WebSphere software

IBM

e business software

Java Development Tool (JDT)

WebSphere Application Server - Express Beta

IBM Software Group

Objectives

- Java Development Tool (JDT)
- Java Perspectives
- Some nice to know features

Java Development Tool

JDT is a complete Development Environment to create, execute, debug and profile Java Applications

Java Perspectives and Default Views

- Java
  - Package Explorer
  - Type Hierarchy
  - Outline
  - Search
  - Console
  - Tasks
- Java Browsing - similar to VAJ Java tools
  - Projects
  - Packages
  - Types
  - Members
- Java Type Hierarchy
  - Type Hierarchy can be in its own perspective or a view within another perspective
- Debug
  - Debug
  - Breakpoints
  - Expressions
  - Variables
  - Display
  - Outline
  - Console

Important Java Tool Features

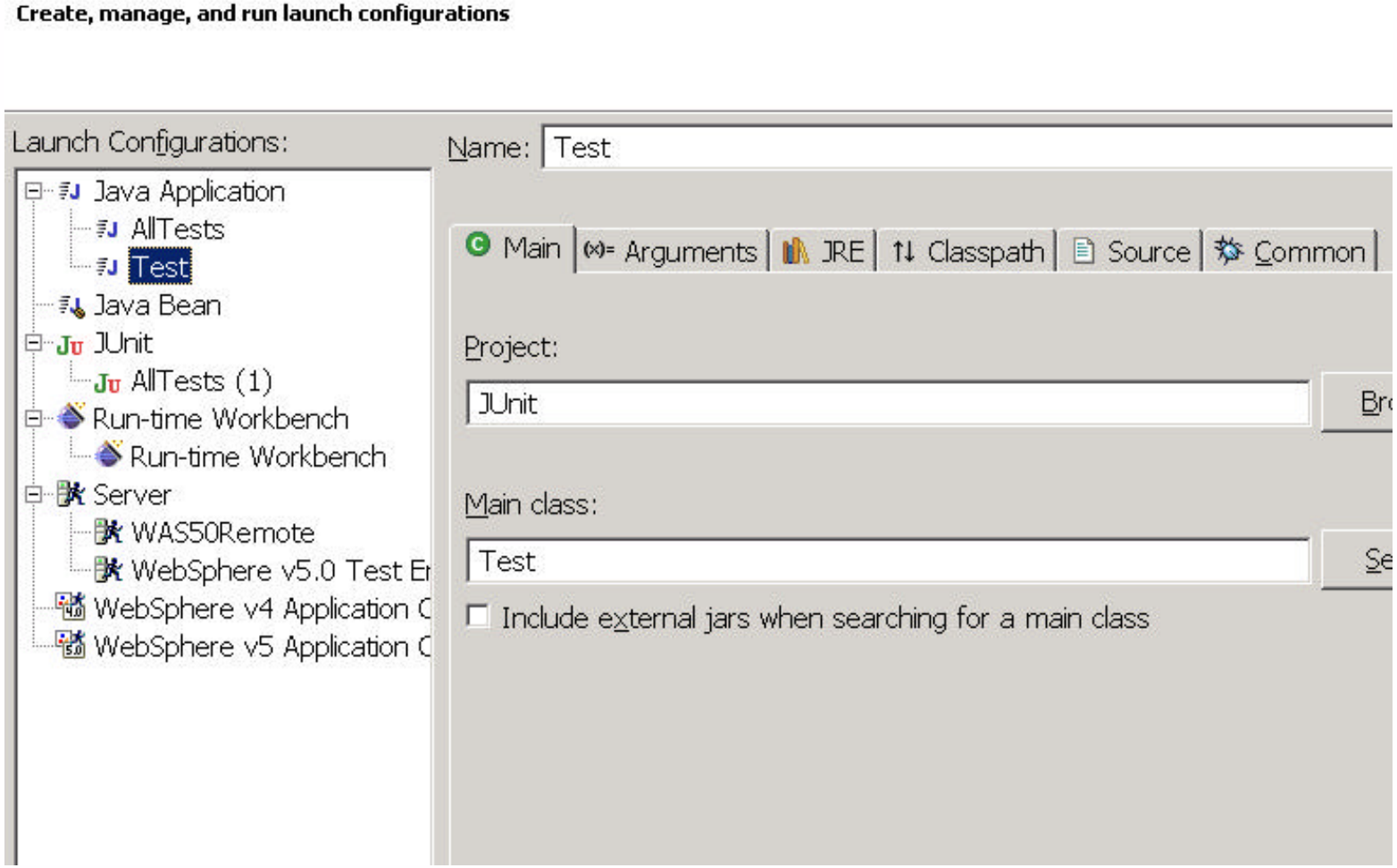
- Views
  - ►error ticks everywhere
  - ►Packages view optionally shows members; working set filtering
  - ►Type hierarchy scoped to packages and projects
- Editor
  - ►parameterized templates
  - ►bracket matching as user types
  - ►structured selection support
- Code Assist - assist in generating code, options
  - ►New: method argument hints; indicate deprecated methods, JavaDoc hints
- JUnit Integration
- Search Facilities - File Search, Java Search and Help Search
  - ►Scope to a working set
  - ►Search in hierarchy
  - ►Read/Write access to fields
  - ►find matches in binaries, inner classes
  - ►Search References, Declarations, Implementors

Refactoring

- Refactoring is behavior-preserving program transformation
  - Make a system-wide code change without affecting the behavior of the system
    - JDT automatically manages the changes
- Refactoring functions
  - On Class, Interface
    - Rename
    - Move
  - On Method
    - Rename
    - Pull-up
  - On Field
    - Rename
    - Pull-up
    - Self Encapsulation
- Can have Refactor with or without preview (set in Windows Preferences)
- Integrated with drag & drop
- Extract local variable can replace all occurrences of an expression

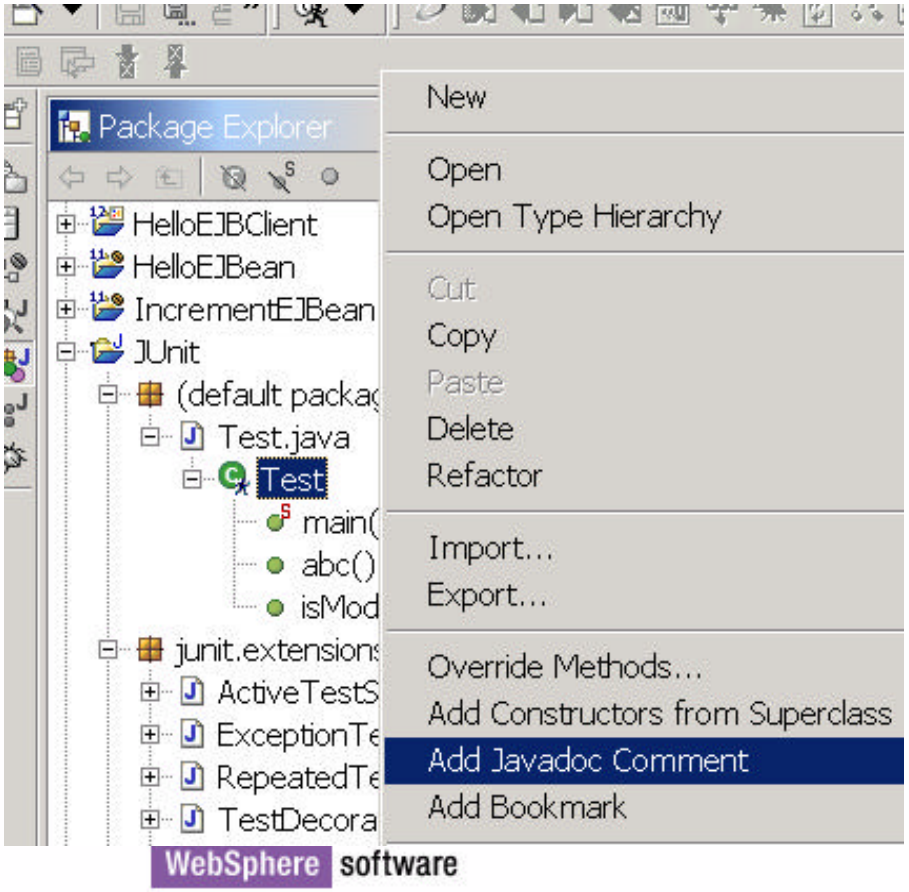
Java Launch Configurations

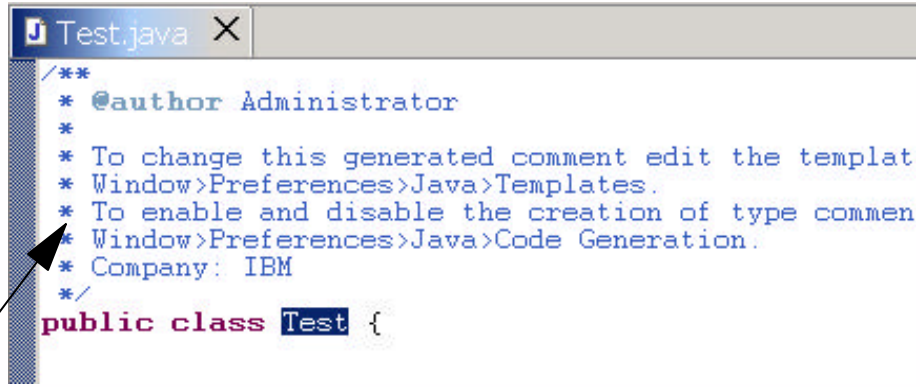
- Launch configurations in Run or Debug mode



Generating JavaDoc

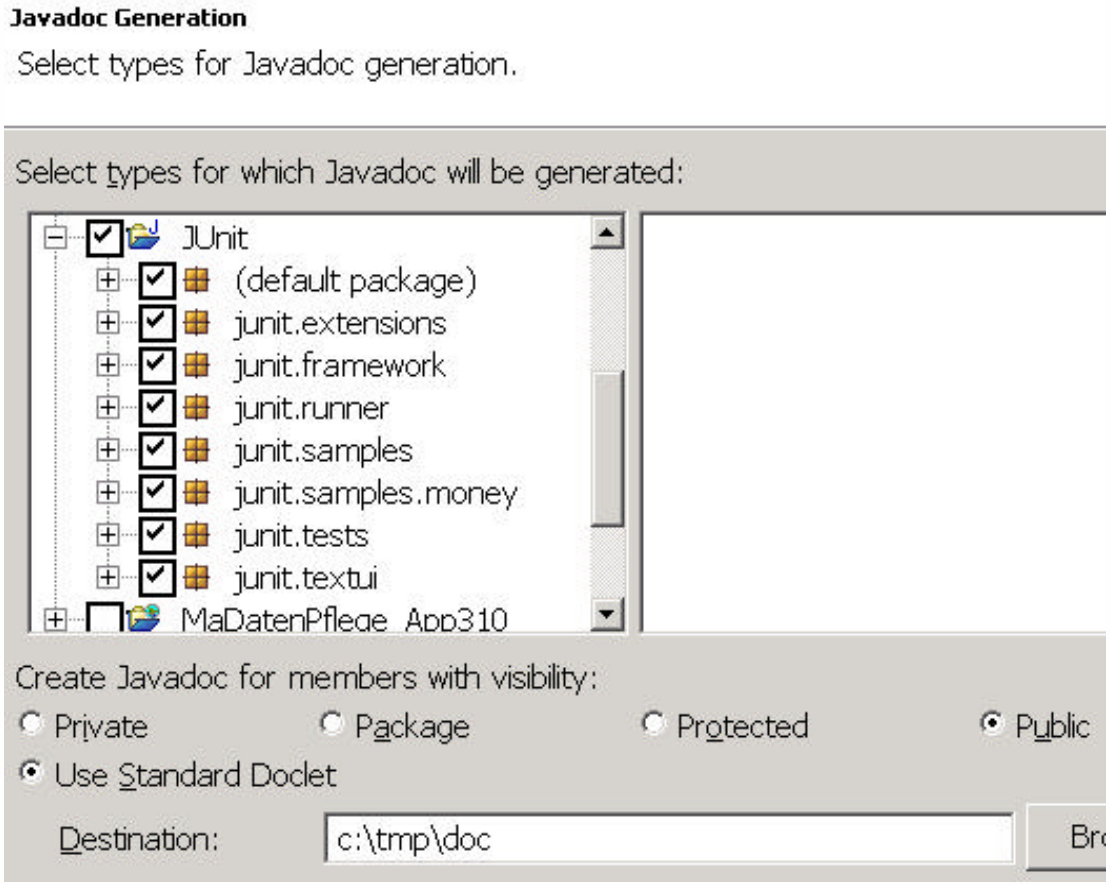
- Right click the on class type or members and apply "Add JavaDoc Comment" action
- Can modify the template being used
  - Edit template variable "typecomment" from Window > Preferences > Java > Templates
- Generating JavaDoc creates the shell of the JavaDoc - you fill the necessary information

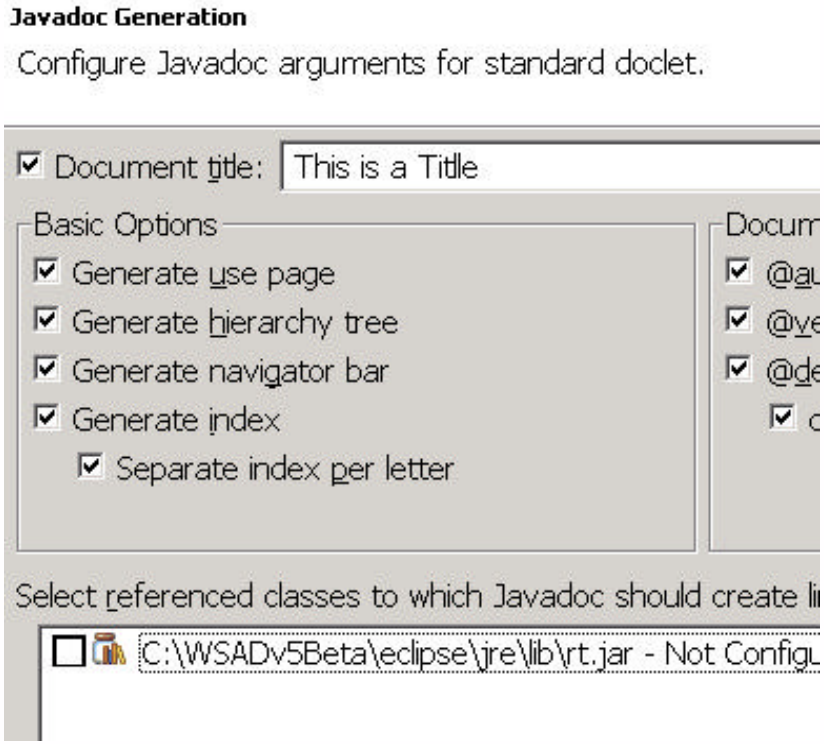




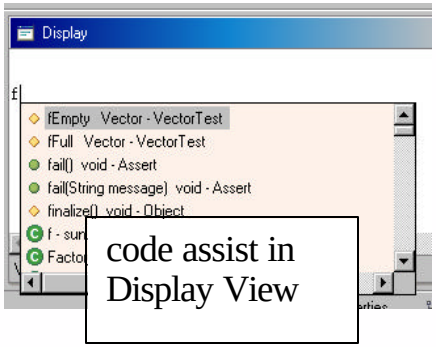
Publishing JavaDoc

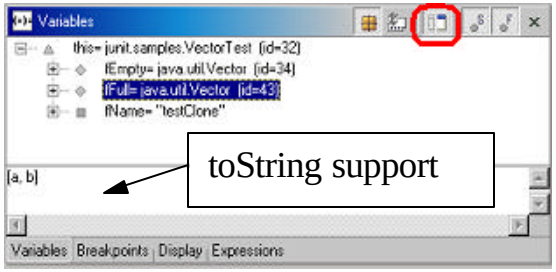
- Setup the JavaDoc command to use for export
  - Window > Preferences > Java > JavaDoc
- Export > JavaDoc
  - Select the project you want to create the JavaDoc and directory
  - Options to generate use page, hierarchy tree, navigator bar, index, etc.
  - Option to select your own Style sheet, and to add an overview HTML page

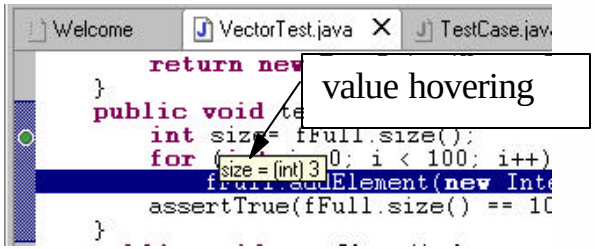




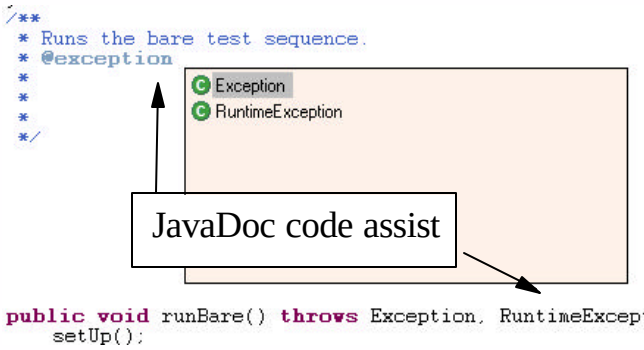
Debugger features you might not know about

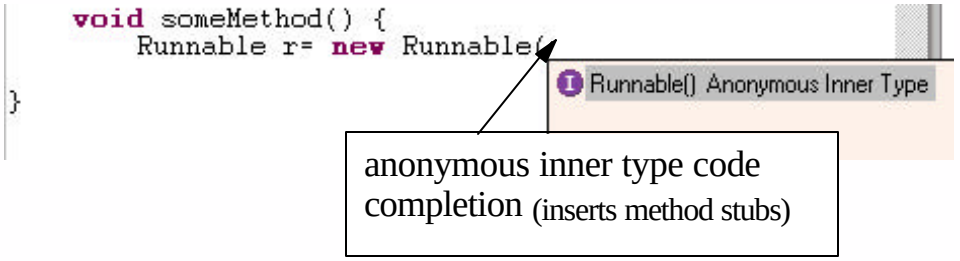


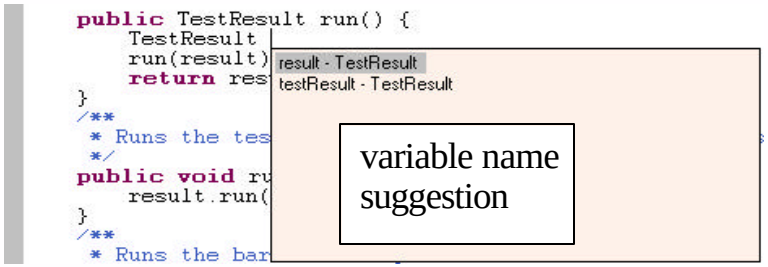


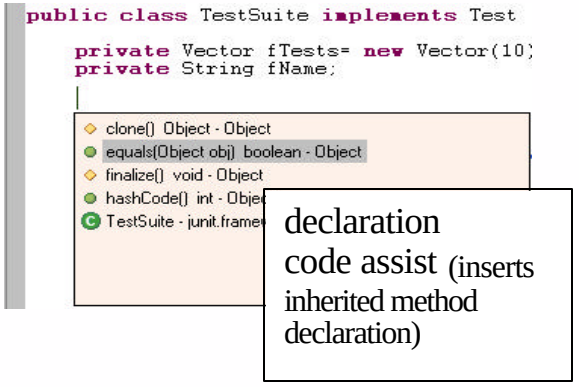


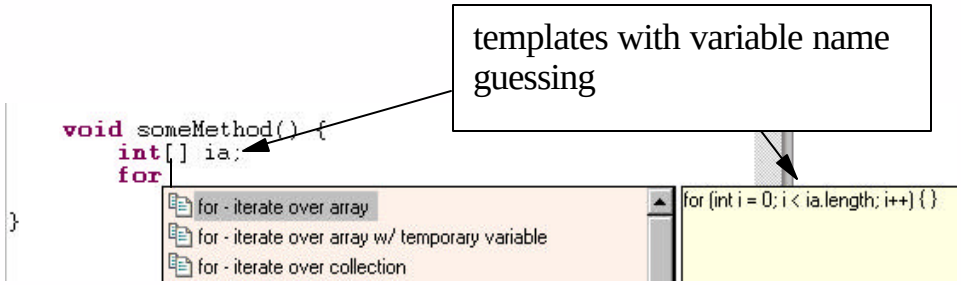
Editor and code assist features you might not know about

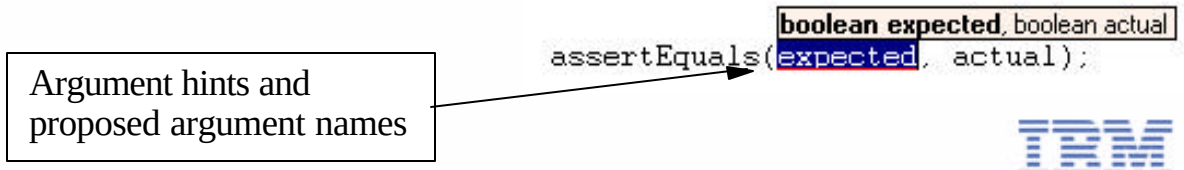




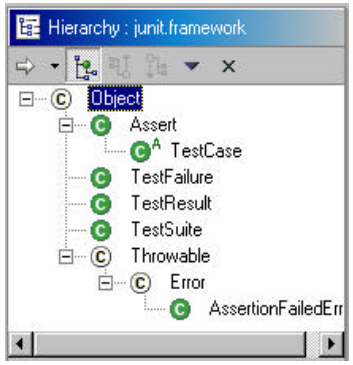




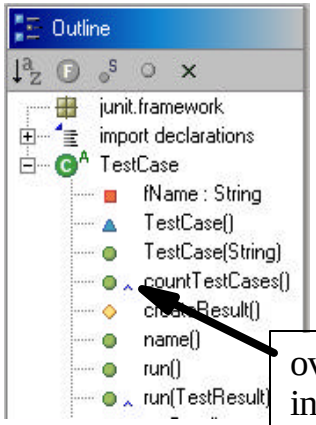




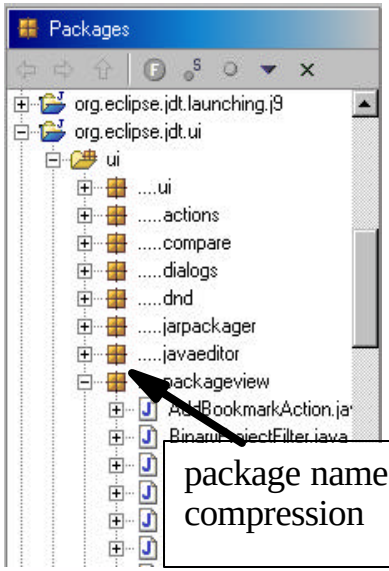
View features you might not know about



open type hierarchy
on a package/project



override
indicator



package name
compression

Other features you might not know about

- **Debugger**
  - run code with errors
  - automatic suspend in debugger when error is encountered
  - method entry/exit breakpoints
  - variable changed indication in Variables view
- **Import management – automatic import on…**
  - code assist type completion
  - override methods, getter setter generation
  - refactoring
- **Compare**
  - structure compare for .java and .properties files
  - option for ignore white space – in Java structure compare ignores formatting and comments

Summary

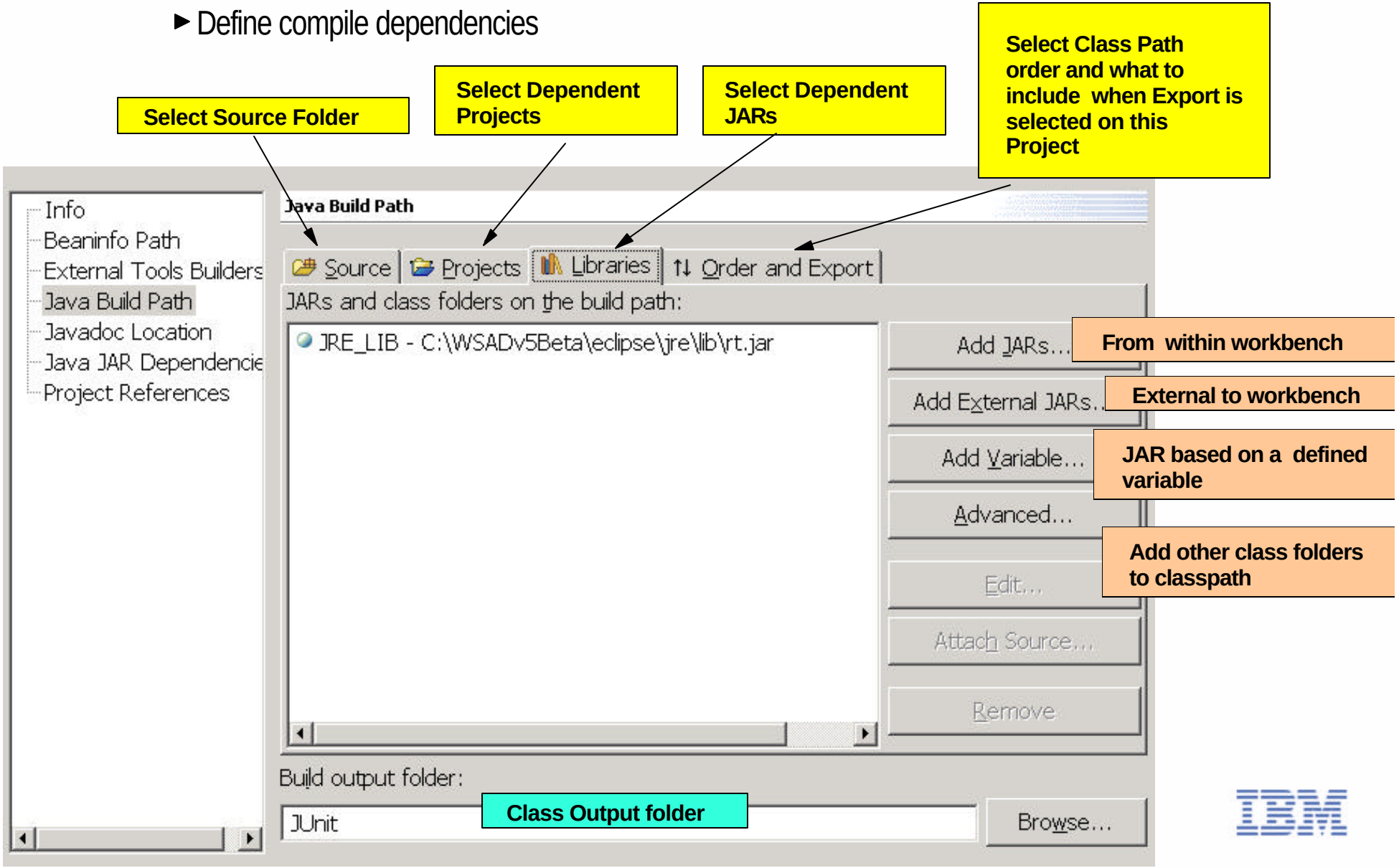
- WSSD provides a very powerful and flexible environment to create and debug Java applications
- Integrated profiler assist developer to find performance bottlenecks at the class and method level
- Only few of the many functions exposed here
- Help documents the detail of many of the features
- Hands-on labs and tutorials are best way to explore these powerful features

Section

Appendix

Java Build Path

- From Project > Properties panel
  - Define compile dependencies

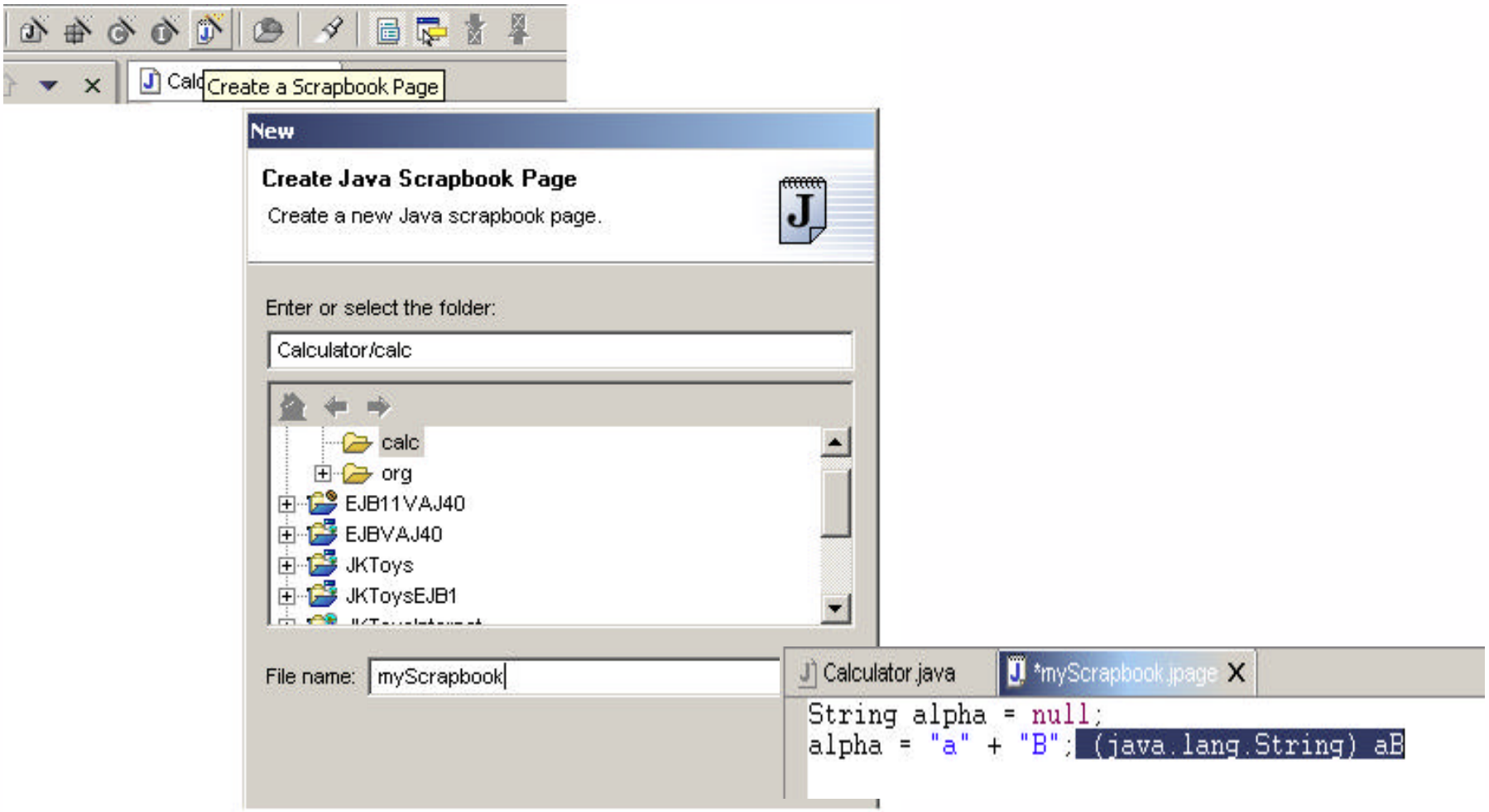


Other useful functions

- Scrapbook Facilities
- Compare and/or Replace Functions
  - From Local History or Team Repository
- Organize Imports

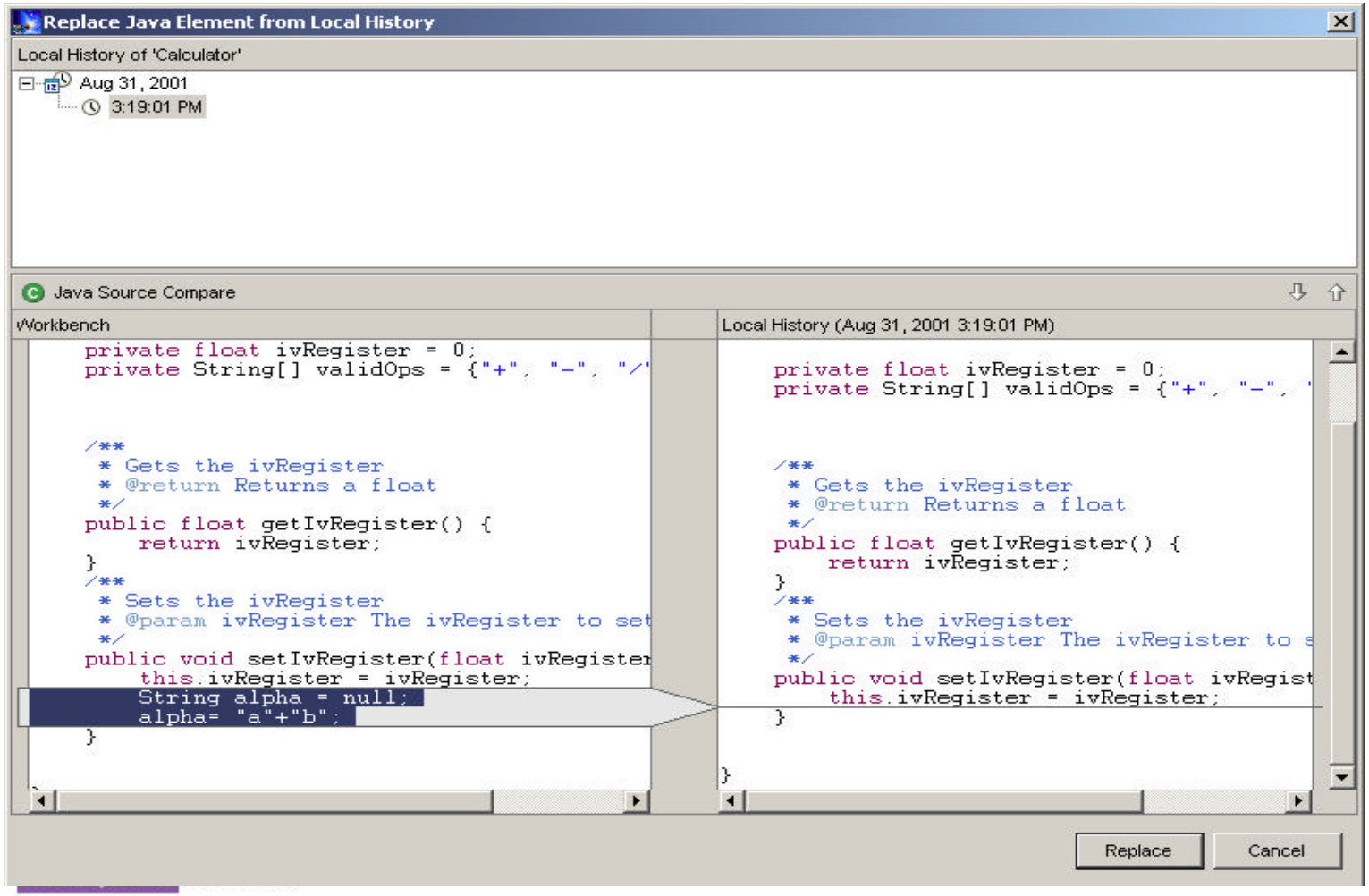
Scrapbook Facilities

- Allows evaluating and testing snippets of code



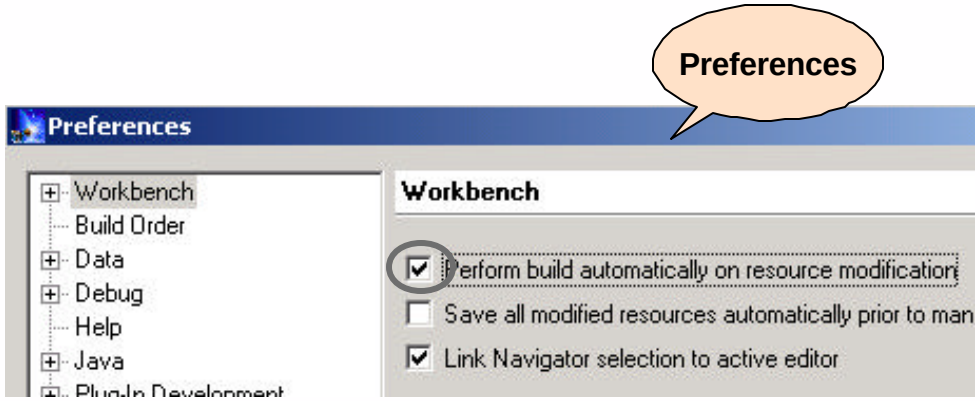
Local History of Code Changes

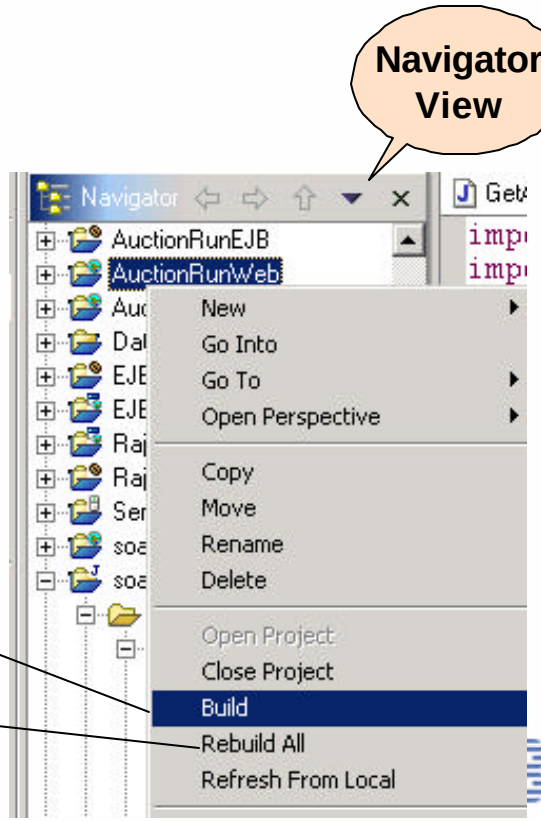
- Compare, replace, or add code from local repository
  - Independent from team repository
  - Customize size of local history through Workbench properties



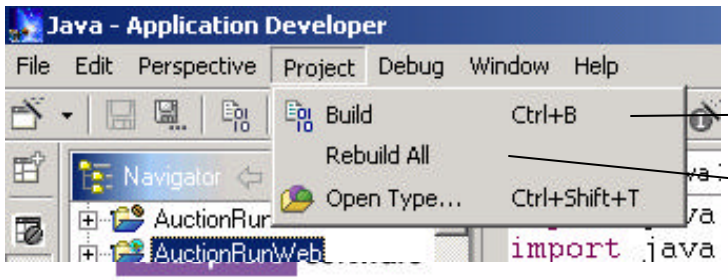
Building Projects - Triggers

- Build triggers - when to build
  - Automatic on file "Save"
    - Default Global attribute
  - Manual
    - Build per project
      - Incremental or Full
    - Build for all projects
      - Incremental or Full





Note
Incremental Build available only if "Automatic" build on file Save is turned OFF



Incremental Build

Full Build

Building Projects with ANT

- ANT: Open source Java based equivalent of *make*
  - ANT scripts written in XML
- ANT documentation
  - http://jakarta.apache.org/ant/index.html