项目源码在文章最后。

这两天写了一个Android的计算器，现在整理一下思路与代码。

首先是功能需求分析：

1、数字我规定最长为九位，如果有小数点则加上小数点最长为十位

2、第二个参数输入时不允许为负数，但是先得出的结果为负数，可以再进行运算（比如：1-2=-1，结果显示的是-1，可以再运算，-1+2=1）

3、得出的结果如果大于等于10位，则把数据进行科学计数

4、科学计数的数据不能再进行运算，再按按钮会直接显示0或者当前按下的数字

5、进行连续运算时，会先计算先一个运算的结果，在把结果作为第一个参数进行下次运算（比如：1+2+3，当按下第二个加号的时候，会显示3+，先把前一个运算的结果算出来再进行下一个运算）

6、当除数为0时，显示 error

7、默认情况下，显示 0

8、是否进行运算过，判断是否按下过等于号，如果按下，显示了结果，再按数字则直接显示当前数字而不是在结果后添加这个数字

接下来是正式开始编写代码：

1、界面布局 （TableLayout 实现按钮排版 ，EditText 显示表达式或值）

2、初始化所有控件，以及为需要点击事件的控件绑定监听器

3、编写监听器事件逻辑代码

①当按下的是数字，先判断是否计算过

----是（按下过等号）：把显示的文本改为当前按下的数字，计算过的标志设为false

----否（没按过等号）：进行下一步判断，计算过的标志设为false

--------判断当前文本是否有科学计数

--------是（有科学计数的标志 e ）：文本置为 0

--------判断文本是否只有一个 0

--------是（当前文本为 0）：文本置为空

--------判断是否有运算符

--------是（包含运算符）：取出运算符后的第二个参数，进行下一步判断

------------判断最后一个是否是运算符

------------是（文本最后一个是运算符号）：则直接添加按下的数字

------------否（文本最后一个不是运算符号）：进行下一步判断

----------------判断第二个参数的长度，包括小数点则可以为10位，不包括小数点只能为9位

--------否（不包含运算符）：判断第一个参数（即整个文本）的长度，包括小数点则可以为10位，不包括小数点只能为9位

②当按下的是小数点，先判断是否计算过

----是（按下过等号）：把文本改为 0. （数字 0 的后面有一个小数点），把计算过的标志位设为 false

----否（没按过等号）：进行下一步判断，，把计算过的标志位设为 false

--------判断是否有运算符

--------是（包含运算符）：取出第二个参数，判断长度是否超过，是：无操作，否：判断参数中是否已经有小数点，有：无操作，否，添加小数点

--------否（不包含运算符）：判断第一个参数（即整个文本）的长度是否超过，是：无操作，否：判断参数中是否已经有小数点，有：无操作，否，添加小数点

③当按下的是特殊键（归零、回退、百分比）

----归零：直接把文本改为 0

----回退：判断文本是否是 error，是：文本改为 0，否：判断文本的长度，当文本的长度大于0时且不等于1时，文本删掉最后一个字符，文本长度为1时，文本改为 0

----百分比：判断文本是否是 error，是：无操作，否：判断是否是表达式，是：无操作，否：数字除百

④当按下的是运算符，判断当前的文本是否是科学计数

----是（文本是科学计数，包含 e）：文本置 0

----否（文本不是科学计数，不包含 e）：进行下一步判断，且第二个参数不为空

--------判断文本是否是表达式

--------是（是表达式，有两个参数）：先得出结果，如果结果不是 error，添加上相应的运算符，如果结果是 error，无操作

--------否（不是表达式，可能是数字，也可能是数字后面带一个运算符）：如果计算过的标志位为 true，改为 false，如果最后一个字符为其他运算符，则改为当前按下的运算符，如果最后一个不是当前按下的运算符，则添加改运算符

⑤当按下的是等号，判断当前文本是否是表达式

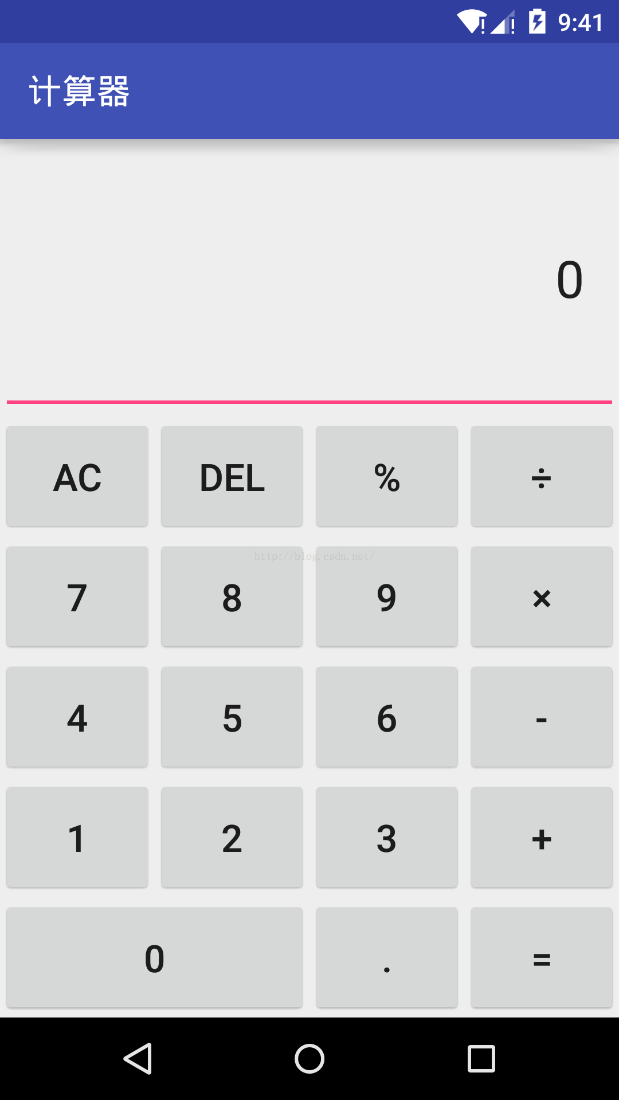
----是：判断包含的是哪种运算符，取出两个参数，如果第二个参数为空，则无操作，如果第二个参数不为空，则进行相应的计算，再把运算结果进行格式化，然后用正则表达式去掉多余的 0  。如果结果的长度大于10位，则把数字转换成科学计数

布局代码如下（activity\_main.xml）：

**[html]** [view plain](http://blog.csdn.net/someone_ikok/article/details/51075839) [copy](http://blog.csdn.net/someone_ikok/article/details/51075839)

1. **<?xml** version="1.0" encoding="utf-8"**?>**
2. **<LinearLayout** xmlns:android="http://schemas.android.com/apk/res/android"
3. android:layout\_width="match\_parent"
4. android:layout\_height="match\_parent"
5. android:orientation="vertical"
6. **>**
8. **<EditText**
9. android:id="@+id/result\_text"
10. android:layout\_width="match\_parent"
11. android:layout\_height="0dp"
12. android:layout\_weight="1"
13. android:textSize="30sp"
14. android:padding="20dp"
15. android:layout\_gravity="end"
16. android:editable="false"
17. android:textAlignment="textEnd"
18. android:singleLine="true"
19. android:text="0"
20. **/>**
22. **<TableLayout**
23. android:layout\_width="match\_parent"
24. android:layout\_height="wrap\_content"**>**
26. **<TableRow**
27. android:layout\_width="match\_parent"
28. android:layout\_height="wrap\_content"**>**
30. **<Button**
31. android:layout\_width="0dp"
32. android:layout\_weight="1"
33. android:layout\_height="70dp"
34. android:textSize="22sp"
35. android:text="AC"
36. android:id="@+id/ac\_btn" **/>**
38. **<Button**
39. android:layout\_width="0dp"
40. android:layout\_weight="1"
41. android:layout\_height="70dp"
42. android:textSize="22sp"
43. android:text="Del"
44. android:id="@+id/delete\_btn" **/>**
46. **<Button**
47. android:layout\_width="0dp"
48. android:layout\_weight="1"
49. android:layout\_height="70dp"
50. android:textSize="22sp"
51. android:text="%"
52. android:id="@+id/percent\_btn" **/>**
54. **<Button**
55. android:layout\_width="0dp"
56. android:layout\_weight="1"
57. android:layout\_height="70dp"
58. android:textSize="22sp"
59. android:text="÷"
60. android:id="@+id/divide\_btn" **/>**
61. **</TableRow>**
63. **<TableRow**
64. android:layout\_width="match\_parent"
65. android:layout\_height="wrap\_content"**>**
67. **<Button**
68. android:layout\_width="0dp"
69. android:layout\_weight="1"
70. android:layout\_height="70dp"
71. android:textSize="22sp"
72. android:text="7"
73. android:id="@+id/num\_seven" **/>**
75. **<Button**
76. android:layout\_width="0dp"
77. android:layout\_weight="1"
78. android:layout\_height="70dp"
79. android:textSize="22sp"
80. android:text="8"
81. android:id="@+id/num\_eight" **/>**
83. **<Button**
84. android:layout\_width="0dp"
85. android:layout\_weight="1"
86. android:layout\_height="70dp"
87. android:textSize="22sp"
88. android:text="9"
89. android:id="@+id/num\_nine" **/>**
91. **<Button**
92. android:layout\_width="0dp"
93. android:layout\_weight="1"
94. android:layout\_height="70dp"
95. android:textSize="22sp"
96. android:text="×"
97. android:id="@+id/multiply\_btn" **/>**
98. **</TableRow>**
100. **<TableRow**
101. android:layout\_width="match\_parent"
102. android:layout\_height="wrap\_content"**>**
104. **<Button**
105. android:layout\_width="0dp"
106. android:layout\_weight="1"
107. android:layout\_height="70dp"
108. android:textSize="22sp"
109. android:text="4"
110. android:id="@+id/num\_four" **/>**
112. **<Button**
113. android:layout\_width="0dp"
114. android:layout\_weight="1"
115. android:layout\_height="70dp"
116. android:textSize="22sp"
117. android:text="5"
118. android:id="@+id/num\_five" **/>**
120. **<Button**
121. android:layout\_width="0dp"
122. android:layout\_weight="1"
123. android:layout\_height="70dp"
124. android:textSize="22sp"
125. android:text="6"
126. android:id="@+id/num\_six" **/>**
128. **<Button**
129. android:layout\_width="0dp"
130. android:layout\_weight="1"
131. android:layout\_height="70dp"
132. android:textSize="22sp"
133. android:text="-"
134. android:id="@+id/subtract\_btn" **/>**
135. **</TableRow>**
137. **<TableRow**
138. android:layout\_width="match\_parent"
139. android:layout\_height="wrap\_content"**>**
141. **<Button**
142. android:layout\_width="0dp"
143. android:layout\_weight="1"
144. android:layout\_height="70dp"
145. android:textSize="22sp"
146. android:text="1"
147. android:id="@+id/num\_one" **/>**
149. **<Button**
150. android:layout\_width="0dp"
151. android:layout\_weight="1"
152. android:layout\_height="70dp"
153. android:textSize="22sp"
154. android:text="2"
155. android:id="@+id/num\_two" **/>**
157. **<Button**
158. android:layout\_width="0dp"
159. android:layout\_weight="1"
160. android:layout\_height="70dp"
161. android:textSize="22sp"
162. android:text="3"
163. android:id="@+id/num\_three" **/>**
165. **<Button**
166. android:layout\_width="0dp"
167. android:layout\_weight="1"
168. android:layout\_height="70dp"
169. android:textSize="22sp"
170. android:text="+"
171. android:id="@+id/plus\_btn" **/>**
172. **</TableRow>**
174. **<TableRow**
175. android:layout\_width="match\_parent"
176. android:layout\_height="wrap\_content"**>**
178. **<Button**
179. android:layout\_width="0dp"
180. android:layout\_weight="2"
181. android:layout\_height="70dp"
182. android:textSize="22sp"
183. android:text="0"
184. android:id="@+id/num\_zero" **/>**
186. **<Button**
187. android:layout\_width="0dp"
188. android:layout\_weight="1"
189. android:layout\_height="70dp"
190. android:textSize="22sp"
191. android:text="."
192. android:id="@+id/dot\_btn" **/>**
194. **<Button**
195. android:layout\_width="0dp"
196. android:layout\_weight="1"
197. android:layout\_height="70dp"
198. android:textSize="22sp"
199. android:text="="
200. android:id="@+id/equal\_btn" **/>**
201. **</TableRow>**
203. **</TableLayout>**
204. **</LinearLayout>**

界面效果如下：



逻辑部分代码如下（MainActivity.java）：

**[java]** [view plain](http://blog.csdn.net/someone_ikok/article/details/51075839) [copy](http://blog.csdn.net/someone_ikok/article/details/51075839)

1. **package** com.ikok.calculator;
3. **import** android.os.Bundle;
4. **import** android.support.v7.app.AppCompatActivity;
5. **import** android.util.Log;
6. **import** android.view.View;
7. **import** android.widget.Button;
8. **import** android.widget.EditText;
10. **public** **class** MainActivity **extends** AppCompatActivity **implements** View.OnClickListener {
11. /\*\*
12. \* 数字
13. \*/
14. **private** Button num0;
15. **private** Button num1;
16. **private** Button num2;
17. **private** Button num3;
18. **private** Button num4;
19. **private** Button num5;
20. **private** Button num6;
21. **private** Button num7;
22. **private** Button num8;
23. **private** Button num9;
24. /\*\*
25. \* 运算符
26. \*/
27. **private** Button plus\_btn;
28. **private** Button subtract\_btn;
29. **private** Button multiply\_btn;
30. **private** Button divide\_btn;
31. **private** Button equal\_btn;
32. /\*\*
33. \* 其他
34. \*/
35. **private** Button dot\_btn;
36. **private** Button percent\_btn;
37. **private** Button delete\_btn;
38. **private** Button ac\_btn;
39. /\*\*
40. \* 结果
41. \*/
42. **private** EditText mResultText;
43. /\*\*
44. \* 已经输入的字符
45. \*/
46. **private** String existedText = "";
47. /\*\*
48. \* 是否计算过
49. \*/
50. **private** **boolean** isCounted = **false**;
51. /\*\*
52. \* 以负号开头，且运算符不是是减号
53. \* 例如：-21×2
54. \*/
55. **private** **boolean** startWithOperator = **false**;
56. /\*\*
57. \* 以负号开头，且运算符是减号
58. \* 例如：-21-2
59. \*/
60. **private** **boolean** startWithSubtract = **false**;
61. /\*\*
62. \* 不以负号开头，且包含运算符
63. \* 例如：21-2
64. \*/
65. **private** **boolean** noStartWithOperator = **false**;


69. @Override
70. **protected** **void** onCreate(Bundle savedInstanceState) {
71. **super**.onCreate(savedInstanceState);
72. setContentView(R.layout.activity\_main);
74. initView();
75. initEvent();
77. }
79. /\*\*
80. \* 初始化控件
81. \*/
82. **private** **void** initView() {
83. /\*\*
84. \* 数字
85. \*/
86. num0 = (Button) findViewById(R.id.num\_zero);
87. num1 = (Button) findViewById(R.id.num\_one);
88. num2 = (Button) findViewById(R.id.num\_two);
89. num3 = (Button) findViewById(R.id.num\_three);
90. num4 = (Button) findViewById(R.id.num\_four);
91. num5 = (Button) findViewById(R.id.num\_five);
92. num6 = (Button) findViewById(R.id.num\_six);
93. num7 = (Button) findViewById(R.id.num\_seven);
94. num8 = (Button) findViewById(R.id.num\_eight);
95. num9 = (Button) findViewById(R.id.num\_nine);
96. /\*\*
97. \* 运算符
98. \*/
99. plus\_btn = (Button) findViewById(R.id.plus\_btn);
100. subtract\_btn = (Button) findViewById(R.id.subtract\_btn);
101. multiply\_btn = (Button) findViewById(R.id.multiply\_btn);
102. divide\_btn = (Button) findViewById(R.id.divide\_btn);
103. equal\_btn = (Button) findViewById(R.id.equal\_btn);
104. /\*\*
105. \* 其他
106. \*/
107. dot\_btn = (Button) findViewById(R.id.dot\_btn);
108. percent\_btn = (Button) findViewById(R.id.percent\_btn);
109. delete\_btn = (Button) findViewById(R.id.delete\_btn);
110. ac\_btn = (Button) findViewById(R.id.ac\_btn);
111. /\*\*
112. \* 结果
113. \*/
114. mResultText = (EditText) findViewById(R.id.result\_text);
115. /\*\*
116. \* 已经输入的字符
117. \*/
118. existedText = mResultText.getText().toString();
120. }
122. /\*\*
123. \* 初始化事件
124. \*/
125. **private** **void** initEvent() {
126. num0.setOnClickListener(**this**);
127. num1.setOnClickListener(**this**);
128. num2.setOnClickListener(**this**);
129. num3.setOnClickListener(**this**);
130. num4.setOnClickListener(**this**);
131. num5.setOnClickListener(**this**);
132. num6.setOnClickListener(**this**);
133. num7.setOnClickListener(**this**);
134. num8.setOnClickListener(**this**);
135. num9.setOnClickListener(**this**);
137. plus\_btn.setOnClickListener(**this**);
138. subtract\_btn.setOnClickListener(**this**);
139. multiply\_btn.setOnClickListener(**this**);
140. divide\_btn.setOnClickListener(**this**);
141. equal\_btn.setOnClickListener(**this**);
143. dot\_btn.setOnClickListener(**this**);
144. percent\_btn.setOnClickListener(**this**);
145. delete\_btn.setOnClickListener(**this**);
146. ac\_btn.setOnClickListener(**this**);
147. }
149. /\*\*
150. \* 点击事件
151. \* @param v  点击的控件
152. \*/
153. @Override
154. **public** **void** onClick(View v) {
156. **switch** (v.getId()){
157. /\*\*
158. \* 数字
159. \*/
160. **case** R.id.num\_zero:
161. existedText = isOverRange(existedText,"0");
162. **break**;
163. **case** R.id.num\_one:
164. existedText = isOverRange(existedText,"1");
165. **break**;
166. **case** R.id.num\_two:
167. existedText = isOverRange(existedText,"2");
168. **break**;
169. **case** R.id.num\_three:
170. existedText = isOverRange(existedText,"3");
171. **break**;
172. **case** R.id.num\_four:
173. existedText = isOverRange(existedText,"4");
174. **break**;
175. **case** R.id.num\_five:
176. existedText = isOverRange(existedText,"5");
177. **break**;
178. **case** R.id.num\_six:
179. existedText = isOverRange(existedText,"6");
180. **break**;
181. **case** R.id.num\_seven:
182. existedText = isOverRange(existedText,"7");
183. **break**;
184. **case** R.id.num\_eight:
185. existedText = isOverRange(existedText,"8");
186. **break**;
187. **case** R.id.num\_nine:
188. existedText = isOverRange(existedText,"9");
189. **break**;
190. /\*\*
191. \* 运算符
192. \*/
193. **case** R.id.plus\_btn:
194. /\*\*
195. \* 判断已有的字符是否是科学计数
196. \* 是 置零
197. \* 否 进行下一步
198. \*
199. \* 判断表达式是否可以进行计算
200. \* 是 先计算再添加字符
201. \* 否 添加字符
202. \*
203. \* 判断计算后的字符是否是 error
204. \* 是 置零
205. \* 否 添加运算符
206. \*/
207. **if** (!existedText.contains("e")) {
209. **if** (judgeExpression()) {
210. existedText = getResult();
211. **if** (existedText.equals("error")){
213. } **else** {
214. existedText += "+";
215. }
216. } **else** {
218. **if** (isCounted) {
219. isCounted = **false**;
220. }
222. **if** ((existedText.substring(existedText.length() - 1)).equals("-")) {
223. existedText = existedText.replace("-", "+");
224. } **else** **if** ((existedText.substring(existedText.length() - 1)).equals("×")) {
225. existedText = existedText.replace("×", "+");
226. } **else** **if** ((existedText.substring(existedText.length() - 1)).equals("÷")) {
227. existedText = existedText.replace("÷", "+");
228. } **else** **if** (!(existedText.substring(existedText.length() - 1)).equals("+")) {
229. existedText += "+";
230. }
231. }
232. } **else** {
233. existedText = "0";
234. }
236. **break**;
237. **case** R.id.subtract\_btn:
239. **if** (!existedText.contains("e")) {
240. **if** (judgeExpression()) {
241. existedText = getResult();
242. **if** (existedText.equals("error")){
244. } **else** {
245. existedText += "-";
246. }
247. } **else** {
249. **if** (isCounted) {
250. isCounted = **false**;
251. }
253. **if** ((existedText.substring(existedText.length() - 1)).equals("+")) {
254. //                    Log.d("Anonymous", "onClick: " + "进入减法方法");
255. existedText = existedText.replace("+", "-");
256. } **else** **if** ((existedText.substring(existedText.length() - 1)).equals("×")) {
257. existedText = existedText.replace("×", "-");
258. } **else** **if** ((existedText.substring(existedText.length() - 1)).equals("÷")) {
259. existedText = existedText.replace("÷", "-");
260. } **else** **if** (!(existedText.substring(existedText.length() - 1)).equals("-")) {
261. existedText += "-";
262. }
263. }
264. } **else** {
265. existedText = "0";
266. }
267. **break**;
268. **case** R.id.multiply\_btn:
270. **if** (!existedText.contains("e")) {
271. **if** (judgeExpression()) {
272. existedText = getResult();
273. **if** (existedText.equals("error")){
275. } **else** {
276. existedText += "×";
277. }
279. } **else** {
281. **if** (isCounted) {
282. isCounted = **false**;
283. }
285. **if** ((existedText.substring(existedText.length() - 1)).equals("+")) {
286. existedText = existedText.replace("+", "×");
287. } **else** **if** ((existedText.substring(existedText.length() - 1)).equals("-")) {
288. existedText = existedText.replace("-", "×");
289. } **else** **if** ((existedText.substring(existedText.length() - 1)).equals("÷")) {
290. existedText = existedText.replace("÷", "×");
291. } **else** **if** (!(existedText.substring(existedText.length() - 1)).equals("×")) {
292. existedText += "×";
293. }
294. }
295. } **else** {
296. existedText = "0";
297. }
298. **break**;
299. **case** R.id.divide\_btn:
301. **if** (!existedText.contains("e")) {
302. **if** (judgeExpression()) {
303. existedText = getResult();
304. **if** (existedText.equals("error")){
306. } **else** {
307. existedText += "÷";
308. }
310. } **else** {
312. **if** (isCounted) {
313. isCounted = **false**;
314. }
316. **if** ((existedText.substring(existedText.length() - 1)).equals("+")) {
317. existedText = existedText.replace("+", "÷");
318. } **else** **if** ((existedText.substring(existedText.length() - 1)).equals("-")) {
319. existedText = existedText.replace("-", "÷");
320. } **else** **if** ((existedText.substring(existedText.length() - 1)).equals("×")) {
321. existedText = existedText.replace("×", "÷");
322. } **else** **if** (!(existedText.substring(existedText.length() - 1)).equals("÷")) {
323. existedText += "÷";
324. }
325. }
326. } **else** {
327. existedText = "0";
328. }
329. **break**;
330. **case** R.id.equal\_btn:
331. existedText = getResult();
332. isCounted = **true**;
333. **break**;
334. /\*\*
335. \* 其他
336. \*/
337. **case** R.id.dot\_btn:
338. /\*\*
339. \* 判断是否运算过
340. \* 否
341. \*   判断是否有运算符，有 判断运算符之后的数字，无 判断整个数字
342. \*   判断数字是否过长，是则不能添加小数点，否则可以添加
343. \*   判断已经存在的数字里是否有小数点
344. \* 是
345. \*   字符串置为 0.
346. \*/
347. **if** (!isCounted){
349. **if** (existedText.contains("+") || existedText.contains("-") ||
350. existedText.contains("×") || existedText.contains("÷") ){
352. String param1 = **null**;
353. String param2 = **null**;
355. **if** (existedText.contains("+")) {
356. param1 = existedText.substring(0, existedText.indexOf("+"));
357. param2 = existedText.substring(existedText.indexOf("+") + 1);
358. } **else** **if** (existedText.contains("-")) {
359. param1 = existedText.substring(0, existedText.indexOf("-"));
360. param2 = existedText.substring(existedText.indexOf("-") + 1);
361. } **else** **if** (existedText.contains("×")) {
362. param1 = existedText.substring(0, existedText.indexOf("×"));
363. param2 = existedText.substring(existedText.indexOf("×") + 1);
364. } **else** **if** (existedText.contains("÷")) {
365. param1 = existedText.substring(0, existedText.indexOf("÷"));
366. param2 = existedText.substring(existedText.indexOf("÷") + 1);
367. }
368. Log.d("Anonymous param1",param1);
369. Log.d("Anonymous param2",param2);
371. **boolean** isContainedDot = param2.contains(".");
372. **if** (param2.length() >= 9){
374. } **else** **if** (!isContainedDot){
375. **if** (param2.equals("")){
376. existedText += "0.";
377. } **else** {
378. existedText += ".";
379. }
380. } **else** {
381. **return**;
382. }
383. } **else** {
384. **boolean** isContainedDot = existedText.contains(".");
385. **if** (existedText.length() >= 9){
387. } **else** **if** (!isContainedDot){
388. existedText += ".";
389. } **else** {
390. **return**;
391. }
392. }
393. isCounted = **false**;
395. } **else** {
396. existedText = "0.";
397. isCounted = **false**;
398. }

401. **break**;
402. **case** R.id.percent\_btn:
403. /\*\*
404. \* 判断数字是否有运算符
405. \* 是 不做任何操作
406. \* 否 进行下一步
407. \*
408. \* 判断数字是否是 0
409. \* 是 不做任何操作
410. \* 否 进行除百
411. \*
412. \* 将字符串转换成double类型，进行运算后，再转换成String型
413. \*/
414. **if** (existedText.equals("error")){
416. } **else** {
418. getCondition();
420. **if** (startWithOperator || startWithSubtract || noStartWithOperator) {
422. } **else** {
423. **if** (existedText.equals("0")) {
424. **return**;
425. } **else** {
426. **double** temp = Double.parseDouble(existedText);
427. temp = temp / 100;
428. existedText = String.valueOf(temp);
429. }
430. }
431. }
432. **break**;
433. **case** R.id.delete\_btn:
434. /\*\*
435. \* 字符串长度大于 0 时才截取字符串
436. \* 如果长度为 1，则直接把字符串设置为 0
437. \*/
438. **if** (existedText.equals("error")){
439. existedText = "0";
440. } **else** **if** (existedText.length() > 0){
441. **if** (existedText.length() == 1) {
442. existedText = "0";
443. } **else** {
444. existedText = existedText.substring(0,existedText.length()-1);
445. }
446. }
447. **break**;
448. **case** R.id.ac\_btn:
449. existedText = "0";
450. **break**;
451. }
452. /\*\*
453. \* 设置显示
454. \*/
455. mResultText.setText(existedText);
456. }


460. /\*\*
461. \* 进行运算，得到结果
462. \* @return  返回结果
463. \*/
464. **private** String getResult() {
466. /\*\*
467. \* 结果
468. \*/
469. String tempResult = **null**;
470. /\*\*
471. \* 两个String类型的参数
472. \*/
473. String param1 = **null**;
474. String param2 = **null**;
475. /\*\*
476. \* 转换后的两个double类型的参数
477. \*/
478. **double** arg1 = 0;
479. **double** arg2 = 0;
480. **double** result = 0;
482. getCondition();
484. /\*\*
485. \* 如果有运算符，则进行运算
486. \* 没有运算符，则把已经存在的数据再传出去
487. \*/
488. **if** ( startWithOperator || noStartWithOperator || startWithSubtract) {
490. **if** (existedText.contains("+")) {
491. /\*\*
492. \* 先获取两个参数
493. \*/
494. param1 = existedText.substring(0, existedText.indexOf("+"));
495. param2 = existedText.substring(existedText.indexOf("+") + 1);
496. /\*\*
497. \* 如果第二个参数为空，则还是显示当前字符
498. \*/
499. **if** (param2.equals("")){
500. tempResult = existedText;
501. } **else** {
502. /\*\*
503. \* 转换String为Double
504. \* 计算后再转换成String类型
505. \* 进行正则表达式处理
506. \*/
507. arg1 = Double.parseDouble(param1);
508. arg2 = Double.parseDouble(param2);
509. result = arg1 + arg2;
510. tempResult = String.format("%f", result);
511. tempResult = subZeroAndDot(tempResult);
512. }

515. } **else** **if** (existedText.contains("×")) {
517. param1 = existedText.substring(0, existedText.indexOf("×"));
518. param2 = existedText.substring(existedText.indexOf("×") + 1);
520. **if** (param2.equals("")){
521. tempResult = existedText;
522. } **else** {
523. arg1 = Double.parseDouble(param1);
524. arg2 = Double.parseDouble(param2);
525. result = arg1 \* arg2;
526. tempResult = String.format("%f", result);
527. tempResult = subZeroAndDot(tempResult);
528. }
530. } **else** **if** (existedText.contains("÷")) {
532. param1 = existedText.substring(0, existedText.indexOf("÷"));
533. param2 = existedText.substring(existedText.indexOf("÷") + 1);
535. **if** (param2.equals("0")){
536. tempResult = "error";
537. } **else** **if** (param2.equals("")){
538. tempResult = existedText;
539. } **else** {
540. arg1 = Double.parseDouble(param1);
541. arg2 = Double.parseDouble(param2);
542. result = arg1 / arg2;
543. tempResult = String.format("%f", result);
544. tempResult = subZeroAndDot(tempResult);
545. }
547. } **else** **if** (existedText.contains("-")) {
549. /\*\*
550. \* 这里是以最后一个 - 号为分隔去取出两个参数
551. \* 进到这个方法，必须满足有运算公式
552. \* 而又避免了第一个参数是负数的情况
553. \*/
554. param1 = existedText.substring(0, existedText.lastIndexOf("-"));
555. param2 = existedText.substring(existedText.lastIndexOf("-") + 1);
557. **if** (param2.equals("")){
558. tempResult = existedText;
559. } **else** {
560. arg1 = Double.parseDouble(param1);
561. arg2 = Double.parseDouble(param2);
562. result = arg1 - arg2;
563. tempResult = String.format("%f", result);
564. tempResult = subZeroAndDot(tempResult);
565. }
567. }
568. /\*\*
569. \* 如果数据长度大于等于10位，进行科学计数
570. \*
571. \* 如果有小数点，再判断小数点前是否有十个数字，有则进行科学计数
572. \*/
573. **if** (tempResult.length() >= 10) {
574. tempResult = String.format("%e", Double.parseDouble(tempResult));
575. } **else** **if** (tempResult.contains(".")) {
576. **if** (tempResult.substring(0, tempResult.indexOf(".")).length() >= 10) {
577. tempResult = String.format("%e", Double.parseDouble(tempResult));
578. }
579. }
580. } **else** {
581. tempResult = existedText;
582. }
584. **return** tempResult;
585. }

588. /\*\*
589. \* 先判断是否按过等于号
590. \* 是 按数字则显示当前数字
591. \* 否 在已有的表达式后添加字符
592. \*
593. \* 判断数字是否就是一个 0
594. \* 是 把字符串设置为空字符串。
595. \*   1、打开界面没有运算过的时候，AC键归零或删除完归零的时候，会显示一个 0
596. \*   2、当数字是 0 的时候，设置成空字符串，再按 0 ，数字会还是 0，不然可以按出 000 这种数字
597. \* 否 添加按下的键的字符
598. \*
599. \* 判断数字是否包含小数点
600. \* 是 数字不能超过十位
601. \* 否 数字不能超过九位
602. \*
603. \* 进行上面的判断后，再判断数字是否超过长度限制
604. \* 超过不做任何操作
605. \* 没超过可以再添数字
606. \*/
607. **private** String isOverRange(String existedText, String s) {
608. /\*\*
609. \* 判断是否计算过
610. \*/
611. **if** (!isCounted){
612. /\*\*
613. \* 判断是否是科学计数
614. \* 是 文本置零
615. \*/
616. **if** (existedText.contains("e")){
617. existedText = "0";
618. }
619. /\*\*
620. \* 判断是否只有一个 0
621. \* 是 文本清空
622. \*/
623. **if** (existedText.equals("0")){
624. existedText = "";
625. }
626. /\*\*
627. \* 判断是否有运算符
628. \* 是 判断第二个数字
629. \* 否 判断整个字符串
630. \*/
631. **if** (existedText.contains("+") || existedText.contains("-") ||
632. existedText.contains("×") || existedText.contains("÷")){
633. /\*\*
634. \* 包括运算符时 两个数字 判断第二个数字
635. \* 两个参数
636. \*/
637. String param2 = **null**;
638. **if** (existedText.contains("+")){
639. param2 = existedText.substring(existedText.indexOf("+")+1);
640. } **else** **if** (existedText.contains("-")){
641. param2 = existedText.substring(existedText.indexOf("-")+1);
642. } **else** **if** (existedText.contains("×")){
643. param2 = existedText.substring(existedText.indexOf("×")+1);
644. } **else** **if** (existedText.contains("÷")){
645. param2 = existedText.substring(existedText.indexOf("÷")+1);
646. }
648. //            Log.d("Anonymous param1",param1);
649. //            Log.d("Anonymous param2",param2);
650. **if** (existedText.substring(existedText.length()-1).equals("+") ||
651. existedText.substring(existedText.length()-1).equals("-") ||
652. existedText.substring(existedText.length()-1).equals("×") ||
653. existedText.substring(existedText.length()-1).equals("÷")){
654. existedText += s;
655. } **else** {
656. **if** (param2.contains(".")){
657. **if** (param2.length() >= 10){
659. } **else** {
660. existedText += s;
661. }
662. } **else** {
663. **if** (param2.length() >= 9){
665. } **else** {
666. existedText += s;
667. }
668. }
669. }
670. } **else** {
671. /\*\*
672. \* 不包括运算符时 一个数字
673. \*/
674. **if** (existedText.contains(".")){
675. **if** (existedText.length() >= 10){
677. } **else** {
678. existedText += s;
679. }
680. } **else** {
681. **if** (existedText.length() >= 9){
683. } **else** {
684. existedText += s;
685. }
686. }
687. }
689. isCounted = **false**;
691. } **else** {
693. existedText = s;
694. isCounted = **false**;
696. }

699. **return** existedText;
700. }

703. /\*\*
704. \* 使用java正则表达式去掉多余的.与0
705. \* @param s 传入的字符串
706. \* @return 修改之后的字符串
707. \*/
708. **public** **static** String subZeroAndDot(String s){
709. **if**(s.indexOf(".") > 0){
710. s = s.replaceAll("0+?$", "");//去掉多余的0
711. s = s.replaceAll("[.]$", "");//如最后一位是.则去掉
712. }
713. **return** s;
714. }
716. /\*\*
717. \* 判断表达式
718. \*
719. \* 为了按等号是否进行运算
720. \* 以及出现两个运算符（第一个参数如果为负数的符号不计）先进行运算再添加运算符
721. \*/
722. **private** **boolean** judgeExpression() {
724. getCondition();
726. String tempParam2 = **null**;
728. **if** ( startWithOperator || noStartWithOperator || startWithSubtract) {
730. **if** (existedText.contains("+")) {
731. /\*\*
732. \* 先获取第二个参数
733. \*/
734. tempParam2 = existedText.substring(existedText.indexOf("+") + 1);
735. /\*\*
736. \* 如果第二个参数为空，表达式不成立
737. \*/
738. **if** (tempParam2.equals("")) {
739. **return** **false**;
740. } **else** {
741. **return** **true**;
742. }
743. } **else** **if** (existedText.contains("×")) {
745. tempParam2 = existedText.substring(existedText.indexOf("×") + 1);
747. **if** (tempParam2.equals("")) {
748. **return** **false**;
749. } **else** {
750. **return** **true**;
751. }
753. } **else** **if** (existedText.contains("÷")) {
755. tempParam2 = existedText.substring(existedText.indexOf("÷") + 1);
757. **if** (tempParam2.equals("")) {
758. **return** **false**;
759. } **else** {
760. **return** **true**;
761. }
763. } **else** **if** (existedText.contains("-")) {
765. /\*\*
766. \* 这里是以最后一个 - 号为分隔去取出两个参数
767. \* 进到这个方法，必须满足有运算公式
768. \* 而又避免了第一个参数是负数的情况
769. \*/
770. tempParam2 = existedText.substring(existedText.lastIndexOf("-") + 1);
772. **if** (tempParam2.equals("")) {
773. **return** **false**;
774. } **else** {
775. **return** **true**;
776. }
778. }
779. }
780. **return** **false**;
781. }
783. /\*\*
784. \* 取得判断条件
785. \*/
786. **private** **void** getCondition() {
787. /\*\*
788. \* 以负号开头，且运算符不是是减号
789. \* 例如：-21×2
790. \*/
791. startWithOperator = existedText.startsWith("-") && ( existedText.contains("+") ||
792. existedText.contains("×") || existedText.contains("÷") );
793. /\*\*
794. \* 以负号开头，且运算符是减号
795. \* 例如：-21-2
796. \*/
797. startWithSubtract = existedText.startsWith("-") && ( existedText.lastIndexOf("-") != 0 );
798. /\*\*
799. \* 不以负号开头，且包含运算符
800. \* 例如：21-2
801. \*/
802. noStartWithOperator = !existedText.startsWith("-") && ( existedText.contains("+") ||
803. existedText.contains("-") || existedText.contains("×") || existedText.contains("÷"));
804. }
806. }

运行效果如下：

