httpd0 Manual

Max Rottenkolber

Friday, 16 September 2016

Table of Contents

See the *httpd0 API documentation* (`api.html`) for a complete API specification.

1 Using httpd0 to serve files

`make-resource-responder` is a file serving backend for httpd0. To serve files under the directory `#p"/var/foo/"` for instance, use `make-httpd` in combination with `make-resource-responder`.

```
(defvar *httpd*
  (make-httpd (make-resource-responder #p"/var/foo/")))
```

Our server instance is bound to `*httpd*`.

The HTTP server instance will now listen on `*:8080` (port 8080 on all hosts). To stop the server instance use `destroy-httpd`.

```
(destroy-httpd *httpd*)
```

Stop listening and free resources of `*httpd*`.

`make-httpd` can be configured by supplying keyword parameters. The parameters `:host` and `:port` specify on which host address the underlying socket will listens.

The special variables `*request-timeout*` and `*request-size*` control the maximum amount of time waited for a request to be sent and the maximum size of requests to be accepted respectively. The special variable `httpd0.responses:*text-mime*` controls the *MIME type* of error and directory listing responses.

The MIME type of individual files is determined by `file-types`[1]. Consult its respective documentation on how to extend the file extension database.

2 httpd0 concurrency

httpd0's concurrency model is based on a *thread pool*[2]. The `:n-threads` keyword parameter to `make-httpd` controls the number of threads used for responding to requests. On the socket level, the keyword parameter `:socket-backlog` to `make-httpd` sets the packet backlog size of the underlying socket.

3 Writing custom responders

`make-resource-responder`, as shown above, is a responder implementation for httpd0. Writing a custom responder is trivial, all the tools you need can be found in the `httpd0.responses` package.

A httpd0 responder is a function that will be called to handle every valid request. It has to accept two arguments, `resource` and `if-modified-since`. The `resource` argument will be a *pathname* representing the resource path of the request while `if-modified-since` will be the parsed date specified in the requests' `If-Modified-Since`[3] header or `nil` if none was specified. The `resource` argument will always be a relative pathname, beware that its `pathname-directory` may contain `:up`.

Consider this minimal example of a responder:

```
(defpackage example-responder
  (:use :cl :httpd0.responses :trivial-utf-8)
  (:export :responder))

(in-package :example-responder)

(defun responder (resource if-modified-since)
  (declare (ignore if-modified-since)) ; We don't use it.
  (cond

    ;; If REQUEST is #p"hello/<string>" respond with "Hello <string>!".
    ((equal '(:relative "hello") (pathname-directory resource))
     (let ((response (string-to-utf-8-bytes
                      (format nil "Hello ~a!~%"
                              (pathname-name resource)))))
       (respond-ok ((length response) *text-mime* (get-universal-time))
         (write-sequence response *standard-output*))))

    ;; Otherwise respond with status code 404.
    (t (respond-not-found))))
```

Definition of `example-responder`.

The responder defined above dispatches a plain text response for *URIs* that match the `/hello/*` pattern, all other requests are responded to with *status code* 404. Note how the body of `respond-ok` writes an *octet sequence* to `*standard-output*`. The call to `respond-ok` also expects the byte size of the response body, its MIME type and a date which designates the time the resource was last modified. For simplicity, the responder ignores the `if-modified-since` argument and always reports the current time as the last modified date.

To test the responder defined above launch an httpd0 instance and visit `http://localhost:8080/hello/Frank`:

```
(make-httpd #'example-responder:responder)
```

Using `example-responder`.

To summarize, the responder is required to:

- Use the `respond-*` routines from `httpd0.responses` (or roll their own).

- Supply the correct byte size of the response, the MIME type and a write date to `respond-ok`. The supplied write date must designate the date at which the requested resource was last modified.

- In the body of `respond-ok`, write the response body as octets to `*standard-output*`.

In return httpd0 will respond to GET and HEAD requests correctly. To view the complete set of `respond-*` routines consult the *httpd0 API* (`api.html`).

- 1. *file-types* (`/software/file-types/`)

- 2. *q-thread-pool* (`/software/q-thread-pool/`)

- 3. *If-Modified-Since* (`http://www.w3.org/Protocols/HTTP/1.0/spec.html#If-Modified-Since`)