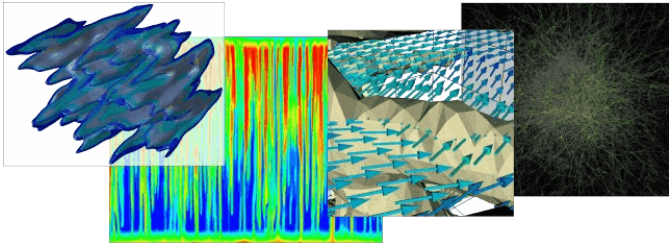
Introducción a git

Una sistema de control de versiones descentralizado



Practique los ejemplos en Gitpod

Open in Gitpod

Disponible en

¡Únete al grupo en Telegram!

Vídeos recomendados en `diode.zone`

- Introducción, clonar un repositorio en Windows 10 con Visual Studio Community 2019.
- Clonar un repositorio, añadir cambios, hacer confirmación y subir objetos a GitHub.
- ¿Qué es un control de versiones?

Instalación de git

En la distribución Arch Linux, el binario se instala con la siguiente directiva

```
[user@host somedir]$ sudo pacman -Syu git
```

Si ya tiene instalado `dune-common`, entonces ya tendrá instalado `git` como dependencia.

En macOS instale `homebrew` y a continuación escriba en el emulador de terminal `brew install git`.
Para Windows 10/11, siga este tutorial.

¿Qué características tiene un control de versiones?

Según la entrada de Wikipedia, un sistema de control de versiones debe proporcionar:

- *Mecanismo de almacenamiento de los elementos que deba gestionar.*
- *Posibilidad de realizar cambios sobre los elementos almacenados.*
- *Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos (normalmente pudiendo volver o extraer un estado anterior del producto).*

(W)

Autentificación

Desde el 13 de agosto del 2021 [1], GitHub no aceptará la autentificación vía contraseña, por lo que podemos continuar vía token o claves SSH. Optaremos por la última opción [2].
En la distribución Arch Linux, el binario se instala con la siguiente directiva

```
[user@host somedir]$ sudo pacman -Syu openssh
```

Es un port del software SSH (Secure Shell) del proyecto OpenBSD bajo la licencia BSD.

[1] https://github.blog/2020-12-15-token-authentication-requirements-for-git-operations
[2] https://docs.github.com/es/github/authenticating-to-github/connecting-to-github-with-ssh

```
[user@host somedir]$ git config --list
```

En el directorio ~/.ssh tendrá id_ed25519.pub id_ed25519 config known_hosts. man ssh.

Utilizando el sistema de firma de clave pública Ed25519

Abra un emulador de terminal y escriba

```
[user@host somedir]$ ssh-keygen -t ed25519 -C "<comentario>"
```

También podría optar el cifrado RSA, de la siguiente manera

```
[user@host somedir]$ ssh-keygen -t rsa -b 4096 -C "<comentario>"
```

En caso de que no desee fijar una contraseña continue con la tecla Intro hasta acabar las preguntas. Si luego desea que no le pida la contraseña en cada intento, agregue en el archivo `~/.gitconfig` lo siguiente:

```
[credential]
        helper = cache --timeout=3600
```

Como paso final, copie su clave pública (.pub) en la siguiente caja

¿Por qué usar un control de versiones?

Si no tiene un SCV, las cosas se salen de control muy rápidamente.
Las razones adecuadas son:

Colaboración el SCV permite que un equipo de personas trabaje en el mismo proyecto al mismo tiempo.

Almacenamiento de versiones el SCV gestiona todas las versiones de todos los archivos, los almacena, los nombra y puede recuperarlos.

Seguimiento de los cambios el SCV registra con precisión lo que se cambió y se debe dar una razón para el cambio.

Restaurar cambios y rutas de regresión el SCV permite restaurar archivos individuales o grupos de archivos a una versión anterior.

Terminología 📖

Repositorio Es una base de datos que contiene toda la información acerca de las versiones de los archivos en un proyecto, ubicado en una carpeta oculta `.git`.

Confirmación Instantánea global de todos los archivos del proyecto con descripción de los cambios con respecto a la versión anterior.

Rama Secuencia de commits que describe una rama (línea) de desarrollo. Un repositorio puede contener más de uno. La rama estándar se llama `main` o `master`.

Tag Nombre persistente (identificador) para un commit, por ejemplo, cuando hace un lanzamiento público.

Es una buena idea presentarse a Git con su nombre y dirección de correo electrónico pública antes de realizar cualquier operación. La forma más sencilla de hacerlo es:

```
[user@host somedir]$ git config --global user.name <user>
[user@host somedir]$ git config --global user.email <user@example.com>
```

Remplace `<user>` por el usuario de GitHub (sin **@**) y `<user@example.com>` por el correo registrado en GitHub.

Confirmaciones 

Una confirmación contiene

- una instantánea de todos los archivos.
- la fecha de creación.
- el nombre y contenido con respecto al autor de los cambios.
- una lista de cambios.
- una lista de referencias a confirmaciones principales.

Las confirmaciones son identificadas usando un hash (suma de verificación) de su contenido, por ejemplo `5cd006a1c044b786ea8196636cd0b62e296dbbe5`.
Los hash de confirmación pueden abreviarse siempre que la abreviatura sea única.
Las confirmaciones no pueden cambiar: contenido diferente ⇒ valor hash diferente.

Ramas 

- Una rama es una línea de desarrollo dentro de un repositorio.
- Los repositorios pueden contener un número arbitrario de ramas.
- `git status` puede decirle en qué rama se encuentra.
- Una rama siempre apunta a un `commit`.
- La creación de una nueva confirmación guarda la confirmación actual como principal y la rama luego apunta la nueva confirmación.

Varios repositorios 

- **git** puede sincronizar cambios entre repositorios.
- Los repositorios adicionales se llaman remotos.
- Las ramas de un repositorio remoto tienen el nombre del repositorio como prefijo.
- Puede clonar un repositorio remoto para obtener su contenido.
- No se pueden ver ramas de repositorios remotos directamente.
- En la primera revisión, git crea una rama de seguimiento.
- Los nuevos cambios se pueden descargar y combinar usando `git pull`.