Contents

Never compile this.

```
;; -*- no-byte-compile: t; -*-
```

1. Watch What You Eat

   ID: org_gcr_2017-05-12_mara:BD133C8B-8DFF-45C0-967D-CB83693C54B4

   (a) Use-Package

   ID: org_gcr_2018-04-11T09-44-43-05-00_mara:55083825-349A-44D0-8026-2E65

   Configure `use-package`.
   Sysop is likely to use this infrequently.
   Start EMACS with this command:

   ```
   emacs --debug-init --no-init-file --no-splash --background-color white
   ```

   ```
   (load-file "~/src/help/.org-mode-fundamentals.emacs.el")
   ```

   Code requiring package-management can only follow this block.
   Before ELPA, I used SVN to manage software packages for EMACS. After ELPA, I used ELPA-packages. Now this system uses MELPA and GNU. MELPA packages always have their issues fixed very quickly. GNU packages rarely have issues.
   Initialize Package.

   ```
   (package-initialize)
   (add-to-list 'package-archives
                '("melpa" . "http://melpa.org/packages/") t)
   (add-to-list 'package-archives
                '("gnu" . "http://elpa.gnu.org/packages/") t)
   ```

   Use-Package is the most configurable and performant way to manage ELPA packages. Startup speed matters but not right now so don't use `:defer`.
   Add the Use-Package distribution the load path.

   ```
   (add-to-list 'load-path"~/src/use-package")
   ```

   Load Use-Package and it's partner Diminish. Every package loaded before this point uses `require`. Every package loaded after this point uses `use-package`.

```
(eval-when-compile
  (require 'use-package))
(use-package diminish)
```

(b) El-Get

ID: org_gcr_2018-04-11T09-44-43-05-00_mara:925A5891-AB13-4972-9C95-3B49

EL-Get handles things that don't easily fit anywhere else.
Initialize EL-Get.

```
(add-to-list 'load-path "~/.emacs.d/el-get/el-get")
(unless (require 'el-get nil 'noerror)
  (with-current-buffer
      (url-retrieve-synchronously
       "https://raw.githubusercontent.com/dimitri/el-get/
   master/el-get-install.el")
    (goto-char (point-max))
    (eval-print-last-sexp)))
```

```
(setq help/el-get-packages nil)
```

Make it really easy to remind yourself and others what EMACS really stands for (in this case it is fun).

```
(add-to-list
 'el-get-sources
 '(:name emacs-name
         :type http
         :url "http://www.splode.com/~friedman/software/
   emacs-lisp/src/emacs-name.el"
         :features emacs-name
         :autoloads nil
         :website "http://www.splode.com/"
         :description "emacs acronym expansions"))
(add-to-list 'help/el-get-packages 'emacs-name)
```

It is not good to flame people on the Internet. It is good to *know* what it is all about, and here is a way to see some examples of the absurdity of it all.

```
(add-to-list
 'el-get-sources
 '(:name flame
         :type http
         :url "http://www.splode.com/~friedman/software/
   emacs-lisp/src/flame.el"
```

```
          :features flame
          :autoloads nil
          :website "http://www.splode.com/"
          :description "automatic generation of flamage, as
   if we needed more"))
(add-to-list 'help/el-get-packages 'flame)
```

People love horoscopes, so, provide them.

```
(add-to-list
 'el-get-sources
 '(:name horoscope
          :type http
          :url "http://www.splode.com/~friedman/software/
   emacs-lisp/src/horoscope.el"
          :features horoscope
          :autoloads t
          :website "http://www.splode.com/"
          :description "generate horoscopes"))
(add-to-list 'help/el-get-packages 'horoscope)
```

James Parry [1] must always be honored.

```
(add-to-list
 'el-get-sources
 '(:name kibologize
          :type http
          :url "http://www.splode.com/~friedman/software/
   emacs-lisp/src/kibologize.el"
          :features kibologize
          :autoloads nil
          :website "http://www.splode.com/"
          :description "generate ravings about kibology, in
   the style of kibo"))
(add-to-list 'help/el-get-packages 'kibologize)
(defun help/kibologize-insert ()
  (interactive)
  (let ((current-prefix-arg '(4)))
    (call-interactively 'kibologize)))
```

You might not always remember your shopping list, but we will remember it for you... though not necessarily for wholesale.

```
(add-to-list
 'el-get-sources
```

[1] https://en.wikipedia.org/wiki/James_Parry

```
 '(:name shop
         :type http
         :url "http://www.splode.com/~friedman/software/
   emacs-lisp/src/shop.el"
         :features shop
         :autoloads nil
         :website "http://www.splode.com/"
         :description "generate random shopping lists"))
(add-to-list 'help/el-get-packages 'shop)
```

Do you remember when those great AT&T adds were on television and it changed your life and bought you a kitten? You will.

```
(add-to-list
 'el-get-sources
 '(:name youwill
         :type http
         :url "http://www.splode.com/~friedman/software/
   emacs-lisp/src/youwill.el"
         :features youwill
         :autoloads t
         :website "http://www.splode.com/"
         :description "generate meaningless marketing hype"
   ))
(add-to-list 'help/el-get-packages 'youwill)
(defun help/youwill-insert ()
  (interactive)
  (let ((current-prefix-arg '(4)))
    (call-interactively 'youwill)))
```

Ask el-get to make sure that those desired packages are installed.

```
(el-get 'sync help/el-get-packages)
```

(c) Stand-Alone

ID: org_gcr_2018-06-12T21-48-52-05-00_mara:D53227C4-5084-4A36-AD92-49B7

```
(add-to-list 'load-path "~/src/help/lisp")
```

(d) Helper

ID: org_gcr_2018-04-11T09-44-43-05-00_mara:27CF9494-7866-4AB9-A2AD-254F

Speed up code execution. Review as desired.

```
(use-package auto-compile
  :ensure t
```

```
  :config
  (auto-compile-on-load-mode)
  (auto-compile-on-save-mode)
  (setq auto-compile-display-buffer 1))
```