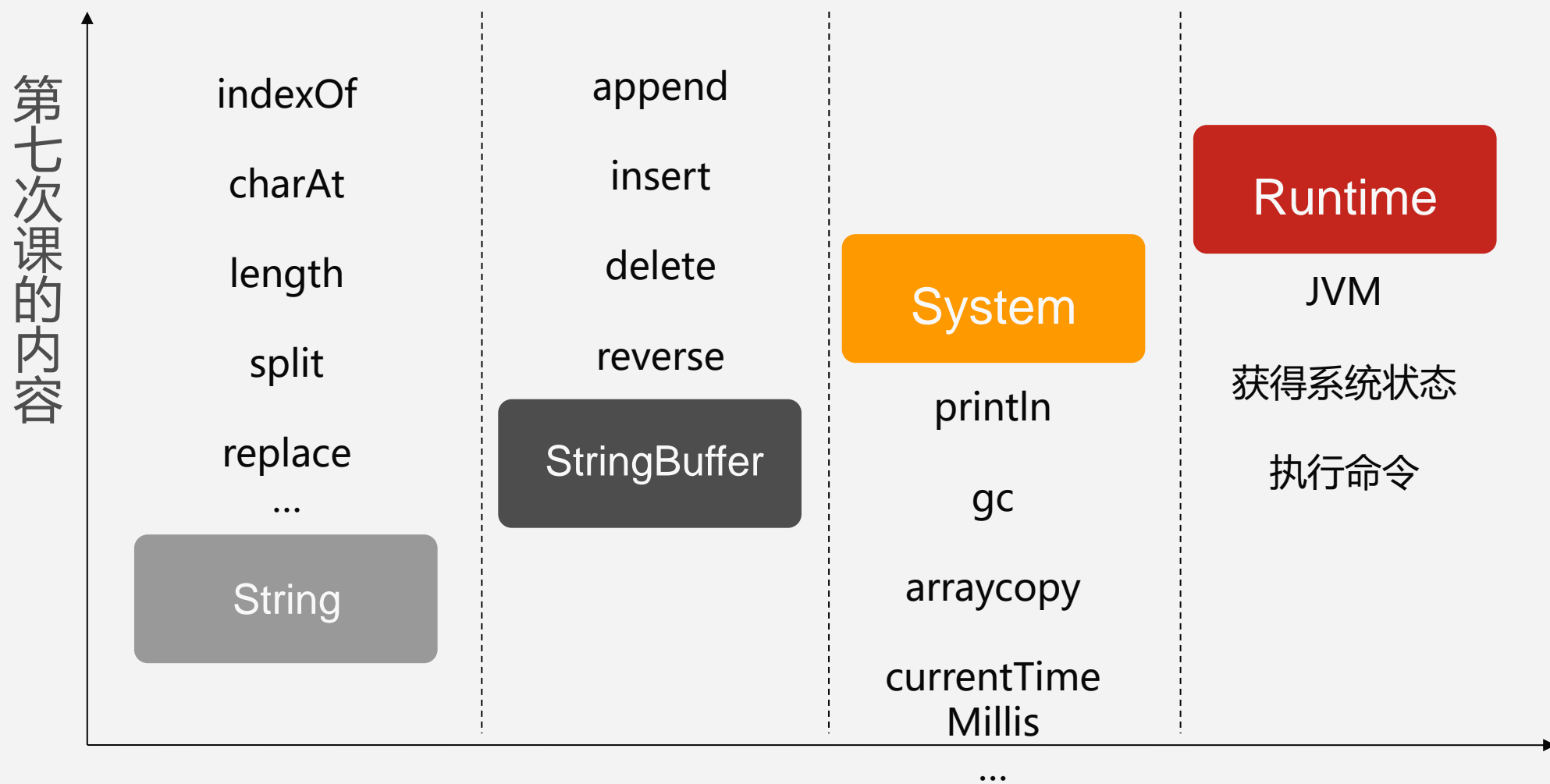




# Java

## 面向对象程序设计

软件学院 贾伟峰



小项目：使用Java API，模拟一个电信计费系统的功能。

## 生成通话记录

如何生成？

01

02

## 计费功能

如何计费？

## 打印消费清单

怎么计费？  
怎么打印？

03

04

## 功能扩展

如何满足不同计费需求？

思路：设计一个类*TelcomUser*  
包含若干方法。

系统运行起来就像这个样子，接下来该如何设计呢？

```
1
2 public class TelcomAccountSystem {
3
4     public static void main(String[] args) {
5         //实例化一个电信用户类TelcomUser
6         TelcomUser telcomUser = new TelcomUser("13800138000");
7         //生成通话记录
8         telcomUser.generateCommunicateRecord();
9         //打印通话详单
10        telcomUser.printDetails();
11    }
12
13 }
```



## TelcomUser类该如何设计?

```
1 import java.util.*;
2 class TelcomUser {
3     private String phoneNumber;
4     private String callTo;
5     private StringBuffer communicationRecords;
6     public TelcomUser(String phoneNumber) {
7         this.phoneNumber = phoneNumber;
8         this.communicationRecords = new StringBuffer();
9     }
10
11     //模拟通话记录的生成
12     void generateCommunicateRecord() {
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32     //随机生成被叫号码（后四位随机）并返回
33     private String getCallToPhoneNumber() {
34
35
36
37
38
39
40     //模拟计费办法，以字符串的形式返回保留4位小数的计费结果
41     private String accountFee(long timeStart, long timeEnd) {
42
43
44
45
46
47
48
49
50     //打印通话记录
51     void printDetails() {
52
53
54
55
56
57
58
59 }
60 }
```

假定是被叫号码

假定是主叫号码

假定通话记录存在这里，以分号相隔；每条记录内部有主叫号码、被叫号码、开始时间、结束时间



## 如何生成通话记录? ——generateCommunicateRecord方法的设计思路

```
11 //模拟通话记录的生成
12 void generateCommunicateRecord() {
13     //随机生成通话记录数目
14     int recordNum = new Random().nextInt(10);
15     for(int i = 0; i <= recordNum; i++) {
16         //随机生成第i条通话记录
17         //开始时间, 当前时间之前的某个随机时间
18         //结束时间开始后的十分钟内随机的一个时间, 至少一分钟
19         //被叫号码
20         //插入通话记录
21     }
22 }
23 }
```

# 开始时间和结束时间

//开始时间，当前时间之前的某个随机时间


```
long timeStart = System.currentTimeMillis() - new Random().nextInt(36000000);
```

//结束时间开始后的十分钟内随机的一个时间，至少一分钟

```
long timeEnd = timeStart + 60000 + new Random().nextInt(600000);
```

## 被叫号码的生成

```
32 //随机生成被叫号码（后四位随机）并返回
33 private String getCallToPhoneNumber() {
34     return "1380372" + String.valueOf(
35         + String.valueOf(
36         + String.valueOf(
37         + String.valueOf(
38     }
39
```



怎么补充?

# 将通话记录存起来

```
1 import java.util.*;
2 class TelcomUser {
3     private String phoneNumber;
4     private String callTo;
5     private StringBuffer communicationRecords;
6     public TelcomUser(String phoneNumber) {
7         this.phoneNumber = phoneNumber;
8         this.communicationRecords = new StringBuffer();
9     }
10
11     //模拟通话记录的生成
12     void generateCommunicateRecord() {}
13
14     //随机生成被叫号码（后四位随机）并返回
15     private String getCallToPhoneNumber() {}
16
17     //模拟计费办法，以字符串的形式返回保留4位小数的计费结果
18     private String accountFee(long timeStart, long timeEnd) {}
19
20     //打印通话记录
21     void printDetails() {}
22 }
```

假定是被叫号码

假定是主叫号码

假定通话记录存在这里，以分号相隔，每条记录内部有主叫号码、被叫号码、开始时间、结束时间

```
23 //插入通话记录
24 this.communicationRecords.append(this.phoneNumber +
25     " " + timeStart +
26     " " + timeEnd +
27     " " + this.callTo +
28     " ");
```

补充完整吧！

## 清单打印功能如何实现呢？




```
1
2 public class TelcomAccountSystem {
3
4     public static void main(String[] args) {
5         //实例化一个电信用户类TelcomUser
6         TelcomUser telcomUser = new TelcomUser("13800138000");
7         //生成通话记录
8         telcomUser.generateCommunicateRecord();
9         //打印通话详单
10        telcomUser.printDetails();
11    }
12
13 }
```

针对communicationRecords中的字符串进行split，然后...

# printDetails方法的设计思路

```
50 //打印通话记录
51 void printDetails() {
52     //获取全部通话记录
53     String allRecords = this.communicationRecords.toString();
54     //分割通话记录
55     String [] recordArray = allRecords.split(";");
56     //循环分割记录内的每一项并输出
57     for(int i = 0; i < recordArray.length; i++) {
58
59     }
60 }
61 }
```



想想循环里面的代码该怎么写?

# 逐一输出该通话记录中的每个信息。计费如何实现？

```
59 String [] recordField = recordArray[i].split(",");
60 System.out.println("主叫: " + recordField[0]);
61 System.out.println("被叫: " + recordField[1]);
62 System.out.println("通话开始时间: " + new Date(Long.parseLong(recordField[2]));
63 System.out.println("通话结束时间: " + new Date(Long.parseLong(recordField[3]));
64 System.out.println("计费: "
65     + accountFee(Long.parseLong(recordField[2]), Long.parseLong(recordField[3]))
66     + " 元。");
```

如何实现?

该如何补上这些代码?

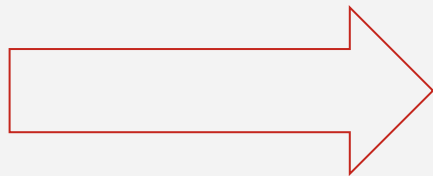
包装类Long的parseLong方法，可以将String转为long



## 包装类：对基本类型的包装，使之具有面向对象特性


byte  
char  
int  
short  
long  
float  
double  
boolean

基本数据类型



Byte  
Character  
Integer  
Short  
Long  
Float  
Double  
Boolean

包装类

```
40 //模拟计费办法，以字符串的形式返回保留4位小数的计费结果
41 private String accountFee(long timeStart, long timeEnd) {
42     //每分钟收费*元
43     double feePerMinute = 0.2;
44     //通话分钟数按四舍五入计算
45     int minutes = Math.((timeEnd - timeStart)/60000);
46     double feeTotal = feePerMinute * minutes;
47     return String.format("%.4f", feeTotal);
48 }
```

哪个方法呢?

什么意思?

以上代码如果顺利完成，程序运行结果应该是这个样子——

```
|-----通话记录分割线-----  
主叫: 13800138000  
被叫: 13803726629  
通话开始时间: Sat Nov 11 17:47:58 CST 2017  
通话结束时间: Sat Nov 11 17:54:15 CST 2017  
计费: 1.2000 元。  
-----通话记录分割线-----  
主叫: 13800138000  
被叫: 13803720179  
通话开始时间: Sat Nov 11 17:26:00 CST 2017  
通话结束时间: Sat Nov 11 17:33:15 CST 2017  
计费: 1.4000 元。  
-----通话记录分割线-----  
主叫: 13800138000  
被叫: 13803720764  
通话开始时间: Sat Nov 11 17:22:40 CST 2017  
通话结束时间: Sat Nov 11 17:31:56 CST 2017  
计费: 1.8000 元。  
-----通话记录分割线-----  
主叫: 13800138000  
被叫: 13803729069  
通话开始时间: Sat Nov 11 16:00:10 CST 2017  
.....
```

## 思考

(1) 如何使用DateFormat对输出格式进行改进?

(2) SimpleDateFormat呢?

提示: *p227-228*

通话开始时间: 2017-11-11 14:43:37  
通话结束时间: 2017-11-11 14:51:06  
计费: 1.4000 元。

-----通话记录分割线-----

主叫: 13800138000

被叫: 13803725787

通话开始时间: 2017-11-11 22:21:21

通话结束时间: 2017-11-11 22:28:23

计费: 1.4000 元。

-----通话记录分割线-----

主叫: 13800138000

被叫: 13803725796

通话开始时间: 2017-11-11 14:54:20

通话结束时间: 2017-11-11 14:59:27

计费: 1.0000 元。

-----通话记录分割线-----

主叫: 13800138000

被叫: 13803725259

通话开始时间: 2017-11-11 22:13:43

通话结束时间: 2017-11-11 22:20:01

计费: 1.2000 元。

就像这样

提示: p229

通话开始时间: 2017年11月11日 08时19分46秒  
通话结束时间: 2017年11月11日 08时30分16秒  
计费: 2.0000 元。

-----通话记录分割线-----

主叫: 13800138000

被叫: 13803727861

通话开始时间: 2017年11月11日 08时09分47秒

通话结束时间: 2017年11月11日 08时17分18秒

计费: 1.4000 元。

-----通话记录分割线-----

主叫: 13800138000

被叫: 13803724171

通话开始时间: 2017年11月11日 04时34分54秒

通话结束时间: 2017年11月11日 04时41分16秒

计费: 1.2000 元。

-----通话记录分割线-----

主叫: 13800138000

被叫: 13803727037

通话开始时间: 2017年11月11日 10时58分32秒

通话结束时间: 2017年11月11日 11时09分10秒

计费: 2.0000 元。

就像这样。

# 开始时间和结束时间，试试Calendar类

```
27 //用Calendar获取当前时间
28 Calendar cal = Calendar.getInstance();
29 //随机减去若干小时（10小时以内）
30 cal.add(Calendar.HOUR, - new Random().nextInt(10));
31 //获得对应毫秒
32 long timeStart = cal.getTimeInMillis();
33 //结束时间开始后的十分钟内随机的一个时间，至少一分钟
34 long timeEnd = timeStart + 60000 + new Random().nextInt(600000);
```

## 思考



不同电信公司计费方法不同，比如：

- (1) accountFee中的单价不同；
- (2) 分钟数向上取整；
- (3) 分钟数向下取整；

若其他功能一样，程序该如何更改？能否引入面向对象技术，减少代码的重复编写呢？

