ServletConfig和ServletContext

1. Servlet初始化参数（使用ServletConfig）
   1. 用途：把程序中用到的参数转移到DD中来
   2. 用法：
      1. 在DD中（web.xml）

<servlet>

<servlet-name>HelloVenus</servlet-name>

<servlet-class>com.ixenos.hahaha</servlet-class>

<init-param>

<param-name>adminEmail</param-name>

<param-value>ixenos@foxmail.com</param-value>

</init-param>

</servlet>

* + 1. 在servlet中

Out.println(getServletConfig().getInitParameter(“adminEmail”));

**每个servlet都继承了一个getServletConfig方法**

而getServletConfig方法返回一个ServletConfig

getInitParameter是ServletConfig的一个方法

1. 在servlet初始化（init）之前不能使用servlet初始化参数（InitParameter），因为ServletConfig对象是在init的时候才创建的。

容器从DD中读出servlet初始化参数，然后把这些参数交给ServletConfig，再把ServletConfig传递给servlet的init方法。

1. 若需覆盖init，选哪个init好？

Servlet超类中有两个init方法，一个有servletConfig参数，一个没有参数，而继承的init(ServletConfig)方法会调用无参数的init方法，所以只需要覆盖无参数的init方法即可

1. Servlet初始化参数只能读取一次，就是在容器初始化servlet的时候

因此，我们还是不能在servlet的生命周期中动态修改email地址，嗨呀……

不过吧，大多web容器允许热部署，只是这样又显得兴师动众，

当然我们还可以让servlet运行时读取文件配置，但这样又在每次运行时有开销，而不是唯一的一次初始化时

1. 上下文初始化参数（使用ServletContext）

Servlet初始化参数只能给servlet调用，所以如果想让参数被Web应用的所有部分访问，那么就必须在DD中为每一个Servlet配置初始化参数，然后在每一个servlet中调用servletconfig参数（把参数传到jsp中只需setattribute然后dispatcher）

用途：上下文参数对整个Web应用可用，这样**应用中的所有servlet和JSP都能自动地访问上下文初始化参数**

用法：

在DD中：

<servlet>

……

</servlet>

<context-param>

<param-name>birth</param-name>

<param-value>0309</param-value>

</context-param>

在Servlet中：

Out.println(getServletContext().getInitParameter(“birth”));

或分步写：

SerletContext context = getServletContext();

Out.println(context.getInitParameter(“birth”));

与servlet初始化参数的区别：在不在servlet标签包裹内

每个servlet一个ServletConfig

每个web应用一个ServletContext

1. 如果Web应用是分布式的，那么每个JVM中也只有一个ServletContext，虽然Web应用有多个ServletContext
2. ServletContext常用方法：

得到初始化参数和得到/设置属性

* 1. getInitParameter(String)
  2. getInitParameterNames()
  3. getAttribute(String)
  4. getAttributeNames()
  5. setAttribute(String, Object) //之前的例子有用到！！！
  6. removeAttribute(String)

得到有关服务器、容器的信息

1. getMajorVersion()
2. getServerInfo()

得到路径、流、请求分派器

1. getRealPath(String)
2. getResourceAsStream(String)
3. getRequestDispatcher(String)

etc.

1. 两种方法得到ServletContext
   1. getServletConfig().getServletContext()

当使用非Http的Servlet时使用这个（ServeltConfig含有SerletContext的引用）

* 1. getServletContext()

一般我们使用HttpServlet就使用这个（已实现getServletContext方法）

1. JSP访问ServletContext，则使用隐藏对象application

JSP访问ServletConfig，使用隐藏对象Config

1. DD中对ServletConfig和ServletContext的部署

<!-- Servlet -->

<servlet>

<servlet-name>SeclectBeer</servlet-name>

<servlet-class>com.example.controller.BeerSelect</servlet-class>

</servlet>

<servlet>

<servlet-name>DownloadIt</servlet-name>

<servlet-class>test.DownloadTest</servlet-class>

<!-- servlet初始化参数，放在对应Servlet中 -->

<init-param>

<param-name>downloadAdd</param-name>

<param-value>/downloadSource/servlet-api.jar</param-value>

</init-param>

</servlet>

<!-- 上下文初始化参数 -->

<context-param>

<param-name>downloadAdd2</param-name>

<param-value>/downloadSource/servlet-api.jar</param-value>

</context-param>

1. ServletConfig和ServletContext
   1. ServletConfig用于向servlet传递部署时信息（将servlet初始化参数放在配置文件，由ServletConfig解析，并在init初始化时传递给servlet，让其成为一个真正的servlet）
   2. ServletConfig还用于访问ServletContext
   3. 每个Servlet只有一个ServletConfig对象
   4. 每个Web应用只有一个ServletContext对象

12.ServletConfig对象和ServletContext对象

1）每一个servlet有一个ServletConfig对象（参数在DD中配置），而每个Web应用有一个ServletContext对象

2）ServletConfig用于向servlet传递部署时信息（使信息不用硬编码到servlet中），而且用于访问ServletContext；而ServletContext用于访问Web应用参数（在DD中配置），用于得到服务器的信息

3）二者在servlet调用init时创建

13.