字符串解析、日期格式化

武永亮

讲授思路

- 字符串
- 日期、日期的格式化、以及字符串的解析

讲授思路-字符串

- 字符串的基本概念
- Java中字符串处理类
  - String类
  - StringBuffer类
  - StringBuilder类
  - StringTokenzier类

字符串

- 定义：n个字符组成的序列
- 字符串常量：一对双引号" "定界起来的字符序列
  - 如："Hello World !"
- 若两个双引号之间没有任何字符，则为空串
  - “”

字符串类

- Java中字符串相关类包括：
  - 字符串常量类
    - java.lang.String
  - 字符串变量类
    - java.lang.StringBuffer
    - java.lang.StringBuilder
    - java.util.StringTokenizer

String类

- String是字符串常量类
  - String对象的值一经赋值，其值不可变
  - 指的是所指向的内存值不可修改，但可以改变指向
- String类型变量的初始化
  - 构造方法初始化
    - String name= new String(“zhangxiao”);
  - 字符串常量初始化
    - String sex = “男”;
- String类是final的，无法被继承

String中常用的字符串处理方法

- 字符串连接
  - concat(String str)
  - “+”运算符
- 字符串查找
  - indexOf (String str)
  - lastIndexOf(String str)
  - charAt(int indexOf)
  - startsWith(String prefix)
- 字符串分割
  - split(String regex)：字符串分割
  - compareTo(String str)：字符串比较
  - equalsIgnoreCase(String str)：忽略大小写

String中常用的字符串处理方法

- 字符串替换
  - replace(char oldChar, char newChar)
- 字符串求子串
  - substring(int beginIndex, int endIndex)
- 字符串大小写转换
  - toUpperCase() 小写转大写
  - ToLowerCase()大写转小写

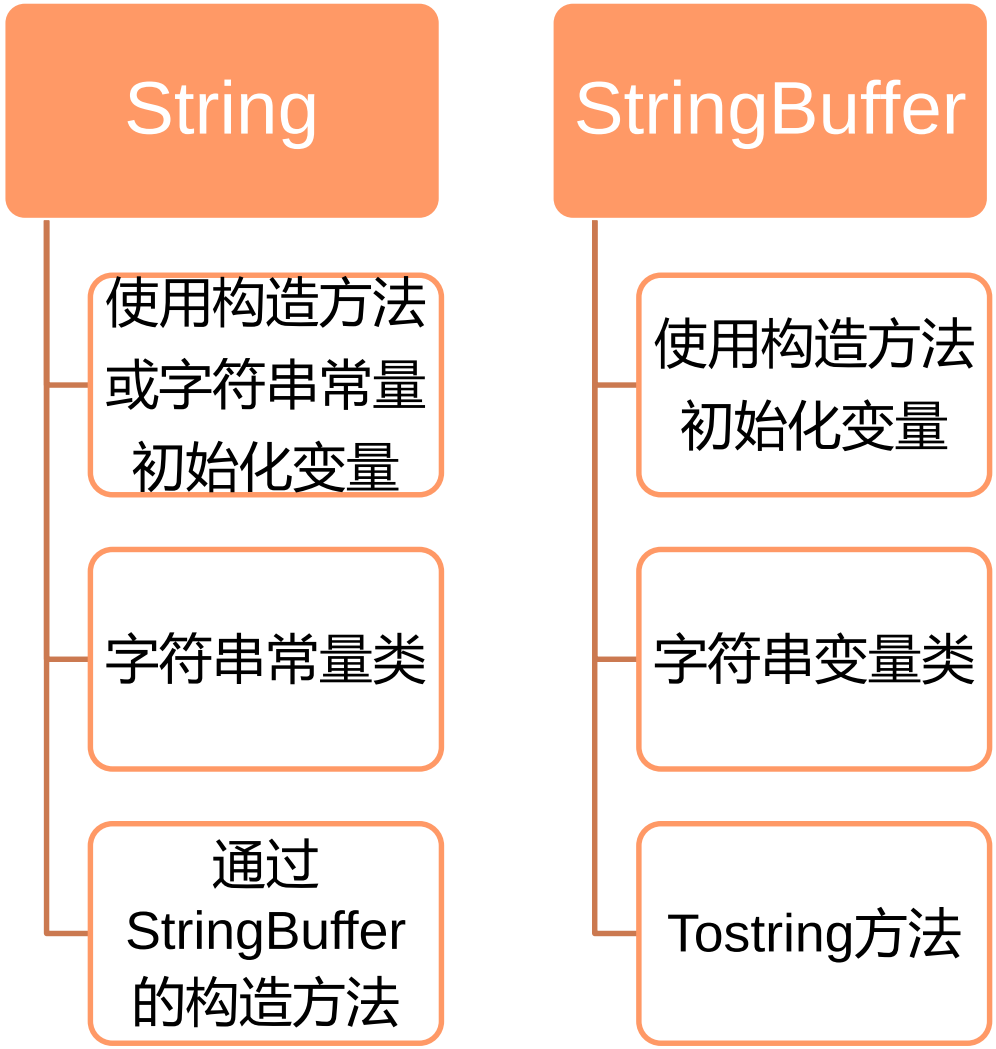
StringBuffer类

- StringBuffer类是一个字符串变量类
  - StringBuffer对象的值可以修改
  - 主要用于对字符串做大量修改的操作时
- StringBuffer变量的初始化
  - 构造方法初始化
  - StringBuffer sb = new StringBuffer(“Hello”);

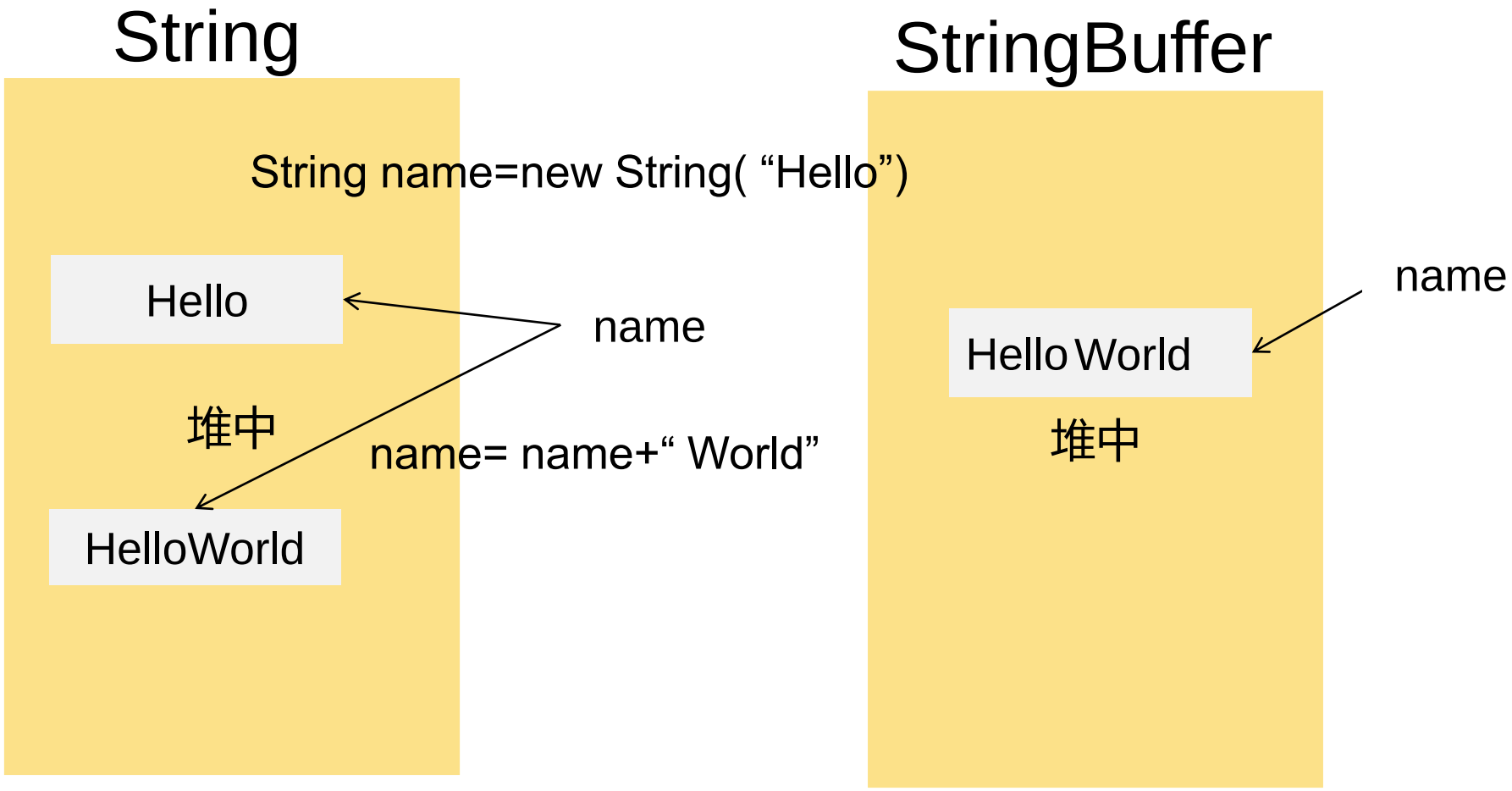
StringBuffer类中常用的方法

- 字符串连接
  - append(Object obj)
- 字符串删除
  - delete(int start,int end)
- 字符串插入
  - insert(int offset,Object obj)
- 字符串逆序
  - reverse()
- 转换为String对象
  - toString()

String与StringBuffer比较



字符串常量类与变量类区别



StringBuilder类

- StringBuilder类与StringBuffer类的方法调用是一致的。
- StringBuilder类与StringBuffer类的区别：
  - StringBuffer是线程安全的
  - StringBuilder是非线程安全的

StringTokenzier类

- StringTokenizer类主要用途是将字符串以定界符为界，分析为一个个独立的token（可理解为单词）。
- StringTokenizer中的定界符可以自己指定。
- StringTokenizer常用的方法：
  - hasMoreTokens：是否有更多的分隔符
  - nextToken：返回下一个分隔符前的内容值

讲授思路-日期的格式化

- Java中用于表示日期的类
  - Date类
  - Calendar类
- 日期的格式化和解析
  - DateFormate
  - SimpleDateFormate

日期类

- Java中常用的表示日期的类有三个
  - java.util.Date：JDK 1.1后，java.util.Date 类中的大多数方法已经不推荐使用。
  - java.sql.Date：主要针对数据库操作中的SQL使用，只有日期没有时间部分。
  - java.util.Calendar：翻译为中文称为“日历”，JDK1.1之后Calendar逐步取代了java.util.Date类，提供了更多的方式来表示日期和时间。

Date类

- java.util.Date类实际上是一个包裹类，它包含的是一个长整型数据, 表示的是从GMT(格林尼治标准时间)1970年1月 1日00:00:00这一刻之前或者是之后经历的毫秒数.
  - Date类提供了两个重载的构造方法
  - Date()：以本地当前时间构造一个Date对象。
  - Date(long date)：以距离基准（1970 年 1 月 1 日 00:00:00 GMT）时间的毫秒值构造Date对象。
  - Date date = new Date();
  - System.out.println(date.getTime()); //得到毫秒值

Calendar类

- java.util.Calendar类中文翻译为“日历”，Calendar类中定义了足够的方法来表述日历的规则。
  - Calendar类的方便之处就在于其可以灵活的设置和获取日期对应的年、月、日、时、分、秒信息，并且可以灵活的对日期进行加减操作。
- Calendar是一个抽象类，Java中提供了Calendar的一个具体实现类GregorianCalendar。
- 创建Calendar类型对象：
  - 通过其中的静态方法Calendar.getInstance()。
  - 得到的对象是一个Calendar类的对象。
- Calendar中的set(int field)和get(int field)方法可以用来设置和读取日期的特定部分。

Calendar类属性常量（一）

- HOUR：get 和 set 的字段数字，指示上午或下午的小时。
- DATE：set和get的字段数字，指示一个月中的某一天。
- MONTH：指示月份的 get 和 set 的字段数字。
- YEAR：指示年的 get 和 set 的字段数字。
- DAY_OF_MONTH：get 和 set 的字段数字，指示一个月中的某天。
- DAY_OF_WEEK：get 和 set 的字段数字，指示一个星期中的某天。
- DAY_OF_WEEK_IN_MONTH：get 和 set 的字段数字，指示当前月中的第几个星期。

Calendar类属性常量（二）

- DAY_OF_YEAR：get 和 set 的字段数字，指示当前年中的天数。
- WEEK_OF_MONTH：get 和 set 的字段数字，指示当前月中的星期数。
- WEEK_OF_YEAR：get 和 set 的字段数字，指示当前年中的星期数。
- HOUR_OF_DAY：get 和 set 的字段数字，指示一天中的小时。
- AM_PM：get 和 set 的字段数字，指示 HOUR 是在中午之前还是在中午之后。
- PM：指示从中午到午夜之前这段时间的 AM_PM 字段值
- AM：指示从午夜到中午之前这段时间的 AM_PM 字段值

Calendar类示例

- 打印当前日期是几月

```
String[] months = { "一月", "二月", "三月", "四月", "五月",
"六月", "七月", "八月",
"九月", "十月", "十一月", "十二月" };
Calendar rightNow = Calendar.getInstance();
int monthConstant = rightNow.get(Calendar.MONTH);
System.out.println(months[monthConstant]);
```

Calendar与Date的转换

- 从一个 Calendar 对象中获取 Date 对象
  - Calendar calendar = Calendar.getInstance();
  - Date date =calendar.getTime();

- 从一个Date对象获得Calendar对象
  - Calendar calendar = Calendar.getInstance();
  - Date date = new Date(long 型参数);
  - calendar.setTime(date);

日期的格式化和解析

- 在不同的应用场景中会需要以不同的格式显示日期，那么日期类型数据的格式化和解析就成为程序设计中日期相关最核心和最主要的操作。

- Java中提供了专门格式化日期的类
  - java.text.DateFormat
  - java.text.SimpleDateFormat

DateFormate

- DateFormat 是日期/时间格式化的抽象类，它以与语言无关的方式格式化并解析日期或时间。
- DateFormat实例创建
  - DateFormat df = DateFormat.getDateInstance(参数)
  - 参数的取值
    - DateFormat.SHORT 完全为数字，如12-9-10
    - DateFormat.MEDIUM 较长，如 2012-9-10
    - DateFormat.LONG 更长，如 2012年9月10日
    - DateFormat.FULL 是完全指定，如2012年9月10日星期一
- DateFormate格式化、解析日期的方法
  - String format(Date date)：格式化日期
  - Date parse(String sateStr)：解析字符串

SimpleDateFormate

- SimpleDateFormate是与环境有关的格式化、解析日期的具体类。
  - 用户可以定义日期-时间格式的模式

```
SimpleDateFormat  sdf = new SimpleDateFormat("yyyy-MM-dd hh:mm:ss");
SimpleDateFormat  sdf2=new SimpleDateFormat("yyyy年MM月dd日 hh点");
try {
     Date date = sdf.parse("2008-12-23 12:23:45");

     System.out.println(sdf2.format(date));  //输出2008年12月23日 12点

} catch (ParseException e) {
      e.printStackTrace();
}
```

日期和时间模式字符表示

字母	日期或时间元素	表示	示例
y	年	Year	2012；12
M	年中的月份	Month	July；Jul；07
w	年中的周数	Number	27
W	月份中的周数	Number	2
D	年中的天数	Number	189
d	月中的天数	Number	23
F	月份中的星期	Number	2
a	am/pm标记	Text	PM
E	星期中的天数	Text	Tuesday；Tue
H	一天中的小时数（0~23）	Number	0
k	一天中的小时数（1~24）	Number	24
K	am/pm中的小时数（0~11）	Number	0
h	am/pm中的小时数（1~12）	Number	12
m	小时中的分钟数	Number	30
s	分钟中的秒数	Number	50
S	毫秒数	Number	978

武

总结

- 字符串
  - 字符串的基本概念
  - Java中字符串处理类
    - String类
    - StringBuffer类
    - StringBuilder类
    - StringTokenzier类
- 日期的格式化
  - Java中用于表示日期的类
    - Date类
    - Calendar类
  - 日期的格式化和解析
    - DateFormate
    - SimpleDateFormate

练习

- 选择合适的日期类型表示现在的时间，并计算现在距离自己的生日还有多少天？
- 编写一个类，其中的功能是获得本周一的日期？
- 编写一个类，其功能是使用SimpleDateFormate类打印同一日期不同的形式？

Thank You