Synchronized cross-device mobile testing

- Tools
  - GhostLab (Mac)
  - Adobe Edge Inspect CC (Mac, Windows)
  - Remote Preview (Mac, Windows, Linux)
  - Grunt + Live-Reload (Mac, Windows, Linux)
  - Emmet LiveStyle
- Conclusions
  - Issues
- Recommendations

If you're a web developer targeting the multi-device web, you likely have to test your sites and web apps across a number of different devices and configurations. Synchronized testing can help here and is an effective way to automatically perform the same interaction across a number of devices and browsers at the same time.

Synchronized testing can help solve two particularly time-consuming problems:

- **Keeping all your devices in sync with the URL you want to test.** Manually loading them on each device is so yesterday, takes longer and makes it easier to miss out on regressions.
- **Synchronizing interactions.** Being able to load up a page is great for visual testing, but for interaction testing you ideally also want to be able to synchronize scrolls, clicks and other behaviours.

Thankfully if you have access to some of your target devices, there are a number of tools aimed at improving the flow from your desktop to your mobile devices. In this article, we will look at Ghostlab, Remote Preview, Adobe Edge Inspect and Grunt.

> {% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/rfxmvUNtDJ2uOtnIblYH.jpg", alt="This is my desk. Well, it used to be my desk. It's now just a mobile museum.", width="800", height="598" %}

This is my desk. Well, it used to be my desk. It's now just a mobile museum.

Tools

GhostLab (Mac)

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/tykqmqiVwnOsRrmQektw.png", alt="GhostLab for Mac by Vanamco", width="800", height="517" %}
GhostLab for Mac by Vanamco

Ghostlab is a commercial Mac application ($49) designed to synchronise testing for sites and web apps across multiple devices. With minimum setup it allows you to simultaneously sync:

- Clicks
- Navigation
- Scrolls
- Form input (e.g login forms, sign-up)

This makes testing your site's end-to-end user experience on multiple devices very straightforward. Once you've opened your page in a browser on your devices, any changes to navigation (including refreshes) cause any other connected devices to refresh instantly. Ghostlab supports watching local directories, so this refresh also happens when you save edits to local files, keeping everything in sync always!

I found setting up Ghostlab a painless process. To get started, download, install and run the trial (or full version if you're in the buying mood). You will then want to connect your Mac and the devices you wish to test to the same Wifi network so that they're discoverable.

Once Ghostlab is running, you can either click "+" to add a URL for testing or simply drag it from your browser's address bar. Alternatively, drag the local folder you wish to test over into the main window and a new site entry will be created. For this article, I'm testing a live version of http://html5rocks.com. Cheeky, eh? ;)

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/OU4JcU5JRVLazL3tUgdf.png", alt="Choose a URL or local directory on your machine", width="800", height="410" %}
Choose a URL or local directory on your machine

You can then kick-start off a new Ghostlab server by clicking the ">" play button next to the name of your site. This starts a new server, available at an IP address specific to your network (e.g http://192.168.21.43:8080).

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/hAHpmYqgtsXruQhdUsIO.jpg", alt="Ghostlab server locally proxying content from our URL", width="800", height="598" %}
Ghostlab server locally proxying content from our URL

Here, I've connected up a Nexus 4 and pointed Chrome for Android to the IP address Ghostlab gave me. That's all I have to do. Ghostlab doesn't require that you install a dedicated client per device like some other solutions and it means you can start testing even more quickly.

Any device you connect to the Ghostlab URL will be added to the list of connected clients in the sidebar to the right of the main Ghostlab window. Double-clicking the device name displays additional details such as the screen size, OS and so on. You should now be able to test navigating and synchronizing clicks! Yay.

Ghostlab is also able to display some stats about connected devices like the UA string, viewport width and height, device pixel density, aspect ratio and more. At any time, you can manually change the base URL you are inspecting by clicking the settings cog next to an entry. This opens up a configuration window that looks like the below:

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/a9Zxmwcg1V108mX42H2e.png", alt="Configure files to watch, HTTP headers, character sets and more.", width="800", height="508" %}
Configure files to watch, HTTP headers, character sets and more.

I can now select one of my other connected devices, click on different links around HTML5Rocks and the navigation is synchronized both on my desktop Chrome (where I entered in the same Ghostlab URL) as well as across all of my devices.

What's even better is that Ghostlab has an option allowing you to proxy all links going through the network so that instead of a click on http://192.168.21.43:8080/www.html5rocks.com navigating to www.html5rocks.com/en/performance (for example), which would break the automatic cross-device refresh suffered by other solutions, it can just translate this link into http://192.168.21.43/www.htm5rocks.com/en/performance so that navigating is completely seamless across all my devices.

To enable, check "Load all content through Ghostlab" under the "Content Loading" tab.

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/xO5WRNa2fc9vzQLZb1QS.png", alt="Ghostlab can rewrite URLs so all resources are loaded through the Ghostlab proxy. Useful for synchronizing navigations to multiple pages", width="800", height="504" %}
Ghostlab can rewrite URLs so all resources are loaded through the Ghostlab proxy. Useful for synchronizing navigations to multiple pages

Seeing it in action:

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/eQE7t4VygF5xq7CJpoQV.jpg", alt="Synchronized testing of an Android, Windows 8 and Firefox OS phone with Ghostlab", width="800", height="598" %}
Synchronized testing of an Android, Windows 8 and Firefox OS phone with Ghostlab

Ghostlab is capable of loading any number of sites or windows across any supported browser. This doesn't just let you test your site at different resolutions, but how your code behaves on different browsers and platforms. Yay!

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/EkUMeViMnNTjdRLrVKQG.gif", alt="Synchronizing scroll, clicks and navigation across all of the test devices", width="623", height="351" %}
Synchronizing scroll, clicks and navigation across all of the test devices

Ghostlab allows you to configure the setup for the project workspace you're previewing and you can specify whether you would like changes to the directory to be watched and refreshed when detected. This means changes cause all connected clients to be refreshed. No more manual refreshes!

You may be wondering what else Ghostlab can help with. Whilst it's certainly not a swiss-army knife, it also supports remote code inspection on connected devices. Through the main interface, double-clicking on any device name should bring up a "debug" option which will launch a version of the Chrome DevTools for you to play around with.

Ghostlab makes this possible via the bundled Weinre remote web inspector, which lets you debug scripts and tweak styles on connected devices. Similar to the remote debugging experience available for Chrome for Android, you can select elements, run some performance profiling and debug scripts.

All in all, I was impressed with how quickly I was able to use Ghostlab for everyday testing across devices. If you're a freelancer, you might find the cost of the commercial license a little high (see below for more options). However, I'm happy to recommend Ghostlab otherwise.

Adobe Edge Inspect CC (Mac, Windows)

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/8U2DXCGXtItfi82pNOv1.png", alt="Adobe's Creative Cloud subscription includes Edge Inspect", width="800", height="512" %}
Adobe's Creative Cloud subscription includes Edge Inspect

Adobe Edge Inspect is part of the Adobe Creative Cloud subscription package, but also comes available as a free trial. It allows you to drive multiple iOS and Android devices with Chrome (via the Edge Inspector Chrome extension), so that if you browse to a particular URL all of your connected devices stay in sync.

To get set up, first sign up for an Adobe Creative Cloud account or login to an existing account if you already have one. Next, download and install Edge Inspect from Adobe.com (available for Mac and Windows at present, but not Linux - sorry!). Note the docs for Edge Inspect are useful to keep at hand.

Once installed, you'll want to get the Edge inspect extension for Chrome so that you can synchronize browsing between connected devices.

> {% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/cae6RFljZG7BS6zuNMrK.png", alt="The Edge Inspect CC Chrome Extension", width="325", height="214" %}
> The Edge Inspect CC Chrome Extension

You will also need to install an Edge Inspect client on each device you wish to sync actions with. Thankfully clients are available for iOS, Android and Kindle.

With the installation process behind us, we can now start inspecting our pages. You'll need to make sure all your devices are connected to the same network for this to work.

Start up Edge Inspect on your desktop, the Edge Inspect extension in Chrome and then the app on your devices (see below):

> {% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/RrPK7JYnuiGoC77xcj1Q.png", alt="Connecting a device up to the Edge Inspect extension", width="605", height="451" %}
> Connecting a device up to the Edge Inspect extension

We can now navigate to a site like HTML5Rocks.com on desktop and our mobile device will automatically navigate to the same page.

> {% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/XAvQlt7WHiuPELPsHJKv.jpg", alt="Driving navigation of a URL across multiple connected devices", width="800", height="598" %}
> Driving navigation of a URL across multiple connected devices

In the extension, you'll also now see your device listed with a <> symbol next to it as in the screenshot below. Clicking this will launch an instance of Weinre allowing you to inspect and debug your page.

> {% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/rWsxtUuWj17uJA92rrzG.png", alt="Connected devices appear with a <> symbol next to them, which can be used to launch the Weinre debugger", width="800", height="549" %}
> Connected devices appear with a <> symbol next to them, which can be used to launch the Weinre debugger

Weinre is a DOM viewer and console, and lacks features from the Chrome DevTools like JavaScript debugging, profiling, and network waterfall. While it is the bare minimum of developer tooling, it's

useful for sanity-checking DOM and JavaScript state.

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/vGeQ7zaldu9MPJCu8iFf.png", alt="Debugging with Weinre", width="800", height="456" %}
Debugging with Weinre

The Edge Inspect extension also supports generating screenshots from connected devices with ease. Useful for layout testing or just getting captures of your page to share with others.

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/Nc1Lf9Fv72IDRWD8o2v1.png", alt="Screenshot generation with Edge Inspect", width="547", height="296" %}
Screenshot generation with Edge Inspect

For developers already paying for CC, Edge Inspect is a great solution. It does however come with a few caveats such as each device requiring a dedicated client to be installed and a little extra setup work that you may not find with an alternative like Ghostlab.

Remote Preview (Mac, Windows, Linux)

Remote Preview is an open source tool where you host HTML pages and content that you can display on multiple devices.

Remote preview executes an XHR call at an interval of every 1100ms to check if a URL in a configuration file has changed. If it finds that it has, the script updates the src attribute of an iframe loaded into the page for each device, loading the page into it. If there are no changes detected, the script will continue polling until a change is made.

{% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/ekGQXOwVV8IPlwEkwVDm.gif", alt="Synchronized URL testing across 27+ devices", width="800", height="450" %}
Synchronized URL testing across 27+ devices

This is great for chaining devices together and easily changing a URL across all of them. To get started:

1. Download Remote Preview and move all of the files for it into a locally accessible server. This can be Dropbox, a development server or something else.
2. Load up "index.html" from this download on all of your target devices. This page will be used as a driver and will load up the page you want to test using an iframe.
3. Edit "url.txt" (included in the download and now served on your server) with the URL you wish to preview. Save this file.
4. Remote Preview will notice that the url.txt file has changed and will refresh all connected devices to load up your URL.

Whilst a lo-fi solution, Remote Preview is free and works.

Grunt + Live-Reload (Mac, Windows, Linux)

Grunt (and Yeoman) are command-line tools used for scaffolding and building projects on the front-end. If you're already using these tools and have live-reload setup, it's easy to update your workflow to enable cross-device testing where each change you make in your editor causes a reload in any of the devices you've opened up your local app on.

You're probably used to using `grunt server`. When run from the root directory of your project, it watches for any changes to your source files and automatically refreshes the browser window. This is thanks to the grunt-contrib-watch task which we run as part of the server.

If you happen to have used Yeoman to scaffold out your project, it will have created a Gruntfile with everything you need to get live-reload working on your desktop. To get it functioning cross-device, you just need to change one property, which is the `hostname` (on your desktop). It should be listed under `connect`. If you notice `hostname` is set to `localhost` just change it to 0.0.0.0. Next run `grunt server` and as usual, a new window should open displaying a preview of your page. The URL will probably look like http://localhost:9000 (9000 is the port).

Fire up a new tab or terminal and use `ipconfig | grep inet` to discover your system's internal IP. It may look like `192.168.32.20`. The last step is to open up a browser like Chrome on the device you would like to synchronize livereloads to and type in this IP address followed by the port number from earlier. i.e `192.169.32.20:9000`.

That's it! Live-reload should now cause any edits you make to source files on your desktop to trigger reloads in both your desktop browser *and* the browser on your mobile device. Awesome!

> {% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/lpWoeIDEbXAI917RMrii.jpg", alt="Saved edits on desktop now trigger a live-reload in your desktop browser as well as mobile browsers on devices with the same URL", width="800", height="598" %}
> Saved edits on desktop now trigger a live-reload in your desktop browser as well as mobile browsers on devices with the same URL
>
> {% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/MzRPsvU8WIadYdcUf81K.gif", alt="Cross-device live-reload in action. Each edit/save gives you a real-time for of your current page, great for responsive design testing.", width="735", height="413" %}
> Cross-device live-reload in action. Each edit/save gives you a real-time for of your current page, great for responsive design testing.

Yeoman also has a Mobile generator available which makes setting this workflow up a breeze.

Emmet LiveStyle

Emmet LiveStyle is a browser and editor plugin that brings live CSS editing to your development workflow. It is currently available for Chrome, Safari and Sublime Text and supports bi-directional (editor to browser and vice-versa) editing.

Emmet LiveStyle doesn't force a complete browser refresh when you make changes, but instead pushes CSS edits across the DevTools remote debugging protocol. What this means is that you can see changes made in your desktop editor in any connected version of Chrome, whether it be on desktop Chrome or Chrome for Android.

LiveStyle has what it calls "multi-view mode", which is ideal for testing and tweaking responsive designs across windows and devices. In multi-view mode, all editor updates are applied to all windows as are DevTools updates for the same page.

With the LiveStyle package installed, to get started with live CSS editing:

1. Start up Sublime Text and open up a CSS file in a project
2. Start Chrome and go to the page with the CSS you would like to edit
3. Open DevTools and go to the LiveStyle panel. Check the "Enable LiveStyle" option. Note: DevTools will need to be kept open during your live editing session for each window in order for style updates to be applied.
4. When this has been enabled, a list of stylesheets will be displayed on the left and a list of your editor files on the right. Select the editor file to be associated with the browser. That's it! Boom.

Now the list of editor files will be automatically updated when you edit, create, open or close files.

> {% Img src="image/T4FyVKpzu4WKF1kBNvXepbi08t52/LHsl7cCjGBztNAN77XKr.gif", alt="Changes to CSS in Sublime being instantly patched across different browser windows and an emulator.", width="781", height="586" %}
> Changes to CSS in Sublime being instantly patched across different browser windows and an emulator.

Conclusions

Cross-device testing is still a new and fast moving space with many new contenders in development. Thankfully there exist a number of free and commercial tools for assuring your compatibility and testing across a wide number of device sets.

That said, there's still a lot of potential for improvement in this area and we would encourage you to think about how the tooling for testing across devices can be further improved. Anything that reduces setup time and improves your cross-device workflow is a win.

Issues

Perhaps the largest issues I ran into during testing with these tools was devices regularly going to sleep. This isn't a deal-breaker, but does get annoying after a while. Where possible it's a good idea to set your devices to not sleep as a workaround, however, keep in mind that this can drain your battery unless you're always plugged in.

I didn't personally run into any major issues with GhostLab. At $49 some may find the price a little steep; however, keep in mind if you're using it regularly it more or less pays for itself. One of the nicest things about the setup was not having to worry about installing and managing a client per target device. The same URL worked everywhere.

With Adobe Edge Inspect, I found having to install and use specific clients on each device a little bit tedious. I also found it didn't consistently refresh all of the clients connected up, meaning I had to do this myself from the Chrome extension. It also requires a subscription to Creative Cloud and is limited to loading up sites in the client rather than in a selected browser on your devices. This may limit your ability to accurately test.

Remote Preview functioned as advertised, but is extremely lightweight. This means for anything more than refreshing your site across devices, you'll need a more advanced tooling option. It won't for example, synchronize clicks or scrolls.

Recommendations

If you're looking for a free cross-platform solution to get you started, I recommend using Remote Preview. For those working in a company looking for a paid-for solution, GhostLab has been consistently excellent in my experience but is only available for the Mac. For Windows users, Adobe Edge Inspect is your best call and minus some quirks, does generally get the job done.

Grunt and LiveStyle are also excellent for augmenting your live iteration during development.

□□□ #2
□ □□□ □□□ 2026-10-31 22:20:11 UTC
□ □□□ □□□ 2026-10-31 22:20:14 UTC