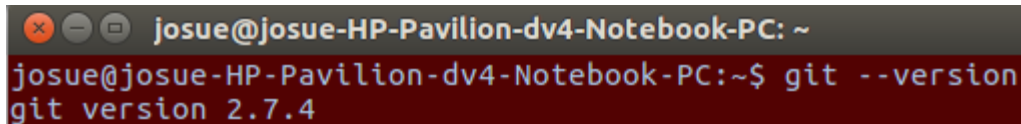


GUIA PARA EL USO DE GIT

Git es una herramienta que se usa y viene por defecto en la mayoría de los sistemas GNU/Linux, es un sistema de control de versiones que maneja un historial de versiones para que tengamos un fácil manejo de proyectos y avances.

Lo primero por hacer para empezar a usar Git es verificar que este instalado con el siguiente comando:

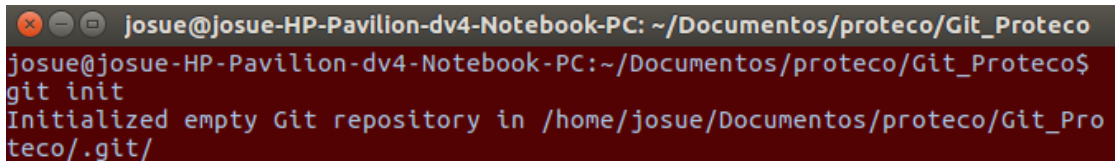


```
Josue@Josue-HP-Pavilion-dv4-Notebook-PC: ~  
josue@josue-HP-Pavilion-dv4-Notebook-PC:~$ git --version  
git version 2.7.4
```

Si al poner el comando nos arroja el mensaje de que no se reconoce el comando, significa que debemos instalar Git con el comando:

```
sudo apt-get install git-all
```

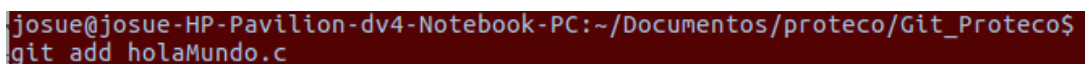
Una vez que confirmamos que tenemos instalado git, podemos crear un repositorio, para ello debemos crear un directorio que será el que usemos exclusivamente para git. Posterior a crear la carpeta, debemos estar en dicho directorio y usar el siguiente comando para la creación de un repositorio:



```
Josue@Josue-HP-Pavilion-dv4-Notebook-PC: ~/Documentos/proteco/Git_Proteco  
josue@josue-HP-Pavilion-dv4-Notebook-PC:~/Documentos/proteco/Git_Proteco$  
git init  
Initialized empty Git repository in /home/josue/Documentos/proteco/Git_Proteco/.git/
```

Como se observa se inicializó el directorio creado como repositorio git.

El siguiente paso es hacer un commit, es decir una primera versión de nuestro repositorio.



```
Josue@Josue-HP-Pavilion-dv4-Notebook-PC:~/Documentos/proteco/Git_Proteco$  
git add holaMundo.c
```

Con el comando anterior, git add "archivo", se le está indicando a git que agregue un archivo al 'index' el cual listará los archivos que se van a registrar en git. Después de añadir

un archivo al index, podemos revisar la lista de archivos que hay en el index y el procedimiento que estamos realizando con el comando:

```
josue@josue-HP-Pavilion-dv4-Notebook-PC:~/Documentos/proteco/Git_Proteco$
git status
En la rama master

Commit inicial

Cambios para hacer commit:
(use «git rm --cached <archivo>...» para sacar del stage)

nuevo archivo: holaMundo.c
```

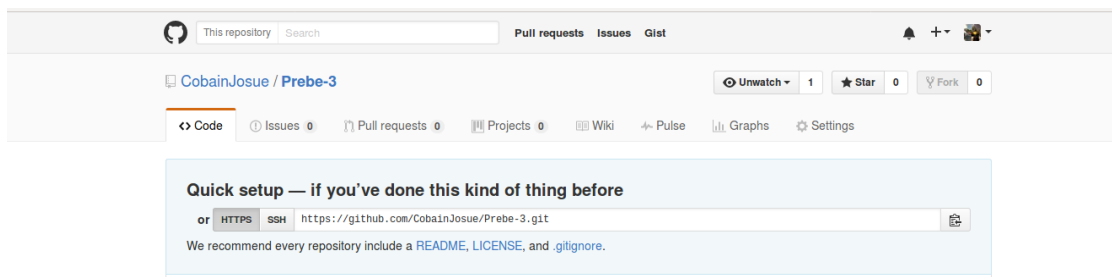
Posteriormente hacemos el commit, que es el que realmente agregara a git la version.

```
josue@josue-HP-Pavilion-dv4-Notebook-PC:~/Documentos/proteco/Git_Proteco$
git commit -m 'Primer commit :v'
[master (root-commit) 663b2b2] Primer commit :v
1 file changed, 8 insertions(+)
create mode 100644 holaMundo.c
```

Habiendo hecho esto, ya tenemos un commit en nuestro git, el paso que sigue consiste en sincronizar el repositorio creado con el que repositorio remoto, es decir el que esta en GitHub, para esto necesitamos crear un repositorio en nuestra cuenta de GitHub, es decir en la pagina, mediante la opcion de nuevo repositorio, habiendo hecho esto, tenemos que sincronizar el repositorio que creamos con el existente en GitHub.

git remote add origin https://urldeturepo.git

El url se obtiene desde GitHub entrando al repositorio:



Despues de esto tenemos que notificarle al repositorio en GitHub que hemos agregado un nuevo commit en Git, para que se registre la version de manera remota tambien y no solo local, esto mediante el comando:

```
josue@josue-HP-Pavilion-dv4-Notebook-PC:~/Documentos/proteco/Git_Proteco$
git push origin master
Username for 'https://github.com': CobainJosue
Password for 'https://CobainJosue@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 303 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/CobainJosue/Prebe-3.git
* [new branch]      master -> master
```

Para la creacion de la llave SSH.

Las claves SSH también conocidas como llaves SSH, son una manera de identificar las computadoras de confianza, sin tener que ingresar una contraseña.

Las claves SSH se deben generar para cada usuario. Luego de realizar la generación obtendremos una clave privada y una clave pública.

Para crear una llave SSH debemos usar el comando:

```
josue@josue-HP-Pavilion-dv4-Notebook-PC:~/Documentos/proteco/Git_Proteco$ ssh-keygen -t rsa -C "chivasagrada@live.com.mx"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/josue/.ssh/id_rsa):
Created directory '/home/josue/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/josue/.ssh/id_rsa.
Your public key has been saved in /home/josue/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:oDjf3pXgg+0xVgVSTyqtmp4EobT7cqMQEp9td7zrSao chivasagrada@live.com.m
x
The key's randomart image is:
+---[RSA 2048]---+
|                . . .                |
|               ...+                  |
|...    0.O..                        |
|.+. = . = .                        |
|+. * + o S.                        |
|... = = +.O .                      |
|..  = = . = O                      |
|... = +.B =                        |
|.+.Eo=. =                          |
+---[SHA256]-----+
```

Esto quiere decir que ya generamos nuestras claves SSH correctamente. Como podemos apreciar, la clave privada a sido guardada en: /home/josue/.ssh/id_rsa y la clave pública en /home/josue/.ssh/id_rsa.pub.

Entonces ya podemos agregar nuestra nueva clave SSH al ssh-agent, escribiendo lo siguiente en nuestro terminal:

```
josue@josue-HP-Pavilion-dv4-Notebook-PC:~/Documentos/proteco/Git_Proteco$ eval "$(ssh-agent -s)" && ssh-add ~/.ssh/id_rsa
Agent pid 12821
Identity added: /home/josue/.ssh/id_rsa (/home/josue/.ssh/id_rsa)
```

Con esto hemos agregado nuestra clave al pequeño programa ssh-agent para que administre nuestras llaves por nosotros. Lo que quiere decir es que tu sólo ingresaras tu clave una sola vez, y después de eso, ssh-agent mantendrá la clave en memoria y cada vez

que sea requerida ssh-agent la entregará por nosotros.

Agregando nuestra clave pública a Github

Primero instalamos xclip, para poder copiar el contenido de nuestra clave pública por consola.

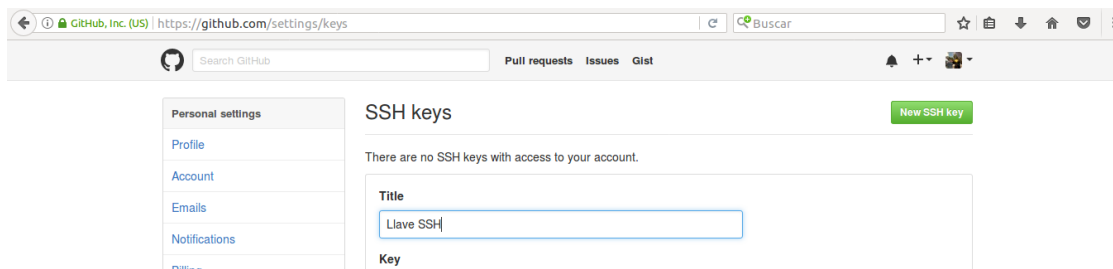
```
sudo apt-get install xclip
```

Luego seleccionamos y copiamos el contenido de nuestra clave pública con la siguiente línea de código:

```
xclip -sel clip < ~/.ssh/id_rsa.pub
```

Con esto tenemos nuestra clave pública en el portapapeles, entonces ingresamos a Github, y En la sección Settings elegimos la opción SSH Keys e ingresamos.

Hacemos click en el botón Add SSH Key(Agrega tu clave SSH).



En el campo Title escribimos algo descriptivo sobre esta clave y en el campo Key pegamos el contenido del archivo ~/.ssh/id_rsa.pub(contenido que tenemos en nuestro portapapeles listo para pegar).

Validando nuestra clave SSH con Github.

Para verificar que esta conectada correctamente:

```
josue@josue-HP-Pavilion-dv4-Notebook-PC:~/Documentos/proteco/Git_Proteco$  
ssh -T git@github.com  
The authenticity of host 'github.com (192.30.253.112)' can't be established.  
RSA key fingerprint is SHA256:nThbg6kXUpJWGL7E1IGOCspRomTxdCARLviKw6E5SY8.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added 'github.com,192.30.253.112' (RSA) to the list of known hosts.  
Hi CobainJosue! You've successfully authenticated, but GitHub does not provide shell access.
```

Con esto ya tenemos registrada nuestra clave pública en github, lo cual nos permitirá subir cambios al servidor de Github creando una conexión segura.