1. springboot：单一应用的开发；springcloud：多个springboot应用
2. 命令行gradle构建
3. 通过<https://start.spring.io/> 创建项目，dependencies输入web选第一个
4. 解压第一步生成的文件，更改文件中的build.gradle中的依赖下载路径
5. 命令行转到此文件夹，使用gradle build 构建项目
6. 构建完成后，会生成可执行的文件，只用第一个就行



1. 转到此目录，使用java –jar start-0.0.1-SNAPSHOT.jar直接运行，为了方便可改名字
2. 可以直接复制生成的项目到另一个文件夹中，变成另一个项目，只要简单改动gradle.build就行，以下两个文件不用复制，因为是构建时生成的。



1. use default gradle wrapper：用各个版本springboot默认的gradle版本
2. use local gradle distribution：用本机自带的gradle构建项目
3. thymeleaf
4. Java模板引擎。能够处理HTML、XML、JavaScript、CSS甚至纯文本。类似JSP、Freemarker。
5. 自然模板。原型即页面，只是后期后端数据覆盖一些原始数据，减少了设计人员和开发人员的沟通。
6. 语法优雅易懂。OGNL、SpringEL。
7. 遵循WEB标准。支持HTML5，HTML5好处，不同客户端效果一样
8. <span th:text=”…”>：需要在html头中引入标签
9. <span data-th-text=””>：html标准
10. 表达式：
11. 变量表达式：${…} <span th:text=”${book.author.name}”>
12. 消息表达式：#{…} <th th:text=”#{header.address.city}”>，也称文本外部化、国际化，指的是英文输出hello，中文输出你好！
13. 选择表达式：\*{…} <div th:object=”${book}”><span th:text=”\*{title}”><di>

取book变量中的title属性，与变量表达式的区别：它们是在当前选择的对象而不是整个上下文变量映射上执行。范围小，效率高。

④ 链接表达式：@{…} <a th:href=”@{}”>

⑤ 分段表达式：th:insert或th:replace

首先定义片段：<div th:fragment=”name”>…<div>

再选择片段：<div th:insert=”~{footer::name}”>

1. 字面量
2. 文本：<span th:text=”’working web’”>，用单引号扩起来
3. 数字：<span th:text=”2014”>，也可以有计算th:text=”2013+2”
4. 布尔：<span th:if=”${user.isAdmin()}==false”>
5. Null：<span th:if=”${variable.something}==null”>
6. 算术计算：<div th:with=”isEven=(${prodStat.count}%2==0)”>
7. 比较：>/</>=/<=(gt,lt,ge,le)

<ul class=”pagination”data-th-if=”${page.totalPages le 7}”>

1. 等价：==,!=(eq,ne)

<option data-th-each=”i:${#arrays.toIntegerArray({5,10,40,100})}”data-th-value=”${i}”data-th-selected=”${i eq page.size}”data-th-text=”i”/>

1. 条件运算符：<tr th:class=”${row.even}? ’even’:’odd’”>,如果row.even是true则为even
2. 无操作：<span th:text=”${user.name}?:\_”>no user authenticated</span>若user.name不存在，则直接显示自定义文本
3. 设置属性值
4. 设置任意属性值 th:attr，定义动态属性值，action和value会覆盖原型了的值，但原型可能没有

<form action=”subscribe.html” th:atrr=”action=@{/subscribe}”>

<fieldset>

<input type=”text” name=”email”/>

<input type=”submit”

value=”subscribe!”th:attr=”value=#{subscribe.submit}”>

</fieldset>

</form>

1. 设置值到指定的属性，原型必须有

<form action=”subscribe.html” th: action=”@{/subscribe}”>

<form action=”subscribe.html” th: value=” #{subscribe.submit}}”>

1. 固定值布尔属性

<input type=”checkbox”name=”option1”checked/> html

<input type=”checkbox”name=”option2”checked=”checked”/> xhtml

<input type=”checkbox”name=”aa”th:checked=”${user.active}”/>thyme

1. 迭代器th:each循环遍历

<li th:each=”book:${books}”th:text=”${book.title}”>en…</li>：遍历books数组，取出book中的title属性，如果有10个book那就有10个li

状态变量：跟踪迭代器的状态，

Index:索引，从0开始

Count:索引，从1开始

Size:总数

Current:当前变量

even/odd:当前是奇偶

first:当前是否第一个

last: 当前是否最后一个

1. 条件语句

① th:if ② th:unless ③th:switch,th:case

1. 模板布局：公有片段

th:fragment=”name”th:insert=”~{页面名称::name}”

也可以通过id来引用：th:insert=”~ {footer::#id名}”

Th:insert th:replace th:include三者的区别

① th:insert它将简单地插入指定的片段作为正文的主标签

② th:replace用指定实际片段来替换其主标签

③ th:include类似th:insert，只插入片段中的内容，不推荐

附图：手机截图

1. 属性优先级，但一个标签中写入多个th:属性时，会发生什么

<li th:each=”item:${items}”th:text=”${item.description}”>

附图：手机截图

1. 注释
2. 标准HTML/XML注释

<!—dd -->

1. Thymeleaf解析器级注释块

删除<!--/\* 和 \*/-->之间的所有内容，静态时保留

1. 原型注释块

<!--/\*/ 和 /\*/-->之间的内容

当模板静态打开是，原型注释块所注释的代码将被注释，模板执行时，代码显示，与解析器级注释相反

1. 内联的逻辑

[[…]]和[(…)]分别对应于th:text和th:utext，utext会对特殊符号转义

<p>the msg is “[(${msg})]”</p> 🡪 <p>the msg is “this is msg”</p>

禁用内联：有时候需要比如以下

<p th:inline=”none”>a double is [[1,2,3],[2,3,4]]</p>

[Js内联说明](#js内联)

[Css内联](#css内联)

1. 表达式基本对象
2. #ctx:上下文对象：${#ctx.request}、${#ctx.response}、${#ctx.Parma}
3. #local
4. Request/session等属性

Param：用于检索请求参数

Session：用于检索session属性

Application：用于检索application/servlet上下文属性

1. Web上下文对象
2. 工具对象

执行信息：#execinfo

消息：#messages

URI/UL：#uris

转换：#conversions

日期：#dates

日历：#calendars

数字：#numbers

字符串：#strings

对象：#objects

布尔：#bools

数组：#arrays

链表：#lists

Set：#sets

Map：#maps

聚合：#aggregates,统计求平均值

ID：#ids，处理重复属性

1. 集成thymeleaf
2. 添加依赖

compile('org.springframework.boot:spring-boot-starter-thymeleaf')

1. 自定义thymeleaf版本，若使用的springboot中集成的thymeleaf版本低，可以指定高版本

buildscript{

ext[‘thymeleaf.version’]=”3.0.3.RELEASE”

ext[‘thymeleaf-layout-dialect.version’]=”2.2.0”

}

1. Thymeleaf实战

[API设计](#app设计)

1. 注意用户添加修改操作，不能操作id，所以，可以根据id进行修改和添加，如果用户没有id，证明是添加，有id则根据此id修改其他属性

10、全文搜索

非结构化数据检索

1. 顺序扫描法
2. 全文搜索：将非结构化的数据，转为结构化的数据，然后创建索引
   1. Lucene ：引擎
   2. ElasticSearch：基于Lucene，只支持json，实时处理
   3. Solr：基于Lucene，支持很多

11、ElasticSearch

1）高度可扩展的开源全文搜索和分析引擎

2）快速地、近实时地对大数据进行存储、搜索和分析

3）用来支撑有复杂数据搜索需求的企业级应用

特点：

1. 分布式
2. 高可用
3. 多类型
4. 多api
5. 面向文档
6. 异步写入
7. 近实时
8. 基于Lucene
9. Apache协议

近实时：大概1秒，lucene可以实现全实时，那么要么牺牲索引效率，要么牺牲查询效率。每个n秒自动刷新，进行增删改操作。根据刷新策略，定期同步到磁盘中去。

集群：

节点：集群中的单台服务器

索引： 类型： 文档：进行索引的基本单位 分片： 副本：

12、bootstrap

1）布局样式、样式例子

2）常用组件、样式

Typography：字体样式

Table：表格组件

Form：表单

Button：按钮

Drop down：下拉组件

Buttongroup：按钮组

Inputgroup：输入框

Navbar：导航栏样式，自适应

Pagination：分页组件

Tag：标签

Alert：提示框

ModalDialog：模式对话框

Progressbar：进度条

Listgroup：列表

Card：卡片

Tooltip：鼠标移到特定地方，的提示

13、常见前端框架

1）bootstrap

2）jQuery

3）Font awesome：图标库

4）Nprogress：进度条

5）Thinker-md：makerdown插件

6）jQuery tags input：博客标签

7）bootstrap chosen：下拉插件

8）toastr：处理信息提示

14、样式什么时候用，用在哪

Container：将body里的所有内容用<div class=”container [样式]”></div>包裹起来

按钮样式：class=”btn btn-”

15、spingsecurity

1）用户登录的时候，先被拦截，进入安全配置类，认证信息管理，configraGlobal，成功，在进行handle映射，之间返回页面，失败，返回自定义登录页面



2）SCRF防护：请求中带有一个SCRF token如果没有，则拒绝，thymeleaf-security集成了，这样form表单就会自带scrf令牌，但是别的操作不行，因为使用的不是thymeleaf，可能是js，所以，这样要自己处理。

16、MongoDB文件服务器



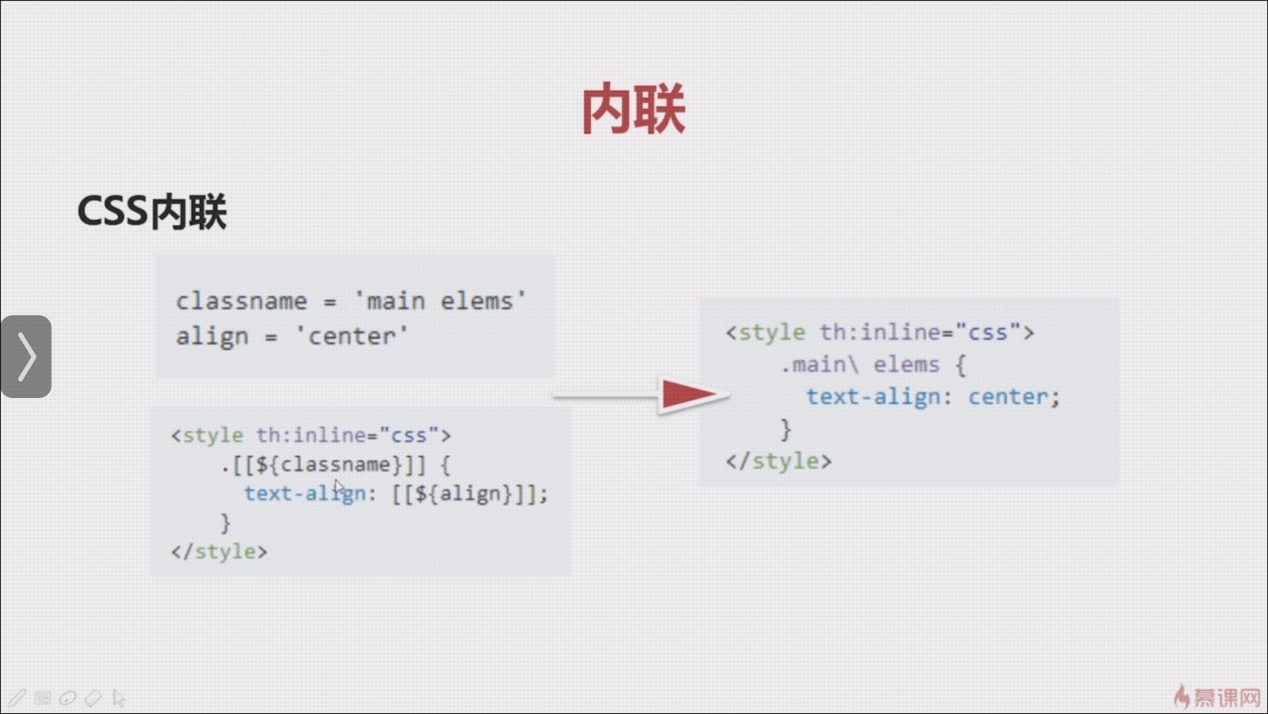
17、编辑图片

Cropper.js已经集成

1. js内联图



1. css内联图



1. 模板布局



1. ThymeleafAPI设计

