2020-6-22

Android移动应用开发

——高级计算器



系(院)名 称：计算机与信息工程学院

专 业 名 称：软件工程

学 生 班 级：计18 软件班

学 生 姓 名：陈 天 莹

学 生 学 号：1815080127

指 导 教 师：邓 育 林 讲师

目录

[一、问题描述与需求分析 2](#_Toc43820174)

[1、问题描述 2](#_Toc43820175)

[2、功能需求分析 2](#_Toc43820176)

[二、概要设计 3](#_Toc43820177)

[1、设计思路 3](#_Toc43820178)

[2、相对于基本需求的新增功能 3](#_Toc43820179)

[3、程序结构 4](#_Toc43820180)

[三、各模块的功能及程序说明 5](#_Toc43820181)

[1、计算器界面设计 5](#_Toc43820182)

[2、界面各模块功能 8](#_Toc43820183)

[3、ScienceCalculator接口 22](#_Toc43820184)

[4、BaseCalculator 接口 26](#_Toc43820185)

[四、操作流程 29](#_Toc43820186)

[1、操作流程步骤 29](#_Toc43820187)

[2、操作流程图 30](#_Toc43820188)

[五、程序运行截图 31](#_Toc43820189)

[1、弧度角度运算 31](#_Toc43820190)

[2、数学表达式 31](#_Toc43820191)

[3、包含科学计算的数学表达式 32](#_Toc43820192)

[4、处理异常 32](#_Toc43820193)

[5、保存运算过程到文件 32](#_Toc43820194)

[6、再按一次退出程序 33](#_Toc43820195)

[六、参考文献 33](#_Toc43820196)

[七、设计过程中遇到的难点与解决方法 34](#_Toc43820197)

[1、替换app图标 34](#_Toc43820198)

[2、“再按一次退出程序”功能 34](#_Toc43820199)

[3、通过屏幕纵横切换来变换Activity 35](#_Toc43820200)

[八、课程总结 35](#_Toc43820201)

[附录：程序主类源代码 36](#_Toc43820202)

[1、AutoScaleTextView.java 36](#_Toc43820203)

[2、BaseCalculator.java 38](#_Toc43820204)

[3、LandActivity.java 43](#_Toc43820205)

[4、MainActivity.java 66](#_Toc43820206)

[5、MyNumberPicker.java 83](#_Toc43820207)

[6、ScienceCalculator.java 84](#_Toc43820208)

一、问题描述与需求分析

1、问题描述

实现一个计算器app，要有输入框清空和删除前一位输入功能，可以实现在前一步计算结果上继续作加、减、乘、除，即连加、减、乘、除操作。除了0~9这10个数字按键以及+、-、\*、/等还应支持（）的优先运算和平方、开平方、幂运算等功能。

2、功能需求分析

1) 有良好的界面UI设计；

2) 设计的app有一个一目了然的中文名称；

3) 为app更换默认的系统图标；

4) 该计算器应是一个简单的科学计算器；

5) 应有0-9这10个数字按键；

6) 本计算器应支持输入框清空和删除前一位输入的功能；

7) 可以进行加、减、乘、除的操作，且可以在前一步的结果上继续进行运算；

8) 可支持括号的优先运算和平方、开平方、幂运算功能；

9) 简单地模仿手机计算器的其他功能。

二、概要设计

1、设计思路

1)设计计算器界面

计算器界面用xml文件编写，包括手机竖屏和横屏两个布局文件：

竖屏：activity\_main.xml 布局为 portrait

横屏：activity\_main\_land.xml 布局为 landscape

竖屏模式可以完成基本的四则运算，不涉及科学计算

横屏模式除了完成基本的四则运算，还添加了科学运算

2)编写计算接口

ScienceCalculator 可以完成包含科学运算函数的 math，先实现可以完成基本四则运算的 BaseCalculator，在此基础上，实现 ScienceCalculator。

运算的思路是先通过 ScienceCalculator 完成math中需要科学计算函数的部分，再用这些部分计算的结果替换原 math 中的这些部分，使包含科学计算函数的 math 转变成可用 BaseCalculator 计算的 math。

2、相对于基本需求的新增功能

1）三个三角函数的运算：sin()、cos()、tan()；

2）两个对数函数的运算：ln()、log()；

3）在左上方的文本框中显示当前的计算过程，在右边的文本区中显示以往的计算过程；

4）实现一个数的倒数运算；

5）增加π的计算；

6）增加e的运算；

7）单击“保存”按钮可以将文本区中的全部计算过程保存到文件；单击“复制”按钮可以将文本区中选中的文本复制到剪贴本；单击“清除”按钮可以清除文本区的全部内容；

8）实现“再按一次提出程序”功能；

9）屏幕的横向和纵向可以使界面在基础运算和高级运算间转换。

3、程序结构

开始

结束

完成计算

用户在计算器的tvNow中输入math表达式

搜索结果显示到tvNow

四

ScienceCalculator调用cal方法

则

BigDecimal格式化处理运算结果

运

算

传入math表达式，精度precision，角度值angel\_metric

部

BaseCalculator调用cal方法

分

预处理math，去掉空格，替换n，自然指数

获得可用BaseCalculator计算的math

计算剩下的科学运算符：sin，cos，tan，ln，log并替换

计算pow运算并替换原math相应部分

科学计算部分

三、各模块的功能及程序说明

1、计算器界面设计

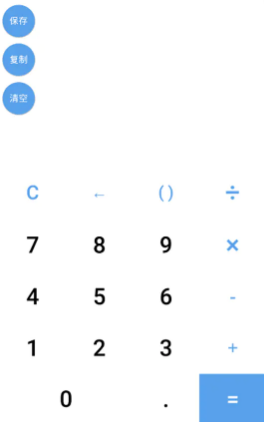
1）竖屏界面

包含控件：

1. 文本框 TextView：tvNowt，vPast 分别显示当前和过去的运算过程；
2. 功能 Buttion：btn\_save，btn\_copy，btn\_clear 用于保存，复制，清空tvPast中的运算过程；
3. 数字 Button：0-9，小数点；
4. 运算符 Button：+ - × / ( ) = ；
5. 运算器基本 Button：btn\_del 退格，btn\_clc 清空当前math。

成员变量：

1. String mathPast，用于存储过去的运算过程；
2. String mathNow，用于存储当前的运算过程，即用户正在输入的部分；
3. int precision，设置默认精度为6位小数；
4. int equal\_flag，设置flag值判断是否需要清空mathNow进行新的运算；
5. ScienceCalculator scienceCalculator，实例化一个科学计算器。



竖屏界面

2）横屏界面

包含控件：

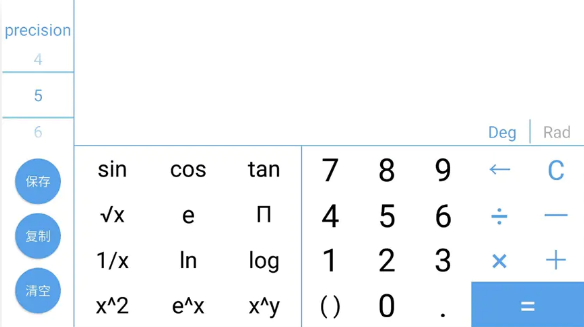
1. 文本框 TextView：tvNow, tvPast 分别显示当前和过去的运算过程；
2. 功能 Buttion：btn\_save，btn\_copy，btn\_clear 用于保存，复制，清空tvPast中的运算过程；
3. 数字Button：0-9，小数点；
4. 基本运算符Button：+ - × / ( ) = ；
5. 科学运算符Button：（12个）sin，cos，tan，√x，e，π，1/x，ln，log，x2，ex，xy ；
6. 运算器基本Button：btn\_del退格，btn\_clc清空当前math；
7. 文本框切换按钮 tvRad，tvDeg 实现弧度制和角度值的切换；
8. 精度选择器 NumberPicker。

成员常量：

1. final int DEG = 0，DEG 表示角度制；
2. final int RAD = 1，RAD 表示弧度制。

成员变量：

1. String mathPast，用于存储过去的运算过程；
2. String mathNow，用于存储当前的运算过程，即用户正在输入的部分；
3. int precision，设置默认精度为6位小数，通过NumberPicker返回用户设置的精度值；
4. int equal\_flag，设置flag值判断是否需要清空mathNow进行新的运算；
5. ScienceCalculator scienceCalculator，实例化一个科学计算器；
6. int angle\_metric，角度制参数，默认为DEG。



横屏界面

2、界面各模块功能

由于横屏界面包括了竖屏界面所有的模块，下文代码功能描述按照 LandActivity.java 文件，即横屏界面对应的 Activity。

**1) 初始化tvPast**

tvPast 用于存储过去的运算过程。

**响应场景设置：**

1.因为tvPast文本框高度有限，为了使用户每次都可以看到最新的运算过程，设置 setMovementMethod(ScrollingMovementMethod.getInstance()) 方法使内容自动滚动到最新的一行；



2.安卓系统集成了很好的文本框内容复制功能，设置setTextlsSelectable的值为true即可实现文本框的长按复制功能；



3.由于计算器具有2个界面，当前的界面可能是从竖屏界面切换来（如果当前界面是竖屏，界面也有可能是从横屏界面切换而来），通过Intent类在两个Activity间传递tvPast的内容，至于用for循环逐行添加过去的运算过程是为了满足使内容自动滚动到最新的一行。

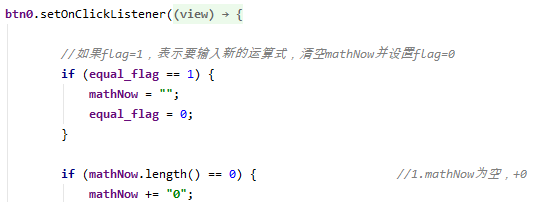


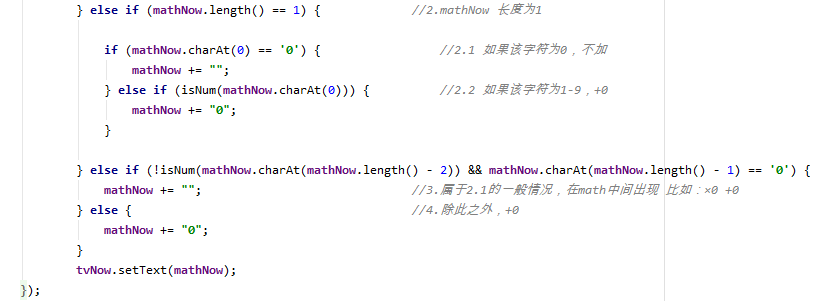
**2) 初始化 NumButtons：0-9，小数点**

按钮需要设置监听事件的应用场景，是为了避免一些错误的math格式。因为不同的数字有不同的处理方式。主要归为以下几类：

**1．btn\_0**

btn\_0 根据响应事件场景在当前 math 表达式中添加 0



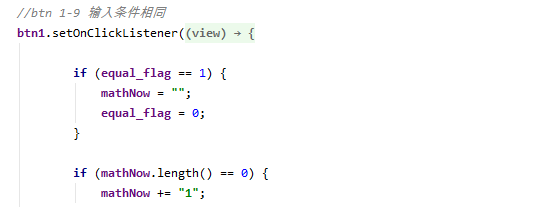


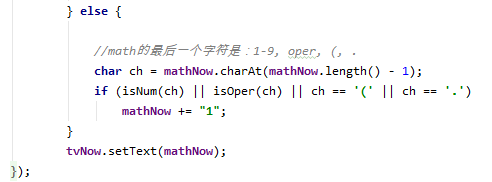
响应场景设置：

* 设置 flag 值判断是否需要清空 mathNow 进行新的运算，该功能是为了方便用户的输入，用户在完成一次计算之后，不需要点击清空按钮就可以直接输入新的运算过程，当 equal\_flag 为1时表示刚刚完成一次运算，可以直接输入新的运算式了，此时完成 mathNow 清空操作，并重置 equal\_flag 为 0；
* 是否添加0的场景设置：
  + mathNow 长度为0，添加0
  + mathNow 长度为1，当前输入1个char了  
    如果当前 char 为0，不添加0  
    如果当前 char 为1-9，添加0
  + mathNow 长度 >1，if中的条件是2.1的一般情况，即在 math 中间出现了，mathNow 的倒数第2个 char 不是 Num 并且 mathNow 的最后一个 char 是0，  
    如 2 + 3 ×0 ，此时也不添加0
  + 除此之外，添加0

**2．btn\_[1-9]**

btn\_1 - btn\_9 的响应场景相同，根据响应事件场景在当前 math 表达式添加 [1-9]



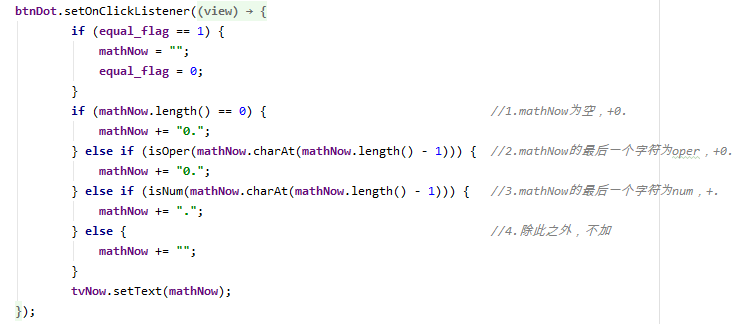


响应场景设置：

* equal\_flag 同 btn\_0；
* mathNow 长度为0，添加[1-9]；
* mathNow 最后一个 char 是 [0-9]，oper，小数点 这几种情况时，加[1-9]；
* 除此之外，不加[1-9]

**3．btn\_dot 小数点**

小数点操作要比普通数字要多一点，有时点击添加的是“0.”btn\_dot 根据响应事件场景在当前math表达式中添加“.”或者“0.”



响应场景设置：

* equal\_flag 同 btn\_0；
* mathNow 长度为0，添加“0.”
* mathNow 的最后一个 char 为 oper，添加“0.”
* mathNow 的最后一个字符为 num，添加“.”
* 除此之外，不添加

**3) 初始化 BaseOperButtons**

包括 + - × / ( ) =

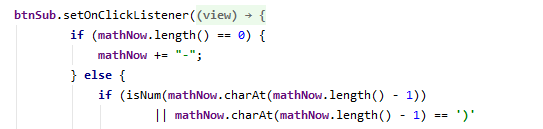
**1. btn\_add +**

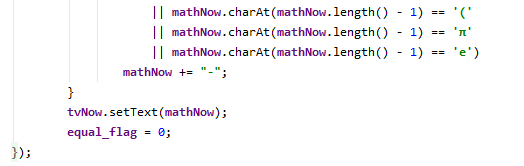


响应场景设置：

1. mathNow长度为0，添加“+”，表示正数
2. 以下5种场景都可以添加“+”，设char是mathNow的最后一个char：
   * char是Num
   * char是“)”
   * char是“(”
   * char是“π”
   * char是“e”，自然指数

**2. btn\_sub –**

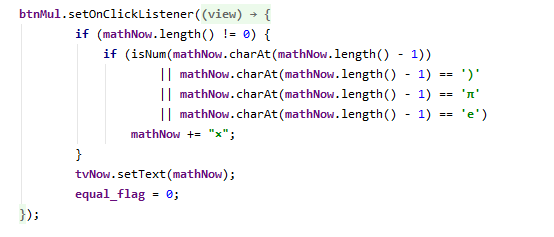




响应场景设置：

* mathNow 长度为0，添加“-”，表示正数
* 同 btn\_add

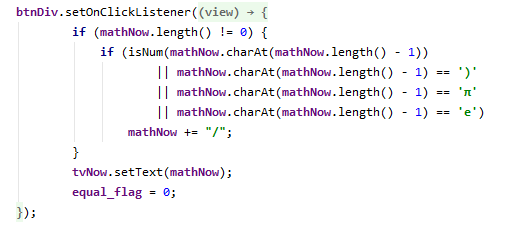
**3．btn\_mul ×**



响应场景设置：

1. × 不能出现在math表达式的首位，所以场景限制在mathNow长度不为0
2. 以下4种场景都可以添加 “×”，设 char 是 mathNow 的最后一个 char：
   * char是Num
   * char是“)”
   * char是“π”
   * char是“e”，自然指数

**4. btn\_div /**



响应场景设置：

1. / 不能出现在math表达式的首位，所以场景限制在mathNow长度不为0
2. 以下4种场景都可以添加 “/”，设 char 是 mathNow 的最后一个 char：
   * char是Num
   * char是“)”
   * char是“π”
   * char是“e”，自然指数

**5. btn\_bracket ( )**



响应场景设置:

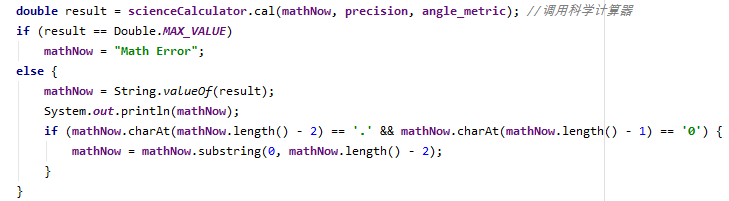
* + equal\_flag同btn\_0；
  + mathNow长度为0，+“(”
  + mathNow最后一个字符是oper，+“(”
  + mathNow最后一个字符是num, π, e
    - * 如果mathNow没有“(”， 加“×(”
      * 如果mathNow已有“(”， 加“(”
  + mathNow最后一个字符是“)”，说明用户是在补全右括号，+“)”

**6. btn\_equal =**

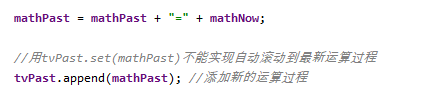
1. 右括号自动补全，通过计算 mathNow 中 “(” 和 “)” 个数的差值，添加右括号，补全当前的 mathNow



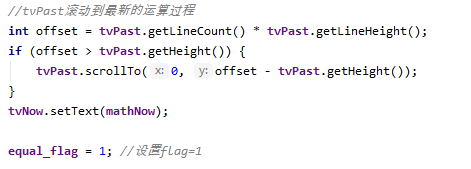
1. mathNow 预处理后进行计算，调用 ScienceCalculator 的 cal 方法计算，并根据返回值情况设定 mathNow 的结果显示为 Math Error 或者正常结果。



1. tvPast 添加新的 mathPast 到文本框



1. 获取 tvPast 文本框属性并滚动到最新的一行，将equal\_flag设为1



**4) 初始化 ScienceOperButtons**

除了x2，xy，其他 ScienceOpers 都要设置 equal\_flag，同btn\_0。

**1. btn\_sin**

****

响应场景设置：

* mathNow 长度为0，添加“sin(”
* mathNow 最后一个 char 是 base opers，(，添加“sin(”

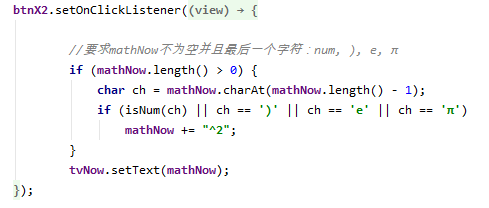
**2. btn\_cos**

****

响应场景设置：

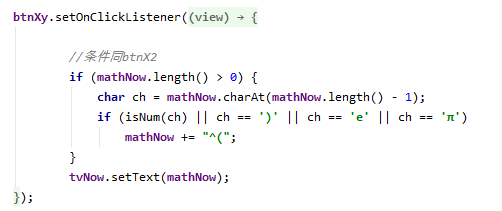
除了 x2，xy，其他 ScienceOper 的场景都和 btn\_sin 相同

**3. btnX2**

****

响应场景设置：mathNow 不为空，并且最后一个字符是：Num，)，e，π

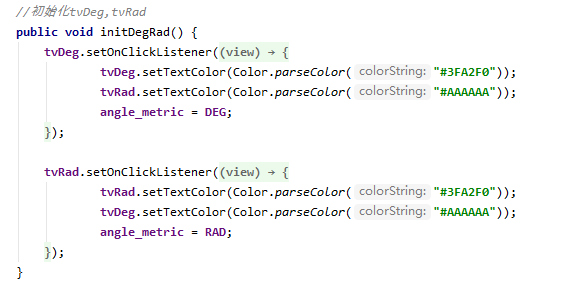
**4. btnXy**

****

响应场景设置：mathNow 不为空，并且最后一个字符是：Num，)，e，π

**5) 初始化 tvDeg，tvRad**

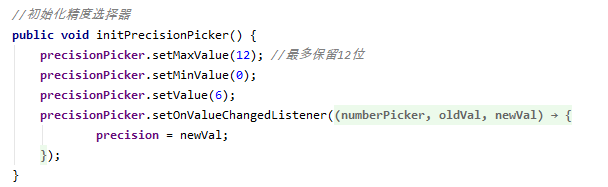
用法：点击 Deg 之后，angle\_metric 设置为 DEG，角度制，界面上 DEG 变为蓝色，RAD 变为灰色，RAD 同样是这样。

****

**6) 初始化精度选择器**

属性设置：

1. 设置精度最大为12位，最小为0位，默认设置值为6
2. NumberPicker 监听事件将用户选择的精度值传给成员变量 precision

****

**7) 初始化功能 Button**

包括 btn\_save，btn\_copy，btn\_clear

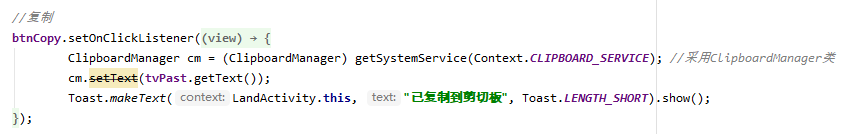
**1. btn\_save 保存**

通过字节流将 tvPast 的内容写道 storage/emulated/0/maht.txt 文件中

****

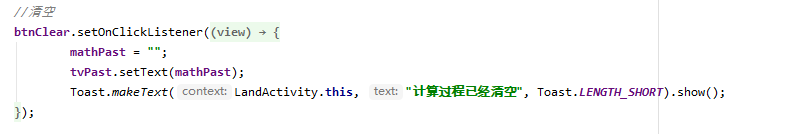
**2. btn\_copy 复制**

调用 ClipboardManager 类 setText 方法复制 tvPast 文本框中过去的运算过程。

****

**3. btn\_clear 清空**

将 tvPast 的内容置为空即可

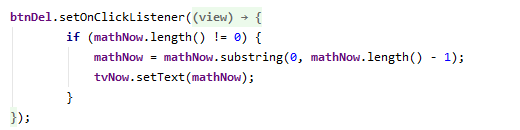
****

**8) 初始化计算器基本Buttons**

包括 btn\_del，btn\_clc

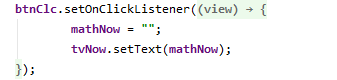
**1. btn\_del 退格**

截取掉mathNow的最后一个char即可

****

**2. btn\_clc 清空mathNow**

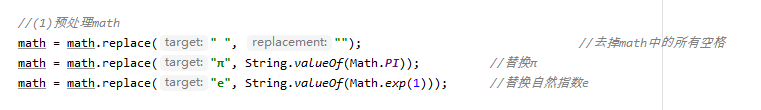
mathNow = “” 即可

****

3、ScienceCalculator接口

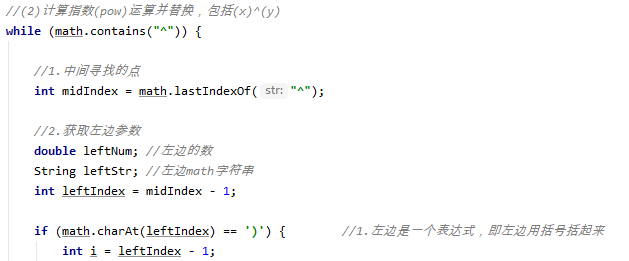
**1) 预处理 math**

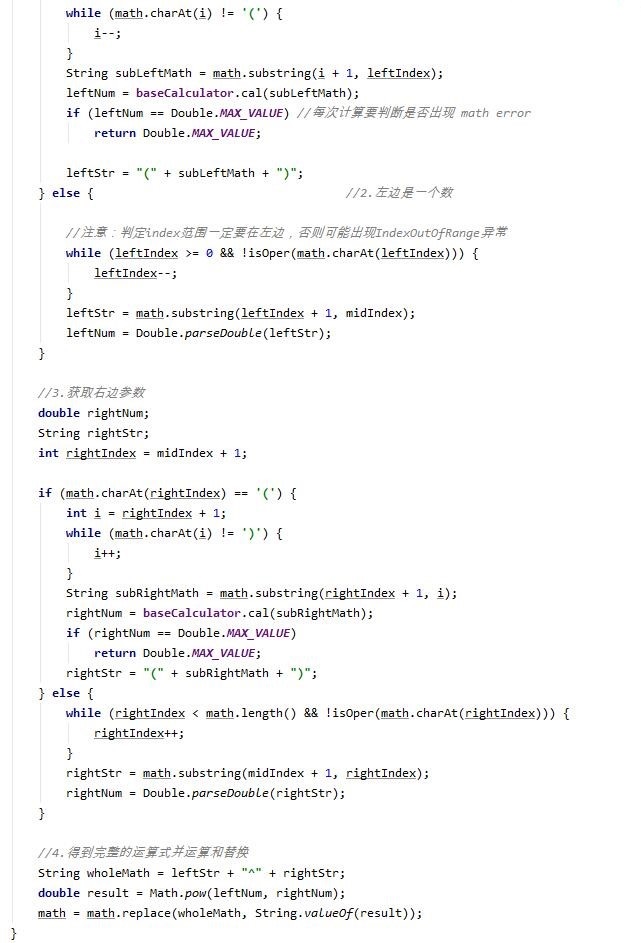
去掉 math 中的空格，替换 π，替换自然指数 e



**2) pow 运算，包含 ^ 的 ScienceOpers**

获取 ^ 左右两边参数进行 Math.pow 计算，如果参数是 Math 表达式，需要调用 BaseCalculator 方法，然后用运算结果替换科学运算式部分





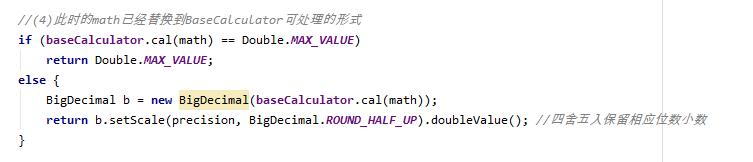
**3) 计算剩下的科学运算**

包括：sin，cos，tan，ln，log，√，通过获取括号位置，如 sin(cos(90°))，先获取 cos(90°) 完成计算，再用 Math.sin 计算，根据 angle\_metric 的情况选择 DEG 或者 RAD。

**while** (math.contains(**"sin"**)  
 || math.contains(**"cos"**)  
 || math.contains(**"tan"**)  
 || math.contains(**"ln"**)  
 || math.contains(**"log"**)  
 || math.contains(**"√"**)) {  
  
 *//1.获取()内运算式并计算出结果，此时假设()不再包含复杂的科学运算* **int** beginIndex = math.lastIndexOf(**"("**);  
 **int** endIndex = getRightBracket(math, beginIndex);  
 String subMath = math.substring(beginIndex + 1, endIndex);  
 **double** subResult = **baseCalculator**.cal(subMath);  
 **if** (subResult == Double.***MAX\_VALUE***) *//每次计算要判断是否出现 math error* **return** Double.***MAX\_VALUE***;  
  
 *//2.获取scienceOper字符串* **int** i = beginIndex - 1;  
 **while** (i >= 0 && !isOper(math.charAt(i))) { *//向左寻找* i--;  
 }  
 String scienceOper = math.substring(i + 1, beginIndex);  
  
 *//3.匹配scienceOper进行科学运算，并替换相应部分* String tempMath;  
 **double** tempResult;  
 **int** DEG = 0; *//判断角度制* **switch** (scienceOper) {  
 **case "sin"**:  
 tempMath = **"sin("** + subMath + **")"**;  
 **if** (angle\_metric == DEG) {  
 tempResult = Math.*sin*(subResult / 180 \* Math.***PI***); *//将默认的 Rad → Deg* } **else** {  
 tempResult = Math.*sin*(subResult);  
 }  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "cos"**:  
 tempMath = **"cos("** + subMath + **")"**;  
 **if** (angle\_metric == DEG) {  
 tempResult = Math.*cos*(subResult / 180 \* Math.***PI***);  
 } **else** {  
 tempResult = Math.*cos*(subResult);  
 }  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "tan"**:  
 tempMath = **"tan("** + subMath + **")"**;  
 **if** (angle\_metric == DEG) {  
 tempResult = Math.*tan*(subResult / 180 \* Math.***PI***);  
 } **else** {  
 tempResult = Math.*tan*(subResult);  
 }  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "ln"**:  
 tempMath = **"ln("** + subMath + **")"**;  
 tempResult = Math.*log*(subResult);  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "log"**:  
 tempMath = **"log("** + subMath + **")"**;  
 tempResult = Math.*log10*(subResult);  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "√"**:  
 tempMath = **"√("** + subMath + **")"**;  
 tempResult = Math.*sqrt*(subResult);  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **default**:  
 **break**;  
 }  
}

**4) BaseCalculaor 运算并格式化 result**

采用 BigDecimal 类四舍五入保留小数位数

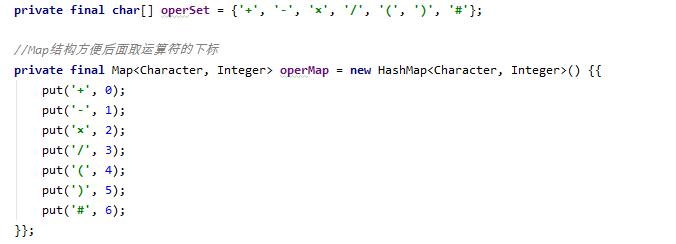


4、BaseCalculator 接口

主要是栈实现四则运算，采用了逆波兰式和运算符优先级表。

**1) operSet 和 operMap**

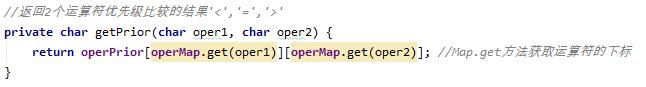
用 Map 是为了方便取运算符下标



**2) operPrior 运算符优先级表**

****

通过 getPrior 方法获取2个运算符优先级比较的结果



**3) 栈实现四则运算**

遍历 math 表达式，num 入 numStack 栈，oper 入 operStack 栈，oper 在入栈时比较其与当前栈顶 oper 的优先级：

1. **“<”**：栈顶 oper 优先级低，新 oper 入栈
2. **“=”**：说明要入栈的 oper 为 “)”，而栈顶 oper 为 “(”，去掉 “(”，其实也是 math 去括号的过程
3. **“>”**： 栈顶 oper 优先级高，oper 出栈，并将 num 运算结果 push 进 numStack  
   直到最后numStack的栈顶元素为计算结果。

在运算过程中涉及了负数的处理，即不将负数的 “-” 视为oper。

**private double** calSubmath(String math) {  
 **if** (math.length() == 0) {  
 **return** Double.***MAX\_VALUE***;  
 } **else** {  
 **if** (!hasOper(math.substring(1, math.length())) || math.contains(**"E-"**)) {  
 **return** Double.*parseDouble*(math);  
 }  
 *//设置flag用于存储math开始位置的负数，如-3-5中的-3，避免-被识别成运算符而出错* **int** flag = 0;  
 **if** (math.charAt(0) == **'-'**) {  
 flag = 1;  
 math = math.substring(1, math.length());  
 }  
 Stack<Character> operStack = **new** Stack<>(); *//oper栈* Stack<Double> numStack = **new** Stack<>(); *//num栈* operStack.push(**'#'**); *//设置栈底元素* math += **"#"**;  
 String tempNum = **""**; *//暂存数字str  
 //计算math* **for** (**int** i = 0; i < math.length(); i++) {  
 **char** charOfMath = math.charAt(i); *//遍历math中的char  
 //(1)num进栈* **if** (!isOper(charOfMath) *//1.不是oper* || charOfMath == **'-'** && math.charAt(i - 1) == **'('**) { *//2.是'-'并且'-'左边有'('，说明是在math中间用负数* tempNum += charOfMath;  
 *//1.1 获取下一个char* i++;  
 charOfMath = math.charAt(i);  
 *//1.2 判断下一个char是不是oper,如果是oper，就将num压入numStack* **if** (isOper(charOfMath)) { *//此条件成功时，下次for循环就直接跳到else语句了* **double** num = Double.*parseDouble*(tempNum);  
 **if** (flag == 1) { *//恢复math首位的负数* num = -num;  
 flag = 0;  
 }  
 numStack.push(num); *//push num* tempNum = **""**; *//重置tempNum* }  
 *//1.3 //回退，以免下次循环for语句自身的i++使得跳过了这个char* i--;  
 }  
 *//(2)oper进栈* **else** {  
  
 **switch** (getPrior(operStack.peek(), charOfMath)) {  
 *//2.1 栈顶oper优先级低，新oper入栈* **case '<'**:  
 operStack.push(charOfMath);  
 **break**;  
 *//2.2 说明当前的charOfMath为')'，而栈顶oper为'('，去掉'('，其实也是math去括号的过程* **case '='**:  
 operStack.pop();  
 **break**;  
 *//2.3 栈顶oper优先级高，oper出栈，并将num运算结果push进numStack* **case '>'**:  
 **char** oper = operStack.pop();  
 **double** b = numStack.pop();  
 **double** a = numStack.pop();  
 **if** (operate(a, oper, b) == Double.***MAX\_VALUE***)  
 **return** Double.***MAX\_VALUE***;  
 numStack.push(operate(a, oper, b));  
 i--; *//继续比较该oper与栈顶oper的关系* **break**;  
 }  
 }  
 }  
 **return** numStack.peek(); *//最后的math变成一个num了* }  
}

四、操作流程

1、操作流程步骤

1. 程序开始；
2. 在手机上点击计算器APP，进入默认的计算器竖屏界面，通过点击按钮输入math表达式，按钮设置了响应事件的场景，避免了一些math 表达式的格式错误，最后完成math 表达式的输入；
3. 点击 = 按钮进行计算，如果运算过程中出现除以0的情况或者格式错误的math表达式，输出Math Error，正常情况下完成math计算，输出计算结果；
4. 此时用户有5个选择：
5. 继续输入math表达式计算
6. 点击保存按钮将文本区中的全部计算过程保存到文件
7. 点击复制按钮将文本区中选中的文本复制到剪贴本
8. 点击清除按钮将文本区的全部内容清除
9. 点击系统返回键退出计算器
10. 用户在完成(3)中的1，2，3，4任意一个之后均可以点击系统返回键退出计算器；
11. 用户将手机横屏，App切换到科学计算器的界面，同样完成(1)，(2)，(3)，(4)操作；
12. 程序结束。

2、操作流程图

开始

点击App进入计算器界面

点击按钮输入math表达式

将文本区中的全部计算过程保存到手机sd卡的math.txt文件

点击保存

完成String类型的math表达式输入

将文本区中选中的文本复制到剪贴本

点击复制

点击等号

Math中出现除以0的情况

清除文本区的全部内容

点击清除

输出math error

完成计算，输出结果

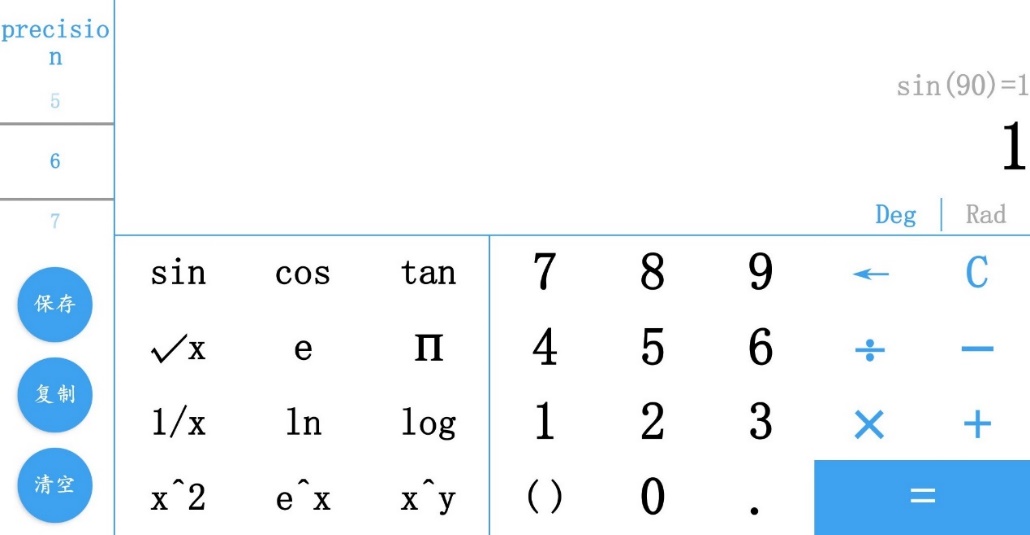
点击退出

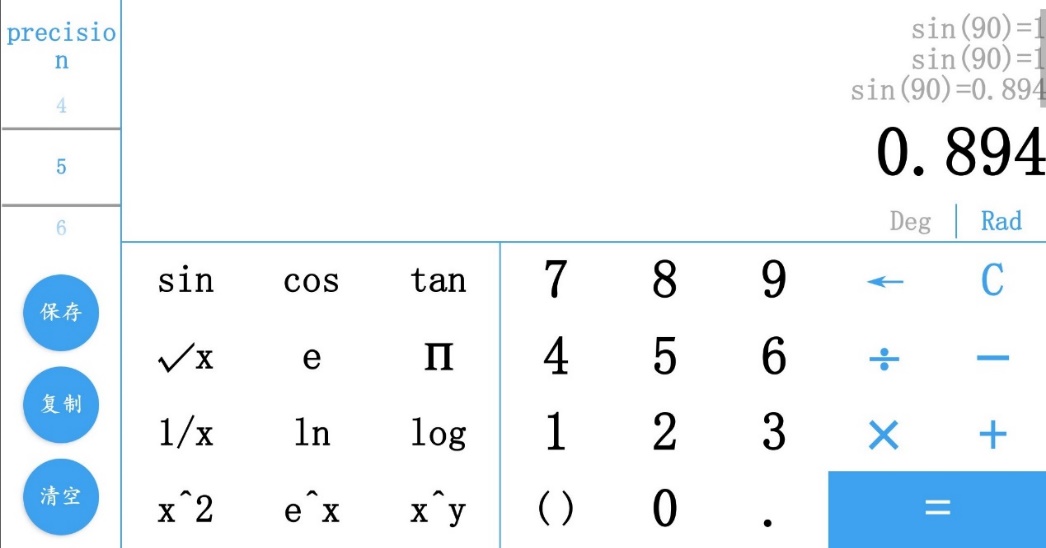
完成一次计算

结束

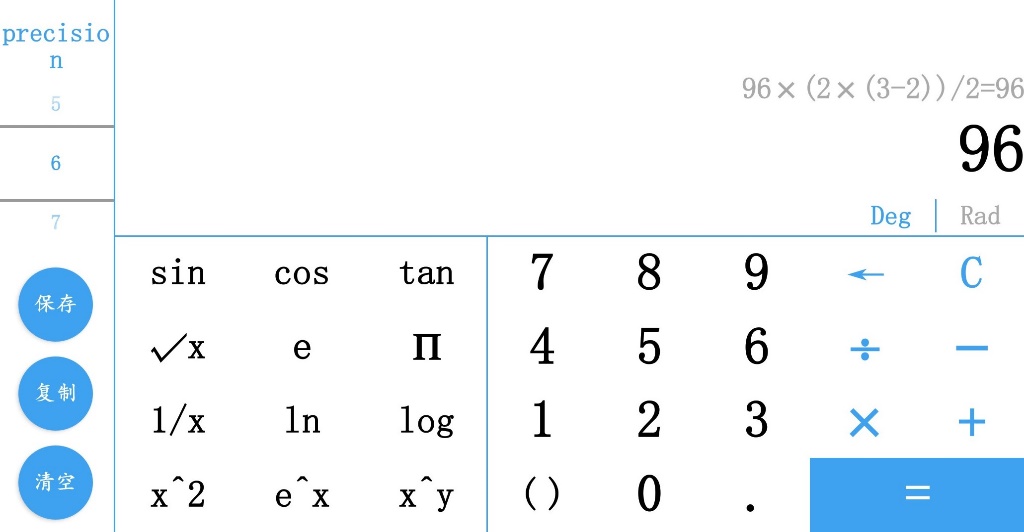
五、程序运行截图

1、弧度角度运算

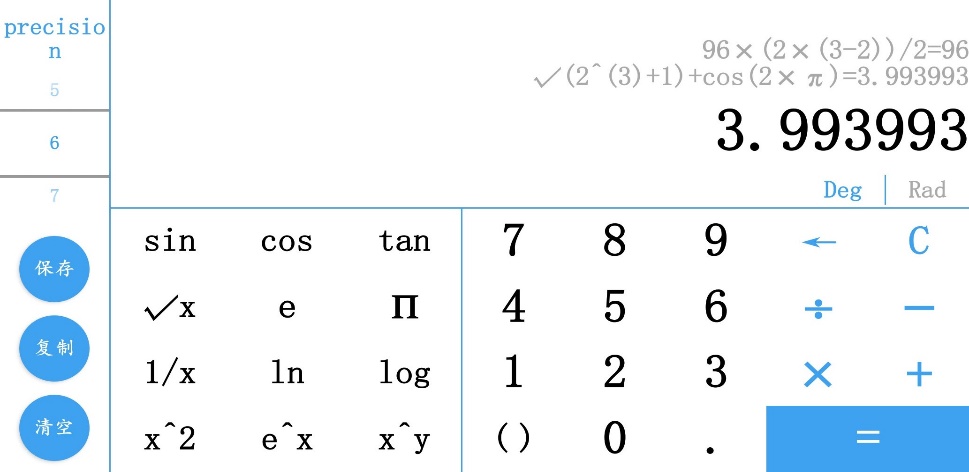




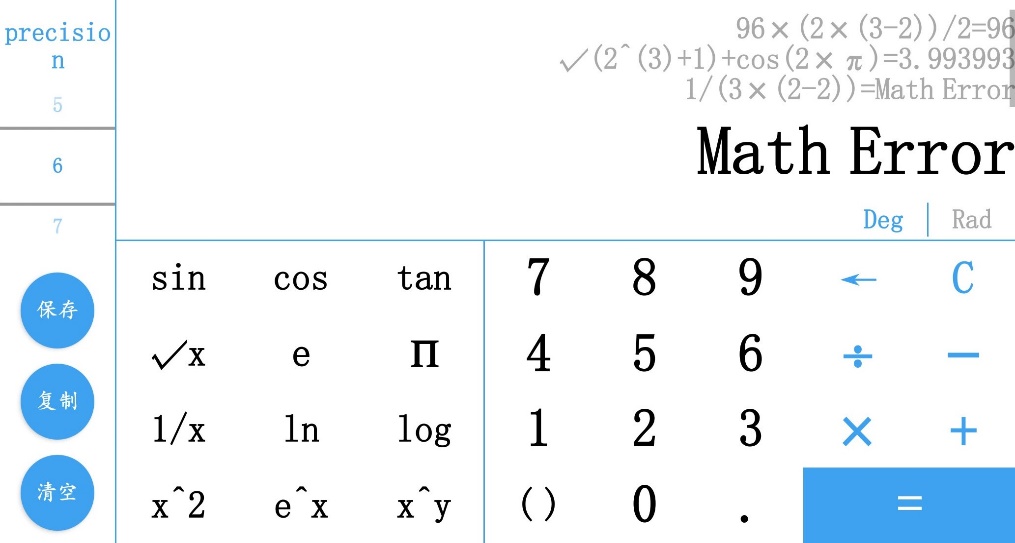
2、数学表达式



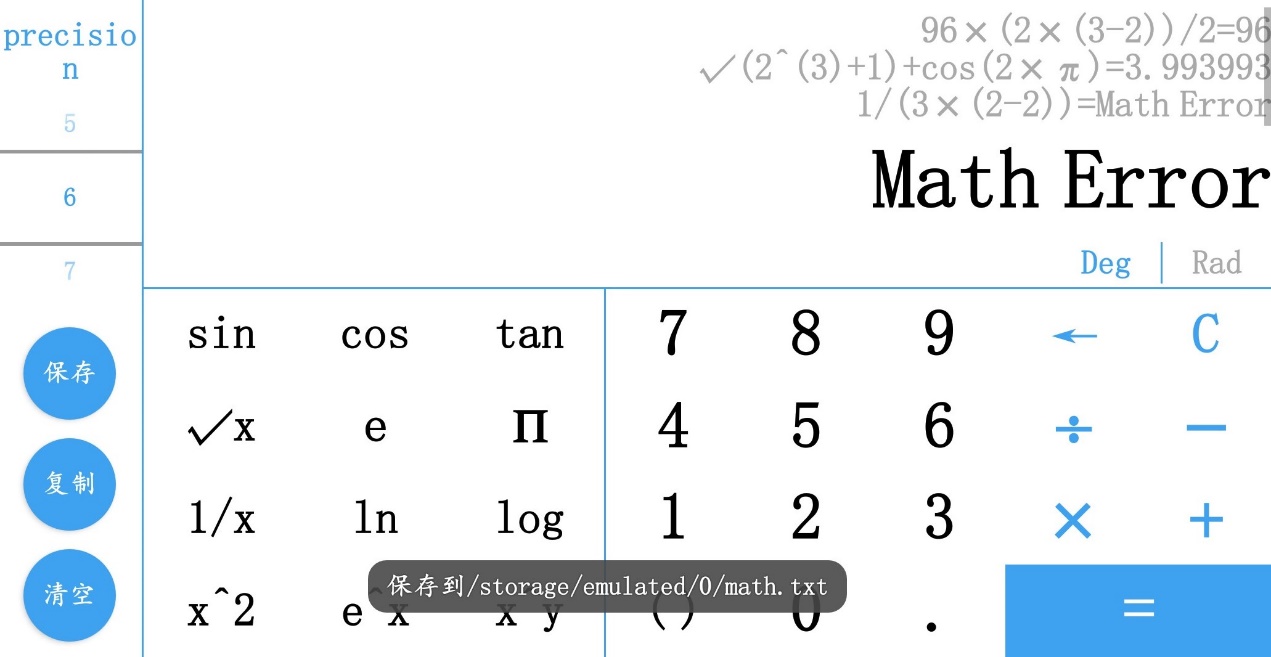
3、包含科学计算的数学表达式



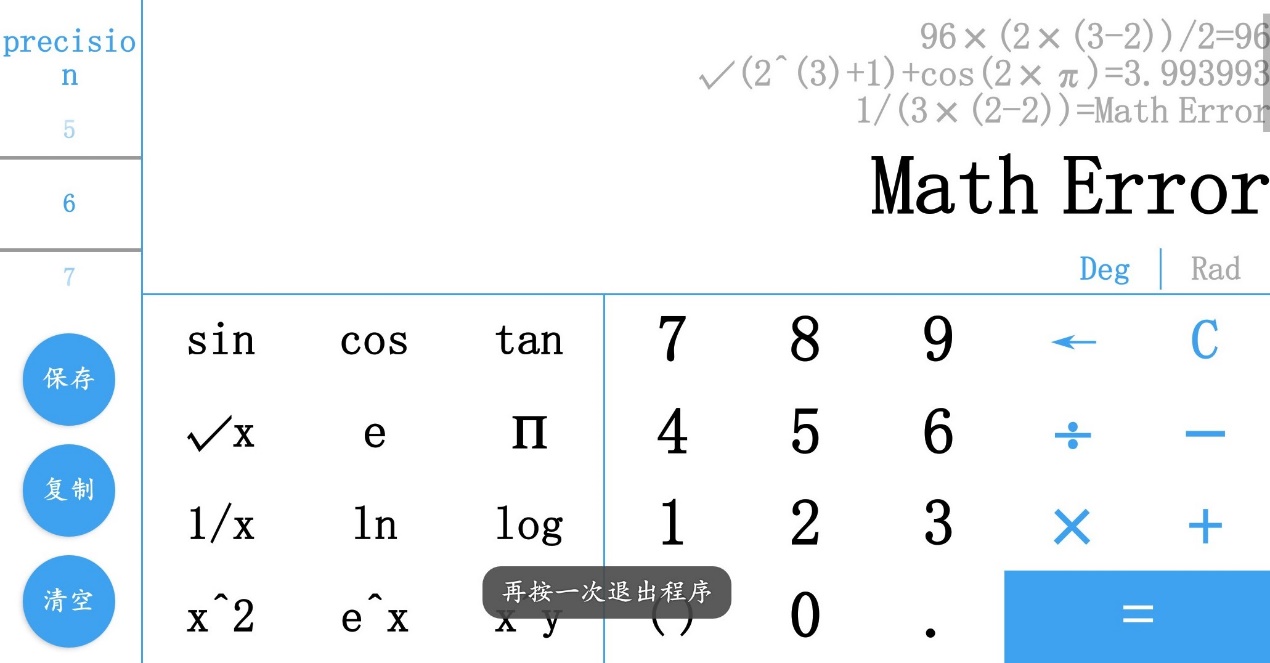
4、处理异常



5、保存运算过程到文件



6、再按一次退出程序



六、参考文献

1、《基于Android Studio的应用程序开发教程》

李宁宁，电子工业出版社，2016年07月

2、《第一行代码：Android（第2版）》

郭霖，人民邮电出版社，2016年11月

3、《Android Studio项目开发实战——从基础入门到趣味开发》

马静,邝楚文,肖国金，电子工业出版社，2020年3月

4、《[Android应用开发技术](https://www.hxedu.com.cn/hxedu/hg/book/bookInfo.html?code=G0306310)》

胡光永,查英华，电子工业出版社，2017年2月

5、《[Android技术及应用](https://www.hxedu.com.cn/hxedu/hg/book/bookInfo.html?code=G0280140)》

张军朝，电子工业出版社，2016年3月

七、设计过程中遇到的难点与解决方法

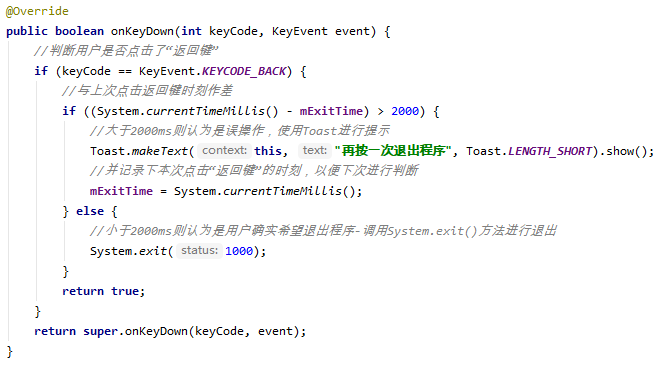
1、替换app图标

问题：在AndroidManifest.xml文件中直接将app图标的引用路径修改至目标图片，程序报错。

解决：打开项目主界面，任意打开一个类文件，点击File-->New-->Image Asset，在Path处选择要替换的图标，选好之后点击ok，点击Next，点击finish，完成后重新更改app图标的引用路径即可。

2、“再按一次退出程序”功能

解决：具体实现的关键是通过在需要退出页面的Activity里复写onKeyDown()方法实现功能逻辑 从而实现该功能。代码如下，



3、通过屏幕纵横切换来变换Activity

问题：屏幕纵横切换只能进行一次，且有时根本不切换，切换过一次后再一次横屏时依然在竖屏界面的布局，无法进入横屏界面布局。

解决：activity横竖屏切换时会销毁重建，重新调用一遍生命周期的方法，所以，我使用Java代码写死横竖屏，并且横竖屏切换时让系统不用处理。代码如下，



八、课程总结

本次实验不仅锻炼了我编写Java程序的能力，还使我对Android系统App设计有了更深的认识。用 Java 做计算器，主要是处理 String 类型的 math 表达式，灵活运用 String 的方法，通过截取原始的 math 分治结果问题：

1. 先预处理 math，去掉影响计算的空格等
2. 再替换 π，e
3. 再计算科学运算式
4. 最后把 math 替换成 BaseCalculator 即可计算的类型
5. 再利用栈实现四则运算的方法计算出最终结果

对于Android程序设计，我学会了以下几点：

1. Android横竖屏切换
2. 保存文件到手机本地
3. 灵活运用layout布局设计App界面，掌握了基本的自适应
4. 自定义控件如NumberPicker，Button边框等，会设计圆形的Button按钮
5. 制作“再按一次退出程序”功能

总的来说，本次实验我收获很多，基本上理解了编写一个 Java 应用的基本架构，应该先编写好接口，再设计界面，最后把响应事件与接口联系起来，做成一个计算器，但我也认识到计算器面临的 math 表达式的类型有很多，在 NumButtons 和 OperButtons 中添加的响应场景可能还不完善，所以我今后还会继续学习，不断完善我的知识短板，提高Android设计能力。

附录：程序主类源代码

1、AutoScaleTextView.java

**public class** AutoScaleTextView **extends** androidx.appcompat.widget.AppCompatTextView{  
 **private static float** *DEFAULT\_MIN\_TEXTSIZE* = 10;  
 **private static float** *DEFAULT\_MAX\_TEXTSIZE* = 20;  
  
 *//Attributes* **private** Paint **testPaint**;  
 **private float minTextSize**;  
 **private float maxTextSize**;  
  
 **public** AutoScaleTextView(Context context, AttributeSet attrs) {  
 **super**(context, attrs);  
 init();  
 }  
  
 **private void** init() {  
 **testPaint** = **new** Paint();  
 **testPaint**.set(**this**.getPaint());  
  
 **maxTextSize** = **this**.getTextSize();  
 **if** (**maxTextSize** <= *DEFAULT\_MIN\_TEXTSIZE*) {  
 **maxTextSize** = *DEFAULT\_MAX\_TEXTSIZE*;  
 }  
 **minTextSize** = *DEFAULT\_MIN\_TEXTSIZE*;  
 }  
  
 **private void** refitText(String text, **int** textWidth) {  
 **if** (textWidth > 0) {  
 **int** availableWidth = textWidth - **this**.getPaddingLeft() - **this**.getPaddingRight();  
 **float** trySize = **maxTextSize**;  
 **testPaint**.setTextSize(trySize);  
  
 *//调试代码1  
 //System.out.println("testPaint.measureText(text)=" + testPaint.measureText(text) \* 3);  
 //System.out.println("availableWidth=" + availableWidth);* **while** ((trySize > **minTextSize**) && (**testPaint**.measureText(text) \* 3 > availableWidth)) {  
 trySize -= 3;  
 **if** (trySize <= **minTextSize**) {  
 trySize = **minTextSize**;  
 **break**;  
 }  
 **testPaint**.setTextSize(trySize);  
 }  
 **this**.setTextSize(trySize);  
 }  
 }  
  
 @Override  
 **protected void** onTextChanged(CharSequence text, **int** start, **int** lengthBefore, **int** lengthAfter) {  
 **super**.onTextChanged(text, start, lengthBefore, lengthAfter);  
 refitText(text.toString(), **this**.getWidth());  
  
 *//调试代码1  
 //System.out.println("this.getWidth()=" + this.getWidth());* }  
  
 @Override  
 **protected void** onSizeChanged(**int** w, **int** h, **int** oldw, **int** oldh) {  
 **if** (w != oldw) {  
 refitText(**this**.getText().toString(), w);  
 }  
 }  
}

2、BaseCalculator.java

**public class** BaseCalculator {  
 **private final char**[] **operSet** = {**'+'**, **'-'**, **'×'**, **'/'**, **'('**, **')'**, **'#'**};  
  
 *//Map结构方便后面取运算符的下标* **private final** Map<Character, Integer> **operMap** = **new** HashMap<Character, Integer>() {{  
 put(**'+'**, 0);  
 put(**'-'**, 1);  
 put(**'×'**, 2);  
 put(**'/'**, 3);  
 put(**'('**, 4);  
 put(**')'**, 5);  
 put(**'#'**, 6);  
 }};  
  
 *//运算符优先级表，operPrior[oper1下标][oper2下标]* **private final char**[][] **operPrior** = {  
 */\* (o1,o2) + - × / ( ) # \*/  
 /\* + \*/* {**'>'**, **'>'**, **'<'**, **'<'**, **'<'**, **'>'**, **'>'**},  
 */\* - \*/* {**'>'**, **'>'**, **'<'**, **'<'**, **'<'**, **'>'**, **'>'**},  
 */\* × \*/* {**'>'**, **'>'**, **'>'**, **'>'**, **'<'**, **'>'**, **'>'**},  
 */\* / \*/* {**'>'**, **'>'**, **'>'**, **'>'**, **'<'**, **'>'**, **'>'**},  
 */\* ( \*/* {**'<'**, **'<'**, **'<'**, **'<'**, **'<'**, **'='**, **' '**},  
 */\* ) \*/* {**'>'**, **'>'**, **'>'**, **'>'**, **' '**, **'>'**, **'>'**},  
 */\* # \*/* {**'<'**, **'<'**, **'<'**, **'<'**, **'<'**, **' '**, **'='**},  
 };  
  
 *//返回2个运算符优先级比较的结果'<','=','>'* **private char** getPrior(**char** oper1, **char** oper2) {  
 **return operPrior**[**operMap**.get(oper1)][**operMap**.get(oper2)]; *//Map.get方法获取运算符的下标* }  
  
 *//简单四则运算* **private double** operate(**double** a, **char** oper, **double** b) {  
 **switch** (oper) {  
 **case '+'**:  
 **return** a + b;  
 **case '-'**:  
 **return** a - b;  
 **case '×'**:  
 **return** a \* b;  
 **case '/'**:  
 **if** (b == 0) {  
 **return** Double.***MAX\_VALUE***; *//处理异常* }  
 **return** a / b;  
 **default**:  
 **return** 0;  
 }  
 }  
  
 *//计算普通的运算式* **private double** calSubmath(String math) {  
 **if** (math.length() == 0) {  
 **return** Double.***MAX\_VALUE***;  
 } **else** {  
 **if** (!hasOper(math.substring(1, math.length())) || math.contains(**"E-"**)) {  
 **return** Double.*parseDouble*(math);  
 }  
  
 *//设置flag用于存储math开始位置的负数，如-3-5中的-3，避免-被识别成运算符而出错* **int** flag = 0;  
 **if** (math.charAt(0) == **'-'**) {  
 flag = 1;  
 math = math.substring(1, math.length());  
 }  
  
 Stack<Character> operStack = **new** Stack<>(); *//oper栈* Stack<Double> numStack = **new** Stack<>(); *//num栈* operStack.push(**'#'**); *//设置栈底元素* math += **"#"**;  
  
 String tempNum = **""**; *//暂存数字str  
  
 //计算math* **for** (**int** i = 0; i < math.length(); i++) {  
  
 **char** charOfMath = math.charAt(i); *//遍历math中的char  
  
 //(1)num进栈* **if** (!isOper(charOfMath) *//1.不是oper* || charOfMath == **'-'** && math.charAt(i - 1) == **'('**) { *//2.是'-'并且'-'左边有'('，说明是在math中间用负数* tempNum += charOfMath;  
  
 *//1.1 获取下一个char* i++;  
 charOfMath = math.charAt(i);  
  
 *//1.2 判断下一个char是不是oper,如果是oper，就将num压入numStack* **if** (isOper(charOfMath)) { *//此条件成功时，下次for循环就直接跳到else语句了* **double** num = Double.*parseDouble*(tempNum);  
 **if** (flag == 1) { *//恢复math首位的负数* num = -num;  
 flag = 0;  
 }  
 numStack.push(num); *//push num* tempNum = **""**; *//重置tempNum* }  
  
 *//1.3 //回退，以免下次循环for语句自身的i++使得跳过了这个char* i--;  
 }  
  
 *//(2)oper进栈* **else** {  
  
 **switch** (getPrior(operStack.peek(), charOfMath)) {  
  
 *//2.1 栈顶oper优先级低，新oper入栈* **case '<'**:  
 operStack.push(charOfMath);  
 **break**;  
  
 *//2.2 说明当前的charOfMath为')'，而栈顶oper为'('，去掉'('，其实也是math去括号的过程* **case '='**:  
 operStack.pop();  
 **break**;  
  
 *//2.3 栈顶oper优先级高，oper出栈，并将num运算结果push进numStack* **case '>'**:  
 **char** oper = operStack.pop();  
 **double** b = numStack.pop();  
 **double** a = numStack.pop();  
 **if** (operate(a, oper, b) == Double.***MAX\_VALUE***)  
 **return** Double.***MAX\_VALUE***;  
 numStack.push(operate(a, oper, b));  
 i--; *//继续比较该oper与栈顶oper的关系* **break**;  
 }  
 }  
 }  
 **return** numStack.peek(); *//最后的math变成一个num了* }  
 }  
  
 *//计算math，添加了一些特殊math的处理* **double** cal(String math) {  
 **if** (math.length() == 0) { *//处理异常* **return** Double.***MAX\_VALUE***;  
 } **else** {  
 *//运算式只是数字的特征：从第2个char开始math中没有oper* **if** (!hasOper(math.substring(1, math.length())) || math.contains(**"E-"**)) {  
 **return** Double.*parseDouble*(math);  
 }  
 *//普通运算* **else** {  
 **return** calSubmath(math);  
 }  
 }  
 }  
  
 *//判断String中是否有运算符* **private boolean** hasOper(String s) {  
 **return** s.contains(**"+"**) || s.contains(**"-"**) || s.contains(**"×"**) || s.contains(**"/"**);  
 }  
  
 *//判断字符是否为运算符* **private boolean** isOper(**char** c) {  
 **int** i;  
 **for** (i = 0; i < **operSet**.**length**; i++) {  
 **if** (c == **operSet**[i]) {  
 **break**;  
 }  
 }  
 *//break出来，说明是oper，i != operSize* **return** i != **operSet**.**length**;  
 }  
}

3、LandActivity.java

**public class** LandActivity **extends** AppCompatActivity {  
  
 *//声明一个long类型变量：用于存放上一点击“返回键”的时刻* **private long mExitTime**;  
  
 *//choose precision, extends NumberPicker* **private** MyNumberPicker **precisionPicker**;  
  
 *//tvNow,tvPast to show math expression  
 //tvRad,tvDeg to switch between Deg and Rad* **private** TextView **tvPast**;  
 **private** AutoScaleTextView **tvNow**; *//extends textView to auto scale the width and decrease the font size* **private** TextView **tvDeg**;  
 **private** TextView **tvRad**;  
  
 *//to save, copy and clear past math expressions* **private** Button **btnSave**;  
 **private** Button **btnCopy**;  
 **private** Button **btnClear**;  
  
 *//common number* **private** Button **btn7**;  
 **private** Button **btn8**;  
 **private** Button **btn9**;  
 **private** Button **btn4**;  
 **private** Button **btn5**;  
 **private** Button **btn6**;  
 **private** Button **btn1**;  
 **private** Button **btn2**;  
 **private** Button **btn3**;  
 **private** Button **btn0**;  
 **private** Button **btnDot**;  
  
 *//basic opers: + - × / ( ) =* **private** Button **btnAdd**;  
 **private** Button **btnSub**;  
 **private** Button **btnMul**;  
 **private** Button **btnDiv**;  
 **private** Button **btnBracket**;  
 **private** Button **btnEqual**;  
  
 *//scienceCalculator calculation  
 // sin cos tan  
 // √x e π  
 // 1/x ln log  
 // x^2 e^x x^y* **private** Button **btnSin**;  
 **private** Button **btnCos**;  
 **private** Button **btnTan**;  
 **private** Button **btnSqrt**;  
 **private** Button **btnE**;  
 **private** Button **btnPi**;  
 **private** Button **btn1x**;  
 **private** Button **btnLn**;  
 **private** Button **btnLog**;  
 **private** Button **btnX2**;  
 **private** Button **btnEx**;  
 **private** Button **btnXy**;  
  
 *//basic calculator function: delete a char,clear the current math* **private** Button **btnDel**;  
 **private** Button **btnClc**;  
  
 **private** String **mathPast** = **""**;  
 **private** String **mathNow** = **""**;  
 **private int precision** = 6;  
 **private int equal\_flag** = 0;  
  
 *//加入角度运算* **private final int DEG** = 0;  
 **private final int RAD** = 1;  
 **private int angle\_metric** = **DEG**; *//默认的度量为角度DEG  
  
 //一个科学计算器* **private** ScienceCalculator **scienceCalculator** = **new** ScienceCalculator();  
  
 @Override  
 **public void** onConfigurationChanged(Configuration newConfig) {  
 **super**.onConfigurationChanged(newConfig);  
  
 **if** (newConfig.**orientation** == Configuration.***ORIENTATION\_PORTRAIT***) {  
 Intent intent = **new** Intent(LandActivity.**this**, MainActivity.**class**);  
 intent.putExtra(**"land"**, **tvPast**.getText().toString());  
 startActivity(intent);  
 finish();  
 }  
 }  
  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
  
 *//隐藏状态栏* getWindow().setFlags(WindowManager.LayoutParams.***FLAG\_FULLSCREEN***, WindowManager.LayoutParams.***FLAG\_FULLSCREEN***);  
 setContentView(R.layout.***activity\_land***);  
  
 **precisionPicker** = findViewById(R.id.***precisionPicker***);  
  
 **tvPast** = findViewById(R.id.***tv\_past***);  
 **tvNow** = findViewById(R.id.***tv\_now***);  
 **tvDeg** = findViewById(R.id.***tv\_deg***);  
 **tvRad** = findViewById(R.id.***tv\_rad***);  
  
 **btnSave** = findViewById(R.id.***btn\_save***);  
 **btnCopy** = findViewById(R.id.***btn\_copy***);  
 **btnClear** = findViewById(R.id.***btn\_clear***);  
  
 **btn7** = findViewById(R.id.***btn\_7***);  
 **btn8** = findViewById(R.id.***btn\_8***);  
 **btn9** = findViewById(R.id.***btn\_9***);  
 **btn4** = findViewById(R.id.***btn\_4***);  
 **btn5** = findViewById(R.id.***btn\_5***);  
 **btn6** = findViewById(R.id.***btn\_6***);  
 **btn1** = findViewById(R.id.***btn\_1***);  
 **btn2** = findViewById(R.id.***btn\_2***);  
 **btn3** = findViewById(R.id.***btn\_3***);  
 **btn0** = findViewById(R.id.***btn\_0***);  
 **btnDot** = findViewById(R.id.***btn\_dot***);  
  
 **btnAdd** = findViewById(R.id.***btn\_add***);  
 **btnSub** = findViewById(R.id.***btn\_sub***);  
 **btnMul** = findViewById(R.id.***btn\_mul***);  
 **btnDiv** = findViewById(R.id.***btn\_div***);  
 **btnBracket** = findViewById(R.id.***btn\_bracket***);  
 **btnEqual** = findViewById(R.id.***btn\_equal***);  
  
 **btnSin** = findViewById(R.id.***btn\_sin***);  
 **btnCos** = findViewById(R.id.***btn\_cos***);  
 **btnTan** = findViewById(R.id.***btn\_tan***);  
 **btnSqrt** = findViewById(R.id.***btn\_sqrt***);  
 **btnE** = findViewById(R.id.***btn\_e***);  
 **btnPi** = findViewById(R.id.***btn\_pi***);  
 **btn1x** = findViewById(R.id.***btn\_1x***);  
 **btnLn** = findViewById(R.id.***btn\_ln***);  
 **btnLog** = findViewById(R.id.***btn\_log***);  
 **btnX2** = findViewById(R.id.***btn\_x2***);  
 **btnEx** = findViewById(R.id.***btn\_ex***);  
 **btnXy** = findViewById(R.id.***btn\_xy***);  
  
 **btnDel** = findViewById(R.id.***btn\_del***);  
 **btnClc** = findViewById(R.id.***btn\_clc***);  
  
 *//设置控件属性* initTvPast();  
 initNumBtns();  
 initBaseOpers();  
 initScienceOpers();  
 initDegRad();  
 initPrecisionPicker();  
 initThreeFunctions();  
 initBaseCalculatorFunction();  
 }  
  
 *//初始化tvPast* **public void** initTvPast() {  
  
 *//设置tvPast一些属性* **tvPast**.setMovementMethod(ScrollingMovementMethod.*getInstance*()); *//内容自动滚动到最新的一行* **tvPast**.setTextIsSelectable(**true**); *//长按复制  
  
 //获取界面切换的tvPast的内容* Intent intent = **this**.getIntent();  
 String tvPastContent = intent.getStringExtra(**"main"**);  
  
 *//如果当前的界面是启动界面，不是从MainActivity切换来的，上面的mathPast就为null了，要处理这种异常* **if** (tvPastContent == **null**) {  
 **tvPast**.setText(**""**);  
 } **else** {  
 String[] maths = tvPastContent.split(**"\n"**);  
 **int** i;  
 **for** (i = 0; i < maths.**length** - 1; i++) {  
 **tvPast**.append(maths[i] + **"\n"**);  
 }  
 **tvPast**.append(maths[i]); *//最后一个math不用加换行* }  
  
 }  
  
 *//初始化数字btns* **public void** initNumBtns() {  
  
 **btn0**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 *//如果flag=1，表示要输入新的运算式，清空mathNow并设置flag=0* **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
  
 **if** (**mathNow**.length() == 0) { *//1.mathNow为空，+0* **mathNow** += **"0"**;  
 } **else if** (**mathNow**.length() == 1) { *//2.mathNow 长度为1* **if** (**mathNow**.charAt(0) == **'0'**) { *//2.1 如果该字符为0，不加* **mathNow** += **""**;  
 } **else if** (isNum(**mathNow**.charAt(0))) { *//2.2 如果该字符为1-9，+0* **mathNow** += **"0"**;  
 }  
  
 } **else if** (!isNum(**mathNow**.charAt(**mathNow**.length() - 2)) && **mathNow**.charAt(**mathNow**.length() - 1) == **'0'**) {  
 **mathNow** += **""**; *//3.属于2.1的一般情况，在math中间出现 比如：×0 +0* } **else** { *//4.除此之外，+0* **mathNow** += **"0"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 *//btn 1-9 输入条件相同* **btn1**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"1"**;  
 } **else** {  
  
 *//math的最后一个字符是：1-9, oper, (, .* **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"1"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn2**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"2"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"2"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn3**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"3"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"3"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn4**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"4"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"4"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn5**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"5"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"5"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn6**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"6"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"6"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn7**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"7"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"7"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn8**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"8"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"8"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn9**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"9"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"9"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnDot**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) { *//1.mathNow为空，+0.* **mathNow** += **"0."**;  
 } **else if** (isOper(**mathNow**.charAt(**mathNow**.length() - 1))) { *//2.mathNow的最后一个字符为oper，+0.* **mathNow** += **"0."**;  
 } **else if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))) { *//3.mathNow的最后一个字符为num，+.* **mathNow** += **"."**;  
 } **else** { *//4.除此之外，不加* **mathNow** += **""**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
 }  
  
 *//初始化基本的运算符* **public void** initBaseOpers() {  
  
 **btnAdd**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"+"**;  
 } **else** {  
 **if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))  
 || **mathNow**.charAt(**mathNow**.length() - 1) == **')'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'('** || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**)  
 **mathNow** += **"+"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 **equal\_flag** = 0; *//可能用运算结果直接运算，flag直接设0* }  
 });  
  
 **btnSub**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"-"**;  
 } **else** {  
 **if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))  
 || **mathNow**.charAt(**mathNow**.length() - 1) == **')'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'('** || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**)  
 **mathNow** += **"-"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 **equal\_flag** = 0;  
 }  
 });  
  
 **btnMul**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() != 0) {  
 **if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))  
 || **mathNow**.charAt(**mathNow**.length() - 1) == **')'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**)  
 **mathNow** += **"×"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 **equal\_flag** = 0;  
 }  
 });  
  
 **btnDiv**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() != 0) {  
 **if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))  
 || **mathNow**.charAt(**mathNow**.length() - 1) == **')'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**)  
 **mathNow** += **"/"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 **equal\_flag** = 0;  
 }  
 });  
  
 **btnBracket**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) { *//1.mathNow为空，+(* **mathNow** += **"("**;  
 } **else if** (isOper(**mathNow**.charAt(**mathNow**.length() - 1))) { *//2.mathNow最后一个字符是oper，+(* **mathNow** += **"("**;  
 } **else if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1)) *//3.mathNow最后一个字符是num, π, e* || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**) {  
 **if** (!hasLeftBracket(**mathNow**)) *//3.1 没有(, 加 ×(* **mathNow** += **"×("**;  
 **else** *//3.2 已有(, 加 )* **mathNow** += **")"**;  
 } **else if** (**mathNow**.charAt(**mathNow**.length() - 1) == **')'**) { *//4.mathNow最后一个字符是)，说明用户是在补全右括号，+)* **mathNow** += **')'**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
  
 **btnEqual**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 *//右括号自动补全* **int** leftNum = 0;  
 **int** rightNum = 0;  
 **for** (**int** i = 0; i < **mathNow**.length(); i++) {  
 **if** (**mathNow**.charAt(i) == **'('**)  
 leftNum++;  
 **if** (**mathNow**.charAt(i) == **')'**)  
 rightNum++;  
 }  
 **int** missingNum = leftNum - rightNum; *//缺失的 ) 数量* **while** (missingNum > 0) {  
 **mathNow** += **')'**;  
 missingNum--;  
 }  
 **tvNow**.setText(**mathNow**);  
  
 **mathPast** = **"\n"** + **mathNow**; *//使得呈现的mathPast自动换行* **double** result = **scienceCalculator**.cal(**mathNow**, **precision**, **angle\_metric**); *//调用科学计算器* **if** (result == Double.***MAX\_VALUE***)  
 **mathNow** = **"Math Error"**;  
 **else** {  
 **mathNow** = String.*valueOf*(result);  
 System.***out***.println(**mathNow**);  
 **if** (**mathNow**.charAt(**mathNow**.length() - 2) == **'.'** && **mathNow**.charAt(**mathNow**.length() - 1) == **'0'**) {  
 **mathNow** = **mathNow**.substring(0, **mathNow**.length() - 2);  
 }  
 }  
  
 **mathPast** = **mathPast** + **"="** + **mathNow**;  
  
 *//用tvPast.set(mathPast)不能实现自动滚动到最新运算过程* **tvPast**.append(**mathPast**); *//添加新的运算过程  
  
 //tvPast滚动到最新的运算过程* **int** offset = **tvPast**.getLineCount() \* **tvPast**.getLineHeight();  
 **if** (offset > **tvPast**.getHeight()) {  
 **tvPast**.scrollTo(0, offset - **tvPast**.getHeight());  
 }  
 **tvNow**.setText(**mathNow**);  
  
 **equal\_flag** = 1; *//设置flag=1* }  
 });  
 }  
  
 *//初始化科学运算符* **public void** initScienceOpers() {  
 **btnSin**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"sin("**;  
 } **else** {  
 *//oper, (, 加 sin(* **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"sin("**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnCos**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"cos("**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"cos("**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnTan**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"tan("**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"tan("**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnSqrt**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"√("**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"√("**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
  
 **btnPi**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"π"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"π"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnE**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"e"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"e"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnLn**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"ln("**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"ln("**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnLog**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"log("**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"log("**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn1x**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"1/"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"1/"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnEx**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"e^("**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isOper(ch) || ch == **'('**)  
 **mathNow** += **"e^("**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnX2**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 *//要求mathNow不为空并且最后一个字符：num, ), e, π* **if** (**mathNow**.length() > 0) {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || ch == **')'** || ch == **'e'** || ch == **'π'**)  
 **mathNow** += **"^2"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnXy**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 *//条件同btnX2* **if** (**mathNow**.length() > 0) {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || ch == **')'** || ch == **'e'** || ch == **'π'**)  
 **mathNow** += **"^("**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 }  
  
 *//初始化tvDeg,tvRad* **public void** initDegRad() {  
 **tvDeg**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **tvDeg**.setTextColor(Color.*parseColor*(**"#3FA2F0"**));  
 **tvRad**.setTextColor(Color.*parseColor*(**"#AAAAAA"**));  
 **angle\_metric** = **DEG**;  
 }  
 });  
  
 **tvRad**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **tvRad**.setTextColor(Color.*parseColor*(**"#3FA2F0"**));  
 **tvDeg**.setTextColor(Color.*parseColor*(**"#AAAAAA"**));  
 **angle\_metric** = **RAD**;  
 }  
 });  
 }  
  
 *//初始化精度选择器* **public void** initPrecisionPicker() {  
 **precisionPicker**.setMaxValue(12); *//最多保留12位* **precisionPicker**.setMinValue(0);  
 **precisionPicker**.setValue(6);  
 **precisionPicker**.setOnValueChangedListener(**new** NumberPicker.OnValueChangeListener() {  
 @Override  
 **public void** onValueChange(NumberPicker numberPicker, **int** oldVal, **int** newVal) {  
 **precision** = newVal;  
 }  
 });  
 }  
  
 *//保存，复制，清空* **public void** initThreeFunctions() {  
 *//保存* **btnSave**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 *//保存文件到sd卡 manifest文件中也要添加2个permission* **if** (Environment.*getExternalStorageState*().equals(Environment.***MEDIA\_MOUNTED***)) {  
 String path = Environment.*getExternalStorageDirectory*().getPath() + **"/math.txt"**; *//设置保存路径和文件名* **try** {  
 FileOutputStream outputStream = **new** FileOutputStream(path);  
 outputStream.write(**tvPast**.getText().toString().getBytes()); *//写字节* outputStream.close(); *//关闭输出流* } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 Toast.*makeText*(LandActivity.**this**, **"保存到"** + path, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
  
 *//复制* **btnCopy**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 ClipboardManager cm = (ClipboardManager) getSystemService(Context.***CLIPBOARD\_SERVICE***); *//采用ClipboardManager类* cm.setText(**tvPast**.getText());  
 Toast.*makeText*(LandActivity.**this**, **"已复制到剪切板"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
  
 *//清空* **btnClear**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **mathPast** = **""**;  
 **tvPast**.setText(**mathPast**);  
 Toast.*makeText*(LandActivity.**this**, **"计算过程已经清空"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
 }  
  
 *//初始化计算器器基本Button* **public void** initBaseCalculatorFunction() {  
 **btnClc**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **mathNow** = **""**;  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnDel**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() != 0) {  
 **mathNow** = **mathNow**.substring(0, **mathNow**.length() - 1);  
 **tvNow**.setText(**mathNow**);  
 }  
 }  
 });  
 }  
  
 *//判断当前字符是否为数字* **private boolean** isNum(**char** c) {  
 **char** num[] = {**'0'**, **'1'**, **'2'**, **'3'**, **'4'**, **'5'**, **'6'**, **'7'**, **'8'**, **'9'**};  
 **int** i = 0;  
 **for** (; i < num.**length**; i++) {  
 **if** (num[i] == c)  
 **break**;  
 }  
 **return** i != num.**length**;  
 }  
  
 *//判断当前字符是否为运算符* **private boolean** isOper(**char** c) {  
 **char** oper[] = {**'+'**, **'-'**, **'×'**, **'/'**};  
 **int** i = 0;  
 **for** (; i < oper.**length**; i++) {  
 **if** (oper[i] == c)  
 **break**;  
 }  
 **return** i != oper.**length**;  
 }  
  
 *//判断当前math是否有')'* **private boolean** hasLeftBracket(String s) {  
 **int** i = 0;  
 **for** (; i < s.length(); i++) {  
 **if** (s.charAt(i) == **'('**)  
 **break**;  
 }  
 **return** i != s.length();  
 }  
  
 @Override  
 **public boolean** onKeyDown(**int** keyCode, KeyEvent event) {  
 *//判断用户是否点击了“返回键”* **if** (keyCode == KeyEvent.***KEYCODE\_BACK***) {  
 *//与上次点击返回键时刻作差* **if** ((System.*currentTimeMillis*() - **mExitTime**) > 2000) {  
 *//大于2000ms则认为是误操作，使用Toast进行提示* Toast.*makeText*(**this**, **"再按一次退出程序"**, Toast.***LENGTH\_SHORT***).show();  
 *//并记录下本次点击“返回键”的时刻，以便下次进行判断* **mExitTime** = System.*currentTimeMillis*();  
 } **else** {  
 *//小于2000ms则认为是用户确实希望退出程序-调用System.exit()方法进行退出* System.*exit*(1000);  
 }  
 **return true**;  
 }  
 **return super**.onKeyDown(keyCode, event);  
 }  
}

4、MainActivity.java

**public class** MainActivity **extends** AppCompatActivity {  
  
 *//声明一个long类型变量：用于存放上一点击“返回键”的时刻* **private long mExitTime**;  
  
 *//tvNow,tvPast显示运算过程* **private** TextView **tvPast**;  
 **private** AutoScaleTextView **tvNow**; *//继承TextView自适应屏幕宽度设置字体大小  
  
 //保存，复制，清空tvPast中的运算过程* **private** Button **btnSave**;  
 **private** Button **btnCopy**;  
 **private** Button **btnClear**;  
  
 *//数字btns* **private** Button **btn7**;  
 **private** Button **btn8**;  
 **private** Button **btn9**;  
 **private** Button **btn4**;  
 **private** Button **btn5**;  
 **private** Button **btn6**;  
 **private** Button **btn1**;  
 **private** Button **btn2**;  
 **private** Button **btn3**;  
 **private** Button **btn0**;  
 **private** Button **btnDot**;  
  
 *//基本运算符: + - × / ( ) =* **private** Button **btnAdd**;  
 **private** Button **btnSub**;  
 **private** Button **btnMul**;  
 **private** Button **btnDiv**;  
 **private** Button **btnBracket**;  
 **private** Button **btnEqual**;  
  
 *//基本计算器功能：删除一个字符，清空当前math* **private** Button **btnDel**;  
 **private** Button **btnClc**;  
  
 *//存储过去和当前的math* **private** String **mathPast** = **""**;  
 **private** String **mathNow** = **""**;  
 **private int precision** = 6;  
 **private int equal\_flag** = 0; *//设置flag值判断是否需要清空mathNow进行新的运算* **private** ScienceCalculator **scienceCalculator** = **new** ScienceCalculator(); *//一个科学计算器  
  
  
 //监听设备当前朝向* @Override  
 **public void** onConfigurationChanged(Configuration newConfig) {  
 **super**.onConfigurationChanged(newConfig);  
  
 *//如果为landscape，切换activity* **if** (newConfig.**orientation** == Configuration.***ORIENTATION\_LANDSCAPE***) {  
 Intent intent = **new** Intent(MainActivity.**this**, LandActivity.**class**);  
 intent.putExtra(**"main"**, **tvPast**.getText().toString()); *//把当前的tvPast中的运算过程传递给LandActivity* startActivity(intent);  
 finish(); *//退出当前的activity* }  
 }  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
  
 *//隐藏状态栏 manifest的application中要设置appTheme为NoActionBar* getWindow().setFlags(WindowManager.LayoutParams.***FLAG\_FULLSCREEN***, WindowManager.LayoutParams.***FLAG\_FULLSCREEN***);  
 setContentView(R.layout.***activity\_main***);  
 **tvPast** = findViewById(R.id.***tv\_past***);  
 **tvNow** = findViewById(R.id.***tv\_now***); *//初始化控件* **btnSave** = findViewById(R.id.***btn\_save***);  
 **btnCopy** = findViewById(R.id.***btn\_copy***);  
 **btnClear** = findViewById(R.id.***btn\_clear***);  
  
 **btn7** = findViewById(R.id.***btn\_7***);  
 **btn8** = findViewById(R.id.***btn\_8***);  
 **btn9** = findViewById(R.id.***btn\_9***);  
 **btn4** = findViewById(R.id.***btn\_4***);  
 **btn5** = findViewById(R.id.***btn\_5***);  
 **btn6** = findViewById(R.id.***btn\_6***);  
 **btn1** = findViewById(R.id.***btn\_1***);  
 **btn2** = findViewById(R.id.***btn\_2***);  
 **btn3** = findViewById(R.id.***btn\_3***);  
 **btn0** = findViewById(R.id.***btn\_0***);  
 **btnDot** = findViewById(R.id.***btn\_dot***);  
  
 **btnAdd** = findViewById(R.id.***btn\_add***);  
 **btnSub** = findViewById(R.id.***btn\_sub***);  
 **btnMul** = findViewById(R.id.***btn\_mul***);  
 **btnDiv** = findViewById(R.id.***btn\_div***);  
 **btnBracket** = findViewById(R.id.***btn\_bracket***);  
 **btnEqual** = findViewById(R.id.***btn\_equal***);  
  
 **btnDel** = findViewById(R.id.***btn\_del***);  
 **btnClc** = findViewById(R.id.***btn\_clc***);  
  
 *//设置控件属性* initTvPast();  
 initNumBtns();  
 initBaseOpers();  
 initThreeFunctions();  
 initBaseCalculatorFunction();  
 }  
  
 *//初始化tvPast* **public void** initTvPast() {  
  
 *//设置tvPast一些属性* **tvPast**.setMovementMethod(ScrollingMovementMethod.*getInstance*()); *//内容自动滚动到最新的一行* **tvPast**.setTextIsSelectable(**true**); *//长按复制  
  
 //获取界面切换的tvPast的内容* Intent intent = **this**.getIntent();  
 String tvPastContent = intent.getStringExtra(**"land"**);  
  
 *//如果当前的界面是启动界面，不是从MainActivity切换来的，上面的mathPast就为null了，要处理这种异常* **if** (tvPastContent == **null**) {  
 **tvPast**.setText(**""**);  
 } **else** {  
 String[] maths = tvPastContent.split(**"\n"**);  
 **int** i;  
 **for** (i = 0; i < maths.**length** - 1; i++) {  
 **tvPast**.append(maths[i] + **"\n"**);  
 }  
 **tvPast**.append(maths[i]); *//最后一个math不用加换行* }  
  
 }  
  
  
 *//初始化数字btns* **public void** initNumBtns() {  
  
 **btn0**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 *//如果flag=1，表示要输入新的运算式，清空mathNow并设置flag=0* **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
  
 **if** (**mathNow**.length() == 0) { *//1.mathNow为空，+0* **mathNow** += **"0"**;  
 } **else if** (**mathNow**.length() == 1) { *//2.mathNow 长度为1* **if** (**mathNow**.charAt(0) == **'0'**) { *//2.1 如果该字符为0，不加* **mathNow** += **""**;  
 } **else if** (isNum(**mathNow**.charAt(0))) { *//2.2 如果该字符为1-9，+0* **mathNow** += **"0"**;  
 }  
  
 } **else if** (!isNum(**mathNow**.charAt(**mathNow**.length() - 2)) && **mathNow**.charAt(**mathNow**.length() - 1) == **'0'**) {  
 **mathNow** += **""**; *//3.属于2.1的一般情况，在math中间出现 比如：×0 +0* } **else** { *//4.除此之外，+0* **mathNow** += **"0"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 *//btn 1-9 输入条件相同* **btn1**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"1"**;  
 } **else** {  
  
 *//math的最后一个字符是：1-9, oper, (, .* **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"1"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn2**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"2"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"2"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn3**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"3"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"3"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn4**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"4"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"4"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn5**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"5"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"5"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn6**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"6"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"6"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn7**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"7"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"7"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn8**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"8"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"8"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btn9**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"9"**;  
 } **else** {  
 **char** ch = **mathNow**.charAt(**mathNow**.length() - 1);  
 **if** (isNum(ch) || isOper(ch) || ch == **'('** || ch == **'.'**)  
 **mathNow** += **"9"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnDot**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) { *//1.mathNow为空，+0.* **mathNow** += **"0."**;  
 } **else if** (isOper(**mathNow**.charAt(**mathNow**.length() - 1))) { *//2.mathNow的最后一个字符为oper，+0.* **mathNow** += **"0."**;  
 } **else if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))) { *//3.mathNow的最后一个字符为num，+.* **mathNow** += **"."**;  
 } **else** { *//4.除此之外，不加* **mathNow** += **""**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
 }  
  
 *//初始化基本的运算符* **public void** initBaseOpers() {  
  
 **btnAdd**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"+"**;  
 } **else** {  
 **if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))  
 || **mathNow**.charAt(**mathNow**.length() - 1) == **')'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'('** || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**)  
 **mathNow** += **"+"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 **equal\_flag** = 0; *//可能用运算结果直接运算，flag直接设0* }  
 });  
  
 **btnSub**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() == 0) {  
 **mathNow** += **"-"**;  
 } **else** {  
 **if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))  
 || **mathNow**.charAt(**mathNow**.length() - 1) == **')'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'('** || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**)  
 **mathNow** += **"-"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 **equal\_flag** = 0;  
 }  
 });  
  
 **btnMul**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() != 0) {  
 **if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))  
 || **mathNow**.charAt(**mathNow**.length() - 1) == **')'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**)  
 **mathNow** += **"×"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 **equal\_flag** = 0;  
 }  
 });  
  
 **btnDiv**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() != 0) {  
 **if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1))  
 || **mathNow**.charAt(**mathNow**.length() - 1) == **')'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**)  
 **mathNow** += **"/"**;  
 }  
 **tvNow**.setText(**mathNow**);  
 **equal\_flag** = 0;  
 }  
 });  
  
  
 **btnBracket**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**equal\_flag** == 1) {  
 **mathNow** = **""**;  
 **equal\_flag** = 0;  
 }  
 **if** (**mathNow**.length() == 0) { *//1.mathNow为空，+(* **mathNow** += **"("**;  
 } **else if** (isOper(**mathNow**.charAt(**mathNow**.length() - 1))) { *//2.mathNow最后一个字符是oper，+(* **mathNow** += **"("**;  
 } **else if** (isNum(**mathNow**.charAt(**mathNow**.length() - 1)) *//3.mathNow最后一个字符是num, π, e* || **mathNow**.charAt(**mathNow**.length() - 1) == **'π'** || **mathNow**.charAt(**mathNow**.length() - 1) == **'e'**) {  
 **if** (!hasLeftBracket(**mathNow**)) *//3.1 没有(, 加 ×(* **mathNow** += **"×("**;  
 **else** *//3.2 已有(, 加 )* **mathNow** += **")"**;  
 } **else if** (**mathNow**.charAt(**mathNow**.length() - 1) == **')'**) { *//4.mathNow最后一个字符是)，说明用户是在补全右括号，+)* **mathNow** += **')'**;  
 }  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
  
 **btnEqual**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 *//右括号自动补全* **int** leftNum = 0;  
 **int** rightNum = 0;  
 **for** (**int** i = 0; i < **mathNow**.length(); i++) {  
 **if** (**mathNow**.charAt(i) == **'('**)  
 leftNum++;  
 **if** (**mathNow**.charAt(i) == **')'**)  
 rightNum++;  
 }  
 **int** missingNum = leftNum - rightNum; *//缺失的 ) 数量* **while** (missingNum > 0) {  
 **mathNow** += **')'**;  
 missingNum--;  
 }  
 **tvNow**.setText(**mathNow**);  
  
 **mathPast** = **"\n"** + **mathNow**; *//使得呈现的mathPast自动换行* **double** result = **scienceCalculator**.cal(**mathNow**, **precision**, 0); *//调用科学计算器* **if** (result == Double.***MAX\_VALUE***)  
 **mathNow** = **"Math Error"**;  
 **else** {  
 **mathNow** = String.*valueOf*(result);  
 **if** (**mathNow**.charAt(**mathNow**.length() - 2) == **'.'** && **mathNow**.charAt(**mathNow**.length() - 1) == **'0'**) {  
 **mathNow** = **mathNow**.substring(0, **mathNow**.length() - 2);  
 }  
 }  
  
 **mathPast** = **mathPast** + **"="** + **mathNow**;  
  
 *//用tvPast.set(mathPast)不能实现自动滚动到最新运算过程* **tvPast**.append(**mathPast**); *//添加新的运算过程  
  
 //tvPast滚动到最新的运算过程* **int** offset = **tvPast**.getLineCount() \* **tvPast**.getLineHeight();  
 **if** (offset > **tvPast**.getHeight()) {  
 **tvPast**.scrollTo(0, offset - **tvPast**.getHeight());  
 }  
 **tvNow**.setText(**mathNow**);  
  
 **equal\_flag** = 1; *//设置flag=1* }  
 });  
 }  
  
 *//保存，复制，清空* **public void** initThreeFunctions() {  
 *//保存* **btnSave**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
  
 *//保存文件到sd卡 manifest文件中也要添加2个permission* **if** (Environment.*getExternalStorageState*().equals(Environment.***MEDIA\_MOUNTED***)) {  
 String path = Environment.*getExternalStorageDirectory*().getPath() + **"/math.txt"**; *//设置保存路径和文件名* **try** {  
 FileOutputStream outputStream = **new** FileOutputStream(path);  
 outputStream.write(**tvPast**.getText().toString().getBytes()); *//写字节* outputStream.close(); *//关闭输出流* } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 Toast.*makeText*(MainActivity.**this**, **"保存到"** + path, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
  
 *//复制* **btnCopy**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 ClipboardManager cm = (ClipboardManager) getSystemService(Context.***CLIPBOARD\_SERVICE***); *//采用ClipboardManager类* cm.setText(**tvPast**.getText());  
 Toast.*makeText*(MainActivity.**this**, **"已复制到剪切板"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
  
 *//清空* **btnClear**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **mathPast** = **""**;  
 **tvPast**.setText(**mathPast**);  
 Toast.*makeText*(MainActivity.**this**, **"计算过程已经清除"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
 }  
  
 *//初始化计算器基本Button* **public void** initBaseCalculatorFunction() {  
 **btnClc**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **mathNow** = **""**;  
 **tvNow**.setText(**mathNow**);  
 }  
 });  
  
 **btnDel**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (**mathNow**.length() != 0) {  
 **mathNow** = **mathNow**.substring(0, **mathNow**.length() - 1);  
 **tvNow**.setText(**mathNow**);  
 }  
 }  
 });  
 }  
  
 *//判断当前字符是否为数字* **private boolean** isNum(**char** c) {  
 **char** num[] = {**'0'**, **'1'**, **'2'**, **'3'**, **'4'**, **'5'**, **'6'**, **'7'**, **'8'**, **'9'**};  
 **int** i = 0;  
 **for** (; i < num.**length**; i++) {  
 **if** (num[i] == c)  
 **break**;  
 }  
 **return** i != num.**length**;  
 }  
  
 *//判断当前字符是否为运算符* **private boolean** isOper(**char** c) {  
 **char** oper[] = {**'+'**, **'-'**, **'×'**, **'/'**};  
 **int** i = 0;  
 **for** (; i < oper.**length**; i++) {  
 **if** (oper[i] == c)  
 **break**;  
 }  
 **return** i != oper.**length**;  
 }  
  
 *//判断当前math是否有')'* **private boolean** hasLeftBracket(String s) {  
 **int** i = 0;  
 **for** (; i < s.length(); i++) {  
 **if** (s.charAt(i) == **'('**)  
 **break**;  
 }  
 **return** i != s.length();  
 }  
  
 @Override  
 **public boolean** onKeyDown(**int** keyCode, KeyEvent event) {  
 *//判断用户是否点击了“返回键”* **if** (keyCode == KeyEvent.***KEYCODE\_BACK***) {  
 *//与上次点击返回键时刻作差* **if** ((System.*currentTimeMillis*() - **mExitTime**) > 2000) {  
 *//大于2000ms则认为是误操作，使用Toast进行提示* Toast.*makeText*(**this**, **"再按一次退出程序"**, Toast.***LENGTH\_SHORT***).show();  
 *//并记录下本次点击“返回键”的时刻，以便下次进行判断* **mExitTime** = System.*currentTimeMillis*();  
 } **else** {  
 *//小于2000ms则认为是用户确实希望退出程序-调用System.exit()方法进行退出* System.*exit*(1000);  
 }  
 **return true**;  
 }  
 **return super**.onKeyDown(keyCode, event);  
 }  
}

5、MyNumberPicker.java

**public class** MyNumberPicker **extends** NumberPicker {  
  
 **public** MyNumberPicker(Context context) {  
 **super**(context);  
 }  
  
 **public** MyNumberPicker(Context context, AttributeSet attrs) {  
 **super**(context, attrs);  
 }  
  
 **public** MyNumberPicker(Context context, AttributeSet attrs, **int** defStyleAttr) {  
 **super**(context, attrs, defStyleAttr);  
 }  
  
 @Override  
 **public void** addView(View child) {  
 **super**.addView(child);  
 updateView(child);  
 }  
  
 @Override  
 **public void** addView(View child, **int** index, android.view.ViewGroup.LayoutParams params) {  
 **super**.addView(child, index, params);  
 updateView(child);  
 }  
  
 @Override  
 **public void** addView(View child, android.view.ViewGroup.LayoutParams params) {  
 **super**.addView(child, params);  
 updateView(child);  
 }  
  
 **public void** updateView(View view) {  
 **if** (view **instanceof** EditText) {  
 *//这里修改字体的属性* ((EditText) view).setTextColor(Color.*parseColor*(**"#3FA2F0"**));  
*// ((EditText) view).setTextSize();* }  
 }  
}

6、ScienceCalculator.java

**public class** ScienceCalculator {  
 **private** BaseCalculator **baseCalculator** = **new** BaseCalculator();  
  
 **double** cal(String math, **int** precision, **int** angle\_metric) {  
  
 *//(1)预处理math* math = math.replace(**" "**, **""**); *//去掉math中的所有空格* math = math.replace(**"π"**, String.*valueOf*(Math.***PI***)); *//替换π* math = math.replace(**"e"**, String.*valueOf*(Math.*exp*(1))); *//替换自然指数e  
  
 //(2)计算指数(pow)运算并替换，包括(x)^(y)* **while** (math.contains(**"^"**)) {  
  
 *//1.中间寻找的点* **int** midIndex = math.lastIndexOf(**"^"**);  
  
 *//2.获取左边参数* **double** leftNum; *//左边的数* String leftStr; *//左边math字符串* **int** leftIndex = midIndex - 1;  
  
 **if** (math.charAt(leftIndex) == **')'**) { *//1.左边是一个表达式，即左边用括号括起来* **int** i = leftIndex - 1;  
 **while** (math.charAt(i) != **'('**) {  
 i--;  
 }  
 String subLeftMath = math.substring(i + 1, leftIndex);  
 leftNum = **baseCalculator**.cal(subLeftMath);  
 **if** (leftNum == Double.***MAX\_VALUE***) *//每次计算要判断是否出现 math error* **return** Double.***MAX\_VALUE***;  
  
 leftStr = **"("** + subLeftMath + **")"**;  
 } **else** { *//2.左边是一个数  
  
 //注意：判定index范围一定要在左边，否则可能出现IndexOutOfRange异常* **while** (leftIndex >= 0 && !isOper(math.charAt(leftIndex))) {  
 leftIndex--;  
 }  
 leftStr = math.substring(leftIndex + 1, midIndex);  
 leftNum = Double.*parseDouble*(leftStr);  
 }  
  
 *//3.获取右边参数* **double** rightNum;  
 String rightStr;  
 **int** rightIndex = midIndex + 1;  
  
 **if** (math.charAt(rightIndex) == **'('**) {  
 **int** i = rightIndex + 1;  
 **while** (math.charAt(i) != **')'**) {  
 i++;  
 }  
 String subRightMath = math.substring(rightIndex + 1, i);  
 rightNum = **baseCalculator**.cal(subRightMath);  
 **if** (rightNum == Double.***MAX\_VALUE***)  
 **return** Double.***MAX\_VALUE***;  
 rightStr = **"("** + subRightMath + **")"**;  
 } **else** {  
 **while** (rightIndex < math.length() && !isOper(math.charAt(rightIndex))) {  
 rightIndex++;  
 }  
 rightStr = math.substring(midIndex + 1, rightIndex);  
 rightNum = Double.*parseDouble*(rightStr);  
 }  
  
 *//4.得到完整的运算式并运算和替换* String wholeMath = leftStr + **"^"** + rightStr;  
 **double** result = Math.*pow*(leftNum, rightNum);  
 math = math.replace(wholeMath, String.*valueOf*(result));  
 }  
  
 *//(3)计算其他的科学运算符* **while** (math.contains(**"sin"**)  
 || math.contains(**"cos"**)  
 || math.contains(**"tan"**)  
 || math.contains(**"ln"**)  
 || math.contains(**"log"**)  
 || math.contains(**"√"**)) {  
  
 *//1.获取()内运算式并计算出结果，此时假设()不再包含复杂的科学运算* **int** beginIndex = math.lastIndexOf(**"("**);  
 **int** endIndex = getRightBracket(math, beginIndex);  
 String subMath = math.substring(beginIndex + 1, endIndex);  
 **double** subResult = **baseCalculator**.cal(subMath);  
 **if** (subResult == Double.***MAX\_VALUE***) *//每次计算要判断是否出现 math error* **return** Double.***MAX\_VALUE***;  
  
 *//2.获取scienceOper字符串* **int** i = beginIndex - 1;  
 **while** (i >= 0 && !isOper(math.charAt(i))) { *//向左寻找* i--;  
 }  
 String scienceOper = math.substring(i + 1, beginIndex);  
  
 *//3.匹配scienceOper进行科学运算，并替换相应部分* String tempMath;  
 **double** tempResult;  
 **int** DEG = 0; *//判断角度制* **switch** (scienceOper) {  
 **case "sin"**:  
 tempMath = **"sin("** + subMath + **")"**;  
 **if** (angle\_metric == DEG) {  
 tempResult = Math.*sin*(subResult / 180 \* Math.***PI***); *//将默认的 Rad → Deg* } **else** {  
 tempResult = Math.*sin*(subResult);  
 }  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "cos"**:  
 tempMath = **"cos("** + subMath + **")"**;  
 **if** (angle\_metric == DEG) {  
 tempResult = Math.*cos*(subResult / 180 \* Math.***PI***);  
 } **else** {  
 tempResult = Math.*cos*(subResult);  
 }  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "tan"**:  
 tempMath = **"tan("** + subMath + **")"**;  
 **if** (angle\_metric == DEG) {  
 tempResult = Math.*tan*(subResult / 180 \* Math.***PI***);  
 } **else** {  
 tempResult = Math.*tan*(subResult);  
 }  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "ln"**:  
 tempMath = **"ln("** + subMath + **")"**;  
 tempResult = Math.*log*(subResult);  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "log"**:  
 tempMath = **"log("** + subMath + **")"**;  
 tempResult = Math.*log10*(subResult);  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **case "√"**:  
 tempMath = **"√("** + subMath + **")"**;  
 tempResult = Math.*sqrt*(subResult);  
 math = math.replace(tempMath, String.*valueOf*(tempResult));  
 **break**;  
 **default**:  
 **break**;  
 }  
 }  
  
 *//(4)此时的math已经替换到BaseCalculator可处理的形式* **if** (**baseCalculator**.cal(math) == Double.***MAX\_VALUE***)  
 **return** Double.***MAX\_VALUE***;  
 **else** {  
 BigDecimal b = **new** BigDecimal(**baseCalculator**.cal(math));  
 **return** b.setScale(precision, BigDecimal.***ROUND\_HALF\_UP***).doubleValue(); *//四舍五入保留相应位数小数* }  
 }  
  
 **private int** getRightBracket(String math, **int** begin) {  
 **int** i;  
 **for** (i = begin; i < math.length(); i++) {  
 **if** (math.charAt(i) == **')'**)  
 **break**;  
 }  
 **return** i;  
 }  
  
  
 *//判断字符是否为运算符* **private boolean** isOper(**char** c) {  
 **char**[] operSet = {**'+'**, **'-'**, **'×'**, **'/'**, **'('**};  
 **int** i;  
 **for** (i = 0; i < operSet.**length**; i++) {  
 **if** (c == operSet[i]) {  
 **break**;  
 }  
 }  
 *//break出来，说明是oper，i != operSize* **return** i != operSet.**length**;  
 }  
}