Introdução ao Git
Comandos para exercício

Residência de Software
Iniciação à Tecnologia da Informação
Professor: Marlan Külberg

Atividade
Instalação

- Fazer uma instalação do Git para futura configuração

- Windows
    - https://gitforwindows.org/

- Linux
    - Via um instalador binário, através do gerenciamento de pacotes (packages), como o yum ou apt-get:

        - $ yum install git-core
        - $ apt-get install git

- Mac
    - https://git-scm.com/download/mac

Comandos iniciais Git

- Verificar versão (também útil para conferir instalação bem sucedida)
  - $ git version

- Configurar Identidade
  - $ git config --global user.name "John Doe"
  - $ git config --global user.email johndoe@example.com

- Verificar configurações
  - $ git config --list

- Verificar o valor que uma determinada chave no Git
  - git config <chave> (Ex.: $ git config user.name)

Ajuda

- 3 formas de comandos de ajuda
  - $ git help <comando> (Ex.: $ git help config)
  - $ git <comando> —help (Ex.: $ git config —help)
  - $ man git -<comando> (Ex.: $ man git -config)

Atividade
Explorar e configurar o Git

- Conferir versão
- Configurar identidade no git
- Explorar os comandos de configuração
- Explorar os comandos de ajuda (pode utilizar para explorar comandos de configuração)

Atividade
Criar working directory

- Criar uma pasta no seu sistema que será o diretório de repositórios do git (sugestão de nome: git)

- Criar uma pasta dentro do diretório de repositórios (sugestão de nome: gitBasico). Essa pasta será o primeiro repositório.

- Identificar o caminho no sistema de arquivos até a pasta de repositório

Comandos recorrente do Git

É muito importante conferir se o git está no diretório correto antes de efetuar qualquer operação.

- Configurar caminho do repositório no git
    - $ cd <caminho>

- Subir a hierarquia de pastas em 1 nível
    - $ cd ..

- Listar as páginas dentro do diretório
    - $ ls

- Conferir o status do repositório
    - $ git status

Atividade
Acessar e verificar repositório

- Acessar a pasta de repositório através do git usando os comandos de direcionamento

- Explorar a movimentação de pastas seja criando novas pastas no git ou através das pastas existentes no sistema

- Verificar o status da futura página repositório

Criar repositório

- Estando na pasta que será o repositório, usar o comando de iniciar como repositório
    - $ git init

Atividade
Iniciar repositório

- Crie o repositório através do comando init
- Crie um arquivo através de um editor de texto
- Transfira ou salve o arquivo no repositório
- Verificar o status do repositório

Adicionar arquivos (Stage Area)

- Transferir um arquivo do working directory para a Stage Area
    - $ git add <arquivo>

- Transferir todos os arquivos de uma mesma extensão do working directory para a Stage Area
    - $ git add *.<extensão>

- Transferir todos os arquivos do working directory para a Stage Area
    - $ git add .

- Para remover arquivos da Stage Area
    - $ git reset HEAD <nome_arquivo>

- Remover arquivo no Working Directory
    - $ git rm <nome_arquivo>

- Se um arquivo estiver na Stage Area mas foi alterado no Working Directory, ele vai constar como modificado na Stage Area.

Atividade
Enviar arquivos para Stage Area

- Enviar o arquivo criado através do comando add

- Criar vários arquivos com mesma extensão e de extensões diversas e testar as variações do comando add

- Remova arquivos da Stage Area através do comando reset

- Verificar o status de arquivo modificado, alterando um arquivo no Working Directory e usando o comando de status

- Remova um arquivo do Working Directory, use status para visualizar o resultado e corrija a inconsistência através do comando rm

Consolidar Arquivos (Commit)

- Fazer o commit dos arquivos na Stage Area
    - $ git commit -m “<descrição do commit>”

- Fazer o commit de arquivos modificados sem a necessidade de usar o comando add antes
    - $ git commit -a -m “<descrição do commit>”

- Cada commit ira gerar uma chave individual e única

- Descartar alterações do arquivo feitas no Working Directory que não é mais possível identificar
    - $ git checkout — <nome_arquivo>

Atividade
Commit

- Fazer o commit dos arquivos dentro da Stage Area

- Fazer modificação em um arquivo no Working Directory e consolidar este arquivo sem passar pelo comando add

- Fazer modificação em um arquivo no Working Directory e reverter as mudanças a partir do último commit através do comando checkout

Consultas no Git

- Consultar as alterações em arquivos que ainda não foram para a Stage Area
    - $ git diff

- Consultar as alterações em arquivos dentro da Stage Area, mas ainda não consolidadas
    - $ git diff —staged

- Partes removidas serão exibidas em vermelho e as adicionadas, em verde

- Históricos de todos os commits
    - $ git log

- Históricos de todos os commits, com as alterações (git log + git diff)
    - $ git log -p

- Limitar o número de entradas do log
    - $ git log -p -<número de entradas>

- Mostrar apenas a identificação e mensagem dos comias
    - $ git log —pretty=oneline

Atividade
Consultar alterações

- Usar os comandos diff para fazer as consultas nos arquivos

- Usar os comandos de log para visualizar os commits de diferentes formas

Reverter Informações (Editar o commit)

- Fazer a edição do arquivo na pasta
- Adicionar para a Stage Area (add)
- Fazer o commit com comando de edição
    - $ git commit -amend —m “<descrição do commit (edição)>”
- Altera a chave do commit mas não altera a quantidade de versões

Atividade
Edição de commit

- Editar um arquivo já consolidado e adicionar à Stage Area
- Usar o comando de edição de commit
- Verificar, através de um dos comandos de log, se o commit foi editado ao invés de ser criado um novo

Gitignore

- Utilizado em arquivos que não se deseja controlar o versionamento

- Criar um arquivo nomeado .gitignore (sem extensão)

- Através de um editor de texto, adicionar os nomes dos arquivos e pastas que não deseja versionar (1 arquivo por linha)

- Pode-se incluir inclusive o arquivo .gitignore

Atividade Gitignore

- Criar ou utilizar arquivos e pastas seguindo os passos para ignorar os commits

- Usar o comando de status para visualizar o resultado

Criar Servidor

- Criar uma pasta que será o repositório
- Configurá-la como compartilhável

- Criar repositório que pode ser usado em outros lugares
    - $ git init —bare

- Para uma estação conseguir acessar os dados do servidor, deve-se criar um clone da pasta do servidor
    - $ git clone file:/// <endereço/nome_servidor>

- Pode-se clonar os dados usando um outro nome
    - $ git clone file:/// <endereço/nome_servidor> <novo_nome>

Operações Servidor Remoto

- Pode-se trabalhar com esses arquivos como se fosse um git local

- As modificações serão apenas locais se não forem enviadas ao servidor

- Para enviar ao servidor, deve-se usar o comando push

- Para buscar dados novos do servidor usar o comando pula

Operações Servidor Remoto

- Para saber o nome do servidor remoto (o nome padrão é origin)
    - $ git remote

- Para enviar dados para o servidor
    - $ git push <nome_servidor> master

- Para buscar dados do servidor para a máquina local
    - $ git pull <nome_servidor> master

GitHub

- Serviço web que oferece diversas funcionalidades extras aplicadas ao git

- Pode-se usar gratuitamente o github para hospedar projetos pessoais.

- projetos/frameworks/bibliotecas sobre desenvolvimento open source estão no github

- https://github.com/

Atividade
Acessar Github

- Acessar o GitHub
- Criar uma conta no GitHub
- Criar novo repositório (com nome, escolher como público e inicializar o repositório com README)

Interagir com GitHub através do Git

- É necessário dar acesso ao computador local para interagir com a conta GitHub

- Para isso, em primeiro lugar é preciso gerar uma chave de autenticação ssh através do comando
    - $ ssh-keygen

- As chaves serão geradas em uma pasta chamada .ssh (geralmente na pasta do usuário, no sistema)

- Dentro da pasta .ssh, abrir o arquivo id_rsa.pub em um editor de texto e copiar o conteúdo

Interagir com GitHub através do Git

- No GitHub, em configurações de conta, acessar ssh keys e depois new ssh key
- Colocar uma identificação (ex.: meu notebook).
- Colar a chave e escolher add ssh key

Operações entre repositórios locais/repositórios GitHub

- As operações entre Git e Github seguem o mesmo modelo de comunicação entre máquina local e servidores remotos (comandos clone, push e pull)

- Na página do repositório, no GitHub, ir na aba security, depois em Quick Setup, e copiar o link da opção SSH.

- Para clonar o repositório do GitHub no Git local:
    - $ git clone <link_copiado> <nome_local (opcional)>

- O programa irá perguntar se deseja continuar a conexão. Digite yes

Operações entre repositórios locais/repositórios GitHub

- Para inserir arquivos, proceder da mesma forma que no modo local

- Para enviar arquivos e alterações para o GitHub
  - $ git push origin master

- Para receber arquivos e alterações do GitHub
  - $ git pull origin master

Atividade
Operações entre repositórios locais/repositórios GitHub

- Criar 2 clones do repositório GitHub com nomes diferentes, para simular a operação entre 2 máquinas locais

- Incluir um novo arquivo em 1 dos clones, commitar e enviar para o GitHub

- Verificar se o arquivo foi para o GitHub

- No segundo clone, fazer um pull do GitHub

- Verificar se recebeu o arquivo enviado pelo primeiro clone