一、get和post的区别

1.编码方式：

GET 请求只能进行 url 编码，**GET**只接受ASCII字符的参数的数据类型，而 POST 支持多种编码方式

GET 比 POST 更不安全参数直接暴露在 URL 上，POST 放在 Request body （请求体）中。

四种常见的 POST 提交数据方式：1.application/x-www-form-urlencoded、multipart/form-data、application/json、text/xml

3.GET产生一个TCP数据包，GET方式的请求，浏览器会把http header和data一并发送出去，服务器响应200（返回数据）；

POST产生两个TCP数据包，浏览器先发送header，服务器响应100 continue，浏览器再发送data，服务器响应200 ok（返回数据）

4.数据传输大小：数据传输大小都会有限制，只是POST的传输大小相对于GET来说比较大

5.缓存角度：get是查询，在一定时间内，查询信息不会那么快的变化，因此它通常用于缓存。

post是修改，向服务器提交表单，POST请求不能被缓存

http有哪些请求方式？

get：从服务器获取一份文档

Post：向服务器发送要处理的数据

Put：将请求的主体部分放在服务器上

head：从服务器获取文档的首部

delete：从服务器删除一份文档

Trace：对可能经过代理服务器传送到服务器上去的报文进行追踪

options：决定可以在服务器上执行哪些哪些方法

二、请求响应

请求行，包含了请求方法、URL地址和协议版本

报文头，报文头包含若干个属性，格式为“属性名:属性值”，服务端据此获取客户端的信息

报文体，它将一个页面表单中的组件值通过param1=value1&param2=value2的键值对形式编码成一个格式化串，它承载多个请求参数的数据

响应头：报文协议及版本； 状态码及状态描述；

响应报文头，也是由多个属性组成；

响应报文体，即我们真正要的“干货”。

三、响应状态码

* 1xx 消息，一般是告诉客户端，请求已经收到了，正在处理，别急...
* 2xx 处理成功，一般表示：请求收悉、我明白你要的、请求已受理、已经处理完成等信息.
* 3xx 重定向到其它地方。它让客户端再发起一个请求以完成整个处理。
* 4xx 处理发生错误，责任在客户端，如客户端的请求一个不存在的资源，客户端未被授权，禁止访问等。
* 5xx 处理发生错误，责任在服务端，如服务端抛出异常，路由出错，HTTP版本不支持等。
* **200状态码 服务器请求正常**

**301状态码：被请求的资源已永久移动到新位置。服务器返回此响应（对 GET 或 HEAD 请求的响应）时，会自动将请求者转到新位置。**

**302状态码：请求的资源临时从不同的URI响应请求，但请求者应继续使用原有位置来进行以后的请求**

**401状态码：请求要求身份验证。 对于需要登录的网页，服务器可能返回此响应。**

**403状态码：服务器已经理解请求，但是拒绝执行它。服务器拒绝了你的地址请求，很有可能是你根本就没权限访问网站，就算你提供了身份验证也没用**

**404状态码：请求失败，请求所希望得到的资源未被在服务器上发现。**

**408链接超时**

**503状态码：由于临时的服务器维护或者过载，服务器当前无法处理请求。**

**服务器501错误。**服务器501错误是服务器还是不具有请求功能的，而且501错误原因是没有实施的，可以用来HttpWebRequest指定一个UserAgent来试试的，有时候你可以换电脑来测试一下的。

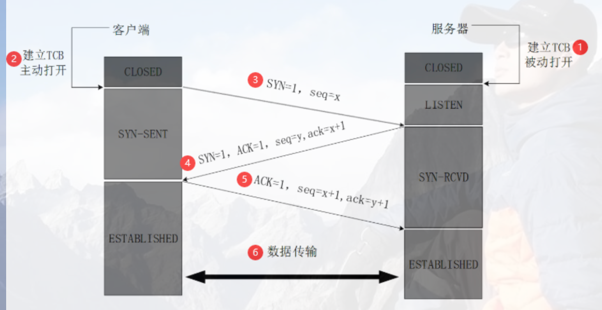
**服务器502错误。**这是服务器上的一个错误网关 ，因此说它是无效的，我们在出现了服务器502错误问题的时候，最好是先清除下缓存或者是在服务器上进行刷新试试的，因为502错误牵扯的问题也是很多的，最好是让程序们来去在服务器上下文章。

**服务器503错误**。服务不可用是的一种状态，那么在服务器503错误出现了之后，大家不必担心的， 服务器或许就是正在维护或者暂停了，你可以联系一下服务器空间商。还有的时候cpu占用的频率大导致的

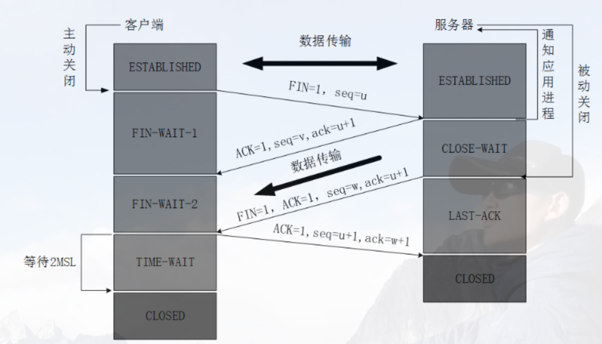
**服务器504错误。**这是代表着网关超时是现象出现了。504错误问题是一个不好办的问题，当然你必须尝试着和网站官方获得联系，认真的去检查不同的电脑简的ip传输的状况。而且这个504错误要专业的负责人才能去解决。

**服务器505错误**。http的版本是不受支持的，一般的请款下浏览器的默认都是1.x 的版本的， 如果出现了HTTP 1.1版本的，那么你需要在Internet 选项的高级下进行设置的。

四、tcp三次握手/四次握手



* 1.服务器准备：TCP服务器进程先创建传输控制块TCB，时刻准备接受客户进程的连接请求，此时服务器就进入了LISTEN（监听）状态
* 2.客户端准备：TCP客户进程也是先创建传输控制块TCB
* **3.第一次握手**：客户端向服务器发出连接请求报文，报文首部中的同步标志SYN=1，同时生成一个初始序列号 seq=x ，此时，TCP客户端进程进入了 SYN-SENT（同步已发送状态）状态。TCP规定，SYN报文段（SYN=1的报文段）不能携带数据，但需要消耗掉一个序号  SYN=1，seq=x
* **4.第二次握手**：TCP服务器收到请求报文后，如果同意连接，则发出确认报文。确认报文中确认标志 ACK=1，SYN=1，确认号是ack=x+1，同时也要为自己初始化一个序列号 seq=y，此时，TCP服务器进程进入了SYN-RCVD（同步收到）状态。这个报文也不能携带数据，但是同样要消耗一个序号.   ACK =1,SYN=1,ack=x+1，seq=y
  + **5.第三次握手**：TCP客户端收到确认后，还要向服务器再次给出确认。确认报文的确认标志ACK=1，确认号ack=y+1，自己的序列号seq=x+1，此时，TCP连接建立，客户端进入ESTABLISHED（已建立连接）状态。TCP规定，ACK报文段可以携带数据，但是如果不携带数据则不消耗序号  ACK=1，ack=y+1，seq=x+1
* 6.连接成功：当服务器收到客户端的确认后也进入ESTABLISHED状态，此后双方就可以开始通信了

****

建立一个连接需要三次握手，而终止一个连接要经过四次握手

FIN=1，seq=u

ACK=1,ack=u+1，seq=v

ACK=1，FIN=1,ack=u+1，seq=w

ACK=1,ack=w+1,seq=u+1

* 客户端由于已经没有任何需要发送给服务端的消息了，所以发起断开客户端到服务端的通道请求
* 服务端收到该请求后同意了 至此客户端到服务端的单项通道断开
* 服务端这个时候不会立刻朝客户端发器请求说那我也断开到你家的通道吧，服务端需要想想我手上还有没有需要发送给客户端的消息，如果还有的话，那我不能立马断开，先把数据发完才能断

等服务端检查完毕之后发送也没有数据要给客户端了，这个时候就会朝客户端发起断开服务端到客户端的通道请求

* 客户端同意该请求，至此四次挥手完成

1. 客户端进程数据发送完毕，停止发送数据，发出连接释放报文，释放标志位FIN=1，序列号seq=u（等于前面已经传送过来的数据的最后一个字节的序号加1），此时，客户端进入FIN-WAIT-1（终止等待1）状态。 TCP规定，FIN报文段即使不携带数据，也要消耗一个序号。
2. 服务器收到连接释放报文，在确认自己受到所有数据后，发出确认报文同意断开，确认标志位ACK=1，确认号ack=u+1，并且带上自己的序列号seq=v，此时，服务端就进入了CLOSE-WAIT（关闭等待）状态。TCP服务器通知高层的应用进程，客户端向服务器方向的一条通道马上就关闭了。这时候TCP处于半连接状态，即客户端已经没有数据要发送了，但是服务器若发送数据，客户端依然要接受。这个状态还要持续一段时间，也就是整个CLOSE-WAIT状态持续的时间。
3. 客户端收到服务器的确认请求后，断开自己的一条通道。此时，客户端就进入FIN-WAIT-2（终止等待2）状态，在此期间还需要接收服务器传过来的数据，等待服务器发送连接释放报文
4. 服务器的数据也发送完毕后，就向客户端发送连接释放报文，FIN=1，ack=u+1，由于在半关闭状态，服务器很可能又发送了一些数据，假定此时的序列号为seq=w，此时，服务器就进入了LAST-ACK（最后确认）状态，等待客户端的确认。
5. 客户端收到服务器的连接释放报文后，确认自己全部接收完数据后，必须发出确认，ACK=1，ack=w+1，而自己的序列号是seq=u+1，此时，客户端就进入了TIME-WAIT（时间等待）状态。注意此时TCP连接还没有释放，必须经过2∗MSL（最长报文段寿命）的时间后，当客户端撤销相应的TCB后，才进入CLOSED状态。
6. 服务器只要收到了客户端发出的确认，立即进入CLOSED状态。可以看到，服务器结束TCP连接的时间要比客户端早一些

四、TCP和UDP的区别

TCP 是面向连接的，UDP 是面向无连接的，可以承担一对多，广播或多播的协议，需要资源少，速度快，应用在直播和实时游戏实现视频传输协议

UDP程序结构较简单

TCP 是面向字节流的，UDP 是基于数据报的

TCP 保证数据正确性，UDP 可能丢包

TCP 保证数据顺序，UDP 不保证

TCP 为什么是可靠连接？

通过 TCP 连接传输的数据无差错，不丢失，不重复，且按顺序到达。

TCP 报文头里面的序号能使 TCP 的数据按序到达

报文头里面的确认序号能保证不丢包，累计确认及超时重传机制

TCP 拥有流量控制及拥塞控制的机制

TCP 的顺序问题，丢包问题，流量控制都是通过滑动窗口来解决的

拥塞控制时通过拥塞窗口来解决的

五、深拷贝浅拷贝赋值

赋值的本质就是将一个对象的内存空间地址赋值给一个变量，让变量指向该内存空间地址。原始列表改变，被赋值的变量也会做相同的改变。其实就是对‘对象'的引用

**浅拷贝**（shallowCopy）只是增加了一个指针指向已存在的内存地址只拷贝父对象，没有拷贝子对象。且浅拷贝是创建一块新的内存空间，但是内存空间内的元素的地址均是父对象元素的地址的拷贝，当父对象内部的子对象发生改变时，拷贝对象的内部的子对象也会跟着改变。

**深拷贝**（deepCopy）是增加了一个指针并且申请了一个新的内存，使这个增加的指针指向这个新的内存，完全的拷贝了父对象及子对象，同时指向一个新的内存空间地址

五、http和https的区别

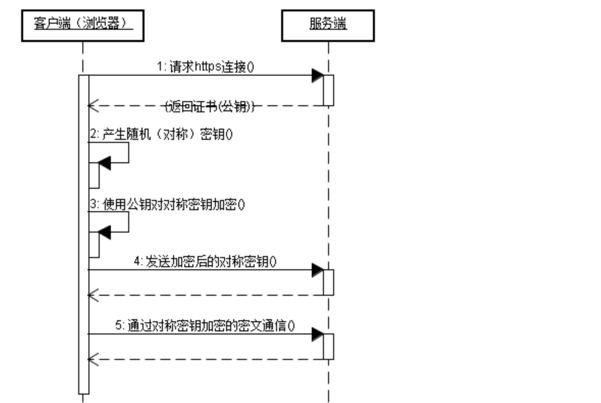
　1、https协议需要到ca申请证书，一般免费证书较少，因而需要一定费用。

　　2、http是超文本传输协议，信息是明文传输，https则是具有安全性的ssl加密传输协议。

　　3、http和https使用的是完全不同的连接方式，用的端口也不一样，前者是80，后者是443。

　　4、http的连接很简单，是无状态的；HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，比http协议安全。

通信过程：



六、linux常用命令

1.ls

Mkdir

Cd

Cat cp mv移动或重命名   rm 删除 rm -r 递归删除。rm -f 强制删除

Grep查找某个字符串。 pwd显示当前目录

七、常见的崩溃原因

常见原因：网络切换、网络带宽、设备多样性系统和分辨率兼容、代码（数组越界、oom、内存泄漏、空指针、多线程并发）

测试崩溃：内存不足、压力测试、数据不兼容、转横屏、多语言

八、adb命令

adb logcat -v time -s AndroidRuntime

adb logcat -v threadtime | find “BaseDetector"adb shell pm clear com.cleanmaster.mguard查看启动时间:adb shell am start -W -n com.cleanmaster.mguard/com.keniu.security.main.MainActivity查看/data/anr/有无log：adb shell cd /data/anr ls查看浮在最上层activity 方法1：1、 adb shell 2、 dumpsys activity | grep "mFocusedActivity” 查看帧率： "adb -s " + deviceName + " shell dumpsys gfxinfo " + packageNameadb shell top -m 10 -s cpu

九、自动化

Krunner

KRunner的底层是基于uiautomator2和WebDriverAgent和yolov5+opencv-python来实现的

* **Android Native部分：**Android native部分是使用uiautomator2，而uiautomator2的底层是基于google uiautomator，原理是在手机上运行了一个http服务器，将UIautomator中的功能，主要有UIDevice、UISelector、UIObject和 UIScrollable，开放出来，然后再将这些http接口，封装成Python库，然后你直接用这个库就可以了。
* **Android H5部分：**如果你的webview是google android系统的原生的，uiautomator2有支持webview的测试，其原理是使用chromedriver，通过chrome的remote debugging来控制chrome的。但是如果你的webview是定制化的，那通过u2底层的chromedriver是无法定位元素的，这时候需要通过注入js代码来实现元素属性的获取和操作。大概原理是通过adb forward，开通chrome Remote Debugging端口，然后创建一个websocket连接，将你的js代码注入到Runtime的程序中。
* **iOS部分：**通过WebDriverAgent在iOS端实现一个webdriver server，然后通过iproxy将端口（default port是8100）映射到对应的执行机器上，借助这个server我们就可以远程控制iOS设备。
* **AI识别模块：**图像识别模块依赖我单独写的一个[kmatch-service](https://git.corp.kuaishou.com/ks_video_qa/kmatch-service)（目标检测和OCR推理服务）。AI目标检测方式识别依赖yolov5，用户提供训标注好的训练集, 触发平台进行训练，生成模型，进行推理上线；OCR方式识别是依赖MMU的识别服务；图像匹配方式识别是是opencv 的Feature Matching和Template Matching两个模块。

[**核心特性**](https://video-quality.corp.kuaishou.com/docs/md/krunner#/?id=%e6%a0%b8%e5%bf%83%e7%89%b9%e6%80%a7)

* 支持Android & iOS & H5 全平台的测试需求
* 高效精准可靠的元素定位方式
* AI识别引擎，支持AI目标检测 /OCR / 图像匹配等方式识别元素
* 提供脚手架命令，一键生成脚本工程模板
* 自动标记生成用例操作流，快速定位用例失败原因
* 高效&可配置化的异常弹窗处理机制
* 支持多机联动操作模式
* 完善的测试用例分层机制，充分实现用例的复用性
* 采用框架和用例分离机制，实现多项目快速接入
* 双端u2/wda服务自管理，自动恢复

脚手架主要是提供一套命令，自动生成对应的测试脚本工程，然后业务方可以在这个工程的基础上编写自己的测试用例，使用者只用专注用例本身的编写，不用再去写setup，tearDown等管理用例生命周期的方法，用例的组织管理脚手架生成的工程都帮你做好了。当然如果你想定制化你的用例组织，你也可以继承后重新里面的方法

测试用例管理是基于unittest进行管理的，PageObject是被测app各个页面控件属性和基本操作，以页面为单位进行组织，针对这个页面上的所有信息，所有控件，相关操作都放到一个文件中。后面如果某个控件属性发生了变化，不用在测试用例中修改，直接在PO里修改

[**Retry机制**](https://video-quality.corp.kuaishou.com/docs/md/krunner#/?id=retry%e6%9c%ba%e5%88%b6)

元素定位重试：我们的脚本只用罗列我们的操作步骤，元素的查找的健壮性全部交给框架去处理。例如框架会帮你查找对应的元素，如果没有出现则会触发查找重试逻辑，直到它出现才进行对应的操作，比如点击，滑动，输入。

失败用例重试：用例执行失败或者用例执行异常时，也会触发重试机制（开关控制），可以设置重试次数，最终的结果决定用例是否执行通过。该实现是通过重写unittest中的run方法，替换test\_method()的方法名进行实现

[**U2和WDA服务管理**](https://video-quality.corp.kuaishou.com/docs/md/krunner#/?id=u2%e5%92%8cwda%e6%9c%8d%e5%8a%a1%e7%ae%a1%e7%90%86)

框架集成了u2，从u2的初始化到u2进程的回收，session的管理；wda方面，框架提供wda从编译到iproxy端口映射，wda server启动校验，wda client的初始化，session断线重连，到最后wda和proxy进程的回收。上面所有的一切krunner都帮你做好了，用例编写者只用专心写用例就可以了。这样大大降低编写用例的难度，提高用例编写效率。

[**插件支持**](https://video-quality.corp.kuaishou.com/docs/md/krunner#/?id=%e6%8f%92%e4%bb%b6%e6%94%af%e6%8c%81)

登录插件、埋点插件、覆盖率插件、kmonkey插件

安卓基础：

长视频feed加载过程，客户端定义接口请求时机，请求数据，数据返回后将数据结构保存到内存序列化成一个对象，写入xml中，使用recycle view，根据返回的数据类型确定卡片样式，然后将数据绑定到卡片上，进行渲染，然后根据图片网络库返回图片，将图片展示出来

上下滑框架是page view ,相当于一个容器，页面内的各个组件的展示位置和规则，

横屏上下滑，转横屏的时候将数据塞进去，展示出来，告诉播放器要起播，播放器根据photo中的码率信息等决定使用哪个播放地址进行播放

卡片上进度条展示规则：相当于一个组件，展示在卡片上，注册监听事件，从播放器获取到播放进度后展示在卡片上

Activity的生命周期分析说明

onCreated()是activity第一次被启动时执行的，主要是初始化一些变量（启动第一步）

onRestart()是当前activity重新被启动时调用的；绑定一些监听器等（可见状态）

onStart()是activity界面被显示出来的时候执行的（可见状态）。

onResume()是当该activity与用户能进行交互时被执行，（能获取到焦点）

onPause()是另一个activity被启动，当前的activity就被暂停了，一般在该函数中执行保存当前的数据（被遮挡住）。

onStop()表示另一个activity被启动完成时，当前activity对用户同时又完全不可见时才调用的（不可见状态）。

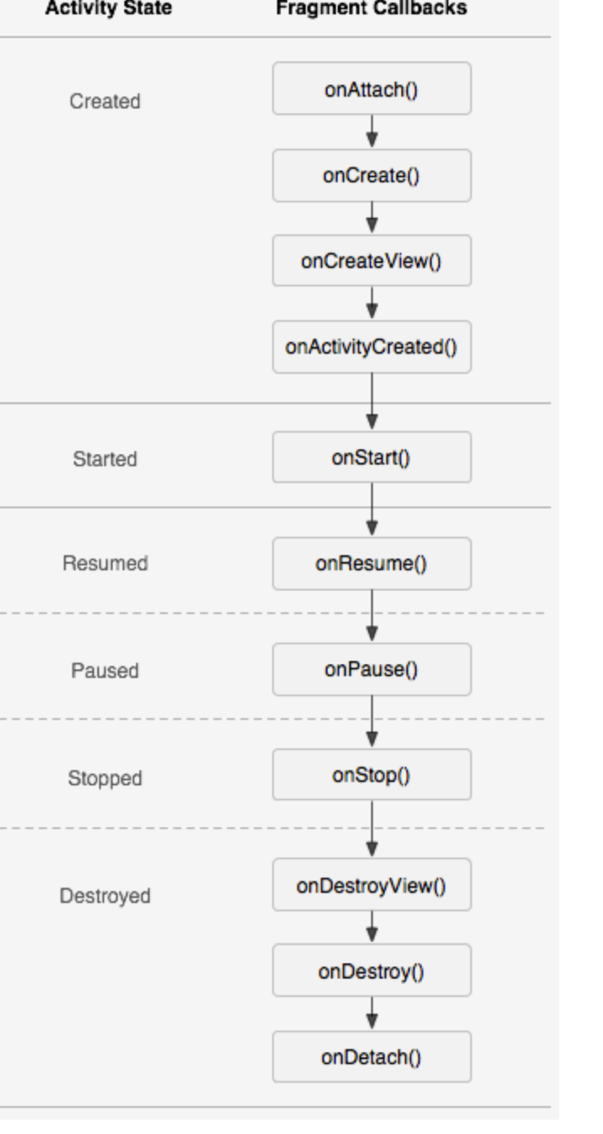
onDestroy()是退出当前activity时调用的，当然如果程序中调用finish()或者说android系统当前资源不够用时就会被调用（销毁）。

当用多个activity在执行时，这时候android系统会自动将这些activity压入栈中并且总是显示最顶的那个activity，这个栈在android叫做task，但是这个栈只支持压入和弹出操作，不支持排序插入等操作。

Activity的7个生命周期函数中的onStop()函数被调用时是在其对应的activity被另外的activity完全遮挡的时候，如果只有部分遮挡，则不会被调用。部分遮挡一般是以消息activtiy的形式出现，这个只需在AndroidManifest.xml中将其对于的activity的主题设置theme中更改即可。当系统资源不够时，其中onPause(),onStop(),onDestroy()是有可能被系统kill掉的，但其它4个是不会被kill掉。

实验验证：两个Activity相互跳转，ActivityA设置一个button跳转到ActivityB

一开始程序运行时是首先调用第1个activity的onCreate(),onStart(),onResume()函数，然后按下button后，调用第1个activity的onPause(),和第2个activity的onCreate(),onStart(),onResume()函数，然后接着调用第1个activity的onStop()，当按下系统返回键时，这时会调用第2个activity的onPause(),接着是第1个activity的onRestart(),onStart(),onResume(), 最后是调用第2个的onStop()和onDestroy().

<https://www.cnblogs.com/seven-sky/p/5316962.html>

onAttached() —— 当fragment被加入到activity时调用（在这个方法中可以获得所在的activity）。

onCreateView() —— 当activity要得到fragment的layout时，调用此方法，fragment在其中创建自己的layout(界面)。

onActivityCreated() —— 当activity的onCreated()方法返回后调用此方法

onDestroyView() —— 当fragment中的视图被移除的时候，调用这个方法。

onDetach() —— 当fragment和activity分离的时候，调用这个方法。

Mvp：

Model：将数据存到内存序列化，保存实例对象

view：程序的UI界面，用于向用户展示数据以及接收用户的输入，还要承担一些生命周期的工作

Presenter：是Model和View之间的桥梁，为了让结构变得更加简单，View并不能直接对Model进行操作

IOC框架

软件系统在没有引入IOC容器之前，如图1所示，对象A依赖于对象B，那么对象A在初始化或者运行到某一点的时候，自己必须主动去创建对象B或者使用已经创建的对象B。无论是创建还是使用对象B，控制权都在自己手上。

    软件系统在引入IOC容器之后，这种情形就完全改变了，如图3所示，由于IOC容器的加入，对象A与对象B之间失去了直接联系，所以，当对象A运行到需要对象B的时候，IOC容器会主动创建一个对象B注入到对象A需要的地方。

    通过前后的对比，我们不难看出来：对象A获得依赖对象B的过程,由主动行为变为了被动行为，控制权颠倒过来了，这就是“控制反转”这个名称的由来。

通过引入IOC容器，利用依赖关系注入的方式，实现对象之间的解耦。

Rxjava

分享，对错误的收集完善，分享错误的原因都能收集起来，方便定位分享错误原因

拉起分享面板，选择分享渠道，请求share/any，组装分享数据，交给分享接收方微信sdk

share/init服务端请求reco，share/any请求对抗中台，请求photo相关信息

依赖倒置：业务方携带指定参数传给sdk，sdk发起请求share/init，sdk接收数据传给业务方，业务方根据数据准备面板，分享面板数据拿到后sdk触发业务方拉起分享面板，用户点击渠道根据渠道ID，sdk请求share/any，分享sdk接收数据组装数据，跟微信sdk通信，然后微信sdk给分享sdk分享结果成功失败的回调，分享sdk告诉业务方分享结果，业务方上报埋点

对抗，打开h5的有上报，打开1000，封禁后打开数下降，

口令：拉起前台的时候，剪贴板的内容是否匹配startup的正则表达式并清空剪贴板，匹配则请求接口，不匹配则不请求

安卓四大组件：

Activity 、service、content provider、broadcast receiver

堆和栈的区别：

1.内存管理方式：栈由操作系统自动分配释放，堆的申请和释放需要程序控制

2.空间大小不同：栈小于堆，栈64字节的windows为1MB，Linux为10M

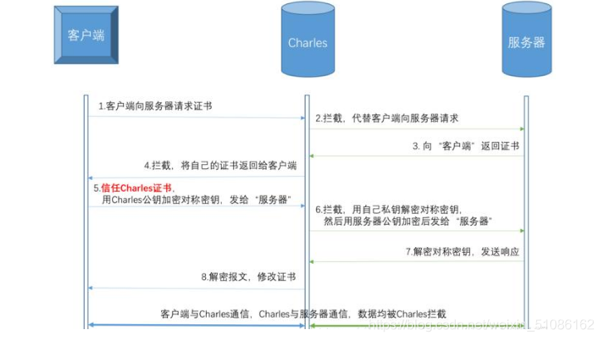
3.生长方向：堆内存地址方向从低到高，栈从高到低

4.分配方式：堆动态分配（c/c++库函数活运算符分配），栈动态（alloca函数分配）和静态分配（操作系统完成）

5.分配效率：栈效率比较高

6.存放内容不同：栈存放的内容函数返回的是地址、实参、局部变量、寄存器内容，堆存放的内容由程序员填充

十、Charles工作原理



1.客户端发起https请求  2.charles拦截像服务端发起请求  3.服务端返回证书（公钥）。4.Charles拦截，发送自己的证书（公钥）给客户端  5.客户端信任证书，生成一个随机的对称密钥，然后用Charles公钥加密，发送给服务端。6.charles拦截，然后用自己的私钥解密，用服务器的公钥加密后发送给服务器  7.服务器用私钥解秘对称密钥，发送客户端响应。8。charlse拦截，并发送自己的证书给客户端  9.链接建立charlse拿到了服务器证书的公钥，和客户端与服务端协商的对称密钥 ，之后就可以解密和修改加密的报文了