Advising Functions

WikiTodo: Add basic advising help.

When to Use Advice?

See AdviceVsHooks for a discussion of when to use advice. ***Summary:***

advice is good to have, but not necessarily good to follow all the time ;-).

Advised hello world

```
 (defun foo () (interactive) (insert "hello"))
 (defadvice foo (after foo-after activate) (insert " world!\n"))
 (foo) ;=> hello world!
```

```
;; now you can unadvise the advised function
(ad-unadvise 'foo)
(foo)
```

Simple Advising

To advise a function, you use \`advice-add\`. It takes the function you want to advise, quoted. An advice combinator (around, before, after, etc). And it takes the function you wrote to be run.

A simple piece of before advice looks like this:

```
 (defun my-message (&optional ARG PRED)
   (message "hello!"))
 (advice-add 'save-some-buffers :before 'my-message)
```

Something to note is the function signatures. Functions used with `:before` and `:after` will receive the same arguments as the functions they advise.

Interactive functions

Maintaining interactive-p inside an advised function can be a pain. For details on this, read up the comments etc. in advice.el. Here’s an example from Le Wang on how to maintain interactive-p inside the function definition of an advised function.

```
(defun bar ()
  (interactive)
  (message "bar interactive: %s" (interactive-p))
  (sit-for 1))
```

```
(defadvice bar (around advice-bar activate)
  (interactive)
  (if (interactive-p)
      (progn
        (message "advice-bar interactive: %s" (interactive-p))
        (sit-for 1)
        (call-interactively (ad-get-orig-definition 'bar)))
    ad-do-it))
```