Git Cheat Seet

Makson Vinicio

11/01/2021

O que e o Git?

Git é um sistema de versionamento, que permite registrar o histórico de edições de qualquer tipo de arquivo.

O que e o GitHub?

O GitHub é uma plataforma de hospedagem de código-fonte e arquivos integrada à ferramenta Git. Ele permite a integralização de múltiplos usuários a múltiplos projetos por meio de repositórios.

Links importantes:

- GitHub
- Download Git
- Download GitHub Desktop
- Meu GitHub
- Curso de Git Curso em Video

Cheats

Configure a ferramenta

Configura o nome que você quer ligado às suas transações de *commit*.

```
git config --global user.name "[nome]"
```

Configura o *email* que você quer ligado às suas transações de *commit*.

```
git config --global user.email "[endereco-de-email]"
```

Criando repositórios

Cria um novo repositório local com um nome especificado.

```
git init [nome-do-projeto]
```

Baixa um projeto e seu histórico de versão inteiro.

```
git clone [url]
```

Refatore nomes de arquivos

Remove o arquivo do diretório de trabalho e o prepara a remoção.

```
git rm [arquivo]
```

Remove o arquivo do controle de versão mas preserva o arquivo localmente.

```
git rm --cached [arquivo]
```

Muda o nome do arquivo e o prepara para o *commit*.

```
git mv [arquivo-original] [arquivo-renomeado]
```

Faça mudanças

Lista todos os arquivos novos ou modificados para serem commitados.

```
git status
```

Mostra diferenças no arquivo que ainda não foram preparadas.

```
git diff
```

Faz o *snapshot* de um arquivo na preparação para versionamento.

```
git add [arquivo]
```

Mostra a diferença entre arquivos preparados e suas últimas.

```
git diff --staged
```

Retira o arquivo da área de preparação, mas preserva seu conteúdo.

```
git reset [arquivo]
```

Grava o *snapshot* permanentemente do arquivo no histórico de versão.

```
git commit -m "[mensagem descritiva]"
```

Mudanças em grupo

Lista todos os *branches* locais no repositório atual

```
git branch
```

Cria um novo *branch*

```
git branch [nome-do-branch]
```

Muda para o *branch* especificado e atualiza o diretório de trabalho

```
git checkout [nome-do-branch]
```

Combina o histórico do *branch* especificado ao *branch* atual

```
git merge [nome-do-branch]
```

Exclui o *branch* especificado

```
git branch -d [nome-do-branch]
```

Revise o histórico

Lista o histórico de versões para o branch atual

```
git log
```

Mostra a diferença de conteúdo entre dois branches

```
git diff [primerio-branch]...[segundo-branch]
```

Retorna mudanças de metadata e conteúdo para o commit especificado

```
git show [commit]
```

Desfaça commits

Desfaz todos os commits depois de *[commit]*, preservando mudanças locais

```
git reset [commit]
```

Descarta todo histórico e mudanças para o commit especificado

```
git reset --hard [commit]
```

Sincronize mudanças

Baixe todo o histórico de um repositório remoto

```
git fetch [nome-remoto]
```

Combina o branch remoto ao branch local atual

```
git merge [nome-remoto]/[branch]
```

Envia todos os commits do branch local para o GitHub

```
git push [alias] [branch]
```

Baixa o histórico e incorpora as mudanças

```
git pull
```

Ordem, do projeto ao *commit*

1. Primeiro passo, criar um diretório
   - `mkdir <nome da pasta>` - Windows/Linux
2. Entrar na pasta
   - `dir <nome da pasta>` - Windows
   - `cd <nome da pasta>` - Linux
3. Iniciar o repositório
   - `git init`
4. Criar um arquivo para inicia o versionamento
   - Ex: `exemplo.py`
5. Adicionar o arquivo no versionamento
   - `git add <nome do arquivo>`
6. Dar o *commit*
   - `git commit -m <mensagem>`
7. Dar o *push*
   - `git push`