参考資料: もっと Emacs を使おう

目 次

1 もっと Emacs を使おう

1.1 *scratch* バッファ

コマンド行に `emacs &` と打って， Emacs を起動してください。Emacs 起動後にウィンドウに表示されているバッファの名前は，ウィンドウ下部のモード行に表示されているとおり，*scratch* です[1]。このバッファに何か書き込みを行うか，しばらく時間が経つと

```
;; This buffer is for notes you don't want to save, and for Lisp evaluation.
;; If you want to create a file, visit that file with C-x C-f,
;; then enter the text in that file's own buffer.
```

というメッセージが現われます。メッセージはよく読みましょう ... といっても見慣れない英単語が含まれているので，意味を理解するのが大変かもしれませんね。

> *scratch* バッファは，保存する必要のない一時的なメモを書いたり，Lisp コード[2] を評価するために使います。**新しいファイルを作成したいなら，`C-x C-f` を押して新しいファイル用のバッファを用意し，そこにテキストを書き込んでください**。

しかしながら，これ以降しばらくは *scratch* バッファを使って作業します。保存する必要のない練習用バッファとして使うからだと思ってください[3]。

1.2 変更の取り消し (undo)

undo は誤って行ってしまった操作を取り消す操作です。

1.2.1 一文字削除と取り消し

1. Emacs のウィンドウに `12345` とタイプし，続いて `<BS>` を 5 回押してください。

   タイプした文字が 5 4 3 2 1 の順に削除 (delete) され，全て無くなりましたね。

2. `C-x u` を 1 回押しましょう。

   最後に行った `<BS>` の操作が取り消されて，1 が復活しました。

3. `C-_` を 1 回押しましょう。

   さらに過去に遡って削除の操作が取り消され， 2 も復活しました。`C-x u` と `C-_` [4]は，共にテキストに加えた変更を取り消す (undo) ためのコマンドで，どちらを使ってもかまいません[5]。

4. undo 操作で，345 も復活させてください。undo を続けて行うには `C-_` の方が操作しやすいでしょう。

---

[1] Emacs を起動するときに，編集するファイル名をコマンド行の引数に与えて，`emacs` *file* や `emacs -nw` *file* とすることも可能です。その場合，起動後の Emacs のウィンドウには *file* という名前のバッファが現れます。

[2] Emacs は “Emacs Lisp” というプログラミング言語を内蔵しています。Emacs 上で Lisp を実行したり，Lisp を使って Emacs の動作を変更 (customize) することができます。

[3] Emacs では，バッファの名前に応じて，自動的に適切な編集モードに切り替わります。Emacs の動作はモードによって異なりますので，不適当なモードで Emacs を使うことは好ましくありません。「普通の」バッファ名で Emacs を利用しているときには,「普通の」文章を編集するための Text モードになりますが，*scratch* バッファだと Lisp Interaction モードで動作します。

[4] メニューでは `Edit -> Undo`

[5] `C-/`でも undo できます。

1.2.2 一行消去と取り消し

1. カーソルをバッファの 1 行 3 列目 (;; This buffer ... の T の位置) まで移動し，この行の終りまでを一度の操作で消去してみましょう。

   カーソル位置から行末までを消去するには C-k (kill の頭文字) を使います。

2. カーソルを 2 行下 (;; then enter ... の t の位置) に移動して，3 行目を消去しましょう。

3. 消去した二つの行を undo 操作で復活させましょう。

1.3 消去 (kill) と貼り付け (再入; yank)

ここで紹介するのは，テキストを切り取って貼り付ける操作です。

1. C-k を使って，*scratch* バッファの 1 行目 (;; This buffer ...) すべてを消去しましょう。続いて 3 行目 (;; then enter ...) もすべて消去しましょう。

2. 今回は，カーソルを移動せずに C-y を 1 回押しましょう。

   3 行目が復活しましたね。

3. 再度 C-y を押しましょう。

   undo のときと違って 1 行目は空のままですが，3 行目の内容がカーソル位置に追加されました。

4. さらに何回か C-y を押しましょう。

C-k によって消去された行は，バッファから消えるとともに kill-ring という場所に保存されます[6]。C-y[7] は，最後に kill-ring に保存した内容を，カーソル位置に貼り付ける (再入する; yank[8]) ための操作です。C-y を押しても kill-ring に保存されたものは無くならないので，再度 C-y を押すと，再び同じ内容が貼り付けられます。なお，kill-ring の中身はメニュー Edit -> Select and Paste で確認することができます。また，<BS> 等で 1 文字を削除しても kill-ring には保存されません。

1.4 領域の利用

任意の範囲のテキストを切ったり貼ったりするには，領域 (region) を使います。

1.4.1 領域指定とテキストの移動

1. undo 操作を繰り返し行って，これまでに変更した *scratch* バッファの内容を，すべて元の状態に戻してみてください。操作の途中で混乱してしまって元の状態に戻せなくなったら，一度 Emacs を終了して，emacs & で再度起動してから，どれかキーを一つ押してください。

[6] C-k を連続して押したときの kill-ring の内容については，チュートリアルの「＊挿入と削除」を参照してください。
[7] Edit -> Paste
[8] エディタ vi でも yank という用語を使いますが，Emacs での yank とは用語の使い方が異なります。vi を使う人は注意が必要です。

2. 1 行目の “This” の直後 (“buffer” の直前) の空白位置にカーソルを移動してから C-<SPACE> を押し，ウィンドウ下部のエコー領域に Mark set と表示されるのを確認してください。この操作を「マークを設定する」といいます。

3. カーソルを，1 行目の “buffer” の直後 (“is” の直前) の空白位置に移動してください。

4. C-w を押してください。 “ buffer” が消去されます。

5. カーソルを下の方に動かして，何も書いてない行の先頭に移動してください。続いて C-y を押してください。消去した “ buffer” がカーソル位置に取り出されます。

2. と 3. で行った操作を，領域 (リージョン) の指定といいます。領域とは,「最後にマークを設定した文字」から,「現在カーソルがある位置の直前の文字」までの範囲のことです。先の操作では, “ buffer” を領域指定しました。buffer の前の空白文字も領域に含まれていることに注意してください[9]。

> 「ポイント」という概念を使うと，領域の定義はもっとすっきりします。
>
> ポイントは，カーソルがのっている文字と，その直前の文字との間に存在する**仮想的な**点であって，実際に表示されるものではありません。四角いカーソルの左下の角がポイントだと思ってもいいでしょう。カーソルを移動すれば，それにつれてポイントも移動します。
>
> マークは C-<SPACE> を押したときのポイントに対して設定されます。マークを設定したポイントと，現在のポイントとの間が領域です。

C-w[10] は，指定した領域内の文字を消去するコマンドです。C-k と同様に，C-w で消去したものは kill-ring に保存されますので，C-y を使って取り出せます。

上の操作では，“ buffer” を領域指定しましたので，これが C-w で消去され，C-y で取り出されました。この操作を用いると，任意のテキストを任意の場所に移動することができます。

1.4.2 テキストの複写

M-w[11] は，領域内のテキストを kill-ring に保存するためのコマンドです。C-w とは違い，領域内のテキストは消去されません。C-y と併用すれば，領域内のテキストを他の場所に複写することができます。

1. 2 行目の行頭 (;; If you want ... の ; の位置) にカーソルを移動して，マークを設定してください。

2. 行末の改行文字[12] を含む 2 行目全体を領域指定してください。マーク位置から現在のカーソル位置の直前までが領域ですから，カーソルを次の行 (3 行目) の行頭に移動すればいいです。

3. M-w を押してください。カーソルがマーク位置に移動して，また戻ってきます。見た目には，それ以外の変化はありません。

4. カーソルを下の方にある空白行まで移動して，C-y を押しましょう。領域指定した内容 (2 行目全体) が複写されました。

---

[9] C-<SPACE> で明示的にマークを設定しなくても，他のコマンドを実行したときに，自動的にマークが設定されることがあります。そのために思いがけない動作が起きたら，落ち着いて undo しましょう。

[10] Edit -> Cut

[11] Edit -> Copy

[12] 表示はされませんが，各行の最後には，改行を表す文字が入っています。C-e で行末の改行文字に移動して，C-d (カーソル位置の 1 文字を削除) でそれを削除すれば，どういうことか確認できるでしょう。

1.5 ミニバッファでの編集機能

ウィンドウ下部のエコー領域がユーザーからのキー入力を受け付ける状態になったとき，そこはミニバッファと呼ばれます。例えば，ファイルを読込むとき，C-x C-f を押してからファイル名を入力しますが，この入力はミニバッファに対して行っているのです。

ミニバッファでは，同一行内のカーソル移動コマンド (C-a, C-b, C-f, C-e等) や文字の削除 (<BS>)，行の消去コマンド (C-k) を使って，入力している文字を編集することができます。加えて，次に説明する入力補完機能を利用することができます。

1.5.1 入力補完機能

C-x C-f によるファイルの読み込みを例にとって入力補完の説明をします。入力補完はミニバッファへの入力において一般的に利用できる機能ですので，C-x C-w でのファイル保存等でも使えます。以下の操作を行ってください。

1. ホームディレクトリに存在するファイルの名前を UNIX のコマンドを使って調べて，

   - ファイル名が emacs で始まる既存のファイルが emacs_motion のみであること
   - ファイル名が com で始まる既存のファイルが commands_file と commands の二つのみであること

   を確認してください。これらを満たしていなければ，ファイル名を変更するか，不要なファイルを削除して，この条件を満たすようにしてください。なお，各ファイルの内容は何であっても構いません。

2. C-x C-f を押し，ミニバッファが

   ```
   Find file: ~/
   ```

   となることを確認してください。

3. これからファイル emacs_motion を読み込みますが，ミニバッファには emacs とだけタイプしてください。<ENTER> は**押さない**でください。

4. <TAB> を押してください。ファイル名をすべてタイプしなくても，ミニバッファの表示が

   ```
   Find file: ~/emacs_motion
   ```

   となるはずです。これが入力補完です。

   ホームディレクトリ (~) に存在するファイルのうち，ファイル名が emacs で始まるものは emacs_motion しかありませんから，emacs を入力した時点で，残りの文字列が何であるかは一意に決まります。<TAB> を押すと，その残りの文字列が補完されるのです。

   emacs_m などとタイプしてから <TAB> を押しても，結果は同じです。

5. 続いて <ENTER> を押すと，emacs_motion がバッファに読み込まれます。

6. 次に，commands を読み込むために，C-x C-f を押してから com とタイプして <TAB> を押しましょう。ミニバッファは

```
Find file: ~/commands
```

となります。com で始まるファイルは，commands_file と commands の二つありますから，com だけだとどちらを読み込むのか決められません。<TAB> は，できる限りの範囲で文字列を補完します。

7. さらに _ をタイプして，ミニバッファを

```
Find file: ~/commands_
```

にしてください。commands_ で始まるファイルは commands_file だけですから，この時点で <TAB> を押せば，残りは完全に補完されるはずです。実際にやってみましょう。

8. ファイル名が補完されたら，今回は <ENTER> の代りに C-g を押してください。

C-g はコマンドの中断です。覚えていましたか？

9. 次に C-x C-f に続き，com? と打ってください。

? は補完できる名前の候補を一覧表示します。どんなファイルがあるのか忘れてしまった場合には，この一覧を見ながら，入力することができます[13]。

10. 読み込みの操作を中断してください。

1.5.2 まとめ：ミニバッファでの入力補完コマンド

<TAB>	補完
?	補完候補の一覧表示

1.6 練習

ファイル commands を入力補完機能を利用して Emacs に読み込んでください。ファイルの内容は

```
ls - list directory contents
mv - (略)
cat - (略)
cp - (略)
rm - (略)
```

になっているはずです。このファイルを編集して，コマンドの順番がアルファベット順になるように，行を並べ替えてください[14]。編集が終わったら，同名のファイルに編集結果を保存してください。

[13] 一覧のウィンドウにカーソルを移動して，望みのファイル名のところで <ENTER> を押し，そのファイルを読み込むこともできます。ウィンドウ間のカーソル移動の方法は後述します。

[14] sort コマンドを使うとすぐにできるんですが，今は Emacs の使い方を練習しているので，Emacs でやってください。行の消去または領域の消去と貼り付けを使います。

1.7 バッファとウィンドウ

1.7.1 複数バッファの利用

ここでは，複数のファイルを一度に編集する方法を扱います。まず，ファイル commands の中にある cat の説明の行を，新しいファイル commands_filter の中に複写してみます。以下の操作を行ってください。

1. Emacs が起動中であれば，一度終了してから，再度起動してください。
2. Emacs のバッファにファイル commands を読み込んでください。
3. commands の中の cat の行を，kill-ring に入れてください。

   そのためには，まず，ウィンドウに表示されている cat の行を領域指定する必要があります。領域指定したい行の行頭でマークを設定してから，カーソルを次の行の行頭まで移動すればいいですね。

   さらに，適当な Emacs のコマンドを使って，指定した領域を kill-ring に入れます。消去はしないでください。操作を誤ったら，焦らずに undo しましょう。
4. 新規のファイル commands_filter 用のバッファを用意してください。

   2. で行ったのと同じ Emacs コマンドを使います。ミニバッファに入力する名前は，もちろん，commands_filter です。
5. commands_filter 用のバッファが用意できたら，cat の説明が入っている kill-ring の内容を取り出して (貼り付けて) ください。
6. バッファ commands_filter をファイルに保存してください。

以上でファイル commands_filter が出来上がりました。ところで，commands というバッファはウィンドウから消えてしまいましたが，このバッファは無くなったのでしょうか?

答えは「いいえ」です。バッファ commands は，今使っている Emacs から無くなった訳ではなくて，新しく用意したバッファの陰に隠れてしまっただけです。次に，バッファ commands の内容を再度ウィンドウに表示してみます。

1. `C-x b` を押してください[15]。これは編集中のバッファ (current buffer) を切り替えるコマンドです。
2. エコー領域に

   ```
   Switch to buffer: (default commands_file)
   ```

   と表示されるはずです。ここで (default commands) とあるのは,「特に指定がなければ commands とします」ということなので，そのまま `<ENTER>` を押します。もしも default の後に commands 以外の文字列が現われたら，それを commands に修正してから `<ENTER>` を押します[16]。

   「デフォルト (default)」という言葉は Emacs 以外の場面でもよく使いますので，この機会に覚えておきましょう。

[15] この場面では，`C-x C-f` でも構わないのですが，バッファというものを意識して，バッファ切り替えのコマンドを使ってみましょう。

[16] この操作をメニューで行うには，メニューバーの Buffers をクリックして，commands という項目を選択します。

commands のバッファがウィンドウに現れましたから，このバッファを引き続いて編集することも可能です。

以上の操作では，二つのバッファを同時に使って編集作業を行いましたが，三つ以上のバッファを同時に使うこともできます。

1.7.2 ウィンドウの分割

1. まず，バッファ切り替えのコマンド (C-x b) を使って，*scratch* バッファをウィンドウに表示してください。

   Emacs を使っている間は，バッファ削除の操作をしなければ，*scratch* バッファは常に存在しています。ミニバッファに入力するバッファ名は *scratch* です。<TAB> による入力補完も利用できます。

2. C-x 2 を押しましょう。ウィンドウが上下二つに分割されました。でも表示されている内容は同じですね。

3. a という文字を入力してください。カーソルは上のウィンドウにありますから，文字はそこに入力されたのですが，下のウィンドウにも同じ文字が現われました。*scratch* という名の一つのバッファを，別々のウィンドウ (窓) から覗いて，編集している状態にあるからです。

4. C-x o を押してください。この o は other の頭文字です。別のウィンドウにカーソルが移動しました。もう一度，C-x o を押しましょう。また，別のウィンドウにカーソルが移動して，上に戻りました。

5. さらに 10 個ほど a を入力して[17]，カーソルの位置を確認してから C-x o を押してください。

   別のウィンドウにカーソルが移動しましたが，カーソルの位置が上と下では違いますね[18]。

   一つのバッファを複数のウィンドウで編集しているときには，あるウィンドウで行ったバッファの内容変更 (文字入力) は，他のウィンドウにも反映されます。一方，カーソル位置や表示位置はウィンドウ毎に独立ですので，一つのバッファの異なる場所を，別々のウィンドウに表示して，それぞれで編集することができます。長い文書を扱うときに便利です。

6. C-x 1 を押しましょう。これは，編集中 (カーソルの存在する) ウィンドウをただ一つだけ残して，他のウィンドウを閉じるコマンドでした。

7. C-x 3 を押してください。結果を確認したら，ウィンドウを一つに戻してください。

1.7.3 複数ウィンドウでの複数バッファの操作

第 1.7.2 節では，異なるウィンドウから一つのバッファの内容を眺めてみましたが，もちろん，異なるウィンドウで異なるファイル (バッファ) を編集することもできます。

1. Emacs のウィンドウを上下二つに分割してください。

[17] 10 回 a のキーを押してもいいですが，C-u 1 0 a でもできます。ほとんどの場合，C-u を使えば，その後に指定した数の分だけ，続くコマンド (今の場合は a を入力するという操作) が繰り返されます。ただし，C-u の動作は後続のコマンドに依存しますので，C-u がコマンドの繰り返しを意味しないこともあります。

[18] 上のウィンドウでのカーソル位置をもう一度確認したければ，C-x o を押して上に移って，すぐ下に戻りましょう。

2. 上のウィンドウで，ファイル commands を読み込んでください。

3. カーソルを下のウィンドウに移動させて，ファイル commands を読み込んでください。

ここでは，実際に作業を行うことは略しますが，カーソルを上下のウィンドウに移動することによって，各ウィンドウ内のバッファを編集することが可能です。

これまで使ってきたファイルへの保存の操作 (C-x C-s) では，保存されるバッファは，カーソルが存在するウィンドウ内のもののみです。複数のウィンドウに表示されている異なるバッファをファイルに保存するには，それぞれのウィンドウで C-x C-s を押す必要があります。

一回のコマンド操作で複数のバッファをファイルに保存したければ C-x s を使います。このコマンドを実行すると，元のファイルから内容変更のあったバッファ各々について，保存するか否かを尋ねられますので，保存する場合には y で答えます。

では，C-x s を試してみましょう。今の場合，変更を施したバッファはないはずですから，保存の対象となるバッファはありません。エコー領域に，その旨表示されます。

1.7.4 バッファの一覧と削除

今 Emacs が持っているバッファの一覧を表示するコマンドは C-x C-b[19] です。

1. C-x C-b を実行してください。

   ウィンドウを分割した状態で実行した場合には，カーソルの存在しないウィンドウに *Buffer List* という名前のバッファ一覧が表示されます。一つのウィンドウのみで作業していた場合には，ウィンドウが分割されて，一覧が現われます。

2. *Buffer List* ウィンドウ[20] の *scratch* という語のある行にカーソルを移動して，<ENTER> を押してみましょう。バッファ切り替えのコマンド (C-x b) を使わなくても，*Buffer List* が *scratch* に切り替わります。

次は不要になったバッファを削除してみましょう。そのためのコマンドは C-x k[21]です。これを使って commands バッファを削除してみます。

1. C-x k を押してください。エコー領域に

   ```
   Kill buffer: (default *scratch*)
   ```

   と表示されます。カーソルの存在するウィンドウに commands を表示した状態でコマンドを実行すると，デフォルトのバッファ名が commands になりますが，今は *scratch* になっているはずなので，commands とタイプしてから <ENTER> を押します。<TAB> による入力補完も可能です。

2. バッファ一覧を表示して，バッファが削除されたか確認しましょう。ただし，ここでバッファ一覧を表示するには，C-x C-b を使ってください。バッファ切り替えコマンド (C-x b) でバッファ一覧 (*Buffer List*) を表示しても，一覧に変更は反映されていません[22]。

なお，第 ?? 節で確認したように，バッファを削除しても，ファイルが削除されるわけではありません。ファイルを編集する必要が生じたら，C-x C-f で再度バッファに読込めばよいのです。

[19] Buffers -> List All Buffers
[20] このバッファ内で使えるコマンドは，ファイル編集用のバッファとは異なりますが，カーソル移動は普通にできます。
[21] File -> Close (current buffer)
[22] バッファ一覧を最新にするには，*Buffer List* バッファで g を押します。

1.7.5 複数フレームの利用

オプション –nw をつけずに新しいウィンドウで Emacs を開いたときには，こんなこともできます。ここではメニューを使った操作を紹介します。試してください。

1. `File -> New Frame`
2. `File -> Delete Frame`

注意 特に理由がない限り，コマンド行から emacs & を何度も打って，複数の Emacs を同時に動かすことは避けましょう。計算機の負荷が大きくなってしまいます。Emacs で複数のウィンドウ (Emacs の用語ではフレーム) を利用したければ，ここで紹介した方法を使いましょう。

1.8 文字列の検索と置換

まずは検索から。

1. Emacs のチュートリアルを開きましょう。やり方を忘れていたら第**??**節を見てください。
2. `C-s` を打って，エコー領域の表示を確認してください。I-search: と表示されましたね。
3. ウィンドウ内のカーソルの動きに注目しながら，C-s という文字列をそのままタイプしてください。CTRL を押しながら s ではありません。
4. ウィンドウの上部に「＊検索」という単語が現われるまで，`C-s` コマンドを何回か実行しましょう。
5. 見つかったら，`<ENTER>` を押してから，そこの説明を読みましょう。

次に置換です。

1. `M-%` を押してください。メタキー (`<ALT>`) と `<SHIFT>` を使います。
2. エコー領域の Query replace: に対して，emacs とタイプし，`<ENTER>` を押してください。
3. エコー領域の Query replace emacs with: に対して，GNU Emacs とタイプし，`<ENTER>` を押してください。
4. エコー領域に Query replacing emacs with GNU Emacs: (? for help) と表示されます。これは,「emacs を GNU Emacs で置き換えますか?」という意味です。y を押しましょう。
5. さらに y を押し，次に n を押してみましょう。
6. `<ENTER>` を押しましょう。エコー領域に置換した回数が表示されます。

1.9 日本語の入力

Emacs で日本語入力を実現する仕組みは幾つかあります。日本語入力の操作の細部は，使っている日本語入力の仕組みによって異なりますので，ここでは，英数字入力と日本語入力を切り替えるキー操作を紹介するにとどめます。

日本語・英数字入力の切り替え: `C-\`

漢字変換や候補の確定は `<SPACE>` や `<ENTER>` で普通にできますので，試してみてください。