Merging branches and solving conflicts

Merge check-list:

1. `Commit.`
2. `bzr merge ...`
3. `Solve conflicts (if any).`
4. `Write merge changelog.`
5. `Commit.`

IMPORTANT

ALWAYS commit
just before *and* after merging!

Conflict check-list:

1. `File myfile has a conflict`
   ⟹ `myfile.BASE: before divergence`
   ⟹ `myfile.THIS: local version`
   ⟹ `myfile.OTHER: from other tree`
2. `kdiff3 myfile.BASE myfile.THIS`
   ↪ `myfile.OTHER # One line only`
3. `Save result as myfile.`
4. `bzr resolve myfile`
   ⟶ `removes files created at step 1!`

IMPORTANT

If *myfile* does not exist when typing `bzr resolved`, it will automatically be marked as *removed* by bzr.

Useful addresses

Abinit Forge
`bzr+ssh://archives.abinit.org/abinit/`
↪ `<developer>/<branch>/`
All of the URL should be typed at once (one line only, no space).

Abinit Website - Developer's Corner
`http://www.abinit.org/developers/`
Reference information for developers.

Abinit Forums
`http://forum.abinit.org/`
All new committers have to ask to be added to the "Committers" group.

Bazaar Website
`http://bazaar-vcs.org/`
Home page of the Bazaar Version Control System.

Mailing lists

Announcements
`announce@abinit.org`
Low-traffic list for important announcements about Abinit.

Committer List
`gnuarch@abinit.org`
For developers having an access to the Abinit Forge (restricted).

Need help?

To get help on all Bazaar commands, just remember `bzr help`.

ABINIT FORGE

Quick reference for committers

Copyright © 2008-2009 Abinit Group.
Written and maintained by Yann Pouillon.

Read this first

The Abinit website contains a section dedicated to Bazaar. Please read the documents located therein carefully (URL: one line, no space).

```
http://www.abinit.org/developers/
    ↪ dev-environment/bazaar/
```

Structure of the Abinit Forge

Full URL of a branch (one line, no space):

```
bzr+ssh://archives.abinit.org/abinit/
      ↪ <repository>/<branch>/
```

Repository name: *trunk*, or committer login.

Branch names:

- `x.y.z-private`
- `x.y.z-public`
- `x.y.z-training`

x: major version number (era)
y: minor version number (development cycle)
z: micro version number (patch level)

The *private* branch is where all developments go.
The *public* branch contains what is merged into *trunk*.
The *training* branch is there for learning, and is created for beginners and tutors only.

BEFORE ACCESSING THE FORGE

`export EDITOR="/path/to/my/editor"` (BASH)
in $HOME/.bashrc

or

`setenv EDITOR "/path/to/my/editor"` (CSH)
in $HOME/.cshrc

Useful commands

bzr ...	Result
`help`	Get help
`info`	Get source tree info
`status`	Get status report

Working with branches (recommended)

A *branch* is an autonomous copy of the source code. All history is kept locally, except when explicitely *published* by the committer.

Action	Command
Get	`bzr branch url` [`local_dir`]
Sync	`bzr pull` [`url`]
Publish	`bzr push` [`url`]
Off-line	`bzr commit`

Typical use: decentralized development.

If the working tree is empty, run `bzr checkout` from within. If it is inconsistent or outdated, run `bzr update` from within.

Public branches can and should only be synchronized with private ones by using the `~abinit/extras/bzr_helpers/bzr-publish` script from already-pushed private branches.

Working with checkouts

A *checkout* is a branch, the history of which is kept on the Forge. It can however be used off-line temporarily.

Action	Command
Get	`bzr checkout url` [`local_dir`]
Sync	`bzr update`
Publish	`bzr commit`
Off-line	`bzr commit --local`

Typical use: fast network connections.

Writing changelogs

Template:

```
One short summary line (no trailing dot)

* dir1/sub1/file1: Some changes. Make
full sentences.

* dir2/sub2/file2,dir3/sub3/file3: Some
other changes.
* dir4/sub4/file4: Related changes (no
blank line before).

* Additional notes and issues.
```

GNU changelog format: please make sure that all lines are < 80 characters and start at the first column. For a complete reference (one-line URL, no space):

```
http://www.gnu.org/prep/standards/
    ↪ html_node/Change-Logs.html
```

Committing

Check-list:

1. `bzr status`
2. `Process files marked as unknown.`
   `Go back to 1 until no file is marked as unknown.`
3. `Write changelog (see above).`
4. `bzr commit --strict` [`-F logfile`]
   `Use -F option if your changelog is in a file.`
5. `Push at least once a day.`

NOTE

Pushing to a public branch triggers heavy processes on various computers. Please use your private branch as much as possible.