Git Básico

Arturo Silvelo

Try New Roads

Trucos

Git Cherry-pick

Git Cherry-pick permite aplicar un commit específico de otra rama a la rama actual, sin necesidad de fusionar toda la rama. Es útil para incorporar cambios puntuales sin mezclar todas las modificaciones de una rama.

Uso básico:

- Identifica el hash del commit que quieres aplicar (usualmente con `git log`).
- Ejecuta el comando:

```
git cherry-pick <commit-hash>
```

Ejemplo:
Si quieres aplicar el commit con el hash `abc1234` de una rama `feature` a tu rama actual, usarías:

```
git cherry-pick abc1234
```

Nota: Si hay conflictos durante el cherry-pick, tendrás que resolverlos manualmente

Git Bisect - Introducción

Git Bisect es una herramienta que permite encontrar un commit específico que introdujo un error en el código utilizando una búsqueda binaria.

Uso básico:

- Inicia el bisecting con el comando:

```
git bisect start
```

- Marca un commit bueno y uno malo:

```
git bisect good <commit-hash>
git bisect bad <commit-hash>
```

Git hará una búsqueda binaria y te pedirá que marques cada commit como bueno o malo hasta encontrar el commit defectuoso.

Git Bisect - Ejemplo y Finalización

Ejemplo de uso:

Si sabes que el último commit estaba funcionando y el actual no, puedes usar:

```
git bisect start
git bisect good <last-good-commit>
git bisect bad <current-commit>
```

Finaliza el bisecting:

Para salir del modo bisecting y volver al estado original:

```
git bisect reset
```

Volver a la rama previa

En Git, para volver a la rama en la que estuviste trabajando antes de la rama actual, puedes utilizar dos métodos:

- **Usar el comando `git switch -`:**

```
git switch -
```

- **Usar `@{-1}` para la misma funcionalidad:**

```
git switch @{-1}
```

Ambos comandos permiten cambiar de vuelta a la rama donde estabas antes de hacer el último cambio, sin necesidad de recordar su nombre.

Auto corrector

En Git, puedes habilitar una funcionalidad de auto-corrección para evitar errores tipográficos en los comandos. Esto se puede lograr con el siguiente comando:

- **Habilitar el auto-corrector:**

```
git config --global help.autocorrect 20
```

El valor `20` indica el número de décimas de segundo que Git esperará antes de intentar corregir automáticamente un error de comando. Si se introduce un comando incorrecto, Git lo corregirá automáticamente después de este tiempo. Puedes ajustar el valor según tus necesidades.

Bibliografía

Recomendaciones

Para profundizar en el uso de Git, recomendamos los siguientes recursos:

- **Aprendiendo Git: ¡Domina y comprende Git de una vez por todas!**
  Un recurso práctico y completo para entender y dominar Git. Disponible en:
  https://leanpub.com/aprendiendo-git
- **Pro Git (oficial):**
  Libro oficial de Git, disponible de forma gratuita en múltiples idiomas.
  https://git-scm.com/book/en/v2