Debugging applications

Overview

This tutorial explores various approaches to debugging the web resources of a MobileFirst application, either before running the application on a device or while running it on a device.

The following debugging options are covered:

- What is debugging?
- Debugging on a desktop browser
- Debugging with the Mobile Browser Simulator
- Debugging with iOS Remote Web Inspector
- Debugging with Chrome Remote Web Inspector
- Debugging with Weinre
- Debugging with IBM MobileFirst Logger
- Testing Adapters
- Debugging with WireShark

What is debugging?

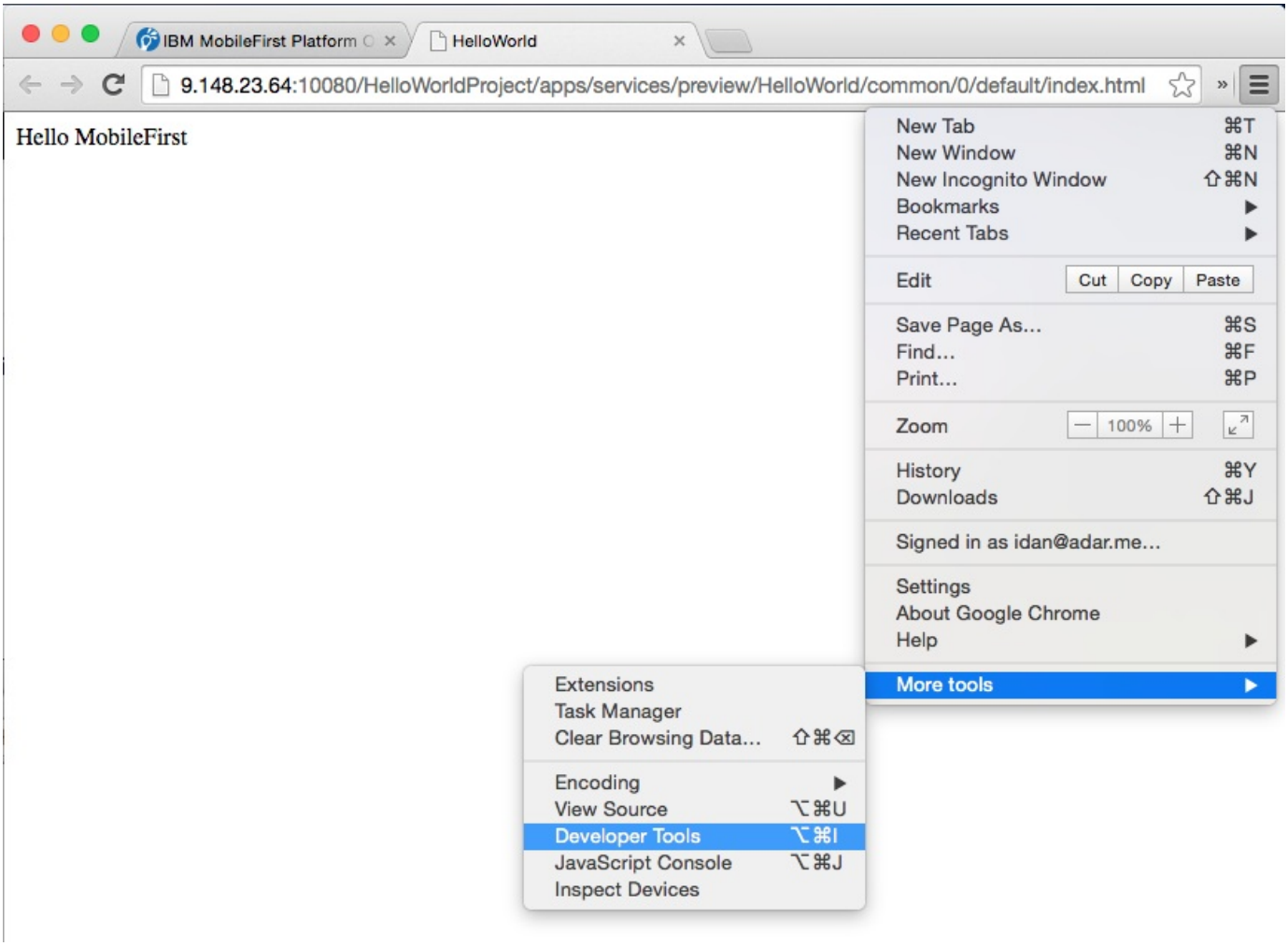
Debugging is a process that consists of finding the cause of defects in the application code and UI.

- MobileFirst applications consist of web-based resources and optional native code (such as Java, Objective-C and C#).
- You can debug native code by using standard tools that are provided by the platform SDK, such as XCode, Android LogCat/ADB, or Microsoft Visual Studio.

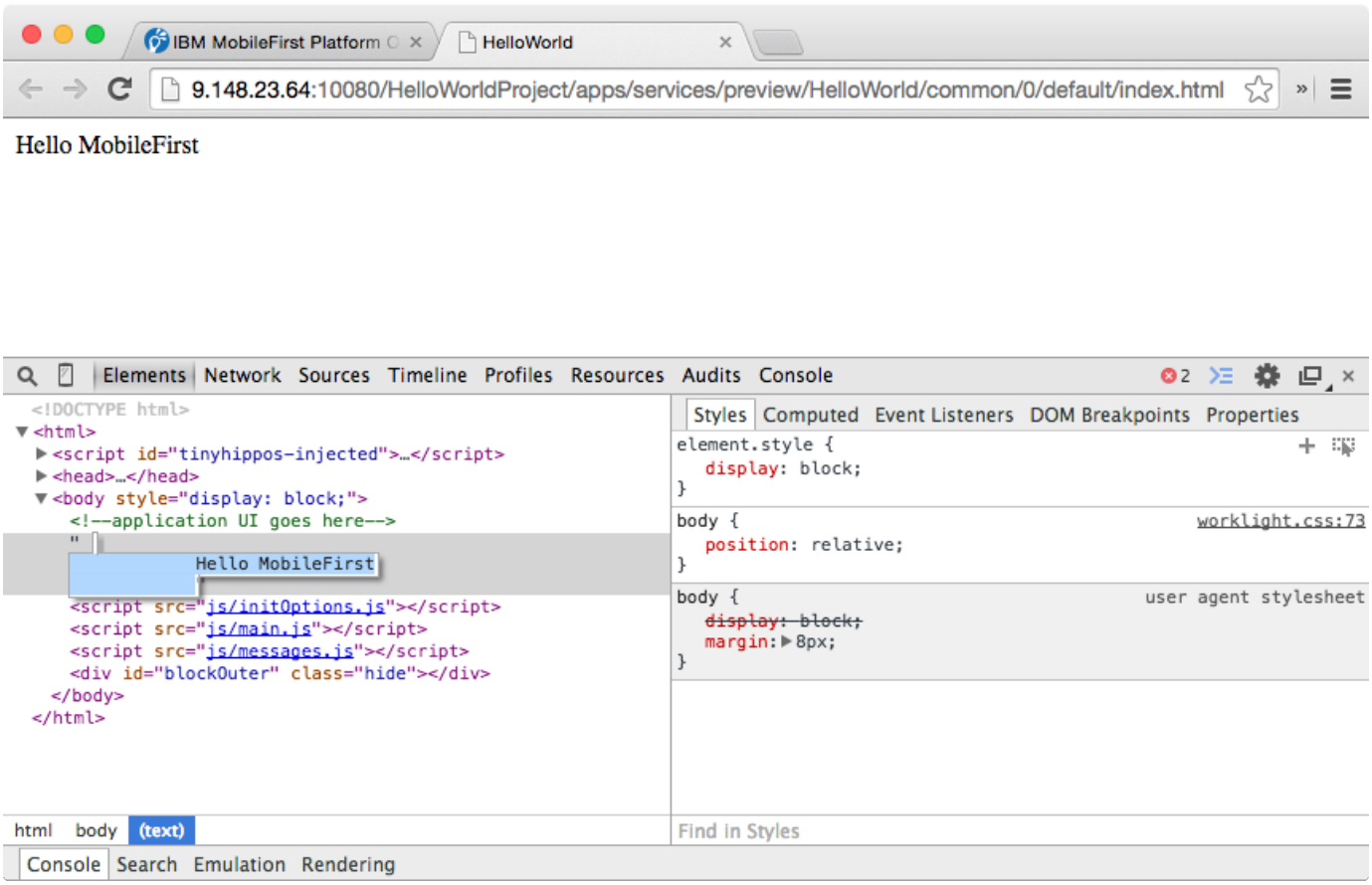
Debugging on a desktop browser

Modern browsers, such as Chrome, Firefox, Safari, or Opera, provide an easy and convenient way to debug web apps. Many web tools for debugging are available on various desktop browsers, for example:

- FireBug
- Chrome Developer Tools
- Internet Explorer Developer Tools
- Dragonfly for Opera
- Safari Web Inspector



In early application development stages, these tools can be used to debug the application just like a regular website. It is not required to install them in a mobile device. You can also preview changes to HTML and CSS in real time by modifying the values in the inspector.



Debugging with the Mobile Browser Simulator

You can also use the Mobile Browser Simulator to preview and debug MobileFirst applications.
In order to use the Mobile Browser Simulator you need to:

1. Follow the Using CLI to manage MobileFirst artifacts (../using-cli-to-manage-mobilefirst-artifacts) tutorial to install the MobileFirst CLI.
2. Use the following command to emulate/debug your application:

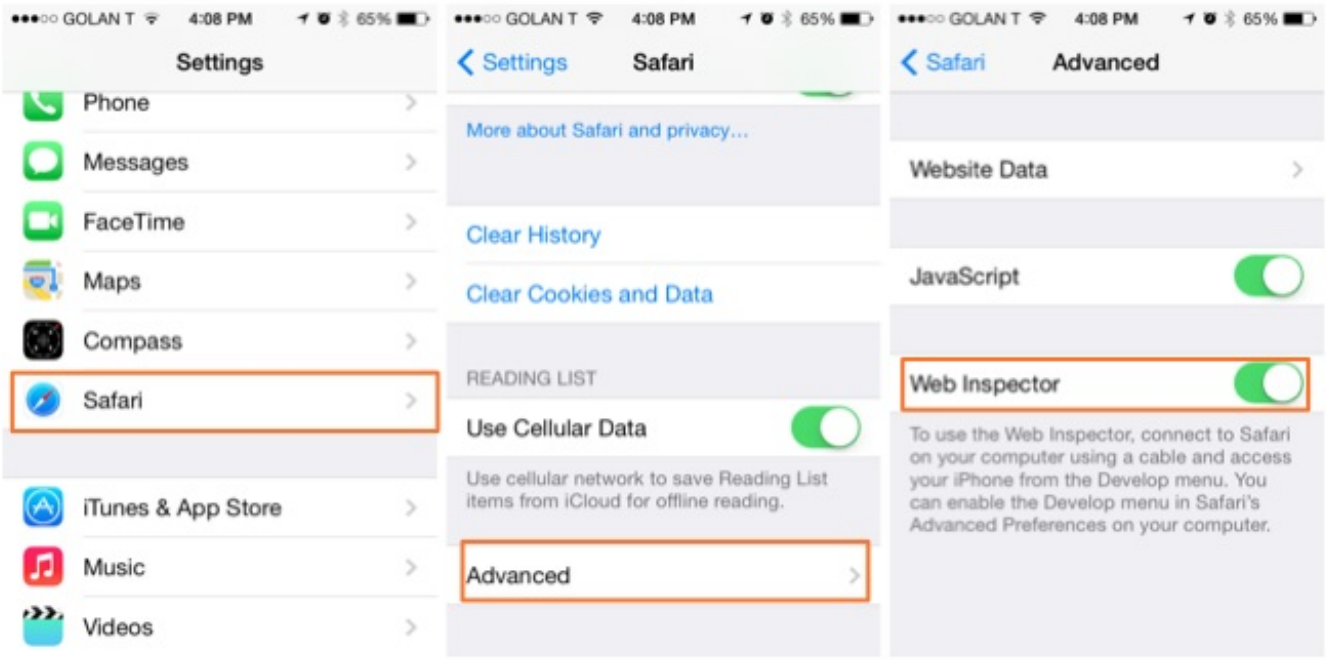
   ```
   $ ./mfpdev app preview
   ```

   If your application consists of more that one platform - specify the platform to preview:

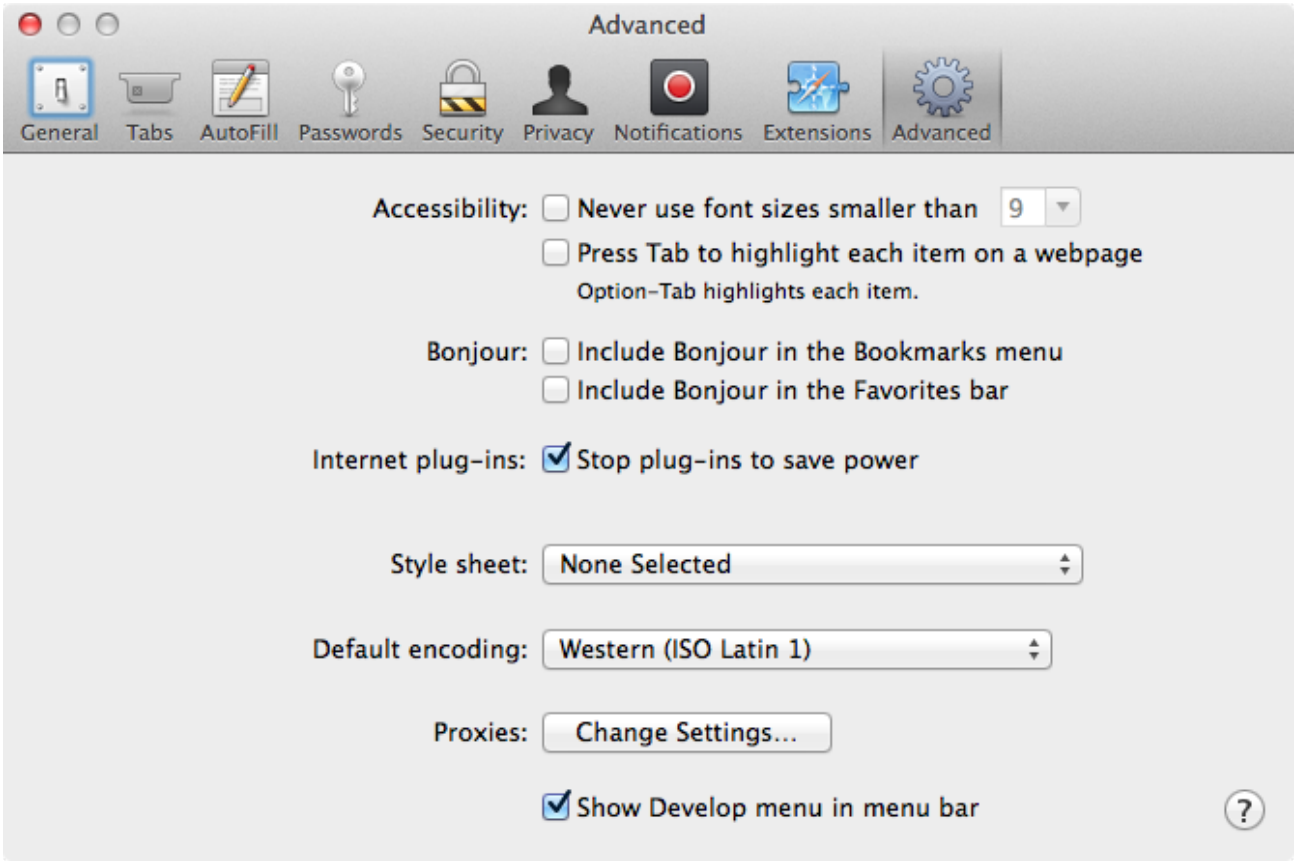
   ```
   $ ./mfpdev app preview <platform>
   ```

Debugging with iOS Remote Web Inspector

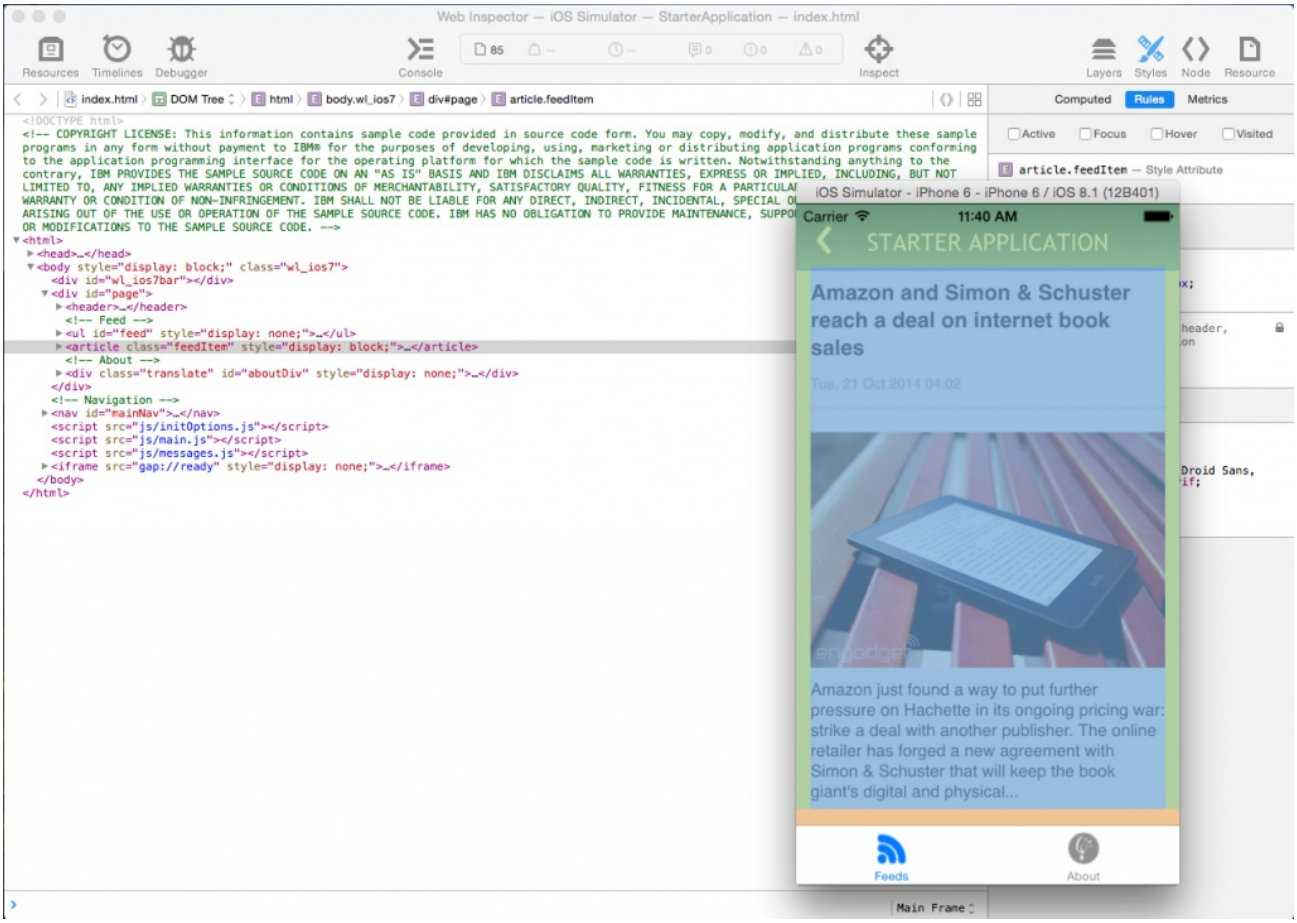
Starting in iOS 6, Apple introduced a remote web inspector for debugging web applications on iOS devices. To debug, make sure that the device (or simulator) has the **Private Browsing** option turned off. To enable Web Inspector on the device, click **Settings > Safari > Advanced > Web Inspector**.



1. To start debugging, connect the iOS device to a Mac, or start the simulator. Safari 6.0 or higher is required. In Safari, go to **Preferences > Advanced**, and select the **Show Develop menu in menu bar** checkbox.



2. Now in Safari, select **Develop > [*your device ID*] > [*your application HTML file*]**. The DOM can now be inspected. It is also possible to alter the CSS and run JavaScript commands, just as in the desktop inspector.

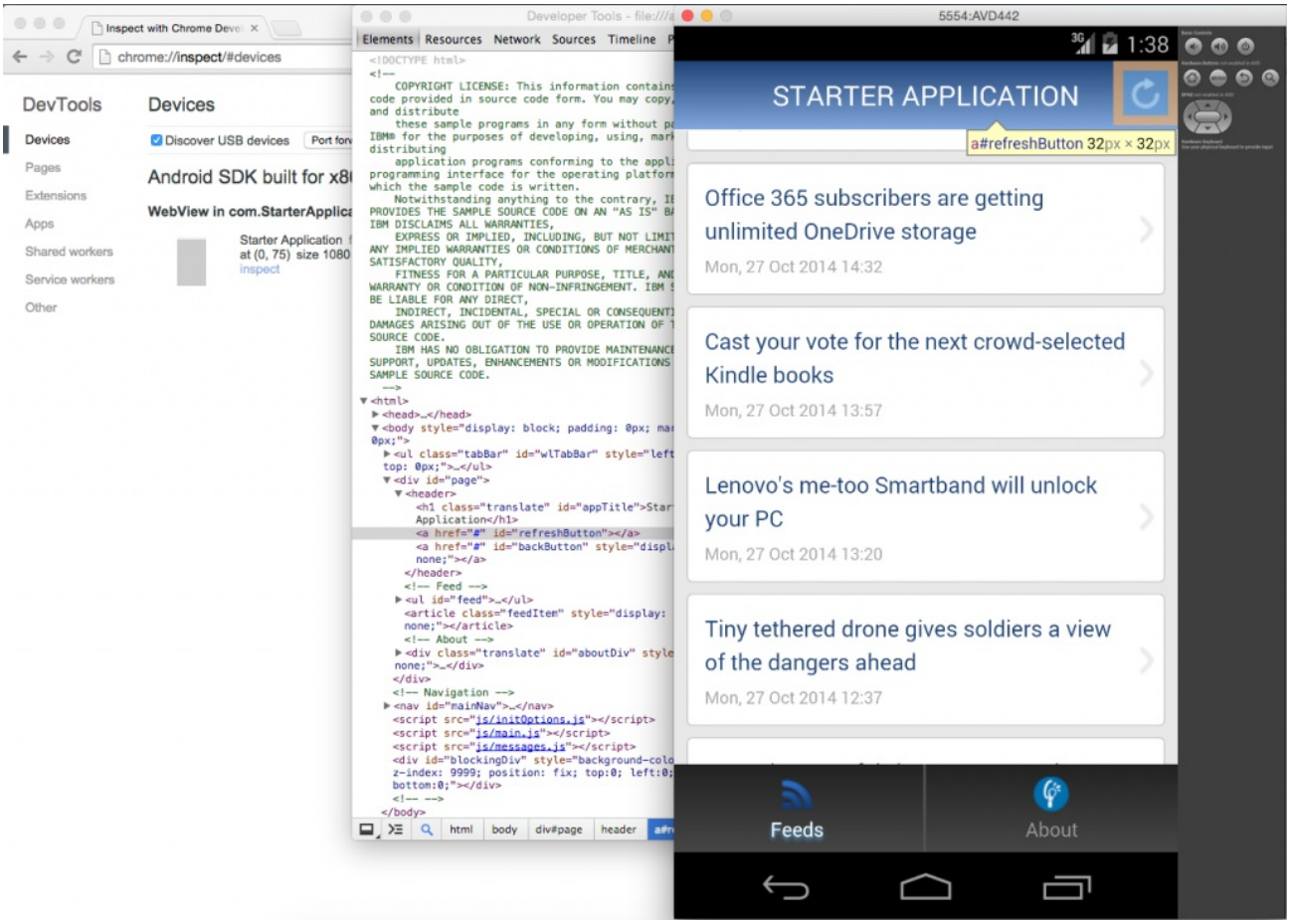


Debugging with Chrome Remote Web Inspector

With Google Chrome, it is possible to remotely inspect web applications on Android devices. This action requires Android 4.4 or later, Chrome 32 or later, and IBM Worklight Foundation V6.2.0 or IBM MobileFirst Platform Foundation 6.3 or later. Additionally, in the `AndroidManifest.xml` file, `targetSdkVersion` = 19 or above is required. In the `project.properties` file, `target` = 19 or above is required.

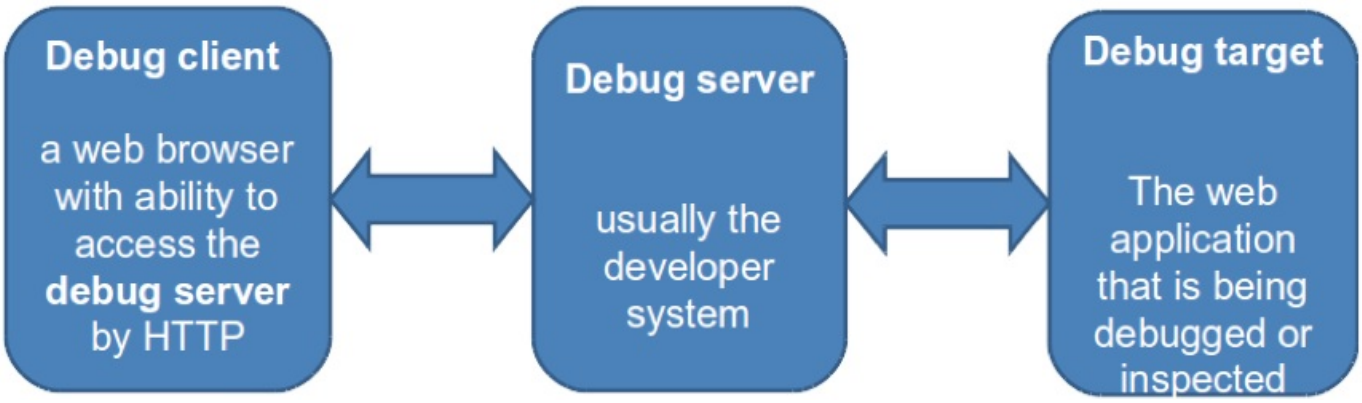
1. Start the application in the Android Emulator or a connected device.
2. In Chrome, enter the following URL: `about:inspect`.
3. Press **Inspect** for the relevant application.

   You can now use all the features of the Chrome Inspector to inspect the Android application.



Debugging with Weinre

Weinre is a debugger for web pages, like Firebug or other Web Inspectors, except that **Weinre is designed to work remotely**. Weinre can be used to inspect and debug web resources such as HTML, JavaScript, CSS, and network traffic on mobile handsets. The Weinre architecture includes the following components:



The Weinre debug server requires a `node.js` runtime. You can find instructions to install Weinre on the Weinre installation page (http://people.apache.org/%7Epmuellr/weinre/docs/latest/Installing.html).

Debug server

When the Weinre server is installed, enter the following command to run it:
`weinre --httpPort 8888 --boundHost -all-`
This command starts a Weinre server.
The default port is `8888` but you can change it.

Target

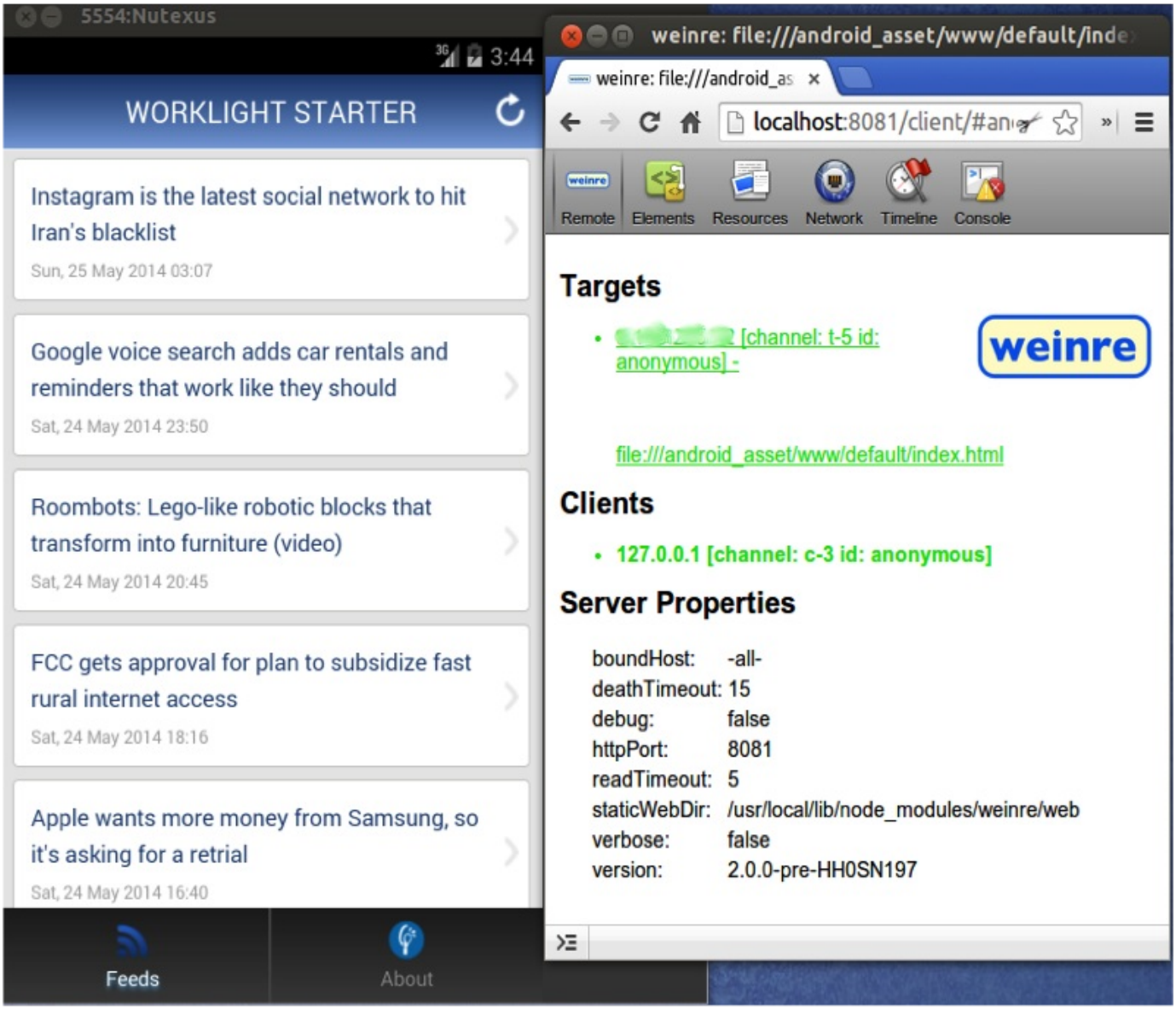
The Weinre server must be accessible from the device that will be used for debugging.
To make it accessible, add the following code line to the web application:

```
<script src="http://a.b.c:8888/target/target-script-min.js"></script>
```

Where a.b.c is the hostname or IP of the Weinre server.

Client

Before you can start debugging, make sure that the application is open and loaded on the browser with this URL:



Debugging with IBM MobileFirst Logger

IBM MobileFirst Platform Foundation provides a `WL.Logger` object which can be used to print log messages to the log for the environment used.
Two of its methods are `WL.Logger.debug()` and `WL.Logger.error()`.
These APIs are multiplatform. The output destination changes according to the platform on which that application runs:

- **Developer console** when it is running on a desktop browser
- **LogCat** when it is running on Android device
- **Visual Studio Output** when it is running on a Windows Phone 8 device and Windows 8 App
- **XCode Console** when it is running on an iOS device

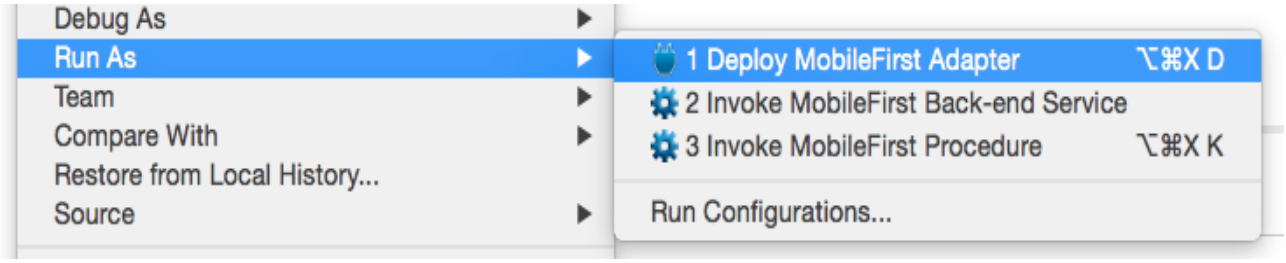
`WL.Logger` contains more methods.

> For more information, see the documentation for `WL.Logger` in the API reference part of the user documentation.

Testing Adapters

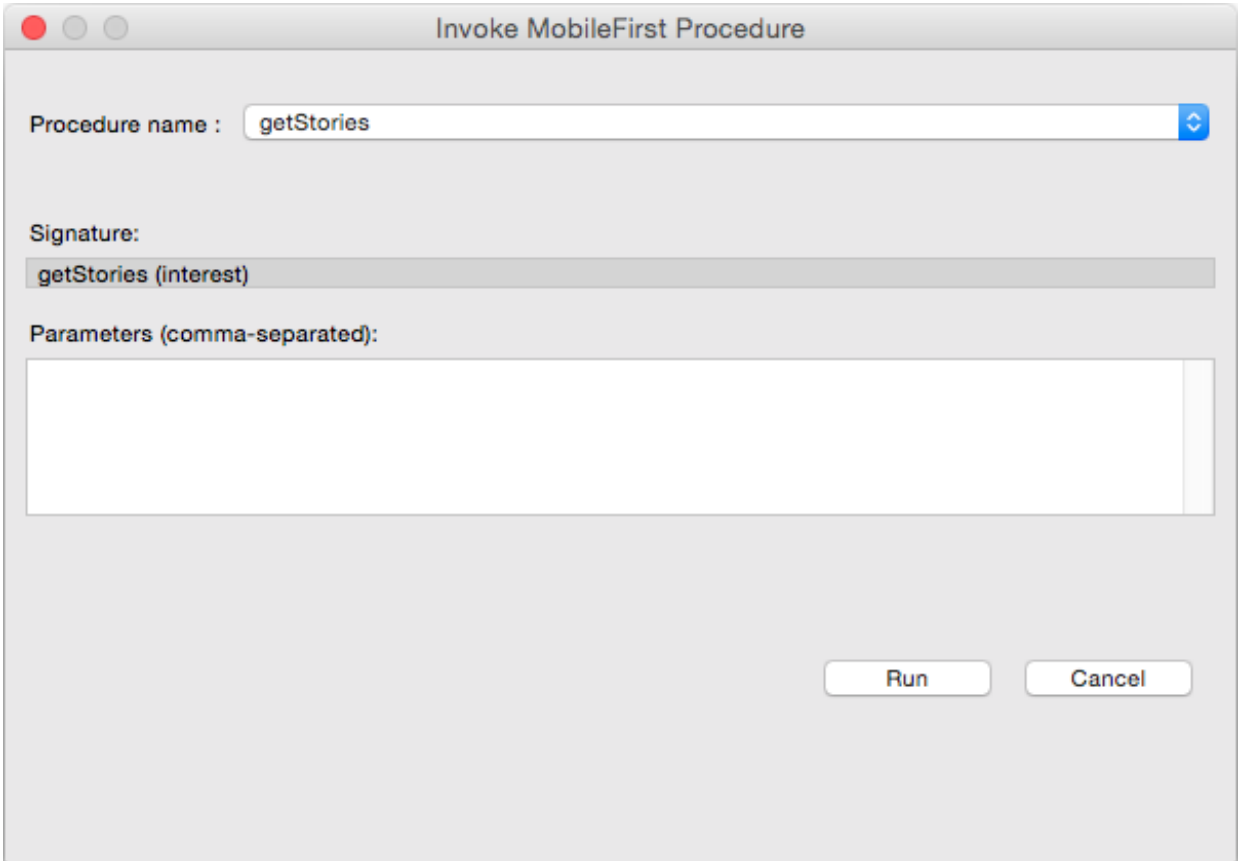
You can test adapter procedures by using MobileFirst Studio. To test a procedure:

1. Right-click an adapter folder and select **Run As < Invoke MobileFirst Procedure**.



(http://developer.ibm.com/mobilefirstplatform/wp-content/uploads/sites/32/2014/07/03_04_deploy_mobilefirst_adapter.png) The adapter and procedure are selected.

2. Optionally, enter comma-separated parameters.



(http://developer.ibm.com/mobilefirstplatform/wp-content/uploads/sites/32/2014/07/03_04_select_adapter_and_procedure.png)
Adapter invocation result:

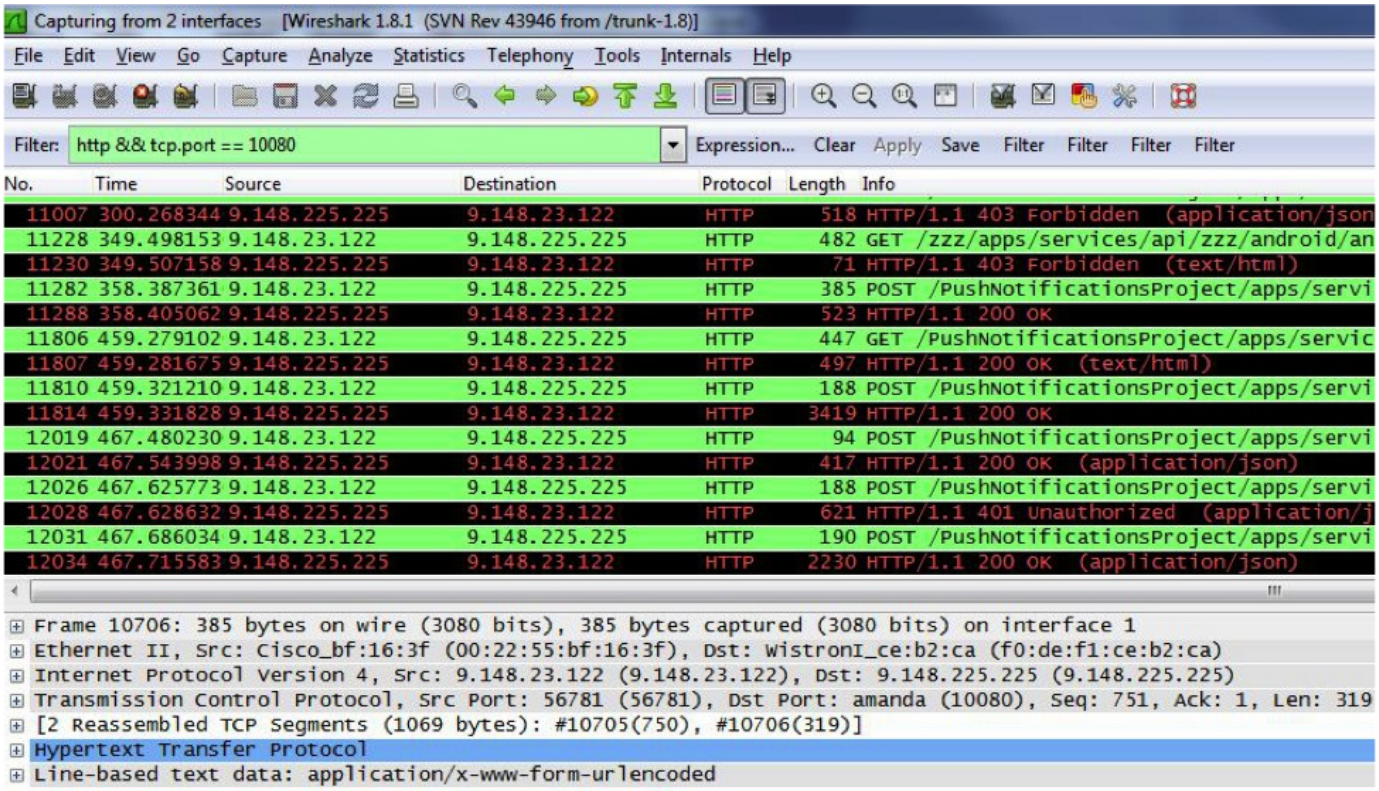
Invocation Result of procedure: 'getStories' from the MobileFirst Server:


```
{
  "errors": [
  ],
  "info": [
  ],
  "isSuccessful": true,
  "responseHeaders": {
    "Alternate-Protocol": "80:quic,p=0.01,80:quic,p=0.01",
    "Cache-Control": "private, max-age=0",
    "Content-Type": "text\/xml; charset=UTF-8",
    "Date": "Tue, 28 Oct 2014 12:44:22 GMT",
    "ETag": "X8aekjl3CvT45xpcepn6EK2pDJw",
    "Expires": "Tue, 28 Oct 2014 12:44:22 GMT",
    "Last-Modified": "Tue, 28 Oct 2014 12:44:19 GMT",
    "Server": "GSE",
    "Transfer-Encoding": "chunked",
    "X-Content-Type-Options": "nosniff",
    "X-XSS-Protection": "1; mode=block"
  },
  "responseTime": 299,
  "rss": {
    "channel": {
      "copyright": "Copyright 2014 Cable News Network LP, LLLP.",
      "description": "CNN.com delivers up-to-the-minute news and information on the latest top stories, weather, entertainment, politics and more.",
      "image": {
        "description": "CNN.com delivers up-to-the-minute news and information on the latest top stories, weather, entertainment, politics and more.",
        "height": "33",
        "link": "http:\/\/edition.cnn.com\/index.html?eref=edition",
        "title": "CNN.com - Top Stories",
        "url": "http:\/\/i.cdn.turner.com\/cnn\/.e\/img\/1.0\/logo\/cnn.logo.rss.gif",
        "width": "144"
      },
      "info": {
        "feedburner": "http:\/\/rssnamespace.org\/feedburner\/ext\/1.0",
        "uri": "rss\/edition"
      },
      "item": [
        {
          "description": "The South African state is to appeal both judgment and sentence after athlete Oscar Pistorius was jailed for five years for shooting his girlfriend.",
          "guid": "http:\/\/edition.cnn.com\/2014\/10\/27\/justice\/south-africa-oscar-pistorius-appeal\/index.html",
```


(http://developer.ibm.com/mobilefirstplatform/wp-content/uploads/sites/32/2014/07/03_04_adapter_proceudre_invocation_result1.png)

Debugging with WireShark

Wireshark is a network protocol analyzer that can be used to see what happens in the network.
You can use filters to follow only what is required.
For more information, see the WireShark (http://www.wireshark.org) website.


Capturing from 2 interfaces [Wireshark 1.8.1 (SVN Rev 43946 from /trunk-1.8)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: http && tcp.port == 10080
Expression... Clear Apply Save Filter Filter Filter Filter
No. Time Source Destination Protocol Length Info
11007 300.268344 9.148.225.225 9.148.23.122 HTTP 518 HTTP/1.1 403 Forbidden (application/json
11228 349.498153 9.148.23.122 9.148.225.225 HTTP 482 GET /zzz/apps/services/api/zzz/android/an
11230 349.507158 9.148.225.225 9.148.23.122 HTTP 71 HTTP/1.1 403 Forbidden (text/html)
11282 358.387361 9.148.23.122 9.148.225.225 HTTP 385 POST /PushNotificationsProject/apps/servi
11288 358.405062 9.148.225.225 9.148.23.122 HTTP 523 HTTP/1.1 200 OK
11806 459.279102 9.148.23.122 9.148.225.225 HTTP 447 GET /PushNotificationsProject/apps/servic
11807 459.281675 9.148.225.225 9.148.23.122 HTTP 497 HTTP/1.1 200 OK (text/html)
11810 459.321210 9.148.23.122 9.148.225.225 HTTP 188 POST /PushNotificationsProject/apps/servi
11814 459.331828 9.148.225.225 9.148.23.122 HTTP 3419 HTTP/1.1 200 OK
12019 467.480230 9.148.23.122 9.148.225.225 HTTP 94 POST /PushNotificationsProject/apps/servi
12021 467.543998 9.148.225.225 9.148.23.122 HTTP 417 HTTP/1.1 200 OK (application/json)
12026 467.625773 9.148.23.122 9.148.225.225 HTTP 188 POST /PushNotificationsProject/apps/servi
12028 467.628632 9.148.225.225 9.148.23.122 HTTP 621 HTTP/1.1 401 Unauthorized (application/j
12031 467.686034 9.148.23.122 9.148.225.225 HTTP 190 POST /PushNotificationsProject/apps/servi
12034 467.715583 9.148.225.225 9.148.23.122 HTTP 2230 HTTP/1.1 200 OK (application/json)
Frame 10706: 385 bytes on wire (3080 bits), 385 bytes captured (3080 bits) on interface 1
Ethernet II, Src: Cisco_bf:16:3f (00:22:55:bf:16:3f), Dst: WistronI_ce:b2:ca (f0:de:f1:ce:b2:ca)
Internet Protocol Version 4, Src: 9.148.23.122 (9.148.23.122), Dst: 9.148.225.225 (9.148.225.225)
Transmission Control Protocol, Src Port: 56781 (56781), Dst Port: amanda (10080), Seq: 751, Ack: 1, Len: 319
[2 Reassembled TCP Segments (1069 bytes): #10705(750), #10706(319)]
Hypertext Transfer Protocol
Line-based text data: application/x-www-form-urlencoded