|  |
| --- |
| Fernando Ismael Pacheco Sánchez |
| Configuración de entorno |
| Configurar entorno para desarrollar o ejecutar los ejercicios de repositorio |
|  |
| **Fernando Ismael Pacheco Sánchez** |
| **21/03/2019** |

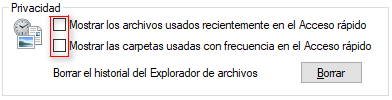
# Introducción

Este documento intenta recopilar toda la información necesaria para preparar un entorno de desarrollo óptimo para poder desarrollar aplicaciones JavaEE y con ello poder probar los ejercicios del repositorio correspondiente de GitHub.

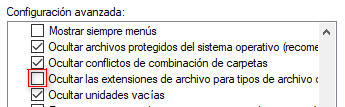
# Carpetas locales

Antes de empezar, vamos a desactivar la opción por defecto que tiene el explorador de archivos de Windows, a la vez que activamos la extensión de ficheros.

Parta ello en el buscador tecleamos: **Opciones del explorador de archivos**. Seleccionamos la primera coincidencia que nos sale, ello nos abre una venta y en la **pestaña** abierta (**General**) quitamos las opciones de privacidad, tal que:



Ahora bien, es útil tener visibilidad de las extensiones de ficheros que por defecto viene desactivado en Windows, por tanto, simplemente en la misma ventana, cambiamos a la **pestaña** de “**Ver**” y desmarcamos la opción:



Es muy importante tener organizadas las carpetas locales para el desarrollo, para ello, a lo largo de mi carrera he probado varias formas de organización, pero la que me gusta es la siguiente:

## Carpeta Desarrollo

En esta carpeta ubicaremos todo el software que usaremos para el desarrollo como Tomcat, Eclipse, SourceTree, etc. No olvidemos linkear como acceso directo en el explorador de archivos, para ello sobre la carpeta:

*Botón derecho > Anclar al Acceso rápido.*

Esta carpeta debe ubicarse en la raíz de un disco, en mi caso:

**C:\Desarrollo**

## Carpeta usuario

Con carpeta usuario nos referimos a la carpeta del usuario logado, pues bien, esta carpeta la linkearemos como acceso directo al igual la carpeta de desarrollo. Ello lo hacemos porque dentro de esta carpeta crearemos dos carpetas, una para los repositorios locales de GitHub y otra para los espacios de trabajo de Eclipse (workspace).

La ubicación de esta carpeta en mi caso es:

**C:\Users\Fernando Pacheco**

### Carpeta Git

Esta carpeta contendrá los repositorios locales que se sincronizarán con GitHub a través de SourceTree. Veremos que los cambios realizados en Eclipse se reflejan en estas carpetas y no en los espacios de trabajo de Eclipse.

No hace falta tener un link de esta carpeta, porque ya tenemos link de la carpeta padre, es decir, del usuario.

La ubicación de la carpeta sería, en mi caso:

C:\Users\Fernando Pacheco**\Git**

### Carpeta Workspaces

Esta carpeta contendrá los espacios de trabajo de Eclipse (Un workspace por cada repositorio), pero veremos más adelante que los códigos fuente que editemos estarán ubicados en la carpeta de Git y no en el workspace del repositorio, ello se debe a que tendremos sincronizados los proyectos de eclipse con Git y este con SourceTree que tiene conexión con GitHub.

Organizarlo de esa forma, nos permite no saturar a Eclipse cuando realizamos alguna sincronización con GitHub, es decir, es más rápido y podemos trabajar de forma segura. Al igual que la carpeta de Git, esta no hace falta tener link.

La ubicación de la carpeta, en mi caso

C:\Users\Fernando Pacheco**\Workspaces**

# Eclipse

Usaremos Eclipse como IDE de desarrollo, lo único importante a tener en cuenta antes de descargarlo es de que debe ser una versión de Eclipse con los complementos de JavaEE, con ello nos ahorramos el tener que estar instalando plugins y configuraciones para usar Git dentro de Eclipse.

Lo mejor es descargarse la versión comprimida zip (Que por defecto suele venir en dicho formato), con ello simplemente los descomprimimos en nuestra carpeta **Desarrollo** y ya sabemos dónde buscar los ficheros de configuración de Eclipse, ya que editaremos un fichero llamado “eclipse.ini”.

En mi caso, la ruta donde lo he descomprimido es:

**C:\Desarrollo\eclipse-jee-oxygen**

## Fichero eclipse.ini

Cuando tenemos varias versiones de eclipse, tendremos que decirle cual será su JDK, ya que cada versión puede trabajar con una versión de JDK. Para ello tendremos que ir a la ubicación y editar el fichero “eclipse.ini”, la ubicación en mi caso es:

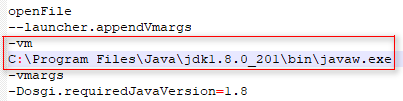
**C:\Desarrollo\eclipse-jee-oxygen\eclipse.ini**

Dentro, pondremos el argumento correspondiente al JDK (dos líneas), para ello pondremos:

**-vm**

**C:/Program Files/Java/** **jdk1.8.0\_201/bin/javaw.exe**

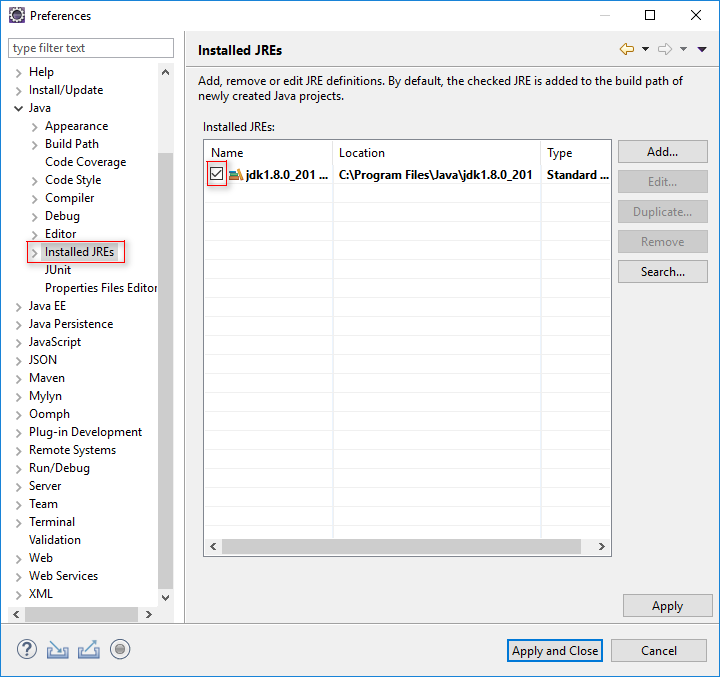
Y la ubicación de estas líneas deben de ser en:



En este caso hemos puesto ese JDK, pero podemos usar cualquiera que sea igual o superior al requerido, en este caso >= JDK 1.8.

Para asegurarnos que usará el JDK que hemos definido en el fichero, tendremos que arrancar Eclipse y configurar el JDK. Arrancado Eclipse, nos vamos a la opción:

*Windows > Preferences*



Debe estar marcado Check. De no tener la misma versión, tendremos que borrar el existente y agregar el nuestro, para ello le damos a:

*Add > Seleccionamos “Standar VM” > Next > Botón “Directory” > Seleccionamos carpeta JDK*

## Configurar codificación

Cuando usamos repositorios, lo mejor es tener un único encoding independiente del sistema operativo en el que desarrollamos, por ello tenemos que fijarnos siempre en el encoding antes de desarrollar.

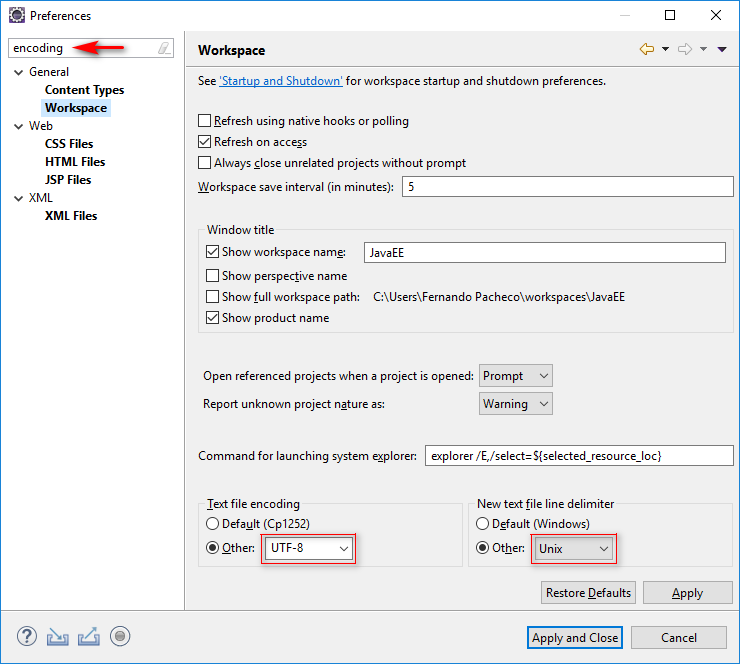
La codificación o encoding que usaremos será la de **UTF-8**, ya que es esa la que todos los servidores y repositorios entienden. En el caso de los repositorios, conseguimos que al sincronizar no tengamos caracteres extraños que se corresponden con los caracteres especiales en español como las tildes.

Para ello, arrancamos eclipse y nos vamos al menú superior a la opción de:

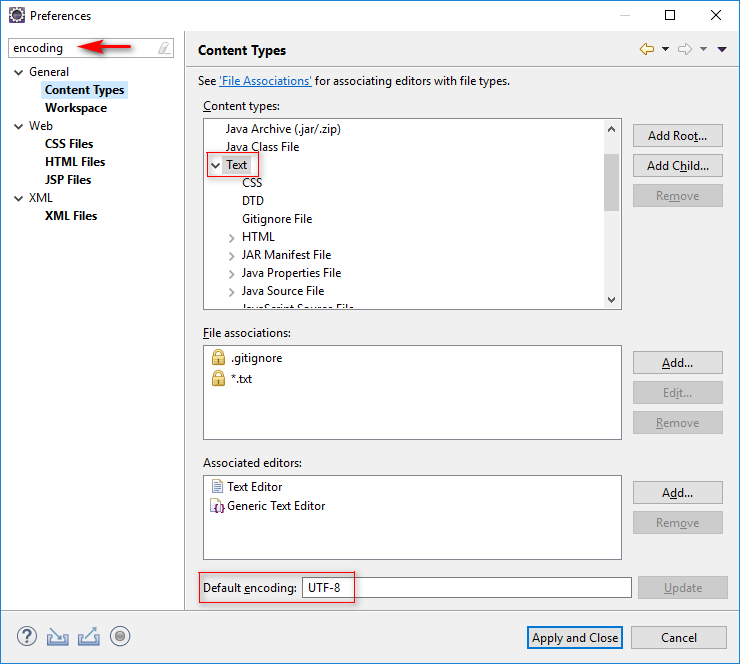
*Windows > Preferences*

En la ventana que se abre, en el buscador pondremos “encoding” para ver las opciones disponibles.

Lo primero, la opción **Workspace** debe de tener:



Ahora tenemos que decirle a Eclipse que todos los ficheros que lleven texto deben tener la codificación **UTF-8**, para ello, en la misma ventana seleccionamos:



Ahora, como vemos, dentro de Text tenemos varios ficheros, CSS, DTD, etc. Pues bien, cada uno de ellos debe tener la codificación **UTF-8**, de tener otra, la cambiamos.

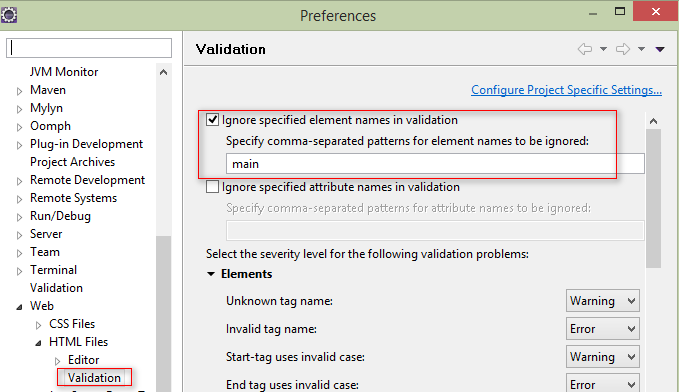
Importante los JSP, poner cada uno de su sub-menú al formato mencionado, para que se aplique, podemos darle al botón de “Update”.

## Configurar etiquetas HTML\_5

Cuando desarrollamos aplicaciones web, y estas queremos que sean HTML\_5, tenemos que configurar Eclipse para que no nos de avisos cuando usamos etiquetas nuevas.

Para ello, tenemos que saber que etiquetas son HTML\_5 e ir añadiéndolas en un listado según las vamos usando.

El listado lo podemos editar en: Menú Eclipse > Windows > Preferences > Web > HTML Files > Validation:



Como observamos, podemos agregar todas las etiquetas que deseemos, obviamente las etiquetas que nos de warning cuando desarrollamos.

## Plugin Web

A la hora de desarrollar tenemos un plugin muy útil para la web, verificar antes instalar que el plugin sigue activo o si no existe, uno parecido ya que nos será de gran utilidad. El plugin es:



## Instalar Tomcat

Pasamos a la descarga e instalación de nuestro contenedor de aplicaciones web, es decir, nuestro servidor donde van a correr cada uno de nuestros ejercicios. Por tanto, lo primero que tenemos que hacer es descargar la versión zip de Tomcat desde su página web:



Descargada, la descomprimimos en nuestra carpeta de Desarrollo, ya que más adelante tendremos que editar algunos ficheros de Tomcat, aunque también lo podemos hacer desde Eclipse, pero eso ya iremos viendo a medida que vamos avanzando en los ejercicios.

La ruta donde hay que descomprimirlo, en mi caso:

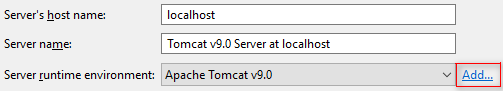
**C:\Desarrollo\apache-tomcat-9.0.16**

Instalado en nuestro equipo, nos queda linkear Tomcat con Eclipse, lo cual es muy sencillo. Arrancamos Eclipse y en las pestañas inferiores seleccionamos la **pestaña** para los **servidores**:

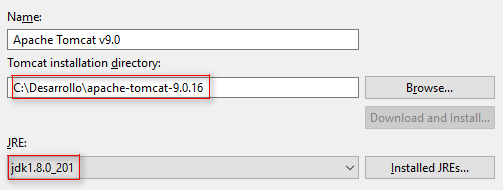


*Click > Apache > Tomcat (La versión que tengas descargada).*

Si ya se tiene instalado un Tomcat, los campos de la imagen saldrán rellenos, de no ser así, debemos darle a la opción de añadir.

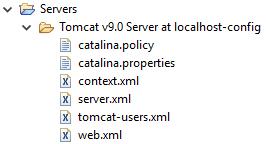


Al darle al botón añadir se nos abre una nueva ventana donde basta seleccionar la ruta de nuestro Tomcat que hemos descomprimido, es decir:



En el campo JRE, tenemos que desplegar el combo y seleccionar el JDK que hemos configurado en Eclipse. Tomcat se ejecutará con esa versión de JDK y cuando lancemos nuestras aplicaciones con Tomcat, este ejecutará la aplicación con el mismo JDK que hemos desarrollado.

Le damos a siguiente y veremos que se nos ha creado una carpeta de proyecto llamada Servers, donde tendremos todos los ficheros de configuración respecto a Tomcat.



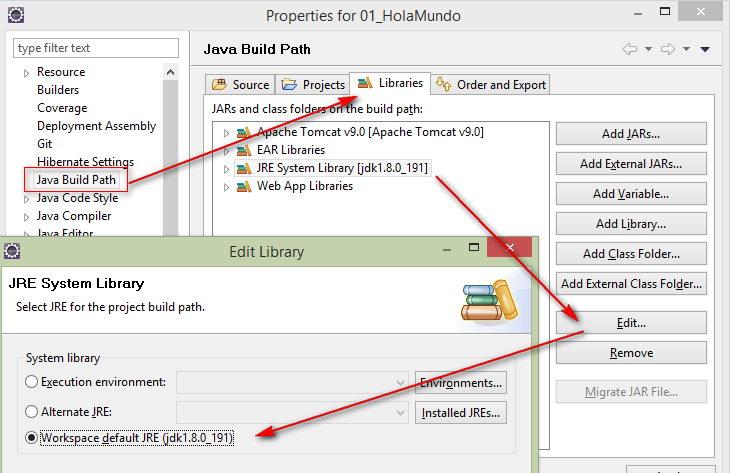
Los ficheros que editaremos para generar **DataSource** son “context.xml” y “server.xml”. Otro detalle que veremos es que en nuestra pestaña de servidores ya vemos nuestro Tomcat tal que:



## Proyectos

Cuando creamos un proyecto, podremos una configuración adaptable a todas las versiones de una versión estándar de Java, es decir, podemos tener JDK 1.8.0\_121, 1.8.0\_142, 1.8.0\_483, Etc. Siempre será la 8, por tanto, para evitarnos errores de este tipo, vamos a definir tal que:

Seleccionamos proyecto que acabamos de crear, Botón derecho > Properties > Java Build Path > En la ventana, vamos a la pestaña Libraries > Seleccionamos el JRE System Library > Botón Edit.



Con esto nos ahorramos el tener una versión exacta de JDK, tenemos que tener en cuenta que en el futuro saldrán nuevas versiones, pero no habría problema, pillamos una cualquiera de la 8 y se la asignamos como predefinida en nuestro Eclipse.

# SourceTree

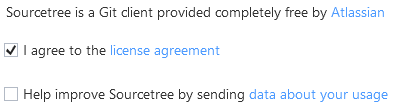
Es el software que usaremos para estar sincronizados con los repositorios de GitHub, para ellos nos descargamos el fichero instalable. Cuando lo lancemos, podemos elegir en ponerlo en la ruta por defecto o en la ruta de nuestra carpeta Desarrollo.

Cuando lo estemos instalando, veremos que nos pedirá un usuario y contraseña, donde podemos elegir en crear una cuenta o usar nuestra **cuenta de Gmail**, yo he optado por la segunda ya que me es más cómodo.

## Instalación

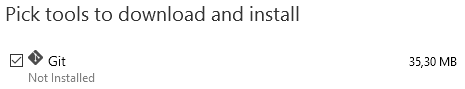
Al ser una herramienta nueva y no cometer errores, describimos los pasos a seguir para tener una instalación correcta.

Nos descargamos el software, es decir, el fichero ejecutable y lo ejecutamos, lo primero que nos saldrá será el aceptar la licencia, total que tenemos que tener:

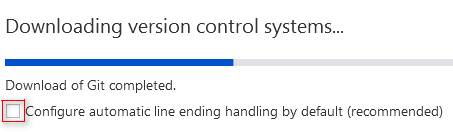


El siguiente paso es logearnos con una cuenta de Atlassian, la cual no tendremos, pero seleccionamos esa opción y automáticamente nos abre una nueva ventana con la opción de logearnos con nuestra cuenta Gmail, que es la opción que yo he elegido por comodidad.

Veremos que automáticamente ya estamos logados, pasamos al paso de instalar las herramientas necesarias para la sincronización de repositorio, solo seleccionamos GitHub, que es donde yo tengo mis repositorios, aunque se tiene la opción de elegir Mercurial.

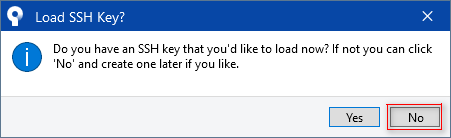


Le damos a siguiente y empezará a descargar todo lo necesario para instalarlo en nuestro equipo, es muy importante quitar el tick que viene seleccionado por defecto, es decir:



Aún desmarcado la opción, la instalación se realiza de forma automática, por tanto no podemos elegir la ubicación para la instalación. Esto no es problema, total, no tendremos que configurar los ficheros de configuración de SourceTree.

Una vez instalado, automáticamente nos abre una mini ventana que nos informa de si tenemos una clave SSH, simplemente le decimos que no y ya tendremos instalado Sourcetree.



## Clonar repositorio

Ahora nos queda sincronizar nuestro equipo con GitHub, es la manera más sencilla a la hora de trabajar con Eclipse, total, tenemos SourceTree abierto y veremos varias opciones, donde una de ella realiza la clonación de un repositorio de GitHub.

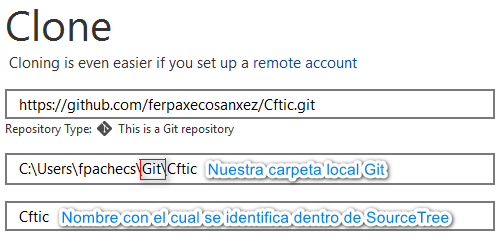
Lo de clonar repositorio es útil, si solo si, se quiere tener un repositorio propio y trabajar sobre ese repositorio y no sobre el repositorio original, es decir, el mío, que es donde están los ejercicios.

Si se decide clonar, simplemente nos vamos al repositorio de GitHub y copiamos la url que nos facilita la propia página web, es decir:



Copiado, nos vamos a SourceTree y damos en el botón Clone, donde tendremos la dirección del repositorio de GitHub, la dirección local y el nombre con el cual estará descrito dentro de SourceTree, por defecto se descarga la rama root, pero si quisiéramos una rama en concreto, basta con desplegar el combo y seleccionarla.

Para seleccionar una rama en concreto debemos abrir las opciones avanzadas.



Clonado el repositorio ya podremos sincronizarlo con Eclipse, aunque podríamos haberlo hecho directamente desde Eclipse, para mi gusta esta forma es más intuitiva y se sabe dónde están los ficheros que se editan o sincronizan.

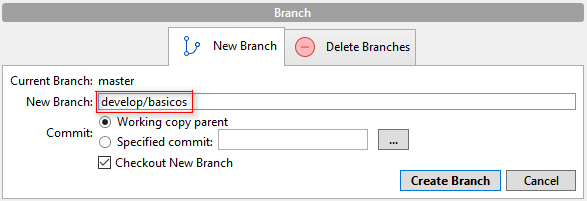
Con todo hecho, podemos pasar al siguiente documento, el cual describe como sincronizar un proyecto de Eclipse con un repositorio previamente ya en local.

## Crear rama

Cuando trabajamos con GitHub, lo más normal es trabajar en ramas de desarrollo, por tanto, si tenemos un repositorio nuevo tendremos que crear una rama, para ello:



Al darle, se nos abre una ventana donde tenemos que poner el nombre de la rama, cualquier rama que vayamos a crear se ubicarán automáticamente dentro de *origin*, por tanto no hace falta poner dicha palabra en la ruta, total, que tenemos:

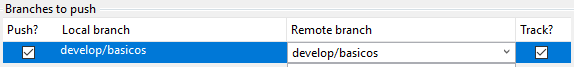


|  |  |
| --- | --- |
| Si nos fijamos en el panel de la izquierda, veremos la siguiente organización donde en la nube no está y es porque hemos creado la rama de forma local, por tanto, nos queda subir la rama a la nube. |  |

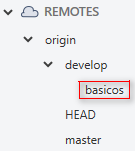
Al crear la rama, automáticamente se cambia a dicha rama de forma automática. Ahora bien, como dijimos, tenemos que subir la rama a la nube, para ello seleccionamos la rama creada y le damos a la opción:



Se nos abre una ventana donde debemos seleccionar la rama recién creada para subirla, con marcar el check nos vale, de tal manera que:



Y veremos que ya la tenemos en nuestra nube, es decir, que ya tenemos sincronizada la rama.



Ahora, para cambiar de rama, basta con darle dos click sobre la rama que se quiere cambiar y si tenemos sincronizado Eclipse con el repositorio local de GitHub, Eclipse automáticamente cambiará la rama.