java.net

Java程序设计

第01-03讲 Servlet模型（二）

Java课程组

本讲教学目标

- 理解Servlet的生命周期
- 熟悉Servlet相关对象

知识回顾/本讲先行知识

- Servlet编写、访问的过程
- Servlet中的请求与响应
- Servlet中的请求数据的传递、请求转发与页面跳转

本讲内容

- Servlet应用开发接口
- Servlet的生命周期
- Servlet应用进阶
- web.xml与@WebServlet注解

Servlet应用开发接口

- 对Servlet应用开发接口及其功能的掌握，决定了是否能做好Servlet开发工作

javax.servlet

抽象层次较高的Servlet接口和类

javax.servlet.genericservlet

不限定协议的Servlet

javax.servlet.http

提供了基于HTTP协议的Servlet基本功能

Servlet接口

方法	功能描述
init()	在Servlet实例化之后，调用Service之前调用init()方法
service()	调用此方法允许Servlet响应请求，在Servlet成功初始化之前无法调用
destroy()	当一个Servlet被从服务中去除时，Servlet容器会调用此方法。在这个对象service()方法所有的线程未全部退出或者没有被容器认为发生超时操作时，此方法不会调用
getServletConfig()	返回一个ServletConfig对象，作为一个Servlet的开发者，应该通过init()方法存储ServletConfig对象以便这个方法能返回这个对象
getServletInfo()	允许Servlet向主机的Servlet运行者提供有关信息。返回的字符串应该是纯文本格式而不是任何标志（HTML、XML）

GenericServlet抽象类

- 实现了Servlet接口
- 提供了Servlet接口中除service()方法外4个方法的简单实现
- 子类必须实现service()方法
- 未指定协议的Servlet

HttpServlet抽象类

- 基于HTTP协议的Servlet
- 继承自GenericServlet
- 必须重写doXXX()方法处理业务逻辑/返回响应
- 扩展
  - protected void service(HttpServletRequest arg0, HttpServletResponse arg1)
  - public void service(ServletRequest arg0, ServletResponse arg1)

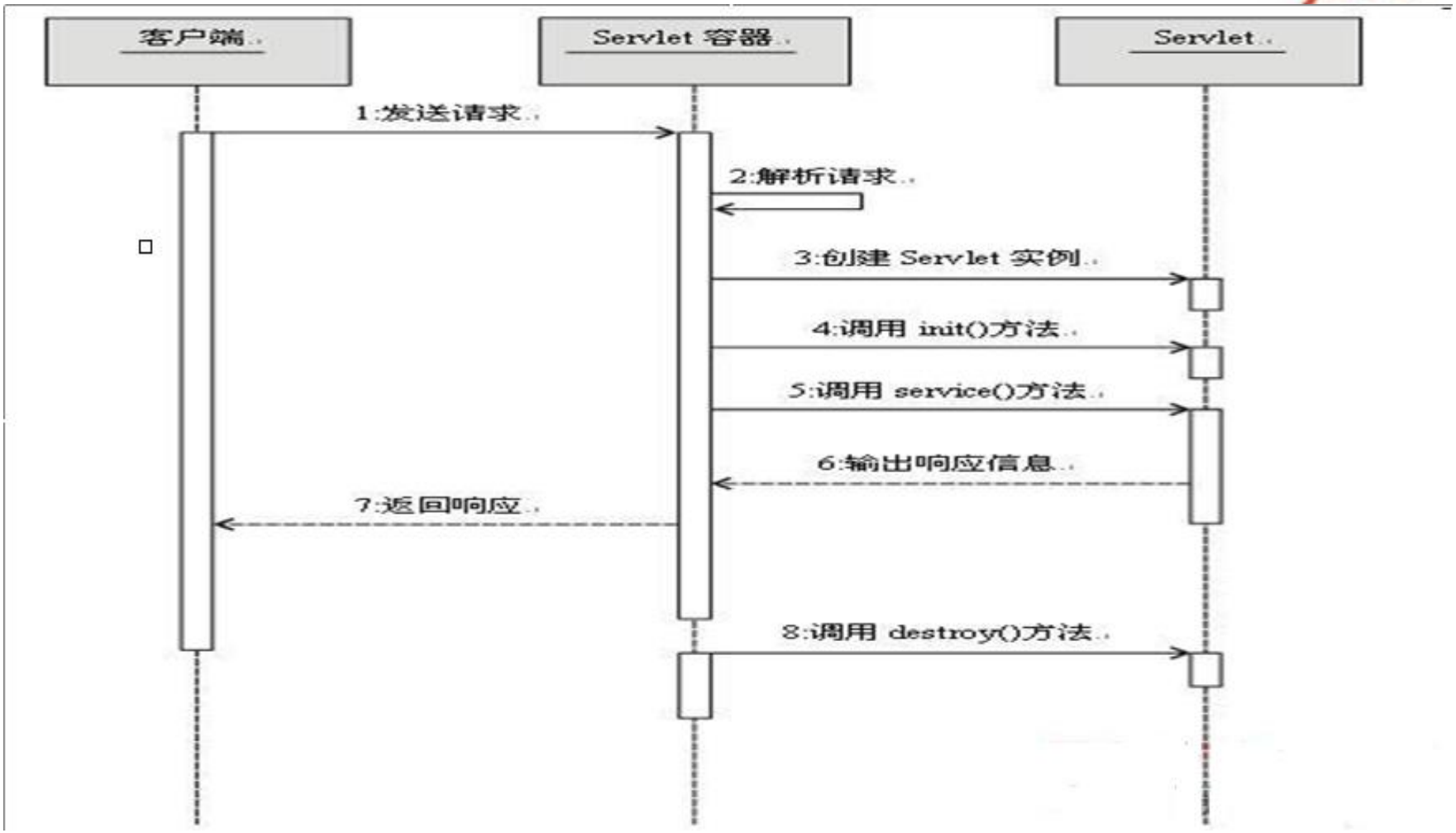
本讲内容

- Servlet应用开发接口
- Servlet的生命周期
- Servlet应用进阶
- web.xml与@WebServlet注解

Servlet生命周期

方法	功能描述
init()	在Servlet实例化之后，调用Service之前调用init()方法
service()	调用此方法允许Servlet响应请求，在Servlet成功初始化之前无法调用
destroy()	当一个Servlet被从服务中去除时，Servlet容器会调用此方法。在这个对象service()方法所有的线程未全部退出或者没有被容器认为发生超时操作时，此方法不会调用
getServletConfig()	
getServletInfo()	

Servlet生命周期序列图



Servlet什么时候实例化

- 在服务器启动加载网站时
- 第一次访问Servlet时
- 具体什么时候加载需要通过web.xml配置文件中，对应Servlet设置项的<load-on-startup>节点进行设置
  - 标记容器是否在启动的时候就加载这个servlet
  - 当值为0或者大于0时，表示容器在应用启动时就加载这个servlet
  - 当是一个负数时或者没有指定时，则指示容器在该servlet被选择时才加载

Servlet对init方法的调用

- init(ServletConfig config)方法
  - 在加载Servlet实例成功后调用init()方法，一般作一些初始化的工作
  - 在一个Servlet的生命周期中只执行一次
  - 通过config. getInitParameter(*Param*)可取得web.xml中的参数

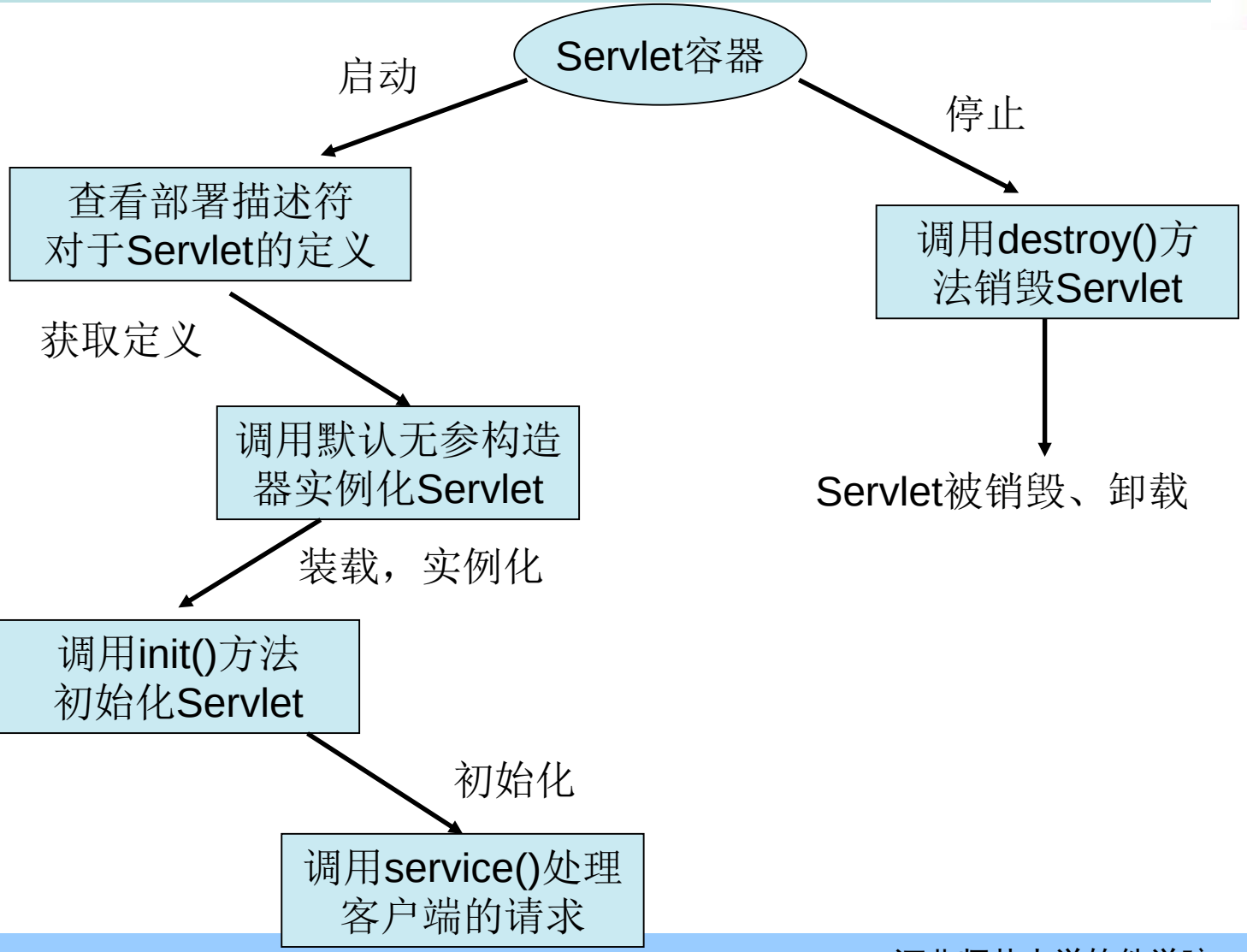
Servlet在什么时候被销毁?

- 服务器关闭
- 更新、停止Web模块的时候
- 由Servlet容器根据情况

Servlet对destroy方法的调用

- destroy()方法
  - 在Servlet实例被销毁之前调用
  - 在一个Servlet的生命周期中只执行一次

容器中Servlet的生命周期



Servlet生命周期总结

- Servlet的生命周期可以分为四个阶段
  - 加载和实例化：由Servlet容器完成
  - 初始化：实例化Servlet后会调用init方法
  - 处理客户请求：service方法(doGet或doPost)
  - 销毁：销毁之前会调用destroy方法

本讲内容

- Servlet应用开发接口
- Servlet的生命周期
- Servlet应用进阶
- web.xml与@WebServlet注解

ServletConfig接口

- ServletConfig包含Web程序部署描述符(web.xml)中定义的参数
- ServletConfig是由Servlet容器实例化
  - Servlet实例化的同时实例化对应的ServletConfig对象，并传入到init(ServletConfig config)

ServletContext接口

- 每个Web应用启动后，Servlet容器都会创建唯一的一个ServletContext对象，该对象包含整个Web程序的信息
- 在自己定义的Servlet中，可以通过getServletContext()方法获得ServletContext对象
  - getServletContext()是在GenericServlet类中定义的
- 示例：读取<context-param>内的数据

数据共享

- 数据共享的方式

1. 在客户端页面和服务器端程序(Servlet)之间，通过请求(request)的getParameter()方法共享数据
2. 在请求(request)和请求(request)之间，通过request的set/getAttribute方法（注意：只能应用在请求转发共享数据）
3. 同一个Servlet对象，通过ServletConfig对象共享数据
4. 在整个Web应用范围内，通过ServletContext的set/getAttribute方法共享数据

数据共享

- 1. 略
- 2. 略
- 3.同一个Servlet对象，通过ServletConfig对象共享数据
  - 初始化参数在web.xml文件中给出

```
<servlet>
  <servlet-name>ForwardServlet</servlet-name>
  <servlet-class>net.onest.ForwardServlet</servlet-class>
  <init-param>
      <param-name>name</param-name>
      <param-value>lww</param-value>
  </init-param>
</servlet>
```


数据共享


```
<servlet>
  <servlet-name>ForwardServlet</servlet-name>
  <servlet-class>net.onest.ForwardServlet</servlet-class>
  <init-param>
      <param-name>name</param-name>
      <param-value>lww</param-value>
  </init-param>
</servlet>
```

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    response.setContentType("text/html;charset=gb2312");
    PrintWriter out = response.getWriter();
    ServletConfig config = getServletConfig();
    String name = config.getInitParameter("name");
    out.println(name);
    request.setAttribute("name", name);
    response.sendRedirect("SecondPage.jsp");
    request.getRequestDispatcher("SecondPage.jsp").forward(request, response);
}
```

数据共享

- 1. 略
- 2. 略
- 3. 同一个Servlet对象，通过ServletConfig对象共享数据
  - 初始化参数在web.xml文件中给出
- 4. 在整个Web应用范围内，通过ServletContext的set/getAttribute方法共享数据
  - 初始化参数在web.xml文件中给出

数据共享


```
    <context-param>
      <param-name>context</param-name>
      <param-value>application</param-value>
    </context-param>
</web-app>
```

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    response.setContentType("text/html;charset=gb2312");
    PrintWriter out = response.getWriter();
    ServletConfig config = getServletConfig();
    String name = config.getInitParameter("name");
    out.println(name);
    String user = request.getParameter("sex");
    out.println("<h1>"+user+"</h1>");

    ServletContext context = getServletContext();
    String app = context.getInitParameter("context");
    out.println(app);
}
```

数据共享总结

- 1. set/getAttribute()，应用于request与request之间（请求转发）
- 2. getParameter()，应用于客户端页面与Servlet之间（参数来自于Html、JSP）
- 3. getServletConfig().getInitParameter(“param”)，应用于同一个Servlet对象
- 4. getServletContext().getInitParameter(“param”)，应用于整个Web应用范围

Servlet非线程安全

- Servlet是多线程应用
  - Servlet容器通常只创建一个Servlet实例，不同的请求都分配一个线程来进行处理，但它们操作的都是同一个Servlet实例，所以可使用Servlet的成员变量共享数据，所以也产生了线程安全的问题
- 示例：

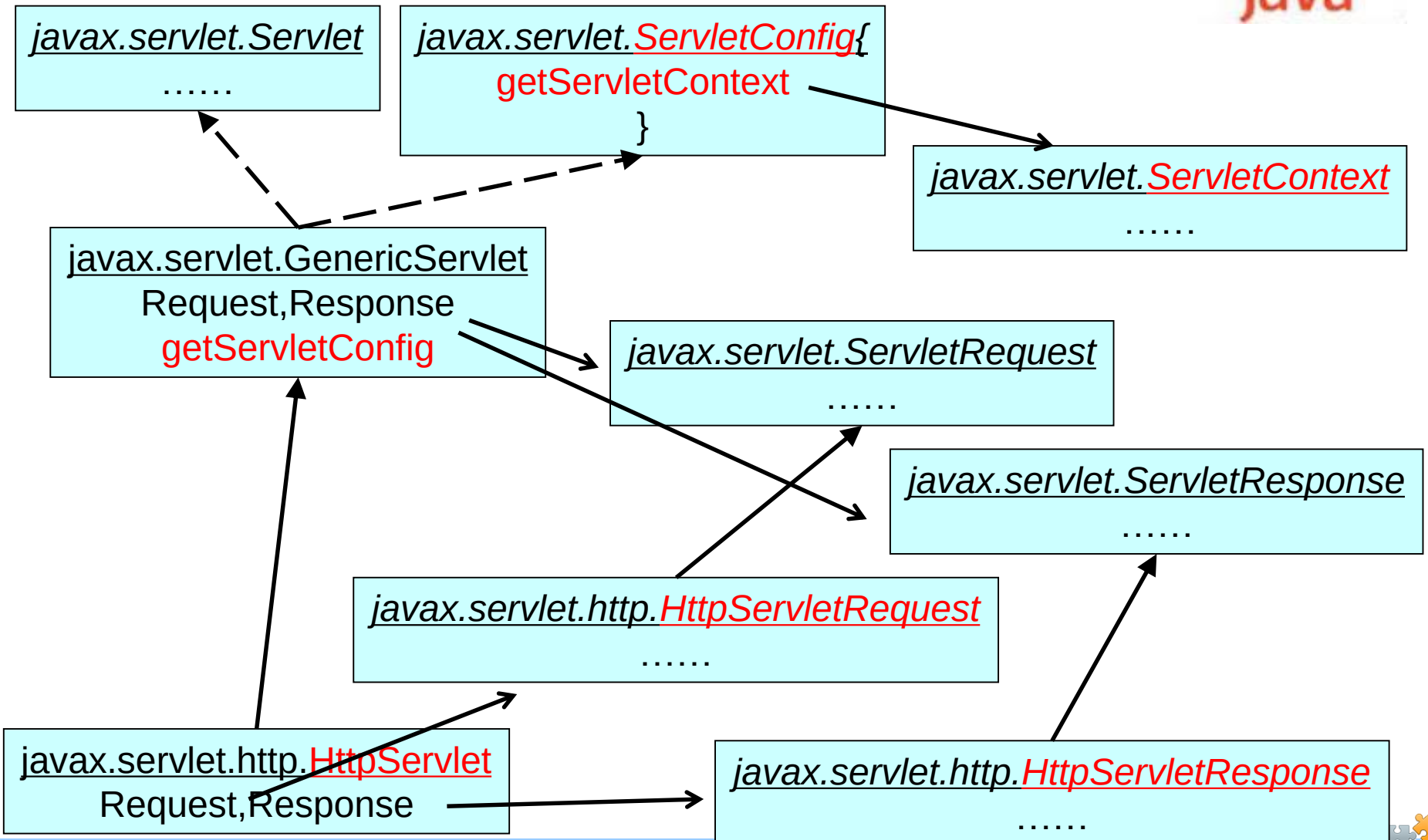
  项目ServletThreadDemo

Servlet进阶

- ContentType属性，输出非文本数据
  - 如：
    response.setContentType("image/jpg");
  - 示例：项目001的ImageServlet

- 编码问题
  - request.set/getCharacterEncoding
  - response.set/getCharacterEncoding
    - 示例： 001的CharacterServlet

目前接触到的类型的总结



本讲内容

- Servlet应用开发接口
- Servlet的生命周期
- Servlet应用进阶
- web.xml与@WebServlet注解

web.xml与@WebServlet注解

- 注解可实现web.xml配置文件的功能

```
@WebServlet(name="ForwardServlet", urlPatterns="/first"
        ,loadOnStartup=1
        ,initParams = { @WebInitParam(name = "name",value="LiLei")
        })
public class ForwardServlet extends HttpServlet {
```

```
<servlet>
  <servlet-name>ForwardServlet</servlet-name>
  <servlet-class>net.onest.ForwardServlet</servlet-class>
  <init-param>
      <param-name>name</param-name>
      <param-value>LiLei</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>ForwardServlet</servlet-name>
  <url-pattern>/first</url-pattern>
</servlet-mapping>
```

@WebServlet注解

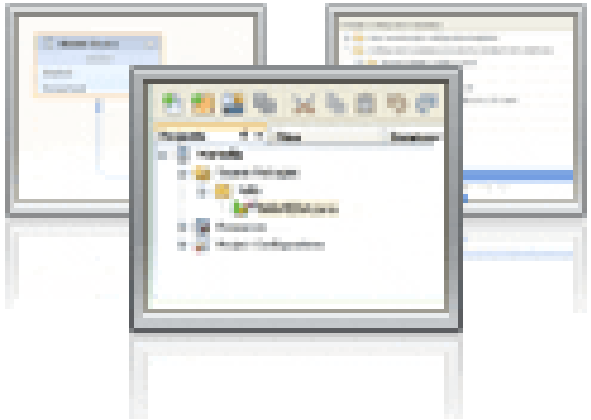
属性名	类型	描述
name	String	指定 Servlet 的 name 属性，等价于 <servlet-name>。如果没有显式指定，则该 Servlet 的取值即为类的全限定名。
value	String[]	该属性等价于 urlPatterns 属性。两个属性不能同时使用。
urlPatterns	String[]	指定一组 Servlet 的 URL 匹配模式。等价于 <url-pattern> 标签。
loadOnStartup	int	指定 Servlet 的加载顺序，等价于 <load-on-startup> 标签。
initParams	WebInitParam[]	指定一组 Servlet 初始化参数，等价于 <init-param> 标签。
asyncSupported	boolean	声明 Servlet 是否支持异步操作模式，等价于 <async-supported> 标签。
description	String	该 Servlet 的描述信息，等价于 <description> 标签。
displayName	String	该 Servlet 的显示名，通常配合工具使用，等价于 <display-name> 标签。

小结

- Servlet的生命周期
- Servlet的一些高级应用

实验

- 与数据库中的数据进行交互
  - 如：将em数据库中“员工信息”表中的所有的员工名字展示在页面上，点击某个名称时显示该员工的详细信息
- 通过例子验证Servlet的生命周期
  - 参照：Servlet模型(二)_实验手册.doc》实验一
- 将服务器的D盘根目录下的soft.jpg展示到客户端的浏览器
  - 参照：Servlet模型(二)_实验手册.doc》实验四



本讲结束

- 谢谢大家