See (info "(emacs)Diff Mode") for more information.

Editing remote files

Emacs can edit remote files transparently (as if they were local) using a feature called Tramp. Whenever Emacs asks for a file, you can indicate a remote file like so: /myname@remotehost:/remote/path/to/file. Emacs retrieves the file over SSH, FTP, or another method and takes care of saving it when you're done. With Tramp you can edit files on different computers using a single Emacs session, even if Emacs is not installed on the remote side.

You can also use Tramp to edit local files with another user's permissions. You can edit files with root privileges via sudo: /sudo::/etc/file, or via su: /root@localhost:/etc/file.

See (info "(TRAMP)") for more information.

Emacs server

Some people like to keep only a single instance of Emacs open and edit all their files in there. Doing this has a few advantages:

You can kill/yank text between buffers in the same instance of Emacs.

Emacs remembers argument histories (what commands you've used, what files you've opened, terms you've searched for, etc.), but only within each instance.

If you have many customizations, starting new instances of Emacs is slow.

Alas, when you type emacs in a shell to edit a file (or when $EDITOR is invoked by an external program), a new instance of Emacs is started. You can avoid this by using emacsclient, which instead opens a new frame connected to an existing instance of Emacs:

In your existing instance of Emacs, type M-x server-start. Or add (server-start) to your .emacs file to make it do that automatically at startup.

To edit a file, type emacsclient -t FILENAME at a prompt. You can also change your $EDITOR to emacsclient -t if you're using programs that automatically invoke $EDITOR. (emacsclient -t opens a new frame on the terminal; alternatively, emacsclient -c opens a new X frame.)

When you're done editing, type C-x C-c, which closes the frame.

For more information, see (info "(emacs)Emacs Server").

Being unproductive with Emacs

Emacs even comes with diversions:

M-x tetris Tetris

M-x hanoi Towers of Hanoi game

M-x doctor Emacs psychotherapist

Common Emacs concepts

Prefix arguments

As we've seen, prefix arguments are sometimes used to indicate repetition:

C-u 10 C-f Forward 10 characters

C-u M-a Backward 4 sentences

We've also seen a prefix argument used to modify the following command (the numeric argument, if provided, is ignored):

M-x shell Create or switch to shell buffer named \*shell\*

C-u M-x shell Create or switch to shell buffer with specified name

If you ever get confused, the documentation for any command (accessible with C-h f or C-h k) describes the effect of the prefix argument, if any.

See (info "(emacs)Arguments") for more information.

Major modes

Every buffer has an associated major mode, which alters certain behaviors, key bindings, and text display in that buffer. The idea is to customize the appearance and features available based on the contents of the buffer.

Emacs ships with dozens of major modes for editing widely used programming languages, markup languages, and configuration file formats. These major modes tell Emacs how to:

Indent your code correctly (usually, simply pressing TAB once will make Emacs indent the current line correctly).

Do syntax highlighting

Identify the boundaries of functions

Invoke interpreters, compilers, or debuggers for your code.

Some commands we've seen, like M-x dired, M-x compile, and M-x shell, in fact use their own special major modes to make their buffers provide certain features (such as highlighting compile errors and making them clickable).

The current major mode is displayed in the mode line. The last thing on the mode line should be one or more items in parentheses, like (Python Narrow). The first of these is the name of the major mode.

You can switch modes in an existing buffer by using M-x and the name of the mode:

M-x java-mode Mode for editing Java files

M-x python-mode Mode for editing Python files

M-x text-mode Mode for editing text files

M-x fundamental-mode Mode with no specializations at all

Emacs is very good at determining the right mode to use when you open a file, so you'll rarely have to use the above commands.

These are examples of the commands provided by language major modes:

Language Some special commands available

Lisp Manipulate s-exps in various ways; execute expressions

Python Indent, unindent blocks; run code in Python shell

HTML Insert and close tags; preview in browser

In almost all cases, major modes for unsupported formats are available as extension packages. You can find many of them on EmacsWiki.

See (info "(emacs)Major Modes") for more information.

Minor modes

Every buffer can also have any number of minor modes, which are extra pieces of functionality you can enable, independently of each other and of the major mode. Minor modes are listed in the mode line after the major mode inside the parentheses. Here are a few commonly used ones: