INSTITUTO FEDERAL
Bahia

Introdução {
‘Git’ & ‘GitHub’
}

Discente >> Manoel Vitor
Docente >> Stéfani Pires

O que é o Git? {

O Git é um sistema de controle de versão distribuído e gratuito que é amplamente utilizado no desenvolvimento de software. Ele permite que desenvolvedores trabalhem em conjunto em um mesmo projeto, gerenciando alterações em código-fonte, documentação e outros arquivos de forma segura e organizada. Com o Git, é possível registrar o histórico de alterações, trabalhar em diferentes versões do código simultaneamente, além de colaborar com outros desenvolvedores de forma simples e eficiente.

}

É o GitHub? {

Já GitHub é uma plataforma online que utiliza o Git como sistema de controle de versão para gerenciar projetos de software de forma colaborativa. Ele oferece uma grande variedade de recursos que facilitam a colaboração, o compartilhamento e o armazenamento de códigos-fonte e outros tipos de arquivos.

}

É o GitHub? {

Com o GitHub, é possível criar, hospedar e compartilhar projetos de software livre, além de contribuir para projetos de outras pessoas. É uma ferramenta essencial para desenvolvedores de software que desejam trabalhar em equipe e controlar as versões de seus projetos de maneira eficiente e segura.

}

Como instalar o [GIT] {

Windows >>
https://git-scm.com/download/win

Linux >>
https://git-scm.com/download/linux

MacOs >>
https://git-scm.com/download/mac

}

Comandos básicos do Git {

- `git clone <URL REPOSITÓRIO>`
- `git branch <NOME DO BRANCH>`
- `git checkout <NOME DA BRANCH>`
- `git status`
- `git add`
- `git commit`
- `git push`
- `git pull`
- `git revert`
- `git merge`

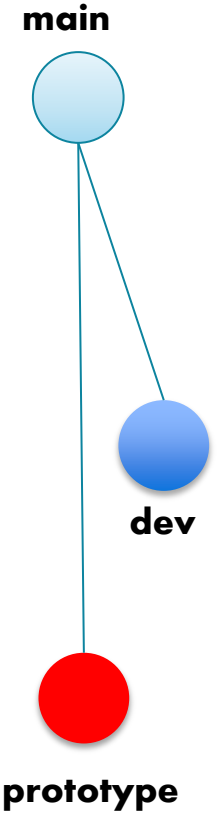
}

Git clone & branch {

- [git clone] - O comando "git clone" é utilizado para criar uma cópia local de um repositório remoto do Git. Ele é útil para baixar um projeto já existente e trabalhar nele localmente. O comando clona todo o histórico de versões do repositório, incluindo os arquivos e as branches existentes.

- [git branch] – O comando "git branch" é utilizado no Git para gerenciar ramificações de um projeto. Ele permite listar, criar, renomear e excluir ramificações. Com o comando "git branch", é possível visualizar todas as ramificações existentes em um repositório, bem como criar e alternar entre elas.

}



Git checkout & status {

- **[checkout] - O comando "git checkout" é usado no Git para alternar entre diferentes branches, que são diferentes versões do código-fonte de um projeto. Ele também pode ser usado para criar uma nova branch a partir de uma existente ou para alternar entre diferentes commits. O comando permite que o usuário navegue facilmente pelo histórico de desenvolvimento do projeto.**

- **[git status] – O comando git status nos dá todas as informações necessárias sobre a branch atual.**

}

Git add & commit {

- **[add] - O comando "git add" é utilizado para adicionar alterações feitas em arquivos ao próximo commit do repositório. Ele prepara os arquivos modificados ou criados para serem incluídos no histórico do Git. É uma etapa fundamental para versionar o código-fonte e controlar o seu desenvolvimento.**

- **[commit] – Git commit é como definir um ponto de verificação no processo de desenvolvimento. Você pode voltar a esse ponto mais tarde, se necessário.**

}

Git push & pull {

- **[push] - Após fazer o commit de suas alterações, a próxima coisa a fazer é enviar suas alterações ao servidor remoto. Git push faz o upload dos seus commits no repositório remoto.**

- **[pull] – O comando git pull é usado para obter as atualizações de um repositório remoto. Esse comando é uma combinação de git fetch e git merge, o que significa que, quando usamos git pull, ele recebe as atualizações do repositório remoto (git fetch) e aplica imediatamente as alterações mais recentes em seu espaço de trabalho local (git merge).**

}

Git revert & merge {

- **[revert] - Às vezes, precisamos desfazer as alterações que fizemos. Existem várias maneiras de se desfazer as alterações em nosso espaço de trabalho local ou remotamente (dependendo do que você necessita), mas devemos usar esses comandos com cuidado para evitar exclusões indesejadas.**

- **[merge] – Quando você concluir o desenvolvimento em sua branch e quando tudo funcionar bem, a etapa final é fazer o merge (mesclar ou unir, em português) da branch com a branch pai (dev ou master/main, em geral). Isso é feito com o comando**

}

Conversão para padronização do Commits

O ***Conventional Commit*** é uma convenção simples de mensagens de commit, que segue um conjunto de regras e que ajuda os projetos a terem um histórico de commit explícito e bem estruturado.

Link do sítio:
https://www.conventionalcommits.org/pt-br/v1.0.0/

Obrigado...

Referências Biográficas

ROSA, Daniel. **10 Comandos do Git que todo desenvolvedor deveria conhecer**. 2022. Disponível em: < https://www.freecodecamp.org/portuguese/news/10-comandos-do-git-que-todo-desenvolvedor-deveria-conhecer/ > Acesso em 07 de Mar. de 2023.

GOMES, Rafael C.. Comandos Git. 2016. Comandos Git: Aprenda git do básico ao avançado. Disponível em: < https://comandosgit.github.io/ >. Acesso em 09 de Mar. De 2023.

AQUILES, Alexandre. **Controlando Versões com Git e Github**. Edição 1. São Paulo. Casa do Código, 2014.