Dicionário Técnico: Conceitos Básicos GitHub

Aluno: Suzana Almeida Coelho

Aluno: Wellingthon Martins Ribeiro

Orientador (a): Lucas Debatin

Git Branch: Segundo Manzieri (2019), os repositórios no GitHub funcionam como uma árvore. Quando criamos um repositório, ele automaticamente é iniciado com a branch *master*, que é equivalente ao tronco da sua árvore, ou seja, é a parte principal do seu projeto. As próximas branches são secundárias e, portanto, ramos da branch master. Usando branches, vários desenvolvedores podem trabalhar em paralelo no mesmo projeto simultaneamente. Podemos usar o comando git branch para criar, listar e deletar branches.

Git Cherry-Pick: esse comando nos permite aplicar alterações introduzidas por commits anteriores – mas cuidado! Se não usado com moderação pode ocasionar conflitos ou reescrita inadequada do histórico de alterações. É um comando útil, pois nos permite adicionar em uma branch somente os commits importantes, evitando commits que estão em testes ou com erros (ESCUDELARIO, 2018).

Git Clone: Segundo Draniceanu (2021), o comando git clone é usado para baixar a versão mais recente de um projeto remoto e copiá-lo para o local selecionado na máquina local. Em outras palavras, o clone do Git basicamente faz uma cópia idêntica da última versão de um projeto em um repositório e a salva no seu computador. Por exemplo, se baixar um projeto do GitHub, tudo o que precisa fazer é clicar no botão verde (clonar ou baixar), copiar o URL na caixa e colá-lo após o comando git clone. Isso fará uma cópia do projeto em sua área de trabalho local para que possa começar a trabalhar com ele.

Git Commit: Um Commit é um pacote de alterações feitas no repositório. Cada commit possui arquivos alterados, autor e uma mensagem de resumo. O Git commit permite criar um commit, ou seja, consegue guardar o estado do seu repositório naquele momento.

Existem diferentes estratégias para fazer commits, mais a ideia principal é que a cada ponto em que o seu código esteja funcionando com uma nova pequena funcionalidade, exista um commit (BERNARDO, 2021).

Git Diff: Segundo Zilberman (2019), é um comando Git multiuso que, quando executado, realiza uma função de comparação nas fontes de dados Git. Essas fontes de dados podem ser commits, ramificações, arquivos e outros.

Git Fetch: O comando git fetch baixa commits, arquivos e referências de um repositório remoto para seu repositório local. Busca (fetching) é o que você faz quando quer ver em que todos estão trabalhando, como é denominado, busca a versão remota, ou seja, baixa a versão remota para que sua versão local fique alinhada com a versão remota (METRING, 2019).

Git Init: Segundo Koushik (2020), o comando git init é geralmente o primeiro comando executado em qualquer novo projeto que ainda não seja um repositório Git. Pode ser usado para converter uma pasta existente sem versão em um repositório Git. Além disso, pode usá-lo para inicializar um repositório vazio, isso transformará o diretório atual em um repositório Git.

Git Pull: O comando git pull é usado para buscar e baixar conteúdo de repositórios remotos e fazer a atualização imediata ao repositório local para que os conteúdos sejam iguais. O git pull busca as últimas alterações enviadas do servidor remoto para o repositório local para que ter as atualizações mais recentes de outros membros da equipe (METRING, 2019).

Git Push: Segundo Eygi (2020), o comando git push é mais usado para publicar modificações locais no repositório central. Após um repositório local ter sido modificado, um comando push é executado para compartilhar as modificações com membros da equipe remota.

Git Remote: Segundo Kulshrestha (2018), o comando git remote permite criar, ver e excluir conexões com outros repositórios. As conexões remotas são mais parecidas com marcadores em vez de links diretos para outros repositórios. Em vez de dar acesso em tempo real a outro repositório, eles funcionam como nomes convenientes que podem ser usados para fazer referência a uma URL não tão conveniente.

REFERÊNCIAS

BERNARDO, Fernanda. **Git commit: confirmando e salvando alterações!** 2021. Disponível em: https://blog.betrybe.com/git/git-commit/. Acesso em: 03 out. 2021.

DRANICEANU, Andreea. 10 Common Git Commands Everyone Should Know. 2021. Disponível em: https://blog.testproject.io/2021/03/22/git-commands-every-sdet-should-know/. Acesso em: 03 out. 2021.

ESCUDELARIO, Bruna de Freitas. **Os 4 comandos do Git que todo desenvolvedor deveria conhecer**. 2018. Disponível em: https://imasters.com.br/desenvolvimento/os-4-comandos-git-que-todo-desenvolvedor-deveria-conhecer. Acesso em: 03 out. 2021.

EYGI, Cem. **10 Git Commands Every Developer Should Know**. 2020. Disponível em: https://www.freecodecamp.org/news/10-important-git-commands-that-every-developer-should-know/. Acesso em: 03 out. 2021.

KOUSHIK, Vikash. **13 Git Commands Every Developer Must Know**. 2020. Disponível em: https://zepel.io/blog/13-git-commands/. Acesso em: 03 out. 2021.

KULSHRESTHA, Saurabh. **Top 20 Git Commands with Example**. 2018. Disponível em: https://medium.com/edureka/git-commands-with-example-7c5a555d14c. Acesso em: 03 out. 2021.

MANZIERI, Catarina. **Entendendo sobre branch e pull request**. 2019. Disponível em: https://medium.com/reprogramabr/entendendo-sobre-branch-e-pull-request-516aea4e364f. Acesso em: 03 out. 2021.

METRING, Ricardo. **Para Que Serve o git fetch - fetch vs pull**. 2019. Disponível em: https://metring.com.br/para-que-serve-o-git-fetch-vs-pull. Acesso em: 03 out. 2021.

ZILBERMAN, Eyar. **10 insanely useful git commands for common git tasks**. 2019. Datree.io. Disponível em: https://www.datree.io/resources/git-commands. Acesso em: 03 out. 2021.