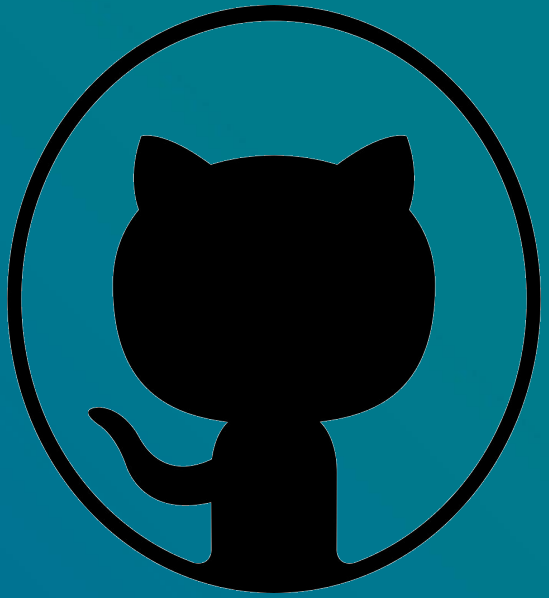
Control de versiones

¿Qué es GitHub?

GitHub es una plataforma de desarrollo colaborativo de software que permite a los desarrolladores alojar proyectos en repositorios remotos y colaborar con otros a través de un control de versiones basado en Git.



UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Programación I

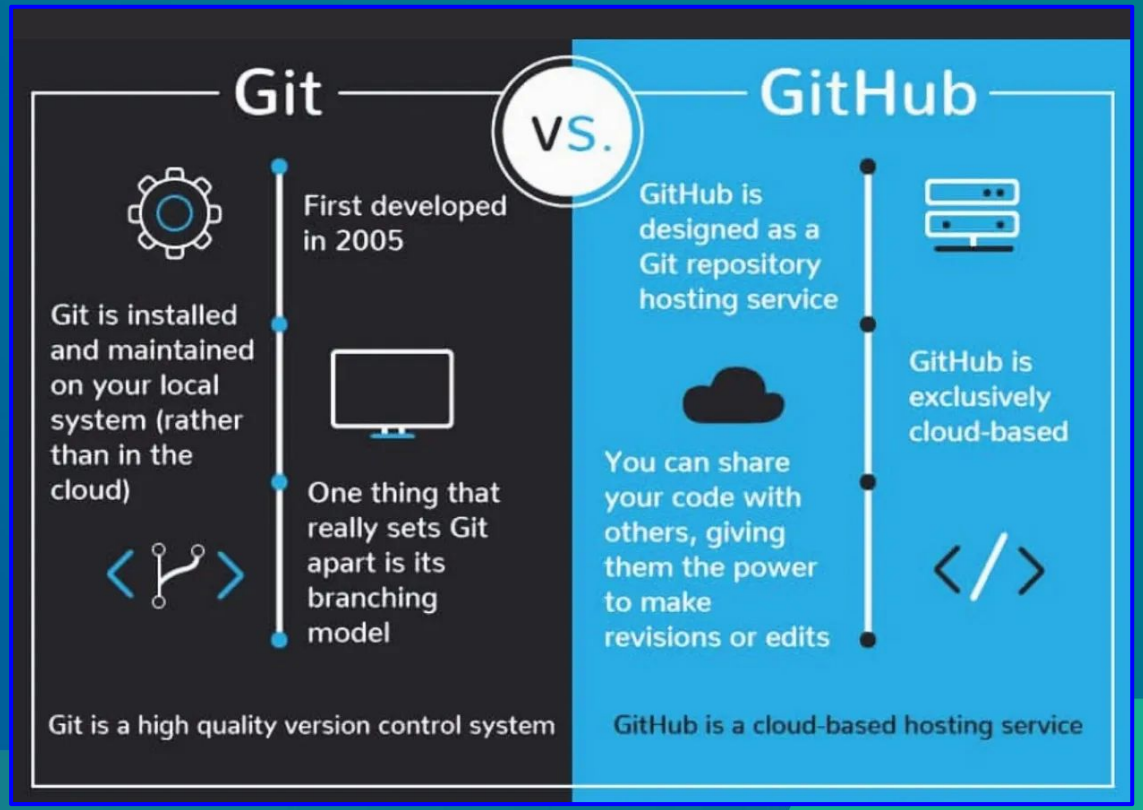
Scarafilo Germán - Quiroz Matias

Control de versiones

¿Qué es Git?

Git es un sistema de control de versiones open source que permite a los desarrolladores realizar un seguimiento de los cambios en el código fuente a lo largo del tiempo, facilitando la colaboración y la gestión de proyectos de software.

Control de versiones



Metodología de trabajo

1. Crear una cuenta en www.github.com
2. Descargar e instalar git desde https://git-scm.com/downloads
3. Configurar git:
   Abrir una terminal y configurar nombre y correo electrónico globalmente.

```
git config --global user.name "Tu Nombre"
git config --global user.email "tuemail@example.com"
```

Comandos git

La tecnología git es muy amplia. Existen infinidad de comandos para poder trabajar de manera local y remota con un repositorio. Muchos de esos comandos los aprenderán a lo largo de la carrera. Aquí va una lista de los comandos básicos que usaremos en esta cátedra.

git init

Inicializa un nuevo repositorio de Git en el directorio actual, creando una carpeta .git.

Nota: para inicializar un repositorio previamente tenemos que crear una carpeta y abrir la misma desde línea de comandos.

```
Initialized empty Git repository in /ruta/a/tu/proyecto/.git/
```

git status

Muestra el estado actual del repositorio, incluyendo los archivos modificados, agregados, y no rastreados (a modo de ejemplo crear un archivo ejemplo.txt dentro de nuestra carpeta donde ejecutamos el comando init).

```
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ejemplo.txt

nothing added to commit but untracked files present (use "git add" to track)
```

git add .

Agrega todos los archivos modificados en el directorio actual al área de preparación (staging area).

```
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   ejemplo.txt
```

Nota: luego de ejecutar nuevamente el comando status, git nos informa que hay cambios que están listos para ser impactados (committed)

git commit -m "mensaje"

Captura de Cambios: Git toma todos los cambios que han sido añadidos al área de preparación (staging area) mediante el comando `git add` y los guarda en el repositorio local como un nuevo commit. Un commit es un "punto de control" en el historial del proyecto, que registra el estado exacto de todos los archivos en ese momento y nos permite agregar un mensaje descriptivo del mismo.

```
git commit -m "Añadir ejemplo.txt al repositorio"
```

```
[master (root-commit) abc1234] Añadir ejemplo.txt al repositorio
 1 file changed, 1 insertion(+)
 create mode 100644 ejemplo.txt
```

git remote add origin <url del repositorio>

Asocia tu repositorio local con el repositorio remoto en GitHub.

```
git remote add origin https://github.com/tu_usuario/repo_prueba.git
```

Nota: verificar en github que el repositorio local se haya asociado al remoto.

git push

Envía los commits locales al repositorio remoto, actualizando el proyecto compartido.

```
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 217 bytes | 217.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/tu_usuario/repo_prueba.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Programación I

Scarafilo Germán - Quiroz Matias

Control de versiones