title: 验证-ios-部署文档
type: 验证-ios-部署文档
order: 0

概述及资源

行为验证 iOS SDK 提供给集成 iOS 原生客户端开发的开发者使用, SDK 不依赖任何第三方库。

环境需求

条目	资源
开发目标	兼容 iOS 9+
开发环境	Xcode 13.0+
系统依赖	`Webkit.framework`
SDK 第三方依赖	无

相关开发资源

条目	资源
产品结构流程	通讯流程, 交互流程
SDK接口文档	gt4-api-ref-ios 或查看头文件注释
错误码列表	Error Code 列表

安装

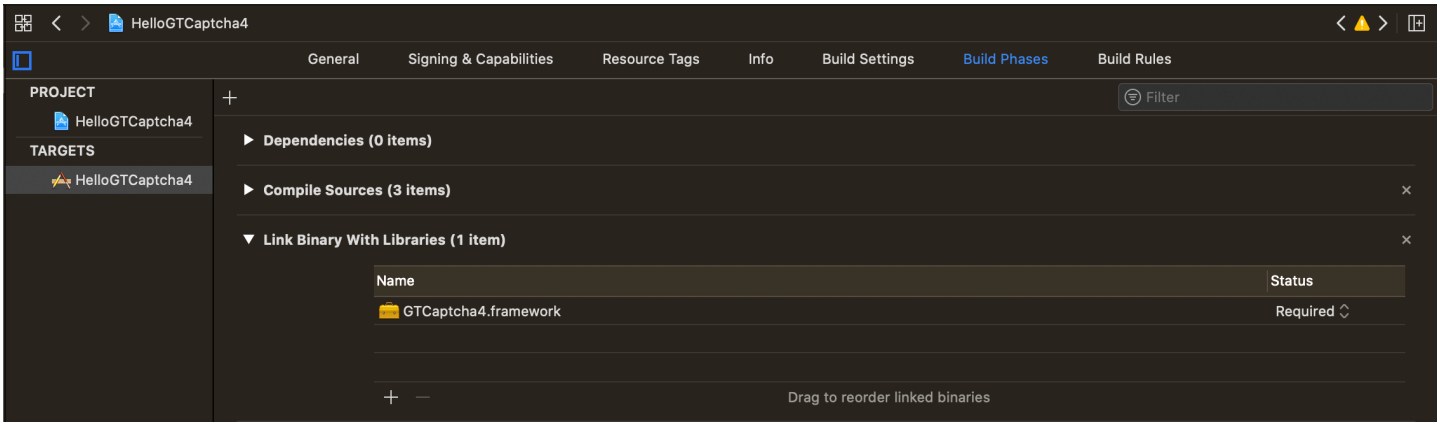
获取SDK

请进入后台下载

导入SDK

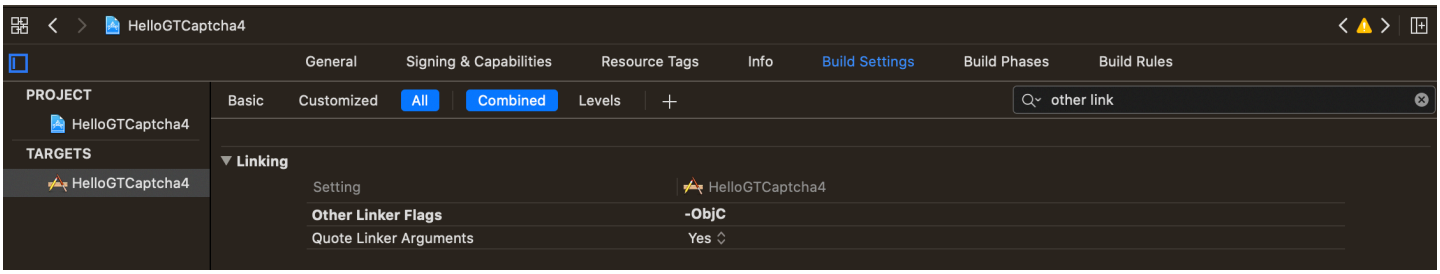
1. 如果您是手动添加 SDK , 将下载获取的`GTCaptcha4.framework`文件拖拽到工程中, 确保`Copy items if needed`已被勾选。

   请使用`Linked Frameworks and Libraries`方式导入framework。在拖入`GTCaptcha4.framework`到工程后, 还要检查`.framework`文件是否被添加到`PROJECT -> Build Phases -> Linked Frameworks and Libraries`。

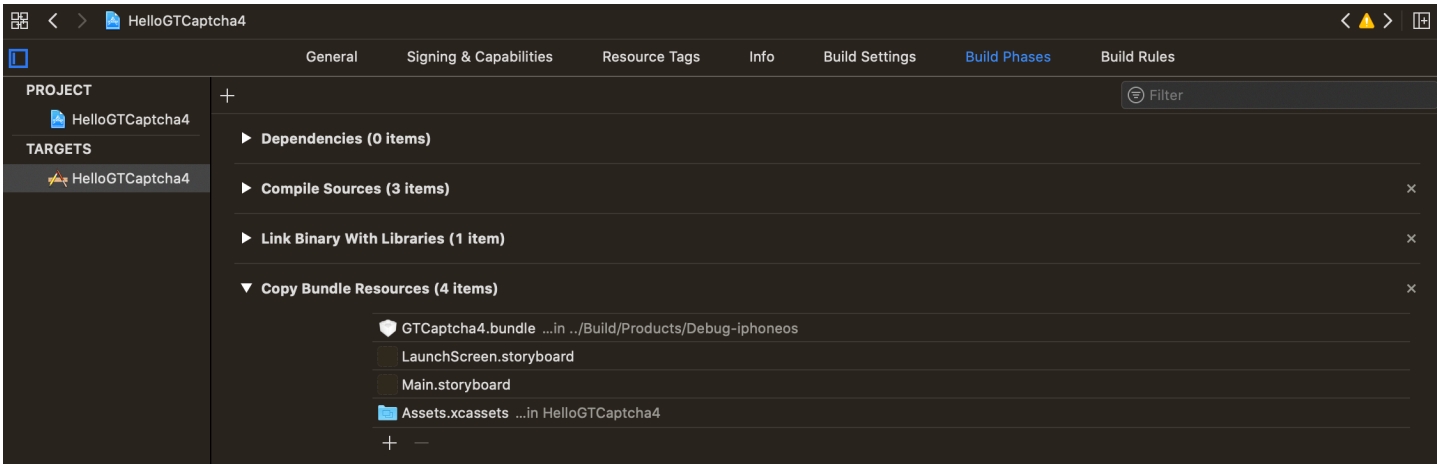


   > SDK 已提供 XCFramework 格式，位于下载文件的`SDK`->`XCFramework`目录下的`GTCaptcha4.xcframework`

2. 针对静态库中的`Category`, 需要在对应target的`Build Settings`->`Other Linker Flags`添加`-ObjC`。



3. 需要在工程中同时引入`GT4Captcha4.Bundle`， 否则无法使用验证。把 SDK 路径下的中的`GTCaptcha4.Bundle`拖入项目中。



配置接口

参考行为验证的 产品结构流程, 必须要先在您的后端搭建相应的 **服务端接口**（请参考服务端部署文档），并配置从极验管理后台 获取的 `captchaId` 和 `Key`

集成用户需要使用 iOS SDK 完成提供的以下接口:

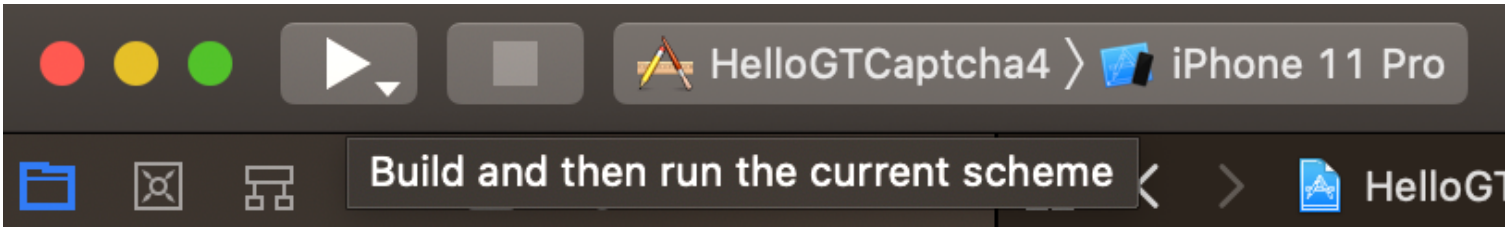
1. 使用 ID 配置验证初始化
2. 启动验证
3. 获取验证结果参数，并对提交结果参数进行二次校验，避免伪造
4. 使用错误代理方法处理验证过程中可能遇到的问题

需要遵守 `<GTCaptcha4SessionTaskDelegate>` 协议以处理验证结果和可能返回的错误。

> 集成代码参考下方的**代码示例**

编译并运行你的工程

编译你的工程, 体验行为验证 4.0!



代码示例

初始化与调用验证

在工程中导入验证动态库`GTCaptcha4.framework`的头文件

```
#import <GTCaptcha4/GTCaptcha4.h>
```

> 以在`UIButton`中集成为参考

1. 初始化

   初始化验证管理器`GTCaptcha4Session`的实例, 在`UIButton`初始化方法中对`GTCaptcha4Session`实例调用注册方法以获得注册数据:

```
#define captchaID @"xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"

@interface ViewController () <GTCaptcha4SessionTaskDelegate>

@property (strong, nonatomic) IBOutlet UIButton *startBtn;

@property (nonatomic, strong) GTCaptcha4Session *captchaSession;

@end

@implementation ViewController

- (GTCaptcha4Session *)captchaSession {
    if (!_captchaSession) {
        _captchaSession = [GTCaptcha4Session
sessionWithCaptchaID:captchaID];
        /// 如需修改默认配置
        /// 可选择下面注释的方式创建实例
        // GTCaptcha4SessionConfiguration *config =
[GTCaptcha4SessionConfiguration defaultConfiguration];
        // config.timeout = 8.0f;
        // ...
        // _captchaSession = [GTCaptcha4Session
sessionWithCaptchaID:captchaID configuration:config];

        _captchaSession.delegate = self;
    }

    return _captchaSession;
}
```

```
- (void)viewDidLoad {
    [super viewDidLoad];

    // 提前初始化验证会话，如果不在此位置调用，将使用懒加载模式
    [self captchaSession];

    [self.startBtn addTarget:self action:@selector(start)
forControlEvents:UIControlEventTouchUpInside];
}
```

> 其他可选配置项见 `GTCaptcha4SessionConfiguration` 中定义的接口或属性。

2. 调用验证

   初始化完成后, 调用下面的方法进行验证:

```
- (void)start {
    [self.captchaSession verify];
}
```

处理验证结果

只有完成对验证结果的校验后, 本次验证才是完整完成。您需要在遵守 `GTCaptcha4SessionTaskDelegate` 协议后, 处理以下代理方法:

```
- (void)gtCaptchaSession:(GTCaptcha4Session *)captchaSession didReceive:
(NSString *)code result:(NSDictionary *)result message:(NSString *)message {
    NSLog(@"result: %@", result);
    // code 为 @"1" 则为用户完成了验证，为 @"0" 则为用户验证失败
    if ([@"1" isEqualToString:code]) {
        if (result && result.count > 0) {
            // 字典转为表单格式进行提交(根据实际接口数据格式要求进行构造)
            __block NSMutableArray<NSString *> *kvPairs = [NSMutableArray
array];

            [result enumerateKeysAndObjectsUsingBlock:^(id  _Nonnull key, id
_Nonnull obj, BOOL * _Nonnull stop) {
                if ([key isKindOfClass:[NSString class]] &&
                    [obj isKindOfClass:[NSString class]]) {
                    NSString *kvPair = [NSString stringWithFormat:@"%@=%@", key,
obj];
```

```
                    [kvPairs addObject:kvPair];
                }
            }];

            NSString *formStr = [kvPairs componentsJoinedByString:@"&"];
            NSData *data = [formStr dataUsingEncoding:NSUTF8StringEncoding];

            // 业务后端提供的验证校验接口
            NSURL *url = [NSURL
URLWithString:@"http://xxx.yyy.zzz/path/validate"];
            NSMutableURLRequest *request = [NSMutableURLRequest
requestWithURL:url];
            request.HTTPMethod  = @"POST";
            request.HTTPBody    = data;

            // 提交到后端对结果进行校验
            [[[NSURLSession sharedSession] dataTaskWithRequest:request
completionHandler:^(NSData * _Nullable data, NSURLResponse * _Nullable response,
NSError * _Nullable error) {
                if (!error && data) {
                    // 处理验证结果
                    NSString *msg = [[NSString alloc] initWithData:data
encoding:NSUTF8StringEncoding];
                    NSLog(@"result: %@", msg);
                }
                else {
                    NSLog(@"error: %@", error);
                }
            }] resume];
        }
    }
}
```

处理验证错误

验证过程中可能发生一些预料之外的错误, 您可以通过遵守 `GTCaptcha4SessionTaskDelegate` 协议后, 在下面的代理方法中进行处理:

```
- (void)gtCaptchaSession:(GTCaptcha4Session *)captchaSession didReceiveError:
(GTC4Error *)error {
    // 向用户展示错误信息和错误码

    // 在日志中记录错误详情
```

```
    NSLog(@"error: %@", error.description);
}
```

强烈建议在向用户展示验证错误原因时，同时展示错误码，方便后续排查线上问题。

可能遇到的错误码请参考后面的列表: GTC4Error

Swift 示例

更多示例代码细节参考官方提供的 Demo，其中 Swift 示例代码请参考 Demo 源码中的 `DefaultDemoViewController.swift` 文件

s