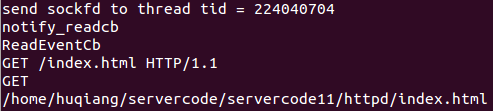
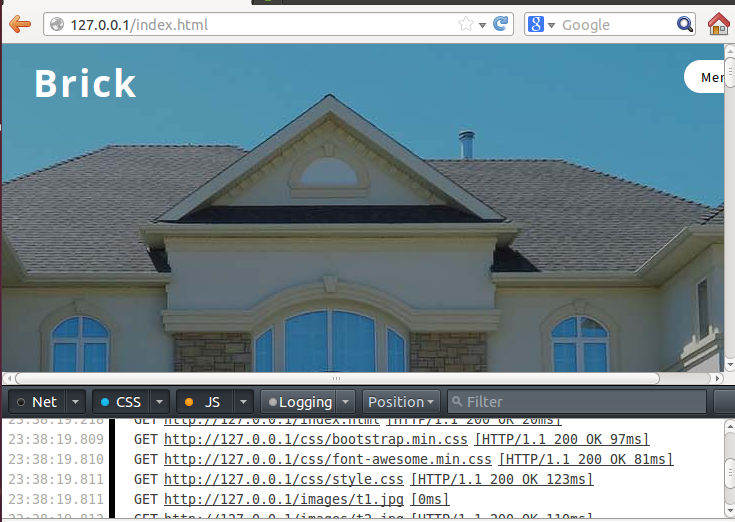
修改程序的配置文件，包括服务器地址，监听端口，工作线程数量等。然后在Linux环境下编译程序，运行编译好的程序。程序以管理员的方式运行，普通用户否则无法监听80端口。如图5.1的打印信息所示，启动服务器并创建了4个工作线程。

**图5.1 启动服务器**

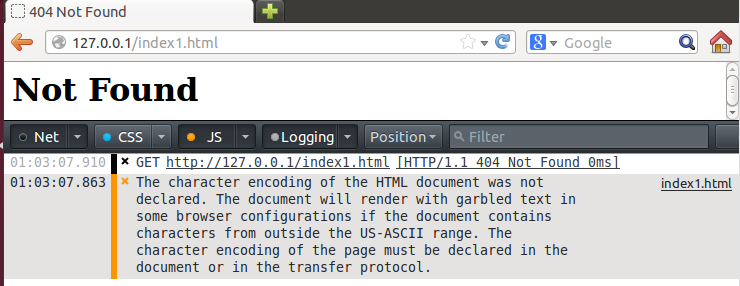
## 5.1 静态

打开系统自带的firefox浏览器，输入本机的网址，请求默认的index.html静态页面。该静态页面包括文字、图片、样式等。浏览器发出请求后可以看到图5.2所示，主线程接受了连接，并将连接随机发送给一个工作线程处理。工作线程触发管道可读回调函数，将新连接加入了工作线程事件机。随后连接触发可读事件，工作线程读取请求报文并解析。打印信息显示了该请求报文的部分信息，包括请求方法和资源绝对路径等。

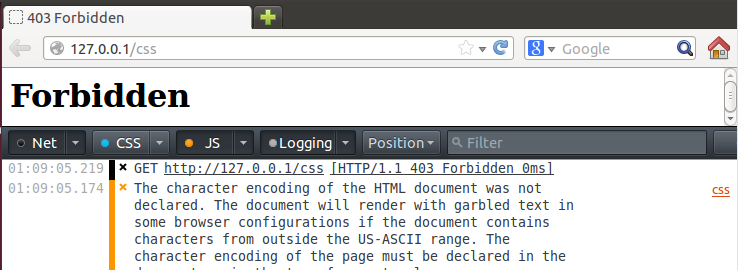
**图5.2 服务器接收请求并解析**

请求成功后，浏览器显示了请求的静态页面，如图5.3所示。文字、图片和样式均能正常显示。打开火狐的状态查看功能，可以看到每一个资源请求的响应报文的状态行，说明静态页面资源请求成功。

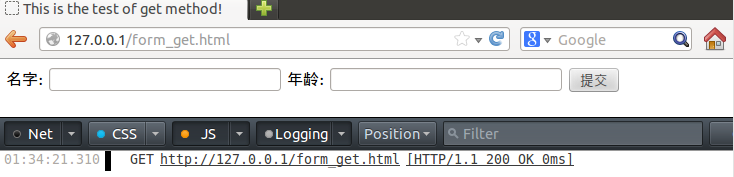
**图 5.3 静态页面请求**

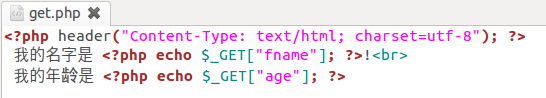
请求资源失败会根据失败原因返回不同页面。如图5.4请求不存在的资源返回404错误。如图5.5因为没有访问目录权限返回403错误。

**图5.4 文件不存在**

**图5.5 禁止访问**

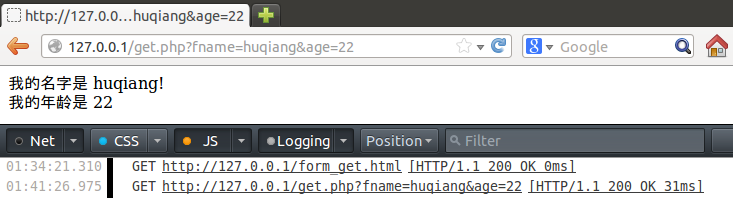
## 5.2 动态页面请求测试

服务器实现了带参数的GET方法和POST方法，首先测试带参数GET方法。浏览器请求一个静态html文件，在表单中输入名字和年龄后，点击提交浏览器将请求发送个服务器。表单和脚本如图5.6和5.7所示。

**图5.6 使用get方法的表单**

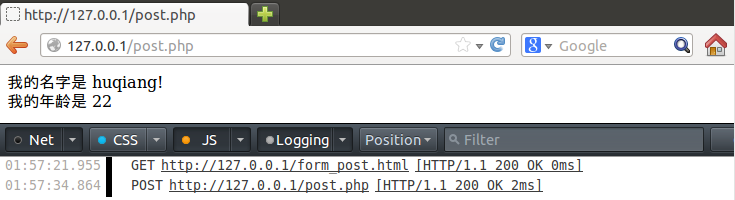
**图5.7 接收get变量的php脚本**

点击提交后，其结果如图5.8所示。请求的参数在url中显示，浏览器成功接收服务器返回的php脚本执行结果。

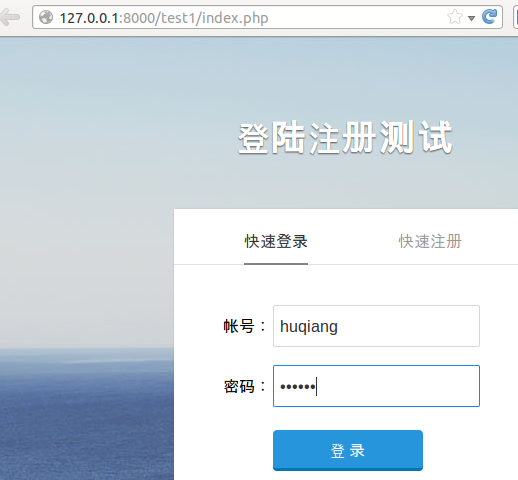


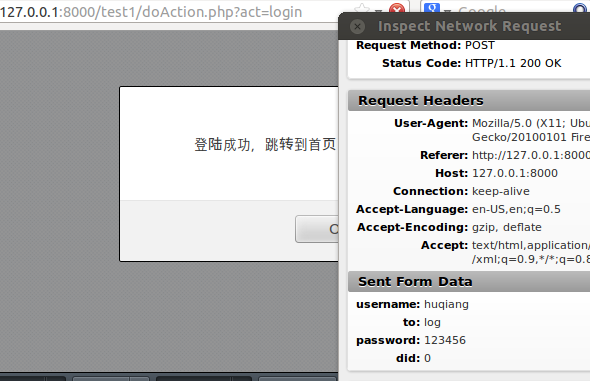
**图5.8 get方法测试结果**

测试POST方法的html文件和php脚本与测试GET方法的类似，只是将html文件表单中的method由get换成post，脚本中的$\_GET变量替换成$\_POST变量。测试html表单结果如图5.9所示，浏览器顺利收到服务器发送的php脚本执行结果。打开firefox的查看工具，可以看见图5.10中POST实体主体内容。

**图5.9 post方法测试结果**

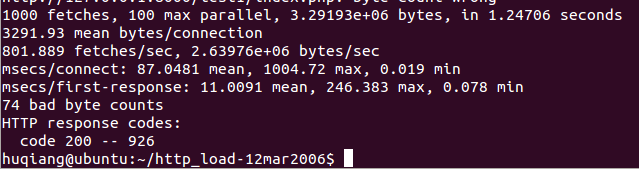
**图 5.10 post方法实体主体数据**





### 5.3 压力测试

服务器的压力测试工具使用的是http\_load，关于此工具的安装和使用方法在这里不做详述，只需关注测试结果即可。测试系统使用的CPU配置为双核Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz，测试模拟100个并发进程进行连续60秒请求，结果如图5.11所示。

**图5.11 测试结果**