# How to Manage and Restore Tmux Sessions in Linux

**tmux cheatsheet:**

**https://gist.github.com/henrik/1967800**

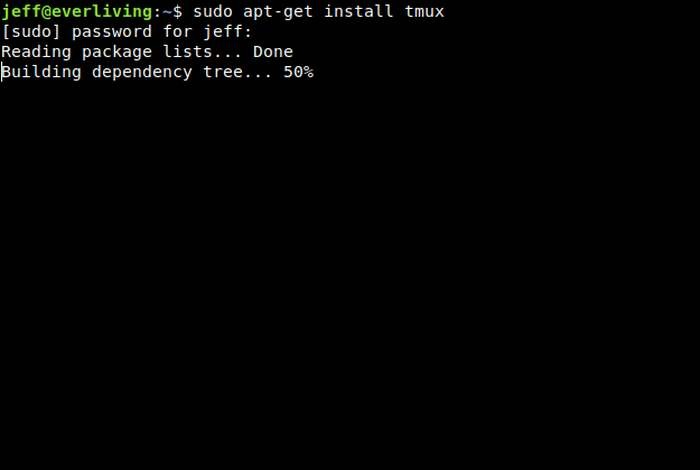
Tmux is a terminal multiplexer that comes stocked with a wide range of useful features and is backed by a surprising number of community-made plugins. Terminal multiplexers like tmux and Screen give your terminal window super productivity powers, allowing you to open multiple sessions in tabs and split screens. Combined with multiple terminal tools and session saving, you can also restore Tmux sessions after a hard reboot.

**Installing tmux**

Getting tmux ready and running on your system is relatively simple if you are using a Linux distribution.

For Ubuntu, this means summoning apt from your terminal and installing the package from the distribution’s official repository. For other distros, use your included package manager to handle the installation. (tmux is available on most distributions.)

**sudo** **apt-get install** tmux



Got it? Great! Now, let’s split our screen.

**Creating Windows and Panes**

In tmux, two terms are used to describe the main types of layout configurations we can create. “Windows” are the tmux term for tabs. Creating a new window will make a tab that you can switch over to with a simple command.

“Panes,” on the other hand, are splits in the current “Window” or tab that can also be switched to using a command.

Before we begin creating these, it is important to note that tmux is modal in function. What this means is that interactions with your terminal session and with tmux happen in separate “modes.” This is useful as you can use each individual terminal session you open normally without accidentally activating a tmux command.

To gain access to tmux and begin issuing commands, we’ll first start our new tmux session and name it:

tmux new -s babytmux

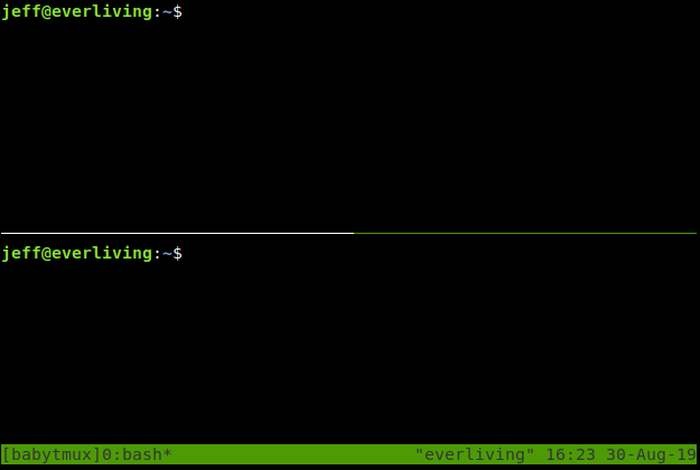


Feel free to change the name (“babytmux”), of course!

Tmux should open immediately, and a new shell session should start up for you. But we want more than one, remember?

To enter tmux’s command mode, we’ll need to use the prefix. This is usually Ctrl and b pressed simultaneously. Alone, you won’t notice anything changing by pressing the prefix, but we can type out commands by entering: immediately after. To leave command mode, either complete the command by pressing Enter or press the ESC key to exit without making changes.

We’ll use a shortcut to create a horizontal split: Press Ctrl and b, then "

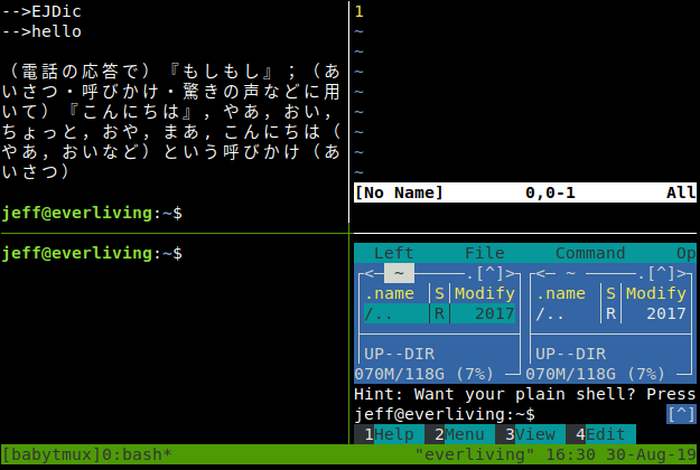


Cool! Now, we can navigate between the two with the following: Ctrl and b, then o.



Each session is independent of the other, so we can open separate programs in each. Let’s quarter our screen with vertical splits and try it out: Ctrl and b, then %

Open whatever you want in each pane, and it will keep running while you access the others.



To make a window, use the following: Ctrl and b, then c

You can switch back to a previous window or move forward one window with:

Ctrl and b, then p

or

Ctrl and b, then n



The status bar at the bottom of the screen shows you a process that is running in each window and which window you are on (with “\*”) for reference.

To leave this tmux session, use this: Ctrl and b, then type :detach-client



Your session will not stop running unless your machine is rebooted or you manually cancel it. To access it again, use this:

tmux a -t babytmux

To create truly immortal tmux sessions that can come back after a reboot, we need to install a plugin or two.

**Installing Plugins**

Using plugins in tmux is relatively straightforward. However, there is a plugin manager we can install to make subsequent plugin installations easier.

**Plugin Manager**

To install [Tmux Plugin Manager](https://github.com/tmux-plugins/tpm), we will clone its files from Github with the following code:

**git clone** https:**//**github.com**/**tmux-plugins**/**tpm ~**/**.tmux**/**plugins**/**tpm

Next, add the following code to “.tmux.conf” in your home folder:

*# List of plugins*

**set** -g **@**plugin 'tmux-plugins/tpm'

**set** -g **@**plugin 'tmux-plugins/tmux-sensible'

*# Other examples:*

*# set -g @plugin 'github\_username/plugin\_name'*

*# set -g @plugin 'git@github.com/user/plugin'*

*# set -g @plugin 'git@bitbucket.com/user/plugin'*

*# Initialize TMUX plugin manager (keep this line at the very bottom of tmux.conf)*

run -b '~/.tmux/plugins/tpm/tpm'



As you can see above, we will need to add the github username and plugin name (found in a plugin’s github URL) for each plugin we want to install from now on. The plugin manager will handle the rest. Use this code to get the plugin manager working:

tmux **source** ~**/**.tmux.conf

Now, for the plugin we need to restore sessions after rebooting, keep reading.

**Resurrect**

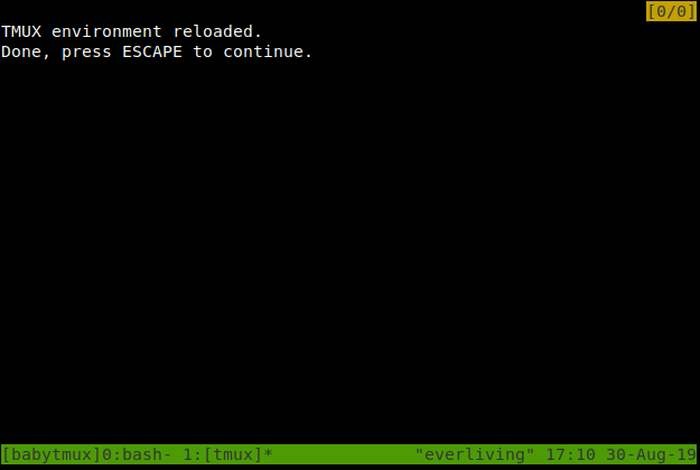
[Tmux Resurrect](https://github.com/tmux-plugins/tmux-resurrect) does just what its name suggests and brings your saved session back to life using a simple command.

This plugin can be installed by adding the following to your .tmux.conf file:

**set** -g **@**plugin 'tmux-plugins/tmux-resurrect'



Now, let Tmux Plugin Manager install it by using the following command in tmux: Ctrl and b, then I (This is “i” in UPPERCASE.)



Once installed, we can start a session and save it with the following: Ctrl and b, then Ctrl + s

To restore our session, we can use Ctrl and b, then Ctrl + r



Using the above, you can maintain an immortal tmux session with all of your preferred tools and processes in place even after rebooting. Give it a try and explore more of tmux’s features to get the most from your terminal.

# List tmux sessions

March 5, 2020  ‐ **3 min read**

Tmux has the concepts of sessions, you can look at a session as a workspace for your terminal. This is convenient when you're working on a project, creating panes and windows, but have to switch to something different in between. In this case you can just keep this specific session running so you can come back to it later. When you're working with multiple sessions it can be useful to list the ones that are active at the moment.

To list all the sessions that are currently managed by the tmux server you can use the tmux list-sessions command. The command shows by default the name of the sessions, the amount of windows per session, when the session was created and whether the session is currently attached to a terminal.

$ tmux list-sessions

first\_project: 4 windows (created Wed Mar 4 18:17:14 2020)

second\_project: 2 windows (created Wed Mar 4 19:00:55 2020) (attached)

Like many tmux commands list-sessions has the shorthand too, tmux ls shows you the same information.

$ tmux ls

first\_project: 4 windows (created Wed Mar 4 18:17:14 2020)

second\_project: 2 windows (created Wed Mar 4 19:00:55 2020) (attached)

Both of the examples above have an equivalent in the command mode of tmux. To start the command mode use <prefix> :. Now you can enter :list-sessions or :ls to see a list of active tmux sessions.

You can use these to make a custom keybinding in your .tmux.conf file. An example would be:

bind <key> list-sessions

By default, list-sessions is binded to the key combination <prefix> s. You can navigate the session list with j and k and activate one by pressing enter. If your main purpose is to switch session you can try out choose-session, it is more verbose and gives you the possibility to switch to a specific window too.

### Format the output

You can specify the format of the session list with the -F option. When you're scripting with tmux this can be a useful feature. For example, if your script requires the session names you can use the following command.

$ tmux ls *-F* "#{session\_name}"

first\_project

second\_project

You can of course use multiple variables in the format string. The output below shows the name of the active window in each session.

$ tmux ls *-F* "#{session\_name}:#{window\_name}"

first\_project:database

second\_project:server

The FORMAT section in the man pages lists a whole lot more options for formatting the output of list-sessions.

In order to scroll up-down:

Ctrl-b then [ then you can use your normal navigation keys to scroll around (eg. Up Arrow or PgDn). Press q to quit scroll mode.

Alternatively you can press Ctrl-b PgUp to go directly into copy mode and scroll one page up (which is what it sounds like you will want most of the time) 🡪 Ctrl+b+repag/avpag