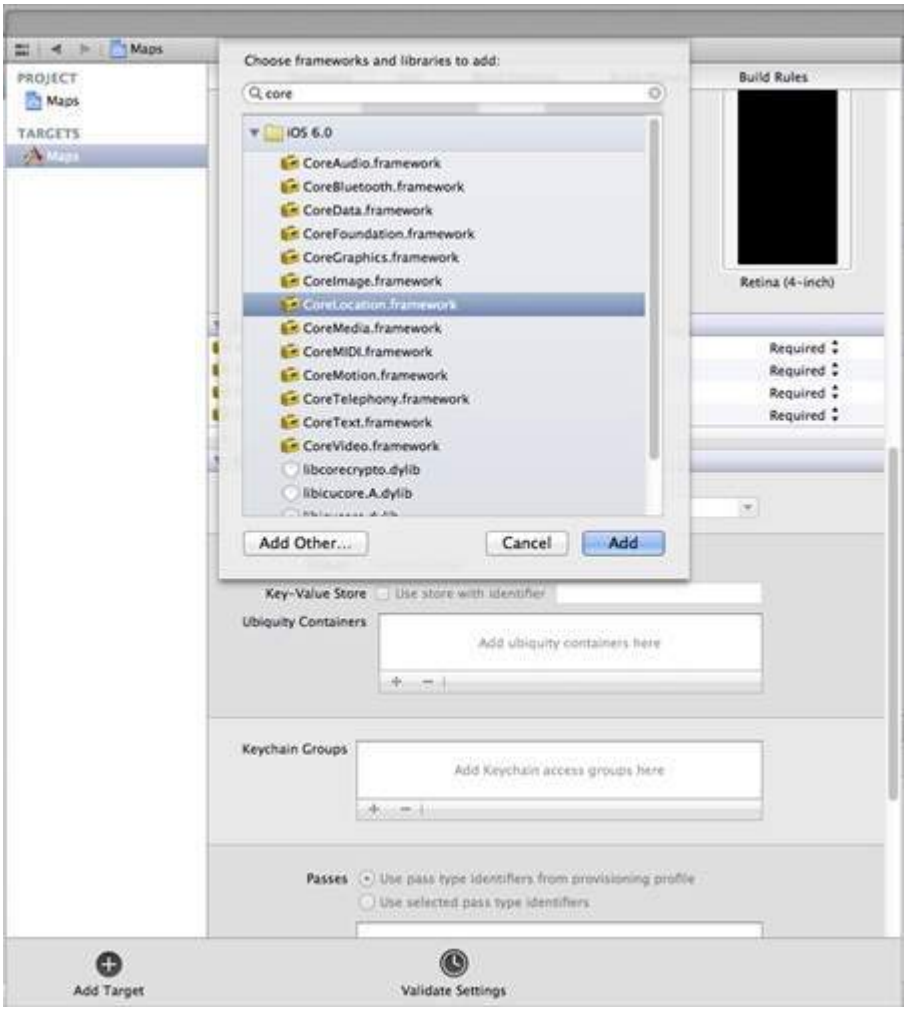
IOS - LOCATION HANDLING

We can easily locate the user's current location in iOS, provided the user allows the application to access the information with the help of the core location framework.

Location Handling – Steps Involved

Step 1. Create a simple View based application.

Step 2. Select your project file, then select targets and then add CoreLocation.framework as shown below −



Step 3. Add two labels in **ViewController.xib** and create ibOutlets naming the labels as **latitudeLabel** and **longitudeLabel** respectively.

Step 4. Create a new file by selecting File-> New -> File... -> select **Objective C class** and click next.

Step 5. Name the class as **LocationHandler** with **"sub class of"** as NSObject.

Step 6. Select create.

Step 7. Update **LocationHandler.h** as follows −

```
#import <Foundation/Foundation.h>
#import <CoreLocation/CoreLocation.h>

@protocol LocationHandlerDelegate <NSObject>

@required
```

```
-(void) didUpdateToLocation:(CLLocation*)newLocation
 fromLocation:(CLLocation*)oldLocation;
@end

@interface LocationHandler : NSObject<CLLocationManagerDelegate>
{
    CLLocationManager *locationManager;
}
@property(nonatomic,strong) id<LocationHandlerDelegate> delegate;

+(id)getSharedInstance;
-(void)startUpdating;
-(void) stopUpdating;

@end
```

Step 8. Update **LocationHandler.m** as follows −

```
#import "LocationHandler.h"
static LocationHandler *DefaultManager = nil;

@interface LocationHandler()

-(void)initiate;

@end

@implementation LocationHandler

+(id)getSharedInstance{
    if (!DefaultManager) {
        DefaultManager = [[self allocWithZone:NULL]init];
        [DefaultManager initiate];
    }
    return DefaultManager;
}
-(void)initiate{
    locationManager = [[CLLocationManager alloc]init];
    locationManager.delegate = self;
}

-(void)startUpdating{
    [locationManager startUpdatingLocation];
}

-(void) stopUpdating{
    [locationManager stopUpdatingLocation];
}
-(void)locationManager:(CLLocationManager *)manager didUpdateToLocation:
 (CLLocation *)newLocation fromLocation:(CLLocation *)oldLocation{
    if ([self.delegate respondsToSelector:@selector
    (didUpdateToLocation:fromLocation:)])
    {
        [self.delegate didUpdateToLocation:oldLocation
        fromLocation:newLocation];

    }
}

@end
```

Step 9. Update **ViewController.h** as follows where we have implemented the **LocationHandler delegate** and create two ibOutlets −

```
#import <UIKit/UIKit.h>
#import "LocationHandler.h"
@interface ViewController : UIViewController<LocationHandlerDelegate>
{
```

```
    IBOutlet UILabel *latitudeLabel;
    IBOutlet UILabel *longitudeLabel;
}
@end
```

Step 10. Update **ViewController.m** as follows −

```
#import "ViewController.h"

@interface ViewController ()

@end

@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
    [[LocationHandler getSharedInstance]setDelegate:self];
    [[LocationHandler getSharedInstance]startUpdating];
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

-(void)didUpdateToLocation:(CLLocation *)newLocation
 fromLocation:(CLLocation *)oldLocation{
    [latitudeLabel setText:[NSString stringWithFormat:
    @"Latitude: %f",newLocation.coordinate.latitude]];
    [longitudeLabel setText:[NSString stringWithFormat:
    @"Longitude: %f",newLocation.coordinate.longitude]];

}

@end
```

Output

When we run the application, we'll get the following output −