高级文本处理

第一节 Java字符编码

一.字符编码

-ASCII码，采用1Byte，8bits，最多256个字符。

-Big5 ASCII+繁体中文

-GB8030>GBK>GB2312

-Unicode(字符集)

  -目标：不断扩充，存储全世界所有的字符。

  -编码方案

    -UTF-8，兼容ASCII，变长，经济，方便运输。

    -UTF-16，UTF-32....

-ANSI编码

 -在Windows上非Unicode的默认编码

二.Java字符编码

源文件编码：采用UTF-8编码，和外界（文本文件）输入输出尽量采用UTF-8编码。

第二节 国际化编程

一.什么是国际化编程

  -多语言版本的软件

  -Java是第一个设计成支持国际化的编程语言

-java.util.ResourceBundle用于加载一个语言_国家语言包

-java.util.Locale定义一个语言_国家

```
import java.util.Locale;
import java.util.ResourceBundle;

public class NewHelloWorld {
public static void main(String[] args) {
    // 取得系统默认的国家、语言环境
    Locale myLocale = Locale.getDefault();
```

```
        System.out.println(myLocale); //zh_CN

        // 根据指定语言_国家环境加载资源文件
        ResourceBundle bundle = ResourceBundle.getBundle("message", myLocale);

        // 从资源文件中取得的信息
        System.out.println(bundle.getString("hello"));

        myLocale = new Locale("en", "US"); //语言_国家 强制转换成en_US
        bundle = ResourceBundle.getBundle("message", myLocale);
        System.out.println(bundle.getString("hello"));  //Hello World

    }
```

}

二.Local类（zh_CN,en_US...）

Local方法

-getAvailableLocales（）返回所有的可用Local

-getDefault（）返回默认的Locale

语言文件

-包含K-V对，每行一个K-V，例如：age=20

-命名规则：

-包名——语言+国家地区.properties（语言和国家能选）

-message_zh_CN.propertities

-制作方法，采用native2ascii.exe进行转码（mac未知怎么转码。。。）

三.ResourceBundle

-根据Local要求，加载语言文件（Properties文件）

-存储语言集合中所有的K-V对

-getString（String ket返回相对应的value）

-ResourceBundle根据key寻找value路径

四.其他国家化

日期/时间国际化

数字/金额国际化

第三节 Java高级字符串处理

正则表达式

-用事先定义好的一些特定字符及这些特定字符的组合，组成一个“规则字符串”.

-java.util.regex包

-Pattern正则表达式的编译表示

  -compile编译一个正则表达式为Pattern对象

  -matcher用pattern对象匹配一个字符串，返回匹配结果。

-Matcher

  -Index Methods(位置方法) //start()，start(int group)，end()，end(int group)

  -Study Methods(查找方法) //lookingAt(),end(),find(),find(int start),matches()

  -Replacement Methods(替换方法) //replaceAll(String replacement)

```
package regex;

import java.util.regex.Pattern;
import java.util.regex.Matcher;

public class MatcherDemo {
private static final String REGEX = "\\bdog\\b";  //\\b表示边界
private static final String INPUT = "dog dog dog doggie dogg";

public static void main(String[] args) {
    //检查字符串里有多少个dog
    Pattern p = Pattern.compile(REGEX);

    Matcher m = p.matcher(INPUT);
    int count = 0;
    while (m.find()) {
        count++;
        System.out.println("Match number " + count);
        System.out.println("start(): " + m.start());
        System.out.println("end(): " + m.end());
    }
```