Curso de Git

**Introdução ao Git e Controle de versão**

Git é um software desenvolvido por Linus Torvalds para gerenciar o Kernel do Linux.

O Git permite fazer o controle de versão de uma aplicação e gerencia a evolução do código fonte.

O Git gerencia diferentes versões e controla as alterações.

Existem, também, ferramentas gráficas para o controle de versão.

**Tipos de ferramentas para controle de versão:**

GIT, CVS, Mercurial, Subversion/SNV

**Conceitos**

O Git é para ser usado via terminal usando o prompt de comando.

Repositório local, é o da sua máquina, quando usa o Git init em um diretório

Repositório Remoto, é o que se encontra no servidor – Compartilhado e distribuído.

Após alterar o arquivo, fazemos o COMMIT no arquivo que se encontra no repositório local.

Quando fazemos um PUSH colocamos o arquivo commitado no repositório remoto, fica disponível para toda a equipe

**Comandos básicos do git**

Git Init – Inicializa um repositório dentro de um diretório (devo estar dentro do diretório)

Git Clone – Usado para baixar base de um repositório remoto para repositório local, clona um projeto.

Git add . – Adiciona todos os arquivos para que o Git possa rastrea-lo. Adiciona os arquivos para ser comitados.

Git commit -m <mensagem> - Salva alteração e torna arquivo rastreavel.

O parâmetro “m”serve para especificar a mensagem Commit.

O parâmetro mensagem, serve para colocar uma mensagem simples e específica sobre a alteração que foi feita até aquele ponto do aplicativo.

Git Push – Usado para enviar arquivo do servidor local para o remoto. Os arquivos devem ser enviados após serem alterados e commitados.

Git Pull – Atualiza repositório local com base no remoto

Git Merge – Usado para juntar dois Branches que estamos alterando. Serve para resolver conflitos de alteração.

Clonar – Git clone + URL ( no terminal)

Quando o git não identifica quem fez uma mudança, ele informa o arquivo e solicita que o conflito seja resolvido manualmente.

Git Status – Apresenta informações sobre o projeto, sempre que ele sofre alteração, esta será apresentada. Mostra ainda arquivos adicionados, se foi commitado. Mostra o estado atual do projeto.

Git Log – Apresenta o histórico de alteração que foram feitas no repositório. Apresenta autor, hora, id, mensagem, commits feitos

Git Branch – Usado quando criamos ramificação no projeto.

* O Git Init cria o ramo MASTER ou principaldo projeto.
* Exemplo: Se temos um site funcionando e criamos uma outra instancia tipo um cadastro, dando Git Branch “cadastro” ele pega todo o código principal no estado atual do repositório master que está funcionando e coloca em um ramo independente do ramo principal.
* Se der algum problema no ramo secundário, isto não afetará o ramo principal.

Exemplo prático:

**Criando uma página no terminal**

mkdir CursoGit – Cria diretório com nome cursogit

cd CursoGit – move para o diretório

git init – Torna este diretório um repositório local git

Criamos um diretório, entramos nele e inicializamos um repositório Git

Podemos usar git status para observar o estado

**Criando um arquivo**

Touch index.html

Com este comando, criamos o arquivo index.html. Após a criação, o mesmo ainda não está monitorado pelo Git.

Se dermos o comando <git status> o arquivo será exibido em vermelho mostrando assim que ele não foi adicionado para ser monitorado pelo Git.

**Adicionando arquivo para ser gerenciado pelo Git**

Git add index.html

Este comando adiciona o arquivo para ser monitorado pelo Git.

Podemos dar um<git status> e veremos que o arquivo será exibido em verde indicando que ele está sendo monitorado e já podemos commitar ou seja fazer alterações que o git gerenciará.

**Commitando o arquivo**

Git commit -m “meu primeiro commit”

Se usarmos< git status>

Veremos o arquivo commitado com a mensagem.

* Commitar alterações significa salvar o estado em que o código fonte está naquele momento, permite rastrear e identificar todo o percurso das alterações feitas no código fonte.

Git log

Este comando retorna o histórico das alterações tais como usuário, hora, etc

**Usando Git em repositório remoto**

Repositório remoto serve para trabalharmos em equipe ou com pessoas

**Tipos de repositórios remotos:**

GitHub, Git Lab, Bit Bucket

**O GitHub**

Site que permite compartilhar projeto local, expor código fonte, serve de repositório, uso do GIT, serve de rede de compartilhamento para desenvolvedor.

**Abas do GitHub**

* Para criar novo repositório no git hub, clica no +

**Pull** **request** – Quando você faz um commit/alteração em um projeto e aguarda o dono analisar

**Issues** – Usado para erros e bugs

**Market** **place** – Tem diversas ferramentas para ajudar no desenvolvimento.

**Explorer** – Exibe noticias, eventos informações, atc

**Readme** – Usado para definir como funciona o código fonte e contem instruções de uso e como baixar. Define como usar biblioteca e clonar.

**Settings** – Usado para adicionar pessoas para ajudar no projeto

**Estrela** – Qualifica o projeto

Criando repositório

Clica no + no git hub e estabelece o nome do repositório

Adicionando uma origem de repositório remoto

Após criar o repositório remoto, informa ao git qual a origem remota associada a este repositório local.

Para isso, entra no diretório / repositório local e da o seguinte comando:

Git remote add origin ‘url apresentada no repositório remoto que foi criado com o +’

Removendo uma origem de repositório remoto

No repositório local, da o seguinte comando:

Git remote rm origin URL

**PUSH – Exemplo prático**

1 – Alteramos um arquivo no terminal ou editor local

2 – <git Status> - O git informa que o arquivo foi modificado

3 - <git add > “nome do arquivo” - Torna o arquivo gerenciável pelo git

4 - <git status> - O git informa que o arquivo está gerenciável

5 - <git commit -m ‘mensagem’ – informa as alterações feitas no arquivo

* Sempre que fizer alteração no arquivo, temos que usar:
* <git add nome do arquivo> para torna-lo gerenciável
* <git commit -m “mensagem”> para colocar a descrição do que foi alterado.

<git log> - mostra os commits realizados

Depois de comitado, temos que enviar o arquivo para o github...

Git push – u origin master

* Vai pedir usuario e senha

<git push> - envia os arquivos para o repositório remoto e iguala as versões local e remota

* Não precisa do parâmetro -U pois os repositórios estão vinculados.