假设Client端发起中断连接请求，也就是发送FIN报文。Server端接到FIN报文后，意思是说"我Client端没有数据要发给你了"，但是如果你还有数据没有发送完成，则不必急着关闭Socket，可以继续发送数据。所以你先发送ACK，"告诉Client端，你的请求我收到了，但是我还没准备好，请继续你等我的消息"。这个时候Client端就进入FIN\_WAIT状态，继续等待Server端的FIN报文。当Server端确定数据已发送完成，则向Client端发送FIN报文，"告诉Client端，好了，我这边数据发完了，准备好关闭连接了"。Client端收到FIN报文后，"就知道可以关闭连接了，但是他还是不相信网络，怕Server端不知道要关闭，所以发送ACK后进入TIME\_WAIT状态，如果Server端没有收到ACK则可以重传。“，Server端收到ACK后，"就知道可以断开连接了"。Client端等待了2MSL后依然没有收到回复，则证明Server端已正常关闭，那好，我Client端也可以关闭连接了。Ok，TCP连接就这样关闭了！

## HTTP协议

http协议包含请求和响应两部分，其中请求包含请求行、请求头和请求正文三部分，响应也包含状态行、响应头和响应正文三部分

请求行包含请求方法、资源路径、协议版本三部分，资源路径包含协议、ip、端口、应用名、资源路径等，如果是get请求，资源路径后面还可以跟请求参数

### http请求

#### 请求行

##### 请求方法（GET/POST）

##### 请求资源路径URI

##### HTTP协议版本（HTTP1.1）

#### 请求报头

##### 请求正文的类型

##### 编码类型

##### Cookie（里面看sessionID）

##### 请求正文长度

##### 签名串sign、token也会放在请求头里面

#### 请求正文

##### 请求的数据格式

##### xml

##### json

##### 键值对，&分割符

##### 字符串

### http响应

#### 状态行

##### http协议版本

##### 状态码

## 状态码 状态码是一个3位整数，以1、2、3、4、或5开头

1. 1xx：信息提示，表示临时的响应

2. 2xx：响应成功，表示服务端成功接收了客户端请求

3. 3xx：重定向 （服务端不处理，返回302错误码，并在响应头指定一个重定向地址，浏览器再次请求新地址）

4. 4xx：客户端错误，表示客户端可能有问题

5. 5xx：服务器错误，表示服务端由于遇到某种错误而不能响应客户端请求

## 常见状态码

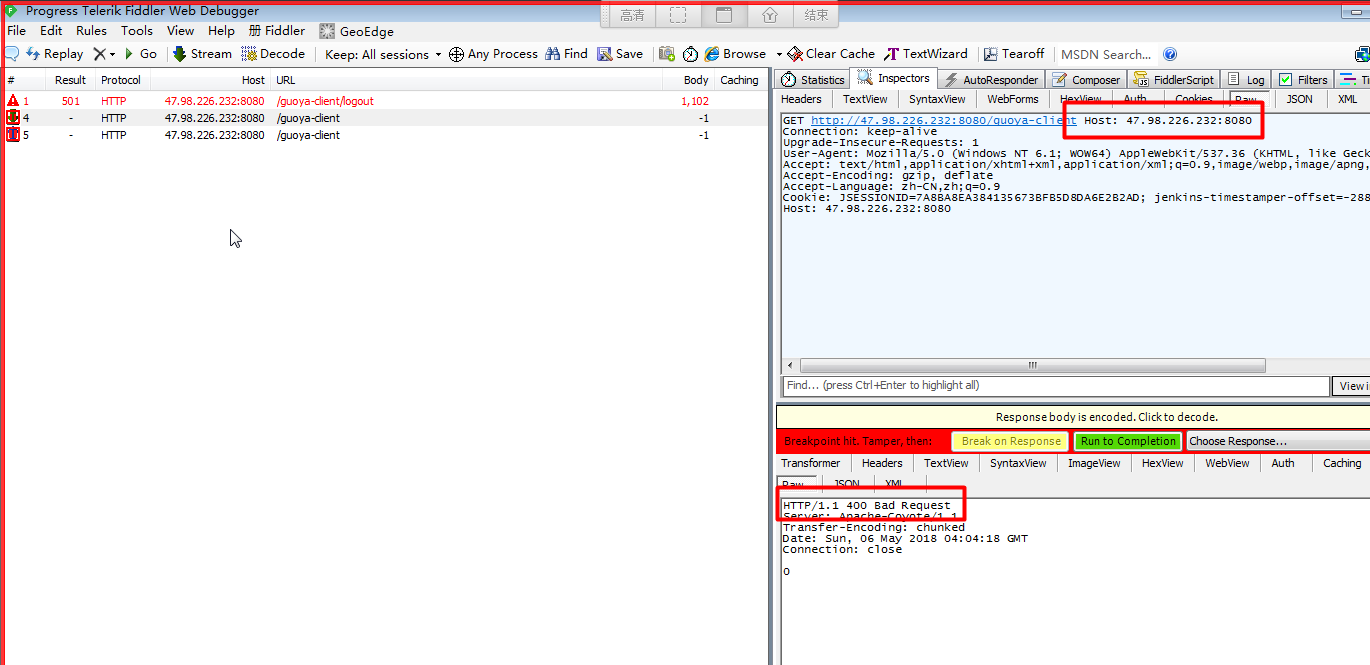
1. 200：响应成功
2. 302：重定向

3. 400：错误的请求，客户发送的http请求不正确

4. 404：文件不存在，在服务器上没有客户端要求访问的文档

5. 405：服务器不支持客户端的请求方式

6. 500：服务器内部错误



##### 描述

#### 响应报头

##### 正文类型

| **文件扩展名** | **MIME类型** |
| --- | --- |
| 未知的数据类型或不可识别的扩展名 | content/unknown |
| .bin、.exe、.o、.a、.z | application/octet-stream |
| .pdf | application/pdf |
| .zip | application/zip |
| .tar | application/x-tar |
| .gif | image/gif |
| .jpg、.jpeg | image/jpeg |
| .htm、.html | text/html |
| .txt、.c、.h、.text、.java | text/plain |
| .mpg、.mpeg | video/mpeg |
| .xml | application/xml |
| 上传文件 | multipart/form-data |

##### 编码类型

utf-8、GBK、ISO8859-1

编码的核心差异：一个字符的**字节数**，**编码表**

##### cookie

#### 响应正文

### 常见响应码

#### 200成功

#### 304重定向

#### 400请求不正确

#### 404资源不存在

#### 405请求方式不支持

#### 500服务器错误

### session和cookie

Cookie是浏览器不同界面共享数据的本地存储

Session是服务器不同接口共享数据，追踪会话状态

Cookie里面会存储sessionid，跟session绑定

#### cookie

cookie是浏览器本地存储

会将响应报文的cookie值自动存到浏览器的cookie里面

每次发请求，浏览器会自动将cookie里面的值加到请求头的cookie字段里面

Cookie值有域和失效时间属性，不是同一个域或者失效的cookie，不会加到请求里面

#### Session

用来追踪用户**会话状态**，也就是多个请求是否是同一个操作的不同步骤

浏览器cookie中无sessionID，发起请求

服务端收到请求，无sessionID，新建，存储并回给浏览器

浏览器cookie中加入sessionID

浏览器发起请求，带sessionID

服务器当作已登录过

## HTTPS协议

### URL地址

### SSL加密

### 端口

### 证书

| **差异点** | **http** | **https** |
| --- | --- | --- |
| 加密 | 明文传输 | SSL加密传输 |
| 证书 | 不需要 | 浏览器需要从服务端下载证书 |
| 地址 | http | https |
| 默认端口 | 80 | 443 |

## SOAP协议

Soap协议可以建立在http协议之上，先用一个servlet接口接收soap请求，再对请求的正文进行解析

### soap和http的用途

soap用于webservice接口

http用于web界面和http接口

### soap和http的消息格式

soap消息只能是xml格式（而且xml里面的部分标签是规定好的，这句只理解，不说）

http消息可以是xml、json、键值对、普通字符串等各种格式

### soap和http的关系

先有http，后有soap，soap消息可以借用http进行传输

Soap接口，会生成一个wsdl文件，可以直接通过界面查看接口详情

## TCP/UDP、HTTP、HTTPS、SOAP关系

### tcp（长连接，四次挥手断开链接结束） 和 udp （短链接，字符串传完就结束）端口层，用来传字符串（传字符串是无序的）

### http是对tcp传输的字符串进行格式规定（请求行、请求头、请求正文）

### soap是对http规定格式的请求正文进一步进行格式规定（XML格式）

### https是防止传输时数据泄露，进行加密传输

双向加密，也就是先加密，再传输，再解密，再处理

# 接口测试

## 测试技术

### jmeter

### soapUI

### postman

### jmeter和soapUI的区别

soapUI只能测soap协议的接口

jmeter既能测soap协议的接口，又能测http协议的接口

http

soap

ftp

java

jdbc

soapUI不需要自己写请问报文（因为wsdl文件都描述规定好了，能自动生成），只需要按照字段填值

jmeter需要自己写请求报文的正文

jmeter功能更强大，内置了很多函数和功能，方便做自动化

## 接口测试流程

### 接口规范

### 项目需求和测试需求

### 测试功能点

### 制定测试计划和分派任务

### 测试用例（可以写粗点，写脚本时细化）

### 写xml或json报文

### 使用jmeter或者soapUI写测试脚本

### 执行查看结果

### 对测试脚本进行增强，做成自动化测试脚本

## Jmeter核心

#### 变量

接口靠变量传数据

#### 变量池

所有接口共享一个变量池

#### debug sampler查变量

可以通过debug sampler查看当前变量

## jmeter常用方法

### 添加线程组

### 添加http请求

### 填写http请求内容

#### IP

#### 端口

#### 请求方法

#### 路径

#### 请求报文（正常数据，保证能成功）

### 添加查看结果树

执行，并查看结果报文内容和格式

### 添加断言

响应断言

Beanshell断言

### 参数化

#### 常量

#### 单接口使用的变量

##### 内置函数（比如随机函数、时间函数）

#### 多个接口共用的变量

##### 添加用户定义变量（常量+函数+变量）

#### 从外部获取或生成

##### Jdbc链接和请求

##### Csv配置和循环控制器

##### Beanshell脚本

#### 从其它接口取

##### 添加正则表达式提取器

### Jmeter接口关联

#### 前一个接口通过正则表达式提取器，把响应数据存放到变量池

#### 后一个接口通过变量从变量池取值

Jmeter的各类请求、断言、beanshell、jdbc、配置之间只能通过变量进行交互

前一个接口添加一个后置处理器中的正则表达式提取器

将响应报文的指定数据提取到变量

后一个接口通过变量使用该数据

### 响应正确，如何保证数据库正确

Jdbc请求，查数据，放变量池

正则表达式提取器，把接口的响应结果提取到变量池

Beanshell断言，从变量池取jdbc的数据和接口响应数据，做对比

1. Failure = **true**;
2. FailureMessage = "和历史数据不匹配";

### Jmeter jdbc

#### 添加jdbc的jar包

#### 新建一个jdbc connection configuration，取个变量名

#### 新建jdbc Request，写sql

#### 给查询结果的列取变量名(jmeter自动给变量添加行号)

#### 通过变量名\_行号取数据，通过变量名\_#取总条数

保证接口和数据库都正确

接口通过响应断言自动判断

数据库通过jdbc前置和后置处理器各查一次数据，再通过beanshell断言对比查询结果

先导jar包

再配置jdbc config

Jdbc请求

### Jmeter 数字签名

#### 找开发要加密字段、排列顺序、key、加密算法类（一般MD5）jar包

#### 再将开发提供的加密jar包导入jmeter

#### 给请求添加beanshell前置处理器

#### vars.get取加密字段，调用加密算法生成加密串

#### 将加密串通过vars.put加入变量

#### 请求报文就可以通过变量名使用这个签名串

var name=vars.get("name");

var age=vars.get("age");

var key="我不是李白";

var sign=MD5Tool.sign("name=" + name+"&age="+age+"&key="+key);

vars.put("sign",sign);

### Jmeter csv文件

一行就是一个测试用例，包含测试数据（参数化请求报文）和最后一列的预期结果（做断言用）

jmeter怎么处理csv文件？添加csvdata set config，指定文件路径，给每列取变量名

怎么把数据都执行一遍？循环控制器或者多线程

循环次数比数据多怎么办？可以配置，要么停下来，或则循环使用

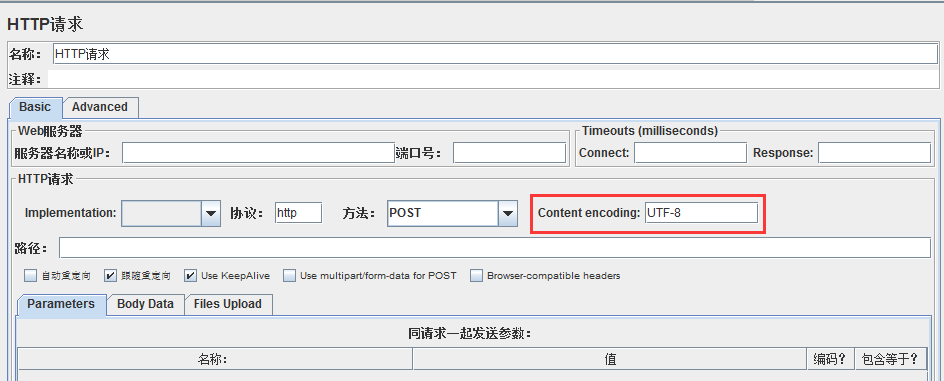
### Jemter如何添加请求报头的内容

添加消息头管理器

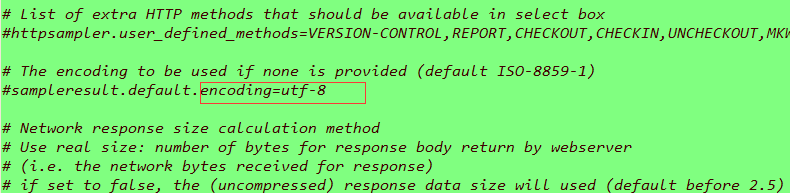
### Jmeter如何做需要提前登录的接口测试

### Jmeter请求和响应 乱码

**①**请求数据显示乱码，可以在请求中如下设置：



**②**返回数据包含乱码时，可以修改jmeter\bin\jmeter.properties文件中的一个属性：将encoding=后面的编码格式改为utf-8，如下：



## 

## soapUI常用方法

### 根据wsdl地址生成工程

### 选择要测试的方法

### 填写请求字段的值

### 执行方法

### 查看结果

## soapUI常用方法

## fidller常用方法

### fiddler如何抓手机包

* + 1. 启动fiddler，设置远程连接和https请求
    2. 查看fiddler所在主机的IP，端口取默认的8888
    3. 把手机连到跟fiddler所在电脑同一个wifi
    4. 设置wifi的代理为主机IP和8888端口，fiddler代理设置完后要重启
    5. 手机浏览器输入电脑ip和端口安装证书（测https需要证书）
    6. 正常使用手机，fiddler抓包

### 抓包原理

#### 正常：浏览器<->服务器

#### 抓包：浏览器 < -> Fiddler < -> 服务器

#### 浏览器与服务器建立TCP连接

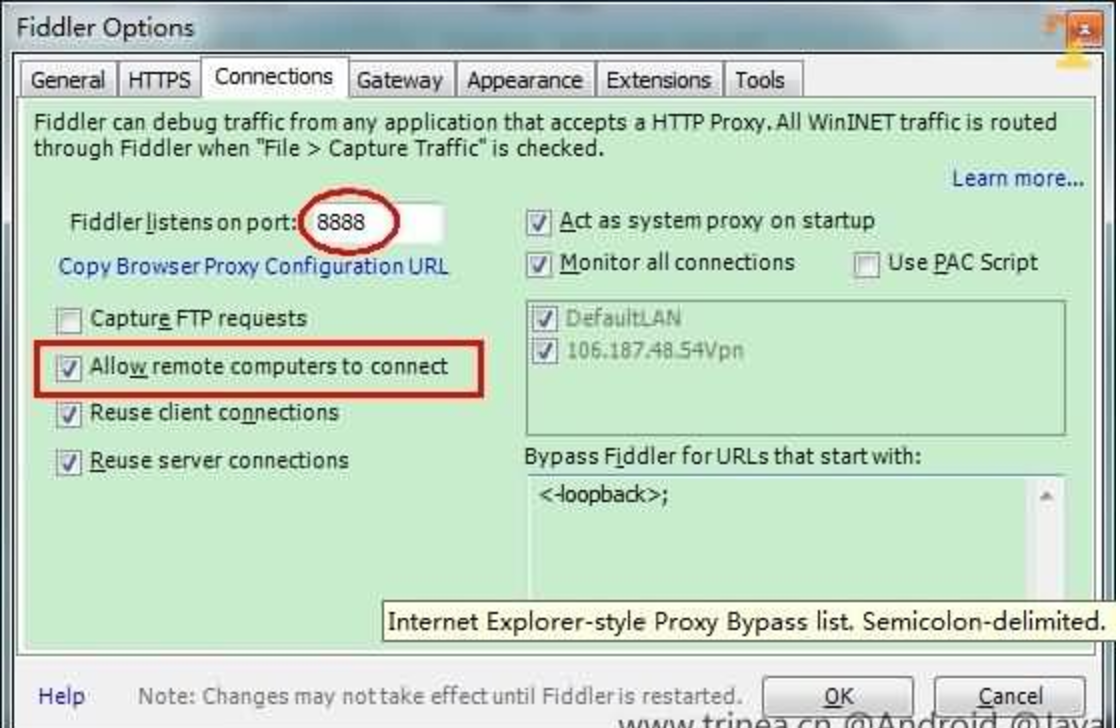
打开Fiddler菜单项Tools->Fiddler Options，选中decrypt https traffic和ignore server certificate errors两项，如下图：



第一次会提示是否信任fiddler证书及安全提醒，选择yes，之后也可以在系统的证书管理中进行管理。

##### 设置允许远程连接

如上图的菜单中点击connections，选中allow remote computers to connect，默认监听端口为8888，若被占用也可以设置，配置好后需要重启Fiddler，如下图：



#### 设置浏览器

##### 设置浏览器代理IP和端口

fiddler启动时，会自动设置浏览器代理为127.0.0.1，端口8888

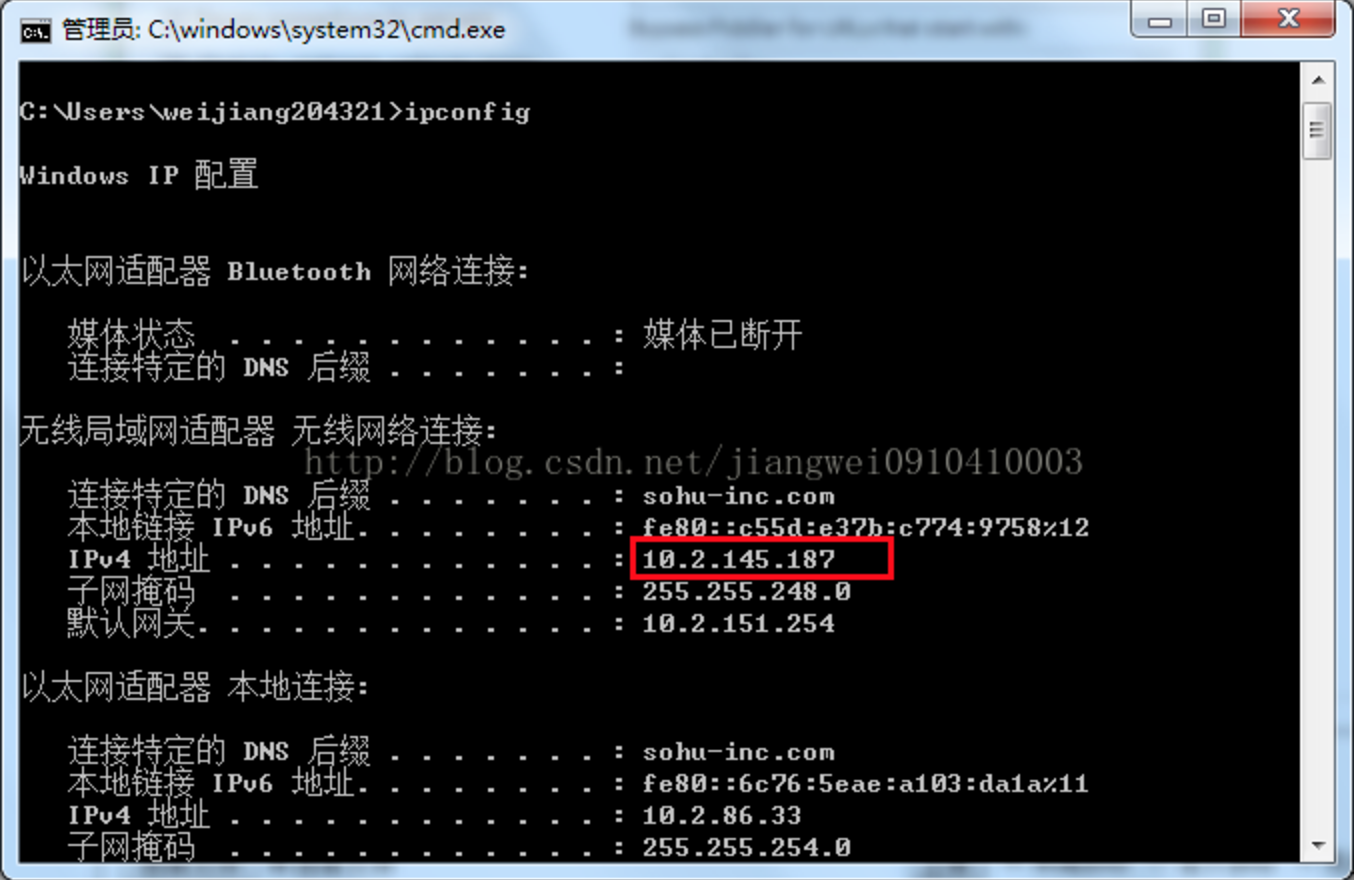
如果没能自动设置成功，则需要在浏览器设置-高级-代理里面进行设置

#### 设置手机

##### 查看fiddler所在电脑的IP

首先获取PC的ip地址：命令行中输入:ipconfig,获取ip地址

首先获取PC的ip地址：命令行中输入:ipconfig,获取ip地址

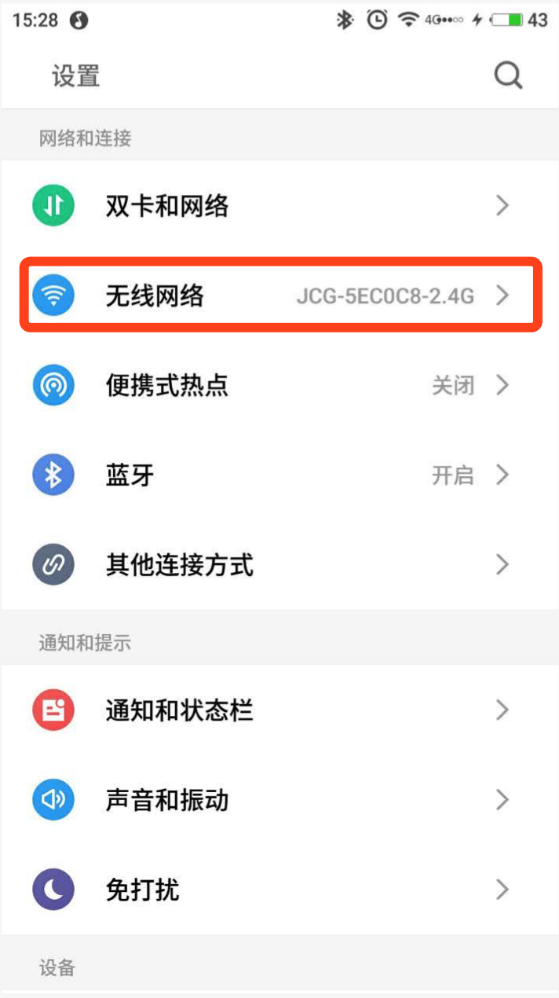


##### 手机和PC连接到同一个网络

连接同一个wifi，或者网线连同一个路由器

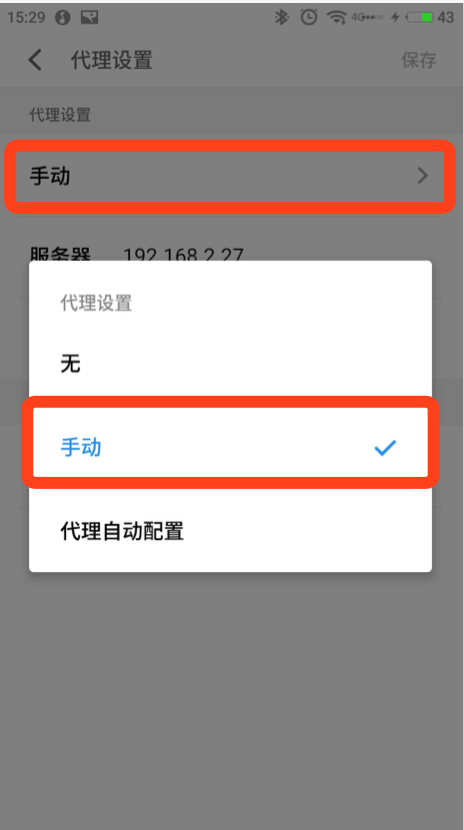
##### 设置WLAN代理IP和端口

进入手机的设置->点击进入WLAN设置->选择连接到的无线网，长按弹出选项框：如图所示:





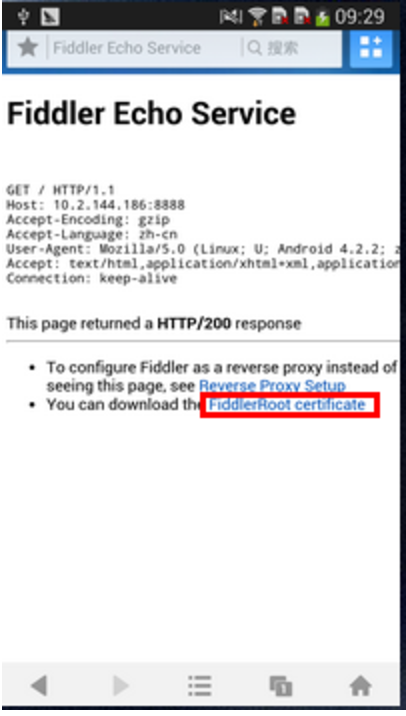






##### 安装fiddler证书

使用Android手机的浏览器打开：http://10.2.145.187:8888， 点"FiddlerRoot certificate" 然后安装证书，如图:



##### 将代理改回“无”

否则fiddler关闭后，手机调用代理（fiddler）不通，则无法上网

http://blog.csdn.net/jiangsanfeng1111/article/details/52448481

# WEB测试

## 测试内容

### 功能测试

### 兼容性测试

#### 不同浏览器厂商：IE、chrome、Firefox等

#### IE不同版本：IE6~IE11

1. 浏览器模拟不同版本（代价最小，但是测的有限）
2. Ietester或者modern.ie测试不同的ie版本



1. 安装虚拟机，每个虚拟机安装1个不同的ie版本（测得最好，代价大）

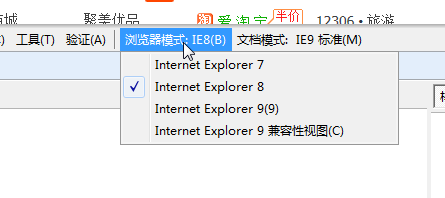
产品有规定的浏览器支持范围，如果超出，需要单独提需求

IMG_256IMG_256

##### IE打开要测试的界面

##### 打开开发者工具

##### 打开仿真页



##### 选择浏览器版本

##### 界面自动重新加载

##### 验证兼容性

### 性能测试

使用fiddler+jmeter模拟界面请求，进行性能测试

### 前用户的进程ps –ef | grep 用户名

### 查看当前用户的指定进程ps –ef | grep 用户名 | grep “正则表达式”

### 强杀进程kill -9

Kill -9 进程号

### 查看使用空间du –sh 文件名

（备注：s是按文件夹统计，h是按适合理解的单位显示如kb、G、M）

df –h

查看文件夹或文件大小

# du -sh

2.2G .

# du -sh \*

2.2G software

8.7M softwareBak

32K temp

# du -sh \*

330M apache-tomcat-7.0.86

1.4G jenkins

4.0K jenkins-slaves

16K mysql

16K mysql\_class

704K slave.jar

74M webapps.zip

270M zbox

55M zentao

43M ZenTaoPMS.biz1.1.4.zbox\_64.tar.gz

### 查看主机剩余空间

只能看磁盘的剩余空间

# df -h \*

Filesystem Size Used Avail Use% Mounted on

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

/dev/vda1 40G 6.8G 31G 19% /

### Top看进程资源占用

### Nmon查看系统性能

### 需要查看的内容：端口号、程序部署路径

# netstat -ano | grep 8080

## 系统环境变量

### 查看环境变量env

Env查看所有环境变量

使用管道符和grep过滤变量结果

# env | grep HOME

HOME=/root

### 环境变量配置文件.profile 或者.bash\_profile

### 加载环境变量配置文件. .profile或者. .bash\_profile

### 添加一个环境变量vi .bash\_profile，然后加入一行export 变量名=变量值

### 添加环境变量的4个步骤：

### ls –al查看环境变量文件（用户根目录）

### vi 配置文件名

Vi 文件名，进入

“i”：进入编辑模式

“esc”：退出，进入命令模式

“:”：开始输命令

“:q!”： 不保存，强制退出

“:wq”：保存并退出

### export 变量名=变量值（添加变量）

### . 环境变量配置文件名（加载配置文件，其中”.”是source命令的简写）

### 配置历史记录

Vi .bash\_profile

Export HISTFILESIZE=400

HISTSIZE=400

## 网络测试

### Ping：网络通不通

### Traceroute：网络各个节点通不通

### ftp：传文件

### Sftp：加密传文件

### Telnet：远程登录

### Ssh：远程加密登录

## Web服务器

### tomcat的配置文件：conf/server.xml

### 默认程序部署路径：webapps

### 启动tomcat脚本：bin/startup.sh

### 停掉tomcat脚本：bin/shutdown.sh

### 查看日志：logs/catania.out

## Shell脚本

### 创建脚本

Touch start.sh

### 指定shell解析程序

可省略，默认使用环境变量指定的shell

#!/usr/bin/sh

### 打印日志

Echo ‘日志内容’

# cat start.sh

#!/usr/bin/sh

echo '------------开始启动服务:/root/software/apache-tomcat-7.0.86/bin/startup.sh'

/root/software/apache-tomcat-7.0.86/bin/startup.sh

echo '-----------------服务启动完成---------------'

tail -200f /root/software/apache-tomcat-7.0.86/logs/catalina.out

## Crontab定时器

### 新增crontab -e

### 查看crontab -l

### 定时的时间格式

### #注释

## 文件切割split和合并cat

### 分割：文本文件

$ split -C 100M large\_file.txt stxt

$ split -l 1000 large\_file.txt stxt

### 分割：二进制文件

$ split -b 100M data.bak sdata

### cat合并文件

$ cat stxt\* > new\_file.txt

## 上传和下载

### sz/rz上传和下载

Sz 文件名：下载文件

Rz：上传文件

默认路径：scurityCRT——》options——》session options——》x/y/zmodem

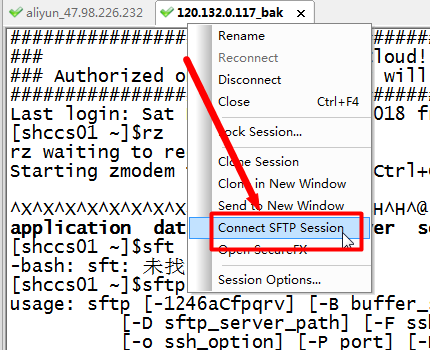
### ftp和sftp命令上传下载

Ls和lls

Cd和lcd

Pwd和lpwd

### Sftp session上传下载



练习题

查找用户下的以cata开头，.out结尾的文件

从查询列表中找到ux下的文件

查看该文件中的所有内容

第一轮，筛选出带8080的行