```
## git --learning --curve


#   difficulty
#
#               ----  nirvana
#              /    \
#             /      \
#            /        ------
#           | conflicts!?#  \
#           | rebase!#       \
#          /                  \
#         ? reset?!            -------
#        / undo?
#       / reset?!
#       | index?
#       | dag?
#   ----
#
#               time
#


# Disclaimer:
# - I'm not an expert
# - This should benefit beginners
# - This covers doubts I had and I was asked
#     however basic it sounds
# - Try to ask questions at the end
# - Try not to treat Git or SCM as secondary
```

```
## git log --gargantuan --intimidating --terrifying


# log has 100+ options
# git log man is -git

$ git log -git
          -tig
          -g -i -t
          -g -t -i
```

```
## consoles


git bash
posh-git
conemu
tortoisegit
...

# What I use?
git bash
scite text editor
winmerge2011 (diff, merge)
```

```
## configs


# system
# global
# local

# edit
$ git config --local -e &
$ git config --global -e &

# global config
# editor
# winmerge
# set some options in configs
# include path
```

```
## aliases


# The freedom to make my own ~~mistakes~~ aliases
# is all I ever wanted.
#              - Mance Rayder
#                Game of Thrones

# git commit
$ git c

# git checkout
$ git co

# git status -s|--short
$ git s

# log aliases
# git log --pretty=twoline --graph \
#   --decorate --abbrev-commit --date=relative
$ git l -1

$ git alias alias
$ git alias conf

# alias with ! run at root level
$ cd to/some/dir
$ git l .
```

## data structures

```
special refs: head, fetch_head
    branch

refs: branch, remote, tag:
    commit

commit:
    hash
    message
    authordate
    author
    committerdate
    committer
    parent|parents
    tree

tree:
    hash
    path
    tree[]
    blob[]
    blobpath[]

blob:
    hash
    binary
```

```
## DAG


#   root
#   o<---o<---A<---x master
#         \       /
#          `--B<-´
#             dev
#

# commits are shown as either of following:
# o
# *
# letters A, B, C ...

# merge commits are shown as:
# x
```

```
## workings


# work offline
# commit represents the whole repo at that point
# for diff, only the commits involved are required


#  o---A---o---o---B
```

```
## upstream


#           upstream
#
#                 --o---B origin/master
#                /
#               /
#   ---o---o---A origin/master
#                master
#
#           downstream

$ git fetch

#           upstream
#
#                         origin/master
#
#   ---o---o---A----o---B origin/master
#              master
#
#           downstream

(master)
$ git reset --hard master@{u} | origin/master
```