7. Java常用类

Contents

- 基本数据类型的包装类
- 字符串相关类
  - 不可变字符序列：String
  - 可变字符序列：StringBuffer、StringBuilder
- 时间处理相关类
  - Date
  - DateFormat、SimpleDateFormat
  - Calendar
- 枚举类
- Math类和Random类
- File类

本章技能点列表

技能点名称	难易程度	认知程度	重要程度
包装类	易	理解	**
自动装箱和自动拆箱	易	理解	**
字符串相关类String	中	应用	***
StringBuffe和StringBuilder	中	应用	***
Date类	易	应用	**
DateFormat类	中	应用	***
Calendar类	难	了解	*
枚举	易	了解	*
Math类和Random类	易	应用	**
File类	中	应用	**

基本数据类型的包装类 A

- 为什么需要 包装类（Wrapper Class）？
  - JAVA并不是纯面向对象的语言。Java语言是一个面向对象的语言，但是Java中的基本数据类型却是不面向对象的。但是我们在实际使用中经常需要将基本数据转化成对象，便于操作。比如：集合的操作中。 这时，我们就需要将基本类型数据转化成对象！
- 包装类均位于java.lang包，包装类和基本数据类型的对应关系：

基本数据类型	包装类
byte	Byte
boolean	Boolean
short	Short
char	Character
int	Integer
long	Long
float	Float
double	Double

如何使用包装类?

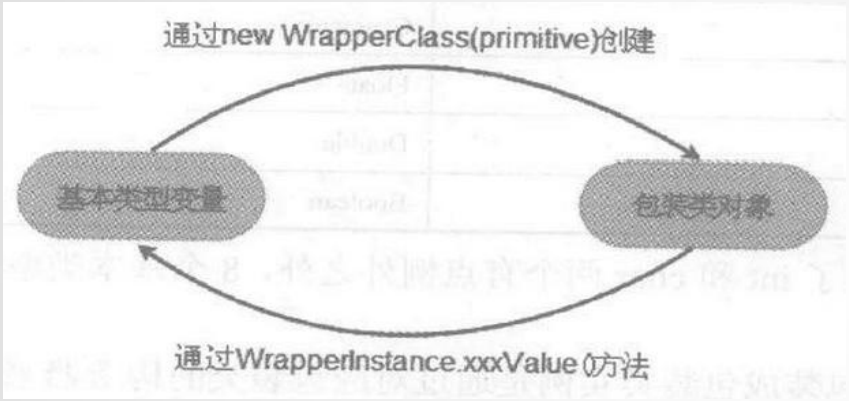
- 包装类的作用：
  - 提供：字符串、基本类型数据、对象之间互相转化的方式！
  - 包含每种基本数据类型的相关属性如最大值、最小值等

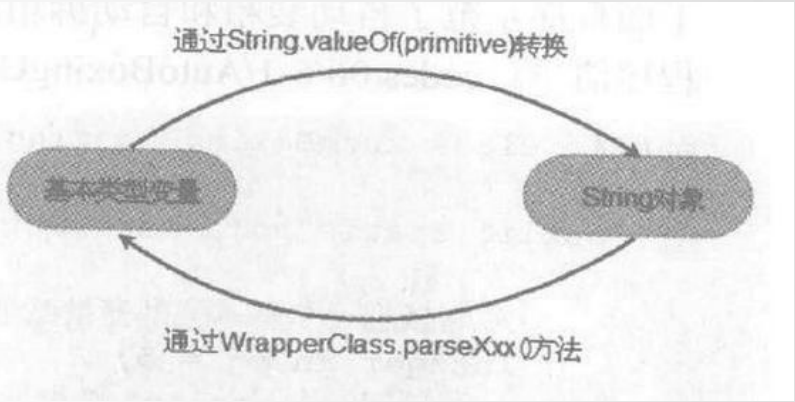
- 所有的包装类(Wrapper Class)都有类似的方法，掌握一个其他都类似！以Integer为例！

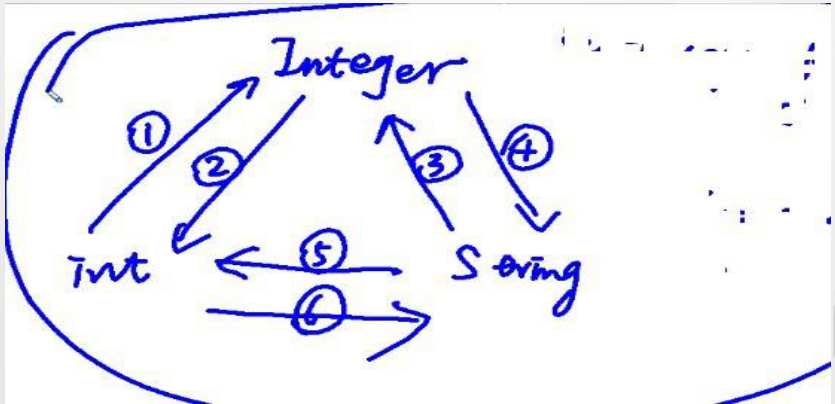
- 代码：TestWrapperClass1.java

如何使用包装类?

- 包装类的基本操作（3类操作）







自动装箱和自动拆箱

- 自动装箱-boxing
  - 基本类型就自动地封装到与它相同类型的包装中，如：
  - Integer i = 100;
  - 本质上是，编译器编译时为我们添加了：
  - Integer i = Integer.valueOf(100);
- 自动拆箱autounboxing
  - 包装类对象自动转换成基本类型数据。如：
  - int a = new Integer(100);
  - 本质上，编译器编译时为我们添加了：
  - int a = new Integer(100).intValue();

课堂练习(15分钟)

- 通过熟悉Integer的用法。同学们自学一下：
- Double，或者其他包装类中的任意一种！

String(不可变字符序列)

- Java字符串就是Unicode字符序列，例如串“Java”就是4个Unicode字符J,a,v,a组成的。

- Java字符串就是Unicode字符序列，例如串“Java”就是4个Unicode字符J,a,v,a组成的。

- Java允许使用符号"+"把两个字符串连接起来
  - String s1 = “Hello”;String s2 = “World!”;
  - String s = s1 + s2; //HelloWorld!

String类的常用方法（1）

- char charAt(int index)
  - 返回字符串中第index个字符。
- boolean equals(String other)
  - 如果字符串与other相等，返回true
- boolean equalsIgnoreCase(String other)
  - 如果字符串与other相等（忽略大小写），则返回true
- int indexOf(String str)　lastIndexOf()
- int length()
  - 返回字符串的长度。
- String replace(char oldChar,char newChar)
  - 返回一个新串，它是通过用 newChar 替换此字符串中出现的所有oldChar而生成的
- 代码：StringTest1.java

String类的常用方法（2）

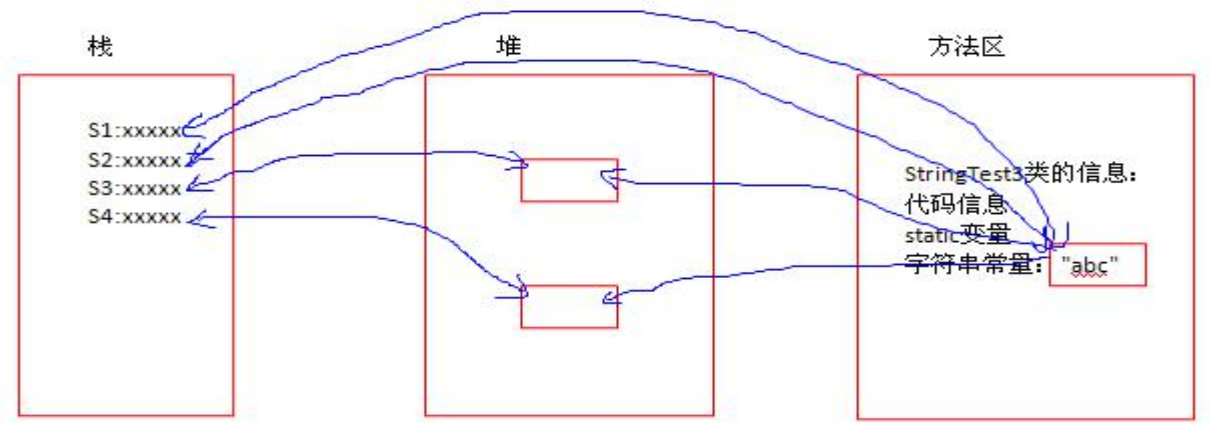
- boolean startsWith(String prefix)
  - 如果字符串以prefix开始，则返回true
- boolean endsWith(String prefix)
  - 如果字符串以prefix结尾，则返回true
- String substring(int beginIndex)
- String substring(int beginIndex,int endIndex)
  - 返回一个新字符串，该串包含从原始字符串beginIndex到串尾或endIndex-1的所有字符
- String toLowerCase()
  - 返回一个新字符串，该串将原始字符串中的所有大写字母改成小写字母
- String toUpperCase()
  - 返回一个新字符串，该串将原始字符串中的所有小写字母改成大写字母
- String trim()
  - 返回一个新字符串，该串删除了原始字符串头部和尾部的空格

字符串相等的判断(一般使用equals方法)

- equals判断字符串值相等，==判断字符串对象引用相等！

```
public class StringTest3 {
        public static void main(String[] args) {
                String s1 = "abc";
                String s2 = "abc";
                String s3 = new String("abc");
                String s4 = new String("abc");
                System.out.println(s1==s2);    //true
                System.out.println(s1==s3);    //false
                System.out.println(s3==s4);    //false
        }
}
```

内存结构图



StringBuffer和StringBuilder

- StringBuffer和StringBuilder非常类似，均代表可变的字符序列，而且方法也一样
- 查看API文档了解相关常用方法：
  - append/delete/deleteCharAt/insert/inverse

- 代码：TestStringBuilder.java

字符串选用

- String：不可变字符序列
- StringBuilder：可变字符序列、效率高、线程不安全
- StringBuilder：可变字符序列、效率低、线程安全

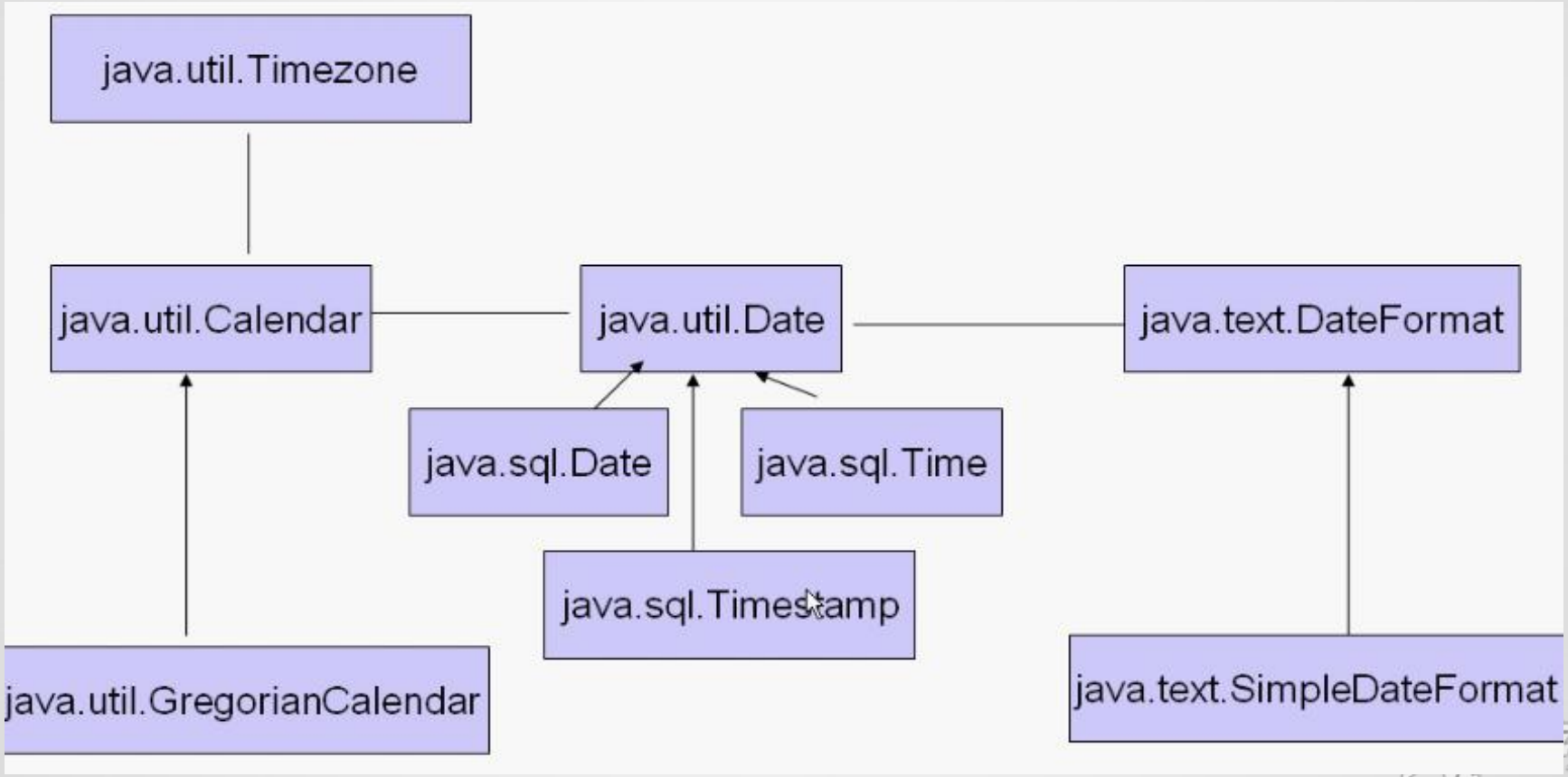
- String使用陷阱：
  - string s="a"; //创建了一个字符串

    s=s+"b"; //实际上原来的"a"字符串对象已经丢弃了，现在又产生了一个字符串s+"b"。如果多次执行这些改变串内容的操作，会导致大量副本字符串对象存留在内存中，降低效率。如果这样的操作放到循环中，会极大影响程序的性能。
- 代码：TestStringBuilder2.java

课堂练习(15分钟)

- 熟悉String用法
- 熟悉API文档查看方法
- 熟悉StringBuilder、StringBuffer用法

时间处理相关类


java.util.Timezone
java.util.Calendar
java.util.Date
java.text.DateFormat
java.sql.Date
java.sql.Time
java.sql.Timestamp
java.util.GregorianCalendar
java.text.SimpleDateFormat

Date时间类(java.util.Date)

- 在标准Java类库中包含一个Date类。它的对象表示一个特定的瞬间，精确到毫秒。
- Java中时间的表示说白了也是数字，是从：标准纪元1970.1.1 0点开始到某个时刻的毫秒数，类型是long。

- 代码：TestDate.java

DateFormat和SimpleDateFormat

- 完成字符串和时间对象的转化！
- format
- parse

- 代码：TestSimpleFormat.java

Calendar日历类

- 人们对于时间的认识是：某年某月某日，这样的日期概念。计算机是long类型的数字。通过Calendar在二者之间搭起桥梁！

Calendar类

表示时间是日期概念，
某年某月某日

表示时间是long类型的数字

GregorianCalendar公历

- GregorianCalendar 是 Calendar 的一个具体子类，提供了世界上大多数国家/地区使用的标准日历系统。
- 注意：
  - 月份：一月是0，二月是1，以此类推，是12月是11
  - 星期：周日是1，周一是2，。。。周六是7

- 代码：TestCalendar.java

课堂作业(15分钟)

- 熟悉时间处理相关类的用法以及他们之间的关系!

Math类

- 包含了常见的数学运算函数。

- random()　生成[0,1)之间的随机浮点数
- 生成：0-10之间的任意整数：
  - int　a =　(int)(10*Math.random());
- 生成：20-30之间的任意整数：
  - int　b = 20 + (int)(10*Math.random());

课堂练习(15分钟)

- 编写程序，利用GregorianCalendar类，打印当前月份的日历，样式如下：
- 今天的日期是 1988-02-23 ,如下为今日所在月份的日历。

```
请输入日期：（年月日，格式：2008-9-20）：
2010-1-1
日      一      二      三      四      五      六
                                        1*      2
3       4       5       6       7       8       9
10      11      12      13      14      15      16
17      18      19      20      21      22      23
24      25      26      27      28      29      30
31
```

- 代码：TestVisualCalendar.java

枚举

- 只能够取特定值中的一个
- 使用enum关键字
- 所有的枚举类型隐性地继承自 java.lang.Enum。（枚举实质上还是类！而每个被枚举的成员实质就是一个枚举类型的实例，他们默认都是public static final的。可以直接通过枚举类型名直接使用它们。）
- 强烈建议当你需要定义一组常量时，使用枚举类型
- 尽量不要使用枚举的高级特性，事实上高级特性都可以使用普通类来实现，没有必要引入复杂性！

- 示例代码：TestEnum.java

File类

- 文件和目录路径名的抽象表示形式。一个File对象可以代表一个文件或目录

- 可以实现获取文件和目录属性等功能
- 可以实现对文件和目录的创建、删除等功能

- File不能访问文件内容

```
File  file = new File("d:\\test\\java.txt");
File  file = new File("d:/test/java.txt");
File  file = new File("java.txt");
```

路径可以是绝对路径和相对路径，分隔符采用\\或者/

File类

- 通过File对象可以访问文件的属性。

  public String getName()　　public String getPath()

  public boolean isFile()　　public boolean isDirectory()

  public boolean canRead()　　public boolean canWrite()

  public boolean exists()　　public long length()

  public boolean isHidden()　　public long lastModified()

  public File [] listFiles();

- 通过File对象创建空文件或目录（在该对象所指的文件或目录不存在的情况下）。
  - public boolean createNewFile()throws IOException
  - public boolean delete()
  - public boolean mkdir()， mkdirs()　注意两个的区别！！