DESARROLLO DE APLICACIONES WEB

10-11-2023

ROBERTO RODRÍGUEZ JIMÉNEZ

roberto.rodjim.1@educa.jcyl.es

Despliegue de aplicaciones Web

Configuración y administración de servidores de aplicaciones

Tarea 03

Contenido

[Tarea online DAW03 2](#_Toc153016458)

[Enunciado 2](#_Toc153016459)

[Caso práctico 2](#_Toc153016460)

[**Caso práctico** 2](file:///C:\xampp\htdocs\daw\daw\rodriguez_jimenez_roberto_DAW03_Tarea\rodriguez_jimenez_roberto_DAW03_Tarea.docx#_Toc153016461)

[¿Qué te pedimos que hagas? 2](#_Toc153016462)

[Recursos necesarios 3](#_Toc153016463)

[Recomendaciones 3](#_Toc153016464)

[Evaluación de la tarea 3](#_Toc153016465)

[Criterios de evaluación implicados 3](#_Toc153016466)

[¿Cómo valoramos y puntuamos tu tarea? 3](#_Toc153016467)

[Respuestas 5](#_Toc153016468)

[1. Una aplicación web puede ser desplegada en diferentes servidores web manteniendo su funcionalidad y sin ningún tipo de modificación en su código debido a la especificación servlet 2.2 , ¿cuál es la estructura de directorios que debe tener? 5](#_Toc153016469)

[2. Ant se basa en ficheros XML, normalmente configuramos el trabajo a hacer con nuestra aplicación en un fichero llamado build.xml. Indica alguna de las etiquetas con las que podemos formar el contenido de este archivo. 7](#_Toc153016470)

[3. Dispones de una máquina que cuenta con el sistema operativo Ubuntu 18.04 LTS o posterior, recientemente actualizado, en la que está el entorno de red configurado y, además, dispones de conexión a Internet y estás trabajando con la cuenta del usuario root . Indica cada uno de los pasos, y comandos implicados en ellos, para conseguir hacer lo siguiente: 8](#_Toc153016471)

[1. Instalar el JDK 8. 8](#_Toc153016472)

[2. Crear usuario para WildFly. 9](#_Toc153016473)

[3. Descargar e instalar WidlFly 19.0.0 Final. 10](#_Toc153016474)

[4. Configurar systemd y el archivo wildfly.conf. 12](#_Toc153016475)

[5. Configurar la autenticación de Wildfly. 14](#_Toc153016476)

# Tarea online DAW03

**Título de la tarea:** Instalación y cobfiguración de un servidor JBoss / WildFly  
**Unidad:** 3  
**Ciclo formativo y módulo:** Desarrollo de Aplicaciones Web - Despliegue de Aplicaciones Web

## Enunciado

### Caso práctico

### **Caso práctico**

En la empresa BK programación, Ada, les pide a sus empleados Juan y María que evaluen el servidor de aplicaciones JBoss / WildFly para instalar en ellos demos, o versiones beta, de las aplicaciones que desarrollan, de esta manera los clientes, o potenciales clientes, podrían probar los productos de BK programación antes de adquirirlos. Entre los parámetros a evaluar cabe destacar los siguientes:

* Seguridad de los servidores de aplicaciones: medidas de seguridad a aplicar para evitar posibles ataques o intrusiones.
* Dimensionamiento del servidor donde se estudian las necesidades físicas del equipo servidor.
* Tipo de servidor a instalar, características específicas del software de servidor seleccionado (Tomcat, JBoss / WildFly, etc.).
* Despliegue de aplicaciones en el servidor donde habría que establecer qué herramientas se deberían utilizar.
* Administración de las conexiones remotas a los servidores.
* Escalabilidad de los servidores, a tener en cuenta en función del número de conexiones simultáneas que se pueden establecer.
* Herramientas de automatización de tareas en el servidor (Ant, etc.).

### 

### ¿Qué te pedimos que hagas?

1. Una aplicación web puede ser desplegada en diferentes servidores web manteniendo su funcionalidad y sin ningún tipo de modificación en su código debido a la especificación servlet 2.2 , ¿cuál es la estructura de directorios que debe tener?
2. **Ant** se basa en ficheros XML, normalmente configuramos el trabajo a hacer con nuestra aplicación en un fichero llamado **build.xml**. Indica alguna de las etiquetas con las que podemos formar el contenido de este archivo.
3. Dispones de una máquina que cuenta con el sistema operativo **Ubuntu 18.04 LTS** o posterior, recientemente actualizado, en la que está el entorno de red configurado y, además, dispones de conexión a Internet y estás trabajando con la cuenta del usuario root . Indica cada uno de los pasos, y comandos implicados en ellos, para conseguir hacer lo siguiente:
   1. Instalar el JDK 8.
   2. Crear usuario para WildFly.
   3. Descargar e instalar WidlFly 19.0.0 Final.
   4. Configurar systemd y el archivo wildfly.conf.
   5. Configurar la autenticación de Wildfly.

Recursos necesarios y recomendaciones

**NOTA IMPORTANTE**

**Para el apartado 3 es necesario entregar las capturas de pantalla de los principales pasos realizados, explicando el proceso seguido en cada uno de ellos. Las capturas de pantalla realizadas deben tener como fondo de pantalla la plataforma con tu usuario mostrando claramente la foto de tu perfil. Aquellos apartados/subapartados que no cumplan esta condición no serán corregidos.**

### Recursos necesarios

* Computador con conexión a Internet.
* Gestor de máquinas virtuales tipo Virtual Box.
* Máquina virtual con última distribución de Ubuntu 18.04 LTS, o posterior.

### 

### Recomendaciones

Incluye la pregunta que se formula antes de cada una de las respuestas.

Detalla las respuestas de una forma estructurada, e intenta razonar y explicar de forma clara las decisiones adoptadas.

Ten en cuenta que para los ejercicios prácticos existen varias respuestas posibles, siendo todas ellas válidas.

## Evaluación de la tarea

### Criterios de evaluación implicados

* a. Se han descrito los componentes y el funcionamiento de los servicios proporcionados por el servidor de aplicaciones.
* b. Se han identificado los principales archivos de configuración y de bibliotecas compartidas.
* d. Se han configurado y activado los mecanismos de seguridad del servidor de aplicaciones.
* e. Se han configurado y utilizado los componentes web del servidor de aplicaciones.
* f. Se han realizado los ajustes necesarios para el despliegue de aplicaciones sobre el servidor.
* h. Se ha elaborado documentación relativa a la administración y recomendaciones de uso del servidor de aplicaciones.
* i. Se ha elaborado documentación relativa al despliegue de aplicaciones sobre el servidor de aplicaciones.

### ¿Cómo valoramos y puntuamos tu tarea?

|  |  |
| --- | --- |
| **Rúbrica de la tarea** | |
| **Apartado 1:** Aplicación Web, estructura de directorios. Respuesta correcta. | 2 puntos |
| **Apartado 2:** Archivo build.xml. Respuesta correcta. | 2 puntos |
| **Apartado 3.1:** Instalar el JDK 8. | 1 punto |
| **Apartado 3.2:** Crear usuario WildFly. | 1 punto |
| **Apartado 3.3:**Descargar e instalar WildFly. | 1 punto |
| **Apartado 3.4:**Configurar systemd y el archivo wildfly.conf. | 2 puntos |
| **Apartado 3.5:**Configurar la autenticación de Wildfly. | 1 punto |

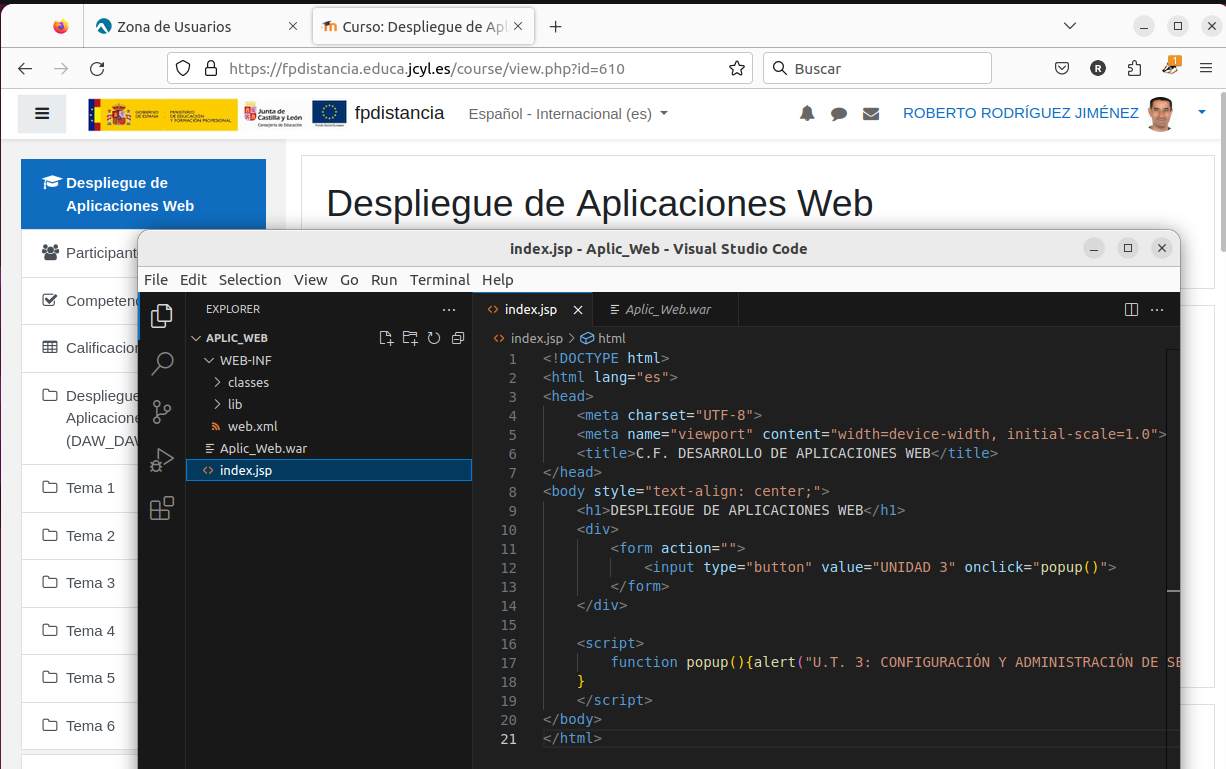
**NOTA IMPORTANTE**

**Aquellos apartados/subapartados en los que las capturas de pantalla no sean claras o no tengan como fondo de pantalla la plataforma con tu usuario mostrando claramente la foto de tu perfil, no serán corregidos.**

## Respuestas

### Una aplicación web puede ser desplegada en diferentes servidores web manteniendo su funcionalidad y sin ningún tipo de modificación en su código debido a la especificación servlet 2.2 , ¿cuál es la estructura de directorios que debe tener?

La aplicación debe tener un directorio raíz para los ficheros estáticos y un directorio WEB-INF para descriptor de la aplicación y subcarpetas con los ficheros compilados y las librerías adicionales.



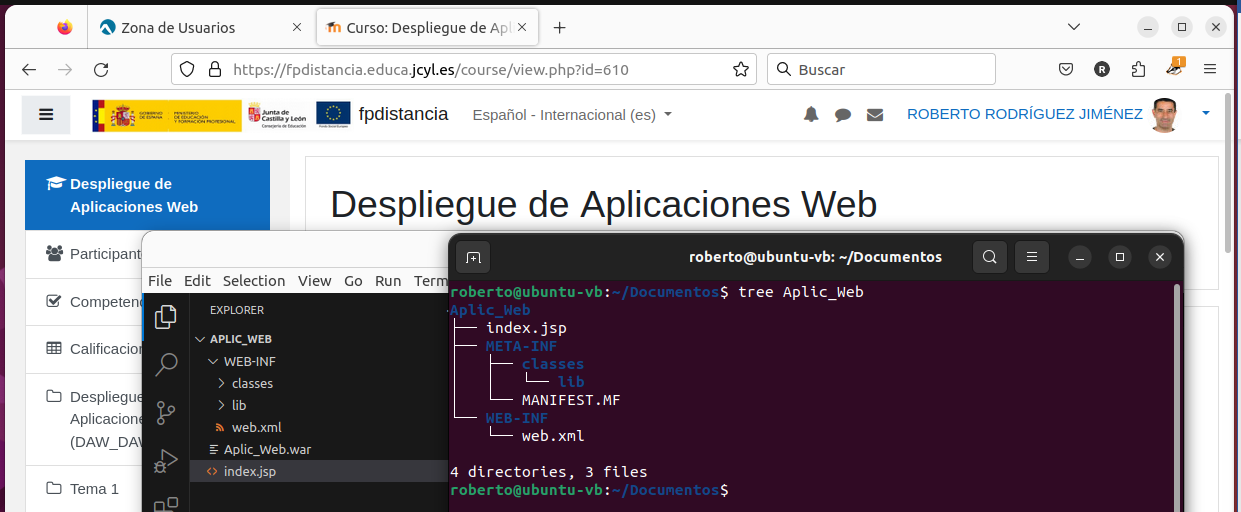
Vista de la estructura de una aplicación según la especificación servlet 2.2

* **Aplicación**
  + **WEB-INF** contiene los archivos necesarios para la aplicación pero que no deben ser servidos al cliente.
    - **classes** contiene los Servlets, Beans y otras clases de Java.
    - **lib** es el directorio en el que colocar las librerías (.jar) necesarias para que la aplicación funcione, como pueden ser con conectores para la base de datos.
    - **web.xml** es el descriptor de la aplicación.
  + **index.jsp** documento de entrada a la aplicación.
  + **\*.css, \*.html, \*.js, etc** resto de archivos que son accesibles para el cliente.

Estos archivos pueden organizarse dentro de directorios propios e, incluso a su vez, en un directorio común, or ejemplo **/webcontent**.

El archivo war (Web Application Archive).

Es un archivo que empaqueta la aplicación completa.



1Directorios y archivos de archivo .war

* **war**
  + **index.jsp** documento para mostrar la web.

En este directorio se almacenan los elementos empleados para la eweb.

* + **META-INF**
    - **classes** contiene las clases empleadas en la aplicación.
      * **lib** directorio para las librerías necesarias (.jar)
    - **MANIFEST.MF**
  + **WEB-INF** es un directorio para los archivos de configuración del archivo war.
    - **web.xml** es un archivo de configuración de la aplicación.

### Ant se basa en ficheros XML, normalmente configuramos el trabajo a hacer con nuestra aplicación en un fichero llamado build.xml. Indica alguna de las etiquetas con las que podemos formar el contenido de este archivo.

* **project** es el elemento raíz.
  + Solamente puede haber un elemento, que es el de la propia aplicación.
* **property** son parámetros necesarios para procesar la aplicación.
  + Las *property* tienen la forma clave-valor.
  + Ant ya incluye algunas de estas propiedades por defecto.
  + Estas propiedades pueden ser usadas en las *task* (tareas).
* **task** es un código que se ejecuta en la aplicación.
  + Los *task* hacen uso de las *porperties*.
  + Un conjunto de tareas se agupa en un *traget*.
* **target** es un conjunto de tareas (objetivo) que se quiere aplicar a la aplicación.
  + Un *target* puede depender de otro.

Ejemplo de archivo *build.xml* en el que los *target* tiene dependencias.

<project name="Proyecto">

<target name="init">

<mkdir dir="classes" />

<mkdir dir="build" />

</target>

<target name="compile" depends="init">

<javac srcdir="src" destdir="classes" />

</target>

<target name="build" depends="compile">

<jar destfile="build/proyecto.jar" basedir="classes" />

</target>

<target name="clean">

<delete dir="classes" />

<delete dir="build" />

</target>

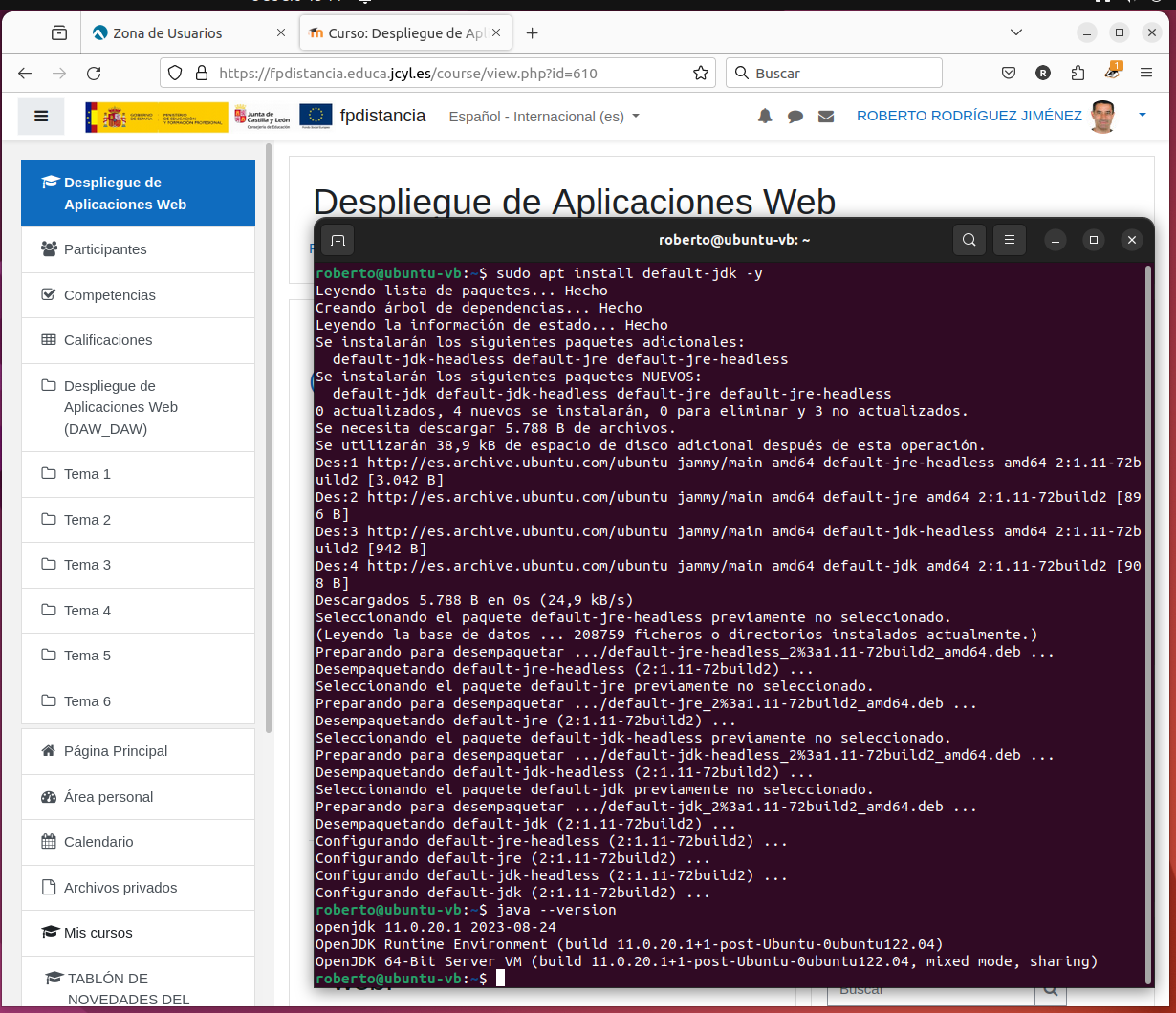
</project>

### Dispones de una máquina que cuenta con el sistema operativo Ubuntu 18.04 LTS o posterior, recientemente actualizado, en la que está el entorno de red configurado y, además, dispones de conexión a Internet y estás trabajando con la cuenta del usuario root . Indica cada uno de los pasos, y comandos implicados en ellos, para conseguir hacer lo siguiente:

#### Instalar el JDK 8.

Instalar la versión por defecto del JDK

sudo apt install default-jdk -y



Instalación y versión del JDK instalado

#### Crear usuario para WildFly.

Opciones que se usan en los comandos:

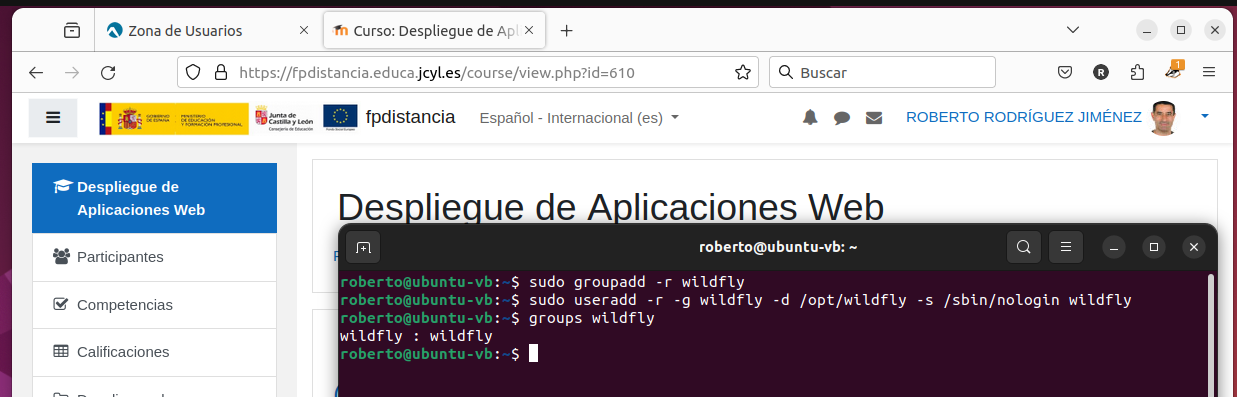
* -r Crea una cuenta del sistema.
* -g Indica el grupo al cual pertenece el usuario.
* -d Directorio personal para el usuario.
* -s Shell para el nuevo usuario: se indica que no puede loguearse.

Crear el grupo llamado *wildfly*

sudo groupadd -r wildfly

Crear el usuario

sudo useradd -r -g wildfly -d /opt/wildfly -s /sbin/nologin wildfly

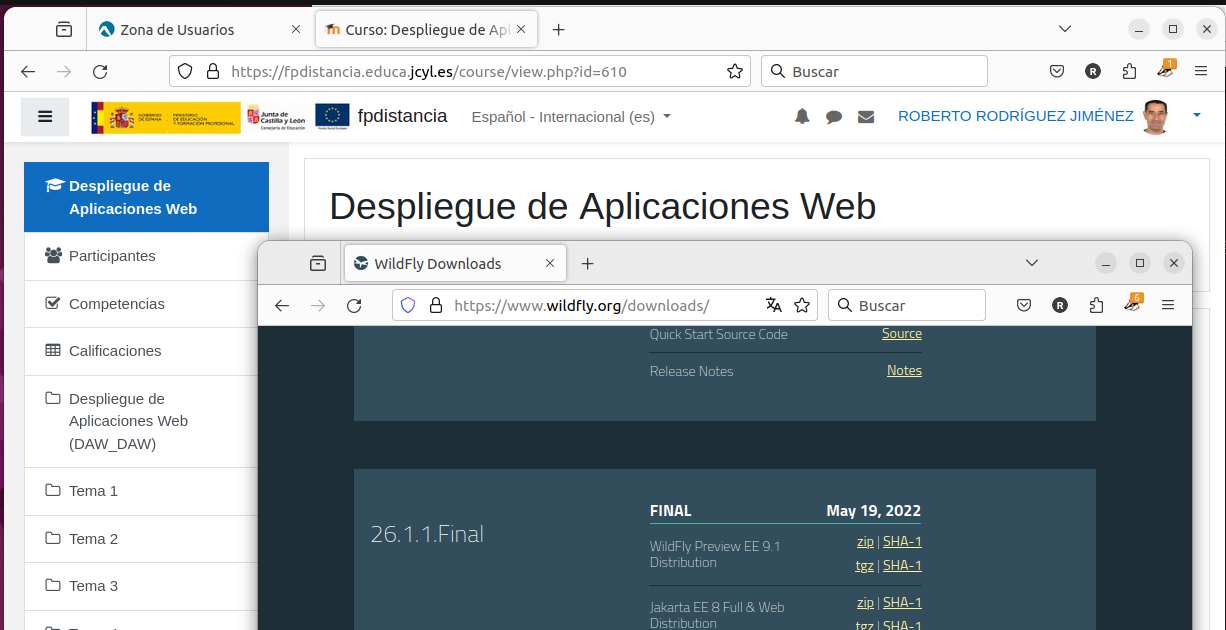


Crear usuario y el grupo al cual añadirlo

#### Descargar e instalar WidlFly 19.0.0 Final.

Buscamos una versión algo más moderna que la del temario, pero de las últimas.  
Decido que voy a instalar la versión 26.1.1 de mayo de 2022.  
La actual es la 30.0.1 Final de Diciembre de 2023.

Copio la ruta del archivo *Jakarta EE 8 Full & Web Distribution* comprimido con *tgz.*



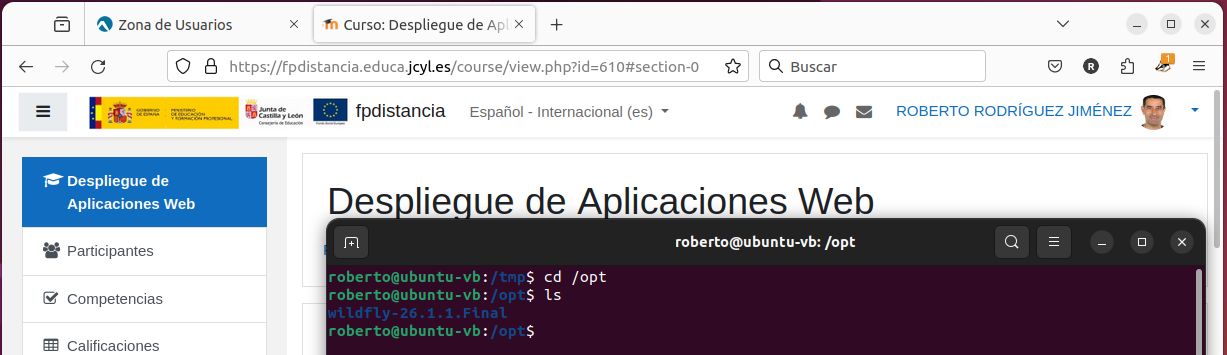
Página de descarga de WildFly

Nos situamos en el directorio */tmp* y descargamos el archivo

cd /tmp  
sudo wget <https://github.com/wildfly/wildfly/releases/download/26.1.1.Final/wildfly-26.1.1.Final.tar.gz>

Descomprimir el archivo y moverlo a */opt*

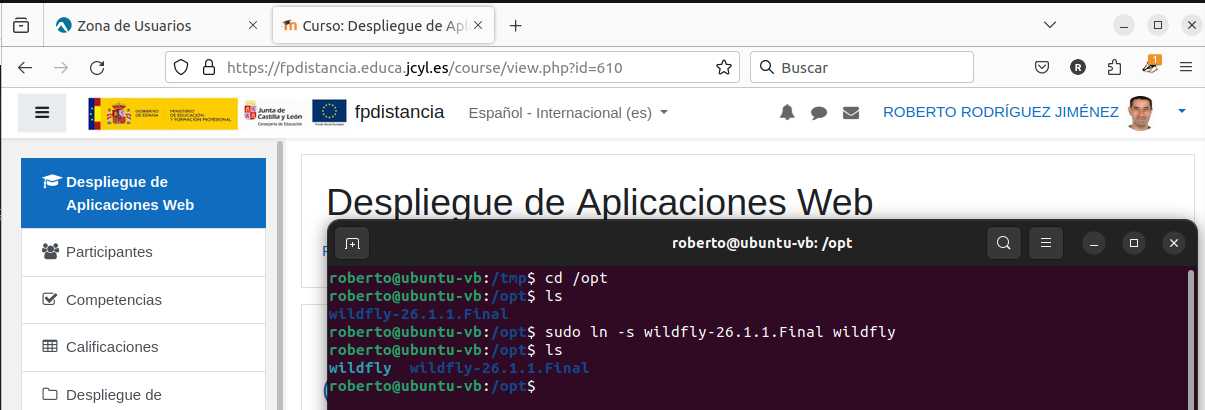
sudo tar xzvf wildfly-26.1.1.Final.tar.gz -C /opt



Archivo descomprimido en /opt

Nos movemos a */opt* y creamos un enlace simbólico.

cd /opt  
sudo ln -s wildfly-26.1.1.Final wildfly



Se ha creado el enlace simbólico a wildfly26.1.1.Final

Hacemos al usuario *wildfly* propietario del grupo *wildfly.*

sudo chown -RH wildfly: /opt/wildfly

Opciones:

* **-R** Hace propietario del directorio y todos los subdirectorios y ficheros.
* **-H** Atraviesa los enlaces simbólicos.

#### Configurar systemd y el archivo wildfly.conf.

Crear el directorio para los archivos de configuración

**-p** crea los directorios padre si no existen.

sudo mkdir -p /etc/wildfly

Copiar el archivo de configuración.

sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.conf /etc/wildfly/

Copiar el script que lanza la aplicación al directorio */opt/wildfly/bin*

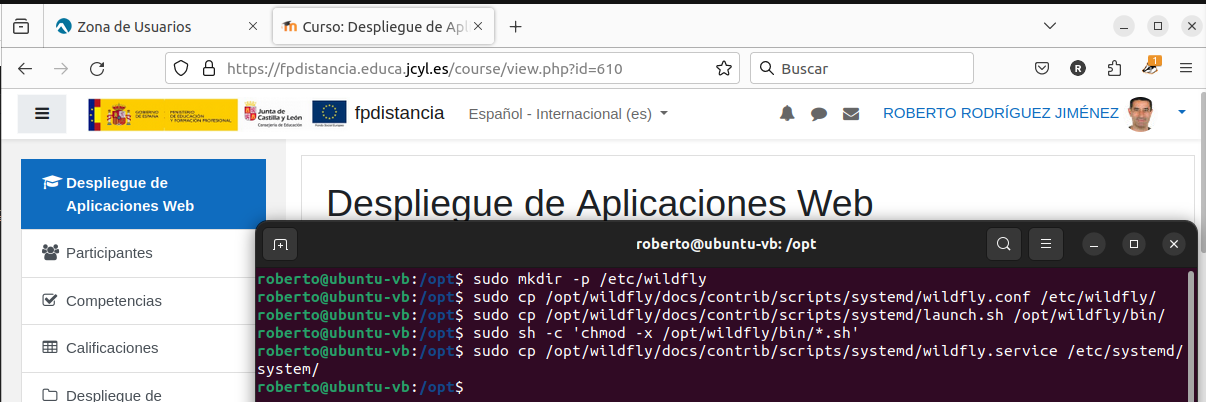
sudo cp /opt/wildfly/docs/contrib/scripts/systemd/launch.sh /opt/wildfly/bin/

Hacer ejecutables los ficheros del directorio *bin*.

sudo sh -c 'chmod +x /opt/wildfly/bin/\*.sh'

Copiar el archivo del servicio a directorio */etc/systemd/system/*

sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.service /etc/systemd/system/



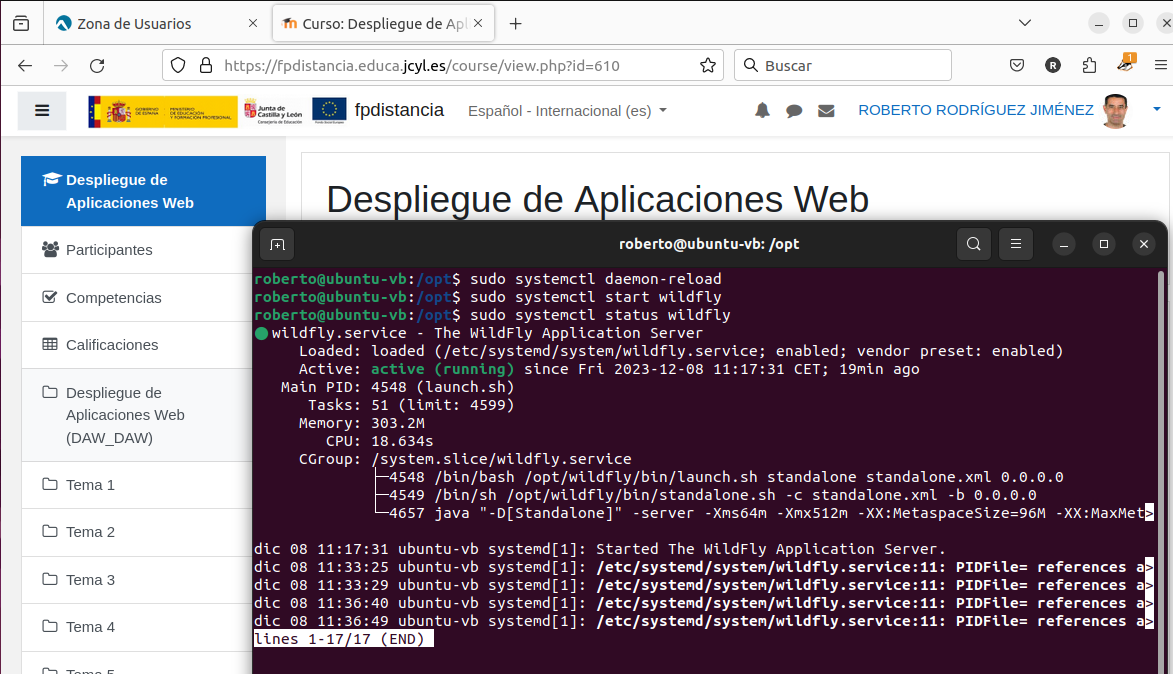
Configuración

Recargar el demonio, iniciar el servicio y comprobar su funcionamiento.

sudo systemctl daemon-reload

sudo systemctl start wildfly

sudo systemctl status wildfly



Comprobación del servicio wildfly

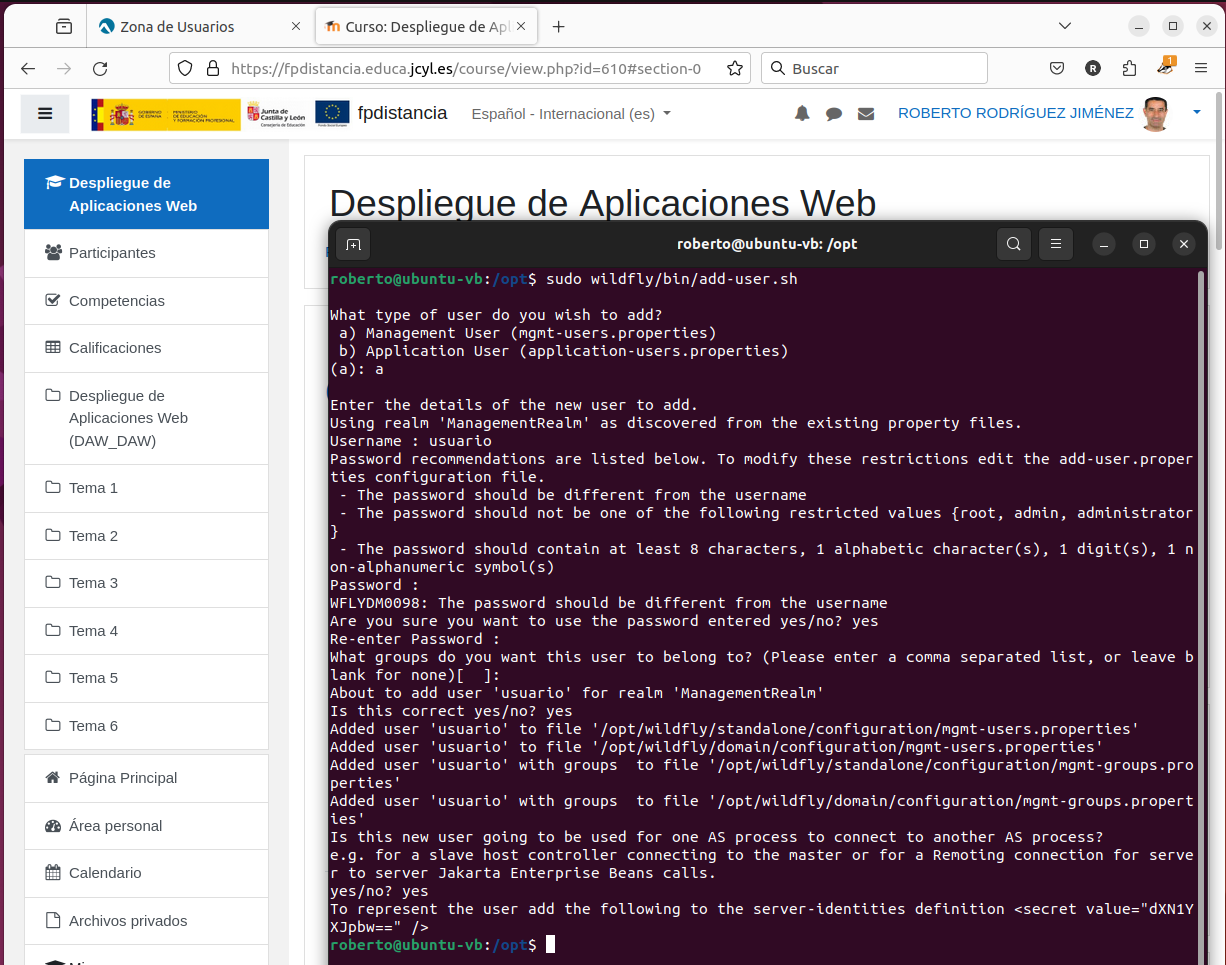
#### Configurar la autenticación de Wildfly.

Crear el usuario *usuario/usuario* para la administración.

Ejecutamos el script *wildfly/bin/add-user.sh*

sudo wildfly/bin/add-user.sh

Marcamos que queremos un usuario de tipo administrativo (a)  
Escribimos usuario tanto para el nombre como para la clave.   
El sistema nos advierte de que no es una buena idea y nos pide confirmación.  
La opción de vincular la cuenta a un grupo la dejamos en blanco.



Configuración de una cuenta administrativa

WildFly funcionando

