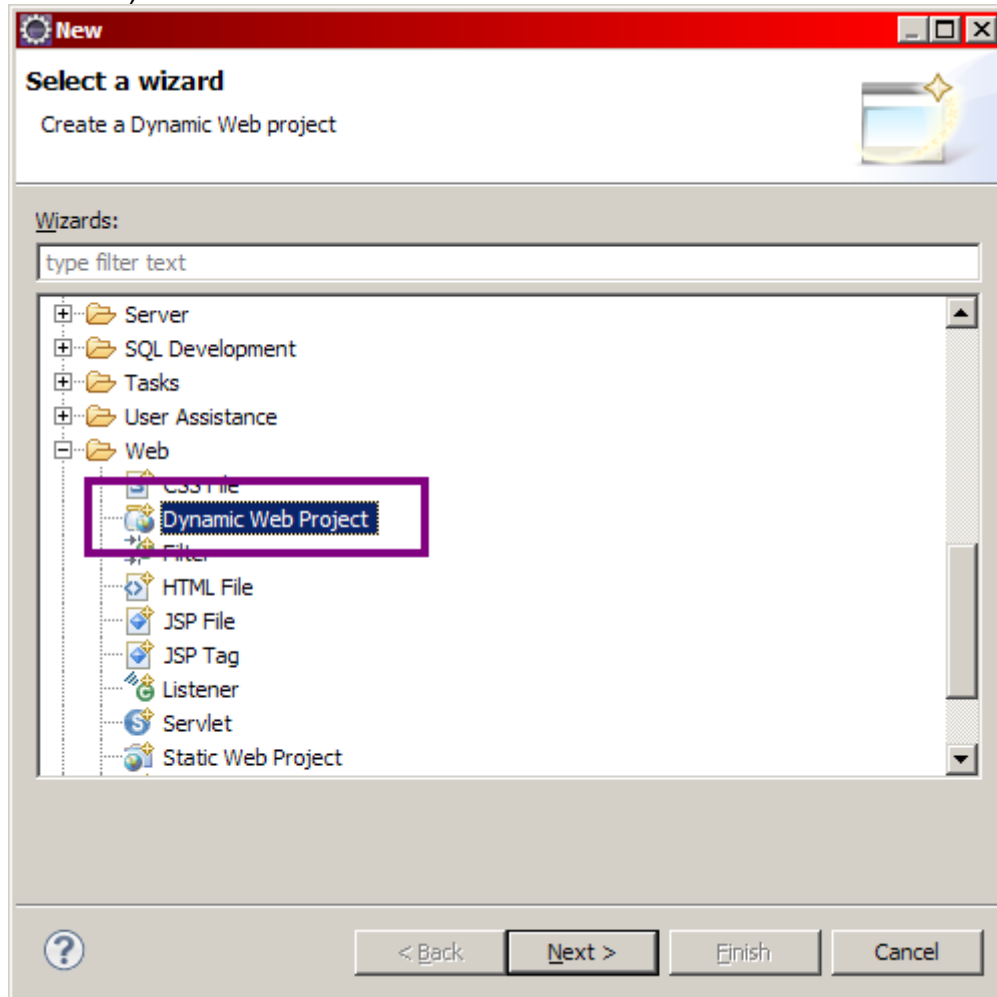


Création du projet web avec Eclipse

Depuis Eclipse, suivez le chemin suivant : `File > New > Project...` (voir la figure suivante).



Sélectionnez alors **Dynamic Web Project** comme le montre l'image ci-dessus, puis cliquez sur : `Next >`

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: test

Project location
☒ Use default location
Location: C:\eclipse_sdz\workspace\test Browse...

Target runtime
<None> New Runtime...

Dynamic web module version
3.0

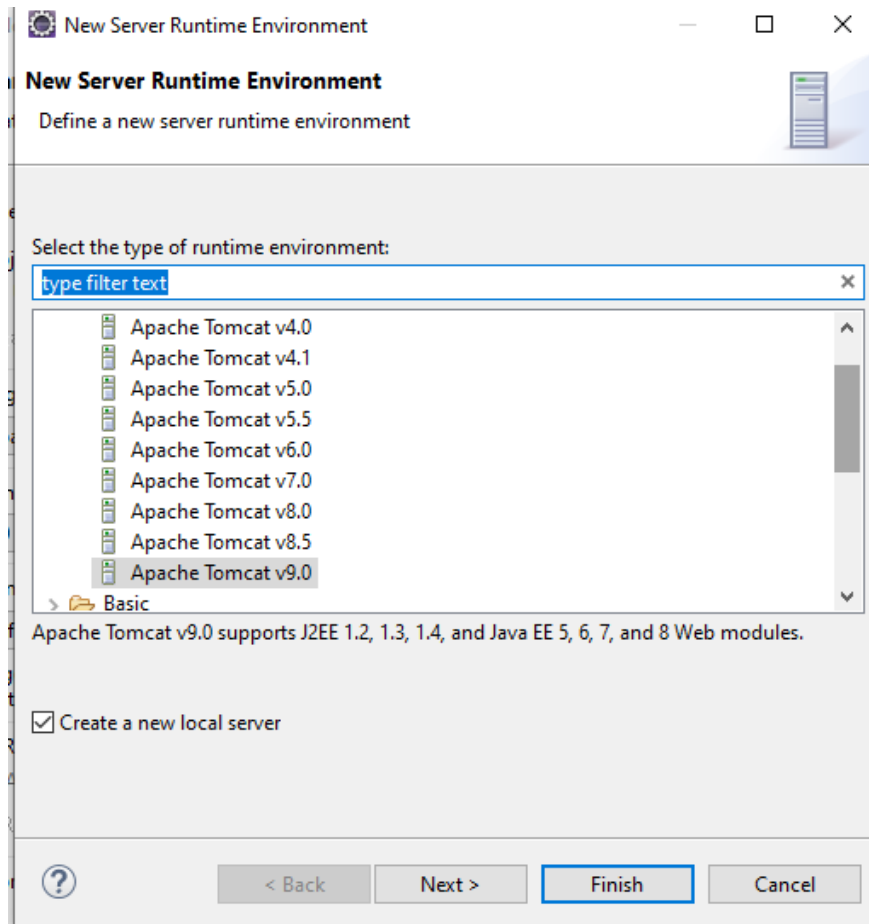
Configuration
Default Configuration Modify...
The default configuration provides a good starting point. Additional facets can later be installed to add new functionality to the project.

EAR membership
☐ Add project to an EAR
EAR project name: EAR New Project...

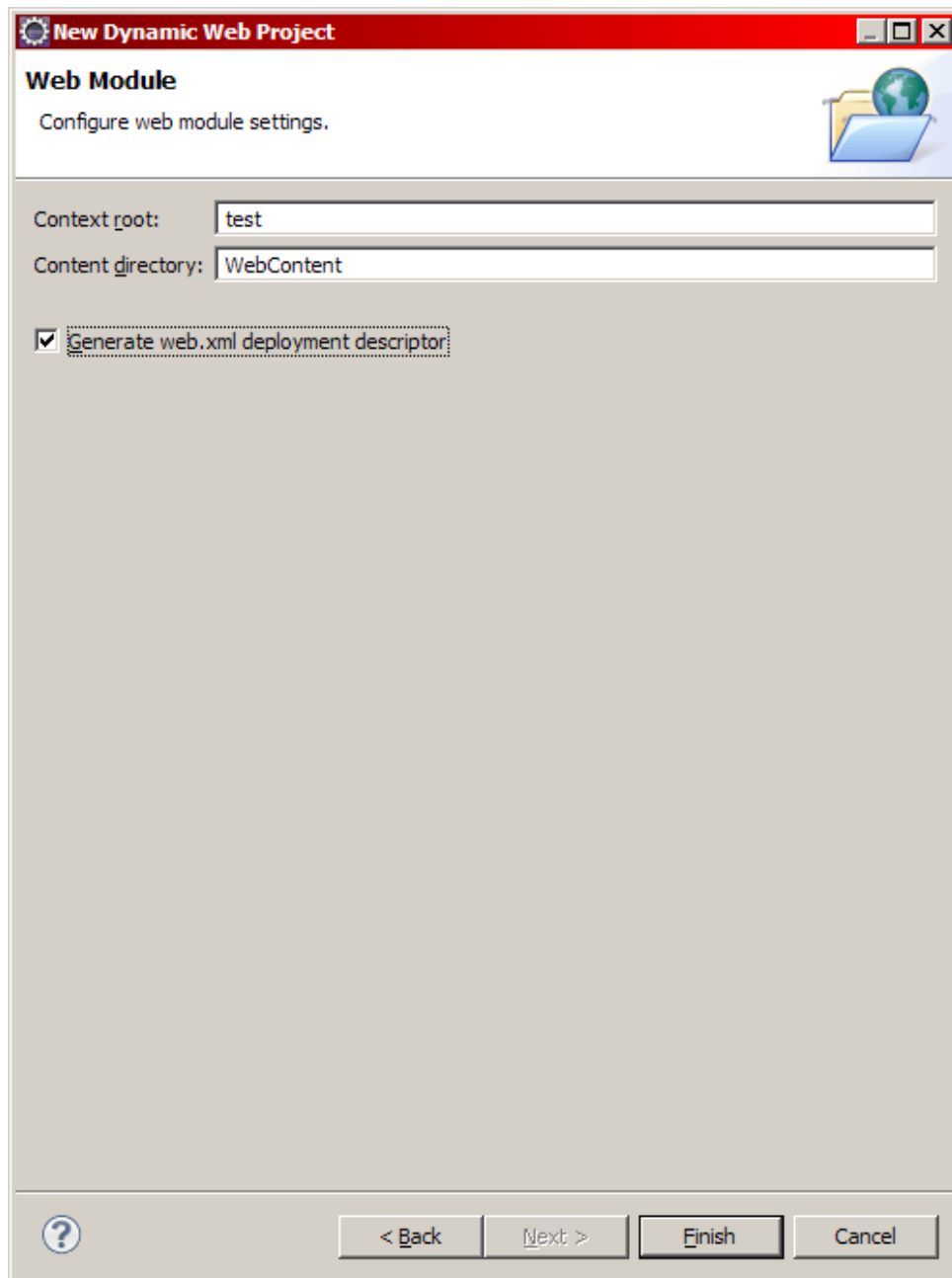
Working sets
☐ Add project to working sets
Working sets: Select...

? < Back Next > Finish Cancel

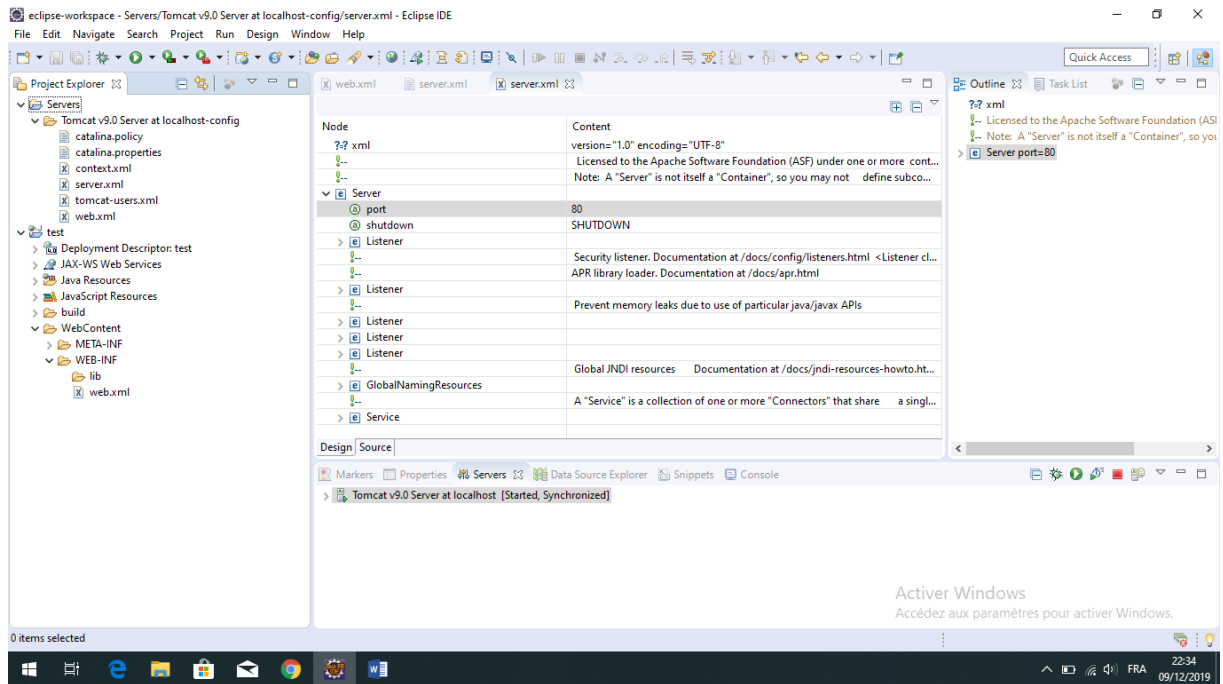
Cliquez sur le bouton **New Runtime...** et sélectionnez alors Apache Tomcat v9.0 et Cochez la case comme indiqué ci-dessus, ce qui signifie que nous allons en plus du projet créer localement une nouvelle instance d'un serveur, instance que nous utiliserons par la suite pour déployer notre application. Cliquez ensuite sur **Next >** et remplissez correctement les informations relatives à votre installation de Tomcat.



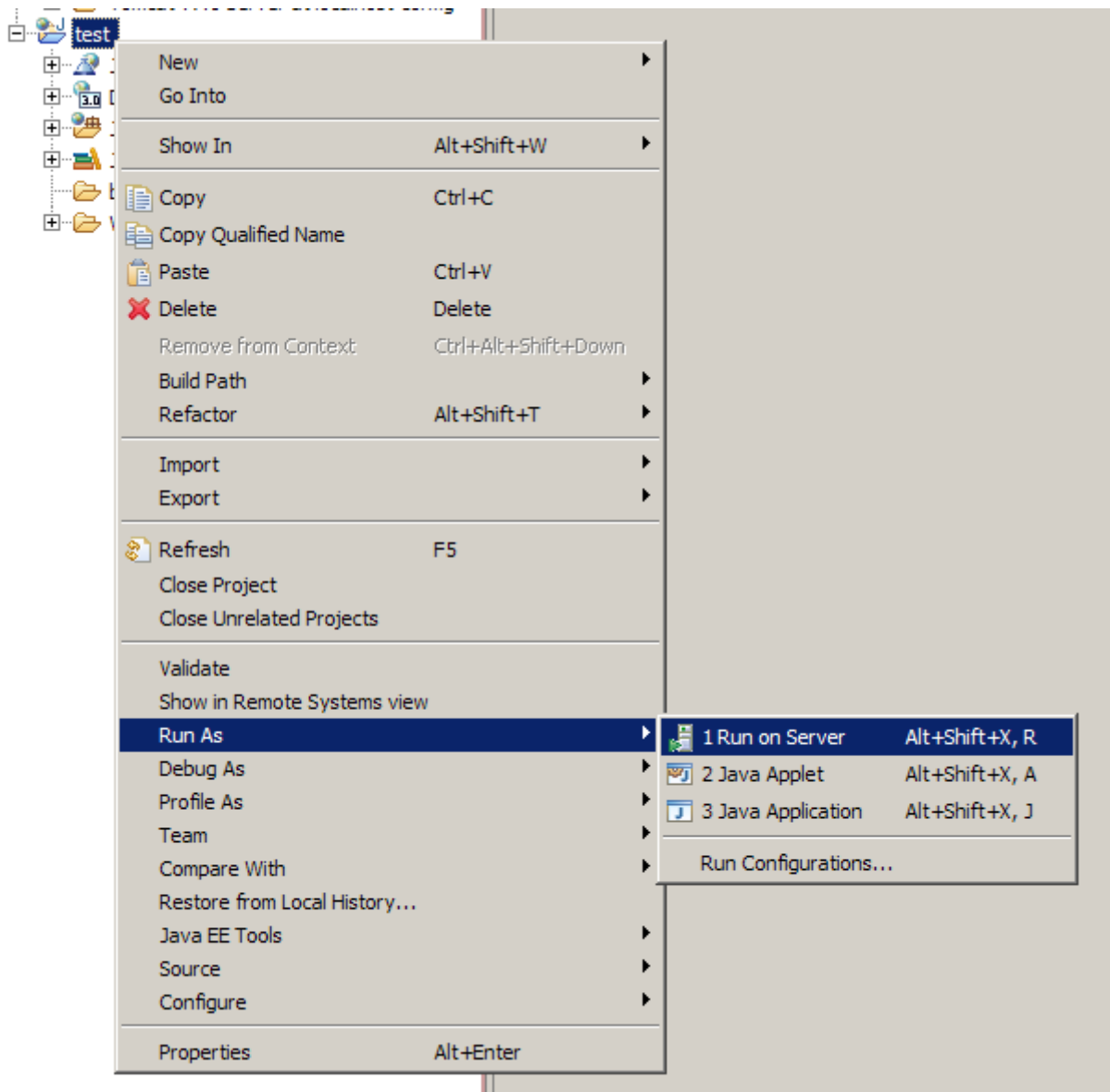
Validez alors en cliquant sur `Finish`, puis cliquez deux fois sur `Next >`, jusqu'à obtenir cette fenêtre (voir la figure suivante).



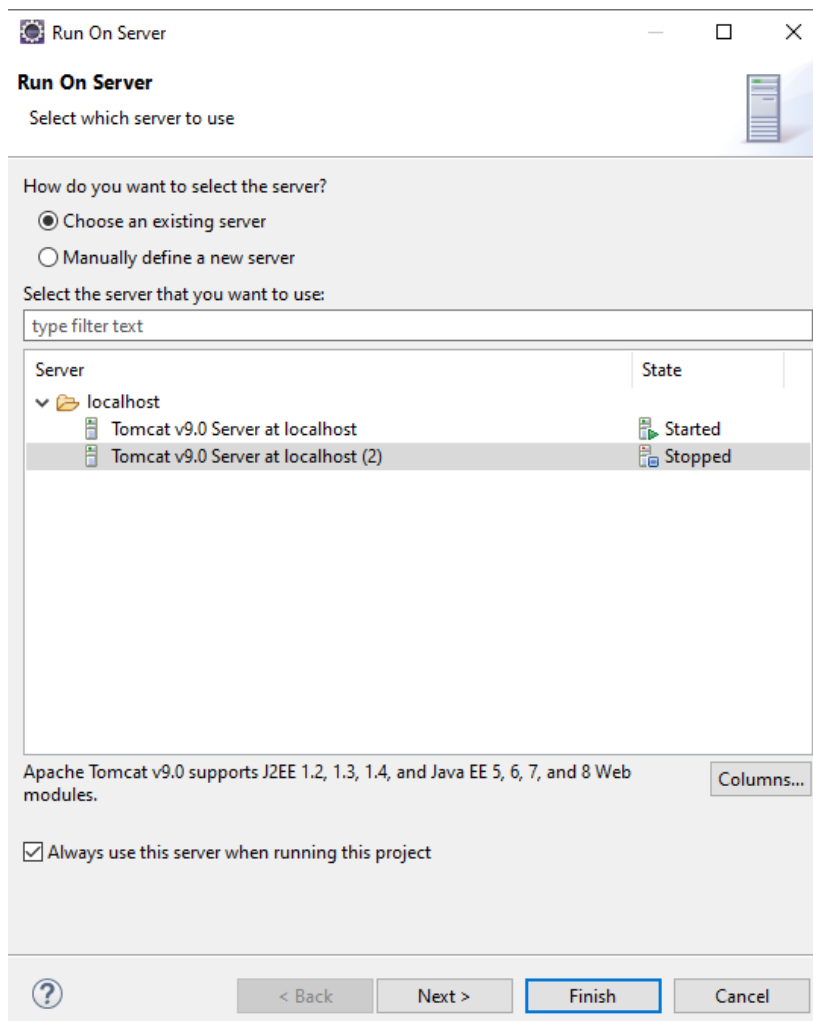
Voici maintenant à la figure suivante ce à quoi doit ressembler votre fenêtre Eclipse.



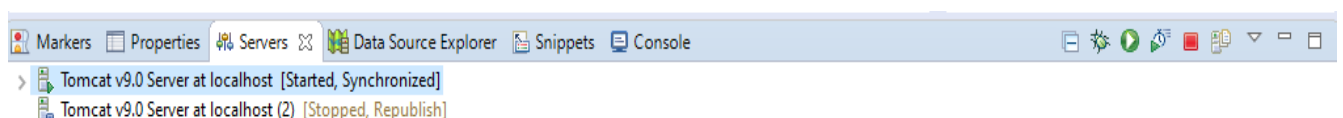
Faites maintenant un clic droit sur le titre de votre projet dans l'arborescence Eclipse, et suivez Run As > Run on Server, comme indiqué à la figure suivante.



Dans la fenêtre qui s'ouvre alors (voir la figure suivante), nous allons sélectionner le serveur Tomcat que nous venons de mettre en place lors de la création de notre projet web, et préciser que l'on souhaite associer par défaut notre projet à ce serveur.

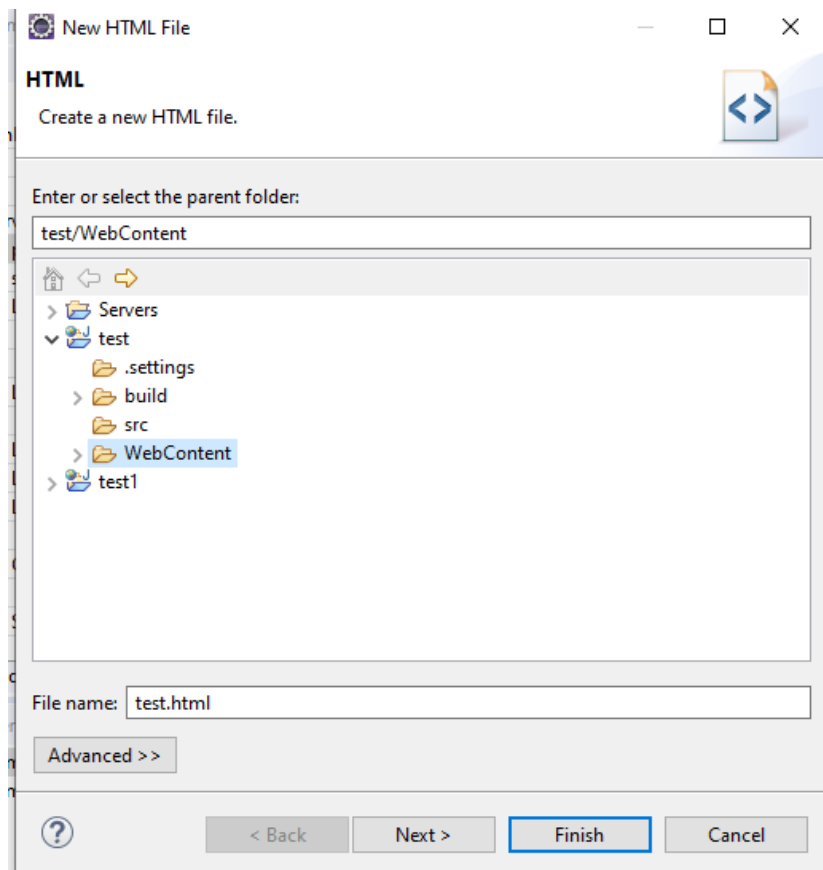
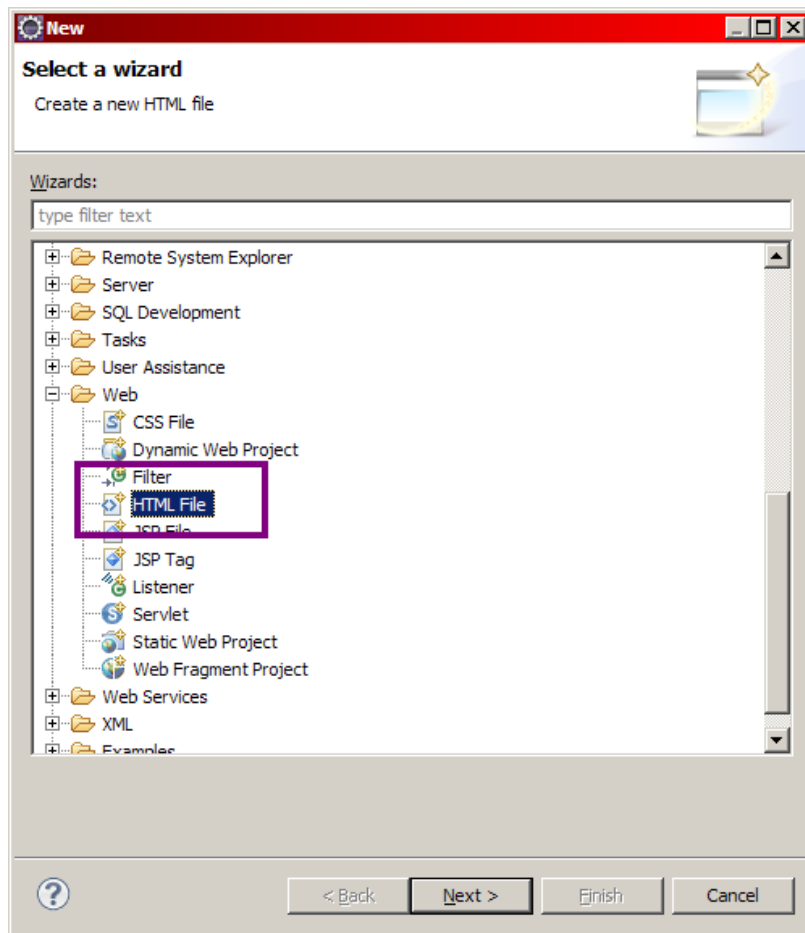


Dorénavant, pour piloter votre serveur Tomcat il vous suffira de vous rendre dans l'onglet **Servers** en bas de votre fenêtre Eclipse, et d'utiliser un des boutons selon le besoin (redémarrage, arrêt, debug), comme indiqué à la figure suivante.




Création d'une page web

Vous pouvez maintenant placer votre première page HTML dans son dossier public, c'est-à-dire sous le dossier **WebContent** d'Eclipse (voir le bloc bleu sur notre schéma). Pour cela, tapez une nouvelle fois **Ctrl** + **N** au clavier, puis cherchez **HTML File** dans le dossier **Web** de l'arborescence qui apparaît alors. Sélectionnez ensuite le dossier parent, en l'occurrence le dossier **WebContent** de votre projet, puis donnez un nom à votre page *test.html* et enfin validez.



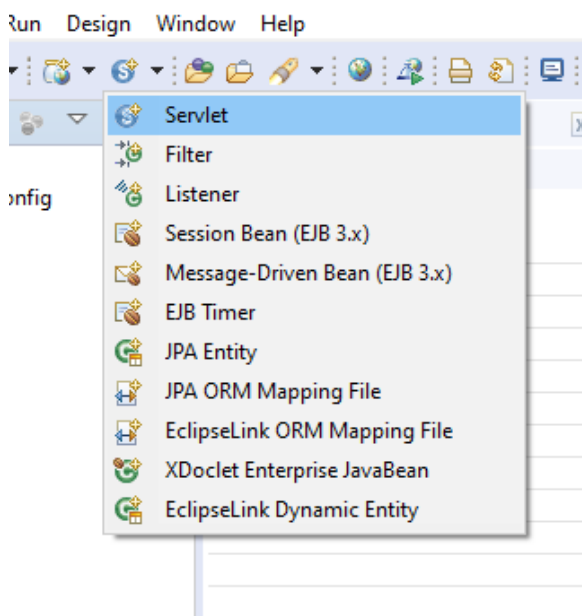
Une page HTML est donc apparue dans votre projet, sous le répertoire **WebContent**. Remplacez alors le code automatiquement généré par Eclipse dans votre page par ce code HTML basique :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Test</title>
  </head>
  <body>
    <p>Ceci est une page HTML.</p>
  </body>
</html>
```

Vous pouvez maintenant tenter d'accéder à votre page web fraîchement créée. Pour ce faire, lancez le serveur Tomcat, via le bouton .

Ouvrez ensuite votre navigateur préféré, et entrez l'URL suivante afin d'accéder à votre serveur : <http://localhost:8080/test/test.html>
Ou bien choisir run on server.

Création d'une première servlet :



Spécifiez la classe de cette servlet : son nom, et le nom du *package* dans lequel cette classe sera placée. Une servlet étend nécessairement `HttpServlet`.

Create Servlet
Specify class file destination.

Project: test

Source folder: \test\src Browse...

Java package: web.servlet.test Browse...

Class name: HelloWorld

Superclass: javax.servlet.http.HttpServlet Browse...

☐ Use an existing Servlet class or JSP

Class name: HelloWorld Browse...

< Back Next > Finish Cancel

Le panneau suivant nous permet de fixer les informations sur cette servlet, qui seront écrites dans le fichier web.xml de notre application.

- Le nom de notre servlet, utilisé pour s'y référer dans le fichier web.xml.
- Ses paramètres d'initialisation.
- Le (ou les) paramètres *URL mappings*, qui fixeront l'URL à laquelle notre servlet sera disponible.

Create Servlet

Enter servlet deployment descriptor specific information.

Name:

Description:

Initialization parameters:

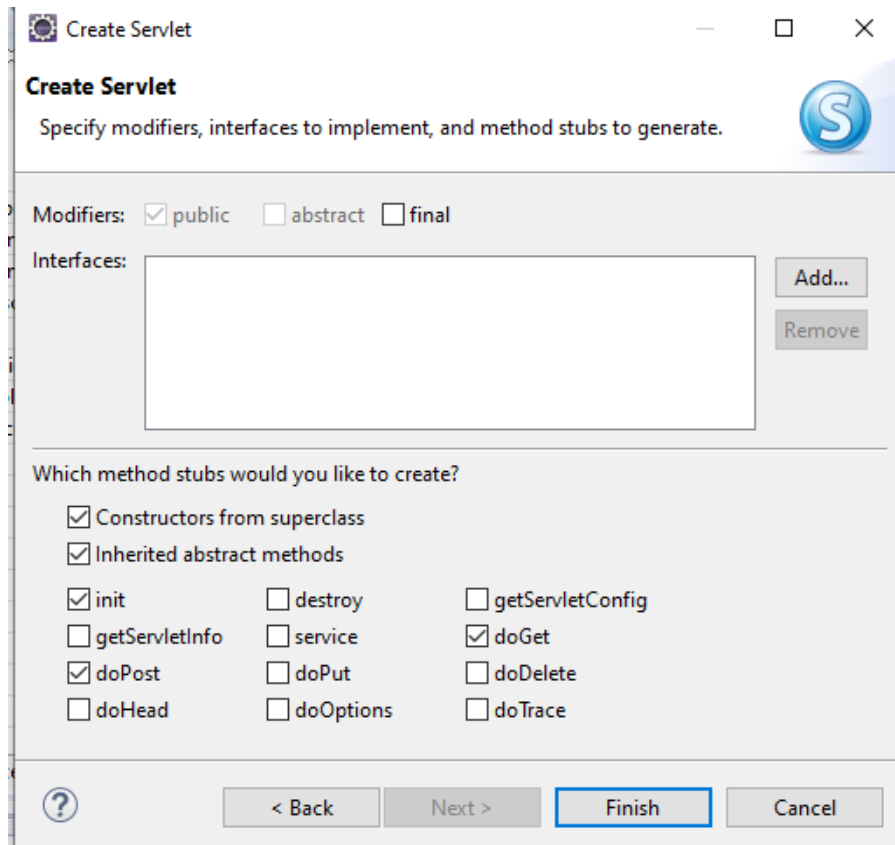
Name	Value	Description
------	-------	-------------

URL mappings:

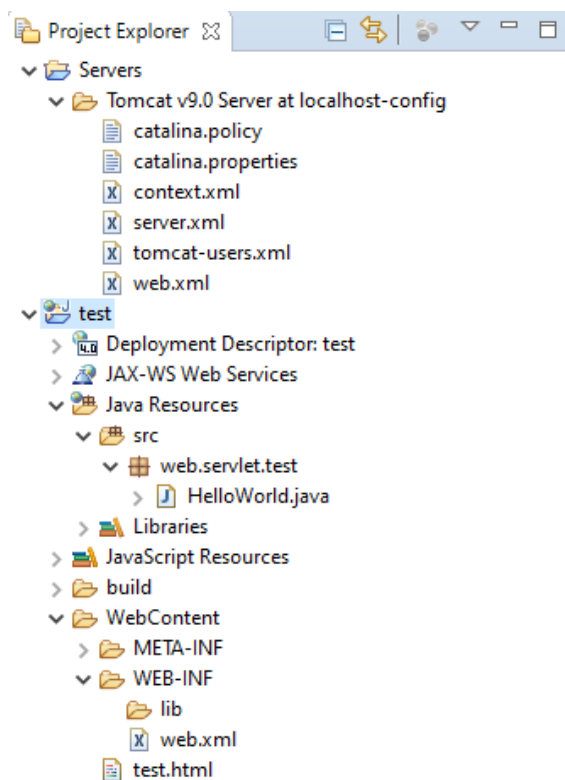
/HelloWorld

☐ Asynchronous Support

Le dernier panneau de création précise le contenu de la classe qu'Eclipse va nous générer. Par commodité, Eclipse peut créer pour nous différentes méthodes standard de notre servlet, telles que `init()`, `doGet()` ou `doPost()`.



Une fois ces étapes déroulées, notre classe de servlet est créée, et apparaît dans la structure de notre projet.



Le fichier web.xml a bien été modifié, et prend bien en compte la servlet qui vient d'être créée. S'il ne l'est pas, il faudra donc saisir les données spécifiques à la servlet et son mapping à la main.

```
12<⊖ <servlet>
13    <description></description>
14    <display-name>HelloWorld</display-name>
15    <servlet-name>HelloWorld</servlet-name>
16    <servlet-class>web.servlet.test.HelloWorld</servlet-class>
17  </servlet>
18⊖ <servlet-mapping>
19  <servlet-name>HelloWorld</servlet-name>
20  <url-pattern>/HelloWorld</url-pattern>
21  </servlet-mapping>
22</web-app>
```

Le code de notre servlet est le suivant.

```
package web.servlet.test;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class HelloWorld
 */
@WebServlet("/HelloWorld")
public class HelloWorld extends HttpServlet {

    /**
     *
     */
    private static final long serialVersionUID = 1L;

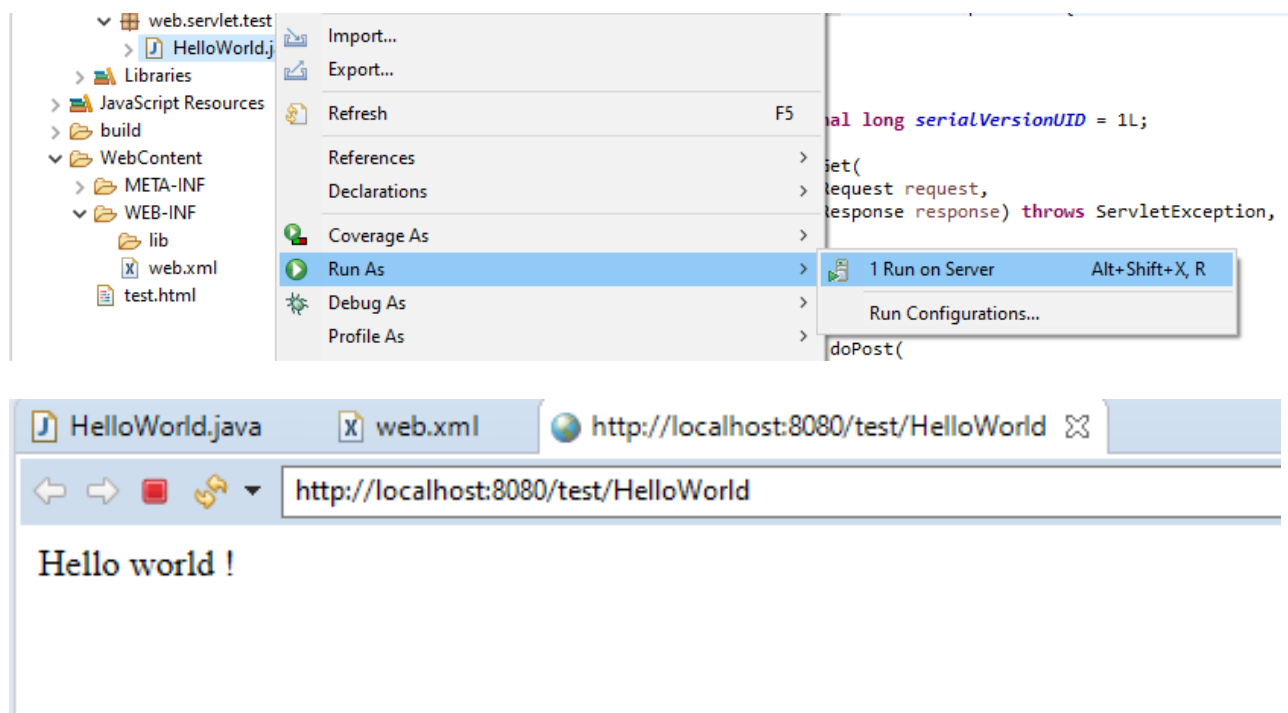
    protected void doGet(
        HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {

        PrintWriter pw = response.getWriter() ;
        pw.write("Hello world !") ;
    }

    protected void doPost(
        HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {

        doGet(request, response) ;
    }
}
```

Ensuite on exécute la servlet :



On peut également copier cette URL et la coller dans un navigateur lancé localement.

Exercice 1

Ecrire une servlet qui renvoie un tableau de 10 cases par 10 selon le modèle suivant :

M(1,1)	M(1,2)	M(1,3)	M(1,4)	M(1,5)	M(1,6)	M(1,7)	M(1,8)	M(1,9)	M(1,10)
M(2,1)	M(2,2)	M(2,3)	M(2,4)	M(2,5)	M(2,6)	M(2,7)	M(2,8)	M(2,9)	M(2,10)
M(3,1)	M(3,2)	M(3,3)	M(3,4)	M(3,5)	M(3,6)	M(3,7)	M(3,8)	M(3,9)	M(3,10)
M(4,1)	M(4,2)	M(4,3)	M(4,4)	M(4,5)	M(4,6)	M(4,7)	M(4,8)	M(4,9)	M(4,10)
M(5,1)	M(5,2)	M(5,3)	M(5,4)	M(5,5)	M(5,6)	M(5,7)	M(5,8)	M(5,9)	M(5,10)
M(6,1)	M(6,2)	M(6,3)	M(6,4)	M(6,5)	M(6,6)	M(6,7)	M(6,8)	M(6,9)	M(6,10)
M(7,1)	M(7,2)	M(7,3)	M(7,4)	M(7,5)	M(7,6)	M(7,7)	M(7,8)	M(7,9)	M(7,10)
M(8,1)	M(8,2)	M(8,3)	M(8,4)	M(8,5)	M(8,6)	M(8,7)	M(8,8)	M(8,9)	M(8,10)
M(9,1)	M(9,2)	M(9,3)	M(9,4)	M(9,5)	M(9,6)	M(9,7)	M(9,8)	M(9,9)	M(9,10)
M(10,1)	M(10,2)	M(10,3)	M(10,4)	M(10,5)	M(10,6)	M(10,7)	M(10,8)	M(10,9)	M(10,10)

Exercice 2 :

Refaire l'exercice précédent de manière à obtenir une table de multiplication de 10 cases par 10 au centre de la page avec une bordure de 1px :

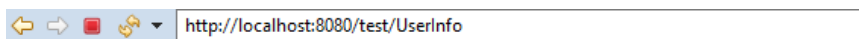
- Le texte de chaque cellule doit être mis au centre de celle-ci.
- La première cellule contient x en gras sur un fond jaune.
- Le texte de des premières ligne et colonne doit être en gras sur un fond gris.

Comme suit :

x	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

Exercice 3 :

- Ecrire une page HTML qui contient un formulaire composé de 3 champs : Nom, Prénom et Age, et un bouton pour l'envoi des informations.
- Ecrire ensuite une servlet UserInfo qui récupère ces informations et les affiche.



Recapitulatif des informations

- Nom: Koulali
- Prenom: Sara
- Age: 31