Empoderar é o primeiro passo para novas revoluções

womakerscode.org

Quem sou?

- @romeumattos
- 25 anos
- Porto Alegre - RS
- Ex estudante de Computação
- Backend Developer @MeSalva!
- Desenvolvedor web há 5 anos

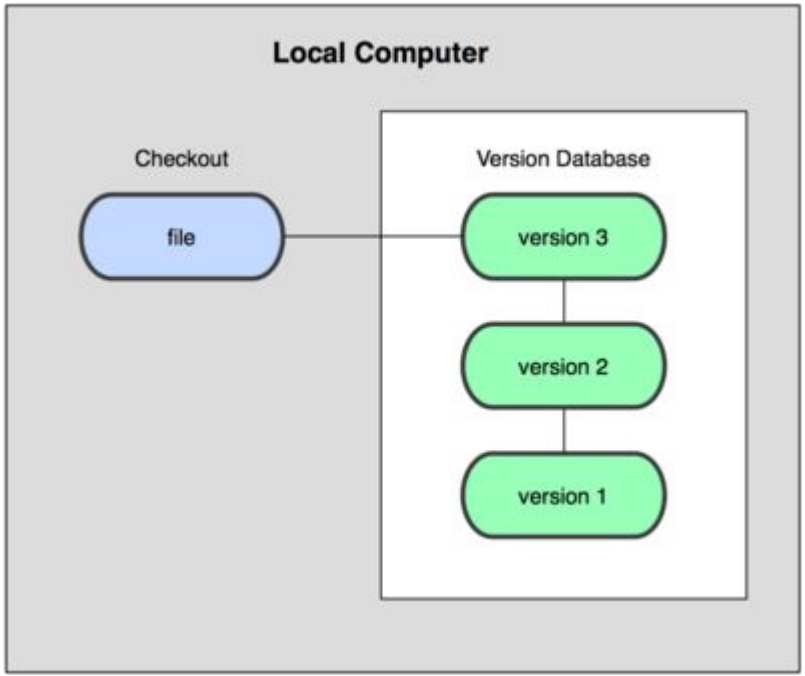
womakerscode.org

Introdução

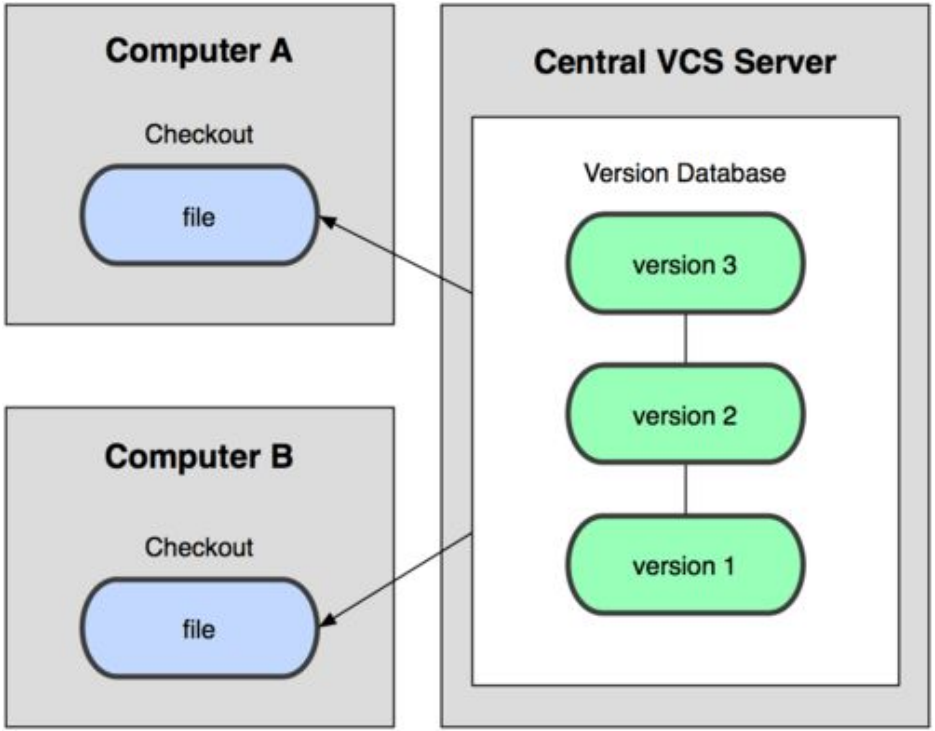
Versionamento

“O controle de versão é um sistema que registra as mudanças feitas em um arquivo ou um conjunto de arquivos ao longo do tempo de forma que você possa recuperar versões específicas.”

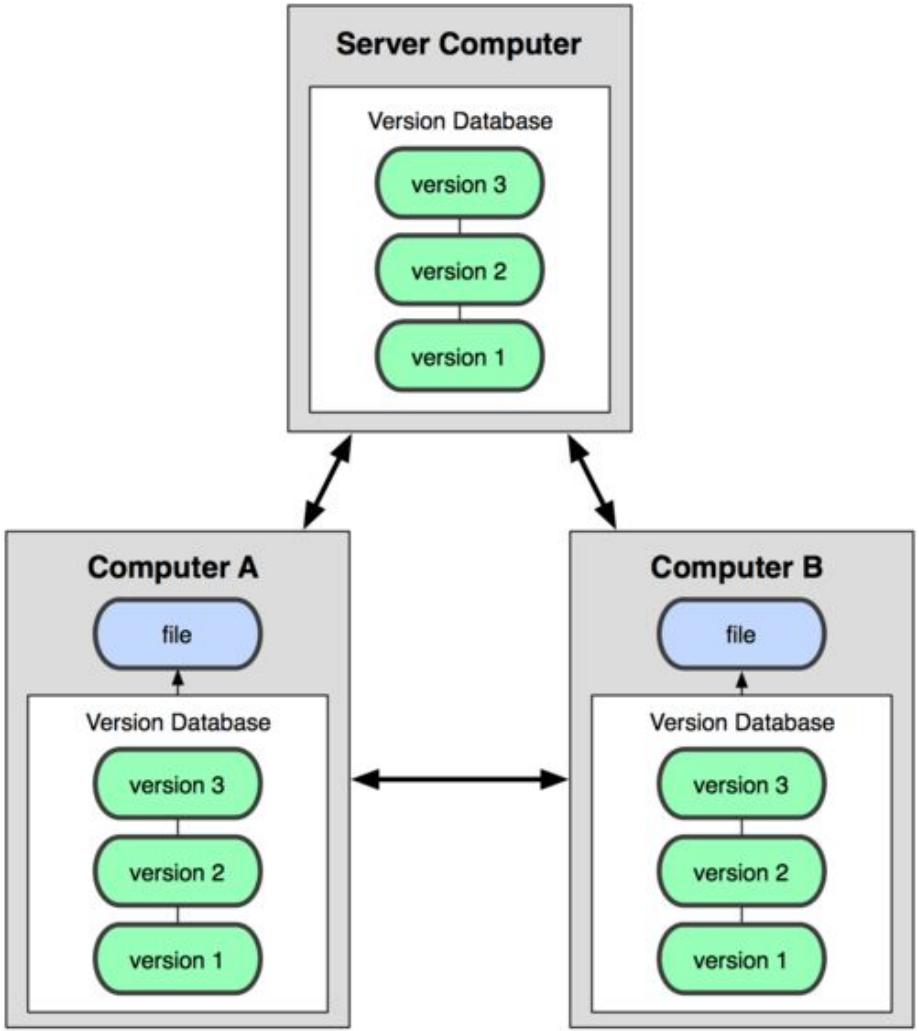
Sistema de Controle de Versão Locais



Sistema de Controle de Versão Centralizados (CVCS)



Versionamento



Sistema de Controle de Versão Distibuidos (DVCS)

(Git, Mercurial, Bazaar ou Darcs)

Sistema de Controle de Versão Distribuídos

- ➔ Não apenas fazem cópias das últimas versões dos arquivos
- ➔ Cópias completas do repositório
- ➔ Se um servidor falha, qualquer um dos repositórios dos clientes pode ser copiado de volta para o servidor
- ➔ Vários repositórios remotos
- ➔ Trabalho em conjunto com diferentes grupos de pessoas
- ➔ Diferentes tipos de workflow - não são possíveis em sistemas centralizados


git

História do Git

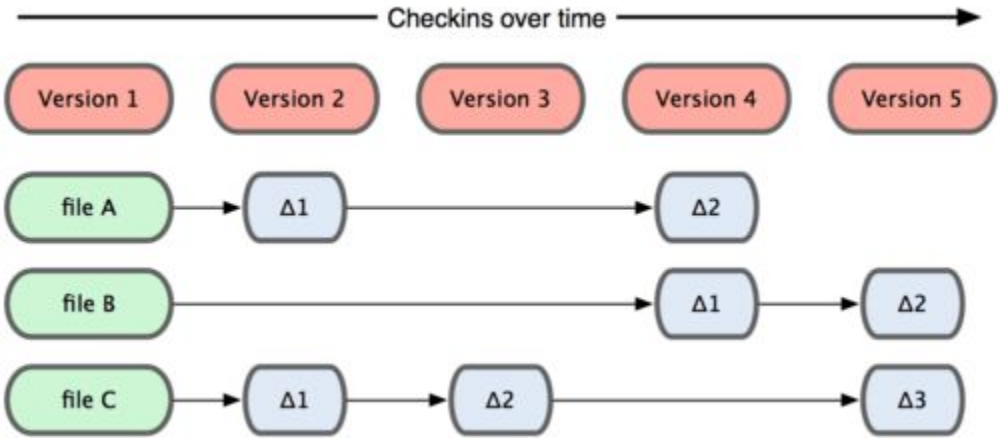
“Assim como muitas coisas boas na vida, o Git começou com um tanto de destruição criativa e controvérsia acirrada. O kernel (núcleo) do Linux é um projeto de software de código aberto de escopo razoavelmente grande. Durante a maior parte do período de manutenção do kernel do Linux (1991-2002), as mudanças no software eram repassadas como patches e arquivos compactados. Em 2002, o projeto do kernel do Linux começou a usar um sistema DVCS proprietário chamado BitKeeper.

Em 2005, o relacionamento entre a comunidade que desenvolvia o kernel e a empresa que desenvolvia comercialmente o BitKeeper se desfez, e o status de isento-de-pagamento da ferramenta foi revogado. Isso levou a comunidade de desenvolvedores do Linux (em particular Linus Torvalds, o criador do Linux) a desenvolver sua própria ferramenta baseada nas lições que eles aprenderam ao usar o BitKeeper.“

Objetivos dos desenvolvedores

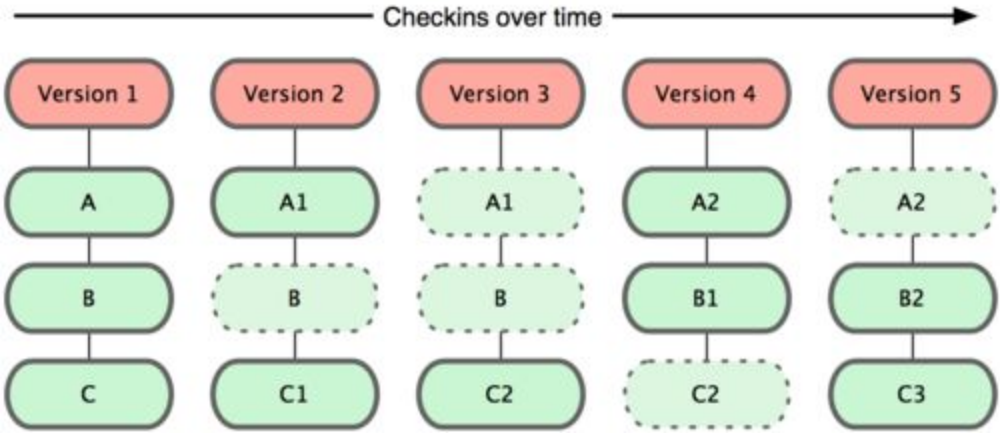
- ➔ Velocidade
- ➔ Design simples
- ➔ Suporte robusto a desenvolvimento **não linear** (milhares de branches paralelos)
- ➔ Totalmente distribuído
- ➔ Capaz de lidar eficientemente com grandes projetos
- ➔ Ex: kernel do Linux (velocidade e volume de dados)

VCS Similares

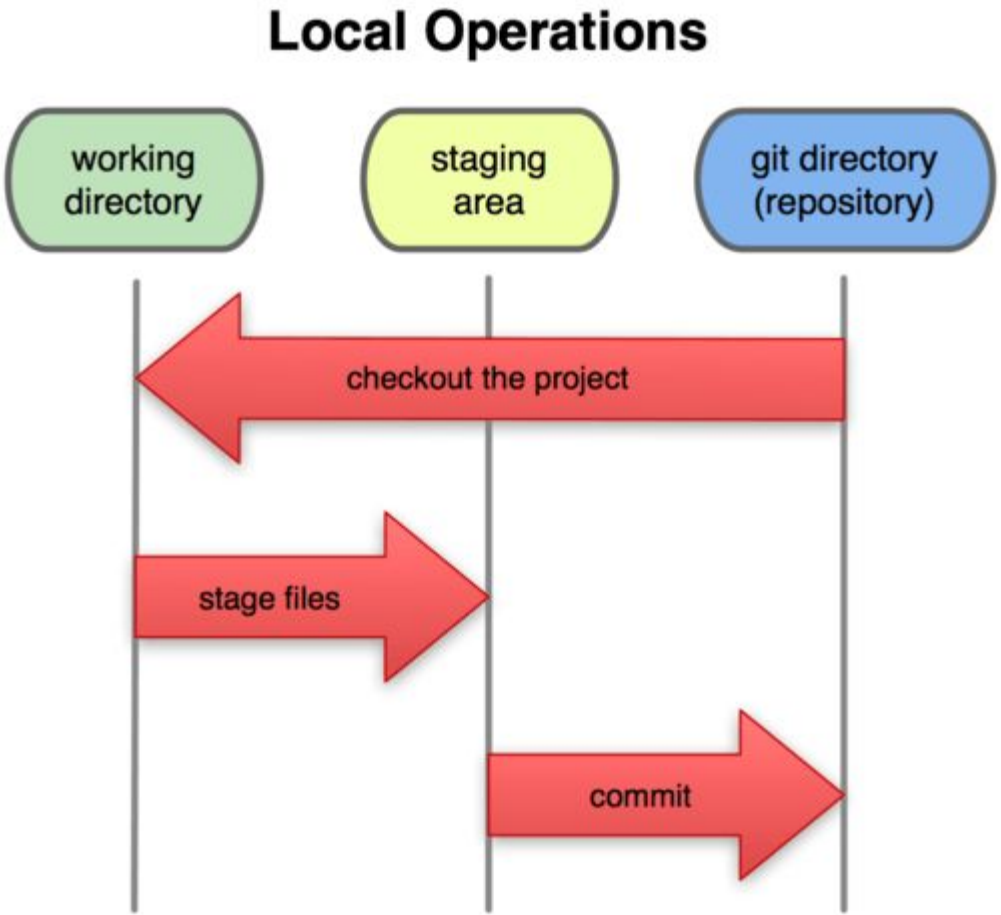


Versionamento

Git



Versionamento



Os 3 estados

Committed: seguramente armazenados em sua base de dados local.

Modified: sofreu mudanças mas que ainda não foi consolidado na base de dados.

Staged: quando você marca um arquivo modificado em sua versão corrente para que ele faça parte do snapshot do próximo commit.

Vamos Praticar =)

Criando um novo repositório

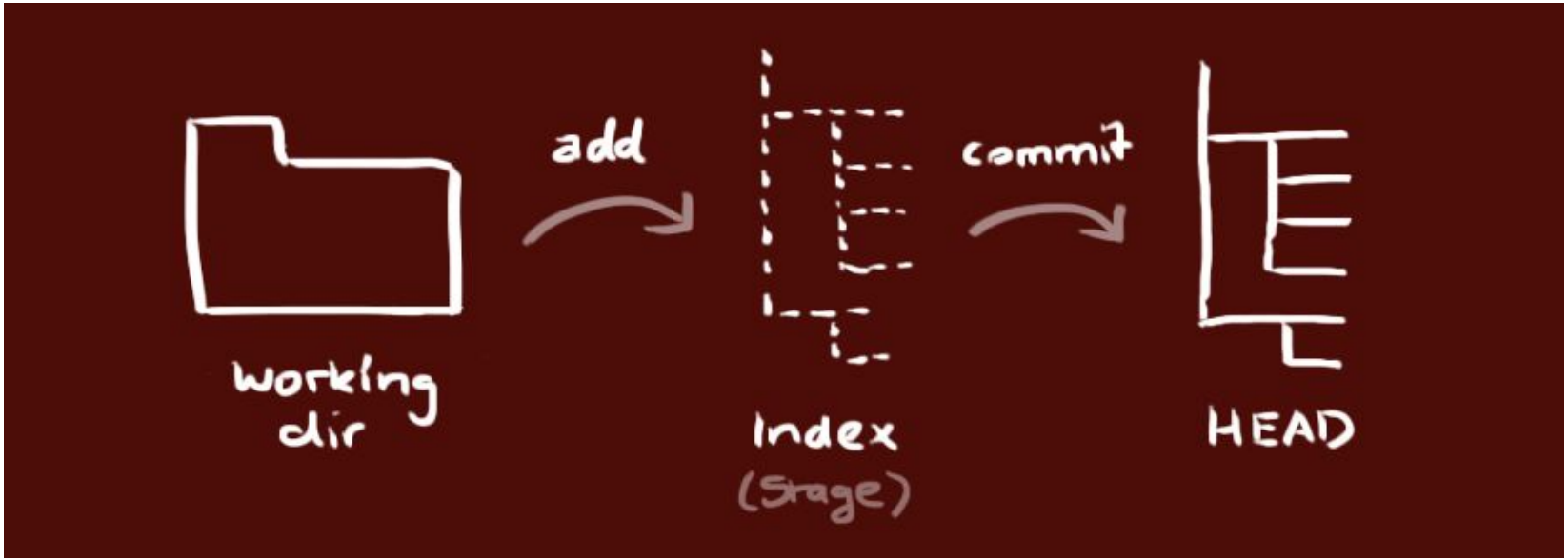
- Instalando: sudo apt-get install git
- crie uma nova pasta
- abra-a e execute o comando:
- git init

Obtendo um repositório

para repositório local:
git clone /caminho/para/o/repositório

para repositório remoto:
git clone https://github.com/romeumattos/womakers.git

Fluxo de Trabalho



Adicionando um repositório remoto

Se você não clonou um repositório remoto e quer conectar seu repositório a um servidor remoto:

```
git remote add origin https://github.com/romeumattos/womakers.git
```


Adicionando

Adicionar um arquivo modificado:

```
git add <arquivo>
```

Adicionando todos:

```
git add *
```


Commitando

Verificando arquivos modificados e staging:

```
git status
```

Fazendo um commit:

```
git commit -m "comentários das alterações"
```

OBS: Após o commit o arquivo vai para o HEAD, porém ainda não para o repositório remoto.

Enviando as alterações

Enviando suas alterações que estão no HEAD para o repositório remoto:

```
git push origin master
```


Branches

“Branches ("ramos") são utilizados para desenvolver funcionalidades isoladas umas das outras. O branch master é o branch "padrão" quando você cria um repositório. Use outros branches para desenvolver e mescle-os (merge) ao branch master após a conclusão. “

Branches

crie um novo branch chamado "funcionalidade_x" e selecione-o usando

`git checkout -b funcionalidade_x`

retorne para o master usando

`git checkout master`

Branches

um branch não está disponível a outros a menos que você envie o branch para seu repositório remoto:

```
git push origin <funcionalidade_x>
```

remova o branch da seguinte forma:

```
git branch -d funcionalidade_x
```


Atualizando - Pull

para atualizar seu repositório local com a mais nova versão, execute:

```
git pull
```


Mesclando - Merge

para fazer merge de um outro branch ao seu branch ativo (ex. master), use:

```
git merge <branch>
```

em ambos os casos o git tenta fazer o merge das alterações automaticamente.

Visualizando as diferenças

você pode também pré-visualizár as diferenças usando:

```
git diff <branch origem> <branch destino>
```


GitHub

GitHub é uma rede com mais de 12 milhões de pessoas e mais de 31 milhões de projetos para colaboração.

Explorar Projetos: https://github.com/explore

Flow de contribuição: https://guides.github.com/introduction/flow/

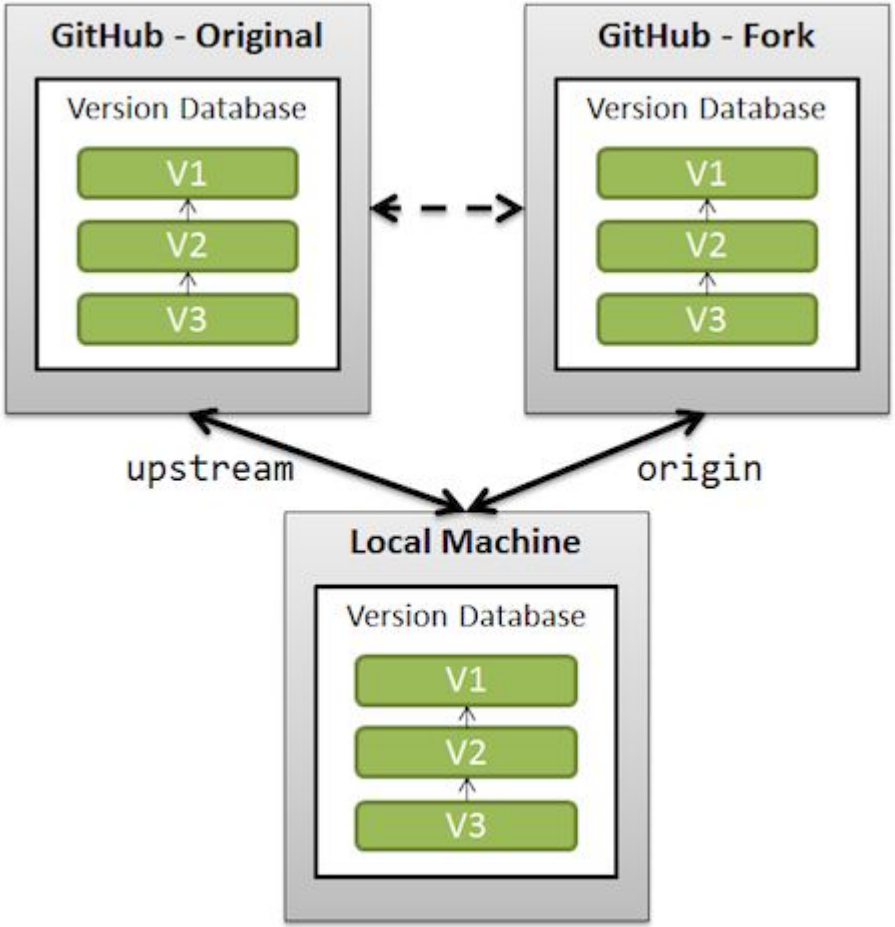
Fork

“Fork nada mais é que uma cópia de um repositório. Esta cópia vira um clone do estado atual do repositório, fazendo assim com que você possa experimentar mudanças e feature novas sem precisar utilizar o repositório principal.”

Pull Request

“[...] é que uma interface web amigável para você como desenvolvedor enviar/propor alterações em um repositório antes de integrar este código ao projeto oficial. Fazendo com que esta funcionalidade seja discutida previamente, e quando aprovada pela equipe que mantém o projeto, pode ser integrada ao código principal.”

GitHub - Contribuindo



Estrutura dos repositórios

Partiu contribuir?

Remotes

- ➔ **git clone usuario@servidor:/caminho/para/o/repositório**
- ➔ **cd /caminho/para/o/repositório**
- ➔ **git remote origin**
- ➔ **git remote add upstream git@servidor:/caminho/para/o/repositório**
- ➔ **git remote origin upstream**

GitHub - Contribuindo

Atualizando o repositório local

Atualizando durante o trabalho:

git rebase -i upstream/master

Atualizando com base nos remotes:

git pull upstream master

Atualizando seu fork:

git push origin master

Por que contribuir?

Referências

Git: https://git-scm.com/book/pt-br/v1/

Git: http://rogerdudler.github.io/git-guide/index.pt_BR.html

Github flow: https://medium.com/@pragmaticivan/forking-workflow-contribuindo-em-projetos-open-source-utilizando-git-parte-2-a1849204d02e#.y5o2cmqtz

Plus: Github Pages

https://pages.github.com/

Muito Obrigado

Site: romeumattos.com.br

E-mail: romeu.smattos@gmail.com

Twitter: @romeumattos

GitHub: @romeumattos