

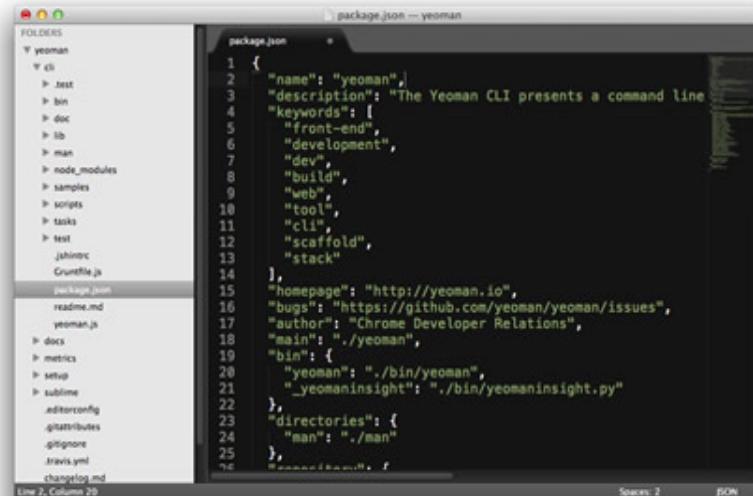


TOOLING FOR THE MODERN WEBAPP DEVELOPER

@addyosmani



Browser + DevTools



Text Editor

```
Last login: Tue Nov 13 14:59:59 on ttys001
addyo at dhcp-172-16-23-68 in ~
$ yeoman
Starting update check...
yeoman v0.9.5
Usage: yeoman [command] [task [task ...]]

Available commands supported by yeoman:

  init  Initialize and scaffold a new project using generator templates
  build Build an optimized version of your app, ready to deploy
  server Launch a preview server which will begin watching for changes
  test   Run a Mocha test harness in a headless PhantomJS
  install Install a package from the clientside package registry
```

Terminal



Devices



In-browser DevTools

- Constantly evolving
- Use [Canary](#) channel for development
- Lots of juicy [experimental](#) features

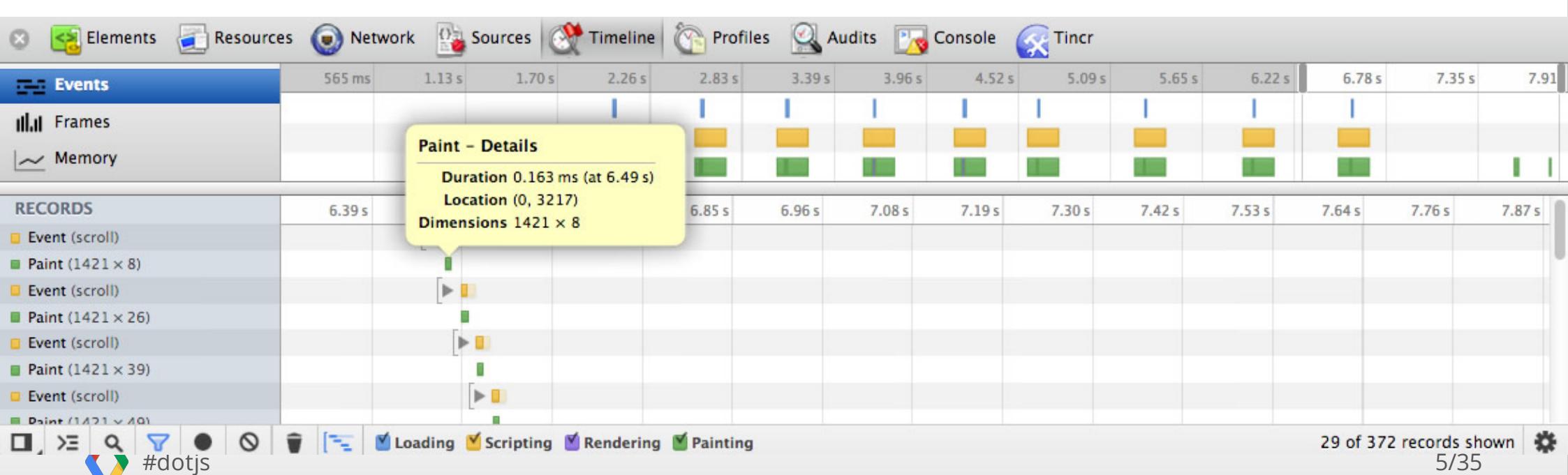


The screenshot shows the Google Chrome DevTools interface with the "Sources" tab selected. The left sidebar lists the project structure, including files from developers.google.com, _static/css, _static/js, chrome, and jsi18n. The main pane displays the code for "script_foot.js". The code is annotated with numbered callouts (1 through 17) pointing to specific lines. Lines 4, 10, and 16 have blue boxes around them. Lines 10 and 16 also have small blue arrows pointing to the right. The right sidebar contains sections for "Watch Expressions", "Call Stack" (which is "Not Paused"), "Scope Variables" (which is "Not Paused"), and "Breakpoints". Three breakpoints are set, indicated by checked checkboxes: one at line 4, one at line 10, and one at line 16. The status bar at the bottom shows "Local modifications".

```
1 var devsite = window.devsite || {};
2 window.devsite = devsite;
3 var _gaq = _gaq || [];
4 WebFontConfig = {google: {families: ["Open+Sans"]}};
5 var gapi = gapi || {};
6 devsite.location = window.location;
7 devsite.reloadWindow = function() {
8     devsite.location.reload()
9 };
10 devsite.devsite = devsite.devsite || {};
11 devsite.devsite.dialogConfig = {autoOpen: !1,modal: !0,show: "fade",di
12 devsite.tag = devsite.tag || {};
13 devsite.tag.admin = devsite.tag.admin || {};
14 devsite.tag.admin.addTagItem = function(a) {
15     var b = "add/" + encodeURIComponent($.trim(a.target.name.value));
16     $.post(b, $(a.target).serialize(), function(a) {
17         a.error ? alert("Error: " + a.error) : null
18     })
19 }
20 })(window);
21 
```

Performance: Timeline + Frames view

- Timeline gives you an overview of memory usage over time
- Summary and detailed views
- Helps remove jank. Layout or scripts - who triggered what?
- Frames view helps achieve that snappy 60fps you ideally want



Finding memory leaks and DOM leaks

- JavaScript, CSS, Heap snapshot Profiles
- What is using memory at a given point in time? Not being GC'd?
- Use [comparison](#) view to identify potential memory leaks
- Use [summary](#) view to identify DOM leaks

The screenshot shows the Chrome DevTools interface with the "Profiles" tab selected. On the left, there's a sidebar titled "HEAP SNAPSHOTS" containing three entries: "Snapshot 1" (1.5 MB), "Snapshot 2" (1.5 MB), and "Snapshot 3" (1.5 MB). The main content area is titled "Select profiling type" and contains three options:

- Collect JavaScript CPU Profile
CPU profiles show where the execution time is spent in your page's JavaScript functions.
- Collect CSS Selector Profile
CSS selector profiles show how long the selector matching has taken in total and how many times a certain selector has matched DOM elements (the results are approximate due to matching algorithm optimizations.)
- Take Heap Snapshot

At the bottom of the screen, there are standard DevTools navigation icons and a status bar indicating "6/35".

A better authoring workflow

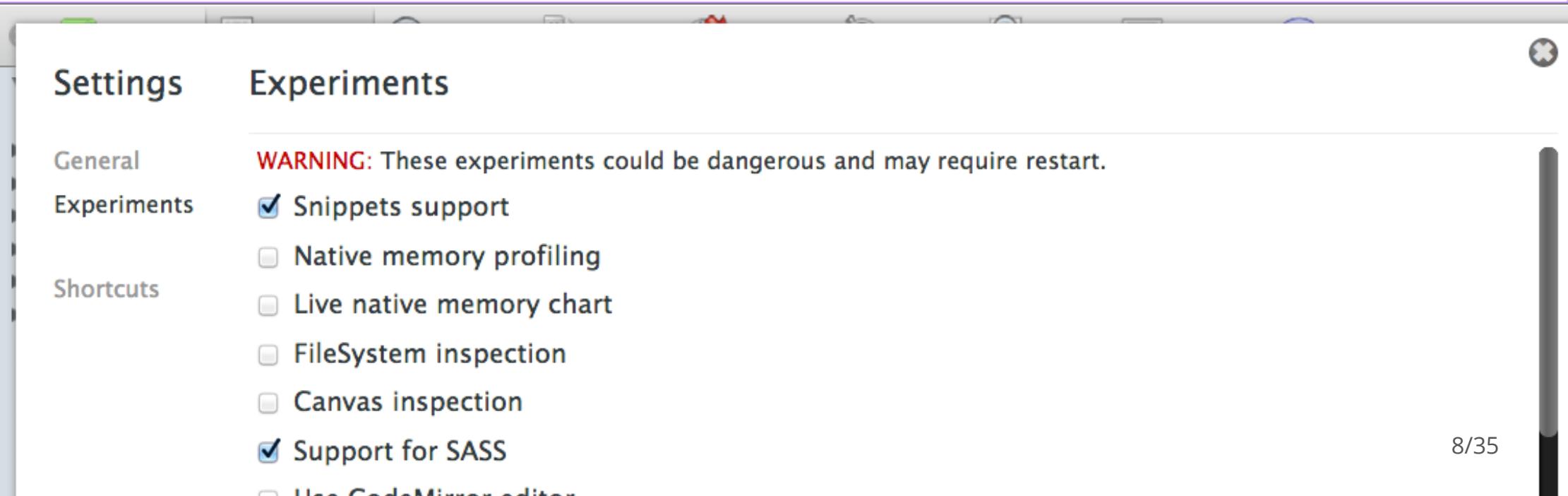
- Live Edit
- Snippets
- Revision history
- AutoSave

The screenshot shows the Chrome DevTools interface with the 'Sources' tab selected. The left sidebar lists various resources, including 'script_foot.js' which is currently open. The main pane displays the code for 'script_foot.js'. The right sidebar contains sections for 'Watch Expressions', 'Call Stack' (which is 'Not Paused'), 'Scope Variables' (which is 'Not Paused'), 'Breakpoints' ('No Breakpoints'), 'DOM Breakpoints', 'XHR Breakpoints', 'Event Listener Breakpoints', and 'Workers' (with page number '7/35').

```
1 var devsite = window.devsite || {};
2 window.devsite = devsite;
3 var _gaq = _gaq || [];
4 WebFontConfig = {google: {families: ["Open+Sans"]}};
5 var gapi = gapi || {};
6 devsite.location = window.location;
7 devsite.reloadWindow = function() {
8     devsite.location.reload()
9 };
10 devsite.devsite = devsite.devsite || {};
11 devsite.devsite.dialogConfig = {autoOpen: !1,modal: !0,show: "fade",dialogClass: "devsite-dialog",width: 400,height: 300};
12 devsite.tag = devsite.tag || {};
13 devsite.tag.admin = devsite.tag.admin || {};
14 devsite.tag.admin.addTagItem = function(a) {
15     var b = "add://" + encodeURIComponent($.trim(a.target.name.value)).toLowerCase();
16     $.post(b, $(a.target).serialize(), function(a) {
17         $("#result").html(a.success ? "Success." : a.error)
18     })
19 }
```

Live reloading + SASS Source Maps

- Editing compiled CSS has little value
- [Enable new SASS hotness](#)
- BOOM! You can edit SASS source files
- Changes automatically reload



CoffeeScript + @sourceURL

- Compile your Coffee sources
- Open up the DevTools
- Review your compiled file
- Whoa! sourceURL comments

The screenshot shows the Google Chrome DevTools interface with the 'Sources' tab selected. The left sidebar displays the file structure, with 'coffee-script.js...' expanded to show its contents. The main pane shows the compiled JavaScript code:

```
9  };
10
11 alert(cube(5));
12
13 }).call(this);
14 //@ sourceURL=test.js
15
16
17
18
19
20
21
22
```

The line `//@ sourceURL=test.js` is highlighted with a blue selection bar. The right sidebar contains the Breakpoints panel, which is currently empty, showing the message `No Breakpoints`.

Mobile Debugging: Overrides Panel

- User Agent
- Device Metrics
- Geolocation
- Orientation
- Emulate touch events

The screenshot shows the Chrome DevTools interface with the "Overrides" panel open. The top navigation bar includes tabs for Elements, Resources, Network, Sources, Timeline, Profiles, Audits, Console, and Tincr. Below the navigation bar, the "Overrides" tab is selected, showing a list of available overrides: Frames, (index.html), and Web SQL. The main content area is titled "Overrides" and contains three sections: "User Agent" (selected), "Override Geolocation", and "Device metrics".

User Agent
Internet Explorer 9

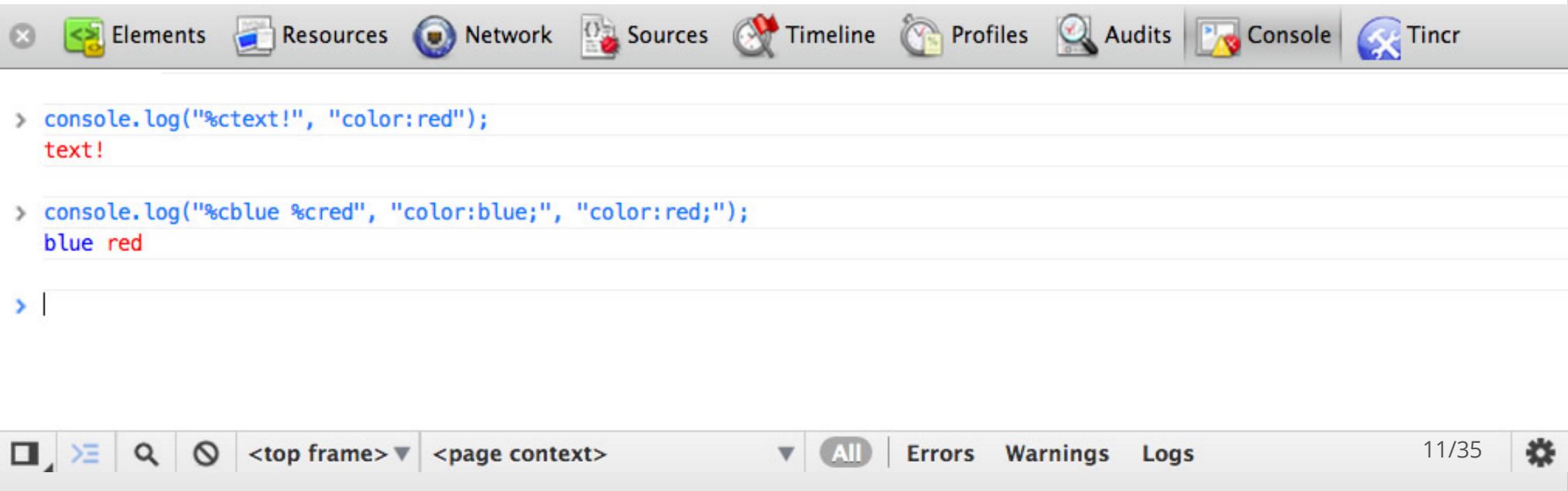
Override Geolocation
Geolocation Position: Lat = 0, L
 Emulate position unavailable

Device metrics

Override Device Orientation

DevTools Console

- Styled-console
- Multi-style support
- inspect() command



The screenshot shows the Chrome DevTools interface with the following details:

- Toolbar:** Elements, Resources, Network, Sources, Timeline, Profiles, Audits, **Console**, Tincr.
- Console Output:**

```
> console.log("%ctext!", "color:red");
text!
> console.log("%cblue %cred", "color:blue;", "color:red;");
blue red
> |
```
- Bottom Navigation:** □, ⌂, 🔎, ⚙, <top frame> ▾, <page context>, ▾, All, Errors, Warnings, Logs, 11/35, ⚙



The dark side of CS6.psd @ 66.7% (smoke, RGB/8) * ×

Learn to love the command line.
It isn't scary.

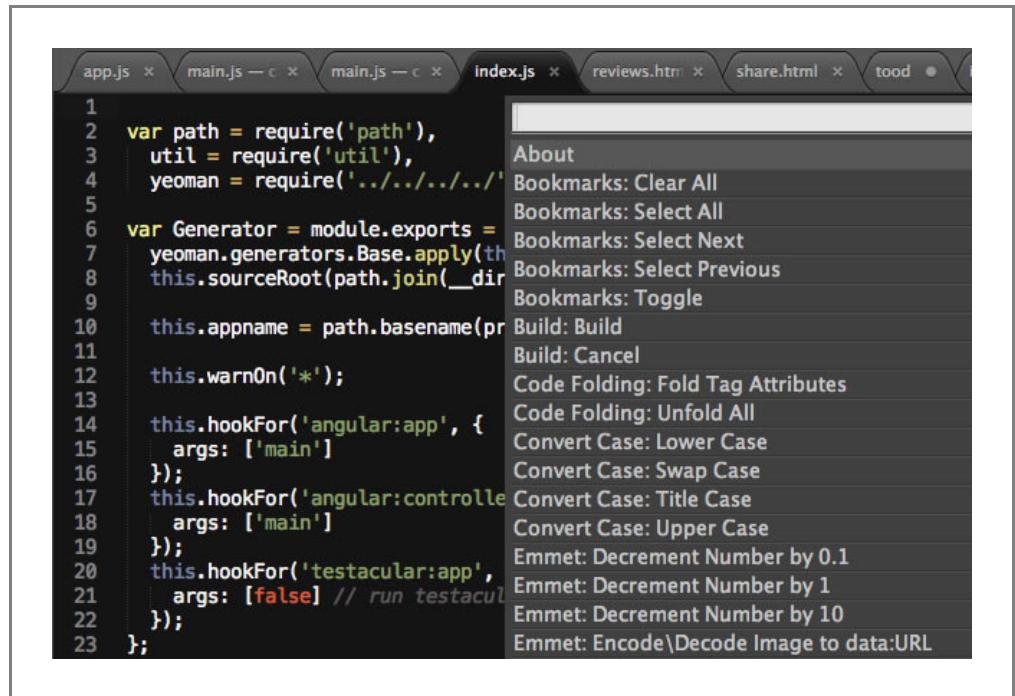
~ Stephen Hay

Command-line

- Make it look hot ❤.
- Capture and replay your command line *history*
- DOTFILE EVERYTHING (mine)
- Aliases are awesome
- Our faves: `gz` , `server` alias and ny
- Some for Browserstack: `win7ie8`, `win8ie10`, `ios3`

Sublime Text #protips

- Run JavaScript From Your Editor In The [Browser](#)
- Use the built-in [Build System!](#)
- Zen coding with [Emmet](#)
- Stack Overflow [code search](#)
- [Snippets](#) for frameworks



The screenshot shows a Sublime Text window with multiple tabs at the top: 'app.js', 'main.js - e', 'main.js - e', 'Index.js', 'reviews.htm', 'share.html', and 'tood'. A context menu is open over the 'Index.js' tab, listing various commands. The visible items include:

- About
- Bookmarks: Clear All
- Bookmarks: Select All
- Bookmarks: Select Next
- Bookmarks: Select Previous
- Bookmarks: Toggle
- Build: Build
- Build: Cancel
- Code Folding: Fold Tag Attributes
- Code Folding: Unfold All
- Convert Case: Lower Case
- Convert Case: Swap Case
- Convert Case: Title Case
- Convert Case: Upper Case
- Emmet: Decrement Number by 0.1
- Emmet: Decrement Number by 1
- Emmet: Decrement Number by 10
- Emmet: Encode\Decode Image to data:URL

The code in 'Index.js' is partially visible:

```
1 var path = require('path'),  
2     util = require('util'),  
3     yeoman = require('.../.../.../  
4  
5 var Generator = module.exports =  
6     yeoman.generators.Base.apply(th  
7         this.sourceRoot(path.join(_dir  
8  
9     this.appname = path.basename(pr  
10    this.warnOn('*');  
11  
12    this.hookFor('angular:app', {  
13        args: ['main']  
14    });  
15    this.hookFor('angular:controlle  
16        args: ['main']  
17    });  
18    this.hookFor('testacular:app',  
19        args: [false] // run testacul  
20    );  
21    this.hookFor('testacular:app',  
22        args: [false] // run testacul  
23});
```

Finding missing semicolons should never be a manual process.

write less, do more.

THANKS OUR GENEROUS SPONSORS



wijm
jobs



Linting

- On file save
- On source control commit
- At build time
- *Anything better?*

```
→ grunt
Running "lint:files" (lint) task
Lint free.

Running "qunit:files" (qunit) task
Testing jquery.demo.html....OK
>> 4 assertions passed (35ms)

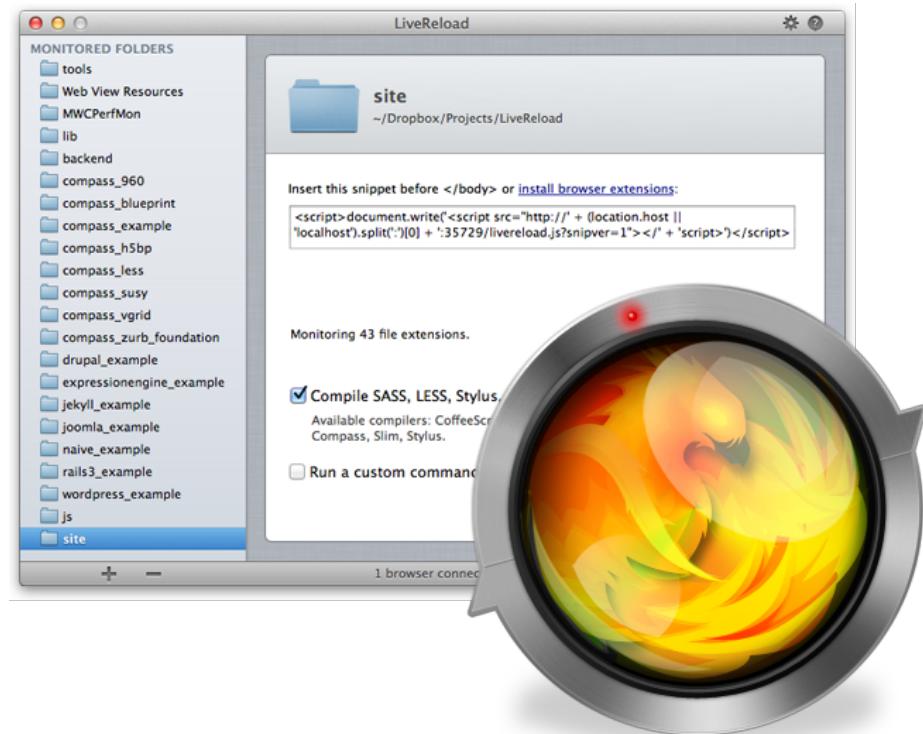
Running "concat:dist" (concat) task
File "dist/jquery.demo.js" created.

Running "min:dist" (min) task
File "dist/jquery.demo.min.js" created.
Uncompressed size: 455 bytes.
Compressed size: 225 bytes gzipped (312 bytes minified).

Done, without errors.
```

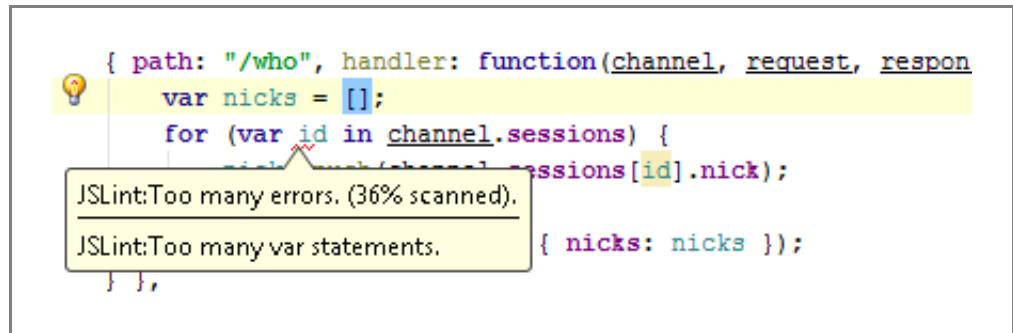
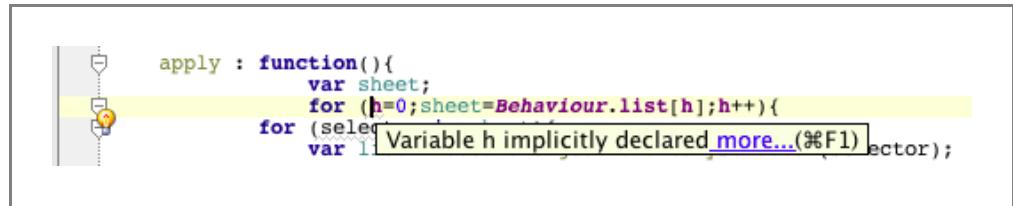
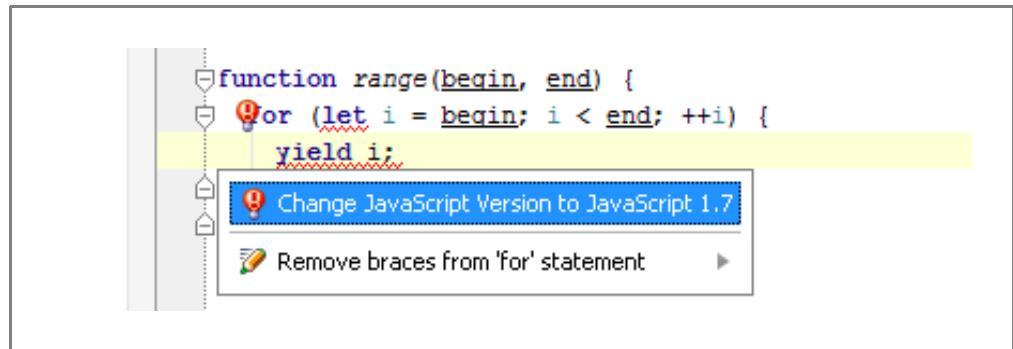
Live feedback

- [Linting](#)
- [Reload](#)
- Recompilation



WebStorm

- Live Edit + Chrome
- JS Language version based suggestions
- Code inspection and zen coding
- Suggestions for DRYer code
- Built in code linting
- and more



Unit Testing In The Cloud

Testing approaches you already know:

- In [the browser](#)
- In a headless browser on-demand via cmd line: `grunt qunit`
- In a headless browser [post-push](#)

New hotness:

- In multiple browsers *in the cloud* via cmd line:

```
bunyip -f Modernizr/test/index.html -c ~/bunyip/config.js -b ios
```

```
bunyip -f index.html local -l "firefox|chrome|safari|phantomjs"
```

Build system



Lint. Resolve dependencies. concatenate modules. compile. Flatten your CSS @imports. Remove debugging statements. Compress images. Precompile templates. Run tests in a variety of environments. Revs asset paths for caching. Affirm code quality.

New hotness: Grunt.js



Getting started

Be sure to read the getting started guide, which is a complete guide to configuring grunt for your project. In addition, check out the [example gruntfiles](#) which highlight a number of fairly common configurations.



Install using [npm install -g grunt](#)

GRUNT

Grunt is a task-
JavaScript projec

[View on GitHub](#)[Docu](#)

Latest plug

[contrib-ember](#) Manage and

[contrib-manifest](#) Precompile

[pistachio-compiler](#) Grunt task t

tailed into G

Grunt.js:

- Task based command line build tool
- Alternative to Rake/Cake/Make/Jake
- Rich community of build tasks
- Generates simple skeleton for new projects
- Lint, test, concat, watch and min out of the box.
- *however..you're still responsible for workflow*

```
answering "?" to any question will show question-specific help and answering "none" to most questions will leave its value blank.

Please answer the following:
[?] Project name (1) toolsRawk
[?] Description (The best project ever.)
[?] Version (0.1.0)
[?] Project git repository (git://github.com/addyo/toolsRawk.git)
[?] Project homepage (https://github.com/addyo/toolsRawk)

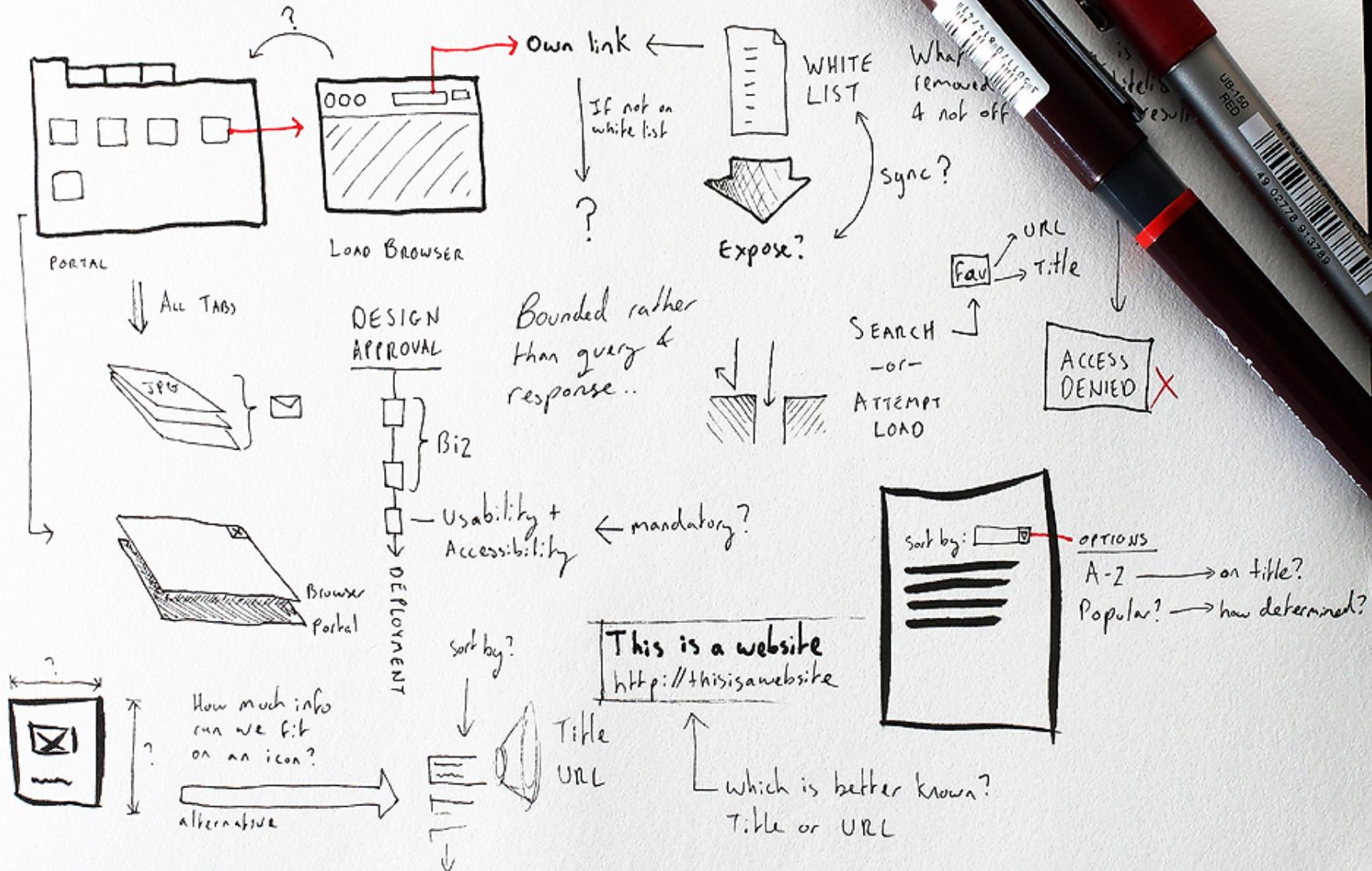
[?] Project issues tracker (https://github.com/addyo/toolsRawk/issues)
[?] Licenses (MIT)
```

*so much choice! you want flexibility.
how could we make this any easier?*

Introducing Yeoman



Go from idea to a rough prototype in 10 min



Limit the time spent on writing boilerplate for your app

```
function() {
    let elements = document.querySelectorAll('input');
    let failedLangs = [];
    for (let i = 0; i < elements.length; i++) {
        if (!elements[i].value) {
            failedLangs.push(elements[i].lang);
        }
    }
    return failedLangs;
}
```

Automate as much of your workflow as possible



Yeoman

- Authoring abstractions
- Scaffolds
- Linting
- LiveReload
- Testing
- Build tools
- and more.



Built on top of great tools like Grunt and Bower



TWITTER ENGINEERING PRESENTS

BOWE

THE BROWSER PACKAGE MANAGER
HTML, CSS, AND JAVASCRIPT

GITHUB



Walkthrough

1. Package management
2. Generators
3. Live Reload
4. Testing
5. Build system



What's next?



Yeoman in 2013

1. Improved flexibility with tools (Grunt, Bower)
2. Better editor integration
3. Support for backends (Rails, PHP)
4. Better mobile helpers and remote debugging
5. Support for all of JS.next (Traceur)
6. Deployment (Heroku, AppEngine)
7. ..plans for a little [more](#)

Yeoman 0.9.5 Just Launched!

- Visit yeoman.io

The image shows the official Yeoman website on the left and a YouTube video player on the right.

Website Screenshot:

- Header:** YEOMAN
- Logo:** A cartoon character wearing a top hat and a mustache.
- Navigation:**
 - Home
 - Why Yeoman?
 - Getting Started
 - Documentation
 - Command Line
 - Package Manager
 - Installation
 - Insight
 - FAQs
 - Discuss
- Social Links:** Google Plus, Twitter, GitHub, Contributing.

YouTube Video Player:

- Title:** Introduction to Yeoman
- Description:** Yeoman is a robust and opinionated set of tools, libraries, and a workflow that can help developers quickly build beautiful, compelling web apps.
- Content:** A video showing a man with glasses and a dark shirt speaking. The video player interface includes a play button, volume control, and a progress bar showing 0:00 / 5:45.
- Controls:** Share, More info, YouTube logo.

Learn to love your workflow and tools



g+
twitter
www
github

addyosmani.com/+
@addyosmani
addyosmani.com
github.com/addyosmani



Thank you!