

```
非贷款,0元入学,不1万就业不给1分钱学费,我们已干四年了!
```

笔记总链接: http://bbs.itheima.com/thread-200600-1-1.html 8. IO

8.1 其他对象API

System类中的字段和方法都是静态的。 常见方法:

示例: public class SystemDemo{ 01. 02.

8.1.1 System类

public static void main(String[] args){ long 11 = System.currentTimeMillis(); 03. System.out.println(11); 04.

long currentTimeMillis();获取当前时间的毫秒值,可以通过此方法检测程序的执行时间。

06. code... 07.

08. long 12 = System.currentTimeMillis(); 09. System.out.println(12-11); 10. } 11. } 复制代码

Properties getProperties();确定当前的系统属性。 Properties集合中存储的都是String类型的键和值。 最好使用它自己的存储和取出的方法来完成元素的操作。

示例1: 01. import java.util.Properties; import java.util.Set; 02. 03. public class SystemDemo{ 04. public static void main(String[] args){ 05. //获取系统的属性信息,并存储到了Properties集合中 06. 07. Properties prop = System.getProperties();

Set<String> nameSet = prop.stringPropertyNames(); 10. 11. for(String name : nameSet){ String value = prop.getProperty(name); 12. System.out.println(name + " = " + value); 13. 14. }

15. } 16. 复制代码 运行结果: 📷 管理员: C:\Windows\system32\cmd.exe D:\code\day20>javac SystemDemo.java D:\code\day20>java SystemDemo

_ _ _ X sun.boot.library.path = D:\Java\jdk1.6.0_21\jre\bin java.vm.version = 17.0-b16 java.vm.vendor = Sun Microsystems Inc. java.vendor.url = http://java.sun.com/ path.separator = ; file.encoding.pkg = sun.io user.country = CN sun.java.launcher = SUN_STANDARD sun.os.patch.level = Service Pack 1

Windows系统中换行为\r\n两个转义字符, Linux只有一个\n。

Ξ

示例2: import java.util.Properties; 01. 02. import java.util.Set; 03. public class SystemDemo{ 04. public static void main(String[] args){ 06. System.out.println("hello-\r\n world"); 07. final String LINE_SEPARATOR = System.getProperty("line.separator"); 08. System.out.println("hello" + LINE_SEPARATOR + "word"); 09. } 10.

11. } 复制代码 运行结果: _ D X ■ 管理员: C:\Windows\system32\cmd.exe D:∖code∖day20>javac SystemDemo.java D:∖code∖day2Ø>java SystemDemo world he 11o word

给属性设置一些属性信息,这些信息是全局的,其他程序都可以使用。例:

System.setPeroperty("myclasspath","c:\myclass");。

P.S.

P.S.

08.

09.

}

复制代码

📷 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day20>java RuntimeDemo

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

public static void main(String[] args) throws Exception{

Runtime r = Runtime.getRuntime();

🧻 RuntimeDemo.java - 记事本

pow(a,b):a的b次方。

import java.util.Random;

public class MathDemo{

public static void main(String[] args){

double d1 = Math.ceil(12.56);

double d2 = Math.floor(12.56);

double d3 = Math.round(12.56);

System.out.println(d1);

System.out.println(d2);

示例1:

01.

02.

03.

04.

05.

06.

07.

08.

09.

10.

26.

27.

28.

12.0 13.0 100.0 d5 = 8.0d5 = 7.0d5 = 10.0

d5

d5

d6

d6

d6

3.0

= 2.0

= 3.0

1.0

2.0

}

运行结果:

复制代码

}

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day2Ø>java MathDemo

8.1.4 Date、DateFormat类

因为可以通过具体的数值进行运算。

将日期对象-->日期格式的字符串。

使用的是DateFormat类中的format方法。

import java.text.DateFormat;

import java.util.Date;

public class DateDemo{

import java.text.SimpleDateFormat;

public static void main(String[] args){

String str_date1 = df.format(date);

String str_date2 = df.format(date);

//如果风格是自定义的如何解决呢?

String str_date3 = df.format(date);

下午01时54分57秒

System.out.println(str_date3);

df = new SimpleDateFormat("yyyy--MM--dd");

System.out.println(str_date2);

System.out.println(str_date1);

//获取日期格式对象,具备着默认的风格。也可以指定为FULL、LONG风格。

df = DateFormat.getDateTimeInstance(DateFormat.FULL,DateFormat.LONG);

_ 0 X

- - X

DateFormat df = DateFormat.getDateInstance(DateFormat.FULL);

Date date = new Date();

对日期对象进行格式化:

示例2:

01.

02.

03.

04.

05.

06.

07.

08.

09.

11.

12.

13.

14.

15.

16.

17.

18.

19.

20.

21.

22.

23.

运行结果:

2015年6月7日 2015年6月7日

2015--06--07

示例3:

}

复制代码

画 管理员: C:\Windows\system32\cmd.exe

D:\code\day20>java DateDemo

D:∖code∖day20>javac DateDemo.java

星期日 星期日

将日期格式的字符串-->日期对象。

画 管理员: C:\Windows\system32\cmd.exe

D:\code\day20>java DateDemo Thu Apr 19 00:00:00 CST 2012 Thu Apr 19 00:00:00 CST 2012

练习:

思路:

两个日期相减就可以了。

必须有两个可以进行减法运算的数。

1. 将日期格式的字符串转成Date对象。

import java.util.Date;

public class Test{

}

import java.text.DateFormat;

import java.text.SimpleDateFormat;

public static void main(String[] args) throws Exception {

public static void test(String str_date1,String str_date2) throws Exception {

DateFormat dateFormat = DateFormat.getDateInstance();

dateFormat = new SimpleDateFormat("yyyy-MM-dd");

Date date1 = dateFormat.parse(str_date1);

Date date2 = dateFormat.parse(str_date2);

String str_date1 = "2014-3-17";

String str_date2 = "2014-4-6";

//1、将日期字符串转成日期对象

long time1 = date1.getTime();

long time2 = date2.getTime();

System.out.println(time);

int day = getDay(time);

System.out.println(day);

long time = Math.abs(time2-time1);

test(str_date1,str_date2);

2. 将Date对象转成毫秒值。

3. 相减,编程天数。

代码:

01.

02.

03.

04.

05.

06.

07.

09.

10.

11.

12.

13.

14.

15.

16.

17.

18.

19.

20.

21.

22.

23.

24.

25.

26.

27.

28.

29.

示例一:

03.

04. 05.

06.

07.

10.

11.

12. 13.

14.

15.

16.

17.

18.

19.

20.

12.

13.

14.

15.

16.

17.

18.

19.

20.

21.

22.

23.

24.

25.

}

期六"};

}

复制代码

2012年4月20日星期五

打印每年2月有多少天

D:∖code∖day20>javac CalendarDemo.java

D:∖code∖day2Ø>java CalendarDemo

import java.util.Calendar;

public class CalendarDemo{

}

运算结果:

练习:

代码:

01.

02.

03.

04.

import java.util.Calendar;

public class CalendarDemo{

}

getWeek(week));

}

public static void main(String[] args){

public static void showDate(Calendar c){

public static String getWeek(int i){

int year = c.get(Calendar.YEAR);

int month = c.get(Calendar.MONTH) + 1;

int day = c.get(Calendar.DAY_OF_MONTH);

int week = c.get(Calendar.DAY_OF_WEEK);

System.out.println(year + "年" + month + "月" + day + "日" +

String[] weeks = { "","星期日" ,"星期一" ,"星期二" ,"星期三" ,"星期

Calendar c = Calendar.getInstance();

//GregorianCalendar

showDate(c);

}

D:∖code∖day20>javac DateDemo.java

"2012-3-17"到"2012-4-6"中间有多少天?

能减可以是毫秒值,如何获取毫秒值?通过date对象。

如何获取date对象呢?可以将字符串转成date对象。

使用的是DateFormat类中的prase方法。

D:\code\day20>javac MathDemo.java

public class RuntimeDemo{

D:∖code∖day20>javac RuntimeDemo.java

}

运行结果:

8.1.2 Runtime类

能创建自己的 Runtime 类实例。

Runtime:没有构造方法摘要,说明该类不可以创建对象。又发现还有非静态的方法,说明该类应该 提供静态的返回该类对象的方法。而且只有一个,说明Runtime类使用了单例设计模式。 示例1: public class RuntimeDemo{ 01. public static void main(String[] args) throws Exception{ Runtime r = Runtime.getRuntime(); 03. 04. Process $p = r.exec("notepad.exe D:\code\day20\RuntimeDemo.java");$ Thread.sleep(5000); 06. p.destroy(); 07.

- - X

- - X

II

每个 Java 应用程序都有一个 Runtime 类实例, 使应用程序能够与其运行的环境相连接。应用程序不

Thread.sleep(5000); p.destroy(); 8.1.3 Math类 Math:提供了操作数学运算的方法,都是静态的。 常用方法: ceil():返回大于参数的最小整数。 floor():返回小于参数的最大整数。 round():返回四舍五入的整数。

 $Process\ p = r.exec(\ "notepad.exe\ D:\\code\\day20\\RuntimeDemo.java");$

System.out.println(d3); 11. 12. 13. double d4 = Math.pow(10,2); System.out.println(d4); 14. 15. 16. Random r = new Random();17. for(int i = 0; i < 5; i++){ 18. double d5 = Math.ceil(Math.random() * 10); 19. System.out.println("d5 = " + d5); 20. 21. } 22. for(int i = 0; i < 5; i++){ 23. 24. double d6 = (int)(r.nextDouble() * 6 + 1);25. System.out.println("d6 = " + d6);

示例1: import java.util.Date; 01. 02. public class DateDemo{ 03. public static void main(String[] args){ long time = System.currentTimeMillis(); 05. System.out.println(time); 06. 08. //将当前日期和时间封装成Date对象 09. Date date1 = new Date(); System.out.println(date1); 10. 11. //将指定毫秒值封装成Date对象 12. Date date2 = new Date(1405244787235I); 13. System.out.println(date2); 14. 15. 16. } 复制代码 运行结果: ■ 管理员: C:\Windows\system32\cmd.exe D:\code\day20>javac DateDemo.java D:∖code∖day20>java DateDemo 1433656405298 Sun Jun 07 13:53:25 CST 2015 Sun Jul 13 17:46:27 CST 2014 日期对象和毫秒值之间的转换 毫秒值-->日期对象: 1. 通过Date对象的构造方法 new Date(timeMillis); 2. 还可以通过setTime设置。 因为可以通过Date对象的方法对该日期中的各个字段(年月日等)进行操作。 日期对象-->毫秒值: 1. getTime方法。

import java.text.DateFormat; 01. import java.text.SimpleDateFormat; 02. 03. import java.util.Date; 04. public class DateDemo{ 05. public static void main(String[] args) throws Exception { 06. String str_date1 = "2012年4月19日"; 07. String str_date2 = "2012--4--19"; 08. 09. DateFormat dateFormat = DateFormat.getDateInstance(DateFormat.LONG); 10. 11. Date date1 = dateFormat.parse(str_date1); System.out.println(date1); 12. 13. dateFormat= new SimpleDateFormat("yyyy--MM--dd"); Date date2 = dateFormat.parse(str_date2); 15. System.out.println(date2); 16. 17. } 18. } 复制代码 运行结果:

private static int getDay(long time){ 30. int day = (int)(time/1000/60/60/24); 31. 32. return day; 33. } 34. } 复制代码 运行结果: 0 ■ 管理员: C:\Windows\system32\cmd.exe D:∖code∖day20>javac Test.java D:∖code∖day20>java Test 1728000000 20 8.1.5 Calendar类 Calendar 类是一个抽象类,它为特定瞬间与一组诸如YEAR、MONTH、DAY_OF_MONTH、HOUR 等日历字段之间的转换提供了一些方法,并为操作日历字段(例如获得下星期的日期)提供了一些方法。

四","星期五","星期六"}; 22. return weeks[i]; 23. } 24. 复制代码 运行结果: 0 ■ 管理员: C:\Windows\system32\cmd.exe D:∖code∖day20>javac CalendarDemo.java D: \code\day20>java CalendarDemo 2015年6月7日星期日 示例二: import java.util.Calendar; 01. 02. public class CalendarDemo{ 04. public static void main(String[] args){ 05. Calendar c = Calendar.getInstance(); 07. c.set(2014,3,20); c.add(Calendar.YEAR,-2); 08. showDate(c); } 10. 11.

public static void showDate(Calendar c){

int year = c.get(Calendar.YEAR);

int month = c.get(Calendar.MONTH) + 1;

int day = c.get(Calendar.DAY_OF_MONTH);

int week = c.get(Calendar.DAY_OF_WEEK); System.out.println(year + "年" + month + "月" + day + "日" + getWeek(week)); public static String getWeek(int i){ String[] weeks = { "","星期日" ,"星期一" ,"星期二" ,"星期三" ,"星期四" ,"星期五" ,"星 return weeks[i]; _ 0 X ■ 管理员: C:\Windows\system32\cmd.exe

```
字符流的抽象基类:Reader, Writer。
 P.S.
 由这四个类派生出来的子类名称都是以其父类名作为子类名的后缀。
 如:InputStream的子类FileInputStream。
 如:Reader的子类FileReader。
 需求:将一些文字存储到硬盘一个文件中。
 记住:如果要操作文字数据,建议优先考虑字符流。
 而且要将数据从内存写到硬盘上,要使用字符流中的输出流:Writer。
 硬盘的数据基本体现是文件,希望找到一个可以操作文件的Writer: FileWriter。
 示例1:
      import java.io.FileWriter;
 01.
      import java.io.IOException;
 02.
 03.
      public class FileWriterDemo{
 04.
 05.
          public static void main(String[] args) throws IOException{
             //创建一个可以往文件中写入字符数据的字符输出流对象
 06.
 07.
             //既然是往一个文件中写入文字数据,那么在创建对象时,就必须明确该文件(用于
      存储数据的目的地)
             //如果文件不存在,则会自动创建
 08.
             //如果文件存在,则会被覆盖
 09.
            FileWriter fw = new FileWriter("demo.txt");
 10.
 11.
             //调用Writer对象中的write(string)方法,写入数据
 12.
             //其实数据写入到临时存储缓冲区中
 13.
            fw.write( "abcde");
 14.
 15.
             //进行刷新,将数据直接写入到目的地中
 16.
            fw.flush();
 17.
 18.
             //关闭流,关闭资源,在关闭前会先调用flush刷新缓冲中的数据到目的地。
 19.
            fw.close();
 20.
 21.
         }
 22.
      复制代码
 运行结果:
                                                                 _ D X
■ 管理员: C:\Windows\system32\cmd.exe
D:∖code∖day20>javac FileWriterDemo.java
                                                                           =
 D:\code\day20>java FileWriterDemo
          🅌 ▶ 计算机 ▶ 本地磁盘 (D:) ▶ code ▶ day20
                                                                       44
 文件(F)
        编辑(E) 查看(V) 工具(T) 帮助(H)
   组织▼
           ___ 打开
                    新建文件夹
                       名称
                                                       修改日期
                                                                      类型
   ☆ 收藏夹
     ▶ 下载
                       demo.txt
                                                       2015/6/7 14:11
                                                                      文本文档
                                                                 🧻 demo.txt - 记事本
 文件(F) 編辑(E) 格式(O) 查看(V) 帮助(H)
 abcde
 P.S.
 1. close方法只能用一次。
 2. 流关闭以后不能,不能再调用write方法,否则会报如下异常错误:
                                                                _ 0 X
■ 管理员: C:\Windows\system32\cmd.exe
D:∖code\day20>javac FileWriterDemo.java
                                                                          П
D:\code\day2Ø>java FileWriterDemo
Exception in thread "main" java.io.IOException: Stream closed
        at sun.nio.cs.StreamEncoder.ensureOpen(StreamEncoder.java:26)
        at sun.nio.cs.StreamEncoder.write(StreamEncoder.java:99)
        at sun.nio.cs.StreamEncoder.write(StreamEncoder.java:116)
        at java.io.OutputStreamWriter.write(OutputStreamWriter.java:203)
        at java.io.Writer.write(Writer.java:140)
        at FileWriterDemo.main(FileWriterDemo.java:22)
 如果构造函数中加入true,可以实现对文件进行续写。
 示例2:
      import java.io.FileWriter;
 01.
 02.
      import java.io.IOException;
 03.
      public class FileWriterDemo{
 04.
          public static void main(String[] args) throws IOException{
 05.
 06.
            FileWriter fw = new FileWriter("demo.txt", true);
 07.
            fw.write( "xixi");
 08.
            fw.close();
 09.
 10.
         }
 11.
      复制代码
 运行结果:
                                                                 - - X
■ 管理员: C:\Windows\system32\cmd.exe
D:∖code∖day20>javac FileWriterDemo.java
D:\code\day20>java FileWriterDemo
                                                                 🧻 demo.txt - 记事本
 文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
 abcdexixi
 IO流的异常处理方式:为防止代码异常导致流无法关闭,因此在finally中对流进行关闭。
 示例3:
      import java.io.FileWriter;
 01.
      import java.io.IOException;
 02.
 03.
 04.
      public class IOExceptionDemo{
            public static void main(String[] args){
 06.
                 FileWriter fw = null;
 07.
 08.
 09.
                  try{
                       fw = new FileWriter("demo.txt" );
 10.
 11.
 12.
                       fw.write( "abcde");
 13.
                 } catch(IOException e){
                       System.out.println(e.toString());
 14.
                 } finally{
 15.
 16.
                       try{
                            fw.close();
 17.
                       } catch(IOException e){
 18.
                            throw new RuntimeException("关闭失败");
 19.
 20.
                       }
 21.
                 }
 22.
            }
      }
 23.
      复制代码
 运行结果:
                                                                 - - X
■ 管理员: C:\Windows\system32\cmd.exe
D:\code\day20>javac IOExceptionDemo.java
D:\code\day2Ø>java IOExceptionDemo
                                                                 - - X
🧻 demo.txt - 记事本
 文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
 abcde
 需求:读取一个文文件,将读取到的字符打印到控制台。(使用FileReader)
 第一种读取方式:使用read()方法读取文本文件数据。
 示例4:
      import java.io.FileReader;
 01.
 02.
      import java.io.IOException;
 03.
      public class FileReaderDemo{
 04.
 05.
            public static void main(String[] args) throws IOException{
 06.
                 FileReader fr = new FileReader("demo.txt" );
 07.
 08.
                  //用Reader中的read方法读取字符
 09.
                  int ch = 0;
 10.
 11.
                  while((ch = fr.read()) != -1){
 12.
                      System.out.println(( char)ch);
 13.
 14.
                 }
 15.
                 fr.close();
 16.
 17.
            }
 18.
      }
      复制代码
 运行结果:
                                                                 - - X
画 管理员: C:\Windows\system32\cmd.exe
D:\code\day20>javac FileReaderDemo.java
D:∖code∖day20>java FileReaderDemo
 说明:
read
public int read()
      throws IOException
    读取单个字符。在字符可用、发生 1/0 错误或者已到达流的末尾前,此方法一直阻塞。
    用于支持高效的单字符输入的子类应重写此方法。
    返回:
        作为整数读取的字符,范围在 O 到 65535 之间(OxOO-Oxffff),如果已到达流的末尾,则返回 -1
    抛出:
        IOException - 如果发生 I/O 错误
 第二种读取方式:使用read(char[])方法读取文本文件数据。
 示例5:
      import java.io.FileReader;
 01.
 02.
      import java.io.IOException;
 03.
      public class FileReaderDemo{
            public static void main(String[] args)throws IOException{
 05.
                 FileReader fr = new FileReader("demo.txt" );
 06.
 07.
                  //使用read(char[])读取文本文件数据
 08.
                  //先创建字符数组
                  char[] buf = new char[3];
 10.
 11.
 12.
                  int len = 0;
 13.
                  while((len = fr.read(buf)) != -1){
 14.
                       System.out.println( new String(buf,0,len));
 15.
 16.
                 }
 17.
                 fr.close();
 18.
 19.
           }
 20.
      }
      复制代码
 运行结果:
                                                                 _ 0 X
画 管理员: C:\Windows\system32\cmd.exe
D:\code\day20>javac FileReaderDemo.java
D:\code\day20>java FileReaderDemo
abc
 de
                                 ~END~
     ... The AC the
                               ~爱上海,爱黑马~
```

05.

06.

07.

08.

09.

10.

11.

12.

13.

14.

15.

16.

17.

18.

19.

20.

21.

22.

23.

24.

25.

27.

28.

29.

30.

31.

}

少天

}

}

期六"};

}

复制代码

■ 管理员: C:\Windows\system32\cmd.exe

D:\code\day20>java CalendarDemo 2014年2月28日星期五

输入流和输出流相对于内存设备而言。

将外设中的数据读取到内存中:输入。

流按操作数据分为两种:字节流与字符流。

这个文字进行操作。简单说:字节流+编码表。

字节流的抽象基类:InputStream,OutputStream。

8.2.2 IO流常用基类-字符流

将内存的数写入到外设中:输出。

字符流的由来:

D:\code\day20>javac CalendarDemo.java

}

运行结果:

8.2.1 IO流

IO包中。

public static void main(String[] args){

public static void showDays(int year){

Calendar c = Calendar.getInstance();

c.add(Calendar.DAY_OF_MONTH,-1);

public static void showDate(Calendar c){

int year = c.get(Calendar.YEAR);

public static String getWeek(int i){

return weeks[i];

int month = c.get(Calendar.MONTH) + 1;

int day = c.get(Calendar.DAY_OF_MONTH);

int week = c.get(Calendar.DAY_OF_WEEK);

//将日期设置为3月1日,然后减一天,打印出2月份的最后一天日期,即可知2月有多

System.out.println(year + "年" + month + "月" + day + "日" + getWeek(week));

String[] weeks = { "","星期日" ,"星期一" ,"星期二" ,"星期三" ,"星期四" ,"星期五" ,"星

8.2 IO流

IO流用来处理设备之间的数据传输。Java对数据的操作是通过流的方式。Java用于操作流的对象都在

其实就是:字节流读取文字字节数据后,不直接操作而是先查指定的编码表,获取对应的文字。再对

- - X

int year = 2014;

showDays(year);

c.set(year,2,1);

showDate(c);