Lecture 15 Demo Code:

FaceIt Segues

Objective

Included below is the source code for the demo in lecture. It is provided under the same Creative Commons licensing as the rest of CS193p's course materials. Code unchanged from previous versions of FaceIt is grayed out. Only a snippet of FaceViewController is included since only one line of code was changed. And here is the complete project.

```
//
//  ExpressionEditorViewController.swift
//  FaceIt
//
//  Created by CS193p Instructor on 3/6/17.
//  Copyright © 2017 Stanford University. All rights reserved.
//

import UIKit

class ExpressionEditorViewController: UITableViewController, UITextFieldDelegate
{
    // MARK: (Read Only) Model

    var name: String {
        return nameTextField?.text ?? ""
    }

    var expression: FacialExpression {
        return FacialExpression(
            eyes: eyeChoices[eyeControl?.selectedSegmentIndex ?? 0],
            mouth: mouthChoices[mouthControl?.selectedSegmentIndex ?? 0]
        )
    }

    private let eyeChoices = [FacialExpression.Eyes.open, .closed, .squinting]
    private let mouthChoices = [FacialExpression.Mouth.frown, .smirk, .neutral, .grin, .smile]
```

```
    // MARK: User Interface Connectivity

    @IBAction func updateFace() {
        faceViewController?.expression = expression
    }

    @IBOutlet weak var nameTextField: UITextField!
    @IBOutlet weak var eyeControl: UISegmentedControl!
    @IBOutlet weak var mouthControl: UISegmentedControl!

    private var faceViewController: BlinkingFaceViewController?

    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
        if segue.identifier == "Embed Face" {
            faceViewController = segue.destination as? BlinkingFaceViewController
            faceViewController?.expression = expression
        }
    }

    @IBAction func cancel(_ sender: UIBarButtonItem) {
        presentingViewController?.dismiss(animated: true)
    }

    // MARK: View Controller Lifecycle

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
        if let popoverPresentationController = navigationController?.popoverPresentationController {
            if popoverPresentationController.arrowDirection != .unknown {
                navigationItem.leftBarButtonItem = nil
            }
        }
        var size = tableView.minimumSize(forSection: 0)
        size.height -= tableView.heightForRow(at: IndexPath(row: 1, section: 0))
        size.height += size.width
        preferredContentSize = size
    }

    // MARK: UITableViewDelegate

    override func tableView(_ tableView: UITableView, heightForRowAt indexPath: IndexPath) -> CGFloat {
        if indexPath.row == 1 {
            return tableView.bounds.size.width
        } else {
            return super.tableView(tableView, heightForRowAt: indexPath)
        }
    }

    // MARK: UITextFieldDelegate

    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        textField.resignFirstResponder()
        return true
    }
}
```

```
extension UITableView
{
    func minimumSize(forSection section: Int) -> CGSize {
        var width: CGFloat = 0
        var height : CGFloat = 0
        for row in 0..<numberOfRows(inSection: section) {
            let indexPath = IndexPath(row: row, section: section)
            if let cell = cellForRow(at: indexPath) ?? dataSource?.tableView(self, cellForRowAt: indexPath) {
                let cellSize = cell.contentView.systemLayoutSizeFitting(UILayoutFittingCompressedSize)
                width = max(width, cellSize.width)
                height += heightForRow(at: indexPath)
            }
        }
        return CGSize(width: width, height: height)
    }

    func heightForRow(at indexPath: IndexPath? = nil) -> CGFloat {
        if indexPath != nil, let height = delegate?.tableView?(self, heightForRowAt: indexPath!) {
            return height
        } else {
            return rowHeight
        }
    }
}
```

```
//
//  ViewController.swift
//  FaceIt
//
//  Created by CS193p Instructor on 1/23/17.
//  Copyright © 2017 Stanford University. All rights reserved.
//

class FaceViewController: UIViewController
{
    func updateUI()
    {
        switch expression.eyes {
        case .open:
            faceView?.eyesOpen = true
        case .closed:
            faceView?.eyesOpen = false
        case .squinting:
//          faceView?.eyesOpen = false
            break
        }
        faceView?.mouthCurvature = mouthCurvatures[expression.mouth] ?? 0.0
    }
}
```

```
//
//  BlinkingFaceViewController.swift
//  FaceIt
//
//  Created by CS193p Instructor on 2/27/17.
//  Copyright © 2017 Stanford University. All rights reserved.
//

import UIKit

class BlinkingFaceViewController: FaceViewController
{
    var blinking = false {
        didSet {
            blinkIfNeeded()
        }
    }

    override func updateUI() {
        super.updateUI()
        blinking = expression.eyes == .squinting
    }

    private struct BlinkRate {
        static let closedDuration: TimeInterval = 0.4
        static let openDuration: TimeInterval = 2.5
    }

    private var canBlink = false
    private var inABlink = false

    private func blinkIfNeeded() {
        if blinking && canBlink && !inABlink {
            faceView.eyesOpen = false
            inABlink = true
            Timer.scheduledTimer(withTimeInterval: BlinkRate.closedDuration, repeats: false) { [weak self] timer in
                self?.faceView.eyesOpen = true
                Timer.scheduledTimer(withTimeInterval: BlinkRate.openDuration, repeats: false) { [weak self] timer in
                    self?.inABlink = false
                    self?.blinkIfNeeded()
                }
            }
        }
    }

    override func viewDidAppear(_ animated: Bool) {
        super.viewDidAppear(animated)
        canBlink = true
        blinkIfNeeded()
    }

    override func viewWillDisappear(_ animated: Bool) {
        super.viewWillDisappear(animated)
       canBlink = false
    }
}
```

```swift
//
//  EmotionsViewController.swift
//  FaceIt
//
//  Created by CS193p Instructor on 1/30/17.
//  Copyright © 2017 Stanford University. All rights reserved.
//

import UIKit

class EmotionsViewController: UITableViewController, UIPopoverPresentationControllerDelegate
{
    // MARK: Model

    private var emotionalFaces: [(name: String, expression: FacialExpression)] = [
        ("Sad", FacialExpression(eyes: .closed, mouth: .frown)),
        ("Happy", FacialExpression(eyes: .open, mouth: .smile)),
        ("Worried", FacialExpression(eyes: .open, mouth: .smirk))
    ]

    // MARK: UITableViewDataSource

    override func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return emotionalFaces.count
    }

    override func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let cell = tableView.dequeueReusableCell(withIdentifier: "Emotion Cell", for: indexPath)
        cell.textLabel?.text = emotionalFaces[indexPath.row].name
        return cell
    }

    // MARK: - Navigation

    @IBAction func addEmotionalFace(from segue: UIStoryboardSegue) {
        if let editor = segue.source as? ExpressionEditorViewController {
            emotionalFaces.append((editor.name, editor.expression))
            tableView.reloadData()
        }
    }

    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
        var destinationViewController = segue.destination
        if let navigationController = destinationViewController as? UINavigationController {
            destinationViewController = navigationController.visibleViewController ?? destinationViewController
        }
       if let faceViewController = destinationViewController as? FaceViewController,
            let cell = sender as? UITableViewCell,
            let indexPath = tableView.indexPath(for: cell) {
            faceViewController.expression = emotionalFaces[indexPath.row].expression
            faceViewController.navigationItem.title = emotionalFaces[indexPath.row].name
       } else if destinationViewController is ExpressionEditorViewController {
            if let popoverPresentationController = segue.destination.popoverPresentationController {
                popoverPresentationController.delegate = self
            }
        }
    }

    // MARK: UIPopoverPresentationControllerDelegate

    func adaptivePresentationStyle(
        for controller: UIPresentationController,
        traitCollection: UITraitCollection
    ) -> UIModalPresentationStyle
    {
        if traitCollection.verticalSizeClass == .compact {
            return .none
        } else if traitCollection.horizontalSizeClass == .compact {
            return .overFullScreen
        } else {
            return .none
        }
    }
}
```