Life Without Storyboards: Single View Apps

Mobile Computing - iOS

Objectives

- Students will be able to:
  - compare app creation using storyboard and from scratch
  - write apps that do not use storyboard

Rationale

- Storyboard is a tremendously powerful tool that allows us to create apps very quickly. However:

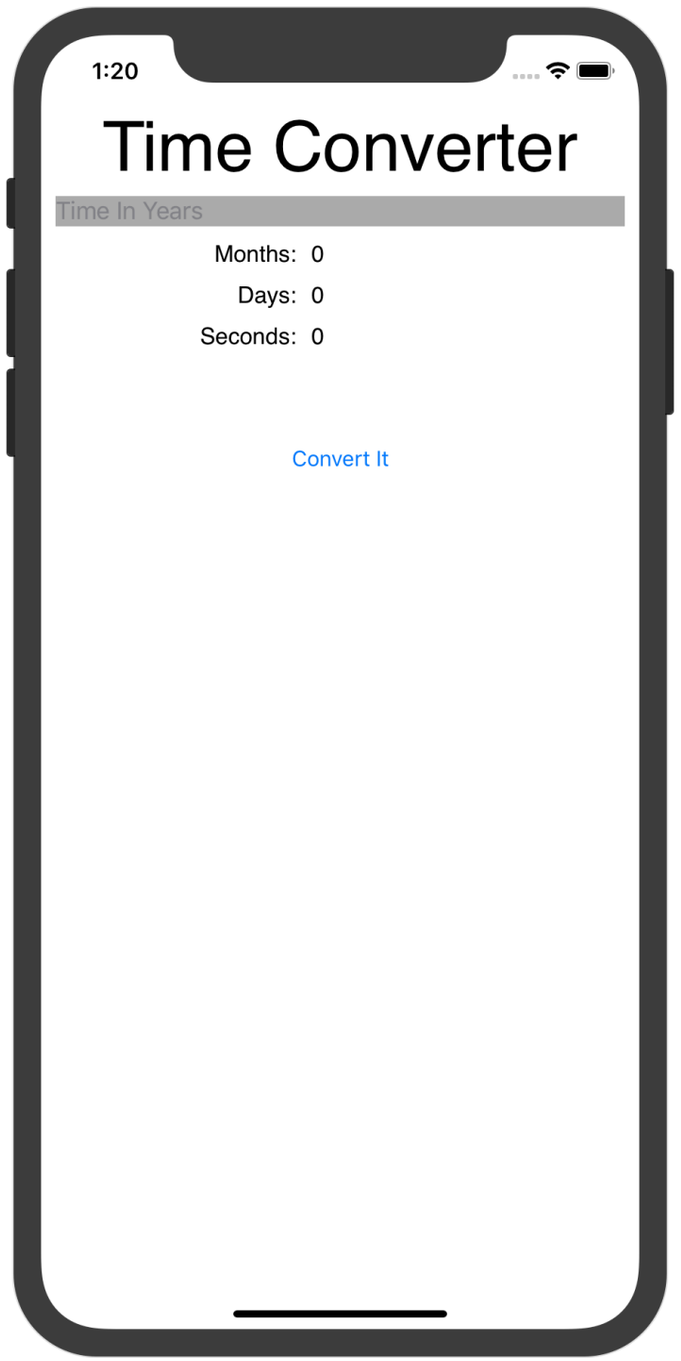
  1. Some developers do not use storyboard, and like to create apps from scratch

  2. You may encounter code that has been written without storyboards

  3. to truly appreciate what storyboard does for you, it is beneficial to see how to create an app without one.

Strategies

- In this presentation we are will recreate Time Converter (created in Outlets and Actions and ViewControllers, Oh, My!):
    - Completely from code
    - By designing the UIViewController's view in storyboard, but instantiating it in code (a technique which may prove useful when you need to dynamically create a UIViewController)



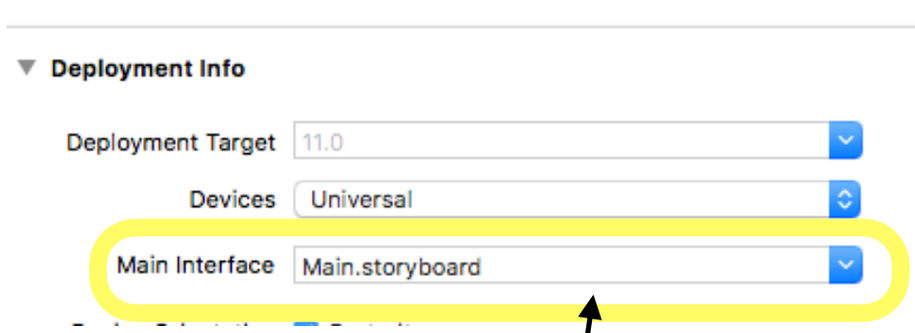
Download and run the Time Converter From Scratch Project, we will use that as the basis for our discussion.

Time Converter Entirely in Code

```
class AppDelegate: UIResponder, UIApplicationDelegate {

 func application(_ application: UIApplication,
        didFinishLaunchingWithOptions launchOptions: [UIApplicationLaunchOptionsKey: Any]?) -> Bool {

        self.window = UIWindow(frame: UIScreen.main.bounds)

        self.window?.rootViewController = ViewController()

        self.window?.makeKeyAndVisible()

        return true
    }
```

In application(_: didFinishLaunchingWithOptions:) we create a window (everything you see must be in a window, instantiate our view controller, and then associate it with that window (so that when the window appears, it knows to ask our view controller for its view)

Note that for this to work in a Single View App, delete Main.storyboard in Target >> General >> Deployment Info



Delete Main.storyboard: a storyboard won't be consulted at app startup

```swift
override func loadView() {
        let view:UIView = UIView()  // first make a view

        view.backgroundColor = UIColor.white // set its color

        let timeConverterLBL:UILabel = UILabel()  // make and configure a label
        timeConverterLBL.text = "Time Converter"
        timeConverterLBL.textAlignment = .center
        timeConverterLBL.font = UIFont(name: "Helvetica", size: 48)
        timeConverterLBL.translatesAutoresizingMaskIntoConstraints = false

        timeTF = UITextField()
        timeTF.placeholder = "Time In Years"
        timeTF.translatesAutoresizingMaskIntoConstraints = false
        timeTF.backgroundColor = UIColor.lightGray

       let monthsTextLBL:UILabel = UILabel()  // make and configure labels
        monthsTextLBL.text = "Months:"
        monthsTextLBL.font = UIFont(name: "Helvetica", size: 16)
        monthsTextLBL.translatesAutoresizingMaskIntoConstraints = false

         let daysTextLBL:UILabel = UILabel()  l
        daysTextLBL.text = "Days:"
        daysTextLBL.font = UIFont(name: "Helvetica", size: 16)
        daysTextLBL.translatesAutoresizingMaskIntoConstraints = false

        let secondsTextLBL:UILabel = UILabel()
        secondsTextLBL.text = "Seconds:"
        secondsTextLBL.font = UIFont(name: "Helvetica", size: 16)
        secondsTextLBL.translatesAutoresizingMaskIntoConstraints = false

        monthsLBL = UILabel()
        monthsLBL.text = "0"
        monthsLBL.font = UIFont(name: "Helvetica", size: 16)
        monthsLBL.translatesAutoresizingMaskIntoConstraints = false

        daysLBL = UILabel()
        daysLBL.text = "0"
        daysLBL.font = UIFont(name: "Helvetica", size: 16)
        daysLBL.translatesAutoresizingMaskIntoConstraints = false

        secondsLBL = UILabel()
        secondsLBL.text = "0"
        secondsLBL.font = UIFont(name: "Helvetica", size: 16)
        secondsLBL.translatesAutoresizingMaskIntoConstraints = false
// continued on the next slide
```

In loadView() we make the UI components, configure them, add them to the view and set up constraints so everything will be in its proper place.

```
let clickMeBTN:UIButton = UIButton(type: UIButtonType.system) as UIButton // notice ... no typecasting
clickMeBTN.setTitle("Convert It", for: .normal)
clickMeBTN.addTarget(self, action: #selector(handleTap), for: UIControlEvents.touchUpInside)
clickMeBTN.translatesAutoresizingMaskIntoConstraints = false // omit this and nothing works!!

view.addSubview(timeConverterLBL) // add it to the view
view.addSubview(timeTF)
view.addSubview(monthsTextLBL) // add it to the view
view.addSubview(daysTextLBL) // add it to the view
view.addSubview(secondsTextLBL) // add it to the view

view.addSubview(monthsLBL) // add it to the view
view.addSubview(daysLBL) // add it to the view
view.addSubview(secondsLBL) // add it to the view

view.addSubview(clickMeBTN)

timeConverterLBL.topAnchor.constraint(equalTo: view.topAnchor, constant: 20.0).isActive = true
timeConverterLBL.leadingAnchor.constraint(equalTo: view.leadingAnchor).isActive = true
timeConverterLBL.trailingAnchor.constraint(equalTo: view.trailingAnchor).isActive = true
timeConverterLBL.centerXAnchor.constraint(equalTo:view.centerXAnchor).isActive = true

timeTF.centerXAnchor.constraint(equalTo: view.centerXAnchor).isActive = true
timeTF.topAnchor.constraint(equalTo: timeConverterLBL.bottomAnchor, constant:6.0).isActive = true
timeTF.leadingAnchor.constraint(equalTo: view.leadingAnchor, constant:10.0).isActive = true

monthsTextLBL.topAnchor.constraint(equalTo:timeTF.bottomAnchor,constant:10.0).isActive = true
monthsTextLBL.rightAnchor.constraint(equalTo:view.centerXAnchor, constant:-30.0).isActive = true
daysTextLBL.topAnchor.constraint(equalTo:monthsTextLBL.bottomAnchor,constant:10.0).isActive = true
daysTextLBL.rightAnchor.constraint(equalTo:view.centerXAnchor, constant:-30.0).isActive = true
secondsTextLBL.topAnchor.constraint(equalTo:daysTextLBL.bottomAnchor,constant:10.0).isActive = true
secondsTextLBL.rightAnchor.constraint(equalTo:view.centerXAnchor, constant:-30.0).isActive = true

monthsLBL.topAnchor.constraint(equalTo:monthsTextLBL.topAnchor).isActive = true
monthsLBL.leftAnchor.constraint(equalTo:monthsTextLBL.rightAnchor, constant:10.0).isActive = true
daysLBL.topAnchor.constraint(equalTo:daysTextLBL.topAnchor).isActive = true
daysLBL.leftAnchor.constraint(equalTo:daysTextLBL.rightAnchor, constant:10.0).isActive = true
secondsLBL.topAnchor.constraint(equalTo:secondsTextLBL.topAnchor).isActive = true
secondsLBL.leftAnchor.constraint(equalTo:secondsTextLBL.rightAnchor, constant:10.0).isActive = true

clickMeBTN.topAnchor.constraint(equalTo: secondsTextLBL.bottomAnchor, constant: 60.0).isActive = true
clickMeBTN.centerXAnchor.constraint(equalTo: view.centerXAnchor).isActive = true
clickMeBTN.leadingAnchor.constraint(equalTo: view.leadingAnchor).isActive = true
clickMeBTN.trailingAnchor.constraint(equalTo: view.trailingAnchor).isActive = true

self.view = view // lastly assign it to the UIViewController's property
}
```

Notice the (huge) amount of code required to set up the constraints so the UI looks right. Storyboard does all this for you 😍

Techy Aside: Time Converter Using Storyboard (a tad)

```
func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplicationLaunchOptionsKey: Any]?) -> Bool {

      self.window = UIWindow(frame: UIScreen.main.bounds)

      let storyboard = UIStoryboard(name: "Main", bundle: nil)
      let viewController = storyboard.instantiateViewController(withIdentifier: "startMeUp")

      self.window?.rootViewController = viewController

      self.window?.makeKeyAndVisible()
      return true
  }


class ViewController: UIViewController {

    @IBOutlet   var timeTF:UITextField!

    @IBOutlet var monthsLBL:UILabel!
    @IBOutlet var daysLBL:UILabel!
    @IBOutlet var secondsLBL:UILabel!

    let monthsPerYear = 12.0
    let daysPerYear = 365.25
    let secondsPerYear = 365.25 * 24.0 * 3600.0

   /* override func loadView() {
        let view:UIView = UIView()  // first make a view
```

Here, we create the ViewController's view in storyboard, and hook up outlets/actions: but rather than designate the ViewController as the initial root view controller, we instead create the storyboard in code, then instantiate the view controller (with identifier startMeUp)