# 尚马教育 JAVA WEB课程

# 数据库连接池 & servlet3.0

文档编号：B19

创建日期： 2017-06-13

最后修改日期：2019-09-23

版 本 号：V2.0

电子版文件名：尚马教育-第二阶段-9.数据库连接池 & servlet3.0.docx

**文档修改记录：**

|  |  |  |  |
| --- | --- | --- | --- |
| 更新日期 | 更新作者 | 更新说明 | 版本号 |
| 2018-08-01 | 王绍成 | 初始版本 | V1.0 |
| 2019-09-13 | 郭雪岩 | 内容更新 | V2.0 |

**主讲人：**

**郭雪岩**

目录

[1. 数据库连接池](#_Toc17722_WPSOffice_Level1) [3](#_Toc17722_WPSOffice_Level1)

[1.1. 数据库连接池](#_Toc19957_WPSOffice_Level2) [3](#_Toc19957_WPSOffice_Level2)

[1.2. Druid](#_Toc30875_WPSOffice_Level2) [3](#_Toc30875_WPSOffice_Level2)

[1.2.1. Druid下载](#_Toc19957_WPSOffice_Level3) [4](#_Toc19957_WPSOffice_Level3)

[1.2.2. Druid对象和设置](#_Toc30875_WPSOffice_Level3) [4](#_Toc30875_WPSOffice_Level3)

[2. Servlet3.0介绍](#_Toc19957_WPSOffice_Level1) [6](#_Toc19957_WPSOffice_Level1)

[2.1. Servlet3.0注解](#_Toc24027_WPSOffice_Level2) [6](#_Toc24027_WPSOffice_Level2)

## 数据库连接池

### 数据库连接池

* 连接池

连接池是在内存中预设好一定数量的连接对象，以备用户在进行数据库操作时直接使用

* 性能

数据库连接的建立、断开均由管理池统一管理

* 连接池技术与传统数据库连接的比较

数据库操作性能得到提升

通过连接池管理数据库的连接与释放、提高了系统资源的使用效率。

|  |
| --- |
|  |

### Druid

* Druid是阿里巴巴开源平台上一个数据库连接池实现
* Druid是Java语言中最好的数据库连接池
* Druid能够提供强大的监控和扩展功能
* Druid可以由不同类型的节点，组成一个集群

#### Druid下载

* druid-x.x.x.jar 项目中导入Jar包

Jar包下载地址：

http://repo1.maven.org/maven2/com/alibaba/druid/1.1.10/druid-1.1.10.jar

源码下载地址：

http://repo1.maven.org/maven2/com/alibaba/druid/1.1.10/druid-1.1.10-sources.jar

#### Druid对象和设置

|  |
| --- |
| * 创建对象配置   DruidDataSource druidDataSource = new DruidDataSource();  druidDataSource.setDriverClassName("com.mysql.jdbc.Driver");//驱动名  druidDataSource.setUrl("jdbc:mysql://192.168.2.2:3306/sqltest");//数据库地址  druidDataSource.setUsername("root");//用户名  druidDataSource.setPassword("root");//密码  //以下为非必填项，不设置也有默认值  druidDataSource.setMinIdle(1);//最小连接池数量 可设置最大值8  druidDataSource.setInitialSize(1);//初始化时建立物理连接的个数  druidDataSource.setMaxActive(10);//最大连接池数量  druidDataSource.setMaxWait(10000);//获得连接最大等待时间 单位毫秒  druidDataSource.setMinEvictableIdleTimeMillis(300000);// 配置一个连接在池中最小生存的时间，单位是毫秒   * 获得连接   Connection conn = druidDataSource.getConnection();  获得连接之后的步骤，与JDBC一致   * 归还连接   Conn.close();  关闭连接并非真的关掉连接，而是把连接返回给数据库连接池，等待下一次调用   * 连接池关闭   druidDataSource.close();  该方法是关闭数据库连接池，执行完之后，不可以再从连接池中获得连接 |

## Servlet3.0介绍

* 使用nio（非阻塞型io）处理 效率相对提高

--8.5及以上版本默认使用nio处理

* 使用注解进行配置

### Servlet3.0注解

* Servlet3.0中可以使用注解（annotation）替代web.xml进行配置；
* 常用的注解有三种类型：
* @WebServlet：对Servlet进行配置

|  |  |
| --- | --- |
| @WebServlet：对Servlet进行配置示例，包括的属性如下，这些属性除了vlaue或urlPatterns是必选的，其他的都是可选的：  1）name：等价于web.xml配置文件中的 <servlet-name>。如果没有指定， Servlet 的<servlet-name>取值为类的全限定名，比如XXX.XXX.XXX。  2）urlPatterns：等价于web.xml配置文件中的 <url-pattern> 标签  3）value：等价于 urlPatterns 属性。  4）loadOnStartup：等价于web.xml配置文件中的<load-on-startup> 标签  5）initParams ：等价于web.xml配置文件中的<init-param> 标签，他的参数是@WebInitParam注解的集合（此注解之后介绍）  6）asyncSupported：等价于web.xml配置文件中的<async-supported> 标签  7）description：等价于web.xml配置文件中的<description> 标签  8）displayName：等价于web.xml配置文件中的 <display-name> 标签 | |
| 注解配置：  @WebServlet(urlPatterns = {"/demo"},  asyncSupported = true,  loadOnStartup = -1,  name = "DemoServlet",  displayName = "DemoServlet ",  initParams = {  @WebInitParam(name = "username",  value = "etc")  } ) | 配置文件配置：  <servlet>  <display-name> DemoServlet </display-name>  <servlet-name>DemoServlet</servlet-name>  <servlet-class>com.sm.SimpleServlet</servlet-class>  <load-on-startup>-1</load-on-startup>  <async-supported>true</async-supported>  <init-param>  <param-name>username</param-name>  <param-value>etc</param-value>  </init-param>  </servlet>  <servlet-mapping>  <servlet-name> DemoServlet </servlet-name>  <url-pattern>/demo</url-pattern>  </servlet-mapping> |

* @WebFilter：配置过滤器

|  |
| --- |
| @WebFilter：配置过滤器此注解为声明一个过滤器，主要属性有以下几个。  在这些属性当中value、urlPatterns、servletNames 三个属性至少要包含其中的一个，并且 value 和 urlPatterns 属性只能有一个，如果两个同时配置，一般情况下value取值将会被忽略。其他的都是可选属性。  filterName：等价于web.xml配置文件中的 <filter-name>标签  value：该属性等价于 urlPatterns 属性  urlPatterns：等价于web.xml配置文件中的 <url-pattern> 标签  servletNames：指定该过滤器将应用的 范围。如果是注解的话取值是 @WebServlet 中的 name 属性的取值，如果servlet这 web.xml 中配置的话，取值是 <servlet-name> 的取值  dispatcherTypes：过滤器的转发模式。取值包括：  ASYNC（异步）、ERROR（错误）、FORWARD（请求转发）、INCLUDE（包含）、REQUEST（请求）。  initParams：等价于web.xml配置文件中的<init-param> 标签  asyncSupported：等价于web.xml配置文件中的<async-supported> 标签  description：等价于web.xml配置文件中的<description> 标签  displayName：等价于web.xml配置文件中的<display-name> 标签 |

* @WebListener ：配置监听器

|  |
| --- |
| @WebListener：配置监听器，此注解是用来声明监听器，它主要的属性只有一个：  value：这个属性表示的是监听器的描述信息，整个配置可以简写成@WebListener("XXX") |