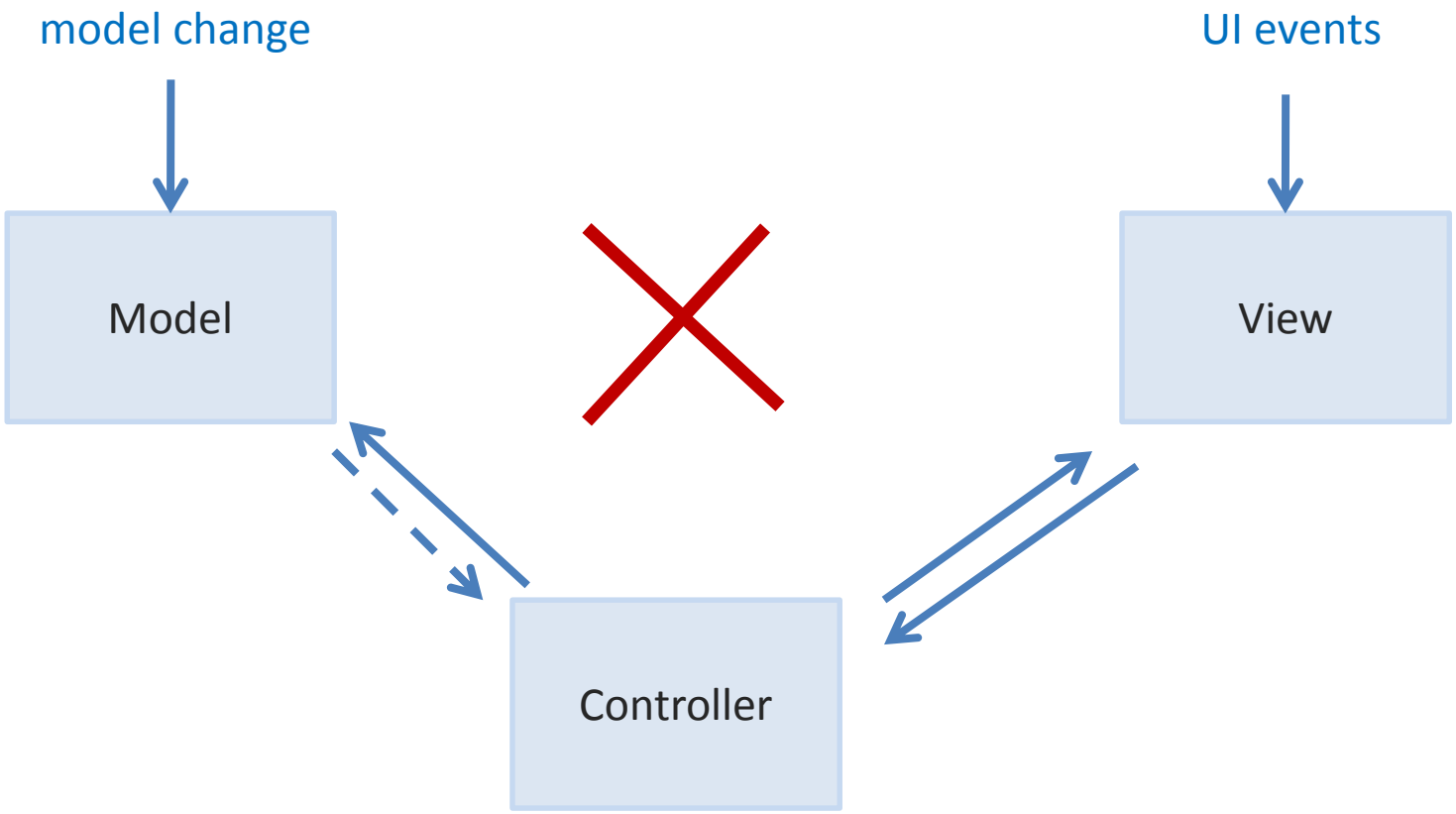
Разработка iOS приложений

Лекция 8.

NSOperation & NSOperationQueue.

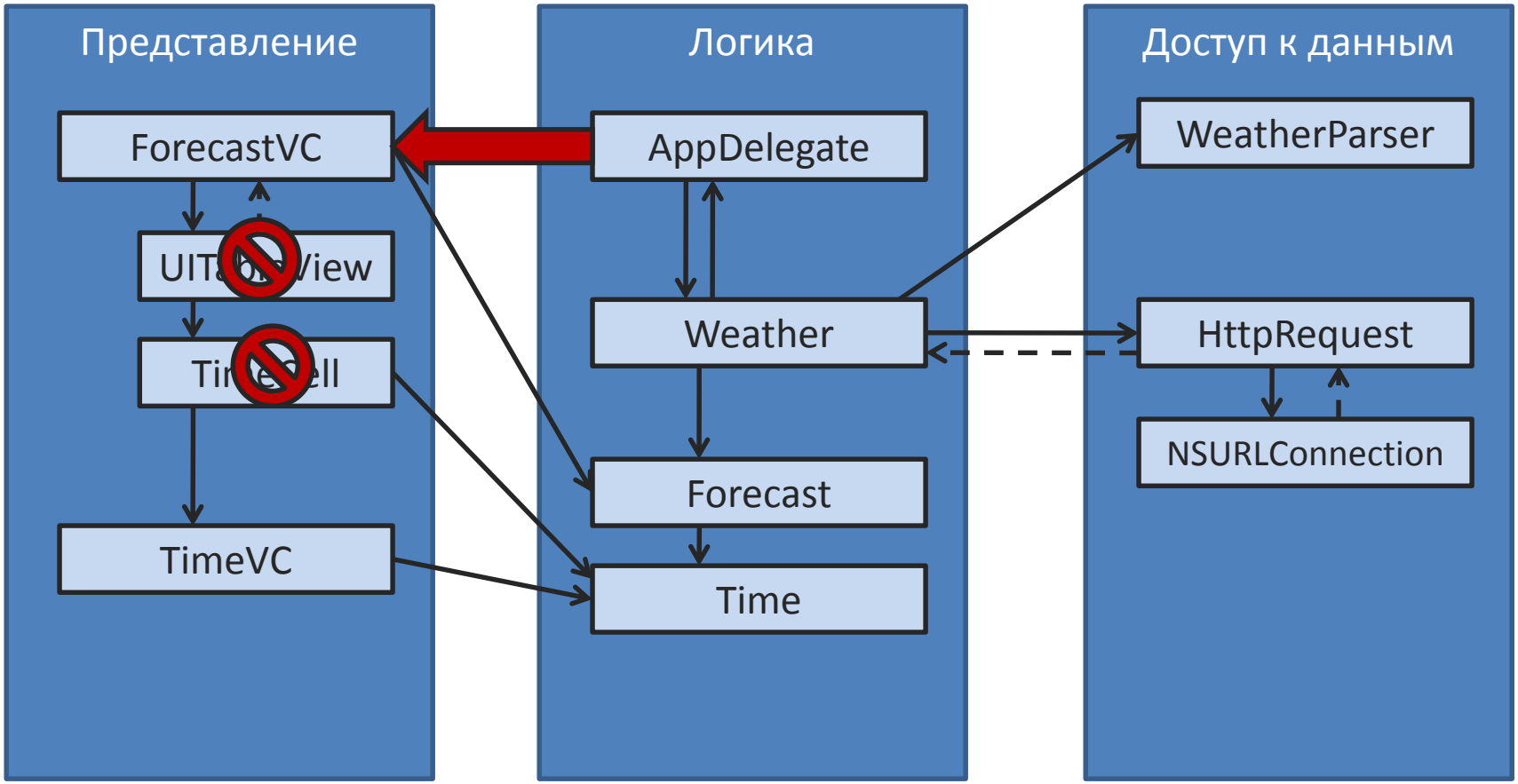
MVC

И снова MVC



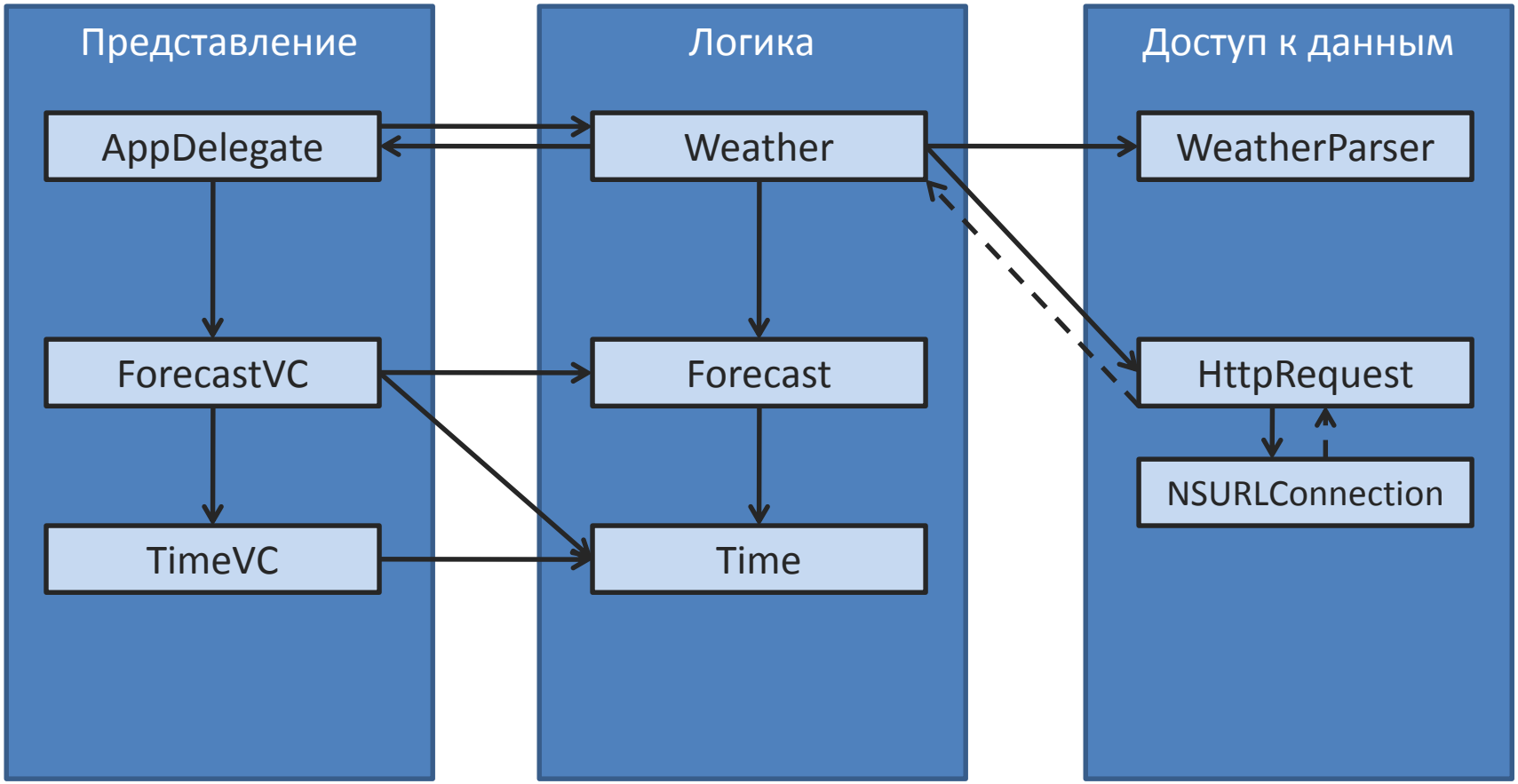
Об архитектуре приложения

- Улучшаем архитектуру дальше



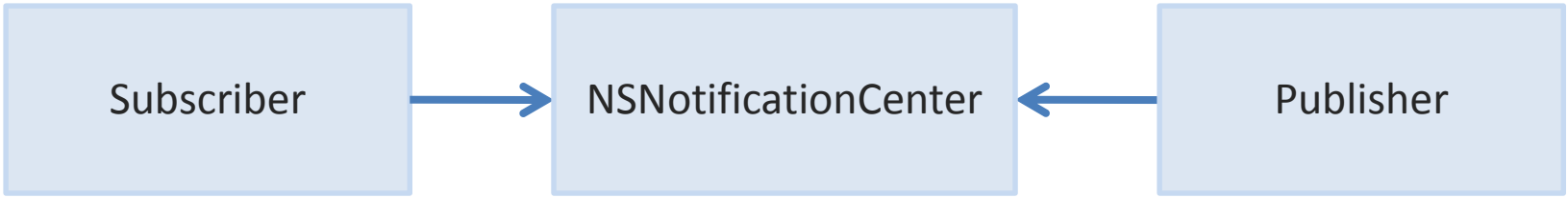
Об архитектуре приложения

- Улучшаем архитектуру дальше



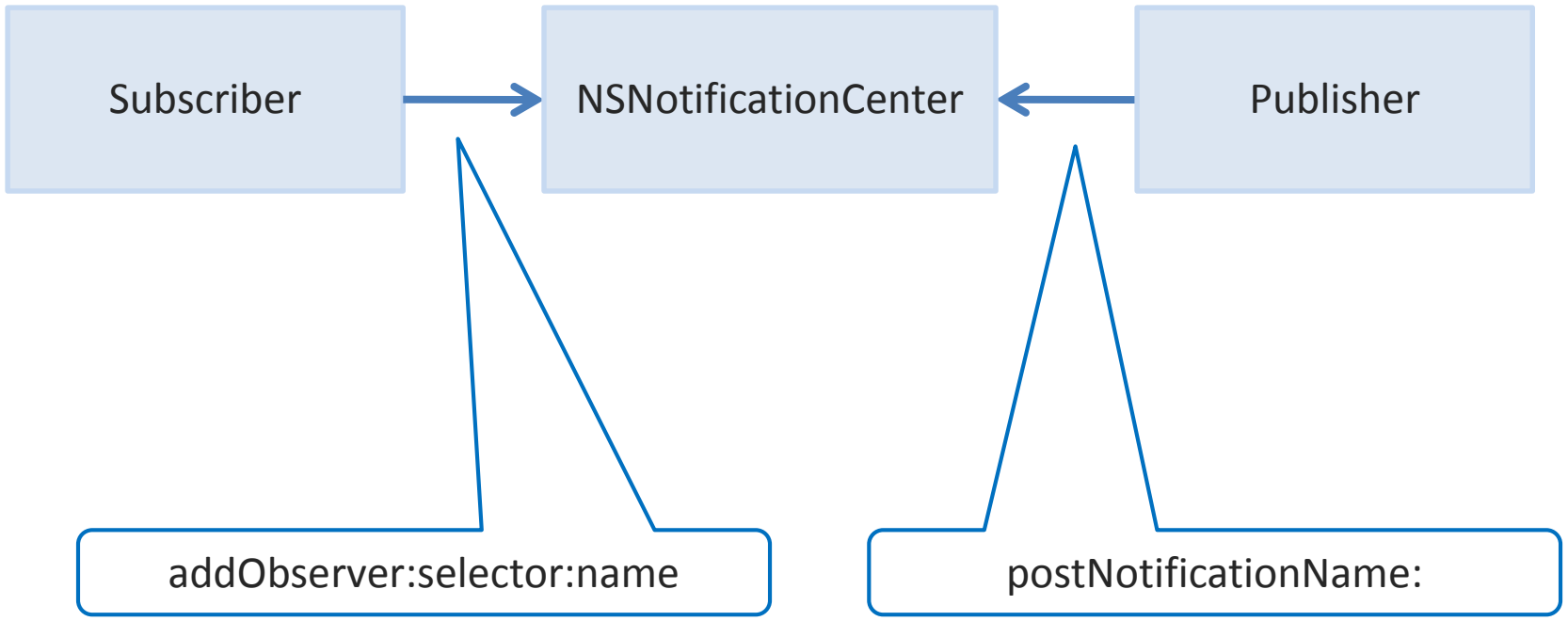
NSNotificationCenter

- NSNotificationCenter
- NSNotification



NSNotificationCenter

- NSNotificationCenter
- NSNotification



Пример

- Subscriber

```
[ [ NSNotificationCenter defaultCenter ]
  addObserver: self
  selector: @selector( timeRemoved: )
  name: @"Weather.time.removed"
  object: nil ];
```

- Publisher

```
[ [ NSNotificationCenter defaultCenter ]
  postNotificatioName: @"Weather.time.removed"
  object: self
  userInfo: paramsDictionary ];
```

Как передать параметры?

- Publisher

```
paramsDictionary = [ NSDictionary dictionaryWith... ]

[ [ NSNotificationCenter defaultCenter ]
  postNotificationName: @"Weather.time.removed"
  object: self
  userInfo: paramsDictionary ];
```

- Subscriber

```
-( void ) timeRemoved: ( NSNotification* )notification
{
  NSDictionary* params = [ notification userInfo ];
  id sender = [ notification object ];
}
```

Удаление подписки

- Subscriber

```
-( void ) dealloc
{
  [ [ NSNotificationCenter defaultCenter ]
    removeObserver: self ];
  [ super dealloc ]
}
```

Сравнение с target/selector

- Subscriber’у не нужно иметь ссылку на publisher’а
- Subscriber’ов может быть много
- Менее формальное описание интерфейса

Многопоточность

- Очень многогранный вопрос
- Вопрос: какие проблемы многопоточности вы знаете? …

Многопоточность

- Очень многогранный вопрос

- Вопрос: какие проблемы многопоточности вы знаете? …
    - Deadlock
    - Гонки (Race condition)

NSOperation

- Отдельно взятая операция, которую нужно выполнить в фоне

- Это абстрактный класс!
- Управляет создание потоков и временем жизни задачи
- Включает в себе некоторый объем работы для выполнения на фоновом потоке
- Содержит приоритеты и зависимости

Подкласс NSOperation

- Для того, чтобы сделать что-то полезное, нужно унаследоваться от NSOperation

- Создать нужный инициализатор (initWith…)

```
-( id ) initWithSomeObject: ( id )someObject
{
  self = [ super init ];
  if ( nil != self )
  {
    self.someObject = someObject;
  }
  return self;
}
```

- Переопределить метод main для выполнения работы

```
-( void ) main
{
  [ someObject doLotsOfTimeConsumingWork ];
}
```

NSInvocationOperation

- Когда делать подкласс слишком много для простой операции…

- Подкласс NSOperation
- Для легковесный задач

```
-( void ) someAction: ( id )sender
{
  NSInvocationOperation* op = [ [ NSInvocationOperation alloc ]
    initWithTarget: self
    selector: @selector( doWork: )
    object: someObject ];
}
```

NSOperationQueue

- Однако чтобы операция выполнилась ее нужно добавить в очередь!

- Очередь хранит в себе операции для выполнения и выполняет их
- У очереди можно установить максимальное количество одновременно выполняемых операций
- Очередь выполняет операции согласно их приоритету и зависимостям

NSOperationQueue

- Создание

```
NSOperationQueue* q = [ [ NSOperationQueue alloc ] init ];
```

- Установка максимального количества одновременных операций

```
[ q setMaxConcurrentOperationCount: n ];
```

- Добавление операции в очередь

```
[ q addOperation: operation ];
```

ВАЖНО!

- NSURLConnection внутри NSOperation работать не будет!...

ВАЖНО!

- NSURLConnection внутри NSOperation работать не будет!...
- … Потому что внутри этот класс использует RunLoop …

ВАЖНО!

- NSURLConnection внутри NSOperation работать не будет!...
- … Потому что внутри этот класс использует RunLoop …
- … Чтобы таки заработало, нужно специально создать поток и передать его в операцию …

ВАЖНО!

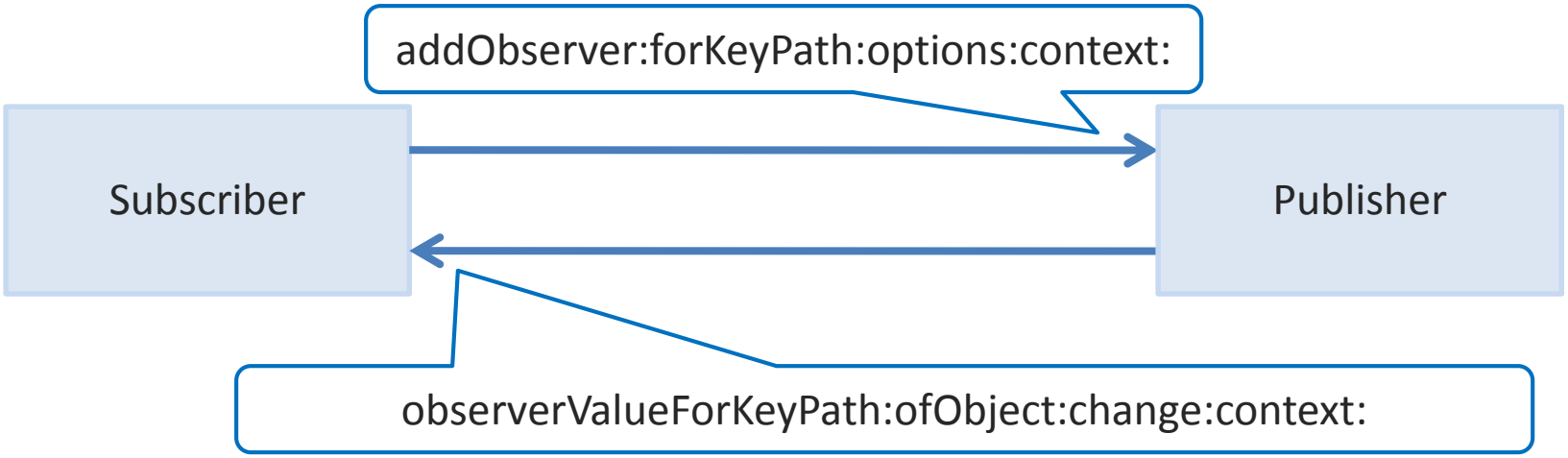
- NSURLConnection внутри NSOperation работать не будет!...
- … Потому что внутри этот класс использует RunLoop …
- … Чтобы таки заработало, нужно специально создать поток и передать его в операцию …
- … См. пример Apple: MVCNetworking

Key-Value Observing

- Еще один метод для получения уведомлений
- Используется для подписки на уведомления об изменении свойств объекта

Key-Value Observing

- NSNotificationCenter
- NSNotification



Key-Value Observing

- Subscriber

```
[ account addObserver: inspector
           forKeyPath: @"openingBalance"
              options: ( NSKeyValueObservingOptionNew |
                         NSKeyValueObservingOptionOld )
              context: nil ];
```

Key-Value Observing

- Subscriber

```
-( void ) observeValueForKeyPath: ( NSString* )keyPath
                        ofObject: ( id )object
                          change: ( NSDictionary* )change
                         context: ( void* )context
{
  if ( [ keyPath isEqual: @"openingBalance" ] )
  {
    id newValue = [ change objectForKey: NSKeyValueChangeNewKey ];
  }
}
```

Key-Value Observing

- Subscriber

```
-( void ) dealloc
{
  [ account removeObserver: self
                forKeyPath: @"openingBalance" ];
  [ super dealloc ];
}
```

Key-Value Observing

- По сравнению с target/selector и delegate
  - Меньше кода для подписки на изменение свойств (т.к. не нужен код на стороне publisher’a)
  - Также требуется ссылка на объект

Практика

• Перевести работу парсера на NSOperation + NSOperationQueue
• Перевести уведомления о подгрузке новых Forecast’ов с сервера на NSNotification + NSNotificationCenter
• Продолжить приведение архитектуры к правильному виду

Сложные задачи для тех кто справится раньше
• Получать картинку прогноза (если есть) и показывать ее в качестве картинки в TimeCell
• Получать картинку через операцию (NSOperation)
• Обновление в таблице сделать на основе Key-Value Observing (при остановке таблицы – методы UIScrollView, стартовать загрузку картинок и подписываться на свойства, при изменении свойств отображать картинку).

Итак…

- Это последнее занятие из основного цикла…
- Дальше я буду рассказывать некоторые темы, но практики по ним не будет

Что дальше:

- На следующем занятии вы должны сообщить мне, какой проект будете делать.
- Если не определитесь, я дам свой
- Необходимо доделать учебный проект и подготовить его к сдаче

Внимание!

Те кто хочет за счет спецкурса получить зачет, должны:

- Сообщить мне об этом
- Закончить учебный проект и сдать его мне.
- Принести зачетку

Требования к учебному проекту

- Два ViewController’а:
  - ForecastViewController: отображает таблицу с Time’ами с помощью специальной ячейки TimeCell
  - TimeViewController: отображает данные конкретного Time’а
  - Они должны выглядеть красиво. Я не требует специфичной графики, только стандартные контролы, но это должно выглядеть понятно.

- Модель:
  - Weather: класс управляет всеми модельными функциями
  - WeatherParser: делегат для XMLParser, осуществляет разбор XML
  - Forecast: класс реализующий элемент прогноза (forecast в xml)
  - Time: класс реализующий один элемент для времени прогноза

- Другие классы
  - Делегат для NSURLConnection
  - Классы NSOperation
  - Любые другие вспомогательные классы, которые вы считаете нужными