

Controle de Versão



eliasfpaiva

Controle de Versão

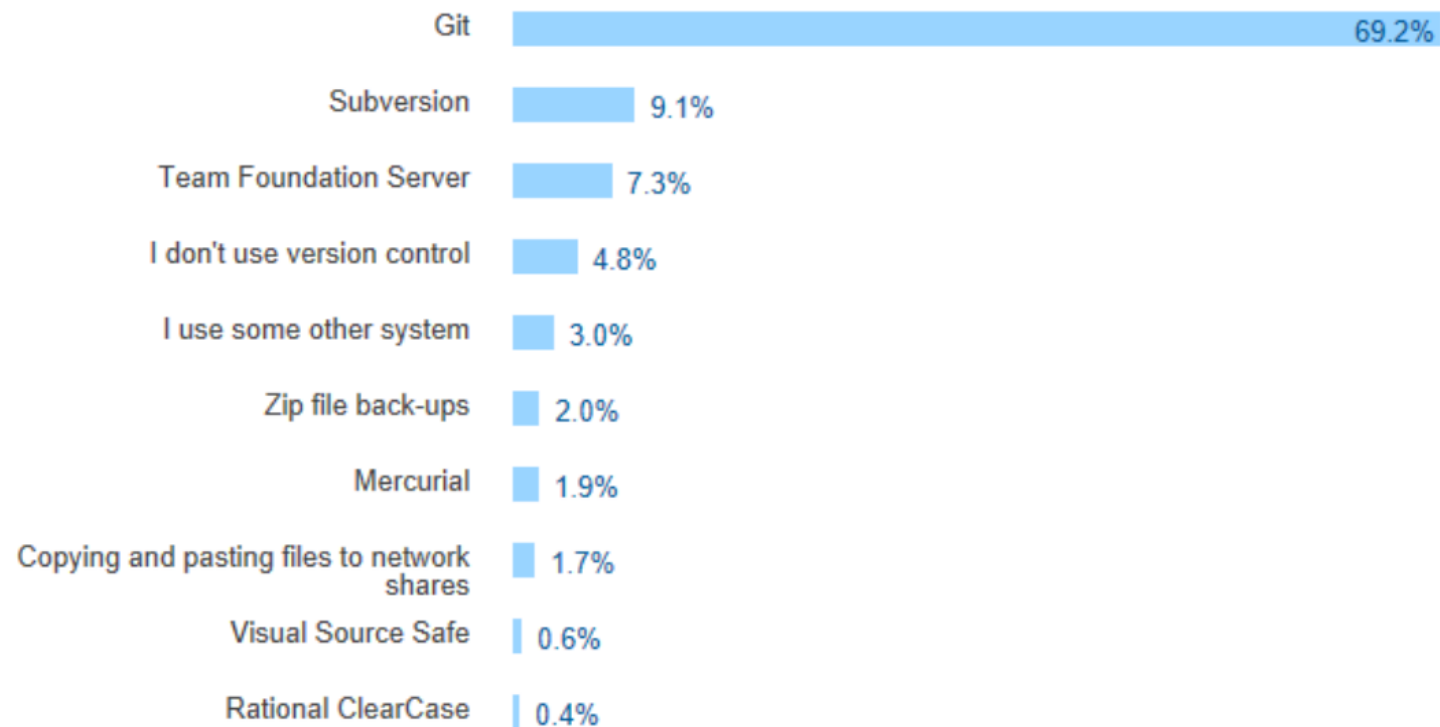


eliasfpaiva

Versionamento

- Controle de versão de arquivo
- Sistemas de Controle de Versão Locais
- Sistemas de Controle de Versão Centralizados
- Sistemas de Controle de Versão Distribuídos
 - Git
 - Subversion
 - Team Foundation Server
 - Mercurial

Distribuição do uso de sistemas de versionamento



30,730 responses; select all that apply

No surprises here: Git is the overwhelmingly clear choice of version control.

Origem do Git

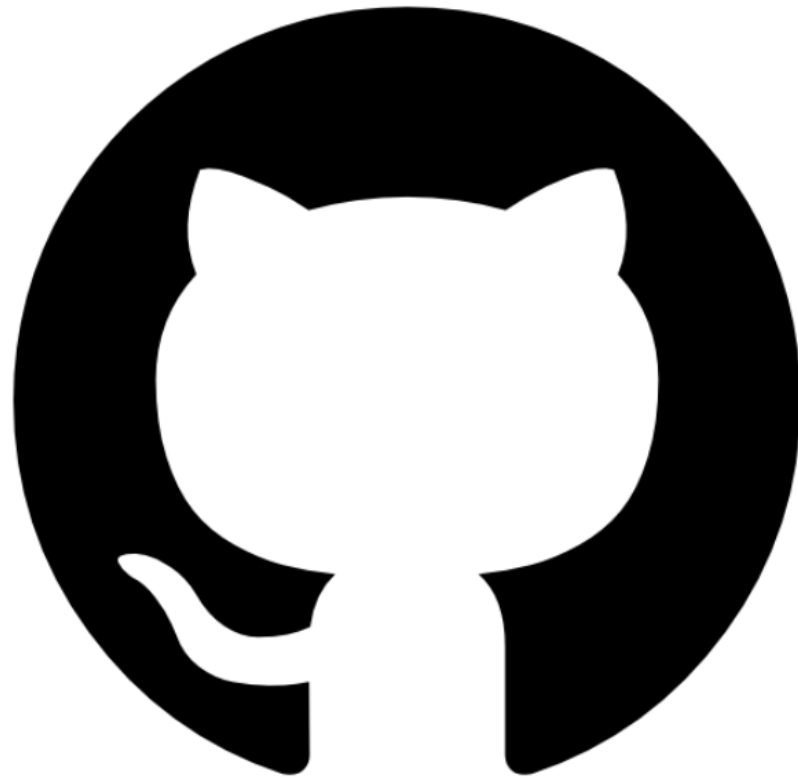
- 1991 - 2002 - Arquivos compactados
- 2002 - 2005 - BitKeeper
- 2005 - Criação do Git



Sites de Controle de Versão



GitLab



GitHub

Primeiro COMMIT em outubro de 2007

Uso do GitHub



Comandos Git

Git Cheat Sheet

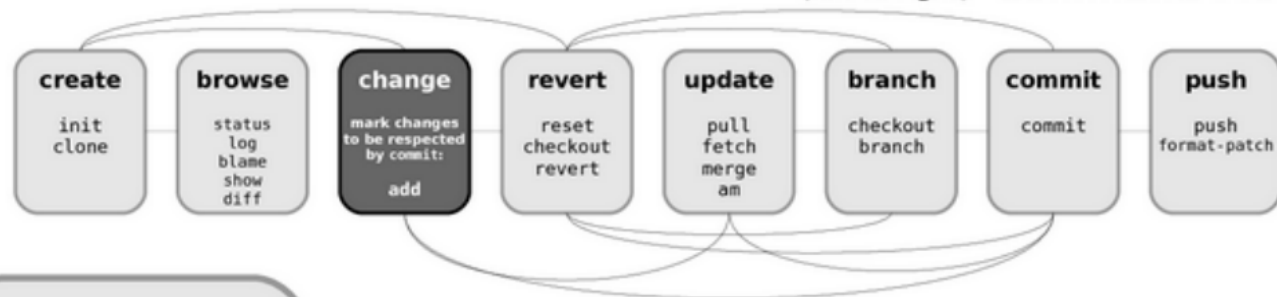
by Jan Krüger <jk@jk.gs>, <http://jan-krueger.net/git/>
Based on work by Zack Rusin

Basics

Use git help [command] if you're stuck.

master	default devel branch
origin	default upstream branch
HEAD	current branch
HEAD^	parent of HEAD
HEAD~4	great-great grandparent of HEAD
foo..bar	from branch foo to branch bar

(left to right) Command Flow



Create

From existing files

```
git init
git add .
```

From existing repository

```
git clone ~/old ~/new
git clone git://...
git clone ssh://...
```

Publish

In Git, commit only respects changes that have been marked explicitly with add.

```
git commit [-a]
(-a: add changed files automatically)
git format-patch origin
(create set of diffs)
git push remote
(push to origin or remote)
git tag foo
(mark current version)
```

Useful Tools

```
git archive
Create release tarball
git bisect
Binary search for defects
git cherry-pick
Take single commit from elsewhere
git fsck
Check tree
git gc
Compress metadata (performance)
git rebase
Forward-port local changes to remote branch
git remote add URL
Register a new remote repository for this tree
git stash
Temporarily set aside changes
git tag
(there's more to it)
gitk
Tk GUI for Git
```

Tracking Files

```
git add files
git mv old new
git rm files
git rm --cached files
(stop tracking but keep files in working dir)
```

View

```
git status
git diff [oldid newid]
git log [-p] [file|dir]
git blame file
git show id (meta data + diff)
git show id:file
git branch (shows list, * = current)
git tag -l (shows list)
```

Update

```
git fetch (from def. upstream)
git fetch remote
git pull (= fetch & merge)
git am -3 patch.mbox
git apply patch.diff
```

Revert

In Git, revert usually describes a new commit that undoes previous commits.

```
git reset --hard (NO UNDO)
(reset to last commit)
git revert branch
git commit -a --amend
(replaces prev. commit)
git checkout id file
```

Branch

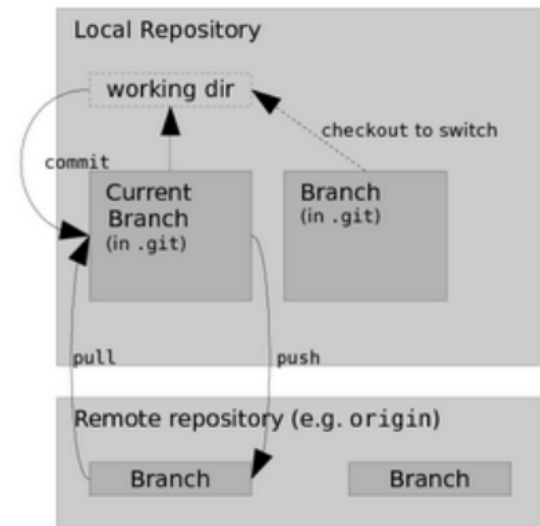
```
git checkout branch
(switch working dir to branch)
git merge branch
(merge into current)
git branch branch
(branch current)
git checkout -b new other
(branch new from other and switch to it)
```

Conflicts



Use add to mark files as resolved.

```
git diff [--base]
git diff --ours
git diff --theirs
git log --merge
gitk --merge
```

Structure Overview





Atividade Prática

- Criação de uma conta no GitHub
- Instalação do Git
- Criação de um novo repositório no GitHub
- Clonagem do repositório
- Acesso à pasta do repositório
- Criação de uma nova Branch
- Edição do projeto 
- Adicionar as alterações
- Criação de um commit
- Envio das alterações para o GitHub
- Merge para a master
- Criação de uma segunda Branch
- Alteração do projeto 
- Enviar para o GitHub
- Merge para a master tratando conflitos

Atividade - Complete

_ata na _ata, que corre
pra _ata, fugintdo da _ata,
que pega a _ata, que come
a _ata, que está na _ata,
que está sem _ata.



Atividade Prática

- Criação de uma conta no GitHub
- Instalação do Git
- Criação de um novo repositório no GitHub
- Clonagem do repositório
- Acesso à pasta do repositório
- Criação de uma nova Branch
- Edição do projeto 
- Adicionar as alterações
- Criação de um commit
- Envio das alterações para o GitHub
- Merge para a master
- Criação de uma segunda Branch
- Alteração do projeto 
- Enviar para o GitHub
- Merge para a master tratando conflitos

Atividade - Resolvida

Bata na **P**ata, que corre
pra **M**ata, fugintdo da **G**ata,
que pega a **R**ata, que come
a **N**ata, que está na **L**ata,
que está sem **D**ata.

Atividade Prática

- Criação de uma conta no GitHub
- Instalação do Git
- Criação de um novo repositório no GitHub
- Clonagem do repositório
- Acesso à pasta do repositório
- Criação de uma nova Branch
- Edição do projeto 
- Adicionar as alterações
- Criação de um commit
- Envio das alterações para o GitHub
- Merge para a master
- Criação de uma segunda Branch
- Alteração do projeto 
- Enviar para o GitHub
- Merge para a master tratando conflitos

Referências

- Primeiros passos - Sobre Controle de Versão; disponível em <https://git-scm.com/book/pt-br/v1/Primeiros-passos-Sobre-Controle-de-Vers%C3%A3o>, acesso em 09/12/2017.
- Developer Survey (Questionário do desenvolvedor) 2017; disponível em <https://insights.stackoverflow.com/survey/2017>, acesso em 09/12/2017
- Primeiros passos - Uma Breve História do Git; disponível em <https://git-scm.com/book/pt-br/v1/Primeiros-passos-Uma-Breve-Hist%C3%B3ria-do-Git>, acesso em 09/12/2017.
- <https://github.com/>, acesso em 09/12/2017
- <https://github.com/about>, acesso em 09/12/2017
- 5 Cheat Sheets que todo Geek deve tener, disponível em <https://geekytheory.com/5-cheat-sheets-que-todo-geek-debe-tener>, acesso em 09/12/2017

Controle de Versão



eliasfpaiva