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 …

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

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

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 …

If necessary, click the « Add » button to define a new server.
Select « Next » button and:
- choose the Tomcat installation directory
- select the JRE

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 )
Select «**Next**» button

Select «**Next**» button

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

Select « Finish » button
Insert “hello” in the <body> tag of the JSP file

```html
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" 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=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    hello
</body>
</html>
```

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

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

Select « Finish » button

Verify the good execution of your JSP file

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` ...

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:
  o `derbyclient.jar`: the JDBC driver for tests with a “Derby” database
  o `telosys-x.x.x.jar`: the Telosys framework

That’s all!

Your project is now a “Telosys project”.

Telosys Starter Kit – v 1.1.0 (September 2010)
3.2 – **Try to run the project**

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

The welcome page appears in the Eclipse web browser:

---

**Telosys test page**

**Test servers presence**

- Test screenshot server without screen name
- Test actions server without action name
- Test services server with nonexistent service

**Test servers with real resources**

- Test screenshot without context ('Test1')
- Test screenshot with context on the server side ('Test2')
- Test screenshot with service on the server side ('Test3')
- Test services server with an existing EPC service ('pong')

**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.

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:

“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:

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:

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)


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”
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 …


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:

```xml
<db id="0">
    <name>"DERBY - TelosysBookStore DB"
    <driver>"org.apache.derby.jdbc.ClientDriver"
    <url>"jdbc:derby://localhost:1527/bookstore"
    <isolationLevel>""</url>
    <poolSize>4</poolSize>
    <property name="name" value="root" />
    <property name="password" value="admin" />
</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.

Select the “Derby Bookstore” database

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

Click the “OK” button.

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:

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:

![Diagram of Book entity relationships]


**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):

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="MARCEL" lastName="DUFENOUJX" manager="0" />
      <Item code="C001" shopCode="CCC" firstName="MARCEL" lastName="JOSIAN" manager="1" />
      <Item code="C002" shopCode="CCC" firstName="KEVIN" lastName="TOTTO" 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="VIRSO" 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.

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…