Teoria Git/Github by Jonathas Carneiro

github.com/sahtcarneiro

Tópicos gerais:

- Git e Github (são iguais mas completamente diferentes)
- Como surgiu e o que é o Git
- O que é o Github e qual sua diferença para o Git
- Conceitos importantes
- Peculiaridades importantes do Github
- Uso prático(documento a parte):
  como baixar e configurar o git
  como conectar com o github e como usar o github

Introdução

Git e GitHub são duas ferramentas amplamente utilizadas por desenvolvedores de software em todo o mundo. Embora sejam frequentemente usados em conjunto, Git e GitHub são coisas diferentes e é importante saber a diferença.

Contexto do surgimento do Git

Antigamente, era muito complicado trabalhar com software a nível empresarial, pois haviam equipes enormes trabalhando no mesmo projeto, e muitas vezes no mesmo trecho de código. Tudo ficava muito confuso: quem fez que alteração? Como fazer com que todos vejam as alterações que fiz em tempo real para evitar que várias pessoas estejam trabalhando na mesma coisa?

Surgiu uma ferramenta para isso, porém além de ser terrível de ser usada, em 2005 a empresa tornou a ferramenta paga, alertando boa parte da comunidade de desenvolvedores para ter possibilidades diferentes e não depender de uma única ferramenta. Foi aí que surgiu o criador do Linux, e lançou o Git, focando em resolver todos os problemas que a antiga ferramenta tinha.

O Git foi produzido com o foco de ser:

- Rápido
- Simples
- Robusto para suportar milhares de ramificações
- Abertamente distribuído
- Capaz de suportar uma grande quantidade de dados

O que é Git?

Git é o que chamamos de software de versionamento. Isso significa que ele é responsável por registrar e gerenciar diferentes versões de um projeto de software. Ele possibilita ir salvando cada passo do projeto, sem perder o passo anterior. É como se, ao você salvar, você deixasse uma cópia de como estava o arquivo antes da alteração registrada, com dados de quem alterou, quando e o que fez. Assim, não só temos tudo registrado como também, em caso de qualquer problema ao longo do projeto, podemos escolher para qual versão dele queremos voltar, pois todas continuarão registradas.
No git, chamamos de repositório um local onde os arquivos e as informações de controle de versão são armazenados. Cada vez que você faz uma alteração em um arquivo, o git registra essa alteração em um commit, que é um registro de todas as alterações feitas desde a última vez que um commit foi feito. Isso permite que você acompanhe as alterações ao longo do tempo e reverta para versões anteriores, se necessário.

O que é GitHub?

O GitHub surgiu com a ideia de manter essa proposta do Git, porém utilizando a internet para conectar os desenvolvedores, fornecendo um lugar online para esse versionamento acontecer gratuitamente. Assim, você "democratiza" o acesso a essa ferramenta, pois o poder computacional e de armazenado necessário para gerir e manter todas as versões do programa não vão ser mais do seu computador, mas de um servidor online.

Então, essa é a ideia do GitHub: ter uma plataforma onde qualquer um pode compartilhar seus códigos com o mundo e ainda permitir o versionamento. Assim, é possível registrar o passo a passo do seu projeto, cada mudança ao longo do tempo e que possa voltar para alguma etapa inicial caso algo dê errado ou tenha algum problema.

Então o github é um serviço de hospedagem para projetos git, permitindo que você compartilhe seus repositórios com outras pessoas e facilite a colaboração em projetos de software.

Conceitos Importantes

Repositório:
Um repositório é um espaço onde os arquivos e informações de controle de versão são armazenados. Em um projeto Git, um repositório contém todos os arquivos do projeto, bem como informações sobre as alterações feitas nesses arquivos ao longo do tempo.
O GitHub é uma plataforma de hospedagem de repositórios Git. Ele permite que você crie, compartilhe e colabore em repositórios com outras pessoas.

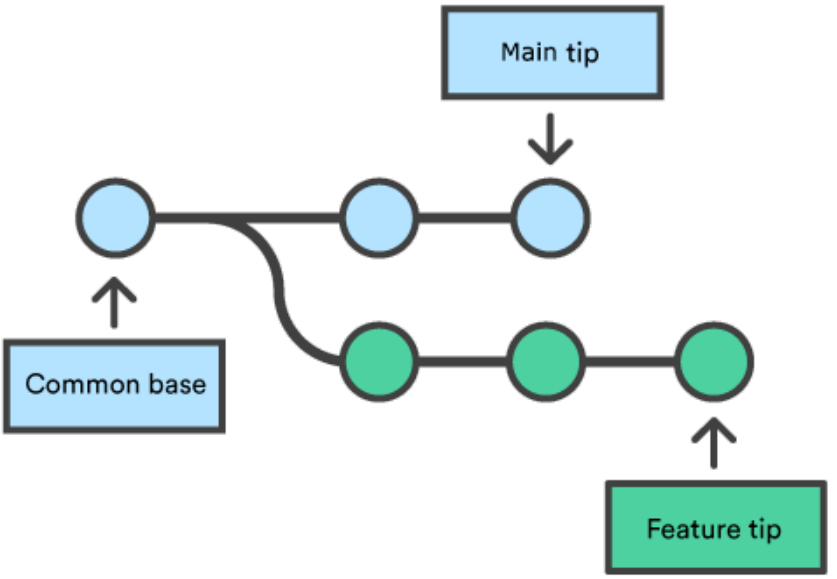
Commits:
Os commits são as alterações registradas. Cada commit é uma imagem instantânea do estado atual do projeto, que inclui as alterações feitas desde o último commit. Os commits são importantes porque permitem que os desenvolvedores trabalhem em diferentes partes do projeto sem afetar o trabalho dos outros desenvolvedores.
É ideal que cada commit tenha uma mensagem de commit associada que descreve as alterações feitas. A mensagem de commit deve ser descritiva e fornecer informações suficientes para outros desenvolvedores entenderem o que foi alterado no projeto. A mensagem de commit também é importante porque permite que você pesquise o histórico de alterações do projeto e encontre alterações específicas com mais facilidade.
Branch:
Branch é uma ramificação no fluxo de trabalho. Ao criar uma ramificação, é possível trabalhar em novas funcionalidades ou correções de bugs sem afetar o código principal do projeto. Isso permite que você teste novas funcionalidades e faça correções sem afetar o trabalho em andamento dos outros desenvolvedores.
Cada ramificação é independente, o que significa que as alterações feitas em uma ramificação não afetam outras ramificações ou o branch principal do projeto. Você pode criar uma nova ramificação a partir de uma ramificação existente ou da branch principal do projeto.

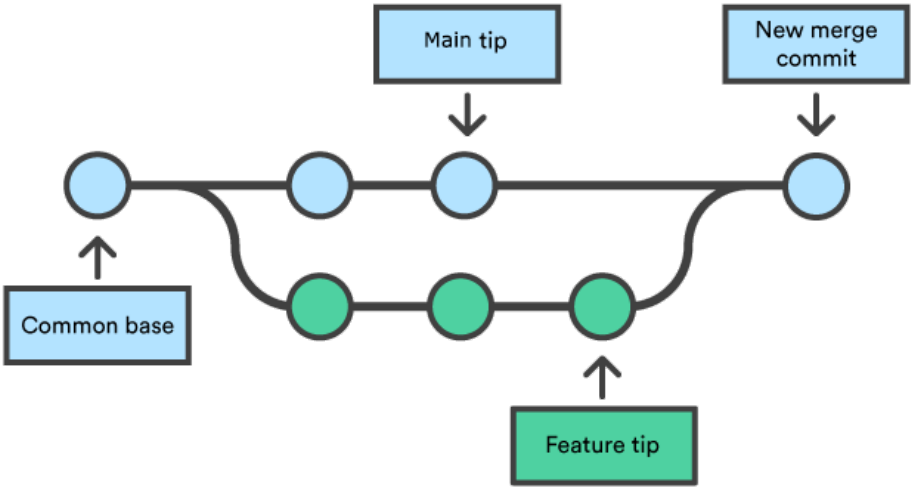
Main branch:

A Main Branch é a branch principal do projeto, também conhecida como branch "master" ou "trunk". Ela contém a versão mais atualizada do código do projeto e é a partir dela que outras ramificações são criadas. Por padrão, o Git cria automaticamente uma Main Branch quando um novo repositório é criado. Antigamente era muito usado o termo "master" como nome padrão da branch principal do Git, mas algumas plataformas como o próprio Git e o Github, por exemplo, têm optado por mudar o nome da branch padrão para "main".

Merge:
O merge é o processo de unir duas ou mais branches (ramificações) do Git em uma única branch. Ele permite que várias pessoas trabalhem em diferentes partes do projeto, sem que suas alterações conflitem entre si.
Ao finalizar as alterações em uma branch, é possível fazer o merge dessas mudanças na branch principal do projeto (também conhecida como main branch), o que pode ser feito de maneira automática ou manual. Isso significa que todas as alterações feitas em uma branch são integradas à main branch, formando uma nova versão do projeto.

Visualização desses conceitos:





Legenda das imagens:
Common base -> Base do projeto, início do projeto;
Main tip -> Main branch, ramificação principal, versão oficial do projeto;
Feature tip -> Branch do projeto, ramificação de desenvolvimento e testes;
New merge commit -> As novas aplicações desenvolvidas na branch(ramificação), ao serem testadas e válidas, sendo integradas na main branch(no projeto principal).

Conclusão

Concluindo, o Git e o GitHub são ferramentas essenciais para qualquer desenvolvedor que deseja trabalhar de maneira colaborativa e manter o controle sobre o versionamento do seu projeto.
O Git, por meio de sua estrutura de versionamento distribuído, permite que várias pessoas trabalhem em um mesmo projeto de forma organizada e eficiente, sem que haja perda de dados ou informações conflitantes.
Já o GitHub, por sua vez, fornece uma plataforma online para hospedar e compartilhar projetos Git, permitindo que os desenvolvedores possam trabalhar de maneira colaborativa de qualquer lugar do mundo. Além disso, o GitHub também fornece diversas funcionalidades adicionais, como integração com diferentes ferramentas, gerenciamento de problemas, e outras.

Peculiaridades do Github

- ReadMe: É padrão que todo projeto tenha um arquivo chamado ReadMe.md, md é o formato markdown, uma linguagem de marcação parecida com html porém mais simples. Ele serve para descrever o projeto, e sempre que tiver um arquivo chamado ReadMe.md na pasta de um projeto, tudo que tem nele será apresentado logo de cara ao abrir a pasta do projeto
- Capa do perfil: Se você criar um repositório com o mesmo nome do seu usuário (exatamente o mesmo nome, ele é case sensitive), tudo que você digitar no ReadMe.md desse repositório será apresentado na entrada do seu perfil, então esse repositório é usado como "capa" do seu perfil
- Porquê usar Git e não só o Github: É possível ir jogando os arquivos diretamente nos repositório pelo site do github, porém é muito difícil de organizar quando se trata de um projeto que envolve muitas pastas e arquivos, por isso o ideal é fazer isso em pastas no seu computador e mandar do git da sua máquina para o github, pois ele já vai com a mesma organização que estiver na sua máquina