第一章

1.3.3 JSP 概述(page5)

- jsp的最大优点是：开放的，跨平台的结构，它可以运行在所有的服务器系统上。
- JSP与ASP的区别：
    - 每次访问asp文件，服务器都要将该文件解释一遍，然后将标准的HTML代码发送到客户端；
    - 而JSP有所不同，当第一次访问JSP文件时，服务器先将JSP编译成二进制码，以后再访问的时候，就直接访问二进制码，这样大大提高了执行的效率。

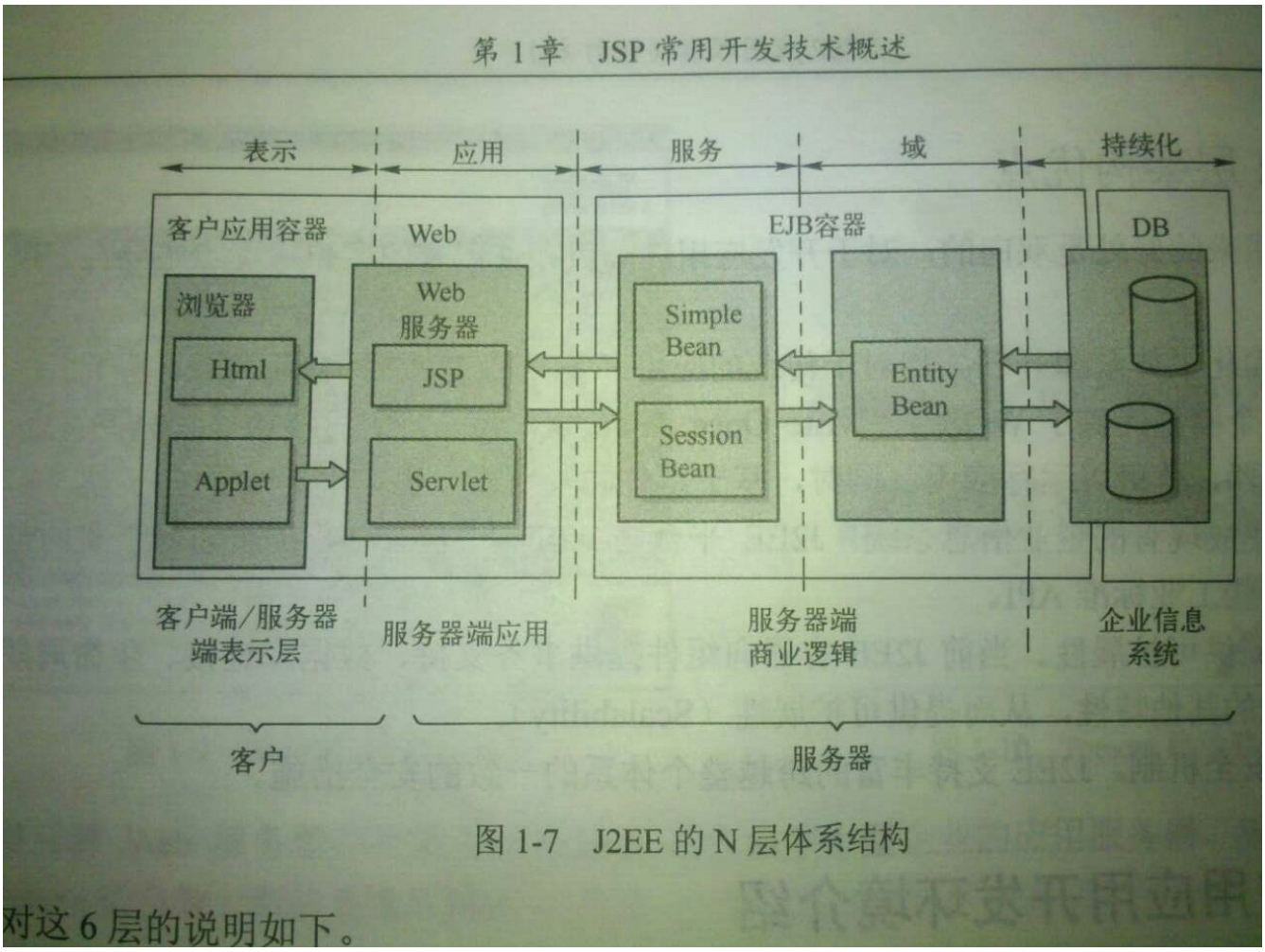
  总结：**JSP**一次编译，多次运行；而**ASP**每次运行需要每次编译。
- JSP的优点如下：
    1. 多平台的支持，可以再所有的服务器操作系统上运行。
    2. 编译后执行，能够大大提高执行效率。
    3. JSP采用Java技术，Java应用比较普遍，因此学习起来比较容易。
    4. JSP是J2EE十三种核心技术中的一种，可以和其他核心技术共同建立企业应用。
- JSP的缺点如下：
    1. 开发环境相对ASP来说，比较复杂。需要先安装JDK，然后安装Web服务器。
    2. 相对ASP的VBsript脚本语言，Java语言学起来稍微复杂。

1.4.3 J2EE体系概述(page6)

- J2EE的任务是提供一个平台独立的，便携式，多用户，安全及标准的企业级平台。
- J2EE的13种核心技术包括（不全，常见的）：
    1. **JDBC** (Java Database Connectivity, Java数据库连接)
    2. **EJB** （Enterprise Java Bean，企业JavaBean）
    3. **JSP** （Java Server Pages, Java服务器端网页）
    4. **Servlet** （服务器端小程序）
    5. **XML** （eXtensible Markup Language）
    6. **JavaMail** (Java邮件)
    7. **JTS** （Java Transaction Service, Java事务服务）
    8. 。。。

1.5.1 N层开发框架（page8）

- 理想的J2EE体系包括**6**个层：表示，应用，服务，域，连通性和持续化。
- 这些层横跨客户机和服务器，而它们逻辑上划分为**Web**容器，**EJB**容器和数据库，如图1-7所示：



第二章

2.3 安装和配置Tomcat(page15)

- Tomcat 服务器的默认端口是 **8080**
- 访问Tomcat的默认路径是：**http://localhost:8080**

2.3.1 配置Tomcat(page18)

- 更改服务端口：在Tomcat下打开conf文件夹找到**server.xml**文件用记事本打开在**< connecter port ="8080" >**中把8080更改即可

2.3.2 测试第一个JSP页面(page20)

- 第一段JSP程序（Hello World）

```
<%
out.print("Hello World");
%>
```

- 对这个程序做简单的说明：
    - 所有的JSP脚本程序都必须用**“<%”**和**“%>”**括起来；
    - 可以用**out**对象的**print**方法输出信息，输出的字符需要用双引号括起来；
    - 每一条JSP语句的末尾用分号结束。

第三章 Servlet编程技术

3.1.1 Servlet概念(page45)

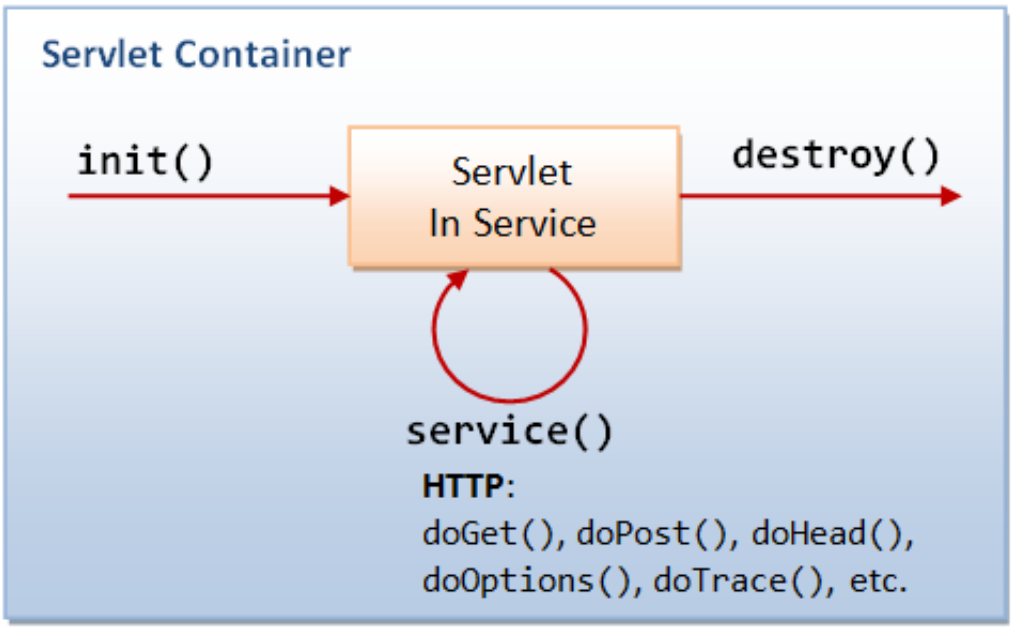
- Servlet

> **Servlet**是一个标准的**Java**类，它的特殊之处是可以处理**HTTP**请求。

- JSP与Servlet有什么关系？

> **JSP**是以另一种方式实现的**Servlet**，**Servlet**是**JSP**的早期版本，在**JSP**中，更加注重页面的表现，而在**Servlet**中则更注重业务逻辑的实现。

3.1.2 Servlet的生命周期(page45)



- **init()**方法：服务器初始化**Servlet**

> 该方法描述为：
> **public void init(ServletConfig config)throws ServletException**
>
> 当**Servlet**第一次被加载时，服务器调用**init()**方法初始化一个**Servlet**对象

- **service()**方法:初始化完毕，**Servlet**对象调用该方法响应客户端请求

> 该方法描述为：
> **public void service(HttpServletRequest request, HttpServletResponse response)throws ServletException, IOException**
>
> **service()**方法可以被多次调用，每次用户请求都导致**service()**方法被调用，调用过程运行在不同的进程中，互不干扰

- **destroy()**方法：调用该方法消灭**Servlet**对象

> 该方法描述为：
> **public void destroy()**
> 当**Servlet**引擎终止服务时，比如关闭服务器等，**destroy()**方法会被执行，销毁**Servlet**对象。

```
总结：在整个Servlet的生命周期中，init()方法和destroy()方法都只被调用一次，
而service()方法被调用多次。
```

3.3.2 GenericServlet类和HttpServlet类 (page51)

> 有上述继承树可知：**GenericServlet**是父类，它实现了**Servlet**接口中的**service()**方法，但它还是一个抽象方法，所有子类都应该实现这个方法。**HttpServlet**继承了**GenericServlet**类，自然也要实现**service()**方法。

- service()方法
  - HttpServlet类有两种形式的service()方法：public 和 protected，描述如下：

    > **public void service(ServletRequest request, ServletResponse response) throws ServletException, IOException**
    >
    > **protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException**

  - 当包容器为一个Servlet收到一个请求时，这个方法的调用顺序是：

> 1. 包容器调用**public**的**service()**方法；
> 2. 在把参数分布转换为**HttpServletRequest**和**HttpServletResponse**后，**public**的**service()**方法调用**protected**的**service()**方法；
> 3. 根据**HTTP**请求方法的类型，**protected**的**service()**调用**doXXX()**方法之一。

注意：常用的**doXXX()**方法有**doGet()**和**doPost()**，我们编写的**Servlet**类继承**HttpServlet**之后，只需要覆盖**doGet()**或**doPost()**方法即可，不需要覆盖**service()**方法。

第四章 JSP编程技术(重点)

- JSP的主要内容包括：3大编译指令，7大操作指令(课本6个)和9大隐含对象(课本7个)。

4.2 JSP页面结构(page68)

- JSP的页面主要包含3种元素：编译指令，操作指令和JSP代码。
    - 编译指令
      > 编译指令告诉JSP解释引擎，需要在编译时做什么动作，例如引入其他类，设置JSP页面使用什么语言编码等。
    - 操作指令
      > 操作指令则是在JSP页面被请求时动态执行的，比如可以根据某个条件动态跳转到另一个页面。
    - JSP代码
      > JSP代码指我们嵌入JSP页面中的Java代码，分两种：变量和方法的声明，使用“<!%”和“%>”标记，另一种是常用到的用“<%”和“%>”包含的JSP代码块。

4.3 编译指令(page68)

- JSP指令的一般形式：<%@ 指令名 属性1 = "value1" %>，<%@ 指令名 属性2 = "value2" %>，多个指令也可以合并写为：<%@ 指令名 属性1 = “value1” 属性2 = "value2" %>
- JSP的3大编译指令：page指令，include指令，taglib指令(不考)。
    - page指令：page 指令是最复杂的JSP指令，它的主要功能为设定整个JSP 网页的属性和相关功能。

属性	定义
language=“语言"	主要指定JSP 容器 要用什么语言来编译JSP 网页。JSP 1.2规范中指出，目前只可以使用Java 语言，不过未来不排除增加其他语言，如C、C++、Perl 等等。默认值为Java语言
extends=“基类名"	主要定义此JSP 网页产生的Servlet 是继承哪个父类
import= "importList"	定义此JSP 网页可以使用哪些Java类库
errorPage="error_url"	表示如果发生异常错误时，网页会被重新指向指定的URL

isErrorPage="true | false"	表示此JSP Page 是否为专门处理错误和异常的网页
contentType = "ctinfo"	表示MIME 类型和JSP 网页的编码方式，其作用相当于HttpServletResponse接口的setContentType()方法
isThreadSafe="true | false"	告诉JSP 容器，此JSP 网页是否能同时处理多个请求。默认值为true，如果此值设为false， 转义生成的Servlet会实现SingleThreadModel接口。
session="true | false"	决定此JSP 网页是否可以使用session 对象。默认值为true

- include指令(page70):用来指定怎样把另一个文件包含到当前的JSP页面中，这个文件可以是普通的文本文件，也可以是JSP页面。例如：**<%@include file="login.htm"%>**。使用include指令，可以实现**JSP**页面的模块化，使**JSP**的开发和维护变得非常简单。

4.4 操作指令**(page71)**

- 操作指令和编译指令不同，编译指令是通知Servlet引擎的处理消息，而操作指令只是运行时的动作。编译指令在将JSP编译成Servlet时起作用；而操作指令通常可替换成**JSP**脚本，它只是**JSP**脚本的标准化写法。

- JSP操作指令主要有如下的6个：

  1. jsp:include：用于引入静态和动态的资源，功能和include指令相同。格式：<jsp:include page="test.htm"/>
  2. jsp:forward：执行页面转向，将请求的处理转发到下一个页面。格式：<jsp:forward page="test.htm"/>
  3. jsp:param：用于传递参数，必须与其他支持参数的标签一起使用。

     ```
     <jsp:forward page="mypage.jsp">
     <jsp:param name="param1" value="value1"/>
     <jsp:param name="param2" value="value2"/>
     </jsp:forward>
     ```

  4. jsp:useBean：创建一个JavaBean的实例。
  5. jsp:setProperty：设置JavaBean实例的属性值。
  6. jsp:getProperty：输出JavaBean实例的属性值
  7.

- JSP的操作指令均以**“/>”**结束。

4.5 JSP代码(page72)

- 变量和方法(略)
- 代码块(略)

4.6 7大隐含对象(page75)

- 隐含对象：有些对象不用声明就可以在JSP页面的脚本代码和表达式中随意使用，这些缺省对象统称为隐含对象。
- 7大隐含对象：
    - **out** 页面输出对象；
    - **request** 请求对象；
    - **response** 响应对象；
    - **application** 应用对象；
    - **session** 会话对象；
    - **cookie** 对象；
    - **pageContext** 对象；

对象名	对象类型	作用域	定义
out	JspWriter	page	一个输出的缓冲流，给浏览器的客户返回内容
request	HttpServletRequest	request	用来得到客户端的信息
response	HttpServletResponse	page	处理服务器端对客户端的一些响应
application	ServletContext	application	用来保存网站的一些全局变量
session	HttpSession	session	用来保存单个用户访问时的一些信息
cookie	Cookie	不清楚呢	将服务器端的一些信息写到客户端的浏览器中
pageContext	PageContext	page	提供了访问和放置页面中共享数据的方式

4.7 response对象(page76)

- 网页转向：sendRedirect()方法，跳转到其他页面
  > 格式：response.sendRedirect("URL 地址");
- 与<jsp:forward>的最大区别
  > 使用<jsp:forward>只能在本站内跳转，但用response.sendRedirect("URL 地址")可以跳转到任何一个地址的页面。
- 其他(略，自己看书)

4.8 request对象(page78)

- 自己看书