**一**

1、启动一个WEB项目的时候，WEB容器会去读取它的配置文件web.xml，读取<listener>和<context-param>两个结点。

2、紧急着，容创建一个ServletContext（servlet上下文），这个web项目的所有部分都将共享这个上下文。

3、容器将<context-param>转换为键值对，并交给servletContext。

4、容器创建<listener>中的类实例，创建监听器。

**二  Load-on-startup**

Load-on-startup元素在web应用启动的时候指定了servlet被加载的顺序，它的值必须是一个整数。如果它的值是一个负整数或是这个元素不存在，那么容器会在该servlet被调用的时候，加载这个servlet。如果值是正整数或零，容器在配置的时候就加载并初始化这个servlet，容器必须保证值小的先被加载。如果值相等，容器可以自动选择先加载谁。

在servlet的配置当中，<load-on-startup>5</load-on-startup>的含义是：

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。

正数的值越小，启动该servlet的优先级越高。

**三  加载顺序**

首先可以肯定的是，加载顺序与它们在 web.xml 文件中的先后顺序无关。即不会因为 filter 写在 listener的前面而会先加载 filter。最终得出的结论是：ServletContext-> listener ->filter -> servlet

      同时还存在着这样一种配置节：context-param，它用于向 ServletContext提供键值对，即应用程序上下文信息。我们的 listener, filter 等在初始化时会用到这些上下文中的信息，那么context-param 配置节是不是应该写在 listener 配置节前呢？实际上 context-param配置节可写在任意位置，因此真正的加载顺序为：context-param -> listener-> filter -> servlet

       对于某类配置节而言，与它们出现的顺序是有关的。以filter 为例，web.xml 中当然可以定义多个 filter，与 filter 相关的一个配置节是filter-mapping，这里一定要注意，对于拥有相同 filter-name 的 filter 和 filter-mapping配置节而言，filter-mapping 必须出现在 filter 之后，否则当解析到 filter-mapping 时，它所对应的filter-name 还未定义。web 容器启动时初始化每个 filter 时，是按照 filter配置节出现的顺序来初始化的，当请求资源匹配多个 filter-mapping 时，filter 拦截资源是按照filter-mapping 配置节出现的顺序来依次调用 doFilter() 方法的。

       servlet 同 filter类似，此处不再赘述。

      由此，可以看出，web.xml 的加载顺序是：ServletContext-> 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。

### 一、web.xml配置文件常用元素及其意义预览

1 <web-app>

2

3 <!--定义了WEB应用的名字-->

4 <display-name></display-name>

5

6 <!--声明WEB应用的描述信息-->

7 <description></description>

8

9 <!--context-param元素声明应用范围内的初始化参数-->

10 <context-param></context-param>

11

12 <!--过滤器元素将一个名字与一个实现javax.servlet.Filter接口的类相关联-->

13 <filter></filter>

14

15 <!--一旦命名了一个过滤器，就要利用filter-mapping元素把它与一个或多个servlet或JSP页面相关联-->

16 <filter-mapping></filter-mapping>

17

18 <!--servlet API的版本2.3增加了对事件监听程序的支持，事件监听程序在建立、修改和删除会话或servlet环境时得到通知。

19 Listener元素指出事件监听程序类-->

20 <listener></listener>

21

22 <!--在向servlet或JSP页面制定初始化参数或定制URL时，必须首先命名servlet或JSP页面。

23 Servlet元素就是用来完成此项任务的-->

24 <servlet></servlet>

25

26 <!--服务器一般为servlet提供一个缺省的URL：http://host/webAppPrefix/servlet/ServletName。

27 但是，常常会更改这个URL，以便servlet可以访问初始化参数或更容易地处理相对URL。

28 在更改缺省URL时，使用servlet-mapping元素-->

29 <servlet-mapping></servlet-mapping>

30

31 <!--如果某个会话在一定时间内未被访问，服务器可以抛弃它以节省内存。可通过使用HttpSession的

32 setMaxInactiveInterval方法明确设置单个会话对象的超时值，或者可利用session-config元素制定缺省超时值-->

33 <session-config></session-config>

34

35 <!--如果Web应用具有想到特殊的文件，希望能保证给他们分配特定的MIME类型，则mime-mapping元素提供这种保证-->

36 <mime-mapping></mime-mapping>

37

38 <!--指示服务器在收到引用一个目录名而不是文件名的URL时，使用哪个文件-->

39 <welcome-file-list></welcome-file-list>

40

41 <!--在返回特定HTTP状态代码时，或者特定类型的异常被抛出时，能够制定将要显示的页面-->

42 <error-page></error-page>

43

44 <!--对标记库描述符文件（Tag Libraryu Descriptor file）指定别名。此功能使你能够更改TLD文件的位置，

45 而不用编辑使用这些文件的JSP页面-->

46 <taglib></taglib>

47

48 <!--声明与资源相关的一个管理对象-->

49 <resource-env-ref></resource-env-ref>

50

51 <!--声明一个资源工厂使用的外部资源-->

52 <resource-ref></resource-ref>

53

54 <!--制定应该保护的URL。它与login-config元素联合使用-->

55 <security-constraint></security-constraint>

56

57 <!--指定服务器应该怎样给试图访问受保护页面的用户授权。它与sercurity-constraint元素联合使用-->

58 <login-config></login-config>

59

60 <!--给出安全角色的一个列表，这些角色将出现在servlet元素内的security-role-ref元素的role-name子元素中。

61 分别地声明角色可使高级IDE处理安全信息更为容易-->

62 <security-role></security-role>

63

64 <!--声明Web应用的环境项-->

65 <env-entry></env-entry>

66

67 <!--声明一个EJB的主目录的引用-->

68 <ejb-ref></ejb-ref>

69

70 <!--声明一个EJB的本地主目录的应用-->

71 <ejb-local-ref></ejb-local-ref>

72

73 </web-app>

### 二、各个配置元素详解

1.Web应用图标：指出IDE和GUI工具用来表示Web应用的大图标和小图标

1 <icon>

2 <small-icon>/images/app\_small.gif</small-icon>

3 <large-icon>/images/app\_large.gif</large-icon>

4 </icon>

2.Web 应用名称：提供GUI工具可能会用来标记这个特定的Web应用的一个名称

<display-name>Tomcat Example</display-name>

3.Web 应用描述：给出于此相关的说明性文本

<desciption>Tomcat Example servlets and JSP pages.</desciption>

4.上下文参数：声明应用范围内的初始化参数

1 <context-param>

2 <param-name>参数名</para-name>

3 <param-value>参数值</param-value>

4 <description>参数描述</description>

5 </context-param>

在servlet里面可以通过 getServletContext().getInitParameter(“context/param”)得到

5.过滤器配置：将一个名字与一个实现javaxs.servlet.Filter接口的类相关联

1 <filter>

2 <filter-name>setCharacterEncoding</filter-name>

3 <filter-class>com.myTest.setCharacterEncodingFilter</filter-class>

4 <init-param>

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

6 <param-value>GB2312</param-value>

7 </init-param>

8 </filter>

9 <filter-mapping>

10 <filter-name>setCharacterEncoding</filter-name>

11 <url-pattern>/\*</url-pattern>

12 </filter-mapping>

6.监听器配置

1 <listener>

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

3 </listener>

7.Servlet配置

1 <servlet>

2 <servlet-name>servlet名称</servlet-name>

3 <servlet-class>servlet类全路径</servlet-class>

4 <init-param>

5 <param-name>参数名</param-name>

6 <param-value>参数值</param-value>

7 </init-param>

8 <run-as>

9 <description>Security role for anonymous access</description>

10 <role-name>tomcat</role-name>

11 </run-as>

12 　 <load-on-startup>指定当Web应用启动时，装载Servlet的次序</load-on-startup>

13 </servlet>

14 <servlet-mapping>

15 <servlet-name>servlet名称</servlet-name>

16 <url-pattern>映射路径</url-pattern>

17 </servlet-mapping>

8.会话超时配置（单位为分钟）

1 <session-config>

2 <session-timeout>120</session-timeout>

3 </session-config>

9.MIME类型配置

1 <mime-mapping>

2 <extension>htm</extension>

3 <mime-type>text/html</mime-type>

4 </mime-mapping>

10.指定欢迎文件页配置

1 <welcome-file-list>

2 <welcome-file>index.jsp</welcome-file>

3 <welcome-file>index.html</welcome-file>

4 <welcome-file>index.htm</welcome-file>

5 </welcome-file-list>

11.配置错误页面

　　(1).通过错误码来配置error-page

1 <!--配置了当系统发生404错误时，跳转到错误处理页面NotFound.jsp-->

2 <error-page>

3 <error-code>404</error-code>

4 <location>/NotFound.jsp</location>

5 </error-page>

　　(2).通过异常的类型配置error-page

1 <!--配置了当系统发生java.lang.NullException（即空指针异常）时，跳转到错误处理页面error.jsp-->

2 <error-page>

3 <exception-type>java.lang.NullException</exception-type>

4 <location>/error.jsp</location>

5 </error-page>

12.TLD配置

1 <taglib>

2 <taglib-uri>http://jakarta.apache.org/tomcat/debug-taglib</taglib-uri>

3 <taglib-location>/WEB-INF/jsp/debug-taglib.tld</taglib-location>

4 </taglib>

如果开发工具一直在报错,应该把<taglib> 放到 <jsp-config>中

1 <jsp-config>

2 <taglib>

3 <taglib-uri>http://jakarta.apache.org/tomcat/debug-taglib</taglib-uri>

4 <taglib-location>/WEB-INF/pager-taglib.tld</taglib-location>

5 </taglib>

6 </jsp-config>

13.资源管理对象配置

1 <resource-env-ref>

2 <resource-env-ref-name>jms/StockQueue</resource-env-ref-name>

3 </resource-env-ref>

14.资源工厂配置

1 <resource-ref>

2 <res-ref-name>mail/Session</res-ref-name>

3 <res-type>javax.mail.Session</res-type>

4 <res-auth>Container</res-auth>

5 </resource-ref>

　　配置数据库连接池就可在此配置

1 <resource-ref>

2 <description>JNDI JDBC DataSource of shop</description>

3 <res-ref-name>jdbc/sample\_db</res-ref-name>

4 <res-type>javax.sql.DataSource</res-type>

5 <res-auth>Container</res-auth>

6 </resource-ref>

15.安全限制配置

1 <security-constraint>

2 <display-name>Example Security Constraint</display-name>

3 <web-resource-collection>

4 <web-resource-name>Protected Area</web-resource-name>

5 <url-pattern>/jsp/security/protected/\*</url-pattern>

6 <http-method>DELETE</http-method>

7 <http-method>GET</http-method>

8 <http-method>POST</http-method>

9 <http-method>PUT</http-method>

10 </web-resource-collection>

11 <auth-constraint>

12 <role-name>tomcat</role-name>

13 <role-name>role1</role-name>

14 </auth-constraint>

15 </security-constraint>

16.登陆验证配置

1 <login-config>

2 <auth-method>FORM</auth-method>

3 <realm-name>Example-Based Authentiation Area</realm-name>

4 <form-login-config>

5 <form-login-page>/jsp/security/protected/login.jsp</form-login-page>

6 <form-error-page>/jsp/security/protected/error.jsp</form-error-page>

7 </form-login-config>

8 </login-config>

17.安全角色：security-role元素给出安全角色的一个列表，这些角色将出现在servlet元素内的security-role-ref元素的role-name子元素中。  
分别地声明角色可使高级IDE处理安全信息更为容易。

1 <security-role>

2 <role-name>tomcat</role-name>

3 </security-role>

18.Web环境参数：env-entry元素声明Web应用的环境项

1 <env-entry>

2 <env-entry-name>minExemptions</env-entry-name>

3 <env-entry-value>1</env-entry-value>

4 <env-entry-type>java.lang.Integer</env-entry-type>

5 </env-entry>

19.EJB 声明

1 <ejb-ref>

2 <description>Example EJB reference</decription>

3 <ejb-ref-name>ejb/Account</ejb-ref-name>

4 <ejb-ref-type>Entity</ejb-ref-type>

5 <home>com.mycompany.mypackage.AccountHome</home>

6 <remote>com.mycompany.mypackage.Account</remote>

7 </ejb-ref>

20.本地EJB声明

1 <ejb-local-ref>

2 <description>Example Loacal EJB reference</decription>

3 <ejb-ref-name>ejb/ProcessOrder</ejb-ref-name>

4 <ejb-ref-type>Session</ejb-ref-type>

5 <local-home>com.mycompany.mypackage.ProcessOrderHome</local-home>

6 <local>com.mycompany.mypackage.ProcessOrder</local>

7 </ejb-local-ref>