绘图

`A.视图`

---

1. 基本概念

许多UIView子类都知道如何绘制自己；不过，特殊情况下，你要做一些你自己的绘图。你需要做的是：子类化UIView，并赋予子类绘制自身的能力。

每当 UIView 需要绘图时，系统会调用它的 **- drawRect:** 方法 就行。你可以 **重写** 该方法来实现自己需要的绘图。

`注意` Apple 不允许你直接调用 drawRect: 方法。

如果你觉得视图需要更新，你希望 drawRect:方法执行，给该视图发送 **- setNeedsDisplay** 消息 即可。 系统会在下一个适当时刻由来调用drawRect:方法。

---

2. CG接口 和 UIKit接口

Core Graphics 是一套完整的绘图API。Core Graphics，通常是指 Quartz 或 Quartz 2D，它是所有iOS绘图的基础绘图系统。UIKit绘图系统也是建立在它之上的。

Core Graphics API是一套C函数接口；UIKit API是一套OC面向对象接口。

绘图上下文，是一个你可以绘图的地方。它包含了绘图时所需要的所有“客观因素”，更重要的是，它关联着绘图图像渲染目的地。其关联目的地可以是：iOS应用的window，位图、PDF文件 或 打印机。

你可以创建一个图像上下文，目的是将绘制内容渲染至位图上；也可以使用系统提供给你的绘图上下文，目的是将绘制内容渲染至屏幕上。

在UIView的drawRect:方法中，调用 **UIGraphicsGetCurrentContext ( )** 函数 会返回当前的绘图上下文，该绘图上下文也就是系统提供给你的上下文。该该上下文中绘制的图像都将显示在屏幕上。

---

3. 设置绘图上下文

当你在一个绘图上下文中绘图时，绘图过程中会使用上下文的当前设置，如线条颜色，粗细等。因此，绘图过程总是首先配置上下文。

线条粗细

CGContextSetLineWidth ( )

线条连接样式

CGContextSetLineJoin ( )

线条颜色

CGContextSetStrokeColorWithColor ( )

CGContextSetRGBStrokeColor ( )

填充颜色

CGContextSetFillColorWithColor ( )

CGContextSetRGBFillColor ( )

阴影

CGContextSetShadowWithColor ( )

---

4. 路径和绘图

想要移动一支假象的笔，可以执行一系列指令，给你描绘出一条路径，对这条路径进行 **描边**（Stroke），或 **填充**（Fill），则构成了绘图。

注意 若无描边和填充操作，单纯的路径并不构成绘图！

当前点的位置

CGContextMoveToPoint ( ) 。移动“笔”到某个点上，注意，在此移动过程中，笔是“悬空”的，并不在“纸”上。

描绘线条

CGContextAddLineToPoint ( )

CGContextAddLines ( )

描绘矩形

CGContextAddRect ( )

描绘椭圆或圆形

CGContextAddEllipseInRect ( )

描绘弧线

CGContextAddArc ( )

关闭当前路径

CGContextClosePath ( ) 。从路径最后一个点到第一个点附加一条线。如果是填充路径，默认会实现该功能。

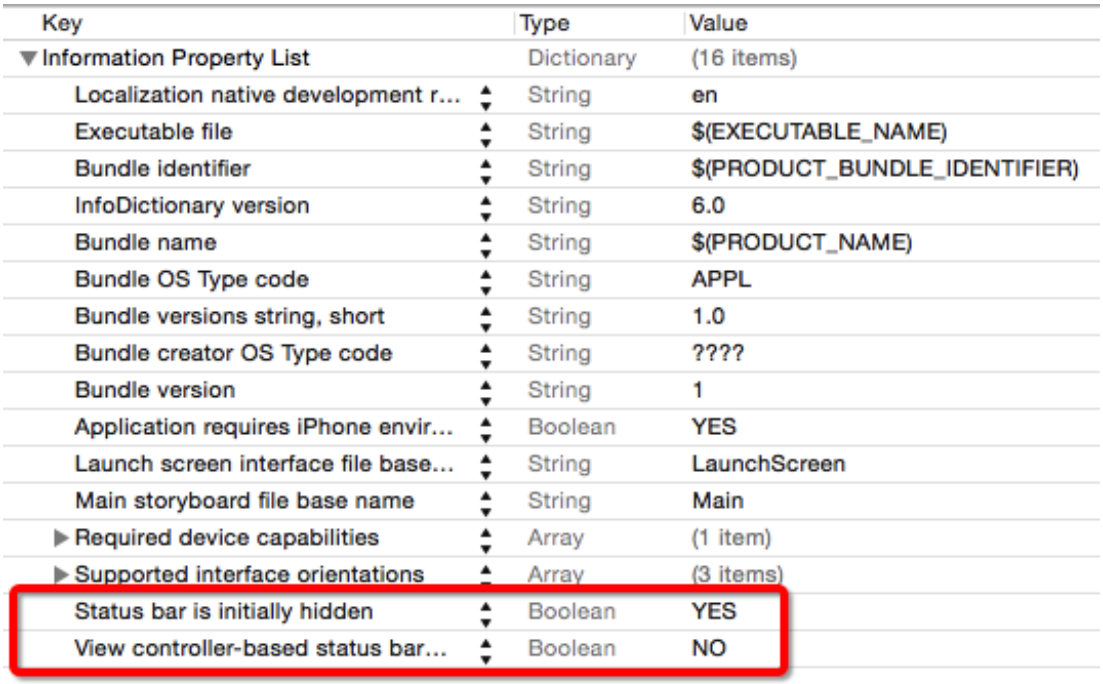
描边或填充当前路径

CGContextStrokePath ( )

CGContextFillPath ( )

---

5. 路径



Key	Type	Value
▼ Information Property List	Dictionary	(16 items)
Localization native development r...	String	en
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle creator OS Type code	String	????
Bundle version	String	1
Application requires iPhone envir...	Boolean	YES
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
▶ Required device capabilities	Array	(1 item)
▶ Supported interface orientations	Array	(3 items)
Status bar is initially hidden	Boolean	YES
View controller-based status bar...	Boolean	NO

CGPathCreateMutable *方法*

Creates a new mutable path of type CGMutablePathRef and returns its handle. We should dispose of this path after we are done with it, as you will soon see.

CGPathMoveToPoint *方法*

Moves the current pen position on the path to the point specified by a parameter of type CGPoint.

CGPathAddLineToPoint *方法*

Draws a line segment from the current pen position to the specified position (again, specified by a value of type CGPoint).

CGContextAddPath *方法*

Adds a given path (specified by a path handle) to a graphics context, ready for drawing.

CGContextDrawPath *方法*

Draws a given path on the graphics context.

CGPathAddRect *方法*

Adds a rectangle to a path. The rectangle’ s boundaries are specified by a CGRect structure.

There are three important drawing methods that you can ask the

CGContextDrawPath procedure to perform:

kCGPathStroke *参数*

Draws a line (stroke) to mark the boundary or edge of the path, using the currently selected stroke color.

kCGPathFill *参数*

Fills the area surrounded by the path with the currently selected fill color.

kCGPathFillStroke *参数*

Combines stroke and fill. Uses the currently selected fill color to fill the path and the currently selected stroke color to draw the edge of the path. We’ ll see an example of this method in the following section.

---

6. 绘制文字

6.1 Fonts

iOS中将所有的文字以关联关系分成了 **文字族** 。通过 UIFont 的类方法 **+ familyNames**，能获取到当前系统支持的所有**文字族**的名字。

通过UIFont的类方法 **+ fontNamesForFamilyName:** ，可以获得参数（文字族的名字）指定所有文字的文字名。

通过UIFont的类方法 **+ fontWithName:size:** ，根据文字的名字，及其字体大小，实例化 UIFont 对象。

6.2 带属性的字符串

自iOS 6以后，系统提供了 **NSAttributedString** 类 。一个NSAttributedString对象概念上包含了字符串的内容，以及显示该字符串时的效果，也叫属性。

属性在字典中被描述。每个可能的属性都有一个预先定义的名称，作为在字典中使用的关键字。

最常见的属性对应名称

NSFontAttributeName

对应一个UIFont对象，决定显示文字时的字体和大小。

NSForegroundColorAttributeName

对应一个UIColor对象，决定显示文字时的颜色。

6.3 绘制文字

NSString提供了绘制自身到context的方法。所以在使用以下方法绘制NSString时，不需要再处理context。

使用NSString对象的 **- drawAtPoint: withAttributes:** 实例方法，可将消息接受者从参数指定地点开始，绘制于其所在的drawRect:方法的视图中。

注意 如果文字内容过长，该方法 **不会** 使文字换行。

使用NSString对象的 **- drawInRect: withAttributes:** 实例方法，可将消息接受者绘制于指定的矩形区间内。文字内容过长时，该方法会使文字换行。

---

7. 绘制图片

与 NSString 类似，UIImage 也提供了绘制自身到当前上下文的方法。

- drawAtPoint: 是按照图片自身尺寸在UIView上绘制。如果UIView过小，则会有部分区域无法绘制的情况。

- drawInRect:，则会按照参数所指明矩形大小绘制。矩形区域大过图片本身大小，则产生拉伸效果；小于图片本身大小，则产生压缩效果。

- drawAsPatternInRect: ，在绘制时，如果参数矩形大于图片本身大小，则会产生平铺效果。单张图片既不拉伸，也不压缩。