Networking with NSURLSession

Hands-on Challenges

Networking with NSURLSession Hands-On Challenges

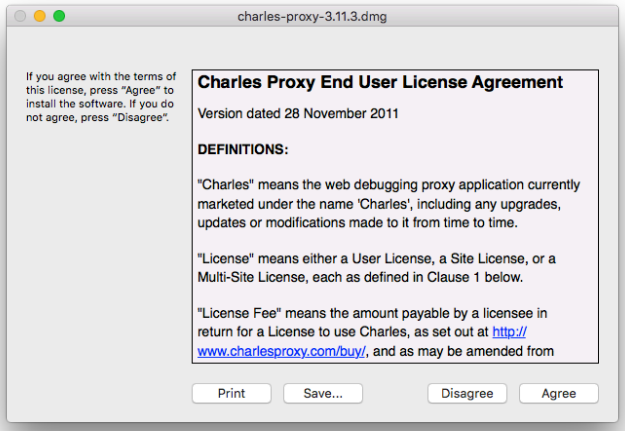
Challenge 2: Using Charles

Sometimes when creating network requests and receiving responses, it can be really helpful to view the network traffic. You can add breakpoints and view the requests and responses, but it can be non-trivial to do. An easier way to see the traffic is to use an app called Charles that intercepts all the network traffic on your computer and logs it.

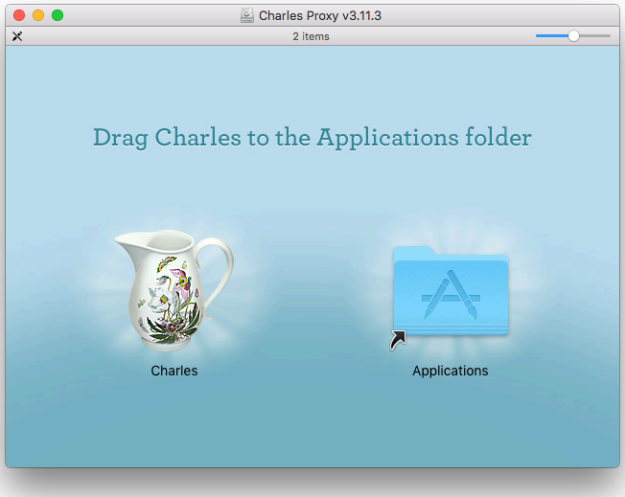
You can download Charles at www.charlesproxy.com/latest-release/download.do. For step-by-step instructions, read on.

Installation

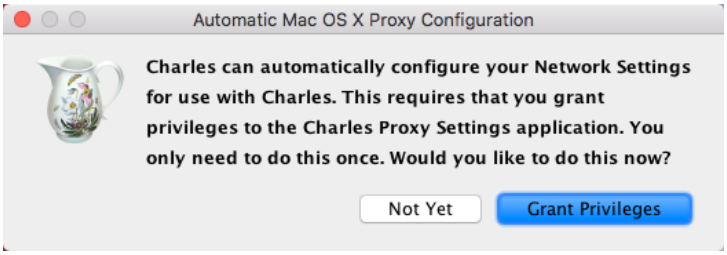
1. Open the downloaded file. When you open the dmg, you'll see a license agreement.



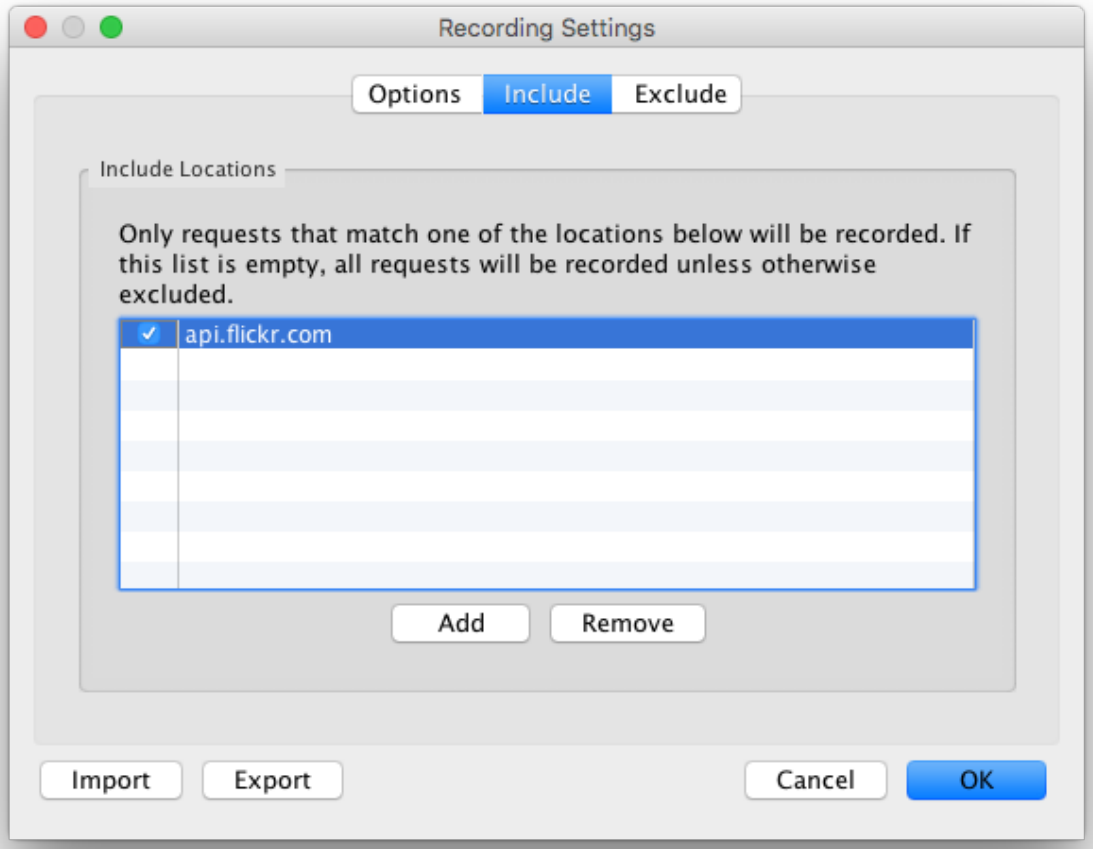
2. After the license agreement, you'll be able to drag the Charles application to the Applications folder.



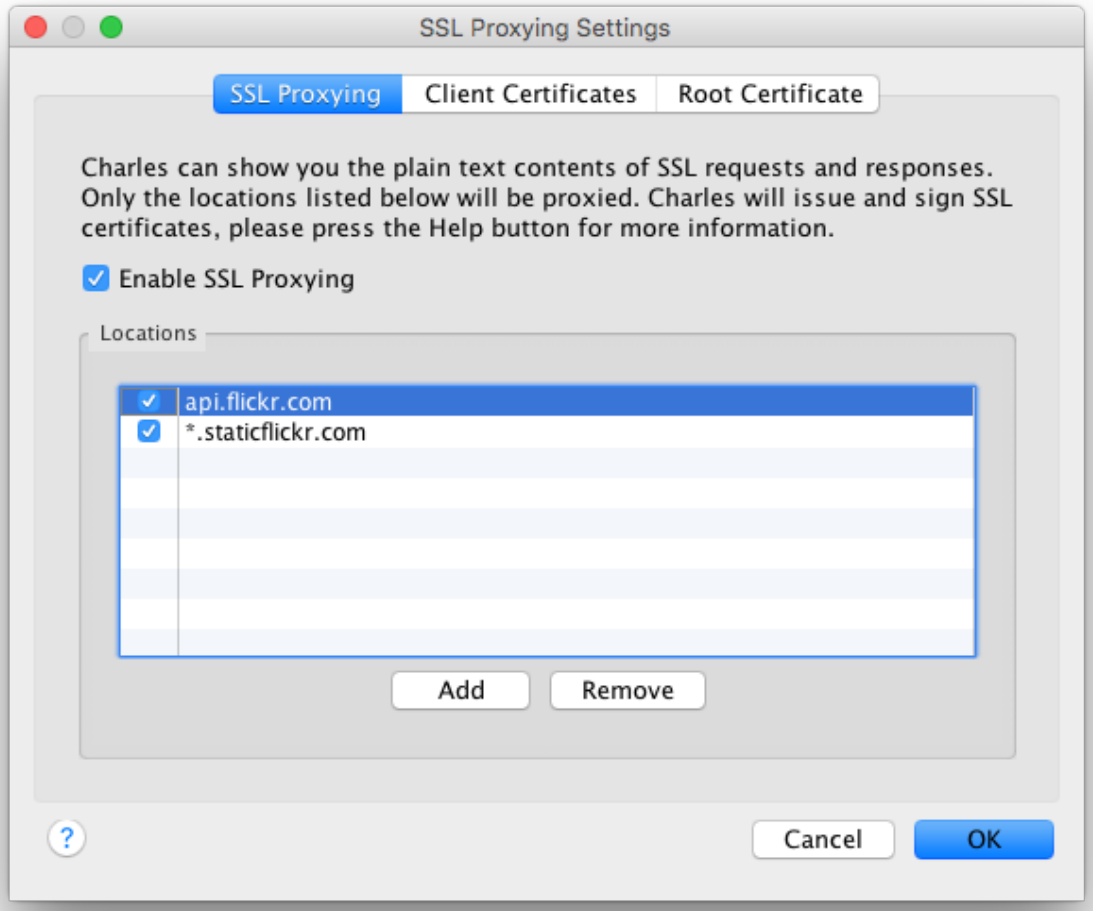
3. When you open the application, Charles will prompt to configure your proxy settings. Pick “Grant Privileges” and type your system password, when prompted.



4. Once Charles starts up, it will start logging all your network traffic. You may be surprised at how much there is! If you want to narrow it down to only certain sites, pick “Proxy/Recording Settings…” and switch to the “Include” tab. Add the sites you want to monitor and click “OK”.



5. If you're monitoring SSL traffic, there are a couple of steps you'll need to take to get it to work. First, you'll have to add the site to a list in Charles. Choose "Proxy/SSL Proxying Settings…" and add the sites to the list.

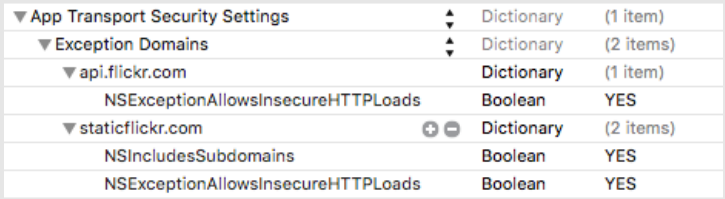


6. Since SSL is meant to be secure, and Charles is basically a man-in-the-middle attack, you'll need some steps in code to allow it to work. For a playground, you'll need a delegate for the NSURLSession to handle the security challenge. Add this code to the playground at the top, just under the import statements:

```
class SSLAuthHandler: NSObject, NSURLSessionDelegate {
  func URLSession(session: NSURLSession, didReceiveChallenge
    challenge: NSURLAuthenticationChallenge, completionHandler:
    (NSURLSessionAuthChallengeDisposition, NSURLCredential?) ->
    Void) {
    if challenge.protectionSpace.authenticationMethod ==
      NSURLAuthenticationMethodServerTrust &&
      challenge.protectionSpace.host == "api.flickr.com" {
        let credential = NSURLCredential(forTrust:
          challenge.protectionSpace.serverTrust!)
        completionHandler(.UseCredential, credential)
    } else {
      completionHandler(.PerformDefaultHandling, nil)
    }
  }
}
```

This code creates a class that can be used as a delegate to handle the security challenge. You do NOT want to ship this code, just use it for when you need Charles to look at the SSL traffic to api.flickr.com. Of course, if you're working with a different site, you'll need to change the host string to match.

If you're working in an app, not a playground, you'll need to implement the method above, but you'll also need an exception in your Info.plist file. In this series, you don't have an app yet. You've only created a playground so far. But, for reference, here's an example of the entries you would need.



These settings should also NOT be shipped. They should be considered temporary while viewing network traffic only.

Then, change the session initializer to use the new class as its delegate. Change this line:

```
let session = NSURLSession.sharedSession()
```

To this:

```
let delegate = SSLAuthHandler()
let session = NSURLSession(configuration:
NSURLSessionConfiguration.defaultSessionConfiguration(), delegate:
delegate, delegateQueue: nil)
```