# 动态WEB的资源。

## 什么是Servlet

Servlet是GenericServlet的父接口，GenericServlet是HttpServlet的父接口，一般我们通用的是实现Servlet孙子类HttpServlet

* Servlet
  + GenericServlet
    - HttpServlet

### 实现Servlet接口，重写5个方法。

**public** **void** service(ServletRequest request, ServletResponse response) ；

**public** **void** init(ServletConfig config)；

**public** ServletConfig getServletConfig() ；

**public** String getServletInfo()；

**public** **void** destroy()；

Servlet是一个小的java程序，运行在服务器中，接收和响应从客户端（浏览器）发送过来的请求。

# 快速入门

## 编写一个类，实现Servlet接口，重写5个方法。

## 编写一个类，继承GenericServlet类，重写一个方法。

## 配置文件，配置Servlet信息。（必须会）

<!-- 先配置Servlet信息 -->

<servlet>

<!-- 配置Servlet名称，名称必须唯一 -->

<servlet-name>HelloServlet</servlet-name>

<!-- 配置Servlet的完全路径（包名+类名） -->

<servlet-class>com.baiHo.servlet.HelloServlet </servlet-class>

</servlet>

<!-- 配置Servlet映射（访问路径） -->

<servlet-mapping>

<!-- 配置Servlet名称，和上面的名称必须相同 -->

<servlet-name>HelloServlet</servlet-name>

<!-- 配置虚拟路径（访问路径） -->

<url-pattern>/HelloServlet.servlet</url-pattern>

</servlet-mapping>

# 编译

## javac -d . HelloServlet.java

-d 表示带目录生成class文件

**. 表示 当前路径**

## HelloServlet.java:4: 软件包 javax.servlet 不存在

### 设置临时的环境变量

set classpath=%classpath%;servlet-api.jar 设置临时的环境变量，只对当前的窗口有效

# Servlet的生命周期（面试题）

## 生命周期：实例被创建，对外提供服务，销毁。

### Servlet被创建后，服务器启动后第一次访问时调用init方法进行初始化

#### public void init(ServletConfig config)；

### 从客户端发送所有的请求是service方法进行处理的。

#### public void service(ServletRequest request, ServletResponse response) ；

### 从服务器中移除服务，调用destroy方法。

#### public void destroy()；

## Servlet的生命周期：第一次请求的时候，Servlet实例被创建，立即调用init方法进行初始化。

### 实例通过service方法提供服务。服务器关闭或者移除服务时，调用destroy方法进行销毁。

## Servlet接口定义了Servlet生命周期

### public void init(ServletConfig config)；方法:服务器调用该方法初始化Servlet

### public void service(ServletRequest request, ServletResponse response) service()方法:初始化完毕，服务器调用该方法响应客户的请求

### public void destroy()；方法:服务器调用该方法消灭servlet对象

### 其中，init()方法只在Servlet第一次被请求加载的时候被调用一次，当有客户再请求Servlet服务时，Web服务器将启动一个新的线程，在该线程中，调用service方法响应客户的请求

# Servlet的关系

* Servlet接口
  + GenericServlet（重写5个方法）
    - HttpServlet（继承GenericServlet实现了Servlet接口）
      * MyServlet（我们自己写的继承HttpServlet）

# 为什么会有GenericServlet？为什么有HttpServlet？

## 注意事项：

### 如果想重写init方法，就得重写无参数的init方法。

# 开发步骤（最终）

## 编写一个类，继承HttpServlet

### 重写doGet和doPost方法

### 在doPost方法中调用doGet方法。

### 表单是get方式，调用doGet

### 表单是post方法，调用doPost方法

# doGet和doPost互相调用。

<form action="http://localhost:80/day09/demo5" method="post">

姓名：<input type="text" name="username" />

<input type="submit" value="查询" />

</form>

# 在web.xml配置

## 修改Servlet模板

## 先找到MyEclipse的安装路径。

## \myeclipse10.7\Common\plugins\com.genuitec.eclipse.wizards.xxxx.jar

### 我自己的：com.genuitec.eclipse.wizards\_9.0.0.me201211011550.jar

## 千万别解压，右键--选择压缩工具打开--templates--Servlet.java

## 拖回去之前，先MyEclipse关闭。

# Servlet自动加载

## Servlet默认是第一次访问时候创建实例。通过配置，服务器启动，创建实例。

## init做初始化的操作，非常消耗时间的。

## 在<servlet>标签下

<servlet>

<servlet-name>ServletDome03</servlet-name>

<servlet-class>com.baiHo.servlet2.ServletDome03</servlet-class>

<!-- 值必须是正整数,如果值越小，优先级越高。Servlet自动加载优先级 -->

<load-on-startup>3</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>ServletDome03</servlet-name>

<url-pattern>/ServletDome03.servlet</url-pattern>

</servlet-mapping>

# 配置虚拟路径（访问路径）

<servlet-mapping>

<servlet-name>ServletDome03</servlet-name>

<url-pattern>/ServletDome03.servlet</url-pattern>

</servlet-mapping>

## 完全路径匹配

### 以/开头的 / ServletDome03.servlet

#### 访问：http://localhost:8080/webServlet01/ServletDome03.servlet

## 目录匹配

### 以/开头的 /\*

### 访问：http://localhost:8080/webServlet01/ServletDome03可以访问

## 扩展名匹配

### 不能以/开头的 \*.do \*.action

#### 访问：http://localhost:8080/webServlet01/ServletDome03.do

## 优先级：完全路径匹配 > 目录匹配 > 扩展名匹配（\*\*\*\*\*）

# ServletConfig对象和配置文件相关

## 配置初始化参数

## 需要在<servlet></servlet>标签下配置。

## 如果要是配置在某个servlet的标签下，那么只能在该servlet中获取初始化参数。

<!-- 演示初始化参数 -->

<servlet>

<servlet-name>SerlectConfigDemo1</servlet-name>

<servlet-class>com.baiHoo.servletConfig.SerlectConfigDemo1</servlet-class>

<!-- 配置初始化参数 -->

<init-param>

<param-name>username</param-name>

<param-value>root</param-value>

</init-param>

<init-param>

<param-name>password</param-name>

<param-value>123</param-value>

</init-param>

<!-- //配置初始化参数 -->

</servlet>

<servlet-mapping>

<servlet-name>SerlectConfigDemo1</servlet-name>

<url-pattern>/SerlectConfigDemo1.servlet</url-pattern>

</servlet-mapping>

<!-- //演示初始化参数 -->

### String getServletName() 获取配置文件中servlet的名称

### String getInitParameter(String name) 获取初始化参数

### Enumeration getInitParameterNames() 获取初始化参数的名称们

# ServletContext对象（域对象） （\*\*\*\*\*）

## 定义：WEB容器在启动时，它会为每个WEB应用程序都创建一个对应的ServletContext对象，它代表当前web应用。

### 一个WEB应用对应一个ServletContext对象

### 一个WEB应用下有多个Servlet程序

### 所有的servlet程序都共享同一个ServletContext对象

### demo1存入内容 ServletContext demo2中取出来

## 作用：

### 获取WEB应用全局初始化参数

### 在web.xml中配置

<!-- 配置全局初始化参数 -->

<context-param>

<param-name>encoding</param-name>

<param-value>GBK</param-value>

</context-param>

<!--// 配置全局初始化参数 -->

#### String getInitParameter(String name)

#### getInitParameterNames()

# 实现数据的共享（\*\*\*\*\*）

## void setAttribute(String name, Object object) 存入数据

## void removeAttribute(String name) 删除数据

## Object getAttribute(String name) 获取数据

# 读取资源文件（\*\*\*\*\*）

## InputStream getResourceAsStream(String path) 通过文件的地址获取输入流

## String getRealPath(String path) 通过文件的地址获取文件的绝对磁盘路径

# HTTP的协议

## 请求

### referer 记住当前网页的来源

### user-agent 浏览器版本信息

### if-modefied-since

## 响应

### 响应头

#### location 和302一起来完成重定向的操作(发送两次求情)

重定向，其实是两次request, 第一次，客户端request   A,服务器响应，并response回来，告诉浏览器，你应该去B。这个时候IE可以看到地址变了，而且历史的回退按钮也亮了。重定向可以访问自己web应用以外的资源。**在重定向的过程中，传输的信息会被丢失。**

#### refresh 页面的定时刷新

#### last-modefied 和 if-modefied-since和304状态码一起来控制缓存。