

# Simple Java Socket HTTP

一个纯 Java Socket 实现的迷你 HTTP 服务器与客户端（含 Swing GUI 客户端），通过底层 Socket 通信完整模拟 HTTP 协议的核心功能。

- 已部署于华为云服务器

后续演示图片是基于本地部署，访问云服务器将url替换为140.210.142.61:8002即可

## 项目概述

本项目是一个 HTTP 服务器和客户端实现，不依赖任何第三方框架，完全使用 Java 标准库实现，涵盖了 HTTP 协议的核心概念，包括请求解析、响应构建、状态码处理、会话管理、文件上传等功能。

## 技术特点

- **纯 Java 实现**: 仅使用 Java 标准库，无第三方依赖
- **多线程架构**: 使用线程池处理并发连接
- **完整 HTTP 支持**: 支持 HTTP/1.1 协议核心功能
- **GUI 客户端**: 基于 Swing 的图形化测试界面
- **项目CI/CD**: 基于GitAction

## HTTP 协议基础

HTTP (超文本传输协议) 是基于请求-响应模式的应用层协议，主要由以下部分组成：

- **请求 (Request)**

- 请求行：包含方法 (GET/POST等) 、路径、协议版本 (如 GET /index.html HTTP/1.1)
- 头部 (Headers) : 键值对形式的元数据 (如 Host: localhost:8080、Content-Type: text/html)
- 空行：分隔头部和体部
- 体部 (Body) : 可选，用于 POST 等方法传递数据 (如表单参数)

- **响应 (Response)**

- 状态行：包含协议版本、状态码、原因短语 (如 HTTP/1.1 200 OK)
- 头部 (Headers) : 与请求头部格式一致 (如 Content-Length: 1024)
- 空行：分隔头部和体部
- 体部 (Body) : 服务器返回的数据 (如 HTML 内容、图片二进制数据)

- **GET 请求示例**

```
GET /hello.txt HTTP/1.1
Host: localhost:8080
User-Agent: SimpleSocketClient/1.0
Connection: keep-alive
```

- 200 响应示例

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8
Content-Length: 30
Server: SimpleSocketServer/1.0
Connection: keep-alive
```

Hello Socket HTTP!  
这是一个纯文本文件

- 301 重定向响应示例

```
HTTP/1.1 301 Moved Permanently
Location: /new
Content-Length: 0
Connection: keep-alive
```

◦ 测试方式：访问<http://localhost:8080/old>资源

- 302 临时重定向响应示例

```
HTTP/1.1 302 Found
Location: /temp-redirect
Content-Length: 0
Server: SimpleSocketServer/1.0
Connection: keep-alive
```

◦ 测试方式：访问<http://localhost:8080/temp>资源

- 304 未修改响应示例

```
HTTP/1.1 304 Not Modified
ETag: "abc123"
Date: Mon, 15 Nov 2023 10:00:00 GMT
Server: SimpleSocketServer/1.0
Connection: keep-alive
```

◦ 测试方式：重复请求相同资源（已经缓存）

- 401 未授权响应示例

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Basic realm="Restricted Area"
```

```
Content-Type: text/plain; charset=UTF-8
Content-Length: 20
Server: SimpleSocketServer/1.0
Connection: keep-alive
```

需要身份验证访问

- **404 未找到响应示例**

```
HTTP/1.1 404 Not Found
Content-Type: text/plain; charset=UTF-8
Content-Length: 25
Server: SimpleSocketServer/1.0
Connection: keep-alive
```

请求的资源不存在

- **405 方法不允许响应示例**

```
HTTP/1.1 405 Method Not Allowed
Allow: GET, POST
Content-Type: text/plain; charset=UTF-8
Content-Length: 25
Server: SimpleSocketServer/1.0
Connection: keep-alive
```

不支持的HTTP请求方法

- 测试方式：使用没有实现的DELETE方法

- **409 User重复注册**

```
HTTP/1.1 409 Conflict
Content-Type: text/plain; charset=UTF-8
Content-Length: 44
Connection: keep-alive
```

注册失败(可能已存在或参数错误)

- **422 上传文件非法命名**

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: text/plain; charset=UTF-8
Content-Length: 15
Connection: keep-alive
```

非法文件名

- **500 服务器内部错误响应示例**

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/plain; charset=UTF-8
Content-Length: 35
Server: SimpleSocketServer/1.0
Connection: keep-alive
```

服务器内部错误，请稍后重试

## 项目结构与文件功能

```
src/main/java/com/example/http/
├── Main.java
├── SimpleHttpServer.java
├── SimpleHttpWorker.java
├── HttpClientGui.java
└── http/
    ├── HttpRequest.java
    └── HttpResponse.java
└── user/
    └── UserService.java

src/main/resources/public/
local
```

## 核心文件清单

文件路径	功能概述
Main.java	程序入口，解析命令行参数并启动服务器/客户端
SimpleHttpServer.java	HTTP 服务器主类，负责监听端口、管理连接和线程池
SimpleHttpWorker.java	处理单个客户端连接的请求，解析请求并生成响应
http/HttpRequest.java	封装 HTTP 请求，解析方法、路径、头部、表单参数等
http/HttpResponse.java	构建 HTTP 响应，包含状态码、头部和体部，支持序列化
HttpClientGui.java	客户端 GUI 实现，支持发送请求、处理响应和重定向
src/main/resources/public/	静态资源目录 (HTML、图片、文本文件等)
local	客户端的资源 (HTML、图片、文本文件等)

## 详细功能说明

### 1. Main.java

程序入口类，负责解析命令行参数并启动对应模式（服务器/客户端）。

- **核心函数**

- `main(String[] args)`: 主入口，根据参数判断启动服务器或客户端
- `startServerMode(String[] args)`: 启动服务器模式，解析端口号（默认8080）
- `startClientMode(String[] args)`: 启动客户端 GUI，支持指定服务器端口
- `parsePortNumber(String portStr)`: 解析端口号，验证范围（1-65535）

### 2. SimpleHttpServer.java

HTTP 服务器主类，基于 `ServerSocket` 监听端口，使用线程池处理并发连接。

- **核心函数**

- `start()`: 启动服务器，循环接受客户端连接
- `handleNewConnection(Socket clientSocket)`: 处理新连接，配置 `Socket` 并提交到线程池
- `configureSocket(Socket socket)`: 配置 `Socket` 选项（超时、TCP 无延迟等）
- `stop()`: 停止服务器，关闭线程池

### 3. SimpleHttpWorker.java

处理单个客户端连接的工作类（实现 `Runnable`），负责解析 HTTP 请求并生成响应。

- **核心函数**

- `run()`: 线程入口，循环处理连接上的多个请求（支持 Keep-Alive）
- `parseRequest(InputStream in)`: 解析请求行、头部和体部，生成 `HttpRequest` 对象
- `processRequest(HttpRequest request)`: 处理请求，路由到对应处理器（静态资源、注册/登录等）
- `sendResponse(OutputStream out, HttpResponse response, boolean keepAlive)`: 发送响应到客户端
- `handle(HttpRequest req)`: 核心路由逻辑，处理静态资源、重定向、注册/登录、文件上传等

### 4. http/HttpRequest.java

封装 HTTP 请求的工具类，提供便捷的请求信息访问方法。

- **核心函数**

- `setStartLine(String method, String uri, String version)`: 设置请求行，拆分路径和查询字符串
- `addHeader(String name, String value)`: 添加请求头部（自动转为小写键）
- `headerFirst(String name)`: 获取指定头部的第一个值
- `setBody(byte[] body)`: 设置请求体，自动解析表单参数（`application/x-www-form-urlencoded`）
- `form(String key)`: 获取表单参数值
- `cookie(String name)`: 获取 Cookie 值

## 5. `http/HttpResponse.java`

构建 HTTP 响应的工具类，支持链式调用，负责生成响应字节流。

- **核心函数**

- `status(HttpStatus status)`: 设置响应状态码（链式调用）
- `header(String name, String value)`: 添加响应头部（链式调用）
- `bodyText(String text, String contentType)`: 设置文本响应体，自动处理编码和长度
- `toBytes(boolean keepAlive)`: 将响应转换为字节数组，包含状态行、头部和体部

## 6. `HttpClientGui.java`

Swing 实现的客户端 GUI，支持发送 GET/POST 请求，处理重定向、Cookie 和缓存。

- **核心函数**

- `sendHttpWithBytes(...)`: 发送 HTTP 请求，返回头部和体部字节数据
- 处理逻辑：
  - 解析 URL 并建立 Socket 连接（支持连接池复用）
  - 构建请求头和体部，发送到服务器
  - 解析响应，处理重定向（301/302）、缓存（304）和 Cookie
  - 根据服务器指令关闭或复用连接

## 7. 静态资源文件

- `index.html`: 默认首页，包含测试链接（文本、图片、重定向示例）
- `hello.txt`: 纯文本测试文件，验证 text/plain MIME 类型
- `test.png`: 图片文件，验证二进制文件传输

# 功能特性

## HTTP 服务器

- **多线程处理**: 使用线程池处理并发连接，高效稳定。
- **HTTP 协议支持**: 支持 HTTP/1.1 协议，包括长连接（Keep-Alive）。
- **静态资源服务**: 支持 HTML、JSON、文本等静态文件的访问。
- **RESTful 风格**: 支持 GET、POST 等基本请求方法。
- **无依赖**: 仅使用 Java 标准库，无任何第三方依赖。

## HTTP 客户端 (GUI)

- **图形化界面**: 基于 Swing 的直观操作界面。
- **请求构建**: 支持自定义 HTTP 方法、路径、Header 和 Body。
- **连接池**: 内置连接池管理，支持复用 TCP 连接。
- **实时日志**: 清晰展示请求报文和响应报文详情。

# 快速开始

## 环境要求

- Java 17 或更高版本

- Maven 3.8 及以上 (推荐 3.9.x)

## 编译打包

```
# 清理并编译项目  
mvn clean package
```

## 运行方式

### 启动 HTTP 服务器

```
# 使用默认端口 8080 启动服务器  
java -jar target/simple-http-socket-1.0-SNAPSHOT.jar server  
  
# 指定端口启动服务器  
java -jar target/simple-http-socket-1.0-SNAPSHOT.jar server 9090
```

### 启动 GUI 客户端

```
# 启动客户端 (默认连接 localhost:8080)  
java -jar target/simple-http-socket-1.0-SNAPSHOT.jar client  
  
# 指定服务器端口启动客户端  
java -jar target/simple-http-socket-1.0-SNAPSHOT.jar client 9090
```

## 使用示例

启动服务器后，可以通过以下方式测试：

1. 浏览器访问：打开浏览器访问 <http://localhost:8080>
2. GUI 客户端：启动客户端进行图形化操作

## 功能演示

以下为主要功能的演示截图与说明，采用折叠分组，阅读更集中：

### 1. 注册

- 说明：可用 GUI 的快捷按钮，也可在资源路径手动构造 POST 请求体。

#### ► 展开查看注册演示

- 正常注册：

The screenshot shows a network testing interface with the following details:

- Method:** POST
- URL:** calhost:8080/register
- Form Fields:** 用户 (User) and 密码 (Password) are empty. Buttons for 注册 (Register), 登录 (Login), and 注销 (Logout) are visible. The status is 未登录 (Not Logged In).
- Request Body (POST Data):** username=role1&password=123456
- HTTP Request Log:**

```
POST /register HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: */*
Content-Type: application/x-www-form-urlencoded
Content-Length: 30
Connection: keep-alive

username=role1&password=123456
```
- HTTP Response Log:**

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8
Content-Length: 12
Connection: keep-alive

注册成功
```

- 重复注册 (409 Conflict) :

The screenshot shows a network testing interface with the following details:

- Method:** POST
- URL:** calhost:8080/register
- Form Fields:** 用户 (User) and 密码 (Password) are empty. Buttons for 注册 (Register), 登录 (Login), and 注销 (Logout) are visible. The status is 未登录 (Not Logged In).
- Request Body (POST Data):** username=role1&&password=123
- HTTP Request Log:**

```
POST /register HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: */*
Content-Type: application/x-www-form-urlencoded
Content-Length: 28
Connection: keep-alive

username=role1&&password=123
```
- HTTP Response Log:**

```
HTTP/1.1 409 Conflict
Content-Type: text/plain; charset=UTF-8
Content-Length: 44
Connection: keep-alive

注册失败(可能已存在或参数错误)
```

## 2. 登录

- 说明：同样支持 GUI 快捷按钮与手动 POST。未授权场景返回 401。
- 展开查看登录演示

- 登录请求：

The screenshot shows a network traffic capture interface. At the top, it displays the method as POST, URL as http://localhost:8080/login, and a status bar indicating '已登录' (Logged In). Below this, there are input fields for '用户名' (User) and '密码' (Password), and buttons for '注册' (Register), '登录' (Login), and '注销' (Logout). A message 'username=role1&password=123456' is shown in the '请求体 (POST数据)' (POST Data Body) section.

**HTTP请求报文**

```
POST /login HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: */*
Content-Type: application/x-www-form-urlencoded
Content-Length: 30
Connection: keep-alive

username=role1&password=123456
```

**HTTP响应报文**

```
HTTP/1.1 200 OK
Set-Cookie: SID=e2e86b5a-eaf6-45f6-8338-f9d8d875613c; Path=/; HttpOnly
Content-Type: text/plain; charset=UTF-8
Content-Length: 12
Connection: keep-alive

登录成功
```

- 401 未授权：

The screenshot shows a network traffic capture interface. At the top, it displays the method as POST, URL as http://localhost:8080/login, and a status bar indicating '未登录' (Not Logged In). Below this, there are input fields for '用户名' (User) and '密码' (Password), and buttons for '注册' (Register), '登录' (Login), and '注销' (Logout). A message 'username=role1&password=123456999' is shown in the '请求体 (POST数据)' (POST Data Body) section.

**HTTP请求报文**

```
POST /login HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: */*
Content-Type: application/x-www-form-urlencoded
Content-Length: 33
Connection: keep-alive

username=role1&password=123456999
```

**HTTP响应报文**

```
HTTP/1.1 401 Unauthorized
Content-Type: text/plain; charset=UTF-8
Content-Length: 38
Connection: keep-alive

登录失败(用户名或密码错误)
```

### 3. 获取服务器资源 (GET)

- 说明：首次 local (客户端) 目录为空，服务端资源包含 index.html 等, GET到的资源会存入客户端也即/local文件夹下。
- ▶ 展开查看 GET 与重定向/缓存/错误演示
- 200 OK (根路径默认返回 index.html) :

The screenshot shows a web-based interface for managing a simple socket server. At the top, there's a header with 'Method: GET', 'URL: http://localhost:8080', and a login form for '用户名' (role1) and '密码' (.....). Below the header, there are buttons for '注册' (Register), '登录' (Login), and '注销' (Logout). A message '已登录:role1' indicates the user is logged in.

**请求报文 (Request Message):**

```
GET / HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: /*
Cookie: SID=9e34ba2b-a89e-456b-9886-007fa712b291
Connection: keep-alive
```

**响应报文 (Response Message):**

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Date: Fri, 28 Nov 2025 14:19:00 CST
Last-Modified: Wed, 19 Nov 2025 19:16:59 CST
Server: SimpleSocketServer/1.0
Connection: keep-alive
Content-Length: 479

<!DOCTYPE html>
<html lang="zh-CN">
<head>
<meta charset="UTF-8"/>
<title>Simple Socket HTTP</title>
</head>
<body>
<h1>欢迎</h1>
<p>这是一个使用 <strong>纯 Java Socket</strong> 实现的迷你 HTTP 服务器。</p>
<ul>
<li><a href="/hello.txt">hello.txt</a></li>
<li><a href="/test.png">test.png</a></li>
<li><a href="/old">/old (301 重定向示例)</a></li>
<li><a href="/temp">/temp (302 重定向示例)</a></li>
</ul>
</body>
</html>
```

- 301 Moved Permanently:

The screenshot shows a web-based interface for managing a simple socket server. At the top, there's a header with 'Method: GET', 'URL: http://localhost:8080/old', and a login form for '用户名' (role1) and '密码' (.....). Below the header, there are buttons for '注册' (Register), '登录' (Login), and '注销' (Logout). A message '已登录:role1' indicates the user is logged in.

**请求报文 (Request Message):**

```
GET /old HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: /*
Cookie: SID=9e34ba2b-a89e-456b-9886-007fa712b291
Connection: keep-alive
```

**响应报文 (Response Message):**

```
HTTP/1.1 301 Moved Permanently
Location: /new
Content-Type: text/plain; charset=UTF-8
Content-Length: 17
Connection: keep-alive

Moved Permanently

-- 自动跟随 -> http://localhost:8080/new --
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Date: Fri, 28 Nov 2025 14:19:23 CST
Last-Modified: Wed, 19 Nov 2025 19:16:59 CST
Server: SimpleSocketServer/1.0
Connection: keep-alive
Content-Length: 338

<!DOCTYPE html>
<html lang="zh-CN">
<head>
<meta charset="UTF-8" />
<title>新页面 /new</title>
</head>
<body>
<h1>这是 /new 页面</h1>
<p>你是从 <code>/old</code> 301 重定向来到这里，或直接访问。</p>
```

- 302 Found:

The screenshot shows the Simple HTTP Client (Socket) interface. At the top, it displays "Simple HTTP Client (Socket) - Request/Response Display". Below that, the "Method" dropdown is set to "GET" and the "URL" field contains "/localhost:8080/temp". There are four buttons: "注册" (Register), "登录" (Login), and "注销" (Logout), followed by the text "已登录:role1". A "请求体 (POST数据)" (Request Body (POST Data)) input area is empty. Below these controls are three buttons: "上传" (Upload), "选择文件" (Select File), and "上传" (Upload again). On the left, under "HTTP请求报文" (HTTP Request Message), the request details are listed:

```
GET /temp HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: /*
Cookie: SID=9e34ba2b-a89e-456b-9886-007fa712b291
Connection: keep-alive
```

On the right, under "HTTP响应报文" (HTTP Response Message), the response details are listed:

```
HTTP/1.1 302 Found
Location: /
Content-Type: text/plain; charset=UTF-8
Content-Length: 5
Connection: keep-alive

Found

-- 自动跟随 -> http://localhost:8080/ --
HTTP/1.1 304 Not Modified
Date: Fri, 28 Nov 2025 14:19:46 CST
Last-Modified: Wed, 19 Nov 2025 19:16:59 CST
Server: SimpleSocketServer/1.0
Connection: keep-alive
Content-Length: 0
```

- 304 Not Modified (缓存验证) :

The screenshot shows the Simple HTTP Client (Socket) interface. At the top, it displays "Simple HTTP Client (Socket) - Request/Response Display". Below that, the "Method" dropdown is set to "GET" and the "URL" field contains "http://localhost:8080/". There are four buttons: "注册" (Register), "登录" (Login), and "注销" (Logout), followed by the text "已登录:role1". A "请求体 (POST数据)" (Request Body (POST Data)) input area is empty. Below these controls are three buttons: "上传" (Upload), "选择文件" (Select File), and "上传" (Upload again). On the left, under "HTTP请求报文" (HTTP Request Message), the request details are listed:

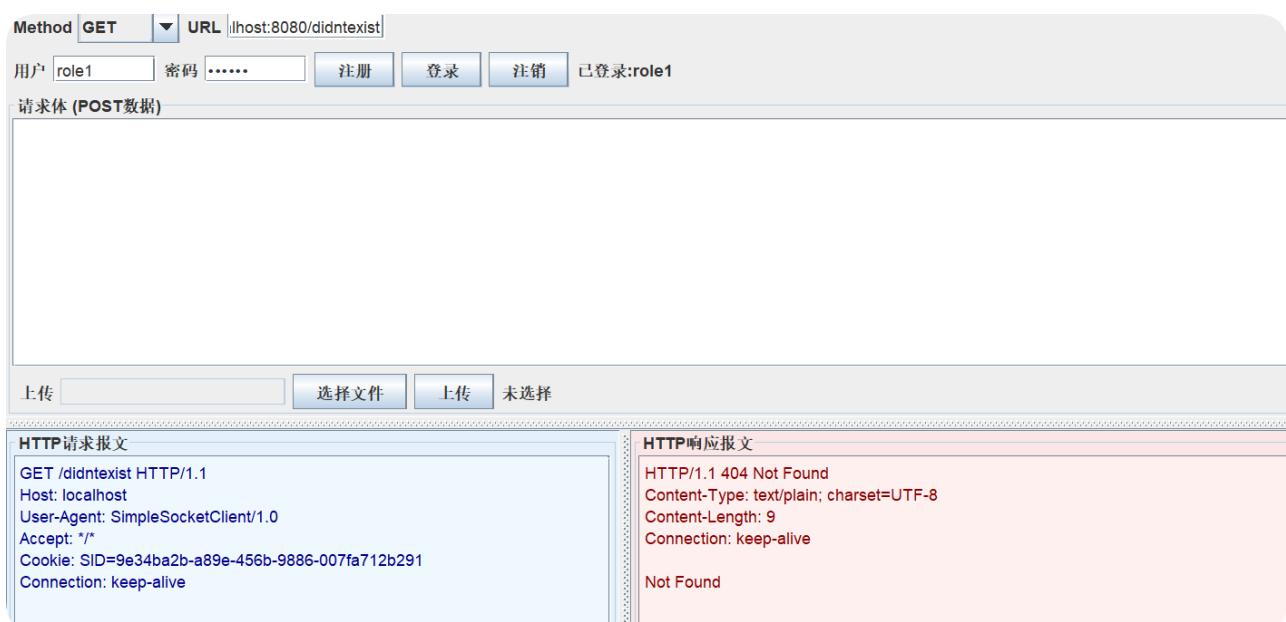
```
GET / HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: /*
If-Modified-Since: Wed, 19 Nov 2025 19:16:59 CST
Cookie: SID=9e34ba2b-a89e-456b-9886-007fa712b291
Connection: keep-alive
```

On the right, under "HTTP响应报文" (HTTP Response Message), the response details are listed:

```
HTTP/1.1 304 Not Modified
Date: Fri, 28 Nov 2025 14:20:10 CST
Last-Modified: Wed, 19 Nov 2025 19:16:59 CST
Server: SimpleSocketServer/1.0
Connection: keep-alive
Content-Length: 0

(使用缓存的 Last-Modified: Wed, 19 Nov 2025 19:16:59 CST)
```

- 404 Not Found:



- 长连接 (Keep-Alive) 验证:

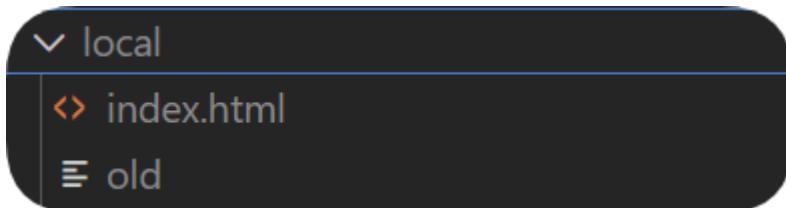
```
[服务器] 接受新连接: /127.0.0.1:61202
[服务器] 开始处理连接: /127.0.0.1:61202
[服务器] 处理请求 #1 - GET / (连接: /127.0.0.1:61202)
[服务器] 保持连接, 等待下一个请求... (已处理: 1 个请求)
[服务器] 处理请求 #2 - GET / (连接: /127.0.0.1:61202)
[服务器] 保持连接, 等待下一个请求... (已处理: 2 个请求)
[服务器] 处理请求 #3 - GET / (连接: /127.0.0.1:61202)
[服务器] 保持连接, 等待下一个请求... (已处理: 3 个请求)
[服务器] 处理请求 #4 - GET / (连接: /127.0.0.1:61202)
[服务器] 保持连接, 等待下一个请求... (已处理: 4 个请求)
[服务器] 处理请求 #5 - GET / (连接: /127.0.0.1:61202)
[服务器] 保持连接, 等待下一个请求... (已处理: 5 个请求)
[服务器] 处理请求 #6 - GET / (连接: /127.0.0.1:61202)
[服务器] 保持连接, 等待下一个请求... (已处理: 6 个请求)
[服务器] 处理请求时发生错误: Read timed out
[服务器] 接受新连接: /127.0.0.1:60713
[服务器] 开始处理连接: /127.0.0.1:60713
[服务器] 处理请求 #1 - GET / (连接: /127.0.0.1:60713)
[服务器] 保持连接, 等待下一个请求... (已处理: 1 个请求)
[服务器] 处理请求时发生错误: Read timed out

```

第一个连接, 没超时继续复用

超时重建连接

- 客户端成功获得资源



#### 4. 不支持的方法 (DELETE 示例)

- 说明: 使用未实现的 HTTP 方法会返回 405。
- 展开查看 405 方法不允许

- 405 Method Not Allowed:

Method: **DELETE** URL: **http://localhost:8080/**

用户: role1 密码: ..... 注册 登录 已登录:role1

请求体 (POST数据)

上传 选择文件 上传 未选择

**HTTP请求报文**

```
DELETE / HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: */*
If-Modified-Since: Wed, 19 Nov 2025 19:16:59 CST
Cookie: SID=9e34ba2b-a89e-456b-9886-007fa712b291
Connection: keep-alive
```

**HTTP响应报文**

```
HTTP/1.1 405 Method Not Allowed
Content-Type: text/plain; charset=UTF-8
Content-Length: 18
Connection: keep-alive

Method Not Allowed
```

## 5. 上传资源 (422 非法命名)

- 说明: 支持文件上传; 若命名非法 (如包含 [ ]) , 返回 422。

### ► 展开查看上传演示

- 上传 **test2.png**:

**HTTP请求报文**

```
POST /upload/test2.png HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: */*
Content-Type: multipart/form-data; boundary=----WSOCK1764311545038
Content-Length: 16416
Cookie: SID=9e34ba2b-a89e-456b-9886-007fa712b291
Connection: keep-alive

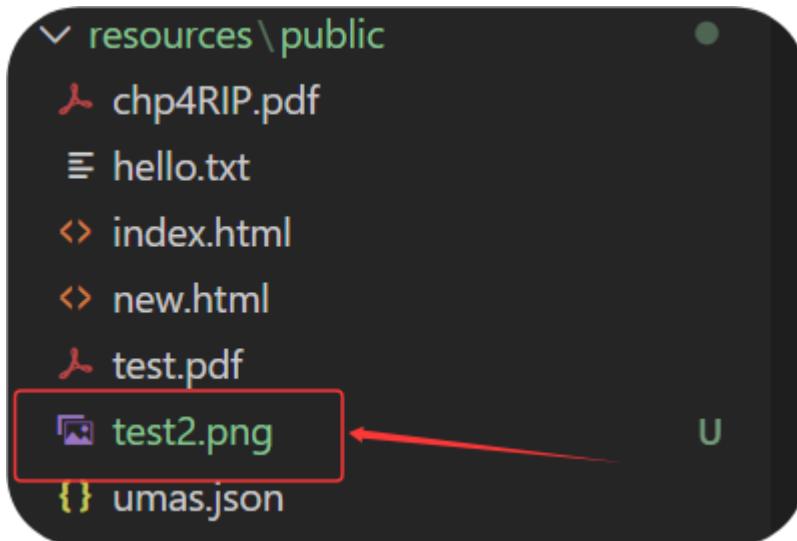
----WSOCK1764311545038
Content-Disposition: form-data; name="file"; filename="test2.png"
Content-Type: image/png

[文件内容: 16266 bytes]
----WSOCK1764311545038--
```

**HTTP响应报文**

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8
Content-Length: 40
Connection: keep-alive

上传成功:
/resource/public/test2.png
```



- 422 非法命名 (示例 test[]2.png) :

The screenshot shows a network traffic capture interface with two panels: "HTTP请求报文" (HTTP Request Message) and "HTTP响应报文" (HTTP Response Message).

**HTTP请求报文 (Request Message):**

```
POST /upload/test[]2.png HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: */*
Content-Type: multipart/form-data; boundary=----WSOCK1764311843742
Content-Length: 16418
Cookie: SID=9e34ba2b-a89e-456b-9886-007fa712b291
Connection: keep-alive

-----WSOCK1764311843742
Content-Disposition: form-data; name="file"; filename="test[]2.png"
Content-Type: image/png

[文件内容: 16266 bytes]
-----WSOCK1764311843742--
```

**HTTP响应报文 (Response Message):**

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: text/plain; charset=UTF-8
Content-Length: 15
Connection: keep-alive

非法文件名
```

## 6. 服务器内部错误 (500)

- 说明：模拟触发除 0 异常，服务端返回 500。

### ► 展开查看 500 错误

The screenshot shows a web application interface and its corresponding HTTP request and response logs.

**Web Application Interface:**

Method: GET URL: http://localhost:8080/test500

用户: role1 密码: ..... 登录 按钮 已登录:role1

请求体 (POST数据):

上传: 18\Desktop\pa报告\test[]2.png 选择文件 上传 失败

**HTTP请求报文 (Request Message):**

```
GET /test500 HTTP/1.1
Host: localhost
User-Agent: SimpleSocketClient/1.0
Accept: */*
Cookie: SID=67a27b0f-ac9d-4254-9075-a189d66ce7c3
Connection: keep-alive
```

**HTTP响应报文 (Response Message):**

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/plain; charset=UTF-8
Content-Length: 58
Connection: keep-alive

500 Internal Server Error: ArithmeticException - / by zero
```