CONDA CHEAT SHEET

Command line package and environment manager

Learn to use conda in 30 minutes at **bit.ly/tryconda**

TIP: Anaconda Navigator is a graphical interface to use conda. Double-click the Navigator icon on your desktop or in a Terminal or at the Anaconda prompt, type `anaconda-navigator`

Conda basics

Verify conda is installed, check version number	`conda info`
Update conda to the current version	`conda update conda`
Install a package included in Anaconda	`conda install PACKAGENAME`
Run a package after install, example Spyder*	`spyder`
Update any installed program	`conda update PACKAGENAME`
Command line help	`COMMANDNAME --help` `conda install --help`

*Must be installed and have a deployable command, usually PACKAGENAME

Using environments

Create a new environment named py35, install Python 3.5	`conda create --name py35 python=3.5`
Activate the new environment to use it	`WINDOWS:  activate py35` `LINUX, macOS: source activate py35`
Get a list of all my environments, active environment is shown with *	`conda env list`
Make exact copy of an environment	`conda create --clone py35 --name py35-2`
List all packages and versions installed in active environment	`conda list`
List the history of each change to the current environment	`conda list --revisions`
Restore environment to a previous revision	`conda install --revision 2`
Save environment to a text file	`conda list --explicit > bio-env.txt`
Delete an environment and everything in it	`conda env remove --name bio-env`
Deactivate the current environment	`WINDOWS: deactivate` `macOS, LINUX: source deactivate`
Create environment from a text file	`conda env create --file bio-env.txt`
Stack commands: create a new environment, name it bio-env and install the biopython package	`conda create --name bio-env biopython`

Finding conda packages

Use conda to search for a package	`conda search PACKAGENAME`
See list of all packages in Anaconda	`https://docs.anaconda.com/anaconda/packages/pkg-docs`

ANACONDA.

CONTINUED ON BACK ➜

Installing and updating packages

Install a new package (Jupyter Notebook) in the active environment	`conda install jupyter`
Run an installed package (Jupyter Notebook)	`jupyter-notebook`
Install a new package (toolz) in a different environment (bio-env)	`conda install --name bio-env toolz`
Update a package in the current environment	`conda update scikit-learn`
Install a package (boltons) from a specific channel (conda-forge)	`conda install --channel conda-forge boltons`
Install a package directly from PyPI into the current active environment using pip	`pip install boltons`
Remove one or more packages (toolz, boltons) from a specific environment (bio-env)	`conda remove --name bio-env toolz boltons`

Managing multiple versions of Python

Install different version of Python in a new environment named py34	`conda create --name py34 python=3.4`
Switch to the new environment that has a different version of Python	`Windows: activate py34` `Linux, macOS: source activate py34`
Show the locations of all versions of Python that are currently in the path **NOTE**: The first version of Python in the list will be executed.	`Windows: where python` `Linux, macOS: which -a python`
Show version information for the current active Python	`python --version`

Specifying version numbers

Ways to specify a package version number for use with conda create or conda install commands, and in meta.yaml files.

Constraint type	Specification	Result
Fuzzy	`numpy=1.11`	1.11.0, 1.11.1, 1.11.2, 1.11.18 etc.
Exact	`numpy==1.11`	1.11.0
Greater than or equal to	`"numpy>=1.11"`	1.11.0 or higher
OR	`"numpy=1.11.1|1.11.3"`	1.11.1, 1.11.3
AND	`"numpy>=1.8,<2"`	1.8, 1.9, not 2.0

NOTE: Quotation marks must be used when your specification contains a space or any of these characters: > < | *

MORE RESOURCES

Free Community Support	groups.google.com/a/continuum.io/forum/#!forum/conda
Online Documentation	conda.io/docs
Command Reference	conda.io/docs/commands
Paid Support Options	anaconda.com/support
Anaconda Onsite Training Courses	anaconda.com/training
Anaconda Consulting Services	anaconda.com/consulting

Follow us on Twitter @anacondainc and join the #AnacondaCrew!
Connect with other talented, like-minded data scientists and developers while contributing to the open source movement. Visit anaconda.com/community

Cheatography

Tmux basics Cheat Sheet

by TheCultOfKaos via cheatography.com/22503/cs/4612/

General Controls

ctrl+b	Default Bind Key
bind-key ?	Lists bind-key combinations
bind-key :	Enter config options directly for current session

Tmux Sessions

tmux	start tmux and attach to default session
tmux ls	prints a list of existing tmux sessions
tmux new -s <name>	Create a new named tmux session
tmux a -t <name>	Attach to a named tmux session
tmux kill-session -t <name>	Kill a session when you're done with it
bind-key, $	(Re)Name a session

256 Color support

Add `alias tmux="TERM=screen-256color-bce tmux"` to your ~/.bash_profile and `set -g default-terminal "xterm"` to ~/.tmux.conf

Other useful config file tricks

set-option -g pane-active-border-fg <color>	Surround the active pane with a specific color for easier identification
bind "" split-window -c "#{pane_current_path}"	When splitting a pane horizontally, create the new pane from the current directory
bind % split-window -h -c "#{pane_current_path}"	When splitting a pane vertically, create the new pane from the current directory
bind c new-window -c "#{pane_current_path}"	When creating a new window , create from the current directory.
bind-key R source-file ~/.tmux.conf; display-message "~/.tmux.conf is reloaded"	Allows you to **bind-key R** to reload your tmux config for the current session
setw -g monitor-activity on	Allows tmux to monitor for command/process exits
set -g visual-activity on	When a command exits in a non-active window, visually change the tab list to reflect that

These go in ~/.tmux.conf

Windows / Tabs -

bind-key c	**C**reate a new window/tab
bind-key w	List **w**indows/tabs (helps on smaller screens)
bind-key ,	(Re)Name a window/tab
bind-key &	Kill current window (confirmation req)
bind-key <number>	Go directly to numbered window
bind-key l	**L**ast Active window/tab
bind-key n	**N**ext window/tab
bind-key	**P**revious window

In the status bar (bottom) are the numbers/names of windows/tabs.

Panes (vertical/horizontal splits)

bind-key %	Split current pane vertically
bind-key "	Split current pane horizontally
bind-key q	Show numeric pane values (red is active)
bind-key o	Cycle through panes
bind-key x	Kill current pane (req confirmation)
bind-key !	Close all panes except current (req conf)
bind-key +	Break pane into new tab/window
bind-key -	Restore pane from new tab to old tab
bind-key <arrow keys>	Navigate around panes

There are many other key-bindings for resizing panes that are worth learning, but out of scope for a basic cheatsheet. Using the mouse modes for resizing is also helpful.

Mouse Support

setw -g mode-mouse on	enable mouse integration
setw -g mouse-select-pane on	use the mouse to select panes
setw -g mouse-resize-pane on	use the mouse to resize panes
setw -g mouse-select-window on	use the mouse to select windows/tabs

These go in ~/.tmux.conf

By **TheCultOfKaos**
cheatography.com/thecultofkaos/

Published 16th July, 2015.
Last updated 16th July, 2015.
Page 1 of 1.