Telosys 1.1.0 Starter Kit

How to create a new **Telosys project**
in
Eclipse 3.x for Java EE
and
how to use the **Telosys Tools** plugin

Table of content

Step 1 – Setup the Eclipse environment 3
1.1 – Install the "TelosysTools plug-in" in Eclipse 3
1.2 – Select the JRE 3
1.3 – Define the Tomcat server 5

Step 2 – Create a standard Java Dynamic Web Project 7
2.1 – Create the Dynamic Web Project 7
2.2 – Test the new projet with Tomcat 9

Step 3 – Transform the project into a Telosys project 12
3.1 – Import the Telosys Starter Kit in the project 12
3.2 – Try to run the project 14
3.3 – Configure the project 16
3.4 – Install the Telosys documentation 18

Step 4 – Setup the database and the repository 19
4.1 – Install the "Bookstore" DERBY database for tests 19
4.2 – Connect to the database 20
4.3 – Generate the repository 22

Step 5 – Code generation 24
5.1 – Generate basic Java classes from the repository 24
5.2 – Test the "Data Access Layer" with the database 26
5.3 – Generate specific files from the repository 27

Step 1 – Setup the Eclipse environment

1.1 – Install the “TelosysTools plug-in” in Eclipse

Stop Eclipse

Download **telosys-tools-plugin_X.X.X.zip** from the Telosys web site
http://telosys.ow2.org/ → Download

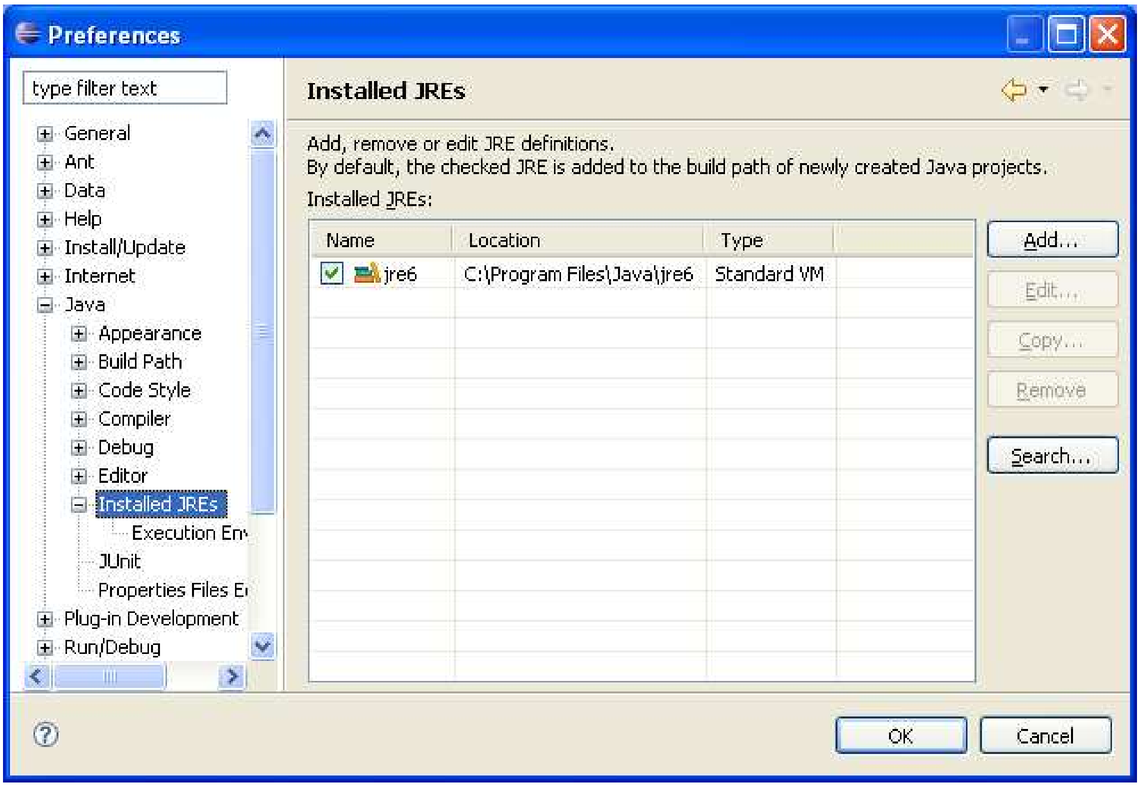
Unzip file **telosys-tools-plugin_X.X.X.zip** in the Eclipse « plugins » directory

Restart Eclipse

Check the “Plugin details” : “Help” → “About Eclipse” → “Plug-in details”
The Telosys Tools plug-in must be in the list.

1.2 – Select the JRE

Window → Preferences …



Installed JREs

Add, remove or edit JRE definitions.
By default, the checked JRE is added to the build path of newly created Java projects.

Installed JREs:

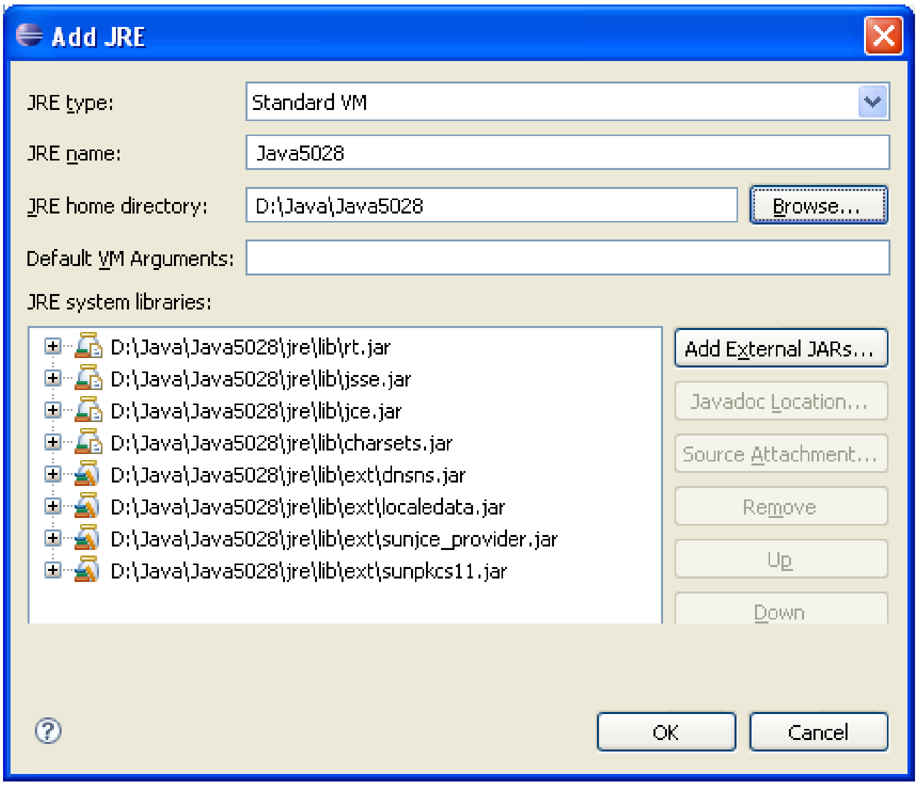
Name	Location	Type
☑ jre6	C:\Program Files\Java\jre6	Standard VM

Choose a **full Java SDK**, because a Java compiler is required to compile the JSP.

Telosys supports all the Java versions from Java 1.4

If necessary, add a JRE as follow :

Click the « Add » button and « Browse » to select a valid JRE home directory



JRE type: Standard VM

JRE name: Java5028

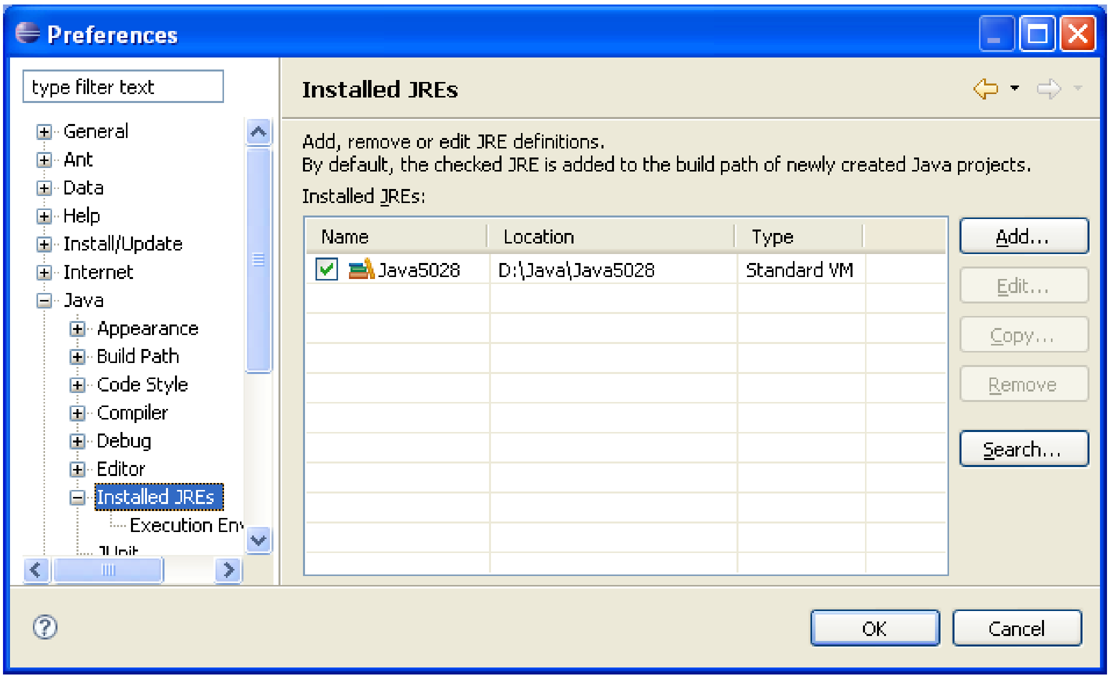
JRE home directory: D:\Java\Java5028

Default VM Arguments:

JRE system libraries:

- D:\Java\Java5028\jre\lib\rt.jar
- D:\Java\Java5028\jre\lib\jsse.jar
- D:\Java\Java5028\jre\lib\jce.jar
- D:\Java\Java5028\jre\lib\charsets.jar
- D:\Java\Java5028\jre\lib\ext\dnsns.jar
- D:\Java\Java5028\jre\lib\ext\localedata.jar
- D:\Java\Java5028\jre\lib\ext\sunjce_provider.jar
- D:\Java\Java5028\jre\lib\ext\sunpkcs11.jar

Select the new JRE added (other unused JRE can be removed)



Installed JREs

Add, remove or edit JRE definitions.
By default, the checked JRE is added to the build path of newly created Java projects.

Installed JREs:

Name	Location	Type
☑ Java5028	D:\Java\Java5028	Standard VM

Click the « OK » button

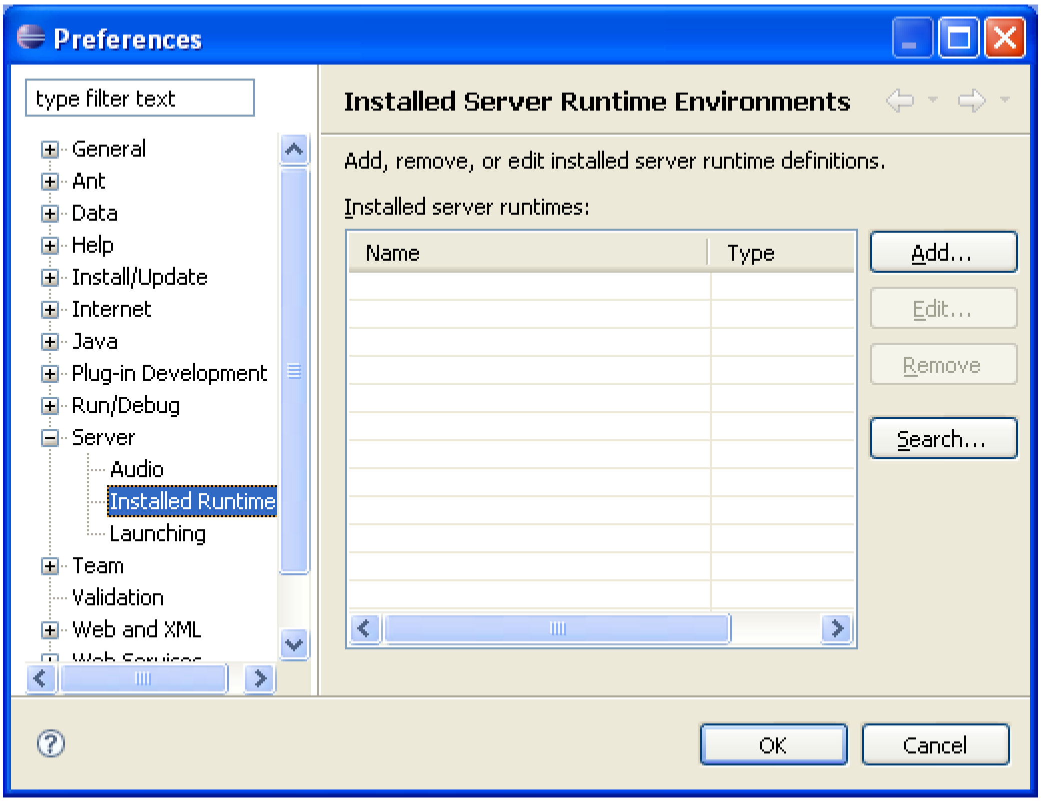
1.3 – Define the Tomcat server

This guide is based on Tomcat because it’s one of the most used servlet container.

But Telosys works fine with any standard servlet container compliant with servlet 2.4 and JSP 1.2 (and more).

Any Tomcat version can be used since Tomcat 4.

Select Window → Preferences …



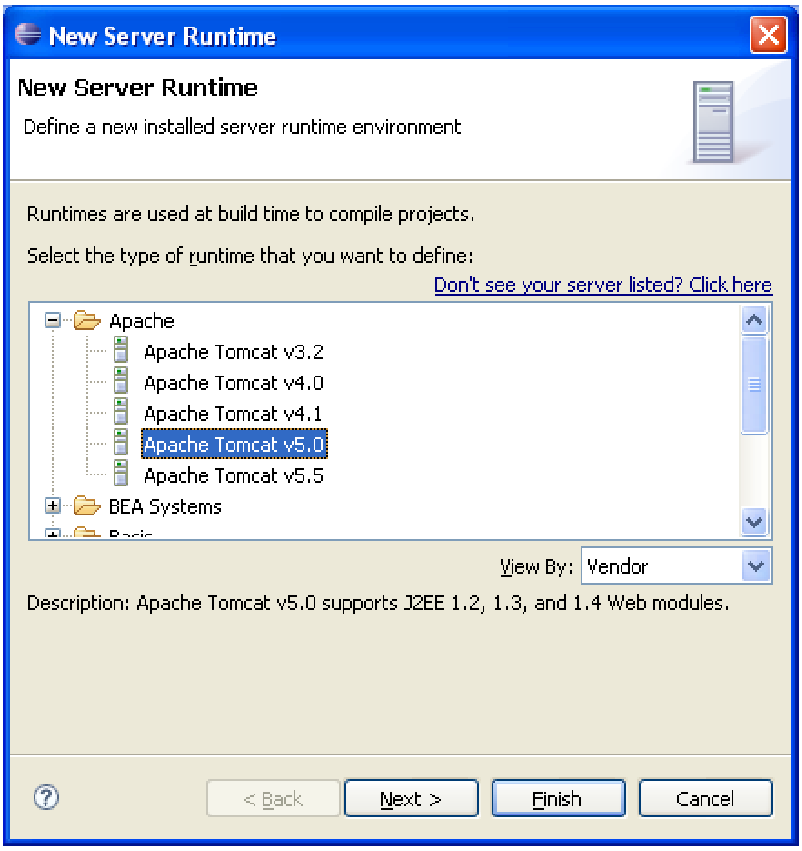
Installed Server Runtime Environments

Add, remove, or edit installed server runtime definitions.

Installed server runtimes:

Name	Type

If necessary, click the « Add » button to define a new server.


New Server Runtime

New Server Runtime

Define a new installed server runtime environment

Runtimes are used at build time to compile projects.

Select the type of runtime that you want to define:

Don't see your server listed? Click here

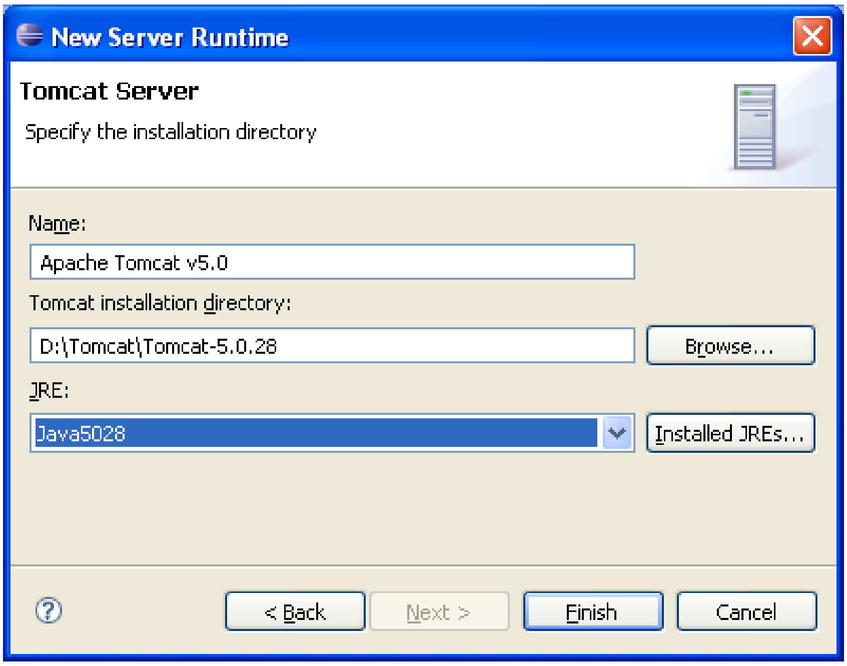
- Apache
  - Apache Tomcat v3.2
  - Apache Tomcat v4.0
  - Apache Tomcat v4.1
  - Apache Tomcat v5.0
  - Apache Tomcat v5.5
- BEA Systems
- Basic

View By: Vendor

Description: Apache Tomcat v5.0 supports J2EE 1.2, 1.3, and 1.4 Web modules.

Select « Next » button and:

- choose the Tomcat installation directory
- select the JRE


New Server Runtime

Tomcat Server

Specify the installation directory

Name: Apache Tomcat v5.0

Tomcat installation directory: D:\Tomcat\Tomcat-5.0.28

JRE: Java5028

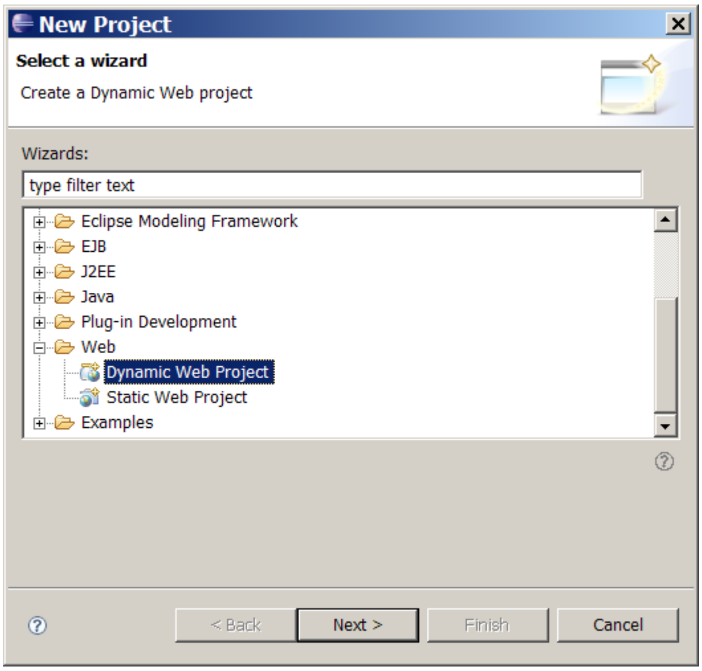
Select « Finish » button

Step 2 – Create a standard Java Dynamic Web Project

2.1 – Create the Dynamic Web Project

File → New (Alt+Shift+N) → Project …

Web → Dynamic Web Project (with Tomcat)



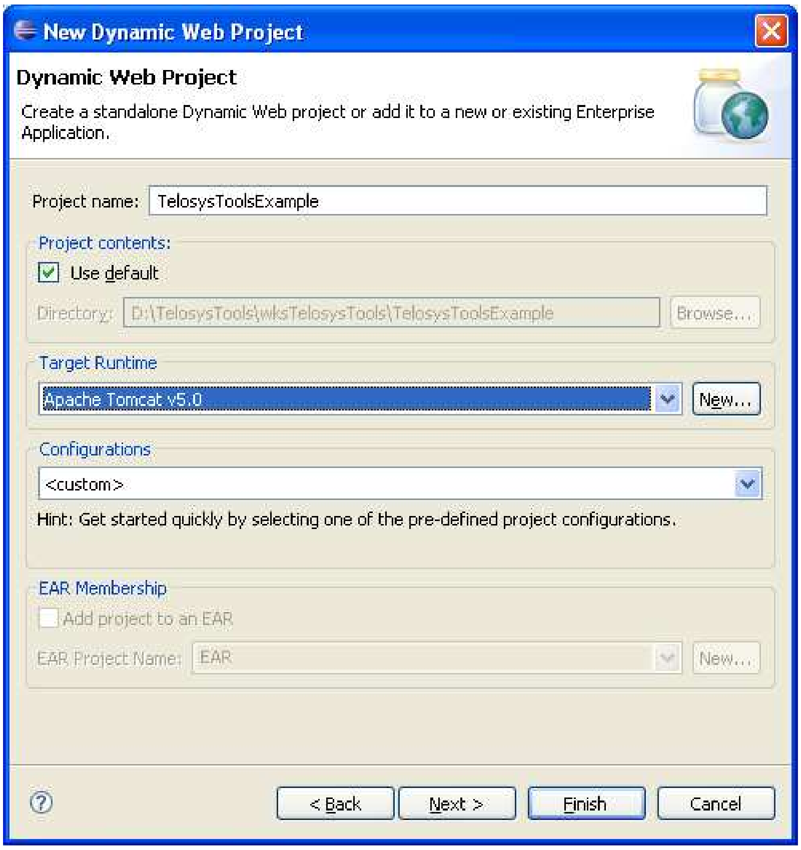
New Project

Select a wizard

Create a Dynamic Web project

Wizards: type filter text

- Eclipse Modeling Framework
- EJB
- J2EE
- Java
- Plug-in Development
- Web
  - Dynamic Web Project
  - Static Web Project
- Examples



New Dynamic Web Project

Dynamic Web Project

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: TelosysToolsExample

Project contents:

☑ Use default

Directory: D:\TelosysTools\wksTelosysTools\TelosysToolsExample

Target Runtime: Apache Tomcat v5.0

Configurations: <custom>

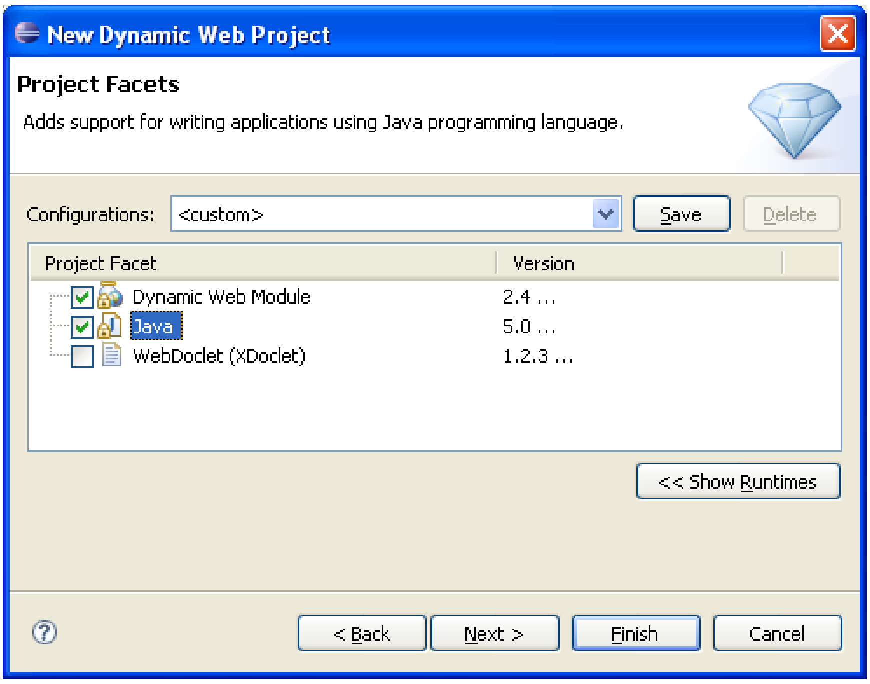
Hint: Get started quickly by selecting one of the pre-defined project configurations.

EAR Membership

□ Add project to an EAR

EAR Project Name: EAR

Select « **Next** » button



New Dynamic Web Project

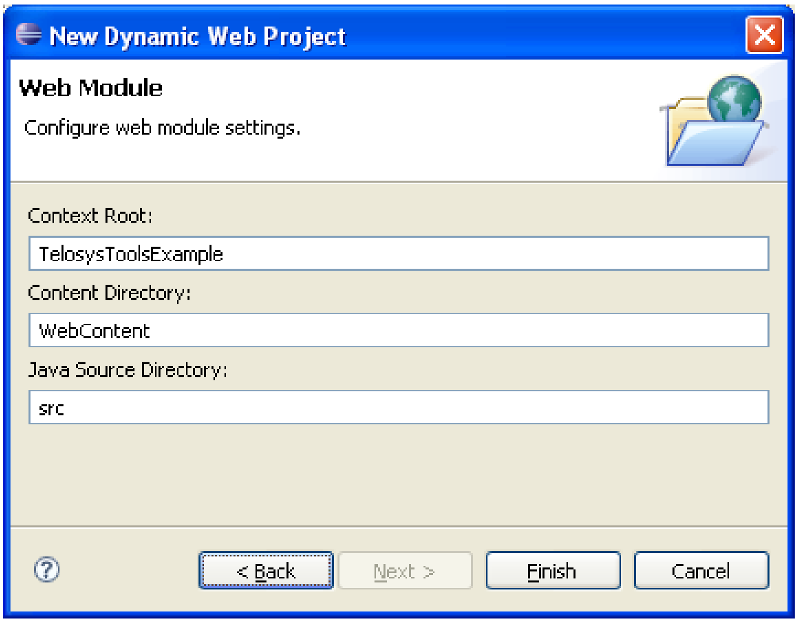
Project Facets

Adds support for writing applications using Java programming language.

Configurations: <custom>

Project Facet	Version
☑ Dynamic Web Module	2.4 ...
☑ Java	5.0 ...
□ WebDoclet (XDoclet)	1.2.3 ...

Select « **Next** » button



New Dynamic Web Project

Web Module

Configure web module settings.

Context Root: TelosysToolsExample

Content Directory: WebContent

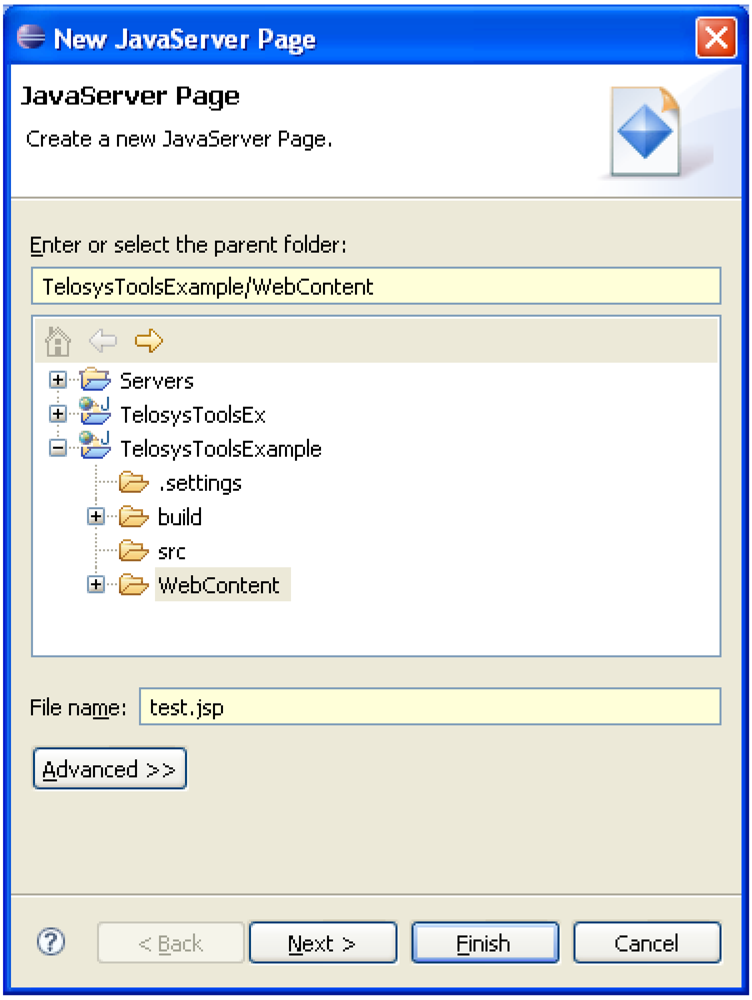
Java Source Directory: src

Select « **Finish** » button

2.2 – Test the new projet with Tomcat

Create a « **test.jsp** » file in **WebContent** directory and try to run it.

Right click on **WebContent** directory → New → JSP



New JavaServer Page

JavaServer Page

Create a new JavaServer Page.

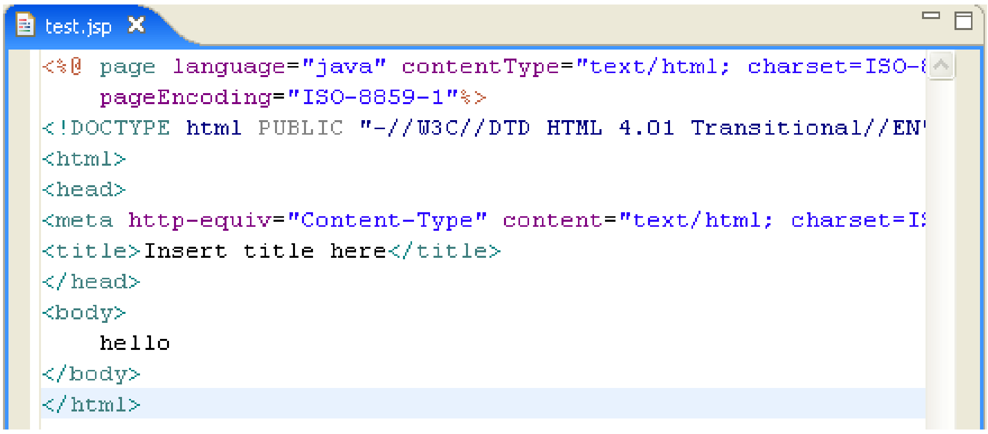
Enter or select the parent folder:

TelosysToolsExample/WebContent

File name: test.jsp

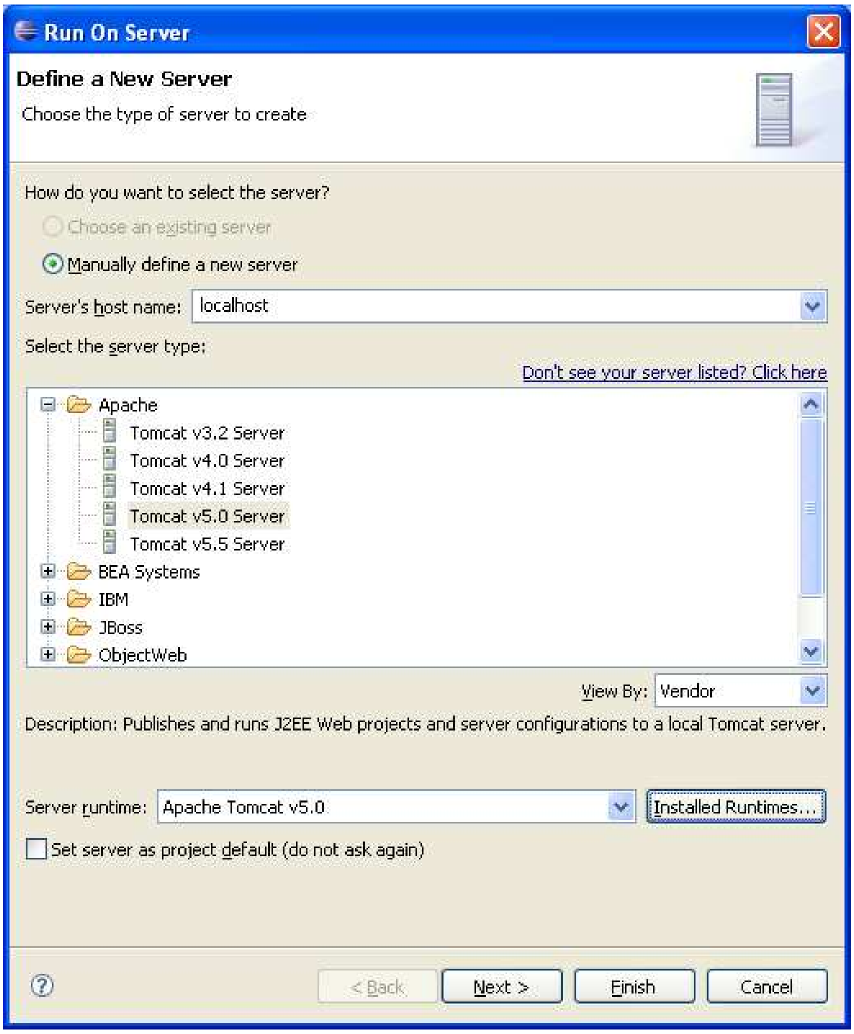
Select « Finish » button

Insert “hello” in the <body> tag of the JSP file



```
<%@ page language="java" contentType="text/html; charset=ISO-8
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN'
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=IS
<title>Insert title here</title>
</head>
<body>
    hello
</body>
</html>
```

Right click on the project directory → Run As → Run on Server



Define a New Server

Choose the type of server to create

How do you want to select the server?

○ Choose an existing server

● Manually define a new server

Server's host name: localhost

Select the server type:

View By: Vendor

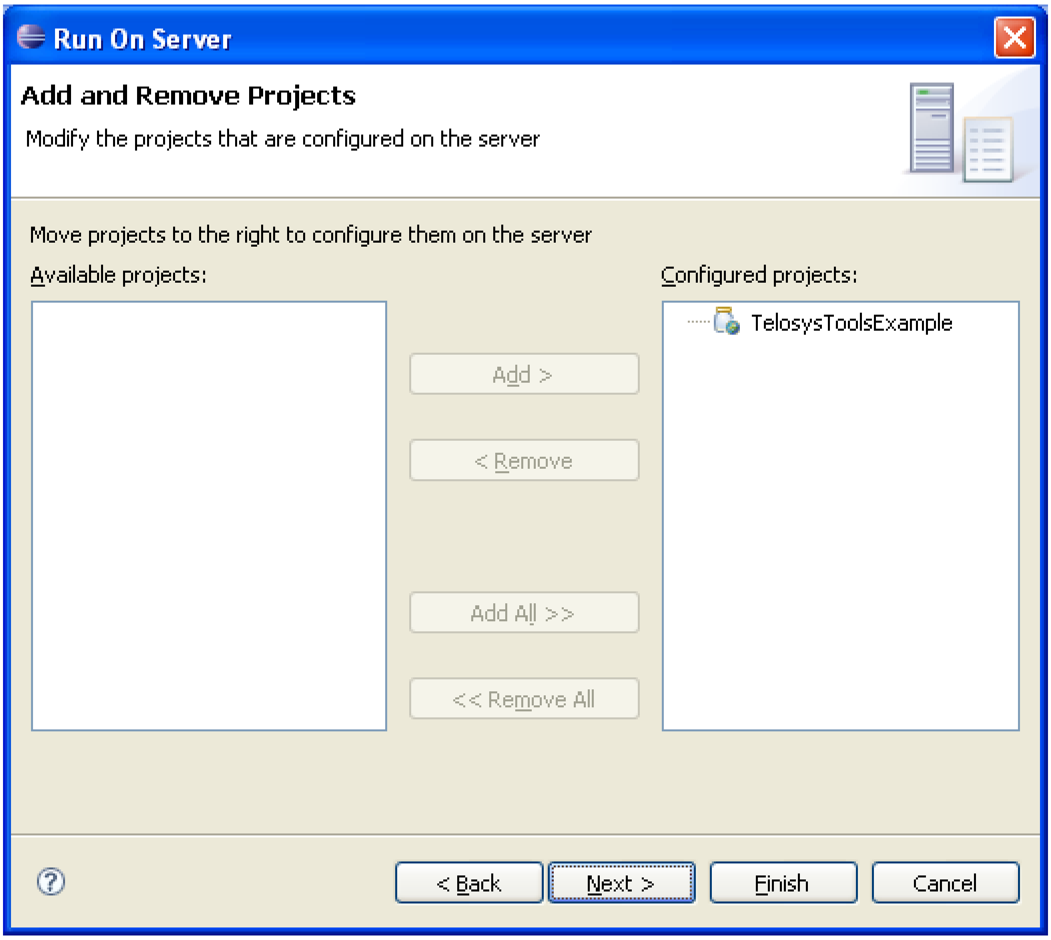
Description: Publishes and runs J2EE Web projects and server configurations to a local Tomcat server.

Server runtime: Apache Tomcat v5.0

□ Set server as project default (do not ask again)

Select « Next » button

Make sure that your project is in the « Configured projects » part



Add and Remove Projects

Modify the projects that are configured on the server

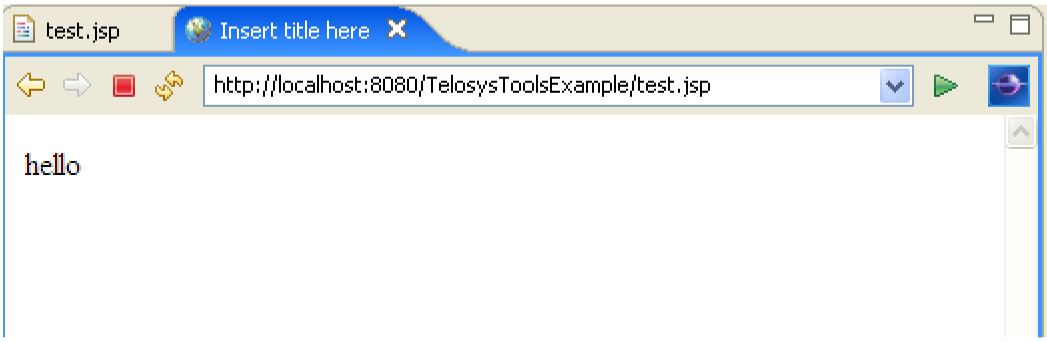
Move projects to the right to configure them on the server

Available projects:

Configured projects:
TelosysToolsExample

Select « Finish » button

Verify the good execution of your JSP file



http://localhost:8080/TelosysToolsExample/test.jsp

hello

At this step you just have a standard Java Dynamic Web Project.

This project works fine, it is ready to become a Telosys project…

Step 3 – Transform the project into a Telosys project

3.1 – Import the Telosys Starter Kit in the project

Download **telosys-x.x.x_starterkit.zip** from the Telosys web site
http://telosys.ow2.org/ → Download

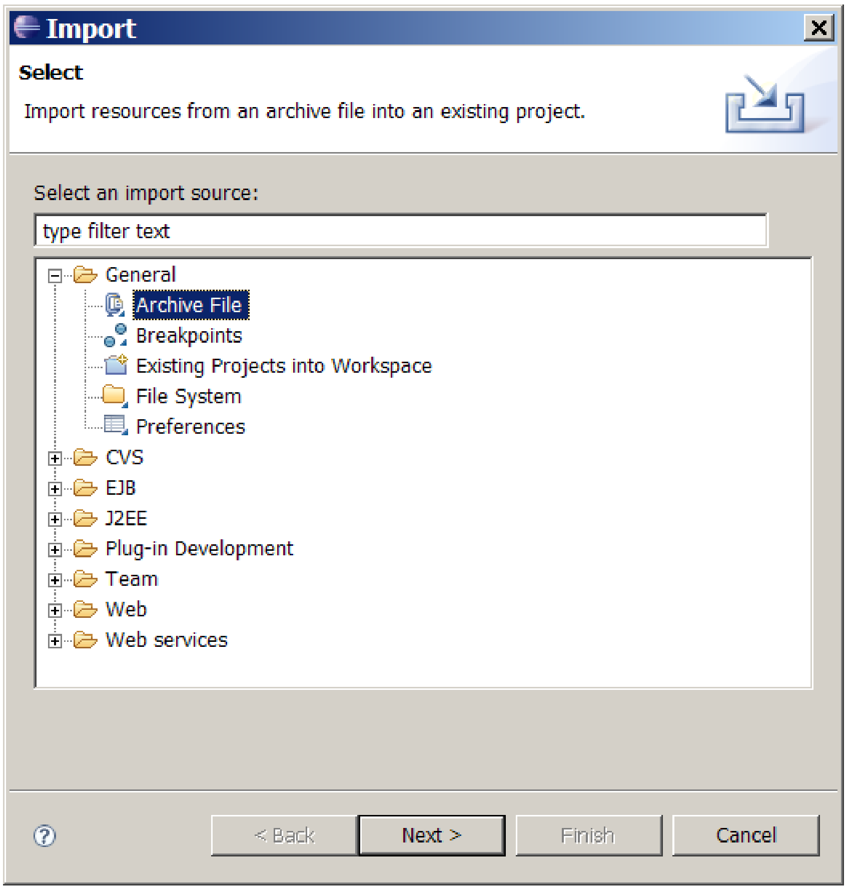
Do not unzip it.

Stop the Tomcat server if it's still running.

Import the starter kit (the "zip" file) in the project :

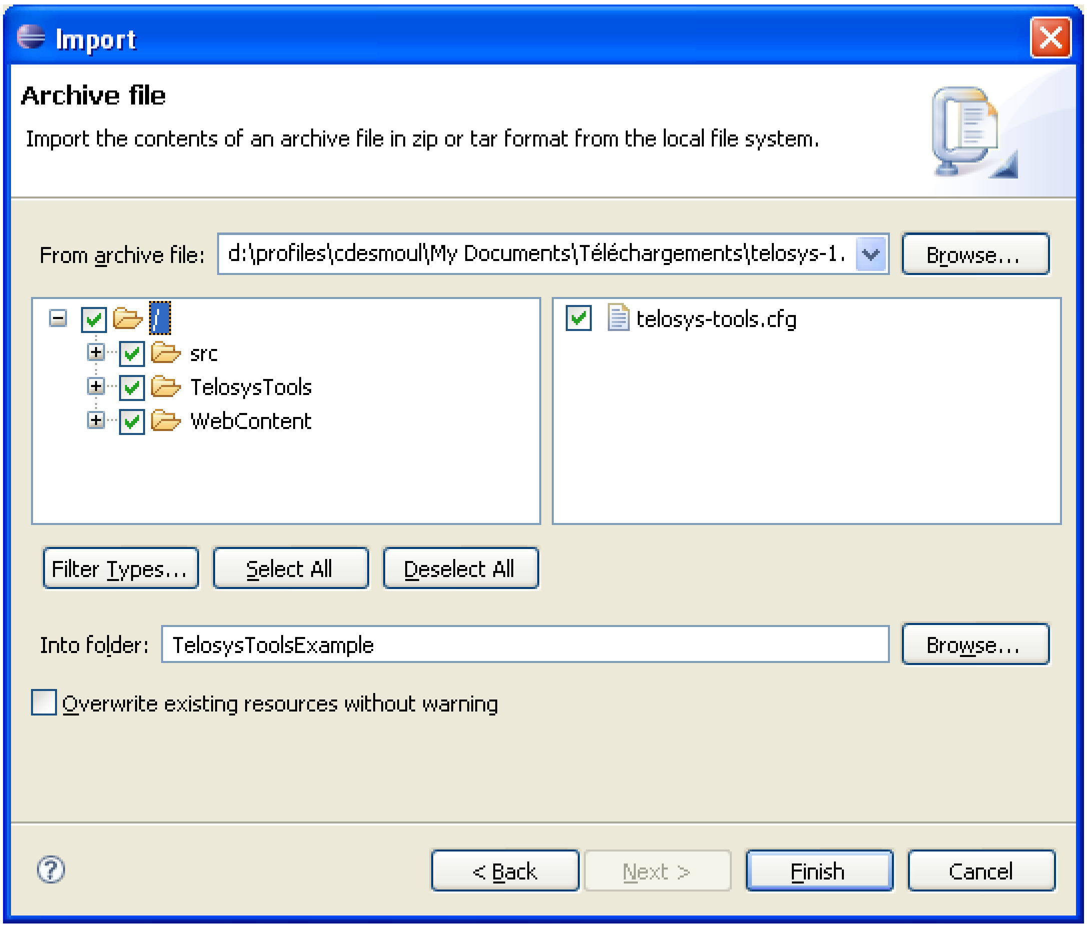
Right click on your project → File → Import → Import …

Select General → Archive File



Select « **Next** » button

Browse the file system to select the file **telosys-x.x.x-starterkit.zip …**



Import

Archive file

Import the contents of an archive file in zip or tar format from the local file system.

From archive file: d:\profiles\cdesmoul\My Documents\Téléchargements\telosys-1.

☑ /
- ☑ src
- ☑ TelosysTools
- ☑ WebContent

☑ telosys-tools.cfg

Into folder: TelosysToolsExample

□ Overwrite existing resources without warning

Click the « **Finish** » button

Overwrite **web.xml** in folder **xxx/WebContent/WEB-INF** ?

→ Answer "**Yes**"

Refresh the project (F5)

After the import there's

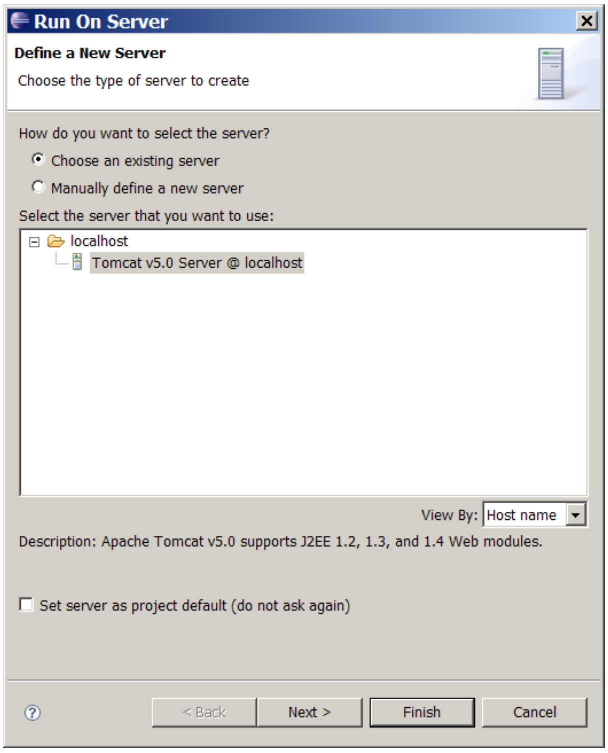
- few Java classes in "src"
- few web resources in "WebContent" (.jsp, .js, …)
- 2 jar files in WEB-INF/lib :
  - derbyclient.jar : the JDBC driver for tests with a "Derby" database
  - telosys-x.x.x.jar : the Telosys framework

That's all !

Your project is now a "Telosys project".

3.2 – Try to run the project

Right click on your project → Run As → Run on server



Run On Server

Define a New Server

Choose the type of server to create

How do you want to select the server?

● Choose an existing server

○ Manually define a new server

Select the server that you want to use:

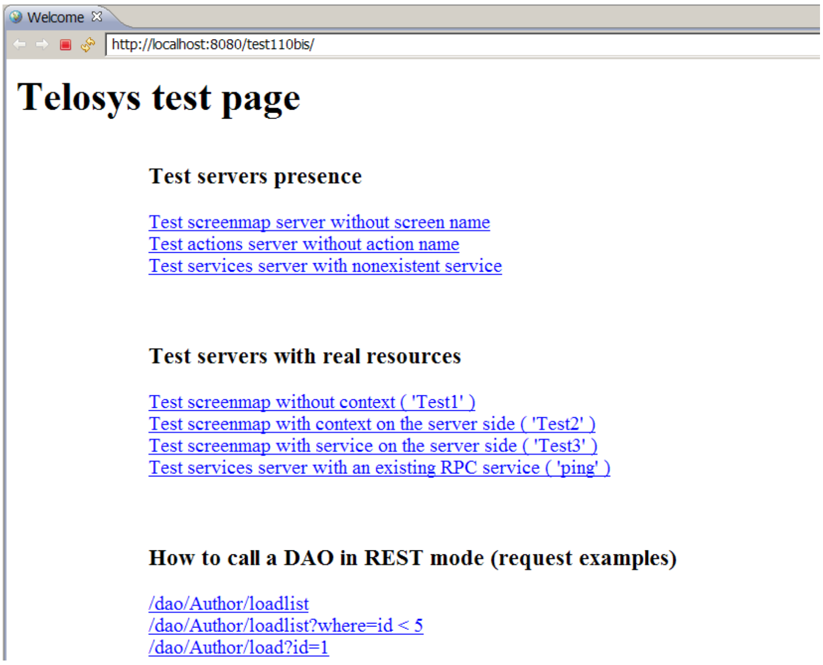
- localhost
  - Tomcat v5.0 Server @ localhost

View By: Host name

Description: Apache Tomcat v5.0 supports J2EE 1.2, 1.3, and 1.4 Web modules.

□ Set server as project default (do not ask again)

The welcome page appears in the Eclipse web browser :



http://localhost:8080/test110bis/

Telosys test page

Test servers presence

Test screenmap server without screen name
Test actions server without action name
Test services server with nonexistent service

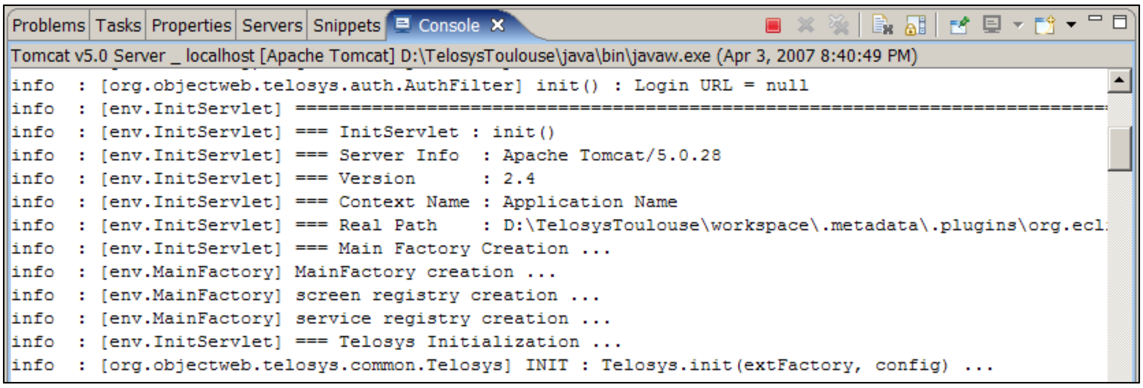
Test servers with real resources

Test screenmap without context ('Test1')
Test screenmap with context on the server side ('Test2')
Test screenmap with service on the server side ('Test3')
Test services server with an existing RPC service ('ping')

How to call a DAO in REST mode (request examples)

/dao/Author/loadlist
/dao/Author/loadlist?where=id < 5
/dao/Author/load?id=1

Check there's no error in the console.



Tomcat v5.0 Server _ localhost [Apache Tomcat] D:\TelosysToulouse\java\bin\javaw.exe (Apr 3, 2007 8:40:49 PM)

```
info  : [org.objectweb.telosys.auth.AuthFilter] init() : Login URL = null
info  : [env.InitServlet] ===========================================================================
info  : [env.InitServlet] === InitServlet : init()
info  : [env.InitServlet] === Server Info  : Apache Tomcat/5.0.28
info  : [env.InitServlet] === Version      : 2.4
info  : [env.InitServlet] === Context Name : Application Name
info  : [env.InitServlet] === Real Path    : D:\TelosysToulouse\workspace\.metadata\.plugins\org.ecl
info  : [env.InitServlet] === Main Factory Creation ...
info  : [env.MainFactory] MainFactory creation ...
info  : [env.MainFactory] screen registry creation ...
info  : [env.MainFactory] service registry creation ...
info  : [env.InitServlet] === Telosys Initialization ...
info  : [org.objectweb.telosys.common.Telosys] INIT : Telosys.init(extFactory, config) ...
```

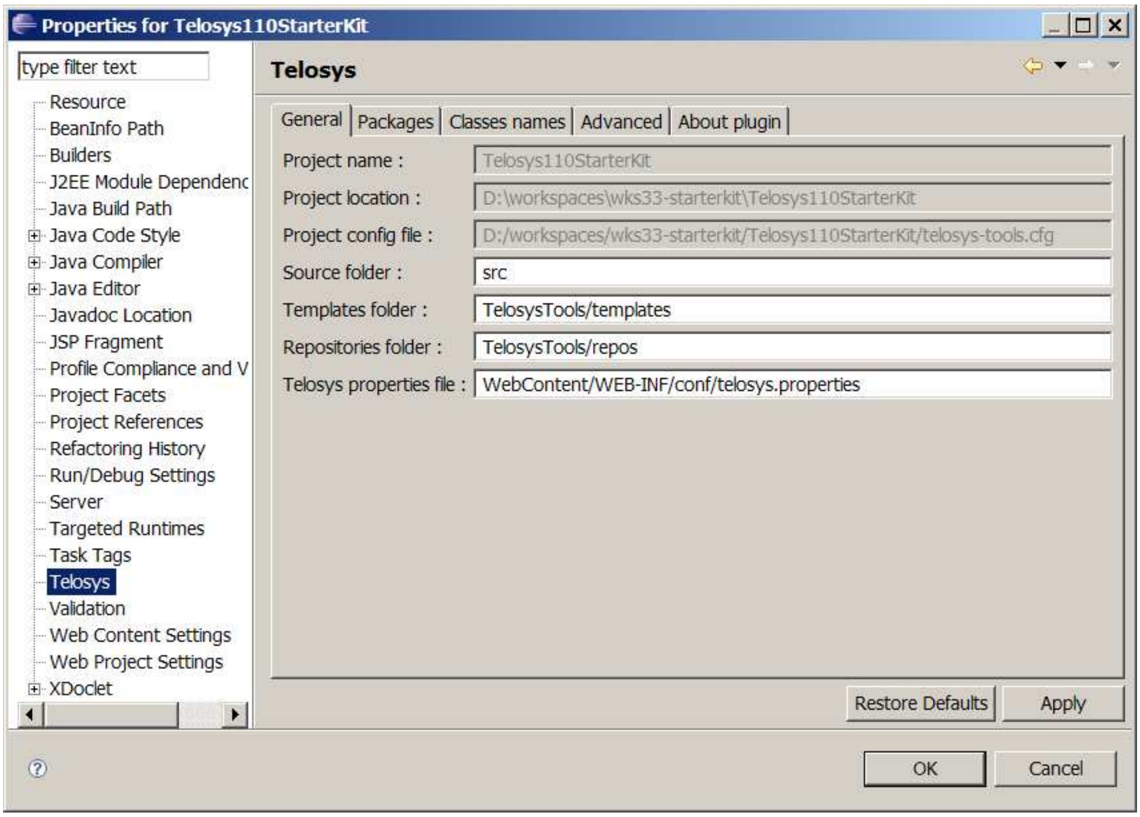
Stop Tomcat Server

3.3 – Configure the project

The Telosys Tools plugin can be used to customize the Telosys project’s properties.

Let’s have a look at these properties …
Select your project + right click → “Properties” → Telosys

The “**General**” tab :



Project name : Telosys110StarterKit
Project location : D:\workspaces\wks33-starterkit\Telosys110StarterKit
Project config file : D:/workspaces/wks33-starterkit/Telosys110StarterKit/telosys-tools.cfg
Source folder : src
Templates folder : TelosysTools/templates
Repositories folder : TelosysTools/repos
Telosys properties file : WebContent/WEB-INF/conf/telosys.properties

“Source folder” : the folder containing the Java sources

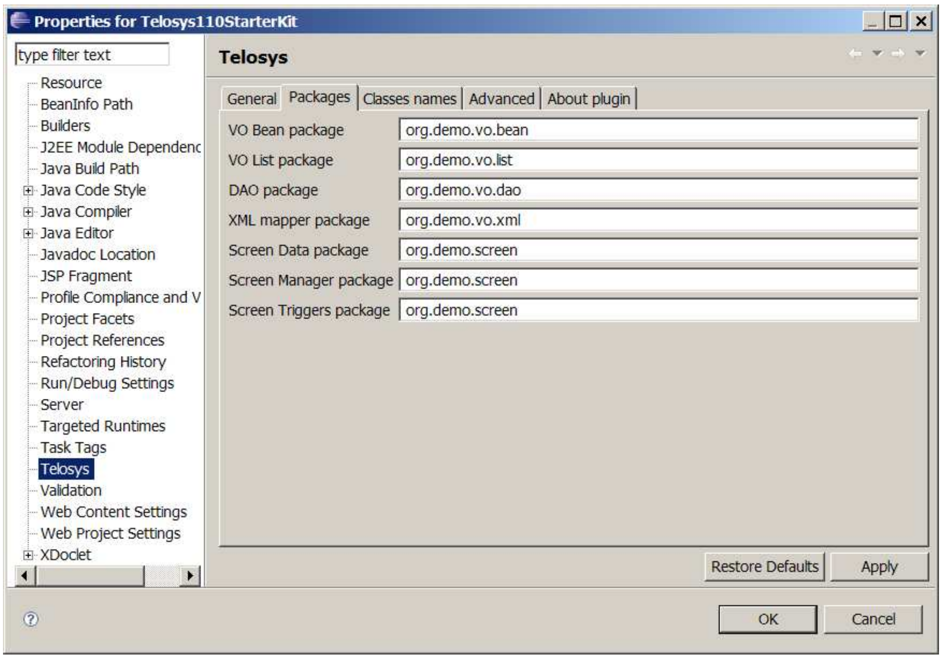
“Templates folder” : the folder containing the Velocity templates used for code generation

“Repositories folder” : the folder containing the repositories (the models used as input for code generation)

“Telosys properties file” : the main Telosys configuration file

→ At this step, you don’t have to change anything

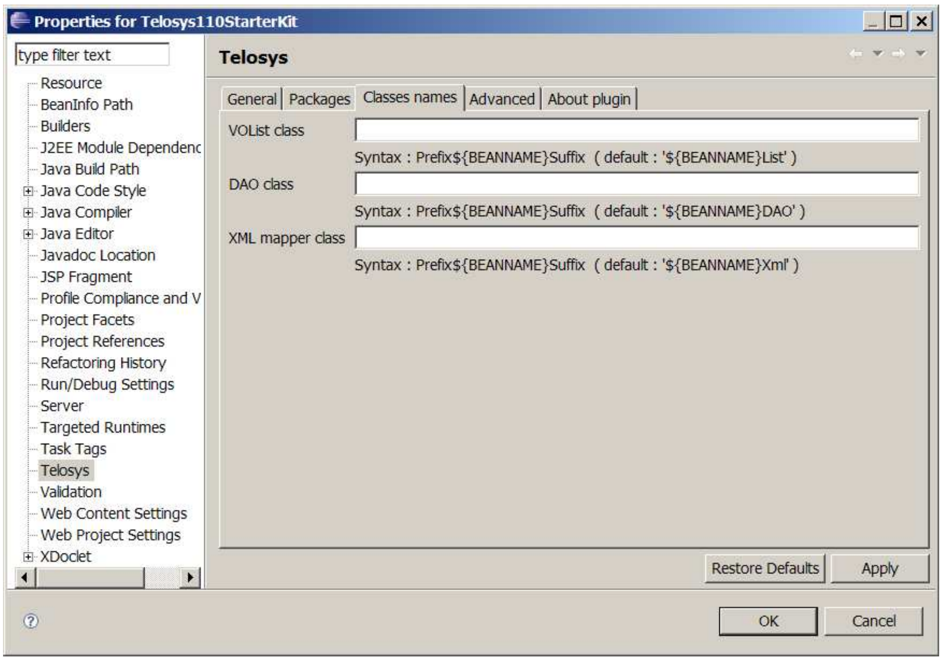
The “**Packages**” tab :



VO Bean package: org.demo.vo.bean
VO List package: org.demo.vo.list
DAO package: org.demo.vo.dao
XML mapper package: org.demo.vo.xml
Screen Data package: org.demo.screen
Screen Manager package: org.demo.screen
Screen Triggers package: org.demo.screen

This tab allows to set the packages for each type of classes.
They must be defined according with the packages organization of your project.

The “**Classes names**” tab :



VOList class:
Syntax : Prefix${BEANNAME}Suffix (default : '${BEANNAME}List')

DAO class:
Syntax : Prefix${BEANNAME}Suffix (default : '${BEANNAME}DAO')

XML mapper class:
Syntax : Prefix${BEANNAME}Suffix (default : '${BEANNAME}Xml')

It’s possible to define classes naming conventions here, by adding a prefix or a suffix at the “entity bean name”.

→ For the moment, do not change anything in the project’s properties.

3.4 – Install the Telosys documentation

Download the documentation files from the Telosys web site

- **telosys-x.x.x_javadoc.zip** (Java classes documentation)
- **telosys-x.x.x_jsdoc.zip** (Javascript functions and classes documentation)
- **telosys-x.x.x_taglibdoc.zip** (TagLib documentation)

http://telosys.ow2.org/ → Download

Unzip these files in the folder of your choice
You obtain one folder for each file :

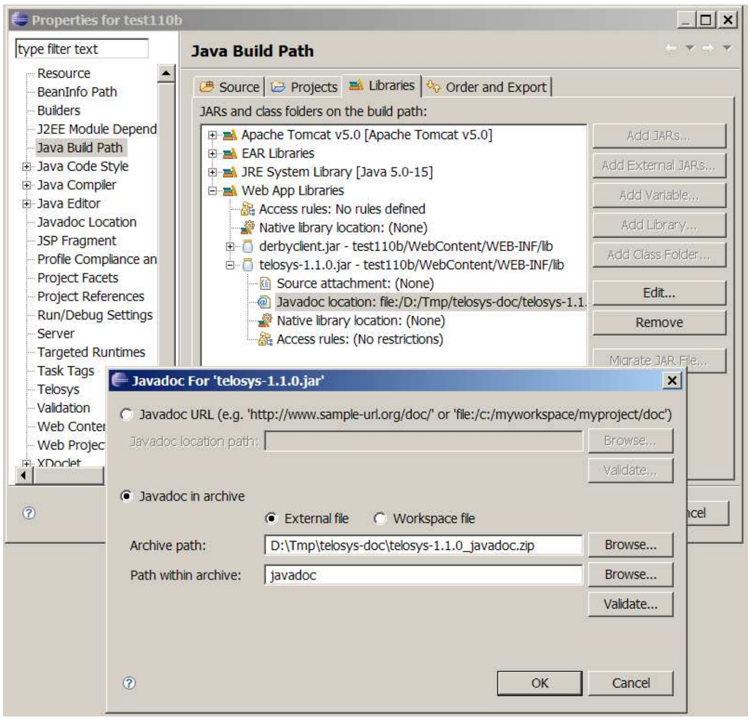
- javadoc
- jsdoc
- taglibdoc

Open the "index.html" of each folder to browse the documentation

Attach the Javadoc to the "telosys-x.x.x.jar" file :

Right-click on your project, select "**Properties**" → "**Java Build Path**"
In the "**Libraries**" tab :

- open "**Web App Libraries**"
- open "**telosys-x.x.x.jar**"
- select "**Javadoc location**"
- click "**Edit**"
  - select **telosys-x.x.x_javadoc.zip** as "External file"
  - set "**javadoc**" as "Path within archive"



Properties for test110b

Java Build Path

JARs and class folders on the build path:

- Apache Tomcat v5.0 [Apache Tomcat v5.0]
- EAR Libraries
- JRE System Library [Java 5.0-15]
- Web App Libraries
  - Access rules: No rules defined
  - Native library location: (None)
  - derbyclient.jar - test110b/WebContent/WEB-INF/lib
  - telosys-1.1.0.jar - test110b/WebContent/WEB-INF/lib
    - Source attachment: (None)
    - Javadoc location: file:/D:/Tmp/telosys-doc/telosys-1.1.
    - Native library location: (None)
    - Access rules: (No restrictions)

Javadoc For 'telosys-1.1.0.jar'

○ Javadoc URL (e.g. 'http://www.sample-url.org/doc/' or 'file:/c:/myworkspace/myproject/doc')

Javadoc location path:

● Javadoc in archive

● External file ○ Workspace file

Archive path: D:\Tmp\telosys-doc\telosys-1.1.0_javadoc.zip

Path within archive: javadoc

Step 4 – Setup the database and the repository

4.1 – Install the "Bookstore" DERBY database for tests

A lightweight database is provided on the Telosys web site for demonstrations, it is very easy to setup …

Download **bookstoreX.X_derby_X.X.X.zip** from the Telosys web site
http://telosys.ow2.org/ → Download

Unzip the file in a folder (this folder will be the DERBY_HOME folder)

Start DERBY by running the command file

DERBY_HOME/start.bat
or
DERBY_HOME/bin/startNetworkServer.bat (Windows)
DERBY_HOME/bin/startNetworkServer (shell Unix/Linux)

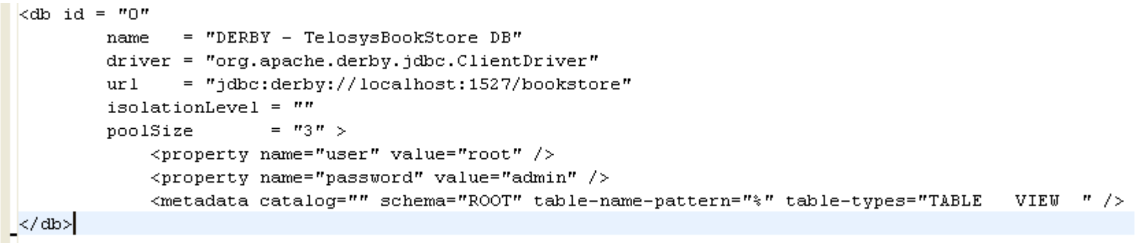
The database engine is running in a command window, and is now ready to accept connections.

To stop the Derby database : just close this window.

4.2 – Connect to the database

Edit the “**databases.dbcfg**” file with a “text editor”

This is an XML file.
Just uncomment the DERBY entry as follow :

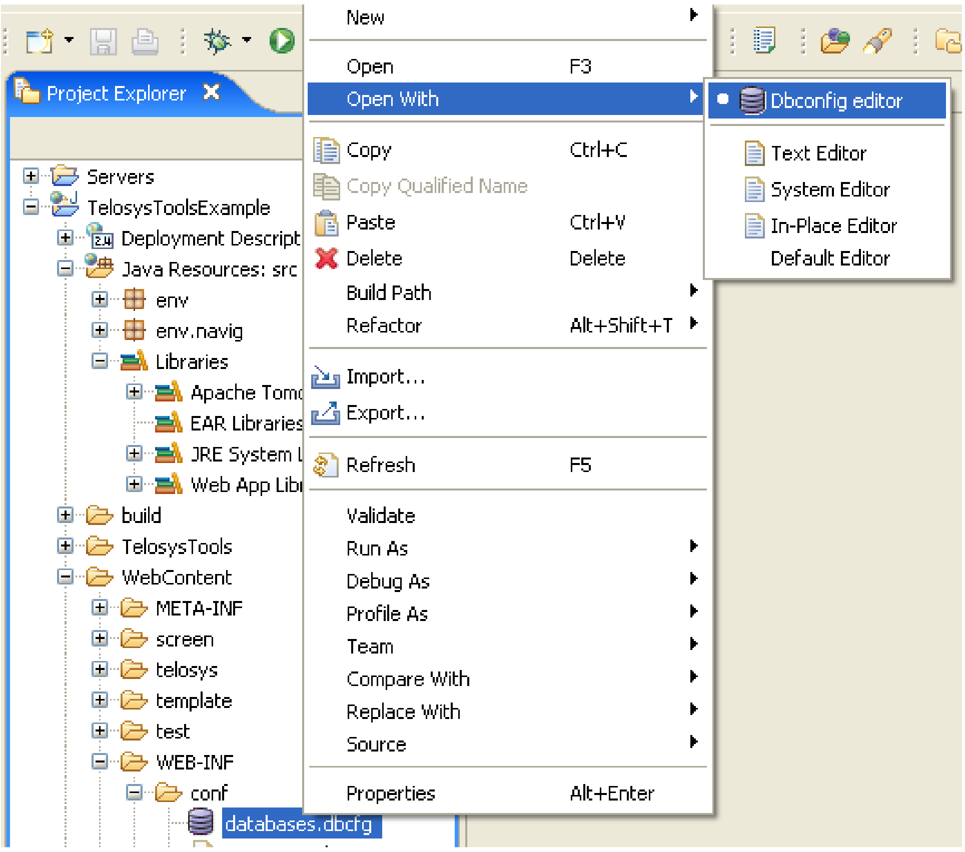


```
<db id = "0"
        name   = "DERBY - TelosysBookStore DB"
        driver = "org.apache.derby.jdbc.ClientDriver"
        url    = "jdbc:derby://localhost:1527/bookstore"
        isolationLevel = ""
        poolSize        = "3" >
            <property name="user" value="root" />
            <property name="password" value="admin" />
            <metadata catalog="" schema="ROOT" table-name-pattern="%" table-types="TABLE   VIEW  " />
</db>
```

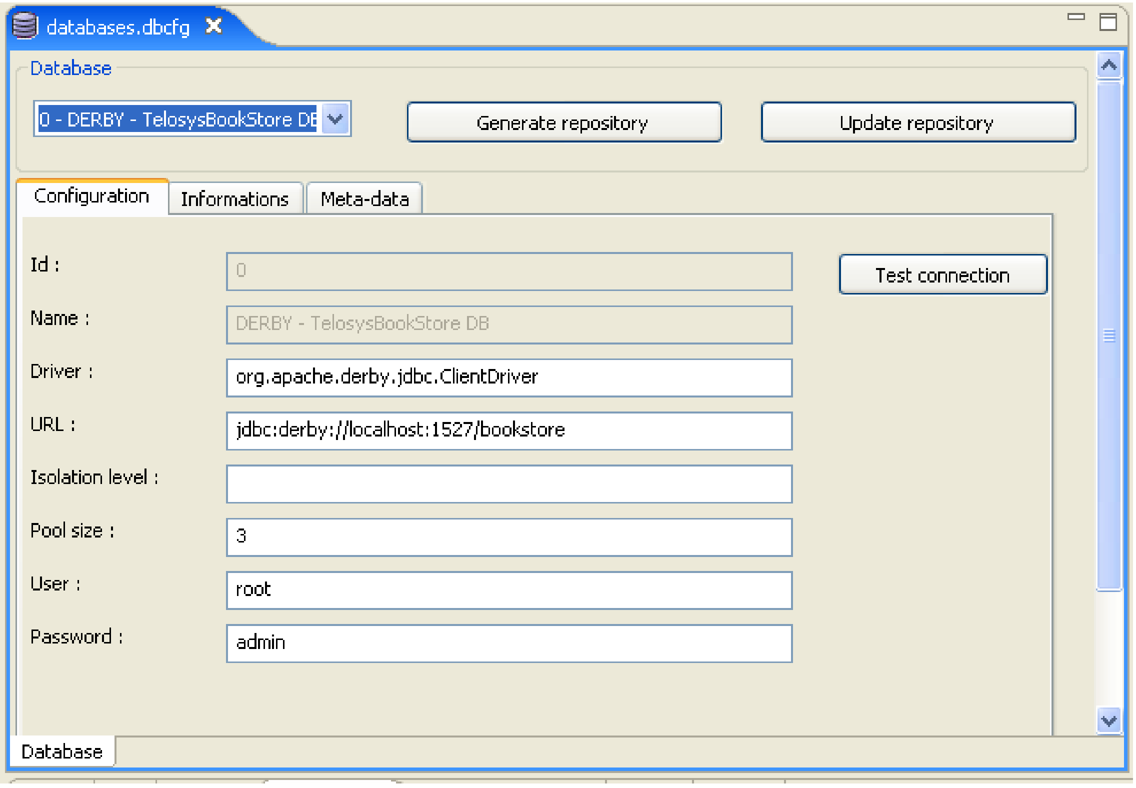
The “id” of the database will be “0”, a connection pool will be created, and the JDBC driver is in the derby “.jar” file imported with the starter-kit.

Save and close **databases.dbcfg** file.

Now, open the **databases.dbcfg** file with the “**Dbconfig editor**”



If the “database.dbcfg” has been correctly modified,
the derby database appears in the combo-box.



Database: 0 - DERBY - TelosysBookStore DE

Id : 0

Name : DERBY - TelosysBookStore DB

Driver : org.apache.derby.jdbc.ClientDriver

URL : jdbc:derby://localhost:1527/bookstore

Isolation level :

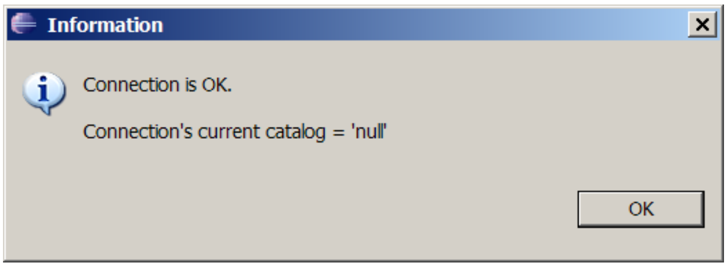
Pool size : 3

User : root

Password : admin

Select the “Derby Bookstore” database

Click the « **Test connection** » button to test connection to database …

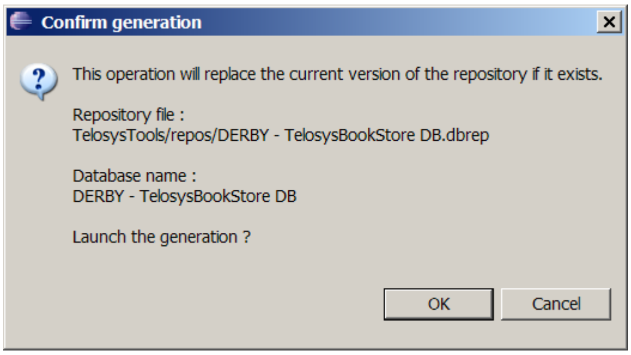


Connection is OK.

Connection's current catalog = 'null'

4.3 – Generate the repository

Once the connection has been verified, just click the « **Generate repository** » button to launch the generation of a new repository.



Confirm generation

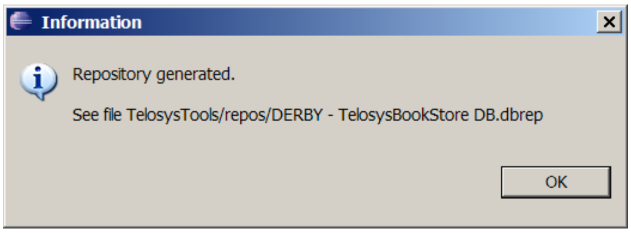
This operation will replace the current version of the repository if it exists.

Repository file :
TelosysTools/repos/DERBY - TelosysBookStore DB.dbrep

Database name :
DERBY - TelosysBookStore DB

Launch the generation ?

Click the "**OK**" button.

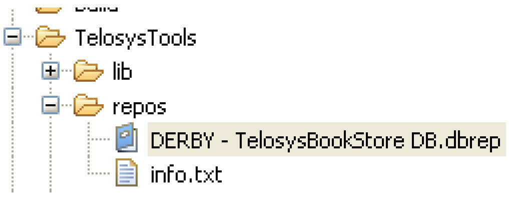


Information

Repository generated.

See file TelosysTools/repos/DERBY - TelosysBookStore DB.dbrep

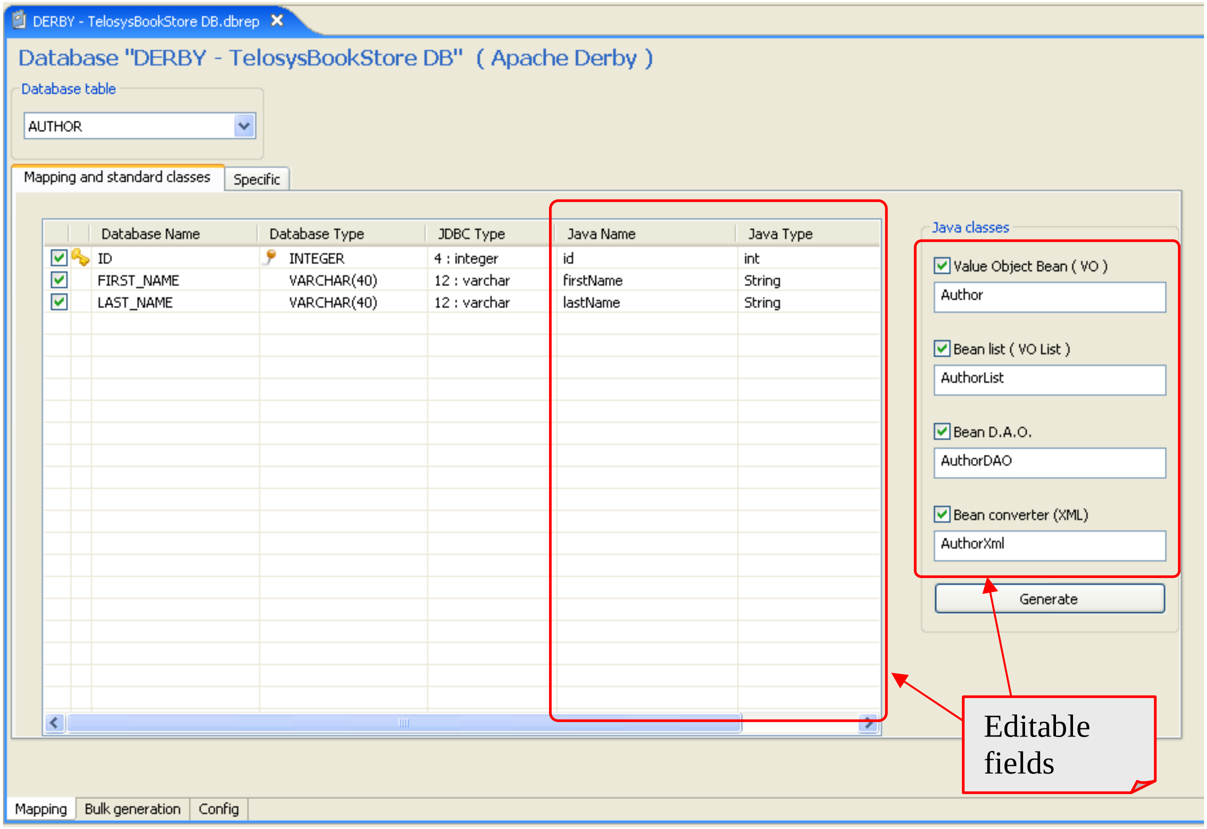
The repository file (**.dbrep**) is generated in the "**TelosysTools/repos**" directory (as defined in the project's properties)



In case of changes in the database tables, an existing repository can be updated with the "**Update repository**" button.

Double-click on the repository file (.dbrep) to open it and to view/edit the database entities

The repository editor looks like this :



DERBY - TelosysBookStore DB.dbrep
Mapping and standard classes
Specific
Generate
Mapping
Bulk generation
Config
Editable fields

Database "DERBY - TelosysBookStore DB" (Apache Derby)

Database table: AUTHOR

	Database Name	Database Type	JDBC Type	Java Name	Java Type
☑	ID	INTEGER	4 : integer	id	int
☑	FIRST_NAME	VARCHAR(40)	12 : varchar	firstName	String
☑	LAST_NAME	VARCHAR(40)	12 : varchar	lastName	String

Java classes

☑ Value Object Bean (VO): Author

☑ Bean list (VO List): AuthorList

☑ Bean D.A.O.: AuthorDAO

☑ Bean converter (XML): AuthorXml

You can modify :

- each attribute Java name and Java type
- the Java classes names to be generated

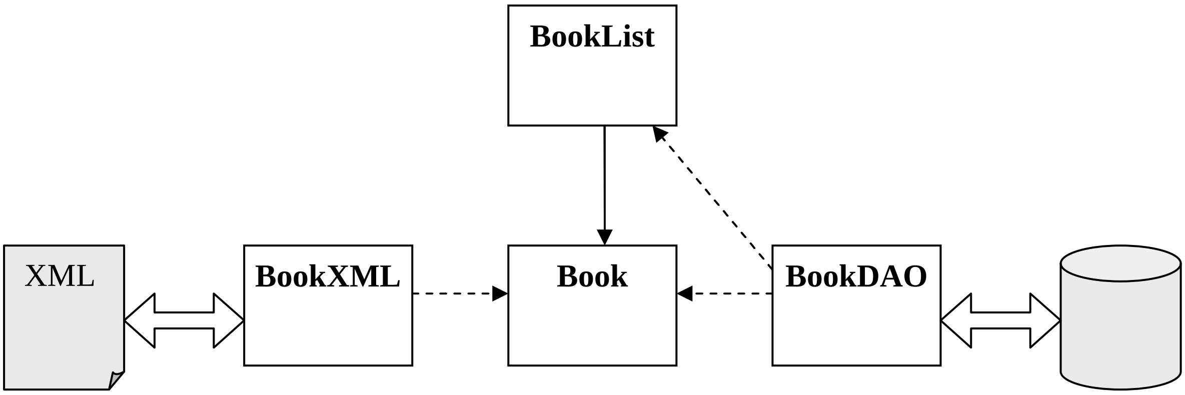
Step 5 – Code generation

5.1 – Generate basic Java classes from the repository

For each table/entity you can generate

- a “**Bean**” (simple “Value Object” / “POJO”)
- a “**Bean List**” (ordered and sortable list of Beans)
- a “**Bean DAO**” (Data Access Object) to manage the database persistence
- a “**Bean converter**” to manage the XML conversion

Example for “Book” entity :



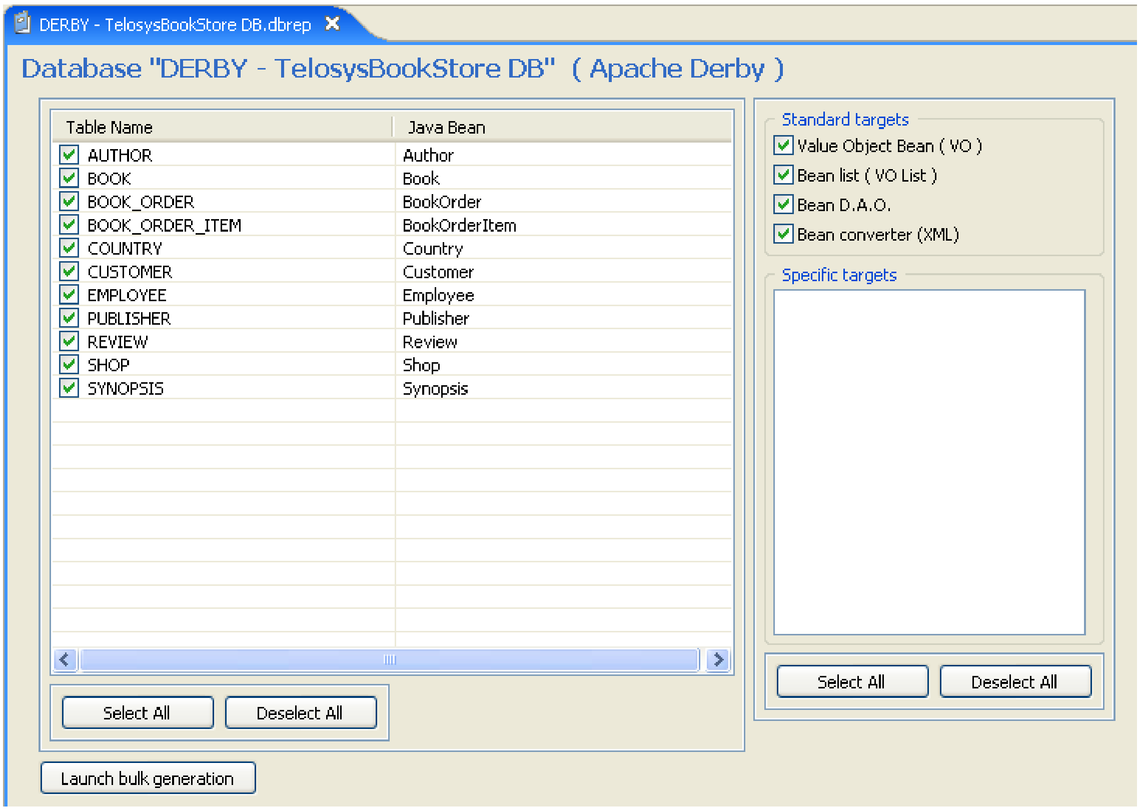
→ Generate Java classes for one entity :

Select the entity in the combo-box
Select the Java classes to generate
Click the « **Generate** » button.

The Java files are generated in the packages defined in the project properties.
Select the "**Config**" tab (at the bottom of the window) to see the current configuration

→ Generate Java classes for all the entities :

Select the « **Bulk generation** » tab (at the bottom of the window) :



DERBY - TelosysBookStore DB.dbrep

Database "DERBY - TelosysBookStore DB" (Apache Derby)

Table Name	Java Bean
☑ AUTHOR	Author
☑ BOOK	Book
☑ BOOK_ORDER	BookOrder
☑ BOOK_ORDER_ITEM	BookOrderItem
☑ COUNTRY	Country
☑ CUSTOMER	Customer
☑ EMPLOYEE	Employee
☑ PUBLISHER	Publisher
☑ REVIEW	Review
☑ SHOP	Shop
☑ SYNOPSIS	Synopsis

Standard targets
☑ Value Object Bean (VO)
☑ Bean list (VO List)
☑ Bean D.A.O.
☑ Bean converter (XML)

Specific targets

Select all the Table/Java Bean : Click the « **Select All** » buttons
Select all the "Standard targets"
Click the « **Launch bulk generation** » button

→ 4 Java classes are generated for each entity.

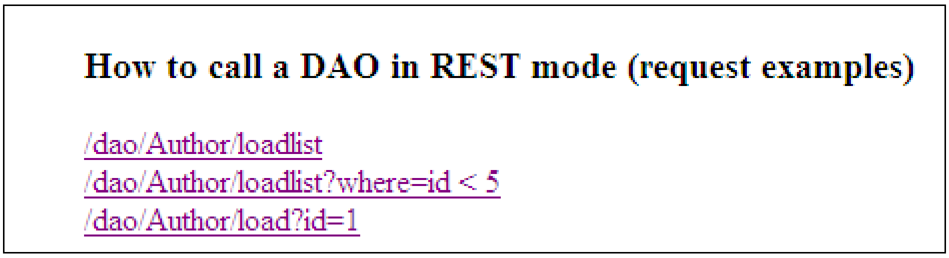
The "Data Access Layer" of the project is now ready to use for all the entities of the database.

5.2 – Test the "Data Access Layer" with the database

Ensure the database is running.

Run the web project :
Right click on the project → Run As → Run on server

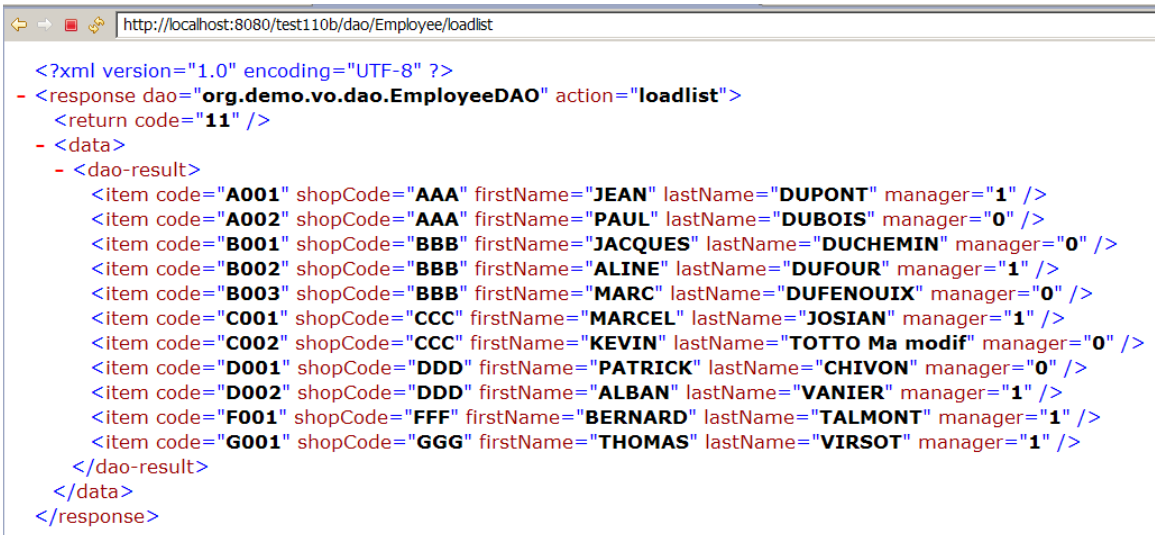
By default each DAO can be called in REST mode (via an URL) :



How to call a DAO in REST mode (request examples)

/dao/Author/loadlist
/dao/Author/loadlist?where=id < 5
/dao/Author/load?id=1

Select one of the links to see the result



```
  <?xml version="1.0" encoding="UTF-8" ?>
- <response dao="org.demo.vo.dao.EmployeeDAO" action="loadlist">
    <return code="11" />
  - <data>
    - <dao-result>
        <item code="A001" shopCode="AAA" firstName="JEAN" lastName="DUPONT" manager="1" />
        <item code="A002" shopCode="AAA" firstName="PAUL" lastName="DUBOIS" manager="0" />
        <item code="B001" shopCode="BBB" firstName="JACQUES" lastName="DUCHEMIN" manager="0" />
        <item code="B002" shopCode="BBB" firstName="ALINE" lastName="DUFOUR" manager="1" />
        <item code="B003" shopCode="BBB" firstName="MARC" lastName="DUFENOUIX" manager="0" />
        <item code="C001" shopCode="CCC" firstName="MARCEL" lastName="JOSIAN" manager="1" />
        <item code="C002" shopCode="CCC" firstName="KEVIN" lastName="TOTTO Ma modif" manager="0" />
        <item code="D001" shopCode="DDD" firstName="PATRICK" lastName="CHIVON" manager="0" />
        <item code="D002" shopCode="DDD" firstName="ALBAN" lastName="VANIER" manager="1" />
        <item code="F001" shopCode="FFF" firstName="BERNARD" lastName="TALMONT" manager="1" />
        <item code="G001" shopCode="GGG" firstName="THOMAS" lastName="VIRSOT" manager="1" />
      </dao-result>
    </data>
  </response>
```

5.3 – Generate specific files from the repository

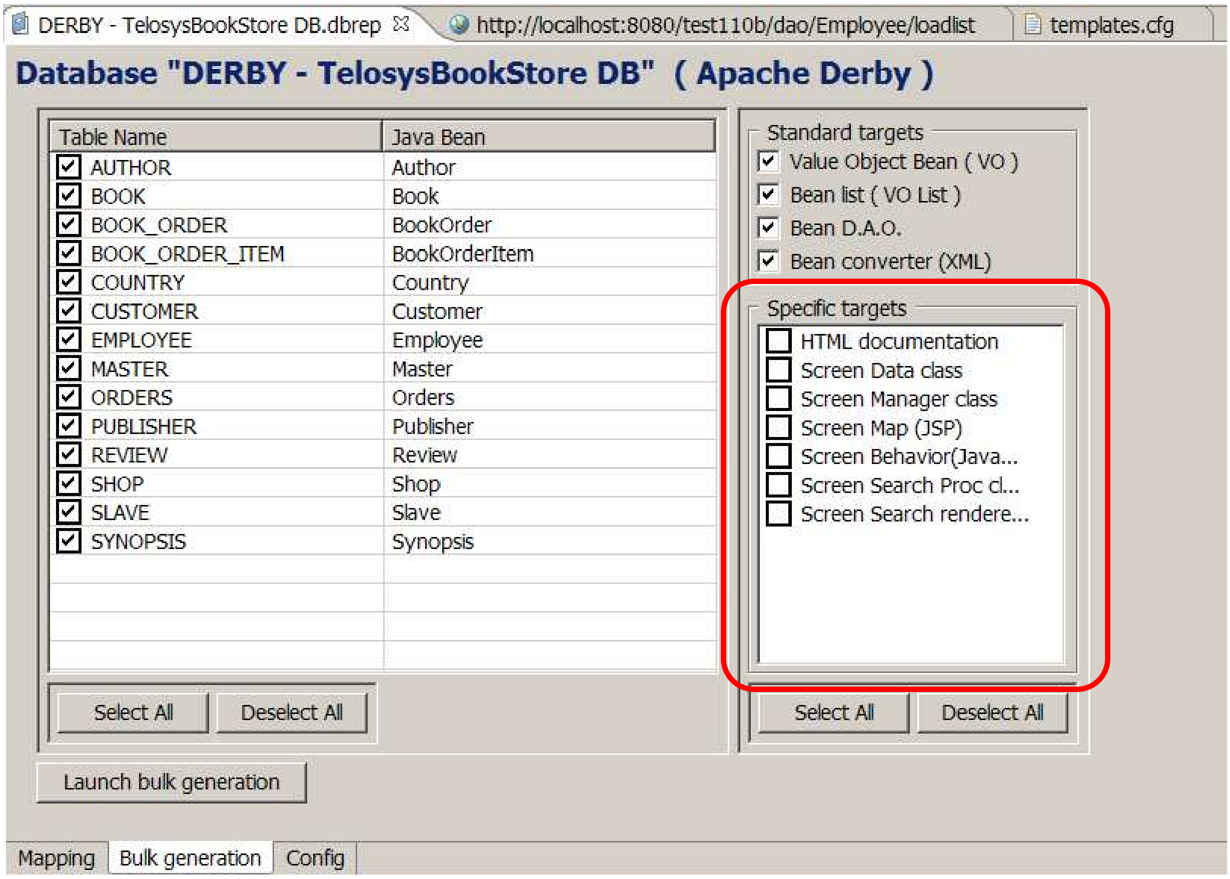
The Telosys Tools plugin is designed to be extended, in order to generate specific files according with the specific needs of a project.

The generation is based on Velocity templates (.vm files), located in the “TelosysTools/templates” folder.

How to add a new generation target ?

1. write the new Velocity template (.vm file)
2. add it in the “TelosysTools/templates” folder
3. reference it in the “TelosysTools/templates/templates.cfg”

When you re-open the repository editor, the specific targets appears in the “Bulk generation” view.


DERBY - TelosysBookStore DB.dbrep
http://localhost:8080/test110b/dao/Employee/loadlist
templates.cfg

Database "DERBY - TelosysBookStore DB" (Apache Derby)

Table Name	Java Bean
☑ AUTHOR	Author
☑ BOOK	Book
☑ BOOK_ORDER	BookOrder
☑ BOOK_ORDER_ITEM	BookOrderItem
☑ COUNTRY	Country
☑ CUSTOMER	Customer
☑ EMPLOYEE	Employee
☑ MASTER	Master
☑ ORDERS	Orders
☑ PUBLISHER	Publisher
☑ REVIEW	Review
☑ SHOP	Shop
☑ SLAVE	Slave
☑ SYNOPSIS	Synopsis

Standard targets
- ☑ Value Object Bean (VO)
- ☑ Bean list (VO List)
- ☑ Bean D.A.O.
- ☑ Bean converter (XML)

Specific targets
- □ HTML documentation
- □ Screen Data class
- □ Screen Manager class
- □ Screen Map (JSP)
- □ Screen Behavior(Java...
- □ Screen Search Proc cl...
- □ Screen Search rendere...

The Telosys starter-kit comes with a set of specific targets allowing “scaffolding” for basic CRUDS (Create/Retrieve/Update/Delete/Search) screens

The screenshot above shows these specific targets.

The specific targets by category :

Client-side components :

- ScreenMap view (.jsp)
- ScreenMap behavior (.js)

Server-side components :

- ScreenData class (.java)
- ScreenManager class (.java)
- ScreenProcedur class (.java)

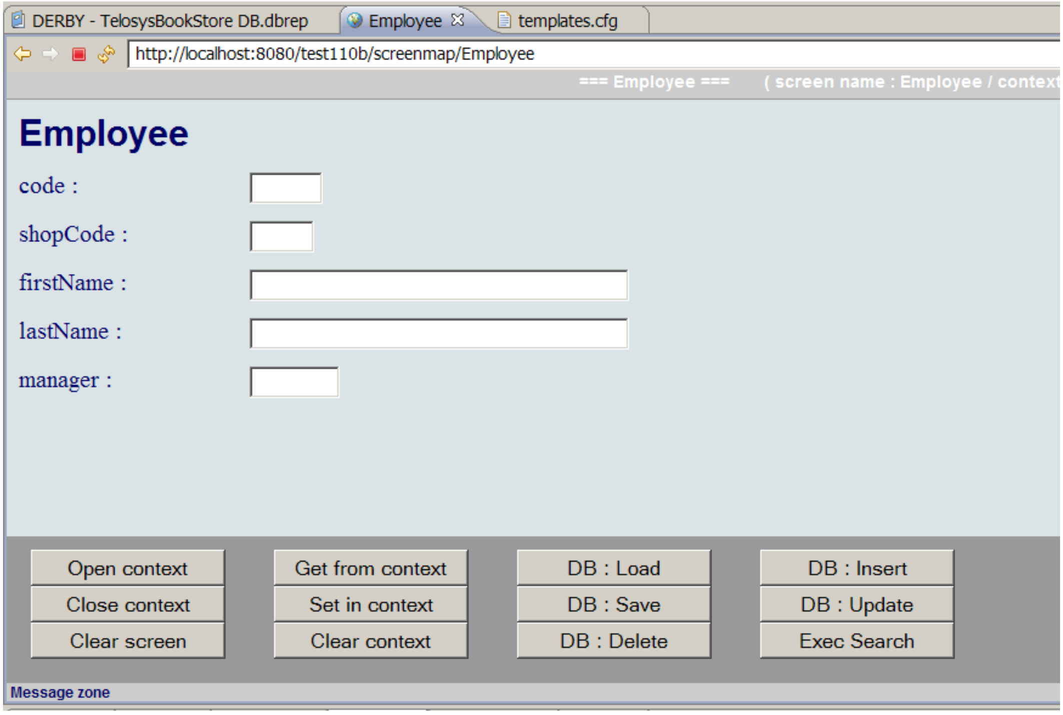
Rendering components :

- Search result renderer (.jsp)

Documentation :

- HTML page describing the entity (.html)

Test the specific targets with the "Employee" entity :
→ generate all the specific targets for the "Employee" entity
→ run the project
→ call the "Employee" screen by the URL :
http://localhost:8080/test110b/screenmap/Employee



Finally, you can generate all the targets for all the entities.

Now let's have a look at the generated components,
it's a good way to discover basic use cases of the framework…