SCIENCES SORBONNE UNIVERSITÉ

draw/drawRect or not? that is he question

Fabrice.Kordon@lip6.fr

LIP6

As an introduction…

What is the question?

- «MyView» example
  - **Use of draw/drawRect**
- «uRotate» example
  - **no use of draw/drawRect**

Let's play with these approaches?

A new ViewController


```
class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
        let screen = UIScreen.main
        let rect = screen.bounds
        let v = MyView(frame: rect)
        self.view = v
        v.drawInFormat(size:rect.size) // Implemented by MyView
    }

    override func viewWillTransition(to size: CGSize,
                                     with coordinator:
UIViewControllerTransitionCoordinator) {
        super.viewWillTransition(to: size, with: coordinator)
        let v = self.view as! MyView // avoids an error
        v.drawInFormat(size:size) // Implemented by MyView
    }
}
```

MyView without draw()


```
import UIKit

class MyView: UIView {

    fileprivate let myDevice = UIDevice.current
    fileprivate let screen = UIScreen.main

    fileprivate let deviceData = UILabel()
    fileprivate let model = UILabel()
    fileprivate let orientation = UILabel()

    fileprivate var top = 0
    fileprivate var incr = 0

    // Required by Xcode (but unused)
    required init?(coder aDecoder: NSCoder) {
        super.init(coder: aDecoder)
        fatalError("init(coder:) has not been implemented")
    }
```

MyView without draw()


```
    override init (frame: CGRect) {
        deviceData.textAlignment = .center
        model.textAlignment = .center
        orientation.textAlignment = .center
        model.text = myDevice.model
        super.init(frame: frame)
        // Handled after the call to "super" (we need self here)
        if myDevice.userInterfaceIdiom == .phone && screen.scale == 3.0 {
            deviceData.text = "Large iPhone (\(myDevice.systemName), \
(myDevice.systemVersion))"
            incr = 50
        } else if myDevice.userInterfaceIdiom == .phone {
            deviceData.text = "iPhone/iPod (\(myDevice.systemName), \
(myDevice.systemVersion))"
            incr = 40
        } else {
            deviceData.text = "iPad (\(myDevice.systemName), \
(myDevice.systemVersion))"
            incr = 100
        }
        // Let's deal with the view's elements
        // Handled after the call to "super" (these are super's properties)
        self.backgroundColor = UIColor.white
        self.addSubview(deviceData)
        self.addSubview(model)
        self.addSubview(orientation)
        self.drawInFormat(size: frame.size)
    }
```

MyView without draw()


```
func drawInFormat (size: CGSize) {
    if myDevice.orientation == UIDeviceOrientation.portrait ||
        myDevice.orientation == UIDeviceOrientation.portraitUpsideDown {
        orientation.text = "Portrait orientation"
        top = 100
    } else if myDevice.orientation == UIDeviceOrientation.landscapeLeft ||
        myDevice.orientation == UIDeviceOrientation.landscapeRight {
        orientation.text = "Landscape orientation"
        top = 50
    } else {
        orientation.text = "Device is laid flat"
    }
    // Setting up orientation information
    deviceData.frame = CGRect(x: CGFloat(size.width / 2.0 - 150.0),
                              y: CGFloat(top), width: 300.0, height: 21.0)
    model.frame = CGRect(x: CGFloat(size.width / 2.0 - 150.0),
                         y: CGFloat(top + incr), width: 300.0, height: 21.0)
    orientation.frame = CGRect(x: CGFloat(size.width / 2.0 - 150.0),
                               y: CGFloat(top + 2 * incr),
                               width: 300.0, height: 21.0)
```

MyView without draw()


```
        // Drawing circles
         let ctx = UIGraphicsGetCurrentContext()
         let table = [(0.9,70.0), (0.8,60.0),(0.7,50.0), (0.6,40.0),
                      (0.5,30.0), (0.3,20.0), (0.0,10.0),]
         for (vert,radius) in table {
             UIColor(red: 1.0, green: CGFloat(vert), blue: 0.0, alpha: 1.0).setFill()
             ctx?.addArc(center: CGPoint(x:size.width / 2.0,
                                         y:size.height / 2.0 + 70.0),
                         radius: CGFloat(radius), startAngle: 0.0,
                         endAngle: CGFloat(Double.pi * 2), clockwise: true)
             ctx?.fillPath();
         }
    }
}
```

Demo (9.7” screen)

What's going on?

Error message

- [Unknown process name] CGContextSetFillColorWithColor: invalid context 0x0. If you want to see the backtrace, please set CG_CONTEXT_SHOW_BACKTRACE environmental variable.

No more error message

- Core Graphics part mus be encapsulated
  - **UIGraphicsBeginImageContext()**
  - **UIGraphicsEndImageContext()**
- You must then fetch an image
  - **UIGraphicsGetImageFromCurrentImageContext()**
  - **And then add this image as a subview**
    be aware that there can be an image already

What's going on?

Error message

- [Unknown process name] CGContextSetFillColorWithColor: invalid context 0x0. If you want to see the backtrace, please set CG_CONTEXT_SHOW_BACKTRACE environmental variable.

No more er

There is another way!

Keep drawing in draw/drawRect

- Core Graphics
  - **UIGraphicsBegi**
  - **UIGraphicsEndImageContext()**
- You must then fetch an image
  - **UIGraphicsGetImageFromCurrentImageContext()**
  - **And then add this image as a subview**
    be aware that there can be an image already

MyView with draw() back


```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
        let screen = UIScreen.main
        let rect = screen.bounds
        let v = MyView(frame: rect)
        self.view = v
        v.drawInFormat(size:rect.size) // Implemented by MyView
    }

    override func viewWillTransition(to size: CGSize,
                                     with coordinator:
UIViewControllerTransitionCoordinator) {
        super.viewWillTransition(to: size, with: coordinator)
        let v = self.view as! MyView // avoids an error
        v.drawInFormat(size:size) // Implemented by MyView
    }
}
```

MyView with draw() back


```
import UIKit

class MyView: UIView {

    fileprivate let myDevice = UIDevice.current
    fileprivate let screen = UIScreen.main

    fileprivate let deviceData = UILabel()
    fileprivate let model = UILabel()
    fileprivate let orientation = UILabel()

    fileprivate var top = 0
    fileprivate var incr = 0

    // Required by Xcode (but unused)
    required init?(coder aDecoder: NSCoder) {
        super.init(coder: aDecoder)
        fatalError("init(coder:) has not been implemented")
    }
```

MyView with draw() back


```
    override init (frame: CGRect) {
        deviceData.textAlignment = .center
        model.textAlignment = .center
        orientation.textAlignment = .center
        model.text = myDevice.model
        super.init(frame: frame)
        // Handled after the call to "super" (we need self here)
        if myDevice.userInterfaceIdiom == .phone && screen.scale == 3.0 {
            deviceData.text = "Large iPhone (\(myDevice.systemName), \
(myDevice.systemVersion))"
            incr = 50
        } else if myDevice.userInterfaceIdiom == .phone {
            deviceData.text = "iPhone/iPod (\(myDevice.systemName), \
(myDevice.systemVersion))"
            incr = 40
        } else {
            deviceData.text = "iPad (\(myDevice.systemName), \
(myDevice.systemVersion))"
            incr = 100
        }
        // Let's deal with the view's elements
        // Handled after the call to "super" (these are super's properties)
        self.backgroundColor = UIColor.white
        self.addSubview(deviceData)
        self.addSubview(model)
        self.addSubview(orientation)
        self.drawInFormat(size: frame.size)
    }
```

MyView with draw() back


```
override func draw(_ rect: CGRect) {
    // Drawing circles
    let ctx = UIGraphicsGetCurrentContext()
    let table = [(0.9,70.0), (0.8,60.0),(0.7,50.0), (0.6,40.0),
                 (0.5,30.0), (0.3,20.0), (0.0,10.0),]
    for (vert,radius) in table {
        UIColor(red: 1.0, green: CGFloat(vert), blue: 0.0, alpha: 1.0).setFill()
        ctx?.addArc(center: CGPoint(x:rect.size.width / 2.0,
                                    y:rect.size.height / 2.0 + 70.0),
                    radius: CGFloat(radius), startAngle: 0.0,
                    endAngle: CGFloat(Double.pi * 2), clockwise: true)
        ctx?.fillPath();
    }
}
```

MyView with draw() back


```
func drawInFormat (size: CGSize) {
    if myDevice.orientation == UIDeviceOrientation.portrait ||
        myDevice.orientation == UIDeviceOrientation.portraitUpsideDown {
        orientation.text = "Portrait orientation"
        top = 100
    } else if myDevice.orientation == UIDeviceOrientation.landscapeLeft ||
        myDevice.orientation == UIDeviceOrientation.landscapeRight {
        orientation.text = "Landscape orientation"
        top = 50
    } else {
        orientation.text = "Device is laid flat"
    }
        // Setting up orientation information
        deviceData.frame = CGRect(x: CGFloat(size.width / 2.0 - 150.0),
                                  y: CGFloat(top), width: 300.0, height: 21.0)
        model.frame = CGRect(x: CGFloat(size.width / 2.0 - 150.0),
                             y: CGFloat(top + incr), width: 300.0, height: 21.0)
        orientation.frame = CGRect(x: CGFloat(size.width / 2.0 - 150.0),
                                   y: CGFloat(top + 2 * incr),
                                   width: 300.0, height: 21.0)
    }
}
```

Demo (9.7” screen)

draw/drawRect only?
some side effect

As a conclusion…

draw/drawRect?

- It is useful if you use Core Graphics…
- But not really
  - **You can generate an image with Core Graphics**

Personal opinion

- View with graphics only (or mainly) on a small device
  - **Use draw/drawRect**
  - **But this method only handles orientation changes**
- Otherwise
  - **draw/drawRect deprecated**

As a conclusion…

draw/drawRect?

- It is useful if you use Core Graphics…
- But not really
  - You can gen… image with Core Graphics

OK!

Let's forget draw/drawRect…

Personal …

- View with gr…
  - Use draw/dr…
  - But this method only handles orientation changes
- Otherwise
  - draw/drawRect deprecated