GitHub
FOLHETO DE AJUDA PARA GIT DO GITHUB

V 1.1.1

Git é um sistema de controlo de versão distribuído open source que facilita ações com o GitHub no seu portátil ou desktop. Este folheto de ajuda resume instruções frequentemente usadas na linha de comando do Git para referência rápida.

INSTALAR O GIT

O GitHub fornece clientes desktop que incluem um interface gráfico para as ações mais comuns. Estes clientes também incluem ferramentas da linha de comandos para cenários mais avançados. Em ambos os casos, os clientes são atualizados automáticamente.

GitHub para Windows
https://windows.github.com

GitHub para Mac
https://mac.github.com

Distribuições de Git para Linux e sistemas POSIX estão disponíveis no site oficial do Git SCM.

Git para todas as plataformas
http://git-scm.com

CONFIGURAR O GIT

Configurar os detalhes de utilizador ao nível do sistema:

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

Configura o nome que ficará associado aos Git commits

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

Configura o email que ficará associado aos Git commits

```
$ git config --global color.ui auto
```

Configura o email que você quer ligado as suas transações de commit

CRIAR REPOSITÓRIOS

Criar um novo repositório ou obter um já existente a partir do URL:

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

Cria um novo repositório local com o nome de projecto especificado

```
$ git clone [url]
```

Faz download de um projeto, incluindo toda a sua história

FAZER ALTERAÇÕES

Ver estado das alterações e criar um commit

```
$ git status
```

Lista todos os ficheiros modificados no projecto atual

```
$ git diff
```

Mostra as diferenças linha a linha dos ficheiros alterados (pre-stage)

```
$ git add [ficheiro]
```

Adiciona um ficheiro à área de preparação para que possa ser incluído em commits (staged)

```
$ git diff --staged
```

Mostra as diferenças linha a linha dos ficheiros preparados e alterados (pre-stage)

```
$ git reset [ficheiro]
```

Remove o ficheiro da área de preparação (staged) sem o remover do sistema de ficheiros

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

Guarda as alterações preparadas permanentemente no histórico de versões

BRANCHING (HISTÓRIA PARALELA)

Crie uma versão parelala do histórico de versões para trabalhar em isolamento.

```
$ git branch
```

Lista todos os branches locais no repositório atual

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

Cria um novo branch

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

Muda para o branch especificado e atualiza o diretório de trabalho

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

Combina a história do branch especificado com o branch atual

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

Remove o branch especificado

GITHUB FOLHA DE DICAS DE GIT

ALTERAÇÃO DE FICHEIROS
Mover e remover os ficheiros já versionados

```
$ git rm [ficheiro]
```
Marca o ficheiro para remoção do repositório local e remove o ficheiro do sistema de ficheiros (staged)

```
$ git rm --cached [ficheiro]
```
Remove o ficheiro do controlo de versão mas preserva-o no diretório de trabalho

```
$ git mv [ficheiro-original] [ficheiro-renomeado]
```
Muda o nome do ficheiro e o prepara-o para o commit

IGNORAR FICHEIROS
Ignore ficheiros e diretórios temporários

```
*.log
build/
temp-*
```
Um ficheiro `.gitignore` na raiz do projecto ignora o versionamento acidental de ficheiros e diretórios correspondentes aos padrões especificados

```
$ git ls-files --other --ignored --exclude-standard
```
Lista todos os ficheiros e directórios ignorados neste projeto

GUARDAR ALTERACÕES TEMPORARIAMENTE
Guarde e restaure mudanças temporárias

```
$ git stash
```
Armazena temporariamente todos os ficheiros modificados

```
$ git stash pop
```
Restaura os ficheiros previamente guardados na stash

```
$ git stash list
```
Lista todos os conjuntos de alterações na stash

```
$ git stash drop
```
Remove os últimos ficheiros guardados na stash

NAVEGAR O HISTÓRICO DE VERSÕES
Navegue e inspecione a história do projeto

```
$ git log
```
Lista o histórico de versões para o branch atual

```
$ git log --follow [ficheiro]
```
Lista o histórico de versões para um ficheiro, incluindo mudanças de nome

```
$ git diff [primeiro-branch]...[segundo-branch]
```
Mostra a diferença de conteúdo entre dois branches

```
$ git show [commit]
```
Mostra as mudanças de conteúdo e metadada do commit especificado

MANIPULAR A HISTÓRIA DE VERSÕES
Remove commits e refazer a história de versões

```
$ git reset [commit]
```
Aponta o ambiente de trabalho para o commit especificado, preservando alteracões locais e descartando o histórico de versões

```
$ git reset --hard [commit]
```
Aponta o ambiente de trabalho para o commit especificado, sem preserver alteracões e descartando o histórico de versões posteriores.

SINCRONIZAR ALTERACÕES
Actualiza e combina alteracões entre repositórios

```
$ git fetch [remote]
```
Faz download de todo o histórico de um repositório remoto

```
$ git merge [remote]/[branch]
```
Combina a história do branch especificado com o branch atual.

```
$ git push [remote] [branch]
```
Envia todos os commits do branch local para o GitHub

```
$ git pull
```
Download e merge num só comando. Igual a executar:
```
git fetch
git merge
```

GitHub Training

Aprenda mais sobre o uso do GitHub e do Git. Envie um email para a Equipe de Treinamentos ou visite nosso site para ver a agenda de eventos ou a disponibilidade de cursos particulares.

✉ **training@github.com**
training.github.com