title: iOS
type: iOS
order: 0

iOS

Overview and resources

The iOS SDK is available to developers who integrate iOS native client development and does not rely on any third-party libraries.

Environmental requirements

Item	Resources
Development object	Compatible with iOS 9+
Developing environment	Xcode 13.0+
System dependent	Webkit.framework
Third-party dependency of SDK	None

Related development resources

Item	Resources
Product structure process	Communication process
SDK interface documentation	gt4-api-ref-ios or view header file comments
Error code list	Error Code List

Installation

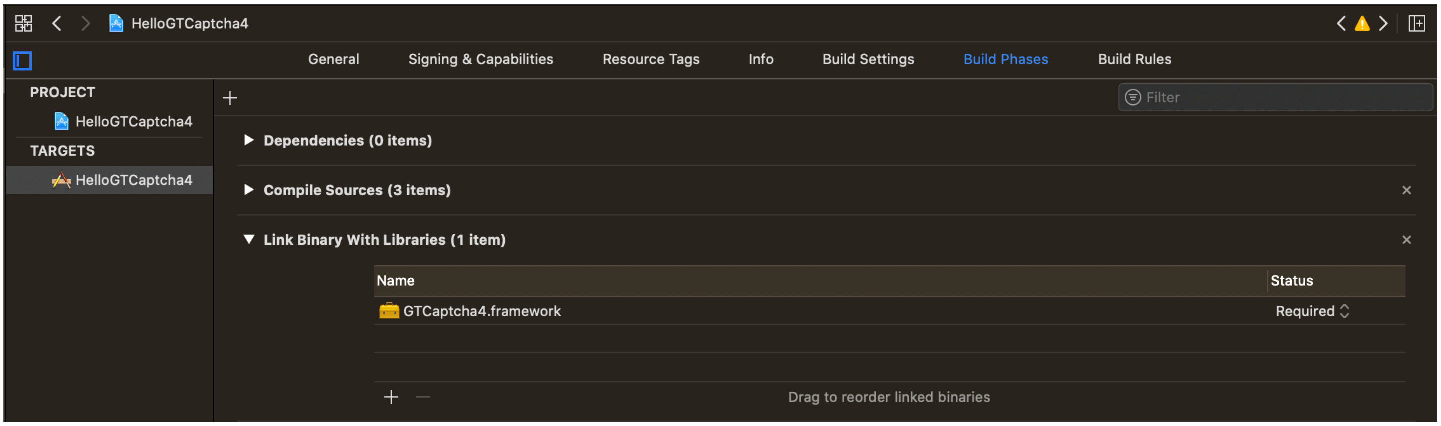
Get SDK

Go to the dashboard to download the current SDK.

Import SDK

1. If you add SDK manually, drag the downloaded `GTCaptcha4.framework` file into the project and make sure that "Copy items" is checked in case it is needed.

   Please import the framework using `Linked Frameworks and Libraries` method. After dragging `GTCaptcha4.framework` into the project, check whether .framework has been added to `PROJECT -> Build Phases -> Linked Frameworks and Libraries`.



Dependencies (0 items)

Compile Sources (3 items)

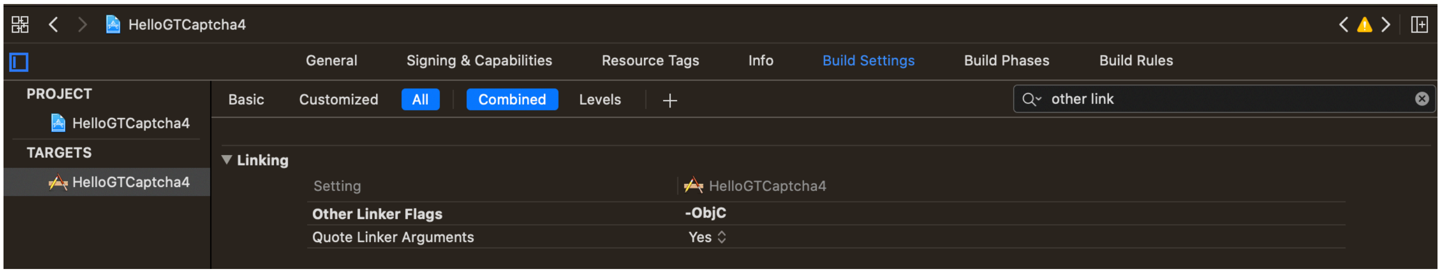
Link Binary With Libraries (1 item)

Name	Status
GTCaptcha4.framework	Required

Drag to reorder linked binaries

> SDK has provided XCFramework format, `GTCaptcha4.xcframework` located in the `SDK` -> `XCFramework` directory of the downloaded file.

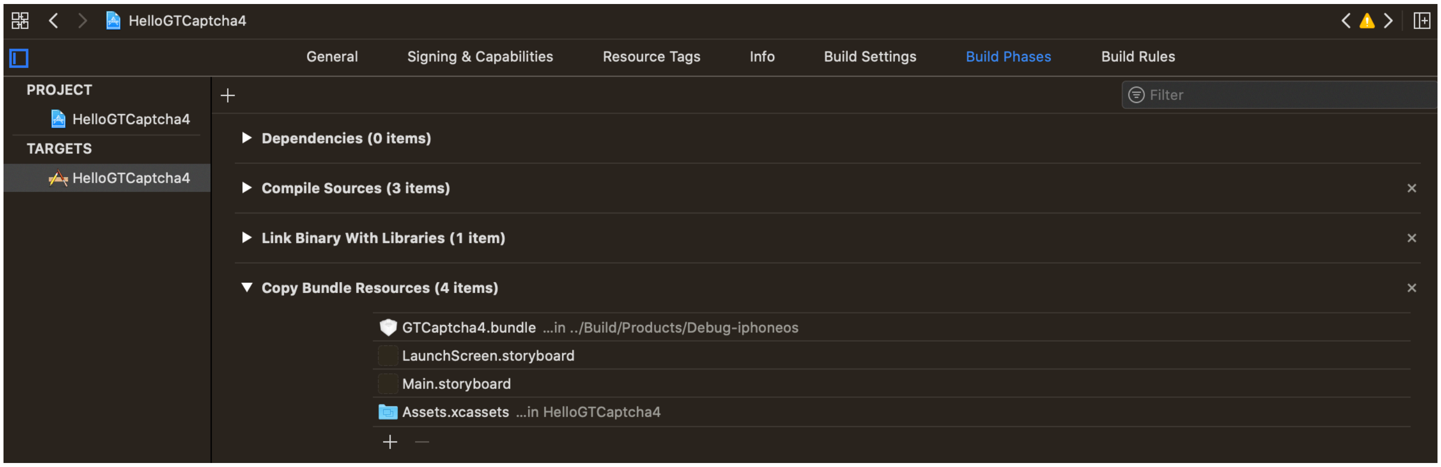
2. As for `Category` in the static library, `-ObjC` needs to be added to `Build Settings->Other Linker Flags` of the corresponding target.



Linking

Setting	HelloGTCaptcha4
Other Linker Flags	-ObjC
Quote Linker Arguments	Yes

3. `GT4Captcha4.Bundle` needs to be included in the project at the same time, otherwise, verification will fail. Drag the `GTCaptcha4.Bundle` to the SDK path in the project.



- Dependencies (0 items)
- Compile Sources (3 items)
- Link Binary With Libraries (1 item)
- Copy Bundle Resources (4 items)
  - GTCaptcha4.bundle ...in ../Build/Products/Debug-iphoneos
  - LaunchScreen.storyboard
  - Main.storyboard
  - Assets.xcassets ...in HelloGTCaptcha4

Configure

As shown in the product workflow process, you need to set up Server interface in your backend first, and configure it with `captchaId` and `Key` you got from GeeTest dashboard.

You need to use iOS SDK to complete the following iconfiguration:

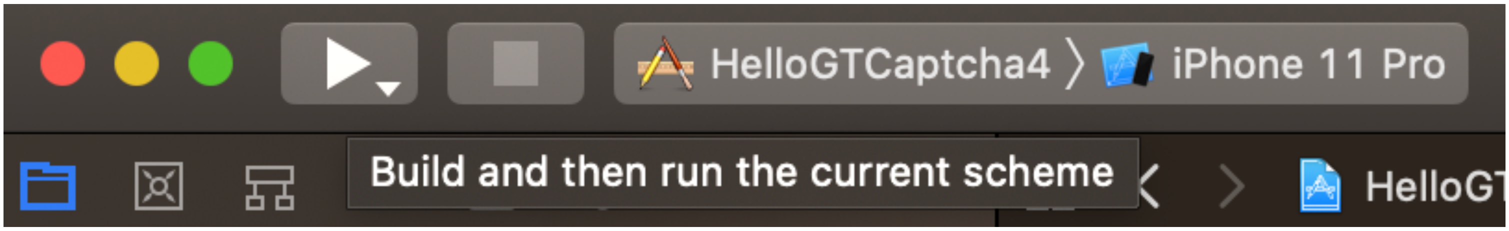
1. Use ID to set up initialization of CAPTCHA
2. Start verification
3. Get verification parameters and carry out secondary verification on the submitted result to avoid forgery
4. Use the error proxy method to deal with problems that may be encountered during verification

Follow the `GTCaptcha4SessionTaskDelegate` protocol to manage verification results and errors that may be returned.

Refer to the code samples below for integration code.

Compile and run your project

Compile your project and experience GeeTest Adaptive CAPTCHA.



Build and then run the current scheme

Sample code

Initialization and call CAPTCHA

Import the header file for verifying the dynamic `library GTCaptcha4.framework` in the project

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

Integration with UIButton

1. Initialization

An instance of initialization validation manager GTCaptcha4Session, which calls the registration method of `GTCaptcha4Session` sample in the `UIButton` initialization method to obtain registration data:

```
#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];

    /// If the default configuration needs to be modified
    /// Create an instance by choosing a form annotated below

    // GTCaptcha4SessionConfiguration *config = [GTCaptcha4SessionConfiguration
    // defaultConfiguration];

    // config.timeout = 8.0f;
    // ...

    // _captchaSession = [GTCaptcha4Session sessionWithCaptchaID:captchaID
    // configuration:config];

    _captchaSession.delegate = self;
  }
  return _captchaSession;
}

- (void)viewDidLoad {
  [super viewDidLoad];

  // Initialize captcha session in advance, as if it is not called in this
  // location, lazy load mode will be used

  [self captchaSession];

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

For other optional configuration items, see the interface or property defined in `GTCaptcha4Session`.

2. Call and start CAPTCHA session

Call the following method for verification after initialization is completed:

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

Process verification results

This verification is complete only after the verification results are verified.
You need to handle the following proxy methods after complying with the `GTCaptcha4SessionTaskDelegate` protocol:

```
- (void)gtCaptchaSession:(GTCaptcha4Session *)captchaSession didReceive:
(NSString *)code result:(NSDictionary *)result message:(NSString *)message {

    NSLog(@"result: %@", result);

     // When the code is @"1", it means a successful user authentication; and
when it is @"0", it means a failed user authentication

     if ([@"1" isEqualToString:code]) {
        if (result && result.count > 0) {

// Submit result data to validate

            __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];



             // Verification interface provided by service backend

            NSURL *url = [NSURL
URLWithString:@"http://xxx.yyy.zzz/path/validate"];

            NSMutableURLRequest *request = [NSMutableURLRequest
requestWithURL:url];

            request.HTTPMethod  = @"POST";

            request.HTTPBody    = data;
```

```
             // Submit to the back end to verify the results

            [[[NSURLSession sharedSession] dataTaskWithRequest:request
completionHandler:^(NSData * _Nullable data, NSURLResponse * _Nullable response,
NSError * _Nullable error) {

                if (!error && data) {

                 // Process validation results

                    NSString *msg = [[NSString alloc] initWithData:data
encoding:NSUTF8StringEncoding];

                    NSLog(@"result: %@", msg);

                }

                else {

                    NSLog(@"error: %@", error);

                }

            }] resume];

        }

    }

}
```

Handle verification errors

Some unexpected errors may occur, which you can handle in the following proxy methods by complying with the `GTCaptcha4SessionTaskDelegate` protocol:

```
- (void)gtCaptchaSession:(GTCaptcha4Session *)captchaSession didReceiveError:
(GTC4Error *)error {
    // Present to end user with error message and code here

    // Log the detailed error description
    NSLog(@"error: %@", error.description);
}
```

It is strongly recommended to present both an error message and error code to the end user. This will facilitate subsequent troubleshooting.

Please refer to the following list for possible error codes: GTC4Error

Swift example

For more example details, please refer to the official Demo, and please refer to the `DefaultDemoViewController.swift` file in the Demo for Swift example code.