ben aston presents

please submit errata to ben@bj.ma

yet another git cheat sheet (using github)

```
set username                  git config --global user.name "<username>"

set email                     git config --global user.email <email-address>

make git case insensitive     git config --global core.ignorecase true

initialize local repo         git init

add a remote (use 'origin' for remote name)
                              git remote add <remote-name> git@github.com:<account-name>/<repo-name>.git

add single file               git add <filename>

add all files not in index    git add .

commit all staged changes     git commit -a

push local to remote & track  git push -u <remote-name> <branch-name>

clone repo for first time     git clone git@github.com:<account-name>/<repo-name>.git

switch branch                 git checkout <branch-name>

checkout and track a remote   git branch -f <local-name> <remote-name>/<branch-name>

pull from remote branch       pull git pull origin <branch-name>

push to remote branch         git push origin <branch-name>

delete remote branch          git push origin :<branch-name>

delete local branch           git branch -d <branch-name>

create branch                 git checkout -b <new-branch-name> <branch-to-branch-off>

rename branch                 git branch -m <old-branch-name> <new-branch-name>

update latest from remote     git remote update

unstage committed changes     git reset HEAD

unstage previous n commits    git reset HEAD~n

view staged changes           git status

lose changes to working copy  git reset --hard

stash changes                 git stash

pop stashed changes           git stash pop

clean everything (ignored, dirs, files)
                              git clean -xdf

who changed file contents     git blame <file>

view remote branches          git branch -r

revert changes to file        git checkout -- <file>
                              git checkout HEAD <file>

undo conflict resolution      git checkout -m <file>

un-stage single file commit   git reset HEAD <file>

cherry pick from a local branch (e.g. if committed to a wrong branch)
                              //get the sha of the commit
                              git checkout <correct-branch-name>
                              git cherry-pick <sha>
                              git push origin <correct-branch-name>
                              git checkout <incorrect-branch-name>
                              git reset -hard HEAD^
                              git push origin <incorrect-branch-name>

interactive add               git add -i
                              2 - update index with changes to existing files
                              4 - add untracked files according to selection

visualize index               git gui

visualize log/history         gitk

pull historical version of branch into another (e.g. known good integration into master)

                              git checkout <sha>                    //now in detached head state
                              git checkout -b <branch-name>         //branch based on <sha>
                              git checkout master                   //for example
                              git pull <temporary-branch-name>
                              git push origin master
                              git branch -d <temporary-branch-name>

add submodule                 git submodule add git@github.com:<account-name>/<repo-name>.git <location>

commit as amend to the previous commit
                              git commit -a --amend
                              //then "i" for interactive ":w" to save, and ":q" to quit

force local to track remote   git branch set-upstream origin/branch-to-track


differences between branches  git diff --name-status master..branch

list commits to branch        git log

check the result of a merge (before commit)
                              git pull <remote-name> //bring yourself up-to-date
                              git merge <branch-name> --no-commit --no-ff

prune knowledge of deleted remotes
                              git remote prune <remote-name> //e.g. 'origin'
compress history archive      git gc

find in files                 git grep -n <string-to-find>

tag a branch                  git tag -a <branch-name> -m "<message>"

"chop the head off a branch"
NOTE: this adds a commit with the tree at the state it was at the given sha. It doesn't actually chop the
head off.
                              //id the sha of the last good commit
                              git reset <sha>             //reset index to the desired tree
                              git reset --soft HEAD@{1}   //move branch pointer to previous HEAD
                              git commit -m "<message>"   //e.g. "revert to <sha>"
                              git reset --hard            //reset working copy to reflect new commit

add color                     git config color.ui true

add alias                     git config --global alias.myalias '<actual-command>'
```