fcd

(F)ast (C)hange (D)irectory for bash shells avoiding a lot of aliases.

Background:

Tired of adding aliases for frequently visited directories?
Tired of searching through your aliases or other configuration file to find out how you named that smart shortcut?

fcd is a solution for that. The benefit of **fcd** is to replace tedious aliases which are cumbersome to create, update and delete when you would like to create shortcuts to frequent visited directories.
Out of own experience I never create an alias until it starts to become boring to repeat the command over and over again. When its done the list of aliases will continuously grow and never get cleaned up.

fcd behaves similar to a `cd <path>` alias and it will also support adding extra commands after the change directory have taken place to be able to tailor the behavior. If you have forgotten the shortcut name you created for a specific alias the script supports a listing function of all created aliases.

Installation / setup:

You have two options:

Option 1: Run the provided `install.sh` script which will create needed bin directory in the users home if it doesn't exists, copying all needed files to the bin, updating the `$PATH` environment if needed and finally creating some supporting shortcut aliases. Logout & login and you are ready to go.

Option 2: Do it manually and this is just a proposal of setup and you can of course tailor it according to your own preferences.

1. If you don't have a bin directory in your `$HOME` directory, create one:
   `mkdir ~/bin`

2. Check if `~/bin` is included in your `$PATH` environment:
   `echo $PATH`

3. If it's not, include the path to ~/bin into your `$PATH` environment. It might vary which configuration file you have to update dependent on your linux distribution and setup. When Bash is invoked as an interactive login shell, or as a non-interactive shell with the `--login` option, it first reads and executes commands from the file `/etc/profile`, if that file exists.
   After reading that file, it looks for `~/.bash_profile`, `~/.bash_login`, and `~/.profile` in that order. Example: `PATH="$HOME/bin:$PATH"`

4. Copy fcd.pl, fcd.sh, README and LICENSE to your ~/bin directory.
5.

The scripts needs to be executable and if needed change the execution flags
`chmod u+x ~/bin/fcd.*`

6. Create a few new aliases to make the use of the **fcd** script more efficient. Update or create the file ~/.bash_alias in your home directory and open it with your favorite editor and add the following lines:

```
alias ++='fcd.pl -w "$@"'
alias fcdrm='fcd.pl -d "$@"'
alias g='source ~/bin/fcd.sh "$@"'
```

Note! Why g as an alias? Simple, g for (g)oto ;)

Done, logout and login again to make sure all path's and aliases are set correctly.

Using other shells than bash?

I haven't tested **fcd** with other shells than bash and tcsh. So if you are using tcsh as a shell environment you can copy the `fcd.csh` file to your `~/bin` directory and use the following aliases instead:

```
alias ++ 'fcd.pl -w \!*'
alias -- 'fcd.pl -d \!*'
alias g 'source ~/bin/fcd.csh \!*'
```

Why different aliases for the delete command?
Haven't figured out how to make -- a valid name in bash since the command alias takes options (-p or -a) before the name and -- always seems to be evaluated as an invalid -- option.

Ready to peek and poke around.

Examples (bash style):

```
$ ++            To add current dir to my list without any shortcut tag.
$ ++ mytag      To add current dir to my list with a tag.
$ g mytag       To change directory to a entry based on tag.
$ g             You will be presented with a list of all entries you have saved
                and you could select one from the list based on list number or
                tag.
$ fcdrm mytag   To delete a entry with a known tag.
> fcdrm         To delete a entry but you are unsure about tag or if no tag
                exists, you can select from the list.
```

If you want to add an extra command after the script have changed directory for you. For example you might want to have a **`ls - la`** of the directory this can be done with the command:
`g -c`

And you will be prompted to add this with:
select entry (number) to add or replace a command to:

For more information type:
`g --help` or `g -h`

Hope you will enjoy using **fcd** and it will save some precious time for you.