MICROOH 麦可网

Android-从程序员到架构师之路

出品人：Sundy
讲师：高焕堂（台湾）

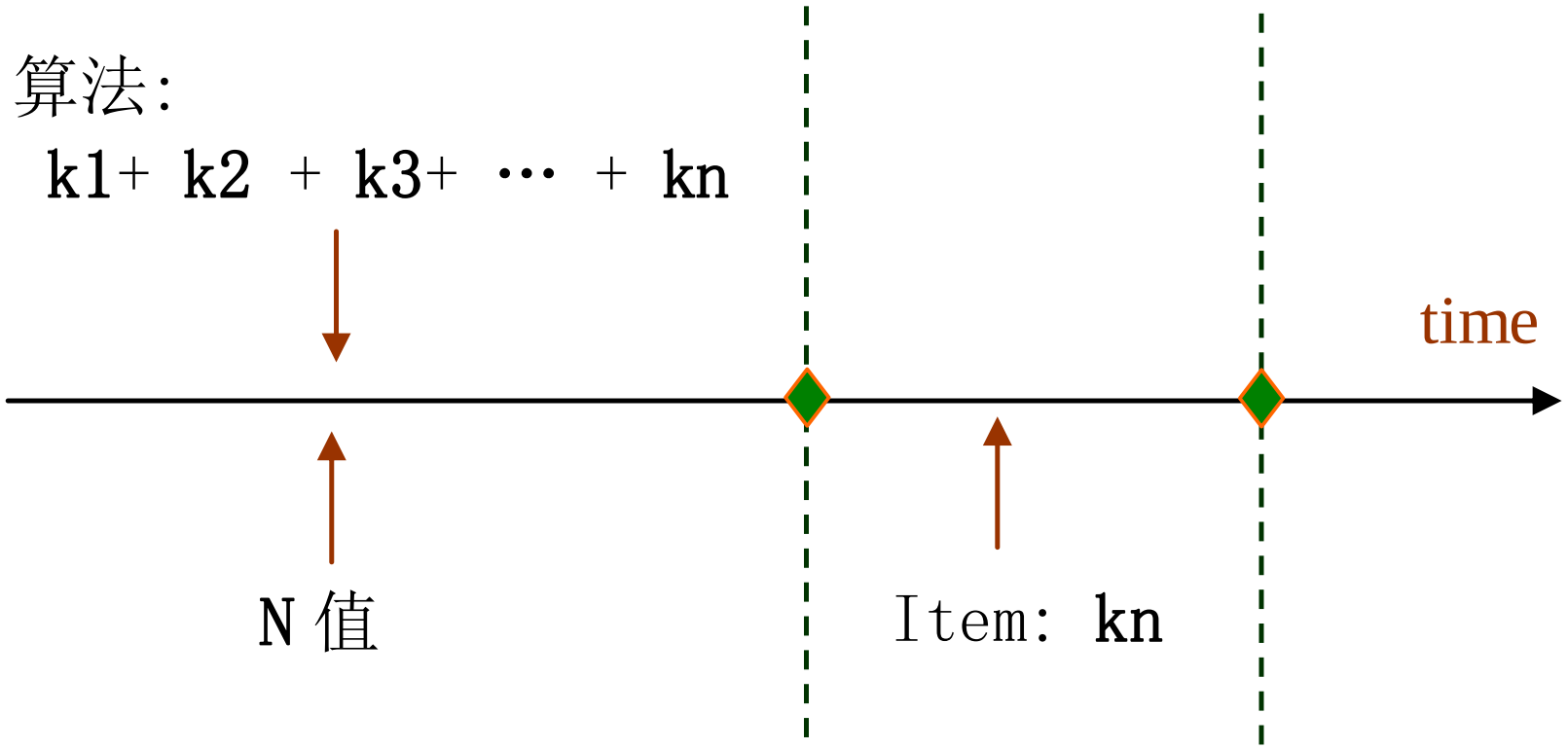
http://www.microoh.com

A07_c

从架构到代码的演练(c)

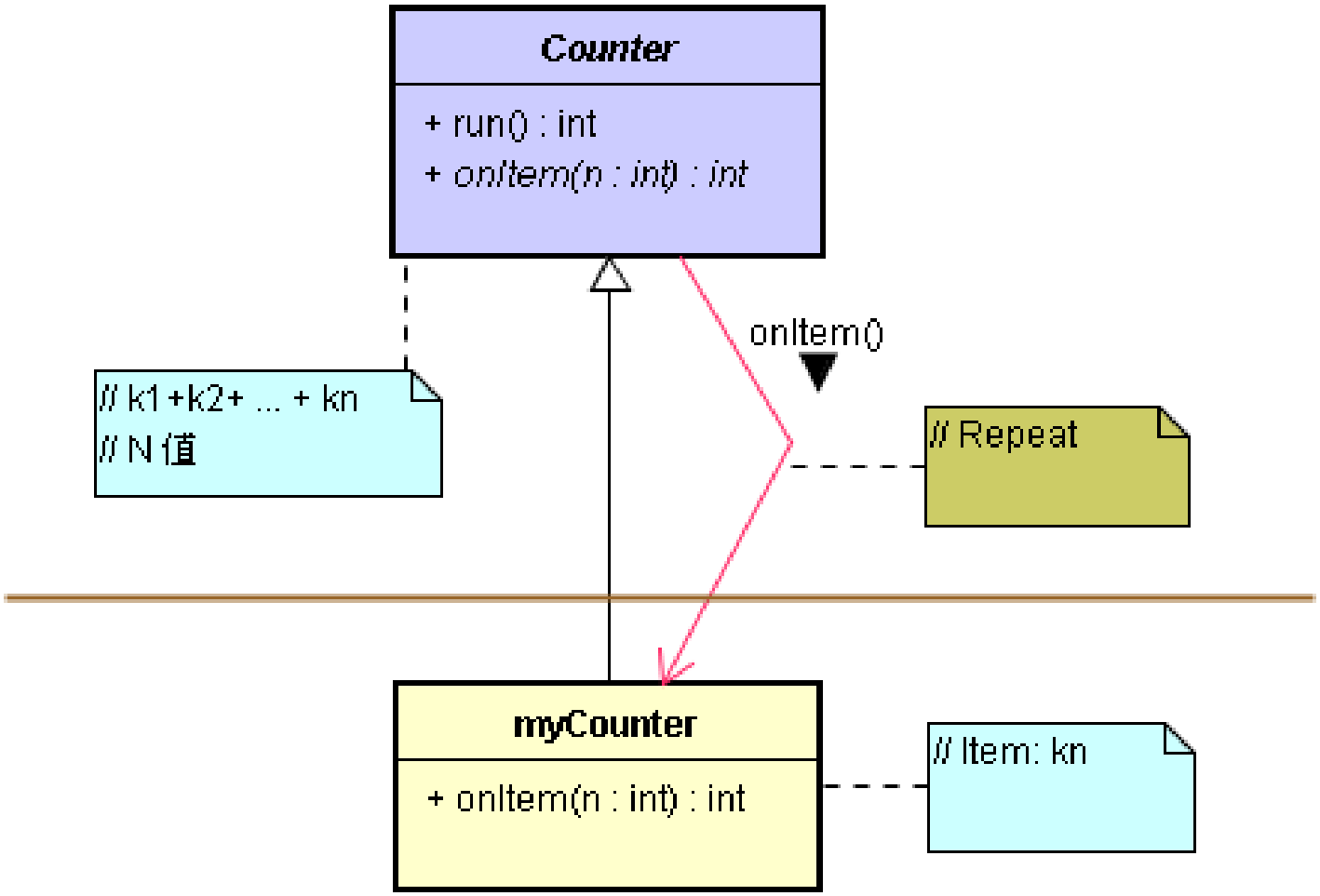
By 高焕堂

亲自演练：题目(三)



- 现在可以试试先想想接口<I>设计：<E>必须有个抽象函数，来反向调用到<T>。
- <E>可以重复调用该函数，总共调用N次，每次回传一个Kn值，在由<E>把它们累加起来。

- 当然，你也能设计一个新的接口函数，<E>只呼叫它一次，呼叫时把N值传递下去给子类别。由子类别回传N项数据，例如从数据库里读取N笔数据并回传给<E>。
- 反正，接口函数的制定权就掌控于你(架構師)的手中，<T>开发者会配合你的。

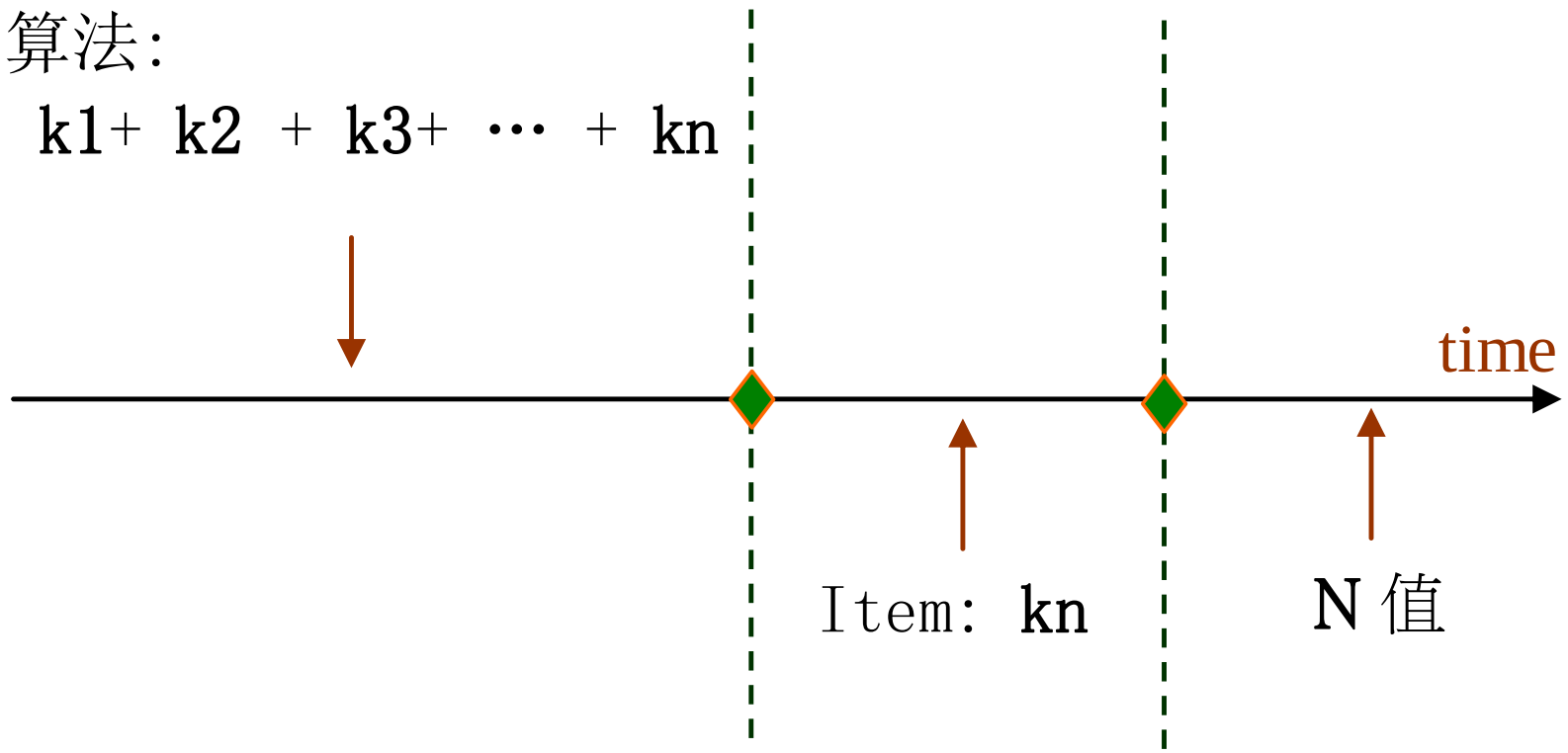

Counter
+ run() : int
+ onItem(n : int) : int
onItem()
// k1 +k2+ ... + kn
// N 值
// Repeat
myCounter
+ onItem(n : int) : int
// Item: kn

```
public abstract class Counter {
        public int run(){
                        int N = getCount();
                        int sum = 0;
                           for(int i=1; i<=N; i++) {
                               sum += onItem(i);
                            }
                           return sum;
                }
        public int getCount() {  return 6;  }
        protected abstract int onItem(int k);
}
```

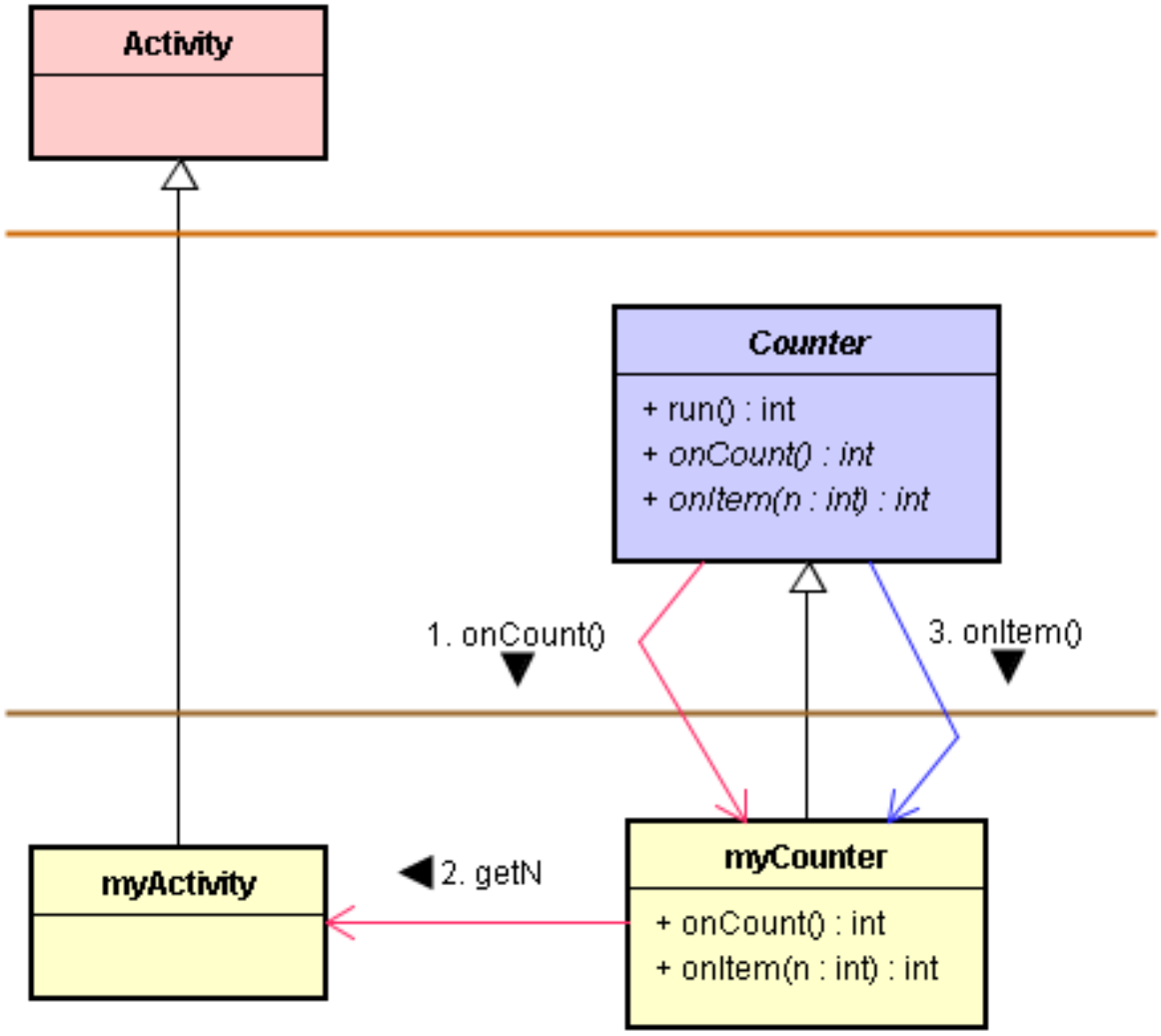
```
public class myCounter extends Counter{
        @Override protected int onItem(int k) {
                return bonus[k-1];
        }
    private int bonus[]
       = {100, 300, 100, 300, 100, 300 };
}
```

```
// myActivity.java
// 与上一题目相同
```

亲自演练：题目(四)



- 依据此图，你必须使用Activity来提供UI上的一个EditText窗口，让User在执行阶段才输入N值。由<E>主动去向Activity的EditText取得N值，然后重复调用<I>的函数，总共呼叫N次，每次回传一个Kn值，再由<E>把它们累加起来。于是，设计出类别架构图，如下：

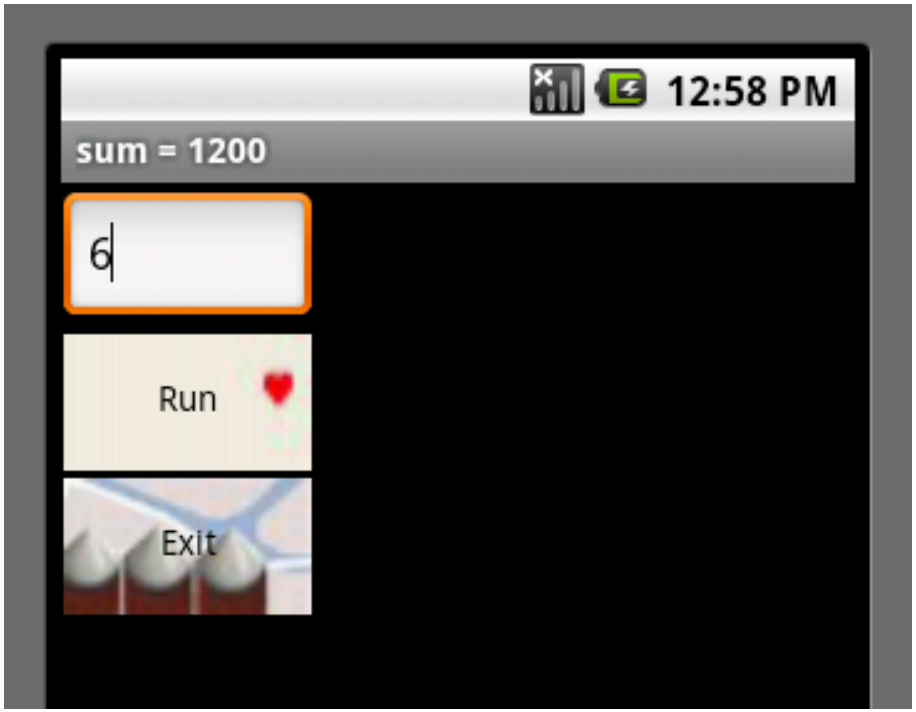

Activity
Counter
+ run() : int
+ onCount() : int
+ onItem(n : int) : int
1. onCount()
3. onItem()
myActivity
2. getN
myCounter
+ onCount() : int
+ onItem(n : int) : int

```
public abstract class Counter {
        public int run(){
                int N = getCount();
                int sum = 0;
                  for(int i=1; i<=N; i++) {
                      sum += onItem(i);
                   }
                  return sum;
            }
           public int getCount() {
                   return onCount();
                }
           protected abstract int onCount();
           protected abstract int onItem(int k);
}
```

```
public class myCounter extends Counter{
        private int bonus[] =
                {100, 300, 100, 300, 100, 300 };

        @Override
        protected int onItem(int k) {
                return bonus[k-1];
        }
        @Override
        protected int onCount() {
                        int n = myActivity.mN;
                        if(n > 6) n = 6;
                                return n;
        }}
```

```
public class myActivity extends Activity implements OnClickListener {
        private Button btn, btn2;
        private EditText et;
        public static int mN;

        @Override
    public void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
      LinearLayout layout = new LinearLayout(this);
      layout.setOrientation(LinearLayout.VERTICAL);
       LinearLayout.LayoutParams param =
              new LinearLayout.LayoutParams(100, 55);
       param.leftMargin = 1;  param.topMargin = 3;

       et = new EditText(this); et.setId(100);
       et.setOnClickListener(this);
       layout.addView(et, param);
       btn = new Button(this);     btn.setId(101);     btn.setText("Run");
       btn.setOnClickListener(this);
       btn.setBackgroundResource(R.drawable.heart);
       layout.addView(btn, param);
       btn2 = new Button(this);    btn2.setId(102);    btn2.setText("Exit");
       btn2.setOnClickListener(this);
       btn2.setBackgroundResource(R.drawable.gray);
       layout.addView(btn2, param);
       setContentView(layout);
    }
```

```
public void onClick(View v) {
        switch(v.getId()) {
        case 101:
            String ss = et.getText().toString();
                mN = Integer.parseInt(ss);
           Counter counter = new myCounter();
           int sum = counter.run();
           setTitle("sum = " + String.valueOf(sum));
              break;
            case 102:
                    finish();       break;
    }}}
```

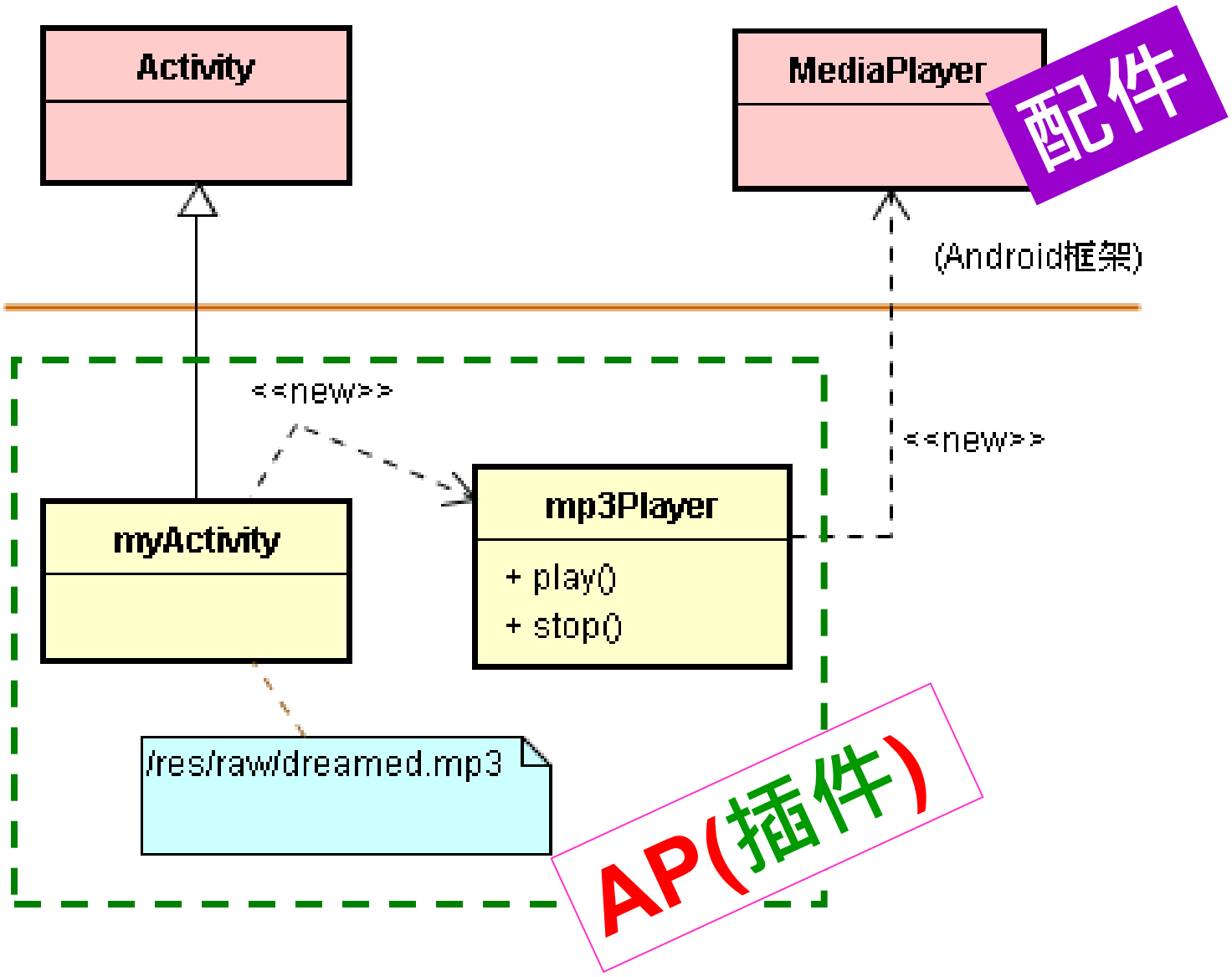

12:58 PM
sum = 1200
6
Run
Exit

8、结语

應用程序(AP)也是框架的一種插件

- 框架(或架构)设计的关键任务就是接口(Interface)设计，这项接口是框架<E>与插件<T>之间的接口，这就是所谓的：框架API。

- 架构师的工作就是聚焦于这件最为关键的事情上，这样子让AP开发工作就显得很轻松了，只要专注于厘清买主知识的内涵，把它分析出来写入<T>里就行了。最后，将相关的<T>组合起来，就成为应用程序(简称AP或App)了。


Activity
MediaPlayer
配件
(Android框架)
<<new>>
<<new>>
myActivity
mp3Player
+ play()
+ stop()
/res/raw/dreamed.mp3
AP(插件)

Thanks...

高煥堂