# C/S模式和B/S模式

## 客户端和服务器

# B/S模式：客户端与服务器一问一答（通讯）遵循：HTTP协议

## HTTP概念：

### HTTP，（超文本传输协议），它是TCP/IP协议中的一个应用层协议。

## HTTP版本：

### HTTP/1.0

### HTTP/1.1

### HTTP-NG 实验室阶段，未投入使用

## HTTP1.0的会话方式

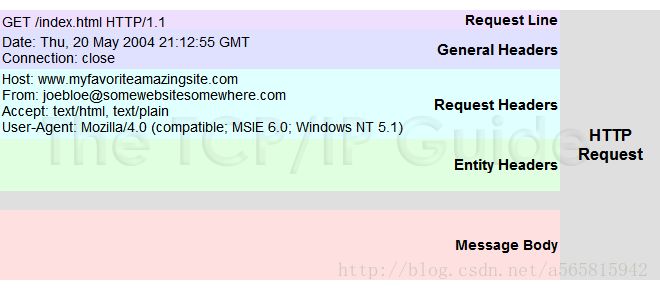
### 建立消息

### 发送请求消息

### 回送响应信息

### 关闭连接  （TCP面向连接；UDP不需要连接的）

## HTTP请求消息



### 请求消息的结构

一个请求行，若干消息头，以及实体内容其中一些消息头是可选的，消息头和实体内容之要用空行隔开

例如：

GET/index.html HTTP/1.1 -请求行

Accept:\*/\* -|

Accept-Lauguage:zh-cn |

Connection:Keep-Alive |

Host:localhost |--多个消息头

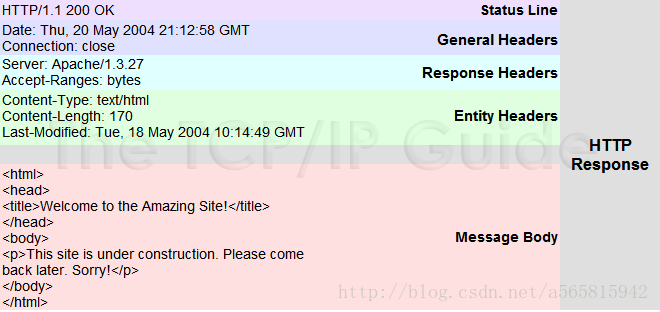
Referer:http://localhost/index.php |

User-Agent:Mozilla4.0 |

Accept-Encoding:gzip,deflate -|

--一个空行

## HTTP响应消息



#### 响应消息的结构

一个状态行，若干消息头，以及实体内容 其中一些消息头和实体内容是可选的，消息头和实体内容之间要用空行隔开

例如

HTTP/1.1 200 OK -状态行

Server.Microsoft-IIS/6.0 -|

Date: Wed Oct 26 11:20:45 CST 2011 |

Content-Length:3389 |-多个消息头

Content-Type:text/html |

Cache-control:private -|

-一个空行

<html> -|

<body> |-实体内容

…… -|

## JAVA访问实例

public class TestHttp {

public static void main(String[] args) {

Socket socket = null;

PrintWriter pw = null;

BufferedReader br = null;

try {

socket = new Socket("www.baidu.com",80);

System.out.println(socket);

pw = new PrintWriter(socket.getOutputStream());

pw.println("GET / HTTP/1.1");

pw.println("Host:www.baidu.com");

pw.println();

pw.flush();

br = new BufferedReader(new InputStreamReader(socket.getInputStream()));

String str = br.readLine();

while(str != null){

System.out.println(str);

str = br.readLine();

}

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally {

try {

if(pw != null){

pw.close();

}

if(br != null){

br.close();

}

if(socket != null){

socket.close();

}

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

}

## HTTP消息的其他细节

### 响应消息的实体内容就是网页文件的内容，也就是我们在浏览器中使用查看源文件的方式看到的内容（HTML代码）

### 一个GET方式的请求消息中不能包含实体内容，且没有采用chunked传输编码方式，那么消息头部分必须包含内容长度的字段，否则，客户端和服务器就无法知道实体内容何时结束

### 在HTTP协议中，还可以使用简单的请求消息和响应消息，他们都没有消息头部分

### 简单的请求消息只能用于GET方式，且请求行中不用指定HTTP版本号

### 对于简单的请求消息，服务器返回简单的响应消息，简单的响应消息中只包含实体内容

## HTTP消息头

### 使用HTTP消息头，可以实现HTTP客户端和服务器之间的条件请求应答，消息头相当于客户端和服务器之间的一些暗号指令  利用消息头可以实现访问统计，例如站长统计，百度统计等

### 每个消息头包含一个投资端名称，然后每一次是冒号，空格，值，回车和换行符  例如：Accept-Lauguage：zh-cn

### 消息头字段名是不区分大小写的，但习惯上将每个单词的第一个字母大写

### 整个小吸头部分中的隔行消息头可按任何顺序排列

### 消息头又可以分为通用消息头，请求头，响应头，实体头四类

### 许多请求头字段都允许客户端在值部分指定多个可接受选项，多个值之间以逗号分隔  例如：Accept-Encoding：gzip，compress

### 有些字段你可以出现多次，例如，响应消息中可以包含有多个“Warning”头字段

## HTTP请求行与状态行

### 请求行

#### 格式：请求方式 资源路径 HTTP版本号<CRLF>

**例如：GET /index.html HTTP/1.1**

**请求方式：GET,POST,HEAD,OPTIONS,DELETE,TRACE,PUT**

### 状态行

#### 格式 HTTP版本号 状态码 描述文本<CRLF>

**例如：HTTP/1.1 20 OK**

**使用GET和POST方式传递参数**

### 在URL地址后面可以附加一些参数

#### URI格式

**协议名://用户名：密码@主机名:端口号/资源路径**

**mailto:vwell@vbaiHoo.net**

#### URN

**用户名@主机名**

### GET方式

**例如：GET /reg.php?username=v.zhang&password=123456 HTTP/1.1**

#### 特点：传送的数据量有限制，一般限制在1kb以内

### POST方式

**例如：**

**POST /res.php HTTP/1.1**

**Host:localhost**

**Content-Type: application/x-www-form-urlencoded**

**Content-Length:10**

**username=v.zhang&password=123456 –实体内容**

#### 特点：传送数据要比GET方式传送大的多

## 响应状态码

### 表示服务器对请求的各种不同处理结果和状态，三位十进制数

### 响应状态码的典型情况

#### 200（正常）

#### 302/307（临时重定向）

#### 301（永久重定向）

#### 304（未修改）

#### 401（未经授权访问）

#### 404（找不到）

#### 500（服务器内部错误）

## HTTPS:HTTP加上加密处理和认证机制。

### 建立连接时，Client把一套加密算法发给Server；

### Server选取一个加密算法作为公钥。（非对称加密，使用私钥才能解开）并把公钥、IP打包在证书里发给Client；

### Client对其验证有效后，使用公钥对密钥（随机生成）加密，发送给公钥。

### Server使用私钥解密。

# 浏览器与web服务器的连接过程是短暂的，每次连接只处理一个请求和响应。对于每一个页面的访问，浏览器与web服务器都要建立一次单独的连接

# 浏览器到web服务器之间的所有通讯都是完全独立分开的请求和响应对

# 支持代理