Useful Commands

Which commit does this branch come from?

```
git merge-base new_branch old_branch
```

Which merged branches were not deleted?

```
for branch in `git branch -r --merged origin /master | grep -v HEAD`;do echo -e `git show --form at= "%ai %
r
```

How to amend a (non-pushed) commit?

```
git commit --amend --no-edit
```

How to clear local tags

```
git fetch --prune origin " +re fs/ tag s/* :re fs/ tag s/* "
```

How to display information about origin?

```
git remote show origin
```

How to setup the upstream of branch?

```
git checkout --track origin /br anc hname
```

How to reset local branch to remote's HEAD?

```
git reset --hard origin
```

How to check out more than one branch at a time?

```
git worktree
```

.gitconfig

diff & stash

How to include stage in diff?

```
git diff --staged
```

How to diff words instead of lines?

```
git diff --word -diff
```

What has changed in a file?

```
git diff HEAD~1..HEAD file.txt
```

How to diff a file with main?

```
git diff HEAD..main file.txt
```

How to diff a stash with the commit it's based on?

```
git diff stash@ {0}^!
```

How to see the most recent stash?

```
git stash show -p stash@{0}
```

Workflow

```
[core]
            exc lud esfile = ~/.git ignore
            aut ocrlf = false
            eol = lf
[include]
            path = ~/.git con fig -local
[fetch]
            prune = true
[push]
            aut oSe tup Remote = true
            default = upstream
[color " bra nch "]
            current = yellow reverse
            local = yellow
            remote = green
[color " dif f"]
            meta = yellow bold
            frag = magenta bold
            old = red bold
            new = green bold
[color " sta tus "]
            added = yellow
            changed = green
            unt racked = cyan
            rem ote Branch = cyan
[alias]
            lg = log --graph --oneline
            hist = lg --decorate --all
            st = status -sb
            co = checkout
            br = branch
            pushf = push --forc e-w ith -lease
            ver = describe --tags
            fuckup = commit --fixup HEAD
            praise = blame
```

```
git fetch
git checkout featur e/f eat ure-x
git commit -m "Add feature x"
git push
git checkout develop
git pull
git checkout featur e/f eat ure-x
git rebase develop
git add fixed files list
git rebase --continue
git checkout develop
git merge --no-ff featur e/f eat ure-x
git push
git branch -D featur e/f eat ure-x
git push origin :featu re/ fea ture-x
```

Log

```
git log --oneline
```

```
git log --stat
```

```
git log --name -status
```

```
git log --follow --pret ty= for mat :'%an (%ae)' file
niq
```