Phần I: GIỚI THIỆU ECLIPSE

A. Eclipse là gì ?

Eclipse là phần mềm miễn phí, được các nhà phát triển sử dụng để xây dựng những ứng dụng J2EE, sử dụng Eclipse nhà phát triển có thể tích hợp với nhiều công cụ hỗ trợ khác để có được một bộ công cụ hòan chỉnh mà không cần dùng đến phần mềm riêng nào khác.Eclipse SDK bao gồm 3 phần chính: **Platform, Java Development Toolkit (JDT), Plug-in Development Environment (PDE)**. Với JDT, Eclipse được xem như là một môi trường hỗ trợ phát triển Java mạnh mẽ. PDE hỗ trợ việc mở rộng Eclipse, tích hợp các **Plug-in** vào Eclipse Platform. Eclipse Platform là nền tảng của toàn bộ phần mềm Eclipse, mục đích của nó là cung cấp những dịch vụ cần thiết cho việc tích hợp những bộ công cụ phát triển phần mềm khách dưới dạng Plug-in, bản thân JDT cũng có thể được coi như là một Plug-in làm cho Eclipse như là một **Java IDE (Integrated Development Enviroment)**.

B. Kiến trúc Eclipse:

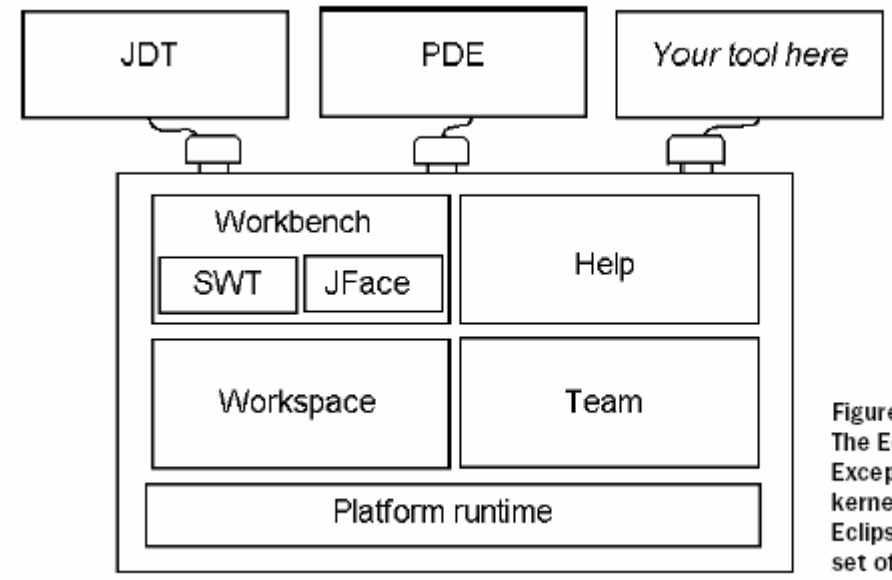


Figure 1.1
The Eclipse architecture. Except for a small runtime kernel, everything in Eclipse is a plug-in or a set of related plug-ins.

1. The Platform runtime :

Công việc chính của **Platform runtime** là phát xem plug-in nào đang có trong thư mục plug-in của Eclipse.Mỗi Plug-in đều có 1 tập tin Manifest liệt kê những kết nối mà plug-in cần.Pug-in chỉ được tải vào Eclipse mỗi khi thực sự cần thiết để giảm lượng tài nguyên yêu cầu và thời gian khởi tạo.

2. The workspace :

- Workspace chịu trách nhiệm quản lý tài nguyên người dùng được tổ chức dưới dạng **Project.** Mỗi Project là một thư mục con trong thư mục Workspace.
- Workspace bảo quản cấp thấp lịch sử những sự thay đổi tài nguyên, tránh thất thoát tài nguyên người dùng.
- Workspace đồng thời chịu trách nhiệm thông bào những công cụ cần thiết cho việc thay đổi tài nguyên.

3. The Workbench :

Workbench là giao diện đồ họa người dùng của Eclipse, gồm có **Standard Widget Toolkit (SWT)** và **JFace**. Eclipse không hòan toàn bắt buộc phải sử dụng SWT hay Jface để lập trình giao diện, bạn vẫn có thể sử dụng **AWT** hay **SWING** của Java thông qua việc cài đặt các Plug-ins.

4. Team support :

Trang bị hệ thống quản trị để quản lý dự án của người dùng : **Concurrent Versions System (CVS)**

5. Help :

Cung cấp hệ thống tài liệu mở rộng, có thể là định dạng HTML hay XML.

PHẦN II: LÀM QUEN VỚI ECLIPSE

A. Cài đặt Eclipse :

Giải nén tập tin Eclipse SDK vào thư mục mà bạn muốn cài đặt (Ví dụ : C:\Eclipse). Chép thư mục JRE của JDK vào thư mục con của thư mục Eclipse. Sau đó, chạy tập tin eclipse.exe để hoàn thành cài đặt. Chú ý, khi cài đặt Eclipse không đòi hỏi cài đặt, đăng ký vào hệ điều hành, sửa đổi biến môi trường hay yêu cầu boot lại máy.

B. Perspective :

Khi khởi động Eclipse, màn hình đầu tiên bạn thấy đó là Eclipse Workbench. Eclipse Workbench đưa ra khái niệm về Perspective, đó là những bối cảnh khác nhau của giao diện Eclipse hỗ trợ cho một công việc nhất định. Khi làm việc với Eclipse bạn sẽ luôn chuyển qua lại giữa các Perspective khác nhau. Ví dụ như khi soạn thảo chương trình, kiểm tra lỗi, biên dịch, sữa lỗi đều có mỗi Perspective khác nhau.

C. Tạo một chương trình Java :

Trước tiên, bạn phải tạo một Project mới:

1. Chọn File → New Project → Hộp thoại New Project mở ra có 3 chọn lựa ở bên cửa sổ trái: Java, Plug-in Development, và Simple. Chọn Java để tạo một Project Java. Sau đó chọn Java Project ở cửa sổ bên phải. Chú ý nếu bạn có cài đặt các Plug-ins phát triển Java thì sẽ được liệt kê hết tất cả ở phần chọn lựa con trong chọn lựa Java bên cửa sổ trái.
2. Click Next → Hộp thoại mới mở ra, yêu cầu đặt tên cho Project mới. Bạn hãy điền vào tên project là **Hello.**
3. Chọn Next → Hộp thoại kế tiếp cho phé bạn thay đổi các cấu hình Java. Chọn Finish để kết thúc.

Kế tiếp, bạn có thể tạo ra một chương trình Java:

1. Click phải chuột trên project **Hello,** chọn New→ Class.
2. Hộp thoại mới mở ra, gồm có :
    a. Source Folder : giữ nguyên, không thay đổi
    b. Package : gỏ vào **org.eclipseguide.hello**
    c. Class Name : gõ vào **HelloWorld**
    d. Ở phần Which Method Stubs Would You Like to Create?, chọn public static void main(String[] args)
3. Chọn Finish để kết thúc.

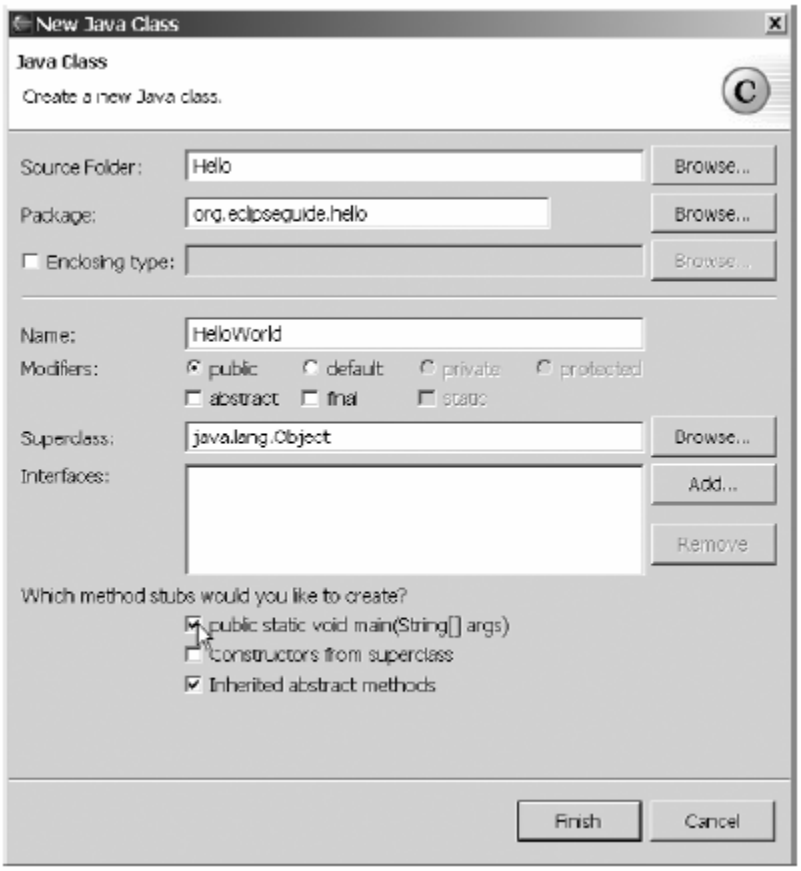


Figure 2.3 Creating the `HelloWorld` class using the New Java Class wizard

Đoạn mã chương trình tự động được sinh ra, với các phương thức rỗng. Bạn phải tự điền thêm vào các chức năng mới cho các phương thức này, thêm dòng lệnh `System.out.println("Hello, world!")` vào hàm `main()`:

```
/*
* Created on Feb 14, 2003
*
* To change this generated comment go to
* Window>Preferences>Java>Code Generation>Code and Comments
*/
package org.eclipseguide.hello;
/**
* @author david
*/
public class HelloWorld {
  public static void main(String[] args) {
  System.out.println("Hello, world!");
   }
}
```

D. Chạy chương trình Java:

1. Chọn chương trình Java bạn muốn chạy bên cửa sổ Package Explorer. Ở đây, sẽ chọn tập tin Hello World.java
2. Trên Menu chính, chọn Run → Run As → Java Application.
3. Cửa sổ Task sẽ thay đổi bằng cửa sổ Console, xuất ra kết quả chương trình như hình dưới đây:

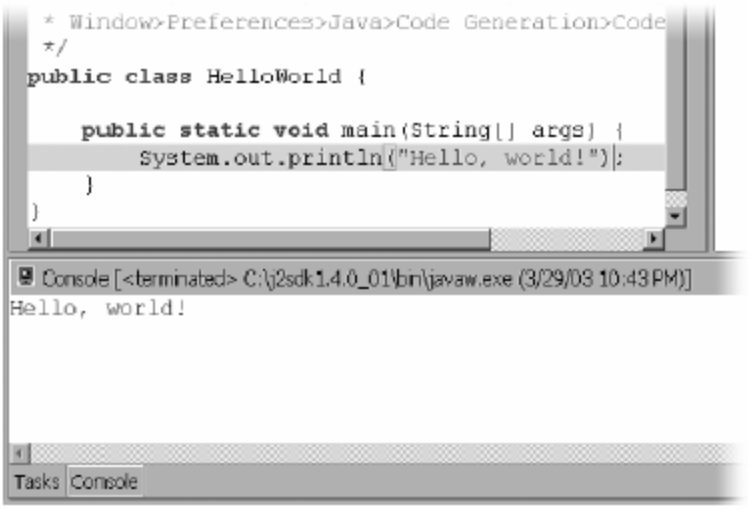


Figure 2.5
The Eclipse Console view displays the output from the HelloWorld program.

E. Gỡ lỗi một chương trình Java (Debug):

Chúng ta sẽ debug chương trình HelloWorld.java. Trước tiên, bạn phải thay đổi chương trình HelloWorld.java theo chương trình dưới đây :

```
public class HelloWorld {
public static void main(String[] args) {
say("Hello, world");
}
public static void say(String msg) {
for (int i = 0; i < 3; i++) {
System.out.println(msg);
}
```

Khi debug một chương trình, bạn cần thiết lập các điểm ngắt cần thiết để kiểm tra lỗi, chương trình sẽ tạm dừng khi gặp các điểm ngắt này, cho phép bạn gỡ lỗi, nếu không có các điểm ngắt này chương trình sẽ tiếp tục thực hiện. Để tạo các điểm ngắt, bạn chọn dòng cần dừng lại, rồi double-click vào lề trái màu xám của cửa sổ soạn thảo phía trước dòng lệnh. Một dấu chấm màu xanh xuất hiện, cho biết đã kích hoạt điểm ngắt. Sau đó, bạn thực hiện các bước sau để debug chương trình:

1. Chọn chương trình cần debug.
2. Chọn Menu Run trên thanh Menu chính → chọn Debug As → Java Application
3. Eclipse tự động chuyển từ Java Perspective sang Debug Perspective.

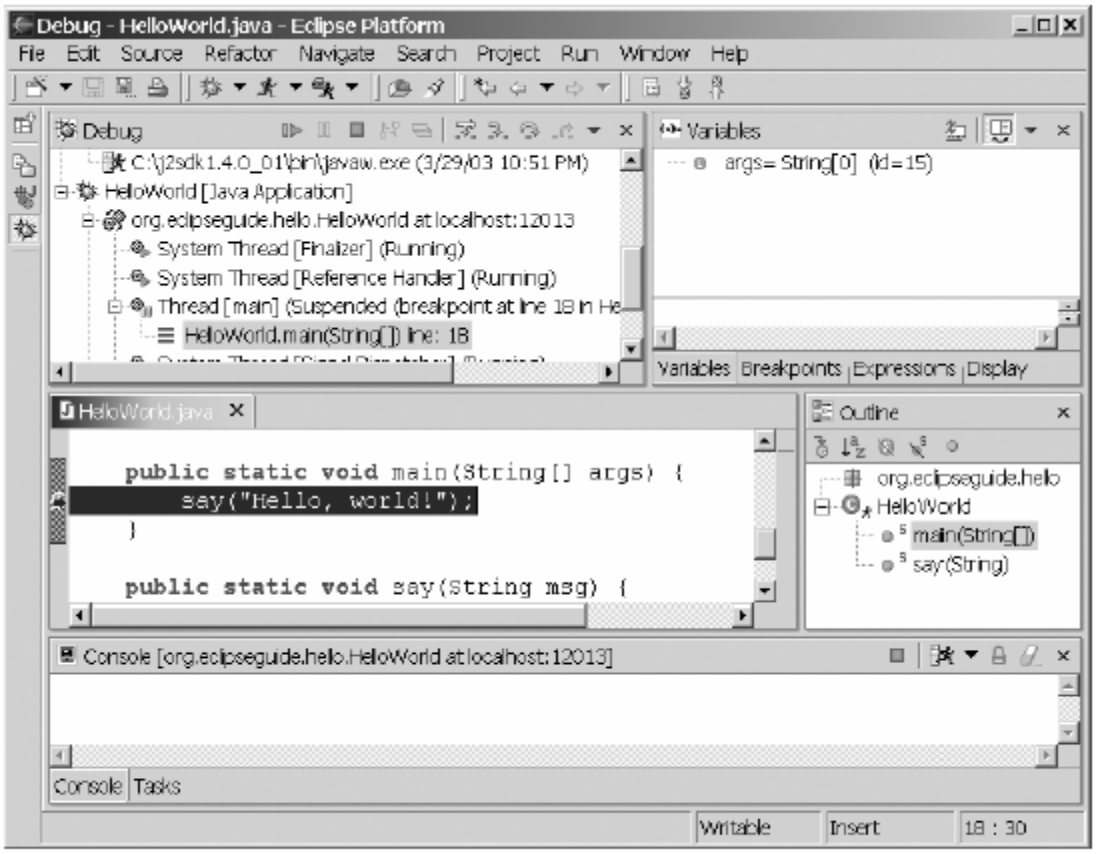


Figure 2.6 Debugging HelloWorld: Execution is suspended at the first breakpoint.

F. Các tính năng mới của Eclipse hỗ trợ cho việc soạn thảo chương trình Java:

1. **Code Completion (Hoàn chỉnh đoạn mã) :** Eclipse hỗ trợ bạn viết mã chương trình Java thêm chính xác, đồng thời giúp không cần phải nhớ nhiều về cú pháp câu lệnh. Mỗi khi bạn đặt dấu chấm hoặc ấn Ctrl + Space (kích hoạt Content Assistant), trình soạn thảo Java của Eclipse sẽ sổ ra một danh sách các câu lệnh hỗ trợ tương ứng, đồng thời xuất hiện phần hướng dẫn sử dụng câu lệnh tương ứng bên cạnh.

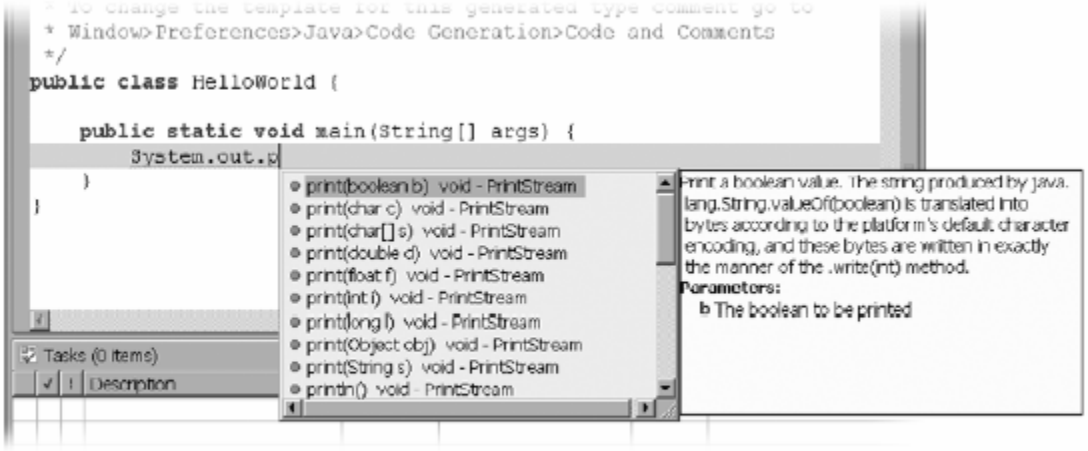


Figure 2.4 The Eclipse code assist feature displays a list of proposed methods and their Javadoc comments. Scroll or type the first letter (or more) to narrow the choice, and then press Enter to complete the code.

Chú ý: để sử dụng tính năng hỗ trợ phần hướng dẫn sử dụng, bạn cần phải chỉ cho Eclipse biết đường dẫn đặt Javadoc của JDK như sau :

- Trên thanh Menu chính, chọn Window → Preferences → ở danh sách cửa sổ bên trái, chọn Java → Installed JREs → phía cửa sổ bên phải chọn Standard VM, click nút Edit.
- Hộp thoại mới xuất hiện → bạn hãy thay đổi đường dẫn ở ô Javadoc URL theo đúng đường dẫn của Javadoc trong máy của bạn.

2. **Quick Fix (Sửa lỗi nhanh):** Mỗi khi bạn gõ vào một câu lệnh mà có vấn đề về lỗi thì Eclipse ngay lập tức thông báo lỗi, khi đó ngay phía trước vị trí dòng lệnh xuất hiện một ký hiệu hình bóng đèn sáng, click vào ký hiệu này (hoặc đặt con trỏ chuột ngay dòng lệnh lỗi, nhấn Ctrl+1) Eclipse sẽ đề nghị bạn một danh sách các phương pháp khắc phục lỗi tương ứng, double-cliclk vào biện pháp thích hợp Eclipse sẽ hỗ trợ bạn sữa lỗi một cách nhanh chóng.
3. **Refactor:** Trong Project của bạn nếu có việc các lớp sử dụng kế thừa hoặc cài đặt từ các lớp khác, khi đổi tên một lớp sẽ ảnh hương đến toàn bộ Project, bắt buộc bạn phải tra lại toàn bộ Project để thay đổi. Eclipse đã giúp bạn làm việc này một các nhanh chóng nhờ tính năng Refactor, Eclipse sẽ tự động cập nhật tòan bộ Project cho phù hợp với tên mới, bạn chỉ cần thụchiện theo các bước sau : nhấn phải chuột trên tên tập tin cần đổi, chọn Refactor → Rename. Ngoài ra, Refactor sẽ giúp bạn rút trích ra được lớp giao diện (Interface) từ các lớp dựng sẵn và Eclipse sẽ tự động cài đặt Interface trên các lớp có sử dụng giao diện này, tương tự như trên bạn chọn Extract Interface trong Refactor và tiếp tục thực hiện theo các hộp thoại xuất hiện kế tiếp.
4. **Local History:** giúp bạn so sánh giữa phiên bản mới và phiên bản cũ của tập tin bạn đang làm việc:
   - Nhấp phải trên tập tin trong cửa sổ Package Explorer.
   - Chọn Compare With → Local History.

   Bạn có thể cho phép Eclipse thay đổi mã chương trình thành trở lại phiên bản cũ bằng cách tương tự, thay vì chọn Compare With, bạn sẽ chọn Replace With.

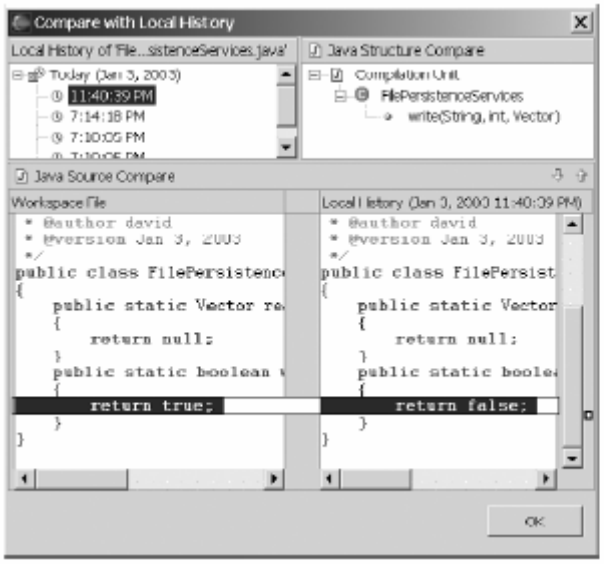


Figure 3.6
Comparing the current code with a previous version in the local history

5. **Java Scrapbook pages (Trang Java rời đính kèm):** Khi viết chương trình Java, đôi khi bạn có một ý tưởng mới mà bạn không chắc nó sẽ hoạt động tốt, bạn chỉ muốn thử trước mà không muốn ảnh hưởng đến chương trình bạn đang làm việc. Eclipse sẽ giúp bạn viết một chương trình nhỏ rời ra bằng cách sử dụng trình soạn thảo đơn giản được thực thi dưới dạng command prompt. Đó là Java Scrapbook Pages. Bạn có thể gõ vào Scrapbook Pages các câu lệnh Java và thực thi nó mà không cần khai báo lớp (class) hay phương thức (method). Để tạo một trnag Scrapbook, bạn phải chuyển sang bối cảnh soạn thảo Java (Java Perspective), rồi thực hiện các bước sau :
   a. Nhấp phải chuột vào tên Project (ở đây là **HelloWorld** project).
   b. Chọn New → Scrapbook Page → Hộp thoại xuất hiện yêu cầu nhập vào tên tập tin : **Test.**
   c. Sau đó, gõ vào một vài dòng lệnh Java để thử :

```
for(int i = 1; i < 10; i++)
{
HelloWorld.say(Integer.toString(i));
}
```

   d. Thực thi đoạn mã này như sau : nhấp phải chuột trên màn hình soạn thảo, chọn Set Imports trong Menu phụ.
   e. Hộp thoại Java Snippet Imports xuất hiện, nhấn vào nút Add Packages.
   f. Hộp thoại mới xuất hiện yêu cầu bạn chọn gói được sử dung : **org.eclipseguide.hello.** Nhấn OK.
   g. Bây giờ, bạn có thể thực thi đoạn mã bằng cách quét chuột chọn toàn bộ đoạn mã → nhấp phải chuột → Execute. Như bất kì chương trình Java bình thường , kết quả sẽ được xuất ra trong ô Console View.

   CHÚ Ý: khi sử dụng với **StringTokenizer**, bạn phải khai bào gói tương ứng trong **java.util.***, bằng cách sau : trong hộp thoại Java Snippet Imports, nhấn vào nút Add Types và gõ vào StringTokenizer , Eclipse sẽ tự động tìm thấy gói tương ứng **java.util.StringTokenizer.**

G. Thay đổi cấu hình và thông số cài đặt, bao gồm kiểu định dạng mã và thiết lập biến đường dẫn môi trường:

1. **Câu chú thích Javadoc:**
   Đây là đoạn văn bản xuất hiện ở phần đầu khi bạn tạo ra một lớp mới. Bạn có thể thay đổi nó theo các bước sau đây:
   - Chọn WindowPreferences → Java → Code Generation
   - Nhấn chuột vào tab Code and Comment.
   - Chọn Code → New Java files → click vào nút Edit.
   - Thay đổi lại như sau :
   ```
   /* ${file_name}
   * Created on ${date}
   */
   ${package_declaration}
   ${typecomment}
   ${type_declaration}
   ```
   - Click OK trong hộp thoại Edit Template.
   - Bằng cách tương tự bạn có thể thay đổi TypeComment trong Code → Types.
2. **Định dạng lại đoạn chương trình:** để đồng bộ hoá toàn bộ đoạn mã chương trình theo một chuẩn định dạng nhất định, bạn click phải chuột trên màn hình soạn thảo → chọn Format. Bạn có thể thay đổi lại chuẩn định dạng này :
   - Chọn Preferences → Java → Code Formatter.
   - Ở phía bên cửa sổ Options, chọn tab New Lines.
   - Đánh dấu kiểm chọn vào Insert a New Line Before an Opening Brace (Cho phép chèn thêm một dòng trước khi mở ngoặc {).
3. **Mẫu đoạn mã chung:** Eclipse cho phép ta có thể tạo ra những mẫu đoạn mã chung, giúp ta dễ dàng sử dụng nhiều lần mà không cần phải gõ lại toàn bộ như trước.
   - Windows → Preferences → Java → Editor → Templates.
   - Nhấn vào nút New.
   - Hộp thoại New Templates xuất hiện : đặt tên cho mẫu sử dụng là **sop** và **Shortcut for System.out.println()** vào ô mô tả (Description).
   - Điền mẫu vào Template: System.out.println("${cursor}").
   - Nhấn OK để đóng hộp thoại Template.
   - Nhấn OK để đóng Preferences.

   Cursor chỉ ra rằng vị trí con trỏ chuột được đặt vào đó khi chèn đoạn lệnh vào. Khi sử dụng, bạn chỉ cần gõ vào **sop** và ấn Ctrll+Space thì câu lệnh System.out.println("${cursor}") xuất hiện.

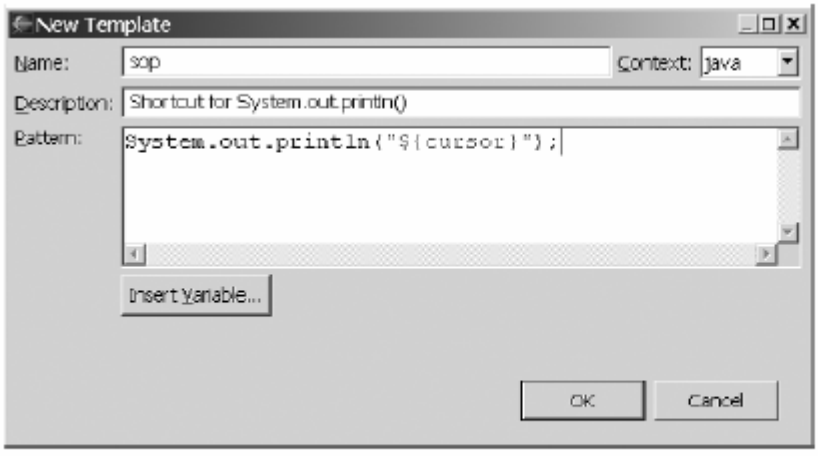


Figure 2.7 Creating a shortcut for `System.out.println()` using a Java editor template

Bạn có thể tạo mẫu Template cho cấu trúc vòng lặp hay cấu trúc điều kiện.

Ví dụ: Template mẫu của vòng lặp **for :**

```
for(int ${index}=0; ${index}< ${cursor}; ${index}++})
{}
```

4. **Đường dẫn và biến đường dẫn:** Để sử dụng lại các gói JAR thêm vào nhiều lần, trong nhiều Project khác nhau, bạn cần phải thiết lập biến đường dẫn trỏ tới các gói JAR này, ví dụ thiết lập đường dẫn cho gói JAR sử dụng cơ sở dữ liệu MySQL :
   - Windows → Preferences → Java → Classpath Variables
   - Nhấn nút New, đặt tên cho biến (**MYSQL_JDBC**) và trỏ đường dẫn tới vị trí gói JAR trên máy của bạn.
   - Nhấn OK kết thúc.

   Sau này mỗi khi cần sử dụng gói JAR này, bạn chỉ cần nhấp phải chuột trên tên Project → chọn Properties → Hộp thoại mới xuất hiện : chọn Java Build Path bên trái và tab Library bên phải → nhấn núy Add Variable → chọn **MYSQL_JDBC.** Ngoài ra, tại đây bạn có thể thêm vào các gói JAR ở bên ngoài ba72ng cách chọn Add External Jars.

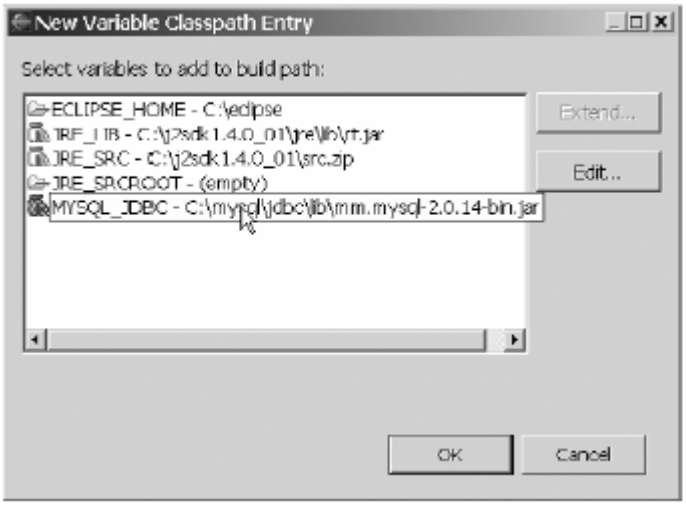


Figure 2.8
Creating a new classpath variable. Classpath variables make it easier to manage your classpath and provide flexibility as well.

5. **Lưu giữ và sử dụng các thông số:**
   Các bước trên cho thấy bạn thường bỏ ra nhiều thời gian để cấu hình các thông số cho phù hợp với các bạn. Do đó, Eclipse cho phép bạn lưu lịa cấu hình này để có thể cất giữ và sử dụng lại nhiều lần tại các máy khác nhau :

   - Chọn Windows → Preferences → chọn 1 trong 2 nút Import và Export
   - Export để lưu cấu hình ra 1 tập tin và Import giúp bạn đưa cấu hình từ 1 tập tin lưu trước đó vào Eclipse.

PHẦN III: ECLIPSE VÀ JUNIT

A. Định nghĩa và cài đặt vùng làm việc (working set):

Khi bạn đang có nhiều Project thì cửa sổ Package Explorer tỏ ra chật chội, khiến bạn khó nhìn và dễ nhầm lẫn khi chọn các tập tin. Eclipse đã đưa ra khái niệm mới : working set (vùng làm việc) để khắc phục chuyện này. Mỗi Project sẽ có một vùng làm việc khác nhau nếu bạn có định nghĩa working set cho từng Project. Như vậy, mỗi lần bạn chỉ nhìn thấy duy nhất một Project mà bạn đang làm việc. Các bước để định nghĩa một working set:

1. Nhấn chuột vào mũi tên đen bên trái cửa sổ Package Explorer.
2. Chọn Working Set → Hộp thoại mới xuất hiện → nhấn vào nút New.
3. Bạn sẽ được hỏi về kiểu working set, chọn Java → nhấn nút Next → Hộp thoại New Working Set xuất hiện yêu cầu đặt tên cho Working Set mới, rồi chọn tài nguyên bạn muốn thể hiện trong Working Set ở cửa sổ phía dưới, có thể là tập tin, thư mục con hay toàn bộ một Project.
4. Nhấn Finish→ nhấn OK.

Ngay lúc này, bạn chỉ còn thấy duy nhất một tài nguyên mà bạn đã chọn khi định nghĩa Working Set, muốn thấy lại hết tòan bộ các Project chỉ cần chọn Deselect Working Set treong menu phụ của cửa sổ Package Explorer.

B. Kiểm tra đơn vị (JUNIT) với Eclipse:

Eclipse cho phép ta cài đặt JUNIT để kiểm tra đơn vị các hàm Java ngay bên tron Eclipse.

1. Cấu hình JUNIT với Eclipse:

- Nhấn phải chuột trên tên Project cần kiểm tra và chọn Properties → Hộp thoại mới xuất hiện.
- Chọn Java Build Path bên cửa sổ phải và tab Libraries bên cửa sổ trái. Nhấn vào nút Add Variable.
- Nhấn New. Điền tên JUNIT vào tên biến và nhấn vào File để chọn gói JUnit JAR trong thư mục plugins của Eclipse, ví dụ : c:\eclipse\plugins\org.junit_3.8.1\junit.jar.
- Nhấn Open để chọn gói JAR và OK chấp nhận một biến mới.
- Tiếp theo bạn cần phải thêm một biến nữa dành cho gói JAR nguồn của JUnit, gói này cần trong việc debug chương trình. Nhấn nút New lần nữa, điền tên JUNIT_SRC vào và chọn tập tin junitsrc.zip trong thư mục JDT, ví dụ : c:\eclipse\plugins\org.eclipse.jdt.source_2.1.0\src\ org.junit_3.8.1\ junitsrc.zip.
- Nhấn OK để trở lại hộp thoại New Variable Classpath Entry.

Bây giờ, bạn sẽ gắn biến JUNIT và biến JUNIT_SRC vào đường dẫn lớp (classpath):

- Chọn biến JUNIT và ấn OK, trở lại cửa sổ Java Build Path trong Properties.
- Nhấn vào kí hiệu dấu + bên cạnh JUNIT, bạn sẽ thất không có Javadoc và nguồn đính kèm.
- Double-click vào Source Attachment và điền vào tên biến **JUNIT_SRC**, nhấn OK và kiểm tra đã đính kèm đúng tên tập tin junitsrc.zip
- Nhấn OK để lưu lại những thay đổi đường dẫn lớp và đóng cửa sổ Properties. Lúc này bạn sẽ thấy thư viện JUNIT được liệt kê trong cửa sổ Package Explorer.

2. Các bước tiến hành kiểm tra đơn vị:

a. Kiểm tra đơn vị sử dụng lớp Test Case:

Chúng ta dễ dàng tạo lớp Test Case theo các bước như sau :

- Nhấn phải chuột vào tập tin cần kiểm tra → chọn New → Other → Hộp thoại xuất hiện.
- Bạn nhấn vào dấu (+) mở rộng tùy chọn Java, chọn JUnit bên cửa sổ trái và chọn TestCase bên cửa sổ phải → Nhấn Next.

- Chọn các thư mục Project, các gói và tập tin cần kiểm tra. Nhớ đánh dấu kiểm chọn vào 2 ô setUp() và tearDown(). setUp() là phương thức tạo ra các đối tượng và dữ liệu cần kiểm tra, còn tearDown() là để hủy bỏ chúng. Nhấn Next tiếp theo.
- Hộp thoại mới xuất hiện cho phép bạn chọn các phương thức cần được kiểm tra đơn vị và lớp cha của chúng.
- Nhấn Finish kết thúc.
- Eclipse sẽ tự động sinh ra tập tin kiểm tra tương ứng mà mình đã chọn ở các bước trước với các phương thức setUp(),tearDown() và các phương thức kiểm tra.

Để chạy lớp Test Case, bạn phải chọn tập tin kiểm tra Test Case, trên thanh Menu chính chọn Run → Run As → JUnit Test, để gở lỗi bạn chọn Run → Debug As→ JUnit Test. Khi chạy JUnit, một cửa sổ mới xuất hiện cho biết kết quả kiểm tra có thành công hay không. Nếu thành công sẽ có thanh màu xanh. Ngược lại sẽ có thanh màu đỏ.

Ví dụ : Ta có một Project tên là Persistence với tập tin FilePersistenceServices nằm trong gói org.eclipseguide.persistence như sau:

```
package org.eclipseguide.persistence;
/**
* File-based persistence class
* Provides methods for maintaining records using files
*
* @author david
* @version 1.0 Dec 30, 2002
*/
import java.util.Vector;
public class FilePersistenceServices
{
	public static boolean write(String fileName, int key, Vector
v)
	{
		return false;
	}
	public static Vector read(String fileName, int key)
	{
		return null;
	}
}
```

Các hình dưới đây minh họa thực hiện tạo Test Case kiểm tra lớp FilePersistenceServices :

Figure 3.1
Creating a JUnit test case with the New JUnit Test Case Wizard

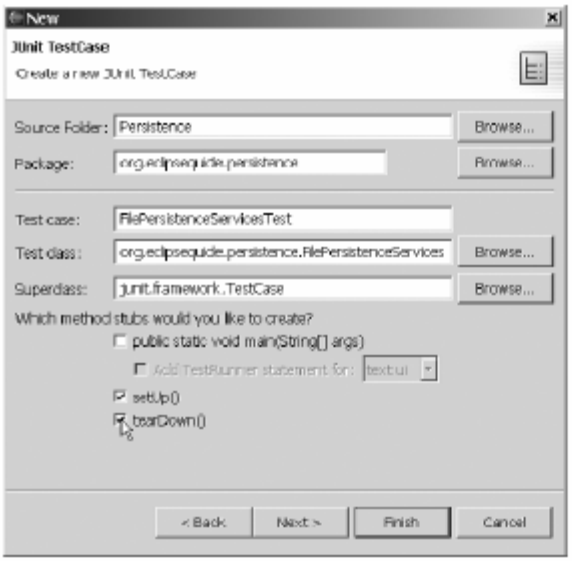


Figure 3.2
Defining the test case and the test class. The JUnit wizard can also provide method stubs for `setup()` and `teardown()` methods.

Figure 3.3
Adding test methods. Check the boxes for the test case methods you want to test.

Eclipse sẽ tạo ra Test Case với nội dung như sau:

```
package org.eclipseguide.persistence;
import junit.framework.TestCase;
/**
* Enter one sentence class summary here.
* Enter class description here.
*
* @author david
* @version Jan 3, 2003
*/
	public class FilePersistenceServicesTest extends TestCase
{
	/**
	* Constructor for FilePersistenceServicesTest.
	* @param arg0
	*/
	public FilePersistenceServicesTest(String arg0)
	{
		super(arg0);
	}
	/**
	* @see TestCase#setUp()
	*/
	protected void setUp() throws Exception
	{
		super.setUp();
	}
	/**
	* @see TestCase#tearDown()
	*/
	protected void tearDown() throws Exception
	{
		super.tearDown();
	}
	public void testWrite()
	{
	}
	public void testRead()
	{
	}
}
```

Bạn tự haàn chỉnh Test Case như sau:

```
// đưa và đầu class.
Vector v1;
// bổ sung vào phương thức setUp()
protected void setUp() throws Exception
{
	super.setUp();
	v1 = new Vector();
	v1.addElement("One");
	v1.addElement("Two");
	v1.addElement("Three");
}
```

```
// bổ sung vào phương thức testWrite()
public void testWrite()
{
     assertTrue(FilePersistenceServices.write("TestTable", 1, v1));
}
// bổ sung vào phương thức testRead()

public void testRead()
{
      Vector w = FilePersistenceServices.read("TestTable", 1);
      assertNotNull(w);
      assertEquals(w, v1);
}
```

b. Kiểm tra đơn vị sử dụng Test Suite:

Một khi bạn đã có nhiều lớp Test Case, bạn muốn tạo ra một bộ kiểm tra Test Suite để chạy tất cả chương trình kiểm tra trong cùng một lớp, bạn thực hiện các bước sau:

- Nhấp phải chuột vào gói chứa các lớp cần kiểm tra.
- Chọn New → Other → Hộp thoại xuất hiện.
- Chọn JUnit bên cửa sổ trái và Test Suite bên cửa sổ phải → nhấn Next.
- Đặt tên cho Test Suite, chọn tất cả các Test Case hiện có và nhấn Finish.
- Chạy Test Suite tương tư như chạy Test Cases.

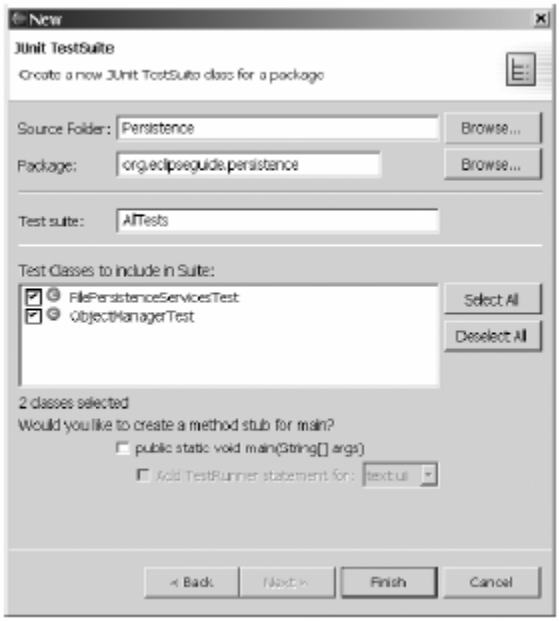


Figure 4.4
Creating the test suite class. The JUnit wizard automatically locates and includes all the test cases in the package.

PHẦN IV: THỰC HIỆN BUILD VỚI ANT TRONG ECLIPSE

Eclipse tích hợp sẵn công cụ build Ant hỗ trợ công việc build chạy ngay bên trong Eclipse mà không phả thiết lập các biến môi trường, các gói JAR, hay chạy rời một chương build nào khác.

Ví dụ : xây dưng tập tin build.xml cho Project HelloWord:

1. Nhấp phải chuột trên tên Project → chọn New → File , điền tên tập tin vào build.xml
2. Mở tập tin build.xml và gõ vào nội dung sau:

```
<?xml version="1.0"?>
<project name="Hello" default="print message">
<target name="print message">
<echo message="Hello from Ant!"/>
</target>
</project>
```

3. Eclipse tuy không hỗ trợ tốt cho việc soạn thảo XML như JAVA nhưng vẫn đảm bảo tính năng cơ bản nhất như chức năng hỗ trợ hoàn chỉnh mã chương trình khi nhấn *Ctrl* + *Space*.
4. Để chạy tập tin XML, bạn nhấp phải chuột trên tập tin XML đó và chọn Run Ant. Kết quả sẽ xuất hiện trên cửa sổ Console :

```
Buildfile: c:\eclipse\workspace\hello\build.xml
print message:
[echo] Hello from Ant!
BUILD SUCCESSFUL
Total time: 2 seconds
```

PHẦN V: ECLIPSE VÀ LOMBOZ

Lomboz là một plug-in khá hay giúp bạn phát triển ứng dụng web trên Eclipse khá dễ dàng và tiện lợi. Cái hay của Lomboz là nó hỗ trợ nhiều loại server, từ tomcat, Websphere, WebLogic, JBoss, Oracle 9iAS đến cả JRun, JonAS, Resin. Và quan trọng hơn cả là nó miễm phí như Eclipse.

Bạn có thể download Lomboz và Tomcat tại các trang web :

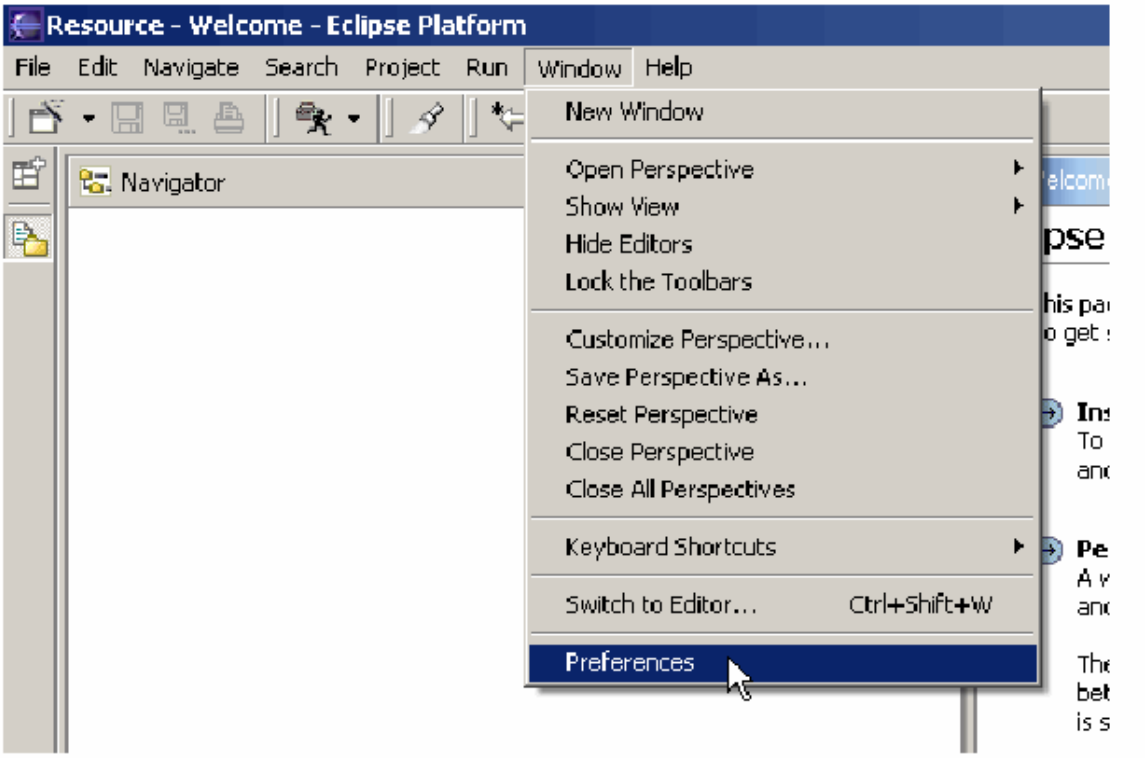
- Lomboz từ **www.objectlearn.com** (chú ý download đúng phiên bản Lomboz tương ứng với phiên bản Eclipse của bạn)
- Tomcat từ **jakarta.apache.org/tomcat**.

A. Cài đặt Lomboz:

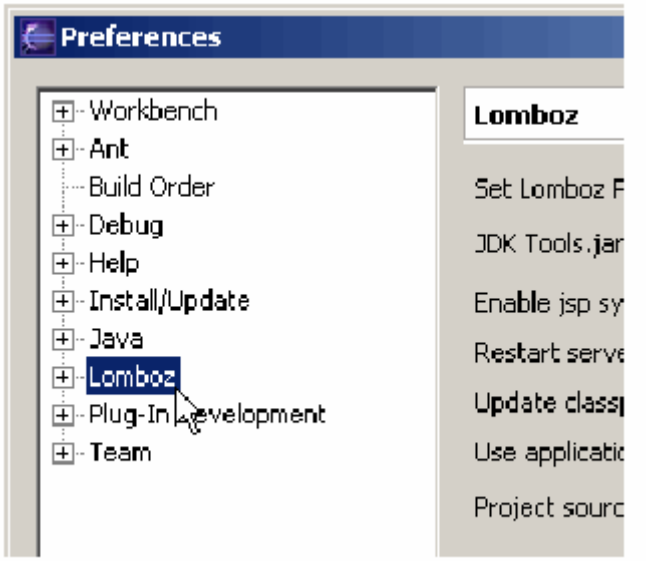
Giải nén tập tin lomboz.212.pl.zip lên ổ cứng, tạo ra một thư mục plugins. Copy thư mục con của plugins vào thư mục plugins của Eclipse. Xong cài đặt Lomboz. Bây giờ chạy eclipse.exe để hoàn thành cài đặt.

B. Cấu hình Lomboz chạy Tomcat:

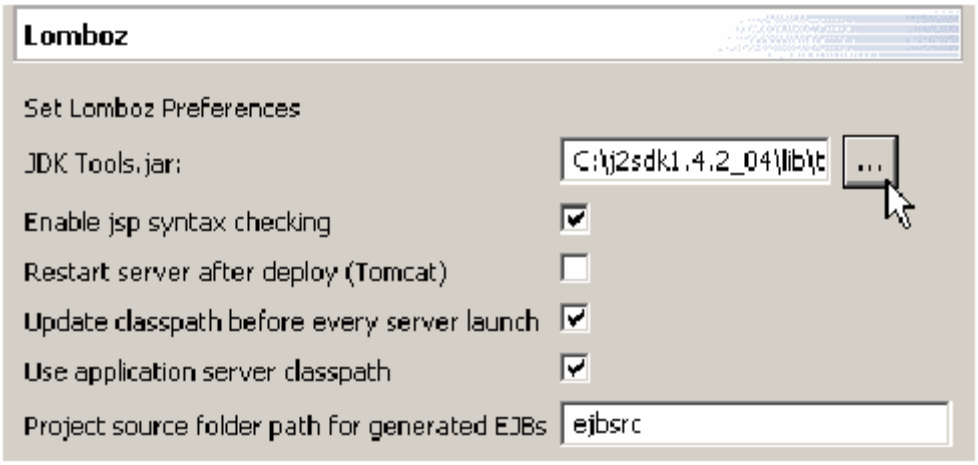
1. Menu Window → Preferences



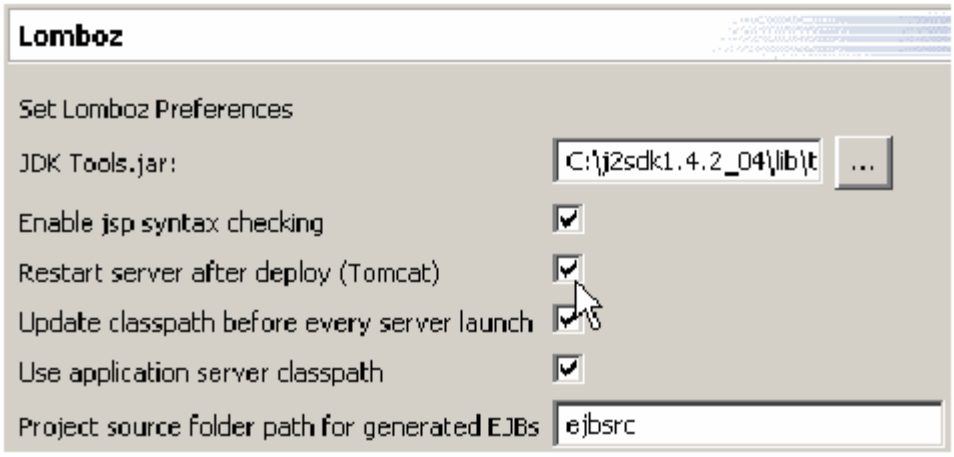
2. Chọn Lomboz.



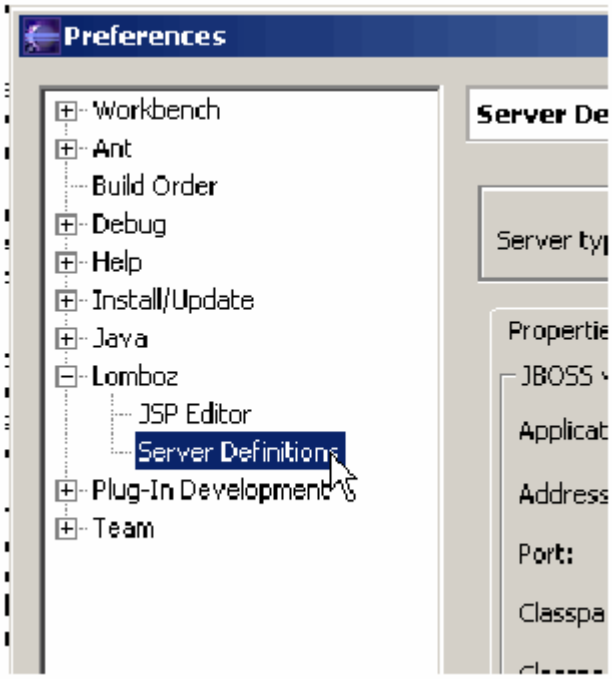
3. Trỏ **JDK Tools.jar** tới file **tools.jar** trong thư mục **lib của JDK.**



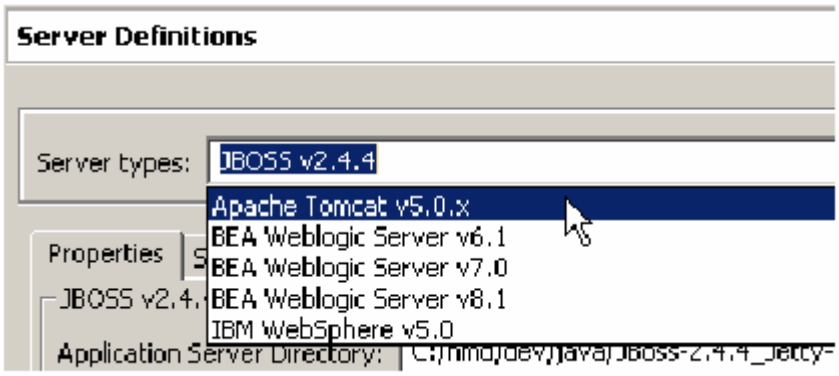
4. Nếu bạn muốn Tomcat tự restart sau khi deploy Project thì bạn chọn vào ô **Restart Ser after deploy (Tomcat)**



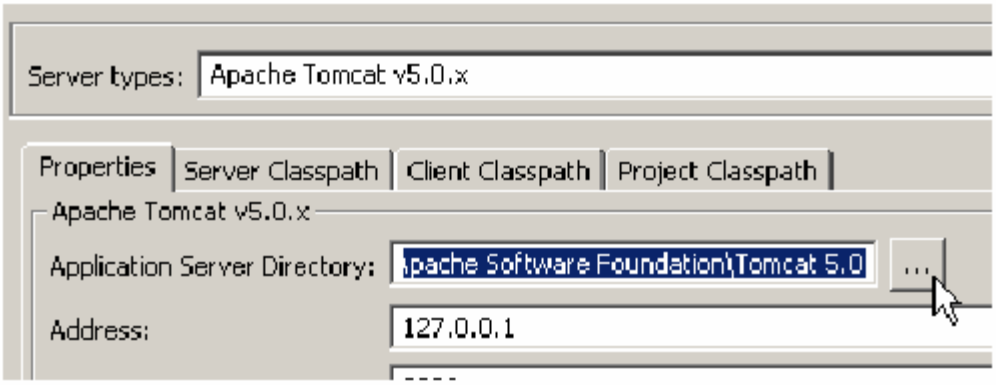
5. Nhấn vào dấu + bên cạnh Lomboz để chọn **Server Definitions**.



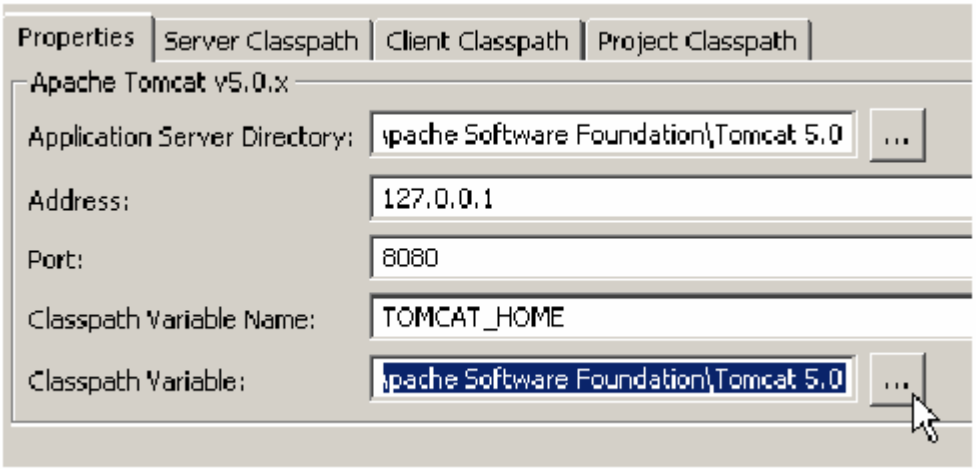
6. Chọn giá trị Server Type như hình dưới đây :



7. Trỏ đường dẫn Application Server Directory tới thư mục gốc của Tomcat :



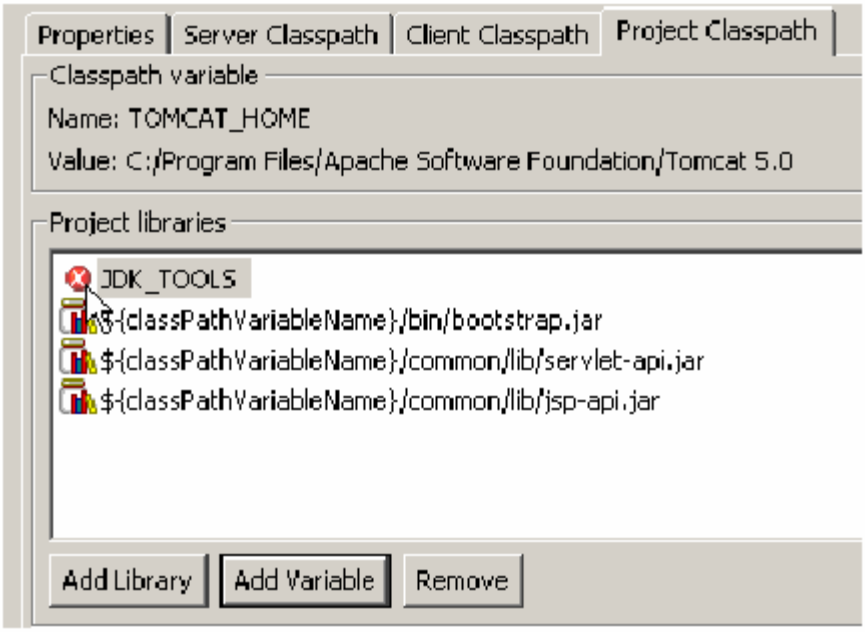
8. Trỏ đường dẫn Classpath Variable tới thư mục gốc của Tomcat :



9. Nhấp Apply rồi OK.

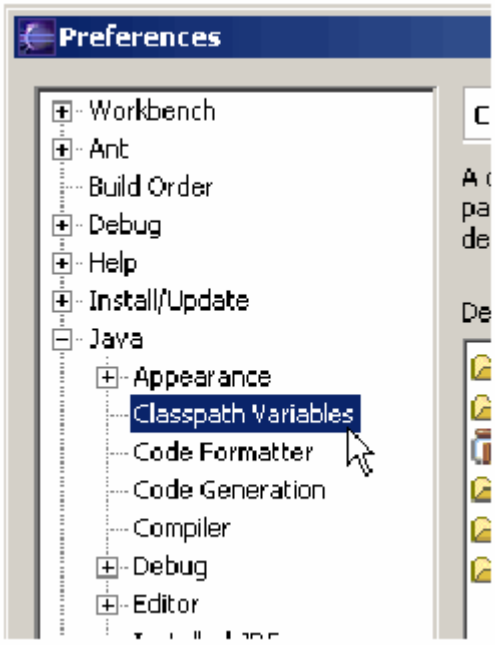
C. Thiết lập biến môi trường JDK_TOOLS:

Nếu bạn chuyển sang tab Project Classpath trong Server Definition, bạn sẽ thấy biến môi trường JDK_TOOLS có dấu màu đỏ.

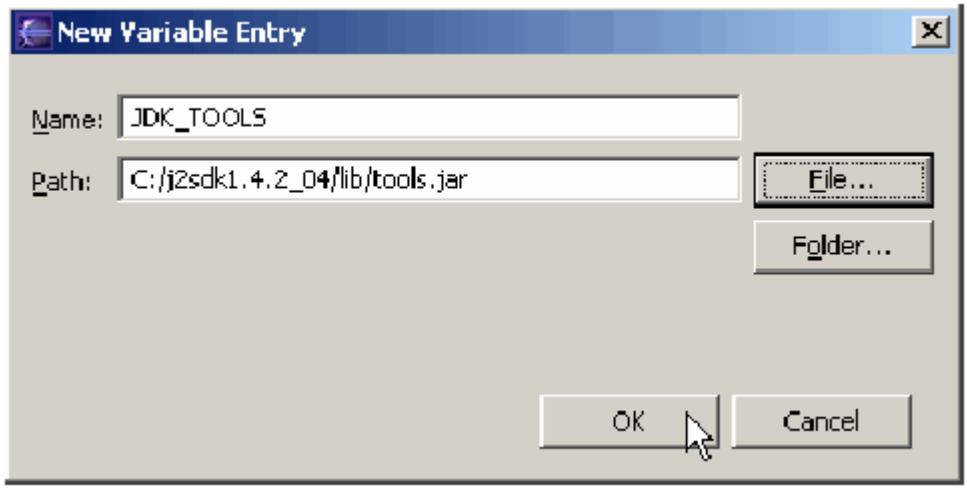


Biến này có nhiệm vụ giúp Tomcat compile trong quá trình chạy. đồng thời bạn cũng đỡ phải compile servlet và các class khác bằng tay. Bạn có thể không cần cài đầy đủ JDK, mà chỉ cần cài JRE và file tools.jar, Tomcat vẫn có thể chạy tốt.

Để thiết lập biến môi trường, bạn vào nhánh con Classpath Variables của Java.



Nhấn New và điền tên biến môi trường và trỏ đường dẫn tới file tools.jar. Sau đó nhấn OK.

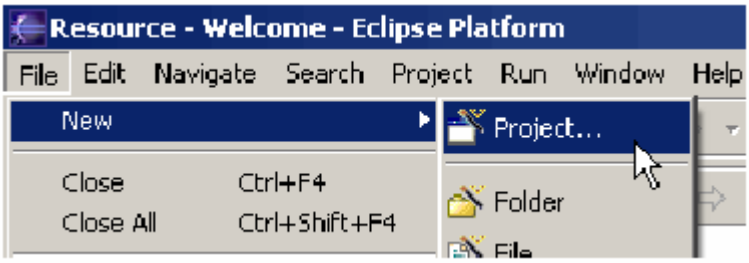


Và nhấn OK nữa để chấp nhận sự thay đổi.

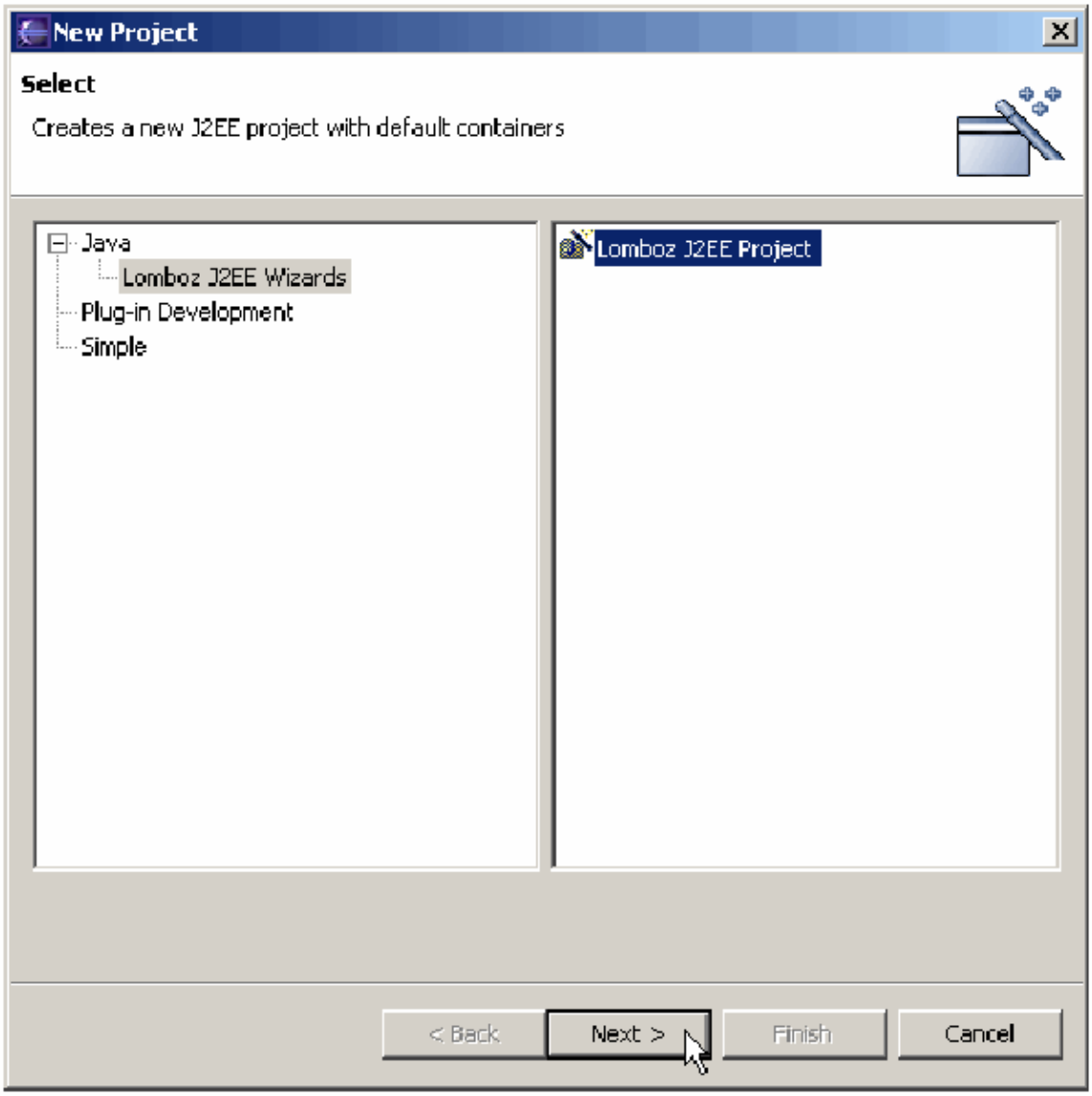
D. Tạo ứng dụng Web:

Để tạo ứng dụng Web trong Eclipse sử dụng lomboz, bạn thực hiện theo các bước sau:

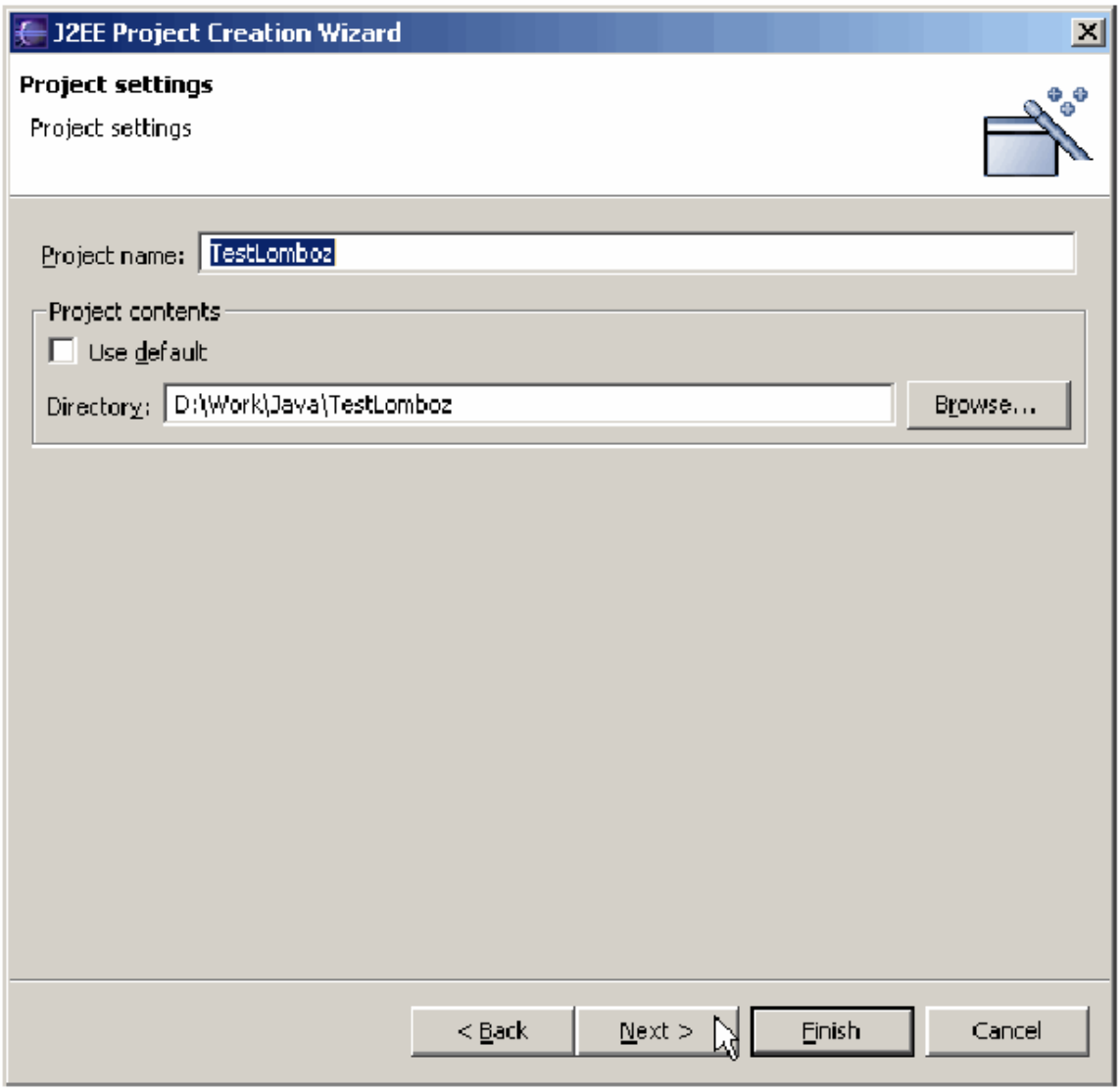
1. File → New → Project…



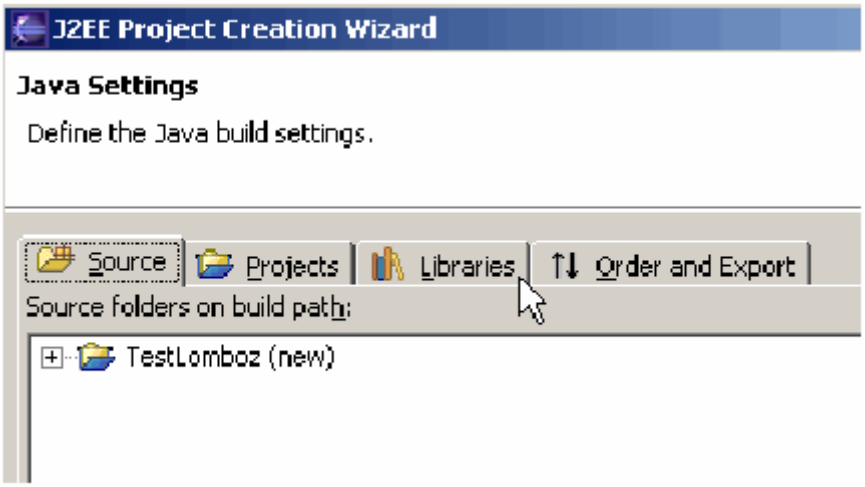
2. Chọn Lomboz J2EE Wizard.



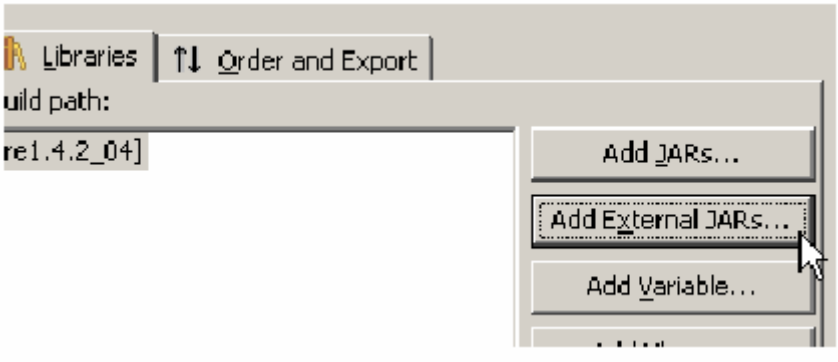
3. Nhấn Next. Nhập tên Project và thư mục lưu Project. Sau đó nhấn Next.



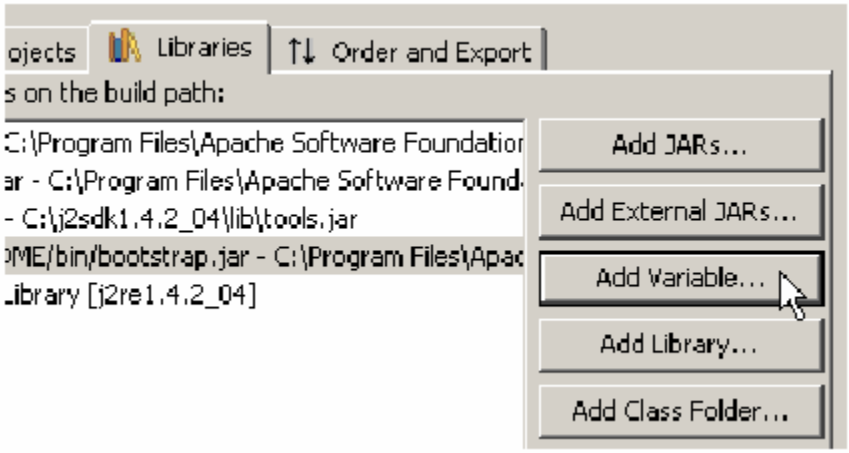
4. Trong cửa sổ Java setting chọn Libraries :



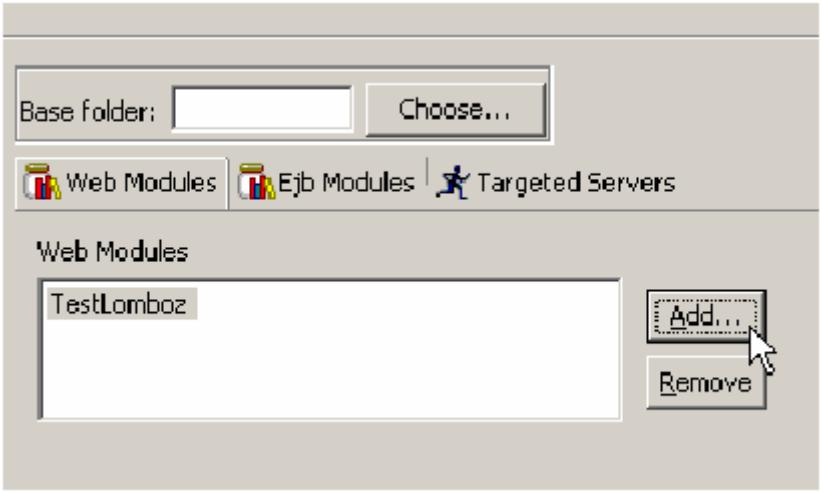
Bạn cần thêm 2 file servlet-api.jar và jsp-api.jar (đối với Tomcat5) bằng cách nhấn nút Add External JARs… 2 file này nằm trong thư mục {TOMCAT_HOME}\common\lib, trong đó {TOMCAT_HOME} là đường dẫn tới thư mục Tomcat. Sau đó nhấn Next.



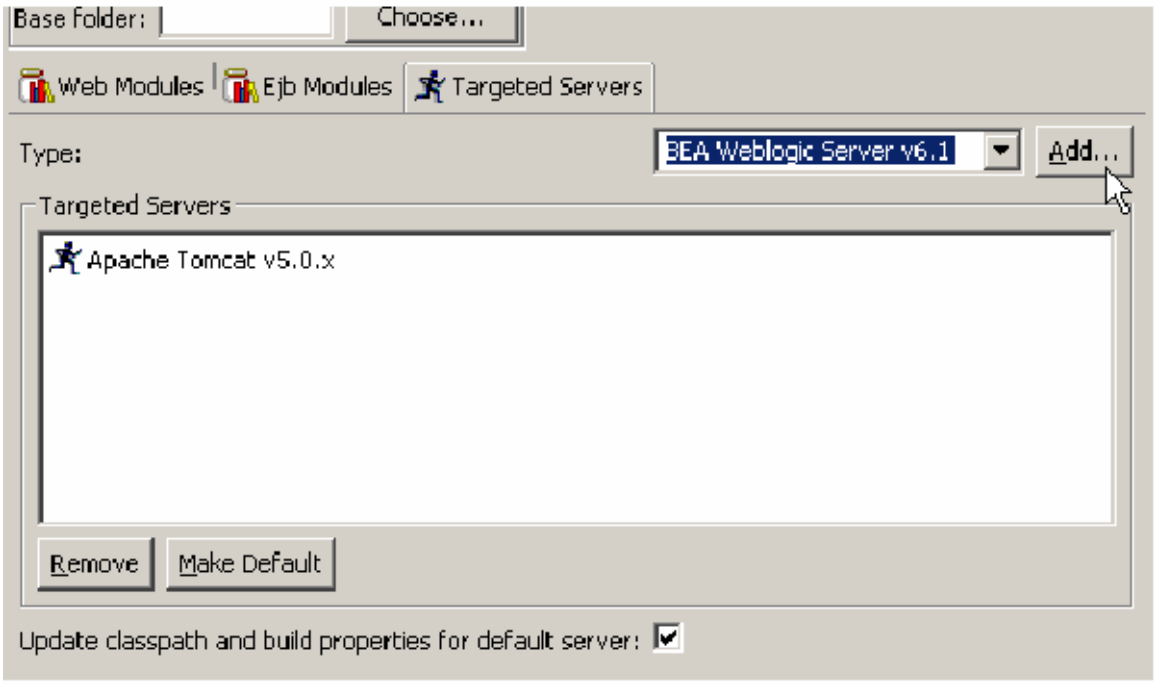
5. Đồng thời bổ sung 2 biến JDK_TOOLS và TOMCAT_HOME mà bạn đã định từ trước bằng cách nhấn nút Add Variable…chọn 2 biến này rồi nhấn OK.



6. Trong cửa sổ Create J2EE Module, bạn đặt tên cho web module bằng cách nhấn nút Add…Nhập tên web module, tên này sẽ tương ứng với web application mà bạn deploy lên Tomcat. Bạn có thể tạo nhiều web module trong một Project.



7. Tiếp thoe chọn Targeted Severs→ chọn **Apache Tomcat v5.0.x** trong Type rồi nhấn nút Add.



8. Sau đó nhấn Finish. Như vậy bạn đã hoàn thành các bước cài đặt, cấu hình và tạo các ứng dụng web.
9. Sau khi coding, bạn deploy project lên Tomcat bằng cách nhấn nút phải chuột trên web module muốn deploy, chọn Lomboz J2EE…, sau đó chọn chọn Deploy Module.

PHẦN 6: SỬ DỤNG CVS TRONG QUẢN LÝ SOURCE CODE

A. Sự cần thiết của việc quản lý Source code:

Quản lý source code thật sự cần thiết khi 2 hoặc nhiều người cùng làm việc chung trên 1 hệ thống file. Nếu như có 2 người cùng làm việc trên 1 file thì chỉ có công việc của người lưu sau cùng là được giữ lại. Hầu hết cách trao đổi cũng như quản lý source đơn giản là thông qua: email, tin nhắn hoặc meetings.

Hệ thống quản lý source (hay hệ thống quản lý version) không thể thay thế hệ thống thông tin truyền thống mà nó chỉ nhằm hổ trợ, cải thiện việc quản lý source code theo 2 cách sau: điều khiển việc truy xuất source bằng hệ thống khóa chỉ cho phép truy xuất tuần tự và lưu giữ những thay đổi trên mỗi file để có thể phục hồi khi có sự cố.

Khả năng bảo quản file là một đặc điểm nổi bật. Nếu có 1 lỗi được phát hiện, bạn có thể quay lại để kiểm tra xem vấn đề gì xảy ra trước khi lỗi xuất hiện. Version control cũng cho phép bạn nhập chú thích khi bạn kiểm tra code để mà bạn có thể thấy cái gì đã xảy ra trong code bị lỗi.

Version control cũng cho phép bạn phân chia project và phát triển song song. Bằng cách này, khi bạn chính thức phát hành sản phẩm, bạn có thể dễ dàng sửa lỗi phần bị lỗi mà không ảnh hưởng đến những phần chính khác

Lưu giữ sửa đổi thì rất quan trọng, để chắc chắn rằng các thay đổi không bị mất, một trong 2 cách sau đây có thể được thực hiện:

- Pessimistic locking – Chỉ một người có thể thay đổi copy của một file tại một thời điểm, đến khi người đó không còn làm việc trên file đó nữa và khóa file, không ai có thể thay đổi nhưng có thể đọc file.
- Optimistic locking – Mọi người tự do thay đổi file, nhưng khi lưu file phần mềm version control sẽ kiểm tra các sung đột và sẽ kết hợp 1 cách tự động hoặc báo cho bạn biết để bạn có thể sửa đổi bằng tay.

Eclipse hổ trợ optimistic locking. Version control có thể dễ dàng cài đặt trên Eclipse thông qua plug-in. để sử dụng CVS bạn phải có kết nối đến CVS server.

B. Sử dụng CVS với Eclipse:

Một vài bước cần thiết để thêm 1 project vào CVS sử dụng Eclipse. Bước thứ nhất là nhập 1 số thông tin cần thiết để Eclipse kết nối đến CVS. Thông tin này được lưu trữ trong 1 đối tượng gọi là Repository location. Sau khi tao repository location, tạo 1 module thích hợp trong CVS và đem các file trong project của bạn vào module.

- **Tạo một nơi lưu trữ:**

Để tạo một repository location bạn phải biết tên CVS server, đường dẫn của CVS repository, và protocol mà server sử dụng. Bạn cũng phải có một username và password cho server và CVS repository. Theo các bước sau:

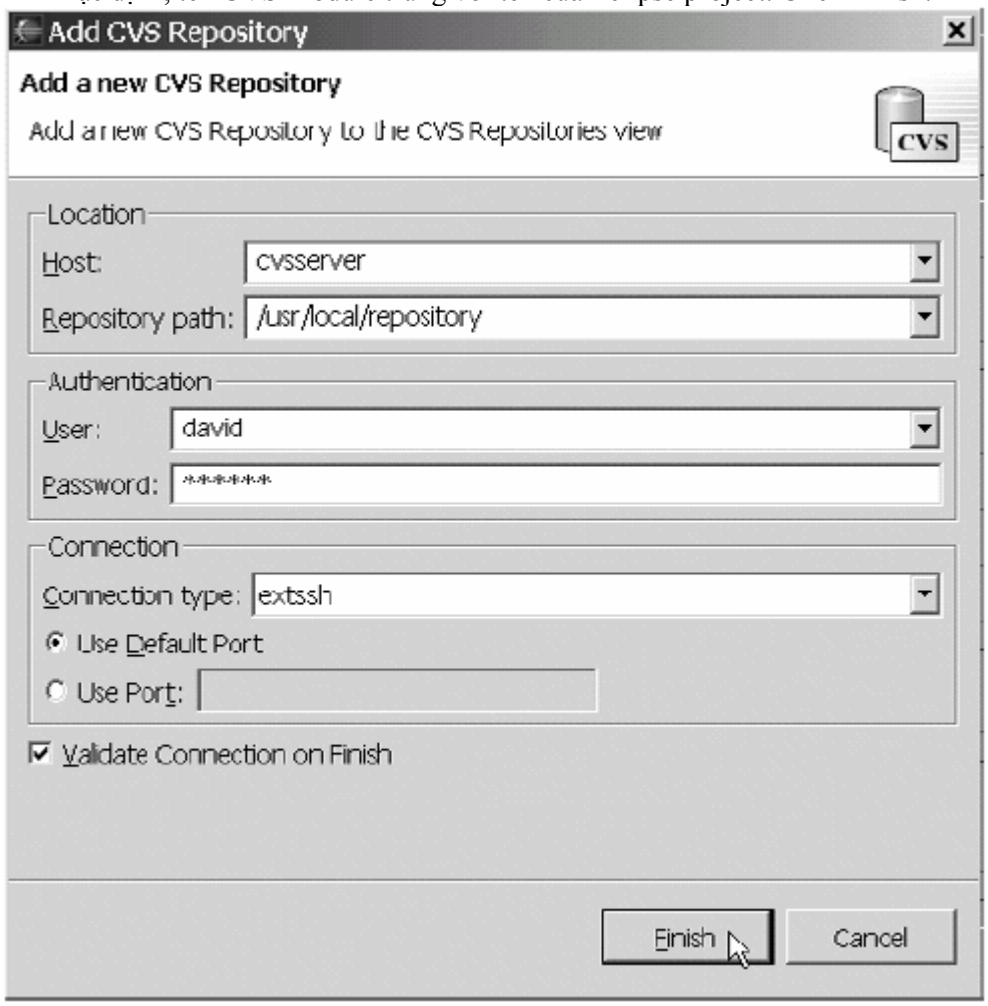
- Từ main menu, chọn Window->Open Perspective->Orther.
- Danh sách các perspective xuất hiện. Chọn CVS Repository Exploring và click OK.
- Trong CVS Repository Exploring View, click phải và chọn New->Repository Location.
- Nhập tên CVS server, đường dẫn repository, username và password. Chú ý: đường dẫn repository phải là đường dẫn đầy đủ, nghĩa là nó chỉ ra nơi repository lưu trử (VD: /usr/local/repository).
- Chọn protocol.
- Chọn port mặc định của CVS.
- Click Finish. Thông tin sẽ được lưu, Eclipse kết nối đến server, nếu thành công bạn sẽ thấy repository location như 1 entry mới trong CVS Repository view.

- **Chia sẻ project:**

Một khi bạn đã kết nối đến repository của bạn, bạn có thể thêm project của bạn vào CVS repository theo cách sau:

- Click phải vào project chọn Team->Share Project

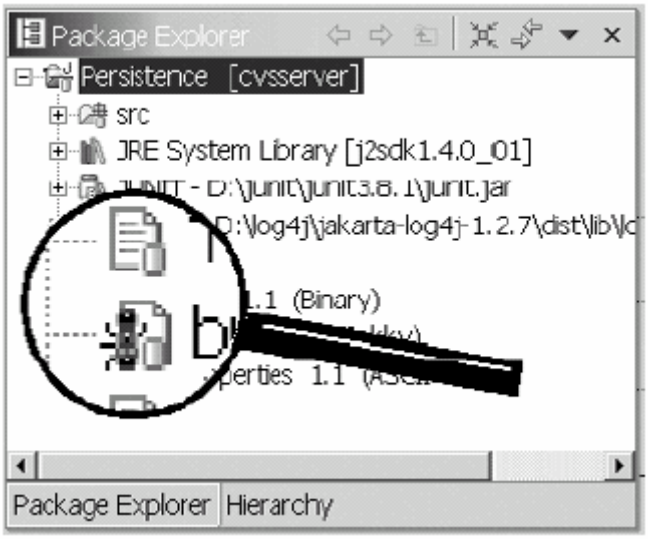
- CVS Repository xuất hiện, bạn nên chắc rằng Use Existing Repository Location và vị trí repository đã được chọn.
- Mặc định, tên CVS Module trùng với tên của Eclipse project. Click Finish.



Hình 6.1: Nhập thông tin repository. Bạn cần lấy 1 vài thông tin từ người quản trị CVS server.

Bước này tạo 1 module trên CVS server nhưng chưa có thêm bất kỳ file nào. Chú ý rằng Eclipse mở 1 cửa sổ CVS Synchronize bên dưới cửa sổ editor. Cửa sổ này cho phép bạn so sánh các file trong repository.

Tại lúc này Eclipse hiển thị các thông tin CVS trong Package Explorer, bạn có thể thấy version và các thông tin khác của file đã share.



Để hiện thị CVS label decorators:

- Chọn Windows->Preferences.
- Workbench->Label decoration.
- Chọn CVS và click OK

CVS console là nơi bạn có thể thấy các dòng lệnh Eclipse gởi nhận từ CVS. Nó rất hữu ích khi bạn muốn biết cái gì đang diễn ra và khi có lỗi. Để mở CVS console, chọn Window->Show view->Other->CVS->CVS console.

- **Thêm và commit file:**

Có 2 bước để tạo mới 1 file trong CVS:

- Thêm file vào CVS
- Commit file

Ngòai ra bạn có thể add nguyên cả project:

- Chọn Persistence project
- Click phải chọn Team->Commit

PHẦN VII: ECLIPSE PLUG-INS

A. Tìm hiểu về plug-in và extension point:

- Plug-in là những đơn vị có khả năng mở rộng trong Eclipse. Nó có thể chứa code, tài nguyên hay cả hai. Eclipse Platform gồm 100 plug in kết hợp với nhau.
- Extension point là 1 cơ chế cho phép 1 plug in có thể thêm các chức năng từ 1 plug in khác.

1. Plug-in:

Để tìm plug-in trong Eclipse, bạn xem trong thư mục plug-ins của Eclipse, tên các thư mục con tương ứng tên các plug-in và version của nó.

VD:

```
C:\ECLIPSE
|
+---features
|
+---plugins
| +---org.apache.ant_1.5.2
| | | lib\
| | | .options
| | | about.html
| | | antsupport.jar
| | | plugin.properties
| | | plugin.xml
| | |
| +---org.eclipse.ant.core_2.1.0
| |
| ...
+---workspace
```

Plug-in với tên org.eclipse.ant.core được tích hợp trong Eclipse như là 1 Ant Builder. Trong mỗi thư mục chứa plug-in đều có 1 file plugin.xml, file này là 1 file manifest dùng để mô tả plug-in như: tên, version. Nó cũng liệt kê tất cả các thư viện yêu cầu và các extension point được sử dụng cũng như được định nghĩa trong plug-in. Những file thường thấy trong thư mục plug-in:

+ plugin.xml – Mô tả plug in
+ plugin.properties – Chứa đựng các thông số hay thuộc tính được tham chiếu bởi plugin.xml
+ about.html – Thông tin về bản quyền
+ *.jar – Code của plug-in
+ lib – Thư mục chứa các file thư viện .jar
+ icons – Thư mục chứa các hình ảnh

2. Vòng đời của plug-in:

Khi khởi động Eclipse sẽ xem trong thư mục plug-ins và tạo 1 danh sách các plug-in, danh sách này được gọi là plug-in registry (danh sách được tạo dựa trên các file manifest). Plug-in được load khi chúng ta gọi nó và chỉ unload khi Eclipse đóng.

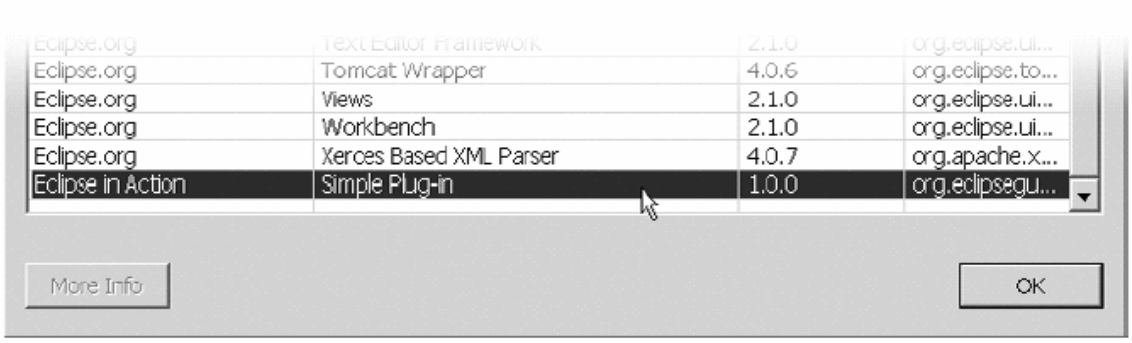
3. Tạo một plug-in đơn giản:

Eclipse plug-in có thể được tạo mà không cần bất kỳ công cụ đặc biệt nào. Chúng ta có thể là 1 plug-in đơn giản bằng cách tạo 1 thư mục org.eclipseguide.simpleplugin_1.00. Bên trong thư mục tạo 1 file plugin.xml có nội dung như sau:

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin
id="org.eclipseguide.simpleplugin"
name="Simple Plug-in"
version="1.0.0"
provider-name="Eclipse in Action">
</plugin>
```

Lưu file và khởi động lại Eclipse. Bạn sẽ không thấy bất kỳ thay đổi nào bởi vì plug-in không làm gì cả. Tuy nhiên bạn có thể biết được nó có được đăng ký hay chưa bằng cách chọn Help-

>About Eclipse Platform và sau đó chọn Plug-in details. Cuộn danh sách xuống và bạn sẽ thấy plug-in như hình:



Thông tin về file plugin.xml:

Line	Purpose
`<?xml version="1.0" encoding="UTF-8"?>`	Required XML prolog; never changes
`<plugin`	Starts defining a new plug-in
`id="org.eclipseguide.simpleplugin"`	Provides the fully qualified id for the plug-in
`name="Simple Plug-in"`	Gives the plug-in a human-readable name
`version="1.0.0"`	Specifies a version number
`provider-name="Eclipse in Action">`	Provides information about the author
`</plugin>`	Finishes defining the plug-in

B. Môi trường phát triển plug-in (PDE):

Eclipse tích hợp một vài tính năng giúp chúng ta có thể dễ dàng tạo những plug-in phức tạp. Nó được gọi là Plug-in Development Environment bao gồm các tính năng như: Plug-in Project, Fragment Project, Feature Project, Update Site Project.

1. Chuẩn bị Workbench:

Trước khi bắt đầu sử dụng PDE, bạn cần mở 1 số tính năng trong Eclipse. Chọn Windows → Preferences:

- Chọn Workbench->Label Decoration và mở Binary Plug-in Project decoration.
- Chọn Plug-in development->Compliers và chọn tất cả các Warning.
- Chọn Plug-in development->Java buid path control và mở Use Classpath Containers for Dependent Plug-ins Option
- Click OK.

2. Cài đặt SDK plug-in:

Eclipse Platform có một tập hợp rất nhiều các Plugins mà không cần phải download trước khi sử dụng, ngoài bạn hoàn toan có thể download thêm các plugins này và chép vào thư mục plugins của Eclipse. Tuy nhiên, đôi khi bạn thích để các plugins này như là một project quản lý trong thư mục workspace.

Để đưa các plugins này vào thư mục workspace, bạn chọn **File →Import → External plug-ins and Fragments** và nhấn **Next;** sau đó click chọn vào ô Copy **Plug-in Contents into the Workspace Location** và nhấn **Next.** Việc cài đặt vào như vậy được gọi là Cài đặt Nhị phân (*binary import*) và project được tạo theo cách này gọi là Plug-ins Nhị phân (*binary plug-ins*). Sau này nếu bạn không muốn để chúng trong thư mục workspace nữa, bạn hoàn toàn có thể xóa chúng đi, mà

không có ảnh hưởng nào đến Eclipse. Bạn cũng có thể tạm thời giấu chúng đi, trong Package Explorer chọn **Filters**, bật lên tùy chọn **Exclude Binary Plug-in and Feature Projects**, và nhấn **OK**.

3. **Tạo một Plug-in mới :**
   - Chọn File → New → Project → mở ra hộp thoại New Project Wizard

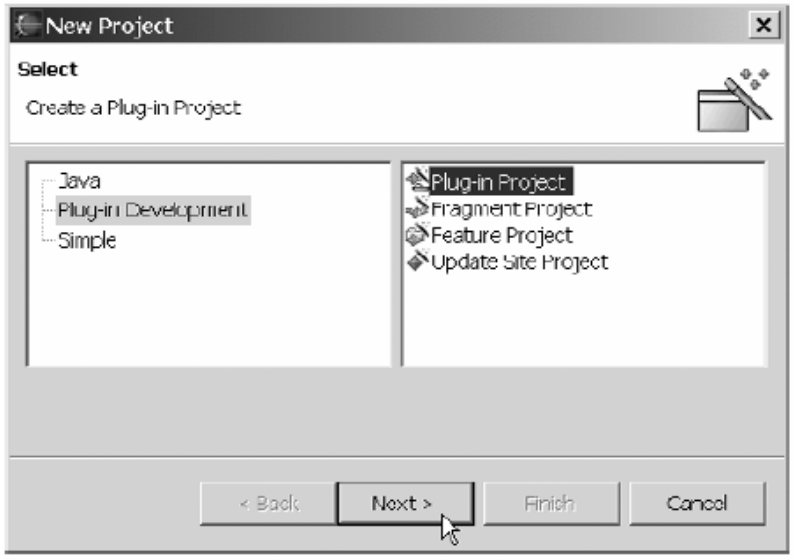


Figure 8.3
The PDE provides several wizards for creating new projects. See section 8.2 for a description of each type of project supported.

   - Chọn Pug-in Development bên trái → chọn Plu-in Project bên phải. Nhấn Next.
   - Nhập vào tên Plug-in như hình dưới đây, nhấn Next.

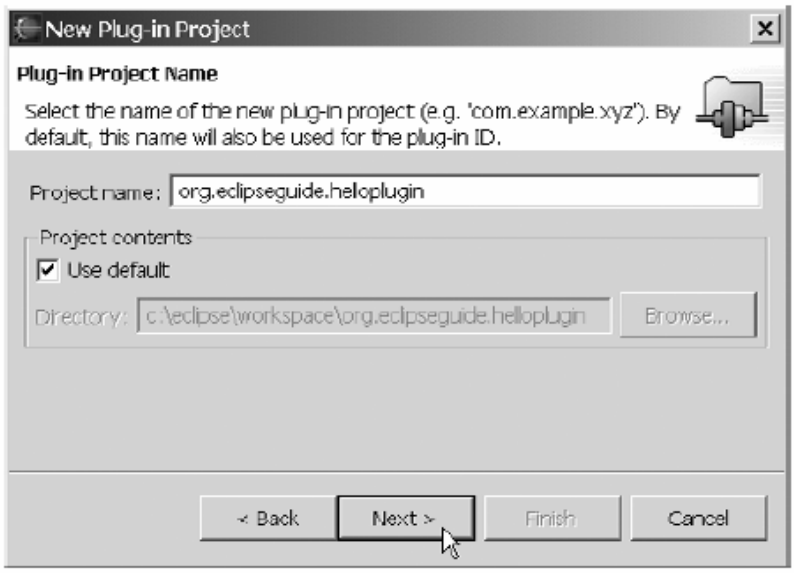


Figure 8.4
Specify the name and location of the project in the first page of the Plug-in Project Wizard.

   - Gõ vào tên ID, mặc định trùng với tên Project, nhấn chọn Create a Java Project. Các thông số khác để mặc định như hình dưới đây, nhấn Next tiếp tục.

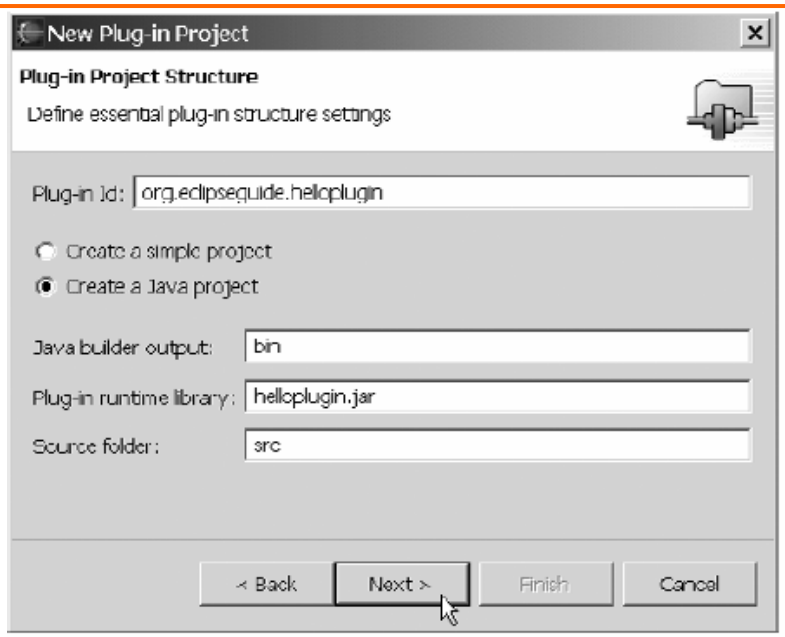


Figure 8.5
Specify the fully qualified ID of the plug-in in the second page of the New Plug-in Project Wizard. You can also control whether this plug-in will contain Java code.

- Cửa sổ mới xuất hiên cho phép bạn chọn lựa một số mẫu plugin có sẵn hỗ trợ cho việc tạo plugin dễ dàng hơn:

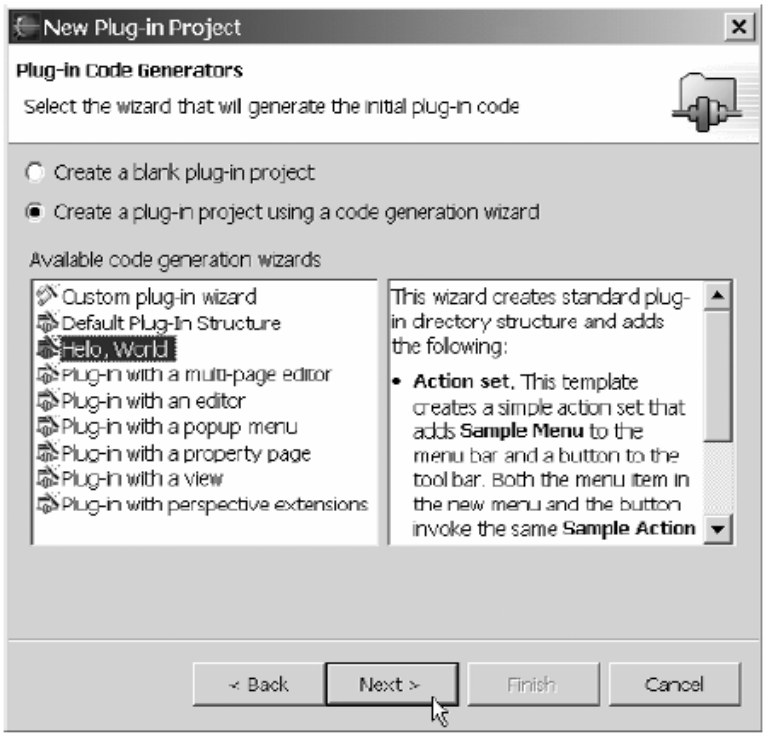


Figure 8.6
Select a code-generation wizard to quickly create a new plug-in from a template. Using the templates lessens the learning curve for Eclipse extensions and is less error-prone than creating the code from scratch. There also is an experimental feature in Eclipse 2.1 for adding your own wizards to this dialog.