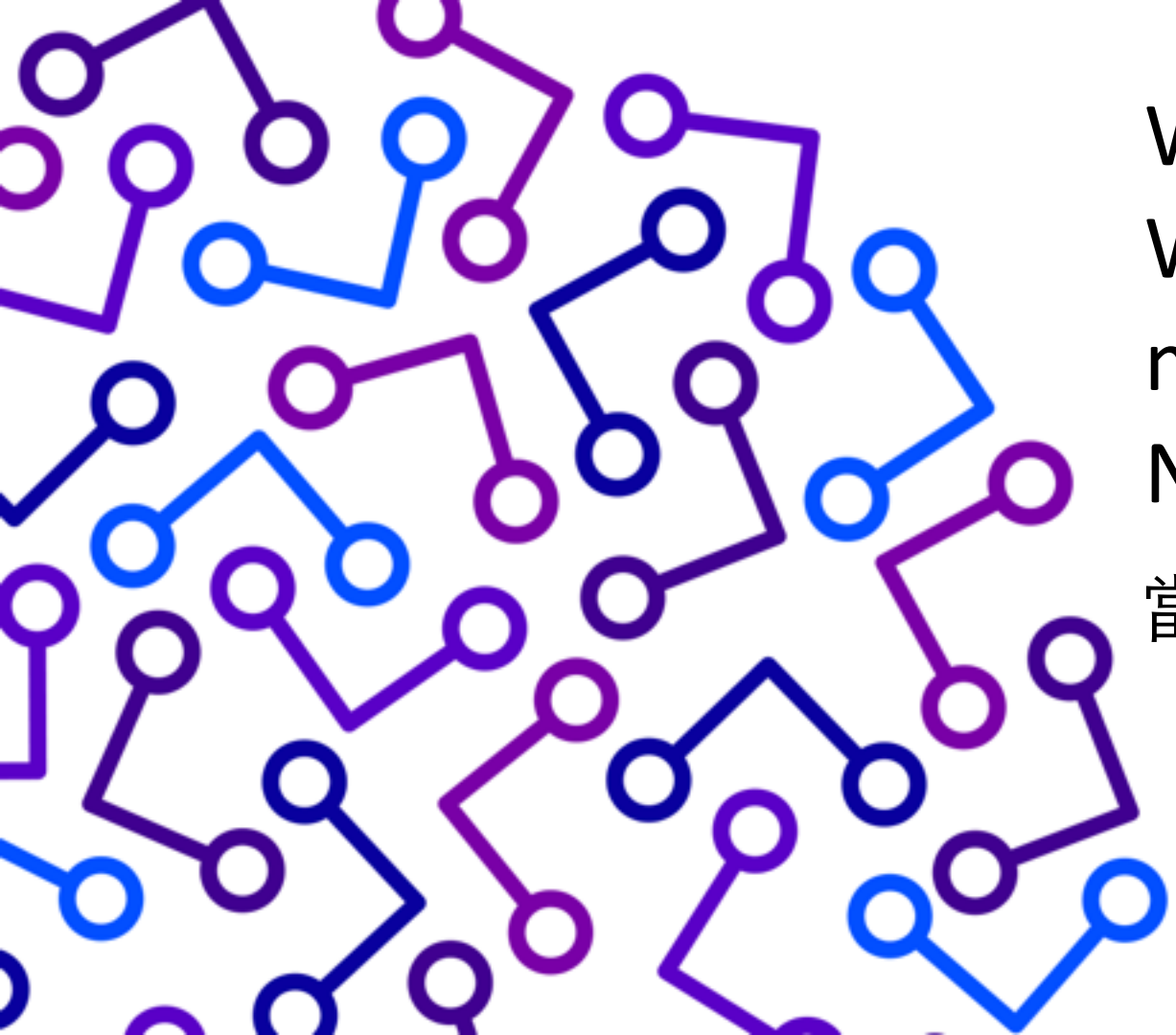


When Web meet Native App

當網頁遇上原生App

Eric Chuang

YAHOO!

When Web meet Native App

Yahoo Taiwan Mobile Team

Eric Chuang

YAHOO!

About Me

- Eric Chuang (ddsakura)
- WebConf 2013 Speaker
- Yahoo Lead Engineer
- Full Stack Engineer ?
  - Developed Yahoo E-Commerce Mobile Web
  - Developed Yahoo E-Commerce 超級商城 iPhone App
  - Developed Yahoo E-Commerce 拍賣 Android App

YAHOO!

Auction Android App

猜猜哪裡是 Web？

Web in Auction Android App

YAHOO!

What is Webview

- A View that displays web pages.
- Android
  - Since API 1
- iOS
  - UIWebView
  - MKWebView

曾經我以為

Android Webview > iOS UIWebview

(can not use Nitro JavaScript Engine)

Android Webview == Android Browser

desktop browsers | tablets | mobiles | other | latest | search

OVERVIEW

	Chrome	Firefox	Internet Explorer	Opera	Safari
Upcoming			TP 343		
Current	39 501	35 449	11 336	26 497	8.0 396
Older	38 499	34 449	10 297	24 485	7.0 352
	37 494	33 444	9 113	23 462	6.0 326
	35 482	32 444	8 33	22 473	5.1 250
	34 475	31 446		20 461	
	26 458	30 434		18 453	
	10 269	28 416		12.10 338	

http://beta.html5test.com/

YAHOO!

desktop browsers | tablets | mobiles | other | latest | search

OVERVIEW

	Android	BlackBerry	Chrome	Firefox	iOS	Opera	Windows Phone
Upcoming							
Current	5.0 452	10.3 449	39 493	35 456	8.0 405	26 489	8.1 346
Older	4.4.3 396	10.2 440	34 479	34 456	7.0 363	12.10 338	8 300
	4.4 379	7 225		33 451	6.0 329	11.50 258	7.5 113
	4.0 222			28 429	5.1 269		
	1.6 34			24 417	4.2 187		
					4.1 171		
					3.1 116		

http://beta.html5test.com/

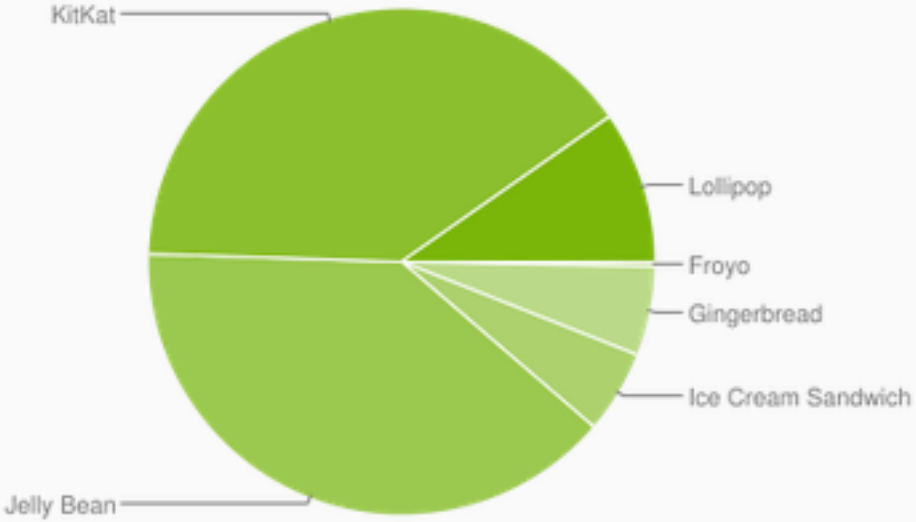
YAHOO!

結果...

Android Webview != Android Browser

Android Webview v.s. iOS UIWebview/MKWebview

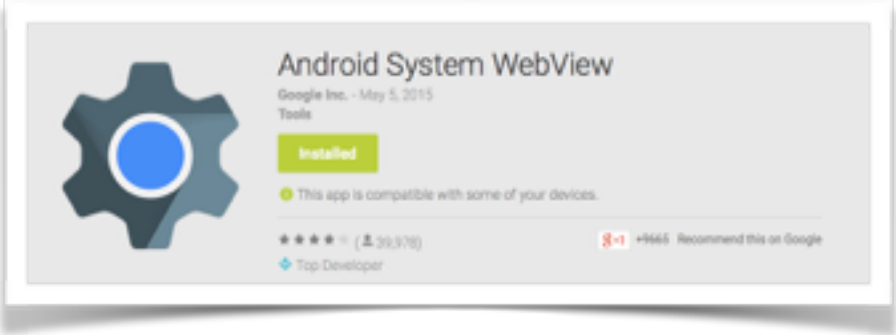
Version	Codename	API	Distribution
2.2	Froyo	8	0.3%
2.3.3 - 2.3.7	Gingerbread	10	5.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	5.3%
4.1.x	Jelly Bean	16	15.6%
4.2.x		17	18.1%
4.3		18	5.5%
4.4	KitKat	19	39.8%
5.0	Lollipop	21	9.0%
5.1		22	0.7%



Data collected during a 7-day period ending on May 4, 2015.
Any versions with less than 0.1% distribution are not shown.

Android Webview version

- Android 4.4
  - Chromium 30
- Android 4.4.3
  - Chromium 33
- Android 5.0
  - Chromium 37
  - And in google play
    - https://play.google.com/store/apps/details?id=com.google.android.webview



YAHOO!

	WebView v30	WebView v33	WebView v36
WebGL	x	x	✓
WebRTC	x	x	✓
WebAudio	x	x	✓
Fullscreen API	x	x	x
Form validation	x	✓	✓
Filesystem API	x	x	x
File input type	x	x	x
<datalist>	x	✓	✓

YAHOO!

Android Webview version

- Android before 4.4
  - Old Webview
  - Vendor may “improve” their webview
  - ref: http://slides.com/html5test/the-android-browser#/12

SO WE NO LONGER HAVE ONE WEBVIEW FOR EACH ANDROID VERSION

BUT

ONE FOR SAMSUNG,

AND ONE FOR HTC,

AND ONE FOR ...

Let’s start webview

- We need permission

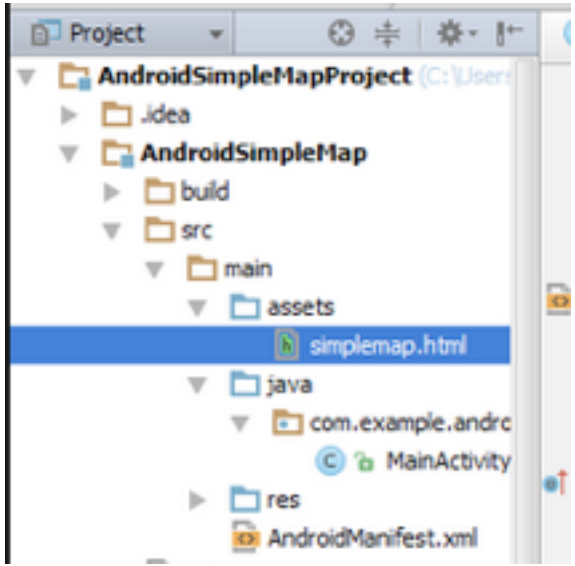
```
<uses-permission android:name="android.permission.INTERNET" />
```

- Basic usage

```
WebView mWebview = new WebView(this);

mWebview.loadUrl(“file:///android_asset/www/index.html”);

mWebview.loadUrl(“http://tw.yahoo.com/“);

String summary = “<html><body>Hello World</body></html>”;
mWebview.loadData(summary, "text/html", null);
```

Local Assets

- file:///android_asset/
  - The assets directory of an Android app is located at **src/main/assets** inside your Android Studio project.

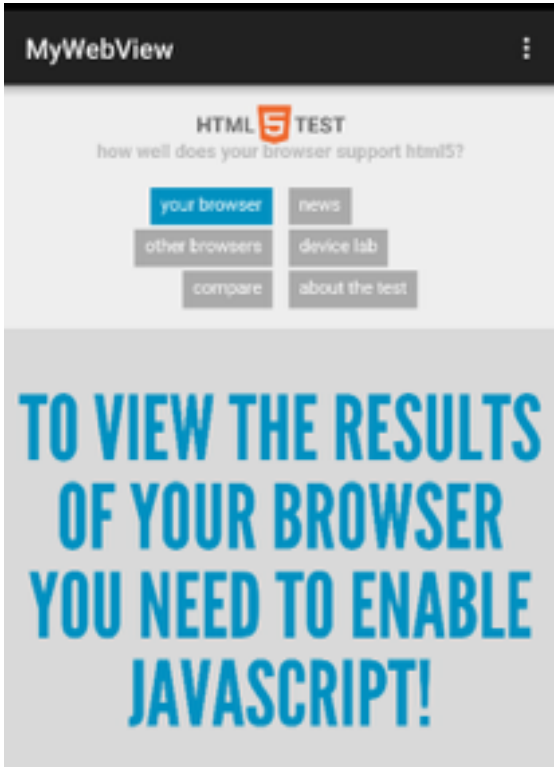


YAHOO!

How about JavaScript

- WebViews don't allow JavaScript by default.

```
// Enable JavascriptWebSettings
webSettings = mWebView.getSettings();
webSettings.setJavaScriptEnabled(true);
```



YAHOO!

Configure webview

- Websettings
  - setJavaScriptEnabled
    - The default is false.
  - setGeolocationEnabled
    - The default is true.
    - ACCESS_COARSE_LOCATION, ACCESS_FINE_LOCATION
  - setBuiltInZoomControls
    - The default is false.
  - setDomStorageEnabled
    - The default value is false.

http://developer.android.com/reference/android/webkit/WebSettings.html

Configure webview cont.

- UserAgent
  - getUserAgentString
  - setUserAgentString

Mozilla/5.0 (Linux; U; Android 4.1.1; en-gb; Build/KLP) AppleWebKit/534.30 (KHTML, like Gecko) Version/4.0 Safari/534.30

Mozilla/5.0 (Linux; Android 4.4; Nexus 5 Build/_BuildID_) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/30.0.0.0 Mobile Safari/537.36

https://developer.chrome.com/multidevice/user-agent

YAHOO!

WebViewClient

WebChromeClient

WebViewClient

- Instance of WebViewClient that is the client callback.
- It will be called when things happen that impact the rendering of the content,
- Functions
  - onLoadResource
  - onPageStart
  - onPageFinish
  - onReceiveError
  - shouldInterceptRequest

http://developer.android.com/reference/android/webkit/WebViewClient.html

WebChromeClient

- Instance of WebChromeClient for handling all chrome functions.
- This class is called when something that might impact a browser UI happens, for instance, progress updates and JavaScript alerts are sent here
- Functions
  - onCloseWindow
  - onCreateWindow
  - onJsAlert
  - onJsPrompt

http://developer.android.com/reference/android/webkit/WebChromeClient.html

Example between WebViewClient and WebChromeClient

```
// Let's display the progress in the activity title bar, like the
// browser app does.
getWindow().requestFeature(Window.FEATURE_PROGRESS);

webview.getSettings().setJavaScriptEnabled(true);

final Activity activity = this;
webview.setWebChromeClient(new WebChromeClient() {
  public void onProgressChanged(WebView view, int progress) {
    // Activities and WebViews measure progress with different scales.
    // The progress meter will automatically disappear when we reach 100%
    activity.setProgress(progress * 1000);
  }
});
webview.setWebViewClient(new WebViewClient() {
  public void onReceivedError(WebView view, int errorCode, String description, String failingUrl) {
    Toast.makeText(activity, "Oh no! " + description, Toast.LENGTH_SHORT).show();
  }
});

webview.loadUrl("http://developer.android.com/");
```

YAHOO!

Handling Links

- Default behavior: load that URL of the link in the default Android browser.

Intercept the url !!

YAHOO!

Intercepting WebView HTTP Requests

- public boolean shouldOverrideUrlLoading (WebView view, String url)
  - Give the host application a chance to take over the control when a new url is about to be loaded in the current WebView.

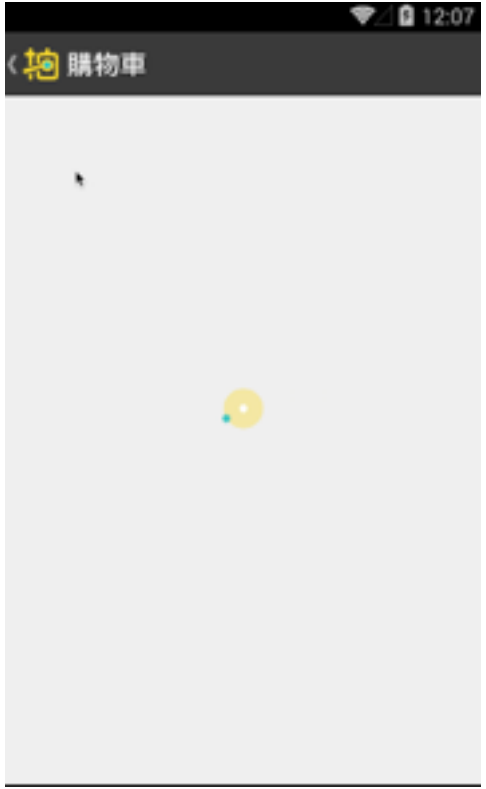
YAHOO!

Intercepting example 1

```
if (path.contains(ECWebView.WEB_URL_ECAUCTION_TYPE_PRODUCT_ITEM)) {
        showItemPage(url); // call native component
        return true;
}

if (path.contains(ECWebView.WEB_URL_ECAUCTION_TYPE_SELLER_BOOTH)) {
        showSellerBooth(url); // call native component
        return true;
}
```

Intercepting example 2

```
if (whiteList.indexOf(host) != -1) {
    toggleLoadingUI(true);
    return false;
}
```



From JavaScript to Java - Android Part

```
public class WebAppInterface {
    Context mContext;

    /** Instantiate the interface and set the context */
    WebAppInterface(Context c) {
        mContext = c;
    }

    /** Show a toast from the web page */
    @JavascriptInterface
    public void showToast(String toast) {
        Toast.makeText(mContext, toast, Toast.LENGTH_SHORT).show();
    }
}
```

```
webView.addJavascriptInterface(new WebAppInterface(this), "Android");
```

YAHOO!

From JavaScript to Java - HTML Part

```
<input type="button" value="Say hello" onClick="showAndroidToast('Hello Android!')" />

<script type="text/javascript">
   function showAndroidToast(toast) {
      Android.showToast(toast);
   }
</script>
```

YAHOO!

From Java to JavaScript

```
mWebView.loadUrl("javascript:window.cartList.closeOverlay()");
```

YAHOO!

Navigating web page history

```
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    // Check if the key event was the Back button and if there's history
    if ((keyCode == KeyEvent.KEYCODE_BACK) && myWebView.canGoBack()) {
        myWebView.goBack();
        return true;
    }
    // If it wasn't the Back key or there's no web page history, bubble up to the default
    // system behavior (probably exit the activity)
    return super.onKeyDown(keyCode, event);
}
```

Cache Web Resources

```
@Override
public WebResourceResponse shouldInterceptRequest(WebView view, String url) {

    if(url.startsWith("http://mydomain.com/article/") {
        String cacheFileName = url.substring(url.lastIndexOf("/"), url.length());
        this.urlCache.register(url, cacheFileName,
                "text/html", "UTF-8", 60 * UrlCache.ONE_MINUTE);
    }

    return this.urlCache.load(url);
}
```

YAHOO!

Android Lollipop

- Mixed content issue

```
if (android.os.Build.VERSION.SDK_INT >= 21) {
    webSettings.setMixedContentMode(WebSettings.MIXED_CONTENT_ALWAYS_ALLOW);
}
```

Cookie

- CookieManager
  - Manages the cookies used by an application's WebView instances.
    - setCookie(String url, String value)
    - getCookie(url)
- CookieSyncManager
  - This class was deprecated in API level 21. The WebView now automatically syncs cookies as necessary. You no longer need to create or use the CookieSyncManager. To manually force a sync you can use the CookieManager method flush() which is a synchronous replacement for sync().

Handle Http Error

```
@Override
public void onReceivedError(WebView view, int errorCode, String description, String
failingUrl) {
    …
}

@Override
public void onReceivedSslError(WebView view, SslErrorHandler handler, SslError error) {
    handler.proceed(); // Ignore SSL certificate errors
}
```

YAHOO!

Handle Http Error bug

- onReceivedError STILL does not receive HTTP Errors
  - https://code.google.com/p/android/issues/detail?id=82069

```
@Override
public void onPageFinished(WebView view, String url) {

    if (WEBPAGE_ERROR_HTML_TITLE.indexOf(mWebContentTitle) != -1) {
       mListener.onPageReceivedError(view, WEBPAGE_ERROR_CODE, mWebContentTitle, url);
    }

}
```

YAHOO!

Debugging webview

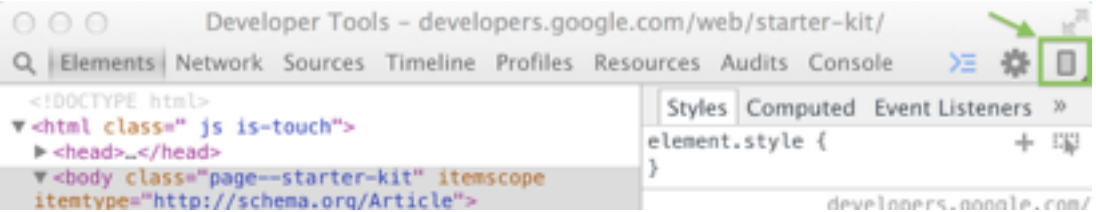
- Requirements
  - Chrome 32 or later installed on your development machine.
  - A USB cable to connect your Android device.
  - For app debugging: Android 4.4+ and a WebView configured for debugging.

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT) {
    WebView.setWebContentsDebuggingEnabled(true);
 }
```

YAHOO!

Debugging webview cont.

- The **chrome://inspect** page displays a list of debug-enabled WebViews on your device.
- To start debugging, click inspect below the WebView you want to debug.
- As of KitKat 4.4.3, **screencast** is available for both browser tabs and Android WebViews.



Developer Tools - developers.google.com/web/starter-kit/

developers.google.com/web/starter-kit/

Elements | Network | Sources | Timeline | Profiles | Resources | Audits | Console

Google Developers

Web Starter Kit

Web **Starter Kit**

Boilerplate & Tooling for Multi-Device Development

Download Kit (0.5.1)

```
<!DOCTYPE html>
▼<html class=" js is-touch">
  ▶<head>…</head>
  ▼<body class="page--starter-kit" itemscope itemtype=
  "http://schema.org/Article">
    ▶<header class="main-header">…</header>
    ▼<div class="wsk-header centered themed--
    background">
      ▼<div class="container">
        ▶<h1>…</h1>
          <p>Boilerplate & Tooling for Multi-Device
          Development</p>
          <a id="download" href="https://github.com/
          google/web-starter-kit/releases/download/v0.5.1/
          web-starter-kit-0.5.1.zip" class="button--
          secondary-variation themed">Download Kit
          (0.5.1)</a>
        </div>
        ::after
      </div>
    ▶<div class="container">…</div>
    ▶<div class="divider divider--secondary">…</div>
    ▶<div class="container">…</div>
    ▶<ul class="guides-list container">…</ul>
    ▶<footer class="main-footer">…</footer>
    <script type="text/javascript" async src="https://
    www.google-analytics.com/analytics.js"></script>
    <script async src="//www.googletagmanager.com/
    gtm.js?id=GTM-MBJLRF"></script>
    ▶<script type="text/javascript">…</script>
    <!-- Google Tag Manager -->
    ▶<noscript>…</noscript>
    ▶<script>…</script>
    <!-- End Google Tag Manager -->
  </body>
</html>
```

html.js.is-touch | body.page--starter-kit

Styles | Computed | Event Listeners »

```
element.style {
}

media="screen"                    developers.google.com/
body {                                  styles.min.css:1
  font-family: Helvetica,Arial,sans-serif;
  font-size: 16px;
  line-height: 1.625em;
  font-weight: 300;
  color: #404040;
  position: relative;
}

media="screen"                    developers.google.com/
body, button, html {                    styles.min.css:1
  -webkit-font-smoothing: antialiased;
  font-smoothing: antialiased;
}

media="screen"                    developers.google.com/
body {                                  styles.min.css:1
  margin: 0;
}

media="screen"                    developers.google.com/
* {                                     styles.min.css:1
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  -ms-box-sizing: border-box;
  box-sizing: border-box;
}

body {                            user agent stylesheet
  display: block;
  margin: 8px;
}

Pseudo ::after element

media="screen"                    developers.google.com/
body:after {                            styles.min.css:1
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  bottom: 0;
```

Find in Styles

Console | Search | Emulation | Rendering

<top frame> | Preserve log

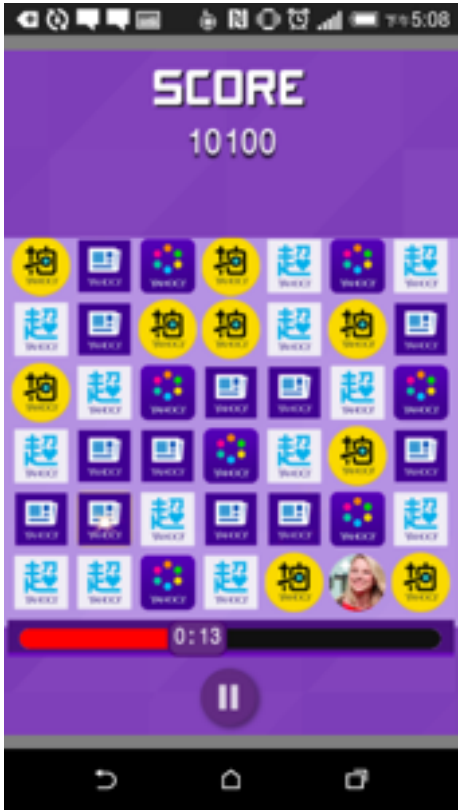
Testing with Webview

- Robotium
  - https://code.google.com/p/robotium/

```
// Type in text box
WebElement txtSearch = solo.getWebElement(By.name("q"), 0);
txtSearch.setTextContent("Hello");

// click button
WebElement btnSearch = (WebElement) solo.getWebElement(By.name("btnG"), 0);
solo.clickOnWebElement(btnSearch);
```

Project that Improve Webview

- Crosswalk
  - https://crosswalk-project.org/
  - Develop around device fragmentation
  - Provide a feature rich experience all on Android 4.x devices
  - Easily debug with Chrome DevTools
  - Improve the performance of HTML, CSS, and JavaScript



YAHOO!

Conclusion

- 這不是非黑即白的世界
  - Web 跟 Native App 亦然
- 讓 Web 和 Native App 共舞
  - 提供使用者最好的體驗

Thank you

http://bit.ly/1JkyR1W

Eric Chuang

YAHOO!