Globally override key binding in Emacs

95

53

How can I set a key binding that globally overrides and takes precedence over all other bindings for that key? I want to override all major/minor mode maps and make sure my binding is always in effect.

This of course doesn't work:

```
(global-set-key "\C-i" 'some-function)
```

It works in `text-mode`, but when I use `lisp-mode`, `C-i` is rebound to `lisp-indent-line`.

I can go through and override this binding in `lisp-mode` and in every other mode individually, but there must be an easier way. Every time I install a new mode for a new file type, I'd have to go back and check to make sure that all of my key bindings aren't being overridden by the new mode.

I want to do this because I want to emulate bindings I've already learned and ingrained from other editors.

emacs keyboard-shortcuts

edited Oct 30 '10 at 16:55

Peter Mortensen

14.2k ● 19 ● 88 ● 115

asked Mar 25 '09 at 20:59

Brian Carper

59.6k ● 24 ● 145 ● 162

8 Answers

141

I use a minor mode for all my "override" key bindings:

```
(defvar my-keys-minor-mode-map
  (let ((map (make-sparse-keymap)))
    (define-key map (kbd "C-i") 'some-function)
    map)
  "my-keys-minor-mode keymap.")

(define-minor-mode my-keys-minor-mode
  "A minor mode so that my key settings override annoying major modes."
  :init-value t
  :lighter " my-keys")

(my-keys-minor-mode 1)
```

This has the added benefit of being able to turn off all my modifications in one fell swoop (just disable the minor mode) in case someone else is driving the keyboard or if I need to see what a default key binding does.

Note that you may need to turn this off in the minibuffer:

```
(defun my-minibuffer-setup-hook ()
  (my-keys-minor-mode 0))

(add-hook 'minibuffer-setup-hook 'my-minibuffer-setup-hook)
```

edited Jan 12 '16 at 16:06

Stefan

24.3k ● 4 ● 34 ● 69

answered Mar 25 '09 at 21:36

scottfrazer

15.4k ● 3 ● 42 ● 44

1 This seems like a good idea. Is there any way to make sure your minor mode doesn't fight with other minor modes? – Brian Carper Mar 25 '09 at 21:51

3 Make sure your minor mode is first on the list minor-mode-map-alist. – Trey Jackson Mar 25 '09 at 21:54

2 Trey is right. Usually putting this near the end of your .emacs is enough. Also, most bindings you'd override would be ones that major modes are setting ... minor modes generally stay out of the way. – scottfrazer Mar 25 '09 at 22:04

Thanks, I think this will work well. – Brian Carper Mar 25 '09 at 22:29

3 Brian Carper: Here's an enhancement to deal with subsequently-loaded minor modes: stackoverflow.com/questions/683425/… – phils Mar 17 '11 at 14:59

30

As an addition to scottfrazer's answer, I've written the following so that my keybindings retain precedence, even if subsequently-loaded libraries bring in new keymaps of their own.

Because keymaps can be generated at compile time, `load` seemed like the best place to do this.

```
(add-hook 'after-load-functions 'my-keys-have-priority)

(defun my-keys-have-priority (_file)
  "Try to ensure that my keybindings retain priority over other minor modes.

Called via the `after-load-functions' special hook."
  (unless (eq (caar minor-mode-map-alist) 'my-keys-minor-mode)
    (let ((mykeys (assq 'my-keys-minor-mode minor-mode-map-alist)))
      (assq-delete-all 'my-keys-minor-mode minor-mode-map-alist)
      (add-to-list 'minor-mode-map-alist mykeys))))
```

edited May 23 '17 at 10:31

Community ♦

1 ● 1

answered Mar 17 '11 at 14:56

phils

60k ● 7 ● 113 ● 151

2 Rather than an advice, you could use the `afer-load-functions` hook. – Stefan Jan 12 '16 at 16:07

Ah, yes indeed. Thanks Stefan. – phils Jan 12 '16 at 19:26

17

Install `use-package`, eval and you're done:

```
(require 'bind-key)
(bind-key* "C-i" 'some-function)
```

answered Dec 12 '14 at 10:39

Mirzhan Irkegulov

10.3k ● 3 ● 74 ● 133

4 Install only bind-key is enough for the use case, though use-package depends on bind-key. – xuchunyang Jun 15 '15 at 4:48

2 This ('bind-key' package) seems to be the most convenient solution; thanks for sharing. – oligilo Feb 25 '16 at 18:22

1 See also emacs.stackexchange.com/a/360/730 – unhammer Apr 19 '16 at 7:43

14

I found this question while searching for "emacs undefine org mode keybindings", because I wanted to unbind the existing C-c C-b behavior to allow my global map to bury-buffer to work in an org buffer.

This ended up being the simplest solution for me:

```
(add-hook 'org-mode-hook
      (lambda ()
        (local-unset-key (kbd "C-c C-b"))))
```

edited Jun 23 '14 at 12:30

Tshepang

6,369 ● 18 ● 74 ● 114

answered May 6 '14 at 16:51

Jay Doane

186 ● 2 ● 4

12

Although scottfrazer's answer is exactly what you asked for, I will mention for posterity another solution.

From The Emacs Manual:

> "Don't define C-c letter as a key in Lisp programs. Sequences consisting of C-c and a letter (either upper or lower case) are reserved for users; they are the only sequences reserved for users, so do not block them."

If you bind your personal global bindings to C-c plus a letter, then you "should" be safe. However, this is merely a convention, and any mode is still able to override your bindings.

answered Nov 18 '09 at 19:57

Kirkland

164 ● 1 ● 4

3 I didn't expect org-mode, of all modes, to break this rule. `C-c C-h' tells me that C-c a, b, c, and l are bound to org-agenda, org-iswitchb, org-capture, and org-store-link, respectively. – Nate Parsons Aug 12 '12 at 22:35

6 Afaik, binding these is the first step org-mode suggests in order to use it, but the user has to define them himself (i.e. it's not done by default), and may choose any other while doing so. (also, it's because these bindings are supposed to be global, not bound to the org major mode) – Nikana Reklawyks Oct 9 '12 at 3:14

3

If you want to "always use the keybinds in the map, unless I explicitly override them for a specific mode-map", and assuming you are using scottfrazier's approach, you want:

```
(defun locally-override (key cmd)
  (unless (local-variable-p 'my-keys-minor-mode-map)
    (set (make-variable-buffer-local 'my-keys-minor-mode-map)
         (make-sparse-keymap))
    (set-keymap-parent my-keys-minor-mode-map
                       (default-value 'my-keys-minor-mode-map)))
  (define-key my-keys-minor-mode-map key cmd))
```

So

```
(locally-override "\C-i" nil)
```

should remove the "\C-i" binding from the minor mode in the current buffer only. Warning: this is completely untested, but seems like the right approach. The point of setting the parent rather than just coping the global value of my-keys-minor-mode-map is so any later changes to the global value are automatically reflected in the local value.

edited May 23 '17 at 12:34

Community ♦
1 ● 1

answered Apr 18 '13 at 18:16

gbrunick
31 ● 1

2

I don't think you can. That is roughly equivalent to saying that you want to define a global variable that cannot be hidden by local variable declarations in functions. Scope just doesn't work that way.

However, there might be a way to write an elisp function to go through the mode list and reassign it in every single one for you.

answered Mar 25 '09 at 21:06

T.E.D.
36.8k ● 7 ● 59 ● 128

This idea of scoping is technically correct, but `overriding-local-map` is specifically designed to override all other maps. However it's dangerous to use it. – event_jr Apr 1 '14 at 13:16

2

Unless you really want to do this yourself, you should check around and see if anyone else already has done it.

There is a package for Emacs which gives your windows-like keybindings. You should be able to find it through google.

answered Mar 25 '09 at 21:37

JesperE
51.9k ● 15 ● 119 ● 184

4 The package you are thinking of is probably `cua-mode`. – Drew Aug 24 '13 at 16:32

1 Yes, that's the package. – JesperE Aug 25 '13 at 10:46