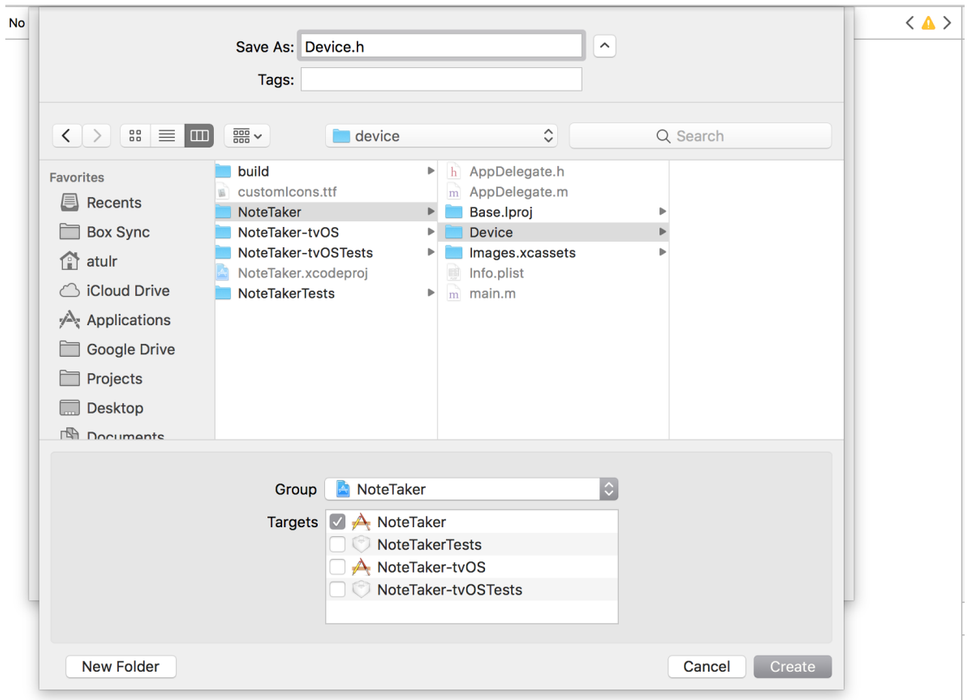
iOS custom native module

Let's build the same native module for iOS that we built for Android in the last chapter. The purpose remains the same, **the native module should get you the device name set on an iPhone.**

Before we continue, just a reminder that if you want to write code for iOS, please **open the iOS project on Xcode**. This is because Xcode is built for iOS development and it will help you resolve the trivial errors that otherwise would take up too much time.

Let's get started

1. Open the `./ios/` project folder. Open the .xcodeproj file or the .xcworkspace file in Xcode.
2. Create a header file `Device.h` by following the steps `File -> New -> File -> Header File` and then name the file `Device.h` and choose the targets. Create a new folder Device if you like to organize files in a folder like me.



3. Let modify our `Device.h` file.

> ios/Device/Device.h

```
#import <React/RCTBridgeModule.h>

@interface Device : NSObject <RCTBridgeModule>
@end
```

This is our main custom native modules header file.

4. Now let's create the corresponding implementation file `Device.m` in the same location.

> ios/Device/Device.m

```
#import "Device.h"

@implementation Device

RCT_EXPORT_MODULE();

@end
```

5. Similar to Android's getName method, here we have RCT_EXPORT_MODULE() macro. If no name is explicitly provided, it will take the name of the module. Here the name of the module is 'Device'. **In order to expose a method from native module to Javascript just write a method inside the RCT_EXPORT_METHOD macro**. These methods can be accessed from `NativeModules` of the `react-native` package.

   See the example below:

   > ios/Device/Device.m

```
#import "Device.h"
#import <UIKit/UIKit.h>

@implementation Device

//export the name of the native module as 'Device' since no explicit name is mentioned
RCT_EXPORT_MODULE();

//exports a method getDeviceName to javascript
RCT_EXPORT_METHOD(getDeviceName:(RCTResponseSenderBlock)callback){
 @try{
   NSString *deviceName = [[UIDevice currentDevice] name];
   callback(@[[NSNull null], deviceName]);
 }
 @catch(NSException *exception){
   callback(@[exception.reason, [NSNull null]]);
 }
}

@end
```

   Here we are exporting a method `getDeviceName()` from native to Javascript. This method can be accessed in JS via

```
import {NativeModules} from 'react-native';
NativeModules.Device.getDeviceName((err ,name) => {
   console.log(err, name);
});
```

   NativeModules has a key named 'Device'. This is basically the same name exported by RCT_EXPORT_METHOD.

   We passed a callback to get the value from the NativeModule.

That's it, let's give it a shot!

In a Javascript file, you can access the module methods using `NativeModules.<moduleName>.<methodName>`

> app/index.js

Just add

```
import {NativeModules} from 'react-native';
...
...
  NativeModules.Device.getDeviceName((err, name) => console.log(err, name));
...
...
```

Running this on an iOS simulator returns.