**Sobre Rebase**

No último vídeo vimos o recurso rebase.

À primeira vista ele parece não ter muita diferença com o merge, afinal de contas **os dois métodos são usados para integrar mudanças de uma branch para outra**. Afinal de contas, quando eu uso um ou outro e quais são realmente as diferenças?

* o *merge* **integra** o conteúdo da branch de trabalho (por exemplo titulo ou lista) com a branch master. Nesse caso **apenas a branch master é alterada para adicionar as mudanças e o histórico da branch de trabalho permanece inalterado** e um novo commit dessa junção é adicionado ao histórico.
* o *rebase* **move** a base da branch de trabalho para o final da branch master; ou seja, Nesse caso, o código também é integrado, porém porém ele faz isso transformando duas branches em uma só. Assim, a linha do tempo das alterações de código permanece sempre como uma linha contínua da branch master, apagando a branch de trabalho.

O merge preserva o histórico "por extenso" e todos os commits feitos nas branches, e é possível consultar exatamente em que momento mudanças no código feitas através de branches de trabalho foram adicionadas à branch principal. Quando as branches são unidas usando rebase esse histórico não é preservado.

Em projetos complexos, algumas pessoas podem achar essas ramificações um pouco confusas. Porém, por outro lado, uma única linha contínua faz com que não seja possível identificar exatamente os pontos de modificação no código.

Traduzindo estes dois processos para um diagrama, teríamos o seguinte resultado:

// usando merge

\* branch master

॰ commit "master1"

|

|

॰ commit "master2"

|\

| \

| \ branch lista

| ॰ commit "lista1"

| |

| |

| ॰ commit "lista2"

| |

| |

| ॰ commit "lista3"

| /

॰ checkout master

| merge lista

|

branch master contém alterações feitas em lista

COPIAR CÓDIGO

// usando rebase

\* branch master

॰ commit "master1"

|

|

॰ commit "master2"

|

|

॰ checkout master

| rebase lista

|

branch master contém alterações feitas em lista

commits em master são preservados

histórico da branch lista não é preservado

COPIAR CÓDIGO

**Quando utilizar um ou outro?**

**Como sempre dizemos em programação, depende. Em geral, partimos das seguintes situações:**

* **merge é mais indicado para trabalhos com branches compartilhadas, onde há várias pessoas envolvidas alterando diversas branches;**
* **rebase é mais indicado para projetos pessoais, times pequenos ou projetos mais simples.**

Em alguns casos específicos é possível usar o rebase mesmo em repositórios compartilhados, ou você pode preferir usar sempre merge mesmo trabalhando em um projeto pessoal privado. Ou seja, tudo depende do caso :) outros pontos a considerar:

* rebase pode ser útil para "limpar" linhas do tempo de repositórios que se tornaram muito poluídas pelo excesso de *merging* e *branching*;
* Se você quer preservar o histórico completo do projeto, independente da complexidade, use sempre merge;
* rebase pode facilitar a resolução de conflitos, porém pode também ser mais complicado voltar o código ao estágio anterior em caso de conflitos mais complexos.