Migrating from OS X to Ubuntu

Björn Haßler, 2020-02-27.

Document available here: https://docs.google.com/document/d/1ofEZURMDOcRoJTc6YF6MBV3aZ71DxQ-sxyHkfbNizuQ/edit?hl=en

Background

Having used a Windows on a laptop but Unix-systems for work in the late 90s / early naughties, I switched to OS X near two decades ago. It offered a good set of desktop applications (like Apple's Final Cut and Adobe Creative Suite) while it was also was based around Unix (i.e., the then-new OS X). This gave me the best of both worlds: Desktop apps like Final Cut Pro, but also access to the Terminal. And indeed, I do think OS X is a great Unix-type system. It works smoothly and has a large user-base.

Having come from Unix systems (including SunOS) I had always mapped the CapsLock key to control, with key strokes like ctrl-a, ctrl-d deeply embedded in muscle memory. Switching to OS X allowed me to retain this: The apple/command key controls window/application functions (like 'new window') while the ctrl key works (for Terminal, and many other apps) as expected in a Terminal (ctrl-a, ctrl-d, etc). So that really worked for me.

I'd been using a range of Macs (both Power PC and Intel, MacBook and MacBook Pro), and eventually a MacBook Air (2011). I had been meaning to replace it but was held back by their Apple's seemingly endless keyboard troubles and other minor issues. In late 2018, the new MacBook Air came out (with an apparently improved keyboard), so I bought one. Within two weeks, parts of the keyboard become unreliable in the same way as had happened for other MacBook models. This 'new improved' keyboard was once again a failure. So that machine went back. In April 2019, my old MacBook Air (2011) went up in a puff of smoke, forcing my hand. (The SSD stayed intact, and I migrated data to a MacMini 2012.) As even that MacBook Air had not been trouble-free (e.g., many battery replacements) I'd decided that it was time to look elsewhere.

So, in April 2019, I decided to switch away from OS X. As opposed to say 10 years ago (when I used many Apple and Adobe desktop applications, like Final Cut, Shake, InDesign, etc etc, and Adobe), I'm now mainly Chrome/Google Drive-based. So my first idea was to use a Chromebook, which have a Linux sub-system. There were really two issues with this. One was that the build-quality of the (£500-ish) Chromebooks I tried wasn't all that great, especially the trackpads were a bit ropy. Obviously, the

Chromebook market is changing quickly, and it may now be possible to get better machines. But at the time, to get a Chromebook that supported Linux and had decent built quality didn't quite work out for me. However, the other big issues was the keyboard. Chrome is a two-modifier system, while OS X is a three modifier system. There are things you can do (see https://github.com/bjohas/kromos, https://github.com/bjohas/extra-keyboards-for-chrome-os) but in the end, I came up against certain issues. This included the issue that — while redefining keyboard layouts works for most keys — there are some keys that seem to be hard-wired, such as speech input to ctrl-d).

Looking around, I could get a Lenovo Yoga 720 as a Grade A machine for around £500, which seemed like a good way to explore Ubuntu. Ubuntu brings greater flexibility regarding keyboards, but as it turns out, it was still a 10-month journey to actually get OS X keyboard replicated. It sounds simple in principle, but in practice, it was hard.

Ubuntu keyboard map like OS X

When switching from OS X to the Ubuntu desktop, you're essentially moving from a three-modifier system (command/alt/ctrl) to a two+1 modifier system (ctrl/alt + super). Super has a special role usually to do with rearranging windows etc. However, in OS X (especially if you use the Terminal) you cannot get away with using all three. In Ubuntu, you could just use ctrl/alt if you didn't have a Super key. The other significant issue is that in Ubuntu, the ctrl key has two separate roles: In desktop applications, it acts like the Apple command key, but in Terminal it acts like ctrl. Given my ctrl-a-etc-muscle memory, that was just hard. So what to do?

Operating system and window manager: Ubuntu 19.04/19.10 with Gnome.

These notes here were developed for Ubuntu 19.04/19.10, which uses Gnome as the default desktop manager. Previous versions of ubuntu used other desktop managers, and some of these notes will not be applicable.

Step 1. Sorting out the overall keyboard shortcuts with gsettings

Many keyboard shortcuts can be configured via Settings and more settings can be controlled by the gnome tweaks tool. It's a good idea to go through the list in the Settings application (keyboard section) and look at the available keys. I do note that there are great things about the shortcuts in Ubuntu, and how configurable they are.

E.g., Super-L/R/U/D are very helpful - this would require an extension on OS X.[1] Screenshot keyboard shortcuts can be set here too.

However, the Settings app (GUI) doesn't have all keyboard shortcuts.

Step 1. Dig into keyboard shortcuts
Go to https://github.com/bjohas/Ubuntu-keyboard-map-like-OS-X/tree/master/scripts Download `gsettings-show.pl` and `setShortcutsGnome.pl`

You can list all settings using `gsettings`, see `gsettings-show.pl` . In particular, the emoji key (set to ctrl+shift+e by default, which conflicts with Google Docs) took a bit of tracking down:

```
> gsettings-show.pl  | grep emoji
...
org.freedesktop.ibus.panel.emoji hotkey ['<...>e']
...
org.freedesktop.ibus.panel.emoji unicode-hotkey ['<...>u']
```

You may want to use `setShortcutsGnome.pl` to set those shortcuts, or to have some examples for how you can define you own.

Step 2. Sorting out non-modifier keys (a UK Mac specific problem)

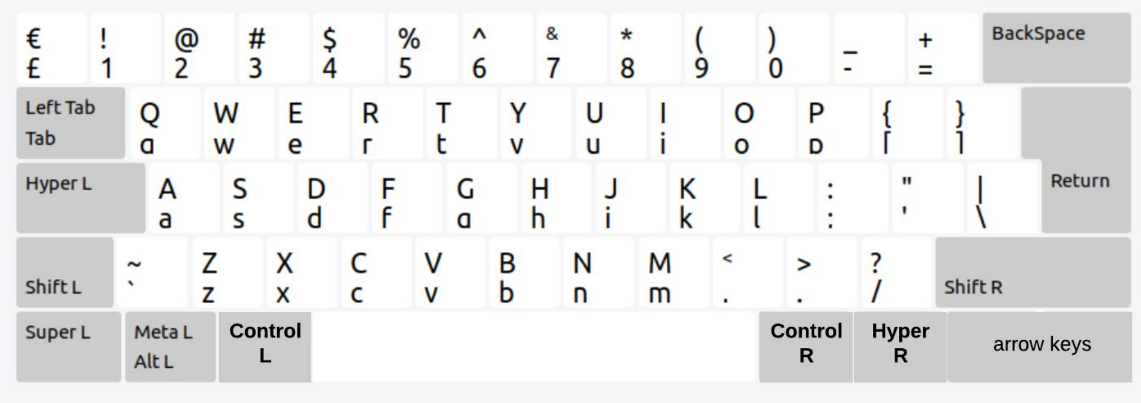
The UK-Mac keyboard is basically a US keyboard with extra keys, so (as I'm touch typing), I've just loaded the US keyboard in Ubuntu. This means you have your usual UK-Mac keys non-modifier keys in the right place. (You don't have £ or € at this stage, see below.)

Step 3. Sorting out modifier keys on the desktop

To get the OS X layout, I've moved the control key next to space (where the apple/command key would be). For most desktop apps, the behaviour (muscle memory) is now like a Mac. Let's call this key 'thumb'. So in the browser 'thumb-t' opens a new tab, 'thumb-a' selects all etc. So that's great. However, in OS X, you can use the Ctrl-key throughout (Cocoa) for going to the beginning of the line, end of line, cut etc. Let's call

[1] One of my go-to applications in OS X was Moom, https://manytricks.com/moom/.

this key 'pinky'. I've used autokey[2] to implement those functions, and mapped this to Hyper, physically located where I would expect 'pinky' to be. So now, pinky-a takes me to the beginning of the line, end of line etc etc. So now, my Ubuntu desktop works like a mac desktop: I can use thumb and pinky like I would on a mac.



Sorting out modifier keys: Ctrl

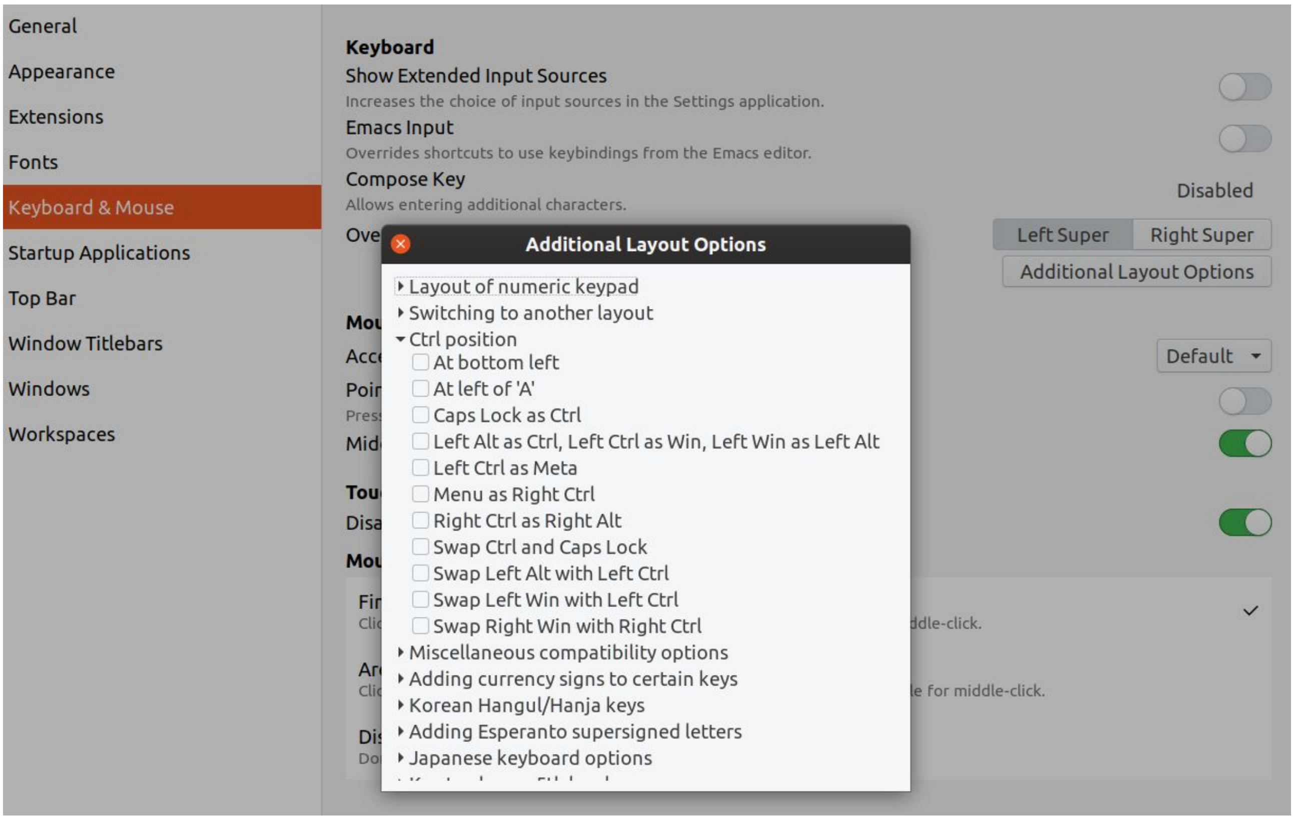
Basically, to place the ctrl key next to space is very natural when you come from OS X. Moving the Ubuntu-control key next to the spacebar means that it essentially works the same as the OS-X-command key (next to the spacebar). I.e., things like copy/cut/paste as well as new tab etc work without retraining muscle memory.

There are options available in Gnome Tweaks. Sometimes I call this the 'thumb-key' (as above): In OS X, it controls various window functions (new window, new tab, cut/copy/paste/select-all etc). That's the same as the role ctrl has on the Ubuntu desktop. By moving the ctrl key to the thumb-key position, much of your OS X muscle memory will just work!

Step 2. Adjust the location of the ctrl key
Install gnome tweaks.

In gnome teaks, move the ctrl-key next to the space bar: Preferences are shown.

[2] https://github.com/autokey/autokey; on OS X, I'd used https://www.keyboardmaestro.com for additional shortcuts where needed. In case you're interested, the autokey commands are all described here: https://github.com/bjohas/Ubuntu-keyboard-map-like-OS-X/blob/master/autokey%20scripts.md.



General

Appearance

Extensions

Fonts

Keyboard & Mouse

Startup Applications

Top Bar

Window Titlebars

Windows

Workspaces

Keyboard

Show Extended Input Sources

Increases the choice of input sources in the Settings application.

Emacs Input

Overrides shortcuts to use keybindings from the Emacs editor.

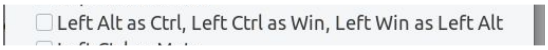
Compose Key — Disabled

Allows entering additional characters.

Additional Layout Options

- ▸ Layout of numeric keypad
- ▸ Switching to another layout
- ▾ Ctrl position
  - □ At bottom left
  - □ At left of 'A'
  - □ Caps Lock as Ctrl
  - □ Left Alt as Ctrl, Left Ctrl as Win, Left Win as Left Alt
  - □ Left Ctrl as Meta
  - □ Menu as Right Ctrl
  - □ Right Ctrl as Right Alt
  - □ Swap Ctrl and Caps Lock
  - □ Swap Left Alt with Left Ctrl
  - □ Swap Left Win with Left Ctrl
  - □ Swap Right Win with Right Ctrl
- ▸ Miscellaneous compatibility options
- ▸ Adding currency signs to certain keys
- ▸ Korean Hangul/Hanja keys
- ▸ Adding Esperanto supersigned letters
- ▸ Japanese keyboard options

Select this option:



□ Left Alt as Ctrl, Left Ctrl as Win, Left Win as Left Alt

From left to right, you now have the keys with the following labels

Ctrl Fn Win Alt Space

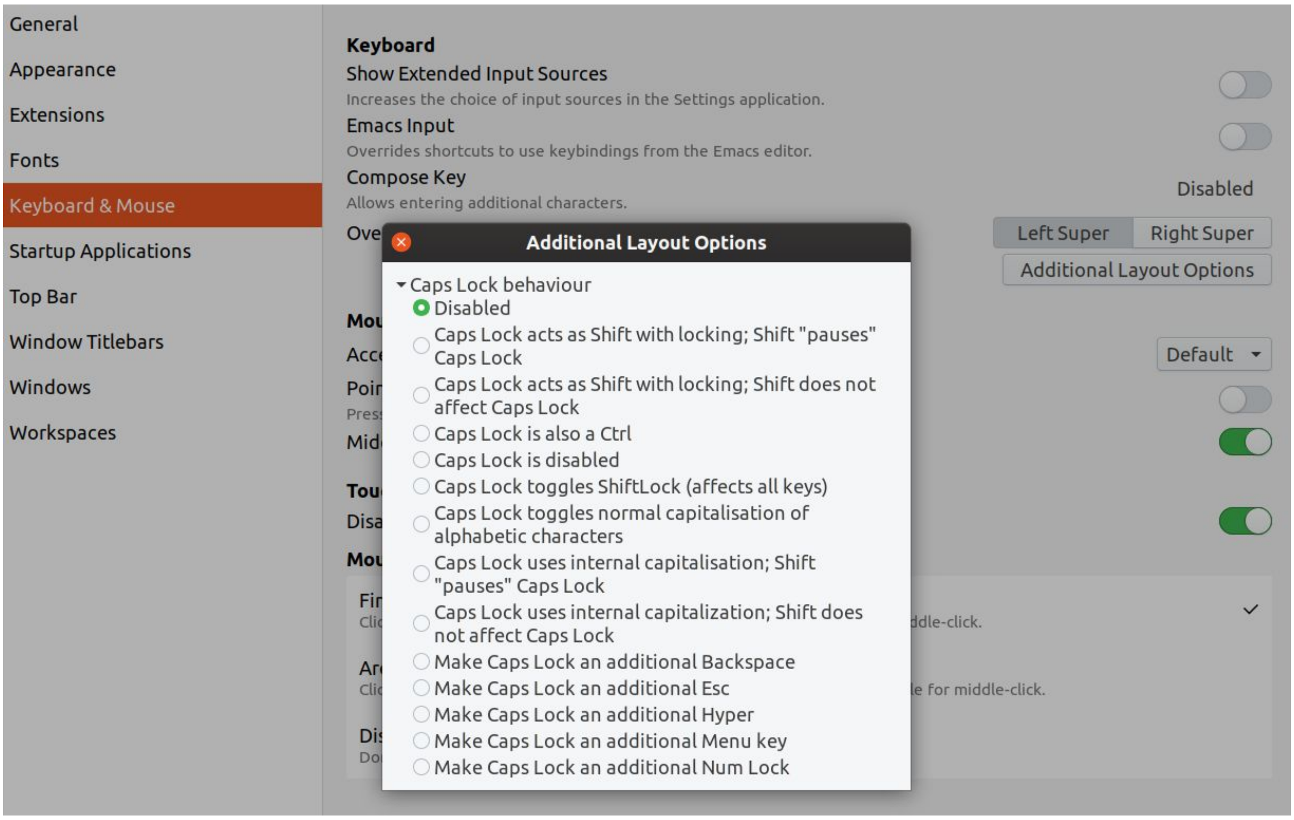
acting as

Super Fn Alt Ctrl Space

That’s now close to how the OS X keyboard works.

Sorting out modifier keys: Hyper.

As discussed above, we now need a Hyper key. If you’re used to having your control on the left bottom edge (labeled Ctrl but now acting as Super) then you’d have to redefine that (I’ll file an extension to gnome-tweaks). However, I’ve always had ctrl in the ‘pinky’ position, so I can just alter the caps lock behaviour. In the same settings go to caps-lock behaviour:



Keyboard

Show Extended Input Sources
Increases the choice of input sources in the Settings application.

Emacs Input
Overrides shortcuts to use keybindings from the Emacs editor.

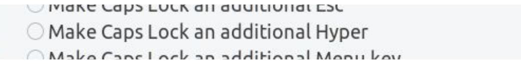
Compose Key — Disabled
Allows entering additional characters.

Additional Layout Options

Caps Lock behaviour

- ● Disabled
- ○ Caps Lock acts as Shift with locking; Shift "pauses" Caps Lock
- ○ Caps Lock acts as Shift with locking; Shift does not affect Caps Lock
- ○ Caps Lock is also a Ctrl
- ○ Caps Lock is disabled
- ○ Caps Lock toggles ShiftLock (affects all keys)
- ○ Caps Lock toggles normal capitalisation of alphabetic characters
- ○ Caps Lock uses internal capitalisation; Shift "pauses" Caps Lock
- ○ Caps Lock uses internal capitalization; Shift does not affect Caps Lock
- ○ Make Caps Lock an additional Backspace
- ○ Make Caps Lock an additional Esc
- ○ Make Caps Lock an additional Hyper
- ○ Make Caps Lock an additional Menu key
- ○ Make Caps Lock an additional Num Lock

Here’s the setting you want:



○ Make Caps Lock an additional Hyper

Now, in the keyboard layout shown above, I’ve also adjusted the right control / right hyper keys. The Gnome Tweaks will not do that for you, but https://github.com/bjohas/Ubuntu-keyboard-map-like-OS-X has got an alternative xkb settings that will do that for you. I’ve also filed an issue to get Gnome Tweaks extended.

Using Hyper

You now have the hyper key and want to use it e.g. for the equivalent of Ctrl-a (in OS X Cocoa). So you need to bring Hyper-a to ‘home’.

Autokey does that via:

```
keyboard.send_key("<home>")
```

See

https://github.com/bjohas/Ubuntu-keyboard-map-like-OS-X/blob/master/autokey%20scripts.md

Caveat: You have to set up a window filter in Autokey, so that these keyboard shortcuts apply only to “anything but Terminal”. Unfortuantely that’s not possible right now (see Autokey bug reports, 2020-02-23). However, you can set a window filter to include your most-used applications (in my case, Chrome, Firefox, autokey, file managers, Zotero - it’s not a short list, but manageable).

For the desktop, this means that the Ubuntu desktop works pretty much like the OS X desktop, as long as you don't need you’re not in the Ubuntu Terminal.

Step 4. Adjusting the terminal

Now the desktop works essentially like the OS X desktop works. The problem is now just the Terminal: On Ubuntu, ctrl works differently in Terminal compared to the desktop; on Mac, it works the same.

On option would be to have a keyboard map specific to terminal Indeed, that’s what I tried for a long time.[3] The idea was to swap ctrl and hyper around. So, in Terminal I move control to the 'pinky' position and the hyper key to the thumb position. That means, as before, I can now do pinky-a, pinky-e, etc etc. So that's great. Moving Hyper to the thumb position, and then use autokey again to map common commands. (thumb-x/c/v to cut/paste etc).

So, this works all pretty simply - and perfectly emulates the Mac Desktop on Ubuntu. The only caveat is that you need to switch the Hyper and Ctrl keys around: The usual position for everything bar Terminal, and swapped round for Terminal. (Btw., the reason I have used Hpyer but haven't used Super is because Super has default roles in Ubuntu. I'm actually using the Caps-Lock key as 'pinky').

This seems simple, and I got some way towards it, but ultimately failed to find a working solution. It sounds trivial, it sounds simple, but in practice, I’ve found it impossible to do.[4]

What to do instead? Adjust the Terminal code.

As you can see here, one route was to explore a ‘terminal only’ keyboard map and then switching to it. While that seems kinda obvious, it’s practically very hard to do (see https://github.com/bjohas/Ubuntu-keyboard-map-like-OS-X/blob/master/README.md). By comparison, modifying Terminal to swap ctrl for hyper was much easier - of course, with the right advice).

You need to rebuild a library and install it. The process is described here:

[3] https://github.com/bjohas/Ubuntu-keyboard-map-like-OS-X

[4] https://github.com/bjohas/Ubuntu-keyboard-map-like-OS-X

https://github.com/bjohas/Ubuntu-keyboard-map-like-OS-X/blob/master/Building%20a%20Gnome-Hyper-Terminal.md

This has the added advantage that the system keeps seeing hyper and ctrl (without those every changing positions). That’s very helpful for autokey.

Thank you to Egmont Koblinger @egmontkob and Christian Persch @chpe for helping me with the compilation of Gnome Terminal. There are a few upstream bugs to be ironed out, but hopefully the modified version of Terminal will make it into an upcoming release.