

# Git vs Subversion

Andrey Kotlarski

13.XII.2011

# Outline

Annoyances with our current source control

Can it get more comfortable?

Git

Appendix

# Rant

- Network traffic
- Hopefully we have good repository backup
- Clunky branch operations
  - svn copy URL1 URL2
  - svn checkout/switch URL2 (tea time)
  - uncomfortable for doing more than one quick fixes that need separate testing
- Merging to trunk?
  - conflicts for files we haven't touched
  - diff + clean up conflicts + patch
  - single commit, bye-bye branch history
  - branch is unusable afterwards, refinements go directly to trunk

# Distributed version control systems

- No central repository
  - No, this is not a limitation
- Each node is a repo on its own
  - no real need for backup
- Network traffic only when interacting with remote repos
- Most operations happen within own repo and are often instantaneous
- Allows alternative work-flows

bla bla

- That kernel git, Linus
- Known as (one of) the fastest
- Seems to scale well for big code bases

# Virtues

- A lot of advanced commands
  - maybe too many for a novice, but hey, you don't need to know them to boot
  - fine-grained control
  - of course you may shoot yourself in the foot
    - however, most things are re-fixable
- (local) Branches are really, really cheap
  - this is what version control is mainly after, right
- Allows the central repository model as well

# Quick start

- Please introduce yourself
  - `git config --global user.name "Andrey Kotlarski"`
  - `git config --global user.email akotlarski@vayant.com`
- initialize repository
  - `git init`
  - `git add .`
  - `git commit -m 'Initial commit.'`
- ...or clone existing one
  - `git clone URL`

# Revisions

- unique SHA1s, not numbers
- only initial characters can be used as long as uniquely identifiable
- HEAD, HEAD<sup>^</sup>, HEAD<sup>~</sup>5



# Branches and tags

- `git branch NAME`
- `git checkout BRANCH/REV/FILE`
- `git tag -a TAG`
- `git tag -l`

# Remote repos, branches and sharing

- `git checkout -track -b BRANCH`
- `git remote add URL`
  - don't switch to remote branch locally!
- `git pull`
- `git push origin`

# Merging

- git merge BRANCH
  - fast-forward merges
- git rebase
  - for brave history manipulators
  - don't do this for public visible commits, havoc ensues!
- git cherry-pick REV

# Get information

- `git log`
- `git status`
- `git show REV/TAG/REV:PATH-TO-FILE-OR-DIR`

# Recommendations

- avoid working in master
  - keep master clean and compilable, this is your face to the world normally
- use branches for whatever little or big changes
  - instantaneous creation
  - commit often
  - it can often be in experimental state (including commits), no problem
  - push important branches to the remote origin for backup

# Links

- `man git-*`
- official site
- Git - SVN crash course
- Successful git branching model