PRESENTACIÓN

NOMBRE: Diorkys

APELLIDOS: Cabrera Del Rosario

CARRERA: Desarrollo de Software

MAESTRO: Kelyn Tejeda

MATERIA: Programación III

MATRÍCULA: 2022-2115

1. **¿Qué es Git?**

Git es un sistema de control de versiones distribuido ampliamente utilizado para el seguimiento de cambios en archivos de código fuente durante el desarrollo de software. Fue creado por Linus Torvalds en 2005 para ayudar en el desarrollo del kernel de Linux, pero desde entonces se ha convertido en una herramienta fundamental en la industria del desarrollo de software.

Git permite a los desarrolladores trabajar en un proyecto de forma colaborativa y simultánea, manteniendo un historial completo de todos los cambios realizados en los archivos. Esto permite rastrear quién realizó qué cambios, cuándo se realizaron y por qué se realizaron. Además, Git facilita la gestión de ramas, lo que permite a los desarrolladores trabajar en nuevas características o correcciones de errores de forma aislada sin afectar al código principal.

Algunos conceptos clave en Git incluyen:

- **Repositorio:** Es el almacenamiento principal de un proyecto en Git, donde se almacenan todos los archivos y su historial de cambios.

- **Commit:** Es una instantánea de los cambios realizados en los archivos en un momento específico. Los commits se utilizan para guardar cambios de forma permanente en el historial del repositorio.
- **Rama (branch):** Es una versión paralela del código en el repositorio. Las ramas se utilizan para desarrollar nuevas características o solucionar problemas sin afectar el código principal del proyecto.
- **Merge**: Es el proceso de combinar los cambios de una rama a otra, generalmente fusionando una rama de características en la rama principal del proyecto.
- **Pull Request:** Es una solicitud para fusionar los cambios realizados en una rama con otra. Se utiliza comúnmente en proyectos de código abierto y en equipos de desarrollo para revisar y discutir los cambios antes de fusionarlos.

2. ¿Cuál es el propósito del comando git init en Git?

El comando git init se utiliza para inicializar un nuevo repositorio Git en un directorio de trabajo existente o vacío. Cuando se ejecuta este comando en un directorio, Git crea un nuevo repositorio vacío en esa ubicación. El propósito principal del comando git init es establecer un nuevo repositorio Git en un directorio para comenzar a realizar un seguimiento de los cambios en los archivos en ese directorio.

Una vez que se ha inicializado un repositorio Git con git init, Git comienza a rastrear los cambios en los archivos dentro del directorio y su subdirectorio.

Los archivos y directorios en el repositorio se convierten en objetos rastreados por Git, lo que significa que Git registrará los cambios en estos archivos y permitirá realizar operaciones como confirmaciones, ramificación y fusión.

3. ¿Qué representa una rama en Git y cómo se utiliza?

Una rama en Git es una línea de desarrollo independiente que permite trabajar en características específicas o correcciones de errores de forma aislada sin afectar el código principal del proyecto. En términos más simples, una rama es una versión paralela del código en un repositorio Git.

Cuando se crea un repositorio Git, se crea automáticamente una rama principal predeterminada llamada master (aunque en algunos casos puede ser llamada main). Además de la rama principal, los desarrolladores pueden crear y trabajar en ramas adicionales según sea necesario.

El proceso básico para utilizar una rama en Git es el siguiente:

Creación de una nueva rama: Se crea una nueva rama utilizando el comando git branch seguido del nombre de la nueva rama. Por ejemplo, para crear una nueva rama llamada "feature-x", se ejecutaría el siguiente comando: git branch feature-x.

Cambio a una rama específica: Para cambiar a la nueva rama que acabamos de crear, se utiliza el comando git checkout seguido del nombre de la rama.

Por ejemplo, para cambiar a la rama "feature-x", se ejecutaría el siguiente comando: git checkout feature-x. A partir de Git 2.23, también puedes utilizar git switch en lugar de git checkout.

Realización de cambios en la rama: Una vez en la rama deseada, se pueden realizar cambios en los archivos del proyecto como se desee. Estos cambios solo afectarán a la rama actual y no afectarán a otras ramas.

Confirmación de cambios: Después de realizar los cambios necesarios, se pueden confirmar en la rama utilizando el comando git commit. Esto guardará los cambios en la historia de la rama actual.

Fusionar cambios: Una vez que se han completado los cambios en una rama y se desean incorporar a la rama principal u otra rama, se puede fusionar la rama actual con la rama objetivo utilizando el comando git merge.

Eliminar una rama: Una vez que los cambios de una rama se han fusionado en otra rama y ya no se necesitan, se puede eliminar la rama utilizando el comando git branch -d seguido del nombre de la rama. Por ejemplo, para eliminar la rama "feature-x", se ejecutaría el siguiente comando: git branch -d feature-x.

4. ¿Cómo puedo determinar en qué rama estoy actualmente en Git?

Para determinar en qué rama te encuentras actualmente en Git, puedes utilizar el comando git branch con la opción -v, que te mostrará una lista de todas las ramas en tu

repositorio junto con un asterisco (*) antes del nombre de la rama en la que te encuentras actualmente. Aquí está el comando:

git branch -v

El resultado puede ser algo así:

```
 master      93d1141 Merge branch 'feature-x'
* feature-x   93d1141 Implement feature X
 develop     e3a8723 Update README.md
```

En este ejemplo, el asterisco (*) indica que la rama actual es "feature-x". Las otras ramas también se enumeran, junto con el último commit en cada una de ellas.

Además, también puedes utilizar el comando git status, que te proporcionará información sobre el estado actual del repositorio, incluyendo la rama en la que te encuentras:

git status

El resultado puede incluir algo similar a esto:

```
On branch feature-x
Your branch is up to date with 'origin/feature-x'.
nothing to commit, working tree clean
```

En este caso, la línea "On branch feature-x" indica que te encuentras en la rama "feature-x".

5. ¿Quién es la persona responsable de la creación de Git y cuándo fue desarrollado?

Git fue creado por Linus Torvalds, quien es mejor conocido como el creador del kernel de Linux. Linus desarrolló Git en 2005 como respuesta a la necesidad de un sistema de control de versiones distribuido eficiente para el desarrollo del kernel de Linux y otros proyectos de código abierto. A lo largo de los años, Git ha ganado una enorme popularidad y se ha convertido en una herramienta fundamental en el desarrollo de software en todo el mundo.

6. ¿Cuáles son algunos de los comandos esenciales de Git y para qué se utilizan?

Algunos de los comandos esenciales de Git y sus usos principales:

- **git init:** Inicializa un nuevo repositorio Git en un directorio.
- **git clone**: Clona un repositorio Git existente desde un servidor remoto a tu máquina local.

- **git add:** Agrega cambios en archivos al área de preparación para ser confirmados en el próximo commit.
- **git commit:** Guarda los cambios confirmados en el repositorio.
- **git status:** Muestra el estado actual del repositorio, incluyendo los archivos modificados, agregados y eliminados, así como la rama actual.
- **git branch:** Lista, crea o elimina ramas. También se utiliza para cambiar entre ramas o mostrar información detallada sobre ellas.
- **git checkout:** Cambia entre diferentes ramas o restaura archivos de la versión actual o de una rama específica.
- **git merge:** Fusiona cambios de una rama a otra. Por lo general, se utiliza para combinar una rama de características en la rama principal.
- **git pull:** Obtiene y fusiona los cambios del repositorio remoto en el repositorio local.
- **git push:** Envía los commits locales al repositorio remoto.
- **git log:** Muestra el historial de commits del repositorio, incluyendo información como el autor, la fecha y el mensaje del commit.
- **git diff:** Muestra las diferencias entre archivos en el directorio de trabajo y el área de preparación, o entre commits, ramas o cualquier otro objeto Git.

7. ¿Puedes mencionar algunos de los repositorios de Git más reconocidos y utilizados en la actualidad?

Los repositorios de Git más reconocidos y utilizados en la actualidad:

Linux Kernel: El repositorio oficial del kernel de Linux, alojado en https://github.com/torvalds/linux.

React: Un repositorio que contiene el código fuente de React, una biblioteca de JavaScript utilizada para construir interfaces de usuario interactivas, alojado en https://github.com/facebook/react.

TensorFlow: Un repositorio que contiene el código fuente de TensorFlow, una biblioteca de código abierto para aprendizaje automático y redes neuronales, alojado en https://github.com/tensorflow/tensorflow.

Visual Studio Code: El repositorio oficial de Visual Studio Code, un editor de código fuente desarrollado por Microsoft, alojado en https://github.com/microsoft/vscode.

Bootstrap: Un repositorio que aloja el código fuente de Bootstrap, un popular framework de desarrollo front-end, alojado en https://github.com/twbs/bootstrap.

Docker: Un repositorio que contiene el código fuente de Docker, una plataforma de virtualización de contenedores, alojado en https://github.com/docker/docker-ce.

Angular: Un repositorio que aloja el código fuente de Angular, un framework de desarrollo web desarrollado por Google para construir aplicaciones web de una sola página (SPA), alojado en https://github.com/angular/angular.

Diorkys Cabrera Del Rosario