Sample Code

Custom Graphics

Demonstrates adding a watermark to a PDF page.

Download

iOS 11.0+ | iPadOS 11.0+ | Xcode 11.3+

Overview

You can add a custom drawing to a page or annotation by overriding that class's draw method. Adding drawings to pages can be useful for adding custom layered effects or, as shown in this sample, watermarking a document.

Register the Delegate and Set the Page Class

The `ViewController` class first initializes a `PDFDocument` instance and sets its delegate to `self`.

```
// 1. Set delegate
document.delegate = self
pdfView?.document = document
```

The delegate, of type `PDFDocumentDelegate`, implements a `classForPage()` method. This method returns `AnyClass` and declares that all instances of `PDFPage` for the document presented through `PDFView` should instantiate the subclass `WatermarkPage`.

This subclass, found in `WatermarkPage.swift`, implements custom drawing.

```
// 2. Return your custom PDFPage class
/// - Tag: ClassForPage
```

```
func classForPage() -> AnyClass {
    return WatermarkPage.self
}
```

ViewController loads a URL to the Sample.pdf file through the app's main bundle. This URL is then used to instantiate a PDFDocument. On success, the document is assigned to the PDFView, which was set up in Interface Builder.

The delegate is assigned before the document, so classForPage(), a PDFDocumentDelegate method, is implemented. This method returns the PDFPage subclass used for custom drawing.

Override the Draw Method

WatermarkPage subclasses PDFPage so that it can override the draw(with:to:) method. This method is called by PDFDocument to draw the page in a PDFView. All custom drawing for a PDF page should be done through this mechanism.

```
// 3. Override PDFPage custom draw
/// - Tag: OverrideDraw
override func draw(with box: PDFDisplayBox, to context: CGContext) {

    // Draw original content
    super.draw(with: box, to: context)

    // Draw rotated overlay string
    UIGraphicsPushContext(context)
    context.saveGState()

    let pageBounds = self.bounds(for: box)
    context.translateBy(x: 0.0, y: pageBounds.size.height)
    context.scaleBy(x: 1.0, y: -1.0)
    context.rotate(by: CGFloat.pi / 4.0)

    let string: NSString = "U s e r   3 1 4 1 5 9"
    let attributes: [NSAttributedString.Key: Any] = [
        NSAttributedString.Key.foregroundColor: #colorLiteral(red: 0.4980392157, gre
        NSAttributedString.Key.font: UIFont.boldSystemFont(ofSize: 64)
    ]

    string.draw(at: CGPoint(x: 250, y: 40), withAttributes: attributes)
```

```
        context.restoreGState()
        UIGraphicsPopContext()

    }
```

Custom drawing methods should always be thread-safe and call the superclass method, which is required to draw the original `PDFPage` content. Custom drawing code can execute before or after this superclass call, although order matters. If your graphics run before the superclass call, they're drawn below the `PDFPage` content. Conversely, if your graphics run after the superclass call, they're drawn above the `PDFPage` content.

See Also

Annotations

Adding Widgets to a PDF Document

Add text, button, and choice widgets to a PDF document.

Adding Custom Graphics to a PDF

Create and add custom annotation and page graphics to your PDF document.

{} PDF Widgets

Demonstrates adding widgets—interactive form elements—to a PDF document.

`class PDFAnnotation`

An annotation in a PDF document.