

Разработка iOS приложений

Лекция 6.

Своя ячейка UITableViewCell.

MapKit

Три способа сделать свой UITableViewCell

- Программно в коде
 - Через InterfaceBuilder
 - Через Storyboard (в данном цикле не рассматривается)
-
- `>= iOS 6 IB + registerNib`

Programmatic UITableViewCell

Надо создать класс-наследник UITableViewCell

CustomCell.h

```
@interface CustomCell : UITableViewCell

@property ( retain ) UILabel* MyLabel;
@property ( retain ) UITextField* MyTextField;

@end
```

Programmatic UITableViewCell

Надо создать класс-наследник UITableViewCell

CustomCell.m

```
@implementation CustomCell
```

```
@synthesize MyLabel;
```

```
@synthesize MyTextField;
```

```
-( id ) initWithStyle: ( UITableViewCellStyle )style  
    reuseIdentifier: ( NSString* )id  
{  
    self = [ super initWithStyle: style reuseIdentifier: id ];  
    if ( self )  
    {  
        <custom code>  
    }  
    return self;  
}
```

```
@end
```

Programmatic UITableViewCell

Важный метод `layoutSubviews`

CustomCell.m

```
@implementation CustomCell

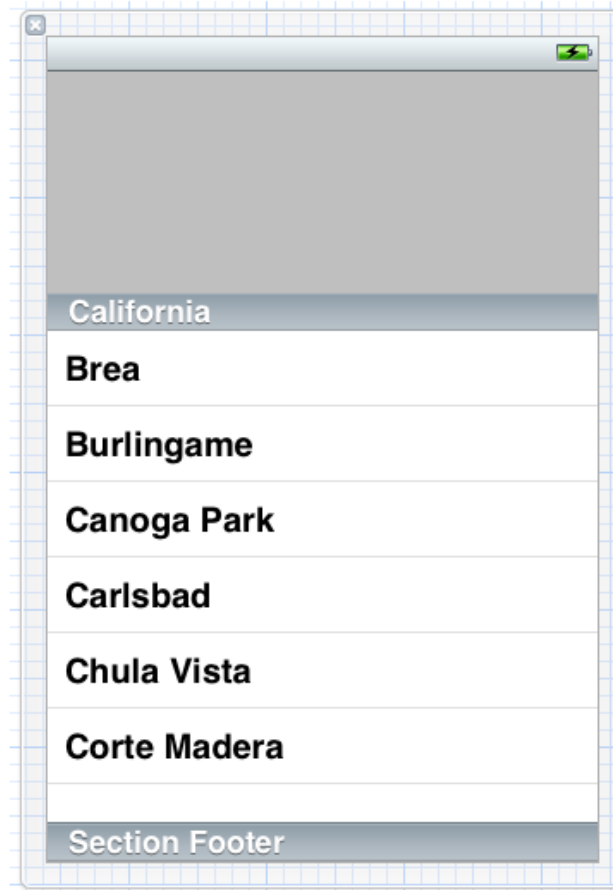
- ( void ) layoutSubviews
{
    [ super layoutSubviews ];

    < custom code >
}

@end
```

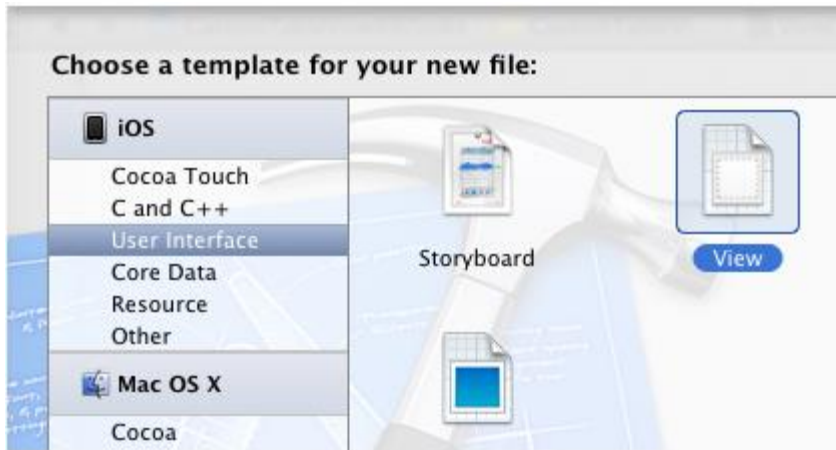
InterfaceBuilt UITableViewCell

Создаем хиб с UITableView (через обычный ViewController!)

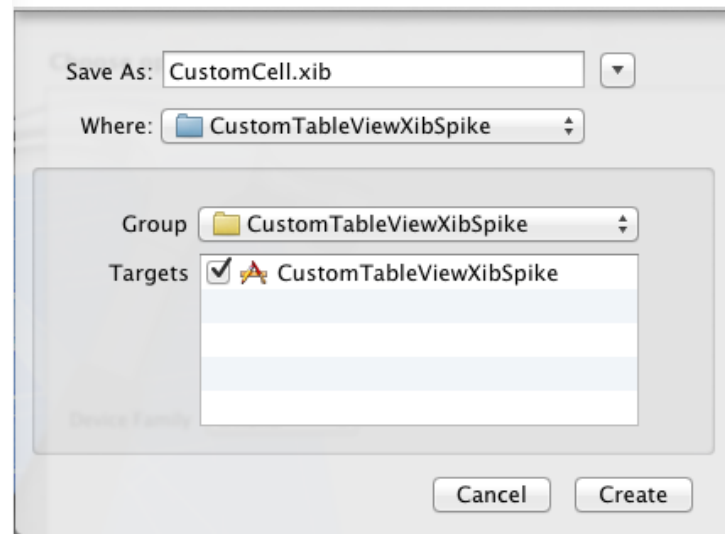


InterfaceBuilt UITableViewCell

Теперь создадим еще один xib для ячейки



И назовем, например, CustomCell



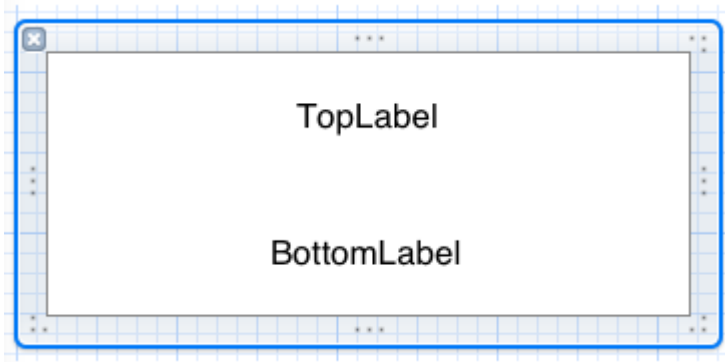
InterfaceBuilt UITableViewCell

Удалим view для нас созданный и перетащим в дизайнер UITableViewCell (из палитры)



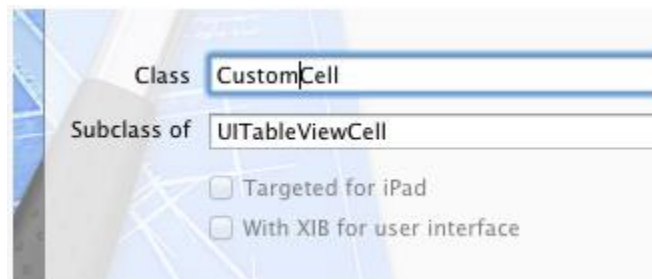
InterfaceBUILT UITableViewCell

Теперь сделаем с ней все что необходимо



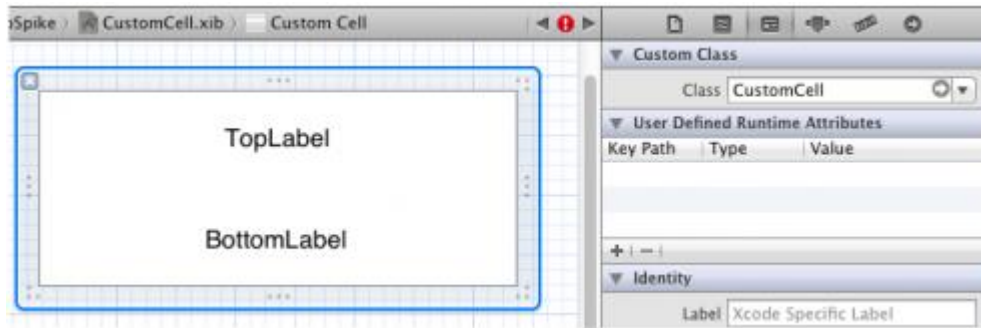
InterfaceBuilt UITableViewCell

Создадим класс для UITableViewCell



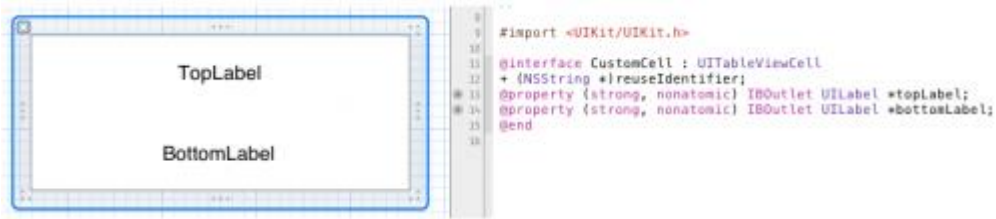
InterfaceBUILT UITableViewCell

Прицепим дизайн к классу



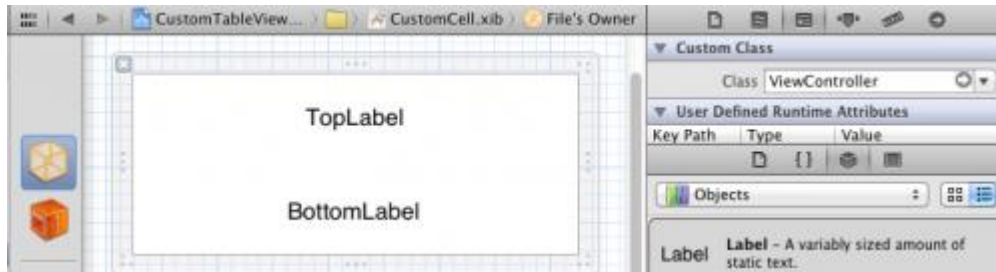
InterfaceBuilt UITableViewCell

Добавим несколько свойств



InterfaceBuilt UITableViewCell

Привяжем к родительскому хіб (с таблицей)



InterfaceBuilt UITableViewCellStyle

В классе родителя создадим специальное свойство

ViewController.h

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

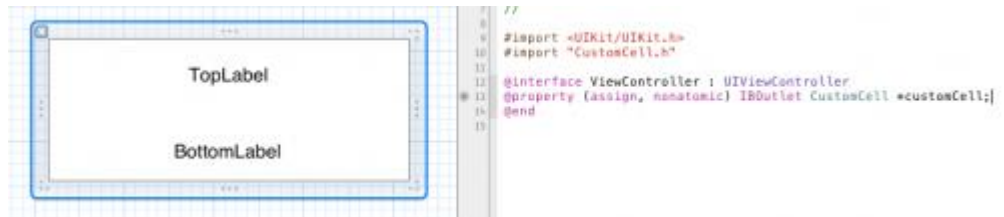
```
@interface ViewController : UIViewController
```

```
@property (assign, nonatomic) IBOutlet CustomCell* customCell;
```

```
@end
```

ViewController.m

```
@synthesize customCell = _customCell;
```



InterfaceBuilt UITableViewCell

Прицепляем созданное свойство к ячейке



InterfaceBuilt UITableViewCell

Создаем (в ViewController'е с таблицей свойства для таблицы)

ViewController.h

```
@property (strong, nonatomic) IBOutlet UITableView* tableView;
```

ViewController.m

```
@synthesize tableView = _tableView;
```


InterfaceBuilt UITableViewCell

Устанавливаем DataSource и Delegate



InterfaceBuilt UITableViewCell

Реализуем нужные методы, особенно

```
- ( NSInteger ) tableView: ( UITableView* ) tableView
    numberOfRowsInSection: ( NSInteger ) section
{ return 1; }

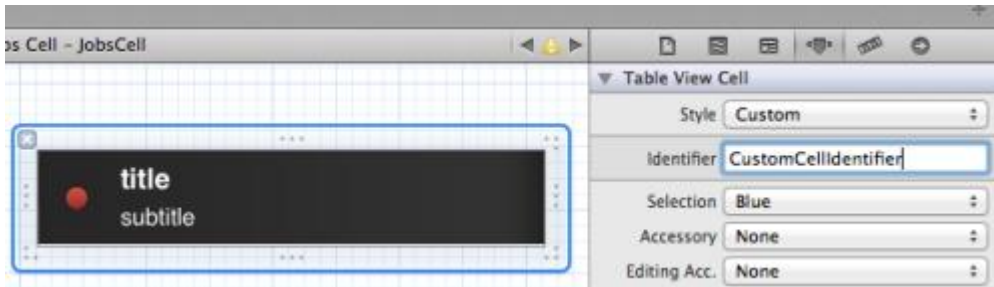
- ( UITableViewCell* ) tableView: ( UITableView* ) tableView
    cellForRowAtIndexPath: ( NSIndexPath* ) indexPath
{
    CustomCell* cell = ( CustomCell* ) [ tableView
dequeueReusableCellWithIdentifier: [CustomCell reuseIdentifier]];
    if ( cell == nil ) {
        [[NSBundle mainBundle] loadNibNamed: @"CustomCell" owner:
self options: nil ];
        cell = _customCell;
        _customCell = nil;
    }

    cell.topLabel.text = @"I am on top";
    cell.bottomLabel.text = @"and I'm on the bottom";

    return cell;
}
```

InterfaceBult UITableViewCell

Не забываем про reuseIdentifier



И теперь можно запускать

CoreLocation & MapKit

CoreLocation – фреймворк для работы с местоположением

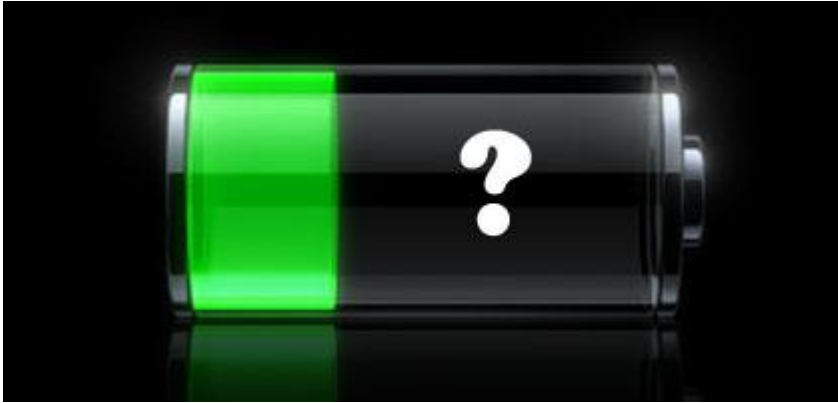
- CLLocationManager
- CLLocation (latitude, longitude, accuracy, ...)

Не содержит GUI

MapKit – фреймворк для работы с картами

CoreLocation

Немного о батарее



Как определяется координата:

- Триангуляция по сотовым вышкам (очень неточно, но батарею не «кушает»)
- Просмотр базы данных WiFi точек (более точно, но и более затратно по батарее)
- GPS (наиболее точно, наиболее затратно)

Общее правило: Чем точнее координата, тем больше тратится заряд батареи

CLLocation

- Координаты
 - `coordinate.latitude` – широта
 - `coordinate.longitude` – долгота
 - `altitude` – высота над уровнем моря
 - `horizontalAccuracy` – точность определения `coordinate`
 - `verticalAccuracy` – точность определения `altitude`
 - `timestamp` – временная метка определения координаты
 - `speed` – скорость
 - `course` – азимут направления

Расстояние между двумя CLLocation

```
- ( CLLocationDistance ) distanceFromLocation: ( CLLocation* ) other
```

Как получить координаты?

Ваш друг – CLLocationManager

Алгоритм:

1. Проверить, поддерживает ли устройство определение координат
2. Создать CLLocationManager и установить delegate для приема координат
3. Настроить менеджер на передачу требуемых данных
4. Начать прием-передачу данных

Просто?

Не тут то было...

Варианты получения местоположения

- Высокоточные регулярные координаты (GPS, например, для навигаторов)
- Получение координат только при значительных изменениях местоположения
- Наблюдение за регионами
- Получение направления движения

Проверка оборудования

CLLocationManager

- +(BOOL) locationServicesEnabled – разрешено ли нам использовать местоположение (см. Настройки)
- +(BOOL) headingAvailable – можем ли мы получать направление (должен быть компас)
- +(BOOL) significantLocationChangeMonitoring – доступно ли получение координат при значительных изменениях (т.е. если у устройства нет GPS)
- +(BOOL) regionMonitoringAvailable – доступно ли наблюдение за регионами
- +(BOOL) regionMonitoringEnabled – разрешено ли наблюдение за регионами

Получение координат

Устанавливаем параметры (CLLocationManager)

- desiredAccuracy
- distanceFilter

Начинаем получать координаты

- -(void) startUpdatingLocation
- -(void) stopUpdatingLocation

Получаем обновления в delegate в функцию

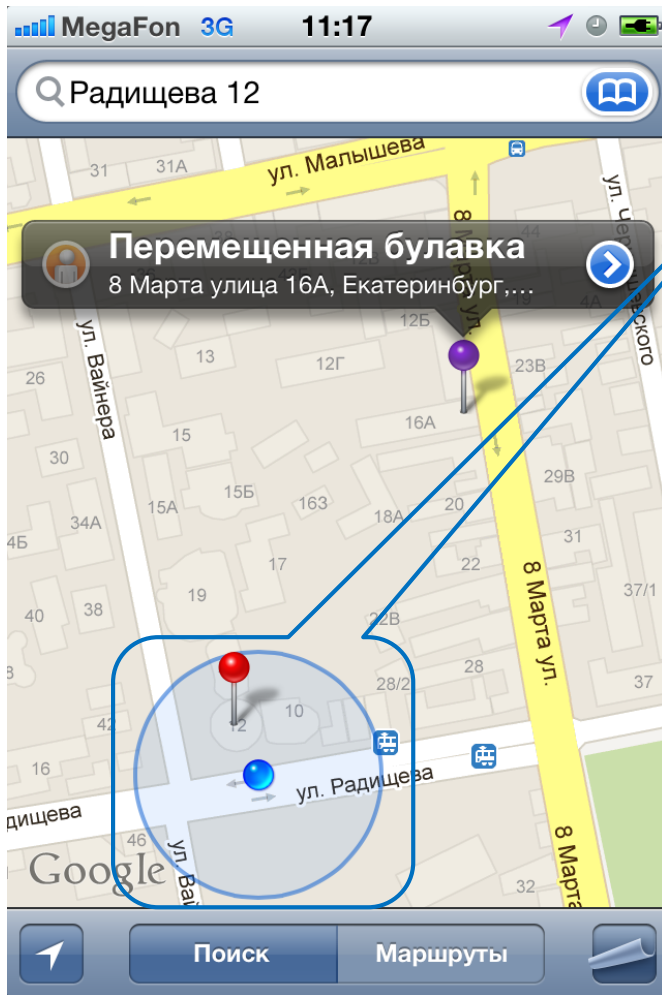
Snow Leopard:

```
-( void ) locationManager: ( CLLocationManager* )manager  
    didUpdateToLocation: ( CLLocation* )newLocation  
        fromLocation: ( CLLocation* )oldLocation;
```

Mavericks, Yosemite

```
-( void ) locationManager: ( CLLocationManager* )manager  
    didUpdateLocations: ( NSArray* )locations;  
{ [ locations lastObject ]; ... }
```

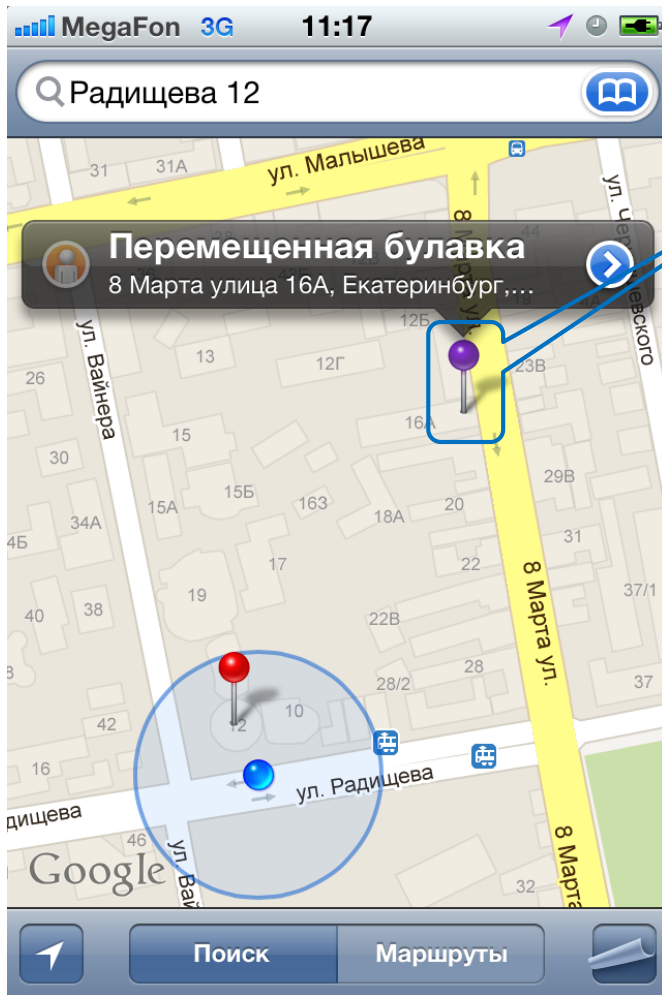
MapKit



Что есть на карте?

Наше местоположение

MapKit



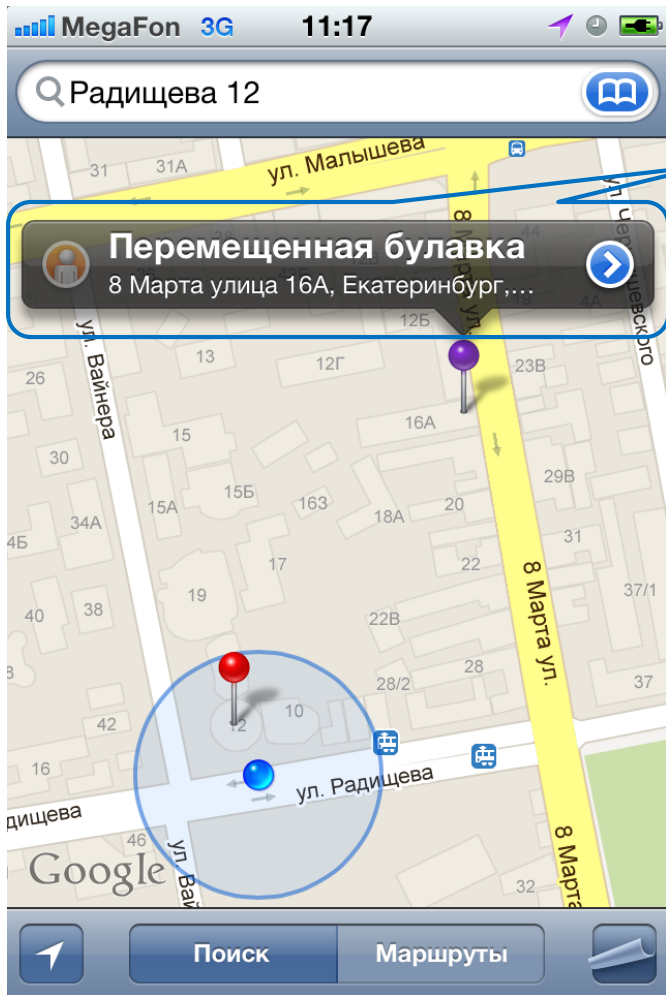
Что есть на карте?

Аннотации

- title
- subtitle

MKAnnotationView
(MKPinAnnotationView)

MapKit



Что есть на карте?

Выноски

MapView

- Показывает карту (схема, спутник, гибрид)
- Показывает набор аннотаций (@property ... annotations)
- Показывает набор оверлеев (@property ... overlays) (не рассматриваем)

MKAnnotation

```
@protocol MKAnnotation
```

```
@property (readonly) CLLocationCoordinate2D coordinate;
```

```
@optional
```

```
@property (readonly) NSString *title;
```

```
@property (readonly) NSString *subtitle;
```

```
@end
```

MapView

Но массив annotations – readonly

Функции для управления аннотациями

- -(void) addAnnotation: (id< MKAnnotation >)annotation;
- -(void) addAnnotations: (NSArray*) annotations;
- -(void) removeAnnotation: (id< MKAnnotation >)annotation;
- -(void) removeAnnotations: (NSArray*) annotations;

По умолчанию все аннотации выглядят как иголки

Но представление можно переопределить ...

MKAnnotationView

... и делается это с помощью MKAnnotationView

```
-( MKAnnotationView* ) mapView: ( MKMapView* )mapView
    viewForAnnotation: ( id< MKAnnotation > ) annotation
{
    MKAnnotationView* view = [ mapView
dequeueReusableAnnotationViewWithIdentifier: IDENTIFIER ];
    if ( nil == view ) {
        view = [ [ MKPinAnnotationView alloc ]
initWithAnnotation: annotation reuseIdentifier: IDENTIFIER ]
autorelease ];
        // view.canShowCallout если необходимо (надо создать view)
    }
    view.annotation = annotation;
    // дополнительные установки
    return view;
}
```


MKMapView

Тип отображения

@property MKMapType mapType

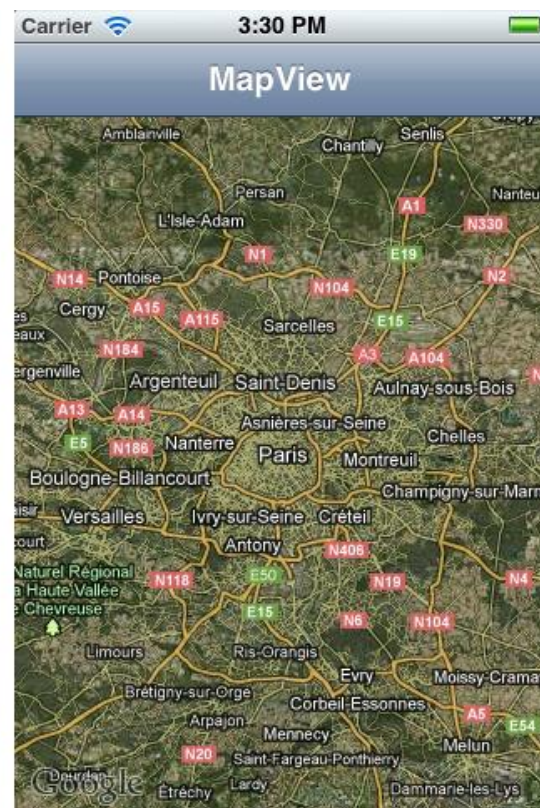
MKMapTypeStandard, MKMapTypeSatellite, MKMapTypeHybrid



Standard



Satellite



Hybrid

MKMapView

Тип отображения

@property MKMapType mapType

MKMapTypeStandard, MKMapTypeSatellite, MKMapTypeHybrid

Отображение текущего местоположения

@property BOOL showUserLocation

Выключение некоторых взаимодействий с пользователем

@property BOOL zoomEnabled

@property BOOL scrollEnabled

Практика

CustomCells

- Для экрана отображения списка элементов Forecast (таблица с Time'ами) сделать кастомную ячейку
 - Чтоб была картинка
 - Чтоб были первые 100 символов из description
 - Чтоб были даты (от, до)
 - Чтоб можно было перейти в экран просмотра отдельной записи

CoreLocation + MapKit

Сделать отдельный проект, в нем

- Отобразить карту
- Отобразить на карте свое местоположение
- Вывести в консоль апдейты местоположения
- Апдейты местоположения выводить на карту в виде MKAnnotation + MKPinAnnotationView