Java Integrated Development Environments: ECLIPSE
Part 1

Installation
Eclipse Installation

- Detailed Instructions
  http://www.3plus4software.de/eclipse/installation_en.html
- Basic Ingredients:
  - Windows, Linux, Solaris, QNX or Mac OS/X operating system.
  - 256 MB RAM or more
  - A Java 2 runtime environment (JRE) or Java 2 Software Development Kit (J2SDK). Eclipse needs version 1.3 or higher.
  - Eclipse SDK version 2.1.3 (the last stable version)
Eclipse Installation

- Extract eclipse-SDK-**-*.zip in the path that will now more be the `%eclipse_home%`. An `eclipse` directory will be created.
  - *E.g. C:\eclipse*
Eclipse Installation

- Eclipse is a Java program. You have to specify the Java path. There are two ways:
  - 1st: Specify the Java path in the PATH environment variable
  - 2nd: Copy the jre folder from the JAVA_HOME directory to the ECLIPSE_HOME directory

- Run eclipse; Click on the eclipse icon
Eclipse Installation

- Alternative. Run eclipse from a command line window specifying the Java path (-vm)
- Gives flexibility in specifying various parameters when starting eclipse
For more details on running eclipse visit this:

The eclipse site http://www.eclipse.org
Part2

Workbench
Workbench

This page will help familiarize you with the Eclipse Workbench. To get started, read the sections below and click on the related links.

**Installed Features**

To find out more about the features installed in your workbench, check and select the feature you are interested in.

**Perspectives, views and editors**

Tasks (0 items)

<table>
<thead>
<tr>
<th>Description</th>
<th>Resource</th>
<th>In Folder</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Workbench

• Editors
  – A visual component within the workbench
  – Used to edit or browse a resource

• Views
  – A visual component within the workbench
  – Used to navigate a hierarchy of information, open an editor or display properties of an active editor

• Perspectives
  – Groups of views an editors within the workbench
A simple example...

This is a sample text file.

The name of this file is sample.txt and the name appears in the tab above. A * indicates that the editor has unsaved changes.
Creating resources

Three alternatives
1. File menu
2. Navigator context menu (right-click on selected resource)
3. New Wizard button
Creating a new project

- From the menubar, select File > New > Project...
Creating a new project
Creating a new project

This page will help familiarize you with the Eclipse Workbench. To get started, read the sections below and click on the related links.

- **Installed Features**
  To find out more about the features installed in your workbench, choose Help > Welcome... and select the feature you are interested in.

- **Perspectives, views and editors**
  A window contains one or more perspectives. A perspective consists of views (e.g. Navigator) and editors for working with your resources.

  The shortcut bar at the far left of the window allows you to open new perspectives and move between perspectives that are already open. The perspective you are currently working with is shown in the title of the window and in the shortcut bar as a pushed-in icon.

- **Configuring your perspectives**

<table>
<thead>
<tr>
<th>Tasks (0 items)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="description" alt="Description" /></td>
</tr>
<tr>
<td>!</td>
</tr>
</tbody>
</table>
Creating a new folder

- From the popup menu of a project in the navigation view, select New > Folder
Creating a new folder

This page will help familiarize you with the Eclipse Workbench. To get started, read the sections below and click on the related links.

- **Installed Features**
  To find out more about the features installed in your workbench, choose Help > Welcome... and select the feature you are interested in.

- **Perspectives, views and editors**
  A window contains one or more perspectives. A perspective consists of views (e.g. Navigator) and editors for working with your resources.

  The shortcut bar at the far left of the window allows you to open new perspectives and move between perspectives that are already open. The perspective you are currently working with is shown in the title of the window and in the shortcut bar as a pushed-in icon.

- **Configuring your perspectives**
Creating a new file

- From the toolbar use the "New" Wizard button and select "File"
Creating a new file
Creating a new file

This is my first file
zowie!
Editors are tiled using tabs (if there is more than one file opened)

The appropriate editor is used according to the type of file
Editors
Stack editors

- Possible both side by side and one above the other
Stack editors

- Possible both side by side and one above the other
Views help you work with resources in addition to editors.
Views (example)

- The Navigator view displays the projects and other resources that you are working with.
Views (example)

- The bookmark view displays all bookmarks in the workbench along with the name of the file the bookmark is associated.
Views

- Display views from Window > Show View menu
Views

- Experiment as much as you want with the views
- Rearrange the Workbench using the Window > Reset Perspective menu
Q: Where are my files actually stored?

A:

1. Open your file system explorer
2. Navigate to the location where you installed eclipse (%ECLIPSE_HOME%). You will find a sub-directory called “workspace”
3. You will find the projects, folders and files in that sub-directory
Import files

1. Drag and Drop files from the file system
2. Copy and Paste files from the file system
3. Use the import wizard either from the popup menu of the navigator or from the File menu
Import files
Import files
Import files

Import from directory

Source must not be empty.

From directory: 

Into folder: Stamatis1/StamatisImport

Options:
- Overwrite existing resources without warning
- Create complete folder structure
- Create selected folders only

Folder: My Documents

Make New Folder  OK  Cancel
Import files

Import resources from the local file system.

From directory: C:\Documents and Settings\karvoun\My Documents

Filter Types... Select All Deselect All

Into folder: Stamatis1\StamatisImport

Options:
- Overwrite existing resources without warning
- Create complete folder structure
- Create selected folders only

< Back Next > Finish Cancel
Import files
If you do not want the contents of the folder to be copied in the “workspace” you have to “link” the folder to a position in the file system upon its creation.
Link folder to file system
Link folder to file system

- Click “Advanced”
- Check “Link to folder in file system”
- Browse to find the desired folder
Link folder to file system

![Link folder to file system dialog box](image)
Link folder to file system
Link folder to file system

<table>
<thead>
<tr>
<th>Description</th>
<th>Resource</th>
<th>In Folder</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Line</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Second Line</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Third Line</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fourth Line</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fifth Line</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Link folder to file system

- If you sneak in the file system explorer you will see that there is no folder “StamatisLink” in the “workspace” folder. Any change will be automatically made to the resources linked in the file system.
Part 3
Java Development
Prepare Workbench

- **Switch to Java Perspective using**
  Window > Open Perspective > Java Menu
Prepare Workbench
Prepare Workbench

- **Java perspective has the following views**
  - Package Explorer
  - Hierarchy
  - Outline
  - Search
  - Console
  - Tasks
Prepare Workbench

- Verify JRE installation and classpath variables
- Select menu Window > Preferences
Prepare Workbench

- Select Java > Installed JREs
- Confirm that a JRE has been automatically detected.
- The use of an SDK rather than a JRE is recommended.
Prepare Workbench

- Select “Workbench”
- Make sure “Perform build automatically on resource modification” is checked
Code Organization

- Projects
- Packages
- Classes
Create a new Java project

- Projects are the containers of packages
- All java projects are directly stored inside a project
- Generated CLASS files are stored along with the JAVA source files
- Two ways to create a new project
  1. From the File > New > Project menu
  2. From the “New Java Project” icon in the toolbar
Create a new Java project

- Select “Java”, then Next
Create a new Java project

- Choose whether you will “import” content in the project. In this case the content is NOT copied in the workspace.
Create a new Java project

- Add any external jars necessary to run your code (if any)
- Click “Finish”
Create a new Java project
Create a new Java package

- Choose the project that will contain the package from the Package Explorer

- Three ways to create the new package
  1. From the pop-up menu (right-click on the project) select New > Package
  2. From the “File” menu select New > Package
  3. From the toolbar using the “New Package” icon
Create a new Java package

- Give a name to the package
- Click “Finish”
Create a new Java package
Create a new Java Class

There are various ways to create a new Java Class
- Select the package where the new class will reside
  1. Use “New Class” icon in the workbench toolbar
  2. From the package’s pop-up menu (right-click) choose New > Class
  3. From the toolbar select the “New” icon’s drop-down and then select “Class”
  4. From the toolbar select the “New” icon and then “Java” in the left pane and “Class” in the right pane
  5. Select File > New > Class menu

You can always type a new class in an existing Java file from the Package Explorer
Create a new Java Class
Create a new Java Class

- Select a class name
- Select possible superclass
- Select possible implementing interfaces
- e.t.c
Create a new Java Class

```java
package stamatisPackage;

/**
 * @author karvoun
 * To change the template, right-click the file and select "Edit Template..."
 */

public class Hello {

    public static void main(String[] args) {
    }
}
```
Java editor features

- Import assistance
Java editor features

- Method Completion (automatically or using Ctrl+Space)
Java editor features

- Hovering over identifier shows Javadoc spec
Java editor features

- On-the-fly spell check

Hover over this to see mistake
Java editor features

- Quick Fix

Double click here to get suggestions.
Java editor features

- Code templates (Ctrl+Space)
Java editor features

- Argument hints
Java editor features

- JavaDoc Assist
Class Hierarchies
Class Hierarchies

Java Class
Create a new Java class.

Source Folder: StamatisJava
Package: stamatisPackage
Enclosing type:

Name: HelloWorld
Modifiers: public
Superclass: stamatisPackage.Hello
Interfaces:

Which method stubs would you like to create?
- public static void main(String[] args)
- Inherited abstract methods

Matching types:
- Hello
- Help

Qualifier:
stamatisPackage - StamatisJava

Ok | Cancel

Finish | Cancel
Class Hierarchies

```java
package stamatisPackage;

/*
 * Created on 11 Mar 2004
 * To change the template for this generated file
 * Window&gt;Preferences&gt;Java&gt;Code Generation
 */

public class HelloWorld extends Hello {
    public static void main(String[] args) {
        HelloWorld hw = new HelloWorld();
        hw.print("Hello World");
    }
}
```
Class Hierarchies
Class Hierarchies
Class Hierarchies

- Select a type and from its pop-up select “Open Type Hierarchy”
- The Hierarchy View will appear
Class Hierarchies
Search for Java Elements

- Declarations or References
Search for Java Elements

- Select a type or method
- From its pop-up select Search > References > Workspace
- You will retrieve all its references in the Search View
Search for Java Elements
Run Java Programs

- Build is performed automatically upon resource modification
- If you want to do it explicitly select menu Project > Rebuild project
- Two ways to run a project:
  - Select menu Run > Run As > Java Application
  - Click on the “Run” icon in the toolbar
Run Java Programs
Run Java Programs

- The Console View shows up with the output!
Part 4

UML
Eclipse UML Installation

- Exit eclipse
- Unzip EMF zip in %ECLIPSE_HOME% directory
- Re-start eclipse
- Exit eclipse
Zip File

Eclipse 2.1.1 is required.

- Get Eclipse 2.1.1 R-2.1.1-200306271545.

The EMF (Eclipse Modeling Framework) plugin is required.

- Get EMF 1.1.0 Build 20030620 1105VL.

The proper GEF (Graphical Editor Framework) plugin is included in our zip file.

Previous GEF installation should be uninstalled.

<table>
<thead>
<tr>
<th>TYPE</th>
<th>VERSION</th>
<th>DATE</th>
<th>SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>EclipseUML</td>
<td>1.2.1.20031103</td>
<td>11/04/2003</td>
<td>9,932,806 Bytes</td>
</tr>
<tr>
<td>EclipseUML</td>
<td>1.2.1.20030806</td>
<td>08/06/2003</td>
<td>10,769,882 Bytes</td>
</tr>
</tbody>
</table>

Eclipse 2.1 is required.

- Get Eclipse 2.1 R-2.1-200303272130.

The EMF (Eclipse Modeling Framework) plugin is required.

- Get EMF 1.1.0 Build 20030620 1105VL.

The proper GEF (Graphical Editor Framework) plugin is included in our zip file.
Eclipse UML Installation

Eclipse Modeling Framework (E...)

EMF is a Java/UML framework for generating tools and other applications based on simple class models.

What is EMF?
Read the overview article to learn about EMF. Then, follow the tutorial to see how easy it is to generate a model.

Learn More
This is the official site of EMF: http://www.eclipse.org/emf
Eclipse UML Installation

- Unzip Eclipse UML zip in %ECLIPSE_HOME% directory
- Re-start eclipse
- That’s it!

More details in:
- http://www.omondo.com/faq/free/index.jsp#emfzipi
Zip File

**Eclipse 2.1.1** is required.

- Get **Eclipse 2.1.1 R-2.1.1-200306271545**.

The **EMF (Eclipse Modeling Framework)** plugin is required.

- Get **EMF 1.1.0 Build 20030620 1105VL**.

The proper **GEF (Graphical Editor Framework)** plugin is included in our zip file.

Previous GEF installation should be uninstalled.

<table>
<thead>
<tr>
<th>TYPE</th>
<th>VERSION</th>
<th>DATE</th>
<th>SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>EclipseUML</td>
<td><strong>1.2.1.20031103</strong></td>
<td>11/04/2003</td>
<td>9,932,806 Bytes</td>
</tr>
<tr>
<td>EclipseUML</td>
<td><strong>1.2.1.20030806</strong></td>
<td>08/06/2003</td>
<td>10,769,882 Bytes</td>
</tr>
</tbody>
</table>

**Eclipse 2.1** is required.

- Get **Eclipse 2.1 R-2.1-200303272130**.

The **EMF (Eclipse Modeling Framework)** plugin is required.

- Get **EMF 1.1.0 Build 20030620 1105VL**.

The proper **GEF (Graphical Editor Framework)** plugin is included in our zip file.
Eclipse UML Installation

Omondo EclipseUML

EclipseUML is a native Eclipse plug-in for UML modeling.

What is EclipseUML?
EclipseUML is a plugin for UML modeling in Eclipse 2.1.
It can be used both for creating new models and for reverse engineering existing models.
This is a release of version 1.2.1 including:
- Class diagram (live synchronization between the class diagram and the Java code)
- Sequence diagram
- Use-case diagram
- Statechart diagram
- Collaboration diagram
- Activity diagram
- Object diagram
- Component diagram
- Deployment diagram
This version also includes a full EMF graphical editor.

Learn More
This is the official site of EclipseUML: http://www.eclipseuml.com/
Create your first class diagram

- Select a package
- From it context menu (right-click) select New>Other>UML->UML Class Diagram
Create your first class diagram
Create your first class diagram
Create your first class diagram

- Choose the kind of relationship you want to be depicted
  - Association
  - Inheritance
  - Dependency
Choose the classes that will participate in the diagram
Create your first class diagram

Here is it!
Prepare Workbench for UML design

- Select menu Window > Preferences
Prepare Workbench for UML design

- We will create a src and a bin output for the UML project
- Select Java > New Projects
- Click on “Folders” checkbox
Prepare Workbench for UML design

- Select “UML”
- Click on checkbox “with text”
Prepare Workbench for UML design

- Select UML>Class Diagrams>Class

- In attributes pane check boxes “Public”, “Private”, “Protected” and “Package”
Eclipse UML Class Diagrams

- Select Java Perspective
- Create a new Java project named “Company”

The src folder is created automatically.
Eclipse UML Class Diagrams

- Select src folder
- From its pop-up menu select UML > Class diagram editor
Eclipse UML Class Diagrams
Eclipse UML Class Diagrams

- Create a New Package.
- Select “new package” in the toolbar and click on the diagram.
- Name it “model”
Eclipse UML Class Diagrams
Eclipse UML Class Diagrams

- Create a new Class Diagram inside the “model” Package
- Right-click on model then Open>model.ucd (ucd is the class diagram extension).
Eclipse UML Class Diagrams
Eclipse UML Class Diagrams

- Create a New class
- Select New class in the toolbar and click on the diagram.
Eclipse UML Class Diagrams

- Enter the name of the New Java Class and check for Person and click on the Finish button.

![New Java Class Window](image)
Eclipse UML Class Diagrams

- We are going to add two new Attributes.
- Click on Person > New > Attribute.
Eclipse UML Class Diagrams

Enter Attributes properties: "age: int" and "name: String"
Eclipse UML Class Diagrams
Eclipse UML Class Diagrams

- Create a New class named Employee.
- We would like to add inheritance from Employee to Person.
- Select Generalization in the Toolbar
Drag the Inheritance from Employee to Person.
Eclipse UML Class Diagrams

- Add "salary: float" attribute to Employee
- Create a New class named Company.
We want to create a new Association between Company and Employee.

Select Association in the Toolbar then drag from Company to Employee.
Eclipse UML Class Diagrams
Eclipse UML Class Diagrams

- It is possible to add a name to the Label, but we will not in this example.
Eclipse UML Class Diagrams

- Select the “1st Association End” tab
- Set multiplicity to 1
Eclipse UML Class Diagrams

- Select the “2nd Association End” tab
- Set name “employees”
- Set multiplicity “0..*”
Eclipse UML Class Diagrams

```
Person
- age: int
- name: String

getAge()
getName()
setAge()
ssetName()

Company
- employees: Collection

setEmployees()
getEmployees()

Company

Employee
- company: Collection
- salary: float

getCompany()
getSalary()
setCompany()
setSalary()

0..*

Person

- company

1

employees

Person

Employee

Company
```
Eclipse UML Class Diagrams

- Click on Company and create two new Attributes: "name: String" and "activity: String"
Eclipse UML Class Diagrams

- Create a New class named Project
- Add attribute “name:String: to class Project
- Add an association between Company and Project.
In association properties
- Label: leave blank
- 1st associations end:
  - Name: company
  - Multiplicity: 1
- 2nd associations end:
  - Name: project
  - Multiplicity: 0..1
Eclipse UML Class Diagrams
Add an Association between Employee and Project
Select Association and drag from Employee to Project
In the properties menu:
- Label: leave blank
- 1st Association End: Name: leader, Dimension 0,Multiplicity 0..1
- 2nd Association End: Name: Project, Dimension 0,Multiplicity 0..1
Eclipse UML Class Diagrams

```
Person
- age: int
- name: String

getAge()
getName()
setAge()
setName()

Company
- activity: String
- employees: Collection
- name: String
- project: Project

getActivity()
getEmployees()
getName()
getCompany()

Employee
- company: Collection
- project: Project
- salary float

getCompany()
getProject()
getSalary()
setCompany()
setProject()

Project
- company: Company
- leader: Employee
- name: String

getCompany()
getLeader()
getName()
```
Show code

- Select a Class
- From the pop-up menu select “Source editor”
The end

- Experiment!!!
- Eclipse Tutorials:
  - http://www.eclipse.org/
  - http://www.3plus4software.de/eclipse/index_en.html
  - http://www.omondo.com/