

**Tarea 3**

# **Asignación Individual**

**Programación III**

**Prof. Kelyn Tejada**

**Diana Calderón  
2022-1921**

# ÍNDICE

	Contenido	Pag.
1.	Introducción	4
2.	¿Qué es Git?	5
3.	¿Para que funciona el comando git init?	5-6
4.	¿Qué es una rama?	6
5.	¿Cómo saber en cual rama estoy?	7
6.	¿Quién creo git?	7



# INTRODUCCIÓN



Git es una herramienta fundamental en el mundo del desarrollo de software que ha revolucionado la forma en que los equipos gestionan y colaboran en proyectos. Este sistema de control de versiones distribuido permite llevar un historial detallado de los cambios realizados en un proyecto, permitiendo a los desarrolladores colaborar de manera eficiente, aun sin conexión a internet. En ese proyecto veremos el uso de esta herramienta, además de ciertos modelos de trabajo y comandos que son sumamente importantes de conocer a la hora de trabajar en el desarrollo de un sistema.

## 1-Desarrolla el siguiente Cuestionario

### 1-Que es Git?

Git es un sistema de control de versiones distribuido.

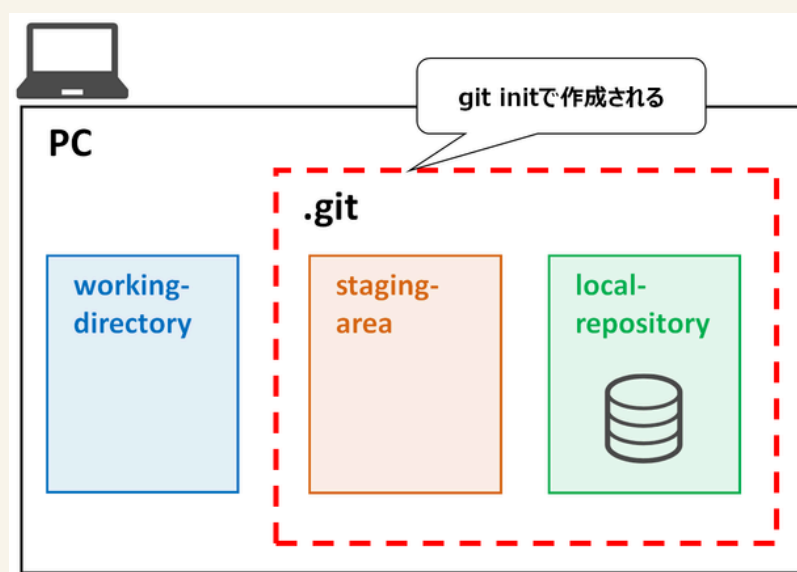
¿Y qué quiere decir esto?

Pues que este crea una réplica exacta de del proyecto en cada dispositivo que lo clones. Esta copia es totalmente independiente, lo que funciona como un repositorio local, gracias a esto, se puede trabajar sin conexión a internet, y luego cuando se restaure la conexión se pueden sincronizar dichos cambios al repos principal. Esto quiere decir, que Git no requiere una conexión constante con el servidor central para trabajar. También este sirve como una máquina del tiempo para el código, permitiéndonos volver a versiones anteriores, comparar cambios y trabajar de forma colaborativa con otros desarrolladores.



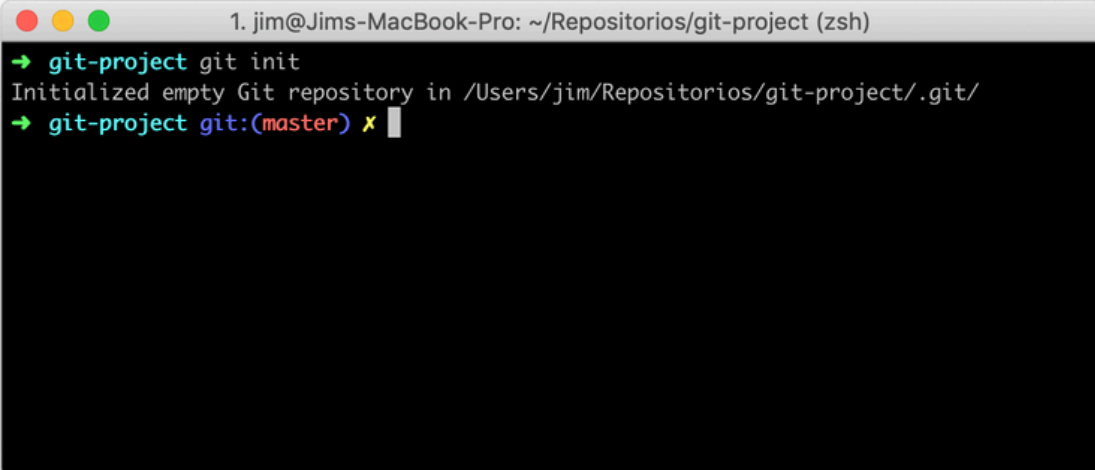
### 2-Para que funciona el comando git init?

El comando git init es utilizado para inicializar un nuevo repositorio de Git en un directorio específico o en el actual. Es decir, convierte una carpeta en un espacio donde Git comenzará a gestionar el historial de tus archivos. Este puede utilizarse para convertir un proyecto existente y sin versión en un repositorio de Git, o para inicializar un nuevo repositorio vacío.



En palabras más simples, el comando `git init` es uno de los primeros comandos que se deben de ejecutar antes de comenzar a utilizar git en un proyecto. Cuando este se ejecuta, le indicas a Git que debe de empezar a rastrear los archivos y cambios en esa carpeta (seguimiento del historial), creando un repos de git en ella. Y ahí es donde se alojaran las informaciones de la versión y el historial de cambios de las carpetas y subcarpetas (directorios y subdirectorios).

Se puede usar; Si ya tienes un proyecto en una carpeta, puedes correr `git init` para empezar a hacer seguimiento de los archivos. Así, Git creará una carpeta oculta llamada `.git` dentro del directorio, que almacenará la información del repositorio. También para crear un repositorio vacío para un proyecto nuevo. Si empiezas desde cero en una carpeta vacía, `git init` también crea el repositorio vacío, listo para que agregues archivos y comiences a versionarlos.

A screenshot of a macOS terminal window. The title bar shows three colored window control buttons (red, yellow, green) and the text '1. jim@Jims-MacBook-Pro: ~/Repositorios/git-project (zsh)'. The terminal has a black background with green text. The first line shows a prompt '→' followed by 'git-project' and the command 'git init'. The second line shows the output 'Initialized empty Git repository in /Users/jim/Repositorios/git-project/.git/'. The third line shows a prompt '→' followed by 'git-project' and the command 'git:(master)' with a red 'x' icon and a cursor.

```
1. jim@Jims-MacBook-Pro: ~/Repositorios/git-project (zsh)
→ git-project git init
Initialized empty Git repository in /Users/jim/Repositorios/git-project/.git/
→ git-project git:(master) x
```

### 3-Que es una rama?

Una rama es una línea de desarrollo independiente dentro de un repositorio de Git. Es por así decirlo una copia de la rama principal (normalmente son llamadas `main` o `master`), y de ahí uno puede trabajar en estas para explorar, crear nuevas funcionalidades o corregir errores del código principal, pero sin alterarlo. Es decir que podemos inventar lo que sea en esta, pero, no afectara nuestro código principal, o mejor dicho no arriesgaríamos a echar a perder nuestro código estable. Eso sí, que si se arregla o mejora algo en una rama del código principal y se han hecho las pruebas correspondientes, estos cambios pueden fusionarse a la rama principal.

#### 4-¿Cómo saber en cual rama estoy?

Puedes ver la rama actual usando el comando “*git branch*” o “*git status*”. Ambos muestran el nombre de la rama activa.

```
► git status
```

```
On branch docs/add-note
```

```
nothing to commit, working tree clean
```

```
Command Prompt

C:\Users\Deborah\Documents\recipes>git checkout main
Switched to branch 'main'

C:\Users\Deborah\Documents\recipes>type file1.txt
file1

C:\Users\Deborah\Documents\recipes>git checkout -b login_issue
Switched to a new branch 'login_issue'

C:\Users\Deborah\Documents\recipes>git branch
* login_issue
  main
  style_change

C:\Users\Deborah\Documents\recipes>
```

#### 5- Quien creo git?

Git fue creado por Linus Torvalds, el creador del sistema operativo Linux, en 2005. Inicialmente el motivo de su desarrollo era para gestionar el código fuente del kernel de Linux de forma distribuida.





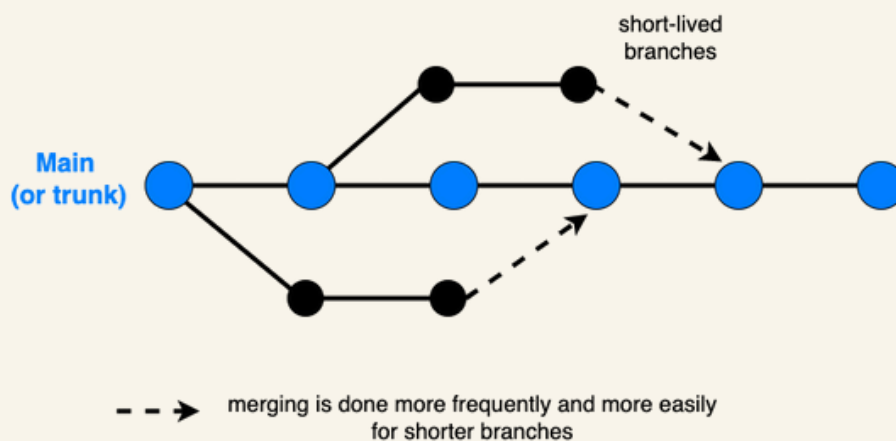


## 8-Que es trunk-based development?

Git Flow es un modelo de ramificación el cual proporciona un marco de trabajo más robusto para gestionar el flujo de trabajo de un proyecto. En palabras sencillas, es una forma de organizar el trabajo en ramas, para distintos tipos de tareas. Este modelo define ramas como develop para el desarrollo activo, main para la versión estable, feature para nuevas funcionalidades, hotfix para correcciones urgentes y release para preparar nuevas versiones. Es especialmente útil para equipos que trabajan en proyectos de gran escala.

### Trunk-based development

StatusNeo





# CONCLUSIÓN

Git no solo es un sistema de control de versiones, sino un pilar esencial para cualquier equipo de desarrollo moderno. Sus funcionalidades, como el manejo de ramas, la creación de repositorios y la colaboración distribuida, facilitan un flujo de trabajo organizado y productivo. Modelos de trabajo como Git Flow y Trunk-Based Development amplían aún más su potencial, adaptándose a distintos enfoques y necesidades de proyectos. Con el uso de comandos fundamentales como `git init`, `git commit`, `git branch`, y `git merge`, Git se convierte en una herramienta versátil y robusta que fomenta una colaboración eficiente y un control preciso sobre la evolución de cualquier proyecto de software.

# BIBLIOGRAFÍA

🇪🇸 M. Á. (2020, octubre 13). ¿Por qué trunk-based development? DEV Community. <https://dev.to/marianocodes/por-que-trunk-based-development-i5n>

Atlassian. (s/f-a). Flujo de trabajo de Gitflow. Atlassian, de <https://www.atlassian.com/es/git/tutorials/comparing-workflows/gitflow-workflow>

Atlassian. (s/f-b). git init. Atlassian, de <https://www.atlassian.com/es/git/tutorials/setting-up-a-repository/git-init>

Atlassian. (s/f-c). Qué es Git. Atlassian. <https://www.atlassian.com/es/git/tutorials/what-is-git>

Git - Fundamentos de Git. (s/f). Git-scm.com. Recuperado el 11 de noviembre de 2024, de <https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Fundamentos-de-Git>

Git - git-init Documentation. (s/f). Git-scm.com. Recuperado el 11 de noviembre de 2024, de <https://git-scm.com/docs/git-init/es>

Git - ¿Qué es una rama? (s/f). Git-scm.com. Recuperado el 11 de noviembre de 2024, de <https://git-scm.com/book/es/v2/Ramificaciones-en-Git-%C2%BFQu%C3%A9-es-una-rama%3F>

¿Qué es Git? (s/f). Microsoft.com. Recuperado el 11 de noviembre de 2024, de <https://learn.microsoft.com/es-es/devops/develop/git/what-is-git>