



Las Americas Institute of Technology

Nombre: Jersey Stevens Jimenez Ulloa

Matricula: 2023-1800

Correo Institucional: 20231800@itla.edu.do

Docente: Kelyn Tejada

Asignatura: Programación 3

Contenido

Parte 1 – Cuestionario	3
1. ¿Qué es Git?	3
2. ¿Para qué sirve el comando git init?	3
3. ¿Qué es una rama en Git?.....	3
4. ¿Cómo saber en cuál rama estoy trabajando?.....	3
5. ¿Quién creó Git?	3
6. ¿Cuáles son los comandos esenciales de Git?	4
7. ¿Qué es Git Flow?	4
8. ¿Qué es el desarrollo basado en trunk (Trunk Based Development)?.....	5
Parte 2 – Enlace GitHub Proyecto Practico	5
Documentación Breve	5
Bibliografía.....	6



Parte 1 – Cuestionario

1. ¿Qué es Git?

Git es un sistema de control de versiones distribuido. Yo lo uso para tener un historial completo de todos los cambios que hago en mi proyecto, lo cual me permite volver atrás si cometo un error o si algo deja de funcionar. También es útil para colaborar con otras personas, porque me deja ver quién hizo qué, cuándo y por qué. A diferencia de otros sistemas de versiones, Git es muy rápido y me permite trabajar incluso sin conexión a internet, lo cual me ha salvado varias veces.

2. ¿Para qué sirve el comando git init?

El comando git init lo uso cuando quiero empezar a usar Git en un proyecto que aún no está siendo controlado por versiones. Lo que hace es crear una carpeta oculta llamada .git, que guarda toda la información que Git necesita para hacer seguimiento de los cambios. Después de eso, ya puedo usar todos los demás comandos de Git en ese proyecto. Es como decirle a Git: “empieza a observar esta carpeta”.

3. ¿Qué es una rama en Git?

Una rama (o *branch*) es una línea de trabajo independiente dentro del proyecto. Es útil porque me permite desarrollar nuevas funcionalidades, hacer pruebas o corregir errores sin tocar directamente la versión principal del proyecto. Por ejemplo, yo puedo estar en la rama develop haciendo pruebas, mientras otra persona está en la rama main subiendo la versión estable. Luego puedo fusionarlas cuando esté seguro de que todo funciona bien.

4. ¿Cómo saber en cuál rama estoy trabajando?

Para saber en qué rama estoy, simplemente uso el comando git branch, y la rama actual aparece marcada con un asterisco (*). También si estoy usando Git Bash, la rama aparece entre paréntesis al final del prompt, como (main) o (develop). En Visual Studio Code también aparece abajo a la izquierda, así que es fácil saberlo.

5. ¿Quién creó Git?

Git fue creado por Linus Torvalds en 2005. Él es también el creador de Linux. Lo desarrolló porque el sistema de control de versiones que estaban usando en el proyecto del kernel de Linux dejó de ser gratuito, y necesitaban una solución libre, segura y rápida. Desde entonces, Git se volvió el estándar en la mayoría de los proyectos de software a nivel mundial.

6. ¿Cuáles son los comandos esenciales de Git?

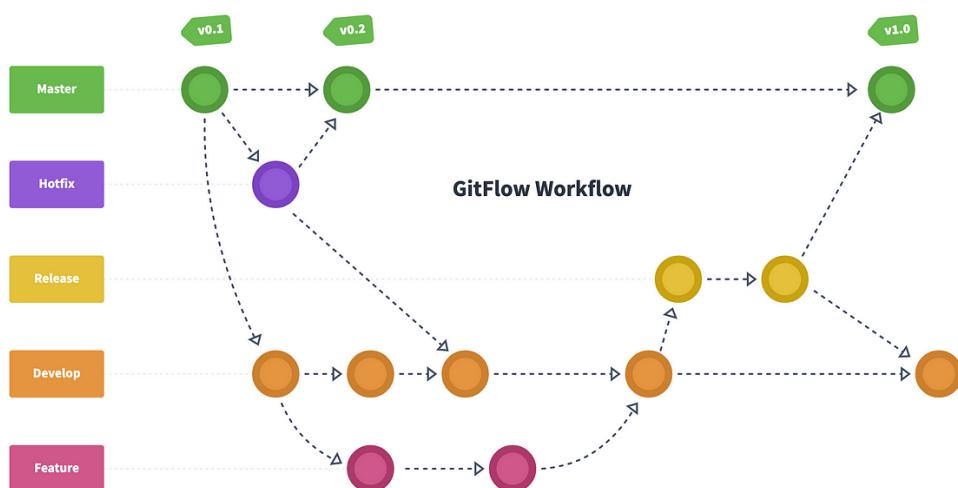
Los comandos que más uso en el día a día son:

- **git init:** para iniciar el repositorio.
- **git status:** para ver el estado de los archivos.
- **git add:** para preparar archivos antes de hacer un commit.
- **git commit:** para guardar un punto de control con los cambios.
- **git push:** para subir los cambios al repositorio remoto (como GitHub).
- **git pull:** para traer los cambios del repositorio remoto.
- **git branch:** para listar o crear ramas.
- **git checkout:** para moverme entre ramas.
- **git merge:** para combinar ramas.

Con estos comandos ya podemos manejar la mayoría de las situaciones que se me presentan al trabajar en un proyecto con Git.

7. ¿Qué es Git Flow?

Git Flow es una metodología para organizar el trabajo con Git. Lo que hace es proponer una estructura de ramas específica que facilita el desarrollo en equipo. Por ejemplo, tiene ramas como main (para versiones estables), develop (para integración), feature (para nuevas funcionalidades), release y hotfix. Yo lo uso cuando quiero mantener todo organizado, sobre todo en proyectos que tienen muchas versiones o en los que varias personas están trabajando al mismo tiempo.



8. ¿Qué es el desarrollo basado en trunk (Trunk Based Development)?

El desarrollo basado en trunk es una estrategia donde todos los desarrolladores trabajan directamente en la rama principal del repositorio, que normalmente se llama main o trunk. En vez de usar muchas ramas separadas, los cambios se hacen pequeños y se integran frecuentemente. Esto ayuda a evitar conflictos grandes y permite que el código siempre esté actualizado. Sin embargo, requiere que todo el equipo tenga mucha disciplina y buenas prácticas, porque cualquier error puede afectar a todos los demás.

Parte 2 – Enlace GitHub Proyecto Practico

<https://github.com/JenseyJim/ProyectoCRUD-T3P3.git>

Documentación Breve

Este proyecto consiste en un sistema CRUD (Crear, Leer, Actualizar y Eliminar) desarrollado como parte de una práctica académica de control de versiones utilizando Git y GitHub. Se creó una aplicación web llamada **Task Manager**, desarrollada con HTML, CSS y JavaScript puro, que permite al usuario gestionar tareas con campos como título, descripción, prioridad y fecha de vencimiento. La estructura visual está basada en Bootstrap, mientras que la lógica de funcionamiento se almacena localmente mediante localStorage. Para organizar el flujo de trabajo se implementó correctamente la metodología **Git Flow**, con ramas como main, develop, qa y al menos cinco ramas feature/, cada una con sus respectivos Pull Requests debidamente documentados y fusionados. Además, el proyecto incluye la integración de **GitHub Actions** con workflows personalizados que validan la existencia de archivos clave y detectan cambios en archivos específicos como parte de una estrategia de automatización básica. Todo el desarrollo se llevó a cabo respetando buenas prácticas de versionado, estructuración y documentación, cumpliendo con los criterios técnicos establecidos por el docente.

Este repositorio cuenta con acciones automatizadas:

- ✓ Verifica que README.md exista
- ✓ Detecta cambios realizados en script.js

Bibliografía

- **Moure, B.** (2023). *Git y GitHub desde cero: Guía de estudio teórico-práctica paso a paso más curso en vídeo*. Recuperado de <https://leanpub.com/git-github>
- **Hinojosa, P., & Merelo, J. J.** *Aprende git*. Recuperado de <https://github.com/JJ/aprende-git>
- **Chacón, S., & Straub, B.** *Pro Git*. Recuperado de <https://git-scm.com/book/es/v2>
- **Midudev.** *Lista de libros sobre programación en Español y gratis*. Recuperado de <https://github.com/midudev/libros-programacion-gratis>
- **Wikipedia.** *Git*. Recuperado de <https://es.wikipedia.org/wiki/Git>