**Git** - versionamento local do código / Software de Controle de Versão (VCS)

* Versionamento é a versão de alguma coisa, hoje lidamos com muitas \*versões\* de alguma coisa

Antes de termos softwares de controle de versão alguns programadores salvavam as versões do site em \*pastas zipadas\* (uma pasta que não pode ser alterada), e os nomes dessas pastas mudavam conforme uma atualização e correção fosse feita, até mesmo se um problema ocorresse. Bom mas aonde armazenar?

* no próprio desktop ou qualquer unidade de armazenamento física, que também possa levar a outros lugares;
* no google drive é uma opção muito melhor, pois aonde eu vou eu acesso o meus arquivos sem a necessidade de um pen-drive;

Nas duas situações tempos um grande problema, uma unidade de armazenamento físico pode ser corrompidos e assim perdemos tudo que foi salvo nela e par ao google drive seria muito confuso trabalhar em equipe, já que em muito pouco tempo ia ser uma bagunça muito grande de arquivos.

E para solucionar todos esses problemas surge os VCS (Software de Controle de Versão) que salvam qualquer nuance que for alterada no código sem que seja "zipado" algum arquivo, ele literalmente cria um histórico de tudo que foi alterado no arquivo para que possamos regredir a estados anteriores.

E temos dois tipos de VCS:

- **Centralizado/Linear**: toda de qualquer alteração que fizéssemos no software, e essa alteração é chamada de

**commit** era salva em um servidor local, e era obrigatório estar conectado a ele para conseguir alterar o código e para ver como ele estava.

- **Distribuído**: aqui conseguimos fazer um repositório local sem necessidade de ter que estar conectado ao servidor e quando chegamos ao ponto crucial do projeto aonde temos quer salva-lo para "ir ao ar" fazemos um **push** que sobe essas alterações para um servidor remoto, diferente do servidor local.

Principais vantagens de Software de Controle de Versão (VCS):

* Controle de Histórico, ou uma máquina do tempo do seu código;
* Trabalho em equipe, ajuda o trabalho ser feito no código por todos o membros da equipe;
* Ramificação do projeto, cria os ramos do projeto como o Front-end, Back-and e todos os outros ramos do projeto, e isso está muito ligado ao Trabalho em equipe;
* Segurança, assegura que nenhum outro membro da equipe mexa naquilo que não for dele, ou seja, quem for do Front-end não pode olhar e nem mexer naquilo que for Back-and;
* Organização, ajuda no ponto mais importante do nosso projeto;

**GitHub** - disponibilização dos códigos / Plataforma de Rede Social de Programadores

GitHub tem uma única diferença básica, enquanto Git gerencia o histórico do seu código localmente o GitHub faz isso em um servidor remoto que pode ser acessado de qualquer lugar.

Mas ainda assim há muitas outras funções:

* Repositórios Ilimitados: não há limite de quantidade de repositórios ou até mesmo de linguagem a ser salva. Somente há uma restrição de quantas pessoas pode trabalhar no mesmo repositório;
* Hospedagem de site: há um funcionalidade dentro do GitHub que conseguimos dar vida ao nosso projeto e mostrar funcionando para outras pessoas sem que tenha o trabalho de baixar o repositório, porém apenas sites simples com **HTML**, **CSS** e **JS** são hospedados na plataforma, sites mais complexos necessitam de uma hospedagem mais robusta;
* Características de Rede Social: aqui podemos interagir com outras pessoas, interagir com os projetos dessas pessoas também;

**Git e GitHub – Issues**

Esse tipo de utilidade para “issues”, é mesmo que um campo de discussão em um fórum. Aonde todos podemos interagir com outros usuários dentro de um repositório, mas somente o dono do repositório tem poderes para fechar uma issue e terminar a discussão.