Introdução ao Git e Docker - Parte 01

Robson Parmezan Bonidia[1]

[1]Fatec - Ourinhos
Tecnologia em Segurança da Informação
Segurança em Sistemas Operacionais e Redes de Computadores I

Ourinhos, SP, Brasil

Índice

Controle de versão

Controle de versão

Controle de versão

Controle de versão

- **O que é "controle de versão" e por que você deve se preocupar?**

Controle de versão

- **O que é "controle de versão" e por que você deve se preocupar?**
  - O controle de versão é um sistema que registra as alterações em um arquivo ou conjunto de arquivos ao longo do tempo para que você possa recuperar versões específicas posteriormente.

Controle de versão

- **O que é "controle de versão" e por que você deve se preocupar?**
  - O controle de versão é um sistema que registra as alterações em um arquivo ou conjunto de arquivos ao longo do tempo para que você possa recuperar versões específicas posteriormente.
  - Aqui, você usará o código-fonte do software como os arquivos com controle de versão, embora na realidade você possa fazer isso com quase qualquer tipo de arquivo em um computador.

Controle de versão

- **O que é "controle de versão" e por que você deve se preocupar?**
  - O controle de versão é um sistema que registra as alterações em um arquivo ou conjunto de arquivos ao longo do tempo para que você possa recuperar versões específicas posteriormente.
  - Aqui, você usará o código-fonte do software como os arquivos com controle de versão, embora na realidade você possa fazer isso com quase qualquer tipo de arquivo em um computador.

Controle de versão

- **O que é "controle de versão" e por que você deve se preocupar?**

Controle de versão

- **O que é "controle de versão" e por que você deve se preocupar?**
  - **Exemplo: Se você é um designer gráfico ou web e deseja manter todas as versões de uma imagem ou layout (o que você certamente desejaria), um Sistema de Controle de Versão (VCS) é o método mais inteligente de se usar.**

Controle de versão

- **O que é "controle de versão" e por que você deve se preocupar?**
  - **Exemplo: Se você é um designer gráfico ou web e deseja manter todas as versões de uma imagem ou layout (o que você certamente desejaria), um Sistema de Controle de Versão (VCS) é o método mais inteligente de se usar.**
  - Ele permite que você reverta os arquivos selecionados para um estado anterior, reverta todo o projeto para um estado anterior, compare as alterações ao longo do tempo, veja quem modificou pela última vez algo que pode estar causando um problema, quem o introduziu e quando, e muito mais.

Controle de versão

- **Tipos de sistemas de controle de versão:**

Controle de versão

- **Tipos de sistemas de controle de versão:**
  - Sistema de controle de versão local;

Controle de versão

- **Tipos de sistemas de controle de versão:**
  - Sistema de controle de versão local;
  - Sistema de controle de versão centralizado;

Controle de versão

- **Tipos de sistemas de controle de versão:**
  - Sistema de controle de versão local;
  - Sistema de controle de versão centralizado;
  - Sistema de controle de versão distribuído.

Controle de versão

- **Sistema de controle de versão local:**

Controle de versão

- **Sistema de controle de versão local:**
  - É um banco de dados local localizado em seu computador local, no qual cada alteração de arquivo é armazenada como um **patch** (Um patch é um conjunto de alterações em um programa de computador);

Controle de versão

- **Sistema de controle de versão local:**
  - É um banco de dados local localizado em seu computador local, no qual cada alteração de arquivo é armazenada como um **patch** (Um patch é um conjunto de alterações em um programa de computador);
  - Para ver a aparência do arquivo em um determinado momento, é necessário adicionar todos os patches relevantes ao arquivo na ordem até aquele momento;

Controle de versão

- **Sistema de controle de versão local:**
  - É um banco de dados local localizado em seu computador local, no qual cada alteração de arquivo é armazenada como um **patch** (Um patch é um conjunto de alterações em um programa de computador);
  - Para ver a aparência do arquivo em um determinado momento, é necessário adicionar todos os patches relevantes ao arquivo na ordem até aquele momento;
  - O principal problema com isso é que tudo é armazenado localmente. Se algo acontecesse com o banco de dados local, todos os patches seriam perdidos;

Controle de versão

- **Sistema de controle de versão local:**
  - É um banco de dados local localizado em seu computador local, no qual cada alteração de arquivo é armazenada como um **patch** (Um patch é um conjunto de alterações em um programa de computador);
  - Para ver a aparência do arquivo em um determinado momento, é necessário adicionar todos os patches relevantes ao arquivo na ordem até aquele momento;
  - O principal problema com isso é que tudo é armazenado localmente. Se algo acontecesse com o banco de dados local, todos os patches seriam perdidos;
  - Se algo acontecesse com uma única versão, todas as alterações feitas após essa versão seriam perdidas.

Controle de versão

- **Sistema de controle de versão local:**

Controle de versão

- **Sistema de controle de versão local:**
  - O Revision Control System (RCS) gerencia várias revisões de arquivos. O RCS automatiza o armazenamento, recuperação, registro, identificação e mesclagem de revisões.

Controle de versão

- **Sistema de controle de versão local:**
  - O Revision Control System (RCS) gerencia várias revisões de arquivos. O RCS automatiza o armazenamento, recuperação, registro, identificação e mesclagem de revisões.
  - O RCS é útil para textos revisados com frequência, incluindo código-fonte, programas, documentação, gráficos, papéis e cartas-padrão.
  - Até mesmo o popular sistema operacional Mac OS X inclui o comando rcs quando você instala as Ferramentas de Desenvolvimento.

Controle de versão

- **Sistema de controle de versão local:**

Controle de versão

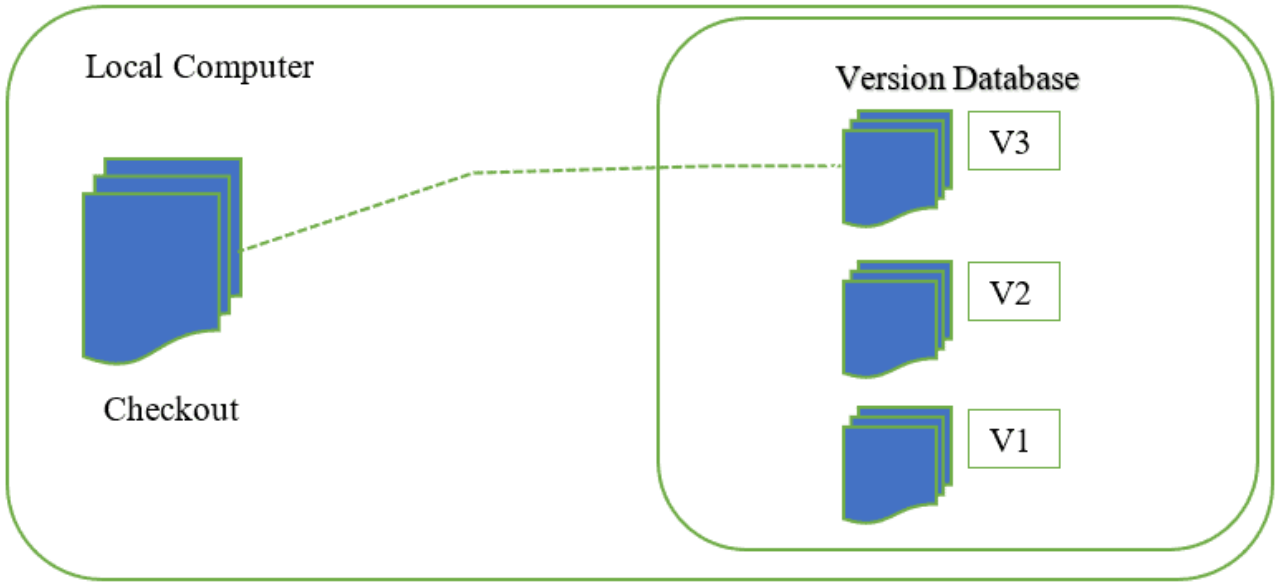
- **Sistema de controle de versão local:**

- https://serengetitech.com/tech/introduction-to-git-and-types-of-version-control-systems/

Controle de versão

- **Sistema de controle de versão local:**

- https://serengetitech.com/tech/introduction-to-git-and-types-of-version-control-systems/



Controle de versão

- **Sistema de controle de versão centralizado:**

Controle de versão

- **Sistema de controle de versão centralizado:**
  - Agora, este sistema possui um único servidor que contém todas as versões dos arquivos;

Controle de versão

- **Sistema de controle de versão centralizado:**
  - Agora, este sistema possui um único servidor que contém todas as versões dos arquivos;
  - Isso permite que vários clientes acessem arquivos simultaneamente no servidor, puxando-os para seu computador local ou empurrando-os para o servidor a partir de seu computador local.

Controle de versão

- **Sistema de controle de versão centralizado:**
  - Agora, este sistema possui um único servidor que contém todas as versões dos arquivos;
  - Isso permite que vários clientes acessem arquivos simultaneamente no servidor, puxando-os para seu computador local ou empurrando-os para o servidor a partir de seu computador local.
  - Dessa forma, todos geralmente sabem o que todos os outros no projeto estão fazendo. Os administradores têm controle sobre quem pode fazer o quê.

Controle de versão

- **Sistema de controle de versão centralizado:**

Controle de versão

- **Sistema de controle de versão centralizado:**
  - Isso permite uma colaboração fácil com outros desenvolvedores ou uma equipe.

Controle de versão

- **Sistema de controle de versão centralizado:**
  - Isso permite uma colaboração fácil com outros desenvolvedores ou uma equipe.
  - Os exemplos mais conhecidos de sistemas de controle de versão centralizados são o Microsoft Team Foundation Server (TFS) e o Apache Subversion (SVN).

Controle de versão

- **Sistema de controle de versão centralizado:**
  - Isso permite uma colaboração fácil com outros desenvolvedores ou uma equipe.
  - Os exemplos mais conhecidos de sistemas de controle de versão centralizados são o Microsoft Team Foundation Server (TFS) e o Apache Subversion (SVN).
  - No entanto, esta configuração também tem algumas desvantagens graves. O mais óbvio é o ponto único de falha que o servidor centralizado representa. Se esse servidor der problema por uma hora, durante essa hora ninguém pode colaborar ou salvar as alterações de versão para o que quer que eles estejam trabalhando.

Controle de versão

- **Sistema de controle de versão centralizado:**

Controle de versão

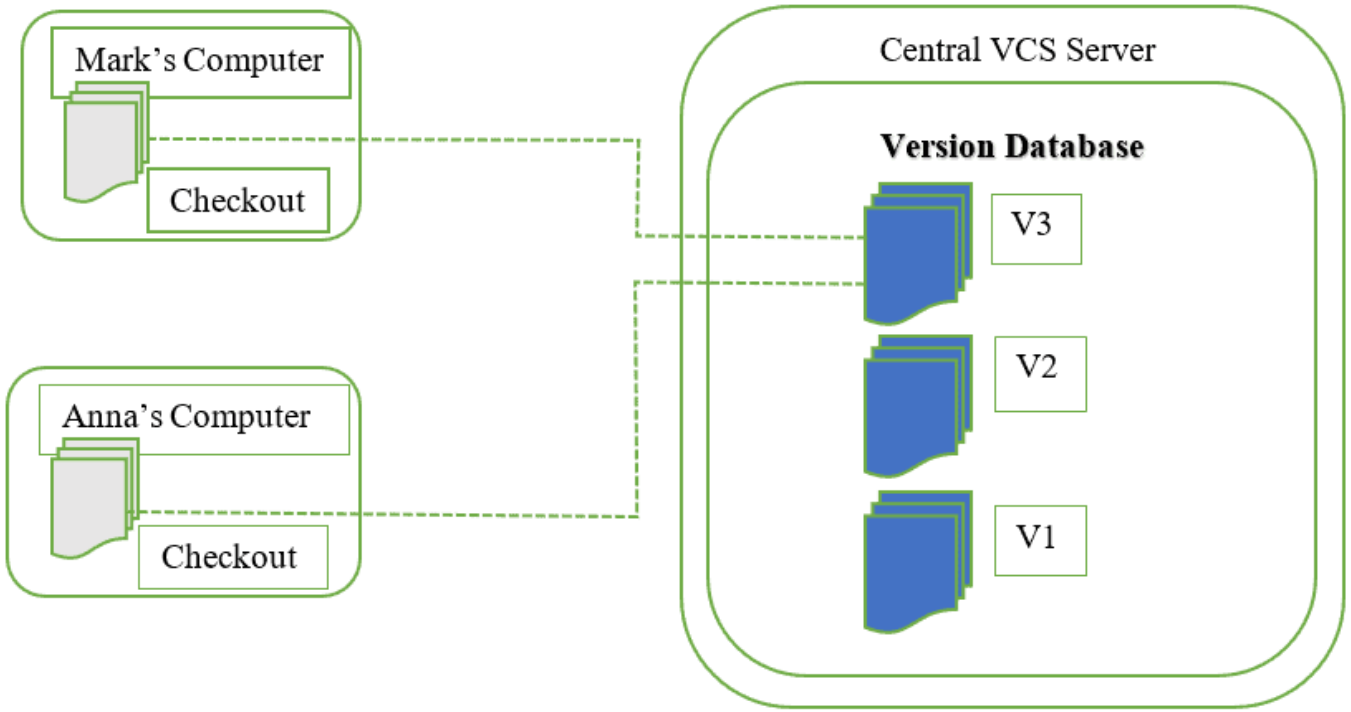
- **Sistema de controle de versão centralizado:**

- https://serengetitech.com/tech/introduction-to-git-and-types-of-version-control-systems/

Controle de versão

- **Sistema de controle de versão centralizado:**

- https://serengetitech.com/tech/introduction-to-git-and-types-of-version-control-systems/



Controle de versão

- **Sistema de controle de versão distribuído:**

Controle de versão

- **Sistema de controle de versão distribuído:**
  - Neste caso, os clientes não apenas verificam o instantâneo mais recente dos arquivos do servidor, eles espelham totalmente o repositório, incluindo seu histórico completo.

Controle de versão

- **Sistema de controle de versão distribuído:**
  - Neste caso, os clientes não apenas verificam o instantâneo mais recente dos arquivos do servidor, eles espelham totalmente o repositório, incluindo seu histórico completo.
  - Assim, todos os que colaboram em um projeto possuem uma cópia local de todo o projeto, ou seja, possuem seu próprio banco de dados local com seu próprio histórico completo.

Controle de versão

- **Sistema de controle de versão distribuído:**
  - Neste caso, os clientes não apenas verificam o instantâneo mais recente dos arquivos do servidor, eles espelham totalmente o repositório, incluindo seu histórico completo.
  - Assim, todos os que colaboram em um projeto possuem uma cópia local de todo o projeto, ou seja, possuem seu próprio banco de dados local com seu próprio histórico completo.
  - Com este modelo, se o servidor ficar indisponível ou morrer, qualquer um dos repositórios do cliente pode enviar uma cópia da versão do projeto para qualquer outro cliente ou de volta para o servidor quando estiver disponível.

Controle de versão

- **Sistema de controle de versão distribuído:**
  - Neste caso, os clientes não apenas verificam o instantâneo mais recente dos arquivos do servidor, eles espelham totalmente o repositório, incluindo seu histórico completo.
  - Assim, todos os que colaboram em um projeto possuem uma cópia local de todo o projeto, ou seja, possuem seu próprio banco de dados local com seu próprio histórico completo.
  - Com este modelo, se o servidor ficar indisponível ou morrer, qualquer um dos repositórios do cliente pode enviar uma cópia da versão do projeto para qualquer outro cliente ou de volta para o servidor quando estiver disponível.
  - **Git é o exemplo mais conhecido de sistemas de controle de versão distribuídos.**

Controle de versão

- **Sistema de controle de versão distribuído:**

Controle de versão

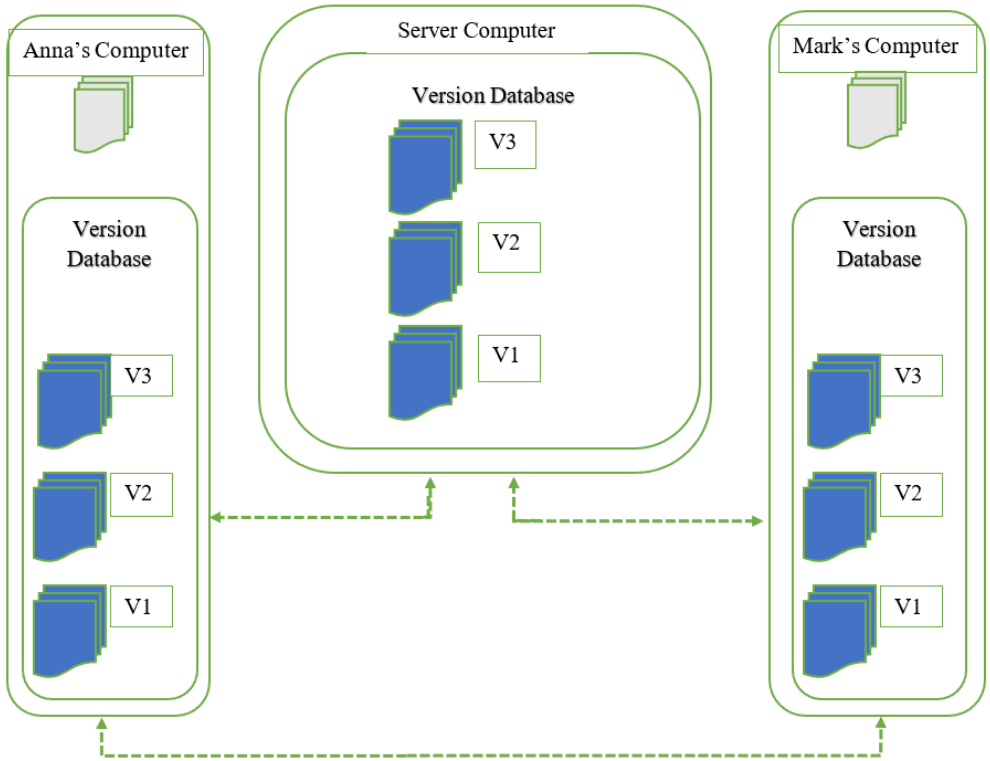
- **Sistema de controle de versão distribuído:**

- https://serengetitech.com/tech/introduction-to-git-and-types-of-version-control-systems/

Controle de versão

- **Sistema de controle de versão distribuído:**

- https://serengetitech.com/tech/introduction-to-git-and-types-of-version-control-systems/



Git - Histórico

Git - Histórico

- **Git - Histórico:**

Git - Histórico

- **Git - Histórico:**
  - Tal como acontece com muitas coisas boas na vida, Git começou com um pouco de destruição criativa e controvérsia ardente.

Git - Histórico

- **Git - Histórico:**
  - Tal como acontece com muitas coisas boas na vida, Git começou com um pouco de destruição criativa e controvérsia ardente.
  - https://destruicaocriativa.com.br/destruicao-criativa/

Git - Histórico

- **Git - Histórico:**
  - Tal como acontece com muitas coisas boas na vida, Git começou com um pouco de destruição criativa e controvérsia ardente.
  - https://destruicaocriativa.com.br/destruicao-criativa/
  - O kernel do Linux é um projeto de software de código aberto de escopo bastante amplo. Durante a maior parte da vida útil da manutenção do kernel Linux (1991–2002), as mudanças no software foram passadas como patches.

Git - Histórico

- **Git - Histórico:**
  - Tal como acontece com muitas coisas boas na vida, Git começou com um pouco de destruição criativa e controvérsia ardente.
  - https://destruicaocriativa.com.br/destruicao-criativa/
  - O kernel do Linux é um projeto de software de código aberto de escopo bastante amplo. Durante a maior parte da vida útil da manutenção do kernel Linux (1991–2002), as mudanças no software foram passadas como patches.
  - Em 2002, o projeto do kernel Linux começou a usar um DVCS (Distributed version control systems) proprietário denominado BitKeeper.

Git - Histórico

- **Git - Histórico:**

Git - Histórico

- **Git - Histórico:**
  - Em 2005, o relacionamento entre a comunidade que desenvolveu o kernel Linux e a empresa comercial que desenvolveu o BitKeeper quebrou, e o status de gratuita da ferramenta foi revogado.

Git - Histórico

- **Git - Histórico:**
  - Em 2005, o relacionamento entre a comunidade que desenvolveu o kernel Linux e a empresa comercial que desenvolveu o BitKeeper quebrou, e o status de gratuita da ferramenta foi revogado.
  - Isso levou a comunidade de desenvolvimento do Linux (e em particular Linus Torvalds, o criador do Linux) a desenvolver sua própria ferramenta com base em algumas das lições que aprenderam ao usar o BitKeeper.

Git - Histórico

- **Git - Histórico:**

Git - Histórico

- **Git - Histórico:**
  - Alguns dos objetivos do novo sistema eram os seguintes: Velocidade • Design simples • Forte suporte para desenvolvimento não linear (milhares de ramificações paralelas) • Totalmente distribuído • Capaz de lidar com grandes projetos como o kernel do Linux de forma eficiente.

Git - Histórico

- **Git - Histórico:**
  - Alguns dos objetivos do novo sistema eram os seguintes: Velocidade • Design simples • Forte suporte para desenvolvimento não linear (milhares de ramificações paralelas) • Totalmente distribuído • Capaz de lidar com grandes projetos como o kernel do Linux de forma eficiente.
  - Desde seu nascimento em 2005, o Git evoluiu e amadureceu para ser fácil de usar e ainda assim manter essas qualidades iniciais.

O que é Git?

O que é Git?

- **O que é Git?**

O que é Git?

- **O que é Git?**
  - A principal diferença entre o Git e qualquer outro VCS (incluindo Subversion e amigos) é a maneira como o Git pensa sobre seus dados.

O que é Git?

- **O que é Git?**
  - A principal diferença entre o Git e qualquer outro VCS (incluindo Subversion e amigos) é a maneira como o Git pensa sobre seus dados.
  - Conceitualmente, a maioria dos outros sistemas armazenam informações como uma lista de alterações baseadas em arquivo.

O que é Git?

- **O que é Git?**
  - A principal diferença entre o Git e qualquer outro VCS (incluindo Subversion e amigos) é a maneira como o Git pensa sobre seus dados.
  - Conceitualmente, a maioria dos outros sistemas armazenam informações como uma lista de alterações baseadas em arquivo.
  - Esses outros sistemas pensam nas informações que armazenam como um conjunto de arquivos e as alterações feitas em cada arquivo ao longo do tempo (isso é comumente descrito como controle de versão baseado em delta).

O que é Git?

- **O que é Git?**

O que é Git?

- **O que é Git?**
  - Git não pensa ou armazena seus dados dessa maneira. Em vez disso, o Git pensa em seus dados mais como uma série de snapshots (Cópia instantânea de volume) de um sistema de arquivos em miniatura.

O que é Git?

- **O que é Git?**
  - Git não pensa ou armazena seus dados dessa maneira. Em vez disso, o Git pensa em seus dados mais como uma série de snapshots (Cópia instantânea de volume) de um sistema de arquivos em miniatura.
  - Com o Git, toda vez que você confirma ou salva o estado do seu projeto, o Git basicamente tira uma imagem de como todos os seus arquivos se parecem naquele momento e armazena uma referência a esse snapshot.
  - Para ser eficiente, se os arquivos não foram alterados, o Git não armazena o arquivo novamente, apenas um link para o arquivo anterior idêntico que ele já armazenou. Git pensa em seus dados mais como um fluxo de snapshots.

O que é Git?

- **O que é Git?**

O que é Git?

- **O que é Git?**
  - Esta é uma distinção importante entre o Git e quase todos os outros VCSs.

O que é Git?

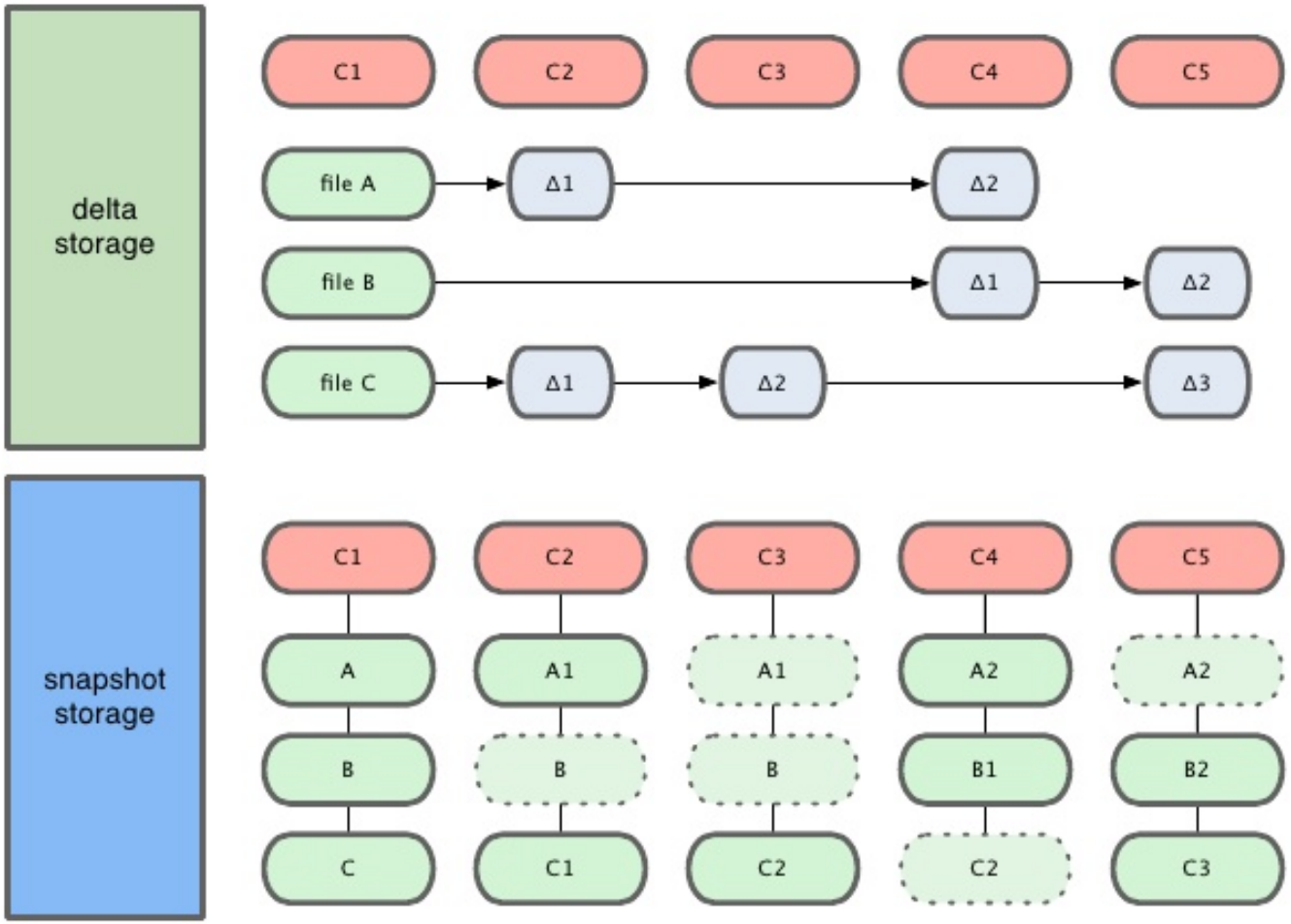
- **O que é Git?**
  - Esta é uma distinção importante entre o Git e quase todos os outros VCSs.
  - Isso faz com que o Git reconsidere quase todos os aspectos do controle de versão que a maioria dos outros sistemas copiava da geração anterior.

O que é Git?

- **O que é Git?**

O que é Git?

- **O que é Git?**



Instalando e Usando Git

Instalando e Usando Git

Instalando no Linux:

Instalando e Usando Git

Instalando no Linux:

Comando
sudo apt install git-all
sudo apt install git
git –version

Configuração Git:

Configuração Git:

Comando
git config –list –show-origin
git config –global user.name "John Doe"
git config –global user.email johndoe@example.com
git config –list

Instalando e Usando Git

Help:

Instalando e Usando Git

Help:

Comando

git help <verb>

git <verb> –help

man git-<verb>

git help config

Instalando e Usando Git

- **Iniciando um repositório:**

Instalando e Usando Git

- **Iniciando um repositório:**
  - **Obtendo um repositório Git:**

Instalando e Usando Git

- **Iniciando um repositório:**
  - **Obtendo um repositório Git:**
  - **Github:** http://gabsferreira.com/criando-e-enviando-arquivos-para-seu-repositorio-no-github/

Instalando e Usando Git

- **Iniciando um repositório:**
  - **Obtendo um repositório Git:**
  - **Github:** http://gabsferreira.com/criando-e-enviando-arquivos-para-seu-repositorio-no-github/
  - O Github é um serviço web que oferece diversas funcionalidades extras aplicadas ao git, gratuitamente.

Instalando e Usando Git

Iniciando um repositório:

Instalando e Usando Git

Iniciando um repositório:

Comando

```
cd /home/user/my_project

git init

git add *.c

git add LICENSE

git add *
```

Instalando e Usando Git

Iniciando um repositório:

Instalando e Usando Git

Iniciando um repositório:

Comando

git commit -m 'Initial project version'

git remote add origin <servidor>

git push origin master

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

Clonando um Repositório Existente:

Clonando um Repositório Existente:

Comando

```
git clone https://github.com/Bonidia/SARS-CoV-Predictor-v1

git clone https://github.com/Bonidia/SARS-CoV-Predictor-v1 SARS-CoV
```

Verificando o status de seus arquivos:

Verificando o status de seus arquivos:

Comando
git status
git add README
git status

Git no servidor - Gerando sua chave pública SSH:

Git no servidor - Gerando sua chave pública SSH:

Muitos servidores Git se autenticam usando chaves públicas SSH. Para fornecer uma chave pública, cada usuário em seu sistema deve gerar uma, se ainda não tiver uma:

Git no servidor - Gerando sua chave pública SSH:

Muitos servidores Git se autenticam usando chaves públicas SSH. Para fornecer uma chave pública, cada usuário em seu sistema deve gerar uma, se ainda não tiver uma:

Comando

cd ~/.ssh

ssh-keygen

cat ~/.ssh/id_rsa.pub

Ignorando arquivos:

Instalando e Usando Git

Ignorando arquivos:

Comando

cat .gitignore

ignore all .a files: *.a

only ignore the TODO file in the current directory, not subdir/TODO: /TODO

ignore all files in any directory named build: build/

ignore all .pdf files in the doc/ directory and any of its subdirectories: doc/**/*.pdf

Removendo/Movendo arquivos:

Removendo/Movendo arquivos:

Comando

git rm PROJECTS.md

git mv README.md README

git log: **Viewing the Commit History**

git log –pretty=format:"%h - %an, %ar : %s"

https://git-scm.com/docs/pretty-formats - marcador de posição

Working with Remotes:

Working with Remotes:

Repositórios remotos são versões de seu projeto que estão hospedadas na Internet ou rede em algum lugar.

Instalando e Usando Git

Working with Remotes:

Repositórios remotos são versões de seu projeto que estão hospedadas na Internet ou rede em algum lugar.

Comando

git clone https://github.com/schacon/ticgit

git remote -v

git fetch <remote>: **O comando vai para esse projeto remoto e puxa todos os dados desse projeto remoto que você ainda não tem.**

git fetch origin - If you clone a repository

git pull

Ramificando:

Ramificando:

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.

Instalando e Usando Git

Ramificando:

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.

Comando

git checkout -b funcionalidade_x: **criar um novo branch**

git checkout master: **retorne para o master**

git branch -d funcionalidade_x: **remova o branch**

git push origin <funcionalidade_x>: **um branch não está disponível a outros a menos que você envie o branch para seu repositório remotol.**

Atualizar & Mesclar:

Atualizar & Mesclar:

Comando

git pull: **atualizar seu repositório local com a mais nova versão**

git merge <branch>: **para fazer merge de um outro branch ao seu branch ativo (ex. master)**

git diff <branch origem> <branch destino>: **fazer o merge das alterações, você pode também pré-visualizá-as**

Rotulando: Criar rótulos para releases de software

Rotulando: Criar rótulos para releases de software

Comando

git tag 1.0.0 1b2e1d63ff

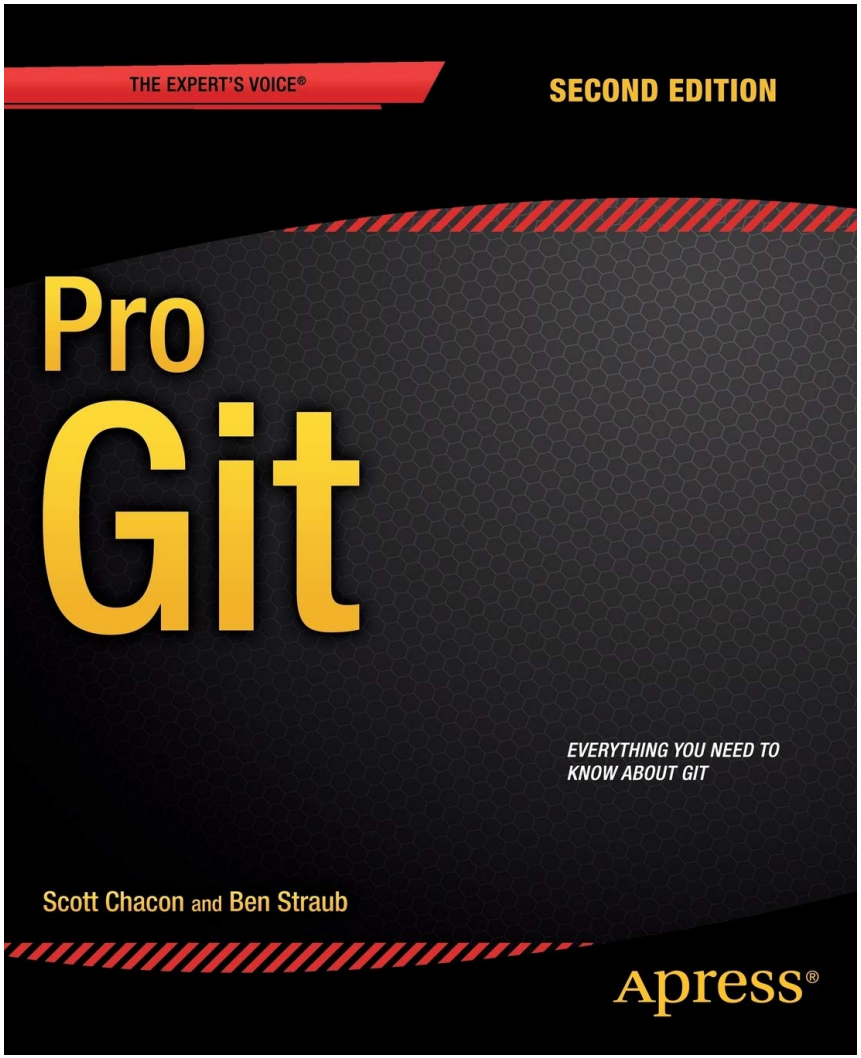
1b2e1d63ff representa os 10 primeiros caracteres do id de commit que você quer referenciar com seu rótulo.

git log: **obter o id de commit**

Refs: https://rogerdudler.github.io/git-guide/index.pt_BR.html

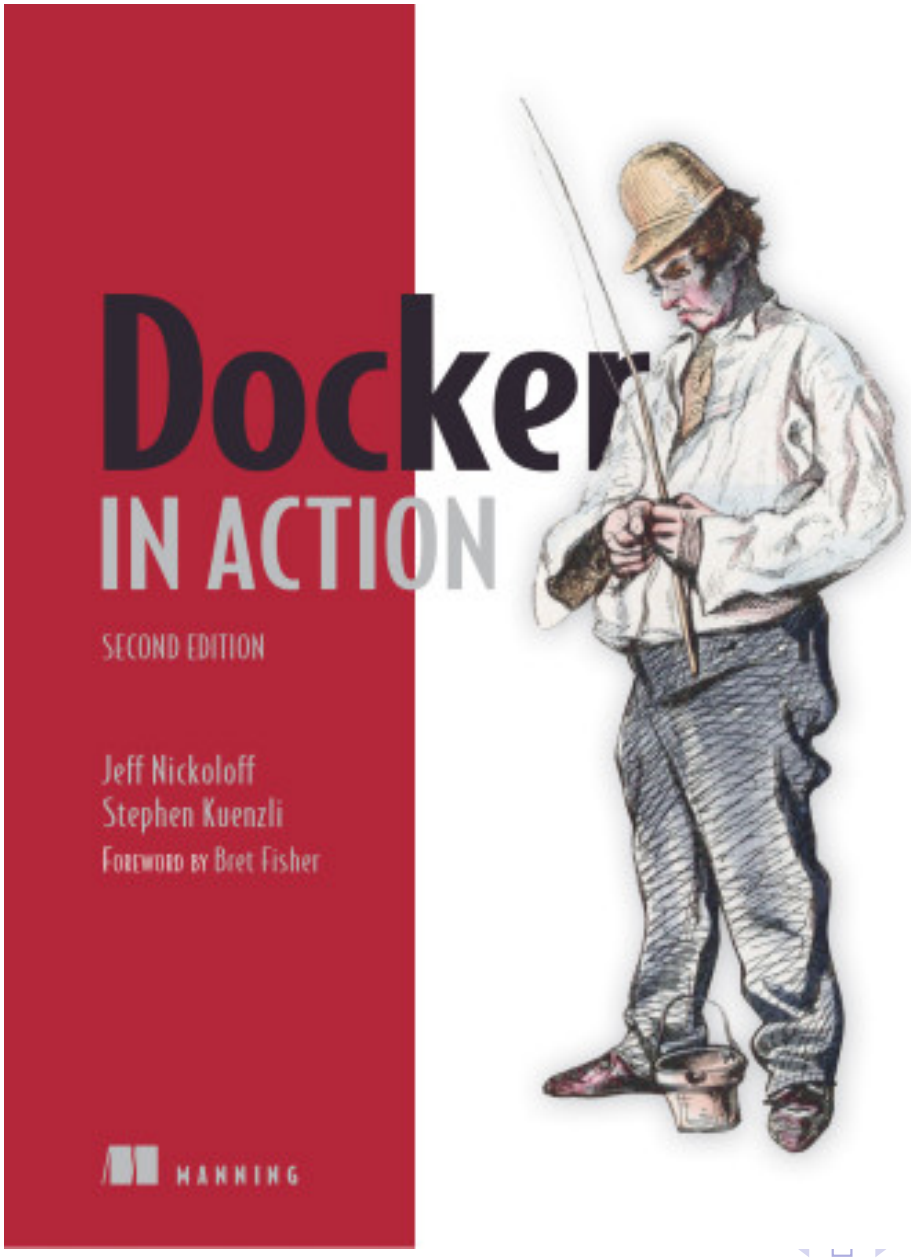
Bibliografia

Bibliografia



Bibliografia

Bibliografia



Agradecimentos

Obrigado pela atenção!!!

Nome: Robson P. Bonidia
E-mail: rpbonidia@gmail.com
Repositório: `https://github.com/Bonidia`
Lattes: http://lattes.cnpq.br/1572375422051077
Site: `https://bonidia.github.io/website/`