O que é git? ↵℘

1. INDEX

---

2. CONTEÚDO

---

1. **Resumo do conteúdo:**
    1. O projeto github permite publicar documentos na nuvem de forma gratuita deste que o documento possa ser compartilhado com o público e o arquivo não seja maior que 50 megas.
    2. **PASSO A PASSO**:
        1. Instalar o git no desktop
        2. Criar um repositório no github
        3. Clonando repositório no github
        4. Crie o script **pushmain.sh** para enviar as alterações para o github
        5. Copie os arquivo do projeto para a pasta do repositório clonado.
        6. Execute o script **pushmain.sh**
        7. Verificando o status atual do projeto.
2. **Introdução**
    1. **Objetivo:**
        1. O git é um programa criado por **Linus Torvalds** cujo objetivo é controlar as versões de múltiplos documentos, podendo ser programas de computador

ou outro documento qualquer digital.

2. O objetivo principal é poder compartilhar as alterações criadas por vários programadores e um administrador fica responsável pelo merge de todas as versões para produzir uma versão única com todas as alterações.
3. [ ]

2. **Pre-requisitos:**
    1. Ter linux ou windows com uma versão acima de 2003 instalado..
    2. [ ]
3. **Benefícios:**
    1. Espero poder controlar as versões dos produtos que forem produzidos de hoje em diante, bem como voltar no tempo caso seja necessário.
    2. [ ]
4. **Desvantagens**.
    1. Não sei ainda....
    2. [ ]

3. **Conteúdo estudado**
    1. **Instalando git no linux e derivados do debian**
        1. Para instalar no linux distribuição baseada em Debian como o Ubuntu, Linux Mint, etc use o apt-get:.

            ```
            sudo apt-get install git-all
            ```

        2. Execute os seguintes comandos para fazer com que o git saiba seu nome e endereço de e-mail:

            ```
            git config --global user.name "Seu Nome"
            git config --global user.email "seu_email@qualquercoisa.com"
            ```

        3. O git funcionando no linux é preciso que os comandos abaixo sejam executado para indicar o término de linha nos arquivos textos:

            ```
            git config --global core.autocrlf input
            git config --global core.safecrlf warn
            ```

        4. [ ]
    2. **Criando repositório git**
        1. Use o comando **git init** para criar o repositório na pasta atual.
            1. Exemplo:

                ```
                git init
                #Initialized empty Git repository in
                ```

```
/Users/alex/Documents/Presentations/githowto/auto/hello/.git/
```

2. Use o comando **git init** para criar o repositório na pasta diferente da pasta atual.
    1. Exemplo

```
cd ~/meuProjeto
git init --separate-git-dir ~/meuProjeto.git
```

    3. **Referências:**
        1. Crie um repositório
        2. Como mover/separar a pasta .git da sua árvore de trabalho
4. [ ]

3. **Adicionando documento ao repositório local ou remoto se estiver configurado**
    1. Suponha você queira adicionar todo o conteúdo na pasta corrente *.* ao repositório então:
        1. Código shellscript

```
# Inicia um novo projeto git
git init

# Este comando pode ser executado várias vezes antes de um commit. Ele apenas adiciona o conteúdo do(s) arquivo(s) especificado(s) no momento em que o comando add é executado; se quiser que as alterações subsequentes sejam incluídas no próximo commit, você deve executar git add novamente para adicionar o novo conteúdo ao índice.

#Diga ao comando para preparar automaticamente os arquivos que foram modificados e excluídos, mas os novos arquivos sobre os quais você não informou o Git não são afetados.
git add --all ou git add *.* ou git add .

# Use o <msg> fornecido como a mensagem de confirmação.
git commit -m "Primeiro repositório da pasta local"

# Renomeie o branch  atual para main
# O comando branch -M não precisa ser feito a todo momento, porque o git sempre envia para
# o ultimo ramo selecionando.
git branch -M main
```

2. **Referências:**
    1. git init
    2. https://git-scm.com/docs/git/en
    3. git init
3. [ ]

4. **Listando o status do repositório**
    1. Use o comando **git status** para checar o estado atual do repositório.
        1. Código shell

```
cd /testGit # A pasta /testGit precisa ter executado o comando git init antes de saber o status

git status

# Resultado: On branch main nothing to commit (working directory clean)
```

    2. **Referências:**
        1. Status do repositório
    3. [ ]

5. **Adicionando modificações ao repositório**
    1. Altere qualquer coisa no arquivo **index.html**, em seguida execute os comandos abaixo:
        1. Código ShellScript

```
# Adicionando o arquivo modificado ao repositório
git add .

# Obtendo o status atual do repositório
git status

# Tornando a alteração definitiva:
git commit -a -m "Alterado o título de index.html"
```

    2. **Referências:**
        1. Adicionando modificações
        2. Exemplo de várias alterações e vários commits
    3. [ ]

6. **Ignorando arquivos no repositório git**
    1. Você pode criar um arquivo **.gitignore** no diretório raiz do seu repositório para informar ao **Git** quais arquivos e diretórios ignorar ao fazer um commit. Para compartilhar as regras de ignorar com outros usuários que clonam o repositório, envie o arquivo **.gitignore** para o seu repositório.

2. Exemplo de arquivo **.gitignore**.
    1. .gitignore

```
# Compiled source #
###################
*.com
*.class
*.dll
*.exe
*.o
*.so
*.ppu
*.bak
*.compiled
```

3. **Referências:**
    1. Configurando arquivos ignorados para um único repositório
    2. Uma coleção de **.gitignore** modelos
4. [ ]

7. **Configurando git para enviar para o github**
    1. Para conectar-se ao github.com é necessário criar um chave do protocolo SSH (Secure Shell Protocol), que fornece um canal seguro em uma rede não segura da seguinte forma:

```
# Gerando uma chave ssh na pasta ~/ssh
ssh-keygen -t ed25519 -C "seu_email@qualquercoisa.com"

# Adicionando sua chave SSH ao agente ssh
eval "$(ssh-agent -s)"

# Adicione sua chave privada SSH ao agente ssh.
# Se você criou sua chave com um nome diferente ou se está adicionando uma
# chave existente com um nome diferente, substitua id_ed25519 no comando
# pelo nome de seu arquivo de chave privada:

ssh-add ~/.ssh/id_ed25519
```

        1. NOTA:
            1. Se você estiver usando um sistema legado que não suporta o

algoritmo Ed25519, use:

```
ssh-keygen -t rsa -b 4096 -C "paulosspacheco"
```

2. Dependendo do seu ambiente, pode ser necessário usar um comando diferente.
    1. Por exemplo, pode ser necessário usar o acesso root executando **sudo -s -H** antes de iniciar o **ssh-agent** ou pode ser necessário usar exec **ssh-agent bash** ou **exec ssh-agent zsh** para executar o **ssh-agent**.
3. **ATENÇÃO**:
    1. Requisitos de autenticação de token para operações Git.
        1. Criação de um token de acesso pessoal
        2. Novo token de acesso pessoal

2. Entre no github para adicionar uma nova chave SSH à sua conta GitHub.
3. Criando repositório no github e configurando a máquina cliente:
    1. Suponha que o grupo seja **mi** e o projeto seja maricarai então:

```
   # criando uma pasta para os arquivos do git separado da
pasta corrente que se deseja versionar
   git init --separate-git-dir ../maricarai.git

   # Associa o repositório remoto ao repositório local.
   git remote add origin
git@github.com:paulosspacheco/maricarai.git

   # Cria um ramo para a versão
     git branch -M main

   # Adiciona todos os arquivo da pasta que se quer versionar
   git add .

   # Finaliza a transação do comando git add e registra o
nome das alterações feitas
   git commit -m "Descreva as alterações feitas"

   # Envia as alterações locais para o repositório remoto.
   git push -u origin main

   # Pull requests (Solicitações de pull)
```

    1. **ATENÇÃO**

        1. Lei sobre Pull Requests para finalizar o processo de atualização
        2. Como criar seu primeiro pull request no GitHub
    4. Verificando as chaves SSH existentes
        1. Digite **ls -al ~/.ssh** para ver se as chaves SSH existentes estão presentes.

```
$ ls -al ~/.ssh
# Lists the files in your .ssh directory, if they exist
```

        2. Verifique a lista de diretórios para ver se você já possui uma chave SSH pública. Por padrão, os nomes de arquivo das chaves públicas com suporte para o GitHub são um dos seguintes.
            1. ~/.ssh/id_rsa.pub
            2. ~/.ssh/id_ecdsa.pub
            3. ~/.ssh/id_ed25519.pub
        3. Copie a chave pública SSH gerada para a área de transferência.
        4. Acesse o site github e cole a chave que está na área de transferência.
        5. Comando para saber se a chave ssh está associada ao github:

```
git status
```

        6. ...
    5. **Referências:**
        1. get-started
        2. Instalando o Git
        3. Preparação
        4. Criando novo projeto no github
        5. Novo token de acesso pessoal
        6. Como trabalhar com Git e GitHub no VsCode | Tutorial
        7. VsCode - [Extensão GitLens], Histórico do GITHUB, no vscode, commit, Linha do código editado (#201)
        8. Verificando as chaves SSH existentes
        9. Configurando arquivos ignorados para todos os repositórios em seu computador.
    6. [ ]

8. **Clonando repositório no github**
    1. **Eureca!!!** a forma mais prática que encontrei para usar o github e vscode foi:
        1. No vscode instalei a extensão **GitLens — Git supercharged**
            1. Essa extensão habilita o vscode a reconhecer o repositório remoto.
        2. Entre no github e crie um repositório;
        3. Na pasta do projeto clone o repositório clonado com o seguinte

comando:

```
git clone git@github.com:paulosspacheco/maricarai.git
cd ./maricarai
code .
```

1. ..

4. Adicione os arquivos do projeto na pasta clonada.
5. Execute o script **pushmain.sh**
6. [ ]

9. **Verificando o status atual do projeto.**
    1. O git status comando exibe o estado do diretório de trabalho e da área de preparação. Ele permite que você veja quais alterações foram testadas, quais não foram e quais arquivos não estão sendo rastreados pelo Git. A saída de status não mostra nenhuma informação sobre o histórico do projeto confirmado. Para isso, você precisa usar o comando **git log**
        1. Exemplo de uso:

```
# Na pasta do repositório executar comando:
git status
```

    2. Exemplo do comando **status** atual do projeto..
    3. **Referências:**
        1. Git Status: inspecionando um repositório
    4. title
    5. [ ]
10. **Criando script para enviar as alterações para o github**
    1. Criar um arquivo **pushmain.sh** e cole os comandos a baixo, em seguida de permissão de execução ao arquivo **pushmain.sh**:

```
#!/bin/bash

# Esse é um parâmetro passado com a descrição do commit
TextoCommit="$1"

# Associa o repositório remoto ao repositório local.
   git remote add origin git@github.com:paulosspacheco/maricarai.git

# Renomeie o branch  atual para main
# O comando branch -M não precisa ser feito a todo momento, porque o
git sempre envia para
```

```
# o ultimo ramo selecionando.
   git branch -M main

# Este comando pode ser executado várias vezes antes de um commit.
   git add .

# Use o <msg> fornecido como a mensagem de confirmação.
   git commit -a -m "$TextoCommit"

# Envia as alterações locais para o repositório remoto.
   git push -u origin main

# imprime o status atual do repositório
git status
```

11. [ ]

4. **REFERÊNCIAS GLOBAIS**
    1. Site oficial para produzir este documento
    2. Guia básico de Markdown
    3. Vídeo aula sobre o github
    4. Git e GitHub - Instalação, Configuração e Primeiros Passos
    5. Voltando um commit do GIT
    6. Aprenda GitLab
    7. COMO USAR O GIT E O GITLAB NA PRÁTICA
    8. Markdown com sabor do GitLab
    9. [ ]
5. **HISTÓRICO**
    1. dd/mm/2021
        1. [ ]
    2. dd/mm/2021
        1. Criar este documento baseado no modelo03.md ;
        2. Escrever tópico Objetivos;
        3. Escrever tópico Pre-requisitos
        4. Escrever tópico Benefícios
        5. Escrever tópico desvantagens
        6. Escrever tópico Conteúdo
        7. Escrever tópico Exemplos
        8. Escrever tópico Referências

9. Atualizar o histórico deste documento.
10. Testar este documento depois após uma semana de concluído.
11. [ ]