

4) Guía de Github



github

Versión Guía 0.16

®DesafioLatam

4) Guía de Github

Configurando un remote

- Github
- SSH

Creando una página web en github pages

El contenido de este documento es una recopilación de la documentación de Git disponible en su página oficial, de la documentación de Github y la incorporación del material de clases de desafiolatam

Objetivos

1. Aprender a trabajar con GIT utilizando los remotes
2. Aprender a hacer los pull y push
3. Aprender a descargar proyectos desde github
4. Descubrir como subir nuestra primera página web a github

Configurando un remote

El remote es el punto a donde nosotros vamos a enviar nuestros cambios, podemos tener un servidor para esto o podemos ocupar github o bitbucket.

Github

Github es una red social de códigos, en cierto sentido se parece a dropbox y sirve para guardar nuestros repositorios de Git.

Para utilizar Github primero debemos crearnos una cuenta, eso lo podemos hacer llenando los datos del

formulario, pero configurarla es un poco más complejo porque necesitamos agregar nuestra clave pública de ssh

SSH

Para subir y bajar archivos de Github lo haremos ocupando ssh, este protocolo ocupa encriptación de clave asimétrica, o sea se necesitan dos claves, una para encriptar y una para desencriptar, estas claves son archivos y estan dentro de nuestro computador usualmente dentro de la carpeta `$HOME/.ssh`

Entramos a la carpeta

```
1 | cd ~/.ssh
```

si la carpeta no existe tenemos que crearla.

```
1 | mkdir ~/.ssh
```

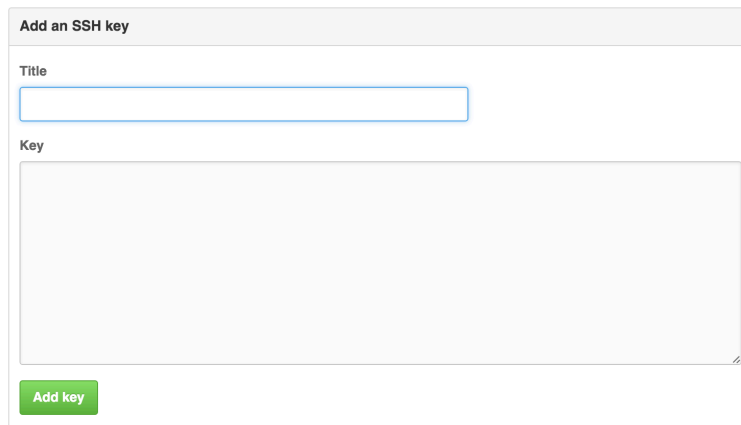
luego con ls podemos revisar dentro de la carpeta si existe un archivo sin extensión y otro que se llame igual que tenga extensión .pub, si es así ese es nuestro juego de claves, si no, podemos generar uno con:

```
1 | ssh-keygen -t rsa
```

Luego tenemos que copiar y pegar el archivo en settings

de github en la sección de ssh keys.

Dentro de esta pantalla veremos la opción para agregar una clave ssh.



The screenshot shows a web form titled "Add an SSH key". It contains two input fields: "Title" and "Key". The "Title" field is a single-line text input. The "Key" field is a multi-line text area. Below the "Key" field is a green button with the text "Add key".

Para que funcione la llave debemos copiarla y pegarla sin añadir espacios ni otros textos. El título puede ser el nombre de tu computador.

Podemos probar que la conexión fue exitosa con:

```
1 | ssh -T git@github.com
```

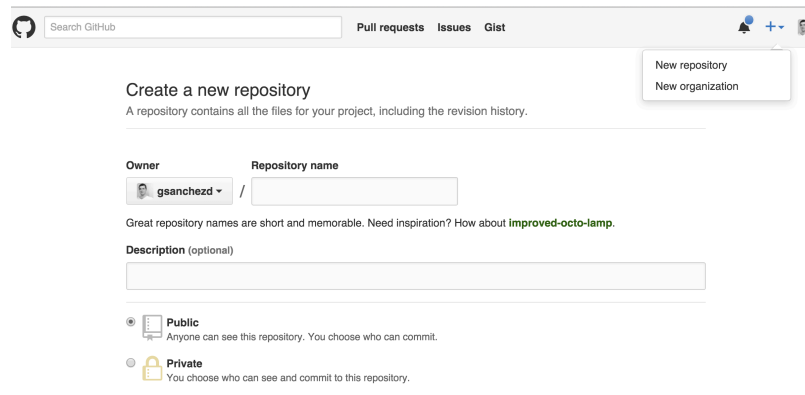
Si obtenemos un mensaje del tipo:

Hi gsanchezd! You've successfully authenticated, but
GitHub does not provide shell access.

Significa que nuestra juego de claves funciona.

Creando un repositorio

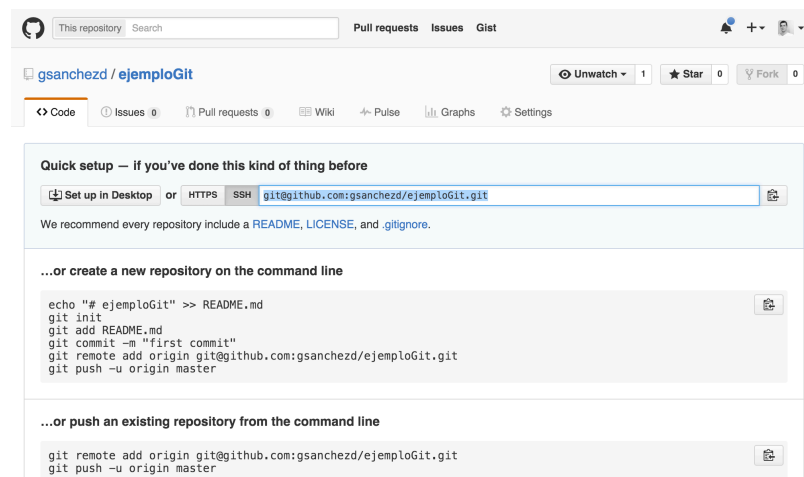
Con la cuenta creada y configurada el siguiente paso es crear un repositorio nuevo, para ambas formas tenemos que ir a la opción de crear nuevo repositorio



The screenshot shows the GitHub interface for creating a new repository. At the top, there's a search bar and navigation links for 'Pull requests', 'Issues', and 'Gist'. A dropdown menu is open, showing 'New repository' and 'New organization'. The main heading is 'Create a new repository' with a subtext: 'A repository contains all the files for your project, including the revision history.' Below this, there are two input fields: 'Owner' (set to 'gsanchezd') and 'Repository name'. A note says: 'Great repository names are short and memorable. Need inspiration? How about **improved-octo-lamp**.' There's a 'Description (optional)' text area. At the bottom, there are two radio buttons for visibility: 'Public' (selected) and 'Private'. The 'Public' option has a subtext: 'Anyone can see this repository. You choose who can commit.' The 'Private' option has a subtext: 'You choose who can see and commit to this repository.'

Podemos crear repositorios públicos o privados pero para los privados necesitamos tener una cuenta pagada de github, en bitbucket podemos tener repositorios privados de forma gratuita, pero siempre es bueno y ayuda al curriculum contribuir al opensource.

Al llenar los campos podremos ver el siguiente página.



The screenshot shows the GitHub repository page for 'gsanchezd / ejemploGit'. At the top, there's a search bar and navigation links for 'Pull requests', 'Issues', and 'Gist'. The repository name is 'gsanchezd / ejemploGit'. There are buttons for 'Unwatch', 'Star', and 'Fork'. Below this, there are tabs for 'Code', 'Issues', 'Pull requests', 'Wiki', 'Pulse', 'Graphs', and 'Settings'. The 'Code' tab is selected. The main content area shows 'Quick setup — if you've done this kind of thing before'. It has two options: 'Set up in Desktop' and 'HTTPS SSH'. The 'SSH' option is selected, and the URL 'git@github.com:gsanchezd/ejemploGit.git' is shown. Below this, there's a note: 'We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).' There are two sections: '...or create a new repository on the command line' and '...or push an existing repository from the command line'. Each section has a code block with the following commands:

```
echo "# ejemploGit" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:gsanchezd/ejemploGit.git
git push -u origin master
```

y esta página nos muestra las dos formas, la primera es crear un repositorio vacío en tu computador y enviarlo, la segunda es ocupar un repositorio existente, las únicas

líneas nuevas que vemos aquí son:

Al copiar la dirección revisar el protocolo, es mejor ocupar ssh en vez de http para que no te pida la clave cada vez que intentas acceder.

```
1 | git remote add origin git@github.com:gsanchezd/ejemploGit.git
2 | git push -u origin master
```

git remote add agrega un punto remoto, en este caso a ese punto le vamos a llamar origin, aunque perfectamente lo pudimos haber llamado github o de cualquier otra forma, finalmente ponemos el punto de destino.

Con el comando `git remote -v` podemos ver todos los puntos remotos que tenemos configurados, si hicimos todo bien obtendremos:

```
origin  git@github.com:gsanchezd/ejemploGit.git
origin  git@github.com:gsanchezd/ejemploGit.git
```

Con un punto remoto configurado podemos enviar nuestros cambios ya confirmados y recuperar cambios confirmados de ese repositorio a través de tres instrucciones, push, fetch y pull.

git push origin master envía los cambios al punto remoto.
git fetch origin master trae los cambios en una rama nueva.

git pull origin master trae los cambios a la rama y hace el merge.

Ahora estos comandos no tienen porque usarse sobre la rama principal, podemos ocupar otras ramas.

Creando una página web en github pages

Si tienes cuenta en github tienes derecho a tener una página web en el dominio github.io, pero no puedes subir esa página por FTP o cpanel, debes subir la página a github a un repositorio especial.

Para hacerlo debemos crear un repositorio que se llame igual que tu cuenta.

Si el repositorio se llama distinto a tu usuario esto no funcionará

Con el repositorio creado tenemos que agregar la página web index.html, no es necesario que tenga código HTML, puede simplemente decir hola

Ahora hay que agregar el archivo a staging (add), confirmar los cambios (commit) y enviar los cambios a github (push)

Preguntas

1. ¿Cuál es la diferencia entre pull y fetch?
2. ¿A qué se debe el error Repository not found al hacer push?
3. ¿A qué se debe el error Updates were rejected because the tip of your current branch is behind hint:

its remote counterpart.?

4. ¿Cómo se puede cambiar el remote?

Comando	Explicación
git remote -v	Muestra todos los remotes disponibles
git remote rm origin	Borra el remote origin
git remote add origin git@github.com:gsanchezd/ejemploGit.git	Agrega el remote origin con punto de destino git@github.com:gsanchezd/ejemploGit.git
git push origin master	Envía los cambios dentro de master al remote origin
git push origin development:development	Envía los cambios dentro de la rama development al remote origin en la rama development
git pull origin master	

