Lecture 7 Demo Code: Cassini Scroll View

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. And here is the complete project.

```
//
//  ImageViewController.swift
//  Cassini
//
//  Created by CS193p Instructor.
//  Copyright © 2017 Stanford University. All rights reserved.
//

import UIKit

class ImageViewController: UIViewController
{
    // MARK: Model

    var imageURL: URL? {
        didSet {
            image = nil
            if view.window != nil { // if we're on screen
                fetchImage()        // then fetch image
            }
        }
    }

    // MARK: Private Implementation

    private func fetchImage() {
        if let url = imageURL {
            // this next line of code can throw an error
            // and it also will block the UI entirely while access the network
            // we really should be doing it in a separate thread
            let urlContents = try? Data(contentsOf: url)
            if let imageData = urlContents {
                image = UIImage(data: imageData)
            }
        }
    }
```

```
    // MARK: View Controller Lifecycle

    override func viewDidLoad() {
        super.viewDidLoad()
        imageURL = DemoURL.stanford // for demo/testing purposes only
    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
        if image == nil { // we're about to appear on screen so, if needed,
            fetchImage()  // fetch image
        }
    }

    // MARK: User Interface

    @IBOutlet weak var scrollView: UIScrollView! {
        didSet {
            // to zoom we have to handle viewForZooming(in scrollView:)
            scrollView.delegate = self
            // and we must set our minimum and maximum zoom scale
            scrollView.minimumZoomScale = 0.03
            scrollView.maximumZoomScale = 1.0
            // most important thing to set in UIScrollView is contentSize
            scrollView.contentSize = imageView.frame.size
            scrollView.addSubview(imageView)
        }
    }

    fileprivate var imageView = UIImageView()

    private var image: UIImage? {
        get {
            return imageView.image
        }
        set {
            imageView.image = newValue
            imageView.sizeToFit()
            // careful here because scrollView might be nil
            // (for example, if we're setting our image as part of a prepare)
            // so use optional chaining to do nothing
            // if our scrollView outlet has not yet been set
            scrollView?.contentSize = imageView.frame.size
        }
    }
}

// MARK: UIScrollViewDelegate
// Extension which makes ImageViewController conform to UIScrollViewDelegate
// Handles viewForZooming(in scrollView:)
// by returning the UIImageView as the view to transform when zooming

extension ImageViewController : UIScrollViewDelegate
{
    func viewForZooming(in scrollView: UIScrollView) -> UIView? {
        return imageView
    }
}
```

```
//
//  DemoURL.swift
//
//  Created by CS193p Instructor.
//  Copyright (c) 2017 Stanford University. All rights reserved.
//

import Foundation

struct DemoURL
{
    static let stanford = URL(string: "http://stanford.edu/about/images/intro_about.jpg")

    static var NASA: Dictionary<String,URL> = {
         let NASAURLStrings = [
            "Cassini" : "http://www.jpl.nasa.gov/images/cassini/20090202/pia03883-full.jpg",
            "Earth" : "http://www.nasa.gov/sites/default/files/wave_earth_mosaic_3.jpg",
            "Saturn" : "http://www.nasa.gov/sites/default/files/saturn_collage.jpg"
        ]
        var urls = Dictionary<String,URL>()
        for (key, value) in NASAURLStrings {
            urls[key] = URL(string: value)
        }
        return urls
    }()
}
```