



## **Mobile User Experience Monitoring - Android & iOS**

## **Steps for Bundling eG Mobile Agent with Application Using Downloaded Artefacts - Android**

The following steps are required to bundle the 'eG Mobile Agent – Flutter' into the apps and activate user experience monitoring.

### **1. Download and install mobile agent**

- a. Download 'eG Mobile agent' artifacts from eG website and copy to the local drive
- b. Create an 'eg-agent' directory under android folder.
- c. Copy agent-gradle-plugin-x.x.x.jar, android-agent-x.x.x.jar and rewriter-agent-x.x.x.jar into this folder just created.

### **2. Add the following lines into android project's build.gradle file:**

```
buildscript {  
    repositories {  
        flatDir {  
            dirs 'eg-agent'  
        }  
    }  
    dependencies {  
        classpath fileTree(include: ['*.jar'], dir: 'eg-agent')  
    }  
}
```

### **3. Add the following lines into application's build.gradle file:**

```
repositories {  
    flatDir {  
        dirs '../eg-agent'  
    }  
}  
apply plugin: 'android'  
apply plugin: 'eg'  
dependencies {  
    implementation fileTree(dir: '../eg-agent', include: ['a*.jar'])  
}
```

### **4. Set application permissions**

Ensure that application requests INTERNET and ACCESS\_NETWORK\_STATE permissions by adding below lines to application's AndroidManifest.xml under manifest tag. Add only if it's not already present.

```
<uses-permission android:name="android.permission.INTERNET" />  
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

Also add ***android:usesCleartextTraffic="true"*** in the application tag of the AndroidManifest.xml file as shown below.

```
<application      android:name=".MyApplication"

    android:allowBackup="true"

    android:icon="@mipmap/logo"

    android:label="@string/app_name"

    android:roundIcon="@mipmap/logo"

    android:supportsRtl="true"

    android:theme="@style/AppTheme"

    android:usesCleartextTraffic="true">
```

## 5. Start the agent

Start the eG Mobile agent while starting the application by implementing the below steps in application's **Main or default activity**.

- a) In application's Main or Default Activity import the eG Mobile Agent class as mentioned in the below box.

```
import android.os.Bundle;
import com.eg.agent.android.AppPlatform;
import com.eg.agent.android.eGAndroidAgent;
```

- b) Copy paste the marked content as shown in the screen shot below which is present in the add/modify component page, into mobile application's onCreate() method of Main or Default Activity.

```
override fun onCreate(savedInstanceState: Bundle?) {

    super.onCreate(savedInstanceState);

    eGAndroidAgent.withCollectorHost("<APP_TOKEN>","<COLLECTOR_URL>").usingSsl(false).withLogging
    Enabled(true).withLogLevel(6).start(getApplicationContext());

    eGAndroidAgent.setAppliactionPlatform(AppPlatform.Flutter)

}
```

**Note:-** Starting the eG mobile agent in any other class is not supported and can cause unexpected or unstable behaviour.

## 6. Build and Run the application

Clean the project and Build the application. While building the application, the console should log messages about transforming classes, artefacts etc. After successful build, run the application in an emulator or mobile device and login to eG manager application to start seeing data.

### Steps for Bundling eG Mobile Agent with Application Using Downloaded Artefacts – iOS

## 1. CocoaPods Installation

### 1.1 Configure using Swift

- 1) In the Podfile for your project, add the following line:

```
pod 'eGMobileAPM', :git => "https://github.com/eginnovation/eGMobileAPM.git"
```

- 2) Close your project in Xcode, and update it by running this command from the Terminal in your project directory:

```
Pod Install
```

- 3) Open your project in Xcode by running this command from the Terminal in your project directory:

```
Open App.xcworkspace
```

- 4) Import the Monitoring Framework into the application (AppDelegate.swift) using the command

```
Import eGMobileAPM
```

- 5) In your AppDelegate.swift file add this call as the first line of application: didFinishLaunchingWithOptions:

```
egAPM.sharedInstance().start(apptoken: "Application  
Token",collectorHost: "Collector Host")
```

In AppDelegate.swift replace the unique application token and collector host that is automatically generated.

- 6) Clean and build your app, and then run it in the simulator or physical device.

## 1.2 Configure using Objective-C

- 1) In the Podfile for your project, add the following line:

```
pod 'eGMobileAPM', :git => "https://github.com/eginnovation/eGMobileAPM.git"
```

- 2) Close your project in Xcode, and update it by running this command from the Terminal in your project directory:

```
Pod Install
```

- 3) Open your project in Xcode by running this command from the Terminal in your project directory:

```
Open App.xcworkspace
```

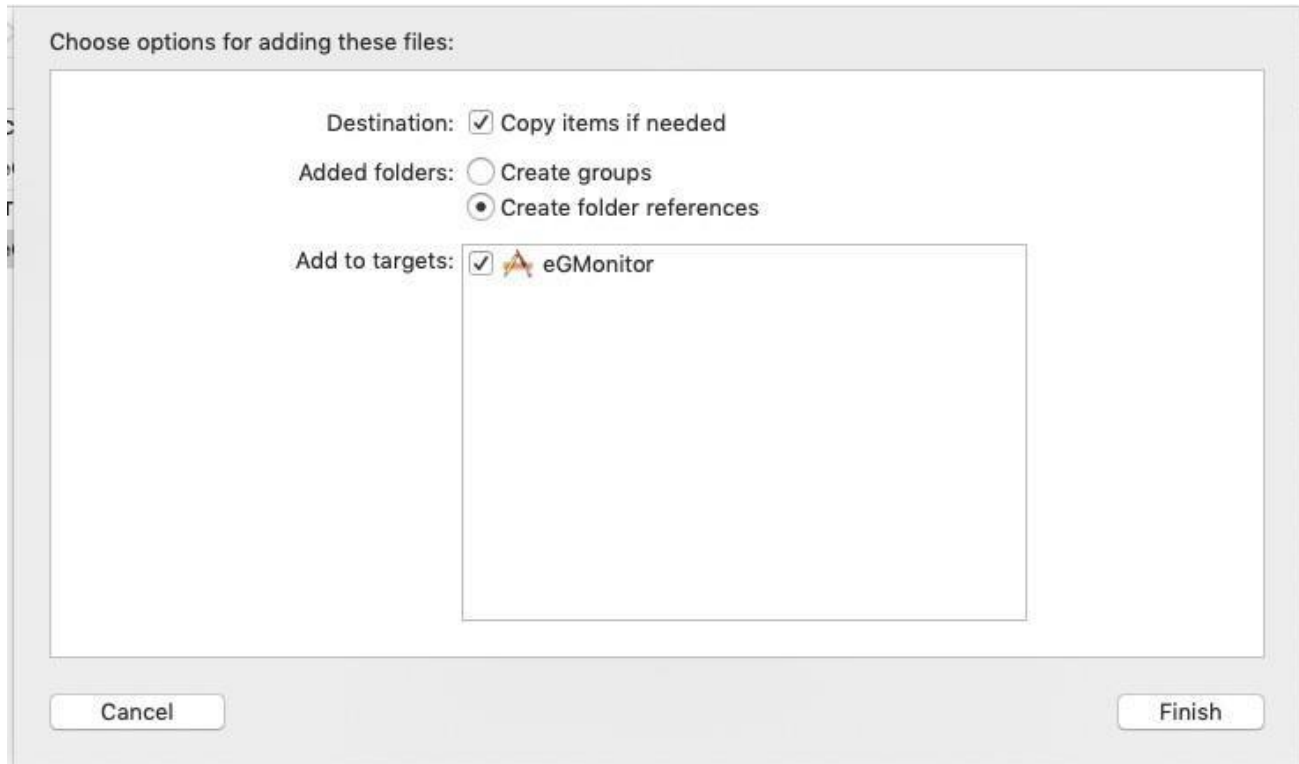
- 4) In your Objective – C project, create a new Swift file. You will be prompted to add a bridging header if you don't have already one. Accept this prompt.
- 5) Import the eGMobileAPM Framework by adding **#import "eGMobileAPM/eGMobileAPM-Swift.h"** to the top of MyApp-Bridging-Header.h
- 6) Import the Monitoring Framework into the application (AppDelegate.m) using the command  
**#import <eGMobileAPM/eGMobileAPM.h>.**
- 7) Add **[egAPM.sharedInstance startWithApptoken:@" Application Token "]  
collectorHost:@" Collector Host "];**  
in your AppDelegate.m and replace the unique application token and collector host that is automatically generated.
- 8) Clean and build your app, and then run it in the simulator or physical device.

## 2. Manual Installation

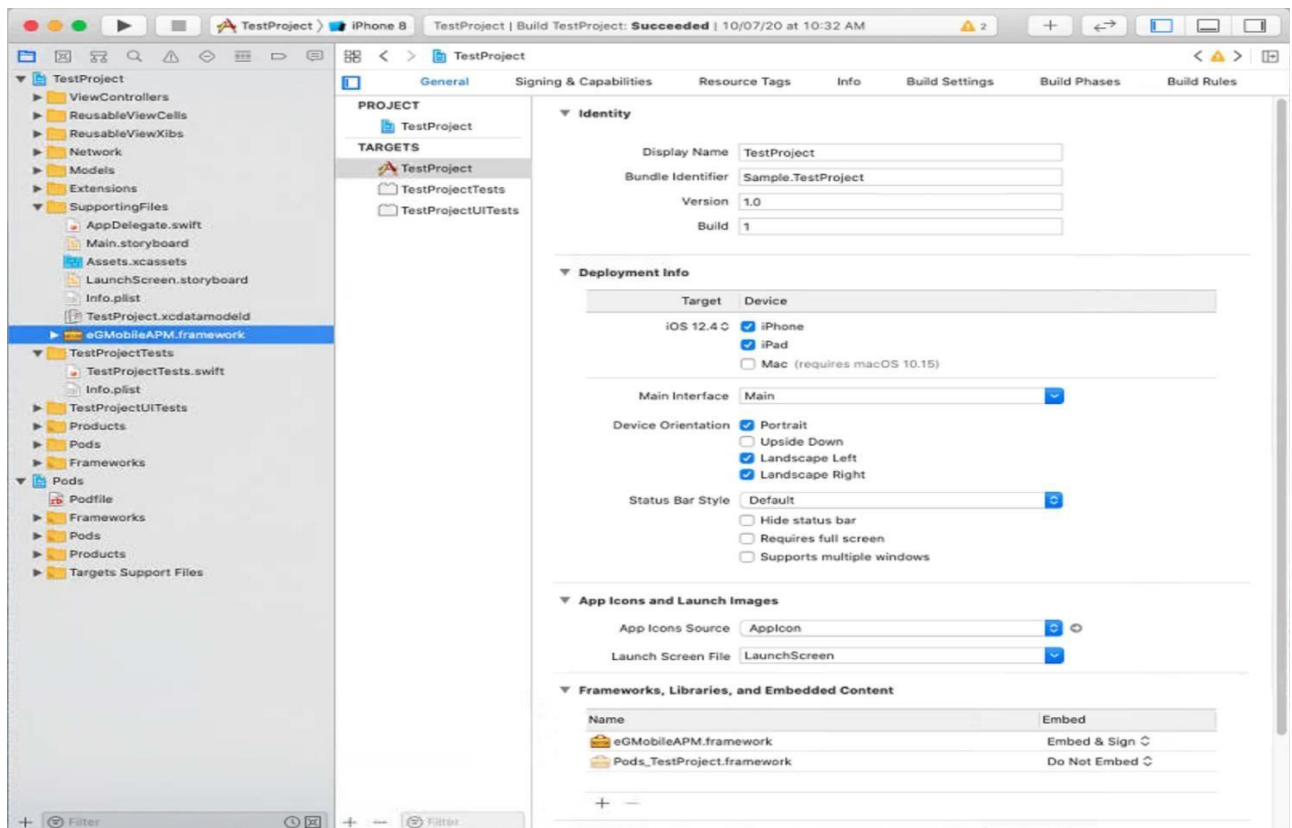
### 2.1 Integrate Monitoring Framework with Swift

The below steps show how to integrate monitoring framework with the swift application.

- 1) Download 'eGMobileAPM Framework' from given path.
- 2) Download and unzip the iOS SDK, drag the "eGMobileAPM.Framework" folder from the finder into your Xcode project. When prompted, select "Copy items into destination" and "Create folder reference"
- 3) After adding choose the below options



- 4) In the **General** settings of your project **Embed&Sign** the eGMobileAPM Framework in Framework, Libraries, and Embedded Content

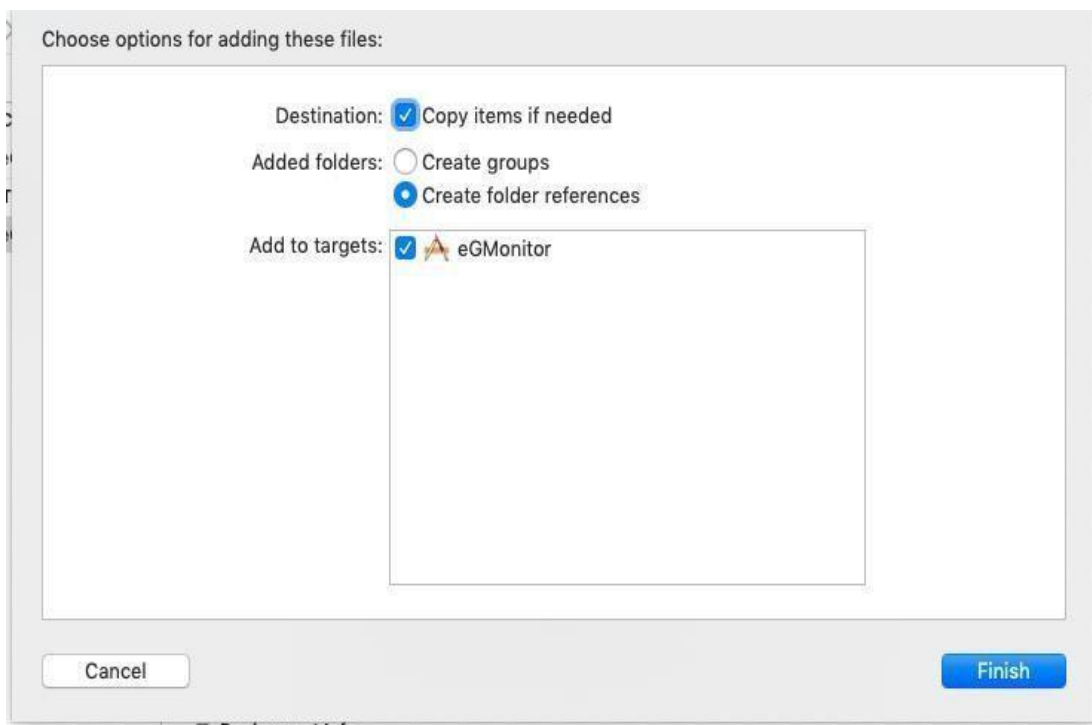


- 5) Import the Monitoring Framework into the application (AppDelegate.swift) using the command: **import eGMobileAPM**
- 6) In your AppDelegate.swift file add this call as the first line of application: `didFinishLaunchingWithOptions:`  
**egAPM.sharedInstance().start(apptoken: "Application Token", collectorHost: "Collector Host")** and replace application token and collector host that is automatically generated.
- 7) Clean the project, Build and Run the application

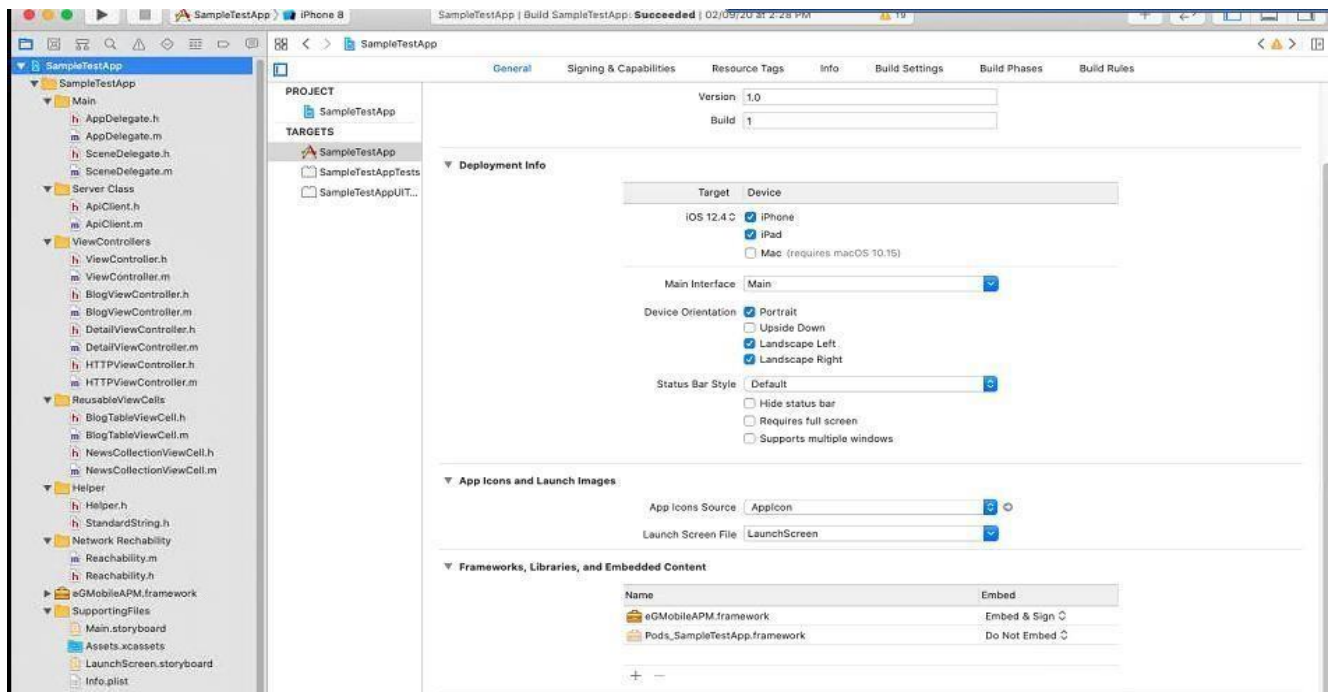
## 2.2 Integrate Monitoring Framework with Objective C

The below steps show how to integrate monitoring framework with the Objective C application.

- 1) Download 'eGMobileAPM Framework' from given path.
- 2) Download and unzip the iOS SDK, drag the "eGMobile.framework" folder from the finder into your Xcode project. When prompted, select "Copy items into destination" and "Create folder reference"
- 3) After adding choose the below options:



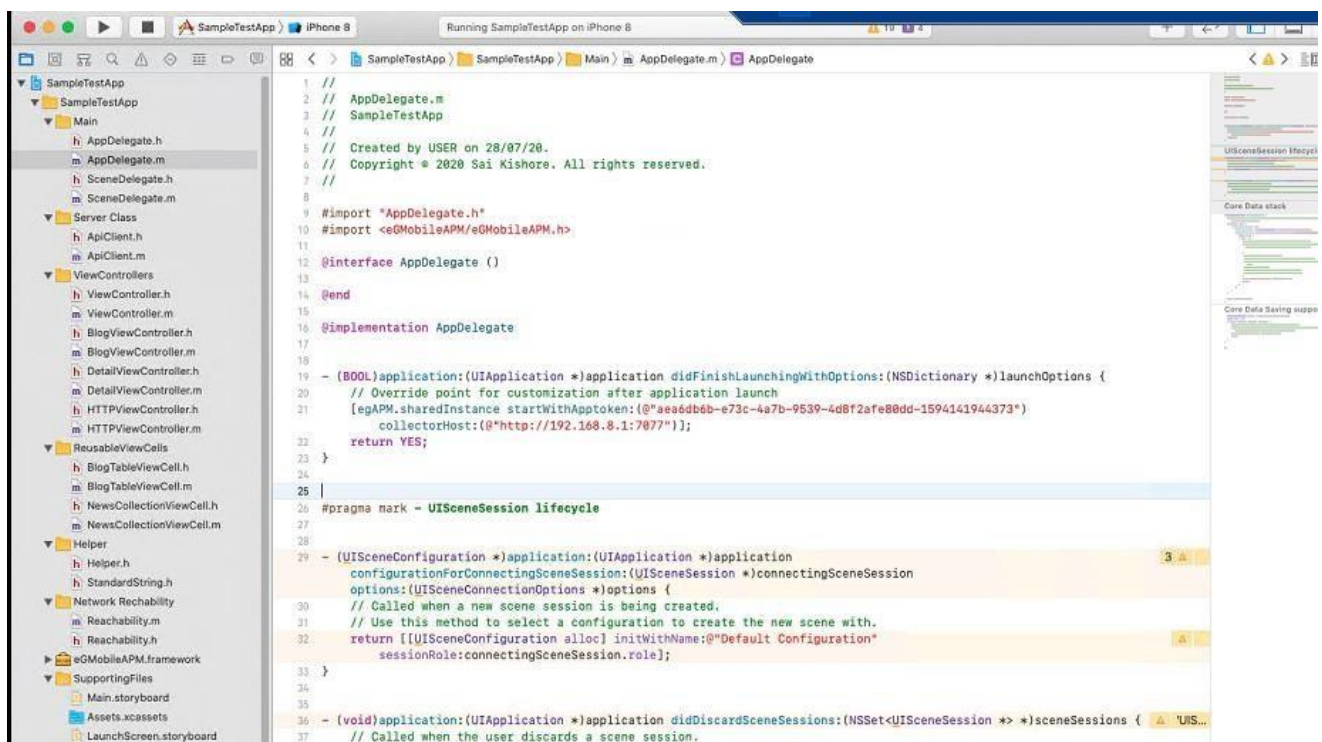
- 4) In the **General** settings of your project **Embed&Sign** the eGMobileAPM Framework in Framework, Libraries, and Embedded Content



- 5) In your Objective – C project, create a new Swift file. You will be prompted to add a bridging header if you don't have already one. Accept this prompt.



- 6) Import the eGMobileAPM Framework by adding:  
**#import "eGMobileAPM/eGMobileAPM-Swift.h"** to the top of MyApp-Bridging-Header.h
- 7) Import the Monitoring Framework into the application (AppDelegate.m) using the command  
**#import <eGMobileAPM/eGMobileAPM.h>**
- 8) In your AppDelegate.m file add this call as the first line of application:  
**didFinishLaunchingWithOptions: [egAPM.sharedInstance startWithApptoken:@"Application Token"] collectorHost:@"Collector Host"];** and replace the unique application token and collector host that is automatically generated.



- 9) Clean and build your app, and then run it in the simulator or physical device.