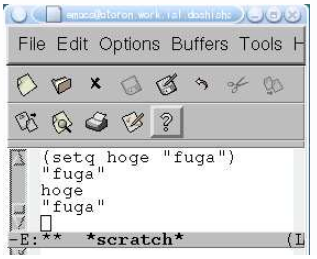
emacs を使ったウェブサービスへの展開

LILO Monthly Seminar
2003 年 1 月

上川純一@LILO

emacs とは

- 文書編集を核とした作業環境
- elisp と呼ばれる lisp の実装



Web service って

- Web 経由でいろいろなサービス
  - ➡ http を利用
  - ➡ 動的に HTML を生成
  - ➡ 個々のケースに対応

elserv

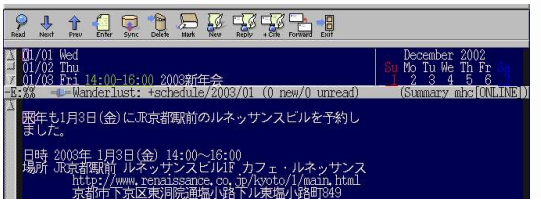
- Yuuichi Teranishi さんの製作した web server
- emacs lisp (+ ruby) で動く
  http を解析し，ページを elisp で処理して返す
- `apt-get install elserv`
- `elserv-0.4.0.tar.gz` を取得, tar xfz で展開，`make install` する

elserv 使用例

- Web ブラウザを利用した emacs アプリケーション
  - ➡ mhc
  - ➡ devscripts-el
  - ➡ wysihtml
  - ➡ emacs-wiki

mhc

- メールから予定を抽出するシステムを核としたスケジューラ
- メールの文面から日付とかを抽出してくれる



mhc with elserv

- elserv を利用して mhc の情報をブラウザから利用できるように

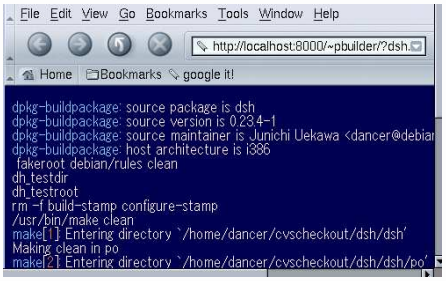
mhc with elserv 利用方法

- `M-x load-file es-mhc.el`
- `M-x elserv-mhc` で `http://localhost:10000/` でアクセス可能

devscripts-el

- Debian パッケージコンパイル環境
- コンパイルログをブラウザ経由で閲覧

devscripts-el ログ表示

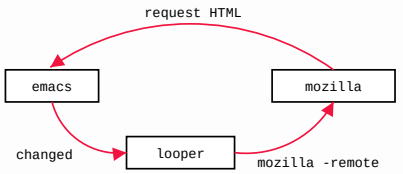


devscripts-el

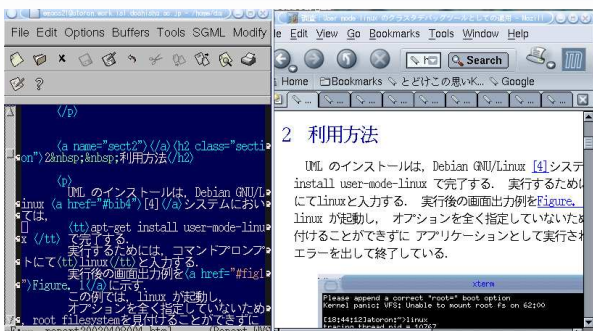
- `http://www.netfort.gr.jp/~dancer/software/downloads/` からソース取得
- devscripts-el-xx.tar.gz を展開
- `M-x load-file` で読み込む
- `debuild` とか
- `M-x pbuilder-log-view-elserv` で起動

wysihtml

- emacs で編集する HTML ファイルを mozilla でプリビューしたい elserv があれば，わざわざセーブする必要もない

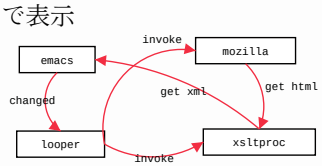


wysihtml 画面写真

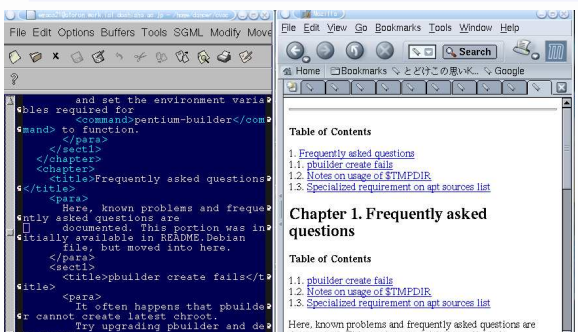


wysidocbookxml

- wysihtml の考えを応用して docbook に適用してみる
- emacs 編集中の XML の一部を抽出，elserv 経由で提供し，xsltproc で処理して mozilla で表示



wysidocbookxml 画面写真

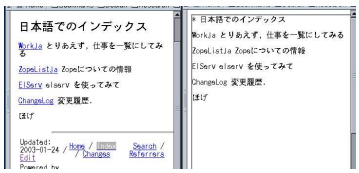


wysihtml

- `http://www.netfort.gr.jp/~dancer/software/wysihtml.html.ja`
- wysihtml-xx.tar.gz を展開 `./configure && make install` でインストール
- `M-x load-file /usr/local/share/wysihtml/wysihtml.el`
- `M-x wysihtml-mode` で wysihtml-mode になる

emacs-wiki

- emacs を利用した wiki
- `WikiName` というように入力したらリンクになる
- ページを生成，共同作業



emacs-wiki

- emacs-wiki をインストール
- `M-x elserv-wiki-start`
- Wiki ファイルが a+w の場合 editable になる
- emacs-wiki 単体だったら emacs から Wiki 的編集ができる

elserv のアプリケーション

- elisp が書ければ比較的簡単に作成できる
- emacs から外のアプリケーションのインタフェース
- elisp のアプリケーションと一般ユーザとの架け橋？
  - ➡ elserv 経由のメールソフト
  - ➡ elserv 経由のシステム管理ツール

elserv の起動

- elserv を指定したポート番号で起動する

```
(setq elserv-id
(elserv-start port-number))
```

- 返り値は elserv のプロセス (デフォルトの値は `(elserv-find-process)` で見付かる)

elserv でウェブページを発行

- 固定文字列のページを発行

```
(elserv-publish (elserv-find-process)
     "/hoge/fuga.html"
     :string "Hello World"
     :content-type "text/plain"
     :description "これは説明文")
```

elserv で動的ページを

- elisp の関数を呼ぶようにする

```
(elserv-publish (elserv-find-process)
    "/hoge/fuga.html"
    :function 'some-handler-function
    :description "これは説明文")

(defun some-handler-function
     (result path ppath request)
       ... body )
```

elserv で動的ページを...続

```
(defun some-handler-function
    (result path ppath request)
  (elserv-set-result-header result
      '(content-type
        "text/html; charset=euc-jp"))
  (elserv-set-result-body
   result
   (encode-coding-string
 "<html><head><title></title></head>
<body><p>テスト</p></body></html>" 'euc-jp)))
```

http://localhost:8000/

http://.../hoge/fuga.html

指定した出力が出る

elserv ハンドラ関数の引数

- result: elserv-set-result-* でクライアントへ返す値を設定するのに利用
- path: 公開パスからの相対的値
- ppath: 公開パス
- request: HTTP リクエストに関する情報

path と ppath

- `(elserv-publish (elserv-find-process) "/published" :function 'myhandler :description "これは説明文")`
- `http://localhost/published/fugafuga` へのアクセス
  - ➡ ppath=http://localhost/published
  - ➡ path=/fugafuga
- URL に何かをエンコードするのに使える

path と ppath

- path を利用して情報を URL に埋め込む
- elisp のプログラムが URL の値によっていろいろと違う反応を返せる
  セッション管理等が可能になる

elserv で HTML をそのまま返す

```
(require 'mcharset)
(setq charset
  (detect-mime-charset-region (point-min) (point-max)))
(elserv-set-result-header
  result
  (list 'content-type (concat "text/html; charset="
    (symbol-name
      charset))))
(elserv-set-result-body result
  (encode-mime-charset-string
  (buffer-string)
  charset))
```

`htmlize.el` を使えば emacs の font-lock をそのままブラウザ表示で使える

まとめ

- emacs で動く elserv というウェブサーバがある
- elserv を使って emacs を外部のアプリケーションとのインタフェースを作成できる