CONTROLE DE VERSÕES POR GIT

Primeiros passos

1. Instalar git
1.1 Linux
- abrir um terminal e acionar o comando

sudo apt-get install git git-core

1.2 Windows

- Download git for Windows

https://git-for-windows.github.io/
(versao atual: Git-2.14.2)

- seguir as instruções para instalação
http://help.github.com/win-set-up-git/

- selecionar as seguintes opções:
- Use Git Bash only
- Checkout Windows-style, commit Unix-style line endings

1.3 Após instalar o git

- abrir terminal (no Linux) ou
acionar git bash (no Windows, a partir do Menu Iniciar)

- configurar o nome do usuário
para identificar futuras interações

$ git config --global user.name "Your Name Here"

- configurar o email do usuário
para identificar futuras interações

$ git config --global user.email "your_email@example.com"

- recomendável providenciar uma credencial para uso
principalmente para repositório HTTPS

http://blob.andrewnurse.net/gitcredentialwinstore/git-credential-winstore.exe

- recomendável providenciar chaves de identificação SSH

  https://help.github.com/articles/generating-ssh-keys

  a chave estará disponível no arquivo

  id_rsa.pub

  e poderá ser copiada usando-se um editor de texto.

- se desejável ignorar arquivos temporários (lixo)

  criar na pasta raiz do projeto arquivo com nome

  .gitignore

  e, usando um editor de textos,
  adicionar as extensões desses arquivos,
  um por linha:

  *.aux
  *.log

2. Para criar seu próprio repositório:
2.01. Abrir terminal ou
acionar git bash (no Windows, a partir do Menu Iniciar)

2.02. Navegar ate' a pasta raiz
(ou ~/ssh, no caso do Windows)

$ ~/ssh

2.03. Mostrar conteúdo da pasta atual

$ ls -lah

2.04. Criar uma pasta (por exemplo, myProject)

$ mkdir myProject

2.05. Mudar para a pasta criada

$ cd myProject

2.06. Preparar sua pasta para se tornar um repositório

$ git init

2.07. Recomendável criar um arquivo descritor

$ touch README.TXT

2.08. Abrir o arquivo criado em um editor de textos
e acrescentar sua identificação (nome, matricula)
e uma descrição do projeto

OBS.:
É possível configurar um editor padrão:

$ git config --global core.editor emacs

OBS.:
É possível configurar um comparador padrão:

$ git config --global merge.tool diff

OBS.:
Para conferir as configurações:

$ git config --list

OBS.:
Para obter informações sobre os comandos,
a qualquer momento:

$ git help comando
$ git comando --help
$ man git-comando

Por exemplo:

$ git help config

2.09. Adicionar o arquivo à sua área de provisórios

$ git add README.TXT

2.10. Submeter o arquivo para o repositório local ou
diretório de trabalho

$ git commit -m "adding readme"

OBS.:
Até esse ponto a interação é local.
Nada terá sido transferido para o repositório global.

3. Para trabalhar com o repositório global
3.1. Para a primeira interação:

$ git remote add origin user@alunos.pucmg/code:username/swd-drupal77.git

3.2. Para atualizar arquivos no repositório:

$ git push origin master

3.3. Para verificar as ultimas atualizações

$ git status

3.4. Se fizer alterações em arquivo (por exemplo, README.TXT)
adicionar a nova versão à área de provisórios

$ git add README

3.5. Identificar a modificação feita no repositório local

$ git commit -m README.TXT "add comment here"

3.6. Submeter as alterações para o repositório global
(mediante uso de sua senha)

$ git push origin master

OBS.:
Para acrescentar todos arquivos de uma vez

$ git add .

OBS.:
Para identificar vários arquivos de uma vez,
no repositório atual, recursivamente

$ git commit -m "upload all files"

OBS.:
Para acrescentar vários arquivos de uma vez,
na área de provisórios
(mediante uso de sua senha)

$ git push origin master

4. Para clonar diretório de um projeto

$ git clone alunos.pucmg/code/projeto.git

OBS.:
Para clonar com outro nome
(por exemplo, "matricula")

$ git clone alunos.pucmg/code/projeto.git matricula

OBS.:
Recomenda-se verificar o conteúdo

$ git status

5. Para remover arquivo(s) do diretório de trabalho

$ git rm arquivo
$ git rm *.tmp

6. Para renomear arquivo no diretório de trabalho

$ git mv arquivo novo

7. Para editar uma submissão recente
(por exemplo, por ter esquecido um arquivo)

$ git commit -m "1 - Primeiro envio"
$ git add esquecido
$ git commit --amend

8. Para retirar o último arquivo transferido para a área de provisórios

$ git reset HEAD ultimo

9. Para reverter um arquivo alterado
a uma versão anterior ainda no diretório de trabalho

$ git checkout -- arquivo

10. Para submeter sua versão do projeto para o repositório remoto (global)

$ git pull origin master

OBS.:
Para verificar o diretório remoto

$ git remote show origin

11. Para criar um variante (*branch*) (básico)

- verificar lista de variantes existentes

$ git branch

- verificar se há variantes ocultos

$ git branch -a

- criar variante (*branch*)

$ git checkout -b novobranch

- alterar entre *branches*:

$ git checkout novobranch

- retornar à raiz:

$ git checkout master

11. Para buscar cópia da versão do projeto no repositório remoto (global)

$ git fetch origin

12. Para remover uma referência remota

$ git remote rm arquivo

13. Para renomear uma referência remota

$ git remove rename arquivo_novo