第四讲要点

第三讲作业

1. `@Data`
   加上这个注解，不用再写private属性的getset方法，会自动填充，可直接调用
2. 先解析后创建
3. beans下面的bean
   ```
   List<Node> list = document.selectNodes("b:beans/b:bean");
   ```
4. xml

   ```
   <beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframew

        <!--方式一：构造方法实例化bean-->
        <bean id="bookDao" class="com.itheima.dao.impl.BookDaoImpl"/>

    </beans>
   ```

   xmlns为namespace，在springboot解析时，是用namespace加上bean的名字来做的
   如果前面没有的话，则默认为缺省的xmlns
   在例子中是 `"http://www.springframework.org/schema/beans"+"bookDao"`
   如果写为 `<xsi:bean/>` ,则是用xsi所代表的namespace来连接bean。
5. isInstance
   判断指定对象是否拥有指定类型,返回类型为bool
   type.isInstance(object)

一、Web App

传统的Web App模式，客户端向服务端发送一个请求，服务端返回一个HTML渲染页面，所有的业务逻辑都要在服务端完成，缺点在于存在着大量的冗余信息

改进后的模式 SPA

页面展示和查询数据分离

服务器前端存放静态资源，诸如渲染等等，后端执行业务逻辑。在浏览器上运行前端，在服务器上运行后端。

请求页面，服务器返回HTML页面

当是数据请求时，客户端发送AJAX，服务器返回JSON格式的信息体

二、Spring Boot

目前多用Spring Boot和MyBatis-Plus的后端开发模式，不再用传统的SSM模式

1. 内置了应用服务器，Embedding Tomcat, Jetty Undertow，程序可以直接运行，而不用再配置其他条件
2. 提供"starter"，可以直接把依赖一次性写入，不需要一条条引用
3. 一般用Annotation，而不是xml进行配置

https://start.spring.io/ 直接通过官网下载完整的应用框架

pom.xml

1. `<parent></parent>` 表示继承某一个项目，其中继承后，内库配置可以直接拿来用
2. `<dependenies></dependenies>` 通过starter机制，可以一次性将所有的依赖存入到"dnpendenies"文件
3. `<plugins><plugins>` 用来定制化，package在运行时调用repackage

Demo

- `@SpringBootApplication`
  表示打开一个端口，标注在main函数的类上
- `@RestController`
  为响应web请求，标注在调用函数的类上
- `@RequestMapping("hello")`
  hello为起始基础路径，basapath
- `@GetMapping("/hello")`
  通过路由，提供给外面调用，标注在调用函数上
- 配置文件
  可以通过.yaml或者.properties来配置，现在一般用的是.yaml
  一般用于配置一些数据库连接等关键字，可以配置内置属性，也可以配置自定义属性
  Eg：配置端口号

```
(.yaml)
server:
    port: 8080


(.properties)
server.port = 8080
```

三、Restful API

`RestController` 是一种访问远程API的一种模式

注意这里需要区别和 `@Controller` 的区别—— `@Controller` 是接受请求，返回跳转页面，是MVC格式的。而 `RestController` 是返回json格式，是Spring Boot格式

Web Service是基于xml标准化消息，响应和请求都放到xml里，而远程API是通过路由来访问的

1. **请求**
   请求=动词+宾语
   动词使用 `GET/POST/PUT/PATCH/DELETE` 五种方法
   宾语URL应该全部使用名词复数（可以有例外）
   过滤信息：限制返回记录，返回记录的开始位置等等
2. **回应**
   回应包括了HTTP状态码和数据俩个部分
   2开头为正确
   4开头为用户（调用者）请求正确
   5开头为服务端内部错误
3. API

API	请求体	响应正文
GET	没有请求体	消息体
POST	发送的时候可以加上一个消息体	消息体
PUT	发送的时候可以加上一个消息体	无消息体
DELETE	没有请求体	无消息体

4. Lombok
   在类上标注 `@Data` ， 自动生成private的getter/setter

调Lombok自动编译

5. 线程安全

```
Map<~> todo = Collections.synchronizedMap(new Hash<~>());
```

6. 自动诸注入
   一个Controller层，一个Service层
   可以在Controller层创建Service对象时，在这一对象上加上 `@Autowired` 实现自动注入
7. **传参**
   路径参数和基本数据
   `@GetMapping("/{id}")` 路径参数一般只用于传标识id
   `@GetMapping("")` 基本数据，通过 `?name=作业` 的路径，最后通过参数匹配来找到调用函数
   特殊的
   传的是一个消息体
   用 `@RequesBoby` 在参数中标记，表示传入的是一个对象
   json格式通过反序列化把值创建到对象中去
   > ***[PAT]一个函数参数只允许有一个ResquestBoby***

8. **状态码**
   要合理的返回状态码，不能只返回数据，包装一下
   Controller层的返回值类型，要用ResponceEntity包装
   可以返回 `OK(result)` ,状态码为200
   还可以返回 `ResponseEntity.noContent().build()` ， 状态码是204

四、Swagger

一些在Swagger显示的汉字注解

```
@Api(tags = "xx")

@ApiOperation("xx")

@ApiParam("xx")
```

五、前端

- vue
- element-plus
- Aios

用vue实现双向的数据绑定

```
axios.get(path)
    .then(response => this.todos = response.data)
    .catch(e => this.$message.error(e.response.data))
```

如果是2xx就到then，如果是4xx就到catch

这里的发送请求是异步的