

DAW06_Tarea

1. Indica cada uno de los pasos que deberías de dar para proceder a la instalación de phpDocumentor, suponiendo que vas a partir de una máquina en la que tienes instalado la distribución Debian / Ubuntu actual, y en la que ya están instalados y correctamente configurados apache y php.

Primero instalamos el paquete **php-pear**:

```
$ sudo apt-get update  
$ sudo apt-get install php-pear
```

Configuramos php-pear para que trabaje en la carpeta donde apache guarda las páginas:

```
$ pear config-set data_dir /var/www/html
```

Ahora podemos instalar pear y sus dependencias:

```
$ sudo pear install --alldeps PhpDocumentor
```

Una vez instalado tenemos que crear un directorio de salida para phpDocumentor y cambiar el propietario de dicho directorio a **www-data**:

```
$ mkdir /var/www/html/docs  
$ chown www-data /var/www/html/docs/
```

2. Explica en qué consisten las plantillas de código en el caso de Javadoc y cada uno de sus componentes.

Las plantillas de código son básicamente formas abreviadas de escribir bloques de código comunes. La más conocida en Java sería en NetBeans cuando, al escribir "sout" y pulsamos el tabulador, escribe directamente System.out.println().

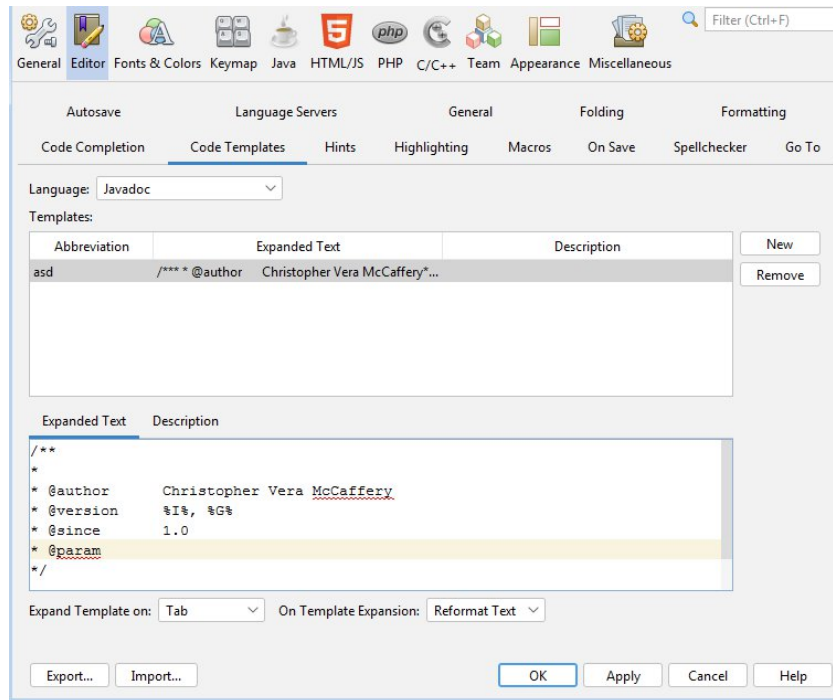
Para Javadoc, la plantilla tendría todas las etiquetas necesarias para escribir la documentación de un método o clase. No hay que ponerlas todas, especialmente @see, @deprecated y @throws, pero si vemos que siempre repetimos el mismo patrón de etiquetas, podemos crear una para cada situación.

Etiqueta
@version
@author
@param
@return
@see
@throws
@deprecated

A la hora de crear una plantilla para JavaDoc, aparte del bloque de texto inmutable, podemos añadir ciertas etiquetas que cambiarán dependiendo de la situación:

- `${cursor}` Esto no escribe nada, simplemente es la posición donde se colocará el puntero tras añadir el bloque de código
- `${enclosing_type}`: clase en la que nos encontramos.
- `${enclosing_method}`: método en el que nos encontramos.
- `${year}`: El año actual.
- `${time}`: La hora actual.

Este es un ejemplo de cómo podría ser una plantilla de javadoc en NetBeans.



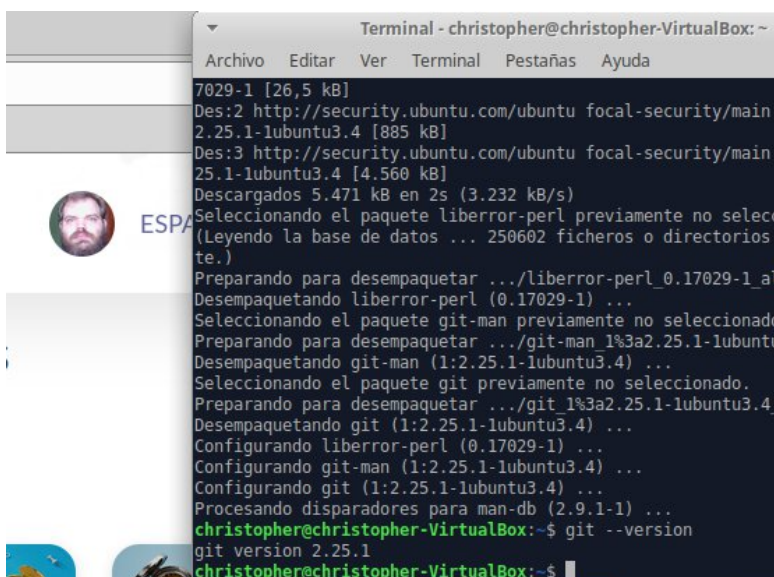
3. Dispones de una máquina que cuenta con el sistema operativo Debian / Ubuntu recientemente actualizado, en la que está el entorno de red configurado y, además, dispones de conexión a Internet y estás trabajando con la cuenta del usuario root . Indica cada uno de los pasos y comandos implicados en ellos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

1. Suponiendo que el sistema ya tiene instalado las siguientes librerías de las que Git depende: curl, zlib, openssl, expat, y libiconv, pasos a realizar la compilación e instalación de Git considerando que ya disponemos del paquete git-1.7.6.tar.bz2

La versión que especifica la tarea no está disponible para descargar, así que lo haremos con la última versión disponible para asegurarnos que no hay problemas de compatibilidad con dependencias.

Usamos el comando:
\$ sudo apt install git

Con esto ya instalamos git junto con sus dependencias.
Para comprobar la versión de git que tenemos instalada:
\$ git --version



```
Terminal - christopher@christopher-VirtualBox: ~
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda

Des:2 http://security.ubuntu.com/ubuntu focal-security/main
2.25.1-lubuntu3.4 [885 kB]
Des:3 http://security.ubuntu.com/ubuntu focal-security/main
2.25.1-lubuntu3.4 [4.560 kB]
Descargados 5.471 kB en 2s (3.232 kB/s)
Seleccionando el paquete liberror-perl previamente no selecc
(Leyendo la base de datos ... 250602 ficheros o directorios
te.)
Preparando para desempaquetar .../liberror-perl_0.17029-1_a
Desempaquetando liberror-perl (0.17029-1) ...
Seleccionando el paquete git-man previamente no seleccionado
Preparando para desempaquetar .../git-man_1%3a2.25.1-lubuntu
Desempaquetando git-man (1:2.25.1-lubuntu3.4) ...
Seleccionando el paquete git previamente no seleccionado.
Preparando para desempaquetar .../git_1%3a2.25.1-lubuntu3.4
Desempaquetando git (1:2.25.1-lubuntu3.4) ...
Configurando liberror-perl (0.17029-1) ...
Configurando git-man (1:2.25.1-lubuntu3.4) ...
Configurando git (1:2.25.1-lubuntu3.4) ...
Procesando disparadores para man-db (2.9.1-1) ...
christopher@christopher-VirtualBox:~$ git --version
git version 2.25.1
christopher@christopher-VirtualBox:~$
```

Index of /pub/software/scm/g

../		
RPMS/	03-Dec-2010	01:58
debian/	24-Dec-2005	08:56
docs/	28-Apr-2022	19:25
testing/	12-Apr-2022	16:59
git-0.01.tar.gz	07-Apr-2005	21:25
git-0.01.tar.sign	08-Aug-2013	19:44
git-0.02.tar.gz	08-Apr-2005	03:07
git-0.02.tar.sign	08-Aug-2013	19:44
git-0.03.tar.gz	09-Apr-2005	00:23
git-0.03.tar.sign	08-Aug-2013	19:44
git-0.04.tar.gz	11-Apr-2005	16:55
git-0.04.tar.sign	08-Aug-2013	19:44
git-0.5.tar.gz	20-Apr-2005	22:26
git-0.5.tar.sign	08-Aug-2013	19:44
git-0.6.tar.gz	21-Apr-2005	17:57
git-0.6.tar.sign	08-Aug-2013	19:44
git-0.7.tar.gz	29-Apr-2005	22:19
git-0.7.tar.sign	08-Aug-2013	19:44
git-1.8.2.3.tar.bz2	08-Aug-2013	19:44
git-1.8.2.3.tar.gz	09-May-2013	21:44
git-1.8.2.3.tar.sign	09-May-2013	21:44
git-1.8.2.3.tar.xz	09-May-2013	21:44
git-1.8.3.1.tar.bz2	10-Jun-2013	21:01
git-1.8.3.1.tar.gz	10-Jun-2013	21:01
git-1.8.3.1.tar.sign	10-Jun-2013	21:01
git-1.8.3.1.tar.xz	10-Jun-2013	21:01
git-1.8.3.2.tar.bz2	28-Jun-2013	22:24
git-1.8.3.2.tar.gz	28-Jun-2013	22:24
git-1.8.3.2.tar.sign	28-Jun-2013	22:24
git-1.8.3.2.tar.xz	28-Jun-2013	22:24
git-1.8.3.3.tar.bz2	15-Jul-2013	21:11
git-1.8.3.3.tar.gz	15-Jul-2013	21:11
git-1.8.3.3.tar.sign	15-Jul-2013	21:11

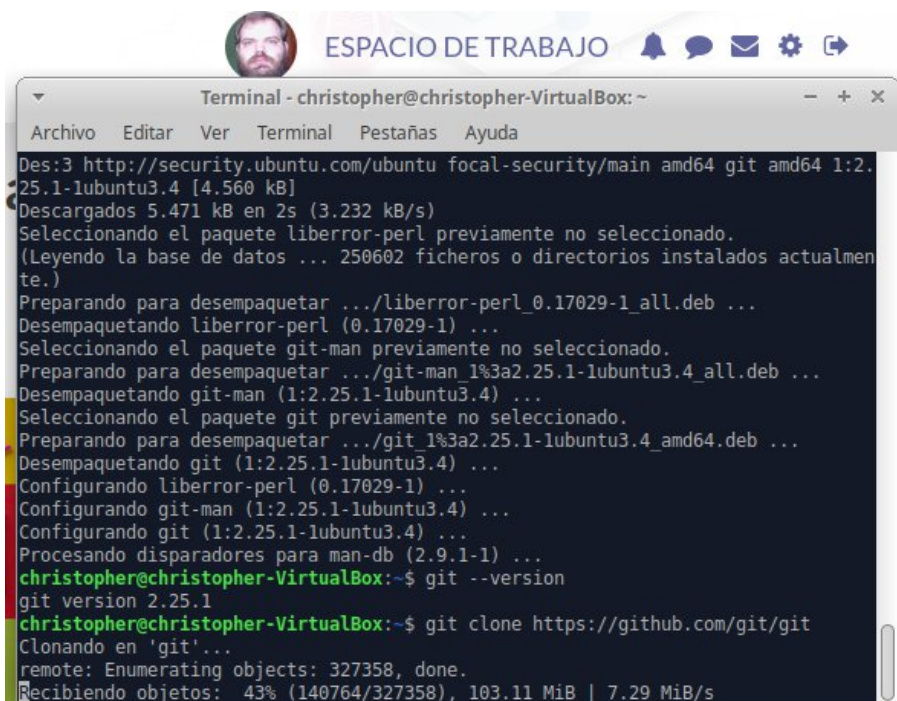
2. Cómo obtener Git a través del propio Git para futuras actualizaciones, de manera que descargaría automáticamente el código fuente desde su repositorio.

3. Comprobar la versión que se ha instalado de Git.

Para descargar git a través de git, usaríamos el comando:

\$ git clone <https://github.com/git/git>

Esto nos habrá descargado una carpeta "git" en nuestra carpeta home.



```
Terminal - christopher@christopher-VirtualBox: ~
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda

Des:3 http://security.ubuntu.com/ubuntu focal-security/main amd64 git amd64 1:2.
25.1-lubuntu3.4 [4.560 kB]
Descargados 5.471 kB en 2s (3.232 kB/s)
Seleccionando el paquete liberror-perl previamente no seleccionado.
(Leyendo la base de datos ... 250602 ficheros o directorios instalados actualmen
te.)
Preparando para desempaquetar .../liberror-perl_0.17029-1_all.deb ...
Desempaquetando liberror-perl (0.17029-1) ...
Seleccionando el paquete git-man previamente no seleccionado.
Preparando para desempaquetar .../git-man_1%3a2.25.1-lubuntu3.4_all.deb ...
Desempaquetando git-man (1:2.25.1-lubuntu3.4) ...
Seleccionando el paquete git previamente no seleccionado.
Preparando para desempaquetar .../git_1%3a2.25.1-lubuntu3.4_amd64.deb ...
Desempaquetando git (1:2.25.1-lubuntu3.4) ...
Configurando liberror-perl (0.17029-1) ...
Configurando git-man (1:2.25.1-lubuntu3.4) ...
Configurando git (1:2.25.1-lubuntu3.4) ...
Procesando disparadores para man-db (2.9.1-1) ...
christopher@christopher-VirtualBox:~$ git --version
git version 2.25.1
christopher@christopher-VirtualBox:~$ git clone https://github.com/git/git
Clonando en 'git'...
remote: Enumerating objects: 327358, done.
Recibiendo objetos: 43% (140764/327358), 103.11 MiB | 7.29 MiB/s
```

Ahora, con los siguientes comandos, instalamos los archivos descargados. Como vamos a dejar que se instale en el directorio por defecto, simplemente nos dirigimos a la carpeta que se ha descargado y escribimos:

```
$ sudo make install
```

Esto reinstalará git, pero como ya teníamos la misma versión, la versión sigue igual.



```
rm -f "$execdir/$p" && \
if test -z ""; \
then \
    test -n "" && \
    ln -s "$destdir_from_execdir_SQ/bin/git" "$execdir/$p" || \
    { test -z "" && \
        ln "$execdir/git" "$execdir/$p" 2>/dev/null || \
        ln -s "git" "$execdir/$p" 2>/dev/null || \
        cp "$execdir/git" "$execdir/$p" || exit; }; \
    fi \
done && \
remote_curl_aliases="git-remote-https git-remote-ftp git-remote-https" && \
for p in $remote_curl_aliases; do \
    rm -f "$execdir/$p" && \
    test -n "" && \
    ln -s "git-remote-http" "$execdir/$p" || \
    { test -z "" && \
        ln "$execdir/git-remote-http" "$execdir/$p" 2>/dev/null || \
        ln -s "git-remote-http" "$execdir/$p" 2>/dev/null || \
        cp "$execdir/git-remote-http" "$execdir/$p" || exit; } \
done
christopher@christopher-VirtualBox:~/git$ git --version
git version 2.25.1
christopher@christopher-VirtualBox:~/git$
```

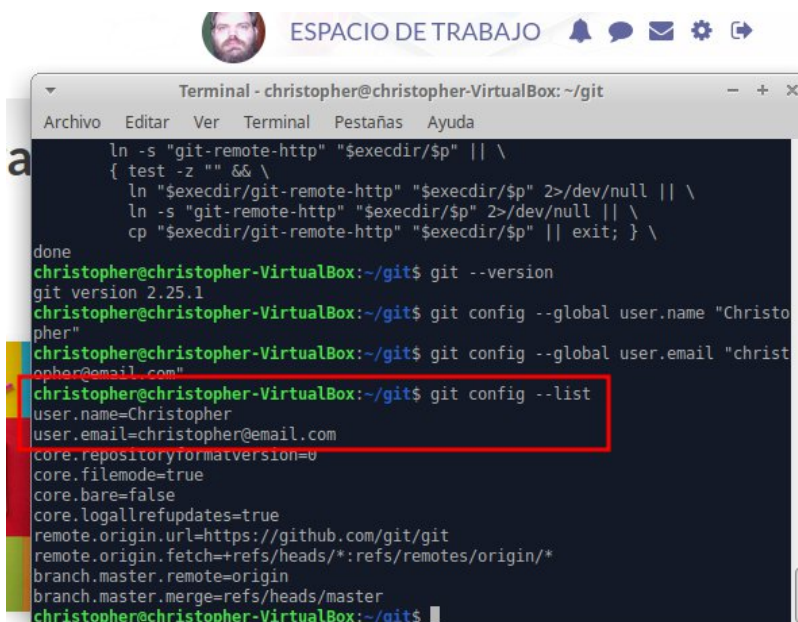
4. Establecer el nombre de usuario y dirección de correo electrónico en la configuración de Git.

Para establecer el nombre de usuario y cuenta de correo, usamos los siguientes comandos:

```
$git config --global user.name "christopher"
$git config --global user.email "christopher@email.com"
```

Para comprobar que están usamos el comando:

```
$git config --list
```



```
ln -s "git-remote-http" "$execdir/$p" || \
{ test -z "" && \
    ln "$execdir/git-remote-http" "$execdir/$p" 2>/dev/null || \
    ln -s "git-remote-http" "$execdir/$p" 2>/dev/null || \
    cp "$execdir/git-remote-http" "$execdir/$p" || exit; } \
done
christopher@christopher-VirtualBox:~/git$ git --version
git version 2.25.1
christopher@christopher-VirtualBox:~/git$ git config --global user.name "Christopher"
christopher@christopher-VirtualBox:~/git$ git config --global user.email "christopher@email.com"
christopher@christopher-VirtualBox:~/git$ git config --list
user.name=Christopher
user.email=christopher@email.com
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
remote.origin.url=https://github.com/git/git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.master.remote=origin
branch.master.merge=refs/heads/master
christopher@christopher-VirtualBox:~/git$
```

bien,

5. Cambiar el editor de texto que trae por defecto Git al editor emacs.

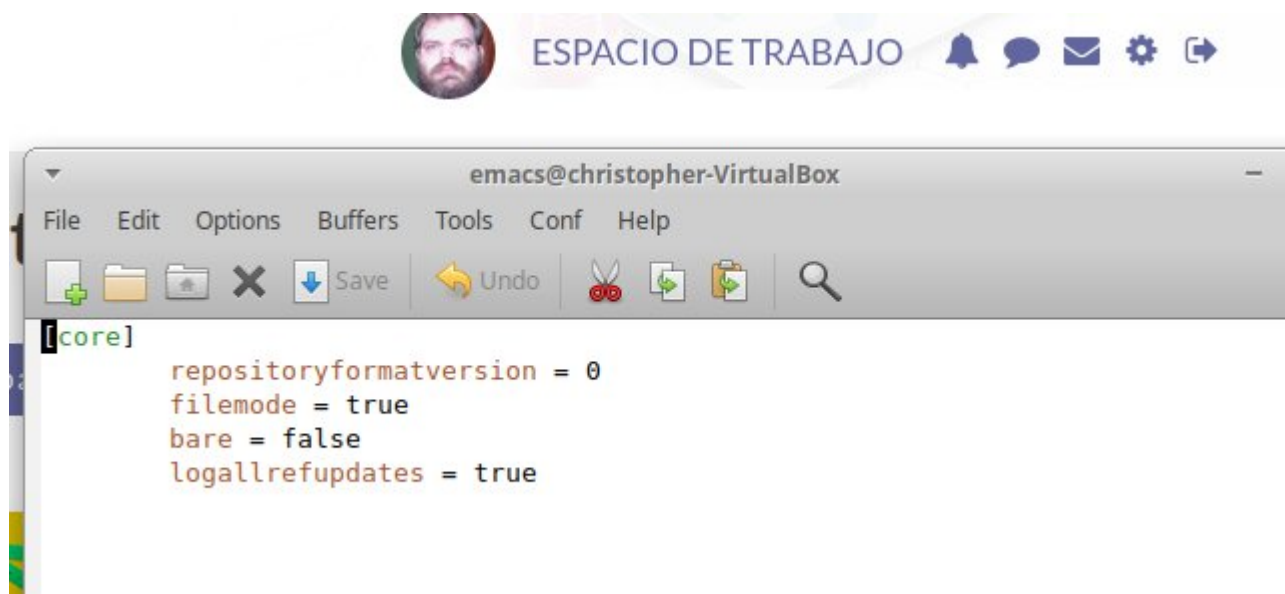
Para cambiar el editor de texto a emacs, usamos el siguiente comando:

```
$ git config -global core.editor emacs
```

Obviamente tenemos que tener emacs instalado en el equipo

Ahora, si por ejemplo abrimos el archivo de configuración de git, lo hará con emacs:

```
$ git config --global -e
```



6. Dentro de la carpeta /var/cache/git/ crear una carpeta para un nuevo proyecto denominado tarea_DAW06 e iniciar un repositorio el nuevo proyecto.

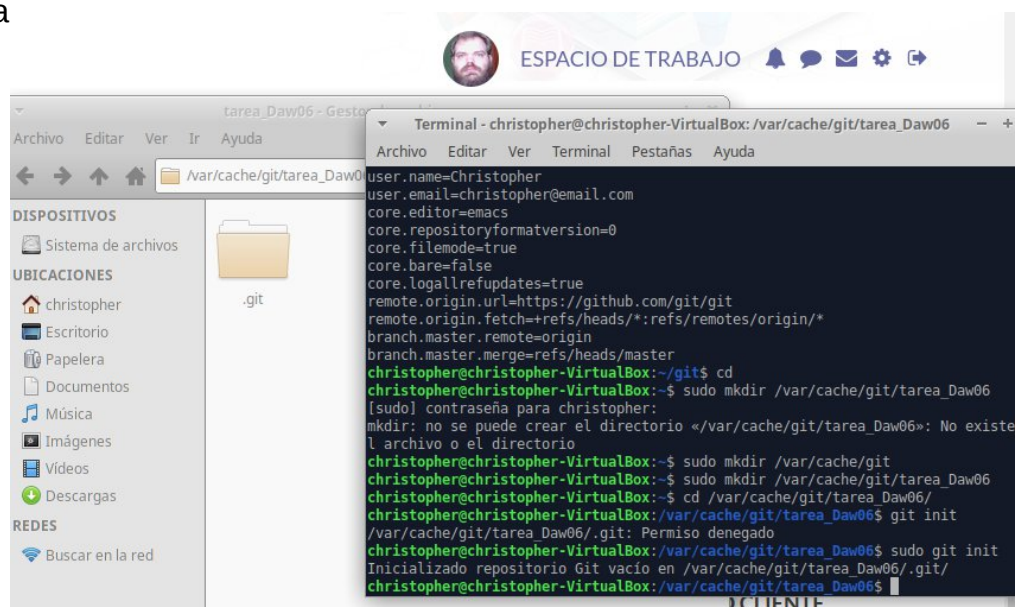
Creamos la carpeta con el comando:

```
$sudo mkdir /var/cache/git/tarea_Daw06
```

Nos dirigimos a la carpeta y escribimos:

```
$ sudo git init
```

Esto nos crea un nuevo repositorio vacío.

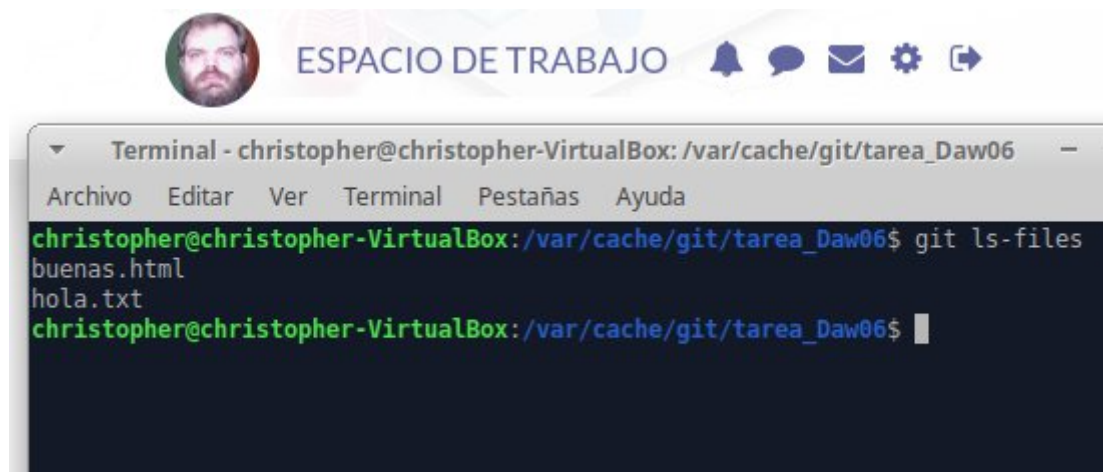


Ahora creamos un par de archivos dentro de la carpeta: **hola.txt** y **buenas.html**
Para añadirlos a nuestro repositorio usamos el comando:

```
$ sudo git add .
```

Esto añade todos los archivos que hay en la carpeta al repositorio.
Ahora escribimos el comando para ver los archivos que forman parte de nuestro repositorio:

```
$ git ls-files
```



Como podemos comprobar, esos dos archivos forman ya parte del repositorio y Git estará al tanto de los cambios que se produzcan.