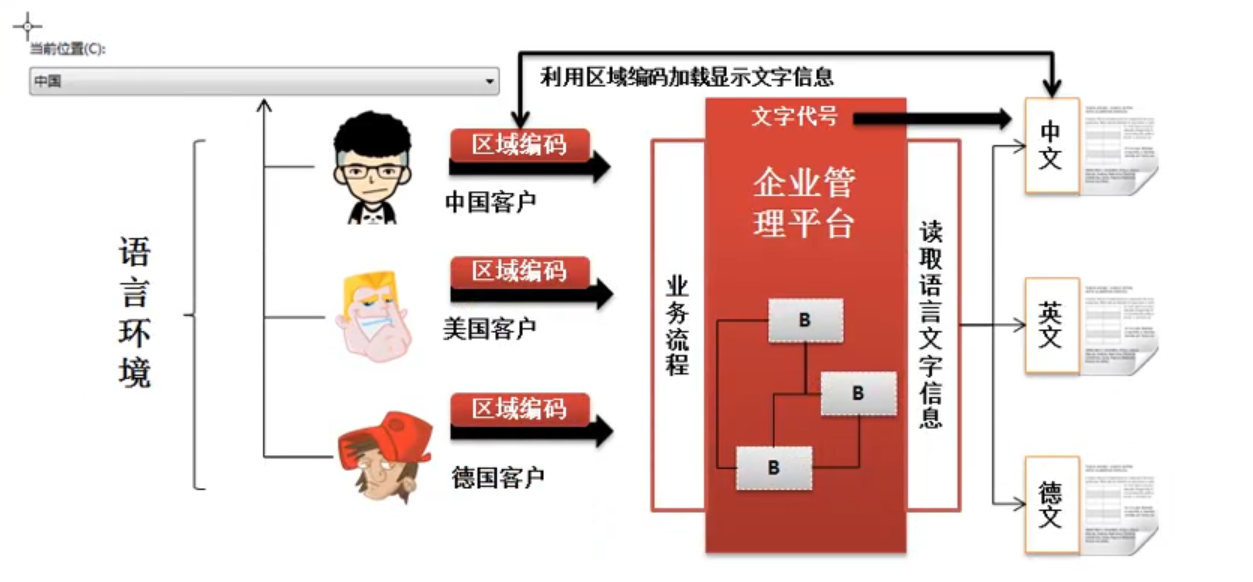
1. 国际化程序的实现
   1. 所谓国际化的程序是指同一段代码可以根据不同的国家实现不同的语言描述，但是程序的核心业务是相同的。

**·国际化问题简介**

现在假设有一款世界都认可的企业管理，那么这个企业的老板决定将这个产品推广到世界的各个上市公司：美国，中国，德国，那么在这种情况下，首先要考虑的问题是说明呢?



通过分析之后我们可以发现，如果想要解决国际化的程序开发，那么要解决的问题就在于以下两点:

·如何可以定义保存文字的文件信息；

·如果可以根据不同的区域语言的编码读取指定的资源信息。

1. Locale类
   1. 通过分析可以发现，如果要想实现国际化，首先要解决的就是不同国家用户的区域和语言编码问题。而在java.util包里面提供有一个专门描述区域和语言的类Locale，而后主要可以使用Locale中的两个构造方法实例化；

·构造方法:public Locale(String language);

·构造方法:public Locale(String language,String country);

此时需要的是国家和语言的代码，而中文的是zh-CN、美国的是en-US。

范例:实例化Locale对象

|  |
| --- |
|  |

如果我们想要自动获取当前环境，可以使用Locale类自身获取默认环境的方法:

·获取默认环境:public static Locale getDefault();

Locale loc = new Locale.getDefault

Locale类中也给我们设置了许多国家地区的常量；

·如Locale loc = new Locale.CHINA;(可以避免区域编码信息的繁琐)

1. 读取资源文件: ResourceBundle

范例:定义一个资源cn.mldn.message.Message.properties配置文件

|  |
| --- |
| Info=欢迎小强同学！ |

资源中主要是一个key=value的结果，程序通过key来获取value的数据。

* 1. 现在已经准备好了资源文件，那么随后就需要进行资源文件的读取操作了，而读取资源文件主要依靠的是java.util.ResourceBundle类完成。

|  |
| --- |
| Public abstract class ResourceBundle extends Object ; |

从上面我们可以发现，ResourceBundle是一个抽象类，如果说现在想要进行此类对象的实例化可以直接利用该类中的一个static方法。

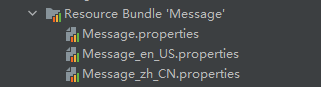
·public static final ResourceBundle getBundle(String baseName);

-baseName描述的是资源文件的名称但是没有后缀：

如: cn.mldn.message.Message

·根据key读取资源内容；public static String getString(String key );

1. 实现国际化程序开发
   1. 现在国际化程序的实现前期准备已经全部完成了，也就是说依靠资源文件、Locale、ResourceBundle类就可以实现国际化的操作（国际化程序的操作关键点就在于读取资源信息）;
   2. 范例:
      1. 首先定义三个资源文件



* + 1. 通过程序进行指定资源的加载

|  |
| --- |
| package com.company.国际化程序实现;  import java.util.ResourceBundle;  /\*\*  \* @author john  \*/  public class InternationalizationProgram {  public static void main(String[] args) {  ResourceBundle resourceBundle = ResourceBundle.getBundle("com.message.Message");  String val = resourceBundle.getString("info");  System.out.println(val);  }  } |

此时我们发现当我们并没有指定明确的Locale时，Message\_zh\_CN起作用了，因为这个方法默认加载的是本地的资源。

当然我们也可以使用Resource类中的方法来指定Locale

public static final ResourceBundle getBundle(String baseName ,Locale locale);

范例:指定locale

|  |
| --- |
| package com.company.国际化程序实现;  import java.util.Locale;  import java.util.ResourceBundle;  /\*\*  \* @author john  \*/  public class InternationalizationProgram {  public static void main(String[] args) {  Locale loc = new Locale("en","US");  ResourceBundle resourceBundle = ResourceBundle.getBundle("com.message.Message", loc);  String val = resourceBundle.getString("info");  System.out.println(val);  }  } |

如果指定的loc没有资源会读取默认的，如local.GERMAN,因为没有德国区域的资源文件，因此会根据getDefault()读取默认地区的资源，如果默认资源 也没有那么会读取没有区域编码的资源。

1. 格式化文本显示
   1. 修改资源内容（Message.properties）

|  |
| --- |
| info={0},Welcom to back,{1}! |

此时如果进行资源读取会将占位符的信息一起读取出来，所以我们需要利用MessageFormat类进行格式处理

代码:

|  |
| --- |
| package com.company.国际化程序实现;  import java.text.MessageFormat;  import java.text.SimpleDateFormat;  import java.util.Date;  import java.util.Locale;  import java.util.ResourceBundle;  /\*\*  \* @author john  \*/  public class InternationalizationProgram {  public static void main(String[] args) {  Locale loc = new Locale("en","US");  ResourceBundle resourceBundle = ResourceBundle.getBundle("com.message.Message", loc);  String val = resourceBundle.getString("info");  System.out.println(MessageFormat.format(val,"易文奇",  new SimpleDateFormat("YYYY-MM-dd").format(new Date())));  }  } |

如果在资源文件中看到了{0}，{1}等都是占位符号，一定要进行格式化处理。