# web.xml中context-param的配置作用

一、定义

<context-param>

<param-name>参数名</param-name>

<param-value>参数值</param-value>

</context-param>

作用：该元素用来声明应用范围(整个WEB项目)内的上下文初始化参数。

param-name 设定上下文的参数名称。必须是唯一名称

param-value 设定的参数名称的值

1. 初始化过程
2. 启动一个WEB项目的时候,容器(如:Tomcat)会去读它的配置文件web.xml。读两个节点: <listener></listener> 和 <context-param></context-param>。
3. 接着容器会创建一个ServletContext(上下文)，整个WEB项目共享这个上下文。
4. 接着容器会将读取到<context-param>转化为键值对,并交给ServletContext。
5. 容器创建<listener></listener>中的类实例,即创建监听（备注：listener定义的类可以是自定义的类但必须需要实现ServletContextListener）。
6. 在监听的类中会有一个contextInitialized(ServletContextEvent event)初始化方法，在这个方法中可以通过event.getServletContext().getInitParameter("param-name设置的值") 来得到context-param 设定的值。在这个类中还必须有一个contextDestroyed(ServletContextEvent event) 销毁方法.用于关闭应用前释放资源，比如说数据库连接的关闭。
7. 得到这个context-param的值之后,你就可以做一些操作了。注意,这个时候你的WEB项目还没有完全启动完成。这个动作会比所有的Servlet都要早。换句话说,这个时候,你对<context-param>中的键值做的操作,将在你的WEB项目完全启动之前被执行。

此处说明一下web.xml 的加载顺序：

****context-param -> listener -> filter -> servlet****，而同个类型之间的实际程序调用的时候的顺序是根据对应的 mapping 的顺序进行调用的。

具体内容可参照此篇博客：

http://www.cnblogs.com/shenliang123/p/3344555.html

1. 举例

如:

<!-- 加载spring的配置文件 -->

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:/applicationContext.xml</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

自定义监听器

<context-param>

<param-name>test1</param-name>

<param-value>value1</param-value>

</context-param>

<context-param>

<param-name>test2</param-name>

<param-value>value2</param-value>

</context-param>

<context-param>

<param-name>test3</param-name>

<param-value>value3</param-value>

</context-param>

<context-param>

<param-name>test4</param-name>

<param-value>value4</param-value>

</context-param>

<listener>

<listener-class>com.anjz.web.listener.CustomListener</listener-class>

</listener>

package com.anjz.web.listener;

import java.util.Enumeration;

import javax.servlet.ServletContext;

import javax.servlet.ServletContextEvent;

import javax.servlet.ServletContextListener;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

/\*\*

\* 自定义监听器

\* @author shuai.ding

\* @date 2017年5月16日下午4:03:34

\*/

public class CustomListener implements ServletContextListener{

private static final Logger LOGGER = LoggerFactory.getLogger(CustomListener.class);

@Override

public void contextInitialized(ServletContextEvent sce) {

LOGGER.info("执行contextInitialized");

ServletContext servletContext = sce.getServletContext();

Enumeration<String> enumeration = servletContext.getInitParameterNames();

while(enumeration.hasMoreElements()){

String key = enumeration.nextElement();

String value = servletContext.getInitParameter(key);

LOGGER.info("键：{},值：{}",key,value);

}

String contextPath = servletContext.getContextPath();

String rootPath = servletContext.getRealPath("/");

LOGGER.info("contextPath:{},rootPath:{}",contextPath,rootPath);

}

@Override

public void contextDestroyed(ServletContextEvent sce) {

LOGGER.info("执行contextDestroyed");

}

}

输出：

2017-05-16 17:21:43,343 - INFO [com.anjz.web.listener.CustomListener] - <执行contextInitialized>

2017-05-16 17:21:43,345 - INFO [com.anjz.web.listener.CustomListener] - <键：test1,值：value1>

2017-05-16 17:21:43,345 - INFO [com.anjz.web.listener.CustomListener] - <键：test4,值：value4>

2017-05-16 17:21:43,345 - INFO [com.anjz.web.listener.CustomListener] - <键：test2,值：value2>

2017-05-16 17:21:43,345 - INFO [com.anjz.web.listener.CustomListener] - <键：test3,值：value3>

2017-05-16 17:21:43,345 - INFO [com.anjz.web.listener.CustomListener] - <contextPath:/web-demo,rootPath:D:\Program\myself\demo\demo-example\web-demo\src\main\webapp>

1. context-param和init-param区别

(1)application范围内的参数，存放在servletcontext中，在web.xml中配置如下：

<context-param>

<param-name>context/param</param-name>

<param-value>avalible during application</param-value>

</context-param>

(2)servlet范围内的参数，只能在servlet的init()方法中取得，在web.xml中配置如下：

<servlet>

<servlet-name>myServlet</servlet-name>

<servlet-class>com.anjz.web.servlet.MyServlet</servlet-class>

<init-param>

<param-name>param1</param-name>

<param-value>avalible in servlet init()</param-value>

</init-param>

<load-on-startup>0</load-on-startup>

</servlet>

package com.anjz.web.servlet;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

/\*\*

\* 自定义servlet

\* @author shuai.ding

\*

\* @date 2017年5月16日下午5:29:18

\*/

public class MyServlet extends HttpServlet{

private static final Logger LOGGER = LoggerFactory.getLogger(MyServlet.class);

/\*\*

\*

\*/

private static final long serialVersionUID = 1L;

@Override

public void init() throws ServletException {

//获取servlet中初始化的参数

String value1 = this.getInitParameter("param1");

LOGGER.info("key:{},value:{}","param1",value1);

//获取ServletContext中初始化的参数

String value2 = getServletContext().getInitParameter("test1");

LOGGER.info("key:{},value:{}","test1",value2);

}

}

第一种参数在servlet里面可以通过getServletContext().getInitParameter("context/param")得到。  
第二种参数只能在servlet的init()方法中通过this.getInitParameter("param1")取得。