# Git "the stupid content tracker"

## História

Git foi criado em 2005 por Linus Torvalds, o criador do Linux. A motivação de criar o Git foi porquê o CVS anterior (SVN) era muito lento para comportar o trabalho do kernel.

Alguns significados para o termo: - "global information tracker" - "goddamn idiotic truckload of sh\*t"

## Pra que serve

## Git serve para recuperar informação

git log  
rm -rf docs  
browser http://localhost:8000  
git reset --hard HEAD  
browser http://localhost:8000

## Git serve para versionar informações

##### Criando

echo "Historia do git" >docs/historia.md  
echo "Historia" >>docs/index.md  
git add .  
git commit -m "adicionado arquivos sobre historia"  
echo "Git foi criado em 2005 por Linus Torvalds, o criador do Linux" >>docs/historia.md  
git commit -a -m "detalhes sobre a historia"

##### Desfazendo

git checkout HEAD~1 docs/historia.md  
git checkout HEAD docs/historia.md

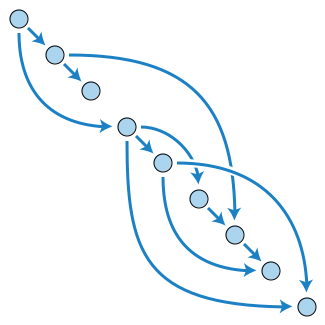
Git também serve para:

* Trabalhar em múltiplas tarefas concorrentemente
* Trabalhar com múltiplas equipes concorrentemente
* Pode-se versionar todo tipo de informação: projetos de software, livros, tcc's, etc.

## Conceitos

## Histórico em grafo

Os commits no git são estruturados em forma de grafo



Grafo acíclico

## Áreas de armazemanto do git

Git tem 3 áreas de armazenamento principais

* Staging (index)
* Repositório Local
* Repositório Remoto



Comandos de trasporte e áreas de armazenamento

# Comandos

## Lista de Comandos

Git conta com vários comandos, a lista completa se encontra em /lib/git-core.

Para mais detalhes do que como cada um opera pode utilizar

man git commando  
git commando --help

## Comandos conhecidos

* git init
* git add
* git commit
* git format-patch
* git apply
* git show
* git log
* git rebase
* git revert
* git checkout
* git reset
* git stash

## Inicializando um projeto

## Init

Para inicializar um projeto git

São criados arquivos na pasta .git com os dados versionados.

git init $DIRETORIO

Para uma lista dos arquivos criados pelo git:

cd /tmp  
git init foo  
find .

## Config

git config --global user.name "John Doe"  
git config --global user.email johndoe@example.com

## Criando Histórico

## Add

Git add adiciona arquivos a uma área temporária para compor um commit (staging).

git add $ARQUIVO|$DIRETORIO

## Commit

Cada mudança no histórico de um projeto é representado por um commit. git show mostra o último commit.

Para uma lista completa dos commits use: git log.

git commit

O commit transfere as informações para o repositório local.

Boas práticas de nomenclatura de commits https://www.alexkras.com/19-git-tips-for-everyday-use/#good-commit-message

## Patches

## Format patch

Patches são mudanças que podem ser transferidas de um repositório para outro:

Para criar um patch do último commit:

git format-patch master

## Appy

Para aplicar um patch:

curl -L https://goo.gl/p1LEc7 -o 0001-historia.patch  
git apply 0001-historia.patch

## Diff e show

O resultado de git show e git diff também são comapatíveis com patches do git.

git diff > my\_patch.patch  
git show > my\_patch.patch  
git diff --cached > my\_patch.patch

## Branches

Uma branch é uma linha de trabalho independente. Podem ser usadas para diversos propósitos.

Pode-se ter uma branch para:

* experimentar uma tecnologia nova;
* uma branch para um bug-fix;
* outra para o trabalho do sprint;

A branch padrão no git é a master. Para listar todas as branches use: git branch

git commit "informações sobre branches no master"  
git checkout -b "recursos\_adicionais" #cria outra linha de trabalho  
curl -L https://goo.gl/XlMc3m -o 0001-recursos\_adicionais.patch  
git apply 0001-recursos\_adicionais.patch

O comando checkout serve para criar a branch e entrar nela.

##### Merge

Merge mescla o conteúdo de branches

git commit -m "mais modificações"  
git merge recursos\_adicionais  
git log

##### Fluxos de trabalho

Existem vários fluxos de trabalho de branches.

* Githubflow
* git flow
* branch por tópico

Mais sobre fluxos de trabalho

* https://git-scm.com/book/en/v2/Distributed-Git-Distributed-Workflows#\_distributed\_git
* https://git-scm.com/book/en/v2/Git-Branching-Branching-Workflows

## Merges

Fast-forward: move o ponteiro da master para o último commit da branch.

git merge --fast-forward

## Resolvendo com nossas alterações

git pull origin master -X ours

## Desfazendo

##### O que ainda não está comitado

git reset --hard HEAD

##### Remover arquivos criados mas não versionados

git ls-files --others --exclude-standard | xargs rm

##### Remover do staging

git reset HEAD $FILE

##### Trocar a última mensagem de commit

git commit --amend

##### Desfazendo últimos 3 commits - sem desfazer o código

git reset --soft HEAD~3

##### Desfazendo últimos 3 commits - desfazendo o código

git reset --hard HEAD~3

## Reescrevendo Histórico

É possível adicionar algo a um commit nos seguintes passos:

* Rebase no commit e marcar ele como editado
* Comitar as alterações
* Fazer um fix up no commit com seu pai

A regra de ouro do rebase https://www.atlassian.com/git/tutorials/merging-vs-rebasing?section=the-golden-rule-of-rebasing

## Revert

Desfaz um commit criando um novo commit. Recomendado em branches públicas onde o histórico não pode ser reescrito.

git checkout hotfix  
git revert HEAD~2

## Remotos

Git é distriuído.

## Remoto no filesystem local

cd /path/to/git-docs  
cd ..  
git clone git-docs other-git-docs  
cd other-git-docs  
git config user.name "other user"  
git config user.email other.user@gmail.com  
  
  
//do some work  
cd ../git-docs  
git remote add other ../other-git-docs  
git merge other/master

## Remoto online

git clone https://github.com/JeanCarloMachado/git-docs  
git push origin new\_branch

## Listar os repositórios remotos

git remote -v

## Github

GitHub é uma rede social.

## Permite

* Criar repositórios
* Colaborar para repositórios existentes:
  + Criando issues
  + Submitando pull requests

## Outros Serviços

##### Githubio Pages

Hospedar um site com seu usuário.

##### Gists

Tipo pastebin

## Ferramentas Auxiliares

## Stash

Para salvar trabalhos temporários

git stash  
git stash apply

## Gitk

Uma interface nativa para quem gosta de interfaces gráficas

## Aliases

git config --global alias.co checkout

## Bash aliases

alias g=git

## [LFS](https://git-lfs.github.com/)

Extensão para trabalhar com arquivos grandes

## Gitignore

.gitignore é uma lista de arquivos bloqueados do versionamento

## Hooks

Com hooks é possível rodar comandos quando eventos ocorrem no git.

* [Referência oficial](https://git-scm.com/docs)
* [Melhor tutorial de Git](http://rypress.com/tutorials/git/index)
* [Encontrando issues no gitub](http://www.jeancarlomachado.com.br/blog/findingissuesongithub.html)