



NOMBRE: GABRIEL

APELLIDO: NÚÑEZ PAULINO

MATRICULA: 2022-1034

PERIODO ACADEMICO: 2024-C1

FECHA DE ENTREGA: 3/4/2024

PROFESOR: KELYN TEJEDA BELLIARD

TEMA DE ESTUDIO: Tarea 3

Calificación: la que indica la plataforma.

Punto1:

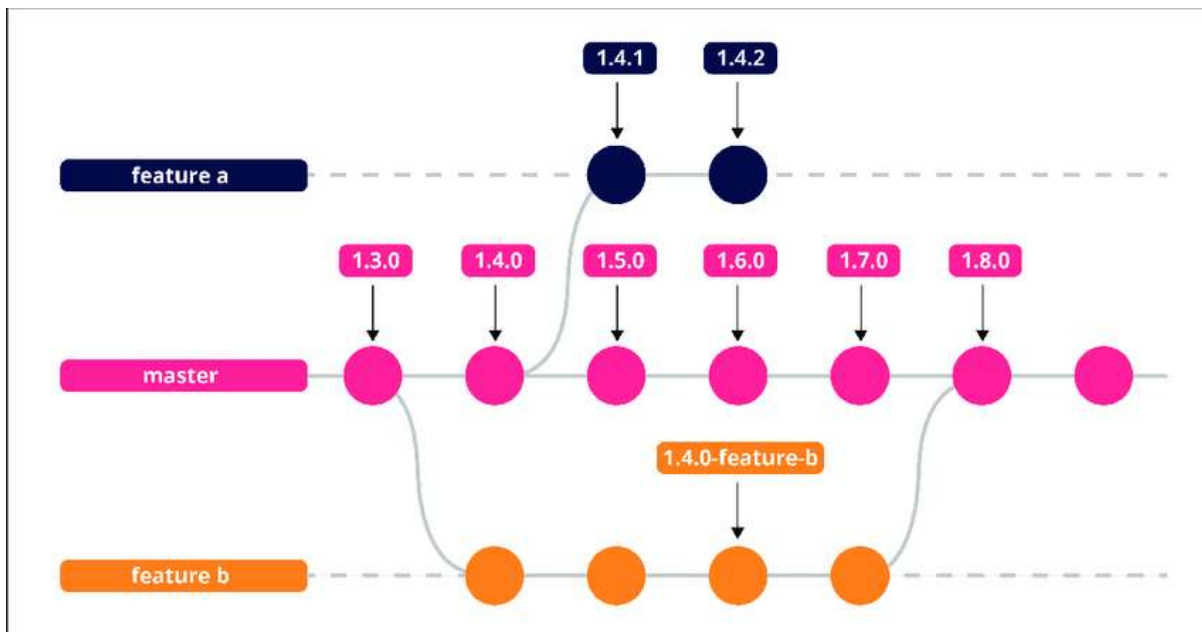
Desarrolla el siguiente Cuestionario

- 1. ¿Qué es Git?**
- 2. ¿Cuál es el propósito del comando git init en Git?**
- 3. ¿Qué representa una rama en Git y cómo se utiliza?**
- 4. ¿Cómo puedo determinar en qué rama estoy actualmente en Git?**
- 5. ¿Quién es la persona responsable de la creación de Git y cuándo fue desarrollado?**
- 6. ¿Cuáles son algunos de los comandos esenciales de Git y para qué se utilizan?**
- 7. ¿Puedes mencionar algunos de los repositorios de Git más reconocidos y utilizados en la actualidad?**

¿Qué es Git?

Hoy en día, Git es, con diferencia, el sistema de control de versiones moderno más utilizado del mundo. Git es un proyecto de código abierto maduro y con un mantenimiento activo que desarrolló originalmente Linus Torvalds, el famoso creador del kernel del sistema operativo Linux, en 2005. Un asombroso número de proyectos de software dependen de Git para el control de versiones, incluidos proyectos comerciales y de código abierto. Los desarrolladores que han trabajado con Git cuentan con una buena representación en la base de talentos disponibles para el desarrollo de software, y este sistema funciona a la perfección en una amplia variedad de sistemas operativos e IDE (entornos de desarrollo integrados).

Las características básicas de rendimiento de Git son muy sólidas en comparación con muchas otras alternativas. La confirmación de nuevos cambios, la ramificación, la fusión y la comparación de versiones anteriores se han optimizado en favor del rendimiento. Los algoritmos implementados en Git aprovechan el profundo conocimiento sobre los atributos comunes de los auténticos árboles de archivos de código fuente, cómo suelen modificarse con el paso del tiempo y cuáles son los patrones de acceso.



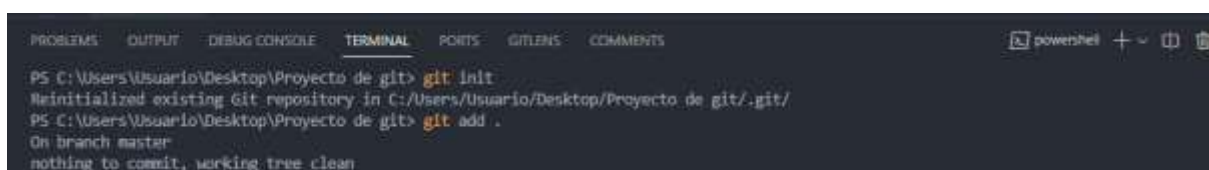
¿Cuál es el propósito del comando git init en Git?

El propósito del comando `git init` en Git es inicializar un nuevo repositorio Git en un directorio existente. Esto significa que se creará un nuevo repositorio Git en la carpeta específica donde se ejecute el comando, lo que permitirá realizar un seguimiento de los cambios en los archivos de ese directorio, además `git init` es el primer comando para comenzar a utilizar Git en un proyecto nuevo o existente.

Al ejecutar `git init`, se crea un subdirectorio de `.git` en el directorio de trabajo actual, que contiene todos los metadatos de Git necesarios para el nuevo repositorio. Estos metadatos incluyen subdirectorios de objetos, referencias y archivos de plantilla. También se genera un archivo `HEAD` que apunta a la confirmación actualmente extraída.

A screenshot of a terminal window titled 'richardkalehoff — bash — bash — 58x10'. The user is in a directory named 'new-git-project'. The command '\$ git init' has been executed, resulting in the output: 'Initialized empty Git repository in /Users/richardkalehoff/Courses/Git-Course/new-git-project/.git/'. The prompt now shows 'richardkalehoff (master #) new-git-project' followed by a new '\$' prompt.

```
richardkalehoff new-git-project
$ git init
Initialized empty Git repository in /Users/richardkalehoff/
Courses/Git-Course/new-git-project/.git/
richardkalehoff (master #) new-git-project
$
```

A screenshot of a VS Code terminal window. The terminal shows the execution of 'git init' and 'git add .' commands. The output indicates that an existing Git repository was reinitialized and that the working tree is clean.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS COMMENTS
PS C:\Users\Usuario\Desktop\Proyecto de git> git init
Reinitialized existing Git repository in C:\Users\Usuario\Desktop\Proyecto de git/.git/
PS C:\Users\Usuario\Desktop\Proyecto de git> git add .
On branch master
nothing to commit, working tree clean
```

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

Las ramas son una de las principales utilidades que disponemos en Git para llevar un mejor control del código. Se trata de una bifurcación del estado del código que crea un nuevo camino de cara a la evolución del código, en paralelo a otras ramas que se puedan generar. En este artículo, repasamos para qué sirven las ramas de Git y cómo podemos trabajar con ellas en un proyecto.

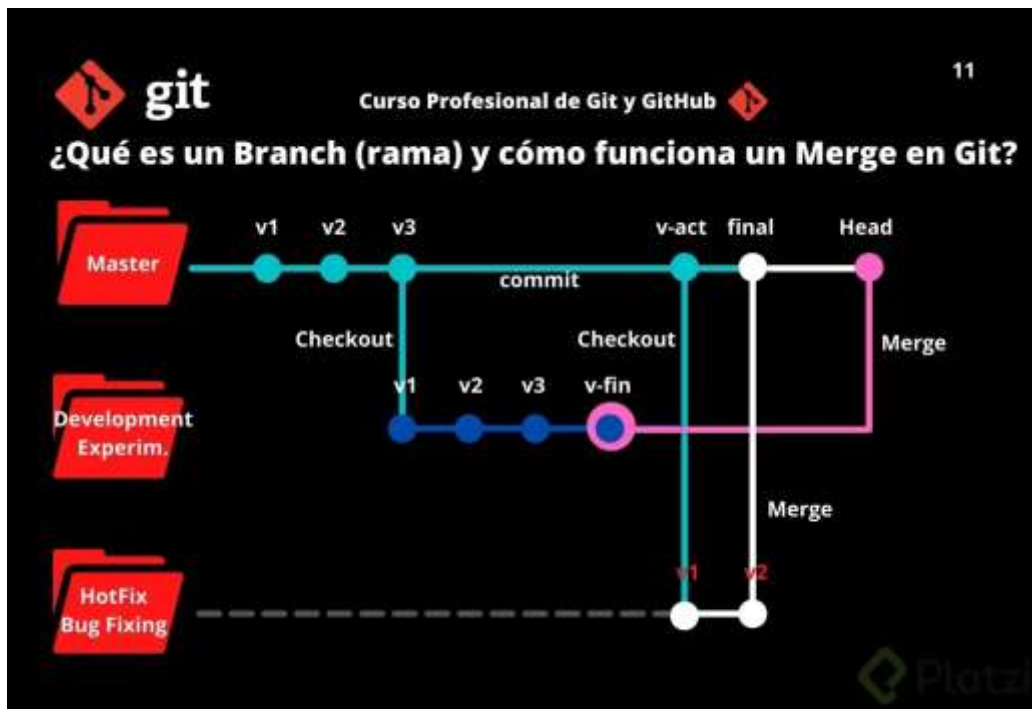
Las ramas nos pueden servir para muchos casos de uso. Por ejemplo, tras la elección del hosting para crear una página web o desarrollar una tienda online, para la creación de una funcionalidad que queramos integrar en un programa y para la cual no queremos que la rama principal se vea afectada. Esta función experimental se puede realizar en una rama independiente, de modo que, aunque tardemos varios días o semanas en terminarla, no afecte a la producción del código que tenemos en la rama principal y que permanecerá estable.

La utilidad principal de las ramas en Git incluye:

Desarrollo Paralelo: Permiten a los desarrolladores trabajar en diferentes características o correcciones de errores al mismo tiempo, sin interferir entre sí.

Experimentación: Las ramas proporcionan un entorno seguro para probar nuevas ideas o cambios sin afectar al código principal del proyecto.

Aislamiento de Cambios: Ayudan a mantener los cambios relacionados con una característica o corrección de errores separados del resto del código, facilitando la revisión, la prueba y la reversión si es necesario.



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

Para saber qué ramas están disponibles y cuál es el nombre de la rama actual, ejecuta **“git branch”**.

```
PS C:\Users\Usuario\Desktop\Proyecto de git> git branch
Dev
QA
feature/Cambio-de-login
feature/Cambios-de-funcionalidad
* master
PS C:\Users\Usuario\Desktop\Proyecto de git>
```

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



Git fue creado por Linus Torvalds, el mismo creador del kernel de Linux. Comenzó el desarrollo de Git en 2005 debido a la necesidad de un sistema de control de versiones distribuido para el desarrollo del kernel de Linux. La primera versión estable de Git, la 1.0, fue lanzada el 21 de diciembre de 2005. Desde entonces, Git ha ganado una amplia adopción en la comunidad de desarrollo de software debido a su velocidad, flexibilidad y robustez.

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

git init: Inicializa un nuevo repositorio Git en el directorio actual o en el directorio especificado.

git clone : Clona un repositorio Git existente desde una URL remota a tu máquina local.

git add : Agrega cambios de archivos al área de preparación (staging area) para ser incluidos en el próximo commit.

git commit -m : Crea un nuevo commit con los cambios en el área de preparación y un mensaje descriptivo.

git status: Muestra el estado actual del repositorio, incluyendo archivos modificados, archivos en el área de preparación y el estado de la rama.

git branch: Lista todas las ramas en el repositorio local y muestra la rama actual.

git checkout : Cambia a la rama especificada.

git merge : Fusiona los cambios de la rama especificada en la rama actual.

git pull: Obtiene y fusiona los cambios desde el repositorio remoto a la rama actual.

git push: Envía los commits locales al repositorio remoto.

git log: Muestra el historial de commits en la rama actual.

git diff : Muestra las diferencias entre los cambios en el área de trabajo y el último commit.

Tecsify presenta...

Comandos básicos de git

GIT es un sistema de **control de versiones**, es decir: un sistema que registra los cambios en un archivo o conjunto de archivos a lo largo del tiempo, esto con el fin de que más adelante no sea un problema obtener versiones anteriores específicas del archivo.

Veamos algunos de los comandos básicos de GIT:

 GIT INIT Inicia un nuevo repositorio, esto creará el "staging" o área de ensayo y un repositorio local.	 GIT COMMIT Guarda los archivos en el repositorio local. Ejemplo: <code>"git commit -m 'Comentario sobre la actualización'"</code>
 GIT CLONE Obtiene una copia de un proyecto que se encuentra en un repositorio público: <code>"git clone [url]"</code>	 GIT PUSH Envía los commits al repositorio remoto. Importante: Git push solamente carga los cambios que han sido confirmados.
 GIT ADD Añade un archivo al área de ensayo. Ejemplo: <code>"git add archivo"</code> o <code>"git add ."</code> para añadir todo.	 GIT PULL Extrae y descarga contenido desde un repositorio remoto y actualiza al instante el repositorio local reflejando ese contenido.
 GIT BRANCH Permite crear, enumerar y eliminar ramas, así como cambiar su nombre.	 GIT MERGE Integra las características de tu rama con todos los commits realizados a las demás ramas del repositorio.
 GIT CHECKOUT Se utiliza principalmente para cambiarse de una rama a otra y para chequear archivos y commits.	 GIT REVERT Deshace cambios efectuados en el historial de confirmaciones de un repositorio.
 GIT STATUS Brinda toda la información necesaria sobre los archivos de la rama actual.	 GIT CONFIG Establece parámetros que Git utiliza por defecto, ejemplo el autor: <code>"git config --global user.name 'Tecsify'"</code>

Tecsify

[/Tecsify](#) [@Tecsify](#) [@Tecsify](#) [www.Tecsify.com/blog](#) [/Tecsify](#) [Tecsify](#) [@Tecsify](#)

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

GitLab



GitLab es una plataforma de gestión de ciclo de vida de desarrollo de software (SDLC) basada en web que proporciona un conjunto completo de herramientas para la gestión de proyectos, la colaboración de equipos y el control de versiones utilizando Git.

Github



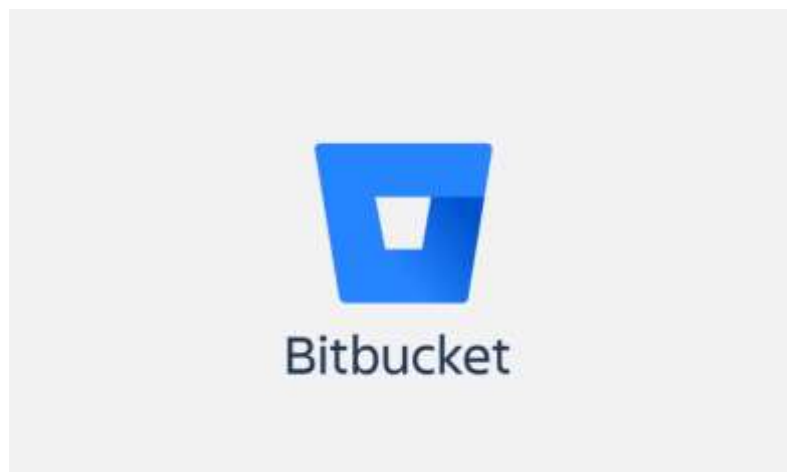
GitHub es una plataforma de alojamiento de repositorios de código basada en la web que utiliza el sistema de control de versiones Git. Es ampliamente utilizada por desarrolladores de software y equipos de desarrollo para colaborar en proyectos, gestionar el control de versiones del código fuente y realizar un seguimiento de los problemas.

AzureDevops



Azure DevOps es una plataforma de colaboración y desarrollo de software basada en la nube proporcionada por Microsoft. Ofrece un conjunto de herramientas integradas que cubren todo el ciclo de vida del desarrollo de software, desde la planificación y el seguimiento hasta la integración continua, la entrega continua y el monitoreo.

Bitbucket



Bitbucket es una plataforma de alojamiento de repositorios de código basada en la nube que utiliza los sistemas de control de versiones Git y Mercurial. Es ampliamente utilizada por equipos de desarrollo de software para alojar, colaborar y gestionar el código fuente de sus proyectos.