Git e GitHub em 1 Hora

<https://brq.udemy.com/course/git-e-github-em-1-hora/learn/>

Versionamento: gerenciamento de versões

Vantagens: trabalho em equipe, resgate de versões estáveis, segurança, backup

Git: ferramenta de controle de versão

GitHub: repositório de projetos e código, dependente do Git

Git:

git --version

git config --global user.name "Usuario 123"

git config --global user.name

git init

git remote add origin https://github.com/orosolin2/teste.git

git status

git add Readme.txt

No GIT se adiciona/remove depois comita: git add <file or folder>

Commit: agrupar em um pacote todas as alterações realizadas: git commit -m “msg”

Para saber o status do repositório: git status

Para enviar para o servidor: git push origin master

Branches podem ser utilizadas para funcionalidades distintas

Para receber do servidor: git pull origin master

Para exibir todas diferenças entre o repositório local e o servidor: git diff

Para exibir o histórico de commits realizados: git log

Para exibir o histórico simplificado de commits realizados: git shortlog

Para exibir o que foi feito no commit: git show <commit id>

Após o commit realizado, quer fazer um restore de um commit anterior: git reset <commit id anterior>

Descartar mudanças: git checkout <nome arquivo>

Github.com/danistefanovic/build-your-own-x

Clonar repositório existente: git clone <url repositório git>

Usar o tab faz auto-complete

Fazer o add de todos os arquivos da pasta atual: git add .

Fazer o add e o commit na mesma linha: git commit -am “”

Arquivo readme.md: arquivo com a “apresentação” do seu projeto, indicado existir

Arquivo .gitignore: indica pastas e arquivos que não devem ser versionados (ex.: senhas)

Dicas:

1. Clone o que puder: achou um repo interessante,
2. Pulls constantes: se você trabalha com mais de uma pessoa
3. Sem medo de commitar: mudanças granulares e incrementais, funcionalidades atômicas
4. Mensagens de commits decentes: padronizar e evitar mensagens genéricas
5. Push com segurança: apenas quando tiver um grau de confiança no código
6. Contribua no GitHub: aprendizado e colaboração
7. Use a “rede social” GitHub
8. Readme elegante: informativo, boa documentação
9. Cuidado com vazamentos: senhas, acessos