ethnio ™

# Native App Intercepts

## *on iOS & Android*

VERSION NO. 3
CREATED AUG 29, 2018

ETHNIO, INC.
6121 W SUNSET BLVD
LOS ANGELES, CA 90028
TEL (888) 879-7439

# Overview

There are two basic methods for implementing an Ethnio layer into your iOS or Android app.

## Method 1: WebView

This method requires you to call a standard browser layer, called many different things like “WebView,” based on your own rules, where you can define when it appears and even how it animates, but populate it with content from an ethnio URL that you can customize to match your app.

## Method 2: Our Library

We have libraries for iOS and Android, which are both outlined in this document, but the important thing to keep in mind is that **most mobile developers are very skeptical of this method**, which might not be part of their regular code. We strongly recommend Method 1 because it’s more likely to actually get approved at your organization.

### Ummmm What is WebView?

*There are tons of different names for what is essentially **a minimal browser** that delivers web content in your native app. This allows us to serve up an ethnio screener without your mobile developers having to really implement any new libraries.*

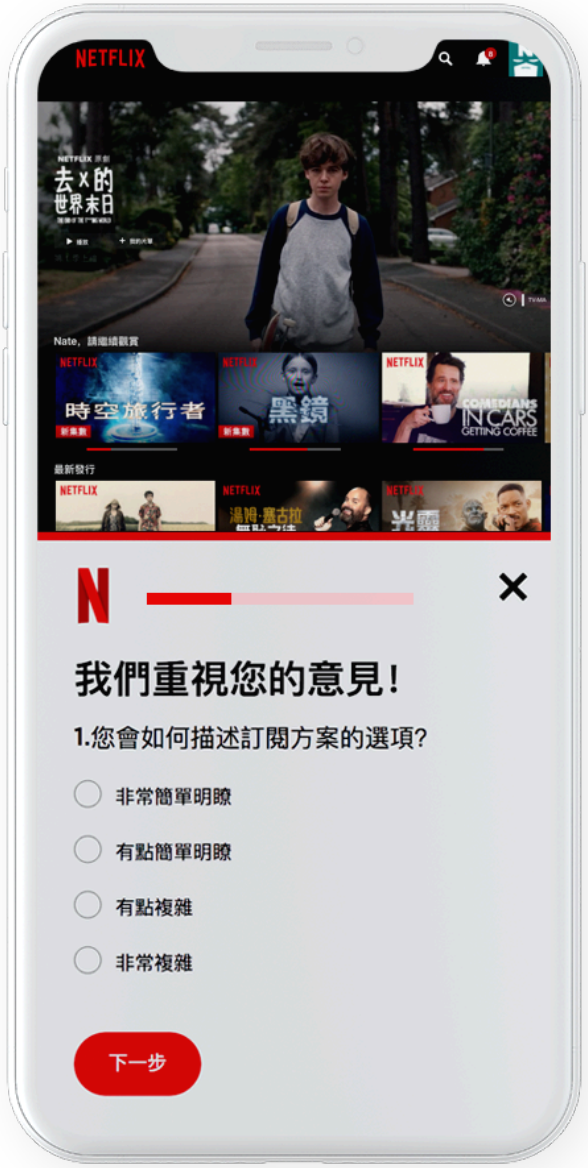
 

# Method 1: WebView

The best method of implementing Ethnio inside a native app on iOS and Android is simply to instantiate a browser layer using standard libraries that are most likely already part of your app. This way you can control what behavior triggers an ethnio screener and there are no external libraries in your app that could make mobile developers nervous. We understand modifying native apps can be much scarier than inserting JavaScript into a web site, so we want to make it as easy as possible. We have special syntax for using ethnio to populate the browser layer inside your app that looks like this:

## ethn.io/preview/22910#app

This is a modification of our standard direct link, where you simply add “mob” into the URL and we include a close button that will work in mobile (app://close) to close that layer. This layer can easily have a little animation to slide up or down, and that’s what AirBnB does, for example. This is what that looks like at right, including custom fonts, which we can do as well.



# Method 2: iOS Library (deprecated)

Conceptually this is very similar to method 1, but the difference is you load a public library so that when you turn an ethnio screener on in your ethnio control panel, it will turn on in your app. In practice, **iOS & Android developers hate this idea, because you need to re-submit an app with another library. Basically, no developers want to do this, hence the need for Method 1.** We no longer support this, but technically it could still work.

## ScreenerViewController

ScreenerViewController is a view controller container UIWebView with the request to service with screeners. Supports all screen sizes and orientations.

## Features

The philosophy behind ScreenerViewController is simple. You create a screener and add ethnio link in XCode project.

## Requirements

ScreenerViewController works starting with iOS 5 version and is compatible with both ARC and non-ARC projects.

## Source files

You can directly add the ScreenerViewController.h and ScreenerViewController.a files to your project. All the steps are outlined on the next page here

1. Download the latest code version.
2. Open your project in Xcode, then drag and drop ScreenerViewController.h and ScreenerViewController.a onto your project (use the "Product Navigator view"). Make sure to select Copy items when asked if you extracted the code archive outside of your project.
3. Include ScreenerViewController wherever you need it with #import “ScreenerViewController.h”.
4. Get ScreenerAppID on site and add it on Info.plist in your project in Xcode, using key "ScreenerAppID".

5. Add base url on Info.plist in your project, using key "ScreenerBaseUrl".

## Usage

Just like any UIViewController, ScreenerViewController can be pushed into a UINavigationController stack or be presented modally, but first you need to check screener status.

```
[ScreenerViewController checkScreenerStatusWithCompletion:^(ScreenerStatusType status) { if (status == ScreenerStatusAvailable) {
ScreenerViewController* screenerWebViewController = [ScreenerViewController new];
[self.navigationController pushViewController:screenerWebViewController animated:YES]; }
}];
[ScreenerViewController checkScreenerStatusWithCompletion:^(ScreenerStatusType status) {

if (status == ScreenerStatusAvailable) {
ScreenerViewController* screenerWebViewController = [ScreenerViewController new];
[self presentModalViewController screenerWebViewController animated:YES completion:nil];
} }];
```

If you are going to show the controller with any other method different from the above you should set dismissBlock. For more examples, including how to use ScreenerViewController, take a look at the demo project.

# Method 2a: Android Library (also deprecated)

All the details are here https://bitbucket.org/Ethnio/ethnio-layer-for-android but below is a basic summary.

## EthnioLib

EthnioLib is an android library (project) with EthnioManager for requests to display an ethnio screener that an existing ethnio creates using the ethnio web app.
Supports all screen sizes and orientations. EthnioLib works starting with android 8 API.

## Source

| | |
|---|---|
| *EthnioDemofolder* | with source code of EthnioDemo project |
| *EthnioLib* | folder with source code of EthnioLib |
| *ethniolib.jar* | latest version of EthnioLib |

## Usage

EthnioLib can open activity or create a fragment with Ethnio content. First of all add EthnioLib.jar to lib folder of your project or add EthnioLib project as a dependency for your project. Get ScrennerAppId from the server and save it as a constant for future use.

*privatestaticfinalStringETHNIO_ID=11111;//example code*
Implement instance of EthnioEventLIstener for callbacks.

*privateEthnioEventListenermEthnioEventListener=newEthnioEventListener{}*
Initialize instance of EthnioManager in onCreate method of activity.

*mEthnioManager=newEthnioManager(this,ETHNIO_ID,mEthnioEventListener>);*
Stop EthnioManager in onDestory method of activity.

*mEthnioManager.stop();*

For showing activity with Ethnio content, register EthnioActivity in AndroidManifest.xml of your project.

*<activityandroid:name ='com.altoros.ethnio.EthnioActivity'/>*
And use the method of EthnioManager's instance for showing it (use code for the difference listeners).

*mEthnioManager.startEthnioActivityIfAvailable(777);*
Activity will appear if Ethnio is available or else send callback with event.instance of EthnioManager as field of activity and initialize it in onCreate method of activity. For creation of a fragment with Ethnio content use the method of EthnioManager's instance.

*mEthnioManager.createEthnioFragmentIfAvailable(4);*
And show fragment from EthnioEventListener's callback.

```
@Override
publicvoidonEthnioFragmentCreated(Fragmentfragment,intlistenerId){
FragmentTransactionfTrans=getSupportFragmentManager().beginTransaction();
fTrans.replace(R.id.frame,fragment);
fTrans.commit();
Toast.makeText(MainActivity.this,listenerId+': Ethnio fragment created’,Toast.LENGTH_SHORT.show();}
```

## Consulting

If you need additional support or help integrating and/or customizing the controller for your project, feel free to get help@ethn.io.