**public void init(ServletConfig servletConfig)**

**ServletConfig：Servlet配置，ServletConfig是一个接口，具体的实现是由Servlet容器开发商实现的，这里我们是以tomcat为例，就是tomcat实现的，具体这个参数的部分参数值，源头在web.xml配置文件中**

**对Servlet配置时配置的：**

**<!-- 配置servlet的初始化值 -->**

**<init-param>**

**<param-name>user</param-name>**

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

**</init-param>**

**<init-param>**

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

**<param-value>123456</param-value>**

**</init-param>**

**ServletConfig接口的定义的方法：**

**String getServletName()  -- 获取当前Servlet在web.xml中配置的名字**

**String getInitParameter(String name) -- 获取当前Servlet指定名称的初始化参数的值**

**Enumeration getInitParameterNames()  -- 获取当前Servlet所有初始化参数的名字组成的枚举**

**ServletContext getServletContext()  -- 获取代表当前web应用的ServletContext对象（一会儿讲）**

**//--获取当前Servlet 在web.xml中配置的名称（用的不多，来演示！）**

**String sName = config.getServletName();**

**System.out.println("当前Servlet 在web.xml中配置的名称:"+sName);**

**//--获取当前Servlet中配置的初始化参数（只能获取一个）经常用到，演示！**

**String value = config.getInitParameter("user");**

**System.out.println(value);**

**// --获取当前Servlet中配置的初始化参数（全部获取）经常用到，演示**

**Enumeration enumration = config.getInitParameterNames();**

**while(enumration.hasMoreElements()){**

**String name = (String) enumration.nextElement();**

**String value = config.getInitParameter(name);**

**System.out.println(name+":"+value);**

**}**

**ServletContext：代表当前web应用（非常重要）**

**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。**

**ServletConfig对象中维护了ServletContext对象的引用，开发人员在编写servlet时，可以通过ServletConfig.getServletContext方法获得ServletContext对象。**

**由于一个WEB应用中的所有Servlet共享同一个ServletContext对象，因此Servlet对象之间可以通过ServletContext对象来实现通讯。ServletContext对象通常也被称之为context域对象。**

**ServletContext的应用：**

**1.做为域对象可以在整个web应用范围内共享数据。**

**这里涉及到一些概念：**

**——域对象：在一个可以被看见的范围内共享数据用到对象**

**——作用范围：整个web应用范围内共享数据**

**——生命周期：当服务器启动web应用加载后创建出ServletContext对象后，域产生。当web应用被移除出容器或服务器关闭，随着web应用的销毁域销毁。**

**动手：写两个servlet，一个放context.setAttribute("name", "smyhvae");另一个取String myName = (String) context.getAttribute("name");**

**context中常用的方法有：**

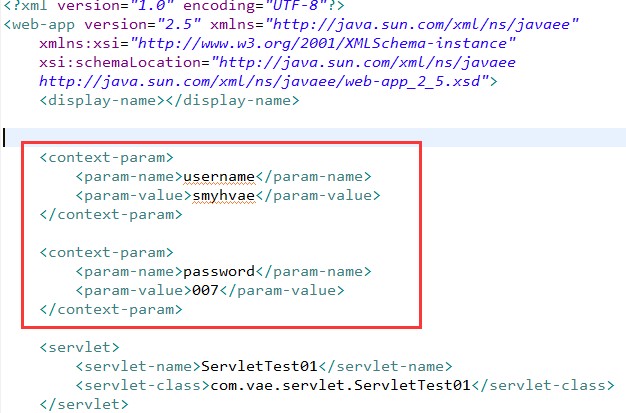
**void setAttribute(String,Object);**

**Object getAttribute(String);**

**void removeAttribute(String);**

**2、获取WEB应用的初始化参数**

**我们在上面，通过<init-param>标签为某一个单独的servlet加配置信息，这种配置信息在其他的Servlet中是无法访问到的。可如果我们使用<context-param>标签（与Servlet标签并列）为整个Web应用配置属性的话，那所有的Servlet就都能访问里面的参数了。例如：**



**然后接下来我们在代码中来获取这些参数。代码如下：**

**// 获取单个的Context里面的初始化参数**

**String value1 = context.getInitParameter("username");**

**String value2 = context.getInitParameter("password");**

**System.out.println(value1 + ";" + value2);**

**System.out.println();**

**// 一次性获取Context里所有的初始化参数**

**Enumeration enumeration = context.getInitParameterNames();**

**while (enumeration.hasMoreElements()) {**

**String name = (String) enumeration.nextElement();**

**String value = context.getInitParameter(name);**

**System.out.println(name + ";" + value);**

**}**

**来操作一下！**

**3、加载资源文件：**

**ServletContext提供了getRealPath方法,在这个方法中传入一个路径,这个方法的底层会在传入的路径的前面拼接当前web应用的硬盘路径，从而得到当前资源的硬盘路径。例如：**

**先在WebRoot的根目录下新建一个专门用了配置信息的文件为config.properties，里面的参数如下：**

**username=smyhvae**

**password=007**

**Properties prop = new Properties(); // 注意导的包是import java.util.Properties;**

**prop.load(new FileReader(context.getRealPath("config.properties")));**

**System.out.println(prop.getProperty("username"));**

**System.out.println(prop.getProperty("password"));**

**演示：**

**4、获取当前WEB应用的名称：**

**getContextPath()**