Изгледи, контролери и делегати

Станимира Железова

Съдържание

Стартиране на iOS приложение

- main.m

```
int main(int argc, char *argv[]) {
NSAutoreleasePool *pool = [[NSAutoreleasePool alloc] init];
int retVal = UIApplicationMain(argc, argv, nil, nil);
[pool release];
return retVal;
}
```

- Създава главния NSAutoreleasePool
- Извиква главната функция - UIApplicationMain
- Аргументите arc и argv не се използват. Има ги само за консистентност със стандартното C

UIApplicationMain

- UIApplicationMain – създава нов обект, наследник на UIApplication, който дава централизиран достъп до приложението.

```
int UIApplicationMain (
    int argc,
    char *argv[],
    // the UIApplication class to instantiate
    NSString *principalClassName,
    // the UIApplicationDelegate class to instantiate
    NSString *delegateClassName
    );
```

UIApplication

- Всяко приложение може да има точно един обект от тип UIApplication (или негов наследник)

  ```
  [UIApplication sharedApplication]; // връща този обект
  ```

- Този клас приема и управлява всички събития идващи от операционната система или потребителя – засечени от хардуерните сензори.
- Държи списък от всички прозорци, които са отворени в приложението.
  - Всички визуални елементи в едно приложение са разположени в дървовидна структура, корен на която е представител на UIWindow.

UIView

- Базов клас за всички визуални елементи в приложението.
- Всички view-та могат да имат много деца (subviews), но само един родител (superview)
  - UIWindow е корена на всяка йерархия от view-та
- Всички наследници на класа могат да обработват събития
  - Класа UIResponder дава възможност за директо обработване на засечените събития (и touches)
  - addGestureRecognizer: - за откриване на стандартни жестове

UIView

- Заема правоъгълна част от екрана на приложението:
  - frame - дава позицията и големината на view-то, като измеренията са в координатната система на съдържащото view (superview)
    - origin - задава позиция в X и Y координати
    - dimensions – задава размери в width и height

```
CGRect viewRect = CGRectMake(10, 10, 100, 100);
UIView* myView = [[UIView alloc] initWithFrame:viewRect];
```

UIView

- Можете да променяте основния изглед на всяко view чрез пропъртита:
  - backgroundColor – фонов цвят
  - hidden – има стойност NO когато view-то е видимо
  - alpha – ниво на прозрачност
- Можете да променяте съдържанието, чрез добавяне и махане на деца:
  - – addSubview:
  - – removeFromSuperview – извиква се на view-то, което трябва да се откачи от своя родител

UIWindow

- Прозорците са наследници на UIView
  - Те служат като контейнери за всички други view-та в приложението, грижат се за тяхното рисуване и разпределят събития до тях.
- Всяко приложение има точно един основен прозорец (keyWindow). Той е видим пред всички други и приема идващите събития.
  - – makeKeyAndVisible
  - – becomeKeyWindow – извиква се автоматично
  - – makeKeyWindow
  - – resignKeyWindow – извиква се автоматично

UIWindow

- Обикновено едно iOS приложение съдържа само един прозорец – основния (keyWindow).
- UIWindowLevel windowLevel – нивото, на което да е показан прозореца.
- Можете да добавяте визуални елементи директно към UIWindow, но е препоръчително да ползвате пропъртито rootViewController
  - Даването на нова стойност на rootViewController замества съдържанието на UIWindow с главното view на този елемент.

UIViewController

- Основен клас за управление на визуалните компоненти и изграждането на потребителски интерфейс за всяко iPhone/iPad приложение
  - Наследява UIResponder (НЕ UIView) – може да приема и препраща събития
  - Всяка инстанция на този клас се грижи за презентацията на йерархия от визуални елементи

    @property(nonatomic, retain) UIView *view

    Обикновено всеки UIViewController заема цял екран от приложението

UIViewController

- Може да управлява модални визуални елементи – най-често други контролери:
  - presentModalViewController:animated:
  - –dismissModalViewControllerAnimated:
  - @property parentViewController – получава стойност автоматично когато бъде показан от друг контролер
- Отговаря на промени в ориентацията на устройството
  - interfaceOrientation – пропърти за четене на текущата ориентация на устройството
  - shouldAutorotateToInterfaceOrientation: - връща YES ако приложението ви поддържа подадената ориентация

UIViewController

- Създаване на UIViewController
  - – initWithNibName:bundle: - Стандартния конструктор за този клас. Подадения .nib файл се зарежда при първото достъпване на view

    ```
    MyViewController *myVC = [[MyViewController alloc] initWithNibName:@”InfoView” bundle:[NSBundle mainBundle]];
    myVC.view;
    ```

  - nibName – името на файла, който държи описанието на визуланите елементи за контролера
  - nibBundle – групата, от която да се вземе nib файла. Ако има стойност nil, се ползва mainBundle

UIViewController

- Дава методи за управление на view-то на всяка стъпка от неговото зареждане и разрушаване:
  - – loadView – дава възможност за програмно изграждане на йерархията от визуални елементи (вместо използване на .nib файл)
  - – viewDidLoad – извиква се щом зареждането на всички обекти от йерархията приключи
  - – viewWillUnload – извиква се преди да започне да разрушава визуалните обекти
  - – viewDidUnload – извиква се след разрушаване на визуланите обекти от йерархията на контролера

UIViewController

- Когато инициализирате view и стойността на пропъртито nibName е nil, а не сте наследили и метода loadView, автоматично се търси .nib файл с име:
  - Ако името на класа завършва с ‘Controller’, се търси файл с име [начало на името на класа].nib

    За класа “MyCoolViewController” -> “MyCoolView.nib”
  - Ако няма такъв файл или името не завършва с ‘Controller‘, се търси файл с име същото като името на класа и разширение „.nib“

    За класа „MyGameStage1” -> “MyGameStage1.nib”

UIViewController

- Дава възможност за задаване на специфични действия на всяка спъпка от показването и скриването на view-то на контролера:
  - – viewWillAppear: - извиква се точно преди показване на view-то на екрана
  - – viewDidAppear: - извиква се след показване на view-то на екрана
  - – viewWillDisappear: - извиква се преди скриването на view-то от екрана
  - – viewDidDisappear:– извиква се след скриването на view-то от екрана

Делегати

- Основен шаблон за дизайн, използван за разработката на iOS приложения.
  - стандартен начин на предаване на контрола от класа, който изпълнява основната функционалнст, към помощен клас - негов деелгат, който да изпълни специфични за конкретната ситуация дейности
  - например - класа UIApplication има делегат, който отговаря на протокола UIApplicationDelegate

Делагати

- Всяко приложение трябва да дефинира клас, който реализира методите от UIApplicationDelegate, за да обработи по специифичен начин, стандартни събития от живота на приложението:

– applicationDidBecomeActive:
– applicationWillResignActive:
– applicationDidEnterBackground:
– applicationWillEnterForeground:
– applicationWillTerminate:
– applicationDidFinishLaunching:

LOG
O

Благодаря за вниманието!

Въпроси?