首先到网上随意找个组，熟悉概念，发现简单例子，但还是想全面了解，于是找Apple的例子，到【https://developer.apple.com/library/ios/samplecode/GenericKeychain/Introduction/Intro.html】下载，打开工程，看Readme文件，有这么一段描述：

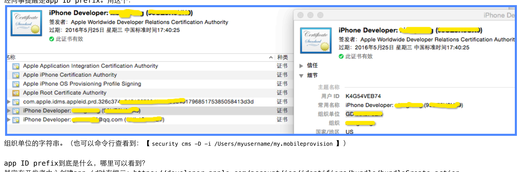
```
The sample must be edited to include your app ID prefix in five places where the text "YOUR_APP_ID_HERE"
appears before building and running the sample apps. The easiest way to do that is to use Xcode's Find in Project
command to locate the places that must be edited first. Just change the app ID prefix; leave the rest unchanged.
```

意思是要用appID的ID prefix去替换YOUR_APP_ID_HERE这个字符串，一开始以为是Apple ID,提示Provision file不匹配，看网上例子是一串字符，以为是这个：

```
iPhone Developer: xiang ding (9NJZSH9XD3)
iPhone Developer: yx113350@qq.com (QNLVG3K7D7)
```

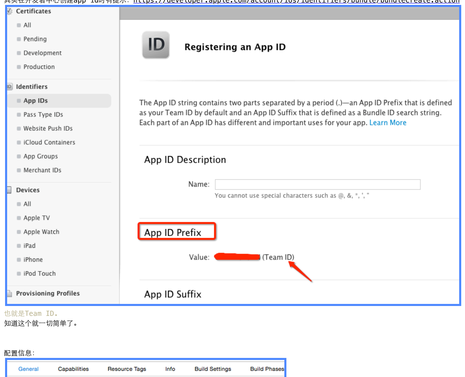
括号里这串，也提示不匹配。

经同事提醒是app ID prefix，用这个：



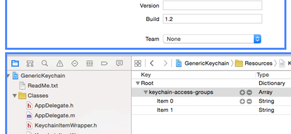
组织单位的字符串，（也可以命令行查看到：【`security cms -D -i /Users/myusername/my.mobileprovision`】）

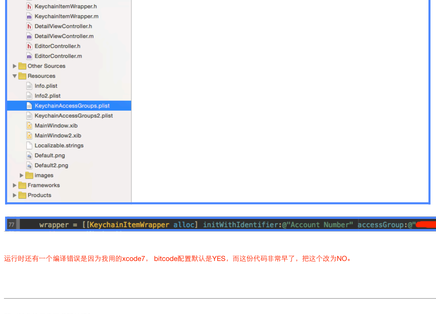
app ID prefix到底是什么，哪里可以看到？
其实在开发者中心创建app id时有提示：https://developer.apple.com/account/ios/identifiers/bundle/bundleCreate.action



也就是Team ID.
知道这个就一切简单了。

配置信息：






```
wrapper = [[KeychainItemWrapper alloc] initWithIdentifier:@"Account Number" accessGroup:@"[redacted].com.[redacted].GenericKeychainSuite"];
```

运行时还有一个编译错误是因为我用的xcode7，bitcode配置默认是YES，而这份代码非常早了，把这个改为NO。

那不付费的开发证书可以吗？
可以

代码理解：

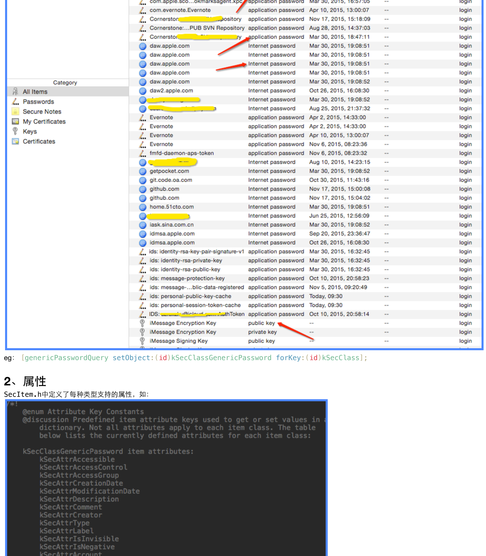
一、查询

1、类型：kSecClass –安全项的类别

系统定义了这几种：

```
extern const CFStringRef kSecClassGenericPassword
    __OSX_AVAILABLE_STARTING(__MAC_10_7, __IPHONE_2_0);
extern const CFStringRef kSecClassInternetPassword
    __OSX_AVAILABLE_STARTING(__MAC_10_6, __IPHONE_2_0);
extern const CFStringRef kSecClassCertificate
    __OSX_AVAILABLE_STARTING(__MAC_10_7, __IPHONE_2_0);
extern const CFStringRef kSecClassKey
    __OSX_AVAILABLE_STARTING(__MAC_10_7, __IPHONE_2_0);
extern const CFStringRef kSecClassIdentity
    __OSX_AVAILABLE_STARTING(__MAC_10_7, __IPHONE_2_0);
```

其实可以参考mac上的钥匙串来理解



eg: `[genericPasswordQuery setObject:(id)kSecClassGenericPassword forKey:(id)kSecClass];`

2、属性

SecItem.h中定义了每种类型支持的属性，如：

```
/*
    @enum Attribute Key Constants
    @discussion Predefined item attribute keys used to get or set values in a
        dictionary. Not all attributes apply to each item class. The table
        below lists the currently defined attributes for each item class:

    kSecClassGenericPassword item attributes:
        kSecAttrAccessible
        kSecAttrAccessControl
        kSecAttrAccessGroup
        kSecAttrCreationDate
        kSecAttrModificationDate
        kSecAttrDescription
        kSecAttrComment
        kSecAttrCreator
        kSecAttrType
        kSecAttrLabel
        kSecAttrIsInvisible
        kSecAttrIsNegative
        kSecAttrAccount
        kSecAttrService
        kSecAttrGeneric
        kSecAttrSynchronizable
```

eg: `[genericPasswordQuery setObject:(id)kSecAttrGeneric];`
如果用了Group，加上这个属性：`[genericPasswordQuery setObject:accessGroup forKey:(id)kSecAttrAccessGroup];`
accessGroup就是多个App中用来标识共享数据的唯一id，以你的带前缀的app id开头且要设定到工程设置的code signing entitlements文件中。

3、数据匹配策略：kSecMatchPolicy系统定义了很多匹配策略，可以根据需要选取

`[genericPasswordQuery setObject:(id)kSecMatchLimitOne forKey:(id)kSecMatchLimit];`

4、返回数据的类型：kSecReturnData

有这几种：

```
extern const CFStringRef kSecReturnData
    __OSX_AVAILABLE_STARTING(__MAC_10_6, __IPHONE_2_0);
extern const CFStringRef kSecReturnAttributes
    __OSX_AVAILABLE_STARTING(__MAC_10_6, __IPHONE_2_0);
extern const CFStringRef kSecReturnRef
    __OSX_AVAILABLE_STARTING(__MAC_10_6, __IPHONE_2_0);
extern const CFStringRef kSecReturnPersistentRef
    __OSX_AVAILABLE_STARTING(__MAC_10_6, __IPHONE_2_0);
eg: [genericPasswordQuery setObject:(id)kCFBooleanTrue forKey:(id)kSecReturnAttributes];
```

5、执行查询

根据设定的返回类型，用合适的变量去接收数据

```
NSDictionary *tempQuery = [NSDictionary dictionaryWithDictionary:genericPasswordQuery];
NSMutableDictionary *outDictionary = nil;
if (! SecItemCopyMatching((CFDictionaryRef)tempQuery, (CFTypeRef *)&outDictionary) == noErr)
```

6、从返回结果中取数据

设定要取的数据类型和类别

```
// Add the proper search key and class attribute.
[returnDictionary setObject:(id)kCFBooleanTrue forKey:(id)kSecReturnData];
[returnDictionary setObject:(id)kSecClassGenericPassword forKey:(id)kSecClass];

// Acquire the password data from the attributes.
NSData *passwordData = NULL;
if (SecItemCopyMatching((CFDictionaryRef)returnDictionary, (CFTypeRef *)&passwordData) == noErr)
```

二、增加和更新数据

用 SecItemCopyMatching检查是否存在，不存在就调用SecItemAdd添加，否则用 SecItemUpdate更新。
对于更新的第二个参数有这样的要求：Only real keychain attributes are permitted in this dictionary (no "meta" attributes are allowed.)
所以例子里把kSecClass移除了：`[tempCheck removeObjectForKey:(id)kSecClass];`

三、删除数据

调用SecItemDelete就行：

```
NSMutableDictionary *tempDictionary = [self dictionaryToSecItemFormat:keychainItemData];
junk = SecItemDelete((CFDictionaryRef)tempDictionary);
```