



## 《Spring Boot 基础教程》

## 第 28 节

## Spring Boot 集成 mybatis

## 一、添加依赖

```
<!-- mybatis -->
<dependency>
    <groupId>org.mybatis.spring.boot</groupId>
    <artifactId>mybatis-spring-boot-starter</artifactId>
    <version>1.1.1</version>
</dependency>
```

版本说明：最新 mybatis-spring-boot-starter 的版本为 1.2.0-SNAPSHOT，依赖的是 spring boot 的 1.4.1，但是还不是 released 版本。教程的版本为 1.1.1 依赖的 spring boot 的版本为 1.3.3.RELEASE，兼容 spring boot 1.4.x。

GitHub: <https://github.com/mybatis/spring-boot-starter>

## 二、基于 mybatis 注解的集成

## 配置：

```
#mysql
spring.datasource.url=jdbc:mysql://localhost/spring_boot_demo?
useUnicode=true&characterEncoding=utf-8
spring.datasource.username=root
spring.datasource.password=123456
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
```

## bean:

```
package com.roncoo.example.bean;

import java.io.Serializable;
import java.util.Date;

public class RoncooUser implements Serializable {
    private Integer id;

    private String name;

    private Date createTime;

    private static final long serialVersionUID = 1L;

    public Integer getId() {
        return id;
    }
}
```



```
public void setId(Integer id) {
    this.id = id;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name == null ? null : name.trim();
}

public Date getCreateTime() {
    return createTime;
}

public void setCreateTime(Date createTime) {
    this.createTime = createTime;
}

@Override
public String toString() {
    StringBuilder sb = new StringBuilder();
    sb.append(getClass().getSimpleName());
    sb.append(" [");
    sb.append("Hash = ").append(hashCode());
    sb.append(", id=").append(id);
    sb.append(", name=").append(name);
    sb.append(", createTime=").append(createTime);
    sb.append(", serialVersionUID=").append(serialVersionUID);
    sb.append("]");
    return sb.toString();
}
}
```

**mapper:**

**package** com.roncoo.example.mapper;

**import** org.apache.ibatis.annotations.Insert;

**import** org.apache.ibatis.annotations.Mapper;

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

**import** com.roncoo.example.bean.RoncooUser;



```
@Mapper
public interface RoncooUserMapper {

    @Insert(value = "insert into roncoo_user (name,
create_time) values ({name,jdbcType=VARCHAR},
#{createTime,jdbcType=TIMESTAMP})")
    int insert(RoncooUser record);

    @Select(value = "select id, name, create_time from
roncoo_user where id = #{id,jdbcType=INTEGER}")
    @Results(value = { @Result(column = "create_time", property
= "createTime", jdbcType = JdbcType.TIMESTAMP) })
    RoncooUser selectByPrimaryKey(Integer id);
}

test:
package com.roncoo.example;

import java.util.Date;

import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.test.context.junit4.SpringRunner;

import com.roncoo.example.bean.RoncooUser;
import com.roncoo.example.mapper.RoncooUserMapper;

@RunWith(SpringRunner.class)
@SpringBootTest
public class SpringBootDemo281ApplicationTests {

    @Autowired
    private RoncooUserMapper mapper;

    @Test
    public void insert() {
        RoncooUser roncooUser = new RoncooUser();
        roncooUser.setName("测试");
        roncooUser.setCreateTime(new Date());
        int result = mapper.insert(roncooUser);
        System.out.println(result);
    }
}
```



```
@Test
public void select() {
    RoncooUser result = mapper.selectByPrimaryKey(2);
    System.out.println(result);
}

}
```

### 三、基于 mybatis xml 的集成

配置：

```
#mybatis
mybatis.mapper-locations: classpath:mybatis/*.xml
#mybatis.type-aliases-package: com.roncoo.example.bean
```

mapper:

```
package com.roncoo.example.mapper;

import org.apache.ibatis.annotations.Mapper;

import com.roncoo.example.bean.RoncooUser;
```

```
@Mapper
public interface RoncooUserMapper {

    int insert(RoncooUser record);

    RoncooUser selectByPrimaryKey(Integer id);

}
```

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="com.roncoo.example.mapper.RoncooUserMapper" >
    <resultMap id="BaseResultMap"
type="com.roncoo.example.bean.RoncooUser" >
        <id column="id" property="id" jdbcType="INTEGER" />
        <result column="name" property="name" jdbcType="VARCHAR" />
        <result column="create_time" property="createTime"
jdbcType="TIMESTAMP" />
    </resultMap>
```



```
<sql id="Base_Column_List" >
    id, name, create_time
</sql>

<select id="selectByPrimaryKey" resultMap="BaseResultMap"
parameterType="java.lang.Integer" >
    select
    <include refid="Base_Column_List" />
    from roncoo_user
    where id = #{id,jdbcType=INTEGER}
</select>

<insert id="insert"
parameterType="com.roncoo.example.bean.RoncooUser" >
    insert into roncoo_user (id, name, create_time)
    values (#{id,jdbcType=INTEGER}, #{name,jdbcType=VARCHAR},
    #{createTime,jdbcType=TIMESTAMP})
</insert>
</mapper>
```

如何快速批量生成 bean, mapper, xml?

使用 mybatis generator, 龙果开源了 roncoo-mybatis-generator, 集成了多个插件

GitHub: <https://github.com/roncoo/roncoo-mybatis-generator>

更多课程信息, 请关注 龙果学院 官方网站 <http://www.roncoo.com/>

或关注 龙果 微信公众号 RonCoo\_com

