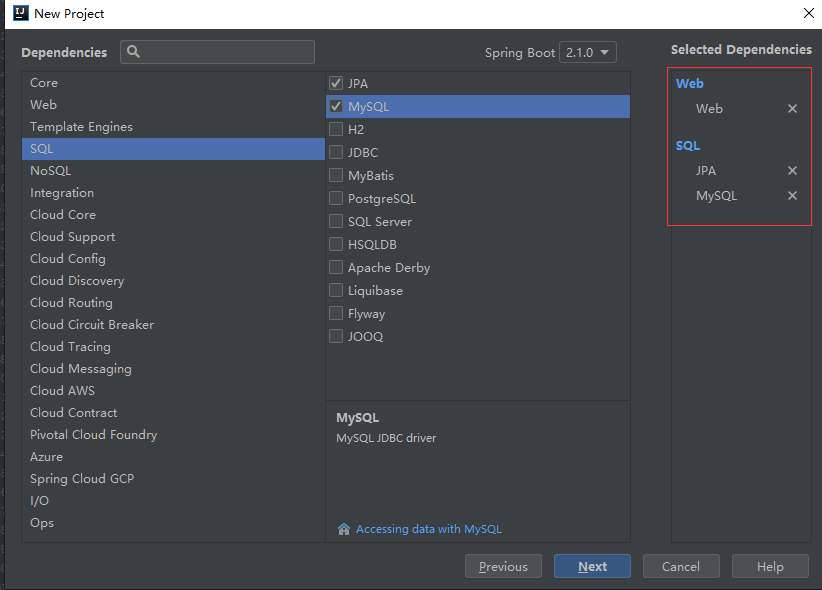
# 实验五

本实验通过Intellij IDEA 代替Visual Studio；Spring平台代替.NET

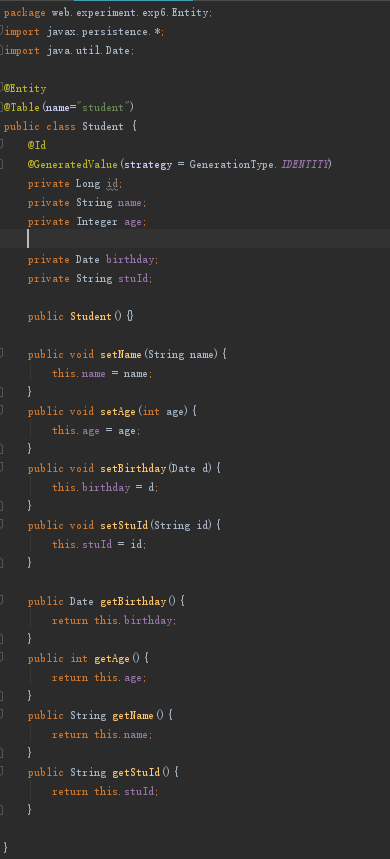
1. 新建项目

在勾选组件：



1. 编写代码

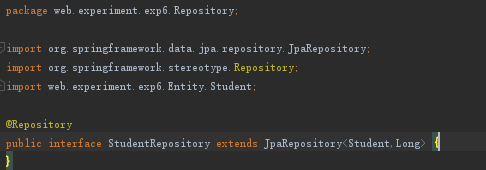
添加实体类，Student，编写代码：



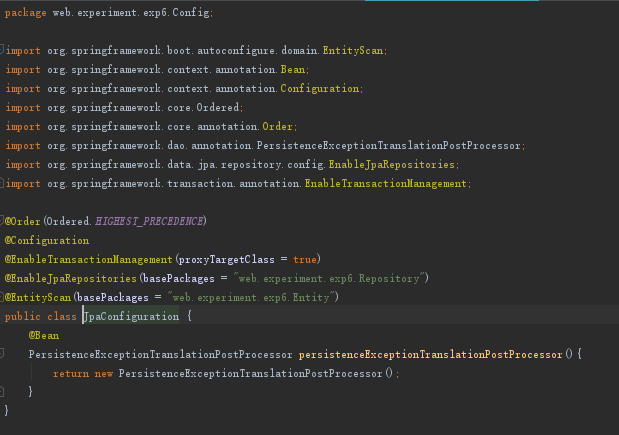
实体持久化

通过Student类的定义，实现了使用Java的普通对象（POJO）与数据库表建立映射关系（ORM），接下来使用JPA来实现持久化。

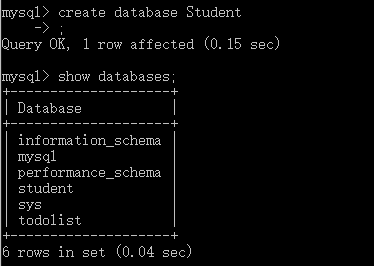
实体持久化的实现在Spring中是通过一个定义一个“接口”，并继承于JPA资源库JpaRepository接口，使用注解@Repository将这个接口也定义为一个资源库，使它能被其他程序引用，并为其他程序提供存取数据库的功能。



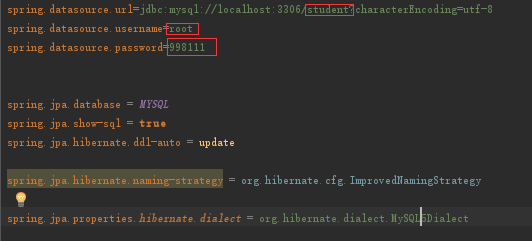
添加配置类



新建数据库Student



编写resources目录下的application.properties文件



编写控制器：

package web.experiment.exp6;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.\*;  
import web.experiment.exp6.Entity.Student;  
import web.experiment.exp6.Repository.StudentRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.PageRequest;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.domain.Sort;  
  
import java.util.ArrayList;  
import java.util.List;  
  
  
@Controller  
@SpringBootApplication  
public class Exp6Application {  
 @Autowired  
 private StudentRepository stuRespo;  
  
 @RequestMapping(path="/text")  
 public String text(){  
 return "text";  
 }  
  
  
 @PostMapping(path="/add")  
 @ResponseBody  
 public String addNewStu(@RequestBody Student stu){  
 stu.setName(stu.getName());  
 stu.setAge(stu.getAge());  
 stu.setBirthday(stu.getBirthday());  
 stu.setStuId(stu.getStuId());  
 stuRespo.save(stu);  
 return "success";  
 }  
  
 @GetMapping(path="/get")  
 @ResponseBody  
  
 public List<Student> getStudents(){  
 List<Student> list = new ArrayList<Student>();  
 Pageable pageable = new PageRequest(0,10, new Sort(Sort.Direction.*ASC*,"id"));  
 Page<Student> page = stuRespo.findAll(pageable);  
 list = page.getContent();  
 return list;  
 }  
  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(Exp6Application.class, args);  
 }  
}

编写视图层界面

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8" />  
 <meta http-equiv="X-UA-Compatible" content="IE=edge">  
 <title>Student</title>  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <script src="/jquery.min.js"></script>  
</head>  
<body>  
 <form>  
 姓名：<input type="text" name="name"><br>  
 年龄：<input type="text" name="age"><br>  
 生日：<input type="text" name="birthday"><br>  
 学号：<input type="text" name="stuId"><br>  
 <button>提交</button>  
 </form>  
 <br><br>  
 <table>  
 <tr>  
 <td>姓名</td>  
 <td>年龄</td>  
 <td>生日</td>  
 <td>学号</td>  
 </tr>  
 </table>  
   
  
</body>  
<script>  
 $(document).ready(**function**(){  
  
 getInfo();  
  
 $("button").click(**function**(){  
 **var** settings = {  
 "async": **true**,  
 // "crossDomain": true,  
 "url": "http://localhost:8080/add",  
 "method": "POST",  
 "headers": {  
 "Content-Type": "application/json",  
 "Cache-Control": "no-cache",  
 "Postman-Token": "43325fb5-7b74-4922-95e5-eaf17f5545a3"  
 },  
 "processData": **false**,  
 "data": "{\n\t\"name\":\"TheShy\",\n\t\"age\":7,\n\t\"birthday\":\"1998-01-11\",\n\t\"stuId\":\"77777777\"\n}"  
 }  
  
 $.ajax(settings).done(**function** (response) {  
 console.log(response);  
 });  
  
 getInfo();  
 })  
 })  
  
 **function** getInfo(){  
 **var** settings = {  
 "async": **true**,  
 // "crossDomain": true,  
 "url": "http://localhost:8080/get",  
 "method": "GET",  
 "headers": {  
 "Cache-Control": "no-cache",  
 "Postman-Token": "b61811a8-335b-4e54-9486-9166cdff905c"  
 }  
 }  
  
 $.ajax(settings).done(**function** (response) {  
 $("table").empty();  
 **for**(**var** i=0;i<response.length;i++){  
 **var** str = '<tr><td>'+response[i].name+'</td><td>'+response[i].age+'</td><td>'+response[i].birthday+'</td><td>'+response[i].stuId+'</td></tr>'  
 $("table").append(str);  
 }  
 });  
 }  
</script>

运行程序，浏览器访问：localhost:8080/text，完成了一个简单的MVC程序

