SpringBoot 实际应用

Spring Boot是一个快速搭建基于Spring应用的框架

MyBatis：半自动化框架，只能完成一半工作：连接数据库、封装结果集、执行sql

不能完成，需要开发者自己完成的：接口定义，sql定义

Hibernate全自动，国外用的比较多

MyBatis Plus：既支持自定义SQL，又可以自动完成SQL

基于MyBatis的一个增强框架，进一步简化MyBatis的使用

MyBatis-plus使用步骤

1.导入依赖

```
<dependency>
    <groupId>com.baomidou</groupId>
    <artifactId>mybatis-plus-boot-starter</artifactId>
    <version>3.4.3.1</version>
</dependency>

<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <scope>runtime</scope>
 </dependency>
```

2.配置application.yml

```
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/test
    username: root
    password: 123456
  thymeleaf:
    prefix: classpath:/templates/
    suffix: .html
mybatis-plus:
  configuration:
    log-impl: org.apache.ibatis.logging.stdout.StdOutImpl
```

3.实体类

```
import lombok.Data;
import org.springframework.stereotype.Component;

@Data
@Component
public class Account {
    private Integer id;
    private String name;
    private String money;
    private String password;
}
```

4.自定义接口 AccountMapper继承BaseMapper<映射的实体类名>

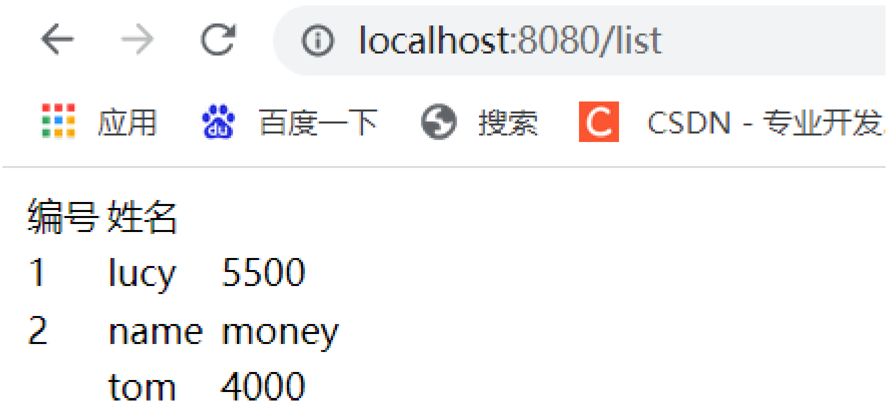
```
import com.baomidou.mybatisplus.core.mapper.BaseMapper;
import com.southwind.entity.Account;
import org.springframework.stereotype.Repository;

@Repository
public interface AccountMapper extends BaseMapper<Account> {

}
```

5.Controller注入mapper

```
@Controller
public class AccountController {

    @Autowired
    private AccountMapper accountMapper;

    @GetMapping("/list")
    public ModelAndView list(){
        ModelAndView modelAndView = new ModelAndView();
        modelAndView.setViewName("index");
        modelAndView.addObject("list", this.accountMapper.selectList(null));
        return modelAndView;
    }
}
```

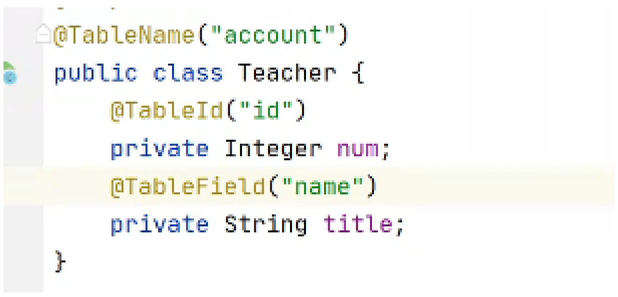
6.启动启动类

```
@MapperScan("com.southwind.mapper")
@SpringBootApplication
public class Springboot002DemoApplication {

    public static void main(String[] args) {

        ConfigurableApplicationContext run =
SpringApplication.run(Springboot002DemoApplication.class, args);

    }
}
```



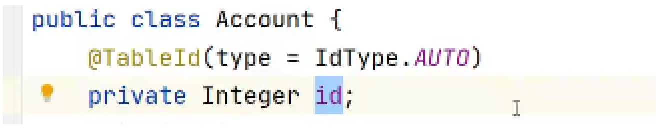
```
编号 姓名
1    lucy   5500
2    name  money
     tom    4000
```

数据库字段名和实体名不对应的时候，使用注解@TableName，@TableId，@TableField来对应



```java
@TableName("account")
public class Teacher {
    @TableId("id")
    private Integer num;
    @TableField("name")
    private String title;
}
```

SELECT id AS num,name AS title FROM account



```java
public class Account {
    @TableId(type = IdType.AUTO)
    private Integer id;
```

MyBatis-plus中使用自定义sql

AccountMapper

```java
@Repository
public interface AccountMapper extends BaseMapper<Account> {
    public List<Account> list();

}
```

AccountMapper.xml

```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.southwind.mapper.AccountMapper">
    <!--namespace根据自己需要创建的的mapper的路径和名称填写-->



    <select id="list" resultType="com.southwind.entity.Account">
            select  * from account
    </select>
</mapper>
```

controller

```
@Controller
public class AccountController {

    @Autowired
    private AccountMapper accountMapper;

    @GetMapping("/list")
    public ModelAndView list(){
        ModelAndView modelAndView = new ModelAndView();
        modelAndView.setViewName("index");
        modelAndView.addObject("list", this.accountMapper.list());
        return modelAndView;
    }
}
```

```
mybatis-plus:
  configuration:
    log-impl: org.apache.ibatis.logging.stdout.StdOutImpl
  mapper-locations: classpath*:com/southwind/mapper/*.xml
```

需要在application.yml中加入mapper-locations

2.mybatis-plus自定义sql有参数时，需要加上@Param

AccountMapper:

```
@Repository
public interface AccountMapper extends BaseMapper<Account> {

    public Account findByName(@Param("username") String username);

}
```

AccountMapper.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.ishang.mapper.AccountMapper">

    <select id="findAll" resultType="com.ishang.entity.Account">
        select * from account
    </select>
    <select id="findByName" resultType="com.ishang.entity.Account">
        select * from account where username= #{username}
    </select>

</mapper>
```

Mybatis-plus常用方法

1.分页查询

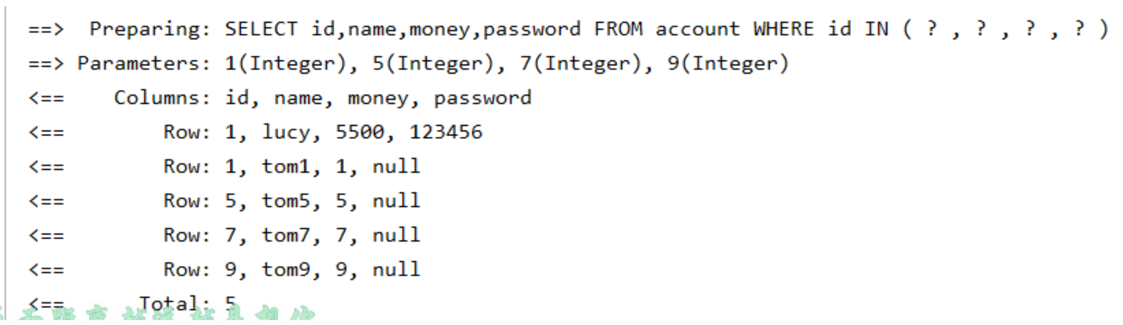
分页查询需要创建pageConfiguration

```java
package com.southwind.configure;


import com.baomidou.mybatisplus.extension.plugins.PaginationInterceptor;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

//mypatis-plus使用分页查询需要有一个分页配置类
@Configuration
public class PageConfiguration {
    @Bean
    public PaginationInterceptor paginationInterceptor(){
        return  new PaginationInterceptor();
    }
}
```

```java
Page<Account> page = new Page<>(2,10);
        IPage<Account> accountIPage = this.accountMapper.selectPage(page, null);
//          一共有多少条数据
        System.out.println(page.getTotal());
//        每页有几条数据
        System.out.println(page.getSize());
//          当前是第几页
        System.out.println(page.getCurrent());
//         一共有多少页
        System.out.println(page.getPages());
        List<Account> records = page.getRecords();
        for (Account record : records) {
            System.out.println(record);
        }
    }
```

2.查询多条

```java
System.out.println(this.accountMapper.selectBatchIds(Arrays.asList(1, 5, 7,
9)));
```



```
==>  Preparing: SELECT id,name,money,password FROM account WHERE id IN ( ? , ? , ? , ? )
==> Parameters: 1(Integer), 5(Integer), 7(Integer), 9(Integer)
<==    Columns: id, name, money, password
<==        Row: 1, lucy, 5500, 123456
<==        Row: 1, tom1, 1, null
<==        Row: 5, tom5, 5, null
<==        Row: 7, tom7, 7, null
<==        Row: 9, tom9, 9, null
<==      Total: 5
```

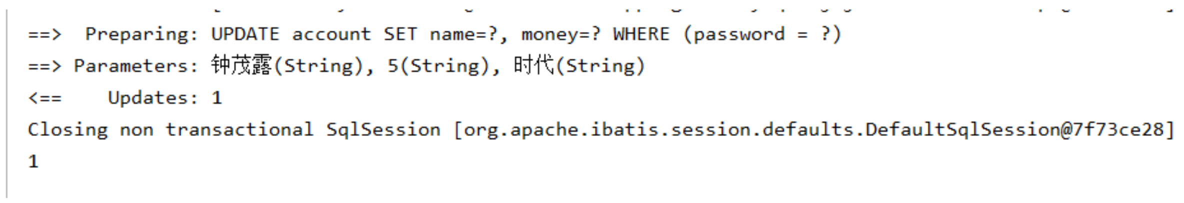
3.map

map与querywrapper不同的是map只能查询等值的，不能像querywrapper可以有模糊查询等

```
Map<String,Object> map = new HashMap<>();
map.put("id",1);
System.out.println(this.accountMapper.selectByMap(map));
```

4.修改

```
/         根据id来修改
        Account account = this.accountMapper.selectById(5);
        account.setName("钟茂露");
        this.accountMapper.updateById(account);

//         根据querywrapper的条件进行修改
        QueryWrapper<Account> queryWrapper = new QueryWrapper<>();
        queryWrapper.eq("password","时代");
        System.out.println(this.accountMapper.update(account, queryWrapper));
```



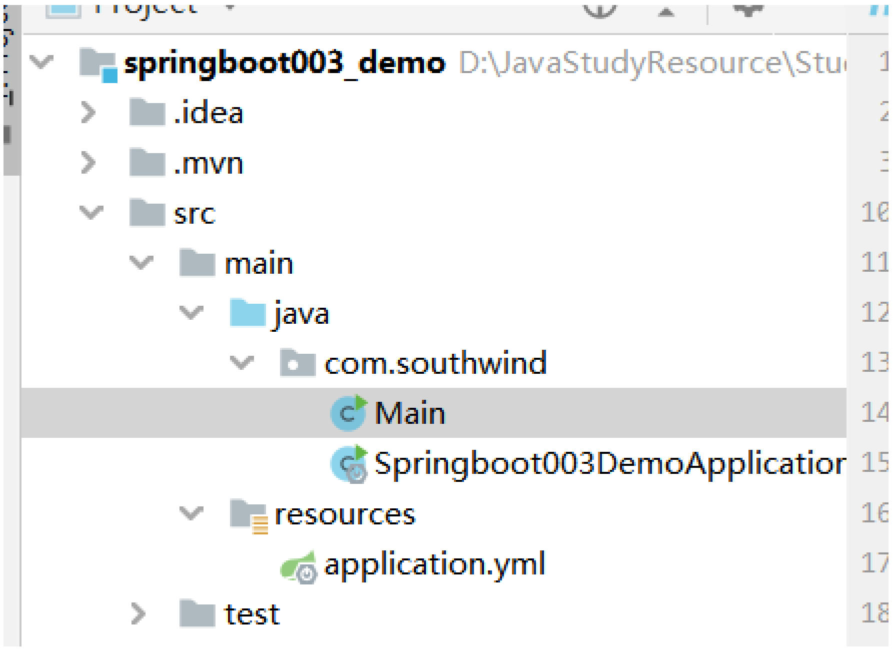
```
==>  Preparing: UPDATE account SET name=?, money=? WHERE (password = ?)
==> Parameters: 钟茂露(String), 5(String), 时代(String)
<==    Updates: 1
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@7f73ce28]
1
```

5.代码自动生成功能

1.导入依赖

```
<dependency>
    <groupId>com.baomidou</groupId>
    <artifactId>mybatis-plus-generator</artifactId>
    <version>3.3.2</version>
</dependency>

<dependency>
    <groupId>org.apache.velocity</groupId>
    <artifactId>velocity</artifactId>
    <version>1.7</version>
</dependency>
```

2.创建一个类，复制以下代码



```
package com.southwind;

import com.baomidou.mybatisplus.annotation.DbType;
import com.baomidou.mybatisplus.generator.AutoGenerator;
import com.baomidou.mybatisplus.generator.config.DataSourceConfig;
import com.baomidou.mybatisplus.generator.config.GlobalConfig;
import com.baomidou.mybatisplus.generator.config.PackageConfig;
import com.baomidou.mybatisplus.generator.config.StrategyConfig;
import com.baomidou.mybatisplus.generator.config.rules.NamingStrategy;

public class Main {
    public static void main(String[] args) {
        //创建对象
        AutoGenerator autoGenerator = new AutoGenerator();
        //数据源
        DataSourceConfig dataSourceConfig = new DataSourceConfig();
        dataSourceConfig.setDbType(DbType.MYSQL);
        dataSourceConfig.setDriverName("com.mysql.cj.jdbc.Driver");
        dataSourceConfig.setUrl("jdbc:mysql://localhost:3306/test");
        dataSourceConfig.setUsername("root");
        dataSourceConfig.setPassword("123456");
        autoGenerator.setDataSource(dataSourceConfig);
        //全局配置
        GlobalConfig globalConfig = new GlobalConfig();

 globalConfig.setOutputDir(System.getProperty("user.dir")+"/src/main/java");
        globalConfig.setAuthor("admin");
        globalConfig.setOpen(false);
        //去掉Service的I
        globalConfig.setServiceName("%sService");
        autoGenerator.setGlobalConfig(globalConfig);
        //包配置
        PackageConfig packageConfig = new PackageConfig();
        packageConfig.setParent("com.southwind");
        packageConfig.setEntity("entity");
        packageConfig.setMapper("mapper");
        packageConfig.setService("service");
        packageConfig.setServiceImpl("service.impl");
        packageConfig.setController("controller");
        autoGenerator.setPackageInfo(packageConfig);
        //策略配置
        StrategyConfig strategyConfig = new StrategyConfig();
//          根据哪张表生成
        strategyConfig.setInclude("account");
        strategyConfig.setEntityLombokModel(true);
        autoGenerator.setStrategy(strategyConfig);
        //启动
        autoGenerator.execute();
    }
}
```