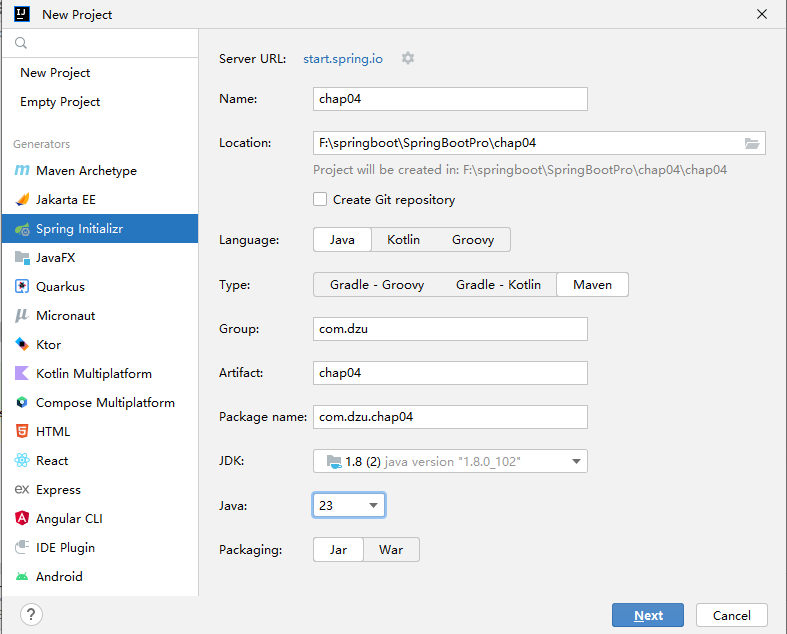
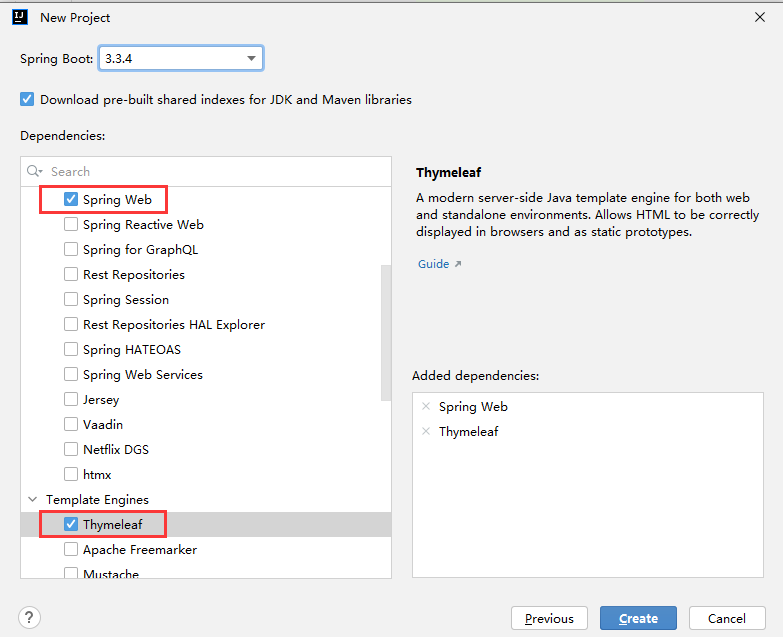
# 实验九 Spring Boot整合Thymeleaf

一、使用Spring Initializr方式创建项目

1. 创建项目



2.选择项目依赖



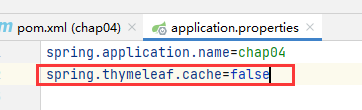
3.修改pom.xml

修改父工程版本和Java版本

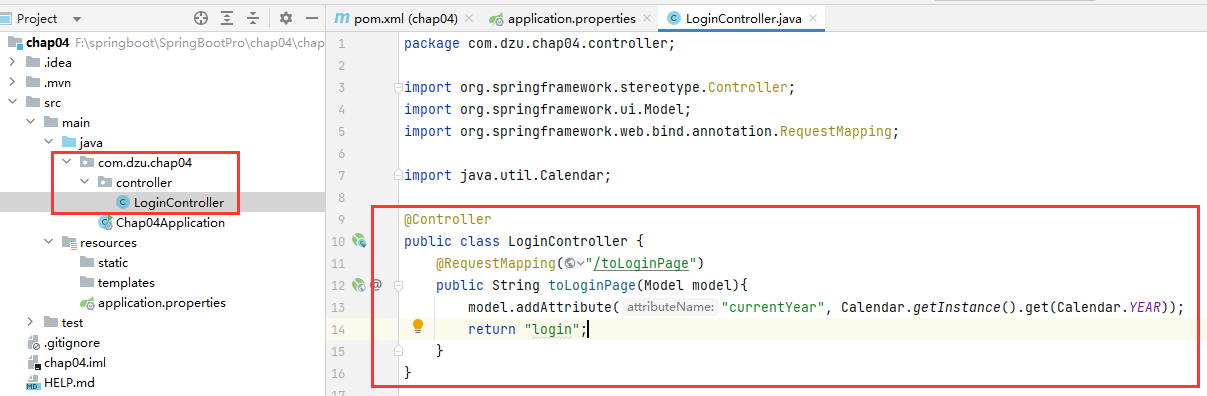
4.设置本地仓库

二、整合Thymeleaf

1.修改配置文件application.properties



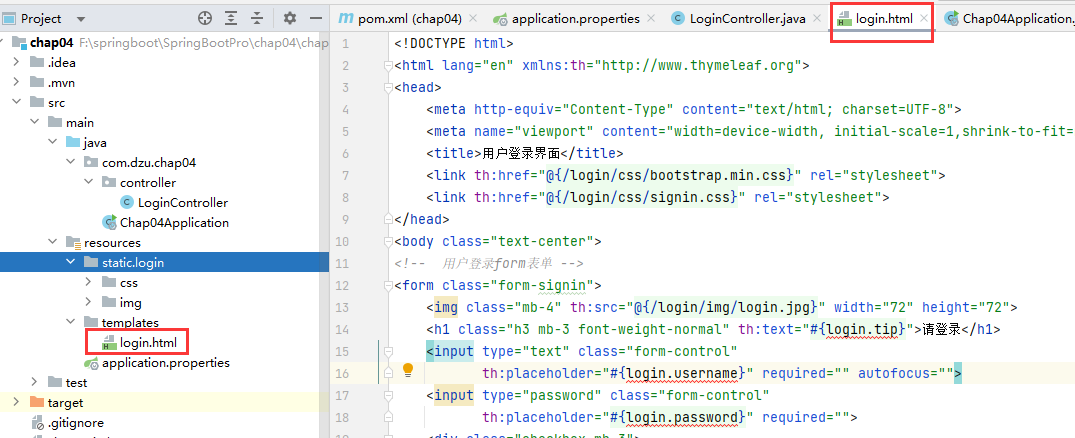
2.创建Web控制器类



@Controller  
public class LoginController {  
 @RequestMapping("/toLoginPage")  
 public String toLoginPage(Model model){  
 model.addAttribute("currentYear", Calendar.*getInstance*().get(Calendar.*YEAR*));  
 return "login";  
 }  
}

3.创建模板页面并引入静态资源文件

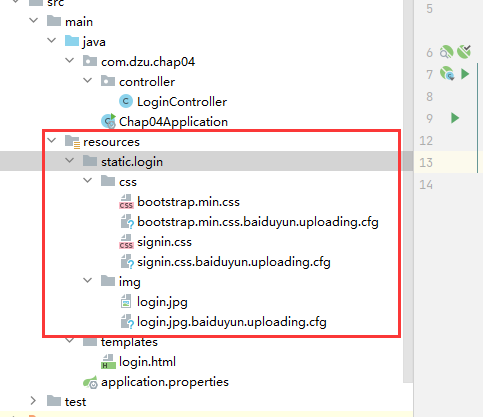
在resources-templates目录下创建login.html



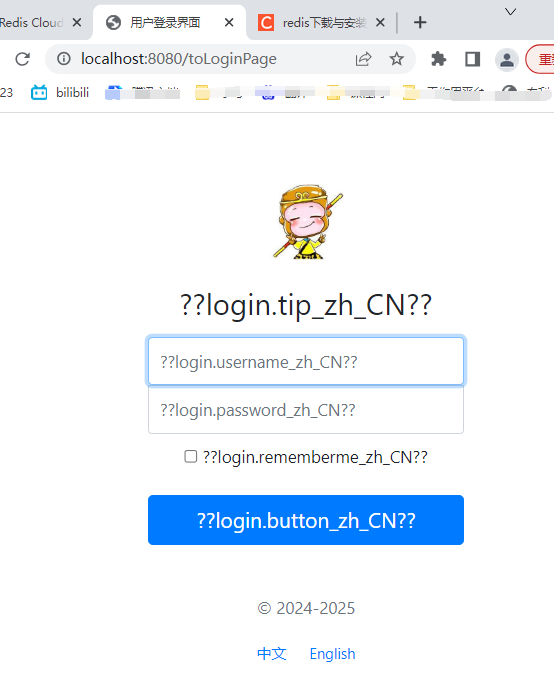


<!DOCTYPE html>  
<html lang="en" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1,shrink-to-fit=no">  
 <title>用户登录界面</title>  
 <link th:href="@{/login/css/bootstrap.min.css}" rel="stylesheet">  
 <link th:href="@{/login/css/signin.css}" rel="stylesheet">  
</head>  
<body class="text-center">  
*<!-- 用户登录form表单 -->*<form class="form-signin">  
 <img class="mb-4" th:src="@{/login/img/login.jpg}" width="72" height="72">  
 <h1 class="h3 mb-3 font-weight-normal" th:text="#{login.tip}">请登录</h1>  
 <input type="text" class="form-control"  
 th:placeholder="#{login.username}" required="" autofocus="">  
 <input type="password" class="form-control"  
 th:placeholder="#{login.password}" required="">  
 <div class="checkbox mb-3">  
 <label>  
 <input type="checkbox" value="remember-me"> [[#{login.rememberme}]]  
 </label>  
 </div>  
 <button class="btn btn-lg btn-primary btn-block" type="submit" th:text="#{login.button}">登录</button>  
 <p class="mt-5 mb-3 text-muted">© <span th:text="${currentYear}">2018</span>-<span th:text="${currentYear}+1">2019</span></p>  
 <a class="btn btn-sm" th:href="@{/toLoginPage(l='zh\_CN')}">中文</a>  
 <a class="btn btn-sm" th:href="@{/toLoginPage(l='en\_US')}">English</a>  
</form>  
</body>  
</html>

4.引入css文件和图片



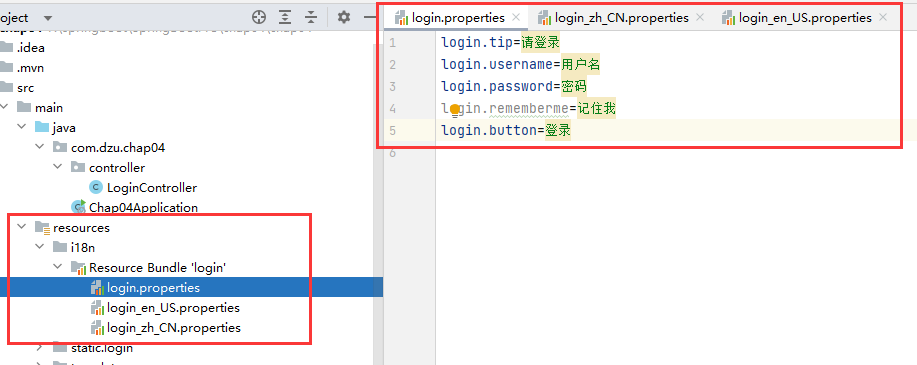
5.测试，运行Chap04Application

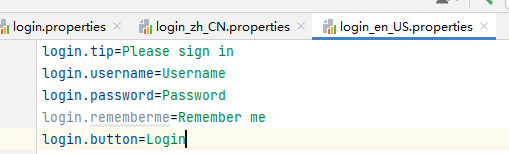
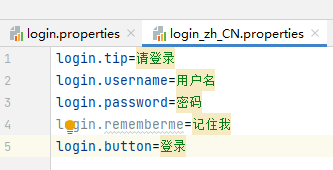


三、使用Thymeleaf配置国际化页面

1.编写多语言国际化文件及配置文件

在resources下创建文件夹i18n，在i18n中新建login.properties、 login\_en\_US.properties和login\_zh\_CN.properties

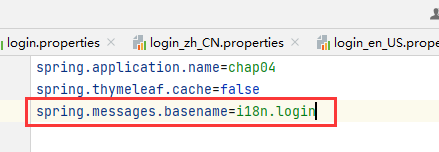




login.tip=请登录  
login.username=用户名  
login.password=密码  
login.rememberme=记住我  
login.button=登录

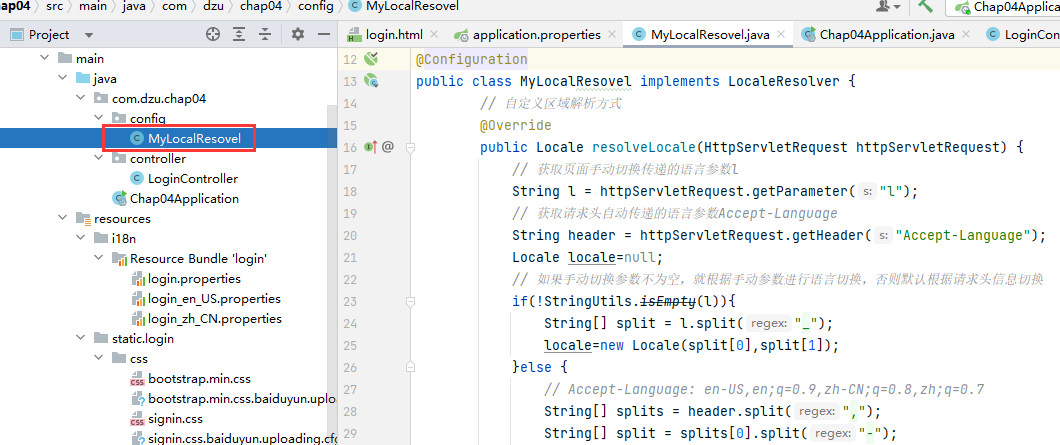
login.tip=Please sign in  
login.username=Username  
login.password=Password  
login.rememberme=Remember me  
login.button=Login

2.修改application.properties文件



spring.application.name=chap04  
spring.thymeleaf.cache=false  
spring.thymeleaf.encoding=utf-8  
spring.thymeleaf.mode=HTML5  
spring.thymeleaf.prefix=classpath:/templates/  
spring.thymeleaf.suffix=.html  
spring.messages.basename=i18n.login

3.定制区域信息解析器

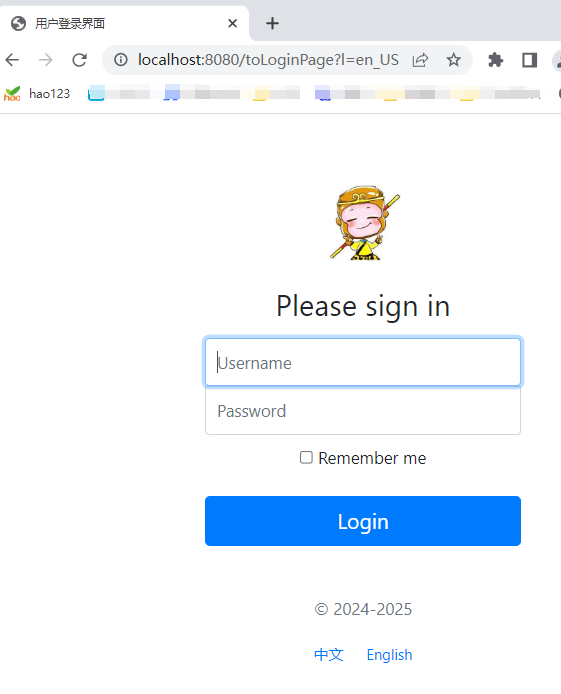


package com.dzu.chap04.config;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.lang.Nullable;  
import org.springframework.util.StringUtils;  
import org.springframework.web.servlet.LocaleResolver;  
  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.util.Locale;

@Configuration  
 public class MyLocalResovel implements LocaleResolver {  
 *// 自定义区域解析方式* @Override  
 public Locale resolveLocale(HttpServletRequest httpServletRequest) {  
 *// 获取页面手动切换传递的语言参数l* String l = httpServletRequest.getParameter("l");  
 *// 获取请求头自动传递的语言参数Accept-Language* String header = httpServletRequest.getHeader("Accept-Language");  
 Locale locale=null;  
 *// 如果手动切换参数不为空，就根据手动参数进行语言切换，否则默认根据请求头信息切换* if(!StringUtils.*isEmpty*(l)){  
 String[] split = l.split("\_");  
 locale=new Locale(split[0],split[1]);  
 }else {  
 *// Accept-Language: en-US,en;q=0.9,zh-CN;q=0.8,zh;q=0.7* String[] splits = header.split(",");  
 String[] split = splits[0].split("-");  
 locale=new Locale(split[0],split[1]);  
 }  
 return locale;  
 }  
 @Override  
 public void setLocale(HttpServletRequest httpServletRequest, @Nullable  
 HttpServletResponse httpServletResponse, @Nullable Locale locale) {  
 }  
 *// 将自定义的MyLocalResovel类重新注册为一个类型LocaleResolver的Bean组件* @Bean  
 public LocaleResolver localeResolver(){  
 return new MyLocalResovel();  
 }  
 }

4.页面国际化使用

5.测试



6.如果出现乱码，修改编码

