Introducción a Git

Parte 1: trabajando solo

Temas

- Definiciones y funcionamiento
- Clonar un repositorio existente
- Crear un repositorio nuevo
- Contribuir modificaciones a un repositorio
- Recuperar una versión anterior de un archivo
- Definición de *branch*

1.
¿Qué es Git?
¿Para qué sirve?

Origen, definiciones y funcionamiento

¿Qué es?

Git es un **control de versiones distribuido**.

Un **control de versiones** es un sistema que permite crear, modificar y eliminar archivos registrando cada uno de esos cambios en un historial, de manera tal que luego podamos revertir los cambios realizados, comparar distintas versiones del archivo en el que estamos trabajando y cambiar entre ellas.

Que sea **distribuido** implica que las copias de los archivos y su historial de cambios son guardados en un servidor remoto y, a su vez, cada miembro del equipo puede tener una copia de ellos guardada localmente.

Creado por

Linus Torvalds

En el año 2005

https://www.cs.helsinki.fi/u/torvalds/

Git (jerga) es un insulto en inglés que denota una persona desagradable, tonta, incompetente, molesta, senil, anciana o infantil.

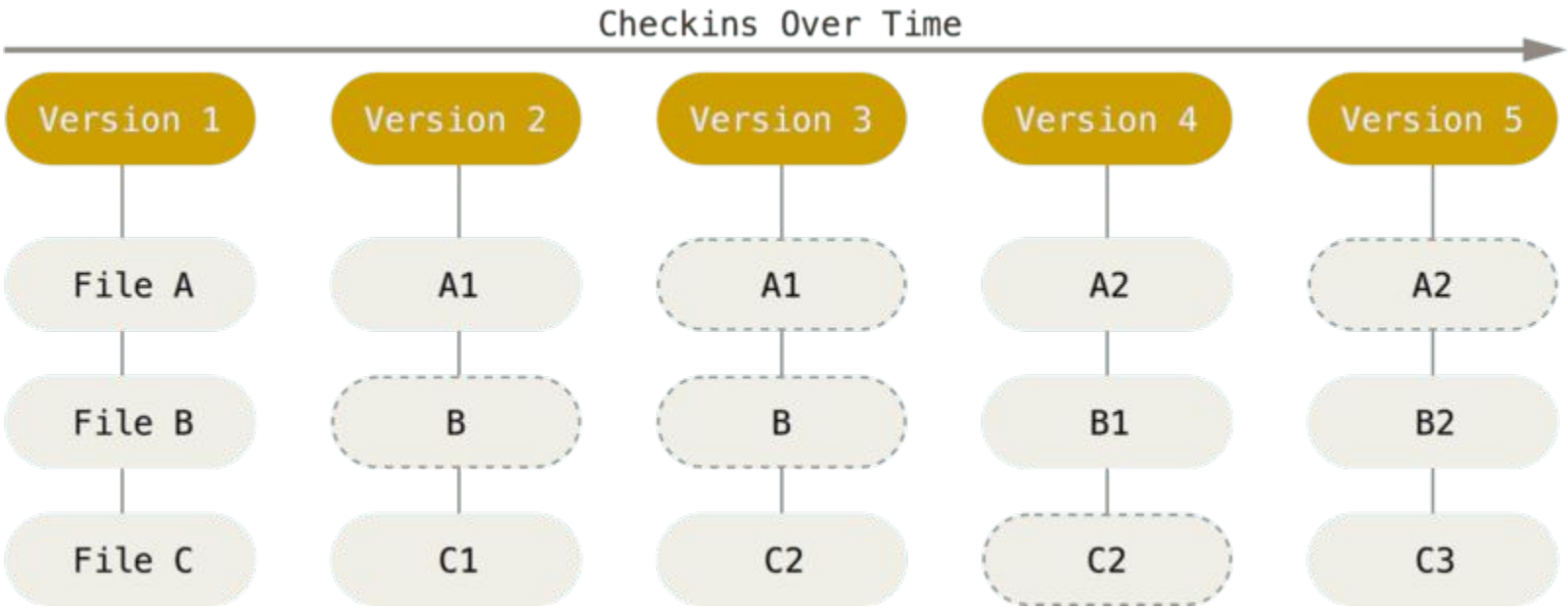
¿Para qué sirve?

Tener una copia en la nube o servidor externo

Resolver problemas :)

Recuperar una versión anterior

¿Cómo funciona?



¿Cómo funciona?

Estado de los archivos

modificado (*modified*)

Significa que has modificado el archivo pero todavía no lo has preparado para su confirmación, no los has añadido al índice

preparado (*staged*)

Significa que has marcado un archivo modificado en su versión actual para que sea almacenado en tu próxima confirmación

confirmado (committed)

Significa que los datos están almacenados de manera segura

¿Cómo funciona?

Secciones de un proyecto

Directorio de Git (*.git directory*)

Donde se almacenan los metadatos y la base de datos de objetos para nuestro proyecto. Es la parte más importante de Git, y es lo que se copia cuando clonamos un repositorio desde la nube a nuestra computadora

Directorio de trabajo (*working directory*)

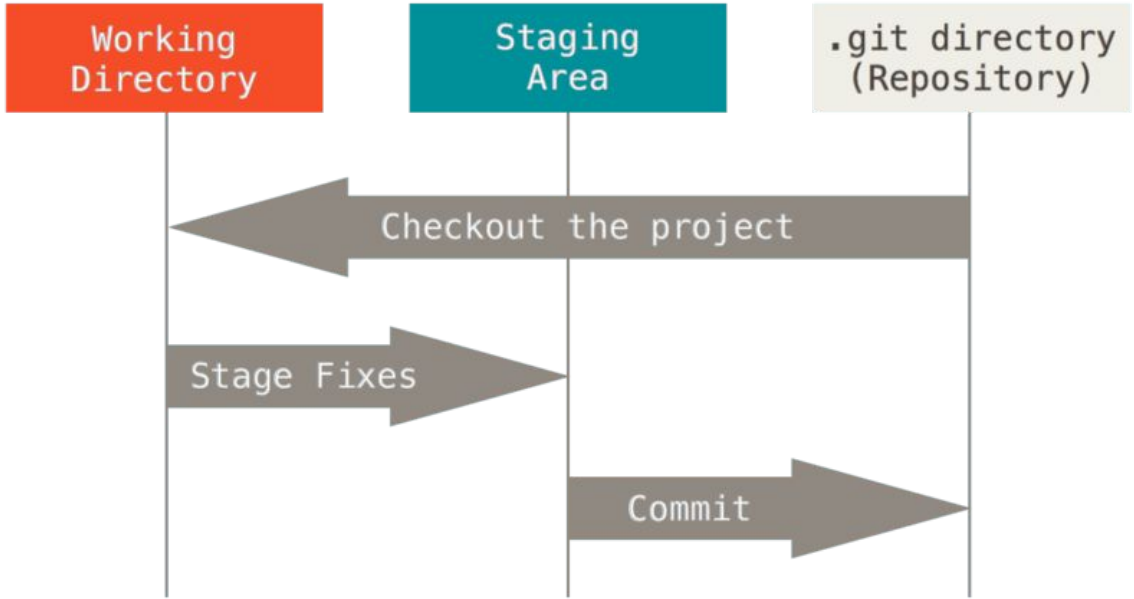
Copia de una versión del proyecto. Estos archivos se sacan de la base de datos comprimida en el directorio de Git, y se colocan en el disco para que los podamos usar o modificar

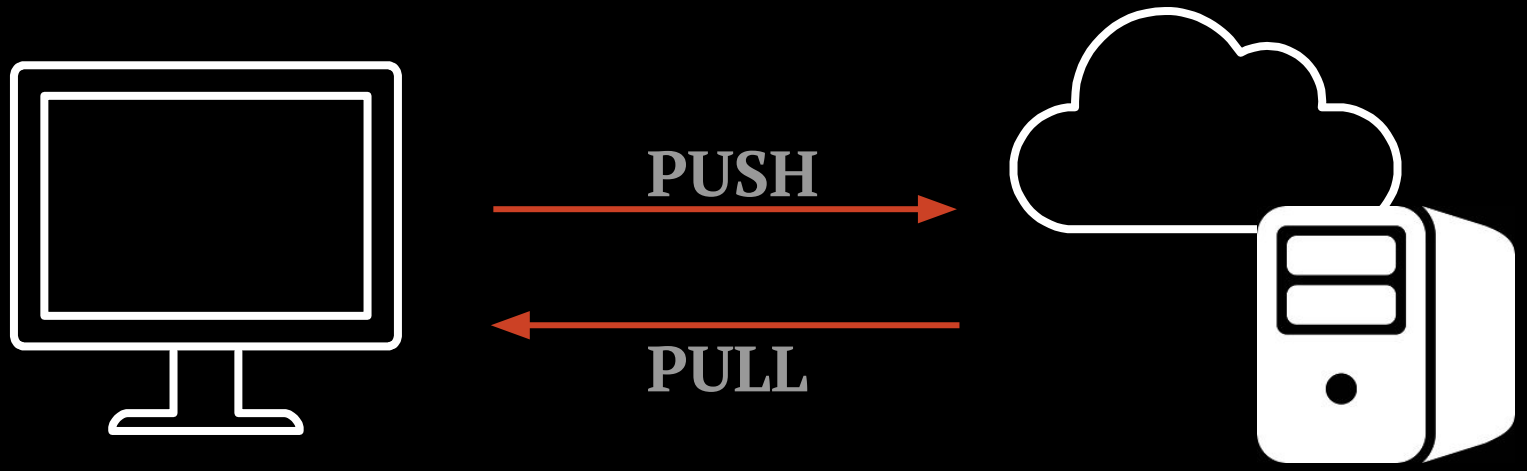
Área de preparación (*staging area)*

Es un archivo, generalmente contenido en nuestro directorio de Git, que almacena información acerca de lo que va a ir en nuestra próxima confirmación

¿Cómo funciona?

Secciones de un proyecto




PUSH
PULL

2.
Preliminares

Creación de cuenta e instalación

Creación de cuenta

Recomendado

Instalación de Git

https://git-scm.com/downloads/

Instalación de GUI (opcional)

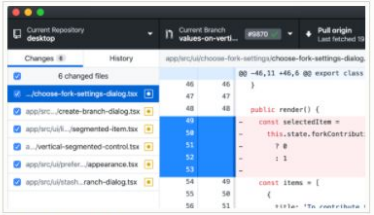
GUI Clients

Git comes with built-in GUI tools for committing (git-gui) and browsing (gitk), but there are several third-party tools for users looking for platform-specific experience.

If you want to add another GUI tool to this list, just follow the instructions.

All | Windows | Mac | Linux | Android | iOS

SourceTree
Platforms: Mac, Windows
Price: Free
License: Proprietary



GitHub Desktop
Platforms: Mac, Windows
Price: Free
License: MIT

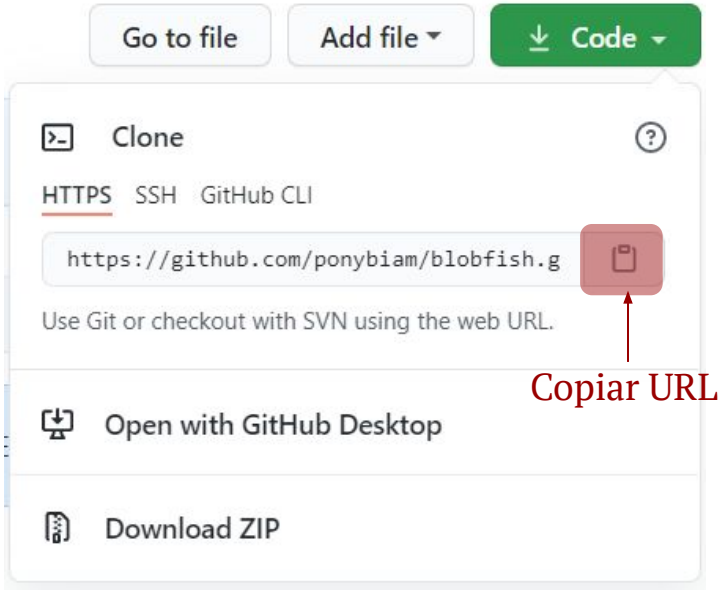
https://git-scm.com/downloads/guis

3.
Obtener un repositorio

Existente o nuevo

Clonar un repositorio existente: de la nube a local

1. En github.com/user/repositorio-deseado:



2. En Git Bash:

```
$ cd <ruta destino>

$ git clone <url-copiada>

# Si se desea renombrar el repositorio:

$ git clone <url-copiada> <nombre-nuevo>
```

El *repositorio deseado* se copiará en *ruta destino*.

Crear un repositorio nuevo: de local a la nube

creación del repositorio

1. En Git Bash:

```
$ cd <ruta del repositorio>

$ git init

# debemos indicar quiénes somos y cuál es nuestro correo electrónico

$ git config --local user.name "nombre"

$ git config --local user.email "email"

$ git add . # Agregar todos los archivos

$ git add <nombre-archivo> # Agregar archivo específico

$ git commit -m "Primer commit"
```

Crear un repositorio nuevo: de local a la nube

conectarlo a la nube

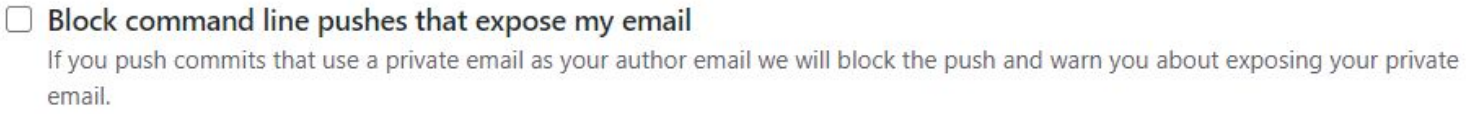
2. En github.com/user/ > Repositories > 

3. En Git Bash:

```
$ git remote add origin <url repositorio>

$ git remote -v # para chequear

$ git push origin master # git push <remote name> <branch name>
```

4. Si tienen autenticación en 2 pasos seguir este instructivo.
5. Si arroja error `push declined due to email privacy restrictions`:

En GitHub > Settings > destildar la opción



4.
Cambios

This is what I paid for

Add & Commit

```
$ git add . # Agregar todos los archivos

$ git add test.txt # Agregar archivo específico

$ git commit -m "Upload test.txt"

$ git push origin master # git push <remote name> <branch name>
```

```
# Para sincronizar de la nube a local

$ git pull origin master # git pull <remote name> <branch name>
```

Gestión de cambios

Algunos comandos útiles

```
# diferencias entre el directorio de trabajo y el remoto, i.e., fue modificado pero no se agregó al staging

$ git diff

# diferencias entre el área de preparación y el remoto

$ git diff --staged

# permite visualizar los commits que han modificado el archivo y muestra los mensajes

$ git log path/to/file

# permite visualizar todos los commits en una línea

$ git log --oneline
```

Gestión de cambios

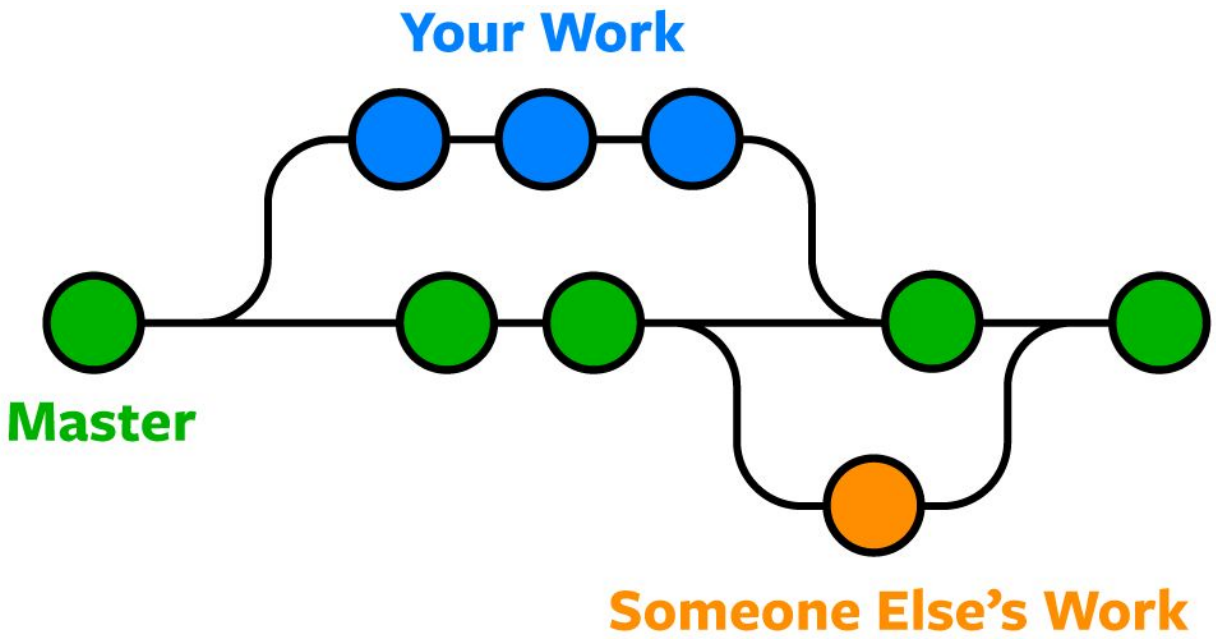
Recuperar una versión anterior

```
# Primero, ver el log para tener el hash de cada commit

$ git log path/to/file

# Deshace los últimos cambios realizados y vuelve el archivo a la versión en la
que se encontraba antes de modificarlo

$ git checkout -- <nombre-del-archivo>

# Deshace los cambios a la versión indicada

$ git checkout <commit hash> <archivo>

$ git add <archivo>

$ git commit -m “Restore version <ID commit>”

$ git push origin <branch>
```

5.
Ramas (*branches*)

Oh no

Branches



Branches

```
# crea una nueva rama y nos mueve hacia ella
$ git checkout -b <new-branch>
# me mueve a la rama indicada
$ git checkout <branch-name>
```

Más sobre checkout, que al parecer hace de todo.

6.
GUI: GitHub Desktop

Gracias Dio

Haciendo lo mismo pero fácil

- Clonar repositorio
- Crear repositorio
- Añadir un repositorio existente localmente
- Agregar, modificar y confirmar cambios
- Pull & Push

Referencias

- https://macfernandez.github.io/hellogit/
- https://rogerdudler.github.io/git-guide/
- https://guides.github.com/
- Pro Git book, Scott Chacon & Ben Straub