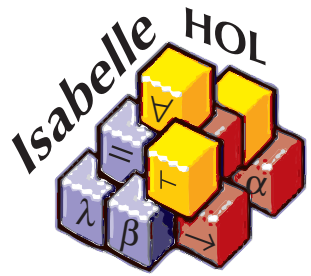
Mercurial and the Isabelle Repository

Alexander Krauss



Theorem Proving Group
Technische Universität München

Isabelle Developer's Workshop, 2009

You aren't losing functionality,
you are losing the illusion that you ever had anything reliable.

— Darren J. Moffat, OpenSolaris developer

Why CVS sucks

Merge fear

cvs: must merge before commit

hg: separates commit - merge - push/pull

Insiders/Outsiders

cvs: Need write access on central server to record history!!!

hg: No outsiders. Anyone can create / share changes

Centralism

cvs: Always need network, slow response

hg: Fully productive anywhere

Empty your mind!

For the rest of this tutorial:

Please **forget** all your cvs/svn knowledge!

Mercurial concepts

Repository: Project sources + history

- local, remote, everywhere
- directory (no big server/database)

  `hg init, hg clone SOURCE`

Changeset: Point in history

- immutable
- globally identified (`a8cb3af50308`)
- relations: parent, children, head, tip

  `commit, push, pull`

Working directory: Where you edit files

- associated to repository & changeset
- state: unchanged/changed

  `commit, update, merge`

Basic Commands

`init`	create empty repository
`clone`	produce identical copy
`update`	update working dir to specific changeset
`commit`	create changeset from working dir
`push,pull`	move changesets between repositories
`merge`	start a merge of two changesets (then: `commit`)
`resolve`	mark conflicts as resolved
`status`	inspect status of files in working dir
`diff`	show diff of uncommitted changes
`(g)log`	show changesets
`serve`	fancy web interface
`revert`	revert files in working dir to some revision
`help COMMAND`	the most important

Example

The Isabelle Repository

```
hg clone http://isabelle.in.tum.de/repos/isabelle
```

Answer questions like

- How old is this function?
- Who wrote this code (Whom can I ask?)

Contribute

- Make changes in your local clone
- Share your local clone (http, email)
- Community can easily review/test/discuss/pull your changes

see: README_REPOSITORY

Changing History

Important

This cannot work once changes have escaped “into the wild”

Repairing major accidents

- E.g., committed your online banking password
- `strip` (removes changesets)

Rebasing changes

- Avoids excessively long edges in the revision graph
- Various extensions, e.g. `mq`

Folding

- `mq`

Configuration

Configuration files

- `~/.hgrc`
- `/path/to/repo/.hg/hgrc`
- key=value format

Activate extensions

```
[extensions]
hgext.graphlog =
```

Set defaults

```
[defaults]
log = -l 5
```

Exercise

Create a child of changeset `A` which has the content of `A`, but with file `f` as in changeset `B`

```
hg up A
hg revert -r B f
hg ci -m "reverted f to B"
```

Thank you!