Glympse® Client API Lite

Step by Step

Integration Guide for iOS

Overview

The Glympse API Lite provides a way of integrating Glympse functionality into any native iOS application. It includes facilities for sending and modifying Glympses.

This guide walks you through the list of all steps required to integrate Glympse API Lite into your iOS application.

1. Project Setup

1.1. Link your project to GlympseKitLite.embeddedframework

Drag the embeddedframework folder to your Xcode project's "Frameworks" folder.

The embeddedframework is located at **GLYMPSE_SDK/lib**.

Once the embeddedframework folder is correctly added to your project...

- **GlympseKitLite.framework** should appear under *Build Phases - Link Binary With Libraries*
- **GlympseResourcesLite.bundle** should appear under *Build Phases - Copy Bundle Resources*

1.2. Add the following other frameworks:

AddressBook.framework

CFNetwork.framework

CoreGraphics.framework

CoreLocation.framework

EventKit.framework

libz.dylib

MapKit.framework

MessageUI.framework

QuartzCore.framework

Security.framework

UIKit.framework

2. Background Location

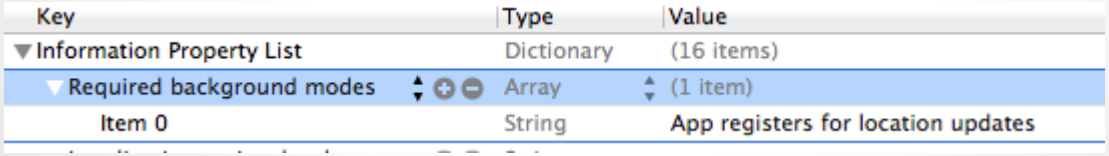
Control-click your application **plist** file, select *Open As > Source Code*

The following section should be added to application **plist** file as a child element of *<dict>*:

```
<key>UIBackgroundModes</key>
<array>
   <string>location</string>
</array>
```

Control-click your application **plist** file, select *Open As > Property List.*

Your **plist** should now appear as follows:



Key	Type	Value
▼ Information Property List	Dictionary	(16 items)
▼ Required background modes	Array	(1 item)
Item 0	String	App registers for location updates

4. Platform Startup

```
@interface GlympseWrapper : NSObject {
@private
    Glympse::GGlympseLite _glympse;
}
+ (GlympseWrapper*)instance; //singleton accessor
- (Glympse::GGlympseLite)glympse;
@end
@implementation GlympseWrapper
- (void)start {
    if ( _glympse == NULL ) {
        _glympse = Glympse::LiteFactory:: createGlympse(BASE_URL, API_KEY);
        _glympse-> start();
    }
}

- (void)stop {
    if ( _glympse != NULL ) {
        _glympse-> stop();
        _glympse = NULL;
    }
}

- (Glympse::GGlympseLite) glympse { return _glympse; }

+ (GlympseWrapper*)instance { ... } //implement your singleton accessor here
@end
```

```
- (BOOL)application:(UIApplication *)application  didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    [GlympseWrapper instance].glympse->start();
}
```

6. Foreground / Background

It is required to notify Glympse platform when application switches between foreground and background states.

```
- (void)applicationWillEnterForeground:(UIApplication *)application
{
    [GlympseWrapper instance].glympse->setActive(true);
}

- (void)applicationDidEnterBackground:(UIApplication *)application
{
    [GlympseWrapper instance].glympse->setActive(false);
}
```

7. Update User Profile

Nickname and avatar can be set in the following way:

```
Glympse::GGlympseLite glympse = [GlympseWrapper instance].glympse;

// Update nickname.
glympse->setNickname(Glympse::CoreFactory::createString(“Sylvia”));

// Update avatar.
UIImage* image = [UIImage imageNamed:@"<< image name >>"];
Glympse::GDrawable avatar = Glympse::CoreFactory:createDrawable(image);
glympse->setAvatar(avatar, 0);
```

NOTE: It is only required to set nickname and avatar once.
There is no need to set them again each time when platform starts.
They should only be re-set when the change is initiated by the user.

8. Send a Glympse

Prepare and send a Glympse.

```
// Create ticket object.
Glympse::GTicketLite ticketLite = Glympse::LiteFactory::createTicket(3600000,
  Glympse::CoreFactory::createString("Hello, world!"), NULL);

// Add invites. You can add as many invites as you need.
ticketLite->addInvite(Glympse::LC::INVITE_TYPE_EMAIL,
  Glympse::CoreFactory::createString(“Sylvia”),
  Glympse::CoreFactory::createString(“Sylvia@glympse.com”));


// Send the ticket.
Glympse::GGlympse glympse = [GlympseWrapper instance].glympse;
glympse->sendTicket(ticket, 0);
```

Next Steps

Use the following resources to get more information on Glympse API Lite:

- Glympse Samples

  Provides three sample projects that demonstrate basics of creating a glympse, sending and modifying a glympse, and retrieving glympse history. Also provides a starting template for a GlympseWrapper singleton.
- Reference Documentation

  Provides complete information on all programming interfaces exposed by the API.

Please contact partners@glympse.com if you need assistance, find a problem, or would like to request a feature.