3.9.验证购买（防止第三方插件漏洞）iOS7新特性

提示：虽然苹果在iOS7提升了购买凭据的安全性，但是处于金钱考虑，购买完成后，一定要做凭据的验证工作。

复制代码

1 #pragma mark 验证购买凭据

2 - (void)verifyPruchase

3 {

4 // 验证凭据，获取到苹果返回的交易凭据

5 // appStoreReceiptURL iOS7.0增加的，购买交易完成后，会将凭据存放在该地址

6 NSURL \*receiptURL = [[NSBundle mainBundle] appStoreReceiptURL];

7 // 从沙盒中获取到购买凭据

8 NSData \*receiptData = [NSData dataWithContentsOfURL:receiptURL];

9

10 // 发送网络POST请求，对购买凭据进行验证

11 NSURL \*url = [NSURL URLWithString:ITMS\_SANDBOX\_VERIFY\_RECEIPT\_URL];

12 // 国内访问苹果服务器比较慢，timeoutInterval需要长一点

13 NSMutableURLRequest \*request = [NSMutableURLRequest requestWithURL:url cachePolicy:NSURLRequestUseProtocolCachePolicy timeoutInterval:10.0f];

14

15 request.HTTPMethod = @"POST";

16

17 // 在网络中传输数据，大多情况下是传输的字符串而不是二进制数据

18 // 传输的是BASE64编码的字符串

19 /\*\*

20 BASE64 常用的编码方案，通常用于数据传输，以及加密算法的基础算法，传输过程中能够保证数据传输的稳定性

21 BASE64是可以编码和解码的

22 \*/

23 NSString \*encodeStr = [receiptData base64EncodedStringWithOptions:NSDataBase64EncodingEndLineWithLineFeed];

24

25 NSString \*payload = [NSString stringWithFormat:@"{\"receipt-data\" : \"%@\"}", encodeStr];

26 NSData \*payloadData = [payload dataUsingEncoding:NSUTF8StringEncoding];

27

28 request.HTTPBody = payloadData;

29

30 // 提交验证请求，并获得官方的验证JSON结果

31 NSData \*result = [NSURLConnection sendSynchronousRequest:request returningResponse:nil error:nil];

32

33 // 官方验证结果为空

34 if (result == nil) {

35 NSLog(@"验证失败");

36 }

37

38 NSDictionary \*dict = [NSJSONSerialization JSONObjectWithData:result options:NSJSONReadingAllowFragments error:nil];

39

40 NSLog(@"%@", dict);

41

42 if (dict != nil) {

43 // 比对字典中以下信息基本上可以保证数据安全

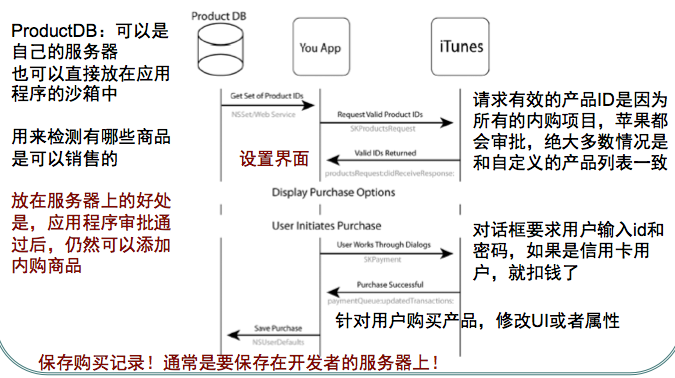
44 // bundle\_id&application\_version&product\_id&transaction\_id

45 NSLog(@"验证成功");

46 }

47 }

三、iOS客户端开发



这幅图很清晰的解释了内购的整个流程，主要包括以下步骤（希望大家认真看这幅图的细节，来理解内购的流程）

•请求有效的产品代号集合

•购买指定产品

•验证购买

•恢复购买

1⃣️请求有效的产品代号集合

// 1) 实例化产品请求

SKProductsRequest \*request = [[SKProductsRequest alloc]initWithProductIdentifiers:identifiers];

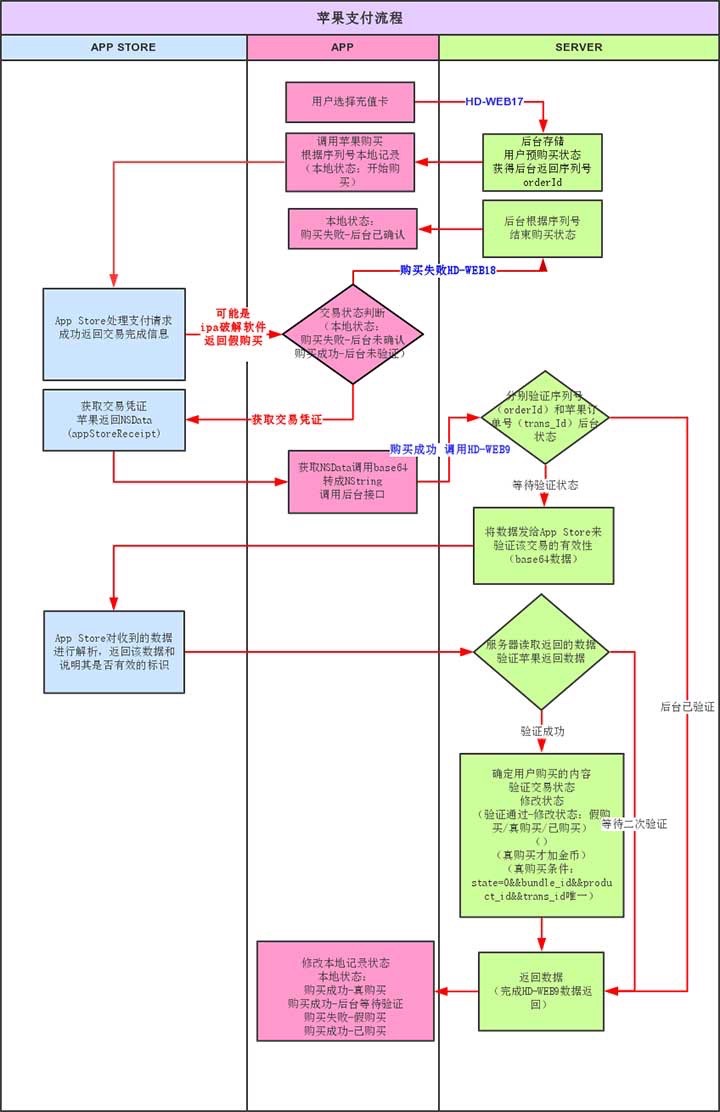
// 2) 设置代理

[request setDelegate:self];

// 3) 启动请求

[request start];

提示：

1. 实例化请求时，必须指定有效的identifiers集合，之所以如此处理，主要是为了确保提交的内购商品真的通过了苹果的审批，处于可用状态！

2. 要想获取到准确的可用产品集合，需要通过代理方法实现

- (void)productsRequest:(SKProductsRequest \*)request didReceiveResponse:(SKProductsResponse \*)response

3. 越狱用户无法测试内购，但是可以购买

2⃣️购买产品

1. 内购的交易过程是通过SKPaymentTransactionObserver监控的，因此需要为IAPHelper添加交易观察者：

// 添加交易观察者对象

[[SKPaymentQueue defaultQueue]addTransactionObserver:sharedInstance];

2. 由于发起交易需要使用SKProduct对象，因此需要使用字典记录所有可用的商品

NSMutableDictionary \*\_productsDict;

3.交易队列回调方法

- (void)paymentQueue:(SKPaymentQueue \*)queue updatedTransactions:(NSArray \*)transactions

{

for (SKPaymentTransaction \*transaction in transactions) {

// 购买完成

if (transaction.transactionState == SKPaymentTransactionStatePurchased) {

NSLog(@"购买完成 %@", transaction.payment.productIdentifier);

[queue finishTransaction:transaction];

} else if (transaction.transactionState == SKPaymentTransactionStateFailed) {

if (transaction.error.code != SKErrorPaymentCancelled) {

NSLog(@"交易失败： %@", transaction.error.localizedDescription);

}

}

}

}

3⃣️验证购买（通常交由服务器来做）

1 接收ios端发过来的购买凭证。

2 判断凭证是否已经存在或验证过，然后存储该凭证。

3 将该凭证发送到苹果的服务器验证，并将验证结果返回给客户端。

4 如果需要，修改用户相应的会员权限。

考虑到网络异常情况，服务器的验证应该是一个可恢复的队列，如果网络失败了，应该进行重试。

与苹果的验证接口文档在 [这里](https://developer.apple.com/library/ios/#documentation/NetworkingInternet/Conceptual/StoreKitGuide/VerifyingStoreReceipts/VerifyingStoreReceipts.html%23//apple_ref/doc/uid/TP40008267-CH104-SW3) 。简单来说就是将该购买凭证用Base64编码，然后POST给苹果的验证服务器，苹果将验证结果以JSON形式返回。

苹果AppStore线上的购买凭证验证地址是 ***<https://buy.itunes.apple.com/verifyReceipt>*** ，测试的验证地址是： ***<https://sandbox.itunes.apple.com/verifyReceipt>***

4⃣️恢复购买——恢复用户购买过的所有商品

[[SKPaymentQueue defaultQueue]restoreCompletedTransactions];