

# um apanhado de dicas de como usar Git (bem, ou um pouco melhor)

Sara Mortara, Andrea Sanchez-Tapia & Diogo Rocha

15 de Julho de 2019

sistema de controle de versão **livre** e de **código aberto**

sistema de controle de versão **livre** e de **código aberto**

- pegada

sistema de controle de versão **livre** e de **código aberto**

- pegada
- alta performance

# o que é git

sistema de controle de versão **livre** e de **código aberto**

- pegada
- alta performance
- múltiplos fluxos de trabalho

sistema de controle de versão **livre** e de **código aberto**

- pegada
- alta performance
- múltiplos fluxos de trabalho
- *branching merging*

sistema de controle de versão **livre** e de **código aberto**

- pegada
- alta performance
- múltiplos fluxos de trabalho
- *branching merging*
- distribuição - *clone*

sistema de controle de versão **livre** e de **código aberto**

- pegada
- alta performance
- múltiplos fluxos de trabalho
- *branching merging*
- distribuição - *clone*
- área intermediária - *staging area*



# o que é GitHub

serviço de hospedagem na internet para controle de versão e colaboração usando **git**

## por onde começar

- guia básico de **Git** no **GitHub**:  
<https://guides.github.com/activities/hello-world/>
- como usar o **Git** no **RStudio**:  
<https://pagepiccinini.com/r-course/lesson-0-introduction-and-set-up/>



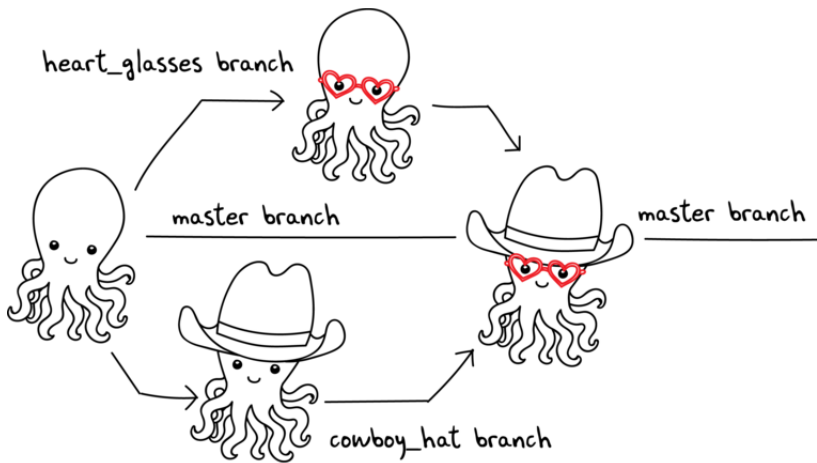
## fluxo de trabalho *feature*



## em outras palavras

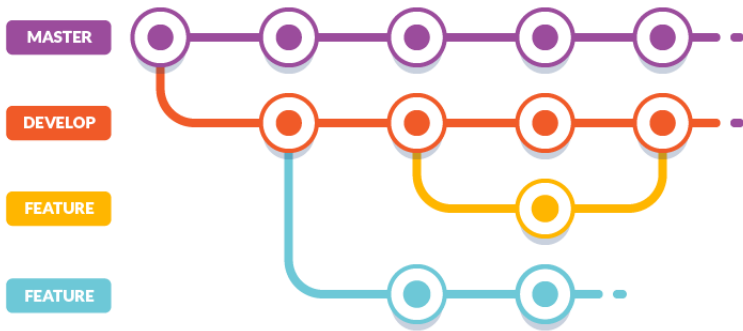
merge it back into the original project.

Here's a simplified version of that workflow:



Each feature is added back into master individually. So if the glasses are finished before the cowboy hat, no problem: those can be

## fluxo de trabalho *gitflow*



## comandos básicos

Para clonar um repositório já existente:

```
git clone https://github.com/Model-R/modelr_pkg.git
```

Para criar um repositório git localmente:

```
git init
```

## os meus **cinco** comandos básicos

1. Para checar em que pé está:

```
git pull status
```

2. Para atualizar o repo localmente:

```
git pull origin master
```

3. Para adicionar um arquivo com mudanças:

```
git add filename
```

4. O *commit*:

```
git commit -m "uma mensagem informativa para você e  
coleguinhxs"
```

5. Enviando para o repositório:

```
git push origin master
```



# boas práticas

## boas práticas

- 1 descrever no ***commit*** (o porquê da) mudança

## boas práticas

- 1 descrever no ***commit*** (o porquê da) mudança
- 2 trabalhar uma tarefa em cada ***branch***

## boas práticas

- 1 descrever no ***commit*** (o porquê da) mudança
- 2 trabalhar uma tarefa em cada ***branch***
- 3 `.gitignore`

## boas práticas

- 1 descrever no ***commit*** (o porquê da) mudança
- 2 trabalhar uma tarefa em cada ***branch***
- 3 `.gitignore`
- 4 usar modo ***diff*** para acompanhar mudanças no código

## boas práticas

- 1 descrever no ***commit*** (o porquê da) mudança
- 2 trabalhar uma tarefa em cada ***branch***
- 3 `.gitignore`
- 4 usar modo ***diff*** para acompanhar mudanças no código
- 5 reportar erros no código em ***issues***