一、导入AUTELSDK

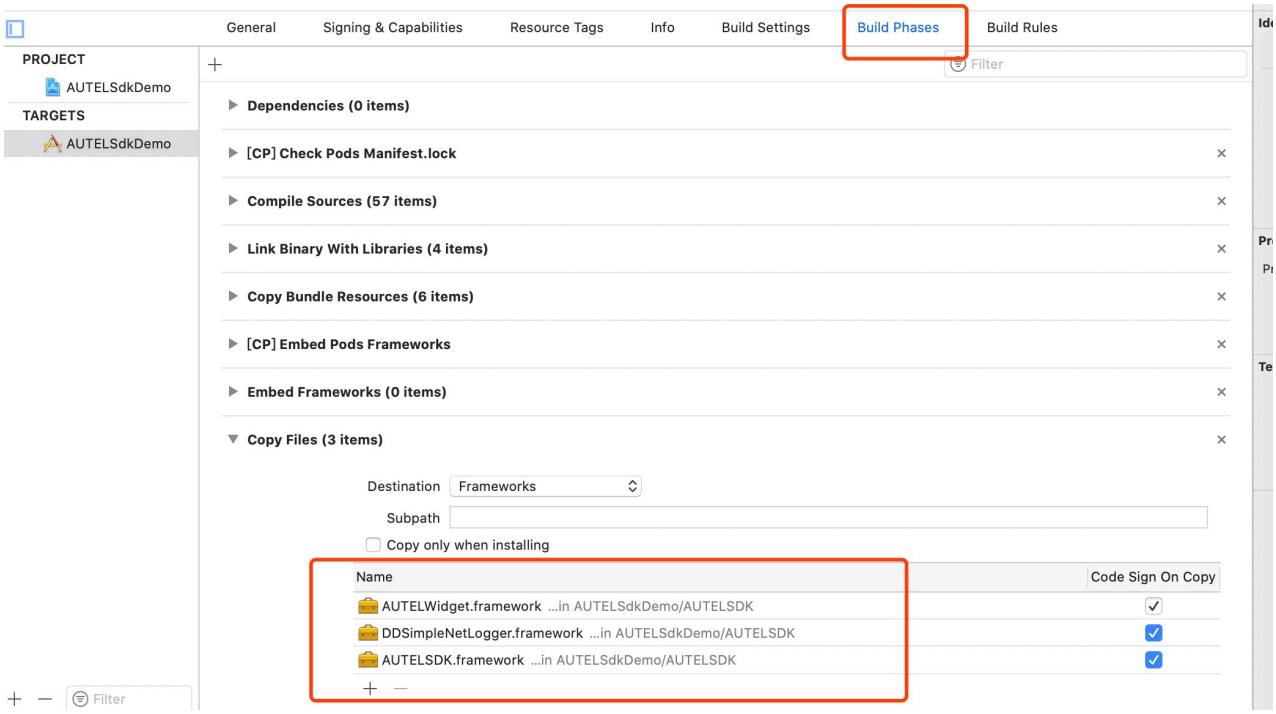
AUTELSDK 支持iOS 11以上的系统。

AUTELSdkDemo 使用XCode 12.4版本编译，支持Swift 5.3。

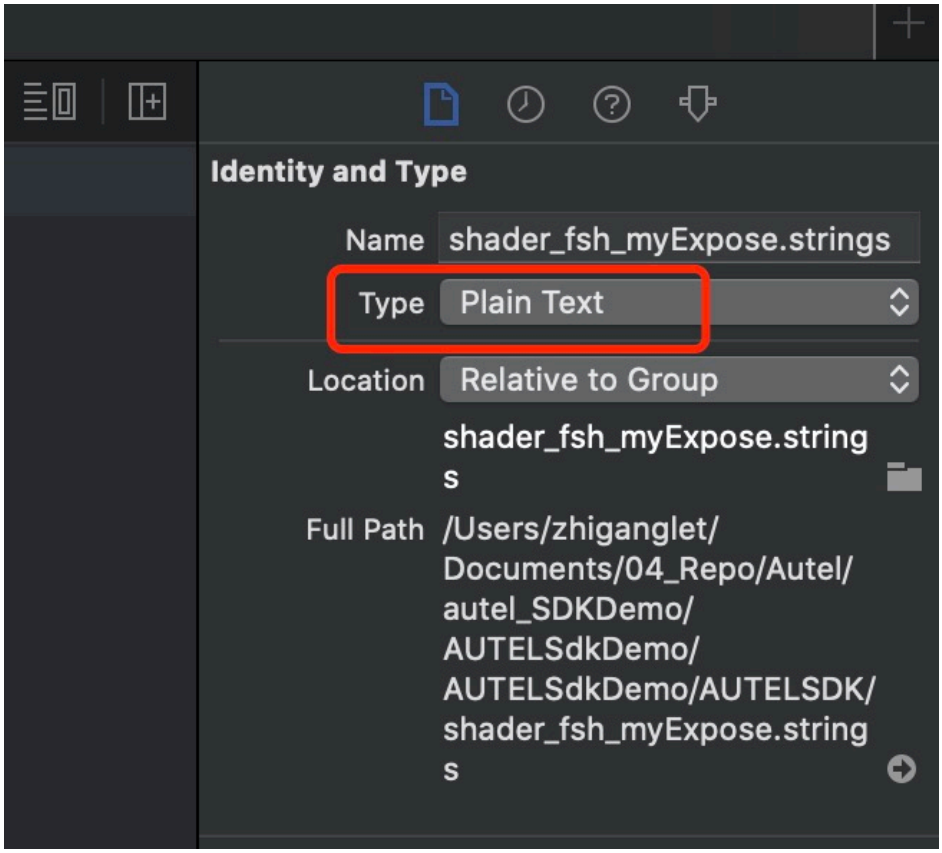
1. AUTELSDK包括AUTELSDK.xcframework、AUTELWidget.xcframework、DDSimpleNetLogger.xcframework三个xcframework和shader_fsh_myExpose.strings、shader_vsh_myExpose.strings两个strings文件。

参考AUTELSdkDemo工程，把AUTELSDK文件夹直接拖入到你的工程；

在AUTELSdkDemo工程的build setting界面：target-〉Build Phases -〉左上角+号 -〉New Copy Files Phase 然后在Copy Files下 Destination选择Frameworks -〉添加动态库



2. shader_fsh_myExpose.strings和shader_vsh_myExpose.strings两个strings文件在Identity and Type下的Type要选择**“Plain Text”**类型，如下图，否则编译不通过。



二、配置Info.plist

Info.plist需要加入如下配置才能正常使用：

```
<key>CFBundleDisplayName</key>
<string>AUTELSdkDemo</string>
<key>UISupportedExternalAccessoryProtocols</key>
<array>
  <string>com.autelrobotics.control</string>
  <string>com.auteltech.control</string>
</array>
<key>NSAppTransportSecurity</key>
<dict>
  <key>NSAllowsArbitraryLoads</key>
  <true/>
</dict>
<key>NSLocationAlwaysAndWhenInUseUsageDescription</key>
<string>App needs your consent to access location</string>
<key>NSLocationAlwaysUsageDescription</key>
<string>App needs your consent to access location</string>
<key>NSLocationWhenInUseUsageDescription</key>
<string>App needs your consent to access location</string>
```

1、CFBundleDisplayName

表示APP名称，必须配置，而且与你申请APP key时填写的APP名称保持一致，否则SDK鉴权不通过；

2、UISupportedExternalAccessoryProtocols

外部附件框架，只有配置了这项，App才可以与道通航空的遥控器建立USB连接；

3、NSAppTransportSecurity

申请http权限

4、NSLocationAlwaysAndWhenInUseUsageDescription等

申请定位权限

三、启动SDK

参考**AUTELSdkDemo**演示代码，AUTELSdkDemo的DataLayer模块代码用于启动SDK和建立飞机各个模块连接，并且监听SDK各个模块上报数据，可直接用于你的工程。

1. 在APP程序入口方法didFinishLaunchingWithOptions调用SDKConnection.shared.connect()方法，如下

```
func application(_ application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {

      SDKConnection.shared.connect()

      return true
  }
```

2. 在SDKConnection.shared.connect()方法里进行APP注册，注册需要传入appKey和代理，SDK会对appKey进行鉴权，鉴权结果通过代理进行回调，鉴权通过会启动SDK。具体请参考demo代码。

```
func connect() -> Void {
      //appKey
      AUTELAppManager.registerApp("your appKey", with: self)
  }
```

NOTE: appKey 为开发者在[Autel开发者平台]申请的应用关联钥匙

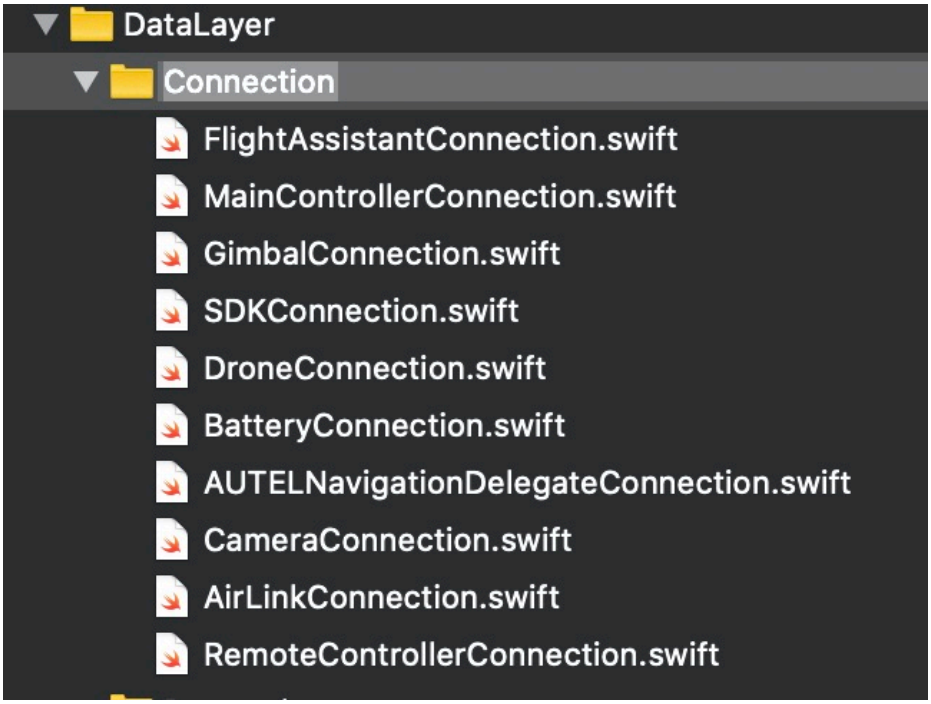
3. 注册成功，数据线分别插入手机和遥控器两端USB口，打开APP，建立连接触发didConnectionStatusChanged回调，把Connection连接层与SDK底层的各个模块建立代理关系

```
// MARK: - AUTELDeviceDelegate
```

```
    func device(_ device: AUTELDevice!, didConnectionStatusChanged status:
AUTELDeviceConnectionStatus) {
        guard let drone = device as? AUTELDrone else {
            self.connectedDrone = nil
            return
        }
        CameraConnection.shared.connect(drone.camera)
        GimbalConnection.shared.connect(drone.gimbal)
        BatteryConnection.shared.connect(drone.battery)
        MainControllerConnection.shared.connect(drone.mainController)
        RemoteControllerConnection.shared.connect(drone.remoteController)
        AirLinkConnection.shared.connect(drone.airLink)

AUTELNavigationDelegateConnection.shared.connect(drone.mainController.navig
ationManager)

FlightAssistantConnection.shared.connect(drone.mainController.flightAssista
nt)

        //执行回调
        queue.async {
            for (_, aProtocol) in self.protocols {
                aProtocol.device(device, didConnectionStatusChanged:
status)
            }
        }

    }
```

Connection连接层实现SDK底层各个模块的代理方法，对各个模块进行状态更新监听：

- FlightAssistantConnection：视觉连接层
- MainControllerConnection：飞行控制器连接层
- GimbalConnection：云台连接层
- BatteryConnection：电池连接层
- AUTELNavigationDelegateConnection：任务连接层
- CameraConnection：相机连接层
- AirLinkConnection：图传连接层
- RemoteControllerConnection：遥控器连接层



应用层想要监听某个模块数据，可以注册监听对应Connection连接层，实现相应的代理方法即可，具体用法见AUTELSdkDemo里的AircraftStatus Demo：

```
fileprivate func registerProtocol() {
    DroneConnection.shared.addProtocol(self)
    MainControllerConnection.shared.addProtocol(self)
    RemoteControllerConnection.shared.addProtocol(self)
    BatteryConnection.shared.addProtocol(self)
    CameraConnection.shared.addProtocol(self)
    GimbalConnection.shared.addProtocol(self)
    FlightAssistantConnection.shared.addProtocol(self)

}
```

四、SDK 功能接口调用

通过各个Connection连接层桥接到SDK对应的模块管理类，进而调用各个模块的接口：

AUTELDrone

通过该类可以对 EVO 1 和EVO 2飞行器进行操作

应用层获取句柄：DroneConnection.shared.drone

AUTELDroneMainController

通过 AUTELDroneMainController 类，你可以获得飞行器主控的当前状态，可以读写主控参数，也向主控发送命令

应用层获取句柄：MainControllerConnection.shared.mainController

AUTELBattery

该类管理电池相关信息，并可获取飞行器电池的实时状态.

应用层获取句柄：MainControllerConnection.shared.mainController

AUTELRemoteController

通过该类可以读写遥控器参数，校准遥控器，还可以实时获得遥控器状态，各摇杆、按钮和波轮的状态。

应用层获取句柄：RemoteControllerConnection.shared.remoteController

AUTELDroneGimbal

通过该类，你可以获得云台的实时状态，也可以向云台发送命令，控制云台

应用层获取句柄：GimbalConnection.shared.gimbal

AUTELBaseCamera

通过 AUTELBaseCamera 类，可以获取到相机的当前状态，可以对相机参数进行读写操作，也可以对相机进行相关操作。

应用层获取句柄：CameraConnection.shared.camera

AUTELAirLink

通过 AUTELAirLink 类，可以获取到通信链路的连接状态

应用层获取句柄：AirLinkConnection.shared.airlink

AUTELFlightAssistant

通过 AUTELFlightAssistant 类，可以获得视觉的实时状态，也可以向视觉发送命令，控制视觉

应用层获取句柄：MainControllerConnection.shared.mainController?.flightLimitation

AUTELNavigation

通过 navigationManager 你可以让飞行器执行自动飞行任务

应用层获取句柄：MainControllerConnection.shared.mainCtl?.navigationManager

五、相机模块使用举例

目前可用相机类型为XT701、XT705、XT706，APP与飞机建立连接成功后，与此同时也会与相机建立连接，连接后通过CameraConnection的代理方法实时上报相机状态及变化，可以根据需要来监听相机各个状态

```
protocol CameraProtocol: CheckProtocol {
```

```
    func camera(_ camera: AUTELBaseCamera!, didUpdateConnectState connectState:
AUTELCameraConnectState) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateRecordRecoverState
recoverState: AUTELCameraRecordRecoverState) -> ()

    func camera(_ camera: AUTELBaseCamera!, didGenerateNewMediaFile newMedia:
AUTELMedia!) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateSDCardState sdCardState:
AUTELCameraSDCardState!) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateMMCState mmcState:
AUTELCameraSDCardState!) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateSystemState systemState:
AUTELCameraSystemBaseState!) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateParameters parameters:
AUTELCameraParameters!, change: [AnyHashable : Array<Any>]!) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateCurrentExposureValues
exposureParameters: AUTELCameraExposureParameters!) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateHistogramTotalPixels
totalPixels: Int, andPixelsPerLevel pixelsArray: [Any]!) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateTimeLapseCaptured
captureCount: Int, andCountDown timeInterval: TimeInterval) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateAutoFocusState focusState:
AUTELCameraAFLensFocusState, andLensFocusAreaRowIndex rowIndex: Int,
andColIndex colIndex: Int) -> ()

    func camera(_ camera: AUTELBaseCamera!, didUpdateAeAfState state: Int) ->
()
}
```

AUTELCameraParameters：可获取相机当前支持的参数范围，示例如下，获取当前相机支持的照片分辨率范围，具体参考demo实现

```
class func requestPhotoSizeWithItem() -> [String] {
        guard let numbers =
CameraConnection.shared.camera?.parameters.supportedCameraPhotoAspectRatioRange
() else { return [] }
        var items: [String] = []
        for num in numbers.reversed() {
            let parameter = AUTELCameraPhotoAspectRatio(rawValue:
UInt8(truncating: num))?.displayName() ?? ""
            items.append(parameter)
        }
        return items
    }
```

应用层调用SDK相机命令，具体使用方法见demo里面的CameraCommandDeliver.swift文件，如设置拍照图片尺寸

```
/// 设置拍照图片尺寸
///
/// - Parameters:
///   - size: 支持的图片尺寸
///   - result: 回调
class func setPhotoSize(_ size: AUTELCameraPhotoAspectRatio, with result:
ResultHandler) {
    CameraConnection.shared.camera?.setPhotoRatio(size, withCompletion: {
result?($0) })
}
```

当前支持的查询参数有：AUTELCameraWorkMode、AUTELCameraVideoResolution、AUTELCameraVideoFileFormat、AUTELCameraPhotoAspectRatio、AUTELCameraExposureMode等

六、AUTELSdkDemo

更多细节请参考AUTELSdkDemo，AUTELSdkDemo实现了如下功能：

- 图传显示
- 地图显示，包括飞机home点和飞机点实时更新
- 飞行状态显示
- 相机模式读写
- 相机分辨率读写
- 任务飞行的航点飞行demo