IOS 定位 SDK 集成文档

创建 IOS 项目工程

定位 SDK （wzLib.framework)

将解压后的 wzLib.framework 文件 **copy** 或 **拖拽** 到工程文件夹中，并在弹框中 choose options for adding these files 选择 add to targets 中选择您的工程

需要申请的权限配置

定位权限

在项目的 Info.plist 添加定位权限申请，根据您的业务需求，选择下列方式设置。

```
<key>UIBackgroundModes</key>
<array>
<string>location</string>
</array>
<key>NSLocationWhenInUseUsageDescription</key>
<string>是否允许使用您的定位？</string>
<key>NSLocationAlwaysAndWhenInUseUsageDescription</key>
<string>是否允许后台一直使用定位？</string>
```

接口调用

第 1 步，引入文件

```
import wzLib
```

第 2 步，初始化

```
var wzClientOption: WzClientLocation?
var location: WzLocation?

init() {
    wzClientOption = WzClientLocation()
    wzClientOption?.callDelegate = self
}
```

第 3 步，开始持续定位

```
wzClientOption?.isLocateOnce = false //设置是否单次定位
wzClientOption?.interval = 5.0 //单位秒
wzClientOption?.startLocation()
```

第 4 步，位置更新

```
func onReceivedLocation(wzLocation: WzLocation) {
    print("onReceivedLocation: ")
    let locateCoordinate =
 HTMCoorTransform.transformFromWGSToGCJ(CLLocationCoordinate2D(latitude:
CLLocationDegrees(wzLocation.latitude!),longitude:CLLocationDegrees(wzLocation.l
ongitude!))) // 坐标系转换，默认是 wgs84 坐标系
self.mapViewState.center = locateCoordinate
}

func onLocationError() {
    //定位失败是坐标返回
}
```

整体代码

```
import SwiftUI
import MapKit
import wzLib
struct ContentView: View,LocationDelegate{
    var mapViewDelegate: MapViewDelegate?
    var wzClientOption: WzClientLocation?

    init() {
        wzClientOption = WzClientLocation()
        wzClientOption?.callDelegate = self
    }

    struct CustomButtonStyle: ButtonStyle {
        func makeBody(configuration: Configuration) -> some View {
            configuration.label
            .padding()
            .background(.blue)
            .cornerRadius(10)
            .foregroundColor(.white)
            .scaleEffect(configuration.isPressed ? 2 : 1)
            .animation(.easeOut(duration: 0.2), value: configuration.isPressed)
        }
    }

    var body: some View {
        HStack{
            // 按钮控件
            Button("单次定位"){
                wzClientOption?.isLocateOnce = true
                wzClientOption?.startLocation()
            }.buttonStyle(CustomButtonStyle())
            Button("连续定位"){
                wzClientOption?.isLocateOnce = false
                wzClientOption?.interval = 5.0
                wzClientOption?.startLocation()
            }.buttonStyle(CustomButtonStyle())
            Button("停止定位"){
                wzClientOption?.stopLocation()
            }.buttonStyle(CustomButtonStyle())
        }.buttonStyle(.bordered)
    }
```

```
    func onReceivedLocation(wzLocation: WzLocation) {
        print("onReceivedLocation: ")
        let locateCoordinate =
HTMCoorTransform.transformFromWGSToGCJ(CLLocationCoordinate2D(latitude:
CLLocationDegrees(wzLocation.latitude!),longitude:CLLocationDegrees(wzLocation.l
ongitude!)))
        self.mapViewState.center = locateCoordinate
    }

    func onLocationError() {

    }
}
struct ContentView_Previews: PreviewProvider {
    static var previews: some View {
        ContentView()
    }
}
```