新测试金字塔之移动互联网应用测试

陈晔

自我介绍

- 陈晔
- 曾担任多家创业公司首任测试主管
- 移动互联网应用测试布道师
- 正测试三观的人
- 上海各个高校测试推广
- 到处刷脸
- 编写《大话移动测试——Android与iOS应用测试指南》（清华大学出版社）
- 目前负责支付宝商户版移动应用的自动化技术

回顾下移动互联网测试的历史

我们的项目流程和现状

移动互联网项目特点

- 迭代快
- 变化快
- 烧钱快
- 见效快
- 用户至上

迭代快

迭代周期：两个月？一个月？半个月？
还有更甚？？

变化快

第N次迭代产品的样子

第N＋1次迭代产品的样子

烧钱快


PK
TAXI
嘀嘀打车
日均订单
183万单
单日最高订单量
262万
微信支付日最高订单量
200万
活动期间用户数
4000万
补贴总额
4亿元
TAXI
快
快的打车
日均订单
128万单
单日最高订单量
162万
支付宝支付日最高订单量
60万
活动期间用户数
未公布
补贴总额
超1亿元

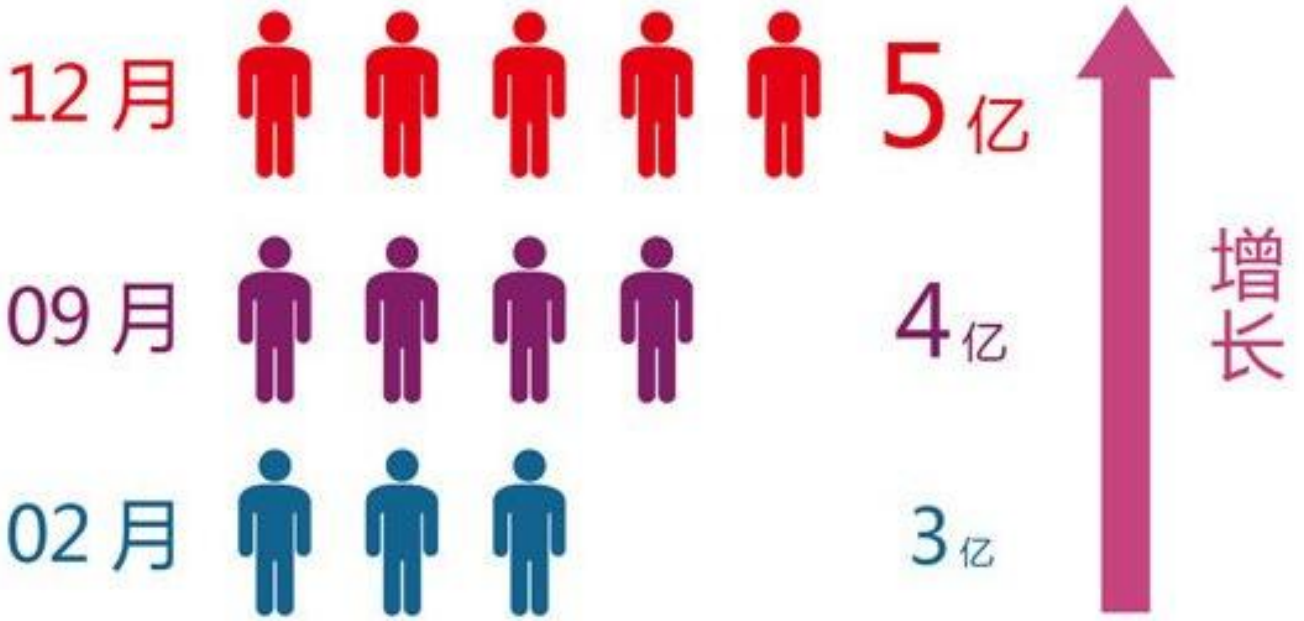
制图 王云涛

见效快

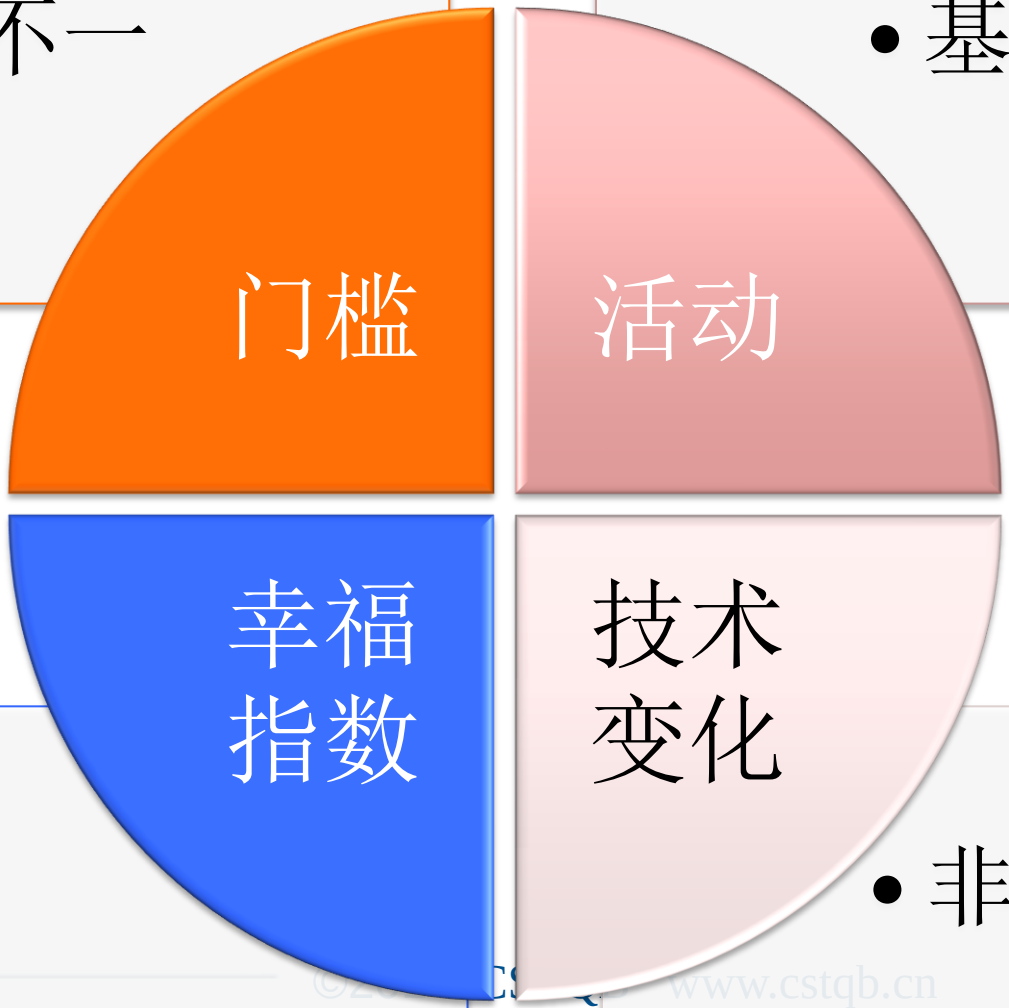
用户至上

一、用户规模

2012 年全年，新浪微博注册用户继续保持**稳定增长**，于 2 月底突破 3 亿，7 个月后突破 4 亿，到 12 月底一举突破 **5 亿大关**。



移动互联网测试特点

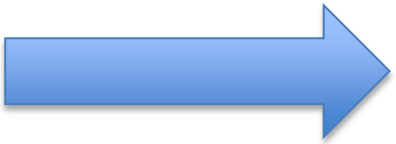


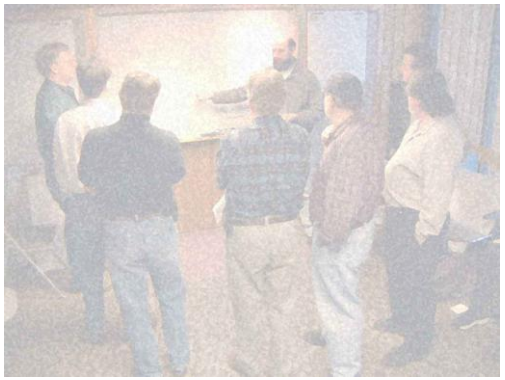
- 高低不一
- 基本手动
- 中等
- 非常快

测试行业特点

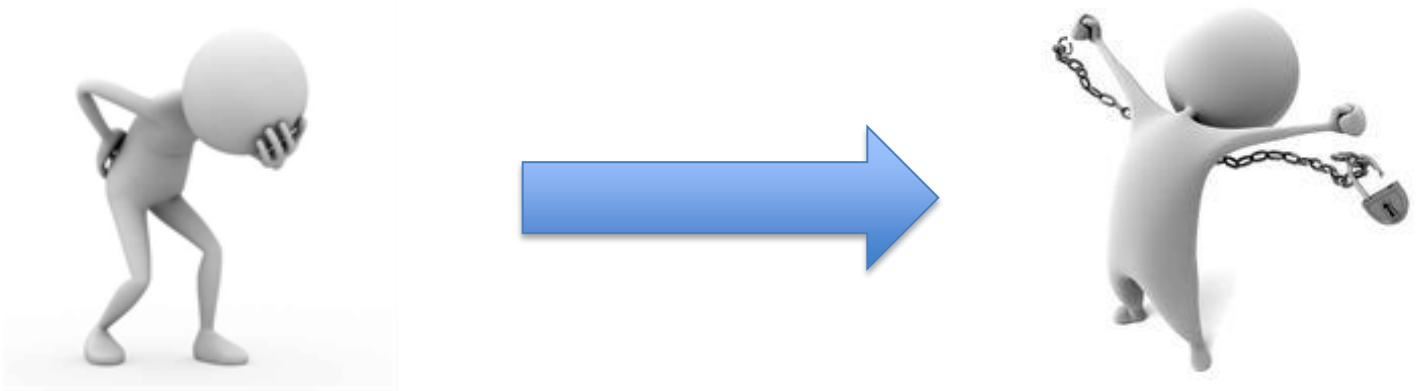
- 各个大会理论忽悠多，落地技术少
- 行业组成人员鱼龙混杂
- 抱怨多，静下心来学习少
- 大部分人都是“拿来主义”
- 大部分人自己看不起自己和测试
- 大部分人也不知道测试到底做啥

问题：移动互联网给我们带来了什么？





在这样的前提下，大部分测试的愿望是什么？

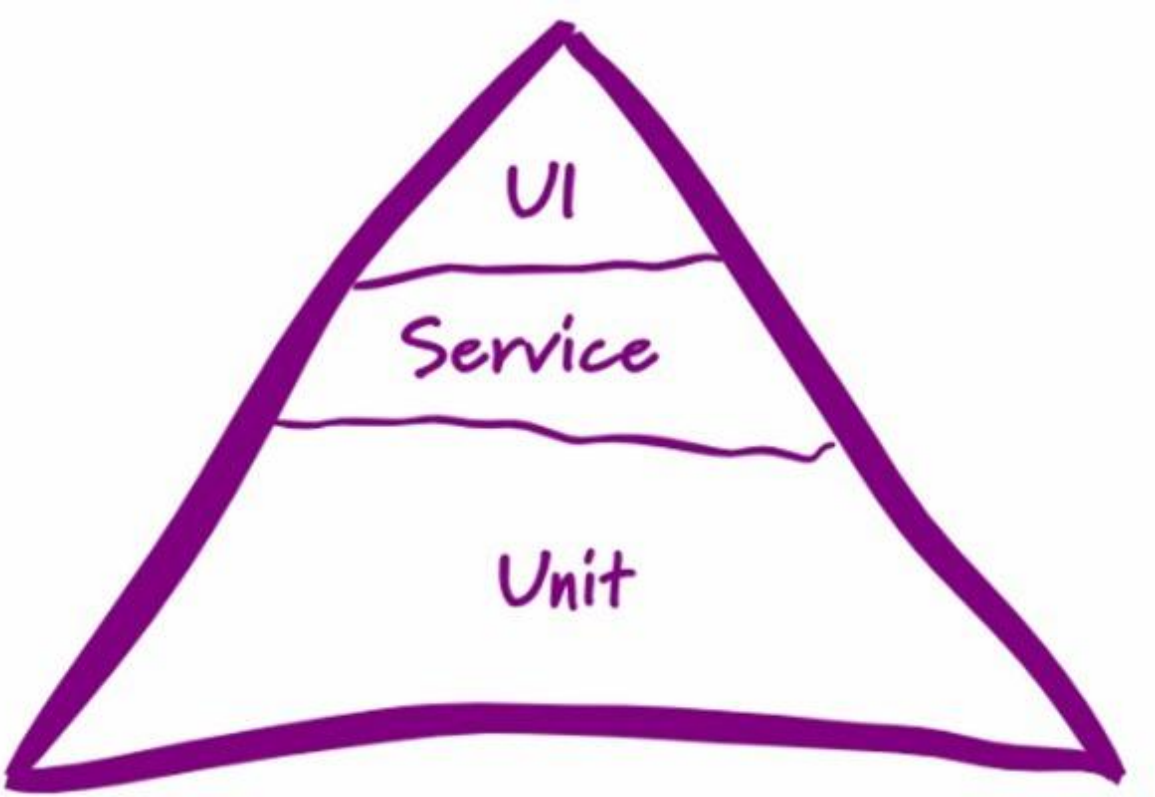


可是事实是

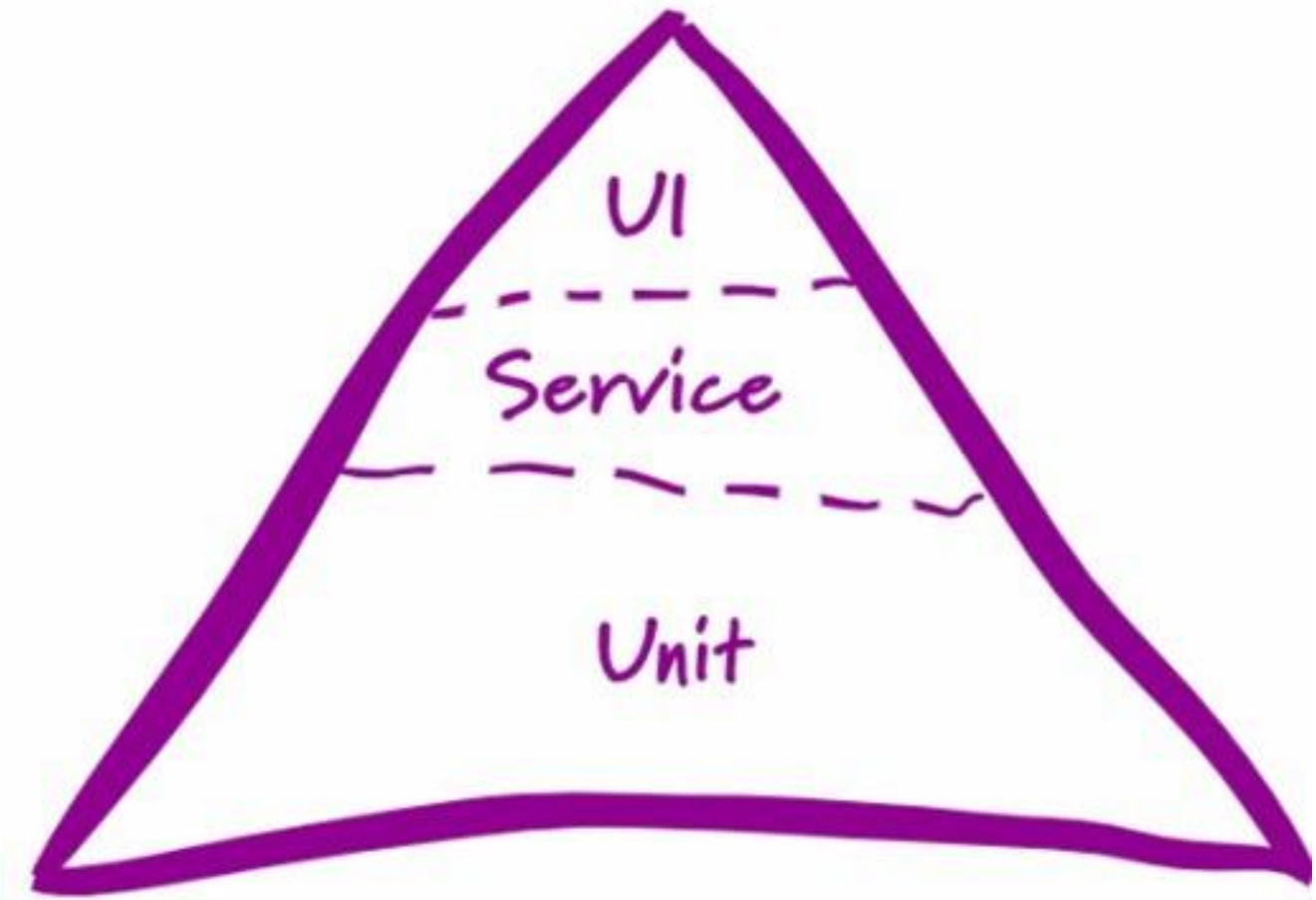
大家有什么问题吗?

- 移动互联网的应用业务测试应该怎么去做?
- 应该关注什么?
- 仅仅关注app，你绝对不是一个合格的测试人员

软件分层是测试的精髓



移动互联网也是一样



问题：你觉得UI层要关注哪些？

最重要的是设计

比如


秘密
攒50个赞，我就曝手游圈某个人的裸照
朋友
36


Shop
Stickers for messaging
Jake
Pepper
Fluffie
Pets
Charlie
Weekend
The Best
The Worst
Nom Nom

- 除了精美的设计以外，还需要遵守规则

测试提升设计敏感度

- 阅读开源设计和交互文档
- 多使用国外的应用
- 多使用竞品
- 多与真正的用户交流，而不是老板
- 真正在生活中使用产品

什么？你认为测试根本不需要关心用户体验？

项目团队中使用产品最多的角色就是测试，他不关心谁关心？至少我是这样认为的

UI测试

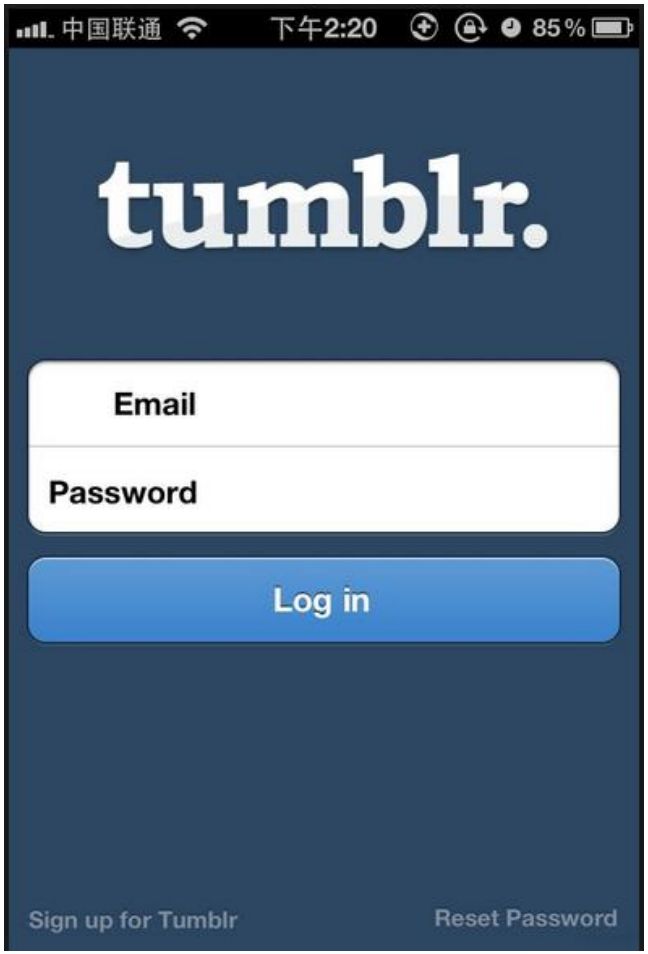
- 毋庸置疑，移动互联网应用最多的测试还是手动功能测试
- 全部自动化是银弹？非也，那是扯淡
- 试问，如果资金类产品都是自动化测试，你们还敢用吗？

UI关注点

- 大部分企业做UI自动化。什么是UI自动化呢？就是将业务的手动功能测试转换成自动化执行吗？请切勿对号入座！

关注点一

- 控件是否存在




```
TextView mTv1;
EditText mEt1;
Button mBt1;

assertNotNull(mTv1);
assertNotNull(mEt1);
assertNotNull(mBt1);
```

关注点二

- 控件位置

public class

ViewAsserts

extends Object

Summary: Methods | Inherited Methods | [Expand All]

Added in API level 1

java.lang.Object
↳android.test.ViewAsserts

Class Overview

Some useful assertions about views.

Summary

Public Methods	
static void	assertBaselineAligned (View first, View second) Assert that two views are aligned on their baseline, that is that their baselines are on the same y location.
static void	assertBottomAligned (View first, View second) Assert that two views are bottom aligned, that is that their bottom edges are on the same y location.
static void	assertBottomAligned (View first, View second, int margin) Assert that two views are bottom aligned, that is that their bottom edges are on the same y location, with respect to the specified margin.

关注点三

- 控件显示内容和状态
- 按钮上的文字
- 文本上的文字
- 单选的按钮是否被选中等

关注点四

- 全屏截屏对比
- MonkeyRunner自带API

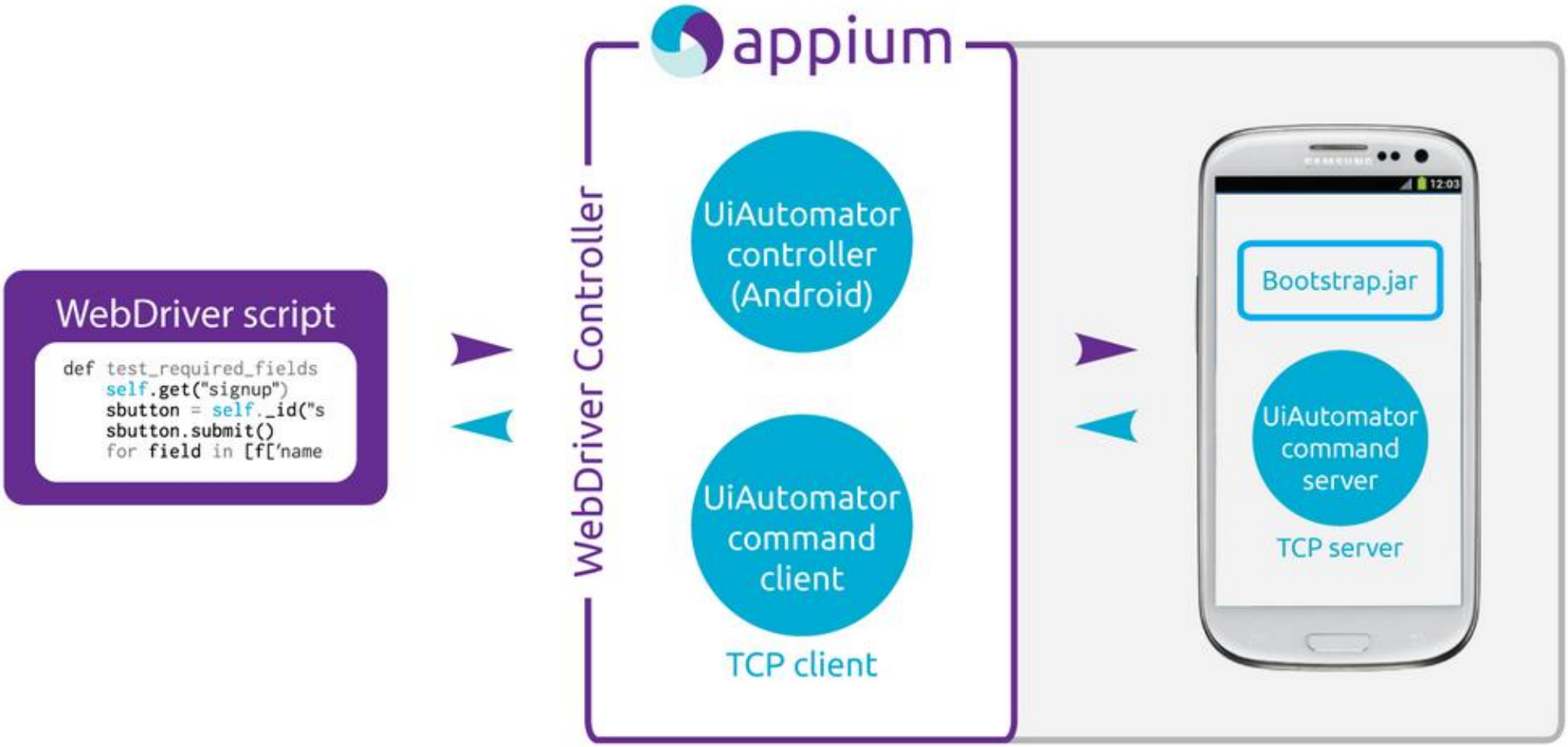
Methods	
string	convertToBytes (*string* format) Converts the current image to a particular format and returns it as a *string* that you can then access as an *iterable* of binary bytes.
tuple	getRawPixel (*integer* x, *integer* y) Returns the single pixel at the image location (x,y), as an a *tuple* of *integer*, in the form (a,r,g,b).
integer	getRawPixelInt (*integer* x, *integer* y) Returns the single pixel at the image location (x,y), as a 32-bit *integer*.
`MonkeyImage`	getSubImage (*tuple* rect) Creates a new `MonkeyImage` object from a rectangular selection of the current image.
boolean	sameAs (`MonkeyImage` other, *float* percent) Compares this `MonkeyImage` object to another and returns the result of the comparison. The `percent` argument specifies the percentage difference that is allowed for the two images to be "equal".
void	writeToFile (*string* path, *string* format) Writes the current image to the file specified by `filename`, in the format specified by `format`.

- 全屏截屏对比
- 其他第三方对比库

```
#sudo pip install PIL
def pil_image_similarity(filepath1, filepath2):
    from PIL import Image
    import math
    import operator

    image1 = Image.open(filepath1)
    image2 = Image.open(filepath2)

#    image1 = get_thumbnail(img1)
#    image2 = get_thumbnail(img2)

    h1 = image1.histogram()
    h2 = image2.histogram()

    rms = math.sqrt(reduce(operator.add,  list(map(lambda a,b: (a-b)**2, h1, h2)))/len(h1) )
    return rms

print pil_image_similarity('/Users/apple/Desktop/git/Vimi_API_Test/Compare_image_test/output.jpg','/Users/apple/Desktop/git/Vimi_API_Test/Compare_image_test/0.jpg')
```

关注点五

- 混合应用中的webview



Appium示例

```
def test(self):
    button = self.driver.find_element_by_name('buttonStartWebviewCD')
    button.click()

    self.driver.switch_to.context('WEBVIEW')

    input_field = self.driver.find_element_by_id('name_input')
    input_field.clear()
    input_field.send_keys('Appium User')
    input_field.submit()

    # test that everything is a-ok
    source = self.driver.page_source
    self.assertNotEqual(-1, source.find('This is my way of saying hello'))
    self.assertNotEqual(-1, source.find('"Appium User"'))
```

行业其他常用工具

- Robotium
- 对Android原生测试框架Instrumentation做了简单明了的封装
- Solo API使用方便
- 兼容Native和Webview

- 缺点：不能跨应用进行相关操作

BDD—Calabash

- 场景即是用例，用例即是场景

my_first.feature × calabash_steps.rb ×

```
1   Feature: Login feature
2
3     Scenario: As a valid user I can log into my app
4       check_element_exists("支付宝商户版")
5       When I press "我是店长"
6       Then I see "支付宝账户登录"
7       Then I enter "xuanyuanyang@alitest.com" into input field number 1
8       Then I enter "111111" into input field number 2
9       When I press "登录"
10      Then I see "支付宝收银"
11
12    Scenario: into app the first test case
13    When I press image button number 1
14
15
```

- 场景中步骤的封装

```ruby
# require 'byebug'
require 'date'
FORMAT = "%Y-%m-%d"
filename = "My first file"
@auth_token = nil

def convert_name( text )
  rv = "#{DateTime.now().strftime( FORMAT )}-#{text.downcase.gsub(/\s+/, '-'
  rv
end

Given /I start the login process/ do
  touch "button text:'Login'"
end

And /I acknowledge the 2 factor bug/ do
  tv = query( "textview", "text" )
  if tv[0] =~ /2\-factor authentication/
    touch "button text:'Ok'"
  end
end
```

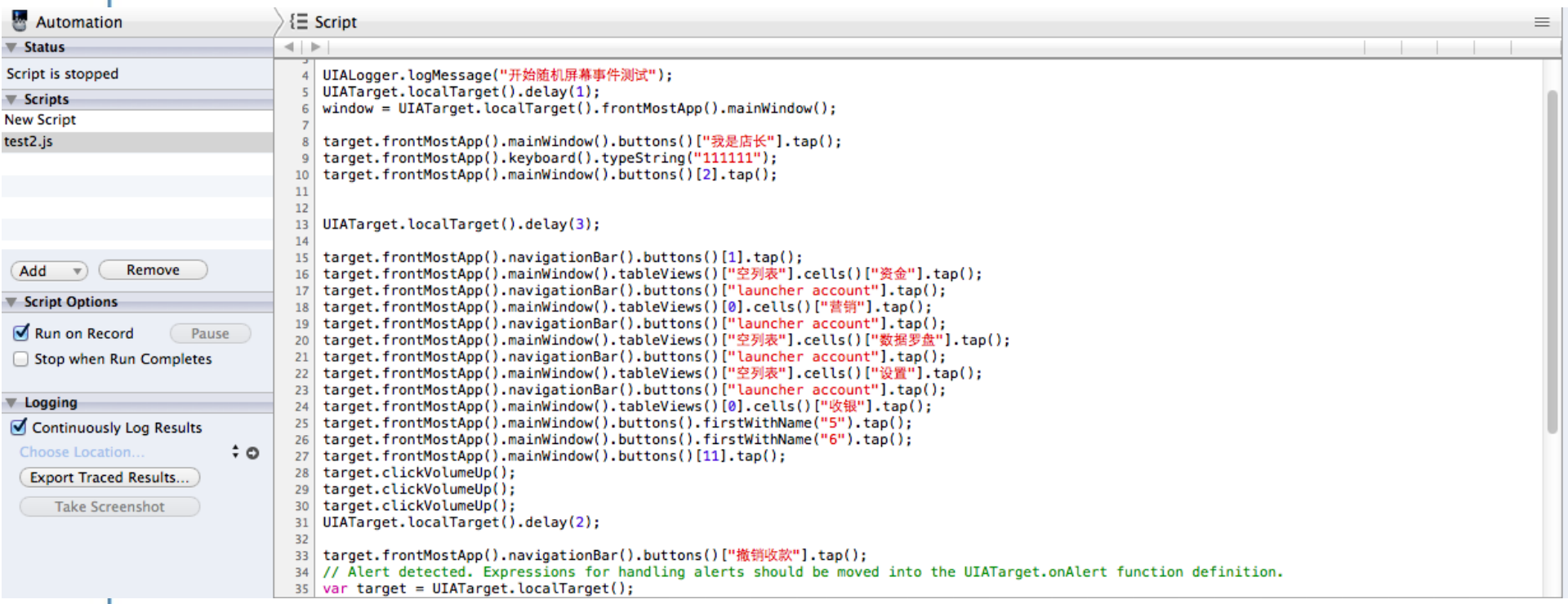
舒服的日志

```
MonkeydeMacBook-Air:calabash_demo monkey$ calabash-android run portal.apk ADB_DE
VICE_ARG=4df7c3f2600b115f
Feature: Login feature

  Scenario: As a valid user I can log into my app
   # features/my_first.feature:3
6784 KB/s (556182 bytes in 0.080s)
7120 KB/s (9098268 bytes in 1.247s)
    When I press "我是店长"
       # calabash-android-0.4.21/lib/calabash-android/steps/press_button_steps.r
b:17
    Then I see "支付宝账户登录"
          # calabash-android-0.4.21/lib/calabash-android/steps/assert_steps.rb:5
    Then I enter "xuanyuanyang@alitest.comoooooooooooo" into input field number
 1 # calabash-android-0.4.21/lib/calabash-android/steps/enter_text_steps.rb:5
    Then I enter "111111" into input field number 2
   # calabash-android-0.4.21/lib/calabash-android/steps/enter_text_steps.rb:5
    When I press "登录"
     # calabash-android-0.4.21/lib/calabash-android/steps/press_button_steps.rb:
17

1 scenario (1 passed)
5 steps (5 passed)
0m39.341s
```

Calabash同样支持iOS

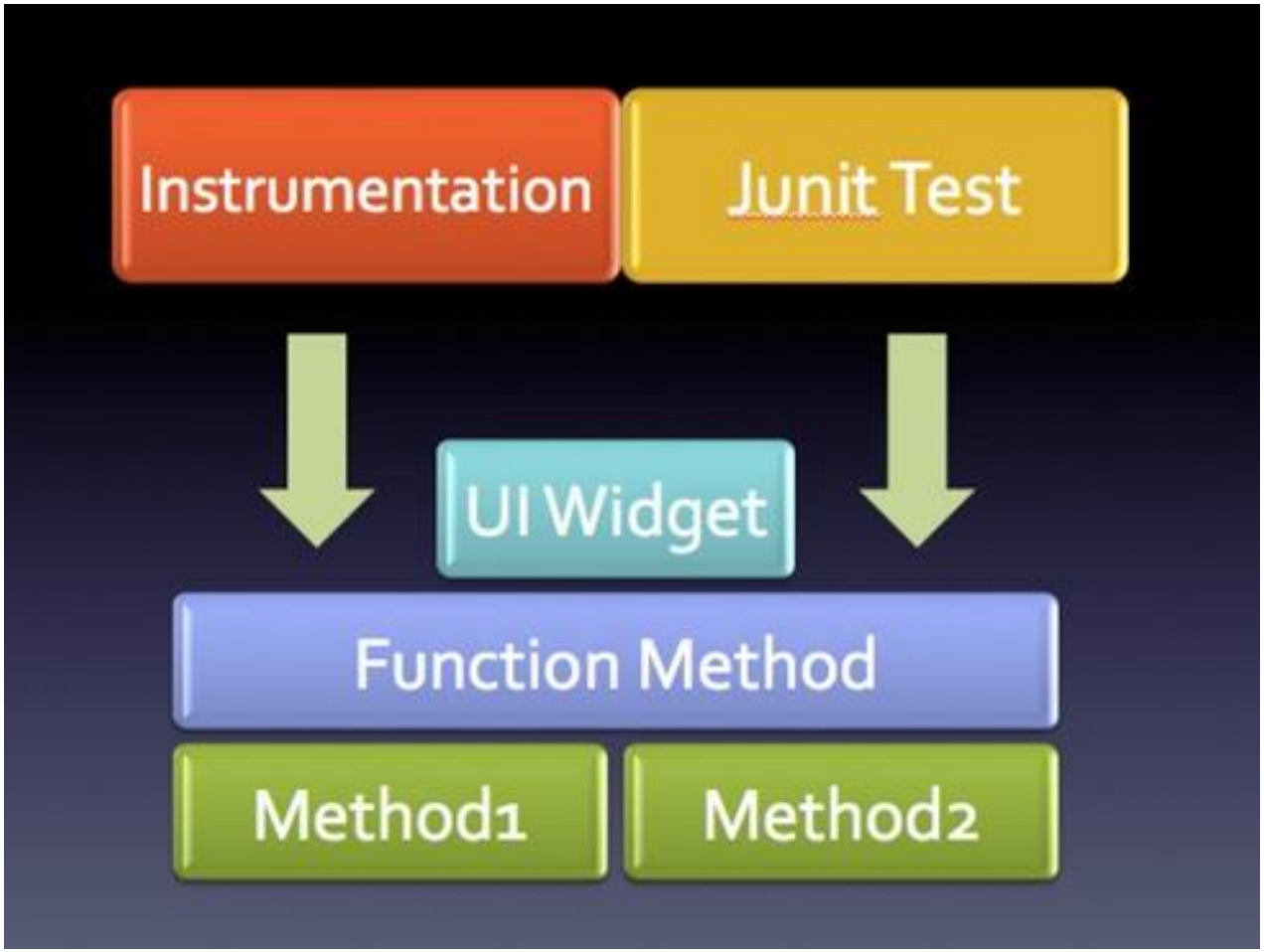
iOS UI测试工具

- Instruments



大家有什么问题吗?

App Inner测试


Instrumentation
Junit Test
UI Widget
Function Method
Method1
Method2

Inner Service测试

- 并非Server端的服务
- 可测试Service的生命周期
- 可测试Service相关的监听方法是否生效
- 可Mock相关Context对Service进行隔离测试

- Inner测试一般分成这样几类：
- Native 方法测试
- 与Server交互的方法测试
- 与业务强绑定的方法测试

Android举例

- Native方法测试
- 使用Android Junit Test
- 并非是Unit test，更多的当作一个API测试
- 比如Aactivity中intent.putExtra()，Bactivity中intent.getStringExtra()；

大部分公司的服务器接口自动化已经很完善，但这样就足够了吗？

移动互联网应用的UI自动化投入产出比大多不高

Android案例二

- 与Server交互的方法测试
- 使用Android Junit Test模拟所有和服务器交互的方法，从而在保证服务器接口正确的情况下，也保证了客户端对于数据的传输和解析也是正确的

Android案例三

- 与业务强绑定的方法测试
- 很多功能和业务的自动化不是只有从UI切入一种方法，更多的可以从UI这层皮下面的核心方法入手

比如联系人搜索功能

- 你可以从UI入手，请问你要准备多少数据，要输入多少次？头大不大
- 但你也可以选择直接启动这个activity，测试搜索的核心算法方法。是不是感觉很爽？

- 是不是发现很多控件很难找到？很多操作很难实现？
- 这就对了
- 当你将这些很难实现的操作，从App inner为切入点去看的时候，一切就迎刃而解了

UT测试

- 这有啥好说的？
- 从代码层面来讲，就是java和object－c的单元测试呗

框架推荐

- Robolectric

```
@Test
public void shouldHaveClickButtonSuccess() throws Exception {
   activity = Robolectric.buildActivity(MainActivity.class).create().get()
    testButton = (Button) activity.findViewById(R.id.button1);
    testText = (TextView) activity.findViewById(R.id.textView1);
    testButton.performClick();
    assertThat((String) testText.getText(),equalTo("the text is changed"));


}
```

- 语法更接近于robotium，但却是shadow了Activity的一种UT

这些思想和技术给我们能带来什么？

- 提升测试效率
- 将测试提前
- 能够轻松的进行各种我们想进行的测试

比如应用启动性能测试

- 你可以使用adb logcat
- 也可以写在android junit test中
- 也可以在代码中插桩

Android junit test

```
055 private long launchActivityUnderTest() {
056 long start = System.currentTimeMillis();
057 Intent i = buildIntent(PACKAGE_UNDER_TEST,
058 ACTIVITY_UNDER_TEST,
059 true);
060 Activity a = getInstrumentation().startActivitySync(i);
061 long end = System.currentTimeMillis();
062 long diff = end - start;
063 a.finish();
064 return diff;
065 }
```

代码插桩

然后可以使用logcat过滤tag，脚本分析文件就可以了

```
    @Override
    public void onCreate(Bundle savedInstanceState) {
           super.onCreate(savedInstanceState);
           Log.e("AppStartTime","AppOnCreate");
           ...
}
    @Override
    protected void onResume() {
        super.onResume();
           Log.e("AppStartTime","AppOnResume");
           ...
}
```

比如Android app性能测试

```
    public void getAppTrafficList() {
        PackageManager pm = getPackageManager();
        List<PackageInfo> pinfos = pm
                .getInstalledPackages(PackageManager.GET_UNINSTALLED_PACKAGES
                        | PackageManager.GET_PERMISSIONS);
        for (PackageInfo info : pinfos) {
            String[] premissions = info.requestedPermissions;
            if (premissions != null && premissions.length > 0) {
                for (String premission : premissions) {
                    if ("android.permission.INTERNET".equals(premission)) {
                        int uId = info.applicationInfo.uid;
                        long rx = TrafficStats.getUidRxBytes(uId);
                        long tx = TrafficStats.getUidTxBytes(uId);
                        if (rx < 0 || tx < 0) {
                            continue;
                        } else {
                            Log.e("网络流量",info.applicationInfo.loadLabel(pm)+Formatter.forma
tFileSize(this, rx+tx)
                        }

                    }
                }
            }
        }
    }
```

Android app电量

```
        @Override
        public void onReceive(Context context, Intent intent) {
            // TODO Auto-generated method stub
            int level = intent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);
            int scale = intent.getIntExtra(BatteryManager.EXTRA_SCALE, -1);
            int status = intent.getIntExtra("status", 0); // 电池状态

            // 若正在充电
            if (status == BatteryManager.BATTERY_STATUS_CHARGING)
                Log.e("电池监控", level + "电池电量11" + "正在充电");

            else
                Log.e("电池监控", level + "电池电量11" + "请及时充电");
        }

    };
```

性能全自动化

- 插桩，然后logcat过滤保存到文本

```
adb logcat -v time -v threadtime *:E | grep ActivityStartTime>StartTimeFile.txt
```

- Top或者dumpsys过滤保存到文本

```
adb shell top -n 400 | grep <your package name>Cpu_MemoryFile.txt
```

- 过滤GC到文本

```
adb logcat -v time -v threadtime *:D | grep GC>GCFile.txt
```

- 编写service，分别输出流量和电量到文本

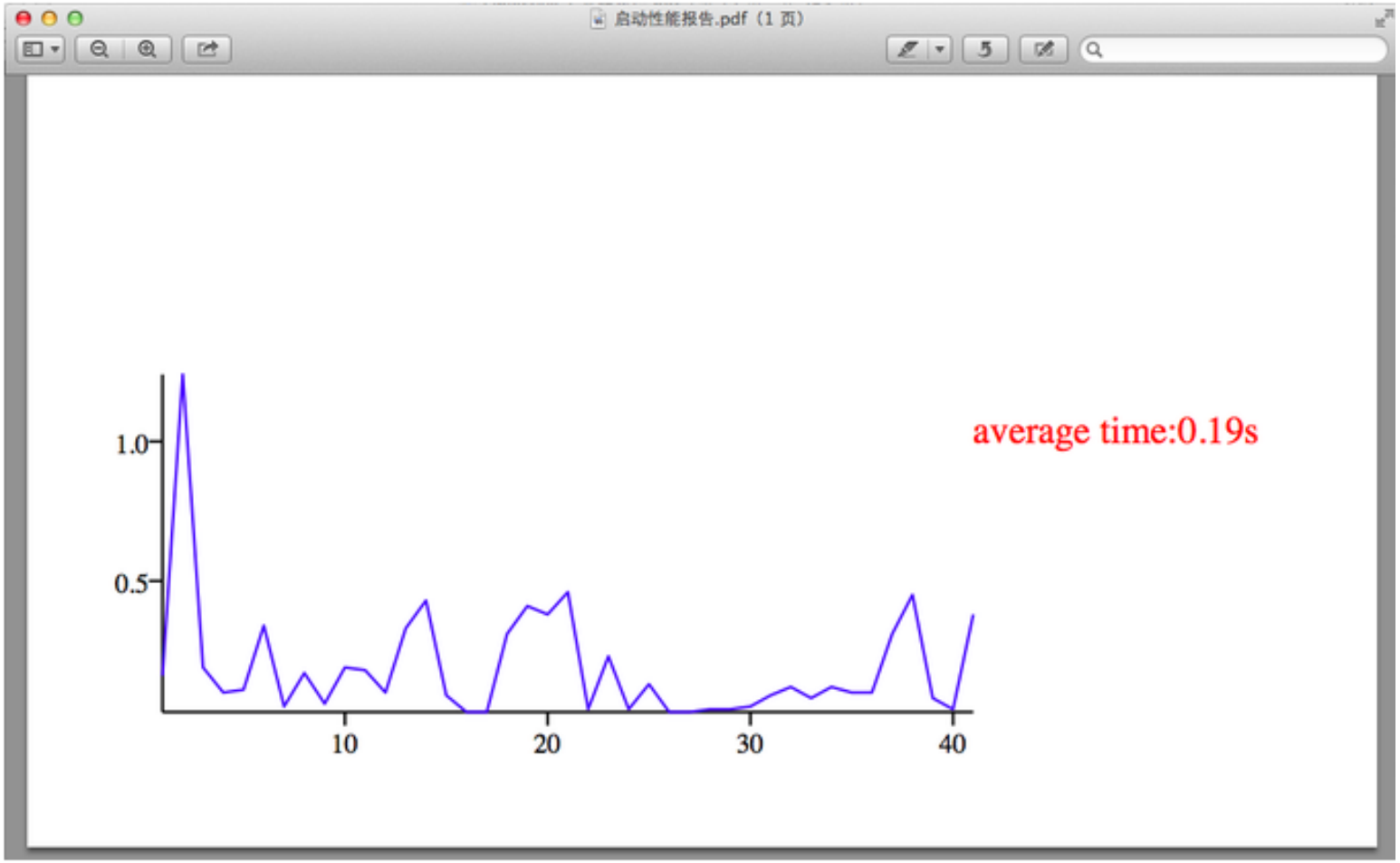
一键出报告

- 使用python第三方库制作报告，同时使用python来解析所有性能指标的数据

```
# -*- coding: utf-8 -*-
from reportlab.graphics.shapes import *
from reportlab.graphics.charts.lineplots import LinePlot
from reportlab.graphics.charts.textlabels import Label
from reportlab.graphics import renderPDF
```

```
def analysisStartFile(list):
    totalcount =0
    totaltime =0
    time_list =[]
    totalcount_list = []
    for i in range(len(list)):
        if 'AppStartTime' in list[i]:
            totalcount =totalcount+1
            totalcount_list.append(totalcount)
            if float(list[i+4].split(' ')[1][-6:])-float(list[i].split(' ')[1][-6:])>0:
                totaltime=totaltime+float(list[i+4].split(' ')[1][-6:])-float(list[i].split('
')[1][-6:])
                time_list.append(float(list[i+4].split(' ')[1][-6:])-float(list[i].split(' ')[
1][-6:]))
    return totalcount_list,'%.2f'%float(totaltime/totalcount),time_list
```

最终报告



其他性能工具和关注点

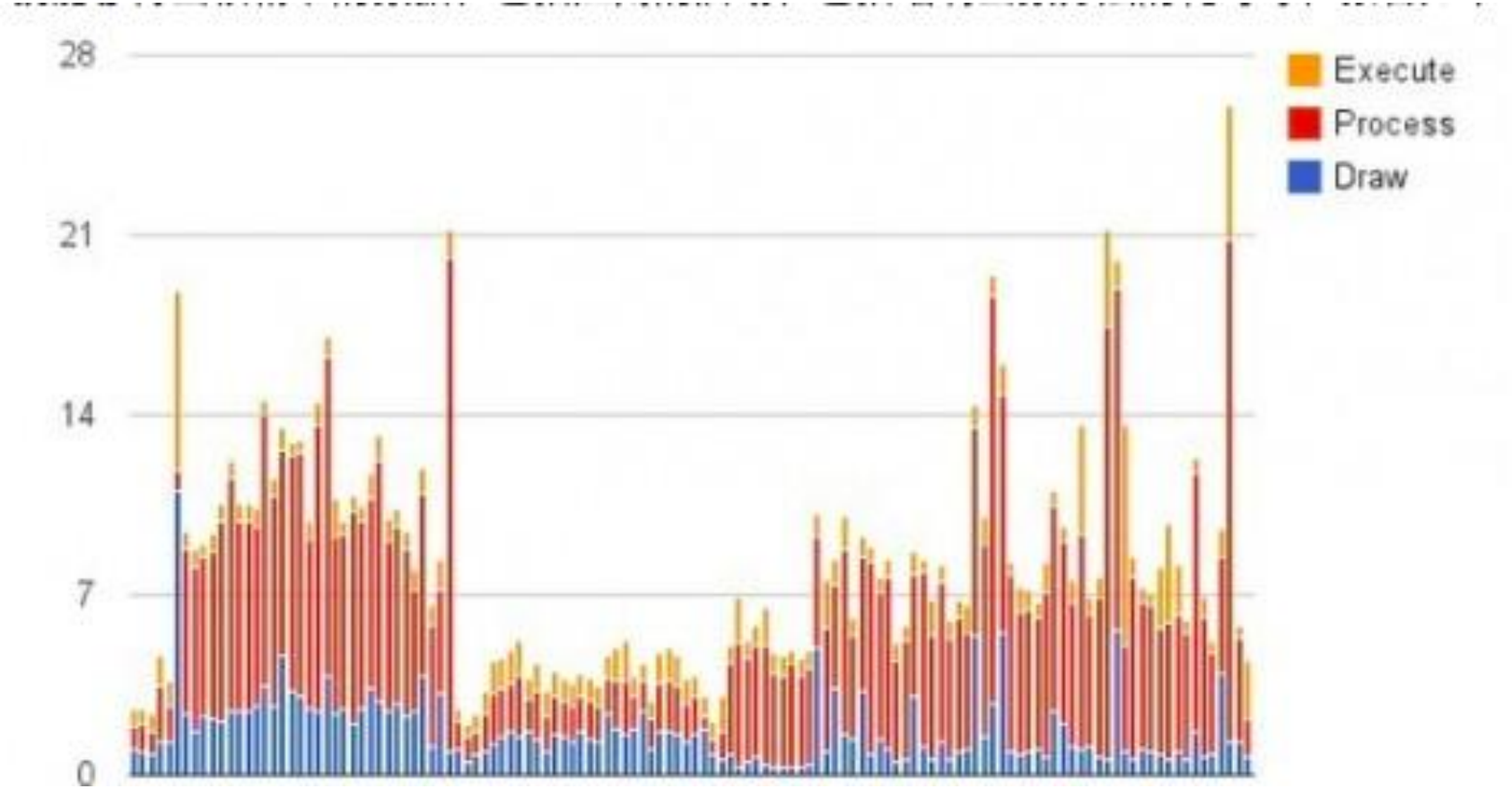
- 主要界面启动时间消耗
- 应用GC_Freed时间
- 应用GC占用空闲内存的百分比
- 应用GC的平均消耗时间
- 电量消耗
- 流量消耗
- Memory消耗平均值
- Cpu平均使用值
- MAT分析
- Lint分析
- Findbugs
- Systrace
- traceView
- Gfxinfo
- 业务测试计算代码覆盖率

Lint

Correctness

1	⚠ DuplicateIds: Duplicate ids within a single layout
3	⚠ ScrollViewSize: ScrollView size validation
2	⚠ DefaultLocale: Implied default locale in case conversion
4	❗ DuplicateDefinition: Duplicate definitions of resources
3	⚠ DuplicateIncludedIds: Duplicate ids across layouts combined with include tags
1	❗ LibraryCustomView: Custom views in libraries should use res-auto-namespace
1	❗ NewApi: Calling new methods on older versions
3	⚠ SpUsage: Using `dp` instead of `sp` for text sizes
1	⚠ Deprecated: Using deprecated resources
1	❗ MangledCRLF: Mangled file line endings

Gnxinfo



systrace


Import errors!
Categories
ms
|2929.99999 ms
|2930.19999 ms
|2930.39999 ms
|2930.59999 ms
|2930.79999 ms
|2930.99999 ms
|2931.19999 ms
|2931.39999 ms
|2931.59999 ms
onMessageReceived
handleMessageRefresh
rebuildLayerStacks
computeVisibleRegions
doComposition
postFramebuffer
onMessage...
han
han...

traceview

我们运行使用一些简单的功能之后，可以在/sdcard/下找到trace文件，使用traceview打开文件之后我们可以看到如下显示：

Name	Incl Cpu Time %	Incl Cpu Time	Excl Cpu Time %	Excl Cpu Time	Calls+Recur Calls/Total
10 com/example/android/notepad/NoteEditor$LinedEditText.onDraw (Landroid/graphics/Canvas;)V	31.3%	1873.123	0.3%	18.169	88+0
Parents					
7 android/view/View.draw (Landroid/graphics/Canvas;)V	100.0%	1873.123			88/88
Children					
self	1.0%	18.169			
11 android/widget/TextView.onDraw (Landroid/graphics/Canvas;)V	84.3%	1579.305			88/92
50 android/widget/TextView.getLineBounds (ILandroid/graphics/Rect;)I	10.9%	204.177			205/205
124 android/graphics/Canvas.drawLine (FFFFLandroid/graphics/Paint;)V	3.3%	61.637			205/205
431 android/widget/TextView.getLineCount ()I	0.5%	9.835			88/88

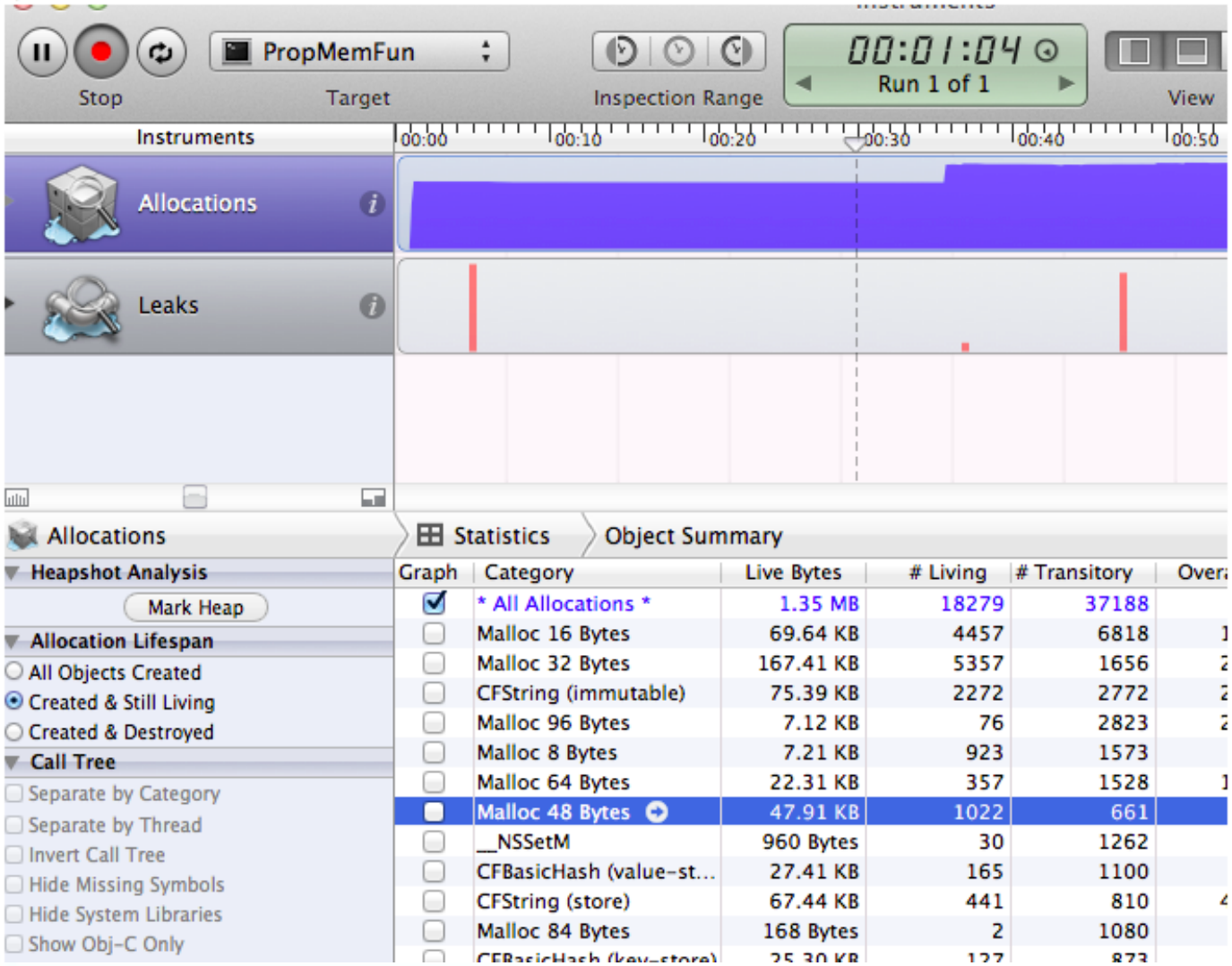
我们能够看到onDraw()以及子方法调用的顺序和消耗。当然还没有结束，我们继续进行修改尝试添加多个 `canvas.drawLine()` ，然后可以看到如下的变化：

Name	Incl Cpu Time %	Incl Cpu Time	Excl Cpu Time %	Excl Cpu Time	Calls+Recur Calls/Total	Cpu Time/Call
10 com/example/android/notepad/NoteEditor$LinedEditText.onDraw (Landroid/graphics/	25.4%	339.049	0.8%	11.233	5+0	67.810
Parents						
7 android/view/View.draw (Landroid/graphics/Canvas;)V	100.0%	339.049			5/5	
Children						
self	3.3%	11.233				
11 android/widget/TextView.onDraw (Landroid/graphics/Canvas;)V	54.2%	183.820			5/9	
27 android/widget/TextView.getLineBounds (ILandroid/graphics/Rect;)I	25.8%	87.505			90/90	
41 android/graphics/Canvas.drawLine (FFFFLandroid/graphics/Paint;)V	16.5%	56.054			270/270	
923 android/widget/TextView.getLineCount ()I	0.1%	0.437			5/5	

然后我们还原到初始代码，继续进行修改尝试增加多个 `canvas.drawColor(Color.BLUE);` ，然后可以看到如下变化：

Name	Incl Cpu Time %	Incl Cpu Time	Excl Cpu Time %	Excl Cpu Time	Calls+Recur Calls/Total
10 com/example/android/notepad/NoteEditor$LinedEditText.onDraw (Landroid/graphics/Canvas;)V	42.4%	3270.929	1.5%	114.846	38+0
Parents					
5 android/view/View.draw (Landroid/graphics/Canvas;)V	100.0%	3270.929			38/38
Children					
self	3.5%	114.846			
11 android/widget/TextView.onDraw (Landroid/graphics/Canvas;)V	46.4%	1517.457			38/42
27 android/widget/TextView.getLineBounds (ILandroid/graphics/Rect;)I	23.2%	758.809			736/736
26 android/graphics/Canvas.drawColor (I)V	22.6%	740.202			2944/3022
86 android/graphics/Canvas.drawLine (FFFFLandroid/graphics/Paint;)V	4.1%	135.154			736/736
620 android/widget/TextView.getLineCount ()I	0.1%	4.461			38/38

Instruments Leaks



移动无线应用安全

- 一类是CS结构的网络安全
- 一类是应用本身的安全性

网络安全

- 其实无疑是传统网络安全一样
- Sql注入
- Xss注入
- 网站钓鱼
- 网络拦截

网络拦截

- 工具很多
- 常用Charles
- Fiddler
- BurpSuite

- 他们都可以更改请求或者返回，从而修改用户的行为

本地安全

- 日志敏感词安全
- ContentProvider数据共享安全
- 本地数据存储
- 本地数据加密
- 混淆
- 业务安全等

常用工具

- 自己编写正则进行代码静态检查
- Fuzzer
- Drozer agent

正则静态扫描敏感词

Now Scan file: a
Match Mobile Phone Numbers:
There are all10
Match Serial Numbers:
There are all 15
Match Emails:
There are all 3
Match Phone Number:
There are all 22
Match Rank Card Id:
There are all 27
Match People Card Id:
There are all 15

Now Scan file: 5
Match Mobile Phone Numbers:
There are all10
Match Serial Numbers:
There are all 1
Match Phone Number:
There are all 4
Match Rank Card Id:
There are all 50
Match People Card Id:
There are all 1

CI是什么？

- CI是将所有测试揉捏起来
- CI在移动互联网测试中必须要有
- 我们使用jenkins

CI什么时候投入呢?

- 必须成规模的自动化用例?
- 必须得有独立的自动化团队
- 必须得有独立的编写自动化用例的时间?

CI的策略怎么制定

- 主动触发
- 被动触发
- 到底哪些测试可以上CI

- CI对我而言，仅仅是一个调度平台
- 插件可用最好，不可好也没有关系

更多信息可参考www.testerhome.com

更多吐槽和案例可见《大话移动测试——Android与iOS应用测试指南》（清华大学出版社）

更多测试行业信息可关注荔枝FM：测试小道消息（FM245329）


大话移动APP测试
Android与iOS
应用测试指南
陈群/著
大话移动APP测试
Android与iOS
应用测试指南

QA

- 微博：Monkey陳曄曄
- Weixin：monkey15chen
- QQ：383750787
- Gmail：snowangelsimon@gmail.com
- Twitter：monkeytest
- Number：18621519900

Thank you

ISTQB®让测试更专业