Springboot+IDEA构建Web应用

- 在springboot下进行web开发
- 静态资源导入探究
- 系统首页在springboot里是如何设置的
- Thymeleaf模版引擎应用
- springboot中MVC的配置原理

在springboot下进行web开发

在springboot下已经没有webapp目录，web项目如何搭建

springboot的最大特点就是：自动装配

xxxAutoConfiguration

自动装配

springboot到底帮我们配置了什么？能不能改？改什么？能不能扩展?

xxxAutoConfiguration 向容器中自动配置组件

xxxProperties 自动配置类，自定义的一些内容

web开发要解决的问题

- 导入静态资源
- 首页
- jsp，模块引擎Thymeleaf
- 装配扩展springMVC
- CURD
- 拦截器
- 国际化

静态资源导入的探究

```
private static final String[] CLASSPATH_RESOURCE_LOCATIONS = { "classpath:/META-INF/resources/",
        "classpath:/resources/", "classpath:/static/", "classpath:/public/" };
```

什么原因：系统只支持这四个路径

优先级：resources/resources、static、public

public：放公共的js、css等

static：放静态资源

resources：上传的资源文件

首页加载方式

jsp，index.html，index.jsp

```
private Optional<Resource> getWelcomePage() {
    String[] locations = getResourceLocations(this.resourceProperties.getStaticLocations());
    return Arrays.stream(locations).map(this::getIndexHtml).filter(this::isReadable).findFirst();
}

private Resource getIndexHtml(String location) {
    return this.resourceLoader.getResource( s: location + "index.html");
}
```

Thyemleaf

---

thyemleaf是一个模版引擎

- 前端交给我们html页面，一般是改成jsp，Ajax
- 通过表达式来进行数据传递
- MVVM
- th:text

为什么Thymeleaf能够加载templates里的html

```
private static final Charset DEFAULT_ENCODING = StandardCharsets.UTF_8;

public static final String DEFAULT_PREFIX = "classpath:/templates/";

public static final String DEFAULT_SUFFIX = ".html";
```

springboot中MVC的配置原理

```
@Configuration
public class MyMvcConfig implements WebMvcConfigurer {

}
```

- 通过方法重写来进行MVC的扩展

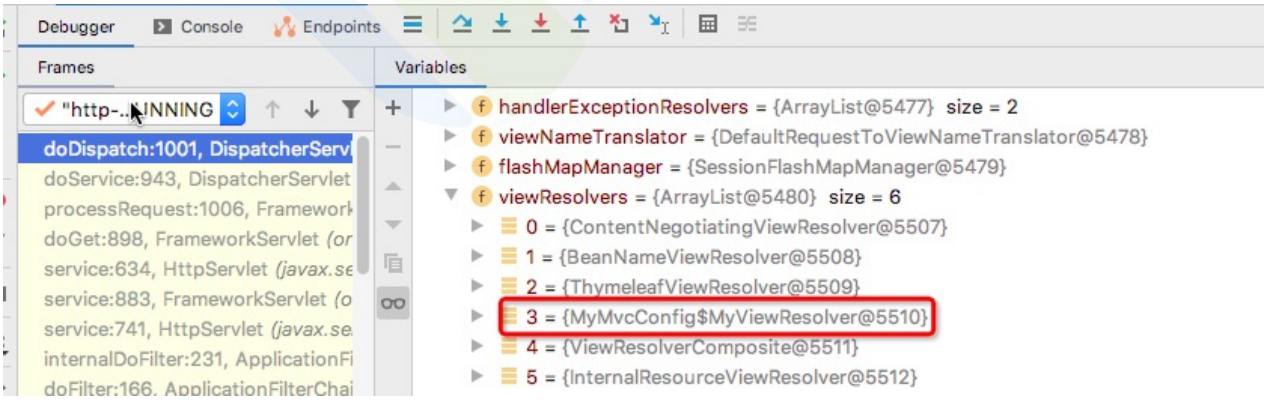
```
@Nullable
public View resolveViewName(String viewName, Locale locale) throws Exception {
    RequestAttributes attrs = RequestContextHolder.getRequestAttributes();
    Assert.state(attrs instanceof ServletRequestAttributes,  message: "No current ServletRequestAttributes");
    List<MediaType> requestedMediaTypes = this.getMediaTypes(((ServletRequestAttributes)attrs).getRequest());
    if (requestedMediaTypes != null) {
        List<View> candidateViews = this.getCandidateViews(viewName, locale, requestedMediaTypes);
        View bestView = this.getBestView(candidateViews, requestedMediaTypes, attrs);
        if (bestView != null) {
            return bestView;
        }
    }
```

获取候选视图，得到最好视图，其实就是从Bean里获取

定义一个自己的视图解析器，一定要注册到Bean里去

```
@Configuration
public class MyMvcConfig implements WebMvcConfigurer {

    @Bean
    public ViewResolver myViewResolver(){
        return new MyViewResolver();
    }
    //自己定义了一个视图解析器
    public static class MyViewResolver implements ViewResolver{
        @Override
        public View resolveViewName(String s, Locale locale) throws Exception
{
            return null;
        }
    }
}
```



总结：

- 定义一个自己的视图解析器，一定要注册到Bean里去
- @Configuration 这个要进行注解
- 要 implements WebMvcConfigurer

- @EnableWebMvc 这个千万不能加，加上springboot的自动autoMVC就失效了

为什么@EnableWebMvc这个加上就失效

- extends 这个类 WebMvcConfigurationSupport
- @ConditionalOnMissingBean(WebMvcConfigurationSupport.class)

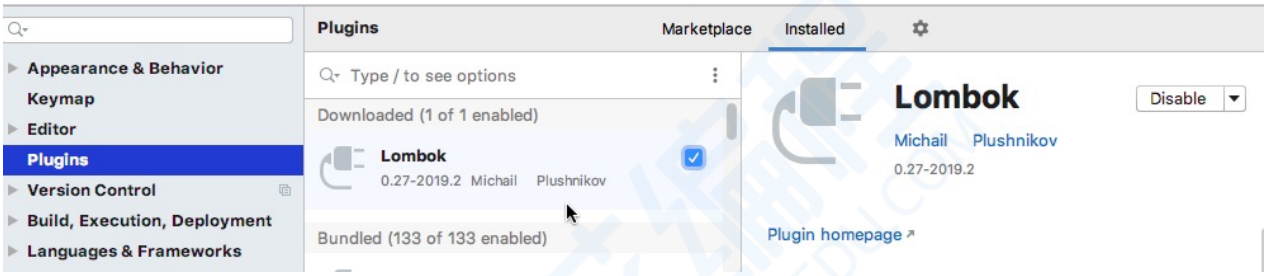
截止目前课程内容的学习

- springMVC能接管了
- 静态资源能导入了
- 首页也映射了，icon也知道咋弄了
- 基本的结构也清晰了

该进入项目工程的应用了

Lombok：通过注解的方式进行构建和getter、setter

step1: 要有plugin



step: 要有POM

```
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
</dependency>
```

国际化

- i18n，User Agent获得我们zh_CN、en_US，这个是在value上写配置
- DNS的方式来做，有一些地方要明确不翻译><