# Druid数据源配置入门

<!-- 阿里 druid数据库连接池 -->

<beanid="dataSource"class="com.alibaba.druid.pool.DruidDataSource"init-method="init"destroy-method="close">

<!-- 基本属性 url、user、password -->

<property name="url" value="${db.mysql.url}" />

<property name="username" value="${db.mysql.username}" />

<property name="password" value="${db.mysql.password}" />

<property name="driverClassName" value="${driverClassName}" />

<!-- 配置初始化大小、最小、最大 -->

<property name="initialSize" value="5" />

<property name="minIdle" value="10" />

<property name="maxActive" value="20" />

<!-- 配置获取连接等待超时的时间 -->

<property name="maxWait" value="60000" />

<!-- 配置间隔多久才进行一次检测，检测需要关闭的空闲连接，单位是毫秒 -->

<property name="timeBetweenEvictionRunsMillis" value="60000" />

<!-- 配置一个连接在池中最小生存的时间，单位是毫秒 -->

<property name="minEvictableIdleTimeMillis" value="300000" />

<property name="validationQuery" value="SELECT 'x'" />

<property name="testWhileIdle" value="true" />

<property name="testOnBorrow" value="false" />

<property name="testOnReturn" value="false" />

<!-- 打开PSCache，并且指定每个连接上PSCache的大小 -->

<property name="poolPreparedStatements" value="true" />

<property name="maxPoolPreparedStatementPerConnectionSize" value="20" />

<!-- 连接泄漏处理。Druid提供了RemoveAbandanded相关配置，用来关闭长时间不使用的连接（例如忘记关闭连接）。 -->

<property name="removeAbandoned" value="true" />

<!-- 1800秒，也就是30分钟 -->

<property name="removeAbandonedTimeout" value="1800" />

<!-- 关闭abanded连接时输出错误日志 -->

<property name="logAbandoned" value="true" />

<!-- 配置监控统计拦截的filters, 监控统计："stat",防SQL注入："wall",组合使用： "stat,wall" -->

<property name="filters" value="stat" /> </bean>

# spring监听器ContextLoaderListener

## 2.1 作用

在启动Web容器时，自动装配Spring applicationContext.xml的配置信息

## 2.2 如何部署

如果在web.xml中不写任何参数配置信息，默认的路径是"/WEB-INF/applicationContext.xml，在WEB-INF目录下创建的xml文件的名称必须是applicationContext.xml。如果是要自定义文件名可以在web.xml里加入contextConfigLocation这个context参数：

<context-param>  
<param-name>contextConfigLocation</param-name>  
<param-value>  
/WEB-INF/classes/applicationContext-\*.xml  
</param-value>  
</context-param>

### 2.2.1 例子

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

如果applicationContext.xml文件没有在/WEB-INF/下，或文件名不一致，或存在多个Spring配置文件，在web.xml文件中根据下面代码修改

  <context-param>  
   <param-name>contextConfigLocation</param-name>  
  <param-value>classpath\*:applicationContext-\*.xml,/WEB-INF/applicationContext.xml,/WEB-INF/classes/applicationContext-\*.xml  
</param-value>  
  </context-param>

# [PropertyPlaceholderConfigurer读取配置文件](https://www.cnblogs.com/dream-to-pku/p/6367396.html)

## 原理

PropertyPlaceholderConfigurer是个bean工厂后置处理器的实现，也就是 BeanFactoryPostProcessor接口的一个实现。PropertyPlaceholderConfigurer可以将上下文（配置文 件）中的属性值放在另一个单独的标准java Properties文件中去。在XML文件中用${key}替换指定的properties文件中的值。这样的话，只需要对properties文件进 行修改，而不用对xml配置文件进行修改。

## 配置

<bean class="com.slp.util.SpringContextHolder"/>

<bean id="propertyConfigurer"

class="com.slp.util.CustomizedPropertyConfigurer">

<property name="locations">

<list>

<value>classpath:conf/jdbc.properties</value>

<value>classpath:conf/config.properties</value>

<value>classpath:conf/server.properties</value>

</list>

</property>

<property name="fileEncoding" value="UTF-8"/>

</bean>