Tutorial

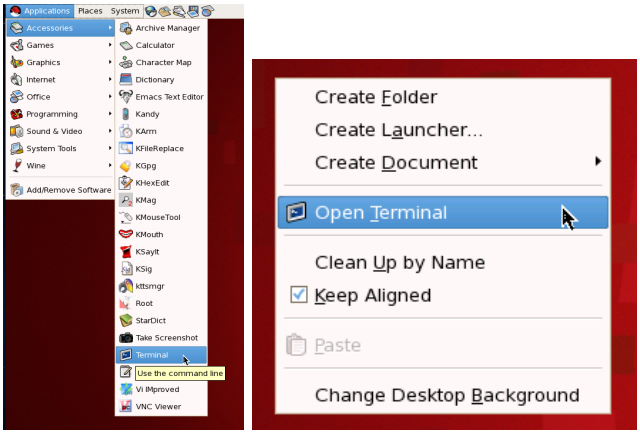
In-class tutorial for Day 1.

GOALS:

1. Log in
2. Open a terminal window
3. Open IDL
4. Execute an IDL statement
5. Start an IDL journal
6. Rename a file

Slightly more detail

1. Log in using your CU identikey.
2. Open a terminal.



3. Type the command `IDL` at the command prompt and hit `enter`
4. Enter the command `print,"hello"` at the `IDL>` command prompt
5. Enter the command `journal` at the `IDL>` command prompt From now on, any command you type will be logged in a file called `idlsave.pro`.
6. Type some other command - `print`, `help`, or anything you'd like
7. Enter the command `journal` (again) at the `IDL>` command prompt
8. Type `exit` and press `enter` to leave IDL
9. Enter the command `ls` (short for "list" or "listing") to list the contents of your home directory. You should see a file called `idlsave.pro`
10. Rename the file `idlsave.pro` to "YourName_Today'sDate" `.pro` (e.g, I would use `AdamGinsburg_2012_08_27.pro`) using the `mv` command: `mv idlsave.pro AdamGinsburg_2012_08_27.pro`

Tutorial: Version Control

Almost all programmers outside of the sciences do their programming collaboratively. Recently, the same has become true of programmers within science as well.

Collaborative programming is when you allow others to help you with your code, or as a group you combine your efforts to write one big code instead of many separate ones.

The simplest way to do this is for each individual to write their own code, pass it around, and have colleagues write down their own comments and suggestions on how to change or improve the code. While simple, this is horribly inefficient: what if the only change your code needed was the addition of a semicolon where you accidentally left one out? There are many other reasons this method isn't great (but it is still the preferred method of editing papers).

There is a powerful unix tool called `diff` that allows you to directly compare files. The `diff` command goes through a pair of text files line-by-line identifying differences on each line, but ignoring lines that are the same.

There is a whole huge framework of tools built on top of `diff`. We're going to skip over a lot of the why and how for those so we can get to more immediately useful stuff.

git

`git` is a "distributed version control system". Those words contain a lot of layered meaning, but for now you need to know that `git`:

1. is a way to keep track of edits to your code
2. provides local (and remote) backups of your code
3. is a great way to share code

There are other similar codes: `hg` (mercurial) is basically the same thing, while `svn` (subversion) is a little different but similar. We're using `git` simply because `github` gave us an educational account.

You're going to make a 'git' account for your own use and for this class.

Go to http://github.com/signup/free.

It will ask you for a username, e-mail address, and password. For your username, pick anything you'd like (but be aware, it will be visible to the internet, including me). For your e-mail address, use your `@colorado.edu` address - this is important, as it will allow you access to some free `.edu` features that are not otherwise available.

E-mail your account name to Adam (astr2600@gmail.com).

At this point, you should be logged in to github. Go to https://github.com/edu and click 'I'm a student'. You should see the text "Before you can request a student plan, you must add and verify your school-issued email address." Click the "add and verify" link - it should take you to https://github.com/settings/emails, where you should now click the Verify button.

You will receive an e-mail sometime soon confirming that you are indeed a student and giving you free access to private repositories (we'll talk about what those are). Follow the instructions in that e-mail.

Summary:

1. Sign up for github
2. Send your account name to astr2600@gmail.com
3. Verify your e-mail address