UniCesumar

PROGRAMAÇÃO FRONT END

GIT

Prof. Esp. Dacio F. Machado


LOADING...

Name

- Super Cool Report v1.xlsx
- Super Cool Report v2.xlsx
- Super Cool Report v3.1.xlsx
- Super Cool Report v3.xlsx
- Super Cool Report v4.xlsx
- Super Cool Report v4a.xlsx
- Super Cool Report v4b.xlsx
- Super Cool Report v5.xlsx
- Super Cool Report vFinal.xlsx
- Super Cool Report vFinal_1.xlsx
- Super Cool Report vFinal_2.xlsx
- Super Cool Report vFinal_Final.xlsx
- Super Cool Report vFinal_Final-UPDATED.xlsx
- Super Cool Report vFinal_Final-UPDATED_NEW.xlsx

How it started

final.doc

How it's going

final final final final v2 submitted final.doc

VERSIONAMENTO - Vantagens

Você provavelmente já usou algum sistema de versionamento de arquivos mesmo antes de estudar sobre isso.

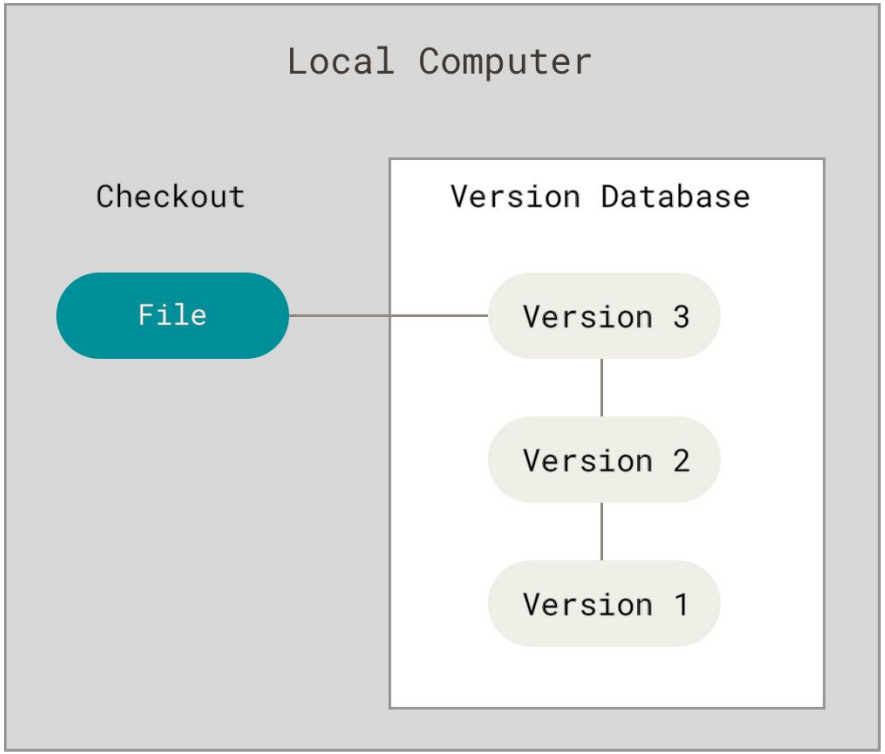
Temos a falta de controle sobre alterações em arquivos em um projeto, especialmente em projetos colaborativos que envolvem várias pessoas trabalhando no mesmo conjunto de arquivos.

No versionamento de arquivos, todas as alterações feitas em um arquivo são registradas e mantidas em um repositório centralizado, permitindo acompanhar e entender facilmente as mudanças realizadas.

VERSIONAMENTO

O versionamento de arquivos é o processo de rastrear e gerenciar diferentes versões de um arquivo ao longo do tempo.

Acompanhar as alterações feitas em um arquivo, permitindo que os usuários mantenham um histórico das mudanças e revertam para versões anteriores, se necessário.



VERSIONAMENTO - Vantagens

Colaboração: Permite que várias pessoas trabalhem em um mesmo arquivo sem sobrescrever as alterações umas das outras. Cada alteração é registrada como uma nova versão.

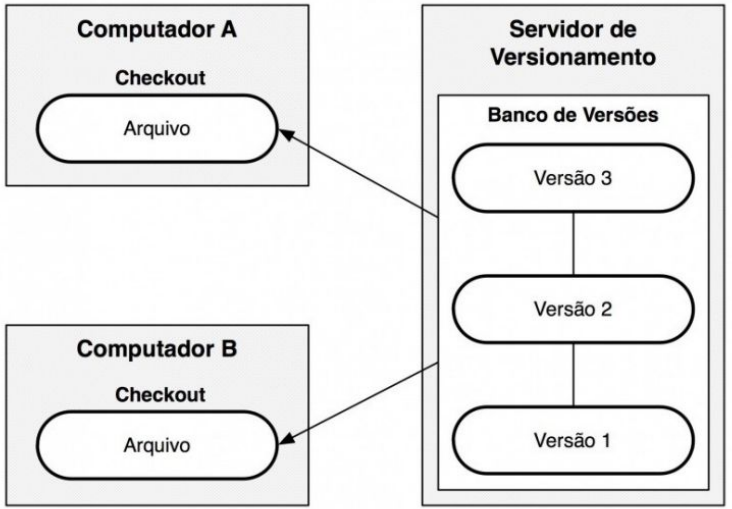
Histórico: Mantém um histórico das mudanças feitas em um arquivo, o que pode ser útil para rastrear quando e por quem uma alteração específica foi feita.

Recuperação: Permite a recuperação de versões anteriores de um arquivo em caso de erros, perda de dados ou necessidade de voltar a uma versão anterior.

Controle de Versão

Controle de versão é a prática de gerenciar alterações em arquivos de código-fonte ao longo do tempo, permitindo que você volte a versões anteriores, se necessário.

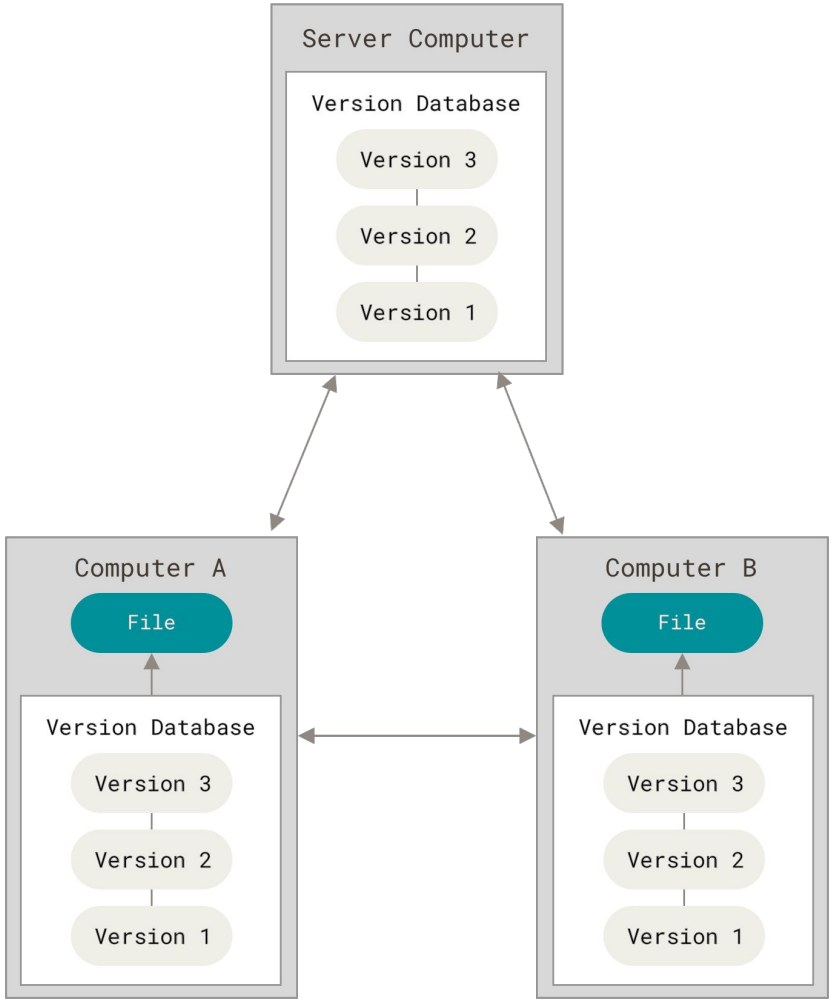
Isso é feito através do uso de um sistema de controle de versão, como o GIT, que permite rastrear as mudanças feitas em cada arquivo e a data em que foram feitas e, principalmente, quem as realizou.



Controle de Versão - GIT

GIT é um sistema de controle de versão, ou versionamento, que permite que você rastreie alterações em seus arquivos ao longo do tempo.

Com isso, você pode criar diferentes versões de um arquivo e alterar entre elas facilmente.



Version Control

GIT

https://git-scm.com/docs

Uma Breve História do Git

Como muitas coisas na vida, (principalmente quando se envolve Linus Torvalds) o Git começou com um pouco de destruição criativa é uma ardente controvérsia.

O núcleo (kernel) do Linux é um projeto de código aberto com um escopo bastante grande. A maior parte da vida da manutenção do núcleo o Linux (1991-2002), as mudanças no código eram compartilhadas como correções e arquivos.

Uma Breve História do Git

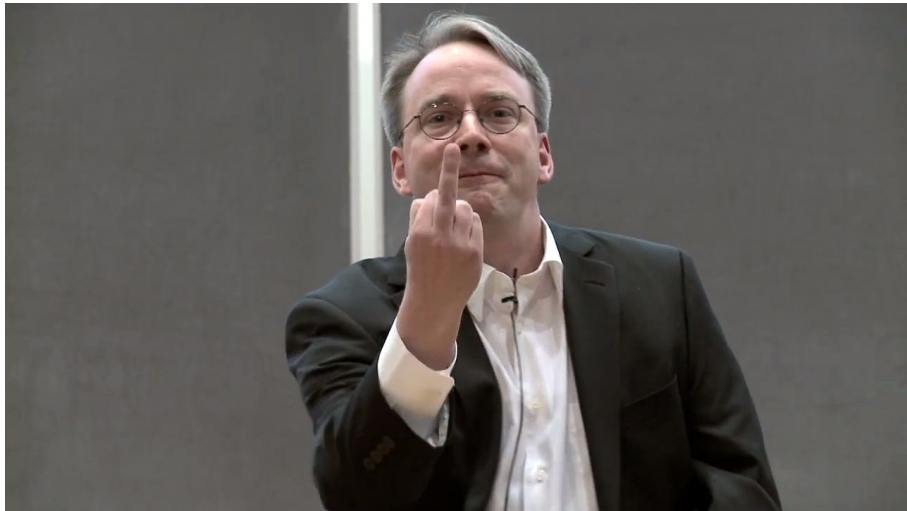
Em 2002, o projeto do núcleo do Linux começou usar uma DVCS proprietária chamada BitKeeper.

Mas como todo software bom e de graça, em 2005 ele se tornou, pago. Isto alertou a comunidade que desenvolvia o Linux (e especialmente Linus Torvalds, o criador do Linux) a desenvolver a sua própria ferramenta baseada em lições aprendidas ao usar o BitKeeper.

World Web Wide

E segundo o próprio Linus em uma entrevista ao GitHub:

> — “ **bem, foram cerca de 10 dias até que eu pudesse usá-lo para o kernel, sim**. Mas, para ser justo, todo o processo começou em dezembro ou novembro do ano anterior, então em 2004.”

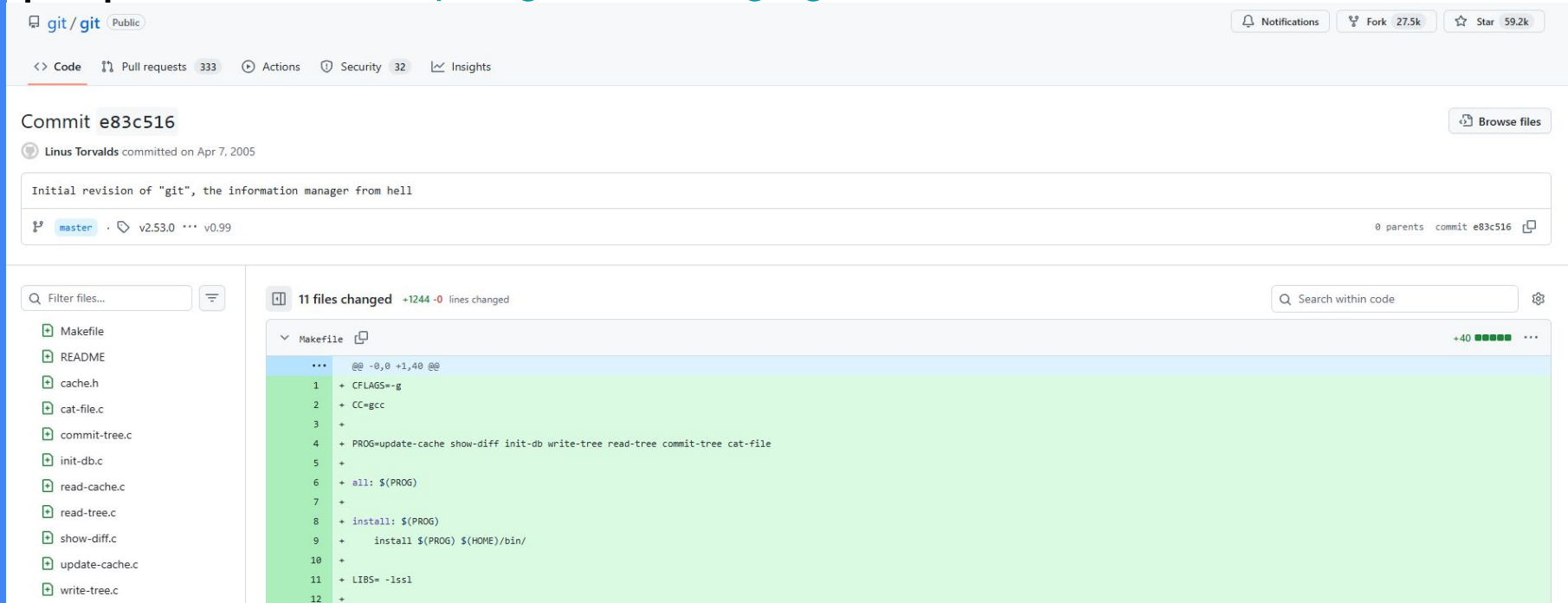


Git turns 20: A Q&A with Linus Torvalds - The GitHub Blog

Linus Torvalds

Repositório do GIT

Sim, podemos encontrar o código fonte do GIT dentro de um repositório no GitHub, feito nada mais nada menos que o próprio Linus https://github.com/git/git

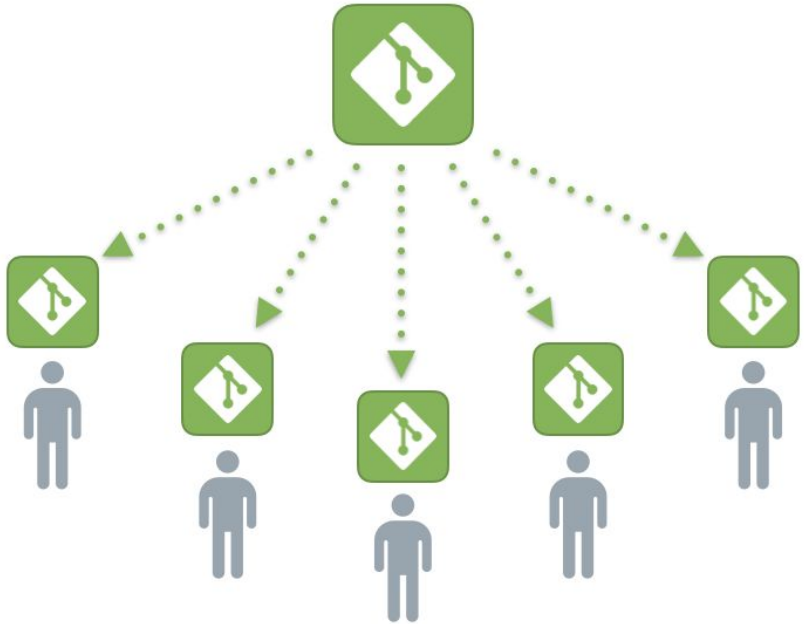


GIT

Várias pessoas podem trabalhar no mesmo arquivo, ao mesmo tempo, sem se preocupar em perder o trabalho um do outro.

O GIT ainda mantém registro de todas as alterações feitas em um arquivo, o que lhe permite voltar às versões anteriores

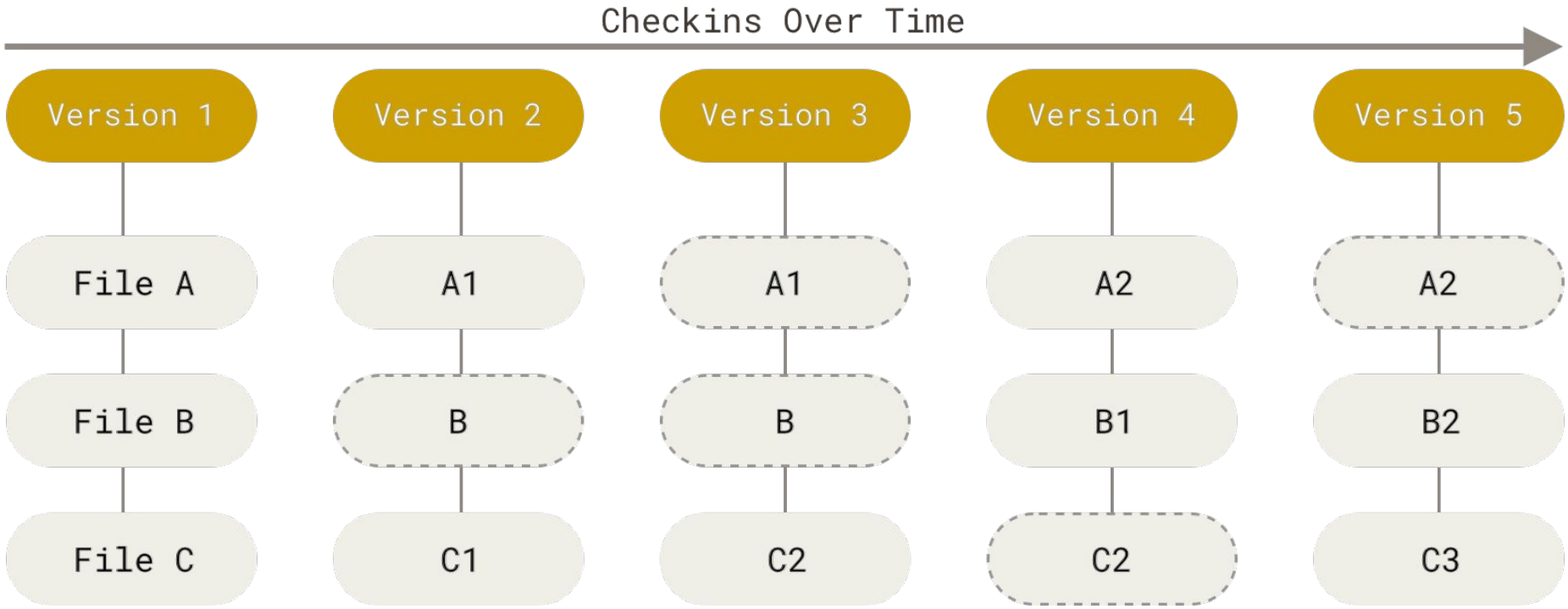
O GIT trata seus dados como um conjunto de imagens de um sistema de arquivos em miniatura



GIT

Toda vez que você salvar uma nova versão o que é salvo é o estado de seu projeto GIT, ele basicamente tira uma foto de todos os arquivos e armazena uma referência para esse conjunto de arquivos.

Caso o arquivo não tenha sido alterado, ele não armazena o arquivo novamente, apenas cria um link para o arquivo idêntico anterior já armazenado.


Checkins Over Time
Version 1
Version 2
Version 3
Version 4
Version 5
File A
File B
File C
A1
B
C1
A1
B
C2
A2
B1
C2
A2
B2
C3

Comandos GIT

git init: inicializa um novo repositório git;

git add: adiciona alterações aos arquivos ao stage;

git commit: cria um novo commit com as alterações adicionadas;

git status: exibe o status atual do repositório, incluindo arquivos modificados e não adicionados ao stage;

git log: exibe um histórico detalhado de commits;

Comandos GIT

git clone: clona um repositório remoto existente na internet ou em sua própria máquina;

git push: envia as alterações locais para um repositório remoto;

git pull: obtém as alterações mais recentes de um repositório remoto;

Comandos GIT

git branch: exibe uma lista de branches (ramos) no repositório atual;

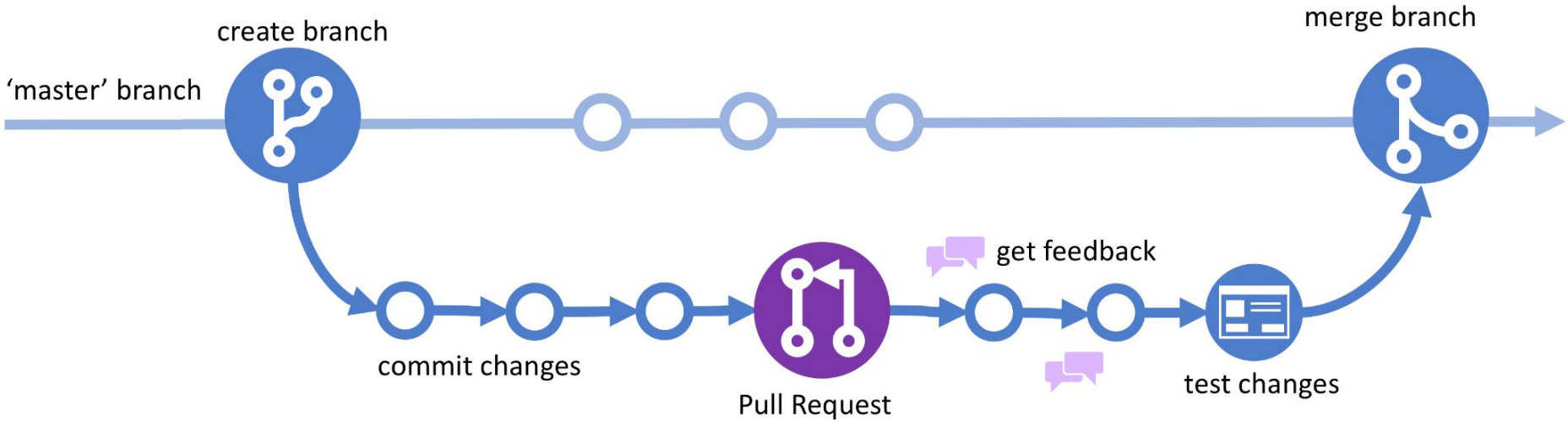
git checkout: muda para um branch (ramo) no repositório atual;

git merge: combina as alterações de um ramo para outro;

git remote: gerencia as conexões com repositórios remotos;

git diff: exibe as diferenças entre duas versões de um arquivo.

GitHub Flow


create branch
merge branch
‘master’ branch
get feedback
commit changes
Pull Request
test changes

COMO CRIAR CONTA NO GITHUB

https://github.com/signup?source=login

OBRIGADO

daciofmf@gmail.com