Instituto Tecnológico de las Américas

Tarea III

Materia

Programación III | Grupo: sábado 9:00AM | Sección: 2

Tema

Módulo 5. Herramientas de administración de fuentes

Estudiante

Amelia Camila Branch Zorrilla| Matrícula: 2022-0008

Profesor

Kelyn Tejada Belliard

Fecha de entrega

25/11/2023

San Pedro de Macorís, República Dominicana

Cuestionario desarrollado de la actividad 1

Git

Git es un sistema de control de versiones distribuido, gratuito y de código abierto, diseñado para gestionar con rapidez y eficacia desde proyectos pequeños a muy grandes. Fue Creado por Linus Torvalds y lanzado el 7 de abril de 2005. El popular comando git init funciona para crear un repositorio Git vacío o reinicializar uno existente.

Rama

Una rama representa una línea independiente de desarrollo. Las ramas ayudan a los equipos de desarrollo a trabajar en paralelo. Separa el "trabajo en curso" del código probado y estable. Puedes saber en qué rama estás en Git corriendo el comando git status.

Comandos más esenciales de Git

- **git branch**
  Este comando es tu herramienta de administración de ramas de uso general. Permite crear entornos de desarrollo aislados en un solo repositorio.
- **git checkout**
  Además de extraer las confirmaciones y las revisiones de archivos antiguas, git checkout también sirve para navegar por las ramas existentes.
- **git add**
  Mueve los cambios del directorio de trabajo al área del entorno de ensayo.
- **git clean**
  Elimina los archivos sin seguimiento de tu directorio de trabajo.
- **git clone**
  Crea una copia de un repositorio de Git existente.

- **git commit**

  Confirma la instantánea preparada en el historial del proyecto.
- **git fetch**

  Con este comando, se descarga una rama de otro repositorio junto con todas sus confirmaciones y archivos asociados.
- **git init**

  Inicializa un nuevo repositorio de Git.
- **git merge**

  Es una forma eficaz de integrar los cambios de ramas divergentes.
- **git pull**

  Este comando es la versión automatizada de git fetch. Descarga una rama de un repositorio remoto e inmediatamente la fusiona en la rama actual.
- **git push**

  Enviar (push) es lo opuesto a recuperar (fetch), con algunas salvedades. Permite mover una o varias ramas a otro repositorio, lo que es una buena forma de publicar contribuciones.
- **git revert**

  Permite deshacer una instantánea confirmada. Si descubres una confirmación errónea, revertirla es una forma fácil y segura de eliminarla por completo del código base.
- **git status**

  Muestra el estado del directorio en el que estás trabajando y la instantánea preparada.

Git Flow

Gitflow es un modelo alternativo de ramificación de Git que implica el uso de ramas de características y múltiples ramas primarias. Fue publicado y popularizado por primera vez por Vincent Driessen en nvie. En comparación con el desarrollo basado en el tronco, Gitflow tiene numerosas ramas de mayor duración y commits más grandes. Bajo este modelo, los desarrolladores crean una rama de características y retrasan su fusión con la rama principal del tronco hasta que la característica esté completa. Estas ramas de características de larga duración requieren más colaboración para fusionarse y tienen un mayor riesgo de desviarse de la rama troncal. También pueden introducir actualizaciones conflictivas.

Gitflow se puede utilizar para proyectos que tienen un ciclo de lanzamiento programado y para la mejor práctica DevOps de entrega continua. Este flujo de trabajo no añade nuevos conceptos o comandos más allá de lo que se requiere para el flujo de trabajo Feature Branch.

Trunk Based Development

El desarrollo basado en troncos es una práctica de gestión de control de versiones en la que los desarrolladores fusionan pequeñas actualizaciones de forma frecuente en un "tronco" o rama principal. Se ha convertido en una práctica habitual entre los equipos de DevOps y parte del ciclo de vida de DevOps, ya que simplifica las fases de fusión e integración. De hecho, el desarrollo basado en troncos es una práctica obligatoria de la CI y la CD. Permite a los desarrolladores crear ramas de corta duración con pequeñas confirmaciones, a diferencia de otras estrategias de ramas de funciones de larga duración. A medida que la complejidad del código base y el tamaño del equipo van creciendo, el desarrollo basado en troncos ayuda a mantener el flujo de publicación de la producción.

Fuentes:

Atlassian. *Comandos básicos de Git*. Atlassian. Recuperado el 23 de noviembre de 2023, de https://www.atlassian.com/es/git/glossary

Atlassian. *Desarrollo basado en troncos*. Atlassian. Recuperado el 23 de noviembre de 2023, de https://www.atlassian.com/es/continuous-delivery/continuous-integration/trunk-based-development

Atlassian. *Gitflow Workflow*. Atlassian. Recuperado el 23 de noviembre de 2023, de https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

Desktop, N. (27 de enero del 2020). Git branches: List, create, switch to, merge, push, & delete. *Nobledesktop.com*. Recuperado el 23 de noviembre de 2023, de https://www.nobledesktop.com/learn/git/git-branches

Git. Git-scm.com. Recuperado el 23 de noviembre de 2023, de https://git-scm.com

Git - git-init documentation. Git-scm.com. Recuperado el 23 de noviembre de 2023, de https://git-scm.com/docs/git-init

Schiestl, B. (28 del febrero 2020). *Code branching definition — what is a branch?* Perforce Software; Perforce. Recuperado el 23 de noviembre de 2023, de https://www.perforce.com/blog/vcs/branching-definition-what-branch