**中软国际考题-笔试**

1. 单选题
2. 在SpringMVC提供了一个语言区域解析器接口LocaleResolver,该接口常用类都在org.springframework.web.servlet.il8n包下，不存在（D）。

A．AcceptHeaderLocaleResolver

B. SessionLocaleResolver

C. CookieLocaleResolver

D. requestLocaleResolver

2、Spring 定义了 MessageSource 接口，用于访问国际化信息，以下哪个是正确的方法 ( B)

    A．get

   B. getMessage

C. getMess

D. Mess

3、<spring:message code=“key" /> （C）

A key 是request范围对象 B key 是session范围对象 C key 是属性文件的key D key 是cookie对象

4、Spring定义了访问国际化信息的接口为（B）

A.MessageLocale B.MessageSource C.MessageFormat D.MessageResource

5、spring哪一个用于国际化（B）

A.Info B.message C.bean D.config

6、ZH-CN 是(A)

A.中国 B.美国 C.澳大利亚

7、以下哪个标签支持国际化。（ ）

A． <s:message>

B． <input>

C． <s:radio>

D． <s:select>

1. 多选
2. SpringMVC中输出国际化消息的方式有（AB）

A．在视图页面上输出国际化消息

B．在Controller的处理方法中输出国际化消息

C．在spring配置文件中输出国际化消息

D．在web.xml配置文件中输出国际化消息

2、spring 国际化语言类型保存可以基于？(A B C )

A．session

B．cookie

C．url

D．request

3.国际化实现原理分为：（ A B C）

A本地化信息获取

B数据获取

C格式化

D 持久化

4、配置LocaleResolver（该类指明了Locale应该存放在何处），有哪三种选择？(ABC)

A.request B. session; C. cookie D.application

5、Java程序国际化主要是通过下面哪些类实现的（A、B、C）

A. java.util.ResourceBundle

B. java.util.Locale

C.java.text.MessageFormat

D.java.util.MessageFormat

E.java.util.NumberFormat

6、国际化支持的语言有（A、B,c,d）

A. 中文 B. 英语 C.简体

D. 都反持

7、ZH-CN和USA分别代表（AB）

A.中国

B.美国

C.印度

D.巴基斯坦

8、 以下哪些类型是MessageSource的子类型。?（ ）

A． Object

B． ApplicationContext

C． Session

D． HierarchicalMessageSource

1. 填空题
2. spring使用MessageSource接口实现国际化。两个实现类为：\_\_\_\_\_\_ResourceBundleMessageSource\_\_\_\_\_、\_\_\_\_\_\_ReloadableResourceBundleMessageSource\_\_\_\_\_。
3. 国际化简称 I18N
4. 配置LocaleResolver（该类指明了Locale应该存放在何处）有3种选择\_\_ request \_\_\_\_\_\_\_\_ 、\_\_ session \_\_\_\_\_\_\_、\_\_\_\_ cookie \_\_\_\_\_
5. spring mvc国际化需要引入标签库\_\_\_ http://www.springframework.org/tags \_\_\_\_\_\_
6. 国际化的属性文件的后缀名\_\_\_\_\_\_\_。
7. java.text.Format是一个用于格式化语言环境敏感的信息（如日期、消息和数字）的抽象基类，直接已知子类有\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 。（DateFormat, MessageFormat, NumberFormat）
8. 国际化的标签是\_\_\_\_\_\_\_\_\_\_ (message)
9. 国际化可以对应不同的 \_\_\_\_\_\_\_ (语言)
10. Spring 定义了 接口，用于访问国际化信息。（MessageSource）
11. i18n是英文单词 的首末字符i和n，18为中间的字符数，是“国际化”的简称。（internationalization）
12. spring中实现国际化，需根据不同语言添加多个 文件即可。（properties）
13. 判断题
14. 在SpringMVC中SessionLocaleResoler是默认的语言区域解析器（F）（不是默认的，需要配置）
15. SpringMVC中CookieLocaleResolver不是默认的语言区域解析器，需要显示对其进行配置（T）
16. SpringMVC中前端显示本地化消息通常使用Spring的message标签（T）
17. 国际化是通讯协议吗（ F ）
18. 国际化不用遵循标准( F )
19. 国际化的意思多个语言的页面的跳转（错）
20. spring mvc国际化需要引入标签库，使用标签（对）
21. java程序可以支持所有的国家和语言。（ F ）
22. java.util.ResourceBundle：用于加载国家、语言资源包。（ T ）
23. java.util.Locale：用于封装特定的国家|区域、语言环境。（T ）
24. java.text.MessageFormat：用于格式化带占位符的字符串。（ T ）
25. 在Spring中，MessageSource接口是从ApplicationContext接口继承的。（ F ）
26. HierarchicalMessageSource接口是Spring国际化中很重要的接口，最重要的两个实现类是 ResourceBundleMessageSource 和ReloadableResourceBundleMessageSource。（ T ）
27. ResourceBundleMessageSource允许用户通过beanName指定一个资源名（包括类路径的全限定资源名），或通过beanNames指定一组资源名。（ T ）
28. ResourceBundleMessageSource可以定时刷新资源文件，以便在应用程序不重启的情况下感知资源文件的变化。（ F ）
29. StaticMessageSource主要用于程序测试，它允许通过编程的方式提供国际化信息。（ T ）
30. 国际化的目的不是为了支持多语言。（ F ）
31. 国际化的思想是编程吗（ F）
32. Spring中不支持国际化，不同语言版本程序必须单独开发。（ F ）
33. Spring中如果想实现中文和英文两种语言显示，属性文件添加messages\_zh\_CN.properties、messages\_en\_US.properties两个即可。（T ）

五、简答题

# 1、简单介绍一下spring使用MessageSource接口实现国际化的两个实现类为?

答案：ResourceBundleMessageSource：基于java的ResourceBundle实现了国际化，配置文件必须放在classpath下。  
ReloadableResourceBundleMessageSource：直接使用读取文件的方式实现国际化，规则跟java的相同，支持动态修改后刷新配置，避免在业务不能中断的情况下重启进程。配置文件可以放在任意目录下，指定目录后，该类会去指定目录中加载配置文件。

2、SpringMVC如何实现国际化/多语言？

（1） 创建SpringMVC项目

配置web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd"

version="3.1">

<!--Spring 配置文件-->

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>/WEB-INF/applicationContext.xml</param-value>

</context-param>

<!--监听器-->

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

<!--配置转发器-->

<servlet>

<servlet-name>dispatcher</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>dispatcher</servlet-name>

<url-pattern>\*.action</url-pattern>

</servlet-mapping>

<filter>

<filter-name>characterEncodingFilter</filter-name>

<filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>

<init-param>

<param-name>encoding</param-name>

<param-value>UTF-8</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>characterEncodingFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

</web-app>

配置Spring文件(dispatcher-servlet.xml)

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:mvc="http://www.springframework.org/schema/mvc"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd">

<!-- 默认使用基于注释的适配器和映射器 -->

<mvc:annotation-driven/>

<!-- 只把动态信息当做controller处理，忽略静态信息 -->

<mvc:default-servlet-handler/>

<!-- 自动扫描包中的Controlller -->

<context:component-scan base-package="controller"/>

<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<property name="viewClass" value="org.springframework.web.servlet.view.JstlView"/>

<property name="prefix" value="/WEB-INF/jsp/"/><!-- 前缀 -->

<property name="suffix" value=".jsp"/><!-- 后缀，自动拼接 -->

</bean>

</beans>

（2）添加多语言的配置文件

添加language\_en\_US.properties到src目录下

language.cn = \u4e2d\u6587

language.en = English

internationalisation = \u0020Internationalisation

welcome = This is the English environment

introduce= This is I18N Demo

添加language\_zh\_CN.properties到src目录下

language.cn = \u4e2d\u6587

language.en = English

internationalisation = \u56fd\u9645\u5316

welcome = \u8fd9\u662f\u4e2d\u6587\u73af\u5883

introduce= \u8fd9\u662f\u56fd\u9645\u5316\u7684\u4e8b\u4f8b

（3） 加入i18n 过滤器到配置文件中

将配置添加到dispatcher-servelet中

<!-- 国际化资源文件 -->

<bean id="messageSource" class="org.springframework.context.support.ReloadableResourceBundleMessageSource">

<!-- 表示多语言配置文件在根路径下，以language开头的文件-->

<property name="basename" value="classpath:language"/>

<property name="useCodeAsDefaultMessage" value="true"/>

</bean>

<mvc:interceptors>

<bean id="localeChangeInterceptor" class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">

<property name="paramName" value="lang"/>

</bean>

</mvc:interceptors>

（5） 在页面中使用多语言

在Controller中添加路径

@Controller

public class HelloController {

@RequestMapping("/hello.action")

public String index() {

return "hello";

}

}

在JSP页面中使用

通过<spring:message code="welcome"/>将配置文件中的内容读取

<%@ page language="java" contentType="text/html; charset=UTF-8"%>

<%@taglib prefix="spring" uri="http://www.springframework.org/tags" %>

<html>

<head>

<title>SpringMVC<spring:message code="internationalisation"/></title>

</head>

<body>

Language:

<a href="?lang=zh\_CN"><spring:message code="language.cn"/></a>

<a href="?lang=en\_US"><spring:message code="language.en"/></a>

<h1>

<spring:message code="welcome"/>

</h1>

当前语言: ${pageContext.response.locale }

</body>

</html>

3、国际化基本规则？

答案：国际化信息”也称为“本地化信息”，一般需要两个条件才可以确定一个特定类型的本地化信息，它们分别是“语言类型”和“国家/地区的类型”。如中文本地化信息既有中国大陆地区的中文，又有中国台湾、中国香港地区的中文，还有新加坡地区的中文。Java通过java.util.Locale类表示一个本地化对象，它允许通过语言参数和国家/地区参数创建一个确定的本地化对象。

语言参数使用ISO标准语言代码表示，这些代码是由ISO-639标准定义的，每一种语言由两个小写字母表示。在许多网站上都可以找到这些代码的完整列表，下面的网址是提供了标准语言代码的信息：http://www.loc.gov/standards/iso639-2/php/English\_list.php。

国家/地区参数也由标准的ISO国家/地区代码表示，这些代码是由ISO-3166标准定义的，每个国家/地区由两个大写字母表示。用户可以从以下网址查看ISO-3166的标准代码：http://www.iso.ch/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html

4、spring mvc国际化的如何实现的？

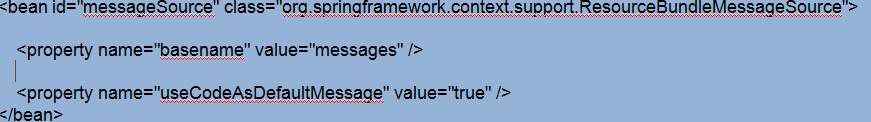
5、spring mvc国际化如何在配置文件中配置属性文件？

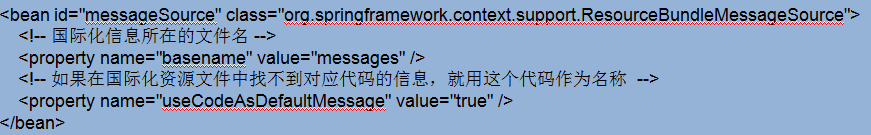
6、spring mvc属性文件可以防在任何包下面吗？

7、<spring:message code=“key" />属性文件如果没有key会不会报错？

8、配置登录页面的国际化

（1）、以下配置文件中value="messages"代表的意思?

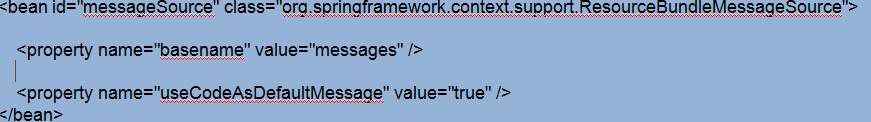




（2）、后台代码如何通过RequestContext 对象的getMessage方法获取国际化信息

RequestContext requestContext = new RequestContext(request);

String msg = requestContext.getMessage(“key”);

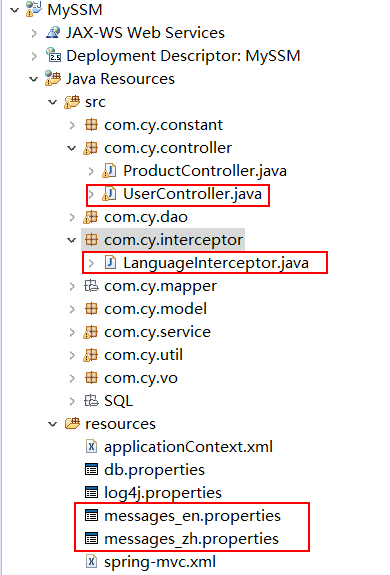
（3）、下面的图中<property name="useCodeAsDefaultMessage" value="true" /> 是什么意思？ 、

9、输入url：展示中文界面

http://localhost:8080/MySSM/user?lang=zh

输入url:  展示英文界面

http://localhost:8080/MySSM/user?lang=en



10、简述使用SpringMVC进行国际化的步骤（5分）

答案：

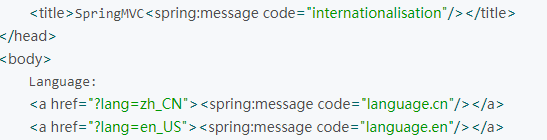
* （1）. 添加资源文件，如language\_en\_US.properties和language\_zh\_CN.properties到src目录下。
* （2）在资源文件中，分别配置中文和英文的输出内容
* （3）. 加入i18n拦截器到配置文件中



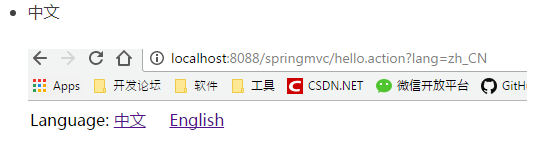
（4）.在JSP页面中配置输出标签

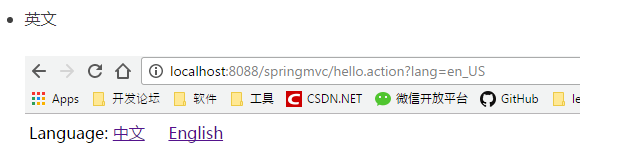
<%@taglib prefix="spring" uri="http://www.springframework.org/tags" %>

* （5）.在jsp页面进行调用



（6）. 使用者切换中文或英文





11、请简单描述Java程序国际化的思路。

答案：

Java程序国际化思路是将程序中的标签、提示等信息放在资源文件中，程序需要支持哪些国家、语言环境，就对应提供哪些相应的资源文件。资源文件就是key-value，每个资源文件的key可以保持不变的，但value则虽不同的国家、语言而改变。

12、Spring国际化时出现org.springframework.context.**NoSuchMessageException**: No message found under ...，可能是什么原因？

答案：

多数情况下是以下3种原因造成的：

第1种：配置 ResourceBundleMessageSource 类的bean 时id 没有设成"messageSource"

第2种：.properties 文件路径配置出错。

第3种：.properties 文件中确实没有定义所要查找的模式字符串pattern 对应的key。

13、解释下国际化？

答：程序中的国际化问题也是一个非常重要的部分，好多的软件都支持国际化，不同的版本对应不同的语言。在进行国际化时，依照程序开发习惯，我们一般选择在页面层或者控制层做国际化处理

14.标签国际化是？

答: 标签国际化我们可以用jstl的fmt方式进行国际化，也可以采用spring自己提供的标签<spring:message/>进行国际化。但尽量还是采用jstl的fmt方式，因为spring自带的标签在找不到国际化值时，整个系统会抛异常，完全不能运行，这也是spring标签的鸡肋。

15.i18n是

答：i18n(其来源是英文单词 internationalization的首末字符i和n,18为中间的字符数)是“国际化”的简称。

16、1、简述软件中国际化的概念？

答案：

软件国际化（Software Internationalization），是全球化的产物。是在软件设计和文档开发过程中，使得功能和代码设计能处理多种语言和文化习俗，在创建不同语言版本时，不需要重新设计源程序代码的软件工程方法。

1. 简述spring中实现国际化的步骤？
2. 基于不同的语言定义多个properties文件，用于后面根据本地化信息从相应的properties文件中获取数据

例如:中英文属性文件

https://img-blog.csdn.net/20180526110541378?watermark/2/text/aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L3FxXzQxNTQxNjE5/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70

1. 编辑Spring配置文件，在原有基础上添加国际化资源文件和本地化信息

|  |
| --- |
| <!-- 获取数据 -->  <bean id="messageSource"  class="org.springframework.context.support.ReloadableResourceBundleMessageSource">  <!-- 表示多语言配置文件在根路径下，以language开头的文件 -->  <property name="basename" value="classpath:language" />  <property name="useCodeAsDefaultMessage" value="true" />  </bean>  <!-- 获取本地化信息 -->  <mvc:interceptors>  <bean id="localeChangeInterceptor"  class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">  <property name="paramName" value="lang" />  </bean>  </mvc:interceptors>  <bean id="localeResolver" class="org.springframework.web.servlet.i18n.CookieLocaleResolver" /> |

六、程序实现题

1、有国际化资源文件zh\_CN，内容为language.cn = \u4e2d\u6587 language.en = \u82f1\u6587 ；Spring配置文件的核心配置代码编写。

|  |
| --- |
| <!-- 存储区域设置信息 -->  <bean id="localeResolver"  class="org.springframework.web.servlet.i18n.SessionLocaleResolver" >  <property name="defaultLocale" value="zh\_CN"/>  </bean>  <!-- 国际化资源文件 -->  <bean id="messageSource"  class="org.springframework.context.support.ReloadableResourceBundleMessageSource">  <property name="basename" value="classpath:messages" /><!--messages与资源文件名字的前缀相对应 -->    </bean>    <mvc:interceptors><!--设置拦截器 -->  <bean id="localeChangeInterceptor"  class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">  <property name="paramName" value="lang" /><!-- lang是url请求时对应的参数-->  </bean>  </mvc:interceptors> |

2、请编写在SpringMVC中实现国际化的核心配置和代码。

|  |
| --- |
| 1. 基于不同的语言定义多个properties文件，用于后面根据本地化信息从相应的properties文件中获取数据。在这里实现的是简体中文和英文的切换，所以定义了两个properties文件，这两个文件均在根目录下  language\_en\_US.properties：  language.cn = \u4e2d\u6587  language.en = English  internationalisation = \u0020Internationalisation  welcome = This is the English environment  introduce= This is I18N Demo  language\_zh\_CN.properties：  language.cn = \u4e2d\u6587  language.en = English  internationalisation = \u56fd\u9645\u5316  welcome = \u8fd9\u662f\u4e2d\u6587\u73af\u5883  introduce= \u8fd9\u662f\u56fd\u9645\u5316\u7684\u4e8b\u4f8b  2.编辑Spring配置文件，在原有基础上添加国际化资源文件和本地化信息  <!-- 获取数据 -->  <bean id="messageSource" class="org.springframework.context.support.ReloadableResourceBundleMessageSource">  <!-- 表示多语言配置文件在根路径下，以language开头的文件 -->  <property name="basename" value="classpath:language" />  <property name="useCodeAsDefaultMessage" value="true" />  </bean>  <!-- 获取本地化信息 -->  <mvc:interceptors>  <bean id="localeChangeInterceptor" class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">  <property name="paramName" value="lang" />  </bean>  </mvc:interceptors>  <bean id="localeResolver" class="org.springframework.web.servlet.i18n.CookieLocaleResolver"  />  3.测试  （1）新建一个controller，用于跳转到指定的国际化测试页面  @Controller  public class Test1 {  @RequestMapping("/hello.action")  public String index() {  System.out.println("进入hello.action");  return "content/hello.jsp";  }  }  （2）新建hello.jsp页面  <%@ page language="java" contentType="text/html; charset=UTF-8"%>  <%@taglib prefix="spring" uri="http://www.springframework.org/tags" %>  <html>  <head>  <title>SpringMVC<spring:message code="internationalisation"/></title>  </head>  <body>  Language:  <a href="?lang=zh\_CN"><spring:message code="language.cn"/></a>  <a href="?lang=en\_US"><spring:message code="language.en"/></a>  <h1>  <spring:message code="welcome"/>  </h1>  当前语言: ${pageContext.response.locale }  </body>  </html> |

3、i18n配置文件是什么:

答：  <!-- 国际化资源文件 -->

<bean id="messageSource" class="org.springframework.context.support.ReloadableResourceBundleMessageSource">

<!-- 表示多语言配置文件在根路径下，以language开头的文件-->

<property name="basename" value="classpath:language"/>

<property name="useCodeAsDefaultMessage" value="true"/>

</bean>

<mvc:interceptors>

<bean id="localeChangeInterceptor" class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">

<property name="paramName" value="lang"/>

</bean>

</mvc:interceptors>

4、

1、实现界面的中英文切换

（1）编写中文、英文属性文件

https://img-blog.csdn.net/20171019220712808?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvbGR3MjAxNTEwODAzMDA2/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

（2）xml文件配置：

|  |
| --- |
| <!-- 开启注解的功能 加载数据验证配置 、配置国际化的资源、Spring mvc 读取国际化资源-->  <mvc:annotation-driven validator="validator" conversion-service="conversion-service"/>  <bean id="conversion-service" class="org.springframework.format.support.FormattingConversionServiceFactoryBean"/>  <bean id="messageSource" class="org.springframework.context.support.ResourceBundleMessageSource">  <property name="basenames" value="message"/>  </bean>  <bean id="localeResolver" class="org.springframework.web.servlet.i18n.SessionLocaleResolver"/>  <mvc:interceptors>  <bean class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor"/> <!-- 需放在拦截器中 -->  </mvc:interceptors>  <bean id="validator" class="org.springframework.validation.beanvalidation.LocalValidatorFactoryBean">  <property name="providerClass" value="org.hibernate.validator.HibernateValidator"/>  <property name="validationMessageSource" ref="messageSource"></property>  </bean> |

1. 页面

|  |
| --- |
| <body>  <a href="${pageContext.request.contextPath}/login.do?new&locale=zh\_CN">中文</a>  <a href="${pageContext.request.contextPath}/login.do?new&locale=en\_us">英文</a>  </body> |