

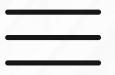


GIT

---

DANNA VALERIA MORALES AGUILAR





# INTRODUCCIÓN A GIT

Git es un sistema de control de versiones distribuido y se suele utilizar en el desarrollo de software. Fue creado por Linus Trovalds en 2005. Su objetivo principal es permitir a los equipos de desarrollo rastrear y gestionar los cambios realizados en el código fuente en lugar de tener una única copia del código en un repositorio central, en Git cada miembro del equipo tiene su propia copia completa del historial del proyecto.





# VENTAJAS VS DESVENTAJAS



## Ventajas

- Facil distribucion
- Alta velocidad
- Ramificacion y fusion sencillas
- Integridad de datos
- Amplia comunidad y soporte



## Desventajas

- Comenzar a aprenderlo lleva su tiempo
- Cuando varios desarrolladores trabajan en la misma parte del código y realizan cambios que entran en conflicto
- Tamaños grandes de espacio afecta el rendimiento
- Perdida de datos sin confirmar

# COMANDOS BÁSICOS

## GIT CLONE

Para descargarte el código fuente existente desde un repositorio remoto

```
git clone <https://link-con-nombre-del-repositorio>
```

## GIT BRANCH

Para crear, listar y eliminar ramas.

```
git branch -d <nombre-de-la-rama>
```

## GIT CHECKOUT

Para cambiar de una rama a otra.

```
git checkout -b <nombre-de-tu-rama>
```

## GIT STATUS

Nos da toda la información necesaria sobre la rama actual.

```
git status
```

## GIT ADD

Para incluir los cambios de los archivos en el siguiente commit.

```
git add <archivo>
```

# COMANDOS BÁSICOS



```
git commit -m "mensaje de confirmación"
```

## GIT COMMIT

Establece un punto de control en el proceso de desarrollo al cual se puede volver mas tarde.

```
git push <nombre-remoto> <nombre-de-tu-rama>
```

## GIT PUSH

Envia tus commits al repositorio remoto.

```
git pull <nombre-remoto>
```

## GIT PULL

Para recibir actualizaciones del respositorio remoto.

```
git revert 3321844
```

## GIT REVERT

Para deshacer cambios que hemos hecho.

```
git merge <nombre-de-la-rama>
```

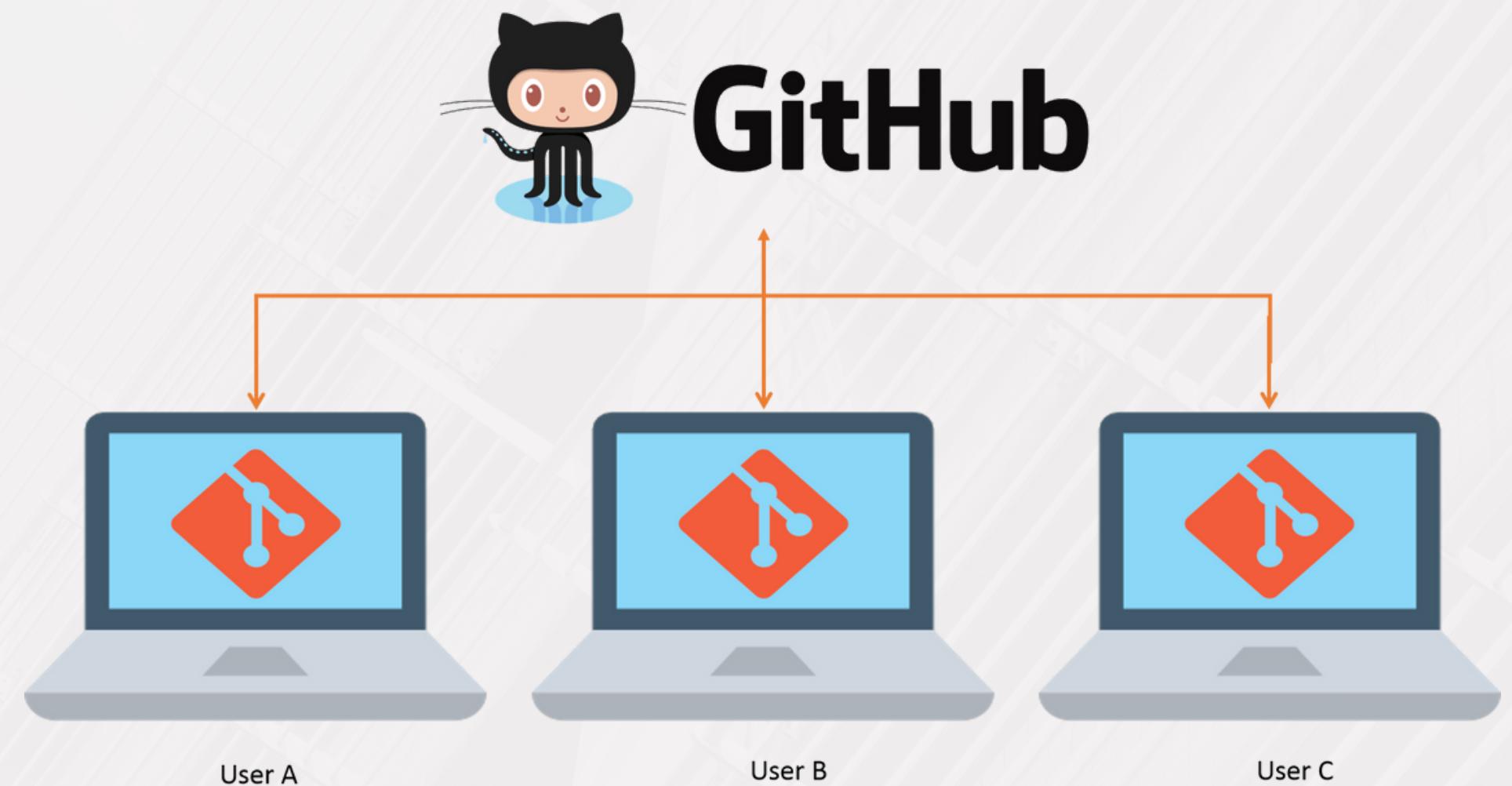
## GIT MERGE

Para fusionar la rama con su rama padre.

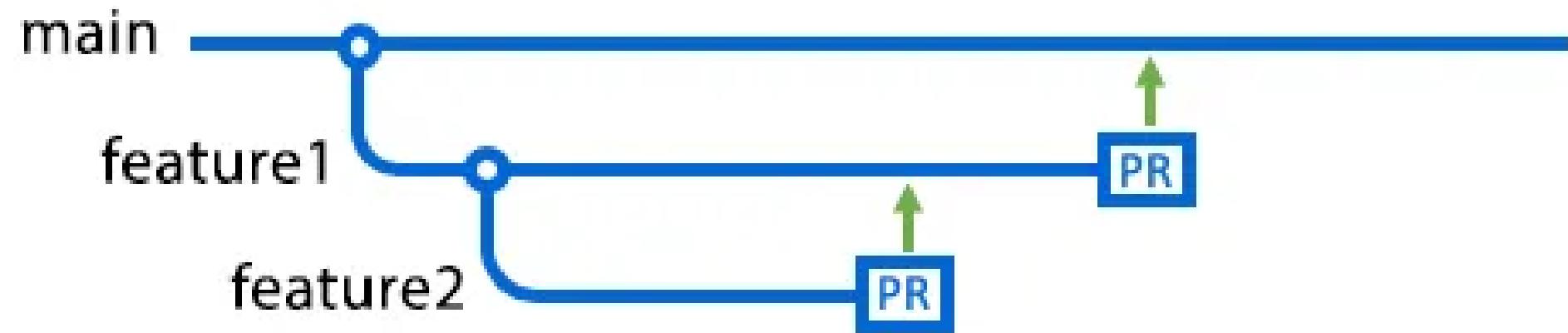
# REPOSITORIOS

Un repositorio es un espacio donde se almacena y gestiona el código fuente y los archivos de un proyecto.

Un repositorio contiene todos los archivos de tu proyecto y el historial de revisiones de cada uno de ellos. Se puede administrar el trabajo del proyecto dentro del repositorio.



# BRANCHES



Las flechas indican la rama base actual para cada solicitud de extraccion, feature1 es la rama base de feauture2.  
Si se combina la solicitud de incorporacion de cambios de feature2, esta se combinara con feature1.

Se usan para identificar tareas de desarrollo sin afectas otras ramas del repositorio. Al crear una nueva rama, se toma una copia exacta de la rama principal llamada master o main, y se trabaja con ella de forma aislada.

Son utiles para desarrollar nuevas caracteristicas, solucionar problemas o realizar experimentos sin comprometer la estabilidad del código en la rama principal.

# MERGE

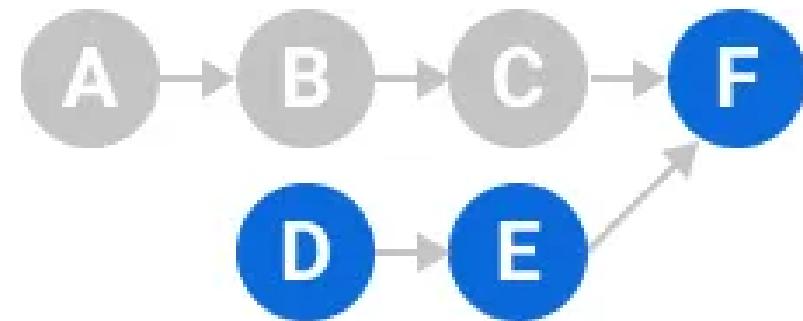
Main



Feature



Main



Merge commit: D + E added to Main via F

Es el proceso de combinar los cambios realizados en una rama con otra.

Despues de desarrollar una nueva caracteristica o correccion en una rama secundaria, es comun fusionar esos cambios con la rama principal una vez que se ha probado y se considera lista para la produccion.

La fusion se encarga de combinar los cambios de ambas ramas, aplicandolos en orden cronologico segun los commits realizados en cada rama. La fusion se realiza automaticamente si no hay conflictos entre los cambios de ambas ramas.



# CONFLICTS

Los conflictos suceden cuando hay cambios en el mismo archivo o línea de código entre dos ramas que se estan fusionando. Esto suele suceder cuando dos desarrolladores han modificado la misma parte del código de manera diferente en sus propias ramas.

Cuando se tiene un conflicto Git no puede determinar como combinar los cambios y solicita que el usuario resuelva este

```
$ git merge BRANCH-NAME
> Auto-merging styleguide.md
> CONFLICT (content): Merge conflict in styleguide.md
> Automatic merge failed; fix conflicts and then commit the
result
```



# PASOS PARA RESOLVER CONFLICTOS

1. Hacer clic en Solicitudes de incorporación de cambios.

A screenshot of a GitHub repository page for 'github/docs'. The top navigation bar shows 'Code', 'Issues 104', 'Pull requests 47', 'Discussions', 'Actions', 'Projects 5', 'Security 6', 'Insights', and 'Settings'. The 'Pull requests' button is highlighted with an orange border. Below the navigation, it shows 'main' branch, '21 branches', and '1 tag'. At the bottom, there are buttons for 'Go to file', 'Add file', 'Code', and 'About'.

2. En la lista de "Pull Requests", hacer clic en la solicitud de extracción con un conflicto de fusión que quieras resolver.

3. Junto a la parte inferior de la solicitud de incorporación de cambios, haga clic en Resolver conflictos.

A screenshot of a 'Resolve conflicts' dialog box. It displays a warning message: 'This branch has conflicts that must be resolved. Use the web editor or the command line to resolve conflicts.' Below this, under 'Conflicting files', it lists 'content/issues/tracking-your-work-with-issues/linking-a-pull-request-to-an-issue.md'. A large orange 'Resolve conflicts' button is at the bottom right.

4. Decidir si maneter los cambios de las otras ramas o hacer un cambio nuevo.

5. Una vez que haya resuelto todos los conflictos en el archivo, haga clic en Marcar como resueltos.

A screenshot of a GitHub commit history for 'content/issues/tracking-your-work-with-issues/linking-a-pull-request-to-an-issue.md'. The commit message is 'Mark as resolved'. The commit details show the file content: '---', 'title: Linking a pull request to an issue', 'intro: You can link a pull request %ifversion link-existing-branches-to-issue %or branch %endif %to an issue to show that a fix is redirect\_from: - /github/managing-your-work-on-github/managing-your-work-with-issues-and-pull-requests/linking-a-pull-request-to-an-issue - /articles/closing-issues-via-commit-message - /articles/rincion-issues-via-commit-mecanismo'. The commit is marked with a red '1 conflict' badge.

6. Seleccionar el siguiente archivo que quieras editar del lado izquierdo de la página en "conflicting files" y se repiten los datos.

7. Hacer clic en Confirmar combinación y por último hacer clic en Combinar solicitud .

A screenshot of a GitHub commit history for 'content/issues/tracking-your-work-with-issues/linking-a-pull-request-to-an-issue.md'. The commit message is 'Commit merge'. The commit details show the file content: '---', 'title: Linking a pull request to an issue', 'intro: You can link a pull request %ifversion link-existing-branches-to-issue %or branch %endif %to an issue to show that a fix is redirect\_from: - /github/managing-your-work-on-github/managing-your-work-with-issues-and-pull-requests/linking-a-pull-request-to-an-issue - /articles/closing-issues-via-commit-message - /articles/rincion-issues-via-commit-mecanismo'. The commit is marked with a green checkmark and the word 'Resolved'.



# CONCLUSIONES

---

Los branches permiten desarrollar características o solucionar problemas de forma aislada, el merge permite combinar los cambios en una rama principal y los conflicts se deben resolver manualmente cuando hay cambios conflictivos entre las ramas que se desean fusionar.

Esto es importante para facilitar la colaboración y el desarrollo de proyectos de manera eficiente y organizada.

