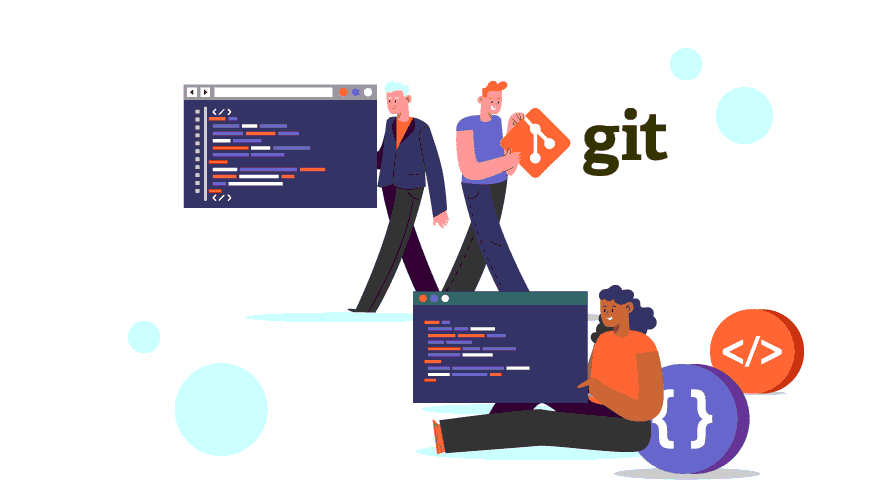
**[README.md](https://github.com/iuricode/padroes-de-commits" \l "readme)**

**📄  
Padrões de commits**

**[](https://github.com/iuricode/padroes-de-commits/blob/main/gitcommit.png)**

De acordo com a documentação do **Conventional Commits**, Commits Semânticos são uma convenção simples para ser utilizada nas mensagens de commit. Essa convenção define um conjunto de regras para criar um histórico de commit explícito, o que facilita a criação de ferramentas automatizadas.

Esses commits auxiliarão você e sua equipe a entenderem de forma facilitada quais alterações foram realizadas no trecho de código que foi commitado.

Essa identificação ocorre por meio de uma palavra e emoji que identifica se aquele commit realizado se trata de uma alteração de código, atualização de pacotes, documentação, alteração de visual, teste...

**🦄 Tipo e Descrição**

O commit semântico possui os elementos estruturais abaixo (tipos), que informam a intenção do seu commit ao utilizador(a) de seu código.

* feat- Commits do tipo feat indicam que seu trecho de código está incluindo um **novo recurso** (se relaciona com o MINOR do versionamento semântico).
* fix - Commits do tipo fix indicam que seu trecho de código commitado está **solucionando um problema** (bug fix), (se relaciona com o PATCH do versionamento semântico).
* docs - Commits do tipo docs indicam que houveram **mudanças na documentação**, como por exemplo no Readme do seu repositório. (Não inclui alterações em código).
* test - Commits do tipo test são utilizados quando são realizadas **alterações em testes**, seja criando, alterando ou excluindo testes unitários. (Não inclui alterações em código)
* build - Commits do tipo build são utilizados quando são realizadas modificações em **arquivos de build e dependências**.
* perf - Commits do tipo perf servem para identificar quaisquer alterações de código que estejam relacionadas a **performance**.
* style - Commits do tipo style indicam que houveram alterações referentes a **formatações de código**, semicolons, trailing spaces, lint... (Não inclui alterações em código).
* refactor - Commits do tipo refactor referem-se a mudanças devido a **refatorações que não alterem sua funcionalidade**, como por exemplo, uma alteração no formato como é processada determinada parte da tela, mas que manteve a mesma funcionalidade, ou melhorias de performance devido a um code review.
* chore - Commits do tipo chore indicam **atualizações de tarefas** de build, configurações de administrador, pacotes... como por exemplo adicionar um pacote no gitignore. (Não inclui alterações em código)
* ci - Commits do tipo ci indicam mudanças relacionadas a **integração contínua** (*continuous integration*).

**☑️ Recomendações**

* Adicione um título consistente com o título do conteúdo;
* Recomendamos que na primeira linha deve ter no máximo 4 palavras;
* Para descrever com detalhes, usar a descrição do commit;
* Usar um emoji no início da mensagem de commit representando sobre o commit;
* Um link precisa ser adicionado em sua forma mais autêntica, ou seja: sem encurtadores de link e links afiliados;

**💈 Padrões de emojis**

| **Tipo de commit** | **Emojis** | **Palavra-chave** |
| --- | --- | --- |
| Acessibilidade | ♿ :wheelchair: |  |
| Adicionando um teste | ✅ :white\_check\_mark: | test |
| Adicionando uma dependência | ➕ :heavy\_plus\_sign: | build |
| Alterações de revisão de código | 👌 :ok\_hand: | style |
| Animações e transições | 💫 :dizzy: |  |
| Bugfix | 🐛 :bug: | fix |
| Comentários | 💡 :bulb: | docs |
| Commit inicial | 🎉 :tada: | init |
| Configuração | 🔧 :wrench: | chore |
| Deploy | 🚀 :rocket: |  |
| Documentação | 📚 :books: | docs |
| Em progresso | 🚧 :construction: |  |
| Estilização de interface | 💄 :lipstick: | feat |
| Infraestrutura | 🧱 :bricks: | ci |
| Lista de ideias (tasks) | 🔜 :soon: |  |
| Mover/Renomear | 🚚 :truck: | chore |
| Novo recurso | ✨ :sparkles: | feat |
| Package.json em JS | 📦 :package: | build |
| Performance | ⚡ :zap: | perf |
| Refatoração | ♻️ :recycle: | refactor |
| Removendo um arquivo | 🔥 :fire: |  |
| Removendo uma dependência | ➖ :heavy\_minus\_sign: | build |
| Responsividade | 📱 :iphone: |  |
| Revertendo mudanças | 💥 :boom: | fix |
| Segurança | 🔒️ :lock: |  |
| SEO | 🔍️ :mag: |  |
| Tag de versão | 🔖 :bookmark: |  |
| Teste de aprovação | ✔️ :heavy\_check\_mark: | test |
| Testes | 🧪 :test\_tube: | test |
| Texto | 📝 :pencil: |  |
| Tipagem | 🏷️ :label: |  |
| Tratamento de erros | 🥅 :goal\_net: |  |

**💻 Exemplos**

| **Comando git** | **Resultado no GitHub** |
| --- | --- |
| git commit -m ":tada: Commit inicial" | 🎉 Commit inicial |
| git commit -m ":books: docs: Atualizaçao do README" | 📚 docs: Atualizaçao do README |
| git commit -m ":bug: fix: Loop infinito na linha 50" | 🐛 fix: Loop infinito na linha 50 |
| git commit -m ":sparkles: feat: Pagina de login" | ✨ feat: Pagina de login |
| git commit -m ":bricks: ci: Modificaçao no Dockerfile" | 🧱 ci: Modificaçao no Dockerfile |
| git commit -m ":recycle: refactor: Passando para arrow functions" | ♻️ refactor: Passando para arrow functions |
| git commit -m ":zap: perf: Melhoria no tempo de resposta" | ⚡ perf: Melhoria no tempo de resposta |
| git commit -m ":boom: fix: Revertendo mudanças ineficientes" | 💥 fix: Revertendo mudanças ineficientes |
| git commit -m ":lipstick: feat: Estilizaçao CSS do formulario" | 💄 feat: Estilizaçao CSS do formulario |
| git commit -m ":test\_tube: test: Criando novo teste" | 🧪 test: Criando novo teste |
| git commit -m ":bulb: docs: Comentários sobre a função LoremIpsum( )" | 💡 docs: Comentários sobre a função LoremIpsum( ) |