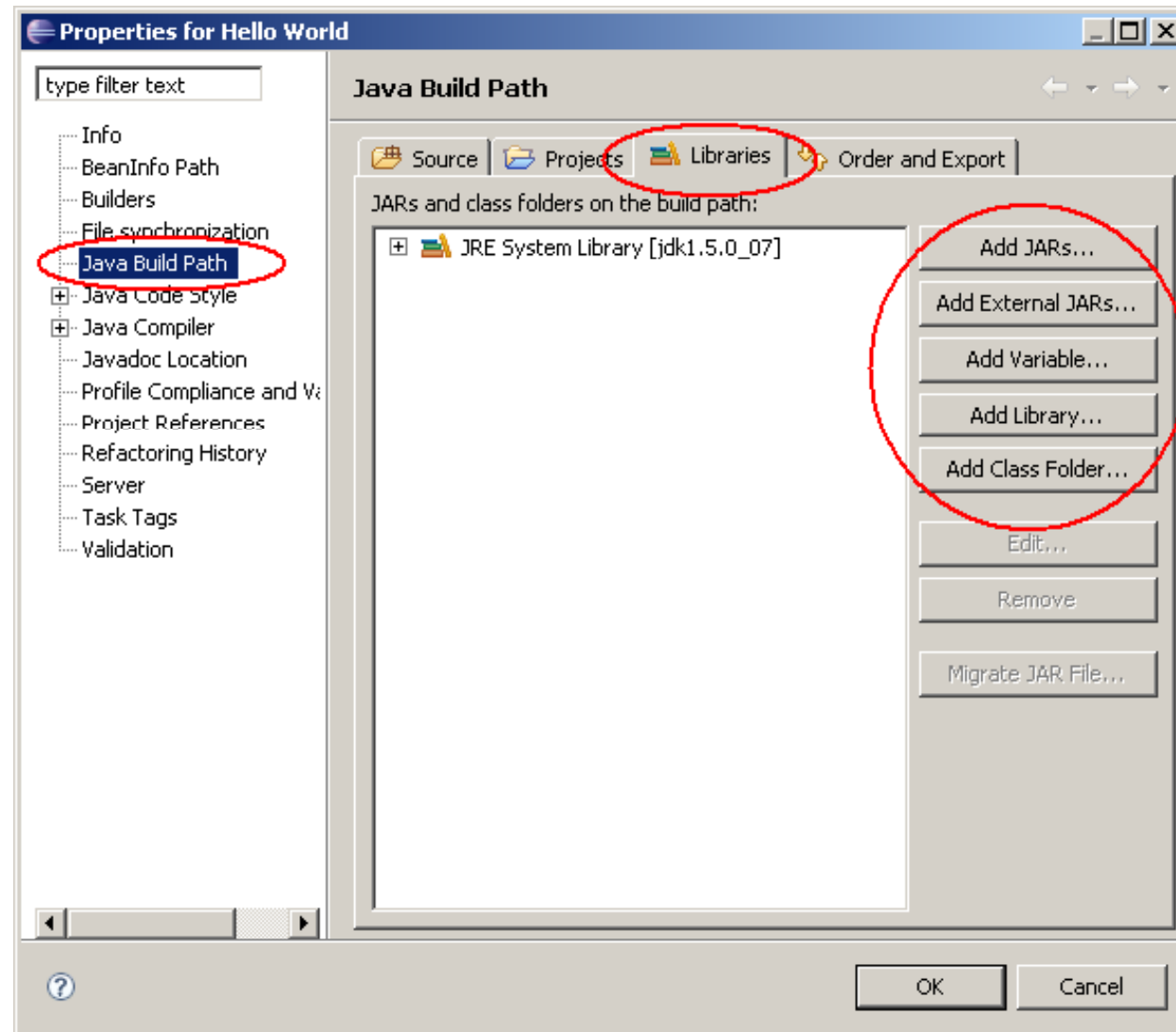
- Source Folder:** A text field containing 'HelloWorld'.
- Package:** A text field containing 'com.ibm.developerworks'.
- Enclosing type:** A checkbox labeled 'Enclosing type:' which is currently unchecked.
- Name:** A text field containing 'HelloWorld'.
- Modifiers:** A group of radio buttons and checkboxes. The 'public' radio button is selected. Other options include 'default', 'private', 'abstract', 'final', and 'static', all of which are unchecked.
- Superclass:** A text field containing 'java.lang.Object'.
- Interfaces:** An empty text area for listing interfaces.
- Which method stubs would you like to create?:** A section with three checkboxes: 'public static void main(String[] args)' (checked), 'Constructors from superclass' (unchecked), and 'Inherited abstract methods' (checked).

## Creare una nuova classe (3/3)

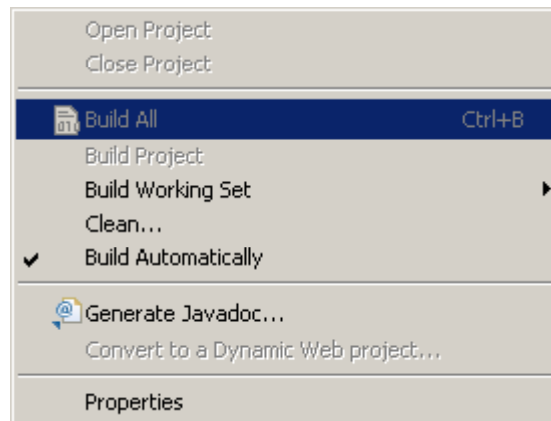


- ◆ La nuova classe appare nella vista sulla sinistra e l'editor mostra il suo codice generato

- ◆ Per compilare ed eseguire un programma Java occorre preoccuparsi che le librerie usate al suo interno siano incluse nel classpath
- ◆ Questo è possibile cliccando la voce di menù alla posizione “Project→Properties”
- ◆ Nel menù di sinistra si scelga “Java Build Path” e nella scheda “Libraries” si possono aggiungere nel classpath jar, directory, ecc...



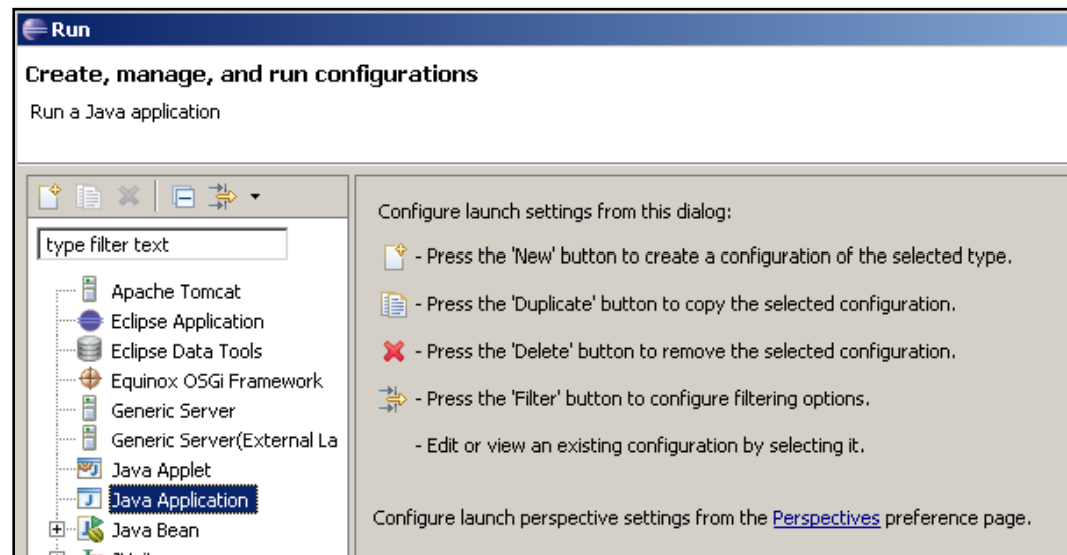
- ♦ Il menù “Project” fornisce accesso alle funzioni di compilazione del sorgente. In eclipse è possibile compilare “a comando” (“Build All”) o attivare la compilazione automatica (“Build Automatically”)



- ♦ Con la compilazione automatica l’editor segnala in tempo reale gli errori di compilazione del codice...



- ◆ Per eseguire il progetto corrente, si clicchi sul menù “Run” e si scelga “Run Configurations...”
- ◆ In questo modo si avvia il Run configurations manager. Qui si crea la “configurazione” che indica all’ambiente qual è la classe principale del progetto da eseguire



- ♦ Il manager è abbastanza “furbo” e di solito, quando si crea una nuova configurazione “Java Application”, individua automaticamente nel nostro progetto la classe con il main da eseguire

