**Mobile Configuration Profile Example App:**

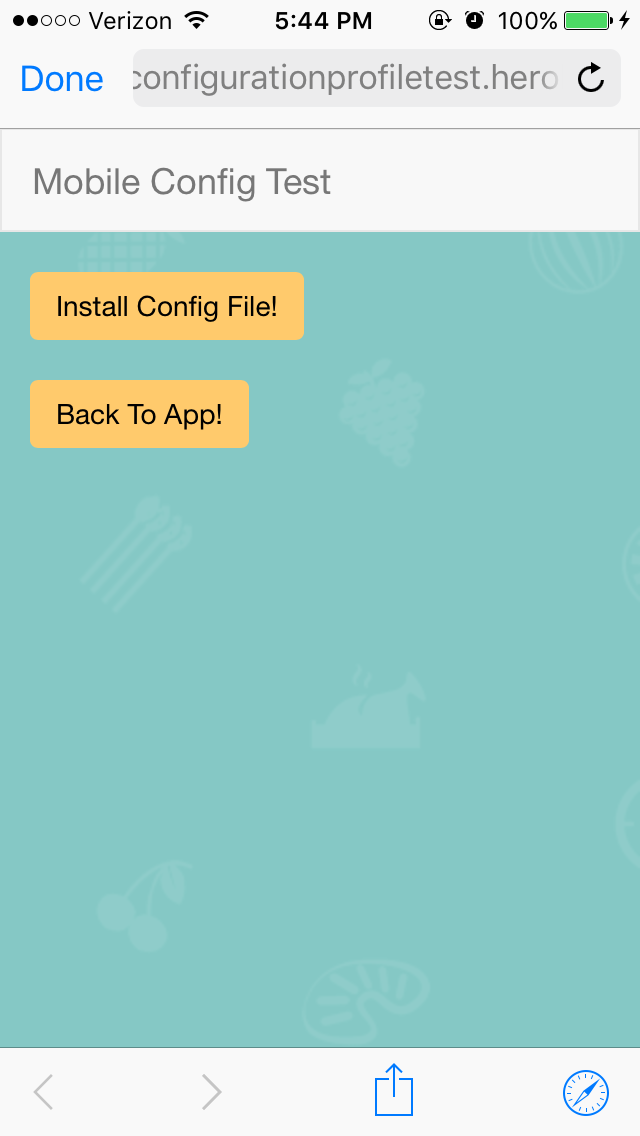
**Purpose:**

* Learn to automatically connect a mobile device to an external device by simply installing a custom mobile config file distributed through the web.
* Once connected, the two devices can interchange data.
* This process will be very important for the increasingly popular “Internet of Things”

**What this example does:**

* In this example, the external device we will be connecting to is a router.
* After setting up the example project, your iOS app will visit a website from which it will download a mobile config profile, install it, then automatically connect you to a Wi-Fi network. All without the need to set up the connection manually (entering passwords and such).
* Follow the step-by-step instructions document to customize everything with your own router. (See Documents: “Step by Step How to Customize this Sample Project” and “How to Sign Config Profiles”)
* Although this example just sets up a Wi-Fi connection, a simple yet purposeful use, mobile config profiles can do much more, such as set up your phone’s restrictions or VPN.

**Screenshots:**

**Steps to test:**

*Note: This example will not work out of the box. You must first complete all steps in the “Step by Step How to Customize this Sample Project” document*

1. Forget the Wi-Fi network your mobile config file points to (if you are already connected).
2. Open the iOS app on your phone
3. Press the “Get Mobile Config” button. This action will redirect you to your sample website
4. On the website, press “Install Config Files!.” This action will download the mobile config file
5. Your phone will automatically display instructions on how to install. Follow these instructions to install the config profile
6. At the final step, after pressing “install", you should be redirected back to the website in Safari. From here, press the “Back To App!” button to be redirected back to the app.
7. You should now be automatically connected to your network, all without manually putting in credentials.

**What this package contains:**

* *Documentation*
  + This document (Overview)
  + How to sign a mobile config file
  + A step-by-step setup of the mobile config file, website, and iOS app
* A *sample rails project*
* A *sample iOS app*

**Mobile Configuration Profile:**

* The mobile configuration profile will contain all the information needed for the device to connect to a specific router. The phone simply needs to install it.
* This file is readable as an XML, which means that you could technically open it up in a text editor, view all the information (including the router password), and edit anything you’d like. For this reason, it is not secure. However, you can encrypt or sign it, making it unreadable for a human.
* In our example, we are loading this mobile config file to a public website, so we definitely do not want any information easily accessible. Therefore, we must sign it.
* Go through all the steps to set up everything from the beginning. (See Documents: “Step by Step How to Customize this Sample Project” and “How to Sign Config Profiles”)
* An easy way to test that the config file works is to email it to yourself and open it on your phone.

**Sample Rails Project:**

* This ruby on rails site contains two buttons.
  + First button links to the mobile config file that will be downloaded with a click.
  + Second button redirects the user back to the sample iOS app by a URL Scheme.

**Sample iOS App**

* This app has one button that opens up Safari with a URL to your sample website. From what I know, you can only download+install mobile configs within the phone via the Safari or Email app. You cannot use SFSafariViewController or a UIWebView to install mobile configs within the app.

**Unsolved Difficulties**:

* In the process of installing the mobile config profile, after the user presses ‘install’ or ‘cancel’, the phone redirects you back to Safari rather than the app.
  + I could not figure out a way to automatically redirect the user back to the app after installing/canceling.
  + Our current work around is the second button on the webpage, “Back To App!,” which brings the user back to the app with a click.
  + For the best UX, it would be ideal to automatically direct the user back to our app after installing. Our work-around is not terrible though. As a little bonus, I’ve made the design fairly similar on both the webpage and app to make the experience more fluid.
* Another difficulty was being able to download the mobile config file through a UIWebView or even SFSafariViewController. Clicking the download link on either of these views does nothing, but clicking the link in the Email app or Safari app works as expected. It would be ideal to keep the user in the app somehow rather than direct them out of the app.
* Another difficulty discussed was how to disconnect from the access point (in our case, it’s a router). Ideally, as soon as all the necessary information is interchanged, we would want the user to automatically disconnect from the device and reconnect to their original router. As far as I know, it is impossible to programmatically delete the mobile config file or to force the user to disconnect from one device and connect back to another device programmatically. It has been suggested that the open access point (in our case, the router) could turn itself off once it has all the data it needs. And at that point there would be nothing for the phone to connect to other than its default (the original router).