



26/07/2024

Capítulo 2: Flujo de trabajo básico

El flujo de trabajo en Git se refiere a la manera en que los desarrolladores utilizan la herramienta de Git para gestionar y colaborar en proyectos en software.

Este capítulo explora los flujos de trabajo más comunes, proporcionando una guía sobre cómo establecer un proceso efectivo para el desarrollo de.

Flujo Básico

El flujo básico es ideal para proyectos pequeños o desarrollos unipersonales. En este modelo, el desarrollador realiza cambios localmente y luego los sube al repositorio remoto. Los pasos típicos incluyen:

1. Crea un nuevo repositorio.


```
bash  
git init
```

2. Hacer un cambio en los archivos.


3. Agregar los cambios al área de preparación (Stating Area)

```
bash  
git add <nombre-del-archivo>
```

4. Confirmar los cambios.

```
bash
git commit -m "Descripción de los cambios" 
```

5. Subir los cambios al repositorio remoto:

```
bash
git push origin master 
```

Este flujo es sencillo y permite al desarrollador trabajar de manera independiente, pero puede no ser suficiente para proyectos más grandes o colaborativos.

Flujo Centralizado

El flujo centralizado es útil para equipos que migran desde sistemas de control de versiones centralizados como Subversion (SVN). En este modelo, todos los cambios se realizan en una rama principal (generalmente llamada **main** o **master**). Las características clave incluyen:

- **Un solo punto de entrada:** Todos los desarrolladores sincronizan sus cambios con un único repositorio central.
- **Control de cambios:** Git evita que se realicen **push** si hay cambios no sincronizados en el repositorio remoto, lo que ayuda a prevenir conflictos.

Este enfoque es familiar para muchos desarrolladores, pero puede limitar la flexibilidad que ofrece Git.

Flujo de Rama de Funcionalidad

Este flujo implica crear ramas específicas para nuevas funcionalidades. Los pasos son:

1. **Crear una nueva rama para la funcionalidad.**

```
bash
git checkout -b feature/nueva-funcionalidad
```

2. **Desarrollar la funcionalidad en esa rama.**

3. **Integrar la funcionalidad en la rama principal.**

```
bash
git checkout master
git merge feature/nueva-funcionalidad
```

Este flujo evita que la rama principal se vea afectada por cambios inestables y permite que varios desarrolladores trabajen simultáneamente en diferentes características.

Flujo GitFlow

GitFlow es un modelo más estructurado que define ramas específicas para diferentes propósitos:

- **Rama **main**:** Contiene el código listo para producción.
- **Rama **develop**:** Contiene los últimos cambios que se están desarrollando.
- **Ramas de características (Feature):** Se crean a partir de **develop** para desarrollar nuevas funcionalidades.
- **Ramas de lanzamiento (Release):** Preparan el código para producción y se integran en **main** y **develop**.
- **Ramas de corrección de errores (Hotfix):** Permiten corregir errores críticos en producción.

Este flujo es adecuado para equipos grandes y proyectos complejos, ya que proporciona una clara separación entre diferentes tipos de trabajo y asegura que la rama principal siempre esté estable.

Conclusión

Elegir el flujo de trabajo adecuado en Git es crucial para la eficiencia y la colaboración en el desarrollo de software. Los flujos básicos, centralizados, de funcionalidad y GitFlow ofrecen diferentes enfoques que pueden adaptarse a las necesidades específicas del equipo y del proyecto. La comprensión de estos flujos permite a los desarrolladores trabajar de manera más efectiva y organizada.