Web Server Manual

Mike Burns (netgeek@speakeasy.net)
Greg Pettyjohn (gregp@ccs.neu.edu)
Jay McCarthy (jay.mccarthy@gmail.com)

June 18, 2006

Copyright notice

Copyright ©1996-2006 PLT

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Library General Public License, Version 2 published by the Free Software Foundation. A copy of the license is included in the appendix entitled “License.”

Contents

1 Quick Start **1**

1.1 Quick Start on Unix 1

I Administration **2**

2 Directory Structure **3**

3 Configuration **4**

3.1 Named Virtual Hosts 4

3.2 The Configuration Tool 4

3.2.1 Managing Virtual Hosts 4

3.3 Configuration File Syntax 4

3.3.1 Host Table Syntax 6

4 Passwords **9**

5 Starting the Server **10**

6 Monitoring the Server **11**

II Programming **12**

7 Before You Begin **13**

7.1 Reloading the Cache 13

7.2 Directories 13

8 Static Content **14**

9 Writing Servlets **15**

9.1 Module-Based Servlets 15
9.2 Unit-Based Servlets 16

10 Servlet Library **17**
10.1 Data Definitions 17
10.1.1 Environment 17
10.1.2 Request 17
10.1.3 Response 18
10.2 Core Procedures 19
10.3 Helpful Servlet Procedures 20
10.4 Example Multiplication Servlet 21
10.5 Example Math Test Servlet 23
10.6 Servlet Development Environment 23

11 Semi-Internal Functions **25**
11.1 Miscellaneous 25
11.2 Starting the Server from a Program 26
11.3 Constructing Configurations 26
11.4 Monitoring the Server 27
11.4.1 Example 27

License **28**

Index **32**

1. Quick Start

To just use a servlet without caring about administrative or programming details:

1.1 Quick Start on Unix

1. If you have root access, just run `plt/bin/web-server`; otherwise, run `plt/bin/web-server -p 8080`.
2. Example servlets are located in `collects/web-server/default-web-root/servlets/examples/`.
3. To run the example servlet **add.ss**, use the URL `http://localhost/servlets/examples/add.ss`, or `http://localhost:8080/servlets/examples/add.ss` if the server was started with `-p 8080`.

Part I

Administration

2. Directory Structure

By default, the configuration tool (see section 3.2) organizes files containing the Web server's configurations, documents, and servlets in one directory tree per virtual host. This tool can be used to manage all trees for the Web server. A default tree exists in `collection/web-server/`, which looks like:

```
configuration-table
default-web-root
  conf
    servlet-error.html
    forbidden.html
    servlet-refresh.html
    passwords-refresh.html
    not-found.html
    protocol-error.html
  htdocs
    Defaults
      index.html
      documentation
        ...
  log
  passwords
  servlets
    configure.ss
my-other-host
  conf ...
  htdocs ...
  log
  passwords
  servlets
still-another-host ...
```

Files may be relocated or shared between hosts by editing the details for that host using the configuration tool. For more on virtual hosts, see section 3.1.

3. Configuration

3.1 Named Virtual Hosts

Many Web sites can be served from one Web server process through the use of virtual hosting. When a client (Web browser) sends a request to the server, the domain name is embedded in the request. The Web server can then service the request in different ways based on that requested domain name.

Zero or more virtual hosts may be defined, in addition to the default host. If a request is sent to the server for a host which is not in the virtual host table, then the default host is used. For example, if the client requests `http://www.plt-scheme.org/`, the server can send back information in the directory `/var/www/plt/www.plt-scheme.org/`; if the client requests, from the save server, `http://www.drscheme.org/`, the server can send back information from the directory `/home/mburns/www.drscheme.org/`.

To configure virtual hosting using the configuration tool, see section 3.2.1. To configure virtual hosting by editing the file directly, see `virtual-host-table`.

3.2 The Configuration Tool

The configuration tool is a normal servlet, run by the Web server, that can manage the configuration files. This is provided as an alternative to editing the file directly.

For security reasons, the configuration servlet is only accessible from the local machine. Thus, one gets to the servlet via `http://localhost/servlets/configure.ss`.

Once there, select the filename for the configuration file (the default should work). As part of the basic configuration, you can change the port to listen on, the maximum number of waiting connections, and the initial timeout. Individual hosts can be added and edited below the basic configuration.

3.2.1 Managing Virtual Hosts

Adding a virtual host is as simple as selecting "Add Host." The page will reload with the information for the new host; simply change the name to be the domain name for the new host, and change the directory to be the correct directory for the virtual host. The Web server must be restarted for these settings to take effect.

In addition to the virtual hosts's domain and Web root, the details of all hosts (including the default host) can be changed by selecting "Edit Minor Details." From there, the log file, static files directory, password file, timeout values, and message files can all be modified. For details, see section 3.3.

3.3 Configuration File Syntax

Instead of using the configuration tool, as described in section 3.2, the configuration file can be edited "by hand". The provided, default configuration file is `collects/web-server/configuration-table`. The file has the following syntax.

```
((port natural-number)
 (max-waiting natural-number)
 (initial-connection-timeout number)
 (default-host-table host-table)
 (virtual-host-table (string host-table) ...))
```

`port : natural-number` Usually `80`, indicating which port to listen on by default.

`max-waiting : natural-number` Usually `40`, indicating the maximum number of clients that can wait for a TCP/IP connection to the Web server. This limit usually does not cause problems because the Web server quickly accepts connections as requested and there is no limit on the number of simultaneous connections. On some platforms, choosing a large number may consume too many resources.

`initial-connection-timeout : natural-number` Usually `30`. After a client connects to the Web server, the server only waits for an HTTP request for this many seconds before closing the connection. Browsers tend to send requests immediately, so a small number is best . This number should be kept small to prevent denial-of-service attacks.

`default-host-table : host-table` Described in section 3.3.1.

`virtual-host-table : (listof (cons string host-table))` A list of `(string host-table)`, where the `host-table` is defined in section 3.3.1 and the `string` represents the named virtual host's domain name.

For example, let the virtual host table be

```
(virtual-host-table
 ("www.plt-scheme.org"
  (host-table
   (default-indices "index.shtml" "index.html")
   (log-format parenthesized-default)
   (messages
    (servlet-message "servlet-error.html")
    (authentication-message "forbidden.html")
    (servlets-refreshed "servlet-refresh.html")
    (passwords-refreshed "passwords-refresh.html")
    (file-not-found-message "not-found.html")
    (protocol-message "protocol-error.html")
    (collect-garbage "collect-garbage.html"))
   (timeouts
    (default-servlet-timeout 120)
    (password-connection-timeout 300)
    (servlet-connection-timeout 86400)
    (file-per-byte-connection-timeout 1/20)
    (file-base-connection-timeout 30))
   (paths
    (configuration-root "errors")
    (host-root "/var/www/www.plt-scheme.org")
    (log-file-path "/var/log/www.plt-scheme.org")
    (file-root "htdocs")
    (servlet-root ".")
    (mime-types "mime.types")
    (password-authentication "passwd")))))
```

Requesting the URL `http://localhost/` will use the `default-host-table`, but requesting the URL `http://www.plt-scheme.org/`, assuming DNS is set up correctly, will return the first of

1. **/var/www/www.plt-scheme.org/index.shtml**
2. **/var/www/www.plt-scheme.org/index.html**

For port-based virtual hosting, run separate instances of the Web server.

3.3.1 Host Table Syntax

The host table is a part of the configuration file with the following syntax

```
(host-table
 (default-indices string ...)
 (log-format parenthesized-default)
 (messages
  (servlet-message string)
  (authentication-message string)
  (servlets-refreshed string)
  (passwords-refreshed string)
  (file-not-found-message string)
  (protocol-message string)
  (collect-garbage string))
 (timeouts
  (default-servlet-timeout number)
  (password-connection-timeout number)
  (servlet-connection-timeout number)
  (file-per-byte-connection-timeout number)
  (file-base-connection-timeout number))
 (paths
  (configuration-root string)
  (host-root string)
  (log-file-path string)
  (file-root string)
  (servlet-root string)
  (password-authentication string)))
```

`default-indices : (listof string)` This is a list of index files for all directories. When a directory is specified as the URL, such as `http://www.plt-scheme.org/`, the Web server searches for a file to display, choosing from this list in the specified order. A good default is

```
(default-indices "index.html" "index.htm")
```

For example, if `default-indices` is set to

```
(default-indices "index.ss" "index.html" "index.htm" "last-resort")
```

and the user requests `http://www.plt-scheme.org/`, the Web server will attempt to serve the following URLS, in order, until a file is found:

1. `http://www.plt-scheme.org/index.ss`
2. `http://www.plt-scheme.org/index.html`
3. `http://www.plt-scheme.org/index.htm`
4. `http://www.plt-scheme.org/last-resort`

`log-format : symbol` The format to use for the log file. Currently only the symbol `parenthesized-default` is supported.

3.3.1.1 MESSAGES

`servlet-message : string` A filename, usually `"servlet-error.html"`, of the page to display when there is an error with the servlet. This file is located relative to the `configuration-root` directory.

`authentication-message : string` A filename, usually `"forbidden.html"`, of the page to display when there user fails authentication. This file is located relative to the `configuration-root` directory.

`servlets-refreshed : string` A filename, usually `"servlet-refresh.html"`, of the page to display when the servlets are refreshed. This file is located relative to the `configuration-root` directory.

`passwords-refreshed : string` A filename, usually `"passwords-refresh.html"`, of the page to display when the password file is refreshed. This file is relative to the `configuration-root` directory.

`file-not-found-message : string` A filename, usually `"not-found.html"`, of the page to display when the requested file was not found (HTTP error number 404). This file is located relative to the `configuration-root` directory.

`protocol-message : string` A filename, usually `"protocol-error.html"`, of the page to display when the Web browser violates the HTTP standard. This should never appear when interacting with the server through a correct Web browser. The server may close the connection instead of sending this error, depending on the severity of the problem.

`collect-garbage : string` A filename, usually `"collect-garbage.html"`, of the page to display when the garbage collected is run. This file is relative to the `configuration-root` directory.

3.3.1.2 TIMEOUTS

`default-servlet-timeout : number` This value, usually `120`, determines how long, in seconds, a servlet can wait in between requests before shutting down, unless the servlet calls `adjust-timeout!` to alter this value. Large values consume more memory, while smaller values annoy end users who must restart servlets.

`password-connection-timeout : number` Usually `300`, this is the number of seconds a user has to enter a username and password.

`servlet-connection-timeout : number` Usually `86400` (one day), this is the number of seconds a servlet has to respond to a single request. Some servlets may perform substantial computation on a loaded server before generating a page. On the other hand, most users will not wait very long for a response.

`file-per-byte-connection-timeout : number` Usually `1/20`. Let *bytes* by the bytes in a file. The total number of seconds the server has to send a file is `file-base-connection-timeout` + `file-per-byte-connection-timeout` $*$ *bytes*

`file-base-connection-timeout : number` Usually `30`. Let *bytes* by the bytes in a file. The total number of seconds the server has to send a file is `file-base-connection-timeout` + `file-per-byte-connection-timeout` $*$ *bytes*

3.3.1.3 PATHS

`configuration-root : string` The directory under which the files in section 3.3.1.1 may be found. Usually `"conf"`.

`host-root : string` The root directory for Web files. Under this directory is usually the `servlet-root`, `configuration-root`, `log-file-path`, `file-root`, and `password-authentication`. Usually `"default-web-root"`.

`log-file-path : string` The name of the log file. Usually `"log"`.

`file-root : string` The directory from which static (HTML) files are served. Usually `"htdocs"`.

`servlet-root : string` The directory from which dynamic (servlet) files are run and served. Causes odd behavior when set to anything besides `"."`. The servlets directory is always `servlets/`, under the `host-root`.

`mime-types : string` The file from which MIME types are read. For information on its format, see Apache's documentation on the subject, http://httpd.apache.org/docs/1.3/mod/mod_mime.html#typesconfig. Usually `"mime.types"`.

`password-authentication : string` The file from which passwords are read. For information on its format, see section 4. Usually `"passwords"`.

4. Passwords

The PLT Web server provides *basic* authentication following RFC 2617. These passwords are stored in **passwords**. The Web server caches passwords for performance reasons. Requesting the URL `http://localhost/conf/refresh-passwords` reloads the password file. The format is:

```
'((realm path-regexp (name password) ...) ...)
```

`realm : string` A unique identifier for this set of passwords.

`path-regexp : string` A regular expression applied to the resource. If it matches, then this set of passwords is used for this resource.

`name : symbol` The login name for a user. Note that case-sensitivity depends on the current settings in MzScheme.

`password : string` The password, *unencrypted*, for the user.

For example, to hide anything in the `/secret/` directory under the `host-root`, but allow access to `bubba` with the password `bbq` and `Billy` with the password `BoB`, use

```
'(("secret stuff" "/secret(/.*)?" (bubba "bbq") (|Billy| "BoB")))
```

5. Starting the Server

The Web server collection provides two programs for starting the Web server. Both accept the same options.

web-server Servlets can use the full MrEd functionality.

web-server-text Servlets can only use MzScheme functionality. This server cannot load servlets written using DrScheme's graphical XML boxes, uses less memory, and works with the MzScheme-only distribution of PLT.

The command line

```
web-server[-text] [-p <port>] [-f <configuration-table-file>] [-a <ip-address>]
```

starts the server on port 80 or *port* and uses the configuration options from the **configuration-table** file of the **web-server** collection or from the specified *configuration-table-file*. If *ip-address* is provided, the server binds to *ip-address*.

6. Monitoring the Server

When launched via

```
web-server-monitor [-p <port>]
                   [-f <frequency>]
                   [-t <timeout>]
                   <alert-email> <host-name>
```

the **web-server-monitor** polls any Web server running on host *host-name* at port *port* (or port 80) every *frequency* seconds (or 1 hour). If the server does not respond to a HEAD HTTP request for the homepage within *timeout* (or 75) seconds or sends an error response, the monitor will notify *alert-email* of the problem.

Part II

Programming

7. Before You Begin

7.1 Reloading the Cache

For efficiency, servlets are cached on their initial invocation. When editing a servlet, before changes can be observed, it is necessary to clear the cache by loading the URL `http://localhost/conf/refresh-servlets`.

7.2 Directories

Within the Web server's root directory, two important directories exist. Note that these names may change. Administrators should see section 2 and section 3.3.1.3.

`htdocs/` Under this directory exists static files which are just presented to the Web browser with an appropriate MIME type. See section 8 for the details.

`servlets/` Under this directory exists dynamic files which are processed by the Web server as servlets. See section 9 for the details.

8. Static Content

The Web server serves content statically or dynamically, depending on the request URL.

On static requests (those that do not match the Web server's servlet filter), the Web server serves files out of the content directory determined from `file-root` for the specified virtual host.

The Web server guarantees that for such static responses, it will serve only those files accessible in subdirectories of the content directory. In the absence of filesystem links, this means that only files which live in the content directory will be made available to the outside world. The server ignores the relative path specifiers `..` and `.` in URLs.

9. Writing Servlets

When a request URL matches the servlet filter, the Web server generates its response dynamically, (see section 7.2 for more explanation, and section 2 and section 3.3.1.3 for administrative details). Instead of serving the files in this directory verbatim, the server evaluates the file as a scheme program to produce output. Servlets are (by default) loaded in a case-sensitive manner. (Search in help-desk for `read-case-sensitive`.)

The path part of the URL supplies the file path to the servlet relative to the `servlets` directory. However, paths may also contain extra path components that servlets may use as additional input. For example all of the following URLs refer to the same servlet:

- http://www.plt-scheme.org/servlets/my-servlet
- http://www.plt-scheme.org/servlets/my-servlet/extra
- http://www.plt-scheme.org/servlets/my-servlet/extra/directories

The server supports two forms of servlets. For useful procedures to handle Web data, see section 10.

9.1 Module-Based Servlets

A module-based servlet is a module that provides three values: an `interface-version`, a `timeout`, and a `start` procedure.

```
(module a-module-servlet mzscheme
  (provide interface-version timeout start)

  (define interface-version 'v1)

  (define timeout +inf.0)

  ; start : request → response
  (define (start initial-request)
    `(html (head (title "A Test Page"))
           (body ([bgcolor "white"])
                 (p "This is a simple module servlet."))))
```

`interface-version : symbol`

The `interface-version` is a symbol indicating how the server should interact with the servlet. The only supported value is `'v1` at this time.

`timeout : number`

The `timeout` is the number of seconds the server will allow the servlet to run before shutting it down. Large values consume more memory, while smaller values annoy users by forcing them to restart their session. The value can be adjusted dynamically by calling the `adjust-timeout!` procedure. For more information, see section 3.3.1.2 and section 10.2.

`start`

```
request -> response
(define (start request) ···)
```

The `start` function consumes a `request` and produces a `response`. Each time a client visits the URL associated with the beginning of the servlet, the server calls the `start` function with the `request` sent from the browser. The server then sends the `response` produced by the `start` function back to the browser.

9.2 Unit-Based Servlets

A unit-based servlet is a `unit/sig` that imports the `servlet^` signature and exports nothing. (See the manual for `unit/sig` and `signatures`.) To construct a signed unit with the appropriate imports, the servlet must require the two modules providing `unit/sig`s and the `servlet` signature:

```
(require (lib "unitsig.ss")
         (lib "servlet-sig.ss" "web-server"))
(unit/sig ()
  (import servlet^)
   ;;; ...insert servlet code here...
)
```

Evaluating

```
(require (lib "servlet-sig.ss" "web-server"))
```

loads the `servlet^` signature which contains the import `initial-request` of type `request`.

10. Servlet Library

To ease the development of interactive servlets, the **web-server** collection also provides the following data types and procedures:

10.1 Data Definitions

10.1.1 Environment

An `environment` is a `(listof (cons symbol string))`. That is, an `environment` is a `list` of improper `cons`-pairs of a `symbol` and a `string`.

A `byte-environment` is a `(listof (cons symbol bytes))`. That is, a `byte-environment` is a `list` of improper `cons`-pairs of a `symbol` and a `bytes`.

10.1.2 Request

A `request` is

```
(make-request symbol url environment environment string string)
(define-struct request (method uri headers bindings host-ip client-ip))
```

`method` One of

- `'get`
- `'post`

`uri` URL, see the **net** collection in help-desk for details.

`headers` An `environment` containing optional HTTP headers for this request.

`bindings` An `environment` containing optional name-value pairs from either the form submitted or the query part of the URL.

`bindings/raw` Either a string or a `byte-environment`. This is primarily used for file uploads.

`host-ip` A string representing what IP address of the server the request came to.

`client-ip` A string representing what IP address is associated with the client.

The bindings contain the information the user supplied when filling out a Web form and hence are usually the most useful part. This makes the following idiom common. For a full example, see Figure 10.2.

```
(extract-bindings/single
  'sym
  (request-bindings
    (send/suspend
      ; ...
      )))
```

10.1.3 Response

A `response` is one of the following:

an X-expression representing HTML Search for `XML` in help-desk. For example

```
'(html
  (head
    (title "Fruits")
    (link ((rev "made")
           (href "mailto:mburns@example.com")
           (title "Webmaster"))))
  (body
    (h1 "Fruits!")
    (ul
      (li "Banana")
      (li "Orange")
      (li "Grapefruit")
      (li "Shirley"))))
```

`(listof (union string bytes))` The first string is the MIME type (often `"text/html"`, but see RFC 2822 for other options). The rest of the strings provide the document's content.

`make-response/full`

`make-response/incremental`

`make-response/full`

`natural-number string natural-number string environment environment(listof (union string bytes)) -> response`
`(define (make-response/full` *`code message seconds mime extras body`*`) ···)`

`code` A natural number indicating the HTTP response code.

`message` A string describing the code to a human.

`seconds` A natural number indicating the time the resource was created. Use `(current-seconds)` for dynamically created responses.

`mime` A string indicating the response type.

`extras` An environment containing extra headers for redirects, authentication, or cookies.

`body (listof (union string bytes))`

`make-response/incremental`

`natural-number string natural-number string environment(((union string bytes) -> void) -> void) -> response`
`(define (make-response/incremental` *`code message seconds mime extras gen`*`) ···)`

`code`, `message`, `seconds`, `mime`, `extras` All the same as for `make-response/full`

`(gen output): ((union string bytes) -> void) -> void` The function `gen` consumes an `output` function. The `output` function consumes a string and sends it to the client. For HTTP/1.1 clients, the server uses the chunked encoding, which is reliable. HTTP/1.0 clients, however, can not distinguish between the end of the document and a lost connection. These facts have two implications. First, it is more efficient to send fewer, larger strings. Second, this response should not be used for data that must arrive reliably.

See also `make-html-response/incremental` in section 10.3.

10.2 Core Procedures

When writing module servlets, loading **servlet.ss** via

```
(require (lib "servlet.ss" "web-server"))
```

provides the following procedures:

send/suspend

```
(string -> response) -> request
(define (send/suspend page) ···)
```

The argument, a function that consumes a string, is given a URL that can be used in the document. The argument function must produce a response corresponding to the document's body. Requests to the given URL resume the computation at the point `send/suspend` was invoked. Thus, the argument function normally produces an HTML `form` with the `action` attribute set to the provided URL. The result of `send/suspend` represents the next request. This procedure is often used for interaction with the user. For an example, see section 10.4.

send/forward

```
(string -> response) -> request
(define (send/forward page) ···)
```

Acts like `send/suspend`, but clears the continuation table first. For example, if the servlet is an on-line quiz where the quiz-taker is not allowed to return to a previous page, use `send/forward`; this is how it is used in the example section 10.5.

send/finish

```
response -> void
(define (send/finish response) ···)
```

This provides a convenient way to report an error or otherwise produce a final response. Once called, all URLs generated by `send/suspend` become invalid. Calling `send/finish` allows the system to reclaim continuations used by the servlet. Often used as the final result of a servlet. For an example, see section 10.4.

send/back

```
response -> void
(define (send/back response) ···)
```

Acts like `send/finish`, but does not clear the continuation table. Useful for allowing the user to correct erroneous data.

To summarize the differences in the `send/` procedures, see Figure 10.1.

Type	Refreshes the continuation table	Does not refresh the continuation table
`response → void`	`send/finish`	`send/back`
`(string → response) → request`	`send/forward`	`send/suspend`

Figure 10.1: Differences in the send/ procedures

`adjust-timeout!`

```
natural-number -> void
(define (adjust-timeout! number) ···)
```

The server will stop an instance of a servlet after it has been idle for a certain amount of time, as described in the section 3.3.1.2. Calling `adjust-timeout!` allows the programmer to change the number of seconds before the servlet times out. Larger numbers consume more resources while smaller numbers force the user to restart computations more often.

Unit servlets receive these interaction procedures as `imports` through the `servlet^` signature.

10.3 Helpful Servlet Procedures

`extract-binding/single`

```
symbol environment -> string
(define (extract-binding/single sym environment) ···)
```

This extracts a single value associated with `sym` in the form bindings. If multiple or zero values are associated with the name, it raises an exception. For an example, see section 10.4.

`extract-bindings`

```
symbol environment -> (listof str)
(define (extract-bindings sym environment) ···)
```

Returns the list of values associated with the name `sym`.

`exists-binding?`

```
symbol environment -> bool
(define (exists-binding? sym environment) ···)
```

Returns true if the name `sym` is bound in the `environment`. This is useful for, e.g., checkboxes.

`extract-user-pass`

```
environment -> (union #f (cons str str))
(define (extract-user-pass environment) ···)
```

Servlets may implement password-based authentication by extracting password information from the HTTP headers. The return value is either a pair consisting of the username and password from the headers or `#f` if no password was provided. This only extracts the provided username and password; the servlet must perform any desired checking.

For more information on passwords, see section 4.

`report-errors-to-browser`

```
(response -> void) -> void
(define (report-errors-to-browser where) ···)
```

Calling this procedure at the beginning of a servlet causes otherwise uncaught exceptions to send an error page displaying the exception to the client. The argument should be based on `send/finish` if errors are fatal or `send/back` if the user may correct the failed form submission via the back button. For some applications, revealing the contents of an exception to a client may cause security problems by leaking sensitive information. For such applications, consider setting the `current-exception-handler` to email the error to the servlet author or store the exception in a log file.

`redirect-to`

```
string [symbol] -> response
(define (redirect-to url . redirection-status) ···)
```

Constructs a response that redirects to the given `url`. The optional argument specifies which kind of redirection to perform:

`permanently` Browsers should send future requests directly to this URL.

`temporarily` Browsers should send future requests to the original URL.

`see-other` The redirection is not replacing the current URL.

See the HTTP 1.1 specification for details on each kind of redirection. The default redirection type is `permanently`.

`make-html-response/incremental`

```
(((union string bytes) -> void) -> void) -> response/incremental
(define (make-html-response/incremental chunk-maker) ···)
```

This fills in default values for `make-response/incremental` to be appropriate for HTML.

Note that the function passed to this will be called in the connection thread, not the servlet thread. This means that e.g. `current-directory` will be the server's current directory, not the servlet's.

10.4 Example Multiplication Servlet

As an example of `send/suspend`, `send/finish`, and `request-bindings`, and `extract-bindings/single`, consider Figure 10.2. This servlet first prompts the user for the first number, then for the second number, and finally returns the product of the two numbers. Note that the final page is abstracted into a simple function, as is the common page to request a number.

For more example servlets, look in the `collects/web-server/default-web-root/sevlets/examples` directory.

```
(require (lib "unitsig.ss")
         (lib "servlet-sig.ss" "web-server"))

(unit/sig ()
  (import servlet^)

  ;; answer-page : Number → Xexpr
  (define (answer-page n)
    `(html
      (head
        (link ((rev "made") (href "mburns@example.com") (title "Webmaster"))))
      (body (h1 "Answer")
            (p ,(string-append "The answer is " (number→string n))))))

  ;; get-number : String → Number
  (define (get-number which)
    (string→number
     (extract-binding/single
      'num
      (request-bindings
       (send/suspend
        (get-number-page which))))))

  ;; get-number-page : String → String → Xexpr
  (define (get-number-page which)
    (lambda (k-url)
    `(html
      (head
        (link ((rev "made") (href "mburns@example.com") (title "Webmaster"))))
      (body (h1 "Enter a number")
        (form ((action ,k-url))
          (p ,(string-append "Please enter the " which " number"))
          (input ((type "text") (name "num") (id "num")))
          (input ((type "submit"))))))))

  (send/finish (answer-page (* (get-number "first") (get-number "second")))))
```

Figure 10.2: Example servlet using send/suspend, send/finish, request-bindings, and extract-binding/single

10.5 Example Math Test Servlet

The servlet in Figure 10.3 administers a simple math quiz to the user, keeping track of how many problems were answered correctly and incorrectly. As such, we don't want the user to simply hit the back button each time she enters a wrong answer; thus, we use `send/forward` to prevent against just that.

10.6 Servlet Development Environment

Choose "Add TeachPack..." from DrScheme's "Language" menu and select the **plt/teachpack/htdp/servlet.ss** teachpack. This provides functions for writing servlets including `send/suspend` and `send/finish`. All the extra servlet helper functions for extracting information from Web requests and building Web responses also become available through the TeachPack.

For information on **plt/teachpack/htdp/servlet2.ss**, see Extended Exercises.

The TeachPacks add the following functionality to the DrScheme environment:

- A Web browser is automatically started when needed.
- All the needed functions, described in section 10.2, as available.
- `send/suspend`, and the other needed functions, are available in the interactions window.

Currently, minor edits will need to be made to your existing servlets to use them with the TeachPacks.

```
(require (lib "unitsig.ss")
         (lib "servlet-sig.ss" "web-server"))

(unit/sig ()
  (import servlet^)

  ;; build-page : String Number Number Xexpr → Xexpr
  (define (build-page title num-wrong num-right body)
    `(html
      (head (title ,title)
            (link ((rev "made")
                   (href "mailto:netgeek@speakeasy.net")
                   (title "Webmaster"))))
      (body (h1 ,title)
            (p ,(string-append "Wrong: " (number→string num-wrong)))
            (p ,(string-append "Right: " (number→string num-right)))
            ,body)))

  ;; get-the-answer : Number Number Number Number → Number
  (define (get-the-answer num-wrong num-right a b)
    (string→number
     (extract-binding/single
      'answer
      (request-bindings
       (send/forward
        (get-the-answer-page num-wrong num-right a b))))))

  ;; get-the-answer-page : Number Number Number Number → String → Xexpr
  (define (get-the-answer-page num-wrong num-right a b)
    (lambda (k-url)
      (build-page
       "What's the answer?"
       num-wrong
       num-right
       `(form ((action ,k-url))
              (p ,(string-append (number→string a)
                                 " * "
                                 (number→string b)))
              (p (input ((type "text")
                         (id "answer")
                         (name "answer"))))
              (p (input ((type "submit"))))))))

  (let loop ((num-wrong 0)
             (num-right 0)
             (a 5)
             (b 7))
    (if (= (* a b) (get-the-answer num-wrong num-right a b))
      (loop num-wrong (+ num-right 1) (+ a 1) (+ b 1))
      (loop (+ num-wrong 1) num-right (+ a 1) (+ b 1)))))
```

Figure 10.3: Simple math test servlet, showing send/forward

11. Semi-Internal Functions

The following functions expose more of the Web server for use by the development environment. They are not intended for general use. They may change at anytime or disappear entirely. These are just for the developers.

11.1 Miscellaneous

`server-loop`

```
custodian (-> iport oport) (-> void) -> void
(define (server-loop custodian tcp-listener config initial-timeout) ···)
```

`custodian` The parent custodian for servlets.

`tcp-listener` Where requests arrive.

`config` Encapsulates most of the state of the server.

`initial-timeout` The number of seconds before timing out connections.

`make-config`

```
(string -> host) (hash-table-of sym script) (unit servlet-sig -> response)
(hash-table-of sym servlet-instance) -> config
(define (make-config hosts scripts instances access) ···)
```

host-table `string → host`

Maps host names to hosts

script-table `(hash-table-of sym script)`

Maps servlet names to servlet units

script `(unit servlet^ → response)`

Represents a servlet that is invoked on each request

instance-table `(hash-table-of sym servlet-instance)`

Maps the path part of a URL to the running servlet.

access-table `(hash-table-of sym (string sym string → (union \#f str)))`

Maps host names to functions that accept a protection domain, a user name, and a password and either return `#f` if access is not denied (i.e. is accessible) or a string prompting for a particular password (i.e. “Course Grades”).

servlet-instance `(make-servlet-instance nat channel (hash-table-of sym (continuation request)))`

The `natural-number` counts the continuations suspended by this servlet. The *channel* communicates HTTP requests from the connection thread to the suspended servlet. The *hash-table* maps parameter parts of URLs to suspended continuations.

11.2 Starting the Server from a Program

Requiring the library **web-server.ss**, via

```
(require (lib "web-server.ss" "web-server"))
```

provides the `serve` function, which starts the server with more configuration options.

`serve`

```
configuration [natural-number (union string #f)] -> (-> void)
(define (serve configuration . port ip-address) ···)
```

The `serve` function starts the Web server just like the launcher does, but the `configuration` argument supplies the server's settings. The optional `port` argument overrides the port supplied by the configuration. The optional `ip-address` binds the server to that address.

The result of invoking `serve` is a function of no arguments that shuts down the server.

`serve`

```
(opt->* (configuration?) ((and/f number? integer? exact? positive?) (union string? false?) (make-mixin-contract frame%)) ((-> void?) (any? . -> . (union false? string?)) (any? . -> . (union false? string?)) (any? . -> . string?) (-> (union false? (is-a?/c frame%))) (-> (is-a?/c frame%))))
(define (serve configuration . frame) ···)
```

In addition, this `serve` function accepts another optional argument. This argument is mixed into the `frame%` class created by the internal browser. The frame is then instantiated with one argument: the URL to visit.

11.3 Constructing Configurations

Constructing configurations requires another library.

```
(require (lib "configuration.ss" "web-server"))
```

`load-configuration`

```
string -> configuration
(define (load-configuration path) ···)
```

This function accepts a `path` to a configuration file and returns a configuration that `serve` accepts. The configuration tool can create configuration files, as explained in section 3.2. Configuration files can also be created by hand, as described in section 3.3.

11.4 Monitoring the Server

Requiring the library **monitor-web-server.ss**, via

```
(require (lib "monitor-web-server.ss" "web-server"))
```

provides the functions:

poke-web-server

```
channel string nat nat -> result
(define (poke-web-server channel server port timeout-seconds) ···)
```

The `poke-web-server` procedure takes a channel on which the results will come in, a server name to test, a port on that server to test, and a timeout (in seconds). The procedure returns immediately. The channel passed to the procedure will receive at least one result within (`timeout-seconds`$+\varepsilon$), for some reasonable ε.

A result is one of:

- `'(fail ,server-name ,server-port ,msg)`
- `'(timeout ,server-name ,server-port ,timeout-seconds)`
- `'(exn ,server-name ,server-port ,exn)`
- `'(ok)`

where `server-name` and `msg` are strings, `server-port` and `timeout-seconds` are numbers, and `exn` is an exception.

result-message

```
result -> string
(define (result-message result) ···)
```

The `result-message` procedure takes a result and produces a multi-line string suitable for inclusion in an error log or an email message.

11.4.1 Example

Here's a simple piece of code which checks the PLT download server:

```
(require (lib "monitor-web-server.ss" "web-server"))
(let ([result-channel (make-channel)]
      [server-name "download.plt-scheme.org"]
      [server-port 80])
    (poke-web-server result-channel server-name server-port 10)
    (let ([result (channel-get result-channel)])
      (match result
          ['(ok) (void)]
          [else (printf (result-message result))])))
```

To have this done automatically, see section 6 or Monit.

License

GNU Library General Public License

Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.

675 Mass Ave, Cambridge, MA 02139, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

   A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

   The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

   "Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a) The modified work must itself be a software library.
   b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
   c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
   d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

      (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

   In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

   Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

   This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

   If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

   However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

   When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

   If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

   Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

   You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

   a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
   b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
   c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
   d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

   For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

   a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
   b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

    If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

    It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

    This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Index

.., 14

adjust-timeout!, 20
authentication-message, 7

`byte-environment`, 17

collect-garbage, 7
configuration-root, 7
configuration-table, 10

default-host-table, 5
default-indices, 6
default-servlet-timeout, 7

`environment`, 17
exists-binding?, 20
extract-binding/single, 20
extract-bindings, 20
extract-user-pass, 20

file-base-connection-timeout, 7
file-not-found-message, 7
file-per-byte-connection-timeout, 7
file-root, 8

gen, 19

host-root, 7

initial-connection-timeout, 5
interface-version, 15

load-configuration, 26
log-file-path, 7
log-format, 6

make-config, 25
make-html-response/incremental, 21
make-response/full, 18
make-response/incremental, 18
max-waiting, 5
mime-types, 8

name, 9

password, 9
password-authentication, 8
password-connection-timeout, 7
passwords, 9
passwords-refreshed, 7

path-regexp, 9
poke-web-server, 27
port, 5
protocol-message, 7

realm, 9
redirect-to, 21
refresh-passwords, 9
refresh-servlets, 13
report-errors-to-browser, 21
`request`, 17
request-bindings, 17
`response`, 18
result-message, 27

send/back, 19
send/finish, 19
send/forward, 19
send/suspend, 19
serve, 26
server-loop, 25
servlet-connection-timeout, 7
servlet-message, 7
servlet-root, 8
`servlet^`, 16
servlets-refreshed, 7
start, 16

timeout, 16

virtual host, 5
virtual-host-table, 5

web-server-monitor, 11