

### Para que sirve?

Normativa simple de commits Facilita la navegacion del historial

<tipo>(contexto *opcional*):

<descripcion>

[cuerpo *opcional*]

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

### Reglas

1. Los commits **DEBEN** incluir un prefijo con los **tipos** definidos, seguido de dos puntos y espacio. Ej: `build:`
2. El contexto, el body y el pie son opcionales
3. El contexto se define en parentesis. Ej: `feat(s ona rqube) :`  
Este provee mas informacion sobre el bloque de codigo que se esta trabajando. En caso de ser varios archivos se ignora
4. Una **descripcion DEBE** seguir al **tipo**. Se puede definir en **imperativo o conjugado solamente**
5. Los **BREAKING CHANGE DEBEN** contener un body y/o notas al pie indicando que ha cambiado en detalle

### Tipo

fix	solucion a un problema de usuario, desde el codigo
feat	introduce una nueva funcion en el codigo
BREAKING CHANGE o !	Se refiere a un cambio de version mayor
docs	cambios a documentos del proyecto
build	cambios relacionados al build stage o dependencias
limpieza (chore)	organizacion de codigo sin agregar nada nuevo
refactor	no resuelve nada pero ayuda a identificar cambios semanticos
ci	cambios en archivos del pipeline y/o scripts de CI, CD

### Ejemplos

Ejemplo de BREAKING CHANGE:

fix!(dependencias): update de restCharp

Este cambio se realizo porque la version 107.2.1 quedo obsoleta en fecha AB/08/1990

fix(BREAKING CHANGE): Cambio de string de conexion

Este cambio se realizo para utilizar la nueva base de datos en azure

Ejemplo de contexto:

fix(SERINCPDRD01): Actualizar apache y OpenSSL

ci(desplieguetemprano): Cambio de rama origen para prueba

feat(266558): Agregando endpoint para validar si el token es valido



By [rafaelreynoso89](#)

Not published yet.

Last updated 11th October, 2022.

Page 1 of 1.

Sponsored by [Readable.com](#)

Measure your website readability!

<https://readable.com>