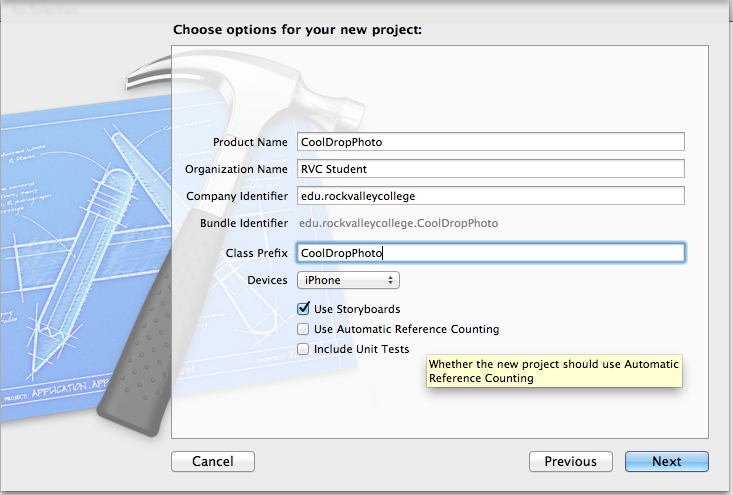
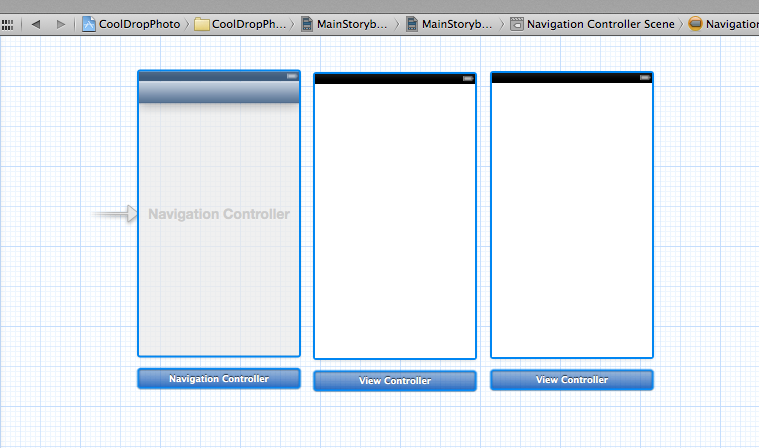
CoolDropPhoto

This app uses the camera, saves a temp photo, and then uploads to dropbox.

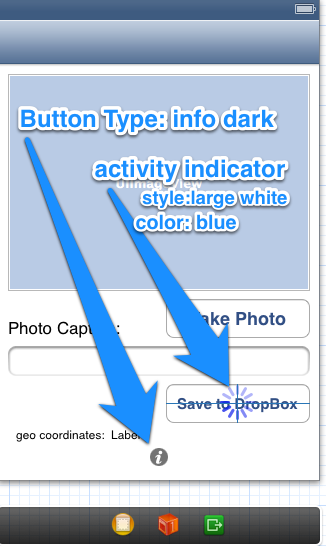
1. Create a new project > single View > storyboard



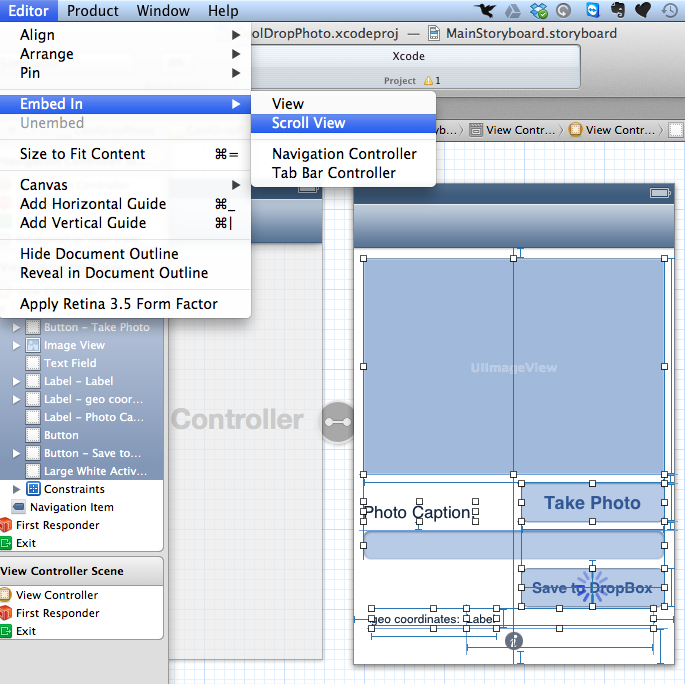
1. Delete ViewController and add a Navigation Controller
2. Delete TableViee
3. Add 2 ViewControllers



1. Ctrl+Drag Naviagation Controller to middle View Controller and select “root view controller”
2. Make sure class CoolDropPhotoViewController is bound to middle view controller.
3. On the middle ViewContoller add:
   1. UIImage
   2. 3 UILabela
   3. UITextfield
   4. 3 UIButton -



1. Ctrl + Drag dark info button to right View Controller and select PUSH segue
2. Select all objects > Editor > Embed In > Scroll View



1. Ctrl + Drag all objects to DropboxPhotoSaveViewController.h. file (except labels)

@property (retain, nonatomic) IBOutlet UIImageView \*Photo;

@property (retain, nonatomic) IBOutlet UITextField \*Name;

@property (retain, nonatomic) IBOutlet UIScrollView \*scrollview;

@property (retain, nonatomic) IBOutlet UIActivityIndicatorView \*activity;

- (IBAction)btnPhoto:(id)sender;

- (IBAction)btnSave:(id)sender;

-(IBAction) doneEditing:(id) sender;

1. DropboxPhotoSaveViewController.m File

#import "DropboxPhotoSaveViewController.h"

@implementation DropboxPhotoSaveViewController

@synthesize Photo;

@synthesize Name;

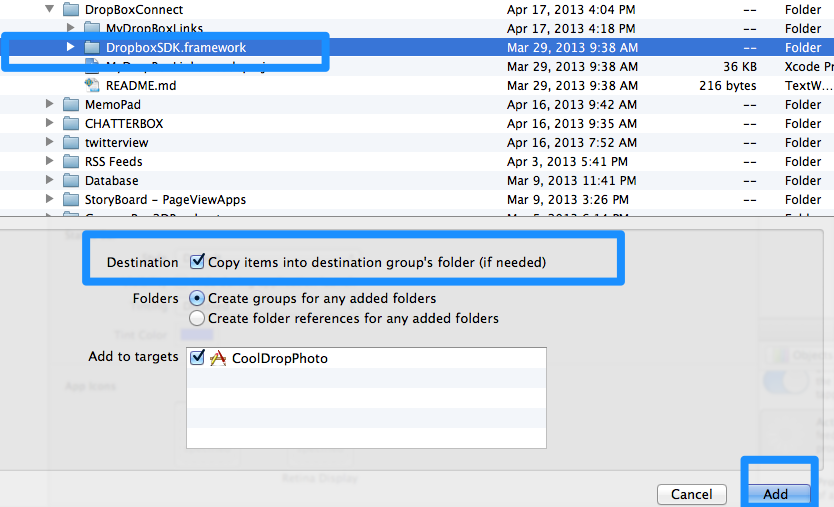
@synthesize scrollview;

@synthesize activity;

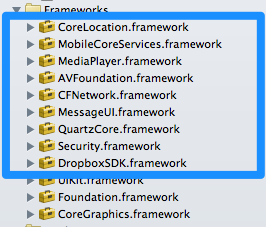
1. Bind outlets and inlets to storyboard objects
   1. UITextField
      1. “did end of exit” - **doneediting**
      2. delegate - viewcontroller
      3. outlet – Name
   2. UIScrollview
      1. outlet – scrollView
   3. UiButtons
      1. outlet – btnPhoto
      2. outlet – btnSave
   4. UIImageView
      1. outlet – Photo
   5. UIActivityIndicator
      1. outlet - activity
2. Enter Title: Cool DropPhoto on both views



1. Insert Dropbox SDK folder



1. Add Needed Libraries. Move into **Frameworks** folder when finished



1. Add Code
2. Create a group named “gpscoordinates”
3. Add New Objective C Class “GPSViewController” subclass of NSObjects
4. GPSViewController.h

#import <CoreLocation/CoreLocation.h>

#import <UIKit/UIKit.h>

@protocol CoreLocationControllerDelegate

@required

- (void)locationUpdate:(CLLocation \*)location; // Our location updates are sent here

- (void)locationError:(NSError \*)error; // Any errors are sent here

@end

@interface GPSViewController : NSObject <CLLocationManagerDelegate> {

CLLocationManager \*locMgr;

id delegate;

}

@property (nonatomic, retain) CLLocationManager \*locMgr;

@property (nonatomic, assign) id delegate;

@end

1. GPSViewController.m (replace with below)

#import "GPSViewController.h"

@implementation GPSViewController

@synthesize locMgr, delegate;

- (id)init {

self = [super init];

if(self != nil) {

self.locMgr = [[[CLLocationManager alloc] init] autorelease]; // Create new instance of locMgr

self.locMgr.delegate = self; // Set the delegate as self.

}

return self;

}

- (void)locationManager:(CLLocationManager \*)manager didUpdateToLocation:(CLLocation \*)newLocation fromLocation:(CLLocation \*)oldLocation {

if([self.delegate conformsToProtocol:@protocol(CoreLocationControllerDelegate)]) { // Check if the class assigning itself as the delegate conforms to our protocol. If not, the message will go nowhere. Not good.

[self.delegate locationUpdate:newLocation];

}

}

- (void)locationManager:(CLLocationManager \*)manager didFailWithError:(NSError \*)error {

if([self.delegate conformsToProtocol:@protocol(CoreLocationControllerDelegate)]) { // Check if the class assigning itself as the delegate conforms to our protocol. If not, the message will go nowhere. Not good.

[self.delegate locationError:error];

}

}

- (void)dealloc {

[self.locMgr release];

[super dealloc];

}

@end

1. DropboxPhotoSaveAppDelegate.m (Replace with below)

#import "DropboxPhotoSaveAppDelegate.h"

#import <DropboxSDK/DropboxSDK.h>

@implementation DropboxPhotoSaveAppDelegate

- (void)dealloc

{

[\_window release];

[super dealloc];

}

- (BOOL)application:(UIApplication \*)application handleOpenURL:(NSURL \*)url {

if ([[DBSession sharedSession] handleOpenURL:url]) {

if ([[DBSession sharedSession] isLinked]) {

NSLog(@"App linked successfully!");

// At this point you can start making API calls

}

return YES;

}

// Add whatever other url handling code your app requires here

return NO;

}

- (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions

{

// Override point for customization after application launch.

DBSession\* dbSession =

[[[DBSession alloc]

initWithAppKey:@"wchopzluxzrwfng"

appSecret:@"slu6hhnzpcofuff"

root:kDBRootAppFolder] // either kDBRootAppFolder or kDBRootDropbox

autorelease];

[DBSession setSharedSession:dbSession];

return YES;

}

- (void)applicationWillResignActive:(UIApplication \*)application

{

// Sent when the application is about to move from active to inactive state. This can occur for certain types of temporary interruptions (such as an incoming phone call or SMS message) or when the user quits the application and it begins the transition to the background state.

// Use this method to pause ongoing tasks, disable timers, and throttle down OpenGL ES frame rates. Games should use this method to pause the game.

}

- (void)applicationDidEnterBackground:(UIApplication \*)application

{

// Use this method to release shared resources, save user data, invalidate timers, and store enough application state information to restore your application to its current state in case it is terminated later.

// If your application supports background execution, this method is called instead of applicationWillTerminate: when the user quits.

}

- (void)applicationWillEnterForeground:(UIApplication \*)application

{

// Called as part of the transition from the background to the inactive state; here you can undo many of the changes made on entering the background.

}

- (void)applicationDidBecomeActive:(UIApplication \*)application

{

// Restart any tasks that were paused (or not yet started) while the application was inactive. If the application was previously in the background, optionally refresh the user interface.

}

- (void)applicationWillTerminate:(UIApplication \*)application

{

// Called when the application is about to terminate. Save data if appropriate. See also applicationDidEnterBackground:.

}

@end

1. DropboxPhotoSaveViewController.h

#import <DropboxSDK/DropboxSDK.h>

#import <UIKit/UIKit.h>

#import <MediaPlayer/MediaPlayer.h>

#import <MobileCoreServices/MobileCoreServices.h>

#import <CoreLocation/CoreLocation.h>

#import "GPSViewController.h"

@interface DropboxPhotoSaveViewController : UIViewController <UINavigationControllerDelegate, UIImagePickerControllerDelegate,CoreLocationControllerDelegate>

{

UIImagePickerController \*imagePicker;

DBRestClient \*restClient;

NSMutableArray \*dropboxURLs;

GPSViewController \*CLController;

IBOutlet UILabel \*locLabel;

}

@property (retain, nonatomic) IBOutlet UIImageView \*Photo;

@property (retain, nonatomic) IBOutlet UITextField \*Name;

@property (retain, nonatomic) IBOutlet UIScrollView \*scrollview;

@property (retain, nonatomic) IBOutlet UIActivityIndicatorView \*activity;

- (IBAction)btnPhoto:(id)sender;

- (IBAction)btnSave:(id)sender;

-(IBAction) doneEditing:(id) sender;

@end

1. DropboxPhotoSaveViewController.m

#import "DropboxPhotoSaveViewController.h"

@implementation DropboxPhotoSaveViewController

@synthesize Photo;

@synthesize Name;

@synthesize scrollview;

@synthesize activity;

NSString \*files;

NSString \*gps;

- (void)viewDidLoad

{

[super viewDidLoad];

[activity stopAnimating];

imagePicker = [[UIImagePickerController alloc] init];

dropboxURLs = [[NSMutableArray alloc] init];

[self didPressLink];

CLController = [[GPSViewController alloc] init];

CLController.delegate = self;

[CLController.locMgr startUpdatingLocation];

// Do any additional setup after loading the view, typically from a nib.

// Do any additional setup after loading the view.

UITapGestureRecognizer \*tap = [[UITapGestureRecognizer alloc]

initWithTarget:self

action:@selector(dismissKeyboard)];

[self.view addGestureRecognizer:tap];

}

-(void)dismissKeyboard {

[Name resignFirstResponder];

}

-(IBAction) doneEditing:(id) sender {

[sender resignFirstResponder];

}

- (void)didPressLink {

if (![[DBSession sharedSession] isLinked]) {

[[DBSession sharedSession] linkFromController:self];

}

}

- (void)didReceiveMemoryWarning

{

[super didReceiveMemoryWarning];

// Dispose of any resources that can be recreated.

}

- (IBAction)btnPhoto:(id)sender {

imagePicker.delegate = self;

imagePicker.sourceType = UIImagePickerControllerSourceTypeCamera;

imagePicker.cameraDevice = UIImagePickerControllerCameraDeviceFront;

NSArray \*mediaTypes =

[NSArray arrayWithObjects:kUTTypeImage, nil];

imagePicker.mediaTypes = mediaTypes;

imagePicker.cameraCaptureMode = UIImagePickerControllerCameraCaptureModePhoto;

imagePicker.allowsEditing = YES;

//—--show the Image Picker—--

[self presentModalViewController:imagePicker animated:YES];

}

- (void)dealloc {

[CLController release];

[Photo release];

[Name release];

[scrollview release];

[activity release];

[super dealloc];

}

- (IBAction)btnView:(id)sender {

}

- (IBAction)btnSave:(id)sender {

[self didPressLink];

[self dismissKeyboard];

// NSString \*localPath = [[NSBundle mainBundle] pathForResource:@"Info" ofType:@"plist"];

// NSString \*filename = @"Info.plist";

// NSString \*destDir = @"/";

[activity startAnimating];

NSArray \*paths =

NSSearchPathForDirectoriesInDomains(

NSDocumentDirectory, NSUserDomainMask, YES);

NSString \*documentsDir = [paths objectAtIndex:0];

CFGregorianDate currentDate = CFAbsoluteTimeGetGregorianDate(CFAbsoluteTimeGetCurrent(), CFTimeZoneCopySystem());

NSString \*strDate = [NSString stringWithFormat:@"%02d%02d%02d:%02d:%2.0f", currentDate.month,currentDate.day,currentDate.hour, currentDate.minute, currentDate.second];

NSString \*oldfilename = @"MyPicture.png";

NSString \*filename = [NSString stringWithFormat:@"%@-%@-%@.%@",

strDate,gps,Name.text,@"png"];

NSString \*localPath = [documentsDir stringByAppendingPathComponent:oldfilename];

NSString \*destDir = @"/";

NSLog(@"FilePath is: %@", localPath);

[[self restClient] uploadFile:filename toPath:destDir

fromPath:localPath];

[[self restClient] loadMetadata:@"/"];

}

- (NSString \*) filePath: (NSString \*) fileName {

NSArray \*paths =

NSSearchPathForDirectoriesInDomains(

NSDocumentDirectory, NSUserDomainMask, YES);

NSString \*documentsDir = [paths objectAtIndex:0];

return [documentsDir stringByAppendingPathComponent:fileName];

}

- (void) saveImage{

//—-get the date from the ImageView—-

NSData \*imageData =

[NSData dataWithData:UIImagePNGRepresentation(Photo.image)];

//—-write the date to file—-

[imageData writeToFile:[self filePath:@"MyPicture.png"] atomically:YES];

}

- (void)imagePickerController:(UIImagePickerController \*)picker

didFinishPickingMediaWithInfo:(NSDictionary \*)info {

UIImage \*image;

NSURL \*mediaUrl;

mediaUrl = (NSURL \*)[info valueForKey:

UIImagePickerControllerMediaURL];

if (mediaUrl == nil) {

image = (UIImage \*) [info valueForKey:

UIImagePickerControllerEditedImage];

if (image == nil) {

//-—-original image selected—--

image = (UIImage \*)

[info valueForKey:UIImagePickerControllerOriginalImage];

//—--display the image—--

Photo.image = image;

}

else { //—--edited image picked—-

//—--get the cropping rectangle applied to the image—--

CGRect rect =

[[info valueForKey:UIImagePickerControllerCropRect]

CGRectValue];

//—--display the image—--

Photo.image = image;

}

//—-save the image captured—-

[self saveImage];

}

//—--hide the Image Picker—--

[picker dismissModalViewControllerAnimated:YES];

}

- (void)imagePickerControllerDidCancel:(UIImagePickerController \*)picker {

//-—-user did not select image; hide the Image Picker—--

[picker dismissModalViewControllerAnimated:YES];

}

- (void)viewDidUnload

{

[super viewDidUnload];

// Release any retained subviews of the main view.

// e.g. self.myOutlet = nil;

}

- (void)viewWillAppear:(BOOL)animated

{

[super viewWillAppear:animated];

}

- (void)viewDidAppear:(BOOL)animated

{

[super viewDidAppear:animated];

}

- (void)viewWillDisappear:(BOOL)animated

{

[super viewWillDisappear:animated];

}

- (void)viewDidDisappear:(BOOL)animated

{

[super viewDidDisappear:animated];

}

- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation

{

// Return YES for supported orientations

return (interfaceOrientation != UIInterfaceOrientationPortraitUpsideDown);

}

- (DBRestClient \*)restClient {

if (!restClient) {

restClient =

[[DBRestClient alloc] initWithSession:[DBSession sharedSession]];

restClient.delegate = self;

}

return restClient;

}

- (void)restClient:(DBRestClient \*)client loadedMetadata:(DBMetadata \*)metadata {

if (metadata.isDirectory) {

[dropboxURLs removeAllObjects];

NSLog(@"Folder '%@' contains:", metadata.path);

for (DBMetadata \*file in metadata.contents) {

NSLog(@"\t%@", file.filename);

[dropboxURLs addObject:file.filename];

}

}

}

- (void)restClient:(DBRestClient \*)client

loadMetadataFailedWithError:(NSError \*)error {

NSLog(@"Error loading metadata: %@", error);

}

- (void)restClient:(DBRestClient\*)client uploadedFile:(NSString\*)destPath

from:(NSString\*)srcPath metadata:(DBMetadata\*)metadata {

NSLog(@"File uploaded successfully to path: %@", metadata.path);

[activity stopAnimating];

UIAlertView \*alert = [[UIAlertView alloc] initWithTitle: @"Success!"

message: @"Picture Uploaded to Dropbox Successful!"

delegate: nil

cancelButtonTitle:@"OK"

otherButtonTitles:nil];

[alert show];

}

- (void)restClient:(DBRestClient\*)client uploadFileFailedWithError:(NSError\*)error {

NSLog(@"File upload failed with error - %@", error);

}

- (void)restClient:(DBRestClient\*)client loadedFile:(NSString\*)localPath {

NSLog(@"File loaded into path: %@", localPath);

}

//gps coordinates

- (void)locationUpdate:(CLLocation \*)location {

//Get nearby address

NSString \* latitude = [[[NSString alloc] initWithFormat:@"%f", location.coordinate.latitude]autorelease];

NSString \* longitude = [[[NSString alloc] initWithFormat:@"%f", location.coordinate.longitude]autorelease];

NSLog(@"latitude:%@",latitude);

NSLog(@"longitude:%@",longitude);

locLabel.text = [NSString stringWithFormat:@"%@,%@",

latitude, longitude];

gps=locLabel.text;

NSLog(@"location: %@", locLabel.text);

}

- (void)locationError:(NSError \*)error {

locLabel.text = [error description];

NSLog(@"location Error: %@", locLabel.text);

}

- (void)textFieldDidBeginEditing:(UITextField \*)textField {

CGPoint scrollPoint = CGPointMake(0, textField.frame.origin.y);

[scrollview setContentOffset:scrollPoint animated:YES];

}

- (void)textFieldDidEndEditing:(UITextField \*)textField {

[scrollview setContentOffset:CGPointZero animated:YES];

}

- (void)textViewDidBeginEditing:(UITextView \*)textView {

CGPoint scrollPoint = CGPointMake(0, textView.frame.origin.y);

[scrollview setContentOffset:scrollPoint animated:YES];

}

- (void)textViewDidEndEditing:(UITextView \*)textView {

[scrollview setContentOffset:CGPointZero animated:YES];

}

@end

1. On the right view controller add Title and add UITextView
   1. Change text property to **attribute**
   2. **add below text**

Cool PhotoDrop  Cool PhotoDrop allows you to take a photo, caption it, and save to your dropbox account.   The photos can be found in the dropbox folder:  apps/coolphotodrop

1. **Run App. Command + R**
2. **Add icons**

**27) Save As PDF with gps Coordinates**

See <https://github.com/ioscourse/PDF-Creation-from-Photo-and-Caption.git>

**28) Add PopUp Viewer to view or download photos or files**

See <https://github.com/ioscourse/PopUp-Viewer.git>