code/netp/tiny/tiny.c

```
1  void serve_static(int fd, char *filename, int filesize)
2  {
3      int srcfd;
4      char *srcp, filetype[MAXLINE], buf[MAXBUF];
5
6      /* Send response headers to client */
7      get_filetype(filename, filetype);
8      sprintf(buf, "HTTP/1.0 200 OK\r\n");
9      sprintf(buf, "%sServer: Tiny Web Server\r\n", buf);
10     sprintf(buf, "%sConnection: close\r\n", buf);
11     sprintf(buf, "%sContent-length: %d\r\n", buf, filesize);
12     sprintf(buf, "%sContent-type: %s\r\n\r\n", buf, filetype);
13     Rio_writen(fd, buf, strlen(buf));
14     printf("Response headers:\n");
15     printf("%s", buf);
16
17     /* Send response body to client */
18     srcfd = Open(filename, O_RDONLY, 0);
19     srcp = Mmap(0, filesize, PROT_READ, MAP_PRIVATE, srcfd, 0);
20     Close(srcfd);
21     Rio_writen(fd, srcp, filesize);
22     Munmap(srcp, filesize);
23 }
24
25 /*
26  * get_filetype - Derive file type from filename
27  */
28 void get_filetype(char *filename, char *filetype)
29 {
30     if (strstr(filename, ".html"))
31         strcpy(filetype, "text/html");
32     else if (strstr(filename, ".gif"))
33         strcpy(filetype, "image/gif");
34     else if (strstr(filename, ".png"))
35         strcpy(filetype, "image/png");
36     else if (strstr(filename, ".jpg"))
37         strcpy(filetype, "image/jpeg");
38     else
39         strcpy(filetype, "text/plain");
40 }
```

code/netp/tiny/tiny.c

图 11-34　TINY serve_static 为客户端提供静态内容

接着，我们将被请求文件的内容复制到已连接描述符 fd 来发送响应主体。这里的代码是比较微妙的，需要仔细研究。第 18 行以读方式打开 filename，并获得它的描述符。在第 19 行，Linux mmap 函数将被请求文件映射到一个虚拟内存空间。回想我们在第 9.8 节中对 mmap 的讨论，调用 mmap 将文件 srcfd 的前 filesize 个字节映射到一个从地址 srcp 开始的私有只读虚拟内存区域。