Advanced Mobile Application Development
Week 10: JSON

petitions
File | New Project
Master-Detail App
iPhone

Setup
Go into the Main storyboard and in the table view change the table view cell to the style subtitle.
Change the title of the master scene to White House Petitions.

In MasterViewController.swift delete the insertNewObject() method entirely, delete everything from viewDidLoad() except the call to super.viewDidLoad(), delete or comment out the table view's commitEditingStyle and canEditRowAtIndexPath methods, and finally delete the as! NSDate text from the prepareForSegue() and cellForRowAtIndexPath methods – not the whole line, just the bit that says as! NSDate.

Make sure your app builds at this point.

Load JSON file
Look at JSON data https://api.whitehouse.gov/v1/petitions.json?limit=100
It's easier to read in XML https://api.whitehouse.gov/v1/petitions
Look at the items in the result, they are key-value pairs
Change objects to store these key-value pairs as dictionaries stored in an array.

```
var objects = [[String:String]]()
```

Create a method to download the json file.

```
    func loadjson(){
        let urlPath =
"https://api.whitehouse.gov/v1/petitions.json?limit=50"
        guard let url = NSURL(string: urlPath)
            else {
                print("url error")
                return
            }

        let session = NSURLSession.sharedSession().dataTaskWithURL(url,
completionHandler: {(data, response, error) in
            let httpResponse = response as! NSHTTPURLResponse
            let statusCode = httpResponse.statusCode
guard statusCode == 200
                else {
                    print("file download error")
                    return
            }
            //download successful
            print("download successful")
})
        //must call resume to run session
```

```
        session.resume()
    }
```

Update viewDidLoad() to call this method.

```
    override func viewDidLoad() {
        super.viewDidLoad()
        loadjson()
    }
```

If you run it now you should get the download successful message.

Parse JSON file

Now we'll create a method to get the json data and parse it so we can use it.

```
    func parsejson(data: NSData){
        do {
            // get json data
            let json = try NSJSONSerialization.JSONObjectWithData(data,
options: NSJSONReadingOptions.AllowFragments) as! NSDictionary
            let meta = json["metadata"] as! NSDictionary
            let response = meta.objectForKey("responseInfo") as!
NSDictionary
            let status = response.objectForKey("status") as! Int
            if status == 200 {
                //download successful
                let allresults = json["results"] as! NSArray
                let results = Array(allresults)
                //add results to objects
                for result in results {
                    //check that data exists
                    guard let title = result["title"]! as? String,
                        let sigCount = result["signatureCount"] as?
NSNumber,
                        let itemurl = result["url"]!as? String
                        else {
                            continue
                    }
                    //create new object
                    let obj = ["title": title, "signature":
sigCount.stringValue, "url": itemurl]
                    //add object to array
                    self.objects.append(obj)
                }
                //reload the table data after the json data has been
downloaded
                //self.tableView.reloadData()
            }
        } catch {
            print("Error with JSON: \(error)")
        }
        tableView.reloadData()
    }
```

Call this from loadjson() right after the print statement that the download was successful.

```
dispatch_async(dispatch_get_main_queue()) {self.parsejson(data!)}
```

Load table data

Now we're ready to load the petition data in our table. Update the following table view delegate method

```
    override func tableView(tableView: UITableView, cellForRowAtIndexPath
indexPath: NSIndexPath) -> UITableViewCell {
        let cell = tableView.dequeueReusableCellWithIdentifier("Cell",
forIndexPath: indexPath)

        let object = objects[indexPath.row]
        cell.textLabel!.text = object["title"]
        if object["signature"] != nil {
            cell.detailTextLabel!.text = object["signature"]! + "
signatures"
        }
        return cell
    }
```

Detail View

In the main storyboard go into the detail view and remove the label (check that this removes its connection as well) and replace it with a web view that fills up the whole view.
Add an activity indicator on top of the web view. (it must be below the web view in the document hierarchy). In the attributes inspector check Hides When Stopped but make sure Hidden is unchecked (down below)
Connect the web view and activity indicator as webView and webSpinner.
Add needed constraints.
Before leaving the storyboard go to the Master view and change the accessory on the cell to a disclosure indicator to give the user the visual cue that selecting the row will lead to more information.

In DetailViewController remove detailDescriptionLabel.

Adopt the UIWebViewDelegate protocol for the class

```
class DetailViewController: UIViewController, UIWebViewDelegate
```

Set up the web view's delegate in viewDidLoad()

```
webView.delegate=self
```

Write a method to load a web page.

```
    func loadWebPage(urlString: String){
        //the urlString should be a propery formed url
        //creates a NSURL object
        let url = NSURL(string: urlString)
        //create a NSURLRequest object
        let request = NSURLRequest(URL: url!)
        //load the NSURLRequest object in our web view
        webView.loadRequest(request)
    }
```

Update detailItem and configureView() as follows:

```
    var detailItem: String?

    func configureView(){
        if let url = detailItem{
            loadWebPage(url)
        }
    }
```

Implement the two delegate methods that are called when the web page starts and stops loading.

```
func webViewDidStartLoad(webView: UIWebView) {
        webSpinner.startAnimating()
    }

func webViewDidFinishLoad(webView: UIWebView) {
        webSpinner.stopAnimating()
    }
```

In MasterViewController update prepareForSegue() to send the detail view the data it needs.

```
override func prepareForSegue(segue: UIStoryboardSegue, sender: AnyObject?){
        if segue.identifier == "showDetail" {
            if let indexPath = self.tableView.indexPathForSelectedRow {
                let petition = objects[indexPath.row]
                let title = petition["title"]
                let url = petition["url"]
                let controller = (segue.destinationViewController as!
UINavigationController).topViewController as! DetailViewController
                controller.detailItem = url
                controller.title = title
                controller.navigationItem.leftBarButtonItem =
self.splitViewController?.displayModeButtonItem()
                controller.navigationItem.leftItemsSupplementBackButton =
true
            }
        }
    }
```

When you run the app if there's a space about the web view you need to update the top constraint to go all the way to the top of the view, not just to the bottom of the nav bar.

Don't forget the launch screen and app icons.