Chrome Devtools Cheatsheet

Opening Devtools

To access the DevTools, on any web page or app in Google Chrome you can use one of these options:

- Open the **Chrome menu** ☰ at the top-right of your browser window, then select **Tools** > **Developer Tools**.
- Right-click on any page element and select **Inspect Element**.

	Windows / Linux	Mac
Open Developer Tools	F12, Ctrl + Shift + I	Cmd + Opt + I
Open / switch from inspect element mode and browser window	Ctrl + Shift + C	Cmd + Shift + C
Open Developer Tools and bring focus to the console	Ctrl + Shift + J	Cmd + Opt + J
Inspect the Inspector (*undock first one and press*)	Ctrl + Shift + J	Cmd + Opt + J

To open up the General Settings dialog type ? or F1 when the Developer Tools window is open. Press Esc to close the settings dialog.

All Panels

	Windows / Linux	Mac
Show General Settings dialog	?, F1	?
Next panel	Ctrl +]	Cmd +]
Previous panel	Ctrl + [	Cmd + [
Backward in panel History	Ctrl + Alt + [	Cmd + Alt + [
Forward in panel history	Ctrl + Alt +]	Cmd + Alt +]
Jump to panel 1-9 (*when enabled in General Settings*)	Ctrl + 1-9	Cmd + 1-9
Toggle Console / close settings dialog when open	Esc	Esc
Refresh the page	F5, Ctrl + R	Cmd + R
Refresh the page ignoring cached content	Ctrl + F5, Ctrl + Shift + R	Cmd + Shift + R
Text search within current file or panel	Ctrl + F	Cmd + F
Text search across all sources	Ctrl + Shift + F	Cmd + Alt + F
Search by filename (*except on Timeline*)	Ctrl + O, Ctrl + O	Cmd + O, Cmd + O
Restore default text size	Ctrl + 0	Shift + 0
Zoom in	Ctrl + +	Shift + +
Zoom out	Ctrl + -	Shift + -

Elements Panel

	Windows / Linux	Mac
Undo change	Ctrl + Z	Cmd + Z
Redo change	Ctrl + Y	Cmd + Y, Cmd + Shift + Z
Navigate	Up, Down	Up, Down
Expand / collapse node	Right, Left	Right, Left
Expand node	Single-click on tag	Single-click on tag
Edit attribute	Enter, Double-click on attribute	Enter, Double-click on attribute
Hide element	H	H
Toggle edit as HTML	F2	

Right-clicking an element you can:

- Force element psuedo states: (`:active` , `:hover` , `:focus` , `:visited`)
- Set breakpoints on the elements: (Subtree modifications, Attribute modification, Node removal)
- Clear console

Styles Sidebar

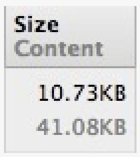
	Windows / Linux	Mac
Edit rule	Single-click	Single-click
Insert new property	Single-click on whitespace	Single-click on whitespace
Go to line of style rule property declaration in source	Ctrl + Click on property	Cmd + Click on property
Go to line of property value declaration in source	Ctrl + Click on property value	Cmd + Click on property value
Go to line of style rule property declaration in source	Ctrl + Click on property	Cmd + Click on property
Go to line of property value declaration in source	Ctrl + Click on property value	Cmd + Click on property value
Cycle through the color definition value	Shift + Click on color picker box	Shift + Click on color picker box
View auto-complete suggestions	Ctrl + Space	Cmd + Space
Edit next / previous property	Tab, Shift + Tab	Tab, Shift + Tab
Increment / decrement value	Up, Down	Up, Down
Increment / decrement value by 10	Shift + Up, Shift + Down	Shift + Up, Shift + Down
Increment / decrement value by 10	PgUp, PgDown	PgUp, PgDown
Increment / decrement value by 100	Shift + PgUp, Shift + PgDown	Shift + PgUp, Shift + PgDown
Increment / decrement value by 0.1	Alt + Up, Alt + Down	Opt + Up, Opt + Down

Emulate an element's pseudo state (`:active` , `:hover` , `:focus` , `:visited`)

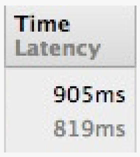
➕ Add new style selectors

Network Panel

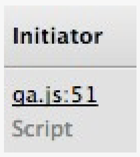
Understanding the information displayed within each column



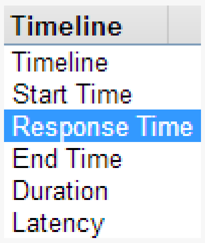
Size: Total size of resource
Content: Gzipped size of resource



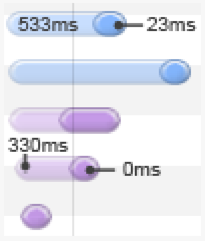
Time: total duration to get response
Latency: time taken to get first byte



File that initialized the resource load
How the resource load was scheduled



Select the Timeline heading to change sort modes for the network.



Transparent: Latency load time
Solid: Total load time

Export network data into HAR format (http://goo.gl/eDSY2)

Sources Panel

	Windows / Linux	Mac
Pause / resume script execution	F8, Ctrl + \	F8, Cmd + \
Step over next function call	F10, Ctrl + '	F10, Cmd + '
Step into next function call	F11, Ctrl + ;	F11, Cmd + ;
Step out of current function	Shift + F11, Ctrl + Shift + ;	Shift + F11, Cmd + Shift + ;
Select next call frame	Ctrl + .	Opt + .
Select previous call frame	Ctrl + ,	Opt + ,
Toggle breakpoint condition	Click on line number, Ctrl + B	Click on line number, Cmd + B
Edit breakpoint condition	Right-click on line number	Right-click on line number
Delete individual words	Alt + Delete	Opt + Delete
Comment a line or selected text	Ctrl + /	Cmd + /
Save changes to local modifications	Ctrl + S	Cmd + S
Go to line	Ctrl + G	Ctrl + G
Search by filename	Ctrl + O	Cmd + O
Jump to line number	Ctrl + P + :<number>	Cmd + P + :<number>
Jump to column	Ctrl + O + :<number> + :<number>	Cmd + O + :<number> + :<number>
Go to member	Ctrl + Shift + O	Cmd + Shift + O
Toggle console and evaluate code selected in Sources panel	Ctrl + Shift + E	Cmd + Shift + E
Run snippet	Ctrl + Enter	Cmd + Enter
Toggle comment	Ctrl + /	Cmd + /

⏸ Don't pause on exceptions

⏸ Pause on All exceptions (including those caught within try/catch blocks)

⏸ Pause on uncaught exceptions (usually the one you want)

Exceptions (http://goo.gl/dZSwr)

Timeline Panel

	Windows / Linux	Mac
Start / stop recording	Ctrl + E	Cmd + E
Save timeline data	Ctrl + S	Cmd + S
Load timeline data	Ctrl + O	Cmd + O

Profiles Panel

	Windows / Linux	Mac
Start / stop recording	Ctrl + E	Cmd + E

Profiling types:

- **CPU profiler (http://goo.gl/igHcw)**: shows where execution time is spent in your page's JavaScript functions
- **Heap profiler (http://goo.gl/9mBFW)**: shows memory distribution by your page's JavaScript objects and related DOM nodes

Search Shortcuts

Find or navigate to specific files, methods or line numbers in an web app within the Sources panel

	Windows / Linux	Mac
Search scripts, stylesheets and snippets by filename	Ctrl + O	⌘ + O
Text search within current file	Ctrl + F	⌘ + F
Text search across all sources	Ctrl + Shift + F	⌘ + Alt + F
Filter/navigate to a JavaScript function/CSS rule when viewing a file	Ctrl + Shift + O	⌘ + SHift + O
Launch line number dialog when viewing a file	Ctrl + K	⌘ + L
Evaluate code selected in scripts in the console	Ctrl + Shift + E	⌘ + Shift + E

Console

	Windows / Linux	Mac
Next suggestion	Tab	Tab
Previous suggestion	Shift + Tab	Shift + Tab
Accept suggestion	Right	Right
Previous command / line	Up	Up
Next command / line	Down	Down
Clear Console	Ctrl + L	Cmd + K, Opt + L
Multi-line entry	Shift + Enter	Ctrl + Return
Execute	Enter	Return

Right-clicking on console:

- XMLHTTPRequest logging: Turn on to view the XHR log
- Preserve log upon navigation
- Filter: Hide and unhide messages from script files
- Clear console: Clear all console messages

Screencasting

	Windows / Linux	Mac
Pinch zoom in and out	Alt + Scroll,Ctrl + Cick and drag with two fingers	Opt + Scroll, Cmd + Cick and drag with two fingers
Inspect element tool	Ctrl + Shift + C	Cmd + Shift + C

Emulation

	Windows / Linux	Mac
Pinch zoom in and out	Shift + Scroll	Shift + Scroll

Console API

Command	Description
console.assert(expression[, object, ...])	Tests that an expression is true. If not, it will write a message to the console and throw an exception.
console.clear()	Clears the console.
console.constructor()	
console.count([title])	Writes the number of times that the line of code where count was called was executed. The optional argument title will print a message in addition to the number of the count.
console.debug(object[, object, ...])	Writes a message to the console, including a hyperlink to the line where it was called.
console.dir(object)	Prints an interactive listing of all properties of the object. This looks identical to the view that you would see in the DOM tab.
console.dirxml(node)	Prints the XML source tree of an HTML or XML element. This looks identical to the view that you would see in the HTML tab. You can click on any node to inspect it in the HTML tab.
console.error(object[, object, ...])	Writes a message to the console with the visual "error" icon and color coding and a hyperlink to the line where it was called.
console.exception(error-object[, object, ...])	Prints an error message together with an interactive stack trace of JavaScript execution at the point where the exception occurred.
console.group(object[, object, ...])	Writes a message to the console and opens a nested block to indent all future messages sent to the console. Call console.groupEnd() to close the block.
console.groupCollapsed(object[, object, ...])	Like console.group(), but block is initially collapsed.
console.groupEnd()	Closes the most recently opened block created by a call to console.group() or console.groupCollapsed().
console.hasOwnProperty()	
console.info(object[, object, ...])	Writes a message to the console with the visual "info" icon and color coding and a hyperlink to the line where it was called.
console.isPrototypeOf()	
console.keys()	Gives you the names of all the elements of an object.
console.log(object[, object, ...])	Writes a message to the console. You may pass as many

	arguments as you'd like, and they will be joined together in a space-delimited line. You can use printf-like string substitution patterns as well which are: • String: %s - `console.log('The %s is a %s', animal, cat);` • Integer: %d, %i - `console.log('Number %d plus %i', 1, 2);` **integer not yet supported* • Floating point number: %f - `console.log('Floating points: %f', 1.5);` **numeric formatting not yet supported* • Hyperlink: %o - `console.log('Live laugh code at $o', 'http://anti-code.com' );` • Style formatting: %c - `console.log('%c This is white text on a black background', 'color:#fff;background:#000;');`
console.memory	An object that returns: jsHeapSizeLimit, totalJSHeapSize, usedJSHeapSize
console.profile([title])	Turns on the JavaScript profiler. The optional argument title would contain the text to be printed in the header of the profile report.
console.profileEnd()	Turns off the JavaScript profiler and prints its report.
console.profiles	An array of profile objects that summarizes the data from profile().
console.propertyIsEnumerable()	
console.table(data[, columns])	Allows to log provided data using tabular layout. The method takes one required parameter that represents table like data (array of arrays or list of objects).
console.time([name])	Creates a new timer under the given name. Call console.timeEnd(name) with the same name to stop the timer and print the time elapsed.
console.timeEnd()	Stops a timer created by a call to console.time(name) and writes the time elapsed.
console.timeStamp()	
console.toLocaleString()	
console.toString()	
console.trace()	Prints an interactive stack trace of JavaScript execution at the point where it is called.
console.values()	Gives you all the values of those elements.
console.warn(object[, object, ...])	Writes a message to the console with the visual "warning" icon and color coding and a hyperlink to the line where it was called.
console.valueOf()	
window.onerror	When exceptions are thrown in the window context and is not caught by any try/catch block, the function will be invoked with the exception's message, the URL of the file where the exception was thrown and the line number in that file passed as three arguments in that order. `window.onerror = function(msg, url, line) { console.log('message: ' + msg, 'url: ' + url, 'line: ' + line); };`

Command Line API

Command	Description
$$	Returns an array of elements that match the given CSS selector.
$0	The currently-selected object in the inspector.
$_	The previously evaluated statement
$1	The previously-selected object in the inspector.
$n(index)	Access to an array of last 5 inspected elements.
dir(object)	Prints an interactive listing of all properties of the object. This looks identical to the view that you would see in the DOM tab.
dirxml(node)	Prints the XML source tree of an HTML or XML element. This looks identical to the view that you would see in the HTML tab. You can click on any node to inspect it in the HTML tab.
clear()	Clears the console.
copy()	Copies everything passed to it to the clipboard.
inspect(object[, tabName])	Inspects an object in the most suitable tab, or the tab identified by the optional argument tabName.
keys(object)	Returns an array containing the names of all properties of the object.
values(object)	Returns an array containing the values of all properties of the object.
monitorEvents(object[, types])	Turns on logging for all events dispatched to an object. The optional argument types may specify a specific family of events to log. The most commonly used values for types are "mouse" and "key". The full list of available types includes "composition", "contextmenu", "drag", "focus", "form", "key", "load", "mouse", "mutation", "paint", "scroll", "text", "ui", and "xul".
unmonitorEvents(object[, types])	Turns off logging for all events dispatched to an object.
performance	
performance.timing	
performance.memory	
performance.navigation	
profile([title])	Turns on the JavaScript profiler. The optional argument title would contain the text to be printed in the header of the profile report.
profileEnd()	Turns off the JavaScript profiler and prints its report.

Flags

Feature	Description
-disable-javascript	Disable JavaScript from command line.
-disable-images	Disable images.
-disable-java	Disable Java.
-disable-plugins	Disable plugins.
-disable-popup-blocking	Disable popup blocking.
-start-maximized	Start Chrome fullscreen

For example: *"C:\Documents and Settings\%username%\Local Settings\Application Data\Google\Chrome" -disable-javascript*

DevTools Themes

- MNML Theme (http://goo.gl/NNzwh)
- Monokai Dark (http://goo.gl/KgNfU)
- Tomorrow Theme (http://goo.gl/tcELr)
- IR_Black Theme (http://goo.gl/Ou6WA)
- IR_Black Theme with sidebar colors (http://goo.gl/5ddPs)
- Solarized Dark (http://goo.gl/W0INT)
- Ruby Blue (http://goo.gl/MxIjz)
- Expresso (http://goo.gl/zK2Sn)
- Inversion (http://goo.gl/T1SeZ)
- Dark Theme (http://goo.gl/viN4b)
- Dark Dev (http://goo.gl/BVIDe)
- WebLight Theme (http://goo.gl/E5NJh)

Tweak your skin for the DevTools using the DevTools themselves by undocking them then hitting `Ctrl + Alt + I` or `^ + Alt + I` on Mac.
Override the classes/IDs of Devtools for your theme via:

- Windows: C:/Users/%username%/AppData/Local/Google/Chrome/User Data/Default/User StyleSheets/Custom.css
- Mac OSX: ~/Library/Application Support/Google/Chrome/Default/User StyleSheets/Custom.css
- Ubuntu: ~/.config/chromium/Default/User StyleSheets/Custom.css

Read about how to customize your Devtools (http://goo.gl/3hOAW).

Other Chrome Shortcuts

Here are some additional Chrome shortcuts which are useful for general use within the browser not specific to the DevTools. View all Chrome shortcuts (http://goo.gl/PsTNm) for Windows, Mac, and Linux.

	Windows / Linux	Mac
Find next	Ctrl + G	Cmd + G
Find previous	Ctrl + Shift + G	Cmd + Shift + G
Open a new window in Incognito mode	Ctrl + Shift + N	Cmd + Shift + N
Toggle Bookmarks bar on and off	Ctrl + Shift + B	Cmd + Shift + B
View the History page	Ctrl + H	Cmd + Y
View the Downloads page	Ctrl + J	Cmd + Shift + J
View the Task Manager	Shift + ESC	Shift + ESC
Next page in a tabs browsing history	Alt + Right	Alt + Right
Previous page in a tabs browsing history	Backspace, Alt + Left	Backspace, Alt + Left
Highlight content in the web address area	F6, Ctrl + L, Alt + D	Cmd + L, Alt + D
Places a ? in the address bar for performing a keyword search using your default search engine	Ctrl + K, Ctrl + E	Cmd + K, Cmd + E

For a list of all Chrome shortcuts for Windows, Mac, and Linux check out http://goo.gl/PsTNm (http://goo.gl/PsTNm)

about:pages

Page	Description
about:about	Displays all the chrome://chrome-urls
about:stats	Display page statistics.
about:memory	Display memory usage in a multi-process browser.
about:plugins	Display installed plug-ins.
about:histograms	Display connection times.
about:dns	Display DNS information.
about:cache	Display cached web pages.
about:network	Display a menu for various network monitoring and testing.
view-cache:stats	Display cached documents.
chrome-resource:/favicon/	Display the binary data for a PNG file.
chrome-resource:/new-tab/	A template for the empty tab page.
about:version	Display information about the browser.

Chrome URLs

- chrome://chrome-urls == about:about
- chrome://appcache-internals
- chrome://blob-internals
- chrome://bookmarks
- chrome://cache
- chrome://crashes
- chrome://credits
- chrome://dns
- chrome://downloads
- chrome://extensions
- chrome://flags
- chrome://flash
- chrome://gpu-internals
- chrome://histograms
- chrome://history
- chrome://ipc
- chrome://media-internals
- chrome://memory
- chrome://net-internals
- chrome://view-http-cache
- chrome://newtab
- chrome://plugins
- chrome://print
- chrome://quota-internals
- chrome://sessions
- chrome://starts
- chrome://sync-internals
- chrome://tcmalloc
- chrome://terms
- chrome://tracing
- chrome://version
- chrome://workers
- chrome://conflicts

For Debugging

The following pages are for debugging purposes only. Because they crash or hang the renderer, they're not linked directly; you can type them into the address bar if you need them

- chrome://crash
- chrome://kill
- chrome://hang
- chrome://shorthang
- chrome://gpuclean
- chrome://gpucrash
- chrome://gpuhang

Devtools Links

- Getting started (http://goo.gl/D0QyI)
- Inspecting Elements and Resources (http://goo.gl/giFgt)
- Debugging JavaScript (http://goo.gl/ZIruf)
- Profiling and Optimizing (http://goo.gl/TIG4q)
- Timeline Panel (http://goo.gl/Fwipo)
- Console API (http://goo.gl/XYgok)
- Command Line API (http://goo.gl/F4JPq)
- JavaScript Console Power User (http://goo.gl/rdlJQ)
- Chrome Devtools (http://goo.gl/BVCnH)
- Chrome Shortcuts (http://goo.gl/gkN0K)
- The Breakpoint #1 (http://goo.gl/qtrvm)
- The Breakpoint #2 (http://goo.gl/w1WyO)
- The Breakpoint #3 - The Sourcemap Spectacular (http://goo.gl/Y10Kw)
- The Breakpoint #4 - The Tour De Timeline (http://goo.gl/qzSf4)
- The Breakpoint #5 - DevTools Grab bag (http://goo.gl/4kDXf)
- Google IO 2010 session (http://goo.gl/q64sW)
- Google IO 2011 session (http://goo.gl/00St7)
- Google IO 2012: Developer Tools Evolution (http://goo.gl/Lp4UJ)
- Secrets of the Chrome Developer Tools (http://goo.gl/YMnNU)
- Wait, DevTools could do THAT? (http://goo.gl/QfVfa)
- Chrome Dev Tools: Networking and the Console (http://goo.gl/FUzll)
- Extending Chrome DevTools for fun and profit (http://goo.gl/o23Cp)
- Better Layout for Devtools Docked Vertically (http://goo.gl/PjIsn)
- The WebKit Inspector (http://goo.gl/bZL3A)
- Visually Re-engineering CSS For Faster Paint Times (http://goo.gl/IQEXd)