Web软件技术

张海腾
htzhang@ecust.edu.cn

课 程 概 述

- 课程目的
  - 了解Web应用程序开发的基础知识；
  - 掌握使用Java技术开发Web应用程序的方法；
  - 为继续学习Java EE高级技术奠定基础。
- 课程要求
  - 掌握Java Web应用开发的主要技术；
    - Servlet技术
    - JSP技术
    - JavaBean技术
    - JDBC数据库访问技术
- 课程结果
  - 能够基于MVC设计模式开发一个简单Web应用程序。

预 备 知 识

- **HTTP、HTML知识、JavaScript、CSS、XML等**
- **Java语言知识**
- **数据库知识**

教材

- 使用教材:

《Java Web 编程技术》（第2版）

沈泽刚 秦玉平 主编 清华大学出版社

- 参考教材:

《Java Web 开发实战经典》 李兴华

王月清 主编 清华大学出版社

主要内容

- 第1章 Java Web技术概述
- 第2章 Servlet技术模型
- 第3章 Servlet容器模型
- 第4章 JSP技术模型
- 第7章 JDBC数据库访问

软件与环境

- 运行环境：**JDK**
- 开发工具：**Eclipse**
- 服务器：**Tomcat**
- 数据库：**MySQL**

平时成绩

- 出勤 10 %
- 作业 10 %
- 实验 10%

第1章 Java Web技术概述

本章主要内容：

1.1 Internet与万维网

1.2 Web常用技术

1.3 服务器资源

1.4 Tomcat服务器

1.5 Servlet与JSP入门

1.6 MVC设计模式

1.1 Internet与万维网

- 1.1.1 主机和IP地址
- 1.1.2 域名和DNS
- 1.1.3 万维网概述
- 1.1.4 服务器和浏览器

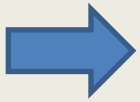
1.1.1 主机和IP地址

- 主机：连接到Internet上的所有计算机
- IP(Internet Protocol)地址：唯一标志每台主机的网络地址。

如某计算机的IP地址可表示为：

10101100 00010000 11111110 00000001

二进制表示法

172.16.254.1  点分十进制表示法

1.1.2 域名和DNS

- **域名(domain name)：** 是由一串用点分隔的名字组成的某一台主机或一组主机的名称。

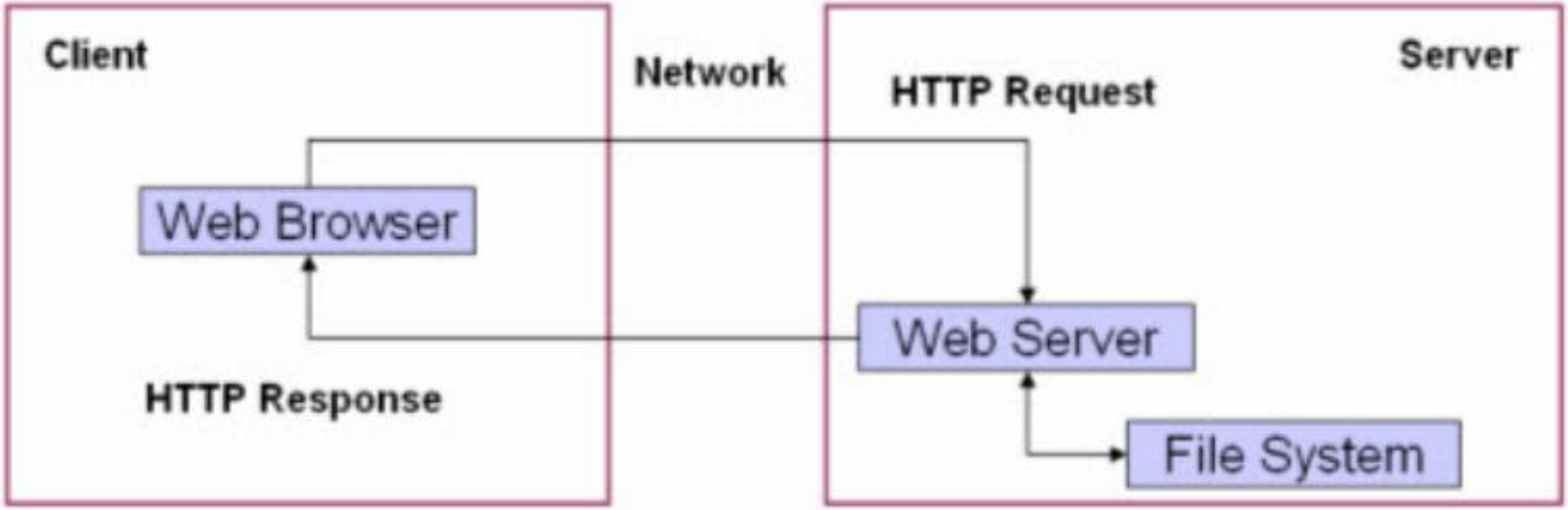
  如：**www.ecust.edu.cn**

- **DNS(Domain Name System)：** 把域名转换成IP地址

  如：**www.ecust.edu.cn** ⇨ **16.228.241.2**

1.1.3 万维网概述

- WWW —World Wide Web—万维网—简称Web。
- 体系结构：Web服务器、Web客户端、通信协议。



- 万维网联盟(World Wide Web Consortium, W3C)，致力于进一步开发Web、对协议进行标准化等工作。主页是： http://www.w3.org/Consortium/

1.1.4 服务器和浏览器

- **Web服务器**
  - 向浏览器提供服务的程序 。
  - 服务器是一种被动的程序，只有当其他计算机的浏览器向它们发出请求时才有所动作。
  - 目前最流行的服务器有**Apache** 服务器和**IIS** 服务器。
- **Web浏览器**
  - 浏览器是一个程序 。
  - 浏览器能够向服务器发出访问某个文档的请求。
  - 能够处理返回的响应并显示**Web**页面。

1.2 Web常用技术

- 1.2.1 HTTP概述
- 1.2.2 URL和URI
- 1.2.3 HTML和XML概述
- 1.2.4 CSS概述
- 1.2.5 JavaScript

1.2.1 HTTP概述

- **HTTP协议**

  -HTTP协议是一个基于请求-响应的无状态的协议。

- **运行机制**

  -客户向服务器发送一个对某种资源的 HTTP 请求,
  -服务器返回对所需要的资源的 HTTP 响应

1.2.2 URL和URI

- **URL（Uniform Resource Locator）**

  -指向Internet上位于某个位置的某个资源。资源包括HTML文件、图像文件和servlet等。

  例如： http://www.ecust.edu.cn/s/2/t/202/main.htm

- **URL的组成部分**

  -协议名称：包括 HTTP、FTP、TELNET、MAIL、FILE协议。

  -所在主机的DNS名或IP地址：www.ecust.edu.cn，116.228.241.2

  -可选的端口号:默认为80

  -资源在服务器上的相对路径和名称：相对路径是相对于服务器上web应用程序根目录。

➢**URI（Uniform Resource Identifier）统一资源标识符。以特定语法标识一个资源的字符串。**

•URI的组成部分

- 模式（schema）： **file（表示本地磁盘文件）、ftp（FTP服务器）、http（使用HTTP协议的Web服务器）、mailto（电子邮件地址）等。**

-模式特有的部分（ schema-specific-part）

例如：mailto:java-net@java.sun.com

news:comp.lang.java

ftp://ftp.scezju.com/

1.2.3 HTML和XML概述

- **HTML(HyperText Markup Language)：超文本标记语言 。它是由一些标签（tag）组成的文本文件，标签标识了文档的内容和类型，Web浏览器通过解析这些标签进行显示。**
  - **HTML语言的标记：HTML标记**
  - **HTML文档例子：程序1.1 register.html**
  - **运行:http://localhost/ch01/html/register.html**
- **XML(Extensible Markup Language)：可扩展标记语言。用于描述结构化数据。XML文档例子**

1.2.4 CSS概述

➢**CSS（Cascading Style Sheets）：层叠样式表。**

它是用来表现HTML或 XML等文件样式的语言。

（1）内联样式。

```
<p style="text-indent:2em">该段落首行缩进2em。</p>
```

（2）内部样式表

```
<style type="text/css">
h1 {color:#f00}
body{background-image:url
(images/bg.gif)}  /style>
```

（3）外部样式表。

```
<link href="css\layout.css"  rel="stylesheet"
type="text/css"/>
```

1.2.4 CSS概述

CSS常用属性及描述

- 程序1.2 index.html
- 程序1.3 layout.css
- 运行:http://localhost/ch01/html/index.html

1.2.5 JavaScript

- 在HTML页面中嵌入脚本，这些脚本是客户机上被执行的而不是在服务器上执行的。
- 从HTML 4.0开始，可以通过<script>标签来使用这样的脚本。最流行的客户端脚本语言是JavaScript。
- http://www.w3school.com.cn/b.asp

例如：一个包含JavaScript的页面源代码

程序1.4　inputCheck.html

运行:http://localhost/ch01/html/inputCheck.html

1.3 服务器资源

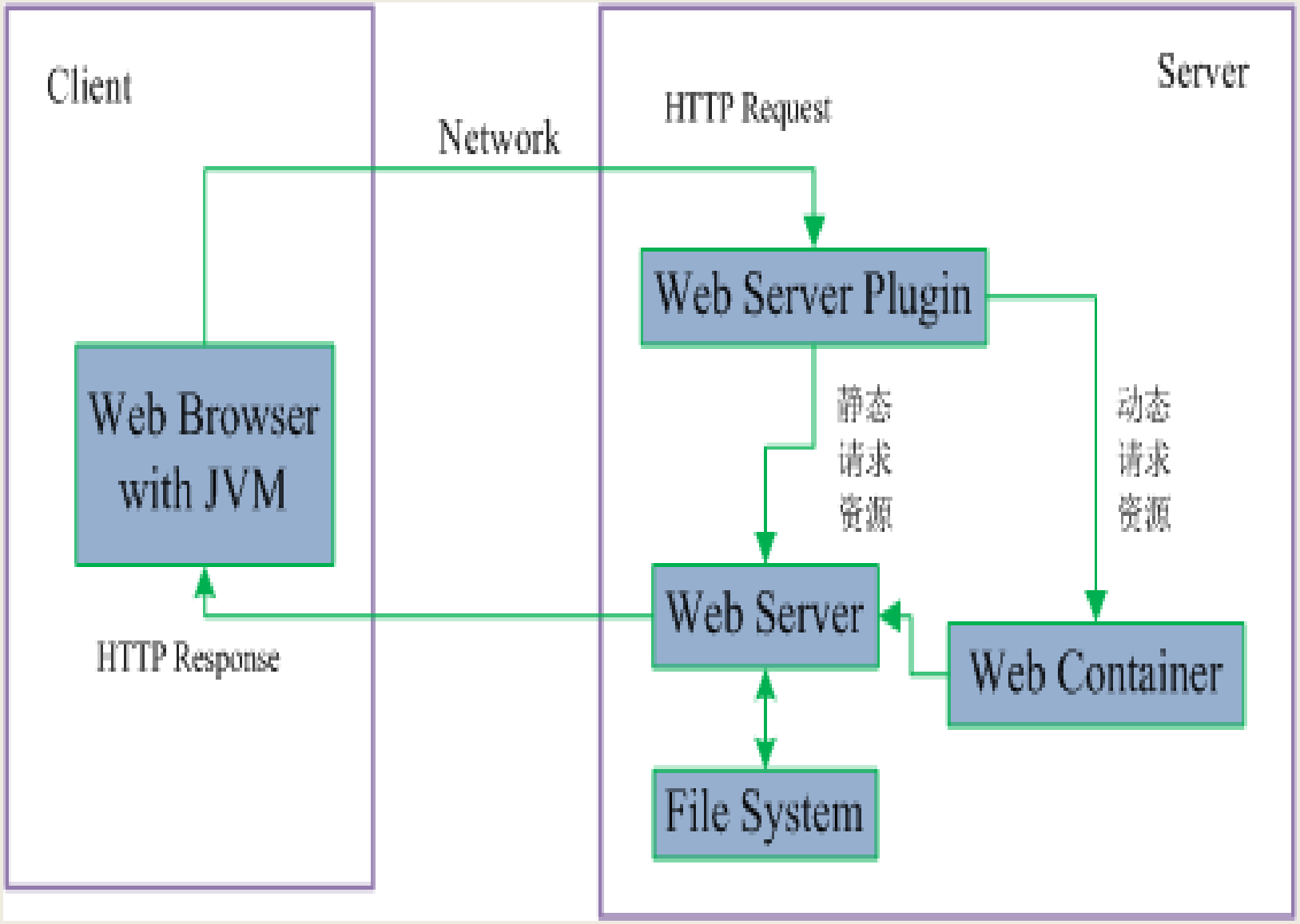
- 1.3.1 主动资源与被动资源
- 1.3.2 静态文档和动态文档
- 1.3.3 服务器端动态Web文档技术
- 1.3.4 客户端动态Web文档技术

1.3.1主动资源与被动资源

- 如果资源本身没有任何处理功能它就是被动的。
- 如果资源有自己的处理功能，它就是主动的。

1.3.2 静态文档和动态文档

- 静态文档是指文档创作完毕后就存放在Web服务器中，在被用户浏览的过程中，其内容不会改变。
- 动态文档是指文档的内容是根据需要生成的，可以分为服务器端动态文档和客户端动态文档。

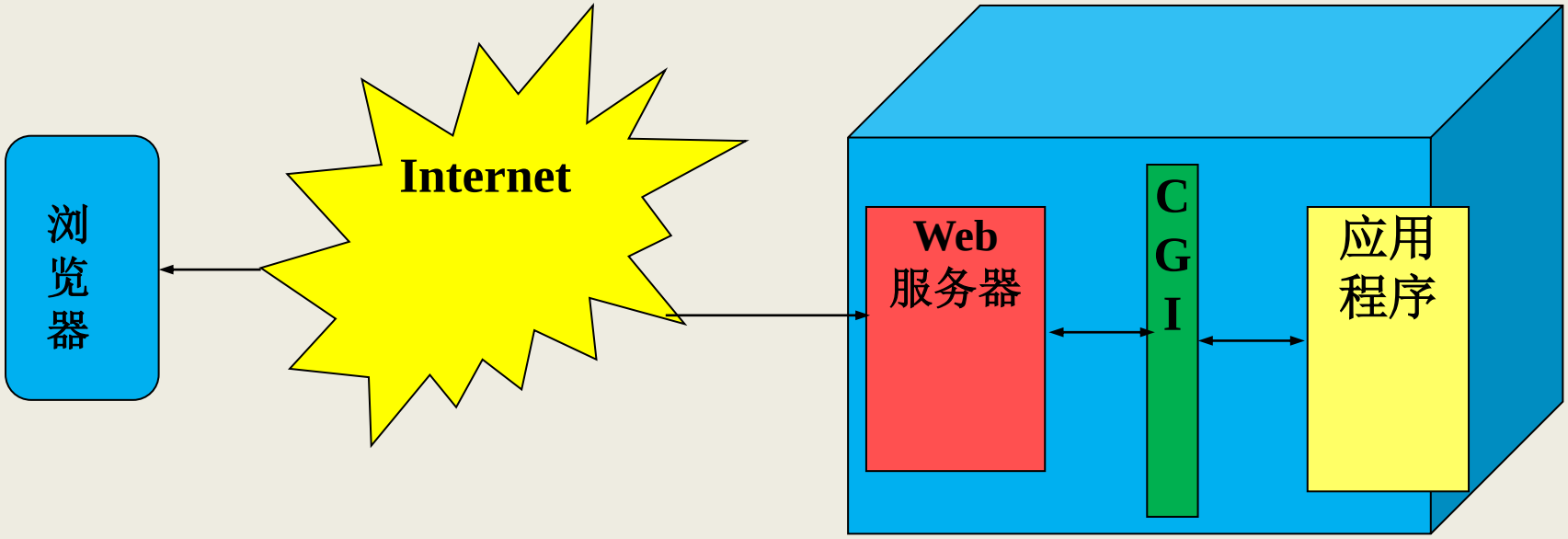

Client
Server
HTTP Request
Network
Web Server Plugin
Web Browser with JVM
静态请求资源
动态请求资源
Web Server
Web Container
HTTP Response
File System

1.3.3 服务器端动态Web文档技术

- CGI技术
- 服务器扩展技术
- HTML页面中嵌入脚本技术

CGI技术

- CGI（Common Gateway Interface，公共网关接口)是一种标准化的接口



•允许Web服务器与后端程序及脚本通信，这些后端程序和脚本能够接受输入信息（例如，来自表单），并生成HTML页面作为响应。

- **CGI程序可以用服务器支持的任何语言来编写，其中最常见的是Perl语言。**
- **服务器在接收到一个对CGI程序的请求时，不会返回该文件，而是运行该文件。**
- **CGI程序采用多进程的机制进行处理。每当一个新用户连接到服务器上时，服务器都会为其分配一个新的进程，**
- **CGI程序执行效率低。**

服务器扩展技术

- 使服务器支持单独的可执行模块，当服务器启动时该模块就装入内存并只初始化一次。然后，就可以通过已经驻留在内存的模块副本为每个请求提供服务。这些独立的可执行的模块称为服务器扩展。
- 在传统CGI中，如果有N个并发的对同一CGI程序的请求，则该CGI程序的代码在内存中重复装载了N次；而对于Servlet，处理请求的是N个线程，只需要一份Servlet类代码。

在HTML页面中嵌入脚本技术

- 在HTML页面中嵌入少量的脚本，然后让服务器来执行这些脚本以便生成最终发送给客户的页面。
- 常用技术包括:PHP、ASP和JSP

PHP是一种HTML内嵌式的语言。 它可以比CGI或Perl更快速地执行动态网页。服务器要求包含PHP的Web页面的文件扩展名为php。

ASP使用VB Script或JavaScript脚本语言编写嵌入在HTML网页中的代码来生成动态内容。使用这种技术的文件的扩展名为asp

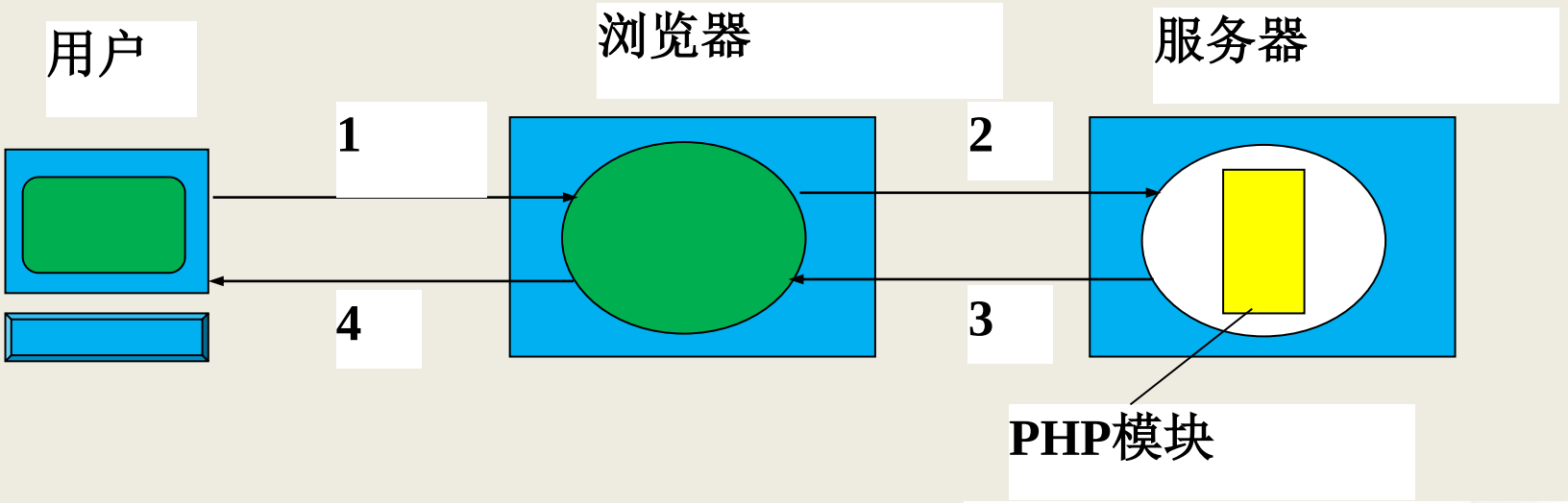
JSP是在传统的网页HTML文件(*.htm,*.html)中插入Java程序段(Scriptlet)和JSP标记(tag)，从而形成JSP文件(*.jsp)。

1.3.4 客户端动态Web文档技术

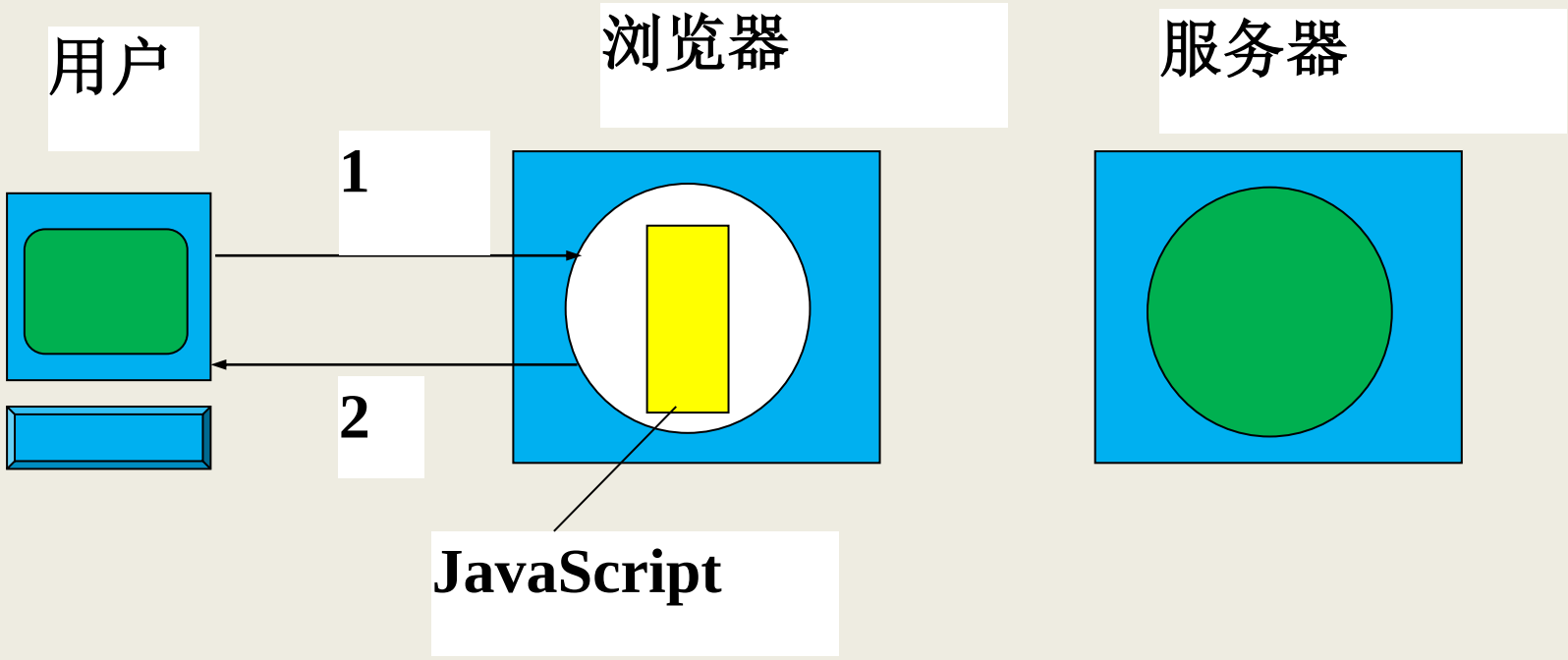
- 客户端动态文档技术的需求：
  - -直接与用户交互
  - -响应事件并处理
  - -客户端验证
- 客户端动态文档技术的实现：
  - -在HTML页面中嵌入脚本，这些脚本是客户机上被执行的而不是在服务器上执行的。
  - -通常使用JavaScript结合DOM技术实现客户端动态Web文档技术。

两种技术区别

- 服务器端动态文档技术的页面，是在服务器端执行的。
- 对一个PHP文件的请求，服务器首先执行该页面，PHP脚本将产生一个新的HTML页面，然后服务器将该页面送回给浏览器以便显示。



- 客户端动态文档技术的页面，是在客户端执行的。
- 对于上面的例子，当我们单击submit按钮时，浏览器解释执行该页面上包含的JavaScript函数。所有的工作都是在本地的浏览器内部完成。浏览器并没有与服务器联系。



1.4 Tomcat服务器

- 1.4.1 Tomcat下载与安装
- 1.4.2 Tomcat的安装目录
- 1.4.3 测试Tomcat
- 1.4.4 配置Tomcat的服务端口
- 1.4.5 Tomcat的启动和停止

1.4.1 Tomcat下载与安装

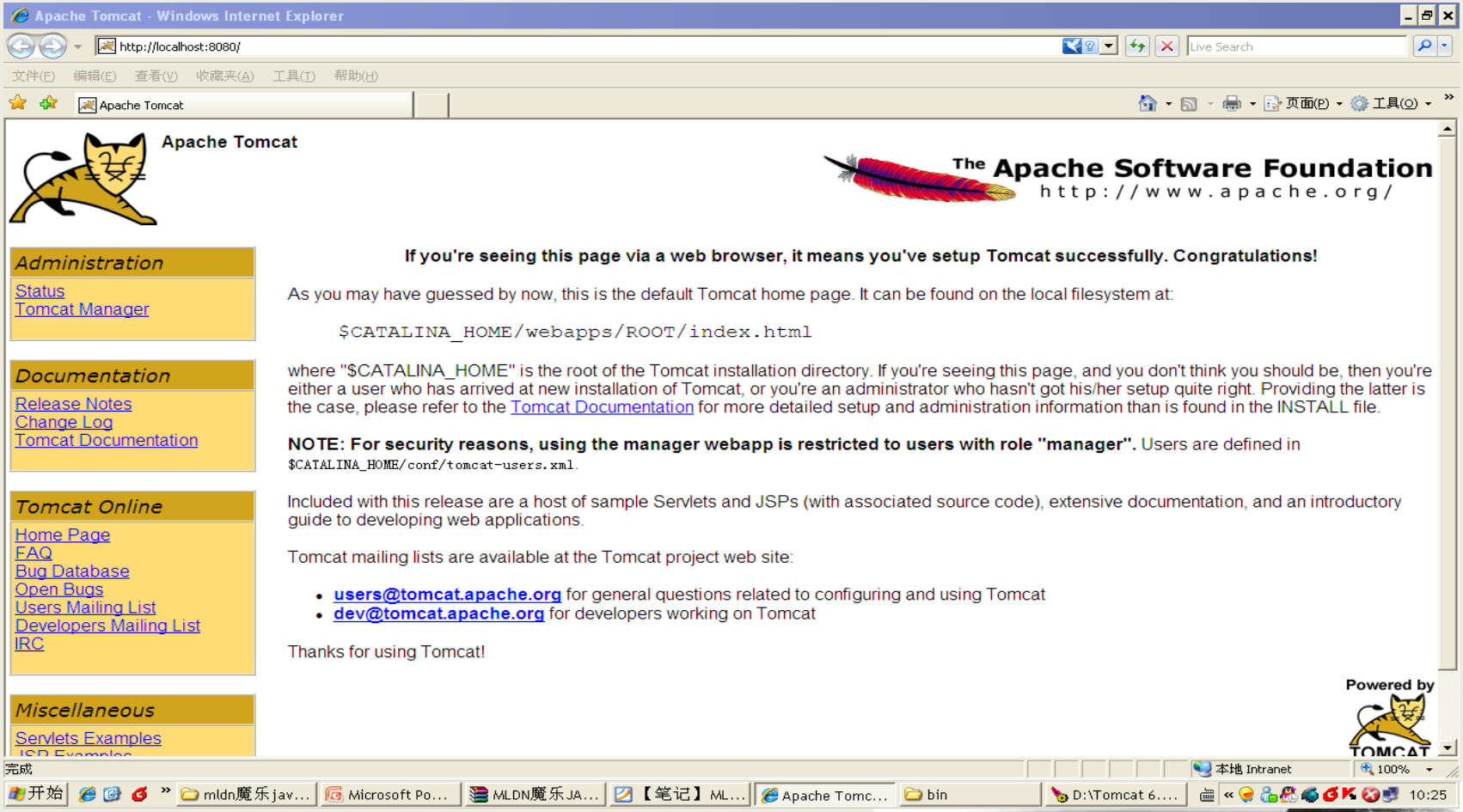
- Tomcat是由Sun公司和Apache 开发小组共同提出的合作项目Apache Jakarta项目下的产品，是为了使 Servlet/JSP能够与 Apache服务器一起运行而开发的Servlet/JSP容器。
- 下载网站:http://tomcat.apache.org/
- 可以下载Windows可执行的安装文件。
- 使用Tomcat时必须有JDK的支持，所以需要先配置JDK的安装环境，直接从www.sun.com上下载JDK的安装版直接安装

1.4.2 Tomcat的安装目录

目录	说明
/bin	存放启动和关闭Tomcat的脚本文件
/conf	存放Tomcat的各种配置文件,其中包括 server.xml、tomcat-users.xml和web.xml
/lib	存放Tomcat服务器及所有web应用程序都能访问的库文件。
/logs	存放Tomcat的日志文件
/temp	存放Tomcat运行时产生的临时文件
/webapps	存放所有web应用程序的根目录
/work	存放jsp页面生成的servlet源文件和字节码文件

1.4.3 测试Tomcat

启动浏览器,输入地址: http://localhost:80/,
出现下图,说明Tomcat 安装成功



1.4.4 配置Tomcat的服务端口

a) 找到Tomcat目录下的conf文件夹
b) 进入conf文件夹里面找到server.xml文件
c) 打开server.xml文件
d) 在server.xml文件里面找到下列信息

```
<Connector port="8080"
protocol="HTTP/1.1"
connectionTimeout="20000"
redirectPort="8443" />
```

e) 把port="8080"改成port= " 8088"，并且保存
f) 重新启动Tomcat，并且在IE浏览器里面的地址栏输入http://localhost:8088/

- 打开Servlet重新载入功能

a) 找到Tomcat目录下的conf文件夹

b) 进入conf文件夹里面找到context.xml文件

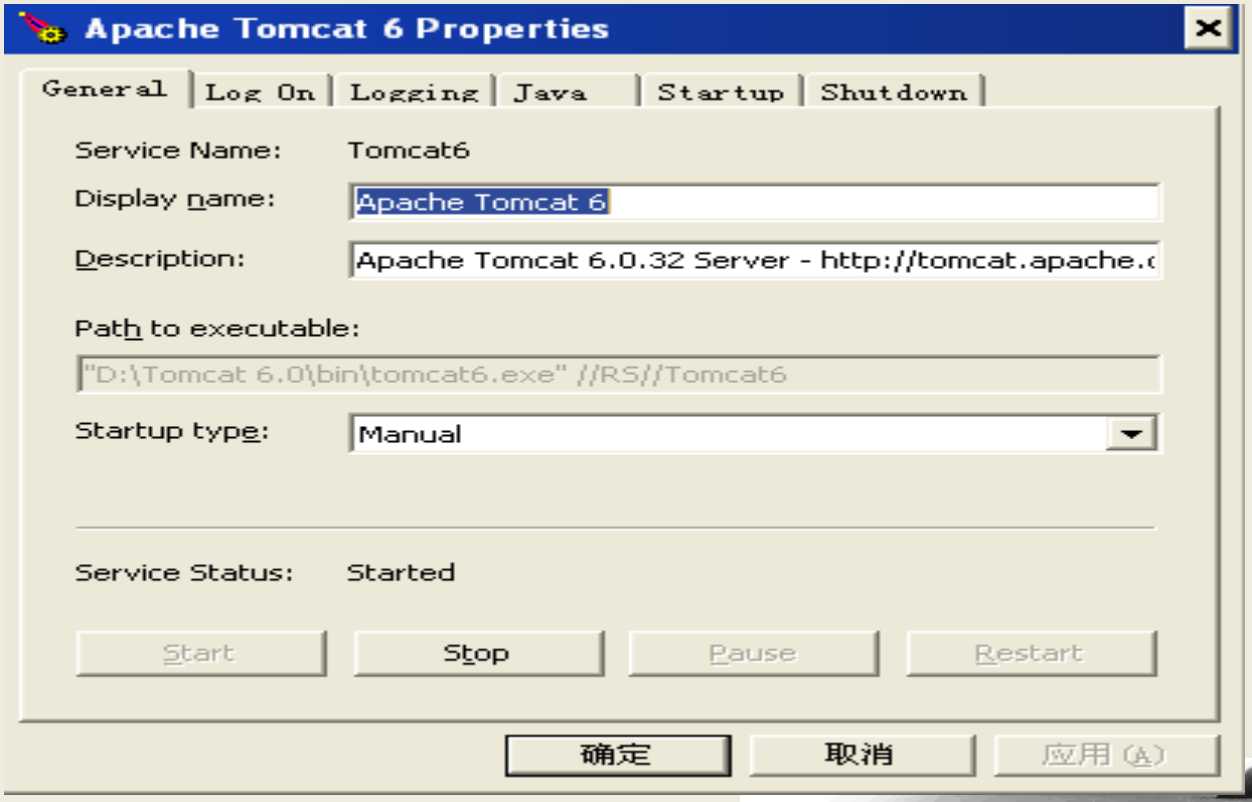
c) 打开context.xml文件

d) 在context.xml文件里面找到< Context >元素，填加一个reloadable属性，其值为true 即"reloadable="true"

1.4.5 Tomcat服务器的启动和停止

a) 打开找到Tomcat目录下bin文件夹中的tomcat6w.exe

b) 也可以打开如下窗口启动和停止Tomcat服务器



1.5 Servlet与JSP入门

- 1.5.1 Servlet
- 1.5.2 Web容器
- 1.5.3 JSP页面

1.5.1 Servlet

- Servlet一般翻译成服务器端小程序，它是使用Servlet API以及相关的类编写的Java程序。主要用来扩展Web服务器的功能。

Servlet的开发步骤 方法一

1、编写Servlet代码 HelloServlet.Java

```
package com.demo;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class HelloServlet extends HttpServlet{
    public void service(HttpServletRequest request, HttpServletResponse response)
            throws ServletException,IOException{
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<body>");
        out.println("<font color='#0000ff'>");
        out.println("<h3>Hello,World!</h3>");
        out.println("The time now is:"+new java.util.Date());
        out.println("</body>");
        out.println("</html>");
    }
}
```

2. 编译Servlet代码

- **import javax.servlet.*;**

  import javax.servlet.http.*;

  不属于JDK的标准包，包含在Tomcat目录中，位于<CATALINA_HOME>\lib\servlet-api.jar文件中。必须将该文件所在的路径添加到Java编译工具下。

- **Eclipse**是一个用于Java程序设计的集成开发环境，具有编辑、调试、运行Java程序的功能。下载地址为：http://www.eclipse.org/downloads/

•编译好的servlet类文件为**HelloServlet.class**

3. 部署Servlet

- Servlet是Web应用程序的一个组件。Web应用程序具有严格定义的目录结构。在Tomcat中，每个应用程序都应在安装目录的webapps目录下有一个目录。
- 假设将该Servlet部署到ch01的Web应用中，因此首先在webapps目录下建立一个名为ch01的子目录，在该目录中再建立WEB-INF子目录，在它的下面建立classes子目录和lib子目录，这样就建立了一个ch01的Web应用程序。

- 将编译好的HelloServlet.class复制到<CATALINA_HOME>\webapps\ch01\WEB-INF\classes\com\demo文件中
- 在WEB-INF目录下建立一个web.xml的文本文件，描述了该Servlet 的部署信息。

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd”   version="2.5">
  <servlet>
    <servlet-name>helloServlet</servlet-name>
    <servlet-class>com.demo.HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>helloServlet</servlet-name>
    <url-pattern>/hello</url-pattern>
</servlet-mapping>
</web-app>
```

4. Servlet的运行

地址栏中输入下面的URL：

http://localhost/ch01/hello

Servlet的开发步骤 方法二

1、编写Servlet代码

程序1.5 HelloServlet2.java

2、编译Servlet代码　HelloServlet2.class

3、部署Servlet

将编译好的HelloServlet2.class复制到<CATALINA_HOME>\webapps\ch01\WEB-INF\classes\com\demo文件中

4. Servlet的运行

http://localhost/ch01/helloServlet.do

Servlet的优缺点

优点：

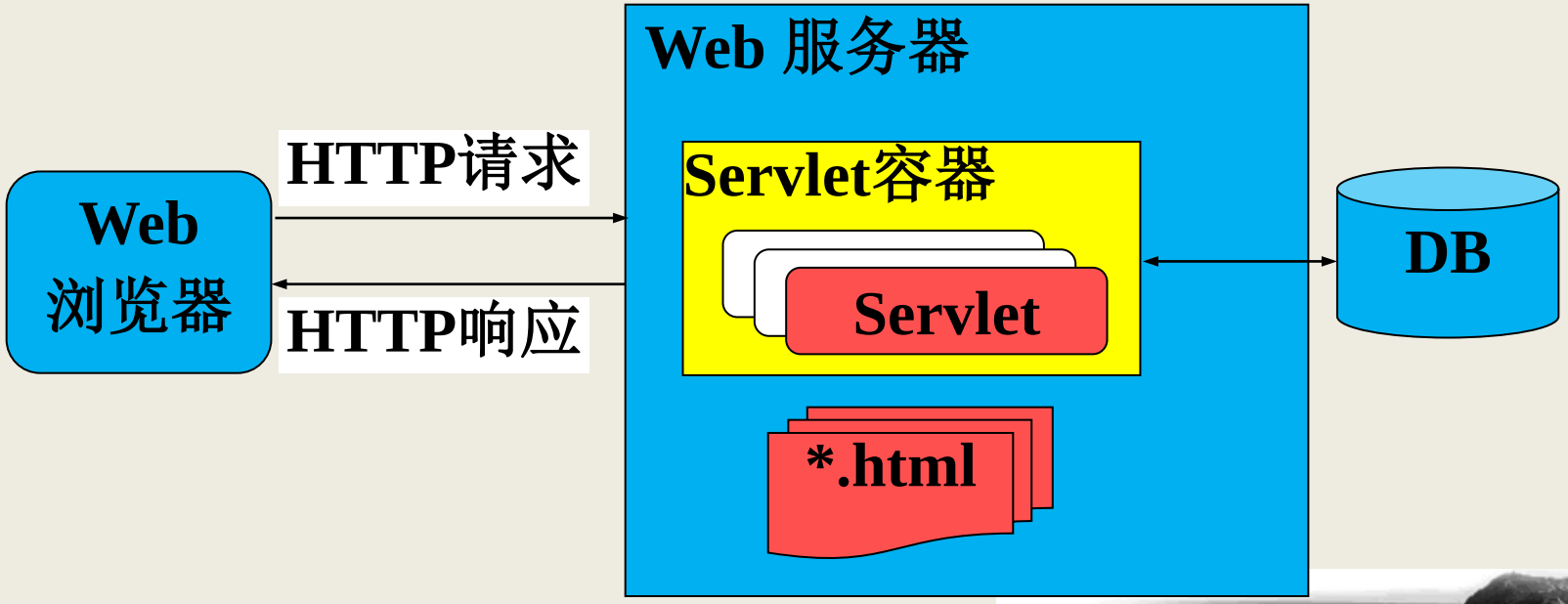
- 高效性。每个请求由一个轻量级的Java线程处理
- 方便性。提供了大量的实用工具例程
- 功能强大。许多使用传统CGI程序很难完成的任务都可以轻松地完成。
- 可移植性好 。为一个服务器编写的Servlet无需任何实质上的改动即可移植到其他服务器上。
- 节省投资 。有许多廉价甚至免费的Web服务器可供个人或小规模网站使用

缺点：

- 缺点是它经常既包含业务逻辑又包含表示逻辑。
- 表示逻辑（presentation logic）是展示给用户的信息，在Servlet中产生HTML响应就是表示逻辑。
- 业务逻辑（business logic）是完成某种数据处理和存储任务的功能。
- Sun在推出Servlet技术后不久又推出了JSP技术。有了JSP技术就可以实现业务逻辑和表示逻辑的分离：
  - Servlet专门处理业务逻辑
  - 用JSP实现表示逻辑。

1.5.2 Web容器

- Web服务器使用一个单独的模块装载和运行Servlet。这个专门用于Servlet管理的单独模块称为Servlet容器（container），或称Web容器。
- Tomcat就是一个Servlet容器。



1.5.3 JSP页面

- JSP（Java Server Pages）页面是包含Java代码和HTML标签的Web页面。
- 由主动的JSP标签和被动的HTML标签混合而成的Web页面。
- 程序1.6 hello.jsp

  http://localhost/ch01/jsp/hello.jsp

Servlet 和JSP的使用

- 可以把Servlet看成是含有HTML的Java代码。
- 可以把JSP看成是含有Java代码的HTML页面。
- 用Servlet可以实现JSP的功能，用JSP也可以实现Servlet的功能 。
  - JSP页面主要实现可视化的表示逻辑 。
  - Servlet主要用来实现业务处理和控制逻辑 。

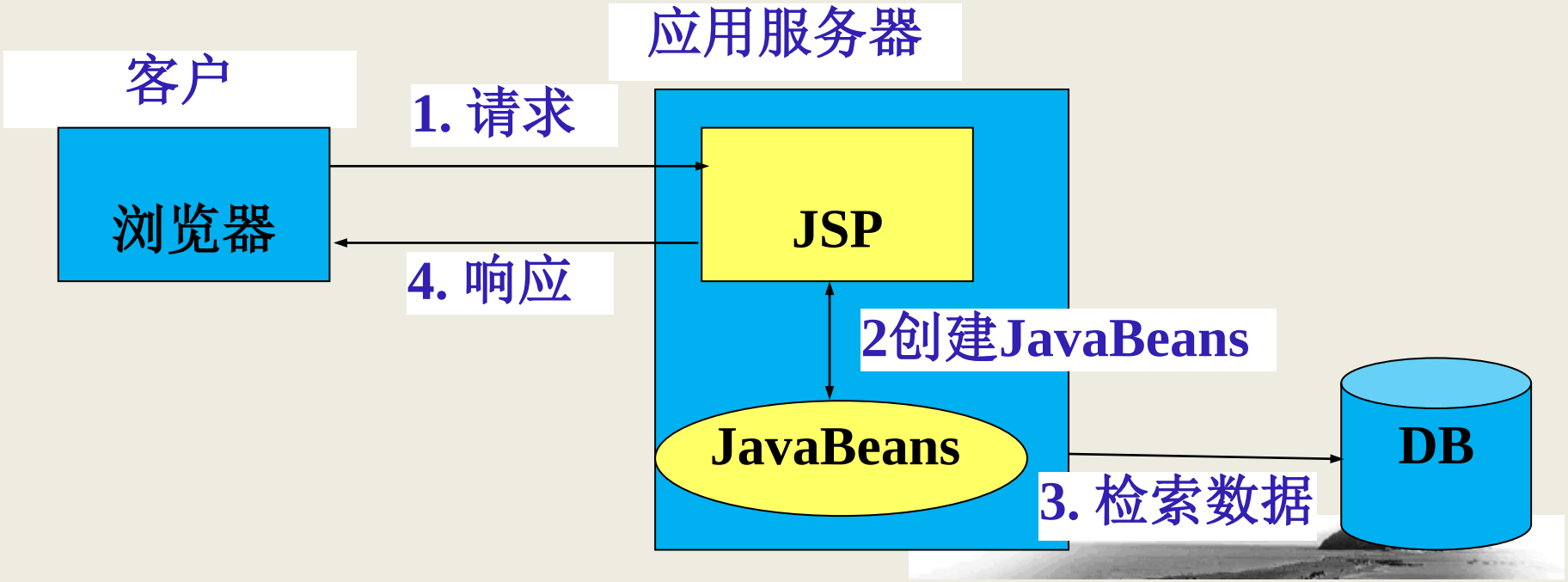
1.6 MVC设计模式

Sun公司在推出JSP技术后提出了建立Web应用程序的两种体系结构方法。

- 1.6.1 Model 1体系结构
- 1.6.2 Model 2体系结构

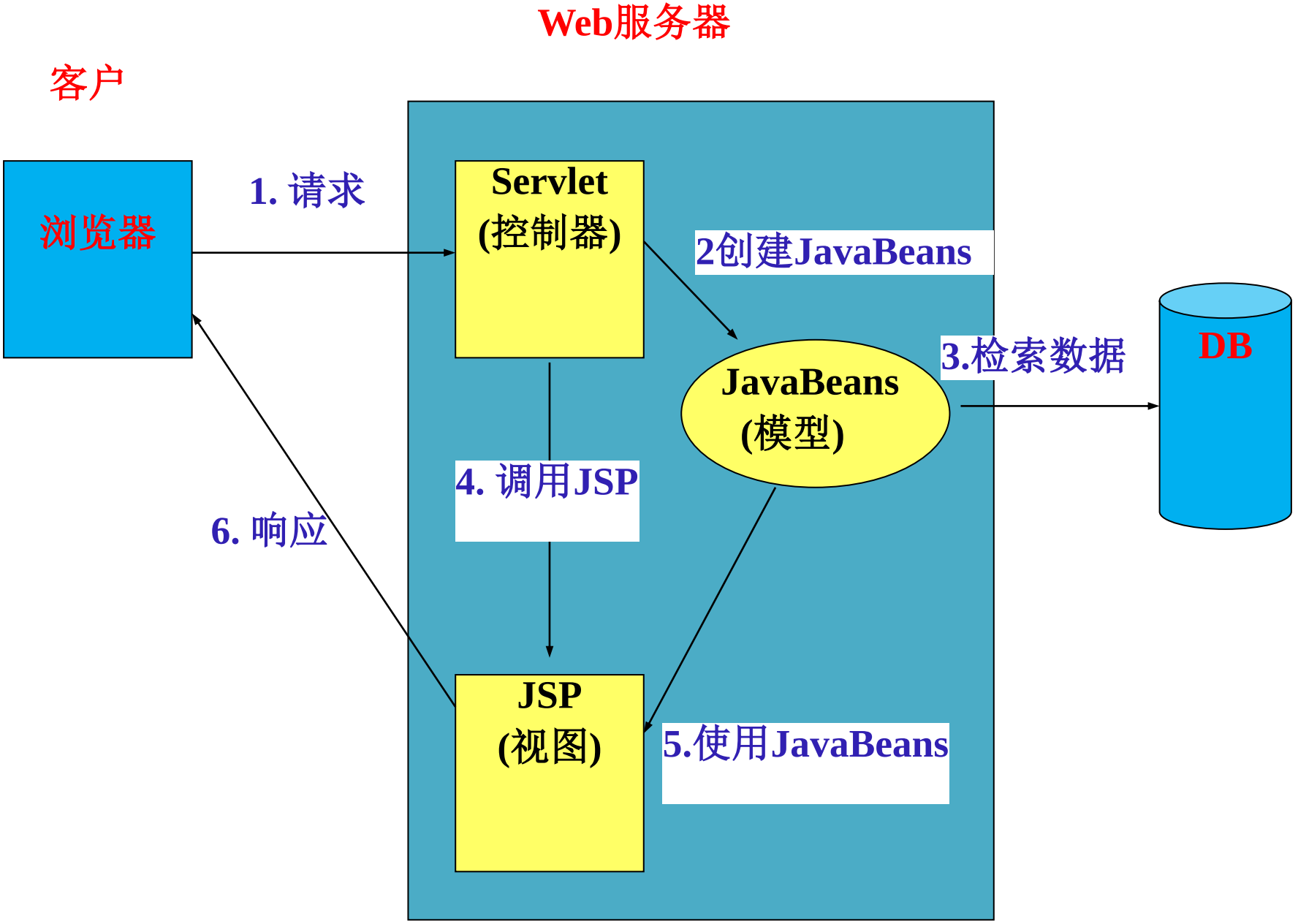
1.6.1 Model 1体系结构

在Model 1体系结构中，每个请求的目标都是JSP页面。该页面完全负责完成请求所需要的所有的任务，其中包括验证客户、使用JavaBean访问数据库及管理用户状态等。



1.6.2 Model 2体系结构

- 遵循MVC（Model-View-Controller）的设计模式。
- 所有请求的目标都是Servlet，它充当控制器（controller）。Servlet分析请求并将产生响应所需要的数据收集到JavaBeans对象中，该对象作为应用程序的模型（model）。最后，Servlet控制器将请求转发到JSP页面。这些页面使用存储在JavaBean中的数据产生响应。


Web服务器
客户
浏览器
1. 请求
Servlet
(控制器)
2创建JavaBeans
JavaBeans
(模型)
3.检索数据
DB
4. 调用JSP
6. 响应
JSP
(视图)
5.使用JavaBeans

1.7 本章小结

- 本章概述了Web应用开发的主要技术和基本原理。其中包括Web技术的基本概念、浏览器和服务器的概念、 HTTP协议、动态Web文档技术等 。
- 本章还简要介绍了Tomcat服务器的安装与配置，讨论了什么是Servlet和 Servlet容器，给出了一个简单的Servlet的开发执行过程。
- 简单介绍了MVC设计模式。