Периодично и отложено извикване на съобщения

EventKit

05.01.2012

Съдържание

- Нишки
- Периодично и отложено извикване на съобщения
- Календарни събития(EventKit framework)

Нишки - NSThread

- При създаване NSAutoreleasePool-а не се наследява
- Стартиране

```
- (NSThread *) initWithTarget:(id)target selector:(SEL)
aSelector object:(id) object
[thread start];
```

или

```
- (void)detachNewThreadSelector:(SEL)aSelector toTarget:(id)
aTarget withObject:(id)anArgument
```

- exit при завършване на aSelector
- aSelector приема един аргумент и не връща стойност
- Заспиване
  `- sleepForTimeInterval:` или `- sleepUntilDate`
- Спиране
  `- exit` или `- cancel` (ако нишката поддържа isCancelled)

Периодично извикване

- Планирано извикване на метод в главната нишка:

```
NSTimer *timer = [NSTimer scheduledTimerWithTimeInterval:
                          (NSTimeInterval)seconds target:self
                          selector:@selector(doSomething:)
                          userInfo:(id)anyObject
                          repeats:(BOOL)yesOrNo];
```

  - блокиране на главната нишка при:
    - твърде малък интервал
    - дълго изпълнение на селектора (затова времеотнемащите операции се изпълняват в главната нишка)
- Спиране на таймер

```
- (void)  invalidate;
```

  - при repeats:No инвалидирането става автоматично

Периодично извикване

- userInfo dictionary – може да се използва за подаване на аргументи към селектора

```
//inside selector:(NSTimer *) timer
NSDictionary *dict = [timer userInfo];
```

- добра практика е да извиквате [timer invalidate] директно в селектора, при repeat:No (за да не пазим отделна истанция на timer)

Отложено извикване

- Алтернатива на NSTimer, NSObject метод:

  - (void) performSelector:(SEL)aSelector withObject:(id)argument afterDelay:(NSTimeInterval)seconds;

- Изпълнява се в run-loop-а на текущата нишка
- не се изпълнява веднага, дори ако `seconds = 0`
- Пример:

```
[self.tableView performSelector:@selector(reloadData)
withObject:nil afterDelay:3];
```

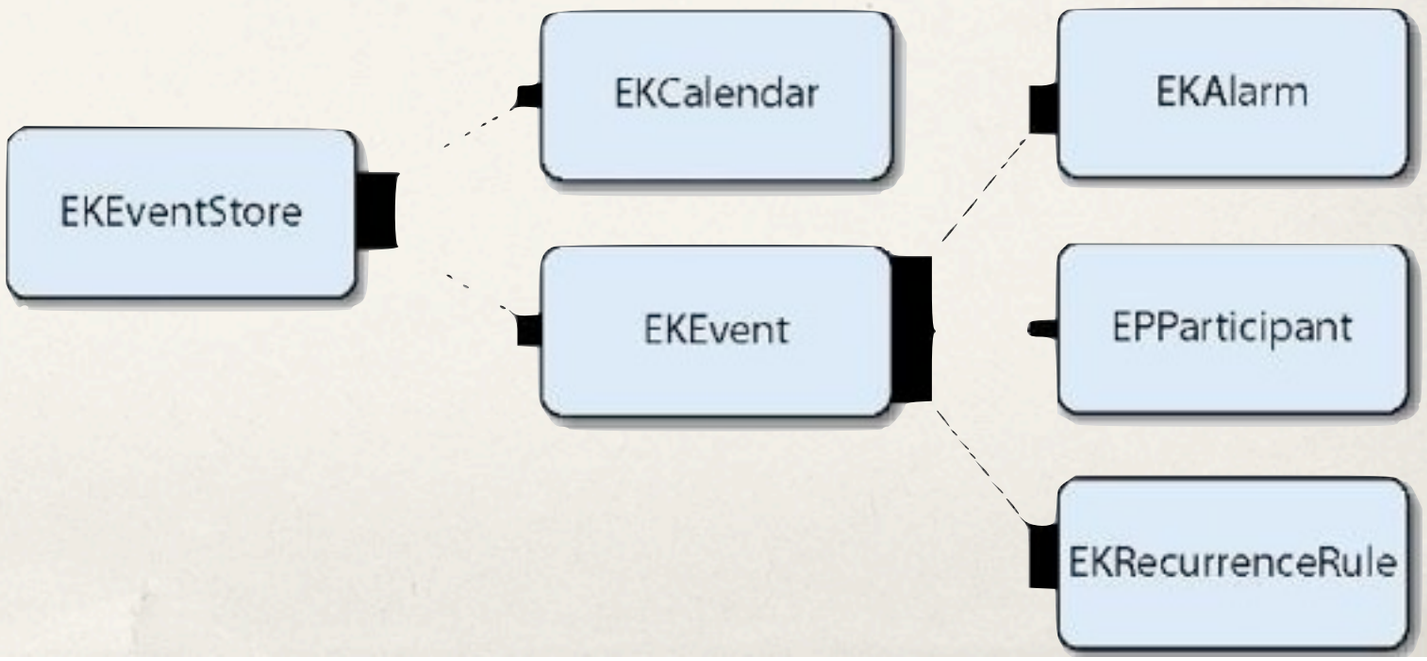
Отложено извикване

- Алтернатива на NSThread, NSObject съобщение:
  `- (void)performSelectorInBackground:(SEL)aSelector withObject:(id)arg`
  - създава нова нишка, в която се изпълнява селектора
- Спиране:
  `+ (void)cancelPreviousPerformRequestsWithTarget:(id) target` - всички регистрирани `performSelector` съобщения към този target спират
- `(void)performSelector:(SEL) aSelector withObject(id)arg`
  - извиква се динамично (не е нужно да знаем името на метода compile-time)

DEMO

EventKit

- Възможност за достъп и модифициране на календарни събития (събитията, видими в Calendar приложението).
- EventKit.framework – предоставя достъп до базата данни, в която се пазят събитията
- EventKitUI.framework – потребителски интерфейс за работа с календара (дублира някои функционалности на Calendar приложението)
- EKEventStore – връща ни два обекта, запълнени с информация от базата – EKCalendar, EKEvent



EventKit

- EKCalendar – съдържа колекция от събития(EKEvent)
  - различни типове – birthday, local, subscription...
  - defaultCalendarForNewEvents свойство
  - calendarWithEventStore: – ако искаме да си направим custom календар
  - allowContentModification свойство
- EKEvent – помни(напомня) кога нещо ще се случи
  - асоциирано с календар
  - модификации – съобщения към EKEventStore
    - saveEvent:(EKEvent):span:(EKSpan):error:(NSError)
    - removeEvent:span:error

EventKit

- EKEvent свойства

Table 1 EKEvent's property table

Property	Details
title	The event title, NSString type
location	The event location, NSString type
allday	BOOL type, indicates the event is an all-day event
startDate	The event start date
endDate	The event end date
calendar	The calendar for new event, EKCalendar type
attendees	An array of participants
alarms	An array of EKAlarm objects
eventIdentifier	The event unique identifier, NSString type.

EventKit

- Как да получим събитие от базата? (питаме EKEventStore)

```
-(NSPredicate *)
predicateForEventsWithStartDate:endDate:calendars:
- (NSArray *)eventsMatchingPredicate:(NSPredicate *)
```

  - NSPredicate – съдържа условие за филтриране на колекция

- Повтарящо се събитие – свойство от тип EKRecurrenceRule
  - честота – EKRecurrenceFrequency – на ден, на седмица, т.н
  - интервал – NSInteger
  - край – EKRecurrenceEnd – не задължително – брой повторения или дата
- EKSpan
  - модификацията засяга само това събитие
  - модификацията засяга това събитие и всичките му бъдещи

EventKit

- UI за редактиране/показване на събитие?
  - имплементираме методи на EKEventEditViewDelegate
  - EKEventViewController – показване на съществуващо събитие (възможност за редактиране – allowEditing свойство)
  - EKEventEditViewController – добавяне на ново събитие
  - наследяват UINavigationViewController (добавят се в стек на navigationController-а)
  - освобожаваме модалния изглед в `eventEditViewController:didCompleteWithAction`
  - показваме в кой календар ще се запазват новите събития в `eventEditViewControllerDefaultCalendarForNewEvents`

EventKit в действие

Благодаря за вниманието!

Въпроси?

Използвана литература

- NSThread class reference
- iPhone SDK threading tutorial
- NSTimer class reference
- how to use NSTimer
- NSObject class reference
- how to use performSelector:withObject:afterDelay
- EventKit programming guide