Advanced Debugging with Chrome Dev Tools

Lather your way to a new intelligence

Wayne Elgin @esjay
Wayne Elgin
@esjay

UX at Cantina.co
WE PROMISE TO
DEMONSTRATE KNOWLEDGE
ASK QUESTIONS & HELP OTHERS
LISTEN CAREFULLY AND BE HUMBLE

WE PROMISE TO
FIGHT MEDIOCRITY
AND EXPERIMENT OFTEN

REASON

PASSION

FIGHT
What You’re Here For

- Dev Tools configuration
  - The Panes Explained
- Element Debugging
  - Computed properties
  - Eyedropper & Color Picker
  - Keyframes editor
  - Bezier curve editor
  - Shadow editor
  - Animations
- Responsive & Simulation

- JavaScript Debugging
  - The Basics
  - Niche console usage
  - Conditional Breakpoints
  - Inline breakpoints
  - Event Breakpoints
  - XHR Breakpoints
  - DOM Breakpoints
  - Performance auditing
If you don’t have a local repo to hack on:


```
  npm i

  npm start
```
Why do Chrome Dev Tools matter?

- Gives full control over DOM
- Allows tweaking CSS rules and properties in real time
- Provides for live observation of JavaScript interpretation
- Audits user accessibility and overall experience
- Simulates multitudes of devices and network conditions
- Analyzes many facets of web performance
Why do Chrome Dev Tools matter?

Besides analyzing accessibility, JavaScript, and CSS in real time...

- Works remotely with [Chrome on Android](https://www.chrome.com)
- Used by [Electron](https://github.com/electron) -based projects (Atom, VS Code, Slack, Discord)
- Debugs Node via built-in `--inspect` and [node-inspector](https://www.npmjs.com/package/node-inspector)
- [Stetho](https://github.com/GoogleChrome/stetho) (Android) and [PonyDebugger](https://github.com/ponydebugger) (iOS) both use them for native apps

Wayne Elgin @esjay
For best results

Dev on Chrome Canary
10 Weeks From Canary to Stable
DevTools on the right
console drawer opened
DevTools experiments enabled

Wayne Elgin @esjay
For best results

AND USE THE DARK THEME

Wayne Elgin @esjay
CSS Debugging
“The rules of CSS club…”

“You do not talk about the color picker & eye dropper”

“Oh, and also, you do not talk about the bezier curve editor”

“Furthermore, let’s have no mention of the text or box shadow editors”

“And if this is your first time at CSS club, you have to use the Animations panel”

“Use the keyframes editor to neutralize the burn”
Wrangling Device Simulation
Wrangling Device Simulation

- A vertical Dev Tools viewport provides built-in resize-ability
- Use built-in presets for mobile devices via the Device Toolbar
- The Rendering tab in the bottom drawer can emulate print styles
Wrangling Device Simulation

- Use the Network Conditions tab...
  - When you are validating your caching strategy
  - When you are struggling with flashes of unstyled or unrendered content
  - When you need to present yourself as alternate browsers
Wrangling Device Simulation

- Use the Sensors tab...
  - When you need to emulate various geolocations
  - When you need to test device orientations
  - When you need to simulate touch-specific events
JavaScript Debugging
The Debugging Basics

deabugger;

console.log();

Set a breakpoint
Debugging in-**console** (ably)

Wayne Elgin [@esjay](https://twitter.com/esjay)
Being a good `console`tant

Use `,` (commas) not `+` for `console.log()` (try `%c`, `%s`, & `%d` formatters).

Use `.group()` or `.groupCollapsed()` & `.groupEnd()` for collapsable log chunks

Use `.assert(expression, string || object)` to print when assertions fail

Use `.table(objectArray)` to pretty-print arrays of objects

Export `console` via right-click, “Save as”

More at Google’s console API reference

Wayne Elgin @esjay
Mastering Breakpoints

Wayne Elgin @esjay
Mastering Breakpoints

- Conditional breakpoints
- Inline breakpoints
- XHR breakpoints
- Event breakpoints
- DOM breakpoints
Async Debugging
Async Debugging (Before a Month Ago)

Stack without Async checked

Stack with Async checked

Wayne Elgin @esjay
Performance Audits
JavaScript Performance Audits

Using the long-enough timeline
@aerotwist

more performance audits

Wayne Elgin @esjay
Bonus Points

Application tab for service workers, cookies, and storage

Set up *workspaces* for source mapping

Use `built-in --inspect` or [Node-Inspector](https://github.com/GoogleChrome/node-inspector) for DevTools in Node

Follow [@addyosmani](https://twitter.com/addyosmani) and [@ChromeDevTools](https://twitter.com/ChromeDevTools) and [@umaar](https://twitter.com/umaar) for great gifs

I tweet Dev Tools things all the time [@esjay](https://twitter.com/esjay)