NAME

baseline - distributed versioning control system

SYNOPSIS

baseline [**add** *[file or dir]*] [**branch -c** | **-l** | **-s**] [**cat -c**] [**commit -m**] [**diff**] [**help**] [**init**] [**log -c** | **-f** | **-n**] [**ls -c** | **-R**] [**version**]

DESCRIPTION

The **baseline** utility provides a command line interface to create or update a **baseline** repository. **baseline** is yet another open-source distributed versioning control system, modeled after git. **baseline** was made to be so simple following a suckless KISS design, and a clean KNF codebase. **baseline** is available under the terms of ISC, a permissive free software licence.

ENVIRONMENT

EDITOR

when **baseline** needs to launch a text-editor and the default editor is not specified in the repository configuration file, **baseline** will then look if the EDITOR environmental variable is set and or not, and if it is set **baseline** will attempt to use it.

FILES

.baseline/config

This file contains the configuration options for a **baseline** repository.

EXIT STATUS

The **baseline** utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

To find out which version of baseline is installed:

```
$ baseline version
```

To create a new repository in the current working directory:

```
$ baseline init
```

To add a specific file or directory to your staging area:

```
$ baseline add <filename>
```

Or, to add all files and directories to your staging area:

```
$ baseline add .
```

To commit your staged changes:

```
$ baseline commit -m 'my commit message'
```

If the -m flag was omitted, baseline will:

- first, search your .baseline/config, for a variable called ‘‘editor’’.
- then, will try the EDITOR environmental variable.
- otherwise, baseline will complain about missing commit message.

To display a diff or generate a patch between a commit and its parent:

```
$ baseline diff <commit id>
```

To display a diff between any two commits:

```
$ baseline diff <commit A id> <commit B id>
```

To list all commits:

```
$ baseline log
```

To list all commits starting from a specific commit:

```
$ baseline log -c <commit id>
```

To limit the number of commits being displayed:

```
$ baseline log -n <number of commits>
```

To format the output of log command:

```
$ baseline log -f <format>
```

Currently the log command supports the following specifiers for each commit:

- %n: a number representing the order of the commit in the log list.
- %an: the author’s name.
- %ae: the author’s email address.
- %at: the author’s commit timestamp.
- %cn: the committer’s name.
- %ce: the committer’s email address.
- %ct: the committer’s commit timestamp.
- %m: the commit’s message.
- \n: a new line.
- \t: a tab space.
- Any other character will be displayed as it is.

To list files and directories:

```
$ baseline ls
```

To recursively list files and directories:

```
$ baseline ls -R
```

To list file and directories for a specific commit:

```
$ baseline -c <commit id>
```

To get the content of a certain file written to the stdout:

```
$ baseline cat </path/to/file>
```

You can easily redirect the output to any other file:

$ baseline cat </path/to/file> > myfile.txt

To get the content of a certain file within a certain commit:

$ baseline cat -c <commit id> </path/to/file>

To find the name of the current branch:

$ baseline branch

To list all the available branches:

$ baseline branch -l

To create a new branch from the current branch:

$ baseline branch -c <branch name>

To switch branches:

$ baseline branch -s <branch name>

AUTHORS

The **baseline** utility was written by Mohamed Aslan <*maslan [at] sce.carleton.ca*>.

BUGS

baseline is currently in pre-alpha status, thus far from having any usefulness. That means you can play with it at your own risk.