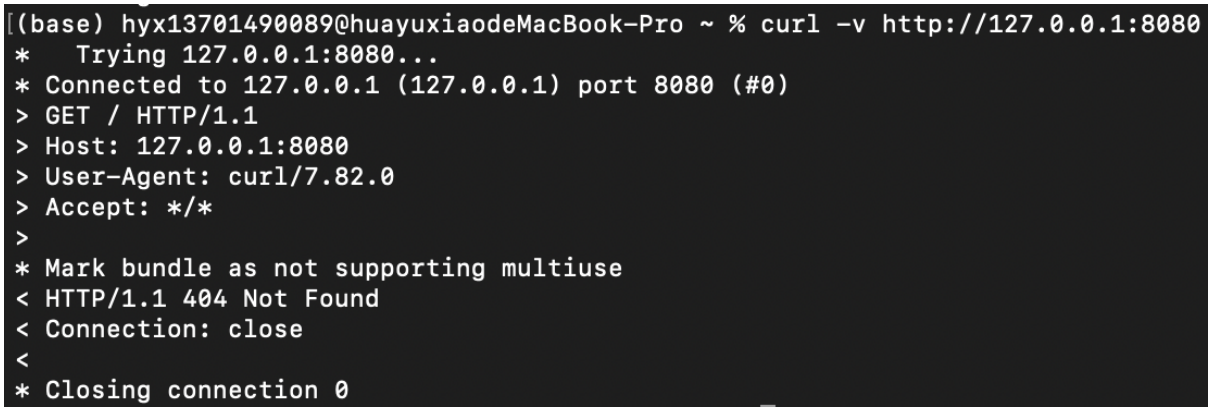
CS305 2022 Fall Assignment 1 - HTTP Server

12010508华羽霄

Task 1: HTTP Message encapsulation and de-encapsulation

```
def default_handler(server: HTTPServer, request: HTTPRequest, response: HTTPResponse):
    response.status_code, response.reason = 404, 'Not Found'
```

Run the server with command python3 main.py and run curl -v http://127.0.0.1:8080 **in a new terminal**. After that, you should get 404 Not Found as intended.



Task 2: Basic Static Content Server

```
def task2_data_handler(server: HTTPServer, request: HTTPRequest, response: HTTPResponse):
    if request.method == 'GET':
        try:
            with open('.' + request.request_target, 'rb') as f:
                response.status_code, response.reason = 200, 'OK'
                response.body = f.read()
                # set the Content-Type and Content-Length in Response Headers properly
                response.add_header('Content-Type', mimetypes.guess_type(request.request_target)[0])
                response.add_header('Content-Length' , str(len(response.body)))
                f.close()
        except FileNotFoundError :
            response.status_code, response.reason = 404, 'Not Found'
    elif request.method == 'HEAD':
        try:
            with open('.' + request.request_target, 'rb') as f:
                response.status_code, response.reason = 200, 'OK'
                # set the Content-Type and Content-Length in Response Headers properly
                response.add_header('Content-Type', mimetypes.guess_type(request.request_target)[0])
                response.add_header('Content-Length' , str(len(f.read())))
                f.close()
        except FileNotFoundError :
```

```
            response.status_code, response.reason = 404, 'Not Found'
    pass
```

If clients GET for http://127.0.0.1:8080/data/index.html , the server should respond with the content of file data/index.html .

```
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/data/index.html
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET /data/index.html HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Connection: close
< Content-Type: text/html
< Content-Length: 225
<
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>

<img src="test.jpg"  alt="" style="width: 500px;"/>
<p>HTML test</p>

<script src="main.js"></script>
</body>
* Closing connection 0
</html>%
```

If clients GET for a non- existence file, such as http://127.0.0.1:8080/data/nosuchfile , the server should respond HTTP 404 Not Found.

```
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/data/nosuchfile
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET /data/nosuchfile HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 404 Not Found
< Connection: close
<
* Closing connection 0
```

Task 3: Handle POST Request

```
def read_message_body(self) -> bytes:
        result: str = "{"
        message = self.buffer.decode()
        # 对信息预处理
        message = message.replace("{","")
        message = message.replace("}","")
        message = message.replace(" ","")
```

```
        message = message.split(",")

        for each in message:
            sub_msg = each.split(":")
            if each != message[-1]:
                result+=f"'{sub_msg[0]}':'{sub_msg[1]}', "
            else:
                result+=f"'{sub_msg[0]}':'{sub_msg[1]}'" + "}"
        result = result.replace('\'', '"')
        return result
```

```
def task3_json_handler(server: HTTPServer, request: HTTPRequest, response: HTTPResponse):
    response.status_code, response.reason = 200, 'OK'
    if request.method == 'POST':
        binary_data = request.read_message_body()
        obj = json.loads(binary_data)
        server.task3_data = str(obj['data'])
        real_data = ('{'+ f"'data': '{server.task3_data}'" + '}').replace('\'', '"')
        with open('.\\post','w') as f:
            f.write(real_data)
        pass
```

When the client GETs http://127.0.0.1:8080/post , you should return the stored data from **last POST**.

```
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/post --data '{"data":"test", "junk":"ignore"}'
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> POST /post HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
> Content-Length: 32
> Content-Type: application/x-www-form-urlencoded
>
* Empty reply from server
* Closing connection 0
curl: (52) Empty reply from server
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/post --get
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET /post HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Connection: close
< Content-Type: application/json
< Content-Length: 12
<
* Closing connection 0
{"data": ""}%
```

Task 4: HTTP 302 Found: URL Redirection

```
def task4_url_redirection(server: HTTPServer, request: HTTPRequest, response: HTTPResponse):
    response.status_code, response.reason = 302, 'Found'
    response.add_header('Location', 'http://127.0.0.1:8080/data/index.html')
    pass
```

```
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/redirect
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET /redirect HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 302 Found
< Connection: close
< Location: http://127.0.0.1:8080/data/index.html
<
* Closing connection 0
```

Task 5: HTTP Cookie and Session

Cookie

```
def task5_cookie_login(server: HTTPServer, request: HTTPRequest, response: HTTPResponse):
    obj = json.loads(request.read_message_body())
    if obj['username'] == 'admin' \
        and obj['password'] == 'admin':
        response.status_code, response.reason = 200, 'OK'
        response.add_header('Set-Cookie', 'Authenticated=yes')
        pass
    else:
        response.status_code, response.reason = 403, 'Forbidden'
        pass
```

```
def task5_cookie_getimage(server: HTTPServer, request: HTTPRequest, response: HTTPResponse):
    if request.method == 'GET':
        for each in request.headers:
            if f'{each.name}' == 'Cookie' and \
                f'{each.value}' == 'Authenticated=yes':
                response.status_code, response.reason = 200, 'OK'
                with open('.\\data\\test.jpg', 'rb') as f :
                    response.body = f.read()
                    # set the Content-Type and Content-Length in Response Headers properly
                    response.add_header('Content-Type', mimetypes.guess_type('.\\data\\test.jpg')[0])
                    response.add_header('Content-Length' , str(len(response.body)))
            else:
                response.status_code, response.reason = 403, 'Forbidden'
    elif request.method == 'HEAD':
         for each in request.headers:
            if f'{each.name}' == 'Cookie' and \
                f'{each.value}' == 'Authenticated=yes':
                response.status_code, response.reason = 200, 'OK'
                with open('.\\data\\test.jpg', 'rb') as f :
                    # set the Content-Type and Content-Length in Response Headers properly
                    response.add_header('Content-Type', mimetypes.guess_type('.\\data\\test.jpg')[0])
                    response.add_header('Content-Length' , str(len(f.read())))
            else:
                response.status_code, response.reason = 403, 'Forbidden'
    pass
```

这道题做得有点问题。

```
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/api/login --data '{"username":"admin", "password":"admin"}'
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> POST /api/login HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
> Content-Length: 40
> Content-Type: application/x-www-form-urlencoded
>
* Empty reply from server
* Closing connection 0
curl: (52) Empty reply from server
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/api/getimage
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET /api/getimage HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 403 Forbidden
< Connection: close
<
* Closing connection 0
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/api/getimage --header "Cookie:Authenticated=yes"
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET /api/getimage HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
> Cookie:Authenticated=yes
>
* Empty reply from server
* Closing connection 0
curl: (52) Empty reply from server
```

Session

```
def task5_session_login(server: HTTPServer, request: HTTPRequest, response: HTTPResponse):
    obj = json.loads(request.read_message_body())
    if obj['username'] == 'admin' \
        and obj['password'] == 'admin':
        response.status_code, response.reason = 200, 'OK'
        session_key = random_string()
        while session_key in server.session:
            session_key = random_string()
        pass
        server.session = {'SESSION_KEY=':f'{session_key}'}
        response.add_header('Set-Cookie', 'SESSION_KEY='+ f'{session_key}')
    else:
        response.status_code, response.reason = 403, 'Forbidden'
```

```
def task5_session_getimage(server: HTTPServer, request: HTTPRequest, response: HTTPResponse):
    if request.method == 'GET':
        for each in request.headers:
            if f'{each.name}' == 'Cookie' \
                and f'{each.value}' == 'SESSION_KEY='+ server.session['SESSION_KEY=']:
                response.status_code, response.reason = 200, 'OK'
                with open('.\\data\\test.jpg', 'rb') as f :
                    response.body = f.read()
                    # set the Content-Type and Content-Length in Response Headers properly.
                    response.add_header('Content-Type', mimetypes.guess_type('.\\data\\test.jpg')[0])
                    response.add_header('Content-Length' , str(len(response.body)))
            else:
                response.status_code, response.reason = 403, 'Forbidden'
    elif request.method == 'HEAD':
         for each in request.headers:
            if f'{each.name}' == 'Cookie' \
                and f'{each.value}' == 'SESSION_KEY='+ server.session['SESSION_KEY=']:
                response.status_code, response.reason = 200, 'OK'
                with open('.\\data\\test.jpg', 'rb') as f :
                    # set the Content-Type and Content-Length in Response Headers properly.
                    response.add_header('Content-Type', mimetypes.guess_type('.\\data\\test.jpg')[0])
                    response.add_header('Content-Length' , str(len(f.read())))
            else:
                response.status_code, response.reason = 403, 'Forbidden'
    pass
```

这道题做得有点问题。

```
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/apiv2/getimage
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET /apiv2/getimage HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 403 Forbidden
< Connection: close
<
* Closing connection 0
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/apiv2/login --data '{"username":"admin", "password":"admin"}'
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> POST /apiv2/login HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
> Content-Length: 40
> Content-Type: application/x-www-form-urlencoded
>
* Empty reply from server
* Closing connection 0
curl: (52) Empty reply from server
(base) hyx13701490089@huayuxiaodeMacBook-Pro ~ % curl -v http://127.0.0.1:8080/apiv2/getimage --header "Cookie:SESSION_KEY=DJK5LAFTY8NEQN6JNRIA"
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET /apiv2/getimage HTTP/1.1
> Host: 127.0.0.1:8080
> User-Agent: curl/7.82.0
> Accept: */*
> Cookie:SESSION_KEY=DJK5LAFTY8NEQN6JNRIA
>
* Empty reply from server
* Closing connection 0
curl: (52) Empty reply from server
```