

Those developers using Emacs might find the following Elisp snippets helpful.

## Parsing compiler messages

GFortran and NAG compiler messages are usually parsed erroneously by the `compile` package of Emacs. The following Elisp lines correct the parsing to allow for `C-x ~` to jump to the next error after a `M-x compile` or `M-x recompile`.

```
;; gfortran 4.1:
(add-to-list 'compilation-error-regexp-alist
  ;; In file pfft3d.f90:182
  ;;
  ;;          MPI_double_precision, &
  ;;
  ;;          1
  ;; Error: Symbol 'mpi_double_precision' at (1) has no IMPLICIT type
  '("^ *In file \\([^\n]*\\):\\([0-9]+\\)[ \\t]*\\n\\n.*\\n.*\\n.*Error:.*\\n" 1 2))
;; NAGWare5 f95:
(add-to-list 'compilation-error-regexp-alist
  ;; Error: mpi.F90, line 116: Implicit type for MPI_COMM_WORLD in MPI_MOD_INIT
  ;;
  ;;          detected at MPI_M@<end-of-statement>
  ;; [f95 terminated - errors found by pass 1]
  '("^Error: \\([^\n]*\\), line \\([0-9]+\\):" 1 2))
(add-to-list 'compilation-error-regexp-alist
  ;; Fatal Error: global.F90, line 25: Cannot find module MPI_M
  ;;
  ;;          detected at MPI_M@<end-of-statement>
  '("^Fatal Error: \\([^\n]*\\), line \\([0-9]+\\):" 1 2))
```

## Inserting new routines

As our coding style wishes the enclosure of Fortran routines (subroutines and functions) by calls to `push_sub()` and `pop_sub()` these Emacs functions can be used to insert a subroutine or function skeleton like

```
subroutine bar()
  call push_sub('current_file.bar')

  call pop_sub()
end subroutine bar
```

at current point position. The functions ask for the desired name of the subroutine. Just insert the following code in your `.emacs` file and call the `octopus-keybindings` function somewhere, e. g. via `f90-mode-hook`. It is then possible to insert a subroutine by `C-c C-s` and a function by `C-c C-u`, the filename is appropriately substituted into `push_sub`'s argument.

```
(require 'f90)

(defun octopus-insert-new-subroutine (name)
  "Inserts a new subroutine skeleton at point."
  (interactive "MSubroutine name: ")
  (octopus-insert-new-routine "subroutine" name))

(defun octopus-insert-new-function (name)
  "Inserts a new subroutine skeleton at point."
  (interactive "MFunction name: "))
```

## Developers:Emacs\_helpers

```
(octopus-insert-new-routine "function" name)
(backward-sentence))

(defun octopus-insert-new-routine (type name)
  "Inserts a new routine skeleton of type with name at point:
<type> <name>()
  call push_sub('<filename>.<name>')

  call pop_sub()
end <type> <name>"
  (let* ((file-base (substring (buffer-name) 0 (string-match "\\.[^.]*$" (buffer-name))))
        (routine (concat type " " name "()\n"
                          "  call push_sub('\" file-base \".\" name '\')\n\n"
                          "  call pop_sub()\n"
                          "end \" type \" \" name \"\n\n")))
    (princ routine (current-buffer))
    (forward-line -2)
    (f90-indent-subprogram)
    (forward-line -4)
    (end-of-line)
    (forward-char -1)))

(defun octopus-keybindings ()
  "Bind octopus-insert-new-subroutine to C-c C-s and octopus-insert-new-function
to C-c C-u."
  (local-set-key "\C-c\C-s" 'octopus-insert-new-subroutine)
  (local-set-key "\C-c\C-u" 'octopus-insert-new-function))
```

---

Back to [Developers Manual](#)