使用RESTful架构完成一个demo

需求

- 添加课程，成功则返回全部课程信息。
- 查询课程，通过id查询对应的课程信息。
- 修改课程，成功则返回修改之后的全部课程信息。
- 删除课程，成功则返回删除之后的全部课程信息。

Spring MVC模型数据解析

Spring MVC提供了以下几种方法添加模型数据：

- Map
- Model
- ModelAndView
- @SessionAttributes
- @ModelAttribute

模型数据绑定到request对象中

- Map，开发者只需要在业务方法中定义Map形参，方法体中对该形参进行操作，Spring MVC框架会自动取出该形参中的数据并存入request对象，可以在JSP通过EL表达式直接获取对象。

```
@RequestMapping("/map")
public String map(Map<String, Course> map){
    Course course = new Course(1,"Java基础",2000);
    map.put("c1",course);
    return "test";
}
```

- Model，Model的使用与Map类似。

```
@RequestMapping("/model")
public String model(Model model) {
    Course course = new Course(1,"Java基础",2000);
    model.addAttribute("c1",course);
    return "test";
}
```

- ModelAndView

与Map、Model不同的是，ModelAndView不但包含模型数据，同时也包含了视图信息，所以使用ModelAndView来处理模型数据，业务方法的返回值必须是ModelAndView，业务方法中对ModelAndView进行两个操作：1、填充模型数据。2、绑定视图信息。

```java
@RequestMapping("/mav1")
public ModelAndView modelAndView1(){
    ModelAndView modelAndView = new ModelAndView();
    Course course = new Course(1,"Java基础",2000);
    modelAndView.addObject("c1",course);
    modelAndView.setViewName("test");
    return modelAndView;
}

@RequestMapping("/mav2")
public ModelAndView modelAndView2(){
    ModelAndView modelAndView = new ModelAndView();
    Course course = new Course(1,"Java基础",2000);
    modelAndView.addObject("c1",course);
    View view = new InternalResourceView("/test.jsp");
    modelAndView.setView(view);
    return modelAndView;
}

@RequestMapping("/mav3")
public ModelAndView modelAndView3(){
    ModelAndView modelAndView = new ModelAndView("test");
    Course course = new Course(1,"Java高级",2000);
    modelAndView.addObject("c1",course);
    return modelAndView;
}

@RequestMapping("/mav4")
public ModelAndView modelAndView4(){
    View view = new InternalResourceView("/test.jsp");
    ModelAndView modelAndView = new ModelAndView(view);
    Course course = new Course(1,"Java高级",2000);
    modelAndView.addObject("c1",course);
    return modelAndView;
}

@RequestMapping("/mav5")
public ModelAndView modelAndView5(){
    Map<String,Course> map = new HashMap<>();
    Course course = new Course(3,"Spring MVC框架",3000);
    map.put("c1",course);
    ModelAndView modelAndView = new ModelAndView("test",map);
    return modelAndView;
}

@RequestMapping("/mav6")
public ModelAndView modelAndView6(){
    Map<String,Course> map = new HashMap<>();
    Course course = new Course(3,"Spring MVC框架",3000);
    map.put("c1",course);
    View view = new InternalResourceView("/test.jsp");
    ModelAndView modelAndView = new ModelAndView(view,map);
    return modelAndView;
}

@RequestMapping("/mav7")
public ModelAndView modelAndView7(){
    Course course = new Course(1,"Java基础",2000);
    ModelAndView modelAndView = new ModelAndView("test","c1",course);
    return modelAndView;
}

@RequestMapping("/mav8")
public ModelAndView modelAndView8(){
    Course course = new Course(2,"Spring框架",3000);
    View view = new InternalResourceView("/test.jsp");
    ModelAndView modelAndView = new ModelAndView(view,"c1",course);
    return modelAndView;
}
```

- HttpServletRequest、HttpSession

```
@RequestMapping("/request")
public String request(HttpServletRequest request){
    Course course = new Course(1,"Java基础",2000);
    request.setAttribute("c1",course);
    HttpSession session = request.getSession();
    session.setAttribute("c1",course);
    return "test";
}

@RequestMapping("/session")
public String session(HttpSession session){
    System.out.println(session);
    session.setAttribute("c1",new Course(1,"Java",2000));
    return "test";
}
```