REFERENCIA RÁPIDA DE CONVENTIONAL COMMITS

Formato de mensaje de commit, tipos, scopes, cambios incompatibles

Formato

Estructura del mensaje de commit

```
<type>[optional scope]: <description>
[optional body]
[optional footer(s)]
```

Partes explicadas

`type`	Categoría del cambio (feat, fix, etc.)
`scope`	Sección del código afectada (opcional)
`description`	Resumen breve en modo imperativo
`body`	Explicación detallada (opcional, tras línea en blanco)
`footer`	Metadatos como BREAKING CHANGE o referencias a issues

Reglas

`Modo imperativo`	"add feature" no "added feature"
`Tipo en minúsculas`	feat: no Feat:
`Sin punto final`	La descripción no termina con "."
`Línea en blanco`	Separar body/footer de la descripción

Tipos

Tipos principales (relevantes para SemVer)

`feat`	Nueva funcionalidad (incrementa versión MINOR)
`fix`	Corrección de bug (incrementa versión PATCH)

Tipos extendidos (convención común)

`build`	Sistema de build o dependencias externas
`chore`	Tareas de mantenimiento (sin cambios en src/tests)
`ci`	Configuración y scripts de CI
`docs`	Solo cambios en documentación
`perf`	Mejora de rendimiento
`refactor`	Cambio de código que no corrige ni agrega
`revert`	Revierte un commit anterior
`style`	Formato, espacios en blanco (no estilos CSS)
`test`	Agregar o corregir tests

Scope

Uso del scope

```
feat(auth): add OAuth2 login flow
fix(parser): handle empty input gracefully
docs(readme): update installation steps
refactor(api): extract validation middleware
```

Guías de scope

`Nombre de módulo`	feat(auth):, fix(parser):
`Nombre de capa`	feat(api):, fix(db):
`Área de funcionalidad`	feat(search):, fix(checkout):
`Dependencia`	build(deps):, chore(npm):
`Omitir si es amplio`	Sin scope para cambios transversales

Cambios incompatibles

Marcar cambios incompatibles

```
feat!: remove deprecated login endpoint
feat(api)!: change response format to JSON:API
fix!: drop Node 14 support
```

Cambio incompatible en footer

```
feat(api): change user endpoint response
BREAKING CHANGE: response now returns array
instead of object. Update client parsing.
```

Reglas

`! after type/scope`	Marcador abreviado de cambio incompatible
`BREAKING CHANGE:`	Token de footer (siempre en mayúsculas)
`BREAKING-CHANGE:`	También válido (con guion)
`Impacto SemVer`	Incrementa versión MAJOR
`Cualquier tipo`	Los cambios incompatibles aplican a cualquier tipo

Ejemplos

Commits simples

```
feat: add email notifications
fix: prevent race condition in checkout
docs: correct typo in contributing guide
style: format with prettier
refactor: simplify error handling logic
```

Con scope

```
feat(blog): add comment threading
fix(auth): refresh token before expiry
test(api): add missing edge case coverage
ci(github): add Node 20 to test matrix
```

Con body y footer

```
fix(parser): handle nested quotes correctly

Previously, nested quotes caused the parser
to enter an infinite loop. This adds a depth
counter to prevent unbounded recursion.

Closes #234
```

Footer

Tokens de footer

`BREAKING CHANGE:`	Describe cambio de API incompatible
`Closes #123`	Cierra issue automáticamente al hacer merge
`Fixes #456`	Cierra issue (referencia de fix)
`Refs #789`	Referencia issue sin cerrarlo
`Reviewed-by: name`	Atribución de revisor
`Co-authored-by: name`	Atribución de co-autor

Múltiples footers

```
feat(api)!: redesign authentication flow

Migrate from session-based to JWT auth.

BREAKING CHANGE: /auth endpoints changed
Closes #101
Refs #98, #99
```

Herramientas

Linting de commits

```
npm install -D @commitlint/cli \
  @commitlint/config-conventional
echo "module.exports = { extends: \
  ['@commitlint/config-conventional'] }" \
  > commitlint.config.js
```

Herramientas populares

`commitlint`	Validar mensajes de commit según la convención
`husky`	Hooks de Git (ejecutar commitlint al hacer commit)
`commitizen (cz)`	Constructor interactivo de mensajes de commit
`standard-version`	Changelog automático + incremento de versión
`semantic-release`	Pipeline de release totalmente automatizado
`release-please`	Herramienta de automatización de releases de Google

Configuración de Commitizen

```
npm install -D commitizen \
  cz-conventional-changelog
npx commitizen init cz-conventional-changelog
# Usar: npx cz (o git cz con alias)
```

Patrones comunes

Mapeo de incremento de versión

`fix:`	PATCH (1.0.0 → 1.0.1)
`feat:`	MINOR (1.0.0 → 1.1.0)
`BREAKING CHANGE`	MAJOR (1.0.0 → 2.0.0)
`docs:, style:, etc.`	Sin incremento de versión

Agrupación en changelog

```
## [1.2.0] - 2026-03-27
### Features
- add email notifications (abc1234)
### Bug Fixes
- prevent race condition (#123) (def5678)
```

Formato de revert

```
revert: feat(blog): add comment threading
This reverts commit abc1234def5678.
```