Convenciones de commits

Las convenciones de los commits son una especificación para dar significado a los mensajes de los archivos que se modifican y suben a los repositorios. Su usan para identificar el objetivo que tiene el o los archivos subidos. Proporciona un conjunto sencillo de reglas para crear un historial de commits explícito

El mensaje del commit debe ser estructurado de la siguiente manera:

<tipo>[ámbito opcional]: <descripción>

[cuerpo opcional]

[nota(s) al pie opcional(es)]

El tipo es el primer elemento del mensaje de commit y define el cambio realizado, como por ejemplo si es una nueva funcionalidad (feat), una corrección de error (fix), una refactorización (refactor) u otro tipo específico. Permite que el historial de cambios sea legible, estructurado y pueda ser filtrado rápidamente.

El ámbito indica la parte específica del sistema que se ve afectada por el commit, sea un módulo, componente o funcionalidad concreta, por ejemplo auth, api, db, ui. Otorga contexto y localiza el cambio dentro de la arquitectura del proyecto, identificando rápidamente qué áreas pueden verse impactadas o qué archivos revisar si surgen conflictos o errores relacionados.

La descripción es un mensaje resumiendo el cambio, escrita de forma imperativa y limitada a 72 letras. Su objetivo es comunicar de manera directa y concisa qué se hizo. El encabezado es lo primero que se ve en herramientas de gestión de código, logs y revisiones, por lo que debe ser conciso.

El cuerpo es opcional, pero recomendado en commits complejos. Explica el motivo del cambio, cómo se implementó y la elección de esa solución. Permite indicar contextos relevantes como decisiones o limitaciones que no se muestran en la descripción. Se describen acciones que pueden afectar otras partes del sistema.

Las notas al pie o footer, se utiliza para enlazar el commit con tickets de seguimiento, issues o tareas específicas como “Refs #133” o “BREAKING CHANGE”. Esto resulta fundamental para mantener la trazabilidad entre el código y la gestión de proyectos, permitiendo saber exactamente qué cambios se implementaron para resolver una tarea o incidente particular.

Un commit que contiene la nota al pie BREAKING CHANGE o que agrega un “!” después del tipo/ ámbito, introduce un cambio de ruptura de API. Puede ser parte de commits de cualquier *tipo*.

Algunos tipos de Commit

Tipo	Uso
feat	Nueva funcionalidad en el código
fix	Corrección de errores en el código
docs	Cambios solo en documentación
style	Cambios de formato, espacios, comas, sin afectar comportamiento
refactor	Refactorización sin cambiar funcionalidad ni corregir errores
perf	Mejoras de rendimiento
test	Añadir o modificar tests
chore	Tareas menores: configs, herramientas, build, etc.
revert	Reversión de un commit anterior

Buenas Prácticas

Los commits deben iniciarse con un prefijo de tipo que consiste en un sustantivo, feat, fix, etc., seguido del ámbito opcional, ! (opcional), dos puntos y un espacio requerido.

Un ámbito puede ser añadido después de un tipo. Un ámbito debe consistir en un sustantivo que describa una sección de la base del código encerrado entre paréntesis, ejemplo: fix(db):

Los cambios de ruptura deben ser indicados en el prefijo de tipo/ámbito de un commit, o como una entrada en la nota al pie.

Cada commit debería cubrir un solo propósito o cambio lógico.

Evitar mensajes genéricos como “arreglos varios” o “cosas”.

Leer el mensaje antes de confirmar el commit.