# [Spring Boot MyBatis 连接数据库](http://blog.csdn.net/catoop/article/details/50553714)

最近比较忙，没来得及抽时间把MyBatis的集成发出来，其实mybatis官网在2015年11月底就已经发布了对SpringBoot集成的Release版本，Github上有代码：<https://github.com/mybatis/mybatis-spring-boot>   
前面对JPA和JDBC连接数据库做了说明，本文也是参考官方的代码做个总结。

先说个题外话，SpringBoot默认使用 org.apache.tomcat.jdbc.pool.DataSource   
现在有个叫 HikariCP 的JDBC连接池组件，据说其性能比常用的 c3p0、tomcat、bone、vibur 这些要高很多。   
我打算把工程中的DataSource变更为HirakiDataSource，做法很简单：   
首先在application.properties配置文件中指定dataSourceType:spring.datasource.type=com.zaxxer.hikari.HikariDataSource

1

然后在pom中添加Hikari的依赖

<dependency>

<groupId>com.zaxxer</groupId>

<artifactId>HikariCP</artifactId>

<!-- 版本号可以不用指定，Spring Boot会选用合适的版本 -->

</dependency>

言归正传，下面说在Spring Boot中配置MyBatis。   
关于在Spring Boot中集成MyBatis，可以选用基于注解的方式，也可以选择xml文件配置的方式。通过对两者进行实际的使用，还是建议使用XML的方式（官方也建议使用XML）。

下面将介绍通过xml的方式来实现查询，其次会简单说一下注解方式，最后会附上分页插件（PageHelper）的集成。

### 一、通过xml配置文件方式

#### 1、添加pom依赖

<dependency>

<groupId>org.mybatis.spring.boot</groupId>

<artifactId>mybatis-spring-boot-starter</artifactId>

<!-- 请不要使用1.0.0版本，因为还不支持拦截器插件，1.0.1-SNAPSHOT 是博主写帖子时候的版本，大家使用最新版本即可 -->

<version>1.0.1-SNAPSHOT</version></dependency>

* 1
* 2
* 3
* 4
* 5
* 6

#### 2、创建接口Mapper（不是类）和对应的Mapper.xml文件

定义相关方法，注意方法名称要和Mapper.xml文件中的id一致，这样会自动对应上   
StudentMapper.java

package org.springboot.sample.mapper;

import java.util.List;

import org.springboot.sample.entity.Student;

/\*\*

\* StudentMapper，映射SQL语句的接口，无逻辑实现

\*

\* @author 单红宇(365384722)

\* @myblog http://blog.csdn.net/catoop/

\* @create 2016年1月20日

\*/public interface StudentMapper extends MyMapper<Student> {

List<Student> likeName(String name);

Student getById(int id);

String getNameById(int id);

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22

MyMapper.java

package org.springboot.sample.config.mybatis;

import tk.mybatis.mapper.common.Mapper;

import tk.mybatis.mapper.common.MySqlMapper;

/\*\*

\* 被继承的Mapper，一般业务Mapper继承它

\*

\*/public interface MyMapper<T> extends Mapper<T>, MySqlMapper<T> {

//TODO

//FIXME 特别注意，该接口不能被扫描到，否则会出错

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13

[IMG_258](http://blog.csdn.net/catoop/article/details/javascript:;)

StudentMapper.xml

<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd"><mapper namespace="org.springboot.sample.mapper.StudentMapper">

<!-- type为实体类Student，包名已经配置，可以直接写类名 -->

<resultMap id="stuMap" type="Student">

<id property="id" column="id" />

<result property="name" column="name" />

<result property="sumScore" column="score\_sum" />

<result property="avgScore" column="score\_avg" />

<result property="age" column="age" />

</resultMap>

<select id="getById" resultMap="stuMap" resultType="Student">

SELECT \*

FROM STUDENT

WHERE ID = #{id}

</select>

<select id="likeName" resultMap="stuMap" parameterType="string" resultType="list">

SELECT \*

FROM STUDENT

WHERE NAME LIKE CONCAT('%',#{name},'%')

</select>

<select id="getNameById" resultType="string">

SELECT NAME

FROM STUDENT

WHERE ID = #{id}

</select>

</mapper>

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34

[IMG_259](http://blog.csdn.net/catoop/article/details/javascript:;)

#### 3、实体类

package org.springboot.sample.entity;

import java.io.Serializable;

/\*\*

\* 学生实体

\*

\* @author 单红宇(365384722)

\* @myblog http://blog.csdn.net/catoop/

\* @create 2016年1月12日

\*/public class Student implements Serializable{

private static final long serialVersionUID = 2120869894112984147L;

private int id;

private String name;

private String sumScore;

private String avgScore;

private int age;

// get set 方法省略

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24

[IMG_260](http://blog.csdn.net/catoop/article/details/javascript:;)

#### 4、修改application.properties 配置文件

mybatis.mapper-locations=classpath\*:org/springboot/sample/mapper/sql/mysql/\*Mapper.xml

mybatis.type-aliases-package=org.springboot.sample.entity

* 1
* 2

[IMG_261](http://blog.csdn.net/catoop/article/details/javascript:;)

#### 5、在Controller或Service调用方法测试

@Autowired

private StudentMapper stuMapper;

@RequestMapping("/likeName")

public List<Student> likeName(@RequestParam String name){

return stuMapper.likeName(name);

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9

[IMG_262](http://blog.csdn.net/catoop/article/details/javascript:;)

### 二、使用注解方式

查看官方git上的代码使用注解方式，配置上很简单，使用上要对注解多做了解。至于xml和注解这两种哪种方法好，众口难调还是要看每个人吧。

#### 1、启动类（我的）中添加@MapperScan注解

@SpringBootApplication@MapperScan("sample.mybatis.mapper")public class SampleMybatisApplication implements CommandLineRunner {

@Autowired

private CityMapper cityMapper;

public static void main(String[] args) {

SpringApplication.run(SampleMybatisApplication.class, args);

}

@Override

public void run(String... args) throws Exception {

System.out.println(this.cityMapper.findByState("CA"));

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17

[IMG_263](http://blog.csdn.net/catoop/article/details/javascript:;)

#### 2、在接口上使用注解定义CRUD语句

package sample.mybatis.mapper;

import org.apache.ibatis.annotations.Param;import org.apache.ibatis.annotations.Select;

import sample.mybatis.domain.City;

/\*\*

\* @author Eddú Meléndez

\*/public interface CityMapper {

@Select("SELECT \* FROM CITY WHERE state = #{state}")

City findByState(@Param("state") String state);

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17

[IMG_264](http://blog.csdn.net/catoop/article/details/javascript:;)

其中City就是一个普通Java类。   
关于MyBatis的注解，有篇文章讲的很清楚，可以参考： <http://blog.csdn.net/luanlouis/article/details/35780175>

### 三、集成分页插件

这里与其说集成分页插件，不如说是介绍如何集成一个plugin。MyBatis提供了拦截器接口，我们可以实现自己的拦截器，将其作为一个plugin装入到SqlSessionFactory中。   
Github上有位开发者写了一个分页插件，我觉得使用起来还可以，挺方便的。   
Github项目地址： <https://github.com/pagehelper/Mybatis-PageHelper>

下面简单介绍下：   
首先要说的是，Spring在依赖注入bean的时候，会把所有实现MyBatis中Interceptor接口的所有类都注入到SqlSessionFactory中，作为plugin存在。既然如此，我们集成一个plugin便很简单了，只需要使用@Bean创建PageHelper对象即可。

#### 1、添加pom依赖

<dependency>

<groupId>com.github.pagehelper</groupId>

<artifactId>pagehelper</artifactId>

#### 2、新增MyBatisConfiguration.java

package org.springboot.sample.config;

import java.util.Properties;

import org.slf4j.Logger;import org.slf4j.LoggerFactory;import org.springframework.context.annotation.Bean;import org.springframework.context.annotation.Configuration;

import com.github.pagehelper.PageHelper;

/\*\*

\* MyBatis 配置

\*

\* @author 单红宇(365384722)

\* @myblog http://blog.csdn.net/catoop/

\* @create 2016年1月21日

\*/@Configurationpublic class MyBatisConfiguration {

private static final Logger logger = LoggerFactory.getLogger(MyBatisConfiguration.class);

@Bean

public PageHelper pageHelper() {

logger.info("注册MyBatis分页插件PageHelper");

PageHelper pageHelper = new PageHelper();

Properties p = new Properties();

p.setProperty("offsetAsPageNum", "true");

p.setProperty("rowBoundsWithCount", "true");

p.setProperty("reasonable", "true");

pageHelper.setProperties(p);

return pageHelper;

}

}