**1. http.**

## 2. Include defintions.

⟨ include files 2 ⟩ ≡

```
#include "platform.h"
#include <microhttpd.h>
#ifdef HAVE_UNISTD_H
#include <unistd.h>
#endif
#ifdef HAVE_SYS_STAT_H
#include <sys/stat.h>
#endif
#ifdef HAVE_FCNTL_H
#include <fcntl.h>
#endif
```

This code is used in sections 4 and 5.

**3. declarations.**

⟨ declarations of functions 3 ⟩ ≡

```
enum MHD_Result ahc_echo(void *cls, struct MHD_Connection *connection, const char *url, const
    char *method, const char *version, const char *upload_data, size_t *upload_data_size, void
    **ptr);
```

This code is used in sections 4 and 5.

**4. main.**

```
⟨ include files 2 ⟩
⟨ declarations of functions 3 ⟩
int main(int argc, char *const *argv)
{
  struct MHD_Daemon *d;
  if (argc ≠ 2) {
    printf("%s PORT\n", argv[0]);
    return 1;
  }
  unsigned int flags = MHD_USE_THREAD_PER_CONNECTION;
  flags |= MHD_USE_INTERNAL_POLLING_THREAD;
  flags |= MHD_USE_ERROR_LOG;
  d = MHD_start_daemon(flags,
      atoi(argv[1]),
      Λ, Λ,      /* accept policy callback */
  &ahc_echo, Λ,      /* access handler callback */
  MHD_OPTION_END);
  if (d ≡ Λ) return 1;
  (void) getc(stdin);
  MHD_stop_daemon(d);
  return 0;
}
```

**5. processing.**

```
⟨ dummy.c 5 ⟩ ≡
  ⟨ include files 2 ⟩
  ⟨ declarations of functions 3 ⟩
  enum MHD_Result ahc_echo(void *cls, struct MHD_Connection *connection, const char *url, const
          char *method, const char *version, const char *upload_data, size_t *upload_data_size, void
          **ptr)
  {
    static char *page = "{\"data\":1}";
    static int aptr;
    struct MHD_Response *response;
    enum MHD_Result ret;
    int fd;
    struct stat buf;
    (void) cls;      /* Unused. Silent compiler warning. */
    (void) version;      /* Unused. Silent compiler warning. */
    (void) upload_data;      /* Unused. Silent compiler warning. */
    (void) upload_data_size;      /* Unused. Silent compiler warning. */
    fprintf(stderr, "ECHO␣url:%s\n␣method:%s\n", url, method);
    if ((0 ≠ strcmp(method, MHD_HTTP_METHOD_GET)) ∧ (0 ≠ strcmp(method, MHD_HTTP_METHOD_HEAD)))
      return MHD_NO;      /* unexpected method */
    response = MHD_create_response_from_buffer(strlen(page), (void *) page, MHD_RESPMEM_PERSISTENT);
    ret = MHD_queue_response(connection, MHD_HTTP_NOT_FOUND, response);
    MHD_destroy_response(response);
    return ret;
  }
```

**6.**

```
⟨ initialize request local data 6 ⟩ ≡
  if (&aptr ≠ *ptr) {      /* do never respond on first call */
    *ptr = &aptr;
    return MHD_YES;
  }
```

## 7. INDEX.

⟨ declarations of functions 3 ⟩ Used in sections 4 and 5.
⟨ `dummy.c` 5 ⟩
⟨ include files 2 ⟩ Used in sections 4 and 5.
⟨ initialize request local data 6 ⟩

# HTTP