01_Servlet的单实例多线程

- 单线程多实例解释
  - Servlet在服务器中只会创建一个实例
  - 当多个请求同时请求一个servlet时,就会根据tomcat中的server.xml文件中的<Connector>标签配置线程池,然后当项目启动时,根据项目中web.xml初始化线程池
  - 当请求到达时，Servlet容器通过程调度线程池中等待状态的线程给请求
  - 线程执行Servlet的service方法
  - 请求结束，放回线程池，等待被调用
- 优点
  - Servlet单实例，减少了产生servlet的开销
  - 通过线程池来响应多个请求，提高了请求的响应时间
- 缺点
  - 在Servlet中尽量避免使用成员变量,因为成员变量属于对象,而Servlet是单实例,所以相当于这个成员变量是给多个线程所共享,所以存在线程安全问题.

02_自定义服务器

- 响应头设置

```
os.write("HTTP/1.1 200 OK\r\n".getBytes());
os.write("Content-Type:text/html\r\n".getBytes());
os.write("\r\n".getBytes());
```

- 完整代码

```
    ServerSocket server = new ServerSocket(6677);
    ExecutorService executorService = Executors.newFixedThreadPool(3);
    while(true){
        Socket socket = server.accept();
        executorService.submit(new Runnable() {
            @Override
            public void run() {
                System.out.println("aaa2");
                try {
                    InputStream is = socket.getInputStream();
                    BufferedReader br = new BufferedReader(new InputStreamReader(is));
                    String line = br.readLine();
                    System.out.println(line);
                    String[] arr = line.split(" ");
                    System.out.println(arr[1]);
                    String htmlpath = arr[1].substring("/test01/".length());
                    FileInputStream fis = new FileInputStream(htmlpath);
                    OutputStream os = socket.getOutputStream();
```

```
                    os.write("HTTP/1.1 200 OK\r\n".getBytes());
                    os.write("Content-Type:text/html\r\n".getBytes());
                    os.write("\r\n".getBytes());
                    int len = 0;
                    byte[] bytes = new byte[1024];
                    while((len = fis.read(bytes))!=-1){
                        os.write(bytes,0,len);
                    }
                    fis.close();
                    socket.close();
                }catch (IOException e){
                    e.printStackTrace();
                }
            }
        });
    }
```