Hidden Gems in Browser Tools (notes)

1. UGM + ACP. <3 CounterMarch. Blog. DJ an hour of metal once a week on CodeBass radio (Tuesdays at 4 Eastern!). Co-founder of Philly Merge, conference for entrepreneurs & developers in the Philly area. Author & lead dev of a few notable CF open source projects: Taffy & CFScript Community Components.
2. Like to start with a quote
3. I can hear what some of you were thinking when he wrote that in 2007… "JavaScript? ... Really?!"
4. CF has come a long way since 1995 so it stands to reason that JS has too.
5. … Because of that, I like to say, "We're living in the future". JS + CSS are getting more powerful, but also more complex.
6. And alerts just don't cut it for debugging complex issues.
7. This is what we're really here to talk about. Tools! Not going to cover Opera in great detail, because of…
8. Reach over 50% of average internet users by supporting IE8 and IE9 and the latest version of Chrome. Add FF10 and IE7 to that and you're reaching more than 75%. But all notable versions of Safari combined add up to less than 7% (probably developers checking Safari compatibility!), and Opera's share is so small it's included as part of "Other" with the likes of Flock. Fun fact: According to Wikipedia, Opera is the most-used browser in both the Ukraine and Belarus.
9. So that's sort of a roundabout introduction to why you should care about these tools in the first place. Now let's talk about how to become a power user. We'll start in the **Console** tab.
10. The first thing you need to learn is console.log. Notice the browser icons. You'll see them throughout the slides, and they indicate which browsers support the features I'm talking about.
11. ** Going to show you lots of different tabs in the dev tools. In any of them other than console, hit **Esc** to get a mini-console!
12. ** Console.log takes multiple arguments, so you *could* log multiple variables, but I like to use this to label my debug output. You can also use **info**, **warn**, **error**, and **assert** methods.
13. ** Pretty self-explanatory. Similar to a terminal window.
14. ** An annoying thing you'll sometimes run into is that you're using console.log but then the page redirects and the console clears. Save yourself that headache and enable this setting.
15. ** Frames are a pain in the ass. But if you've got to deal with them here's how.
16. ** You can copy the result of something to your clipboard.
17. ** The console does math.
18. ** Let's move into the all-important **SCRIPTS** tab.

19. ** If you learn absolutely nothing else from this presentation, it should be that the step debugger is your best friend. [DEMO: breakpoints, TALK: JS error breakpoint, DOM change breakpoint, XHR URL match breakpoint, call stack]
20. ** Keyboard shortcuts! Of course: IE deviates from everyone else for seemingly no reason. And Mac users will recognize these as the system KB shortcuts for Exposé / Mission Control, so Chrome on OSX automatically remaps them. <3
21. ** Variable watches: cut down on the #of console.log's you do if interested in how a variable's value changes over time.
22. ** Un-minify de-compresses minified JS and makes it slightly more readable. Does not undo variable/function renaming that might happen during minification.
23. Moving into the **"Elements"/"HTML"/"Documents"** tab.
24. ** Getting element dimensions is really easy! How wide is that div? How tall is that UL?
25. When you look at this tab at first or after a page refresh, the code is collapsed. Expand all with a single keystroke.
26. ** Ever feel caught in an infinite loop of: change css, alt+tab to browser, refresh, alt+tab to IDE, repeat? Never again!
27. ** While doing real-time editing, use arrows & modifier keys to jump around sizes faster.
28. Pretty self-explanatory once you see it. (screenshots)
29. ** Maybe you like HEX? RGB? HSL?
30. Now we'll move on to the **Resources** tab.
31. Mostly handy when working with static HTML.
32. ** Snoop around in client-side storage without writing any JS. Includes HTML5 localStorage, cookies, cache manifest, sqlite, etc.
33. Lastly, the **Network** tab.
34. Disabling caching is useful for testing first-time users or those with cookies disabled, cases like that.
35. Can even spoof mobile browser UA strings, like iPhone, iPad, Android, and so on.
36. Network tab's biggest job is showing download latency & duration.
37. Also shows the two primary DOM events. DomContentLoaded = finished parsing, images/etc may not be finished downloading.
38. Chrome's KB shortcut is Ctrl/Cmd+Alt+I; FireBug is F12 by default but I like to remap this to F2 so it doesn't conflict with System stuff.
39. (Answer: Expand HTML KB Shortcut)
40. I've found un-docking useful when working on mobile-friendly websites, where I want the browser window very thin.