Event Kitプログラミングガイド

2010-09-22

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
U.S.A.

アップルジャパン株式会社
〒163-1450 東京都新宿区西新宿
3 丁目20 番2 号
東京オペラシティタワー
http://www.apple.com/jp/

目次

リスト

はじめに

Event KitおよびEvent Kit UIフレームワークはともに、iOSアプリケーションがユーザのカレンダーデータベースのイベント情報にアクセスできるようにします。また、日付範囲または一意の識別子に基づいたイベントのフェッチ、イベントレコードが変更されたときの通知の受け取り、およびカレンダーのいずれかに対するユーザによるイベントの作成と編集が可能になります。Event Kitを使用してユーザのカレンダーデータベースのイベントに対して行われた変更は、適切なカレンダー（CalDAV、Exchangeなど）と自動的に同期されます。この文書では、Event Kitの概念と、一般的なプログラミング作業について説明しています。

対象読者

この文書は、iOSアプリケーションでの、カレンダーのイベントデータの表示、またはカレンダーのイベントデータをユーザが編集できるようにしたい方を対象としています。Event Kitにより、ユーザのカレンダー情報へのアクセス制限が提供されます。Event Kitは、フル機能のカレンダーアプリケーションの実装には適していません。

この書類の構成

この文書は次の章で構成されています。

- 「イベントのフェッチ」（9 ページ）では、カレンダーデータベースのイベントをフェッチする方法を説明します。
- 「Event View Controllerの使用」（11 ページ）では、ユーザがイベントの作成と編集を行えるようにするために、Event View Controllerを表示する方法について説明します。
- 「プログラムによるイベントの作成と編集」（13 ページ）では、プログラムでイベントの作成と編集を行う方法について説明します。
- 「繰り返しのイベントの作成」（17 ページ）では、イベントを繰り返すイベントを作成する方法について説明します。
- 「イベントの変更の監視」（19 ページ）では、カレンダーデータベースに対する外部的な変更に関する通知を受け取るための登録を行う方法について説明します

関連項目

Event KitおよびEvent Kit UI APIの詳細な説明については、次の文書を参照してください。

- 『*Event Kit Framework Reference*』は、Event Kit API.を詳細に説明しています。
- 『*Event Kit UI Framework Reference*』は、Event Kit UI APIを詳細に説明しています。
- 『*SimpleEKDemo*』サンプルコードは、Event KitおよびEvent Kit UIフレームワークを使用してカレンダーデータにアクセスし、編集する基本例です。

イベントのフェッチ

EKEventStoreクラスを使用して、ユーザのカレンダーデータベースのイベントをフェッチできます。また、作成したプレディケート(Predicate)に一致するカスタムのイベントセットのフェッチや、一意の識別子による個々のイベントのフェッチを行うこともできます。イベントのフェッチ後は、EKEventクラスのプロパティを使用して、関連するカレンダー情報にアクセスできます。

イベントストアの初期化

次のデフォルトのイニシャライザを使用して、EKEventStoreオブジェクトを初期化します。

```
EKEventStore *store = [[EKEventStore alloc] init];
```

EKEventStoreオブジェクトは、初期化して解放するのに比較的長い時間がかかります。したがって、それぞれのイベント関連タスクについて、個別のイベントストアを初期化して解放するべきではありません。代わりに、アプリケーションがロードするときにイベントストアを1つ初期化して、それを繰り返し使用してください。

プレディケート(Predicate)を使用したイベントのフェッチ

一般に、特定の日付範囲内のイベントをフェッチします。イベントストアメソッドeventsMatchingPredicate:は、作成したプレディケートに指定されている日付範囲内のすべてのイベントをフェッチします。EKEventStoreのpredicateForEventsWithStartDate:endDate:calendars:メソッドを使用して、eventsMatchingPredicate:メソッドのプレディケートを作成する必要があります。リスト 1-1に、現在の日付の前30日と、現在の日付の後15日の間に発生したすべてのイベントのフェッチを示します。

リスト 1-1　プレディケートを使用したイベントのフェッチ

```
// プレディケートの開始と終了の日付を作成
CFGregorianDate gregorianStartDate, gregorianEndDate;
CFGregorianUnits startUnits = {0, 0, -30, 0, 0, 0};
CFGregorianUnits endUnits = {0, 0, 15, 0, 0, 0};
CFTimeZoneRef timeZone = CFTimeZoneCopySystem();

gregorianStartDate = CFAbsoluteTimeGetGregorianDate(
    CFAbsoluteTimeAddGregorianUnits(CFAbsoluteTimeGetCurrent(), timeZone,
startUnits),
    timeZone);
gregorianStartDate.hour = 0;
gregorianStartDate.minute = 0;
gregorianStartDate.second = 0;
```

```
gregorianEndDate = CFAbsoluteTimeGetGregorianDate(
    CFAbsoluteTimeAddGregorianUnits(CFAbsoluteTimeGetCurrent(), timeZone,
endUnits),
    timeZone);
gregorianEndDate.hour = 0;
gregorianEndDate.minute = 0;
gregorianEndDate.second = 0;

NSDate* startDate =
    [NSDate
dateWithTimeIntervalSinceReferenceDate:CFGregorianDateGetAbsoluteTime(gregorianStartDate,
 timeZone)];
NSDate* endDate =
    [NSDate
dateWithTimeIntervalSinceReferenceDate:CFGregorianDateGetAbsoluteTime(gregorianEndDate,
 timeZone)];

CFRelease(timeZone);

// プレディケートの作成
NSPredicate *predicate = [eventStore predicateForEventsWithStartDate:startDate
 endDate:endDate calendars:nil]; // eventStoreはインスタンス変数

// プレディケートに一致するすべてのイベントをフェッチ
NSArray *events = [eventStore eventsMatchingPredicate:predicate];
[self setEvents:events];
```

predicateForEventsWithStartDate:endDate:calendars:メソッドの*calendars*パラメータとして、EKCalendarオブジェクトの配列を渡すことによって、検索するカレンダーのサブセットを指定できます。ユーザのカレンダーは、イベントストアのcalendarsプロパティから取得できます。nilを渡すと、ユーザのカレンダーのすべてから、フェッチするようにメソッドに伝えます。

eventsMatchingPredicate:メソッドは同期的であるため、アプリケーションのメインスレッドで実行しないでください。非同期動作の場合は、dispatch_async関数、またはNSOperationオブジェクトを使用して、ほかのスレッドでメソッドを実行します。

識別子(Identifier)を使用した個々のイベントのフェッチ

個々のイベントをフェッチする場合で、イベントの一意の識別子を前に使用したプレディケートでフェッチして知っている場合、EKEventStoreのeventWithIdentifier:メソッドを使用してイベントをフェッチします。イベントの一意の識別子は、eventIdentifierプロパティを使用して取得できます。

開始日によるイベントのソート

アプリケーションは、多くの場合、開始日でソートしたイベントデータをユーザに表示します。EKEventオブジェクトの配列を日付でソートするには、配列のsortedArrayUsingSelector:を呼び出して、compareStartDateWithEvent:メソッドのセレクタを提供します。

Event View Controllerの使用

Event Kit UIフレームワークでは、次に示すように、イベントを処理するための2つのタイプのView Controllerが提供されています。

- `EKEventViewController`クラス。既存のイベントの表示や、ユーザによる編集を可能にする場合に使用します。
- `EKEventEditViewController`クラス。ユーザによるイベントの作成、編集、削除を可能にする場合に使用します。

イベントの表示と編集

`EKEventViewController`クラスを使用するには、イベントストアから取得する既存のイベントが必要です。このタイプのView Controllerを表示する前に、`event`プロパティと、ほかの表示オプションを設定する必要があります。リスト 2-1に、Event View Controllerの作成と、Navigation Controllerへの追加を行う方法を示します（`myEvent`がすでに存在すると仮定）。ユーザにイベントを編集させないようにするには、`allowsEditing`プロパティを`NO`に設定します。

リスト 2-1　既存のイベントの編集

```
    EKEventViewController *eventViewController = [[EKEventViewController alloc]
 init];
    eventViewController.event = myEvent;
    eventViewController.allowsEditing = YES;
    navigationController = [[UINavigationController alloc]
                            initWithRootViewController:eventViewController];
    [eventViewController release];
```

ユーザがイベントの表示を終了したときに通知を受け取るには、Event View Controllerにデリゲートを指定する必要があります。デリゲートは、`EKEventViewDelegate`プロトコルに準拠し、`eventViewController:didCompleteWithAction:`メソッドを実装する必要があります。

イベントの作成と編集

ユーザによるイベントの作成、編集、削除を可能にするには、`EKEventEditViewController`クラスと、`EKEventEditViewDelegate`プロトコルを使用します。Event Edit View Controllerの作成は、`eventStore`プロパティを設定する必要があることと、`event`プロパティを任意で設定することを除いて、Event View Controllerと同じです。

- View Controllerを表示するときに`event`プロパティが`nil`の場合、ユーザはデフォルトのカレンダーに新しいイベントを作成して指定されたイベントストアに保存します。

- eventプロパティを設定した場合、ユーザは既存のイベントを編集します。イベントは指定されたイベントストアに存在する必要があり、そうでない場合は例外が発生します。

EKEventEditViewControllerクラスのインスタンスは、リスト 2-2に示すように、モーダルモードで表示されるように設計されています。このコードでは、selfがNavigation Controllerの最上位のView Controllerです。モーダルモードでのView Controllerの詳細については、「Presenting a View Controller Modally」 in *View Controller Programming Guide for iOS*を参照してください。

リスト 2-2　モーダルモードでのEvent Edit View Controllerの表示

```
    EKEventEditViewController* controller = [[EKEventEditViewController alloc]
 init];
    controller.eventStore = myEventStore;
    controller.editViewDelegate = self;
    [self presentModalViewController: controller animated:YES];
    [controller release];
```

ユーザがイベントの編集を終了したときに通知を受け取るため、デリゲートを指定する必要もあります。デリゲートは、EKEventEditViewDelegateプロトコルに準拠し、モーダルモードのView Controllerを閉じるeventEditViewController:didCompleteWithAction:メソッドを実装する必要があります（リスト 2-3を参照）。一般に、モーダルモードのView Controllerを表示するオブジェクトは、それを閉じる責任もあります。

リスト 2-3　デリゲートがモーダルビューを閉じる

```
- (void)eventEditViewController:(EKEventEditViewController *)controller
didCompleteWithAction:(EKEventEditViewAction)action {
    [self dismissModalViewControllerAnimated:YES];
}
```

デリゲートでは、編集が終了したときにユーザが行ったアクションも渡されます。ユーザは、変更のキャンセル、イベントの保存、またはイベントの削除のいずれかを行います。デベロッパは、さらにアクションを行う必要がある場合、eventEditViewController:didCompleteWithAction:デリゲートメソッドを実装します。

プログラミングによるイベントの作成と編集

Event Kitフレームワークを使用することにより、ユーザは、カレンダーデータベース内で新しいイベントの作成と、既存のイベントの編集が可能になります。

注： イベントデータの変更をユーザが行えるようにするためのお勧めの方法は、Event Kit UIフレームワークで提供されている、Event View Controllerを使用することです。これらのEvent View Controllerを使用する方法については、「Event View Controllerの使用」（11 ページ）を参照してください。 Event View Controllerがアプリケーションに不適切な場合にのみ、この章で説明されている手法を使用してください。

常にユーザに通知

アプリケーションがユーザのカレンダーデータベースをプログラムで変更する場合、*変更を行う前に、ユーザの確認を得る必要があります*。アプリケーションは、ユーザから明確な指示がなければ、カレンダーデータベースを変更すべきではありません。

イベントの作成と編集

`EKEvent`クラスの`eventWithEventStore:`メソッドを使用して新しいイベントを作成します。

カレンダーデータベースからフェッチした、新しいイベントまたはイベントの詳細を編集するには、イベントの対応するプロパティを設定します。編集可能な詳細には、次のものが含まれます。

- イベントのタイトル
- イベントの開始日と終了日
- イベントが関連付けられたカレンダー

  イベントストアプロパティ`calendars`を使用して、ユーザのカレンダーの配列を取得できます。
- 反復するイベントの場合の、イベントの反復ルール
- イベントに関連付けられたアラーム

アラームの追加と削除

addAlarm:メソッドを使用して、イベントにアラームを追加できます。アラームは、絶対的な日付、またはイベントの開始日から相対的な日数を使用して作成されます。相対的な日数を使用して作成されたアラームは、イベントの開始前または開始日に発生する必要があります。イベントのアラームは、removeAlarm:メソッドを使用して削除できます。

イベントの保存

イベントに行った変更は、保存されるまで効果はありません。EKEventStoreのsaveEvent:span:error:メソッドを使用して、カレンダーデータベースに変更を保存します。それにより、イベントが所属するカレンダー（CalDAV、Exchangeなど）と、変更が自動的に同期されます。

繰り返しのイベントを保存する場合、saveEvent:span:error:メソッドのspanパラメータにEKSpanFutureEventsを指定することによって、発生するすべてのイベントに変更を適用できます。

イベントの削除

EKEventStoreのremoveEvent:span:error:メソッドを使用して、カレンダーデータベースからイベントを永久に削除します。

繰り返しのイベントを削除する場合、removeEvent:span:error:メソッドのspanパラメータにEKSpanFutureEventsを指定することによって、発生するすべてのイベントを削除できます。

プレディケート(Predicate)を使用したイベントの処理

EKEventStoreのenumerateEventsMatchingPredicate:usingBlock:メソッドを使用して、指定されたプレディケートに一致するすべてのイベントを処理できます。EKEventStoreのpredicateForEventsWithStartDate:endDate:calendars:メソッドを使用して、このメソッドにプレディケートを作成する必要があります。指定する処理は、EKEventSearchCallbackタイプのブロックです。

```
typedef void (^EKEventSearchCallback)(EKEvent *event, BOOL *stop);
```

ブロックでは、次の2つのパラメータが渡されます。

event
: これは、現在処理対象のイベントです。

stop
: このブロックが返されるときにイベントの処理を停止するには、このパラメータの値をYESに設定してenumerateEventsMatchingPredicate:usingBlock:メソッドに通知します。プレディケートに一致していても処理されていないイベントは、すべて未処理のまま残ります。

このメソッドを使用すると、ユーザのカレンダーデータベースに大きな変更をもたらす結果となることに留意してください。ユーザの確認を要求するとき、実行しようとしているアクションの完全な情報をユーザが確実に持つようにしてください。

`enumerateEventsMatchingPredicate:usingBlock:`メソッドは同期的であるため、アプリケーションのメインスレッドで実行しないでください。非同期動作の場合は、`dispatch_async`関数、または`NSOperation`オブジェクトを使用して、ほかのスレッドでメソッドを実行します。

繰り返しのイベントの作成

繰り返しのイベントは、日次、週次、月次、または年次で、繰り返すイベントです。繰り返しのパターンは組み合わせることができます。たとえば、年次で毎月第1週目と第2週目の火曜日と木曜日ごとに、イベントをスケジュールすることができます。

イベントがいつ発生するかを記述した、繰り返しのルールをイベントに割り当てることによって、イベントを繰り返すイベントを作成できます。繰り返しのルールは、`EKRecurrenceRule`クラスのインスタンスによって表されます。

基本的な繰り返しのルールの作成

`initRecurrenceWithFrequency:interval:end:`メソッドを使用して、日次、週次、月次、年次のシンプルなパターンで、繰り返しのルールを作成できます。このメソッドには次の3つの値を指定できます。

- **繰り返しの頻度(initRecurrenceWithFrequency)。**これは、繰り返しのルールが、日次、週次、月次、年次のいずれであるかを示す、`EKRecurrenceFrequency`タイプの値です。
- **繰り返しの間隔(interval)。**これは、パターンの繰り返しの頻度を指定する、0より大きい整数値です。たとえば、繰り返しのルールが週次の繰り返しルールで間隔が1の場合、パターンは毎週繰り返されます。繰り返しのルールが月次の繰り返しルールで間隔が3の場合、パターンは3か月ごとに繰り返されます。
- **繰り返しの終了(end)。**この省略可能なパラメータは、いつ繰り返しのルールが終了するかを示す、`EKRecurrenceEnd`クラスのインスタンスです。繰り返しは、特定の終了日、または発生回数を指定して終了することができます。

  繰り返しのルールの終了を指定しない場合は、`nil`を渡します。

複雑な繰り返しのルールの作成

`initRecurrenceWithFrequency:interval:daysOfTheWeek:daysOfTheMonth:monthsOfTheYear:weeksOfTheYear:daysOfTheYear:setPositions:end:`メソッドを使用して、複雑なパターンで繰り返しのルールを作成できます。基本的な繰り返しのルールと同様に、頻度、間隔、省略可能な繰り返しの終了を指定します。さらに、次の値を指定できます。

- **週のうちの曜日(daysOfTheWeek)。**日次の繰り返しのルールを除き、すべての繰り返しのルールに適用され、イベントが発生する曜日を示す`EKRecurrenceDayOfWeek`オブジェクトの配列を指定できます。

たとえば、EKRecurrenceDayOfWeekオブジェクトの配列と曜日の値EKTuesdayおよびEKFridayを指定して火曜日ごとおよび金曜日ごとに発生する繰り返しを作成できます。

- **月のうちの日付(daysOfTheMonth)。**月次の繰り返しのルールのみに適用され、イベントが発生する月の日付を示すNSNumberオブジェクトの配列を指定できます。値は、1～31、および-1～-31です。負の値は、月末から逆算して数えていることを示します。

  たとえば、1および-1の値を含む配列を指定すると、毎月の最初と最後の日に発生する繰り返しを作成できます。

- **年のうちの月(monthsOfTheYear)。**年次の繰り返しのルールのみに適用され、イベントが発生する1年のうちの月を示すNSNumberオブジェクトの配列を指定できます。値は、1～12です。

  たとえば、起因するイベントの発生が1月10日の場合、1および2の値を含む配列を指定すると、毎年1月10日と2月10日に発生する繰り返しを作成できます。

- **年のうちの週(weeksOfTheYear)。**年次の繰り返しのルールのみに適用され、イベントが発生する1年のうちの週を示すNSNumberオブジェクトの配列を指定できます。値は、1～53、および-1～-53です。負の値は、年の終わりから逆算して数えていることを示します。

  たとえば、起因するイベントの発生が水曜日の場合、1および-1の値を含む配列を指定すると、年次で最初と最後の週の水曜日に発生する繰り返しを作成できます。指定した現在の年の週に水曜日が含まれない場合（年初または年末など）、イベントは発生しません。

- **年のうちの日付(daysOfTheYear)。**年次の繰り返しのルールのみに適用され、イベントが発生する1年のうちの日を示すNSNumberオブジェクトの配列を指定できます。値は、1～366、および-1～-366です。負の値は、年の終わりから逆算して数えていることを示します。

  たとえば、1および-1の値を含む配列を指定すると、毎年の最初と最後の日に発生する繰り返しを作成できます。

- **位置の設定(setPositions)。**日次の繰り返しのルールを除き、すべての繰り返しのルールに適用され、繰り返しのルールに含まれるイベントの発生をフィルタするNSNumberオブジェクトの配列を指定できます。このフィルタは、デベロッパが指定したほかのパラメータによって決定される、一連の繰り返し発生に適用されます。値は、1～366、および-1～-366です。負の値は、発生のリストの終わりから逆算して数えていることを示します。

  たとえば、1および-1の値を含む配列を指定すると、曜日の値として月曜日から金曜日までが指定された年次の繰り返しのルールにおいて、毎年最初と最後の平日にのみ発生する繰り返しを作成できます。

前述のパラメータの値はいくつでも指定できます（特定の繰り返しルールに適合しないパラメータは無視されます）。前述のパラメータの2つ以上の値を指定した場合、指定したすべての値に適合する日にのみ繰り返しが発生します。

イベントの変更の監視

ユーザのカレンダーデータベースは、アプリケーションの実行中に、ほかのプロセスまたはアプリケーションによって変更される可能性があります。アプリケーションがカレンダーイベントをフェッチする場合、カレンダーデータベースへの変更についての通知を受け取ることを登録する必要があります。これにより、ユーザに表示するカレンダー情報が最新であることを確実にします。

通知の監視

カレンダーデータベースへの変更が検出されると、`EKEventStore`オブジェクトによって`EKEventStoreChangedNotification`通知が送信されます。アプリケーションがイベントデータを処理する場合、この通知を受け取ることを登録します。

次のコードは、`EKEventStoreChangedNotification`通知を受け取ることを登録します。

```
[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(storeChanged:)
    name:EKEventStoreChangedNotification object:eventStore];
```

通知への応答

`EKEventStoreChangedNotification`通知を受け取った場合、フェッチして保持した`EKEvent`オブジェクトに変更が加えられている可能性があります。これらの変更の影響は、イベントの追加、変更、削除のうち、いずれが行われたかによって異なります。

- イベントが追加された場合、保持しているイベントのいずれにも影響はないが、追加されたイベントはユーザに表示するイベントの日付範囲内になる。
- イベントが変更または削除された場合、そのイベントを表す`EKEvent`オブジェクトのプロパティが期限切れになる。

カレンダーデータベースに変更が発生した場合、ローカルデータがしばしば無効または不完全になるため、`EKEventStoreChangedNotification`通知を受け取ったときには常に、イベントの現在の日付範囲を解放してフェッチしなおす必要があります。現在イベントを変更しており、どうしても必要ない限りフェッチしなおしたくない場合は、イベントで`refresh`メソッドを呼び出せます。メソッドから`YES`が返された場合はイベントの使用を継続できますが、そうでない場合はイベントを解放してフェッチしなおす必要があります。

Event View Controller内で変更されたイベントは、カレンダーデータベースに変更が発生したときに自動的に更新されます。

書類の改訂履歴

この表は「*Event Kit プログラミングガイド*」の改訂履歴です。

日付	メモ
2010-09-22	全体を通して改訂を行いました。
2010-08-03	SimpleEKDemoのサンプルコードへのリンクを追加しました。
2010-04-29	Event Kitフレームワークを使用してiOSのカレンダーデータにアクセスする方法について説明した新規文書。

書類の改訂履歴