Thursday, October 25, 2018

Lab 1 - Create your first Git repo and configuration

SETUP Steps: (**WARNING: if you already have a global config, be sure to have a current backup before the lab)**

```
git config --global --list
```

Should return an error saying the global config file does not exist.

Otherwise backup the current config file `.gitconfig` *(located in your user's home directory)*

LAB 1:

Goal: create a basic git configuration file

Note: all commands expect typical a `bash` shell environment.

1. Create a new directory for the first lab and navigate to that directory:

```
mkdir ./lab1/
cd ./lab1/
```

2. From that directory create your *`git`* repo from the CLI with *`git init`*

```
git init .
```

3. edit the local git config to verify https ssl when connecting to `github.com`:

```
git config --local http.https://github.com/*.sslVerify true
```

4. Check the file is created

```
git config --local --list
```

You should see a line with in the resulting output with the following:

```
http.https://github.com/*.sslVerify=true
```

For Example:

```
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
core.ignorecase=true
core.precomposeunicode=true
http.https://github.com/*.sslVerify=true
```

5. Edit your user settings in the global config: (This will edit your global .gitconfig)

5a. Configure your git user name: (replace FIRST LAST with your actual name)

```
git config --global user.name FIRST LAST
```

5b. Configure your git user email (for these labs please use the free github user email you setup as part of the perquisites as this will be used to authenticate your user in later labs)

```
git config --global user.email USERNAME@example.com
```

(Note to hide your email github allows you to hide your email in which case you'll have probably have a `@users.noreply.github.com` *style email)*

5c. Configure your git username (for these labs please use the free github user email you setup as part of the perquisites as this will be used to authenticate your user in later labs)

```
git config --global user.username USERNAME
```

5d. Configure your git editor:

```
(which open && git config --global core.editor open) || (which xdg-
open && git config --global core.editor xdg-open)
```

6. Now test the config by listing it:

```
echo "Global config" ;
git config --global --list
echo "Local Config" ;
git config --local --list
echo "system config (optional)" ;
git config --system
```

7. Now open the global .gitconfig and let's add some basic error checking (*Caveat: for repos over 2GB you might want to turn these off as they do add some overhead*)

7a. open the file `~/.gitconfig` in your default editor

```
open ~/.gitconfig || xdg-open ~/.gitconfig
```

7b. Now add the following template for use with the labs (feel free to customize these for use as a future template after the presentation). Template on following Page.

```
[core]
     protectNTFS = true
     protectHFS = true
     trustctime = false
     logAllRefUpdates = true
     warnAmbiguousRefs = true
     safecrlf = true

[status]
     showStash = true
     submoduleSummary = true

[diff]
     renames = copies
     renameLimit = 10
     algorithm = minimal

[fetch]
     recurseSubmodules = true

[transfer]
     fsckObjects = true

[gui]
     matchTrackingBranch = true
     pruneDuringFetch = true

[merge]
     log = 50
     renormalize = true
     branchdesc = true
     verbosity = 3

[notes]
     mergeStrategy = cat_sort_uniq
     rewriteMode = cat_sort_uniq

[rebase]
     stat = true

[rerere]
     enabled = true

[gpg]
     program = gpg2

[i18n]
     logOutputEncoding = utf-8
```

7c. Also add security settings for github.com (github supports https with tlsv1.2):

```
[http.https://github.com/*]
        sslVerify = true
        sslVersion = tlsv1.2
```