

[Toggle](#) [TesterHome](#)

移动性能测试 使用 adb shell 抓取 Android 性能数据

- [社区](#)
- [Bug 曝光台](#)
- [问答](#)
- [社团](#)
- [招聘](#)
- [Wiki](#)
- [酷站](#)

- [注册](#)
- [登录](#)

-

移动性能测试 使用 adb shell 抓取 Android 性能数据

[quqing](#) · 发布于 2015年12月24日 · 最后由 [cloudwind](#) 回复于 2016年07月29日 · 2722 次阅读

本帖已被设为精华帖！

[目录](#)

- [流量抓取方式也略有不同，前一种方法获取tcp流量，而且在有的设备上无法获取到数据，无法做到普适性：](#)
- [流畅度阈值的界定：按官网的建议每秒小于60帧就能感觉到不流畅，也就是说每帧的阈值=1000/60 =16ms](#)
- [app启动时间获取原理：](#)
- [计算时间差：](#)
- [获取设备的一些临界值](#)
- [命令模板](#)
- [支持实时抓取数据和异步抓取数据，异步抓取每个抓取类型会分配一个线程，异步抓取的数据以文件形式持久化到本地。](#)

前段时间在看Android客户端性能测试，处于兴趣写了个性能数据抓取的插件，可以抓取的数据有app启动时间、cpu、pss、流量上下行、流畅度等。

关于内存要说一下为什么只抓取pss，因为没有root的情况下无法获取到uss，pss是最有参考价值的（进程占用内存+按比例分配共享库占用的内存）

流量抓取方式也略有不同，前一种方法获取tcp流量，而且在有的设备上无法获取到数据，无法做到普适性：

- 改之前： shell adb -s #udid# shell cat /proc/uid_stat/#uid#/tcp_rcv adb -s #udid# shell cat /proc/uid_stat/#uid#/tcp_snd
- 改之后： shell adb -s #udid# shell cat /proc/net/xt_qtaguid/stats | grep #uid# | awk '{tx_bytes+=\$6}END{print "\$(date +%Y%m%d%H%M%S)",tx_bytes}' adb -s #udid# shell cat /proc/net/xt_qtaguid/stats | grep #uid# | awk '{tx_bytes+=\$8}END{print "\$(date +%Y%m%d%H%M%S)",tx_bytes}'

流畅度阈值的界定：按官网的建议每秒小于60帧就能感觉到不流畅，也就是说每帧的阈值=1000/60 =16ms

app启动时间获取原理：

- 获取logcat日志： shell adb -s \$1 shell logcat -v time -b events > output/\$1/latest/logcat_\$1.txt &
- 根据起始关键字和结束关键字过滤时间戳：

```
handlLogcat = new Thread(new CmdThread("./shell/appLanchTimeClient.sh " + udid + " " + beginKeywords + " " + endKeywords), "handlLogcat_" + udid);
```

计算时间差：

```
startTime = CmdInvoke.run("tail -n 1 output/" + udid + "/latest/start_" + udid + ".txt | awk '{print $2}'");
endTime = CmdInvoke.run("tail -n 1 output/" + udid + "/latest/end_" + udid + ".txt | awk '{print $2}'");
lanchTime = DateUtil.getDiffTwoTime(DateUtil.parsehmsS(endTime), DateUtil.parsehmsS(startTime), "S");
```

获取设备的一些临界值

```
heapgrowthlimit=`adb -s $udid shell getprop dalvik.vm.heapgrowthlimit`
heapstartsize=`adb -s $udid shell getprop dalvik.vm.heapstartsize`
heapsize=`adb -s $udid shell getprop dalvik.vm.heapsize`
echo "#单个应用程序最大内存限制" > output/$udid/devMemLimit.txt
echo "heapgrowthlimit=$heapgrowthlimit >> output/$udid/devMemLimit.txt"
echo "#单个java虚拟机最大的内存限制" >> output/$udid/devMemLimit.txt
echo "heapsize=$heapsize >> output/$udid/devMemLimit.txt"
echo "#每帧刷新耗时阈值(ms)" >> output/$udid/devMemLimit.txt
#60FPS<==>每帧<=16ms FPS=1000/(Draw+Process+Execute)
#打开android手机 设置->开发者选项->GPU呈现模式分析
echo "gfxinfo=16.00" >> output/$udid/devMemLimit.txt
```

命令模板

```
public class CmdTemplate {
    public static String PID = "adb -s #udid# shell dumpsys meminfo #packageName# | grep pid | awk '{print $5}'";
    public static String UID = "adb -s #udid# shell dumpsys package #packageName# | grep packageSetting | cut -d \"\\\"\" -f2 | cut -d \"\\\"\" -f1";

    /**实时抓取**/
    // TIMESTAMP(yyyymmddHHMMSS) CPU(%) PID/PACKAGE
    public static String CPU_USAGE = "adb -s #udid# shell dumpsys cpuinfo | grep #filter# | awk '{print \"$(date +%Y%m%d%H%M%S)\", $1, $2}' | grep #pid#";

    // TIMESTAMP(yyyymmddHHMMSS) PSS(KB) PACKAGE
    public static String PSS = "adb -s #udid# shell dumpsys meminfo | awk '/process:/, /adjustment:/ {if(i>1) print x; x=$0; i++;}' | grep #filter# | grep #pid# | awk '{print \"$(date +%Y%m%d%H%M%S)\", $1, $3}'";

    // TIMESTAMP(yyyymmddHHMMSS) RX_BYTES(BYTE)
    public static String RX_BYTES = "adb -s #udid# shell cat /proc/net/xt_qtaguid/stats | grep #uid# | awk '{rx_bytes+= $6} END {print \"$(date +%Y%m%d%H%M%S)\", rx_bytes}'";

    // TIMESTAMP(yyyymmddHHMMSS) TX_BYTES(BYTE)
    public static String TX_BYTES = "adb -s #udid# shell cat /proc/net/xt_qtaguid/stats | grep #uid# | awk '{tx_bytes+= $8} END {print \"$(date +%Y%m%d%H%M%S)\", tx_bytes}'";

    /**异步抓取**/
    // TIMESTAMP(yyyymmddHHMMSS) PID VSS(KB) RSS(KB) PACKAGE
    public static String SAVE_TOP_INFO = "adb -s #udid# shell top -n 1 -d 0 | grep #filter# | awk '{print \"$(date +%Y%m%d%H%M%S)\", $1, $6, $7, $10}' >> output/#udid#/latest/top_#udid#.txt";

    // TIMESTAMP(yyyymmddHHMMSS) CPU(%) PID/PACKAGE
    public static String SAVE_CPU_INFO = "adb -s #udid# shell dumpsys cpuinfo | grep #filter# | awk '{print \"$(date +%Y%m%d%H%M%S)\", $1, $2}' >> output/#udid#/latest/cpu_#udid#.txt";

    // TIMESTAMP(yyyymmddHHMMSS) PSS(KB) PACKAGE
    public static String SAVE_PSS_INFO = "adb -s #udid# shell dumpsys meminfo | awk '/process:/, /adjustment:/ {if(i>1) print x; x=$0; i++;}' | grep #filter# | awk '{print \"$(date +%Y%m%d%H%M%S)\", $1, $3}' >> output/#udid#/latest/pss_#udid#.txt";

    // TIMESTAMP(yyyymmddHHMMSS) RX_BYTES(BYTE) TX_BYTES(BYTE)
    public static String SAVE_TRAFFIC_INFO = "adb -s #udid# shell cat /proc/net/xt_qtaguid/stats | grep #uid# | awk '{rx_bytes+= $6}{tx_bytes+= $8} END {print \"$(date +%Y%m%d%H%M%S)\", rx_bytes, tx_bytes}' >> output/#udid#/latest/traffic_#udid#.txt";

    // Draw(MS) Process(MS) Execute(MS)
    public static String SAVE_GFX_INFO = "adb -s #udid# shell dumpsys gfxinfo #packageName# | awk '/Execute/, /hierarchy/ {if(i>1) print x; x=$0; i++;}' | sed /^[[:space:]]*/d | awk '{if(length($0)==16) print $1, $2, $3}' >> output/#udid#/latest/gfxinfo_#udid#.txt";
}
```

支持实时抓取数据和异步抓取数据，异步抓取每个抓取类型会分配一个线程，异步抓取的数据以文件形式持久化到本地。

pss抓取内容

```
20151201163741 41414 com.yzt
20151201163741 22044 com.yzt:remote
20151201163741 21048 com.yzt:pushservice
20151201163741 19309 com.yzt:push
20151201163747 48010 com.yzt
20151201163747 22047 com.yzt:remote
20151201163747 21132 com.yzt:pushservice
20151201163747 19333 com.yzt:push
```

cpu抓取内容

```
20151201163739 0.8% 9819/com.yzt:remote:
20151201163739 0% 9699/com.yzt:
20151201163743 5.2% 9699/com.yzt:
20151201163743 1% 9819/com.yzt:remote:
20151201163743 0% 14177/com.yzt:pushservice:
```

流量抓取内容

```
20151201163740 1119317 1313028
20151201163743 1119317 1313028
20151201163746 1429836 1328958
20151201163749 1509060 1337427
20151201163752 2525902 1381403
```

每帧耗时抓取内容

```
1.30 1.25 0.53
0.63 0.62 1.64
0.24 0.51 0.44
0.25 1.76 0.40
0.25 1.55 0.43
0.25 0.53 0.37
0.49 1.51 2.12
0.45 0.67 0.27
0.43 0.71 2.83
0.43 0.76 3.04
```

报表展示：

[26 个赞](#)

转载文章时务必注明原作者及原始链接，并注明「发表于 TesterHome」，并不得对作品进行修改。

如果觉得我的文章对您有用，请随意打赏。您的支持将鼓励我继续创作！

[打赏支持](#)

共收到 **41** 条回复



[monkey](#) · #1 · [2015年12月24日](#)

使用markdown更新下格式。。 = =



[quqing](#) · #2 · [2015年12月24日](#) 作者

#1楼 [@monkey](#) 已修改



[kinget007](#) · #3 · [2015年12月24日](#)

谢谢分享！



[Lihuazhang](#) · #4 · [2015年12月24日](#)

#2楼 [@quqing](#) markdown的格式，如果编程语言，代码高亮要有吧。现在的markdown 有点敷衍哦。



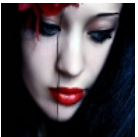
[quqing](#) · #5 · [2015年12月24日](#) 作者

#4楼 [@lihuazhang](#) 已更新



[testly](#) · #6 · [2015年12月24日](#)

感谢分享



[alice](#) · #7 · [2015年12月25日](#)

写的不错呦！



[xiaoluosun](#) · #8 · [2015年12月25日](#)

多谢分享



[shixue33](#) · #9 · [2015年12月25日](#)

感谢楼主分享，但是感觉有点乱的感觉.....= = !

另外问一下，最后一张图，那根红色的标准线是怎么画出来的？



[quqing](#) · #10 · [2015年12月25日](#) 作者 [1个赞](#)

[#9楼 @shixue33](#) highchart官网有资料的，yAxis: {min: 0,plotLines :[{color:'red',dashStyle:'solid',value:16,width:1,label:{text:'threshold:16ms',align:'left',x:0,style:{fontSize:'12px',color:'red'}}}],title: {text: 'time(ms)/per frame'}},



[xwgoss](#) · #11 · [2015年12月25日](#)

对于APP启动，我们采用的是adb shell am start -W packagename/activity指令，回头研究下两者的差别性



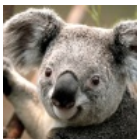
[adfgzhzhang](#) · #12 · [2015年12月29日](#)

awk: not found



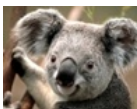
[quqing](#) · #13 · [2015年12月29日](#) 作者

[#12楼 @adfghzhang](#) Linux、mac系统，直接敲awk，按回车看下提示



[adfghzhang](#) · #14 · [2015年12月29日](#)

[#13楼 @quqing](#) 已找到原因，我使用的手机shell里面没有这个功能，模拟器可以正常使用。另外发了个邮件给您的163邮箱，盼复



[y1i1n1](#) · #15 · [2015年12月30日](#)

小白弱弱的问一句，这个脚本怎么运行？Windows可以使用吗？



[quqing](#) · #16 · [2015年12月30日](#) 作者

[#15楼 @y1i1n1](#) windows有adb环境就可以使用



[quqing](#) · #17 · [2015年12月30日](#) 作者

有人会疑惑，`/proc/net/xt_qtaguid/stats|grep #uid#`，每个uid对应的两条数据有什么区别呢？

Lines with `cnt_set==0` are for background data

Lines with `cnt_set==1` are for foreground data

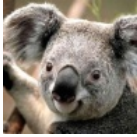
Total traffic is a sum of both

参考资料：<http://stackoverflow.com/questions/15163549/interpreting-android-xt-qtaguid-stats>



[rickyzhan](#) · #18 · [2015年12月31日](#)

赞！



[groot](#) · #19 · [2016年01月04日](#)

赞~



[wyb199026](#) · #20 · [2016年01月06日](#)

赞，回复mark，抽空学习一下



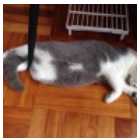
[xushizhao](#) · #21 · [2016年01月08日](#)

我在群里没找到你，能给我个QQ号不？



[quqing](#) · #22 · [2016年01月08日](#) 作者

[#21楼 @xushizhao](#) 393472146



[codeskyblue](#) · #23 · [2016年01月11日](#)

testerhome这个社区做的不错嘛。文章写的也不错，有些地方我简单的说明下

- cpu

文章中写cpu的获取是这样子的shell `dumpsys cpuinfo` 不过这种方法延迟很厉害，直接解析`/proc/<pid>/stat`会更好一些。

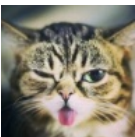
- 帧率

文章中似乎提到的很少，我猜应该是通过`dumpsys SurfaceFlinger --latency SurfaceView`获取到的，这个命令的输出有一些垃圾数据也需要过滤掉。获取帧率还有很多其他办法，比如root的手机可以通过调用service call `SurfaceFlinger 1013`获取当前已经刷新过的帧数，稍微计算下就能计算出帧率了。



[quqing](#) · #24 · [2016年01月11日](#) 作者

[#23楼 @codeskyblue](#) 帧率抓取的命令文中有：adb -s #udid# shell dumpsys gfxinfo #packageName#|awk '/Execute/,/hierarchy/{if(i>1)print x;x=\$0;i++}'|sed /[:space:]]*\$/d|awk '{if(length(\$0)==16)print \$1,\$2,\$3}'



[m13890](#) · #25 · [2016年01月13日](#)

有个问题我一直不清楚，adb指令是否会对正在运行的设备造成性能压力。相对比通过adb来获取实时数据，在设备上开发专门监测这些数据然后传回的App哪个更合适？



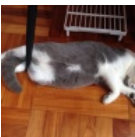
[quqing](#) · #26 · [2016年01月13日](#) 作者

没法说哪个更合适。我的方案实行简单，无需兄弟部门配合，但对控制机的压力比较大；打埋点收集监控数据，需公司层面支持，如果能解决这个前提，当然是后者比较好。



[appium_liang](#) · #27 · [2016年01月15日](#)

谢谢分享，已收藏



[codeskyblue](#) · #28 · [2016年01月25日](#)

[#24楼 @quqing](#) 使用dumpsys gfxinfo packagename 对于游戏好像测试不出来



[quqing](#) · #29 · [2016年01月25日](#) 作者

[#28楼 @codeskyblue](#) 打开android手机 设置->开发者选项->GPU呈现模式分析，这个设置了没？



[sz_cw](#) · #30 · [2016年01月28日](#)

[#13楼 @quqing](#) windows下执行命令，提示awk不是内部命令。。。请问怎么解决、



[quqing](#) · #31 · [2016年01月28日](#) 作者

[#30楼 @sz_cw](#) 之前是基于mac、linux写的，后来为了支持windows，提取样本数据的脚本用java写的内置方法替换掉了



[sz_cw](#) · #32 · [2016年02月01日](#)

[#31楼 @quqing](#) 此前在BAT的外包公司执行过这样的脚本，但获取数据不没有UDP,也一直在寻找方法，现在楼主写出来，大赞，无奈在win下操作一直报错，望能指点下：

```
C:\Users\Administrator>adb shell cat /proc/net/xt_qtaguid/stats | findstr 10146 | awk '{tx_bytes+=$6}END{print '$<date+%Y%m%d%H%M%S',tx_bytes}'
awk: '{tx_bytes+=$6}END{print
awk: ^ invalid char ''' in expression
```



[quqing](#) · #33 · [2016年02月02日](#) 作者

[#32楼 @sz_cw](#) 这篇文章的内容不是最新的，只针对mac、linux，后面为了兼容windows，过滤方法是通过代码实现的；如果非要用脚本实现，看这篇文章，回帖有答案，<https://testerhome.com/topics/3856>



[jira](#) · #34 · [2016年03月22日](#)

[#23楼 @codeskyblue](#) 解析proc/pid/stat会有好多数据，怎么算cpu的使用率？

—— 来自TesterHome官方 [安卓客户端](#)



[ansonwoo](#) · #35 · [2016年03月24日](#)

读取proc/pid/stat下整个手机的CPU. 怎么区分出是待测app消耗的CPU资源呢?



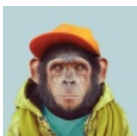
[fantasyty](#) · #36 · [2016年03月31日](#)

[@quqing](#) 帧率那个，你的采样间隔是多少



[quqing](#) · #37 · [2016年03月31日](#) 作者

[#36楼 @fantasyty](#) 自定义的，单位：秒



[fenfenzhong](#) · #38 · [2016年03月31日](#)

[#37楼 @quqing](#) 你好楼主，有几个关于FPS的问题：

1. 执行一次命令会得到最近128帧的渲染情况，怎么决定采样间隔？比如1秒后再次执行命令，可能只有30帧是新的（在输出的底部），其他98帧都是旧数据，那这种情况怎么处理？又如果间隔太大，那中间其实已经过去很多帧，FPS统计本来是一个连贯的过程，那这样的数据还有意义吗？
2. 最后面那张报表，我看帧数已经超过了128，如果报表是静态的（数据全在一个list里，只画一次图，那么还是问题1），如果报表是动态的，要怎么实现？



[quqing](#) · #39 · [2016年03月31日](#) 作者

[#38楼 @fenfenzhong](#) 系统文件抓取的原始数据是每一帧耗费多少时间，以此换算出fps。测试数据本来就是采样的，我的理解是能反映其基本本面，就到达目的了



[doublecchen](#) · #40 · [2016年05月07日](#)

非常棒，但是一般测试具体场景下的性能，所以会把自动化和adb命令结合起来



[cloudwind](#) · #41 · [2016年07月29日](#)

adb做性能，收集数据感觉也很牛逼。

需要 [登录](#) 后方可回复, 如果你还没有账号请点击这里 [注册](#)。

相关话题

- [android 端取 cpu.fps.men.wifi/gprs 流量等值](#)
- [shell 管理 monkey 压力测试续——监控方案重构及 MCM 监控维护](#)
- [shell 脚本通过 dumpsys SurfaceFlinger --latency 数据计算 FPS 和评价流畅度。](#)
- [Android 性能测试实践 \(四\) 流量](#)
- [移动端性能测试方法总结](#)

作者



会员

quqing (测男)

第 6469 位会员 / 2015-12-19

云端

[26 个赞](#)

共收到 41 条回复

[打赏支持](#)

[有新回复！点击这里立即载入](#)



[关于](#) / [活跃用户](#) / [中国移动互联网测试技术大会](#) / [反馈](#) / [Github](#) / [API](#) / [酷站](#) / [帮助推广](#)

TesterHome 移动测试社区，由众多移动测试工作者维护，致力于推进国内测试技术。Inspired by RubyChina

2017/10/10

使用 adb shell 抓取 Android 性能数据 · TesterHome

友情链接 [WeTest](#) [腾讯质量开放平台](#) / [InfoQ](#) / [测试之道](#) / [测试窝](#) / [百度测试吧](#)
[简体中文](#) / [正體中文](#) / [English](#)

