GiGA Genie Mobile Plugin 2.0

Developer Guide

for iOS

융합기술원 Service연구소

PEOPLE. TECHNOLOGY. kt

목차

1. 개요

1.1. 문서 연혁

버전	변경일	변경사항	작성자
V 0.1.0	2019.11.25	- 최초	KT

1.2. 목적

본 문서는 ㈜케이티(이하 KT)의 GiGA Genie AI 서비스를 모바일 앱에서 이용할 수 있도록 제공하는 GiGA Genie Mobile Plugin(이하, G-Plugin) 라이브러리를 3rd Party Application 에 연동하는 방법에 대해 기술한다.

1.3. 적용범위

- 3rd Party Application 에서 AI 비서 라이브러리를 연동하는 방법을 설명한다.
- 지원 버전은 2.3 항목에 작성하였다.
- 내용은 KT 서비스 제공 계획의 변경 및 추가에 따라 추후 보완 및 변경 될 수 있다.

1.4. 약어 정리

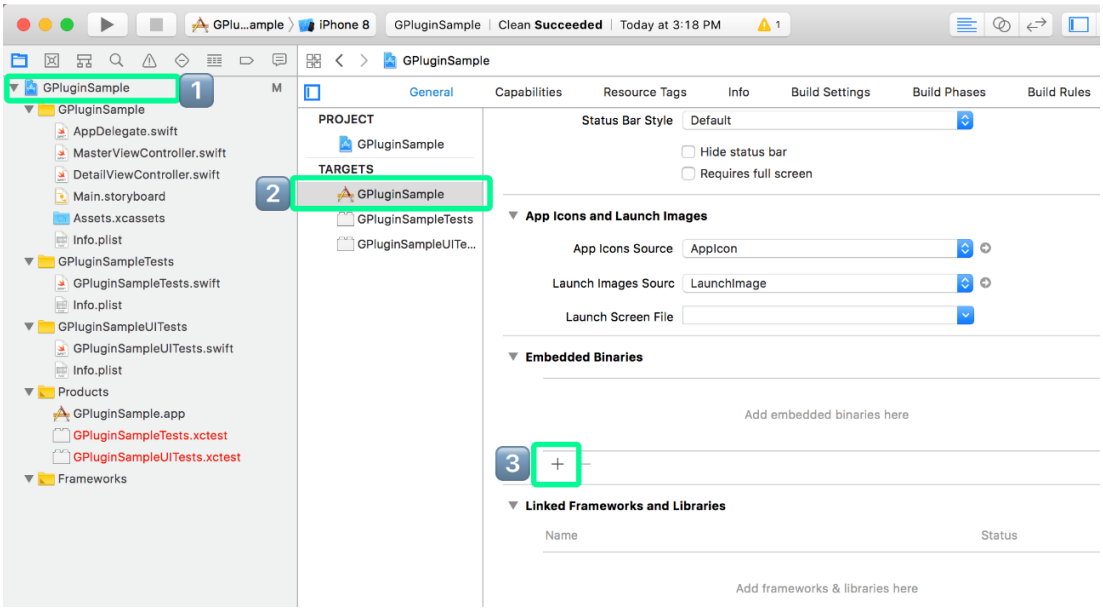
- G-Plugin – GiGA Genie Mobile Plugin 의 약어
- KWS – Keyword Spotting 의 약어
- DSS – 대화서버의 약어

2. 일반 사항

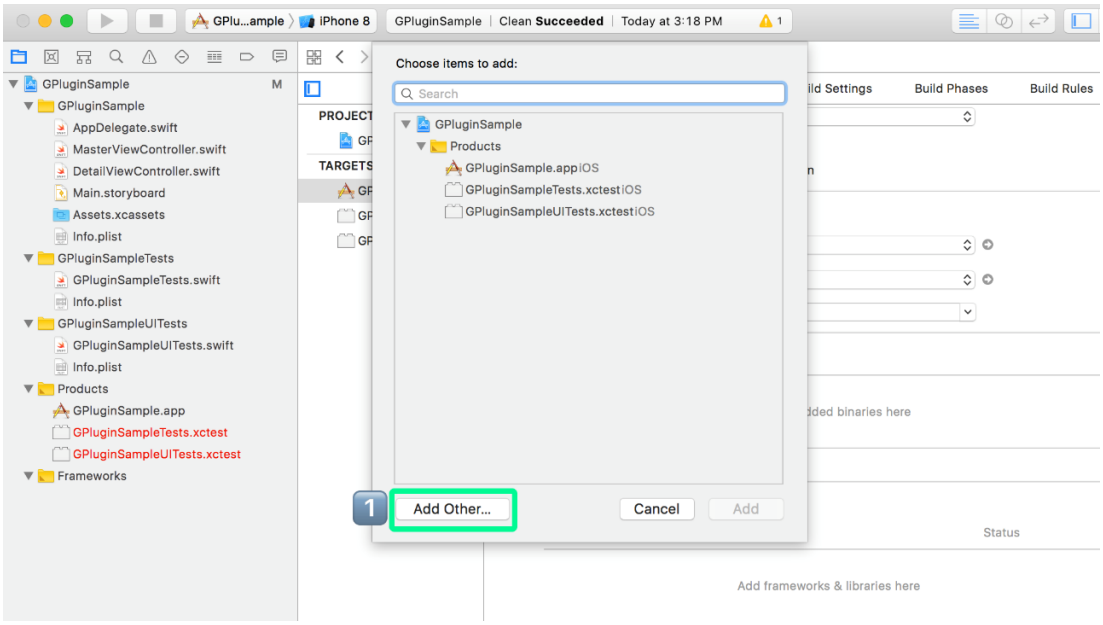
3rd Party Application 이 G-Plugin 과 연동하기 위한 API 사용 방법을 설명한다.

2.1. AI 비서 라이브러리 프로젝트 적용 방법

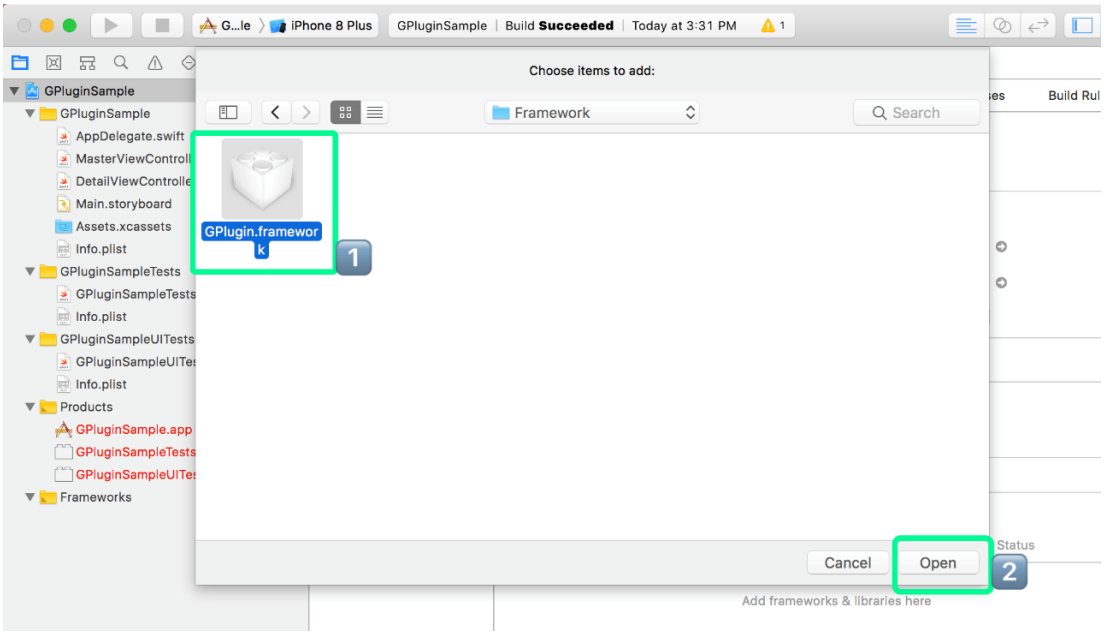
1) Groups & Files(좌측메뉴) > (1)Project 선택 > (2)Target 선택 > (3)Embedded Binaries (화면 중간) 에서 '+' 버튼을 클릭한다.



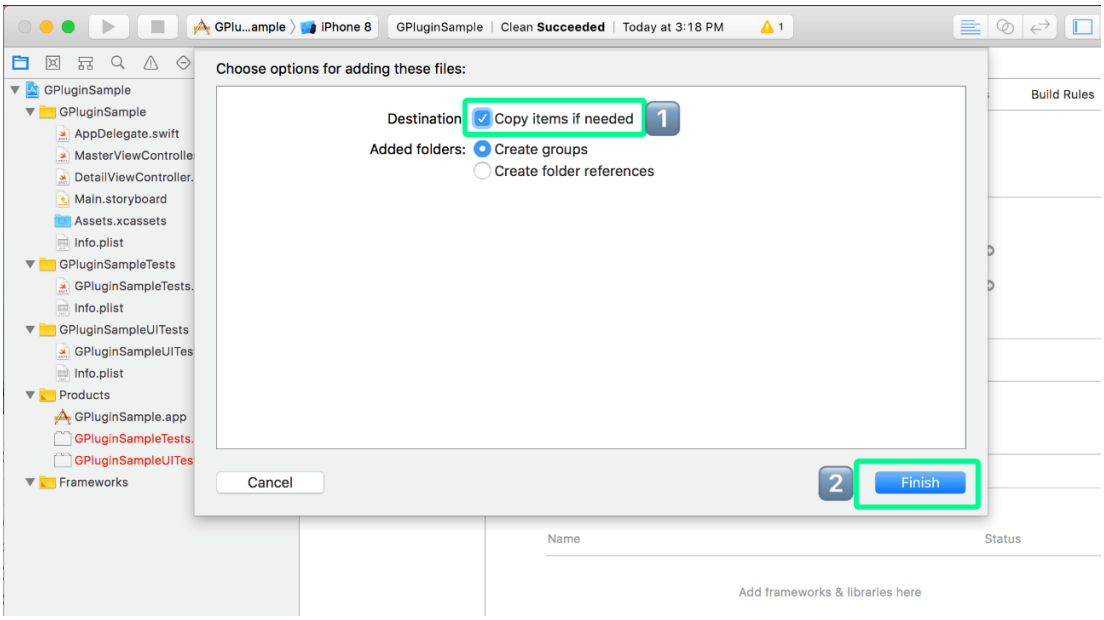
2) 다음의 화면에서 'Add Other ...' 버튼을 클릭한다.



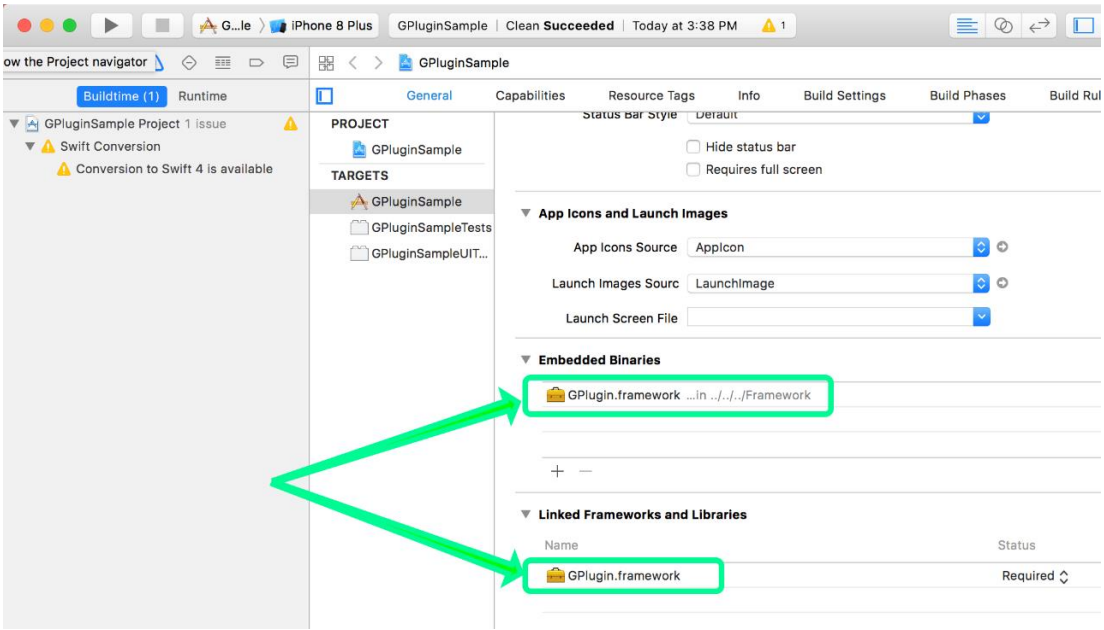
3) 전달받은 GPlugin-iOS 폴더에서 'GPlugin.framework'를 선택한 후 'Open' 버튼을 클릭한다.



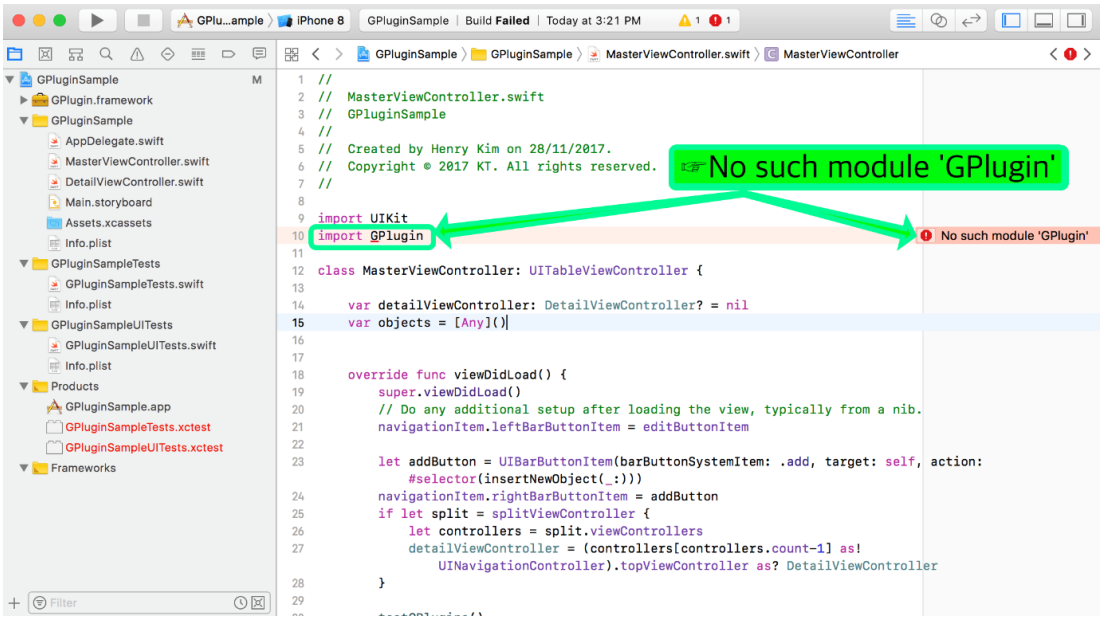
4) 'Copy items if needed'(1)를 선택한 후 'Finish'(2)를 클릭한다.



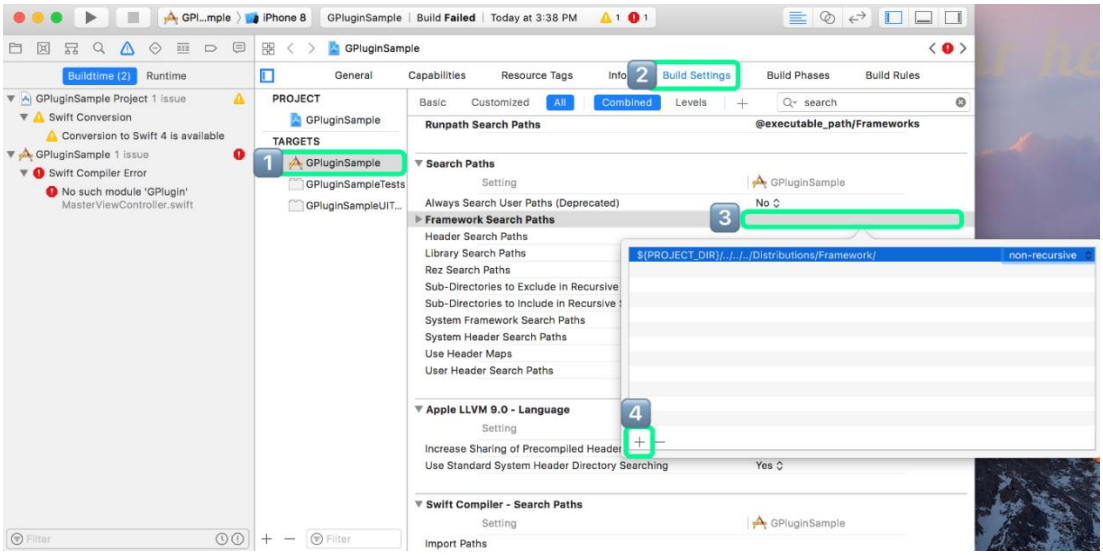
5) 다음 그림과 같이 'GPlugin.framework' 항목이 표시되었다면 성공적으로 Framework 가 import 된 것이다.



6) 소스코드 상에서 GPlugin 을 import 한 후 다음과 같은 오류가 발생했을 경우의 조치 방법을 안내하겠다.



7) 다음과 같이 Target (1) > Build Settings (2) > Framework Search Paths (3) 의 우측 공란을 더블 클릭하여 '+'버튼(4)을 눌러서 생성된 항목에 '${PROJECT_DIR}'과 같이 입력한 후 탭을 누른다. ${PROJECT_DIR}은 Xcode 로 생성한 프로젝트의 최상위 폴더이다. 만약 다른 위치에 'GPlugin.framework' 파일을 복사했다면 path 를 확인 후 기입해준다.

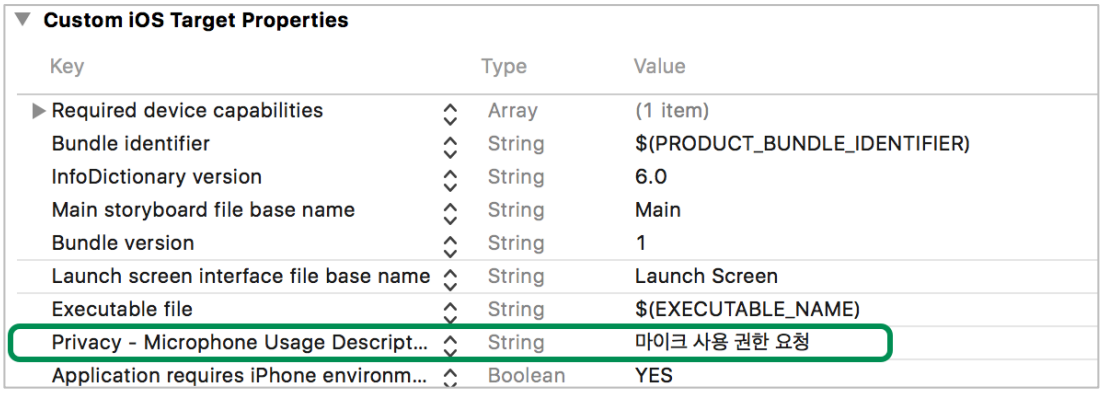


8) 마이크 및 음성인식 사용 권한 추가

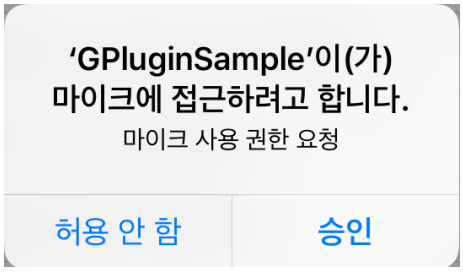
- **Sample 파일에 다음과 같이 AVFoundation Class Import 필요**

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

- **음성 인식을 위하여 '마이크 사용 권한'을 획득하여야 한다. G-Plugin 라이브러리를 사용하는 프로젝트의 Info.plist 에 아래와 같이 'Micorophone Usage Description' 를 추가해야 한다.**



- **다음과 같이 iOS 시스템 alert 이 사용자에게 표시된다.**



2.2. 주의 사항 (TBD)

2.3. 지원 버전

G-Plugin 라이브러리는 아래 버전으로 빌드되었으며, 3rd Party Application 에서도 아래 범위 내의 버전 정보로 설정해야 한다

- 본 문서는 Objective-C 언어 기준으로 작성 되어있다.
- Swift 는 4 이상, Xcode 10.0 이상, iOS 9.0 이상을 지원한다.

3. API description

3.1. API 목록

* Objective-C 기준, Swift 는 사용예제 참조

	APIs	설명
void	**init**:(**id**)delegate preferSingleton:(**BOOL**)usingNoti clientId:(**NSString***)clientId clientKey:(**NSString***)clientKey clientSecret:(**NSString***)clientSecrt userId:(**NSString***)userId complete:(**id**)complete	G-Plugin 라이브러리 생성 - G-Plugin 라이브러리를 사용하기 전 반드시 호출되어야 한다. - ClientId, ClientKey, ClientSecret 은 기가지니 포탈에서 발급 - userId 는 3rd party 가 생성하여 전달
GPlugin	**shared** * Notification Key: GPLUGIN_NOTIFICATION_RESULT	Singleton mode 를 선호하는 경우 사용한다. Delegate 와 함께 Notification 통해 결과를 전달받을 수 있다. GPLUGIN_NOTIFICATION_RESULT 를 addObserver 로 선언하면 noti 받을 수 있다.
void	**startGPlugin**	G-Plugin 서비스 시작 - 음성인식 + AI 서비스 이용 가능
void	**stopGPlugin**	G-Plugin 서비스 종료
void	**setGPluginListener**:(**id** <**GPluginListener**>)listener GPluginListener <NSObject> -(**void**)gPlugin:(GPlugin*)gPlug **onError**:(**NSError** *)error; -(**void**)gPlugin:(GPlugin*)gPlug **onEvent**:(**NSDictionary** *)event; -(**void**)gPlugin:(GPlugin*)gPlug **onCommand**:(**NSDictionary** *)event;	G-Plugin 으로부터 음성명령 결과나 이벤트 등을 수신하기 위한 delegate 등록 - onEvent: 음성인식 시작/중단 이벤트를 수신하기 위한 Callback - onCommand: 음성 또는 텍스트 인식 해석 결과(json object)를 수신하기 위한 Callback 등록
void	**updateLocInfoV2:**(**NSString** *)longi **latitude**:(**NSString** *)latti **address**:(**NSString** *)Addr;	현재 위치정보 및 도착지(목적지) 및 경유지 주소를 업데이트

		- startGPlugin() 호출 후 최초 한번 위치정보를 업데이트 하고 그 이후 변경 시마다 또는 주기적로 업데이트 해야 한다. (위치 정보 서비스 확장을 위하여 사용, 현재는 날씨 정보에 사용 중) (예시) addr: "서울 서초구 우면동" 사용하지 않는 인자: null
String	**getGPluginVersion**	G-Plugin 라이브러리 버전정보 조회
int	**getKwsId**	현재 GPlugin 음성인식 호출어 조회 - 0(기가지니), 1(지니야)
void	**setKwsId**:(**int**)kwsId	GPlugin 음성인식 호출어 변경 - 0(기가지니), 1(지니야)
void	**enableKws**:(**BOOL**)isOn	GPlugin 음성인식 호출어 사용 여부 설정 - true : 호출어 인식 활성화 - false : 호출어 인식 비활성화
Boolean	**checkKwsAvailable**	GPlugin 음성인식 호출어 사용여부 조회 - true : 호출어 인식 사용 - false : 호출어 인식 미사용
void	**startKws**	GPlugin 호출어 인식 시작
void	**stopKws**	GPlugin 호출어 인식 종료
void	**voiceCommand**	음성 명령으로 Service 요청
void	**textCommand**:(**NSString** *)msg type:(**NSString** *)type	텍스트 명령으로 Service 요청
void	**setMediaStatus**:(**int**)channel status:(**NSString** *)status playtime:(**int**)playtime	단말에서 미디어를 제어(재생시작, 일시정지, 종료)하는 경우 해당 동작을 서버로 노티
void	**serviceLogin**: (**NSString** *)appId type:(**NSString** *)type	서비스 로그인 페이지 요청 (예: 지니뮤직 등)
void	**cancelVoiceCmd**	G-Plugin 음성인식 종료

void	**getTTS**:(**NSString** *)msg	G-Plugin TTS Stream 요청
void	**setServerInfo:**(**NSString** *)serverIP grpcPort:(**NSString** *)grpcPort restPort:(**NSString** *)restPort	G-Plugin 연동 서버 정보 변경

3.2. G-Plugin Common API

3.2.1. G-Plugin 라이브러리 생성

3.2.1.1. [**init:(id)**delegate **preferSingleton:(BOOL)**usingNoti **clientId:(NSString*)**Id **clientKey:(NSString*)**Key **clientSecret:(NSString*)**Secret **userId:(NSString*)**userId **complete:(id)**complete]

3rd Party Application 이 AI 비서와 연동하기 위해서 GPlugin 을 생성한다. 3rd Party Application 이 G-Plugin 라이브러리를 사용하기 전에 반드시 호출하여야 한다.

Parameter 정의

Column Name	Type	설명
delegate	id	라이브러리 생성과 동시에 delegate 를 지정 가능
preferSingleton	Boolean	GPlugin 을 싱글톤 객체로 사용할 지 여부 싱글톤으로 사용하는 경우에만 Notification 작동
clientId	String	기가지니 포탈 또는 KT 담당자를 통해 발급
clientKey	String	기가지니 포탈 또는 KT 담당자를 통해 발급
clientSecret	String	기가지니 포탈 또는 KT 담당자를 통해 발급
userId	String	사용자 식별을 위한 정보로 3rd party Application 에서 생성하여 입력한다. VOC 대응을 위한 식별자이며, 만약 이 정보가 없는 경우 단말의 MAC 주소를 사용자 식별 정보로 사용하게 된다. (44 자리 이내로 권장)
complete	id	라이브러리 생성 결과를 전달 받음

3.2.1.2. init 호출 예시 (Objective-C)

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

- (void)initGPlugin {
    [GPlugin init:kUsingNoti?nil:self preferSingleton:kUsingNoti
clientId:_clientId clientKey:_clientKey clientSecret:_clientSecret
userId:_userId complete:^(id gplugObject) {
        if (gplugObject != nil) {
            dispatch_async(dispatch_get_main_queue(), ^{
                self.gPlug = (GPlugin *)gplugObject;
                // Setup GPlugin
            });
        }
        else {
            NSLog(@"gplug2.0 init fail 입력소스 확인 필요");
        }
    }];
}
```

3.2.1.3. init 호출 예시 (Swift)

```
import GPlugin

func initGPlugin() {
    GPlugin.init(delegate,
             preferSingleton: kUsingNotification.boolValue,
             clientId: clientId,
             clientKey: clientKey,
             clientSecret: clientSecret,
             userId: userKey) { (gplugObject) in

        if let gplug = gplugObject as? GPlugin {
            DispatchQueue.main.async {
                self.gPlug = gplug
                // Setup GPlugin
            }
        }
        else {
            print("gplug2.0 init fail 입력소스 확인 필요")
        }
    };
}
```

3.2.2. G-Plugin 서버 정보 변경

3.2.2.1. [setServerInfo:(NSString *)serverIP grpcPort:(NSString *)grpcPort restPort:(NSString *)restPort]

3rd Party Application 이 G-Plugin 라이브러리에서 통신할 서버의 정보를 설정한다. 해당 API 를 호출하지 않으면 Default 는 상용서버로 연동하게 된다. 연동 서버 정보를 변경하기 위해서는 Init 하기 전에 해당 API 를 먼저 호출하여야 한다.

Parameter 정의

Column Name	Type	설명
serverIP	String	연동 서버 IP
grpcPort	String	연동 서버 gRPC Port
restPort	String	연동 서버 rest Port

3.2.2.2. setServerInfo 사용 예시 (Objective-C, Swift)

```
// Objective-C
[gPlug setServerInfo:@“inside.gigagenie.ai” grpcPort:@“50051”
restPort:@“9080”];

// Swift
gPlug.setServerInfo(“inside.gigagenie.ai”, grpcPort:“50051”, restPort:“9080”)
```

3.2.3. G-Plugin 서비스 시작

3.2.3.1. [startGPlugin]

3rd Party Application 이 G-Plugin 라이브러리를 사용하기 위해 G-Plugin 서비스를 시작한다.

Parameter 정의

Column Name	Type	설명

3.2.3.2. startGPlugin 사용 예시 (Objective-C, Swift)

```
// Objective-C
[gPlug startGPlugin];

// Swift
gPlug.start()
```

※ 스위프트 3.2 부터는 Class 이름이 메소드에 포함된 경우 자동으로 생략됨

3.2.4. G-Plugin 서비스 종료

3.2.4.1. [stopGPlugin]

3rd Party Application 이 G-Plugin 라이브러리를 사용을 종료하기 위해 G-Plugin 서비스를 종료한다.

Parameter 정의

Column Name	Type	설명

3.2.4.2. stopGPlugin 사용 예시 (Objective-C, Swift)

```
// Objective-C
[gPlug stopGPlugin];

// Swift
gPlug.stop()
```

※ 스위프트 3.2 부터는 Class 이름이 메소드에 포함된 경우 자동으로 생략됨

3.2.5. G-Plugin 서비스 Callback 등록

3.2.5.1. –[setGPluginListener:(id <GPluginListener>)listener]

3rd Party Application 이 G-Plugin 라이브러리로부터 Notification 이나 이벤트 등을 수신하기 위한 callback 을 등록한다.

1) **onError:**(NSError *)error
   - G-Plugin 에러 전달. UI 작업을 쉽게 하기 위해서 Error 도 이벤트로 전달한다.

2) **onCommand**:(NSDictionary *)results
   - 음성 또는 텍스트 인식 해석 결과 전달
   - 전달받은 actionType 종류에 따라 아래와 같은 시나리오로 처리하면 된다.
     - dialog_response: 대화서버 응답을 파싱하여 서비스 앱의 flow 에 맞도록 처리
     - media_stream: 전달받은 음성 stream 을 재생
     - media_url: 전달받은 url 을 통해 미디어 재생.

- webview_url: 전달받은 url 을 웹뷰에서 실행.
- start_voice: 음성인식(voiceCommand) 시작 요청.

3) **onEvent**:(NSDictionary *)event **[첨부 1] G-Plugin EventCode 참고**

- 음성인식(음성명령 시작/종료/에러), TTS(재생완료/취소) 이벤트 전달
- 전달받은 이벤트 코드를 수신하여, UI 및 음성인식 재시작 혹은 음성 명령어 모드를 동작시킨다.

Parameter 정의

Column Name	Type	설명
listener	GPluginListener	G-Plugin callback 수신

3.2.5.2. setGPluginListener 등록 및 Delegate methods 예시 (Objective-C)

```
// 헤더파일 정의
#import <GPlugin/GPlugin.h>

@interface ObjcViewController() <GPluginListener>
@end
```

```
// *.m 파일 정의
[gPlug setGPluginListener:self];

// MARK: - GPlugin Handler by Notification
- (void)gPluginHandler:(NSNotification *)noti {
    if (noti && noti.userInfo) {
        NSDictionary *userInfo = noti.userInfo;

        if ([userInfo[GPLUGIN_NOTIFICATION_TYPE_KEY]
isEqualToString:GPLUGIN_NOTIFICATION_TYPE_ERROR]) {
            NSError *error = userInfo[@"error"];

            [self gPlugin:self.gPlug onError:error];
        }
        else if ([userInfo[GPLUGIN_NOTIFICATION_TYPE_KEY]
isEqualToString:GPLUGIN_NOTIFICATION_TYPE_EVENT]) {
            NSDictionary *event = userInfo[@"event"];

            [self gPlugin:self.gPlug onEvent:event];
        }
        else if ([userInfo[GPLUGIN_NOTIFICATION_TYPE_KEY]
isEqualToString:GPLUGIN_NOTIFICATION_TYPE_ONCOMMAND]) {
            NSDictionary *command = userInfo[@"command"];

            [self gPlugin:self.gPlug onCommand:command];
```

```
        }
    }
}

// MARK: - GPlugin Listener (Delegate)
- (void)gPlugin:(GPlugin *)gPlug onError:(NSError *)error {
    ...
}

- (void)gPlugin:(GPlugin *)gPlug onEvent:(NSDictionary *)event {
    if (event != nil) {
        if (event[@"eventCode"] != nil) {
            NSString *eventCode = event[@"eventCode"];
            NSString *eventData = @"";

            if (event[@"data"] != nil) {
                eventData = event[@"data"];
            }

            if ([eventCode isEqualToString:@"KWS_STARTED"]) {
                // 호출어 인식 시작
            }
            else if ([eventCode isEqualToString:@"KWS_DETECTED"]) {
                // 호출어 인식 완료
            }
            else if ([eventCode isEqualToString:@"KWS_ERROR"]) {
                // 호출어 인식 에러
            }
            else if ([eventCode isEqualToString:@"VOICE_STARTED"]) {
                // 음성 인식 시작
            }
            else if ([eventCode isEqualToString:@"VOICE_STOPPED"]) {
                // 음성 인식 완료
            }
            else if ([eventCode isEqualToString:@"SERVER_ERROR"]) {
                // 서버 에러
            }
        }
    }
}

- (void)gPlugin:(GPlugin *)gPlug onCommand:(NSDictionary *)result {
    NSString *actionType = result[@"actionType"];
    NSString *commandType = result[@"commandType"];
    NSData *mediastream = result[@"mediastream"];
    NSDictionary *payload = result[@"payload"];

    if ([actionType isEqualToString:@"media_stream"]) {
        // 미디어 스트림 Play
    }
```

```
    else if ([actionType isEqualToString:@"media_url"]) {
        // 미디어를 play 하거나 제어하는 명령 수행
    }
    else if ([actionType isEqualToString:@"webview_url"]) {
        // 전달받은 url을 웹뷰로 실행
    }
    else if ([actionType isEqualToString:@"start_voice"]) {
        // 음성인식 시작
    }
    else if ([actionType isEqualToString:@"dialog_response"]) {
        // 단말 특화 명령 수행
    }
}
```

3.2.5.3. setGPluginListener 등록 및 Delegate methods 예시 (Swift)

```
// SwiftViewController.swift

import GPlugin

class SwiftViewController: UIViewController, GPluginListener {

    gPlug.setGPluginListener(self)

    // MARK: - GPlugin Handler by Notification
    @objc func gPluginHandler(_ noti: Notification) {
        guard let userInfo = noti.userInfo else {
            return
        }
        if let resultKey = userInfo[GPLUGIN_NOTIFICATION_TYPE_KEY] as? String,
            resultKey == GPLUGIN_NOTIFICATION_TYPE_ERROR,
            let error = userInfo["error"] as? NSError {

            gPlugin(self.gPlug!, onError: error)
        }
        else if let resultKey = userInfo[GPLUGIN_NOTIFICATION_TYPE_KEY] as?
String,
            resultKey == GPLUGIN_NOTIFICATION_TYPE_EVENT,
            let event = userInfo["event"] as? [String : String] {

            gPlugin(self.gPlug!, onEvent: event)
        }
        else if let resultKey = userInfo[GPLUGIN_NOTIFICATION_TYPE_KEY] as?
String,
            resultKey == GPLUGIN_NOTIFICATION_TYPE_RMS_CHANGED,
            let isVoiceBegun = userInfo["isVoiceBegun"] as? Bool,
            let rmsdB = userInfo["rmsdB"] as? Float {

            gPlugin(self.gPlug!, onRmsChanged: isVoiceBegun, rmsdB: rmsdB)
```

```
        }
        else if let resultKey = userInfo[GPLUGIN_NOTIFICATION_TYPE_KEY] as?
String,
            resultKey == GPLUGIN_NOTIFICATION_TYPE_ONCOMMAND,
            let command = userInfo["command"] as? [String : String] {

            gPlugin(self.gPlug!, onCommand: command)
        }
    }

    // MARK: - GPlugin Delegate
    func gPlugin(_ gPlug: GPlugin, onError error: Error) {
        print(String.init(format: "error = %@", error! as CVarArg))
    }

    func gPlugin(_ gPlug: GPlugin, onEvent event: [AnyHashable : Any]) {
        guard let eventCode: String = event["eventCode"] as? String,
            let eventData: String = event["data"] as? String else {
                print("No eventCode or data")
                return
        }

        switch eventCode {
        case " KWS_STARTED":
            // 호출어 인식 시작
        case " KWS_DETECTED":
            // 호출어 인식 완료
        case " KWS_ERROR":
            // 호출어 인식 에러
        case " VOICE_STARTED":
            // 음성 인식 시작
        case " VOICE_STOPPED":
            // 음성 인식 완료
        case " SERVER_ERROR":
            // 서버 에러
        default:
        }
    }

    func gPlugin(_ gPlug: GPlugin, onCommand result: [AnyHashable : Any]) {

        var actionType: String?
        var commandType: String?
        var mediastream: Data?
        actionType = result["actionType"] as? String
        commandType = result["commandType"] as? String
        mediastream = result["mediastream"] as? Data

        if actionType == "media_stream" {
            if let voice = mediastream {
```

```
                // 미디어 스트림 Play
            }
        }
        else if actionType == "media_url" {
            // 미디어를 play 하거나 제어하는 명령 수행
        }
        else if actionType == "webview_url" {
            // 전달받은 url을 웹뷰로 실행
        }
        else if actionType == "dialog_response" {
            // 단말 특화 대화 수행
        }
        else if actionType == "start_voice" {
            // 음성 인식 시작
        }
    }

}
```

3.2.6. G-Plugin 버전정보 조회

3.2.6.1. [getGPluginVersion]

3rd Party Application 이 현재 사용중인 G-Plugin 라이브러리 버전정보를 조회한다.

3.2.6.2. getGPluginVersion 사용 예시 (Objective-C, Swift)

```
// Objective-C
NSString *gPluginVer = [gPlug getGPluginVersion];


// Swift
let gPluginVer = gPlug.getVersion()
```

3.2.7. 라이브러리 내부 로그 보기 설정

3.2.7.1. [setLogVisible:(bool)flag]

라이브러리 내부 API 로그를 보기 위한 설정

Column Name	Type	설명
flag	Boolean	내부 로그 보기 설정 - true : visible - false : invisible

3.2.7.2. setLogVisible 호출 예시

```
// Objective-C
[gPlug setLogVisible:true]; // 내부 로그 보기 설정

// Swift
gPlug.setLogVisible(true) // 내부 로그 보기 설정
```

3.2.8. 단말 위치 정보 업데이트

3.2.8.1. [updateLocInfoV2:(NSString *)x latitude:(NSString *)y address:(NSString *)addr]

라이브러리 내부 API 로그를 보기 위한 설정

Column Name	Type	설명
longitude	NSString	위도 값 설정
latitude	NSString	경도 값 설정
address	NSString	주소 설정

3.2.8.2. setLogVisible 호출 예시

```
// Objective-C
[gPlug updateLocInfoV2:"127.027530" latitude:"37.4724275" address:"서울시 서초구 우면동"];

// Swift
gPlug.updateLocInfoV2("127.027530", latitude:"37.4724275", address:"서울시 서초구 우면동")
```

3.3. G-Plugin Main Service API

3.3.1. G-Plugin Main Service Flow

3rd Party Application 이 G-Plugin 서비스를 사용하기 위해서는 먼저 인증 및 초기화(init API 호출)작업을 통해 Gplugin 객체를 생성하고 startGplugin 을 호출해야 한다. GiGA Genie AI 서비스를 요청하는 방법은 "3.3.1.1 G-Plugin Command 요청 Flow" 에서 설명한다. G-Plugin 은 GiGA Genie AI 서비스에서 해석된 내용을 콜백으로 리턴해주며, 3rd party 에서는 onCommand

콜백으로 전달받은 액션을 수행해야 한다. 해당 시나리오에 대한 Flow 는 "3.3.1.2 G-Plugin Service 제어 Flow"에서 설명한다.

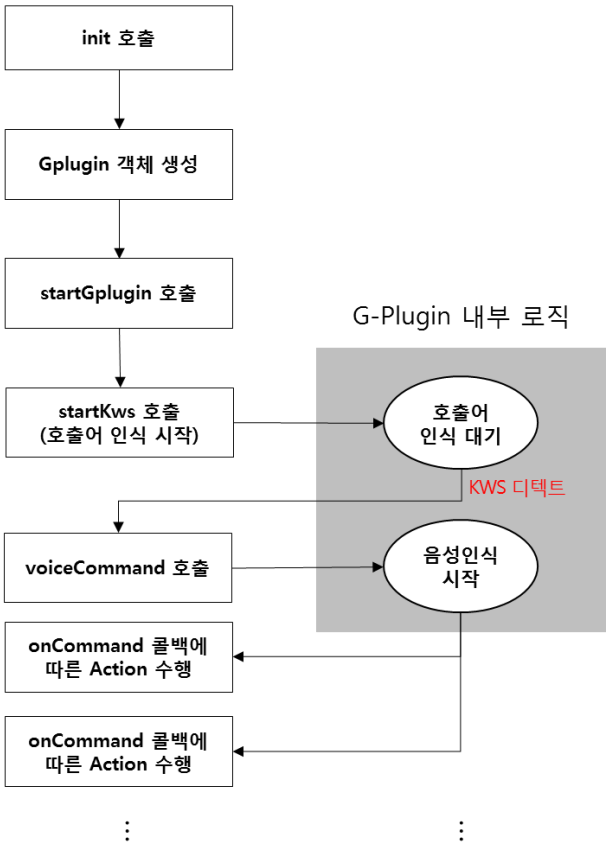
3.3.1.1. G-Plugin Command 요청 Flow

3rd Party Application이 G-Plugin을 통해 GiGA Genie AI 서비스를 요청하기 위해서는 아래 2 가지 방법을 사용할 수 있다. 호출어 인식 및 음성인식을 위한 마이크 제어 권한은 G-Plugin 이 가지고 있다.

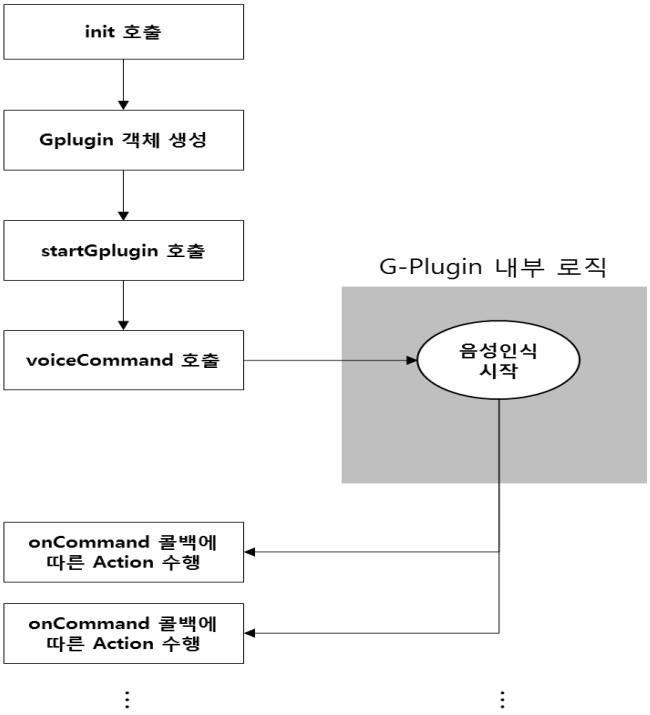
1) startKws API를 호출하여 호출어 인식 대기 상태를 요청하고, 이 상태에서 호출어가 인식된 경우(호출어 인식 이벤트 수신), voiceCommand API를 호출하면 음성인식을 시작하게 된다.

2) voiceCommand API를 호출한 경우 G-Plugin에서 바로 음성인식을 시작하게 된다.

KWS(호출어) 디텍트를 통해 음성인식을 시작하는 Flow

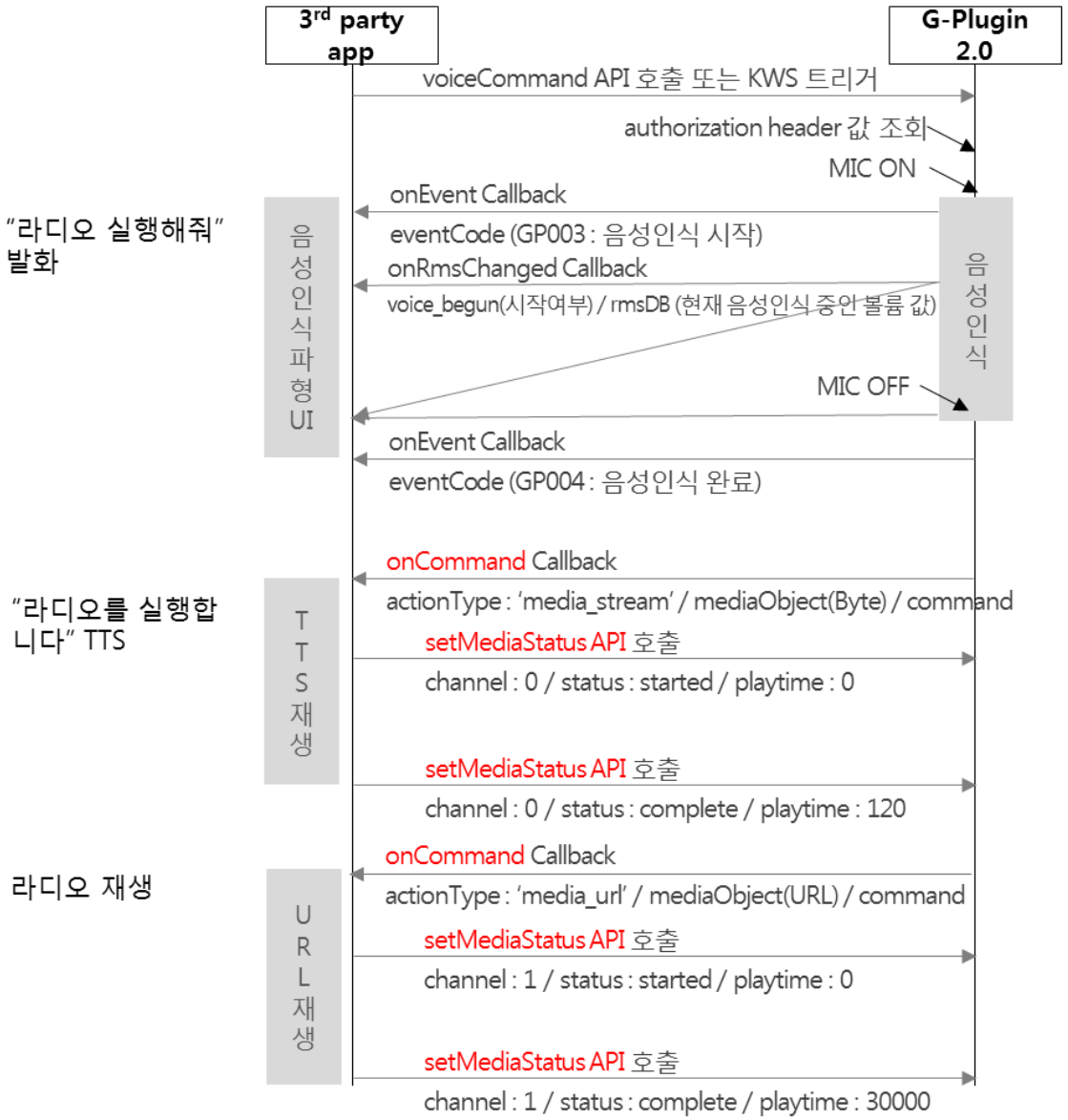


3rd party app이 음성인식을 직접 요청하여 시작하는 Flow



3.3.1.2. G-Plugin Service 제어 Flow

3rd Party는 GiGA Genie AI 서비스를 요청한 이후 onCommand 콜백으로 전달받은 액션을 수행해야 한다. 또한 3rd party application이 미디어를 제어하는 경우(재생, 일시정지, 중단, 완료 등) G-Plugin으로 현재 미디어 재생 상태를 G-Plugin으로 전달해주어야 다음 Command를 정상적으로 콜백 받을 수 있다. Callback 종류에 따른 시나리오 및 각 옵션에 대한 설명은 "3.3.4. Callback 및 미디어 재생 상태 업데이트"에서 설명한다.



3.3.2. G-Plugin 호출어 API

3.3.2.1. 현재 호출어 조회

3.3.2.1.1. [getKwsId]

3rd Party Application 이 G-Plugin 을 통해 현재 호출어를 조회한다.

호출어 ID : 0(기가지니), 1(지니야)

- default 호출어 : 지니야(1)

Column Name	Type	설명

3.3.2.1.2. getKwsId 사용 예시 (Objective-C, Swift)

```
// Objective-C
int kwsID = [gPlug getKwsId];

// Swift
let kwsID = gPlug.getKwsId()
```

3.3.2.2. 호출어 변경

3.3.2.2.1. [setKwsId:(int)kwsId]

3rd Party Application 이 G-Plugin 을 통해 호출어를 변경한다.

Column Name	Type	설명
kwsId	int	호출어 변경 0 : 기가지니 1 : 지니야

3.3.2.2.2. setKwsId 사용 예시 (Objective-C, Swift)

```
// Objective-C
[gPlug setKwsId:1]; // 호출어가 지니야로 설정됨

// Swift
gPlug.setKwsId(1)
```

3.3.2.3. 호출어 인식 사용여부 설정

3.3.2.3.1. [enableKws:(BOOL)isOn]

3rd Party Application 이 G-Plugin 을 통해 호출어 인식 사용여부를 설정한다.

Column Name	Type	설명
isOn	boolean	호출어 인식 사용 여부 설정 - true : 호출어 인식 활성화 - false : 호출어 인식 비활성화

3.3.2.3.2. enableKws 사용 예시 (Objective-C, Swift)

```
// Objective-C
[gPlug enableKws:false];

// Swift
gPlug.enableKws(false)
```

3.3.2.4. 호출어 인식 사용여부 조회

3.3.2.4.1. [checkKwsAvailable]

3rd Party Application 이 GPlugin 을 통해 호출어 사용여부를 조회한다. (enableKws 에 의해 설정 값을 조회하며 true(사용), false(미사용)으로 리턴된다.)

Column Name	Type	설명

3.3.2.4.2. checkKwsAvailable 사용 예시 (Objective-C, Swift)

```
// Objective-C
BOOL isKwsAvailable = [gPlug checkKwsAvailable];

// Swift
let isKwsAvailable = gPlug.checkKwsAvailable
```

3.3.2.5. 호출어 인식 시작

3.3.2.5.1. [startKws]

3rd Party Application 이 GPlugin 을 통해 호출어 인식을 시작한다.

Column Name	Type	설명

3.3.2.5.2. startKws 사용 예시 (Objective-C, Swift)

```
// Objective-C
 [gPlug startKws];

// Swift
gPlug.startKws()
```

3.3.2.6. 호출어 인식 종료

3.3.2.6.1. [stopKws]

3rd Party Application 이 GPlugin 을 통해 호출어 인식을 종료한다.

Column Name	Type	설명

3.3.2.6.2. stopKws 사용 예시 (Objective-C, Swift)

```
// Objective-C
[gPlug stopKws];

// Swift
gPlug.stopKws()
```

3.3.3. Command 요청

G-Plugin 으로 Command 를 요청하기 위해서 아래 3 가지 방법을 사용할 수 있다.

1. KWS(호출어) 디텍트되어 G-Plugin 내부에서 voiceCommand 함수 호출하여 음성인식 시작
2. 3rd party application 이 직접 voiceCommand 함수 호출하여 음성인식 시작
3. 3rd party application 이 직접 텍스트 명령어를 textCommand 함수로 호출

3.3.3.1. voiceCommand()

3rd Party Application 이 G-Plugin 을 통해 음성인식을 시작한다.

해당 API 사용 시 호출어가 인식된 이후 동작과 동일하게 동작하게 된다.

결과는 onCommand() callback 을 통해 전달된다.

3.3.3.2. voiceCommand 사용 예시

```
// Objective-C
[gPlug voiceCommand];

// Swift
gPlug.voiceCommand()
```

3.3.3.3. textCommand(String msg, String type)

3rd Party Application 이 G-Plugin 을 통해 텍스트 명령을 요청한다. 결과는 onCommand() callback 을 통해 전달된다.

Column Name	Type	설명
msg	String	명령 텍스트를 입력한다.
type	String	명령 텍스트의 종류 - query: 텍스트 쿼리 전송 - event: 이벤트 전송

3.3.3.4. textCommand 사용 예시

```
// Objective-C
[gPlug textCommand:@"오늘 날씨 알려줘" type:@"query"]; // 텍스트 쿼리 전송
[gPlug textCommand:@"ED:D101E200", type:@"event"); // 앱 구동 이벤트 전송

// Swift
gPlug.textCommand("오늘 날씨 알려줘", type:"query")
gPlug.textCommand("ED:D101E200", type:"event")
```

3.3.4. Command 취소

3.3.4.1. 음성인식 종료

3.3.4.1.1. cancelVoiceCmd()

3rd Party Application 이 voiceCommand 요청으로 진행중인 음성인식을 종료한다.

Column Name	Type	설명

3.3.4.1.2. cancelVoiceCmd 사용 예시

```
// Objective-C
[gPlug cancelVoiceCmd];

// Swift
gPlug.cancelVoiceCmd()
```

3.3.5. Callback 및 미디어 재생 상태 업데이트

3.3.5.1. G-plugin Callback

G-plugin 은 3rd Party Application 으로 음성 또는 텍스트 인식 해석 결과를 onCommand Callback 으로 전달한다. 3rd Party Application 은 onCommand Callback 의 actionType 에 따라 미디어 재생 등의 action 을 수행해야 한다. **[3.2.5] G-Plugin 서비스 Callback 등록 참고**

3.3.5.1.1. onCommand - dialog_response

onCommand Callback 의 actionType 이 dialog_response 인 경우, 대화서버 응답을 파싱하여 서비스 앱의 flow 에 맞도록 처리해야 한다. 기가지니 포탈의 Dialog Kit 으로 서비스 시나리오를 구성하면 대화서버의 응답을 custom 할 수 있다.

예제)

```
"actionType":"dialog_response",
"dialogResponse": JSON,   //JSON포맷의 대화서버 응답 결과로 3rd party에서 파싱하여 처리
```

① dialogResponse 의 JSON 포맷

key	Description
Intent	DialogKit에서 정의한 Intent
appInfo	DialogKit에서 정의한 NE정보

3.3.5.1.2. onCommand - media_stream

onCommand Callback 의 actionType 이 media_stream 인 경우, Callback 의 mediastream 을 재생해야 한다. 특히 TTS 재생이 요청될 때 사용된다. mediastream 정보는 String 형태로 전달되며, 이를 다음과 같이 byte array 로 변환하여 사용할 수 있다.

- byte[] decodedString = Base64.decode(mediastream, Base64.DEFAULT);

예제)

```
“actionType”:”media_stream”,
“mediastream”: AUDIO_STEAM_STRING, //음성 ByteArray (바로 재생 또는 audio file 생성 가능)
```

3.3.5.1.3. onCommand - media_stream_meta

media_stream Callback 이 전달될때, 해당 stream 의 meta 정보는 media_stream_meta Callback 으로 연속적으로 전달된다. 이는 TTS 재생 메시지를 화면에 표시하거나, 응답의 감성 정보를 표현할 때 사용할 수 있는 정보이다.

예제)

```
"actionType":"media_stream_meta",
"commandType":"Req_PLMD",
"payload":
{
    "cmdOpt":{
        "actOnOther":"pause",
        "channel":"0",
        "errorCode":0,
        "metaInfo":
        {"actFeel":"Neutral",
        "media_playtime":0,
        "media_size":0,
        "mesg":"안녕하세요?",
        "sentencePattern":""
        },
        "playNotiTime":0,
        "playTime":0
        }
    }
}
```

- cmdOpt: 명령어 옵션으로 다음의 옵션 전달
    - channel: 출력할 채널 번호
    - actOnOther: 다른 출력 미디어에 대한 action (다른 출력 미디어에 대한 액션 표 참조)

- metaInfo: media 메타 정보가 설정됨(metaInfo: TTS 재생하는 Text 정보 / actFeel: 응답의 감성 정보)

① 다른 출력 미디어에 대한 액션(actOnOther)

Action	Description
mute	현재 다른 미디어가 재생 중이면 mute 시키고, 미디어 출력 완료 후에도 mute 상태 유지
muteR	현재 다른 미디어가 재생 중이면 mute 시킴, 미디어 출력 완료 후 unmute 시킴
pause	현재 다른 미디어가 재생 중이면 pause 시키고, 미디어 출력 완료 후에도 pause상태 유지
pauseR	현재 다른 미디어가 재생 중이면 pause 시킴, 미디어 출력 완료 후 다른 미디어 resume 시킴
stop	현재 다른 미디어가 재생 중이면 stop 시킴, 다른 미디어 완전 재생 종료
volDown	현재 다른 미디어가 재생 중이면 volume-down 시킴, 미디어 출력 완료 후에도 volume-down 상태 유지
volDownR	현재 다른 미디어가 재생 중이면 volume-Down 시킴, 미디어 출력 완료 후 기존 볼륨 복구

3.3.5.1.4. onCommand - media_url

onCommand Callback 의 actionType 이 media_url 인 경우, 전달받은 url 을 통해 미디어 재생해야 한다. 미디어 재생에 대한 옵션과 metadata 는 onCommand Callback 의 payload(JSON 포맷)로 아래와 같이 전달된다.

예제)

```
"actionType":"media_url",
"payload":
{
    "cmdOpt":{
        "channel": 1,
        "actOnOther": "pause",
        "url": "http://dev-apis.ginie.co.kr/api/v1/tracks/87101805",
        "playNotiTime": 60000,
        "metaInfo": { media_type:'music', title:'좋은날', artist:'아이유', ……},
        }
```

```
    }
}
```

- cmdOpt: 명령어 옵션으로 다음의 옵션 전달
    - channel: 출력할 채널 번호
    - actOnOther: 다른 출력 미디어에 대한 action(다른 출력 미디어에 대한 액션 표 참조)
    - url: 출력 URL (channel 이 0 이면 null)
    - playNotiTime: 특정 시간 플레이 되었는지 노티해주는 시간(지니뮤직은 60 초)
    - metaInfo: media 메타 정보가 설정됨(미디어 메타 정보 표 참조)

① 다른 출력 미디어에 대한 액션(actOnOther)

Action	Description
mute	현재 다른 미디어가 재생 중이면 mute 시키고, 미디어 출력 완료 후에도 mute 상태 유지
muteR	현재 다른 미디어가 재생 중이면 mute 시킴, 미디어 출력 완료 후 unmute 시킴
pause	현재 다른 미디어가 재생 중이면 pause 시키고, 미디어 출력 완료 후에도 pause상태 유지
pauseR	현재 다른 미디어가 재생 중이면 pause 시킴, 미디어 출력 완료 후 다른 미디어 resume 시킴
stop	현재 다른 미디어가 재생 중이면 stop 시킴, 다른 미디어 완전 재생 종료
volDown	현재 다른 미디어가 재생 중이면 volume-down 시킴, 미디어 출력 완료 후에도 volume-down 상태 유지
volDownR	현재 다른 미디어가 재생 중이면 volume-Down 시킴, 미디어 출력 완료 후 기존 볼륨 복구

② 미디어 메타 정보(metaInfo)

Key		Value		
media_type	mand	**music**	**radio**	**podcast**
title	mand	노래 제목	라디오 채널명	팟캐스트 제목
description	opt	설명		
artist	opt	가수명	방송사	제작자
imageurl	opt	앨범 이미지	채널 이미지	팟캐스트 이미지

media_name	opt	노래제목		에피소드명
media_playtime	opt	재생 시간		
media_size	opt	Bytes		
category	opt	카테고리(가요, 팝송)		
album_name	opt	앨범 명		

3.3.5.1.5. onCommand - webview_url

onCommand Callback 의 actionType 이 webview_url 인 경우, 전달받은 url 을 웹뷰로 실행해야 한다. (예, 지니뮤직 로그인 요청이 콜백으로 온 경우)

예제)

```
“actionType”: ”webview_url”,
"oauth_url":"https://dev-auth.genie.co.kr/oauth/authorize?response_type=code&client_id=YjEyYWM0NDEtYzBkNC00MjkzLTkyODQt&redirect_uri=https%3A%2F%2Fagent.gigagenie.ai%3A10089%2Foauthresult%2Fgeniemusic%2F07d87279-0cf8-5ce5-b04f-76941a6fc4f6&state=DlA0K3vlBG9"
```

3.3.5.1.6. onCommand - start_voice

onCommand Callback 의 actionType 이 start_voice 인 경우, voiceCommand API 를 호출하여 음성인식을 진행해야 한다. (예, ‘일정 등록’ 등 연속 발화가 필요한 시나리오의 경우)

예제)

```
“actionType”: ”start_voice”,
“payload”:
{
   “cmdOpt”:{
      ……
   }
}
```

3.3.5.2. 미디어 재생 상태 업데이트

3.3.5.2.1. setMediaStatus()

3rd Party Application 이 미디어를 제어하는 경우(재생, 일시정지, 중단, 완료 등) G-Plugin 으로 현재 미디어 재생 상태를 전달해야 한다. 상태 정보는 필수로 전달해야 하며, 해당 정보를 GiGA Genie AI 서버에서 있어야 상태에 따른 적절한 command 를 판단 가능하다.

Column Name	Type	설명
channel	int	출력하는 채널 정보
status	String	started(시작됨), paused(중지됨), resumed(재시작됨), complete(플래이 완료), stopped(외부 요인에 의한 종료), noti(playNotiTime 에 의한 noti)
playTime	int	플레이 시간(ms)

3.3.5.2.2. setMediaStatus 사용 예시

```
mGplugin.setMediaStatus(0, “paused”, 1200);
```

3.3.6. 음성인식 Audio Session 설정

3.3.6.1. [gPlugin:(GPlugin *)gPlug onInitAudioSession:(AVAudioSession *)audioSession]

음성인식을 위한 오디오 세션을 설정한다. 3rd Party App 에서 아래 호출 예시와 같이 오디오 세션을 설정하면 음성인식 시 (호출어 또는 명령어 인식 모드 진입 시) 매번 호출된다.

잘못 설정하는 경우 음성인식이 동작하지 않을 수 있으며, 음성인식 시작 실패 시 Error Code 를 리턴 한다. 본 delegate 메소드를 구현하지 않는 경우 G-Plugin 내부에 설정된 default audio session 으로 동작하도록 되어있다.

3.3.6.2. onInitAudioSession 세팅 방법

음성인식 오디오 세션을 3rd Party App 에서 설정하기 위해서는 **아래 필수 사항이 구현되어야 한다**.

- AVAudioSessionCategoryPlayAndRecord: 음성 녹음을 위해 세팅 되어야 하는 설정이다.
- AVAudioSessionCategoryOptionMixWithOthers: 오디오 재생과 녹음을 동시에 하기 위해 세팅 되어야 하는 설정이다.

```
-(void)gPlugin:(GPlugin*)gPlug onInitAudioSession:(AVAudioSession*)Session {
    NSError *error = nil;
    [Session setActive:NO error:&error]; //재생중 오디오 중단되는 이슈 방지

    if(gplug.isWakeUp) { // 명령어 모드인 경우
        [Session setCategory: AVAudioSessionCategoryPlayAndRecord
         withOptions: AVAudioSessionCategoryOptionMixWithOthers error:&error];
    }
    else { // 호출어 모드인 경우
        [Session setCategory: AVAudioSessionCategoryPlayAndRecord
         error:&error];
    }

    [Session setActive:YES error:&error]; //반드시 Active: Yes 되어야 함
}
```

3.3.6.3. onInitAudioSession 호출 예시 (전체 코드)

```
- (void)gPlugin:(GPlugin*)gPlug onInitAudioSession:(AVAudioSession*)Session {
    NSError *error = nil;
    [Session setActive:NO error:&error]; //재생중 오디오 중단되는 이슈 방지

    if(self.gplug.checkKwsAvailable) {
        //키워드 인식일 때에는 다른 앱에서 출력되는 Audio와 Mix하여 출력
        // AVAudioSessionCategoryOptionInterruptSpokenAudioAndMixWithOthers 는
반드시 필요한 옵션이며, 서비스 시나리오(블루투스 스피커 연결/해제, 이어폰 연결/해제
등)에 맞게 조정
        if (@available(iOS 10.0, *)) {
            [audioSession setCategory: AVAudioSessionCategoryPlayAndRecord
                  withOptions:  AVAudioSessionCategoryOptionMixWithOthers |
                                AVAudioSessionCategoryOptionDefaultToSpeaker |
                                AVAudioSessionCategoryOptionAllowBluetoothA2DP
                  error:&error];
        } else {
            [audioSession setCategory: AVAudioSessionCategoryPlayAndRecord
                    withOptions: AVAudioSessionCategoryOptionMixWithOthers
                    error:&error];
        }
    }
    else {
        //명령어 인식일 때는 다른 앱에서 출력되는 Audio 정보는 Mute 처리
        if (@available(iOS 10.0, *)) {
            [audioSession setCategory: AVAudioSessionCategoryPlayAndRecord
                   withOptions:AVAudioSessionCategoryOptionDefaultToSpeaker |
                               AudioSessionCategoryOptionAllowBluetoothA2DP
```

```
                    error:&error];
        } else {
            [audioSession setCategory:AVAudioSessionCategoryPlayAndRecord
                      error:&error];
        }
    }
    //AudioSession을 설정 후 setActive로 설정정보를 AudioSession에 반영(필수)
    //setActive설정이 호출되지 않으면, 음성인식 시작이 failed될 수 있음
    [audioSession setActive:YES error:&error];
}
```

3.4. GPlugin TTS API

3rd Party Application 이 GPlugin TTS 서비스를 이용할 수 있다.

3.4.1. TTS 재생 요청

3.4.1.1. [startTTS:(NSString*)msg]

3rd Party Application 이 TTS 재생을 요청한다.

Parameter 정의

Column Name	Type	설명
msg	String	TTS 재생할 메시지

3.4.1.2. startTTS 호출 예시 (Objective-C, Swift)

```
// Objective-C
[gPlug startTTS:@"테스트 중입니다."];

// Swift
gPlug.startTTS("테스트 중입니다.")
```

3.4.2. TTS Audio Session 설정

3.4.2.1. [(GPlugin *)gPlug **onInitAudioSessionTTS**:(AVAudioSession *)session]

TTS 재생을 위한 오디오 세션을 설정한다. 3rd Party App 에서 아래 호출 예시와 같이 오디오 세션을 설정하면 TTS 시작 전 매번 호출된다.

본 delegate 메소드를 구현하지 않는 경우 G-Plugin 내부에 설정된 default audio session 으로 동작하도록 되어있다.

3.4.2.2. onInitAudioSessionTTS 세팅 방법

TTS 오디오 세션을 3rd Party App 에서 설정하기 위해서는 아래 필수 사항이 구현되어야 한다.

- AVAudioSessionCategoryPlayback: 단말이 무음모드이더라도 TTS 소리가 나도록 하는 설정이다.

```
- (void)gPlugin:(GPlugin *)gPlug onInitAudioSessionTTS:(AVAudioSession
*)session {
    NSError *error = nil;

    [[AVAudioSession sharedInstance]
setCategory:AVAudioSessionCategoryPlayback error:&error];
    [[AVAudioSession sharedInstance] setActive:YES error:&error];
}
```

4. 개발 참고 사항 및 이슈 사항

4.1. 예외처리

4.1.1. Siri 에 의한 인터럽트 발생 이슈 해결 코드

Objective-C	swift

- (void)viewDidLoad {
[super viewDidLoad];
 	
[[NSNotificationCenter defaultCenter] addObserver:selfselector: @selector(willEnterForeground) name:UIApplicationWillEnterForegroundNotification object:nil];
}
 	
- (void)willEnterForeground {
if(checkKWSAvailable) {
[self.gPlug startKWS];
}
}
	- override func viewDidLoad() {
	Super.viewDidLoad()
	
	NotificationCenter.default.addObserver(self,selector:#selector(willEnterForeground), name: .UIApplicationWillEnterForegroundNotification, object: nil)
	}
	
	- @objc func willEnterForeground() {
	if checkKWSAvailable {
	self.gPlug?.startKWS ()
	}
	}

4.2. FAQ(TBD)

4.2.1. "CFK_FAILED_CONN" 에러코드 발생하는 경우

음성인식 서버는 TB(TestBed)서버와 상용서버로 구분되어 있다. TB 음성인식 서버에 연동하는 경우, 단말이 WiFi 나 타사 통신망(SKT, LGU+)에 연결되어 있으면 "CFK_FAILED_CONN" 에러코드가 발생한다.

TB 음성인식 서버 연동은 KT 통신망에서만 가능하며, 상용 음성인식 서버는 통신사나 WiFi 연결 상태와 관련 없이 정상 동작한다.

[첨부 1] G-Plugin EventCode

Event Code	설명	전달 데이터
"VOICE_STARTED"	음성인식 시작	event.putString("eventCode", "VOICE_STARTED");
"VOICE_STOPPED"	음성인식 종료	event.putString("eventCode", "VOICE_STOPPED"); event.putString("data", " 음성인식 결과");

"SERVER_ERROR"	서버 오류	event.putString("eventCode", "SERVER_ERROR");
		event.putString("data", errorCause);
		- errorCause: 서버 오류 코드(첨부 2)

"KWS_STARTED"	호출어 인식 시작	event.putString("eventCode", "KWS_STARTED");
"KWS_DETECTED"	호출어 인식 완료	event.putString("eventCode", "KWS_DETECTED");

"KWS_ERROR"	호출어 오류	event.putString("eventCode", "KWS_ERROR");
		event.putString("data", errorCause);
		- errorCause: 호출어 오류 코드(첨부 3)

[첨부 2] G-Plugin ErrorCause

Error Cause	설명
900	음성인식 서버 접속 실패
901	음성인식 시작 후 5 초 이내에 실제 음성 감지가 발생하지 않는 경우 (발화를 하지 않는 경우)
902	음성인식 시작 후 10 초 이내에 음성 끝 감지가 발생하지 않는 경우 (발화를 10 초 이상 계속 하는 경우)
910	TTS 서버 접속 실패
911	TTS 변환 요청 실패
912	TTS 요청 텍스트의 포맷이 잘못되었거나 빈 스트링인 경우
920	대화 서버 접속 실패
921	클라이언트 타입에 상응하는 대화 서버 프로파일 없음
930	Filter 서버 접속 실패
601	음성인식 시작 요청이 오지 않는 경우
602	음성인식 중단 요청이 오지 않는 경우

[첨부 3] KWS ErrorCause

Error Cause	설명	Error 처리 방식
E_INIT	KWS 라이브러리 초기화 실패	호출어 정지(stopKws)
E_MIC	KWS 마이크 연동 실패	호출어 정지(stopKws)
E_NETWORK	KWS 네트워크 연동 실패	호출어 정지(stopKws) 및 시작(startKws)
E_AUDIO	KWS 오디오 오류	호출어 정지(stopKws)
E_RECOGN_TIMEOUT	KWS 인식 결과 타임아웃	호출어 정지(stopKws) 및 시작(startKws)
E_AUDIO_TIMEOUT	KWS 오디오 데이터 입력 대기 초과	호출어 정지(stopKws) 및 시작(startKws)
E_INTERNAL	KWS 내부 오류	호출어 정지(stopKws)
E_SPEECH_TIMEOUT	KWS 음성입력 신호 타임아웃	호출어 정지(stopKws) 및 시작(startKws)
E_UNKNOWN	정의되지 않는 오류	호출어 정지(stopKws)

[첨부 4] 음성 및 대화 Life-cycle

1. startKws 로 호출어 인식 대기
2. 호출어 발화
3. 음성인식 시작
4. 사용자 발화
5. 음성인식 종료
6. 서버에서 response 수신
7. App 에서 action 수행
8. startKws 로 호출어 인식 대기