04_repositorios_remotos

October 15, 2020

1 Repositorios remotos.

Git tiene la particularidad de poder gestionar repositorios distribuidos. Es decir, que puede realizar operaciones en repositorios locales y repositorios remotos.

En vista de que los repostorios de *Git* son directorios comunes y corrientes, lo único que se debe de hacer es crear las condiciones para que un usuario pueda acceder de forma remota a dicho directorio.

De este modo, la forma en la que un usuario accede a el repositorio no depende de *Git* sino del servidor que lo contiene.

1.1 Protocolos soportados por *Git*.

Git fue diseñado para poder realizar conexiones mediante los protocolos:

- *SSH*.
- *HTTP/HTTPS*.

1.2 Autenticación soportada por *Git*.

Del mismo modo, *Git* puede utilizar los siguientes esquemas de autenticación.

- Los esquemas de autenticación de *SSH*.
- Los esquemas de autenticación de *HTTP*.
- Autenticación federada con *OAuth*.

1.3 El comando `git clone`.

El comando `git clone` permite traer el contenido un repositoro remoto a un repositorio local.

`git clone <URL> <ruta>`

Donde:

- `<URL>` es la ruta en la que se encuentra el repositorio remoto.
- `<ruta>` es la ruta del nuevo repositorio local.

Ejemplo:

- La siguiente celda creará una copia del repositorio remoto localizado en https://github.com/PythonistaIO/prueba en el subdirectorio `experimental`.

[ ]:
```
git clone https://github.com/PythonistaIO/prueba.git experimental
```

[ ]:
```
cd experimental
```

[ ]:
```
ls -a
```

1.4 El comando `git remote`.

El comando `git remote` permite realizar operaciones d gestión de repositorios remotos.

La referencia al comando `git remote` puede ser consultada en:

https://git-scm.com/docs/git-remote

1.4.1 Listado de repositorios remotos.

La sintaxis para enlistar los repositorios remotos ligados a un repositorio local.
`git remote show`.

Nota: Como regla general, el repositorio remoto principal tiene como nombre `origin`.

[ ]:
```
git remote show
```

1.5 Configuración de los repositorios remotos en el ámbito `--local`.

La configuración de un repositorio remoto es guardada en el ámbito `--local` en un registro con la estructura.

`remote.<nombre>`

Donde:

- `<nombre>` es el nombre dado al repositorio remoto.

Ejemplo:

- La siguiente celda traerá la configuración del ámbito `--local` del repositorio actual. Se podrá apreciar que el registro `remote.origin` corresponde a la configuración del reposotorio remoto `origin`.

[ ]:
```
git config --local --list
```

1.5.1 Adición de un repositorio remoto.

Para relacionar unrepositorio remoto a un repositorio local se usa la siguiente sintaxis.

`git remote add <nombre> <URL>`

Donde:

- `<nombre>` es el nombre que se le dará al repositorio remoto.
- `<URL>` es la URL del repositorio remoto.

Ejemplo:

- Se relacionará al repositorio localizado en la URL `git@github.com:PythonistaIO/prueba.git` con el nombre `alterno`

[ ]:
```
git remote add alterno git@github.com:PythonistaIO/prueba.git
```

[ ]:
```
git remote show
```

1.6 Acceso a la rama de un repositorio remoto.

Git permite acceder a las ramas de un repositorio remoto utilizando el comando `git checkout`, Sin embargo, ya que dichas ramas no pertenecen al repositorio local, la rama remota a la que se acceda estará en estado `'detached HEAD'`.

1.6.1 Referencia a la rama de un repositorio remoto.

Para hacer referencia a un archivo remoto se utiliza la siguiente estructura:

`<remoto>/<rama>`

Donde:

- `<remoto>` es el nombre del repositorio remoto.
- `<rama>` es el nombre de la rama en el repositorio remoto.

Nota: Es muy importante hacer notar que el acceso a un repositorio remoto no se realiza mediante una conexión continua, por lo que es necesario actualizar la información contenida en dichos repositorios mediante los comandos `git fetch` y `git pull`.

Ejemplo:

- La siguiente celda acccederá al repositorio `origin/master`.

[ ]:
```
git checkout origin/master
```

[ ]:
```
ls
```

- La siguiente celda regresará al repositorio `master`.

[ ]:
```
git checkout master
```

[ ]:
```
touch archivo-local
```

[ ]:
```
git add --all
```

[ ]:
```
git commit -m "commit local"
```

- La siguiente celda desplegará las diferencias entre las ramas `origin/master` y `master`.

[ ]:
```
git diff origin/master master
```

[ ]:
```
git status
```

1.7 Actualización de un repositorio local con respecto a un repositorio remoto.

Es posible traer las actualizaciones de un repositorio remoto mediante dos comandos.

- `git fetch`
- `git pull`

1.7.1 El comando `git fetch`.

Este comando actualiza los cambios del repositorio remoto.

`git fetch <remoto>.<rama>`

- Donde `<remoto>` corresponde al nombre del repositorio remoto. El valor por defecto es `origin`.
- – Donde `<rama>` corresponde a la etiqueta de la rama en el repositorio remoto. El valor por defecto es la rama actual del repositorio local.

La documentación de referencia del comando `git fetch` está disponible en:

https://git-scm.com/docs/git-fetch

Ejemplo:

- La siguiente celda ejecutará el comado `git fetch` para `origin/naster`.

[ ]:
```
git fetch
```

[ ]:
```
git status
```

[ ]:
```
git diff master origin/master
```

[ ]:
```
git checkout master
```

[ ]:
```
ls
```

1.7.2 El comando `git pull`.

Este comando es similar a `git getch`, pero realiza un `merge` de forma automática.

[ ]:
```
git pull
```

[ ]:
```
git commit -m "cambios remoto"
```

[ ]:
```
git merge
```

[ ]:
```
git checkout origin/master
```

[ ]:
```
git checkout master
```

1.8 El comando `git push`.

El comando `git push` permite enviar los cambios de una rama de un repositorio local a un repostiorio remoto.

Nota: Es importante que el usuario tenga los permisos necesarios para realizar esta acción.

```
git push <remoto> <rama>
```

Donde: * `<remoto>` es el nombre de un repositorio remoto. * `<rama>` es la rama que se quiere afectar en el repositorio remoto.

[ ]:
```
git push
```

1.9 El servicio *GitHub*.

Github Es un sitio que ofrece múltiples servicios para desarrolladores de software tales como:

- Repositorios remotos de *Git* públicos y privados.
- Wikis.
- Gists.
- Gestión de proyectos.
- Automatización de creación de artefactos.
- Herramientas de DevOps.
- Servicios de nivel "Enterprise".

1.9.1 Forks.

Github permite realizar copias del repositorio de un usuario al repositorio de otro usuario.

1.9.2 Pull requests.

A partir de los *forks* es posible colaborar con un proyecto clonado (fork), mediante *Pull Requests*, en las que un usuario permite compartir y discutir los cambios ehchos en su repositorio clonado a otro usuario para que incluya dichos cambios a su propio repositorio.