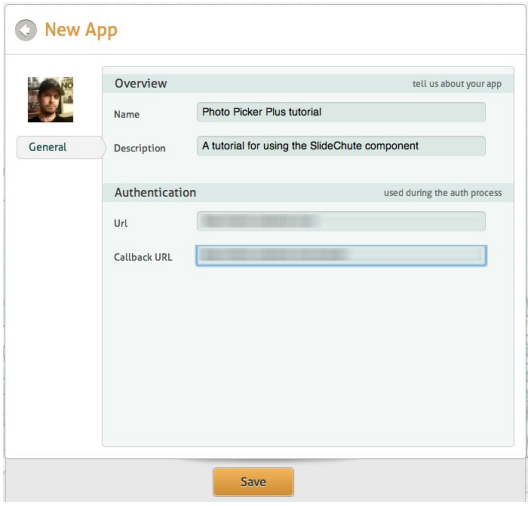
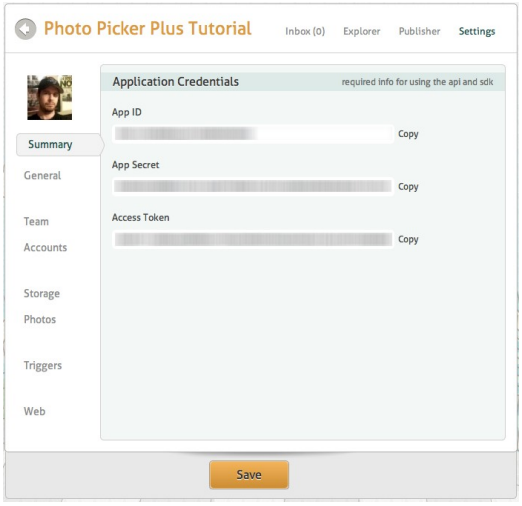
Photo Picker+

This tutorial will show you how to use the Photo Picker Plus component to present the user with a multi-service photo picker. This tutorial was written using version 5.0 of the iOS SDK and version 4.2 of Xcode. Uses Chute SDK version 1.120206 or newer (the version number can be found in the GCConstants.h file). Some changes may need to be made for other software versions.

Preparation

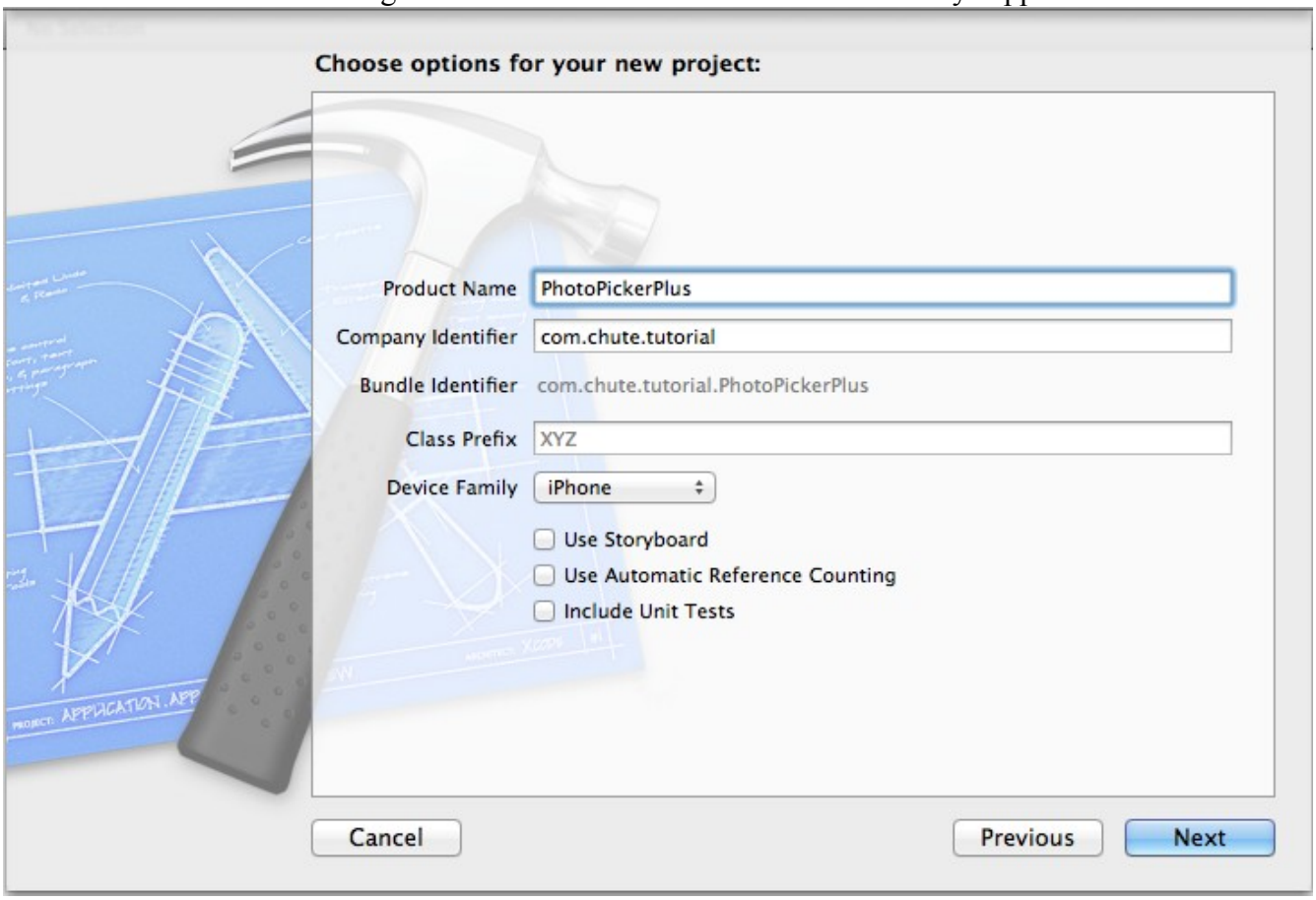
1. Download the Chute SDK from https://github.com/chute/Chute-SDK
2. Download the PhotoPickerPlus component from https://github.com/chute/photo-picker-plus/tree/master/iOS/PhotoPickerPlus
3. Create a Chute developer account and register your app with Chute at http://apps.getchute.com/





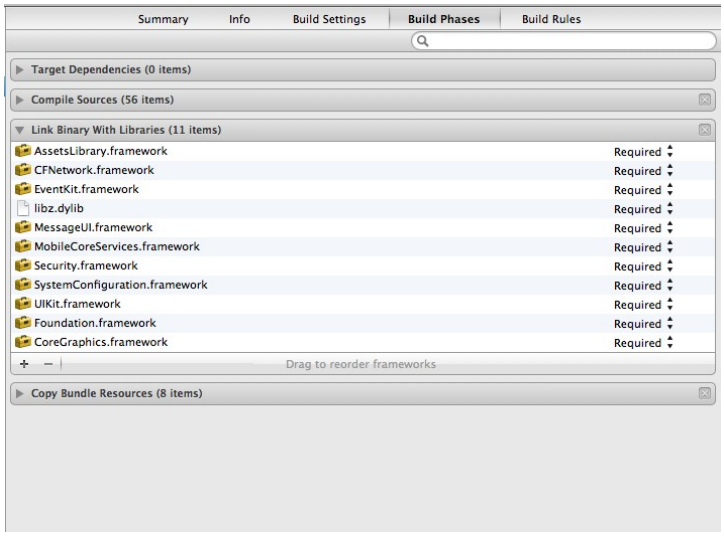
Create A New Project

Start by creating a new Xcode project. A single view application will be easiest to modify for this tutorial. You can choose whatever name you like, I'll call it PhotoPickerPlus. Be sure that "Use Automatic Reference Counting" is unchecked as the SDK does not currently support ARC.



Add The SDK And Component And Link Dependancies

1. Add the SDK to the project
2. Add the picker component
3. Link the required libraries
    - AssetsLibrary
    - CFNetwork
    - EventKit
    - libz.dylib
    - MessageUI
    - MobileCoreServices
    - Security
    - SystemConfiguration



At this point you may want to try running the project to make sure that everything is added ok. You will get a few warnings, but if there are no errors then everything should be correctly added and linked.

Edit Your App ID And Secret

The next step is to enter your chute client information in the GCConstants.h file. This file can be found in SDK/Classes/Core directory. You will need to fill in your APP ID and APP secret from the settings tab of your admin panel. You will also need to adjust the redirect URL to match the callback url from the admin panel. Then set the redirect relative url to everything after the base in the callback url.

Set ViewController As Delegate And Add Objects/Methods

In your viewController.h file import PhotoPickerPlus.h and set up the class as a photoPickerPlusDelegate. Then add an object for the image view and a method for pressing the pick photo button. This should look similar to this

viewController.h

```
#import <UIKit/UIKit.h>
#import "PhotoPickerPlus.h"

@interface ViewController : UIViewController <photoPickerPlusDelegate>

@property (nonatomic, readonly) IBOutlet UIImageView *imageView;

-(IBAction)pickPhotoSelected:(id)sender;

@end
```

Synthesize ImageView And Write The Display Method

In viewController.m you now need to synthesize your imageView object and write the method to display the photo picker plus component. The method will initialize the controller, set itself as the delegate, present it and release it. We also want to have out current view visible behind the first screen

of the picker so we will add a line for that as well. The code for all this is

viewController.m

```
@synthesize imageView;

-(IBAction)pickPhotoSelected:(id)sender{
    PhotoPickerPlus *temp = [[PhotoPickerPlus alloc] init];
    [temp setDelegate:self];
    [self setModalPresentationStyle:UIModalPresentationCurrentContext];
    [self presentViewController:temp animated:YES completion:^(void){
        [temp release];
    }];
}
```

Write The Delegate Methods

The photoPickerPlusDelegate methods are photoPickerPlusControllerDidCancel: and photoPickerPlusController:didFinishPickingMediaWithInfo:. These work exactly the same as the UIImagePickerController delegate methods to make things easier. You can refer to Apple's documentation on UIImagePickerControllerDelegate to see what the keys in the dictionary are. The only one we are going to be concerned with is UIImagePickerControllerOriginalImage. So our cancel method will just dismiss the picker and our success method will display the image in the imageView and dismiss the picker. The code for these methods is

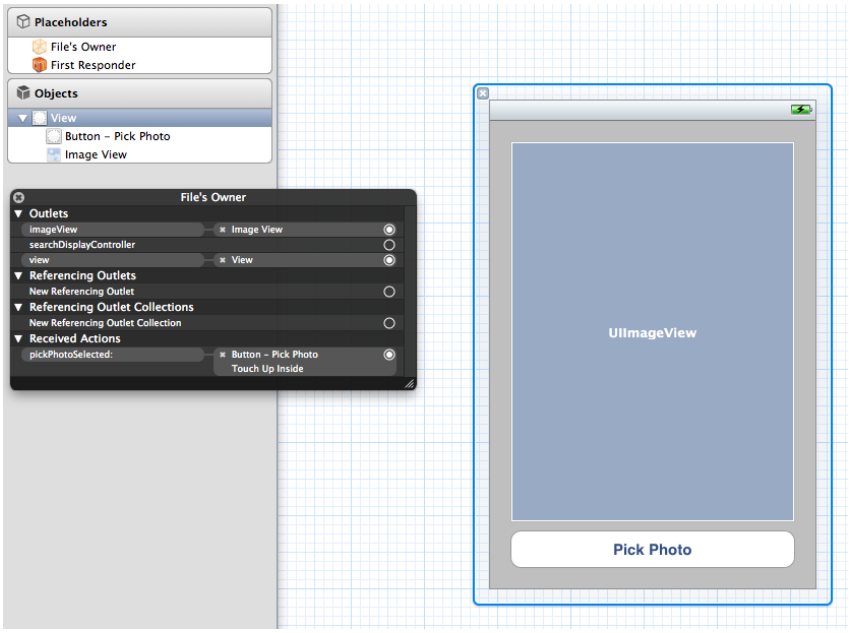
viewController.m

```
-(void) photoPickerPlusControllerDidCancel:(PhotoPickerPlus *)picker{
    [self dismissViewControllerAnimated:YES completion:^(void){

    }];
}
-(void) photoPickerPlusController:(PhotoPickerPlus *)picker didFinishPickingMediaWithInfo:
(NSDictionary *)info{
    [self dismissViewControllerAnimated:YES completion:^(void){
        [[self imageView] setImage:[info objectForKey:UIImagePickerControllerOriginalImage]];
    }];
}
```

Create The UI

Open viewController.xib and add an UIImageView covering most of the view and a UIButton below it. Hook the UIImageView up to the imageView object and hook up the pickPhotoSelected method to the touchUpInside event of the button. You can also title the button whatever you want. I called mine pick photo. You also probably want to set the mode for the UIImageView to aspectFit.



Conclusion

You should have a fully working app now that allows you to take a photo, pick an image from the device, or pick an image from a variety of online sources. Due to the picker accessing the ALAssets it will present the user with a dialog asking if they want to allow location services. This is due to ALAssets having location data associated with the images. If the user declines then the picker will not allow them to choose images on the device.