Lecture 6

Autolayout

Attendance

tiny.cc/cis195-att

Today

- Swift Topic: Extensions { }
- Swift Topic: Code Formatting with MARK
- Segues, Navigation Controller, TabBarController
- Live Demo

Extensions

Consult the docs

- *Documentation*

- “Extensions add new functionality to an existing class, structure, enumeration, or protocol type...
  This includes the ability to extend types for which you do not have access to the original source code.”

- **This means we can add functionality to any part of UIKit!**

Syntax

```
extension SomeType {
    // new functionality to add to SomeType goes here
}
```

Syntax

```
extension SomeType {
    // new functionality to add to SomeType goes here
}

extension SomeType: SomeProtocol, AnotherProtocol {
    // implementation of protocol requirements here
}
```

Computed Properties

```
extension Double {

    var km: Double {
        return self * 1000.0
    }

}

let distanceInMeters = 3.km  //equal to 3000.0
```

Computed Properties

```
extension Double {

    var km: Double { return self * 1000.0 }
    var ft: Double { return self / 3.28084 }
}

let distanceInMeters = 3.km  //equal to 3000.0
let inFeet = distanceInMeters.ft  //equal to 9842.52
```

Computed Properties

```
extension Double {
    var km: Double { return self * 1000.0 }
    var m: Double { return self }
    var cm: Double { return self / 100.0 }
    var mm: Double { return self / 1000.0 }
    var ft: Double { return self / 3.28084 }
}

let threeFeet = 3.ft
print("Three feet is \(threeFeet) meters")
// Prints "Three feet is 0.914399970739201 meters"
```

Add Initializers to Existing Types

```
struct Rect {
    var topLeft = Point()
    var size = Size()
}
let myRect = Rect(topLeft: Point(x: 2.0, y: 2.0),
                  size:   Size(width: 5.0, height: 5.0))


// Extend rect with a new init() that takes a “center” rather than topLeft
extension Rect {
    init(center: Point, size: Size) {
        let topLeftX = center.x - (size.width / 2)
        let topLeftY = center.y - (size.height / 2)
        self.init(topLeft: Point(x: topLeftX, y: topLeftY), size: size)
    }
}
```

Add Methods to Existing Types

```
extension Int {
    func repetitions(task: () -> Void) {
        for _ in 0..<self {
            task()
        }
    }
}
```

Add Methods to Existing Types

```
extension Int {
    func repetitions(task: () -> Void) {
        for _ in 0..<self {
            task()
        }
    }
}
```

Notice this is a closure type.
Essentially an uncalled function

We call the closure with “task()“

Add Methods to Existing Types

```
extension Int {
    func repetitions(task: () -> Void) {
        for _ in 0..<self {
            task()
        }
    }
}

3.repetitions {
    print("Hello!")
}
// Hello!
// Hello!
// Hello!
```

INT.repetitions(CLOSURE)

Executes the CLOSURE, INT times

Mutating Self in a Method Extension

```
extension Int {
    mutating func square() {
        self = self * self
    }
}
```

Mutating Self in a Method Extension

```
extension Int {
    mutating func square() {
        self = self * self
    }
}
var someInt = 3
someInt.square()
// someInt is now 9
```

Adding Subscript Support with Extensions

Recall the subscript syntax we use with Arrays and Dictionaries:

```
var numberOfLegs = ["spider": 8, "ant": 6, "cat": 4]
numberOfLegs["bird"] = 2
```

Wouldn't it be cool if we could subscript an Int? For example:

123456789[0] returns 9
123456789[2] returns 7

Adding Subscript Support with Extensions

```
extension Int {

    subscript(digitIndex: Int) -> Int {

        var decimalBase = 1
        for _ in 0..<digitIndex {
            decimalBase *= 10
        }
        return (self / decimalBase) % 10

    }

}
```

Adding Subscript Support with Extensions

```
extension Int {
    subscript(digitIndex: Int) -> Int {
        var decimalBase = 1
        for _ in 0..<digitIndex {
            decimalBase *= 10
        }
        return (self / decimalBase) % 10
    }
}
746381295[0]
// returns 5
746381295[1]
// returns 9
746381295[2]
// returns 2
746381295[8]
// returns 7
```

Adding Nested Types with Extensions

```
extension Int {
    enum Kind {
        case negative, zero, positive
    }

    ...
}
```

Adding Nested Types with Extensions

```
extension Int {
    enum Kind {
        case negative, zero, positive
    }
    var kind: Kind {
        switch self {
        case 0:
            return .zero
        case let x where x > 0:
            return .positive
        default:
            return .negative
        }
    }
}
```

Adding Nested Types with Extensions

```
func printIntegerKinds(_ numbers: [Int]) {
    for number in numbers {
        switch number.kind {
        case .negative:
            print("- ", terminator: "")
        case .zero:
            print("0 ", terminator: "")
        case .positive:
            print("+ ", terminator: "")
        }
    }
    print("")
}
printIntegerKinds([3, 19, -27, 0, -6, 0, 7])
// Prints "+ + - 0 - 0 + "
```

Adding Nested Types with Extensions

```
if number < 0 {
    print(“\(number) is negative")
}

if number.negative {
    print(“\(number) is negative")
}
```

Q: How could we add this syntax?

Adding Nested Types with Extensions

```
if number < 0 {
    print(“\(number) is negative")
}

if number.negative {
    print(“\(number) is negative")
}
```

Q: How could we add this syntax?

A: With a computed property for each Kind case, which checks if self.kind == .negative/pos/zero

Adding Nested Types with Extensions

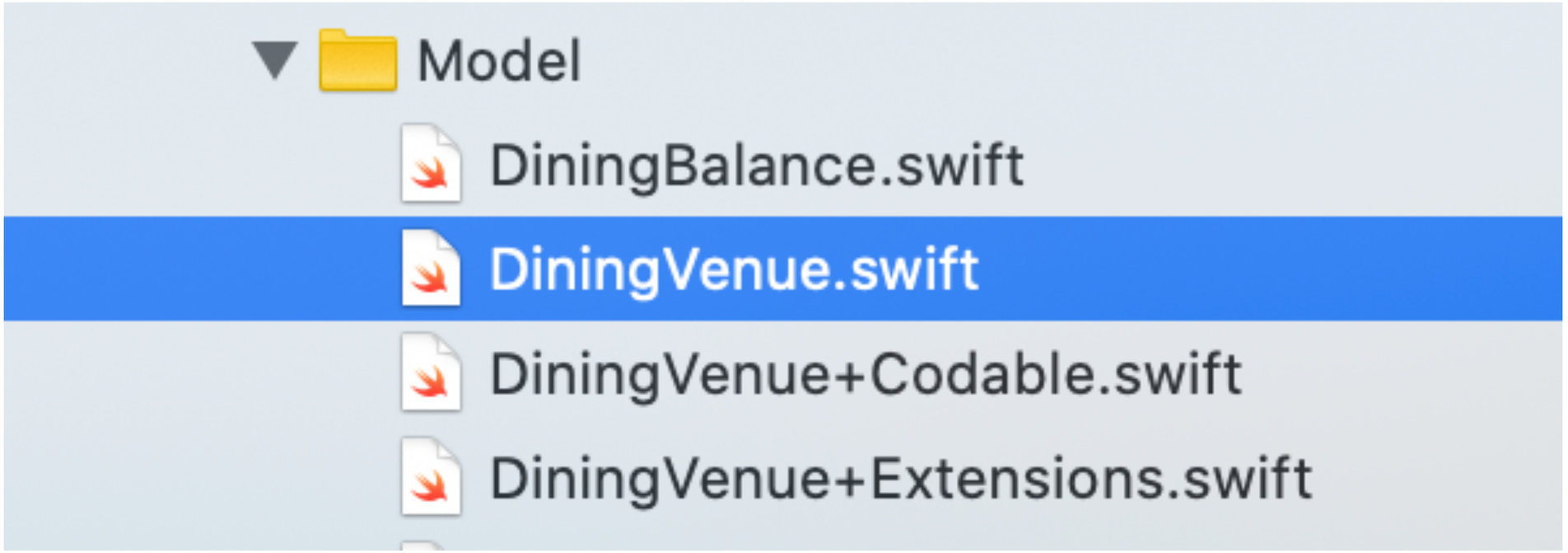
```
extension Int {
    enum Kind { case negative, zero, positive }

    // … implementation for the self.kind property …

    var negative: Kind { return self.kind == .negative }

    var zero: Kind { return self.kind == .zero }

    var positive: Kind { return self.kind == .positive }
}
```

Code Separation with Extensions

```
// In MyViewController.swift
class MyViewController {
    . . .
}
```

```
// In MyViewController+TableViewDelegate.swift
extension MyViewController: UITableViewDelegate {
    func cellForRowAt(:)
    . . .
}
```

Code Separation with Extensions



```swift
extension UIColor {

    // MARK: - UI Palette
    static let navigation = UIColor(named: "navigation")!
    static let uiCardBackground = UIColor(named: "uiCardBackground")!
    static let uiGroupedBackground = UIColor(named: "uiGroupedBackground")!
    static let uiGroupedBackgroundSecondary = UIColor(named: "uiGroupedBackgroundSecondary")!
    static let uiBackground = UIColor(named: "uiBackground")!
    static let uiBackgroundSecondary = UIColor(named: "uiBackgroundSecondary")!
    static let labelPrimary = UIColor(named: "labelPrimary")!
    static let labelSecondary = UIColor(named: "labelSecondary")!
    static let labelTertiary = UIColor(named: "labelTertiary")!
    static let labelQuaternary = UIColor(named: "labelQuaternary")!

    // MARK: - Primary Palette
    static var baseDarkBlue = UIColor(named: "baseDarkBlue")!
    static let baseLabsBlue = UIColor(named: "baseLabsBlue")!

    // MARK: - Neutral Palette
    static var grey1 = UIColor(named: "grey1")!
    static var grey2 = UIColor(named: "grey2")!
    static var grey3 = UIColor(named: "grey3")!
    static var grey4 = UIColor(named: "grey4")!
    static var grey5 = UIColor(named: "grey5")!
    static var grey6 = UIColor(named: "grey6")!
```

We can also use extensions to define custom UIColors and UIFonts that should be used across the app.

Credit: Swift Docs

https://docs.swift.org/swift-book/LanguageGuide/Extensions.html

Code Style

Compiler Directives

```
// TODO: This isn't finished yet

// FIXME: This code is awful.

// FIXME: - This code is really broken -

// MARK: UITableViewDataSource methods

// MARK: -
```

[Credit: HackingWithSwift]

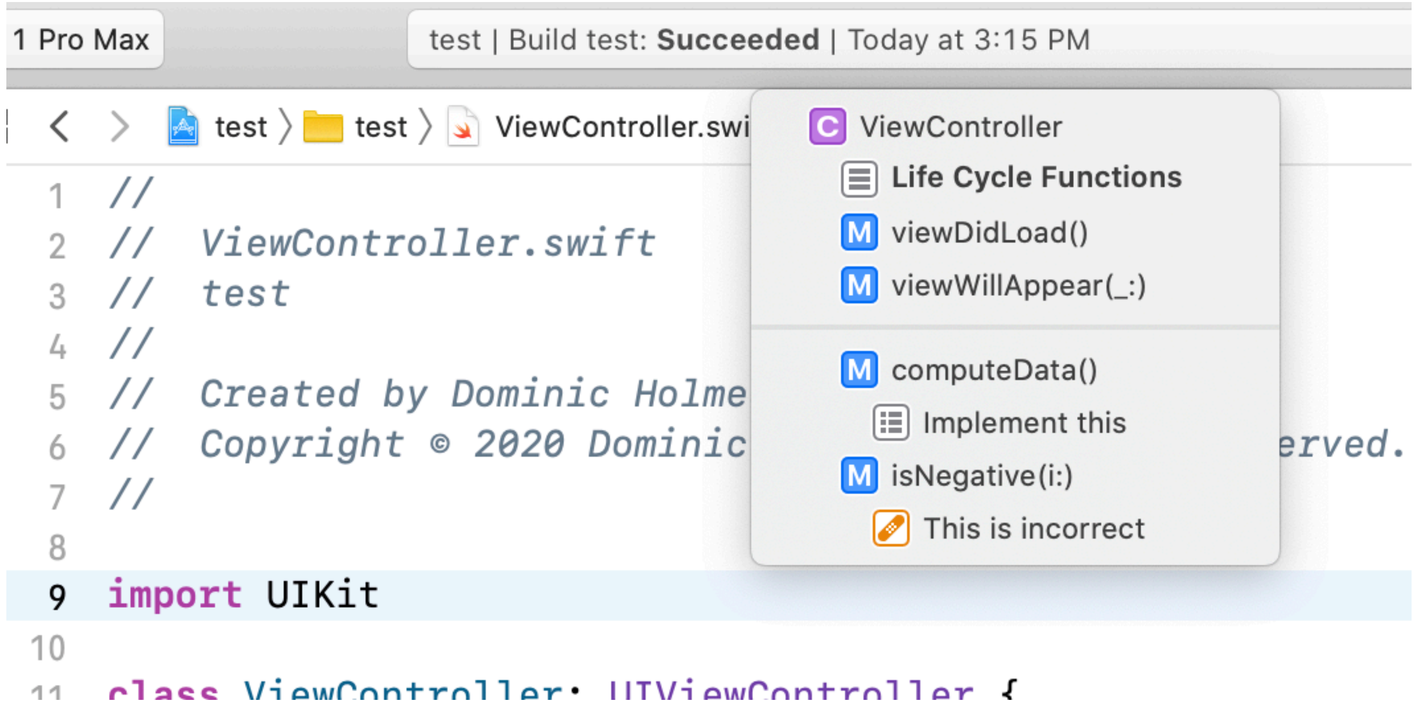
Compiler Directives

```
// TODO: This isn't finished yet

// FIXME: This code is awful.

// FIXME: - This code is really broken -

// MARK: UITableViewDataSource methods

// MARK: -
```

Credit: HackingWithSwift

```
import UIKit

class ViewController: UIViewController {

    // MARK: Life Cycle Functions
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
    }

    // MARK: -

    func computeData() {
        // TODO: Implement this
    }

    func isNegative(i: Int) -> Bool {
        // FIXME: This is incorrect
        return i > 0
    }
}
```

```swift
import UIKit

class ViewController: UIViewController {

    // MARK: Life Cycle Functions
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
    }

    // MARK: -

    func computeData() {
        // TODO: Implement this
    }

    func isNegative(i: Int) -> Bool {
        // FIXME: This is incorrect
        return i > 0
    }
}
```



```swift
extension UIColor {

    // MARK: - UI Palette
    static let navigation = UIColor(named: "navigation")!
    static let uiCardBackground = UIColor(named: "uiCardBackground")!
    static let uiGroupedBackground = UIColor(named: "uiGroupedBackground")!
    static let uiGroupedBackgroundSecondary = UIColor(named: "uiGroupedBackgroundSecondary")!
    static let uiBackground = UIColor(named: "uiBackground")!
    static let uiBackgroundSecondary = UIColor(named: "uiBackgroundSecondary")!
    static let labelPrimary = UIColor(named: "labelPrimary")!
    static let labelSecondary = UIColor(named: "labelSecondary")!
    static let labelTertiary = UIColor(named: "labelTertiary")!
    static let labelQuaternary = UIColor(named: "labelQuaternary")!

    // MARK: - Primary Palette
    static var baseDarkBlue = UIColor(named: "baseDarkBlue")!
    static let baseLabsBlue = UIColor(named: "baseLabsBlue")!

    // MARK: - Neutral Palette
    static var grey1 = UIColor(named: "grey1")!
    static var grey2 = UIColor(named: "grey2")!
    static var grey3 = UIColor(named: "grey3")!
    static var grey4 = UIColor(named: "grey4")!
    static var grey5 = UIColor(named: "grey5")!
    static var grey6 = UIColor(named: "grey6")!
```

In our UIColor extensions file, we use //MARK to separate the different color palettes of the app.

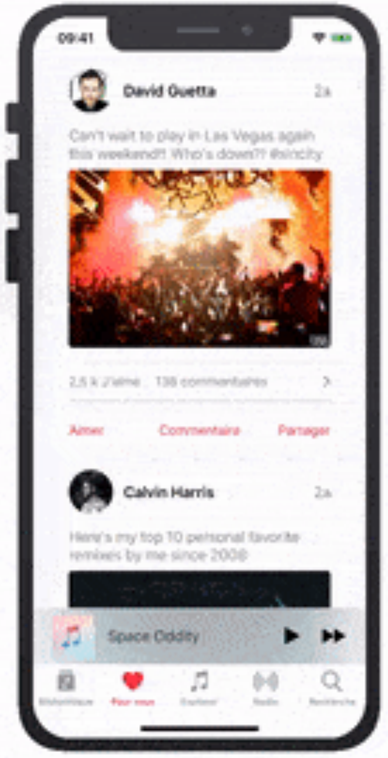
A Fantastic Swift Style Guide:

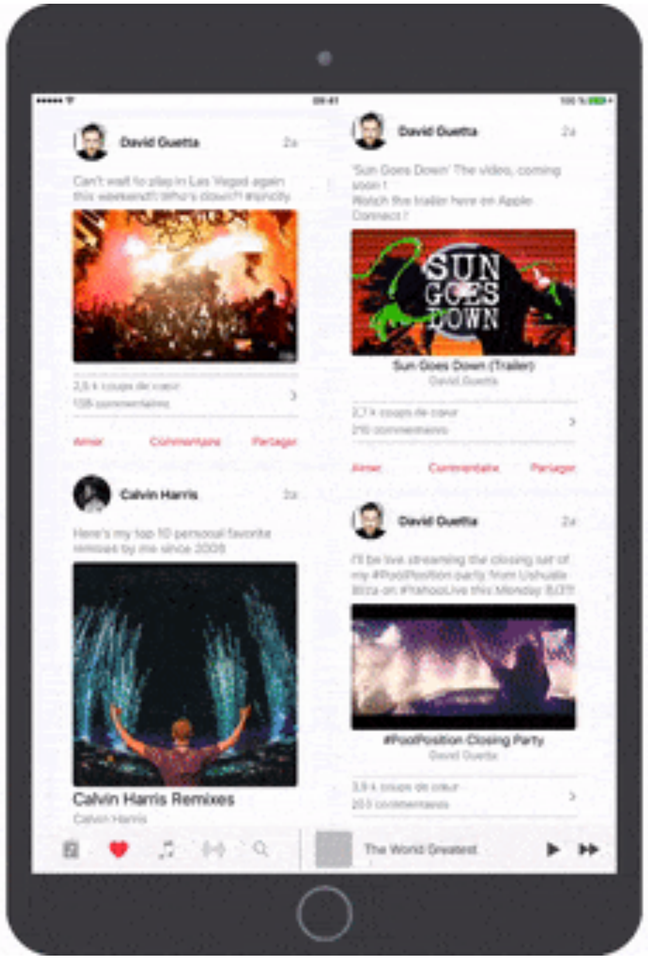
https://github.com/raywenderlich/swift-style-guide

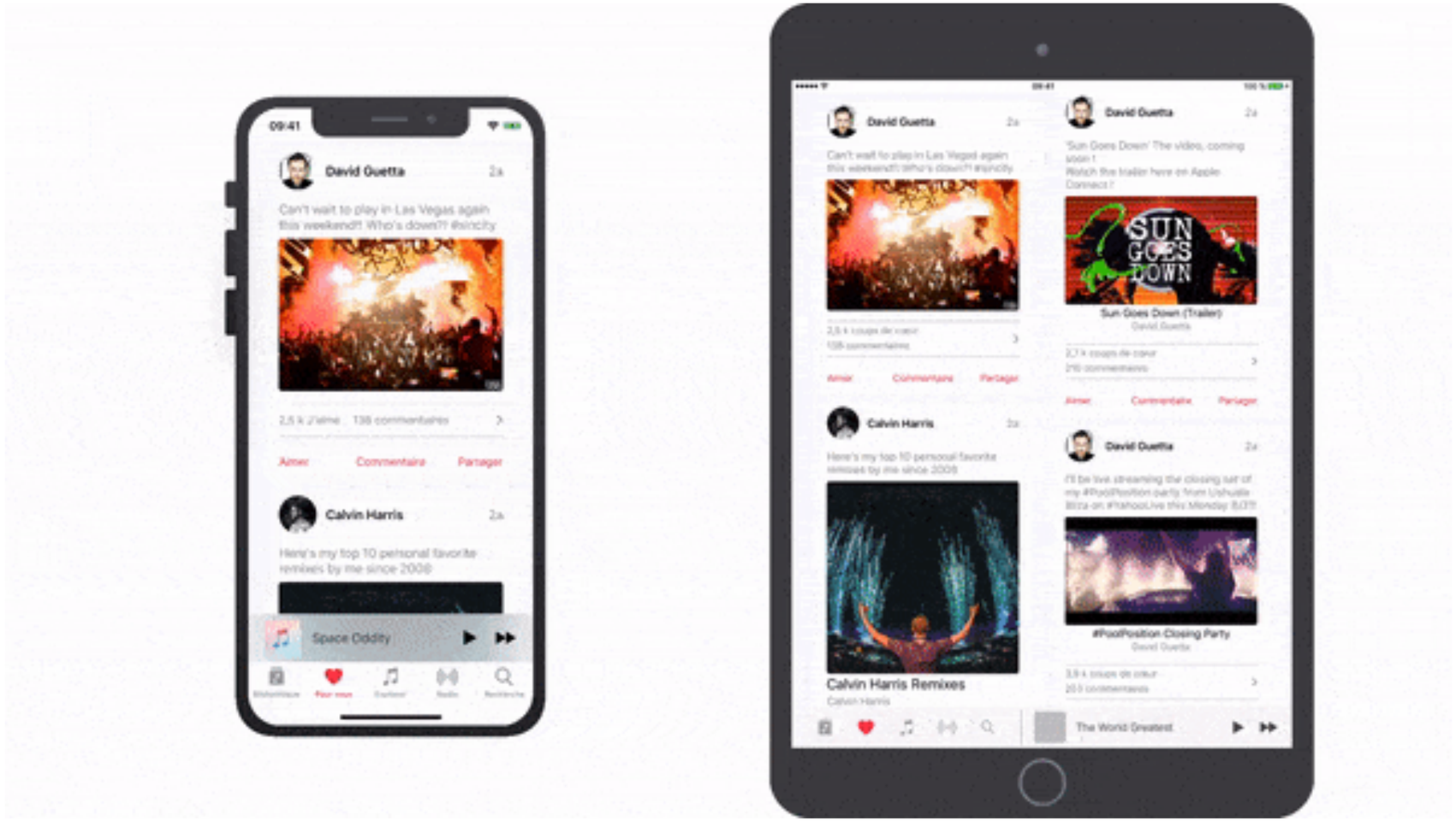
Segues

Segue Types

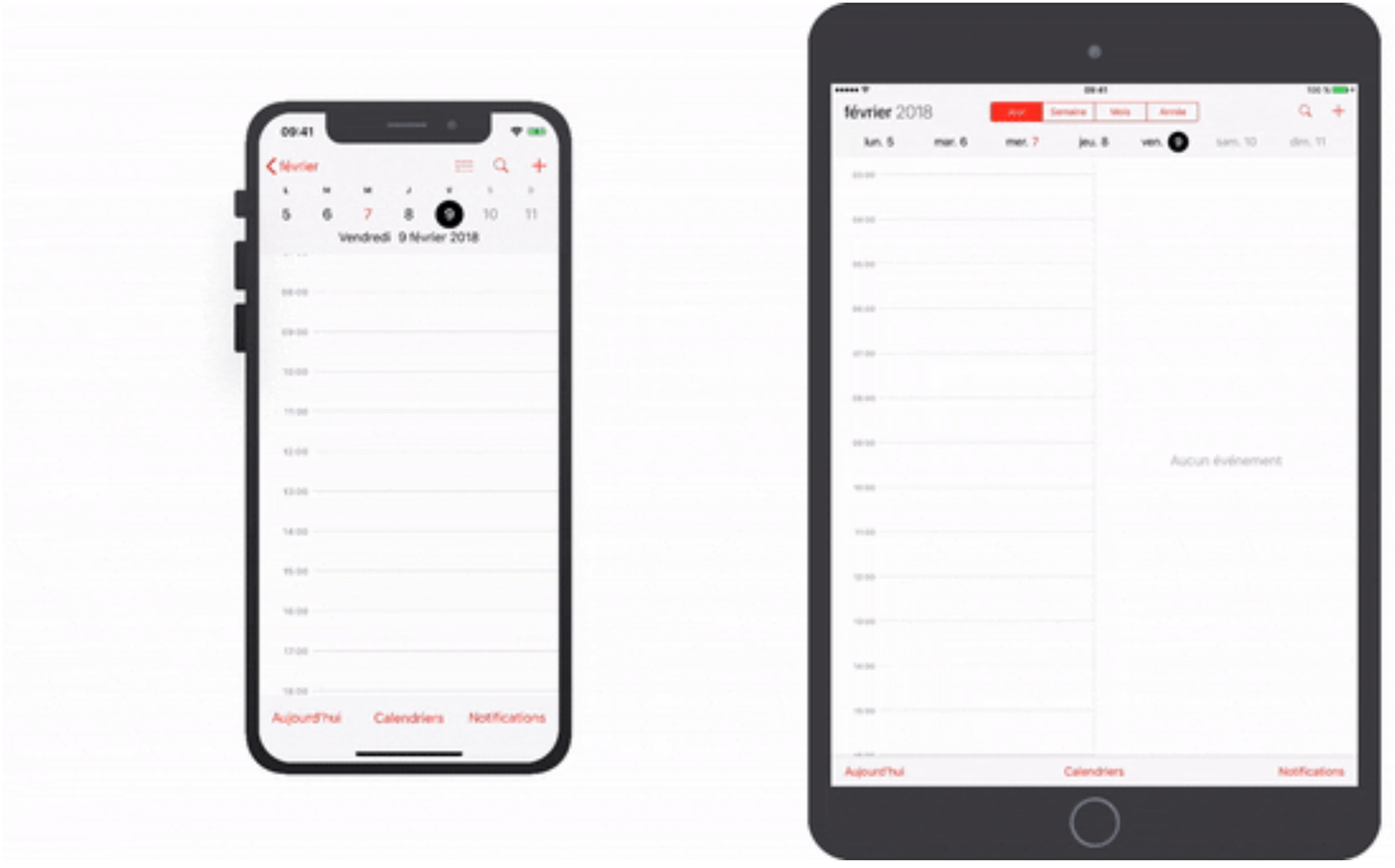
- Show (Push)
  - Modally presented if outside of a Nav Controller
  - Nav Controller overrides this for a “sideways” push transition
- Present Modally
  - Can customize this for different transition styles
- Present as Popover
  - Shown as a popover
  - This looks like a Modal segue on iPhone — but different on iPad!
- Show Detail (only used in a UISplitViewController)

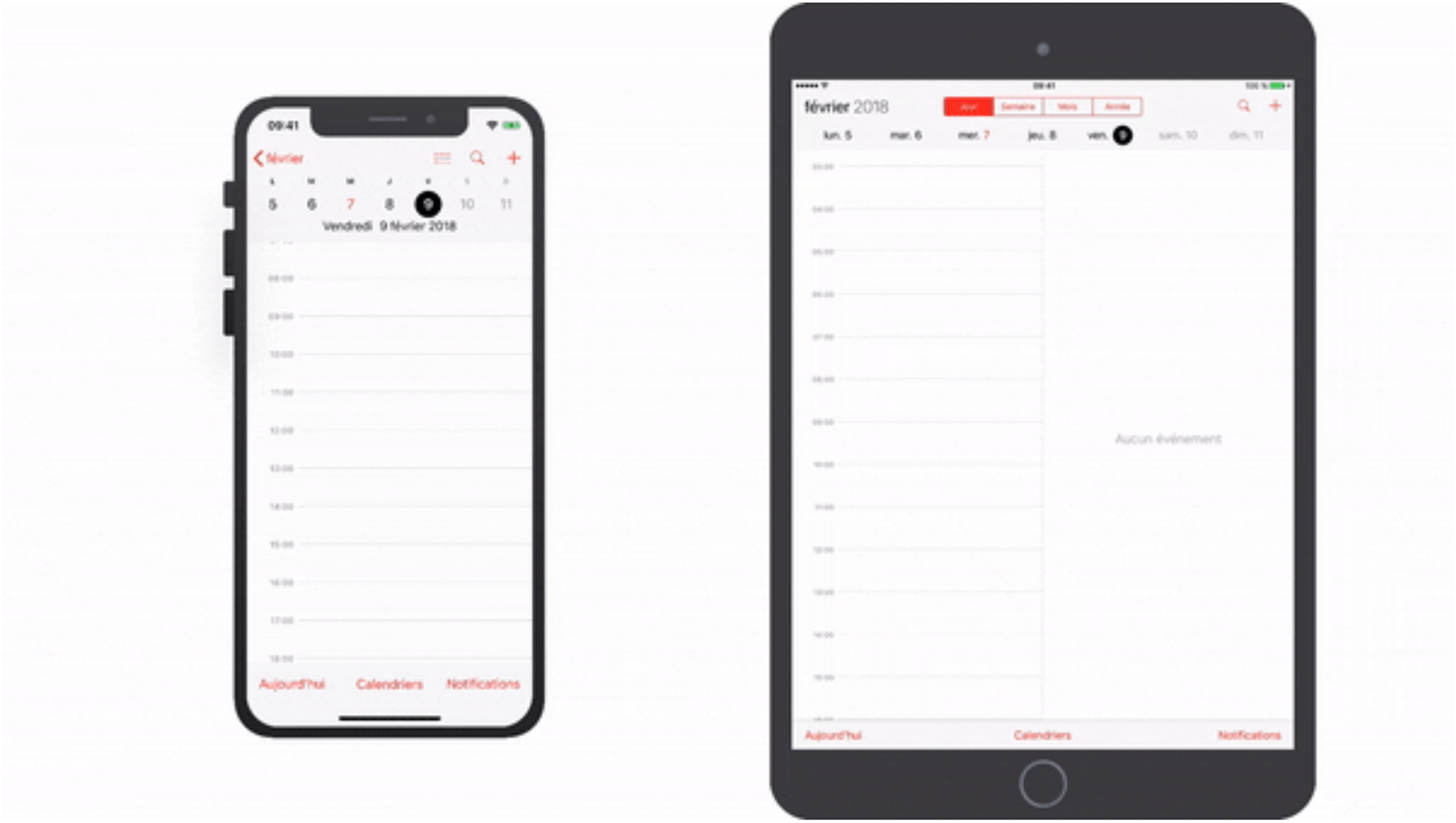

09:41
David Guetta
2a
Can't wait to play in Las Vegas again this weekend!! Who's down?? #sincity
2,5 k J'aime 138 commentaires
Aimer
Commentaire
Partager
Calvin Harris
2a
Here's my top 10 personal favorite remixes by me since 2006
Space Oddity


David Guetta
2a
Can't wait to play in Las Vegas again this weekend!! Who's down?? #sincity
2,5 k coups de coeur
138 commentaires
Aimer
Commentaire
Partager
Calvin Harris
2a
Here's my top 10 personal favorite remixes by me since 2006
Calvin Harris Remixes
Calvin Harris
David Guetta
2a
'Sun Goes Down' The video, coming soon !
Watch the trailer here on Apple Connect !
SUN GOES DOWN
Sun Goes Down (Trailer)
David Guetta
2,7 k coups de coeur
210 commentaires
Aimer
Commentaire
Partager
David Guetta
2a
I'll be live streaming the closing set of my #PoolPosition party from Ushuaïa Ibiza on #YahooLive this Monday 8.07!
#PoolPosition Closing Party
David Guetta
3,9 k coups de coeur
203 commentaires
The World Greatest

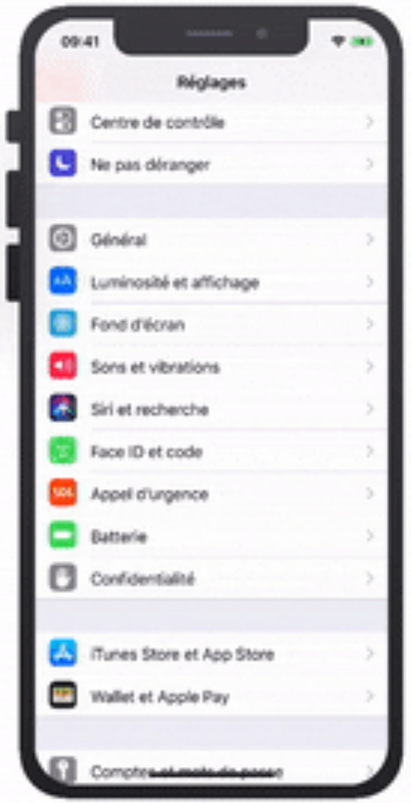

09:41
David Guetta
2a
Can't wait to play in Las Vegas again this weekend!! Who's down?? #sincity
Aimer
Commentaire
Partager
Calvin Harris
2a
Here's my top 10 personal favorite remixes by me since 2006
Space Oddity
David Guetta
2a
Can't wait to play in Las Vegas again this weekend!! Who's down?? #sincity
Calvin Harris
2a
Here's my top 10 personal favorite remixes by me since 2006
Calvin Harris Remixes
Calvin Harris
David Guetta
2a
'Sun Goes Down' The video, coming soon !
Watch the trailer here on Apple Connect !
SUN GOES DOWN
Sun Goes Down (Trailer)
David Guetta
David Guetta
2a
#PoolPosition Closing Party
The World Greatest

Show

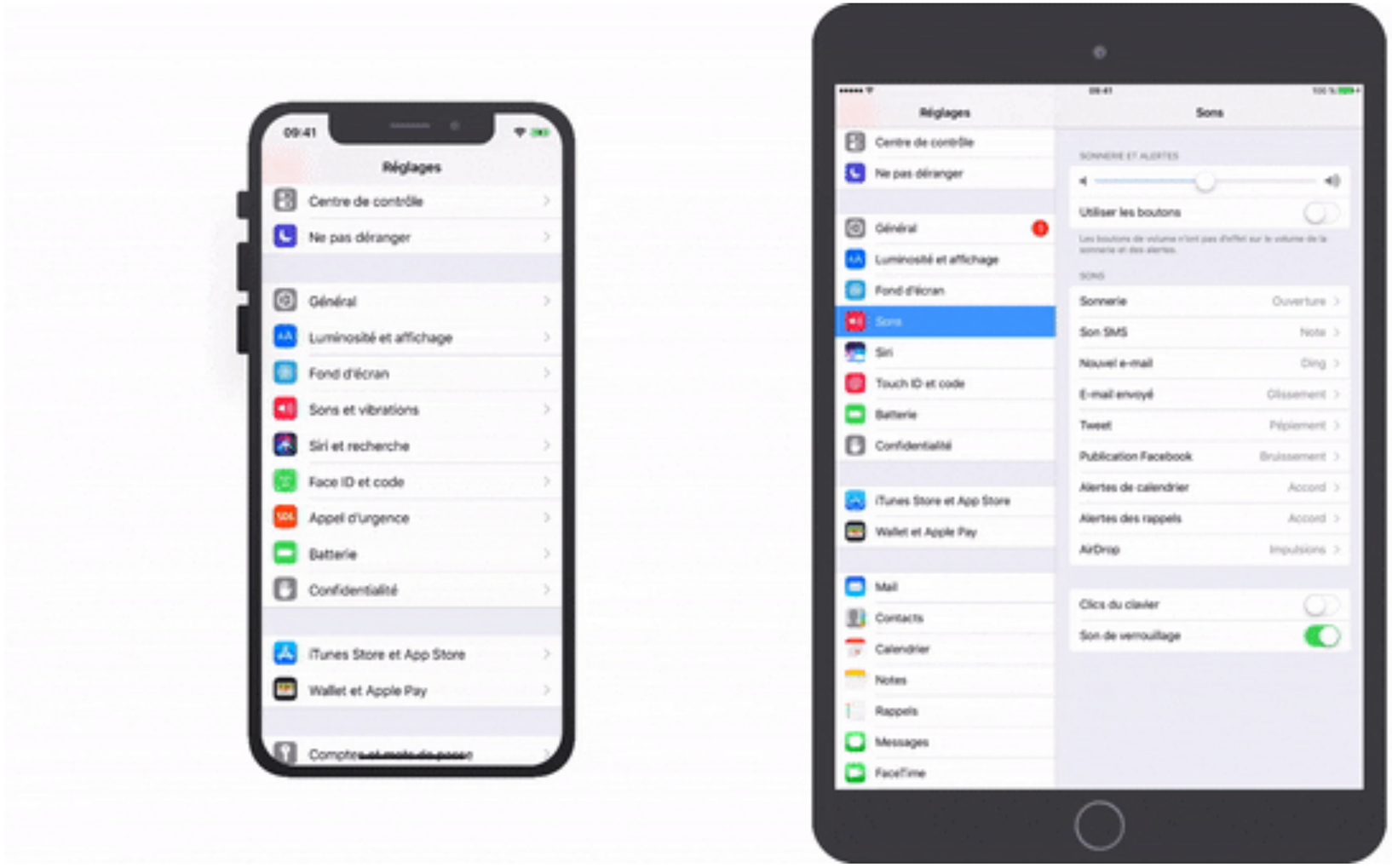

09:41
février
5 6 7 8 9 10 11
Vendredi 9 février 2018
Aujourd'hui
Calendriers
Notifications
février 2018
lun. 5
mar. 6
mer. 7
jeu. 8
ven. 9
Aucun événement
Aujourd'hui
Calendriers
Notifications


09:41
février
Vendredi 9 février 2018
Aujourd'hui
Calendriers
Notifications
février 2018
Aujourd'hui
Calendriers
Notifications

Popover

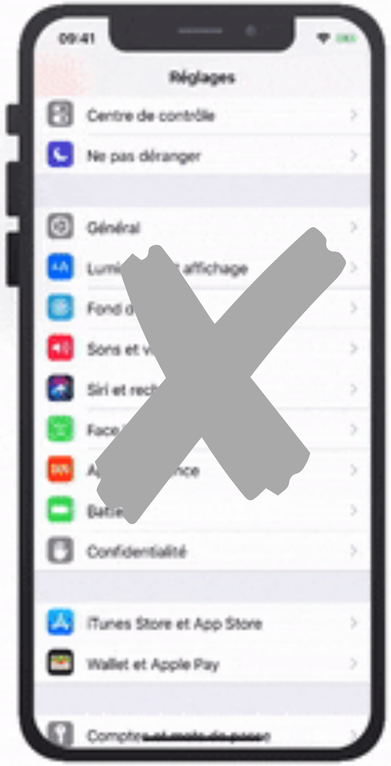

09:41
Réglages
Centre de contrôle
Ne pas déranger
Général
Luminosité et affichage
Fond d'écran
Sons et vibrations
Siri et recherche
Face ID et code
Appel d'urgence
Batterie
Confidentialité
iTunes Store et App Store
Wallet et Apple Pay

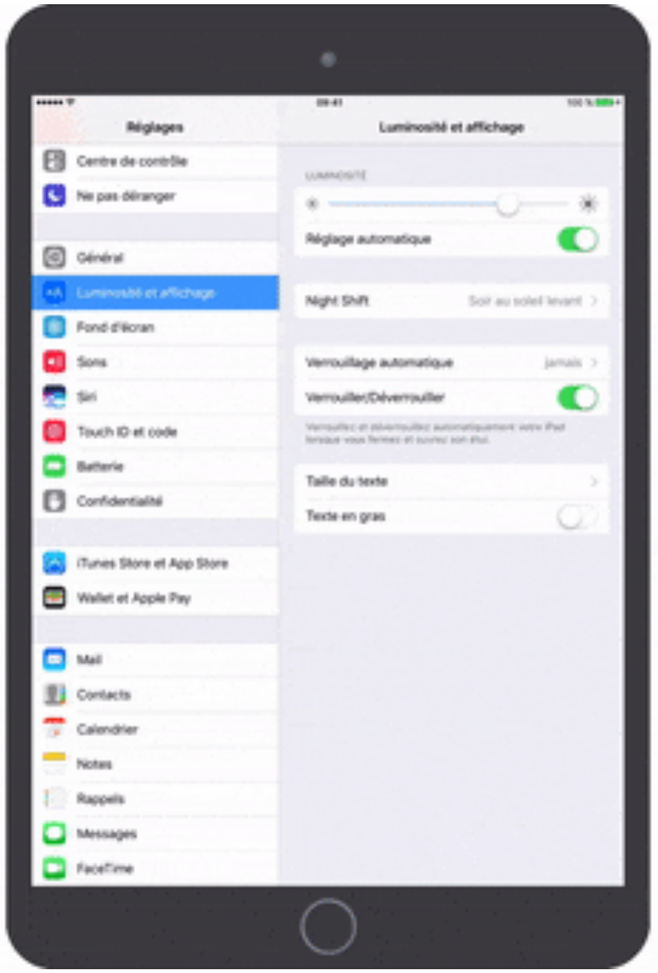

Réglages
Sons
Centre de contrôle
Ne pas déranger
Général
Luminosité et affichage
Fond d'écran
Sons
Siri
Touch ID et code
Batterie
Confidentialité
iTunes Store et App Store
Wallet et Apple Pay
Mail
Contacts
Calendrier
Notes
Rappels
Messages
FaceTime
SONNERIE ET ALERTES
Utiliser les boutons
SONS
Sonnerie
Ouverture
Son SMS
Note
Nouvel e-mail
Ding
E-mail envoyé
Glissement
Tweet
Pépiement
Publication Facebook
Bruissement
Alertes de calendrier
Accord
Alertes des rappels
Accord
AirDrop
Impulsions
Clics du clavier
Son de verrouillage



Present Modally

GIF is incorrect here





Show Detail

Passing Data with Segues

- prepareForSegue() is your best friend
  - *From the Docs: "The default implementation of this method does nothing.* ***Subclasses override this method and use it to configure the new view controller prior to it being displayed.*** *The segue object contains information about the transition, including* ***references to both view controllers that are involved****. Because segues can be triggered from multiple sources, you can use the information in the segue and sender parameters to disambiguate between different logical paths in your app."*

prepareForSegue:sender:

Notifies the view controller that a segue is about to be performed.

Declaration

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue
                 sender:(id)sender;
```

Parameters

segue
The segue object containing information about the view controllers involved in the segue.

sender
The object that initiated the segue. You might use this parameter to perform different actions based on which control (or other object) initiated the segue.

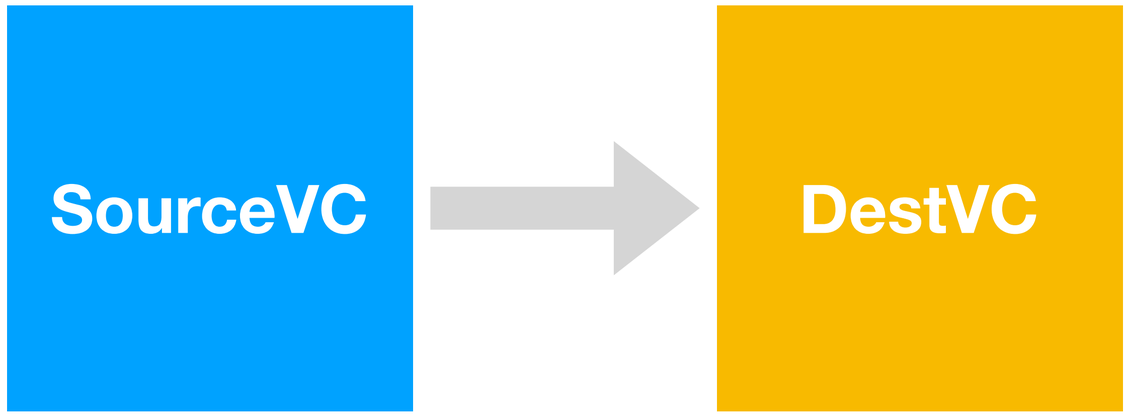
Passing Data

- prepareForSegue() is your best friend
  - Inside this function (called from the **source view controller** of your segue), you get a **reference** to the **destination** of your segue
  - `segue.destination as? YourVCType`

Passing Data

- Inside this function (called from the **source view controller** of your segue), you get a **reference** to the **destination** of your segue

```
let dest = segue.destination as? DestVC
```



Special Case: Segue to a Navigation Controller

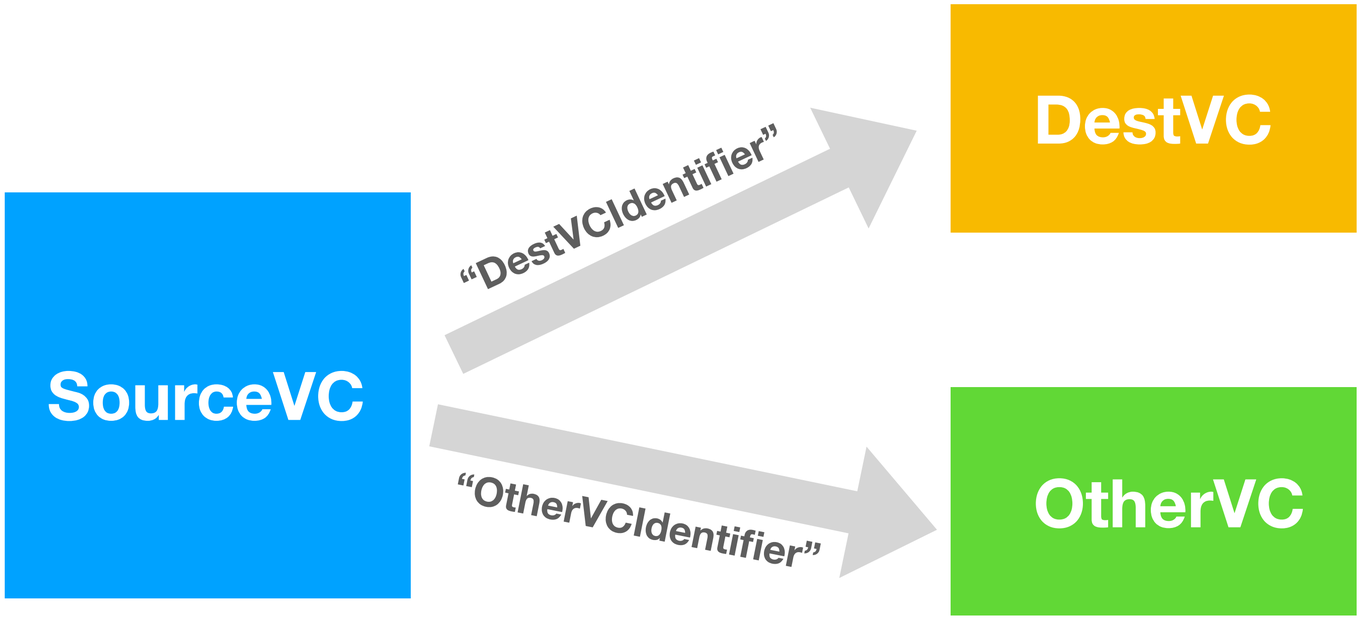
```
let navVC = segue.destination as?
UINavigationController

let dest = navVC.topViewController as?
DestinationVC
```

Special Case: Segue to a Navigation Controller

Ex: We present a modal segue to a view that is embedded in a nav controller. In this case, **the destination is of type UINavigationController.** To access the actual vc you first get the destination, cast it to type UINavigationController, and then use .topViewController.

Identifying Segues

- We tell segues apart with the `segue.identifier`
  - *We can also trigger them from code* with the method `performSegue(withIdentifier:String)`



Stopping a Segue

- `shouldPerformSegue(withIdentifier)`
  - Called before performSegue(...)
  - Just return False if you don’t want it to happen
  - Both of these methods are part of UIViewController

App 5

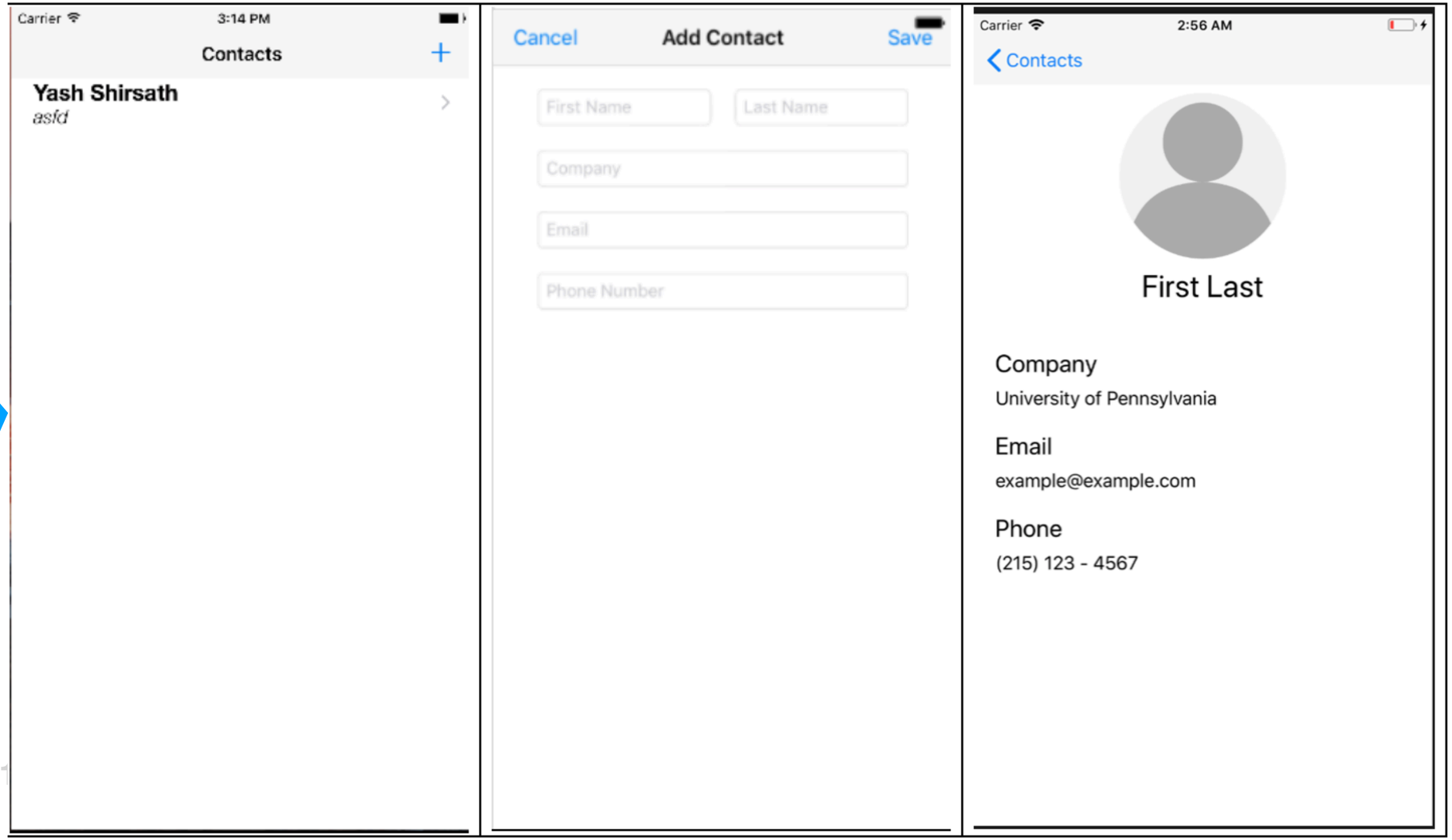
App 5: Contacts

- *Our first "full app"*
  - **Our toolbox is growing!**
- You'll combine things you learned in Apps 1, 2, 3, and 4
- Consists of
  - A table view of contacts w/ swipe-to-delete
  - A segue to a screen for adding a new contact
  - ALL VIEWS resize correctly (for the remainder of the course)

App 5: Contacts

App 3!

App 2, kinda!



Live Demo: Segues

Due Before Next Class

- App 5: Contacts
- Tutorial 5: Networking

Links

- Survey: tiny.cc/cis195-att
- Piazza: tiny.cc/cis195-piazza