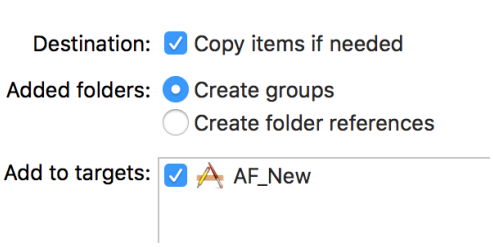
AudioFetch SDK for iOS

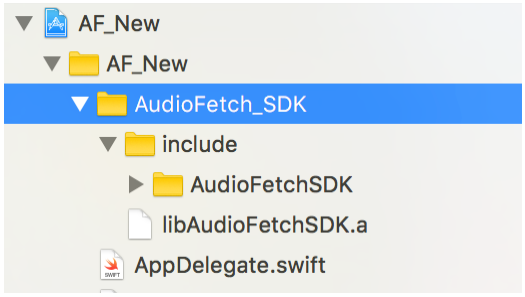
The AudioFetch SDK allows developers to add the AudioFetch system channel discovery and audio management features directly to their third-party native iOS applications.

Documentation Date: June 27, 2016
Documentation Version: 1.0

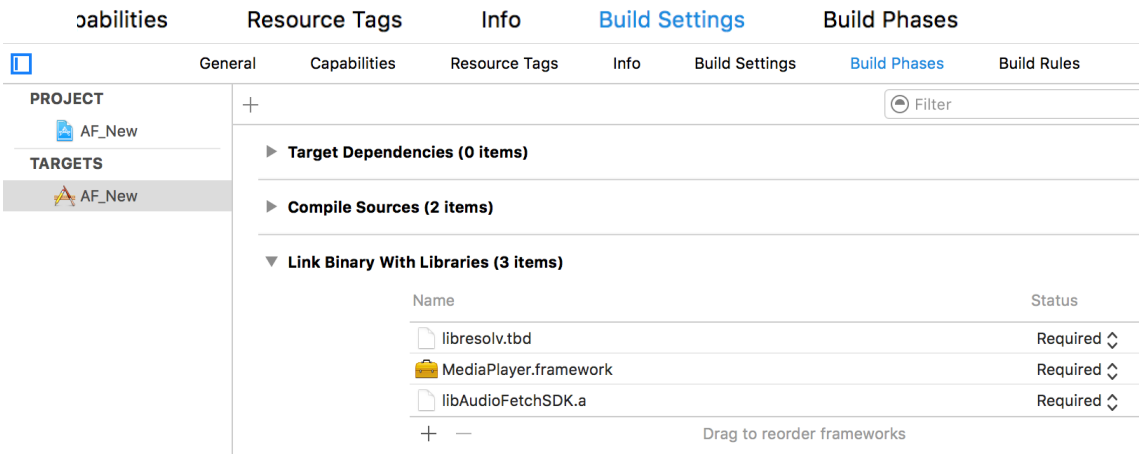
The following is a break down of how to add and integrate the AudioFetch SDK to your iOS App.

1. Add the SDK package to your project
   1.1. Drag the *libAudioFetchSDK.a* and the *include* folder into your project's navigator window. Make sure you check *Copy items if needed.* Your project should look similar to the screen shot on the right.



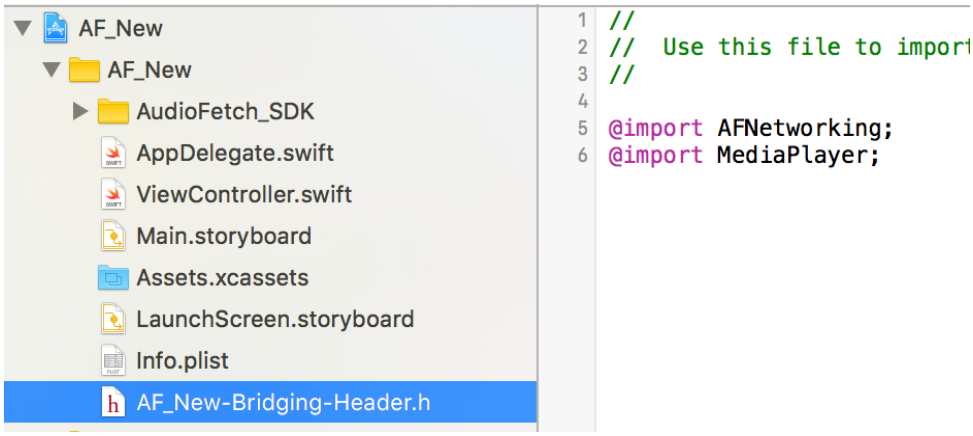


   1.2. Go to your App's Target *Build Settings* tab and search for *Other Linker Flags.* Add *-ObjC* and *-lstdc++.*
   1.3. Go to your App's Target *Build Phases* tab and expand *Link Binary with Libraries*. Add *MediaPlayer.framework* and *libresolv.tbd*. Also Add *AFNetworking* by adding the library manually to the project like you did our SDK or by using CocoaPods. Here's a link



describing how to add using CocoaPods https://cocoapods.org/pods/AFNetworking.

1.4. For a Swift Project, create a Bridging Header and import both MediaPlayer and AFNetworking. Following this step, your project should now be able to compile without errors.



2. Integrate Audio Manager with your ViewController

2.1. Add Import references to the four header files in the AudioFetch SDK to your Bridge Header.

```
//

@import AFNetworking;
@import MediaPlayer;

#import "AudioManager.h"
#import "Apb.h"
#import "Channel.h"
#import "Notifications.h"
```

2.2. Create your Audio Manager class

```
class ViewController: UIViewController {

    // MARK: CREATE AUDIO MANAGER
    private lazy var audioMgr = AudioManager.sharedInstance()
```

2.3. Create a method for setting the current channel and a method for setting the current volume. The current channel method should first check whether that channel exists using the *hasChannel* property and if it does set the current channel using the *currentChannel* property. The current volume can be set using the *volume* property

```
/**
 Sets the volume on the AudioManager
 */
// MARK: SET THE VOLUME
func setVolume(volume : Float) {
    audioMgr.volume = volume
}

/**
 Sets the current chanel etc
 */
// MARK: SET THE CURRENT CHANNEL
func setCurrentChannel(channel : Int) {
    let curCnl = UInt(channel)

    if audioMgr.hasChannel(curCnl) {
        audioMgr.currentChannel = curCnl
    }
}
```

2.4. Call the new *setCurrentChannel* and *setVolume* methods in the *viewDidLoad* method.

```
override func viewDidLoad() {
    super.viewDidLoad()
    // Do any additional setup after loading the view, typically from a nib.

    setCurrentChannel(0)

    setVolume(0.50)
}
```

2.5. Build and Run your app to the simulator or device. Provided you have audio input plugged into the first (number 1) channel spot on the AudioFetch hardware device, your app should now be playing the audio of the first channel at 50% iOS device volume.