一、UDID (**Unique Device Identifier**)

UDID的全称是Unique Device Identifier，顾名思义，它就是苹果IOS设备的唯一识别码，它由40个字符的字母和数字组成。在很多需要限制一台设备一个账号的应用中经常会用到。在iOS5中可以获取到设备的UDID，后来被苹果禁止了。

二、UUID（Universally Unique Identifier）

UUID是Universally Unique Identifier的缩写，中文意思是通用唯一识别码。它是让分布式系统中的所有元素，都能有唯一的辨识资讯，而不需要透过中央控制端来做辨识资讯的指定。这样，每个人都可以建立不与其它人冲突的 UUID。在此情况下，就不需考虑数据库建立时的名称重复问题。苹果公司建议使用UUID为应用生成唯一标识字符串。

开发者可以在应用第一次启动时调用一 次，然后将该串存储起来，替代UDID来使用。但是，如果用户删除该应用再次安装时，又会生成新的字符串，所以不能保证唯一识别该设备。使用UUID，就要考虑应用被删除后再重新安装时的处理。一个解决的办法是：UUID一般只生成一次，保存在iOS系统里面，如果应用删除了，重装应用之后它的UUID还是一样的，除非系统重置 。但是不能保证在以后的系统升级后还能用（如果系统保存了该信息就能用）。

三、MAC Address

用来表示互联网上每一个站点的标识符，采用十六进制数表示，共六个字节（48位）。其中，前三个字节是由IEEE的注册管理机构

RA负责给不同厂家分配的代码(高位24位)，也称为"编制上唯一的标识符”（Organizationally Unique Identifier)，后三个字节(低位24位)由各厂家自行指派给生产的适配器接口，称为扩展标识符（唯一性）。

MAC地址在网络上用来区分设备的唯一性，接入网络的设备都有一个MAC地址，他们肯定都是不同的，是唯一的。一部iPhone上可能有多个MAC地址，包括WIFI的、SIM的等，但是iTouch和iPad上就有一个

WIFI的，因此只需获取WIFI的MAC地址就好了，也就是en0的地址。

MAC地址就如同我们身份证上的身份证号码，具有全球唯一性。这样就可以非常好的标识设备唯一性，类似与苹果设备的UDID号，通常的用途有：1）用于一些统计与分析目的，利用用户的操作习惯和数据更好的规划产品；2）作为用户ID来唯一识别用户，可以用游客身份使用app又能在服务器端保存相应的信息，省去用户名、密码等注册过程。

使用Mac地址生成设备的唯一标识主要分三种：

1、直接使用“MAC Address”

2、使用“MD5(MAC

Address)”

3、使用“MD5(Mac Address+bundle_id)”获得“机器＋应用”的唯一标识（bundle_id 是应用的唯一标识）

在iOS7之后，如果请求Mac地址都会返回一个固定值。

四、IDFA（**identifierForIdentifier**）

广告标示符，适用于对外：例如广告推广，换量等跨应用的用户追踪等。

是iOS 6中另外一个新的方法，提供了一个方法advertisingIdentifier，通过调用该方法会返回一个NSUUID实例，最后可以获得一个UUID，由系统存储着的。不过即使这是由系统存储的，但是有几种情况下，会重新生成广告标示符。如果用户完全重置系统（(设置程序 -> 通用 -> 还原 -> 还原位置与隐私)，这个广告标示符会重新生成。另外如果用户明确的还原广告(设置程序-> 通用 -> 关于本机 -> 广告 -> 还原广告标示符)，那么广告标示符也会重新生成。关于广告标示符的还原，有一点需要注意：如果程序在后台运行，此时用户“还原广告标示符”，然后再回到程序中，此时获取广 告标示符并不会立即获得还原后的标示符。必须要终止程序，然后再重新启动程序，才能获得还原后的广告标示符。

在同一个设备上的所有App都会取到相同的值，是苹果专门给各广告提供商用来追踪用户而设的，用户可以在 设置|隐私|广告追踪 里重置此id的值，或限制此id的使用，故此id有可能会取不到值，但好在Apple默认是允许追踪的，而且一般用户都不知道有这么个设置，所以基本上用来监测

推广效果，是戳戳有余了。

注意：由于idfa会出现取不到的情况，故绝不可以作为业务分析的主id，来识别用户。

代码：

```
#import <AdSupport/AdSupport.h>


  NSString *adId = [[[ASIdentifierManager sharedManager] advertisingIdentifier] UUIDString];
```

五、IDFV（**identifierForVendor**）

Vindor标示符，适用于对内：例如分析用户在应用内的行为等。

是给Vendor标识用户用的，每个设备在所属同一个Vender的应用里，都有相同的值。其中的Vender是指应用提供商，但准确点说，是通过BundleID的DNS反转的前两部分进行匹配，如果相同就是同一个Vender，例如对于com.somecompany.appone,com.somecompany.apptwo

这两个BundleID来说，就属于同一个Vender，共享同一个idfv的值。和idfa不同的是，idfv的值是一定能取到的，所以非常适合于作为内部用户行为分析的主id，来标识用户，替代OpenUDID。

注意：如果用户将属于此Vender的所有App卸载，则idfv的值会被重置，即再重装此Vender的App，idfv的值和之前不同。

代码：

```
  NSString *idfv = [[[UIDevice currentDevice] identifierForVendor] UUIDString];
```

六、OPEN

UDID

每台iOS设备的OpenUDID是通过第一个带有OpenUDID SDK包的App

生成，如果你完全删除全部带有OpenUDID SDK包的App（比如恢复系统等），那么OpenUDID会重新生成，而且和之前的值会不同，相当于新设备；

优点是没有用到MAC地址；不同设备能够获取各自唯一的识别码，保证了唯一性，可以用于以往UDID的相关用途；从代码分析OpenUDID的获取，识别码获取方便并且保存谨慎。缺点是当将设备上所有使用了OpenUDID方案的应用程序删除，且设备关机重启，xcode彻底清除并重启，重装应用程序去获取OpenUDID，此时OpenUDID变化，与之前不一样了，所有OpenUDID应用卸载后，由UIPasteboard保存的数据即被清除，重装故会重新获取新的OpenUDID。

那么当因为用户干预或者恶意程序，致使UIPasteboard数据清除，从而导致OpenUDID被删除，重装也会获取新的OpenUDID。

OpenUDID生成唯一识别码的代码：

```
1.  unsigned char result[16];
2.
        const charchar *cStr = [[[NSProcessInfo processIn
    fo] globallyUniqueString] UTF8String];
3.      CC_MD5( cStr, strlen(cStr), result );
4.      _openUDID = [NSString stringWithFormat:
5.
                @"%02x%02x%02x%02x%02x%02x%02x
    %02x%02x%02x%02x%02x%02x%02x%02x%02x%08x
    ",
6.              result[0], result[1], result[2], result[3],
7.              result[4], result[5], result[6], result[7],
8.              result[8], result[9], result[10], result[11],
9.
                result[12], result[13], result[14], result[15]
    ,
10.             arc4random() % 4294967295];
```

/
****

****
**********/

一.UDID(Unique Device Identifier)

UDID是Unique Device Identifier的缩写,中文意思是设备唯一标识.

在很多需要限制一台设备一个账号的应用中经常会用到,在Symbian时代,我们是使用IMEI作为设备的唯一标识的,可惜的是Apple官方不允许开发者获得设备的IMEI.

ios5 sdk中的获取方法:

[UIDevice currentDevice] uniqueIdentifier]

uniqueIdentifier在UIDevice.h中的定义如下:

@property(nonatomic,readonly,retain) NSString *uniqueIdentifier __OSX_AVAILABLE_BUT_DEPRECATED(__MAC_NA,__MAC_NA,__IPHONE_2_0,__IPHONE_5_0); // a string unique to each device based on various hardware info.

意思是iOS2.0以上及iOS5.0以下的系统可用,但不建议使用.Apple有可能在ios5.0之后删除该函数.

经过测试,未越狱的iPhone,系统版本为5.0.1,依然可以获取UDID.

但是我们需要注意的一点是,对于已越狱了的设备,UDID并不是唯一的.使用Cydia插件UDIDFaker,可以为每一个应用分配不同的UDID.

所以UDID作为标识唯一设备的用途已经不大了.

二.UUID(Universally Unique Identifier)

UUID是Universally Unique Identifier的缩写,中文意思是通用唯一识别码.

由网上资料显示,UUID是一个软件建构的标准,也是被开源软件基金会(Open Software Foundation,OSF)的组织在分布式计算环境(Distributed Computing Environment,DCE)领域的一部份.UUID的

目的,是让分布式系统中的所有元素,都能有唯一的辨识资讯,而不需要透过中央控制端来做辨识资讯的指定.

根据以上定义可知,同一设备上的不同应用的UUID是互斥的,即能在改设备上标识应用.但是并没有明确指出能标识出装有同一应用的不同设备,但是根据我推测,这个UUID应该是根据设备标识和应用标识生成唯一标识,再经过加密而来的(纯推测).

iOS中获取UUID的代码如下:

```
1 -(NSString*) uuid { 2 CFUUIDRef puuid = CFUUIDCreate( nil ); 3 CFStringRef uuidString = CFUUIDCreateString( nil, puuid ); 4 NSString * result = (NSString *)CFStringCreateCopy( NULL, uuidString); 5 CFRelease(puuid); 6 CFRelease(uuidString); 7 return [result autorelease]; 8 }
```

虽然UUID是官方提出的一种替代UDID的建议方案,但网上有资料说UUID不能保证在以后的系统升级后(IOS6,7)还能用.

经过我测试目前,UUID在IOS4和IOS5下均可以使用,而且UUID每次生成的值都不一样,需要开发者自行保存UUID.

如果使用UUID为标识保存用户的资料在网络上,当用户重装软件后,UUID的值就可能会发生改变(基本上可说是百分百会发生改变),用户则无法重新下载原来的网络资料.

三.一个可行的解决方案

经过上述的探讨,我们不难发现,无论是使用UDID或是UUID,我们的目的通常都是为了让用户可以自动注册,而不需要账号密码.而使用UDID和UUID作为用户的ID也并不是毫无缺陷.

现在网上有一现成的解决方案,使用设备的Mac地址,因为Mac地址也是唯一的.unix有系统调用可以获取Mac地址.但有些事情需要注意:

1.iPhone可能有多个Mac地址,wifi的地址,以及SIM卡的地址.一般来讲,我们取en0的地址,因为他是iPhone的wifi的地址,是肯定存在的.(例外情况依然有:市面上依然存在一部分联通的阉割版无wifi的iPhone)

2.Mac地址涉及到隐私,不应该胡乱将用户的Mac地址传播!所以我们需要

将Mac地址进行hash之后,才能作为DeviceId上传.

关于第一个注意点的问题,经过我测试,没有Wifi功能的iPhone3GS一样可以获得Mac地址,所以这应该是目前标识设备唯一最好的一个解决方案.

解决方案github下载地址github。com/gekitz/UIDevice-with-UniqueIdentifier-for-iOS-5

背景:

大多数应用都会用到苹果设备的UDID号，UDID通常有以下两种用途:

1）用于一些统计与分析目的；【第三方统计工具如友盟，广告商如ADMOB等】

2）将UDID作为用户ID来唯一识别用户，省去用户名，密码等注册过程。

不过，2011年时，苹果就宣布ios5.0以后的系统中将不再支持以下方法获取用户的UDID【苹果设备的唯一识别码】：

[UIDevice currentDevice] uniqueIdentifier];

【注：对于已越狱了的设备,UDID并不是唯一的.使用Cydia插件UDIDFaker,可以为每一个应用分配不同的UDID】

同时，苹果公司建议使用UUID【一种开放的软件构建标准】来替代:

```
-(NSString*) uuid {
    CFUUIDRef puuid = CFUUIDCreate( nil );
    CFStringRef uuidString = CFUUIDCreateString( nil, puuid );
    NSString * result = (NSString *)CFStringCreateCopy( NULL, uuidString);
    CFRelease(puuid);
    CFRelease(uuidString); return [result autorelease];
}
```

该方法每次都会获取一个唯一的标识字符串，开发者可以在应用第一次启动时候调用一次，然后将该串存储起来，以便以后替代UDID来使用。

问题是如果用户删除该应用再次安装时，又会生成新的字符串，所以不能保证唯一识别该设备。

而最近（2012年3月），有消息称苹果应用商店开始拒绝使用UDID的应用上架。

替代方案：

现在网上有一现成的解决方案,使用设备的Mac地址,因为Mac地址也是唯一的.unix有系统调用可以获取Mac地址.但有些事情需要注意:

1.iPhone可能有多个Mac地址,wifi的地址,以及SIM卡的地址.一般来讲,我们取en0的地址,因为他是iPhone的wifi的地址,是肯定存在的.

2.Mac地址涉及到隐私,不应该胡乱将用户的Mac地址传播!所以我们需要将Mac地址进行hash之后,才能作为DeviceId上传.

该解决方案源码地址：https://github.com/gekitz/UIDevice-with-UniqueIdentifier-for-iOS-5

该方案提供了两个方法：

uniqueDeviceIdentifier (返回MAC和CFBundleIdentifier的MD5值)

uniqueGlobalDeviceIdentifier(返回MAC的MD5值)

使用方法：

#import "UIDevice+IdentifierAddition.h" NSLog(@"%@", [[UIDevice currentDevice] uniqueDeviceIdentifier]);

NSLog(@"%@",[[UIDevice currentDevice] uniqueGlobalDeviceIdentifier]);

测试结果：

WIFI下：

UDID：XXXX21f1f19edff198e2a2356bf4XXXX

新生成的：XXXX7dc3c577446a2bcbd77935bdXXXX

3G下：

UDID：XXXX21f1f19edff198e2a2356bf4XXXX

新生成的：XXXX7dc3c577446a2bcbd77935bdXXXX

GPRS下

UDID：XXXX21f1f19edff198e2a2356bf4XXXX

新生成的：XXXX7dc3c577446a2bcbd77935bdXXXX

飞行模式下：

UDID：XXXX21f1f19edff198e2a2356bf4XXXX

新生成的：XXXX7dc3c577446a2bcbd77935bdXXXX

删除应用重装后：

UDID：XXXX21f1f19edff198e2a2356bf4XXXX

新生成的：XXXX7dc3c577446a2bcbd77935bdXXXX