Remote debugging on Android or iOS devices

Matt YIU, Man Tung (mtyiu@cse)

SHB 118

Office Hour: Tuesday, 3-5 pm

2015.03.19
Outline

• Why remote debugging?
• Requirements
• Remote debugging on Android
• Remote debugging on iOS
Why remote debugging?

- **Mobile web development** is becoming more popular
  - “The way your web content behaves on mobile can be dramatically different from the desktop experience” (from https://developer.chrome.com/devtools/docs/remote-debugging)
Sidetrack: Designing for mobile devices

• Differences for designing for mobile devices:
  – Working with small screens
    • Solved by responsive web design
  – Working with touch screens
  – Optimizing images
    • Also solved by responsive web design
    • Use images with retina screen support (keyword: @2x)
  – Mobile APIs
    • Introduced by HTML5
    • New possibilities offered by mobile devices, such as orientation and geolocation

Why remote debugging?

- Mobile web development is becoming more popular
- For desktop, we have the developer tools for debugging
- For mobile devices, how can we debug our webpages?
- Remote debugging addresses the need!
  - With remote debugging, you can debug live content on your mobile device from your development machine!
Requirements

• For **Android** users, you need:
  – Chrome 32 or later installed on your development machine (Mac/Windows/Linux)
  – A USB cable to connect your Android device
  – **Chrome for Android** installed on your **Android 4.4+** device

• For **iOS** users, you need:
  – Safari on your development machine (**Mac only**)
  – A USB cable to connect your iOS device
  – Safari and **iOS 6** or later on your iOS device

• For those do not have an Android or iOS device:
  – Use **emulator** for **Android** (Mac/Windows/Linux) or **iOS** (Xcode on Mac)

rgba(237, 28, 36, 0.25):
Debugging Firefox for Android is also possible. But, we will skip it because of the time limit...
Remote debugging on Android

Warning: Chrome on iOS is not supported!
Setting up your Android device

• **Step 1**: Enable **USB debugging**
  – Select “**Settings > Developer options**”
Setting up your Android device

• **Step 1:** Enable **USB debugging**
  – Select “**Settings > Developer options**”

*If you cannot find the options...*

• It is because the developer options are hidden by default on Android 4.2 and later
• To enable the developer options, select “**Settings > About phone/tablet**” and tap “**Build number**” 7 times
Setting up your Android device

- **Step 2:** In **Developer options**, select the **USB debugging** checkbox.
  - An alert prompts you to allow USB debugging. Tap **OK**.
Setting up your Android device

• **Step 3:** Connect the device to your development machine using a USB cable
  – Install the appropriate USB driver for your device for Windows
  – See OEM USB Drivers on the Android Developers’ site
Discovering devices in Chrome

• After setting up remote debugging on Android, **discover** your device on your **desktop** Chrome browser

• **Step 1:** Navigate to “**chrome://inspect**” and confirm that **Discover USB devices** is checked
  – Alternatively, you can select **Chrome menu > More tools > Inspect Devices**
Discovering devices in Chrome

- **Step 2:** On your **Android device**, an alert prompts you to allow USB debugging from your computer. Tap **OK**
  - The message **USB debugging connected** displays in the device's notification drawer.

  To skip this alert in the future, check **Always allow from this computer**.
Discovering devices in Chrome

• **Step 3**: On your computer, the `chrome://inspect` page displays every connected device, together with its **open tabs** and **debug-enabled WebViews**
  – Consult the Troubleshooting section if you cannot see them
Debugging remote browser tabs

• Now you are ready to launch DevTools and debug your remote browser tabs
• Click inspect below the browser tab you want to debug
Debugging remote browser tabs

- A new instance of **Chrome DevTools** launches on the computer
  - The usage of this DevTools is the same as that of the one for desktop
Debugging tips

• Use **F5** (or **Cmd+R** on Mac) to reload a remote page from the DevTools window

• Keep the device on a **cellular network** and use the **Network** panel to view the **network waterfall** under actual mobile conditions

• Use the **Timeline** panel to analyze **rendering** and **CPU usage**

• If you’re running a local web server, use **port forwarding** or **virtual host mapping** to access the site on your device
References

- The instructions are mostly copied from https://developer.chrome.com/devtools/docs/remote-debugging
  - Content available under the CC-By 3.0 license
Remote debugging on Android or iOS devices

Will be skipped

Remote debugging on Android

In case there are some Firefox fans...
Prerequisites

• A desktop computer with **Firefox 36+** installed
• An Android device with **Firefox for Android 35+** installed
• A **USB cable** to connect the two devices
Setting up your Android device

- **Step 1**: Again, you need to enable **USB debugging** on your Android device.
- Please follow **pp. 8-11** to enable USB debugging.
Setting up your Android device

- **Step 2**: Enable **remote debugging** in Firefox for Android
- Open the browser and open its menu, select **Settings > Developer tools**. Check the “Remote debugging” box.
**Configuring Firefox on Desktop**

- You desktop Firefox needs to have the **ADB Helper** add-on, version 0.7.1 or higher.
- It will be installed automatically the first time you open the **WebIDE**.
- To open WebIDE, click the menu button > **Developer** > **WebIDE**.
Configuring Firefox on Desktop

- To verify that ADB Helper is installed, navigate to “about:addons” and select the “Extension” tab.
Connecting devices in Firefox

• **Step 1:** In WebIDE, click “Select Runtime” and select the Android device you want to debug.
Connecting devices in Firefox

• **Step 2:** On your **Android device**, an alert prompts you to allow USB debugging from your computer. Tap **OK**

![Incoming Connection](image_url)

An incoming request to permit remote debugging connection was detected. A remote client can take complete control over your browser! Allow connection?

- Disable
- Cancel
- OK
Debugging remote browser tabs

• Now you are ready to launch **Firefox Developer Tools** and debug your **remote browser tabs**

• In the WebIDE, click “**Main Process**” and select the remote tab that you want to debug
Debugging remote browser tabs

• The Firefox Developer Tools will open up and you can start debugging!
References

• The instructions are mostly adapted from https://developer.mozilla.org/en-US/docs/Tools/Remote_Debugging/Debugging_Firefox_for_Android_with_WebIDE
Remote debugging on iOS

**Warning:** Safari for Windows is not supported!
Setting up your iOS device

- Enable **Web Inspector**
  - Go to the **Settings** app and choose **“Safari > Advanced”**
  - Toggle on the **Web Inspector** switch

---

Remote debugging on Android or iOS devices

---

Prepared by Matt YIU, Man Tung  
2015.03.19
Setting up Safari on Mac

- Enable **Develop menu**
  - In desktop Safari, go to “**Safari > Preferences...**” menu
  - Select the **Advanced** tab
  - Select the “**Show Develop menu in menu bar**” checkbox
Discovering devices in Safari

• Connect the device to your development machine using a **USB cable**
• To begin inspecting, simply enter the URL of the page you want to inspect on the device
• At the same time, open Safari on Mac
Debugging remote browser tabs

- Choose the site from the “Develop” menu in Safari
Debugging remote browser tabs

- **Safari’s Web Inspector** will open up and you can start remote debugging!

*Mobile*

*Desktop*
Remote debugging for other browsers

• You may find some useful information on:
  – http://stackoverflow.com/questions/11262236/ios-remote-debugging

• Disclaimer: I didn’t try the tools listed on these pages 😊

• Good luck to your assignment and project (if you are using the mobile browser as the controller)!

– End –