

### 1) Guía de preguntas

#### ¿Qué es GitHub?

Es una plataforma online donde distintos usuarios pueden cooperar para la creación y modificación de código de manera colaborativa.

#### ¿Cómo crear un repositorio en GitHub?

Desde nuestro perfil vamos a Repositories > New. Le ponemos nombre y descripción y lo seteamos como público o privado

#### ¿Cómo crear una rama en Git?

Luego de haber creado un repositorio local con a través del comando `git init`, con el comando `git branch nombre_rama` creamos una rama y le damos un nombre en particular (*master*, por defecto)

#### ¿Cómo cambiar a una rama en Git?

Por medio del comando `git checkout nombre_otra_rama`. Lo que en verdad hacemos al movernos entre ramas es actualizar el HEAD, indicador de Git de dónde nos encontramos trabajando.

También se puede crear otra rama y moverse ahí al mismo tiempo con el comando `git checkout -b nueva_rama`

#### ¿Cómo fusionar ramas en Git?

Con el comando `git merge nombre_rama` (vamos a fusionar la rama que indiquemos a aquella en la cual nos encontramos trabajando en el momento de hacerlo)

#### ¿Cómo crear un commit en Git?

Con `git commit -m mensaje`, donde vamos a hacer un comentario de referencia sobre esta modificación.

Previamente deberemos haber hecho un `git add` . para agregar al *Stage* (espacio de preparación previa a subir los commits) los archivos sobre los cuales hacemos el commit.

#### ¿Cómo enviar un commit a GitHub?

Con `git push origin nombre_rama`

### ¿Qué es un repositorio remoto?

Es una copia de un proyecto local, alojada en GitHub

### ¿Cómo agregar un repositorio remoto a Git?

Con `git remote add origin url`, donde url va a ser la dirección GitHub del repositorio.

Podemos ver una lista de ellos con el comando `git remote -v`. Y con podemos obtener información sobre repositorios remotos `git fetch nombre_repo_remoto`

### ¿Cómo empujar cambios a un repositorio remoto?

Con `git push origin master`

### ¿Cómo tirar de cambios de un repositorio remoto?

Con `git pull origin master`

### ¿Qué es un fork de repositorio?

Es una bifurcación de un proyecto, donde se clona un repositorio de un tercero a nuestro GitHub para que podamos modificarlo sin afectar el proyecto original.

### ¿Cómo crear un fork de un repositorio?

Desde el repositorio de interés en GitHub, haciendo click en el botón Fork

### ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

Desde GitHub, haciendo click en el botón *Contribute > Open pull request*. Además deberemos indicar en qué es que mejora nuestro código al original

### ¿Cómo aceptar una solicitud de extracción?

Abriendo la solicitud de extracción, y seleccionando Aprobar, para que los cambios se realicen

### ¿Qué es una etiqueta en Git?

Es la marca que usa Git para señalar que hubo un cambio en el historial de versiones. Se usan para marcar cambios de versión del programa, parches, modificaciones de código, etc. y hay livianas (una simple marca de modificación en el historial) y anotadas (detallan checksum, autor, email, mensaje de interés, etc)

### ¿Cómo crear una etiqueta en Git?

Con `git tag nombre`

### ¿Cómo enviar una etiqueta a GitHub?

Con `git push -tags` para todas las etiquetas. También se puede enviar una en particular con el comando `git push nombre_etiqueta`

### ¿Qué es un historial de Git?

Es el registro secuencial de todos los commits de un proyecto. Además, cada rama tiene su propio historial de commits.

### ¿Cómo ver el historial de Git?

Con `git log` vemos todos los commit (en orden descendente). El comando tiene además las siguientes opciones: `-n` (donde n es la cantidad de últimos commits que queremos ver), `--oneline` (para una línea por commit), `--decorate` (muestra referencias de ese commit), `--graph` (arma pequeño gráfico en ASCII para visualizarlo mejor), `--all` (muestra historial de commits para todas las ramas)

### ¿Cómo buscar en el historial de Git?

Con `git log --grep`, para filtrar una búsqueda específica (string) entre todos los commits. También se puede usar: `git log --nombre_archivo` si necesitamos ese en particular, `git log --since="yyyy-mm-dd" -until="yyyy-mm-dd"` para buscar en un rango de fechas, o `git log --author="nombre_autor"` para buscar por autor

### ¿Cómo borrar el historial de Git?

Localmente, usamos `git reset` para borrar todo del Stage, `git reset nombre_archivo` para borrar uno específico.

Remotamente, con el comando `git rebase`, para borrar commits consecutivos.

### ¿Qué es un repositorio privado en GitHub?

Un repositorio al que tienen acceso permitido ciertos usuarios

### ¿Cómo crear un repositorio privado en GitHub?

Al momento de crear el repositorio en GitHub seleccionar la opción Privado

### ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Yendo a *Colaboradores > Agregar personas*. Se debe indicar por el mail del usuario, establecer su rol y elegir el repositorio al cual será invitado

### ¿Qué es un repositorio público en GitHub?

Un repositorio al que tienen acceso cualquier persona

### ¿Cómo crear un repositorio público en GitHub?

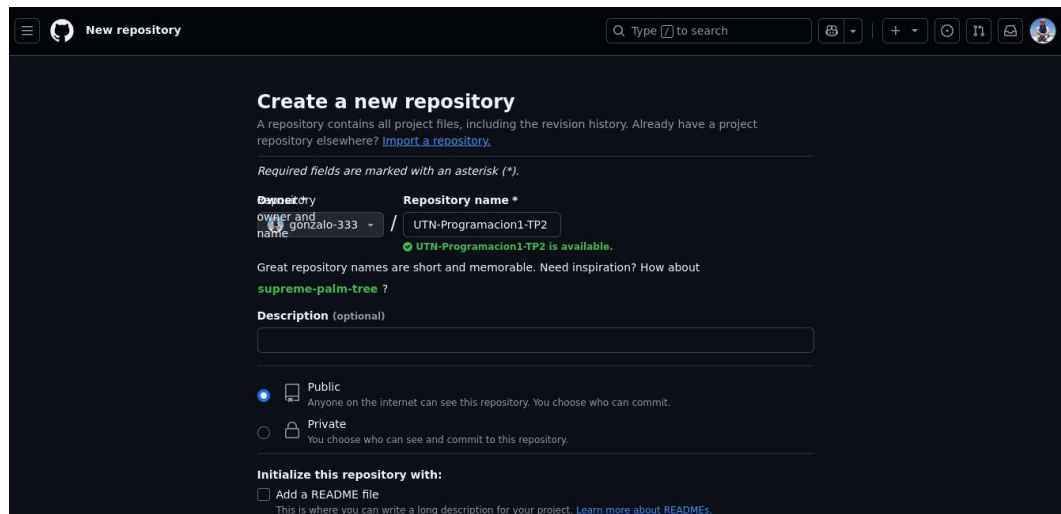
Al momento de crear el repositorio en GitHub seleccionar la opción Público

### ¿Cómo compartir un repositorio público en GitHub?

En el repositorio de interés, desde *Configuración > Acceso > Agregar personas*

## 2) Actividad GitHub n°1

- Creo repositorio



**Create a new repository**

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (\*).

**Owner and name** **Repository name \***

gonzalo-333 / UTN-Programacion1-TP2

UTN-Programacion1-TP2 is available.

Great repository names are short and memorable. Need inspiration? How about [supreme-palm-tree](#)?

**Description** (optional)

☒ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with:**

☐ Add a README file  
This is where you can write a long description for your project. [Learn more about READMEs](#).

- Creo archivo, lo agrego al stage, hago commit y “pusheo”

```
root@olivia-PC:/home/olivia/UTN/TP2# git commit -m "Agregando mi-archivo.txt"
En la rama master
nada para hacer commit, el árbol de trabajo está limpio
```

```
root@olivia-PC:/home/olivia/UTN/TP2# git branch
* master
```

```
root@olivia-PC:/home/olivia/UTN/TP2# git push git@github.com:gonzalo-333/UTN-Programacion1-TP2.git master
Enumerando objetos: 3, listo.
Contando objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (3/3), 230 bytes | 230.00 KiB/s, listo.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:gonzalo-333/UTN-Programacion1-TP2.git
 * [new branch]      master -> master
```

- Creo nueva rama y me muevo a ella, modifico archivo y lo subo a ella

```
root@olivia-PC:/home/olivia/UTN/TP2# git checkout -b nueva_rama
Cambiado a nueva rama 'nueva_rama'
root@olivia-PC:/home/olivia/UTN/TP2# git branch
master
* nueva_rama
root@olivia-PC:/home/olivia/UTN/TP2# echo "Nueva rama agregada :)" > mi-archivo.txt
root@olivia-PC:/home/olivia/UTN/TP2# git add .
root@olivia-PC:/home/olivia/UTN/TP2# git status
En la rama nueva_rama
Cambios a ser confirmados:
  (usa "git restore --staged <archivo>..." para sacar del área de stage)
    modificados:   mi-archivo.txt

root@olivia-PC:/home/olivia/UTN/TP2# git commit -m "Modificando en nueva_rama"
[nueva_rama 0aa8c41] Modificando en nueva_rama
 1 file changed, 1 insertion(+)
root@olivia-PC:/home/olivia/UTN/TP2# cat mi-archivo.txt
Nueva rama agregada :)
```

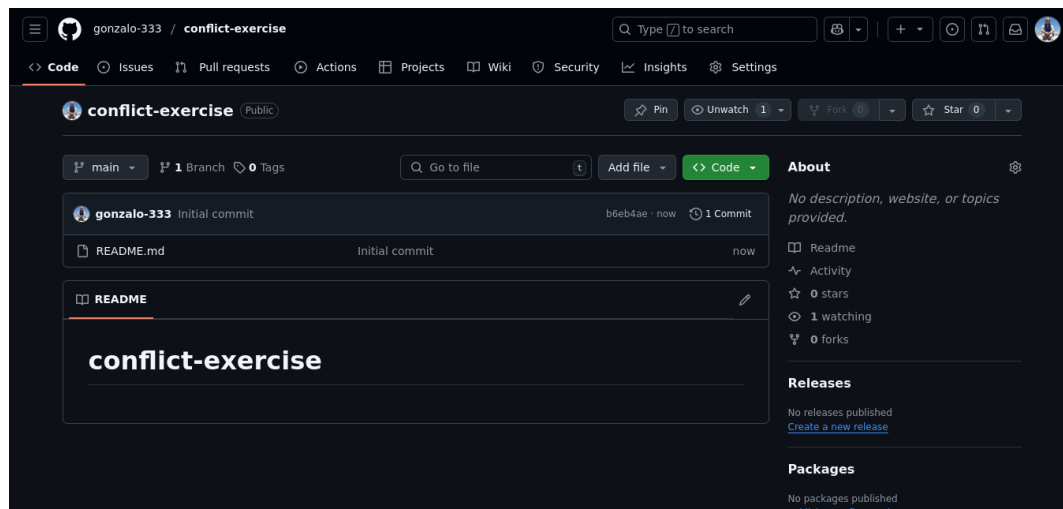
```

root@olivia-PC:/home/olivia/UTN/TP2# git push git@github.com:gonzalo-333/UTN-Programacion1-TP2.git nueva_rama
Enumerando objetos: 5, listo.
Contando objetos: 100% (5/5), listo.
Escribiendo objetos: 100% (3/3), 284 bytes | 284.00 KiB/s, listo.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'nueva_rama' on GitHub by visiting:
remote:   https://github.com/gonzalo-333/UTN-Programacion1-TP2/pull/new/nueva_rama
remote:
To github.com:gonzalo-333/UTN-Programacion1-TP2.git
 * [new branch]      nueva_rama -> nueva_rama

```

### 3) Actividad GitHub nº2

- Creo repositorio



- Clono repositorio a maquina local

```

root@olivia-PC:/home/olivia/UTN/TP2# git checkout master
Cambiado a rama 'master'
root@olivia-PC:/home/olivia/UTN/TP2# git push git@github.com:gonzalo-333/UTN-Programacion1-TP2.git master
Everything up-to-date
root@olivia-PC:/home/olivia/UTN/TP2# git clone git@github.com:gonzalo-333/conflict-exercise.git
Clonando en 'conflict-exercise'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 9 (delta 0), reused 6 (delta 0), pack-reused 0 (from 0)
Recibiendo objetos: 100% (9/9), listo.
root@olivia-PC:/home/olivia/UTN/TP2# cd conflict-exercise

```

- Creo nueva rama, me voy a ella, modifico archivo, lo agrego al Stage y hago commit

```

root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git checkout -b feature-branch
Cambiado a nueva rama 'feature-branch'

```

```

root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# echo -e "\nEste es un cambio en la feature branch." >> README.md
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# cat README.md
# conflict-exercise

Este es un cambio en la feature branch.
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git add README.md
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git commit -m "Added a line in feature-branch"
[feature-branch 5dae792] Added a line in feature-branch
1 file changed, 3 insertions(+), 1 deletion(-)

```

- Vuelvo a rama main, edito archivo, guardo cambio y hago commit

```

root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git checkout main
Cambiado a rama 'main'
Tu rama está actualizada con 'origin/main'.
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# echo -e "\nEste es un cambio en la main branch." >> README.md
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# cat README.md
# conflict-exercise

Este es un cambio en la main branch.
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git add README.md
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git commit -m "Added a line in main branch"
[main 88d74b9] Added a line in main branch
1 file changed, 2 insertions(+), 1 deletion(-)
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# 

```

- Hago merge, generando el conflicto y lo resuelvo conservando los dos contenidos

```

root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git merge feature-branch
Auto-fusionando README.md
CONFLICTO (contenido): Conflicto de fusión en README.md
Fusión automática falló; arregle los conflictos y luego realice un commit con el resultado.
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# cat README.md
# conflict-exercise
<<<<<< HEAD
Este es un cambio en la main branch.
=====
Este es un cambio en la feature branch.
>>>>>> feature-branch
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# cat README.md
# conflict-exercise

Este es un cambio en la main branch.
Este es un cambio en la feature branch.
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# 

```

```

root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git add README.md
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git commit -m "Resolved merge conflict"
[main fa054f1] Resolved merge conflict
root@olivia-PC:/home/olivia/UTN/TP2/conflict-exercise# git push origin main
Enumerando objetos: 11, listo.
Contando objetos: 100% (11/11), listo.
Compresión delta usando hasta 2 hilos
Comprimiendo objetos: 100% (6/6), listo.
Escribiendo objetos: 100% (9/9), 855 bytes | 285.00 KiB/s, listo.
Total 9 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To github.com:gonzalo-333/conflict-exercise.git
b6eb4ae..fa054f1 main -> main

```

gonzalo-333 / conflict-exercise

Type / to search

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

Files

main

Go to file

README.md

conflict-exercise / README.md

gonzalo-333 Resolved merge conflict fa054f1 · 4 minutes ago History

Preview Code Blame 5 lines (3 loc) · 99 Bytes Code 55% faster with GitHub Copilot Raw

# conflict-exercise

Este es un cambio en la main branch. Este es un cambio en la feature branch.