INSTITUTO TECNOLÓGICO DE LAS AMÉRICAS

Las Americas Institute of Technology

Nombre
Hector Jhoan

Apellido
Martinez Mendoza

Matricula
20231789

Carrera
Desarrollo de software

Materia
Programación III

Nombre del docente
Kelyn Tejada Belliard

Tema
Tarea 3

Fecha
4/4/2025

Tabla de contenido

Cuestionario

1. ¿Qué es Git?

`Git` es un controlador de versiones que se instala localmente en la máquina donde se está desarrollando. Cuando inicializamos un repositorio `Git` mantiene un historial o una bitácora de todos los cambios realizados en los archivos que está trackeando, esto te permite volver a versiones anteriores del código que se está programando en caso de que existan errores en las nuevas versiones o quieras revisitar una solución, etc.

`Git` facilita trabajar en equipo porque al llevar un registro claro de lo que se va haciendo, quien, porque y cuando lo hace, te ayuda a dividir mejor las tareas. Además de que puedes crear varias ramas de desarrollo de un mismo proyecto lo que te permite asignar tareas a cada quien.

2. ¿Para qué sirve el comando git init?

Con `Git init` se inicializa un repositorio, esto se refiere a que se crea un subdirectorio .git oculto que rastrea los cambios que se hacen en los archivos que estén dentro del repositorio.

3. ¿Qué es una rama en Git?

Una rama o *branch* en Git es una línea de desarrollo separada de las demás. Lo ideal es tener ramas distintas para diferentes etapas o tareas del desarrollo. La rama principal de publicación suele llamarse `master` (o actualmente `main`), y generalmente cada *commit* en esa rama representa una versión lista para producción.

Si quiero arreglar algo o desarrollar una nueva función sin afectar el contenido de la rama principal, entonces creó una nueva rama. Esta nueva rama es básicamente una copia del contenido actual que puedo modificar libremente. Cuando los cambios estén listos, se pueden unir nuevamente a la rama principal mediante un proceso llamado *merge*.

4. ¿Cómo saber en cuál rama estoy trabajando?

Con el comando en la terminal “`Git branch`” te muestra todas las ramas activas y con un asterisco la rama que estás trabajando.

5. ¿Quién creó Git?

El creador del sistema operativo Linux, Linus Torvalds en el 2005 creó git cuando estaban creando Linux como controlador de versiones.

6. ¿Cuáles son los comandos esenciales de Git?

- `git init`: inicia un repositorio.
- `git add`: añade archivos al área de preparación (staging).
- `git commit`: guarda los cambios en el historial.
- `git status`: muestra el estado actual del repositorio.
- `git log`: muestra el historial de commits.
- `git branch`: gestiona ramas.
- `git checkout`: cambia de rama (con -b nombre_de_la_rama puedes crear una rama nueva)
- o de versión.
- `git switch:` cambia de rama (con -c nombre_de_la_rama puedes crear una rama nueva)
- `git merge`: fusionar ramas.
- `git pull`: descarga y fusiona cambios del repositorio remoto.
- `git push`: sube cambios al repositorio remoto.

7. ¿Qué es Git Flow?

Es una forma de gestionar un proyecto utilizando ramas de `git` distintas para puntos del flujo de trabajo diferentes.
`Main/Main`: En esta rama pondremos el código que está listo para producción, es decir, cada commit de esta rama debe ser una versión estable y lista para publicar del código que se está trabajando.

`Release`: En esta rama va la versión del código que está justo antes de ser publicado es decir, el código debe llegar listo, funcional, ya que esta rama está solo para revisar el código y confirmar que está listo no para hacer cambios o arreglos en el código. Si se descubren errores, a menos que sean ortográficos, debe llevarse de nuevo a `develop` (la rama anterior).
Comúnmente sale desde develop y se fusiona con master/main.

`Develop`: En esta rama se junta todo el trabajo de desarrollo, y se fusiona comúnmente con release si las funciones están correctas.

`Feature`: En esta rama se codifican las nuevas funcionalidades o cambios grandes en el código. Comúnmente salen desde develop y luego de que están listas vuelven a develop.

`Hotfix`: Aquí es donde se realizan los arreglos de errores críticos que se descubren en master/main, comúnmente se fusionan con main/master después de arreglarse.

Básicamente el flujo es el siguiente, creamos el repositorio con `git init`, y se hace el primer `commit` en la rama main. Luego a partir de esta rama creamos la develop `git checkout -b develop main` o `git switch -c develop main`, se utilizará para trabajar el desarrollo del proyecto y cada vez que vayamos a trabajar una funcionalidad distinta

creamos una rama llamada feature y le añadimos el nombre de la funcionalidad, con el comando `git checkout -b feature/nombre-funcionalidad develop.` Al finalizar el desarrollo de la funcionalidad se fusiona la rama feature con develop utilizando `git merge feature/nombre-funcionalidad`. Cuando el proyecto está listo para una nueva versión, se crea una rama `release` desde `develop` con `git checkout -b release/1.0 develop`, donde se realizan pruebas finales. Una vez lista, se fusiona la rama `release` tanto a `main` (`git checkout main` → `git merge release/1.0`) como a `develop` para mantener todo sincronizado. Si aparece un error urgente en producción, se crea una rama `hotfix` desde `main` con `git checkout -b hotfix/1.0.1 main`, se corrige el problema y se fusiona tanto a `main` como a `develop` para que todos los entornos queden actualizados.

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

Es una forma de trabajar donde todos los desarrolladores trabajan dentro una sola rama main y en lugar de tener muchas ramas con commits largos se hacen cambios pequeños y frecuentes en el código.

Bibliografía

- Atlassian. (s.f.). *¿Qué es Git?*. Recuperado el 31 de marzo de 2025, de https://www.atlassian.com/es/git/tutorials/what-is-git

- Git SCM. (s.f.). *git-init Documentation*. Recuperado el 1 de abril de 2025, de https://git-scm.com/docs/git-init

- Chacon, S., & Straub, B. (s.f.). *Pro Git - ¿Qué es una rama?*. Git SCM. Recuperado el 2 de abril de 2025, de https://git-scm.com/book/es/v2/Ramificaciones-en-Git-%C2%BFQu%C3%A9-es-una-rama%3F

- Chacon, S., & Straub, B. (s.f.). *Pro Git - Gestión de ramas*. Git SCM. Recuperado el 2 de abril de 2025, de https://git-scm.com/book/es/v2/Ramificaciones-en-Git-Gesti%C3%B3n-de-Ramas

- Wikipedia. (2024). *Git*. En Wikipedia, La enciclopedia libre. Recuperado el 1 de abril de 2025, de https://es.wikipedia.org/wiki/Git

- Atlassian. (s.f.). *Glosario de comandos de Git*. Recuperado el 3 de abril de 2025, de https://www.atlassian.com/es/git/glossary

- Atlassian. (s.f.). *Gitflow Workflow*. Recuperado el 3 de abril de 2025, de https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

- Hammant, P. (s.f.). *Trunk Based Development*. Recuperado el 31 de marzo de 2025, de https://trunkbaseddevelopment.com/