

Aplicaciones JAVA EE

IVÁN CÓRDOBA DONET

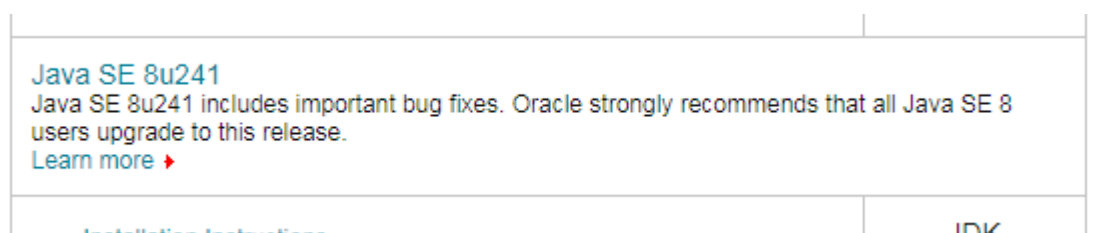
Indice

| | |
|--|----------------|
| <u>1. Tomcat en Windows.....</u> | <u>pág. 3</u> |
| <u>1.1. Instalación de java en Windows.....</u> | <u>pág. 3</u> |
| <u>1.2. Instalación de Tomcat en Windows.....</u> | <u>pág. 4</u> |
| <u>1.2.1. Instalación de Tomcat en Windows con asistente.....</u> | <u>pág. 4</u> |
| <u>1.2.2. Instalación de Tomcat en Windows de forma manual.....</u> | <u>pág. 10</u> |
| | |
| <u>2. Servlets.....</u> | <u>pág. 13</u> |
| <u>2.1. Instalación de Eclipse.....</u> | <u>pág. 13</u> |
| <u>2.2. Integración de Eclipse y Tomcat.....</u> | <u>pág. 14</u> |
| <u>2.3. Despliegue en Tomcat.....</u> | <u>pág. 27</u> |
| <u>2.4. Servlets y base de datos.....</u> | <u>pág. 38</u> |
| <u>2.4.1. Instalación de mysql y conectores.....</u> | <u>pág. 38</u> |
| <u>2.4.2. Creación de un Servlet con acceso a base de datos.....</u> | <u>pág. 43</u> |
| | |
| <u>3. Tomcat en Linux.....</u> | <u>pág. 49</u> |
| <u>3.1 Instalación de Java en Linux.....</u> | <u>pág. 49</u> |
| <u>3.2 Instalación de Tomcat en Linux.....</u> | <u>pág. 49</u> |
| <u>3.3 Estructura de Tomcat.....</u> | <u>pág. 52</u> |

1. Tomcat en Windows

1.1. Instalación de Java en Windows

Para instalar java vamos a la url <http://www.oracle.com/technetwork/java/javase/downloads> y bajamos hasta la versión que queramos descargar, en este caso la versión 8.



Vamos a la parte de descarga aceptamos los términos y condiciones y descargamos la versión que necesitamos. En este caso **Windows x86**

| Java SE Development Kit 8u241 | | |
|---|-----------|---|
| You must accept the Oracle Technology Network License Agreement for Oracle Java SE to download this software. | | |
| <input checked="" type="radio"/> Accept License Agreement <input type="radio"/> Decline License Agreement | | |
| Product / File Description | File Size | Download |
| Linux ARM 32 Hard Float ABI | 72.94 MB | jdk-8u241-linux-arm32-vfp-hflt.tar.gz |
| Linux ARM 64 Hard Float ABI | 69.83 MB | jdk-8u241-linux-arm64-vfp-hflt.tar.gz |
| Linux x86 | 171.28 MB | jdk-8u241-linux-i586.rpm |
| Linux x86 | 186.1 MB | jdk-8u241-linux-i586.tar.gz |
| Linux x64 | 170.65 MB | jdk-8u241-linux-x64.rpm |
| Linux x64 | 185.53 MB | jdk-8u241-linux-x64.tar.gz |
| Mac OS X x64 | 254.06 MB | jdk-8u241-macosx-x64.dmg |
| Solaris SPARC 64-bit (SVR4 package) | 133.01 MB | jdk-8u241-solaris-sparcv9.tar.Z |
| Solaris SPARC 64-bit | 94.24 MB | jdk-8u241-solaris-sparcv9.tar.gz |
| Solaris x64 (SVR4 package) | 133.8 MB | jdk-8u241-solaris-x64.tar.Z |
| Solaris x64 | 92.01 MB | jdk-8u241-solaris-x64.tar.gz |
| Windows x86 | 200.86 MB | jdk-8u241-windows-i586.exe |
| Windows x64 | 210.92 MB | jdk-8u241-windows-x64.exe |

Una vez descargado ejecutamos el programa y seguimos el asistente de instalación.

1.2. Instalación de Tomcat en Windows

Se puede instalar Apache Tomcat en Windows de dos maneras, con el asistente de instalación o de forma manual.

1.2.1. Instalación de Tomcat en Windows con asistente de instalación

Primero iremos a la página oficial de Apache Tomcat (<http://tomcat.apache.org/>), y bajaremos hasta la versión que queramos descargar, en este ejemplo será la versión 7.0.99 y clicamos en 'Download' o 'Descarga' (depende del idioma).

Tomcat 7.0.99 Released

2019-12-17

The Apache Tomcat Project is proud to announce the release of version 7.0.99 of Apache Tomcat. This release contains a number of bug fixes and improvements compared to version 7.0.96.

Full details of these changes, and all the other changes, are available in the [Tomcat 7 changelog](#).

[Download](#)

Luego iremos en la sección de 'CORE' y clicamos en el instalador de windows

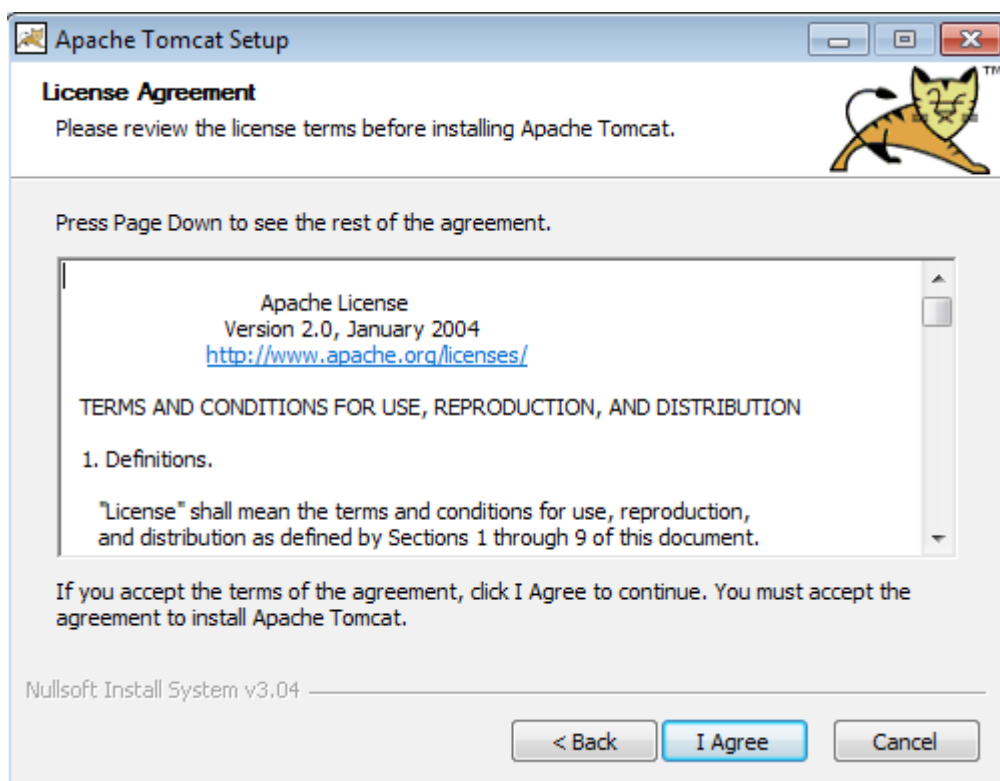
Binary Distributions

- Core:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [32-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [64-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [32-bit/64-bit Windows Service Installer](#) ([pgp](#), [sha512](#))
- Full documentation:
 - [tar.gz](#) ([pgp](#), [sha512](#))
- Deployer:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
- Extras:
 - [JMX Remote jar](#) ([pgp](#), [sha512](#))
 - [Web services jar](#) ([pgp](#), [sha512](#))
 - [JULI adapters jar](#) ([pgp](#), [sha512](#))
 - [JULI log4j.jar](#) ([pgp](#), [sha512](#))
- Embedded:
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [zip](#) ([pgp](#), [sha512](#))

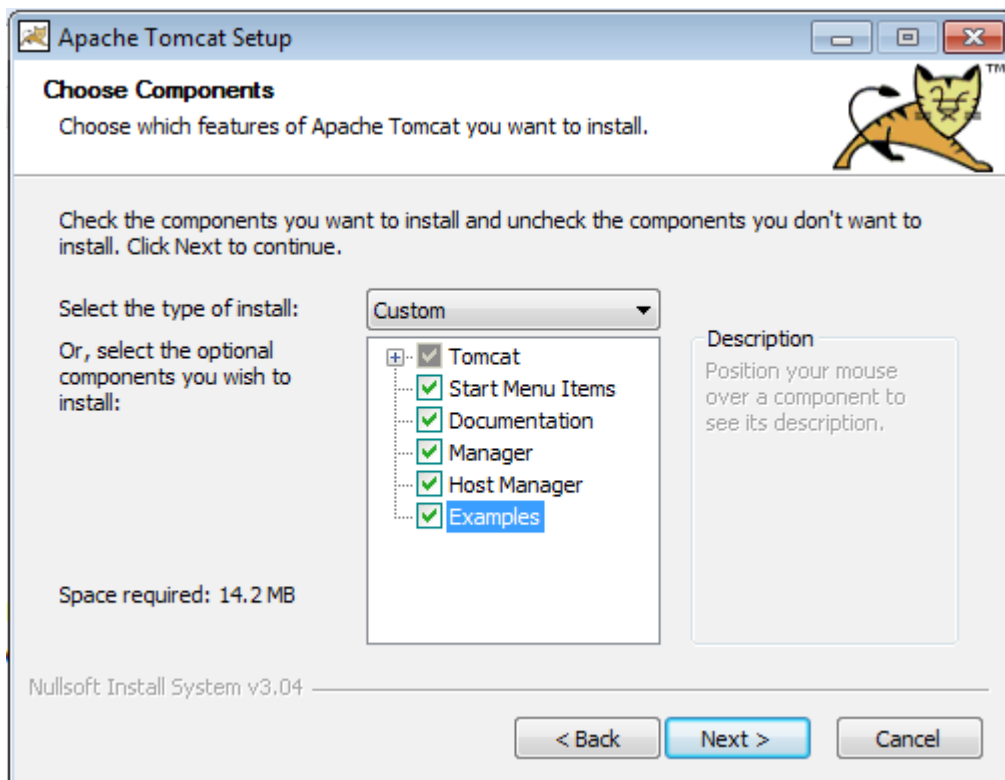
Source Code Distributions

- [tar.gz](#) ([pgp](#), [sha512](#))
- [zip](#) ([pgp](#), [sha512](#))

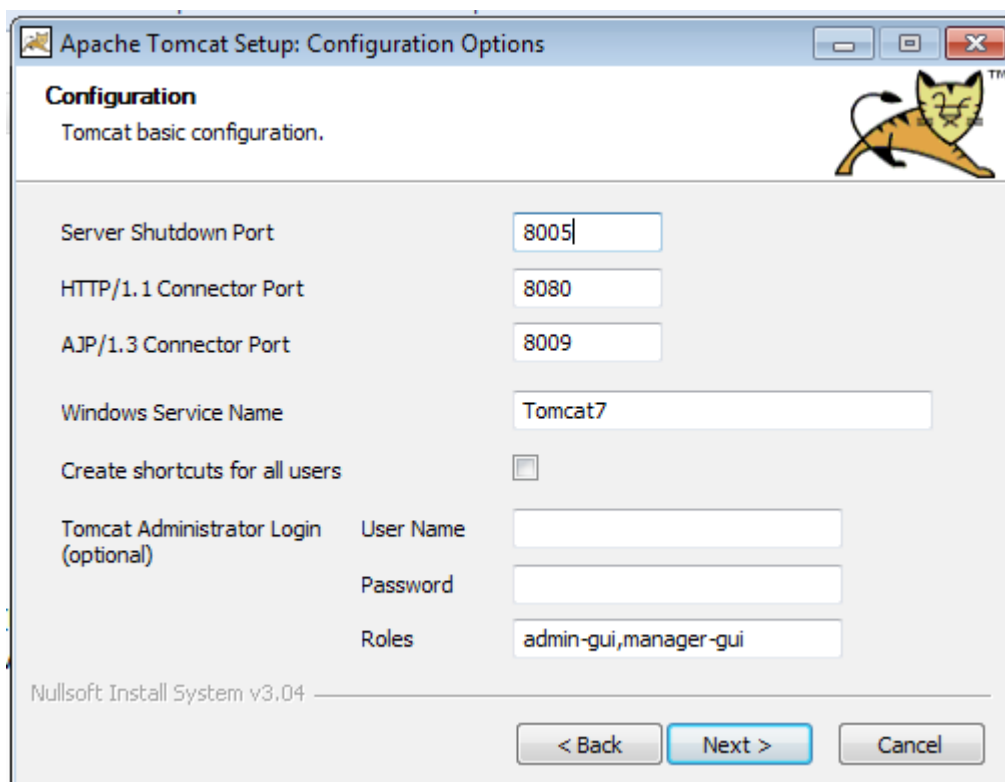
Una vez descargado el ejecutable lo iniciamos y seguimos el asistente.



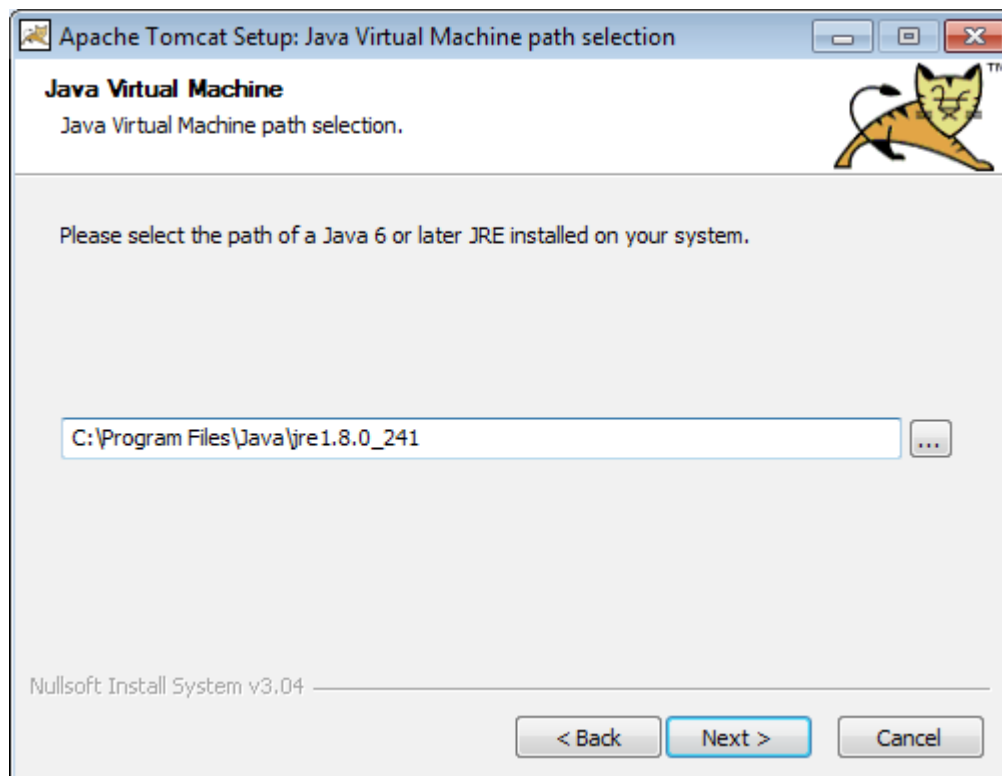
Selecciona los paquetes que quieras instalar y pulsa siguiente



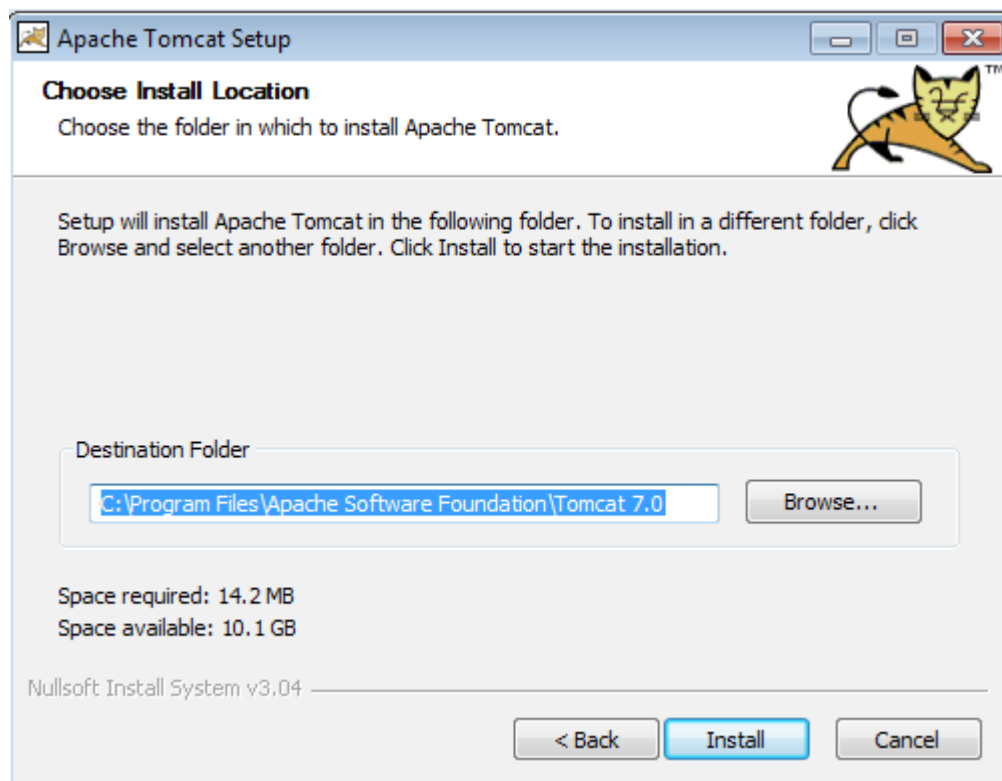
Configura los puerto del servidor de aplicaciones y siguiente.



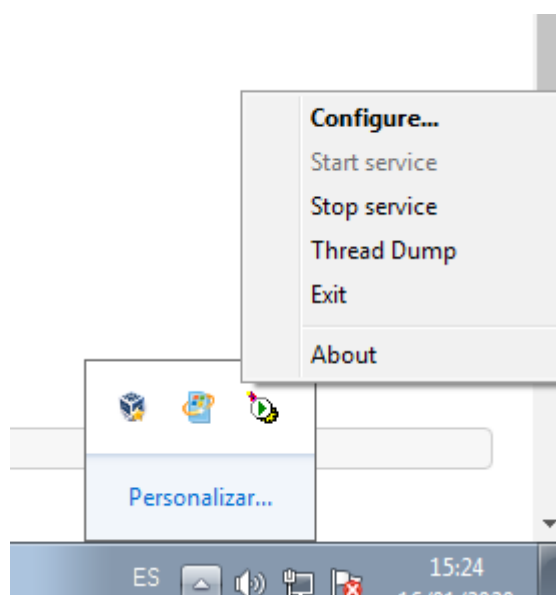
Indica la ruta en tu sistema de JAVA



Indica donde quieres instalar Tomcat



Cuando acabe la instalación veras un servicio corriendo en la barra de tareas que indica que Apache Tomcat esta en funcionamiento



1.2.2. Instalación de Tomcat en Windows de forma manual

Primero iremos a la página oficial de apache Tomcat (<http://tomcat.apache.org/>), y bajaremos hasta la versión que queramos descargar, en este ejemplo será la versión 7.0.99 y clicamos en ‘Download’ o ‘Descarga’ (depende del idioma).

Tomcat 7.0.99 Released

2019-12-17

The Apache Tomcat Project is proud to announce the release of version 7.0.99 of Apache Tomcat. This release contains a number of bug fixes and improvements compared to version 7.0.96.

Full details of these changes, and all the other changes, are available in the [Tomcat 7 changelog](#).

[Download](#)

Luego iremos en la sección de ‘CORE’ y clicamos la versión zip a descargar (depende de si tu sistema es de 32 o de 64 bits)

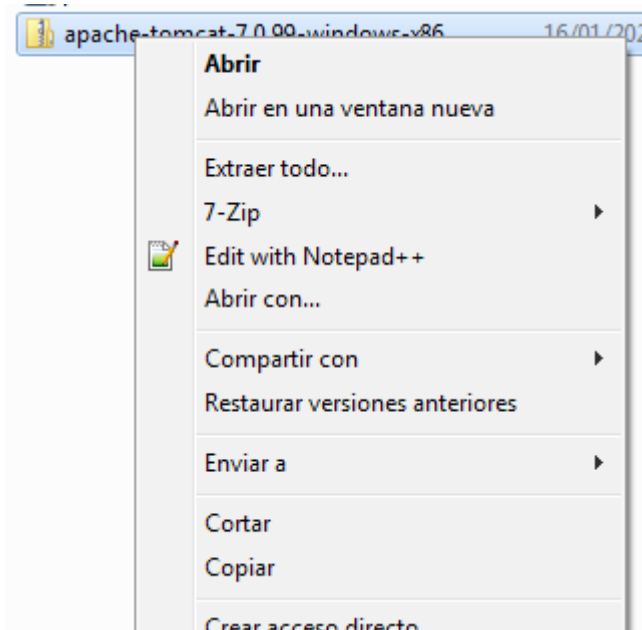
Binary Distributions

- Core:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [32-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [64-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [32-bit/64-bit Windows Service Installer](#) ([pgp](#), [sha512](#))
- Full documentation:
 - [tar.gz](#) ([pgp](#), [sha512](#))
- Deployer:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
- Extras:
 - [JMX Remote jar](#) ([pgp](#), [sha512](#))
 - [Web services jar](#) ([pgp](#), [sha512](#))
 - [JULI adapters jar](#) ([pgp](#), [sha512](#))
 - [JULI log4j.jar](#) ([pgp](#), [sha512](#))
- Embedded:
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [zip](#) ([pgp](#), [sha512](#))

Source Code Distributions

- [tar.gz](#) ([pgp](#), [sha512](#))
- [zip](#) ([pgp](#), [sha512](#))

Se descargará un zip que comprimirémos en la ruta '[C:\](#)' de forma que la ruta se queda en [C:\apache-tomcat-7.0.99](#).



Una vez hecho eso añadimos las variables del sistema de JAVA_HOME y de CATALINA_HOME

Panel de control

Sistema y seguridad

Sistema

Buscar en el Panel de control

Ventana principal del Panel de control

Administrador de dispositivos

Configuración de Acceso remoto

Protección del sistema

Configuración avanzada del sistema

Vea también

Centro de actividades

Windows Update


Información y herramientas de rendimiento

Ver información básica acerca del equipo


Edición de Windows

Windows 7 Ultimate

Copyright © 2009 Microsoft Corporation. Reservados todos los derechos.



Sistema

Evaluación:  Evaluación de la experiencia en Windows

Procesador: Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz 3.29 GHz

Memoria instalada (RAM): 2,00 GB

Tipo de sistema: Sistema operativo de 32 bits

Lápiz y entrada táctil: La entrada táctil o manuscrita no está disponible para esta pantalla


Configuración de nombre, dominio y grupo de trabajo del equipo

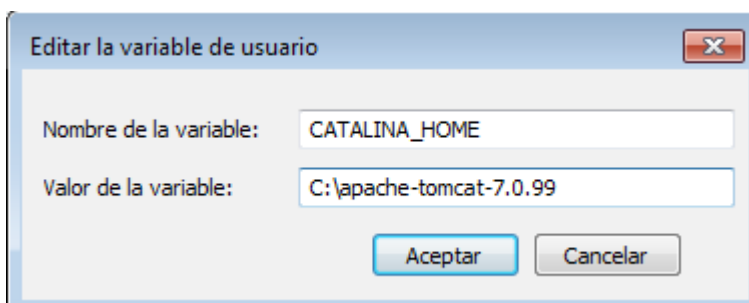
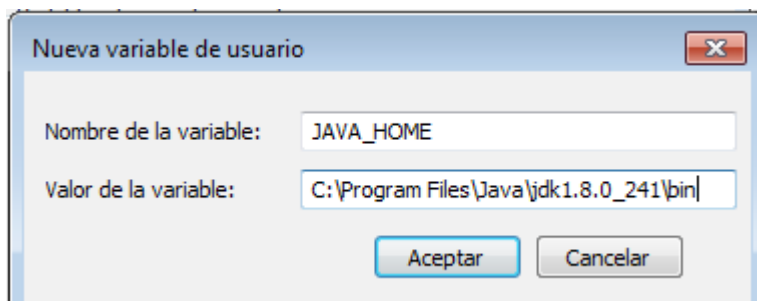
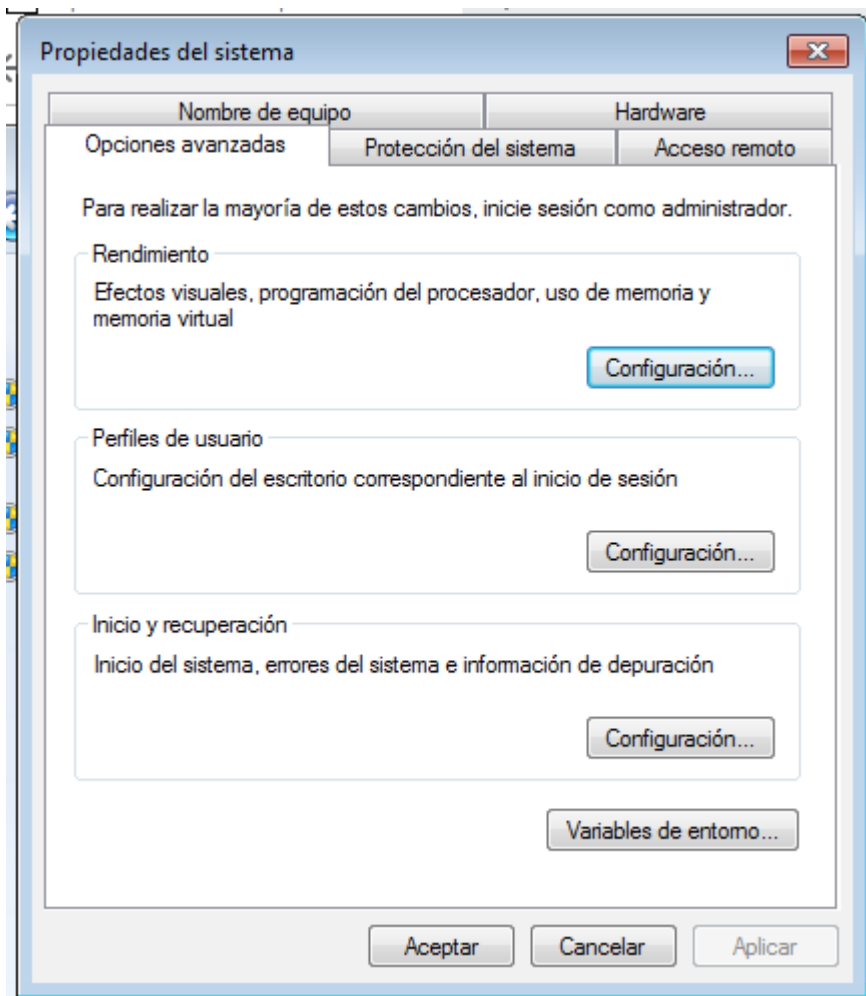
Nombre de equipo: DesarrolloW720

Nombre completo de equipo: DesarrolloW720.daw20.net

Descripción del equipo: DesarrolloW720

Grupo de trabajo: DESD1TEGUE20

 Cambiar configuración



Una vez puestas las variables podemos ejecutar el servidor de aplicaciones en una terminal poniendo '**%_CATALINA_HOME%\bin\startup.bat**' y para parar el servicio de aplicaciones usaremos '**%_CATALINA_HOME%\bin\shutdown.bat**'.

El servidor se ejecutara con el host local y el puerto configurado anteriormente.

2. Servlets

Eclipse es un entorno de desarrollo de software opensource. Para poder instalar eclipse es necesario tener Java instalado

2.1. Instalación de Eclipse

Vamos a la página de descarga de Eclipse

<https://www.eclipse.org/downloads/packages/release/kepler/sr2/eclipse-ide-java-ee-developers> y descargamos la versión que nos interese según el sistema operativo. Este eclipse es una versión portable

This package was released on 02/28/2014. A newer package is available [here](#).



Eclipse IDE for Java EE Developers

Package Description

Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn, EGit and others.

This package includes:

- Data Tools Platform
- Eclipse Git Team Provider
- Eclipse Java Development Tools
- Eclipse Java EE Developer Tools
- JavaScript Development Tools
- Maven Integration for Eclipse
- Mylyn Task List
- Eclipse Plug-in Development Environment

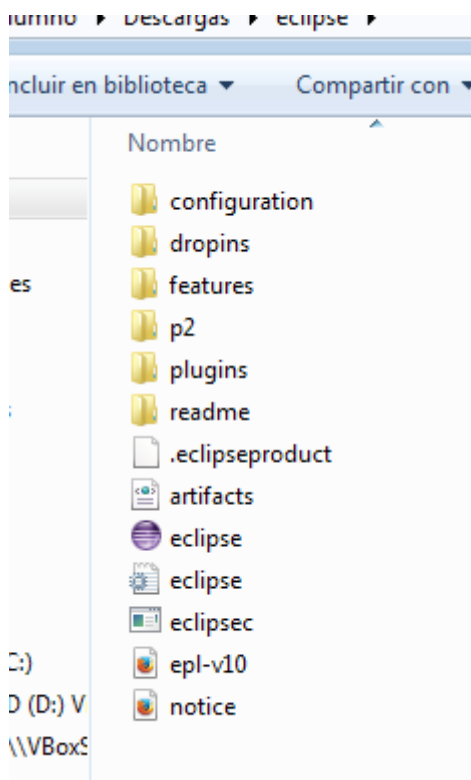
Download Links

[Windows 32-bit](#)
[Windows 64-bit](#)
[Mac OS X \(Cocoa\) 32-bit](#)
[Mac OS X \(Cocoa\) 64-bit](#)
[Linux 32-bit](#)
[Linux 64-bit](#)

Downloaded 3,741,922
Times

► [Checksums...](#)

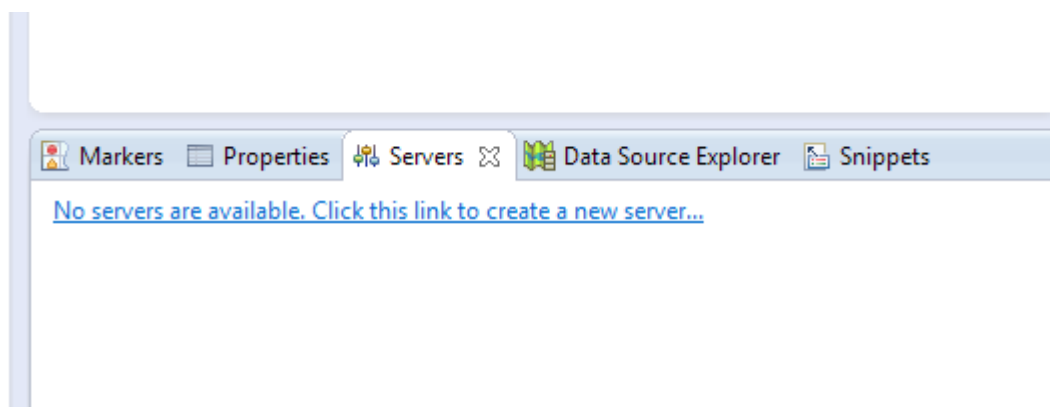
Esto descargará un archivo comprimido, lo descomprimos y ejecutamos el ejecutable **eclipse** y nos pedirá en que workspace queremos trabajar



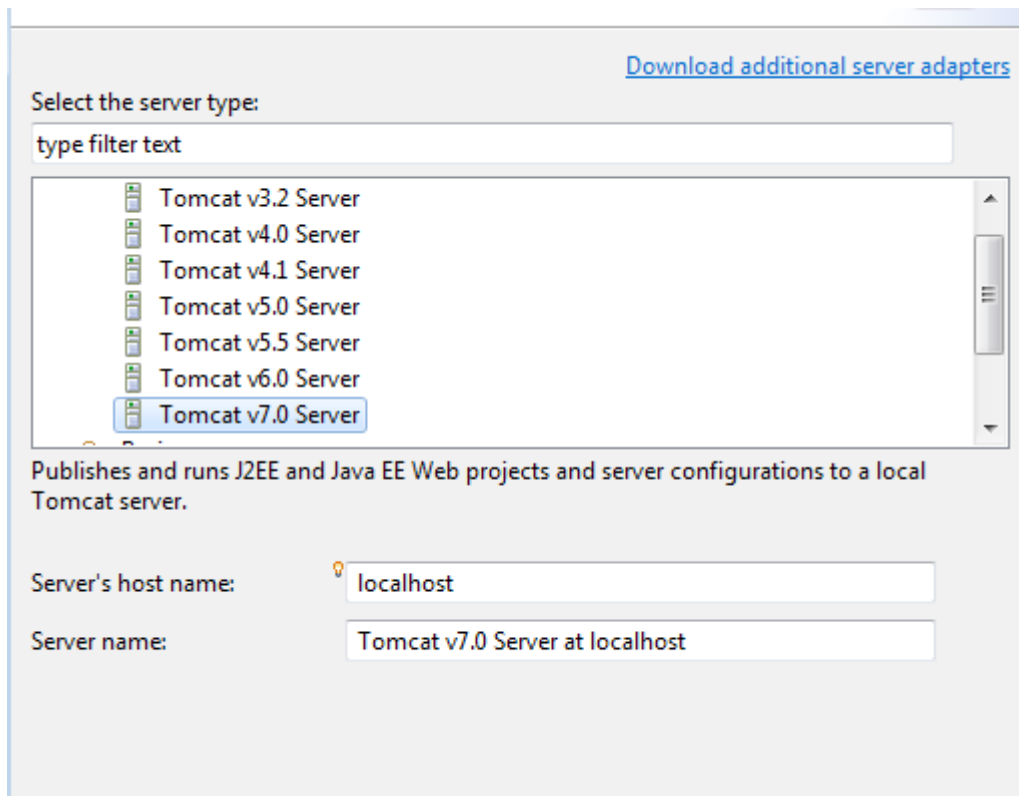
2.2. Integración de Eclipse con Tomcat

Podemos trabajar con eclipse y Tomcat de dos maneras. Una integrando apache Tomcat dentro de eclipse (nos da ciertas comodidades para desplegar en desarrollo) o podemos compilar el proyecto en un war y dejarlo caer en la carpeta publica de apache Tomcat llamada webapp. Vamos a ver la primera.


Nos movemos a la pestaña de server que hay en el bloque de abajo



y clicamos en el link para crear un nuevo server



nos pedirá que tipo de servidor queremos (ya que tiene integración con diferentes servicios de servidores. JBOSS, WINDFLY, TOMCAT....). Elegimos la versión que usemos y clicamos siguiente (en nuestro caso seleccionaremos Tomcat v7)

Tomcat Server 

Specify the installation directory

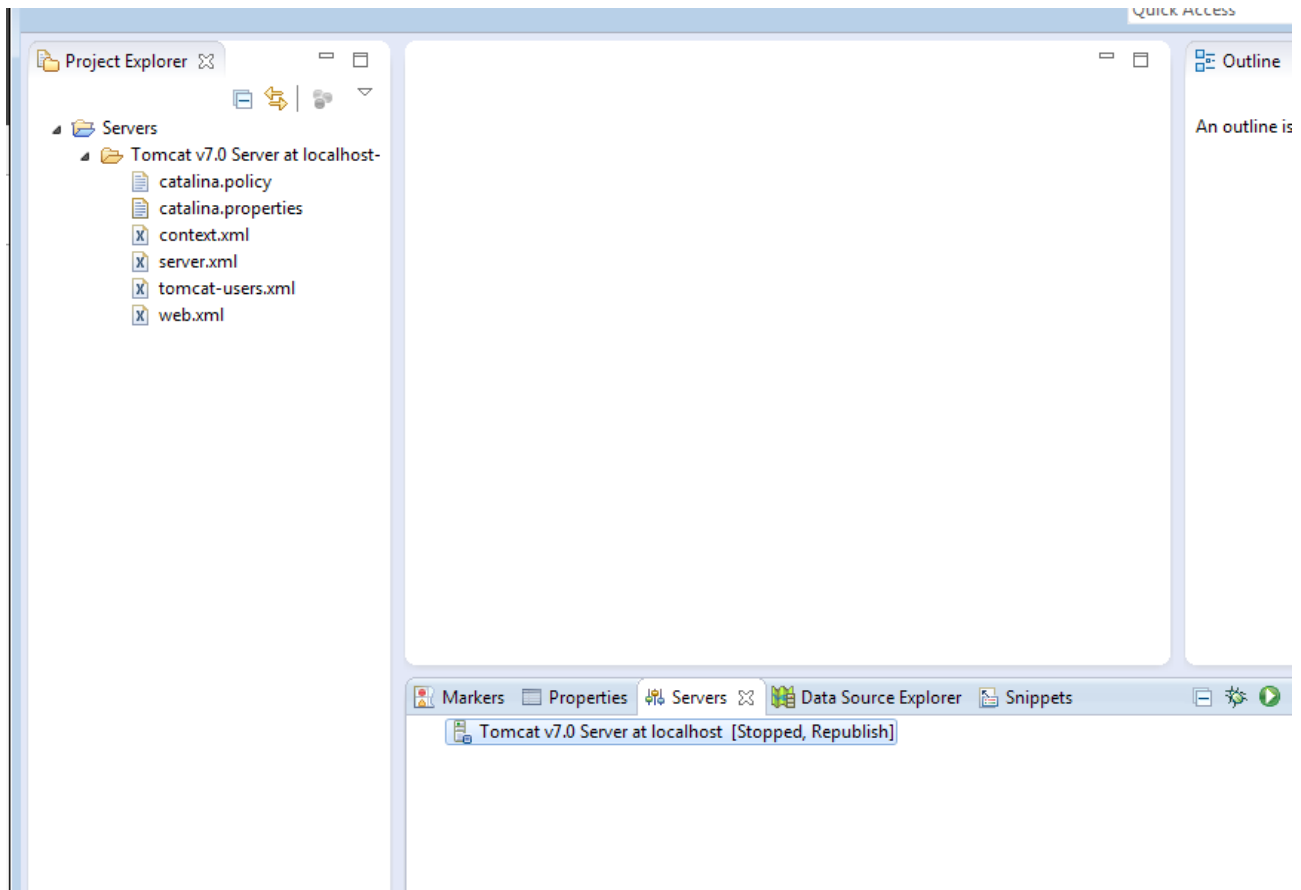
Name:

Tomcat installation directory:

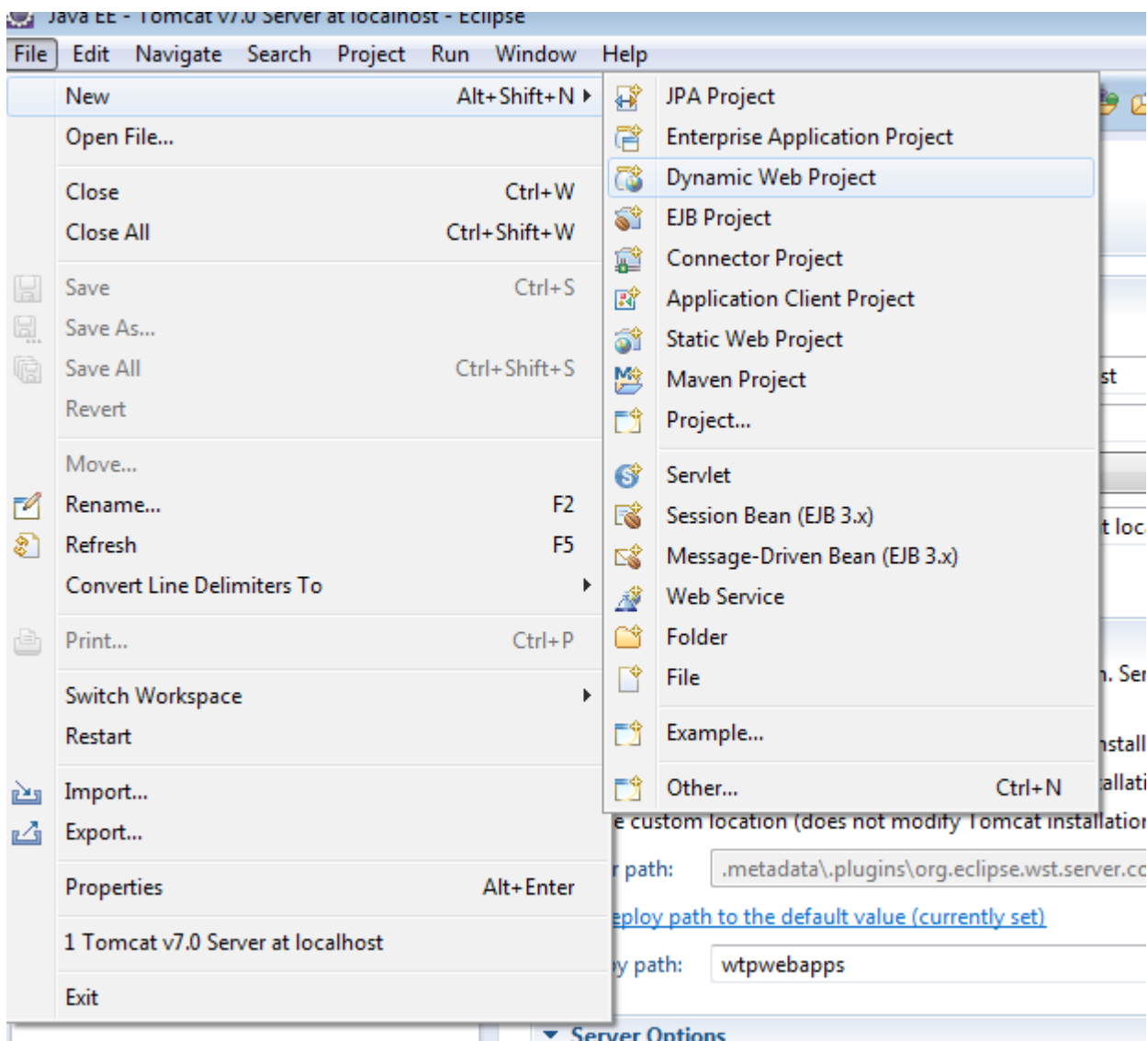
apache-tomcat-7.0.47

JRE:

configuraremos el nombre del servidor y indicaremos la ruta a nuestro apache Tomcat anteriormente instalado (ya que lo único que hará esto es enlazar nuestro servidor con nuestro IDE) y clicamos finalizar. Veremos ahora en el bloque de servers nuestro servidor



Ahora crearemos un nuevo proyecto web y lo enlazaremos en este servidor para ver como se despliega. Para ello vamos al apartado File → New → Dynamic Web Project



Le ponemos un nombre al proyecto (en nuestro caso PrimerProyecto) y le damos Siguiente. Las otras pantallas son de configuración pero las dejaremos por defecto. En la última pantalla nos da la opción de crear un fichero descriptor. De momento no seleccionaremos esta opción ya que es un proyecto de prueba para ver como vincularlo con el servidor integrado.

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:

Project location

☒ Use default location

Location:

[Browse...](#)

Target runtime

[New Runtime...](#)

Dynamic web module version

Configuration

[Modify...](#)

A good starting point for working with Apache Tomcat v7.0 runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

☐ Add project to an EAR

EAR project name:

[New Project...](#)

Working sets

☐ Add project to working sets

Working sets:

[Select...](#)



[< Back](#)

[Next >](#)

[Finish](#)

[Cancel](#)

Java

Configure project for building a Java application.



Source folders on build path:

 src

Add Folder...

Edit...

Remove

Default output folder:

build\classes



< Back

Next >

Finish

Cancel

Web Module

Configure web module settings.



Context root:

Content directory:

☐ Generate web.xml deployment descriptor



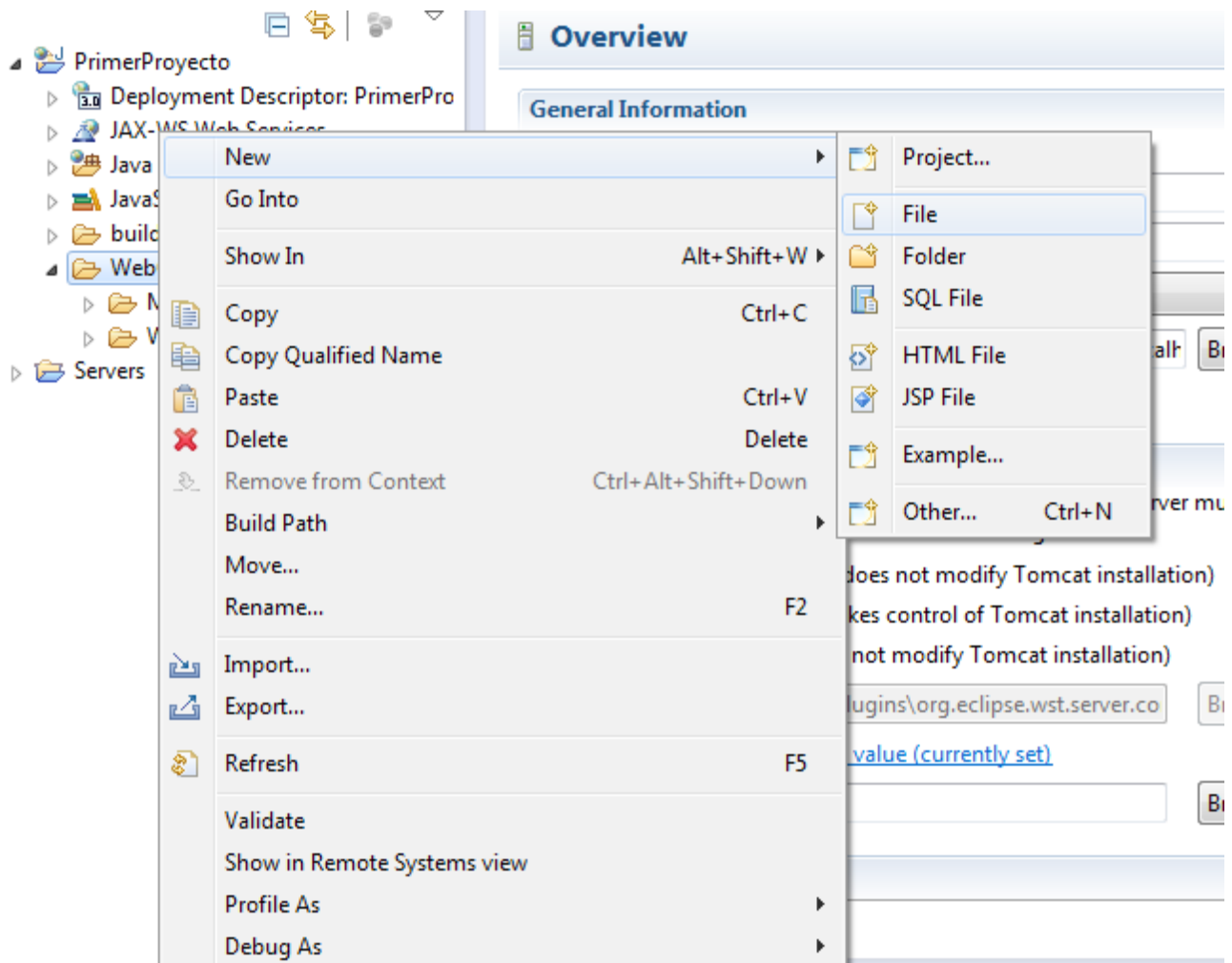
< Back

Next >

Finish

Cancel

vamos a crear un fichero 'index.html' para poder visualizar algo de contenido





New File



File

Create a new file resource.



Enter or select the parent folder:

PrimerProyecto/WebContent



PrimerProyecto



.settings



build



src



WebContent



RemoteSystemsTempFiles



Servers

File name:

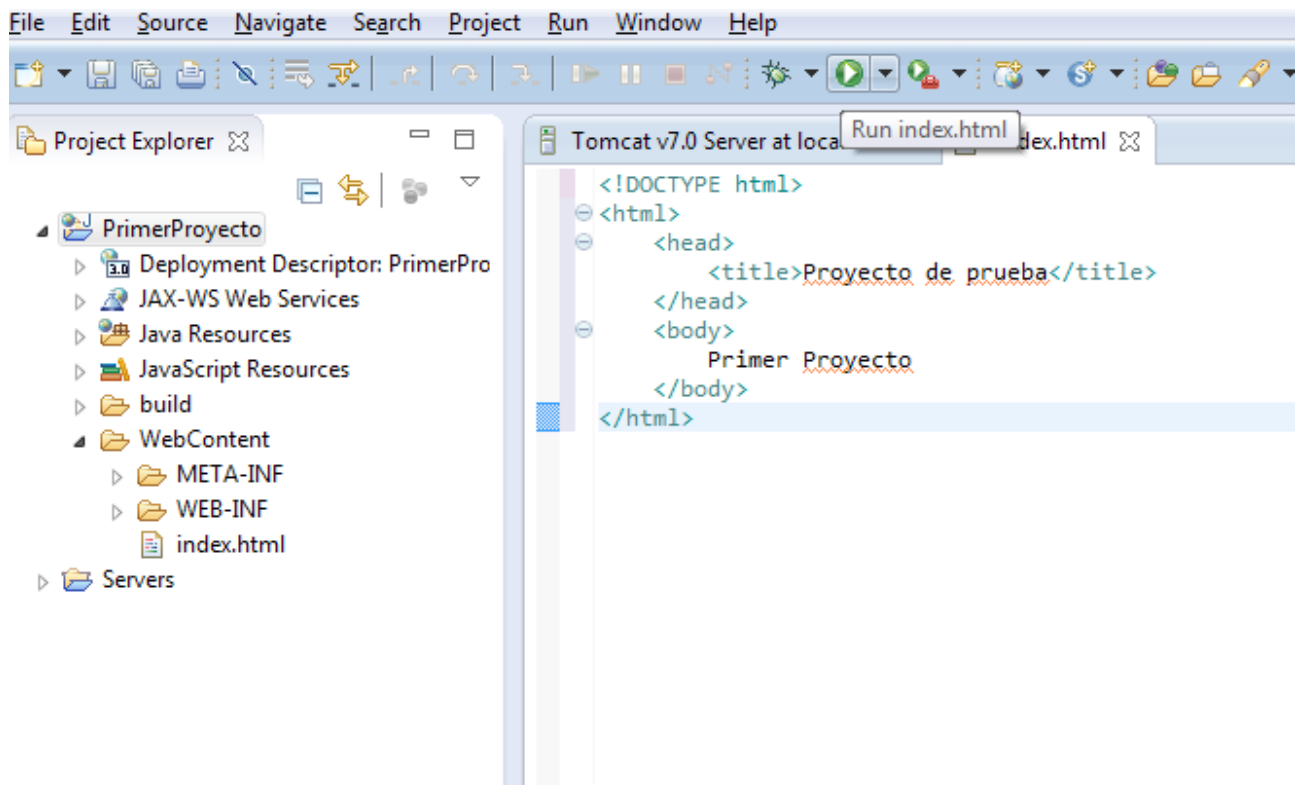
Advanced >>



Finish

Cancel

Una vez creado el fichero clicamos en el botón verde de play para ejecutar el proyecto en el servidor anteriormente creado.



Seleccionamos el servidor creado y le damos finalizar

Run On Server

Select which server to use



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:

type filter text

| Server | State |
|---------------------------------|---------|
| localhost | |
| Tomcat v7.0 Server at localhost | Stopped |

Apache Tomcat v7.0 supports J2EE 1.2, 1.3, 1.4, and Java EE 5 and 6 Web modules.

Columns...

☐ Always use this server when running this project

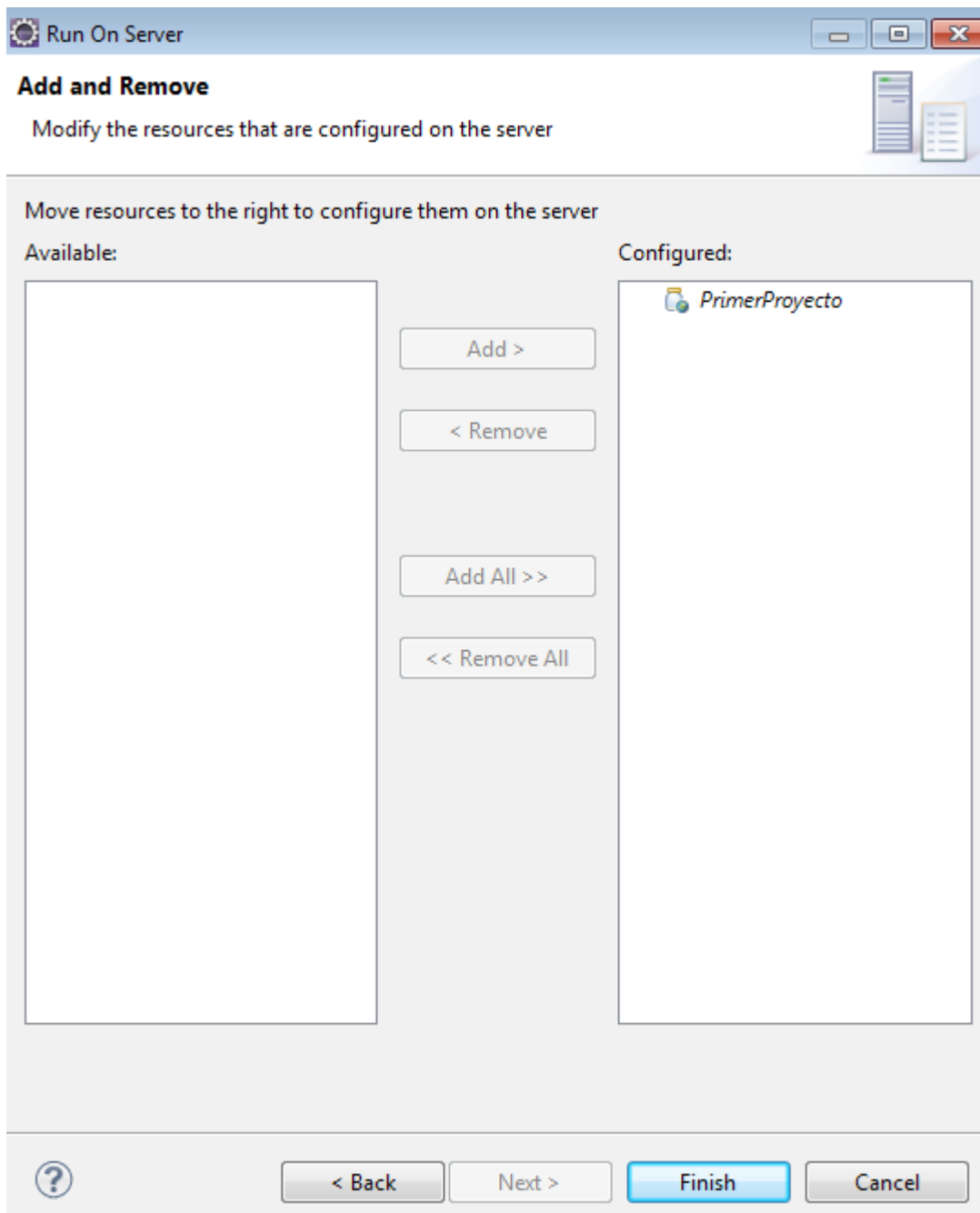


< Back

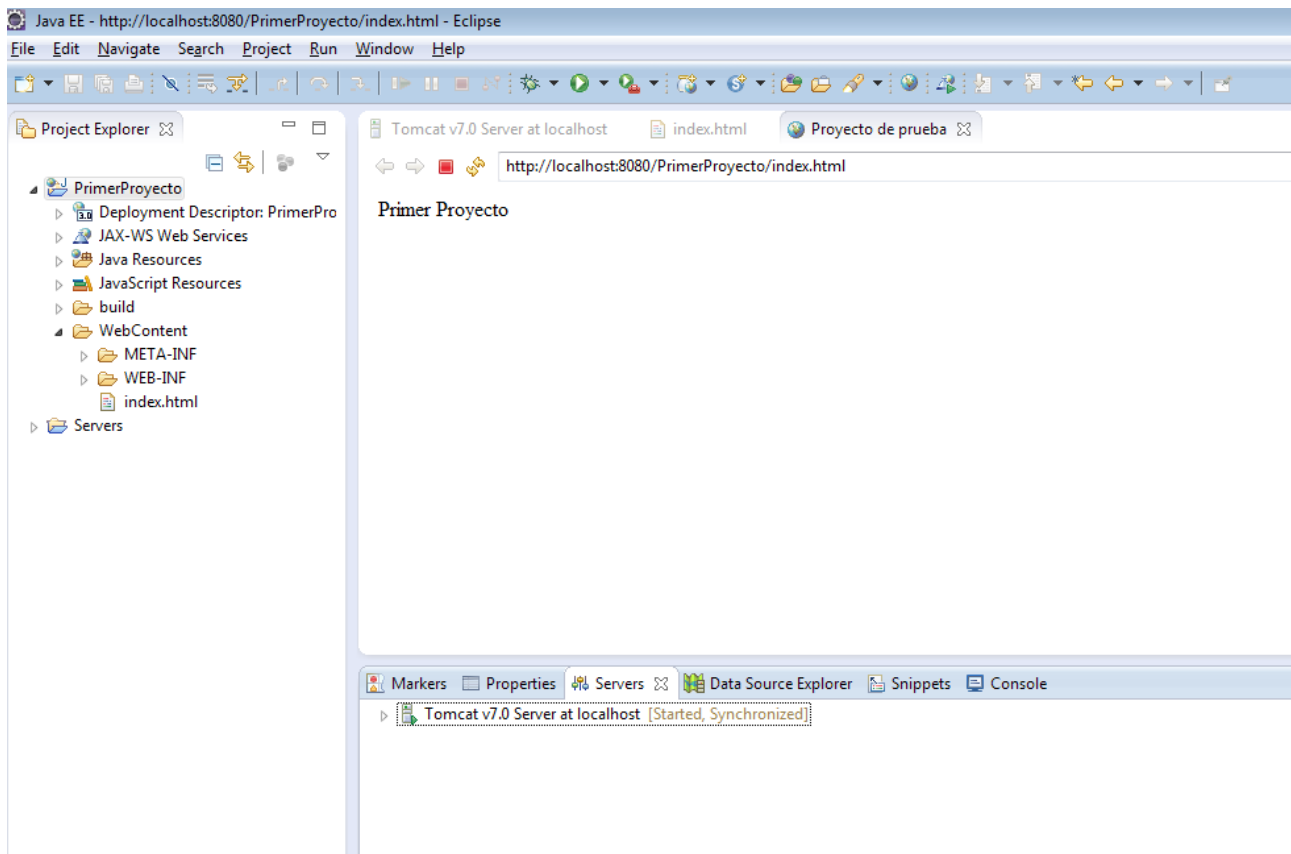
Next >

Finish

Cancel



Al acabar la vinculación del proyecto con el servidor nos ejecutara un navegador dentro de eclipse con la url del servidor creado que contiene ahora el proyecto.



2.3. Despliegue en Tomcat

Hemos visto como trabajar en local para la creación de servlet vinculando el proyecto con el servidor y eso nos ayuda a la hora del desarrollo pero no cuando queremos desplegar el proyecto en un servidor. Eso es lo que vamos a ver.

Primero de todo vamos a crear otro proyecto de prueba y esta vez seleccionaremos que genere el fichero web.xml

Web Module

Configure web module settings.



Context root: HolaMundo

Content directory: WebContent

☒ Generate web.xml deployment descriptor



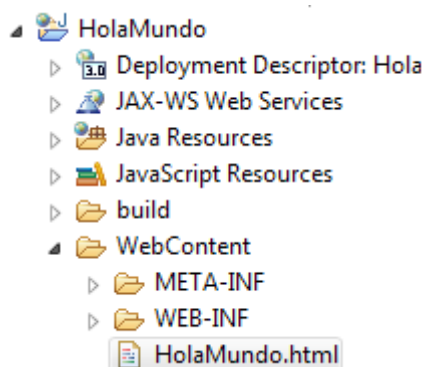
< Back

Next >

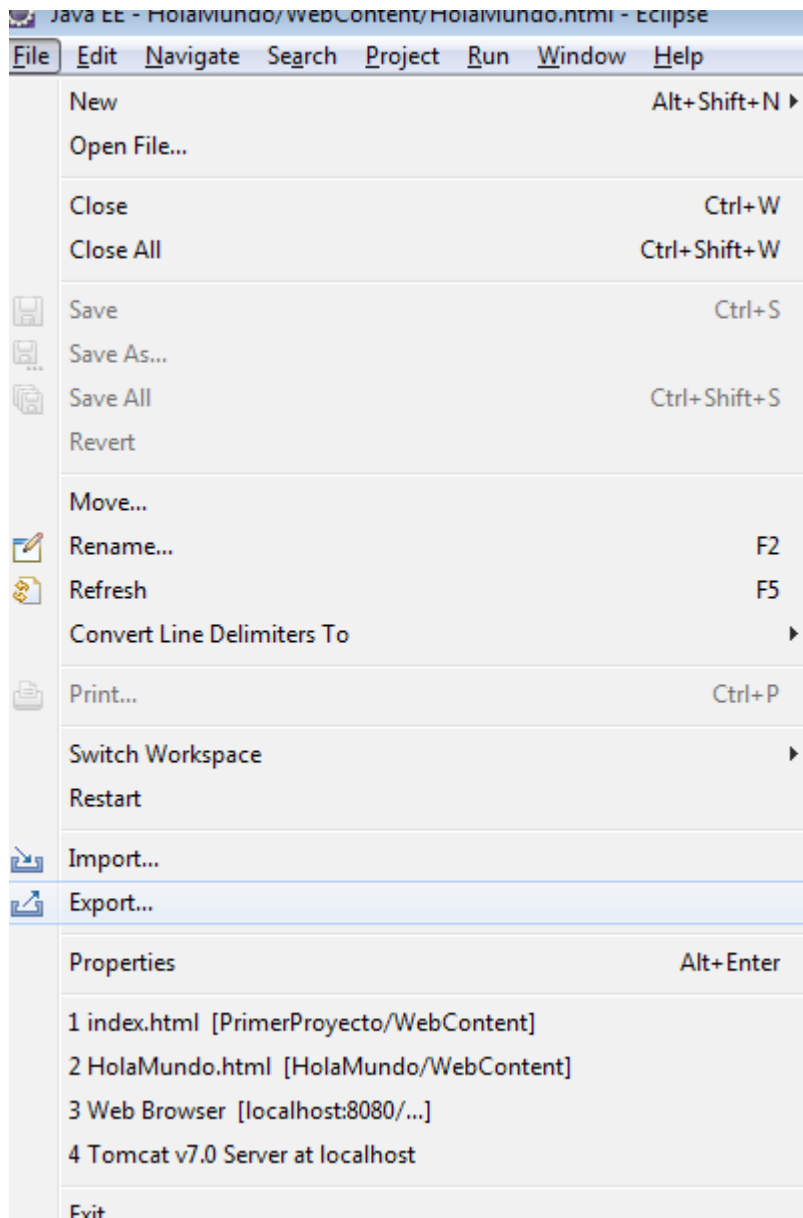
Finish

Cancel

Creamos un fichero html



y exportamos el proyecto para generar un 'war' (fichero compilado del proyecto)



Select

Export a Web Module into an external WAR file



Select an export destination:

- ▶ General
- ▶ EJB
- ▶ Install
- ▶ Java
- ▶ Java EE
- ▶ Plug-in Development
- ▶ Remote Systems
- ▶ Run/Debug
- ▶ Tasks
- ▶ Team
- ▶ Web
 - WAR file
- ▶ Web Services
- ▶ XML



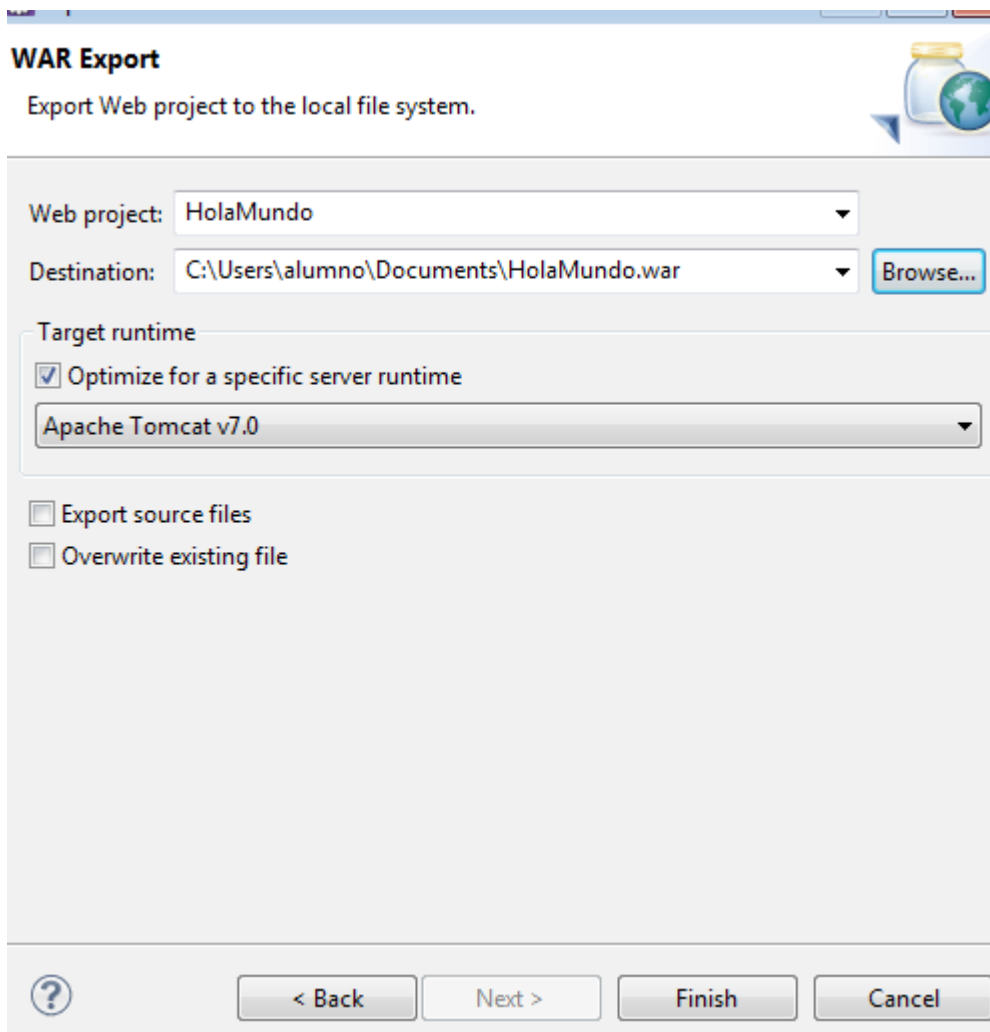
< Back

Next >

Finish

Cancel

Y seleccionamos donde guardar el fichero, ahora mismo nos da igual porque lo moveremos a la carpeta que corresponda



The image shows a 'WAR Export' dialog box from an IDE. It has a title bar and a header section with the title 'WAR Export' and a subtitle 'Export Web project to the local file system.' To the right of the subtitle is a small icon of a jar with a globe. The main area contains several fields: 'Web project:' with a dropdown menu showing 'HolaMundo'; 'Destination:' with a text field showing 'C:\Users\alumno\Documents\HolaMundo.war' and a 'Browse...' button to its right; a 'Target runtime' section with a checked checkbox 'Optimize for a specific server runtime' and a dropdown menu showing 'Apache Tomcat v7.0'; and two unchecked checkboxes at the bottom: 'Export source files' and 'Overwrite existing file'. At the very bottom are four buttons: a help button with a question mark, '< Back', 'Next >', and 'Finish', followed by a 'Cancel' button.

WAR Export
Export Web project to the local file system.

Web project: HolaMundo

Destination: C:\Users\alumno\Documents\HolaMundo.war [Browse...](#)

Target runtime

☒ Optimize for a specific server runtime

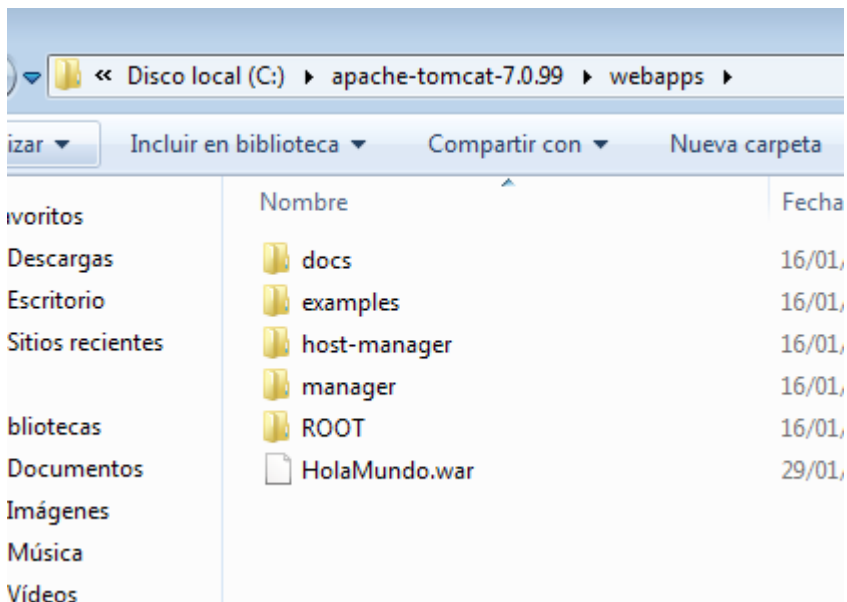
Apache Tomcat v7.0

☐ Export source files

☐ Overwrite existing file

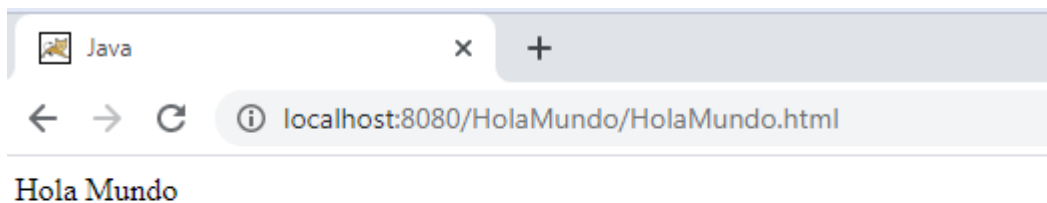
[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

El servidor Tomcat, como todos los servidores, tiene una carpeta publica la cual es la que muestra al exterior. En Tomcat esa carpeta se llama webapps y sera donde dejemos el fichero compilado que hemos exportado antes

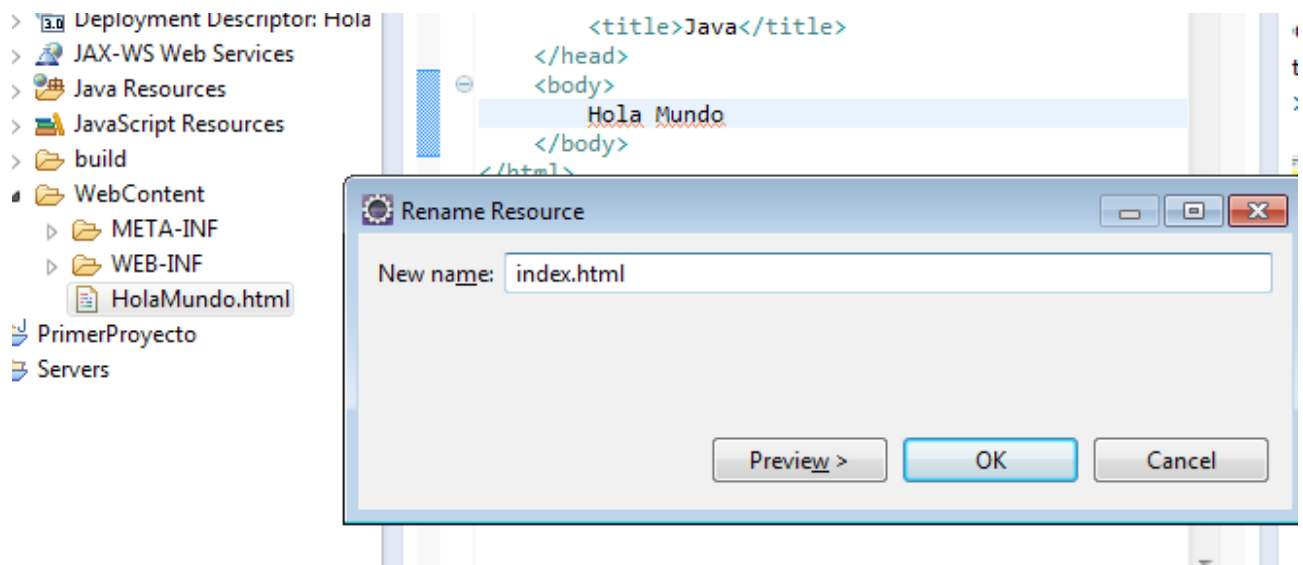


Ejecutamos el servidor Tomcat y accedemos a la url
[‘http://localhost:8080/HolaMundo/HolaMundo.html’](http://localhost:8080/HolaMundo/HolaMundo.html)

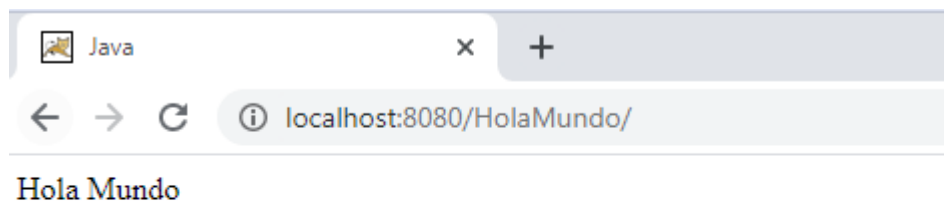
```
C:\Users\alumno>%CATALINA_HOME%\bin\startup.bat
Using CATALINA_BASE:   "C:\apache-tomcat-7.0.99"
Using CATALINA_HOME:   "C:\apache-tomcat-7.0.99"
Using CATALINA_TMPDIR: "C:\apache-tomcat-7.0.99\temp"
Using JRE_HOME:        "C:\Program Files\Java\jdk1.8.0_241"
Using CLASSPATH:       "C:\apache-tomcat-7.0.99\bin\bootstrap.jar;C:\apache-tomcat-7.0.99\bin\tomcat-juli.jar"
C:\Users\alumno>
```



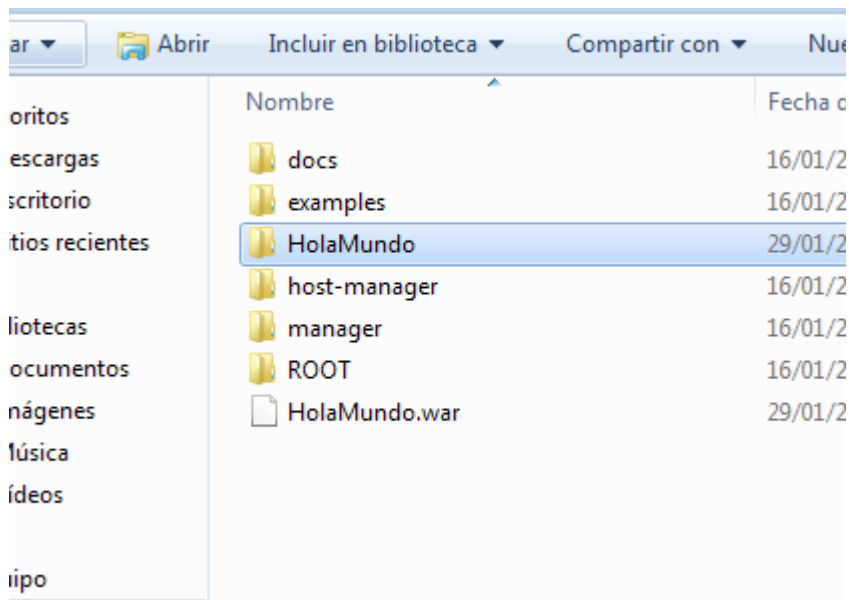
Vamos a modificar el nombre del fichero y en vez de HolaMundo.html vamos a poner index.html. Volvemos a exportar el proyecto como war y dejarlo en la carpeta webapps de tomcat



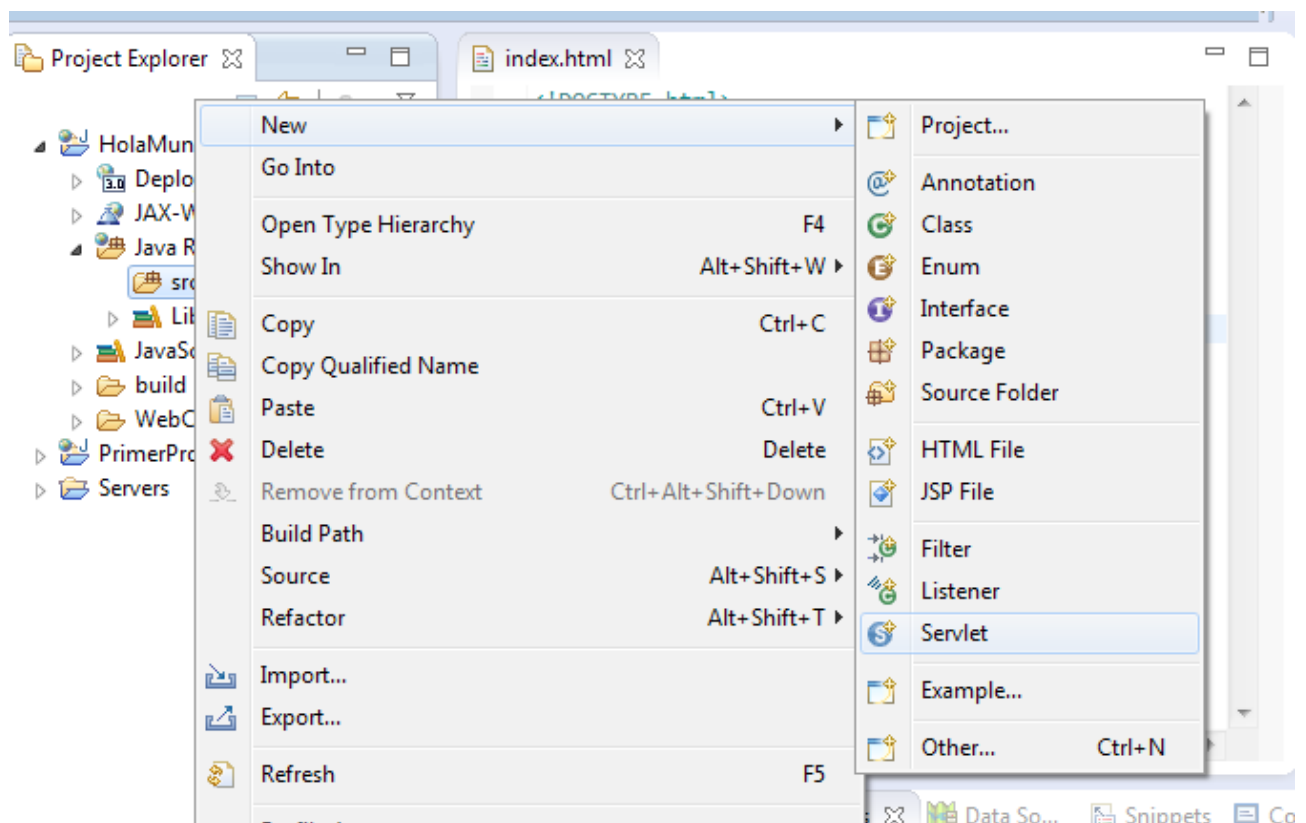
Ahora para acceder al proyecto basta con poner solo [‘http://localhost:8080/HolaMundo’](http://localhost:8080/HolaMundo) (Este ‘HolaMundo’ se debe al nombre del proyecto). Al ser index.html el fichero principal por defecto lo muestra sin necesidad de ponerlo en la url.



vemos que el servidor Tomcat nos a creado dentro de su carpeta webapps una carpeta HolaMundo, que lo que contiene es básicamente es el proyecto.



Para ver mejor el funcionamiento de un servlet vamos a añadir algo de lógica. Vamos a crear en el proyecto HolaMundo un servlet dentro de la carpeta Java Resources → SRC



Create Servlet

Specify class file destination.

Project: HolaMundo

Source folder: /HolaMundo/src Browse...

Java package: Browse...

Class name: HolaMundoServlet

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

☐ Use an existing Servlet class or JSP

Class name: HolaMundoServlet Browse...

? < Back Next > Finish Cancel

Create Servlet

Enter servlet deployment descriptor specific information.

Name: HolaMundoServlet

Description:

Initialization parameters:

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

Add... Edit... Remove

URL mappings:

/HolaMundoServlet

Add... Edit... Remove

? < Back Next > Finish Cancel

Create Servlet

Specify modifiers, interfaces to implement, and method stubs to generate.

Modifiers: ☒ public ☐ abstract ☐ final

Interfaces: Add... Remove

Which method stubs would you like to create?

☒ Constructors from superclass

☒ Inherited abstract methods

☐ init ☐ destroy ☐ getServletConfig

☐ getServletInfo ☐ service ☒ doGet

☒ doPost ☐ doPut ☐ doDelete

☐ doHead ☐ doOptions ☐ doTrace

? < Back Next > Finish Cancel

Después de esto nos habrá creado una clase java con algunos métodos creado predefinidos. Nosotros solo modificaremos el método doGet poniendo este código. Lo que nos hace es mostrar datos de la petición (URI, protocolo, remote address) y mostrar un número aleatorio.

```
/**
 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
 */
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    try {
        out.println("<html>");
        out.println("<head><title>Hola Mundo</title></head>");
        out.println("<body>");
        out.println("<h1>Hola Mundo</h1>");
        out.println("<p>Request URI: " + request.getRequestURI() + "</p>");
        out.println("<p>Protocolo: " + request.getProtocol() + "</p>");
        out.println("<p>Dirección remota: " + request.getRemoteAddr() + "</p>");
        out.println("<p>Número aleatorio: <strong>" + Math.random() + "</strong></p>");
        out.println("</body></html>");
    } finally {
        out.close();
    }
}
```


Después de esto modificaremos el fichero web.xml para indicar el servlet que acabamos de crear y ponerle una ruta de acceso, en este caso ‘/DiHola’.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schema
  <display-name>HolaMundo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>HolaMundo</servlet-name>
    <servlet-class>HolaMundoServlet</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>HolaMundo</servlet-name>
    <url-pattern>/DiHola</url-pattern>
  </servlet-mapping>
</web-app>
```

Para acceder a nuestro servlet ponemos la url
‘<http://localhost:8080/HolaMundo/DiHola>’ (HolaMundo por el nombre del proyecto y DiHola por la ruta que hemos puesto)

 localhost:8080/HolaMundo/DiHola

Hola Mundo

Request URI: /HolaMundo/DiHola

Protocolo: HTTP/1.1

Dirección remota: 0:0:0:0:0:0:1

Número aleatorio: 0.27376751286817946

2.4. Servlets y base de datos

2.4.1 Instalación de mysql y conectores

Para poder utilizar los servlets junto a las bases de datos primero tenemos que tener instalada una base de datos (en este caso mysql, se puede descargar desde aquí '<http://dev.mysql.com/download/installer>') y su conector. También descargaremos el conector Microsoft .NET Framework 4 (necesario para mysql) desde '<http://www.microsoft.com/en-us/download/details.aspx?id=42642>' (Esta url puede cambiar dependiendo de la versión del **mysql connector** y es importante descargar la versión que se requiera).

Primero instalamos .Net Framework 4. Es una instalación fácil con las típicas pantallas de asistencia.

Descargamos el instalador de mysql


MySQL Installer 8.0.19

Select Operating System:

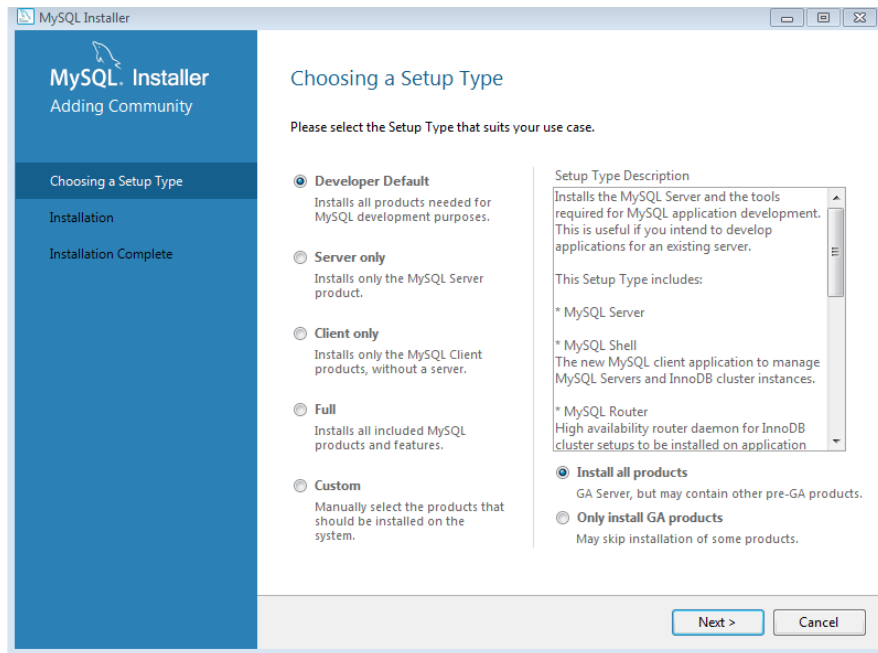
Microsoft Windows ▼

Looking for previous GA versions?

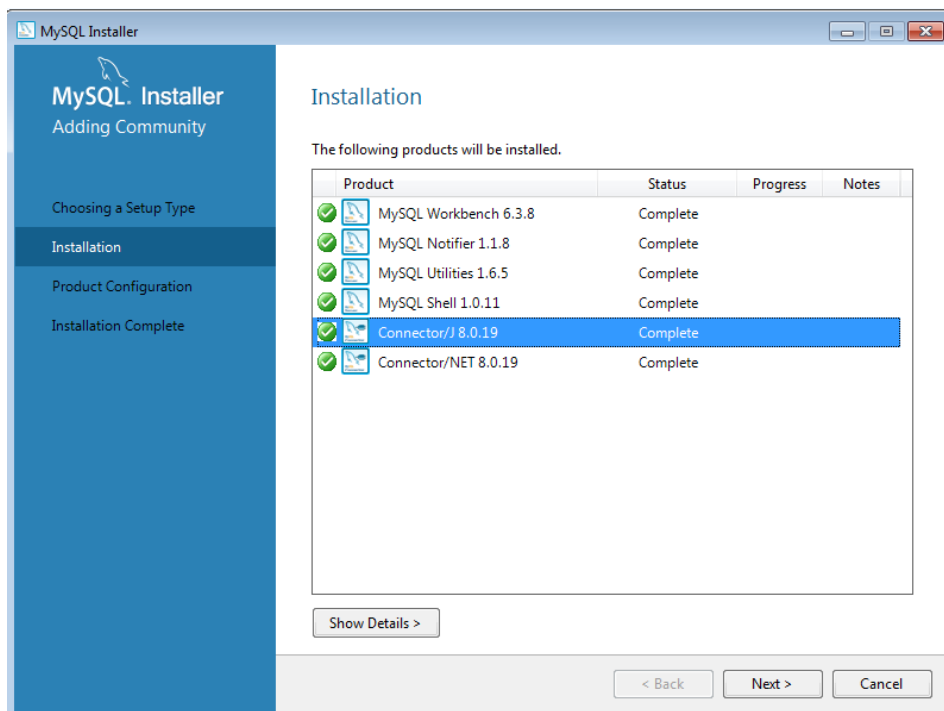
| | | | |
|---|--------|--------|--------------------------|
| Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.19.0.msi) | 8.0.19 | 18.6M | Download |
| MD5: 32043776cb2239db45fddaa86dc0ad61 Signature | | | |
| Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.19.0.msi) | 8.0.19 | 398.9M | Download |
| MD5: 1a882015da7fb93f20c4717e63b6817c Signature | | | |

 We suggest that you use the [MD5 checksums and GnuPG signatures](#) to verify the integrity of the packages you download.

Elegimos la opción por defecto

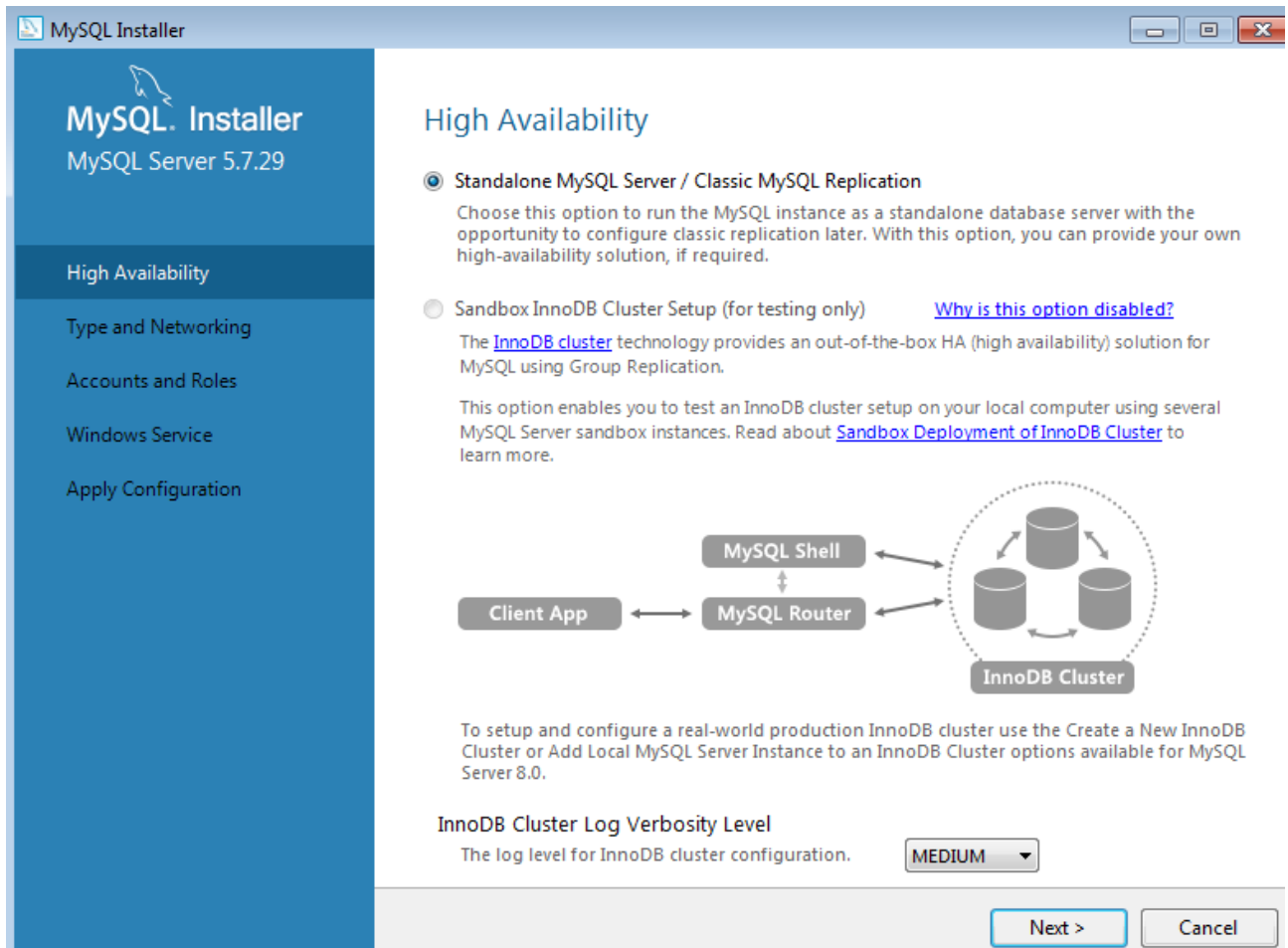


Al pasar a la siguiente pantalla se descargará los diferentes drivers y controladores. (si antes de esta pantalla saliera otra pantalla de instalación de drivers y da error no pasa nada, clicar siguiente y llegareis a este punto).



Una vez instalado mysql y el conector de java vamos a configurar mysql server.

Seleccionamos la primera opción.



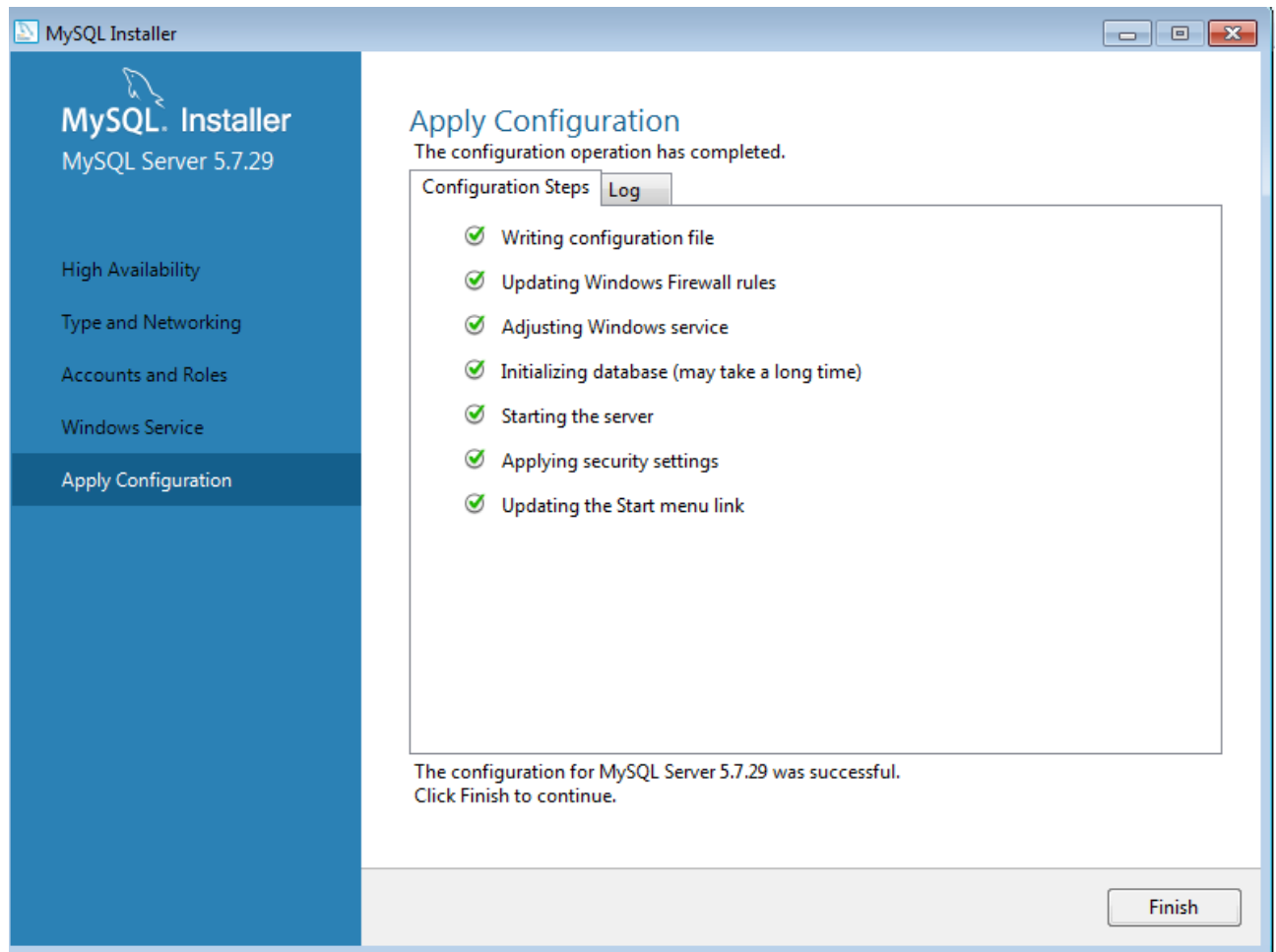
Elegimos el tipo de configuración desarrollo, el puerto de mysql, usuario y contraseña. Son pantallas sencillas del típico asistente con siguiente, siguiente...

The screenshot shows the 'MySQL Installer' window for 'MySQL Server 5.7.29'. The left sidebar contains a list of configuration steps: 'High Availability', 'Type and Networking' (which is the active step), 'Accounts and Roles', 'Windows Service', and 'Apply Configuration'. The main area is titled 'Type and Networking' and contains the following sections:

- Server Configuration Type**: A text block explaining that this setting defines system resources. Below it, a dropdown menu for 'Config Type' is set to 'Development Computer'.
- Connectivity**: A text block asking how to connect to the server. It includes three checked options: 'TCP/IP' (with a 'Port' field set to '3306'), 'Open Windows Firewall port for network access', and 'Named Pipe' (with a 'Pipe Name' field set to 'MYSQL'). There is also an unchecked 'Shared Memory' option with a 'Memory Name' field set to 'MYSQL'.
- Advanced Configuration**: A text block explaining that additional configuration pages are available. Below it, an unchecked checkbox is labeled 'Show Advanced and Logging Options'.

At the bottom right of the window, there are three buttons: '< Back', 'Next >' (which is highlighted with a blue border), and 'Cancel'.

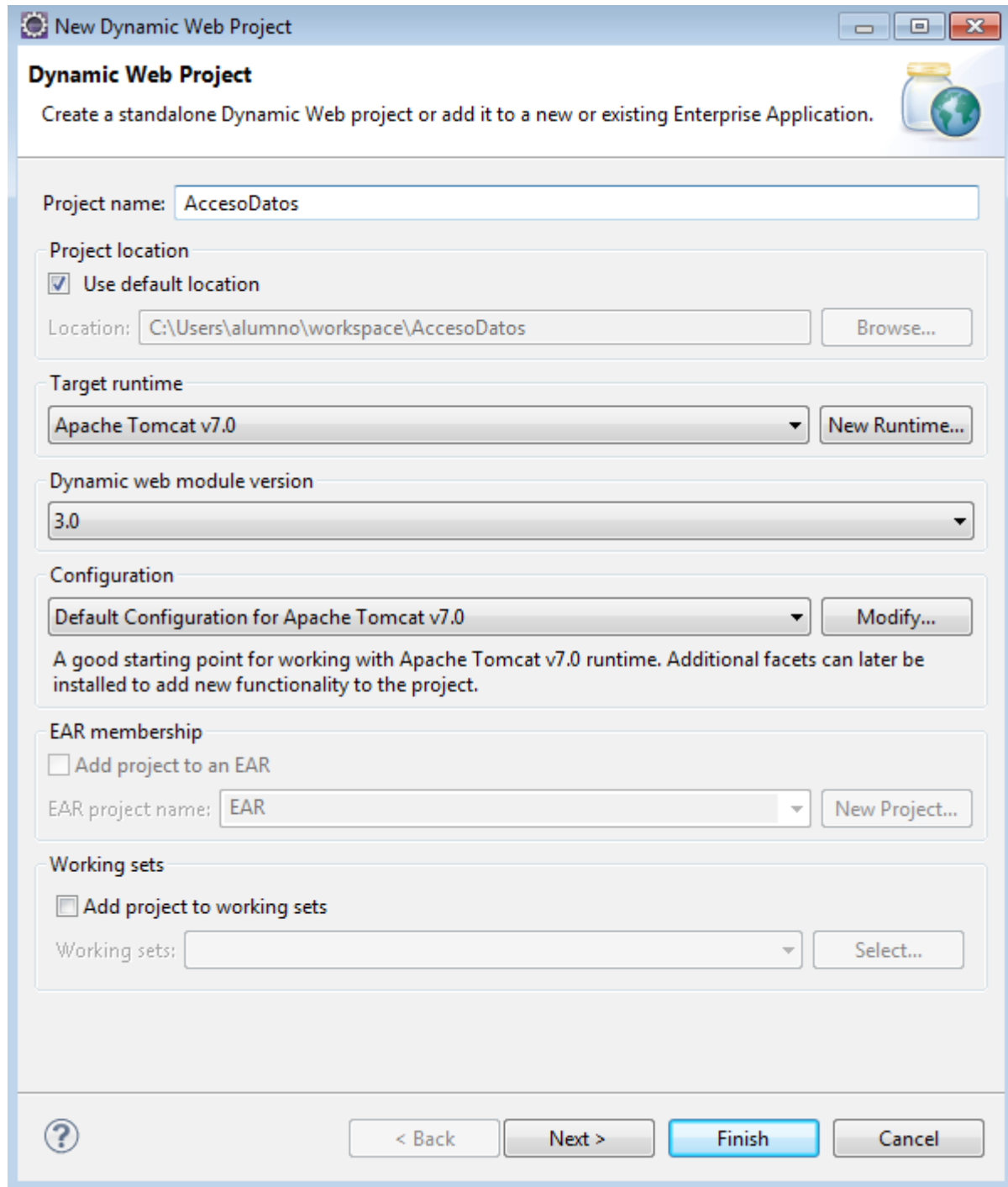
Al final nos sale una pantalla resumen para verificar que todo este bien configurado



Una vez hecho todas las instalaciones necesarias creamos una base de datos llamada TiendaLibros con una tabla llamada libros con tres columna en esta tabla que se llamen autor, titulo y precio.

2.4.2. Creación de un Servlet con acceso a base de datos

Una vez instalado todo abrimos eclipse y creamos un **dynamic web project** llamado AccesoDatos y seleccionamos que nos genere el fichero web.xml.



The screenshot shows the 'New Dynamic Web Project' wizard in Eclipse. The project name is 'AccesoDatos'. The location is set to 'C:\Users\alumno\workspace\AccesoDatos'. The target runtime is 'Apache Tomcat v7.0'. The dynamic web module version is '3.0'. The configuration is 'Default Configuration for Apache Tomcat v7.0'. The EAR membership is 'Add project to an EAR'. The working sets are 'Add project to working sets'. The 'Finish' button is highlighted.

New Dynamic Web Project

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

Project name:

Project location

☒ Use default location

Location:

Target runtime

Dynamic web module version

Configuration

A good starting point for working with Apache Tomcat v7.0 runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

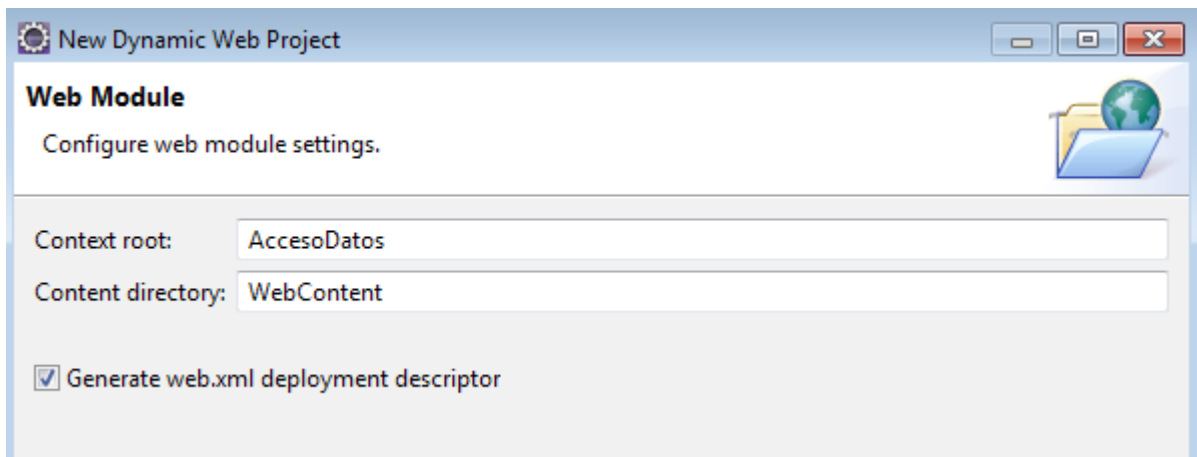
☐ Add project to an EAR

EAR project name:

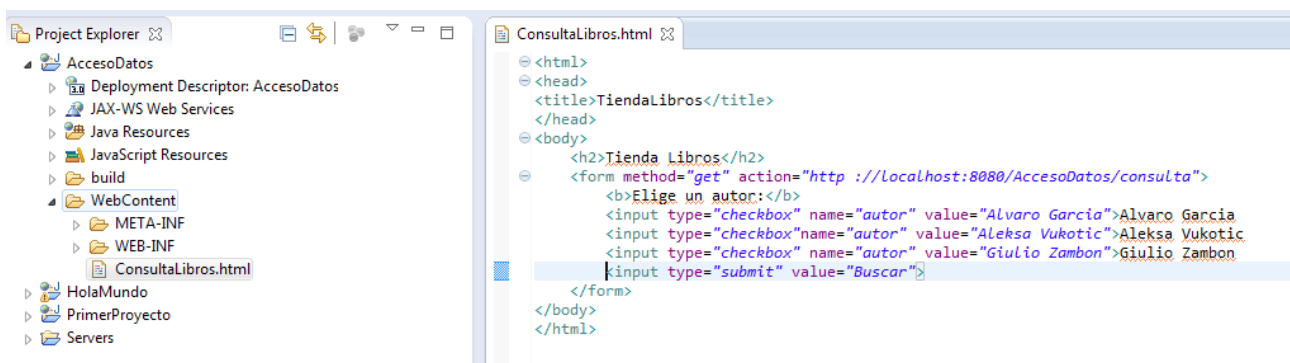
Working sets

☐ Add project to working sets

Working sets:



Creamos un fichero llamado ConsultaLibros.html dentro de la carpeta WebContent y creamos este formulario.



Creamos el servlet y ponemos en el doGet el siguiente código.

(es importante poner la zona horaria en la conexión con la base de datos).

```
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    Connection conn = null;
    Statement stmt = null;
    try {
        Class.forName("com.mysql.cj.jdbc.Driver").newInstance();
        String userName = "root";
        String password = "root";
        String url = "jdbc:mysql://localhost/TiendaLibros?serverTimezone=UTC";
        conn = DriverManager.getConnection(url, userName, password);

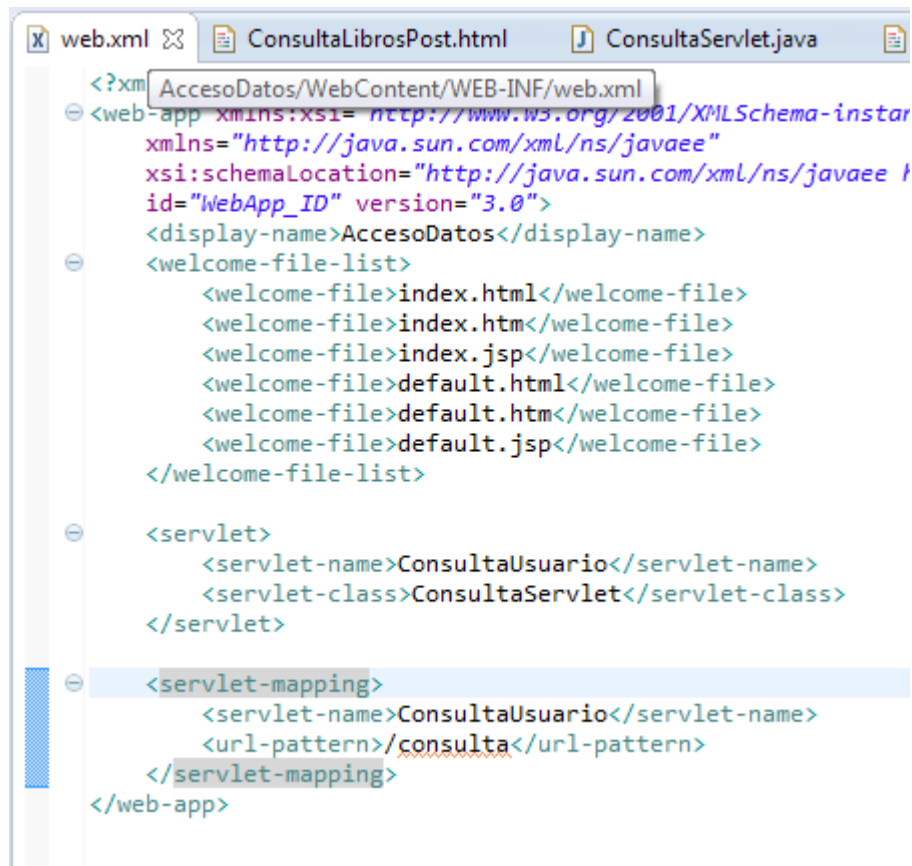
        stmt = conn.createStatement();
        String sqlStr = "SELECT * FROM libros WHERE autor='" + request.getParameter("autor") + "'";
        ResultSet rset = stmt.executeQuery(sqlStr);

        out.println("<html><head><title>Resultado de la consulta</title></head><body>" );
        out.println("<h3>Gracias por tu consulta.</h3>");
        out.println("<p>Tu consulta es: " + sqlStr + "</p>");

        int count = 0;
        while(rset.next()) {
            count++;
            out.println("<p>" + rset.getString("autor") + " " + rset.getString("titulo") + " " + rset.getDouble("precio") +
                out.println("<p>====" + count + " registros encontrados =====</p>");
            out.println("</body> </html>");
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    } finally {
        out.close();
    }

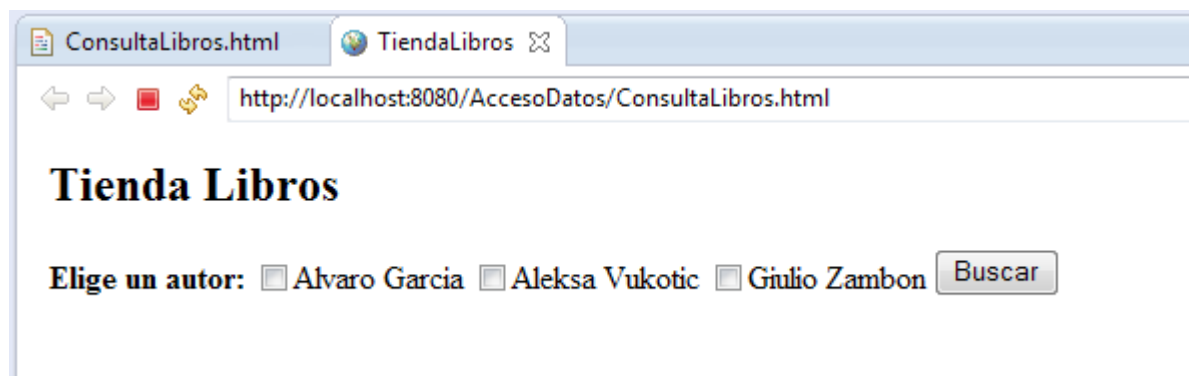
    try {
        if (stmt != null) stmt.close();
        if (conn != null) conn.close();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
}
```

Editamos el fichero web.xml para indicar la ruta

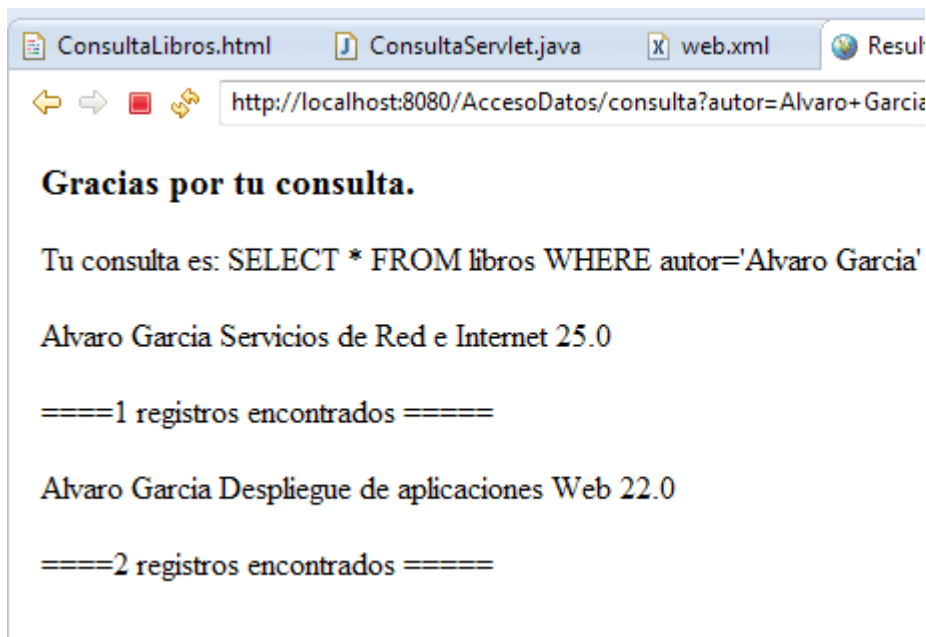


```
<?xml xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
id="WebApp_ID" version="3.0">
  <display-name>AccesoDatos</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
  <servlet>
    <servlet-name>ConsultaUsuario</servlet-name>
    <servlet-class>ConsultaServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>ConsultaUsuario</servlet-name>
    <url-pattern>/consulta</url-pattern>
  </servlet-mapping>
</web-app>
```

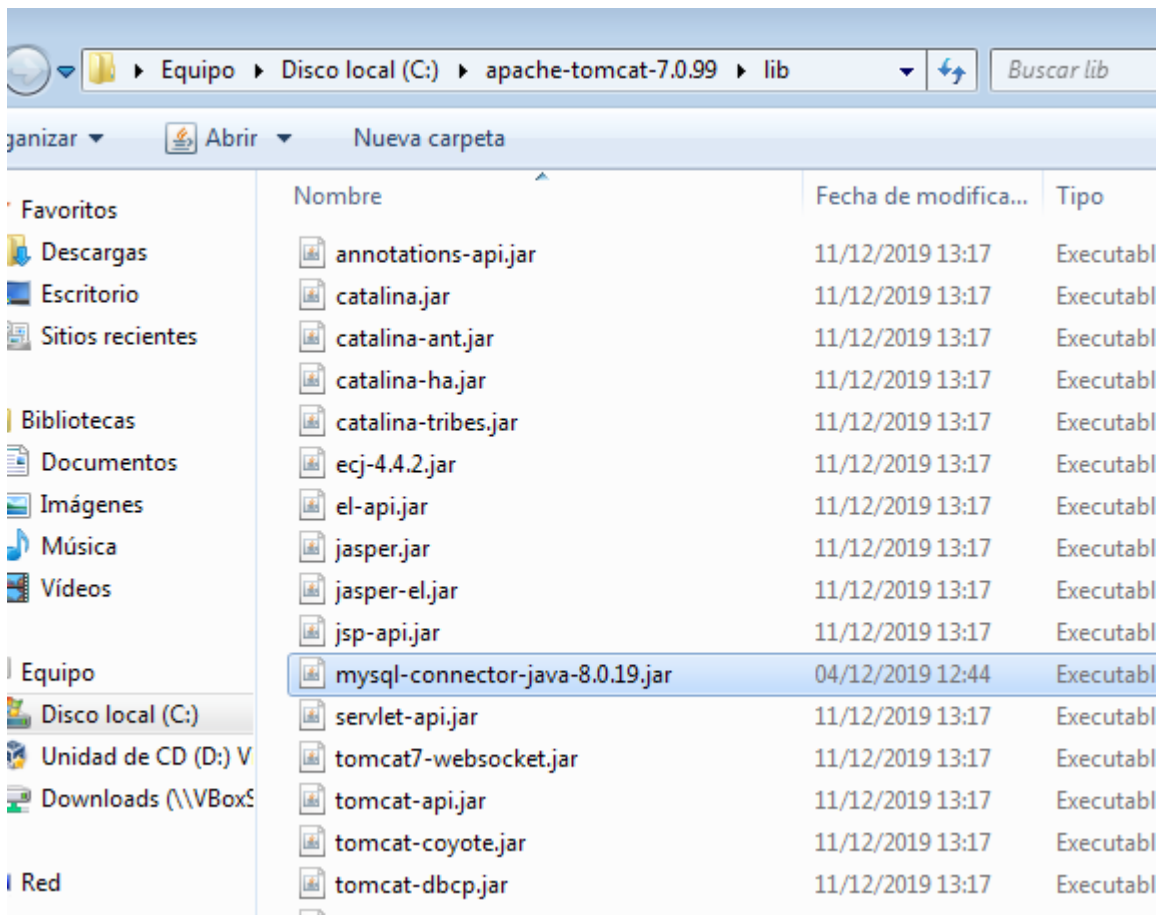
Ejecutamos el servlet en el servidor de eclipse (hay que recordar que la url de acceso es accediendo a AccesoDatos/ConsultaLibros.html)



Elegimos un autor y le damos buscar



Para ejecutarlo en el servidor Tomcat tenemos que exportar el proyecto a war y lo dejamos en la carpeta webapps de Tomcat como hemos hecho con anterioridad. (Si queremos ejecutar el proyecto en el servidor Tomcat tenemos que poner el conector de mysql en este servidor. Para ello vamos al fichero jar del connector y lo copiamos dentro de la carpeta lib que hay en Tomcat).



3. Tomcat en Linux

3.1 Instalación de Java en Linux

Para instalar Tomcat es necesario primero instalar java por tanto antes de nada actualizamos los repositorios de linux.

```
alumno@ServidorLinux20:~$ sudo apt update_
```

depués instalamos la versión libre de java para linux llamada openjdk

```
Des:10 http://security.ubuntu.com trusty-security/restricted Sources [4.931 B]  
Des:11 http://security.ubuntu.com trusty-security/universe Sources [102 kB]  
Descargados 7.299 kB en 12s (584 kB/s)  
Leyendo lista de paquetes... Hecho  
alumno@ServidorLinux20:~$ sudo apt install openjdk-7-jdk
```

teniendo ya instalado java creamos las variables de entorno necesarias llamadas JAVA_HOME y JRE_HOME apuntando al directorio donde se han instalado

```
alumno@ServidorLinux20:~$ sudo nano /etc/environment
```

```
JAVA_HOME=/usr/lib/jvm/java-7-openjdk-i386  
JRE_HOME=/usr/lib/jvm/java-7-openjdk-i386/jre_  
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games"
```

para que las variables estén accesibles en el sistema hay que reiniciar el sistema operativo

```
alumno@ServidorLinux20:~$ sudo reboot
```

3.2 Instalación de Tomcat en Linux

Con todo esto ya tenemos java y sus variables de entorno funcionales, ahora toca el turno de instalar Tomcat

```
alumno@ServidorLinux20:~$ sudo apt install tomcat7_
```

Para el buen funcionamiento de Tomcat tenemos que crear sus variables de entorno

```
alumno@ServidorLinux20:~$ sudo nano /etc/environment
```

```
CATALINA_HOME=/usr/share/tomcat7  
CATALINA_BASE=/var/lib/tomcat7  
JAVA_HOME=/usr/lib/jvm/java-7-openjdk-i386  
JRE_HOME=/usr/lib/jvm/java-7-openjdk-i386/jre  
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games"
```

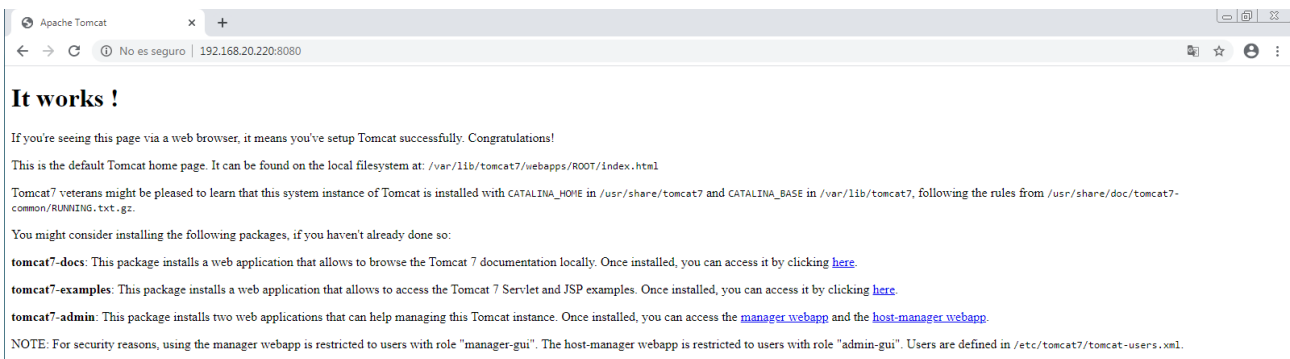
Al crear las variables de entorno nuevas tenemos que volver a reiniciar el sistema operativo

```
alumno@ServidorLinux20:~$ sudo reboot
```

Con el sistema ya iniciado comprobamos que el servidor Tomcat este corriendo en el puerto 8080 (por defecto)

```
alumno@ServidorLinux20:~$ ps -ef | grep tomcat
tomcat7 1281 1 8 15:04 ? 00:00:03 /usr/lib/jvm/java-7-openjdk-amd64/bin/java -Djava.util.logging.config.file=/var/lib/tomcat7/conf/logging.properties -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djava.awt.headless=true -Xmx128m -XX:+UseConcMarkSweepGC -Djava.endorsed.dirs=/usr/share/tomcat7/endorsed -classpath /usr/share/tomcat7/bin/bootstrap.jar:/usr/share/tomcat7/bin/tomcat-juli.jar -Dcatalina.base=/var/lib/tomcat7 -Dcatalina.home=/usr/share/tomcat7 -Djava.io.tmpdir=/tmp/tomcat7-tomcat7-tmp org.apache.catalina.startup.Bootstrap start
alumno 1420 1404 0 15:04 tty1 00:00:00 grep --color=auto tomcat
alumno@ServidorLinux20:~$ netstat -ltn
Conexiones activas de Internet (solo servidores)
Proto Recib Envíad Dirección local Dirección remota Estado
tcp6 0 0 127.0.0.1:8005 :::* ESCUCHAR
tcp6 0 0 :::8080 :::* ESCUCHAR
tcp6 0 0 :::80 :::* ESCUCHAR
tcp6 0 0 :::443 :::* ESCUCHAR
alumno@ServidorLinux20:~$
```

Para comprobar que el servidor funciona abrimos otra maquina (Desarrollo Windows 7) y accedemos a la ip y el puerto del servidor Tomcat y nos debería salir algo así, una pagina por defecto indicando que funciona.



Viendo que el servidor Tomcat ya esta en funcionamiento vamos a proceder a instalar la documentación de Tomcat y unos ejemplos.

```
alumno@ServidorLinux20:~$ sudo apt install tomcat7-docs tomcat7-examples
```

Una vez instalado volvemos a la maquina windows y accedemos a la documentación que hemos instalado, podemos hacerlo desde la url entrando a /docs o en la página principal clicando en el enlace para ver la documentación

Apache Tomcat 7 (7.0.52) - Docs

192.168.20.220:8080/docs/

Apache Tomcat 7

Version 7.0.52, Oct 10 2018

The Apache Software Foundation
<http://www.apache.org/>

Documentation Index

Introduction

This is the top-level entry point of the documentation bundle for the **Apache Tomcat** Servlet/JSP container. Apache Tomcat version 7.0 implements the Servlet 3.0 and JavaServer Pages 2.2 specifications from the [Java Community Process](#), and includes many additional features that make it a useful platform for developing and deploying web applications and web services.

Select one of the links from the navigation menu (to the left) to drill down to the more detailed documentation that is available. Each available manual is described in more detail below.

Apache Tomcat User Guide

The following documents will assist you in downloading, installing Apache Tomcat 7, and using many of the Apache Tomcat features.

1. **Introduction** - A brief, high level, overview of Apache Tomcat.
2. **Setup** - How to install and run Apache Tomcat on a variety of platforms.
3. **First web application** - An introduction to the concepts of a *web application* as defined in the Servlet Specification. Covers basic organization of your web application source tree, the structure of a web application archive, and an introduction to the web application deployment descriptor (*/WEB-INF/web.xml*).
4. **Deployer** - Operating the Apache Tomcat Deployer to deploy, precompile, and validate web applications.
5. **Manager** - Operating the **Manager** web app to deploy, undeploy, and redeploy applications while Apache Tomcat is running.
6. **Realms and Access Control** - Description of how to configure *Realms* (databases of users, passwords, and their associated roles) for use in web applications that utilize *Container Managed Security*.
7. **Security Manager** - Configuring and using a Java Security Manager to support fine-grained control over the behavior of your web applications.
8. **JNDI Resources** - Configuring standard and custom resources in the JNDI naming context that is provided to each web application.
9. **JDBC DataSource** - Configuring a JNDI DataSource with a DB connection pool. Examples for many popular databases.
10. **Classloading** - Information about class loading in Apache Tomcat, including where to place your application classes so that they are visible.
11. **JSPs** - Information about Jasper configuration, as well as the JSP compiler usage.
12. **SSL** - Installing and configuring SSL support so that your Apache Tomcat will serve requests using the [https](#) protocol.
13. **SSI** - Using Server Side Includes in Apache Tomcat.
14. **CGI** - Using CGI with Apache Tomcat.

Probamos a ir también a los ejemplos instalados navegando a la url /examples o clicando en el enlace para ver los ejemplos

Apache Tomcat Examples

192.168.20.220:8080/examples/

Apache Tomcat Examples

- [Servlets examples](#)
- [JSP Examples](#)
- [WebSocket \(JSR356\) Examples](#)
- [WebSocket Examples using the deprecated Apache Tomcat proprietary API](#)

Vamos a entrar en la parte de ejemplos de servlets

Servlet Examples

192.168.20.220:8080/examples/servlets/

Servlet Examples with Code

This is a collection of examples which demonstrate some of the more frequently used parts of the Servlet API. Familiarity with the Java(tm) Programming Language is assumed.

These examples will only work when viewed via an http URL. They will not work if you are viewing these pages via a "file://..." URL. Please refer to the *README* file provide with this Tomcat release regarding how to configure and start the provided web server.

Wherever you see a form, enter some data and see how the servlet reacts. When playing with the Cookie and Session Examples, jump back to the Headers Example to see exactly what your browser is sending the server.

To navigate your way through the examples, the following icons will help:

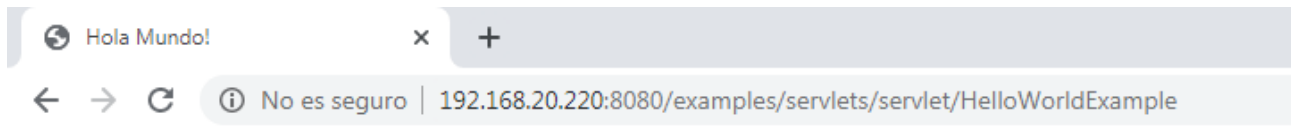
- Execute the example
- Look at the source code for the example
- Return to this screen

Tip: To see the cookie interactions with your browser, try turning on the "notify when setting a cookie" option in your browser preferences. This will let you see when a session is created and give some feedback when looking at the cookie demo.

| | | |
|--------------------|-------------------------|------------------------|
| Hello World | Execute | Source |
| Request Info | Execute | Source |
| Request Headers | Execute | Source |
| Request Parameters | Execute | Source |
| Cookies | Execute | Source |
| Sessions | Execute | Source |

Note: The source code for these examples does not contain all of the source code that is actually in the example, only the important sections of code. Code not important to understand the example has been removed for clarity.

Y clicamos en ejecutar el ejemplo de HolaMundo (HelloWorld)



Hola Mundo!

3.3 Estructura de Tomcat

Viendo que ya tenemos instalado el servidor Tomcat y sus ejemplos y documentación y todo funciona bien y es accesible desde otro ordenador. Vamos ahora a analizar la estructura de Tomcat.

Por un lado tenemos el directorio en el cual se ha instalado Tomcat siendo /usr/share/tomcat aquí se encuentran los binarios necesarios para el programa

```
alumno@ServidorLinux20:~$ ll /usr/share/tomcat?_
```

Por otro lado tenemos la carpeta pública de Tomcat en la que hay algunos ficheros de configuración y la parte publica para mostrar las diferentes páginas web. Esta parte de Tomcat se encuentra en /var/lib/tomcat7. Vemos que la carpeta de configuración es un enlace simbólico a el directorio /etc/tomcat7 en el cual dentro hay un fichero server.xml donde nos indica la configuración del servidor (como el puerto, redirecciones, servicios...).

```
alumno@ServidorLinux20:~$ sudo cat /var/lib/tomcat7/conf/server.xml
```

Y la carpeta pública es webapps

```
alumno@ServidorLinux20:~$ ll /var/lib/tomcat7/
total 28
drwxr-xr-x  7 root    root    4096 feb 13 15:04 ./
drwxr-xr-x 47 root    root    4096 feb 13 14:59 ../
drwxr-xr-x  3 tomcat7 tomcat7 4096 feb 13 14:59 common/
lrwxrwxrwx  1 root    root      12 oct 10  2018 conf -> /etc/tomcat7/
lrwxrwxrwx  1 root    root     17 oct 10  2018 logs -> ../../log/tomcat7/
drwxr-xr-x  2 root    root    4096 feb 13 15:04 policy/
drwxr-xr-x  3 tomcat7 tomcat7 4096 feb 13 14:59 server/
drwxr-xr-x  3 tomcat7 tomcat7 4096 feb 13 14:59 shared/
drwxrwxr-x  3 tomcat7 tomcat7 4096 feb 13 14:59 webapps/
lrwxrwxrwx  1 root    root     19 oct 10  2018 work -> ../../cache/tomcat7/
alumno@ServidorLinux20:~$
```

Vemos que dentro de webapps se encuentra ROOT y dentro de este un fichero index.html el cual es el por defecto que crea Tomcat en la instalación que contiene la página que hemos visto antes en la máquina windows 7

```
alumno@ServidorLinux20:~$ cat /var/lib/tomcat7/webapps/ROOT/index.html _
```