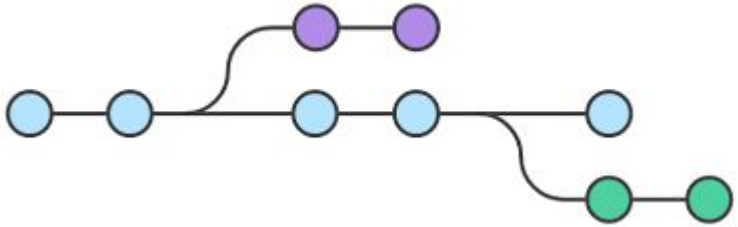
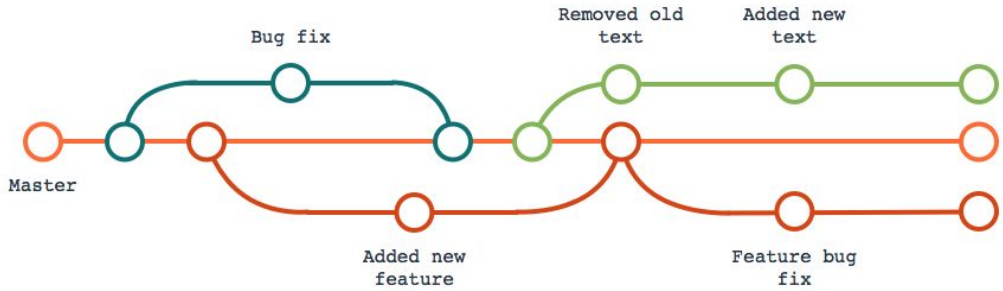
Herramientas de gestión de código abierto: el caso GIT

Paulo Cordero Villalobos
Jason Alvarado Rojas



¿Qué son?

- Las herramientas de gestión de código son utilizadas para darle seguimiento a las modificaciones de un proyecto almacenado en un repositorio de código fuente.
- Se genera un historial de cambios que ayuda a solucionar posibles conflictos al fusionar actualizaciones de múltiples contribuidores.



Importancia

- Evita conflictos entre código escrito por varios desarrolladores en un mismo proyecto.
- Se logra trabajar de manera más ordenada permitiendo un mejor flujo de trabajo.
- Permite controlar la evolución del proyecto fácilmente y observar las distintas versiones que se han hecho a lo largo del tiempo

Ejemplos de sistemas de control de versiones

git

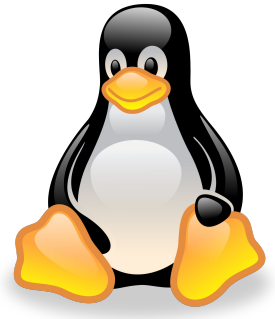
ATLASSIAN

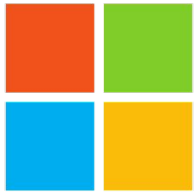
Bitbucket

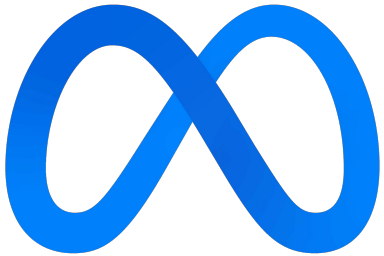
git

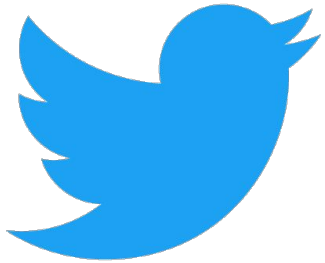
GitHub

Ejemplos de empresas que utilizan Git

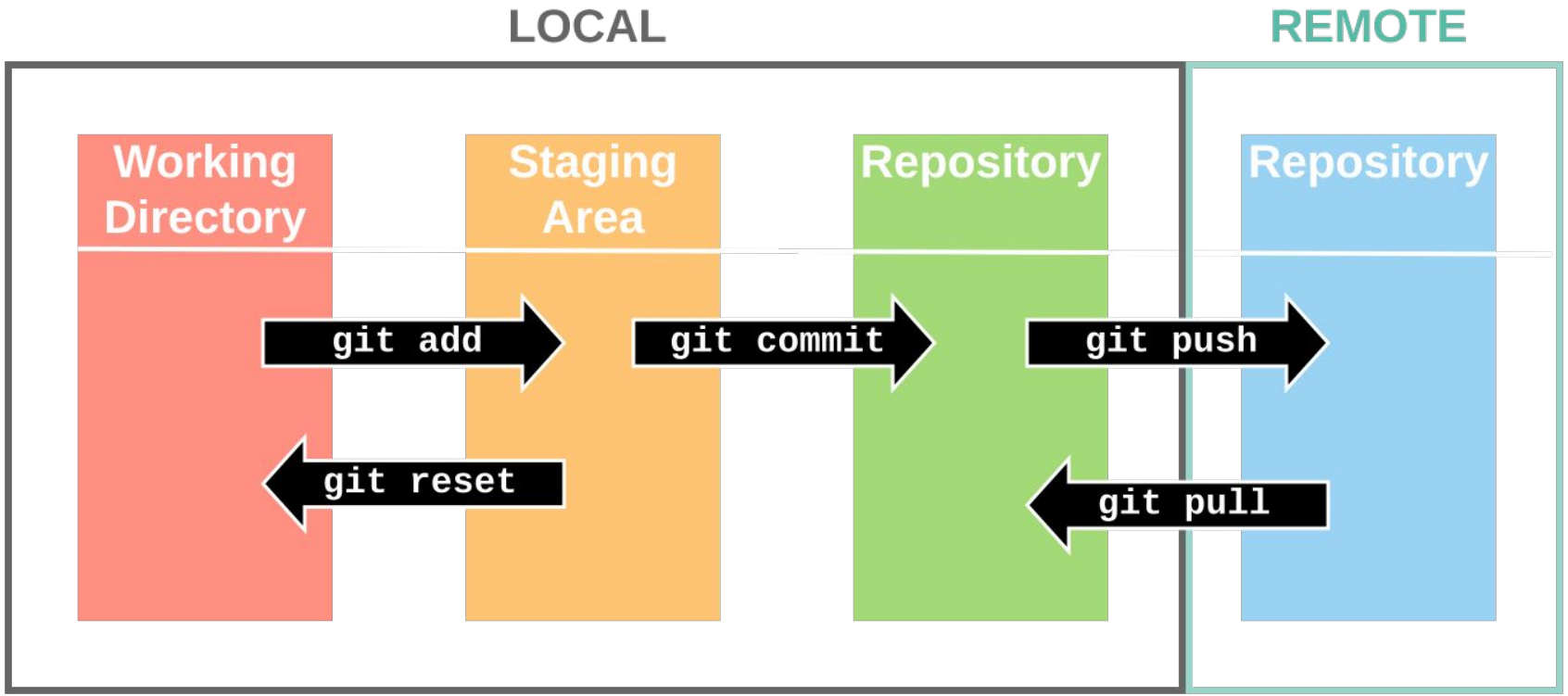








¿Cómo funciona Git?



Creando clave SSH

Abrimos la terminal y utilizamos los siguientes comandos:

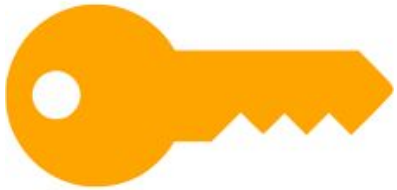
```
$  ssh-keygen -t rsa
```

Nos pedirá una ruta en la que guardar la llave creada.

Luego ingresamos un passphrase, y una vez creada la llave, en la terminal usamos el comando:

```
$ ssh-add
```

¿Cómo vincular mi llave a mi cuenta de Github?



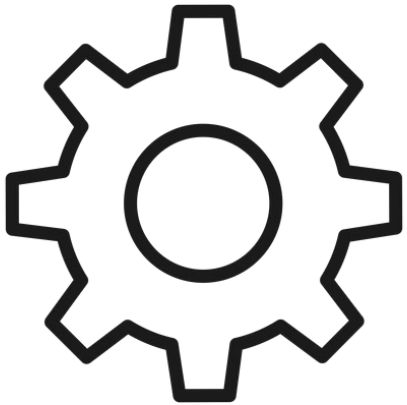
-Carpeta oculta (**.ssh**) en la carpeta home

-Copiar contenido del archivo **.pub**

-Agregar llave a nuestra cuenta de Github mediante **Settings**>**SSH and GPG keys**>**New SSH key**

Setup de git en el equipo

```
$ git config --global user.name “usuario”
$ git config --global user.email “usuario@correo.com”
```



Comandos básicos

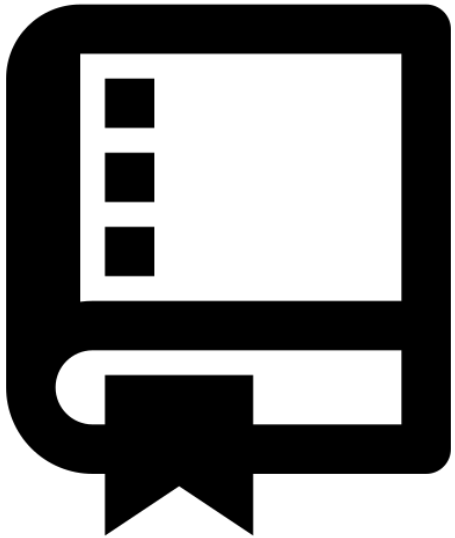
Crear un repositorio

```
$ git init
```

Crear un repositorio local

```
$ git clone [url]
```

Clonar un repositorio de GitHub



Comandos de uso frecuente localmente

```
$ git status
```

Nos informa el estado de nuestro working directory con respecto al repositorio.

```
$ git add archivo
```

Agrega el archivo modificado indicado, al staging area.

```
$ git commit -m Descripción de los cambios
```

Agrega todo lo que se encuentre en el staging area, al repositorio local. Entre las comillas se especifica de forma general, los cambios hechos por el usuario.

Comandos de uso frecuente localmente

Branch & Merge

`$ git branch`

Listar las ramas.

`$ git branch branch-name`

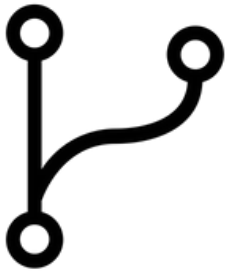
Crear una nueva rama.

`$ git checkout branch-name`

Cambiar a una rama distinta.

`$ git merge branch-name`

Combinar la rama especificada con la rama actual.



Comandos de uso frecuente localmente

```
$ git rm archivo
```

Remueve el archivo indicado, del staging area.

```
$ git log
```

Muestra el historial de commits hechos al branch actual.

```
$ git diff archivo
```

Muestra los cambios actuales entre el working directory y el staging area.

```
$ git diff --staged
```

Muestra los cambios actuales entre el staging area y el repositorio.

Comandos de uso frecuente para el repositorio remoto

```
$ git fetch
```

Extraer los cambios del repositorio remoto pero sin actualizar las ramas

```
$ git pull
```

Extraer los cambios del repositorio remoto agregando las modificaciones a las ramas

```
$ git push
```

Sube los cambios del repositorio local al repositorio remoto

¿Preguntas antes del mini-taller?