heig-vd

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD

www.heig-vd.ch

# 03 - Towards deployment

## Getting familiar with cloud solutions

**TWEB 2017**

**Olivier Liechti**

https://softeng-heigvd.github.io/Teaching-HEIGVD-TWEB-2017-Main/

https://t.me/joinchat/CPPWmAsLLgWdXQhoXTaNHw

https://t.me/joinchat/AAAAAEE3IWzr-jZRRMq3qg

# Weekly menu

How are we going to spend the next 6 periods?

# Goals (1)

- **Introduce concepts**
  - Javascript dialects and transpiling
  - pipelines: task manager & asset bundlers
  - *We will put them in practice a bit later*

# Goals (2)

- Prepare the deployment of our apps
  - Present a way to deploy web assets
  - Present a way to deploy the back-end API and the crawler
  - Get familiar with these environments (tutorials and experiments)

# Goals (3)

- Continue to work on the logic of your project:
    - Crawler: everything ok with async?
    - Front-end: everything ok presentation?
    - Back-end: be aware of CORS

# Problem

How do I make sure that my “dialect” of Javascript will work on all browsers?

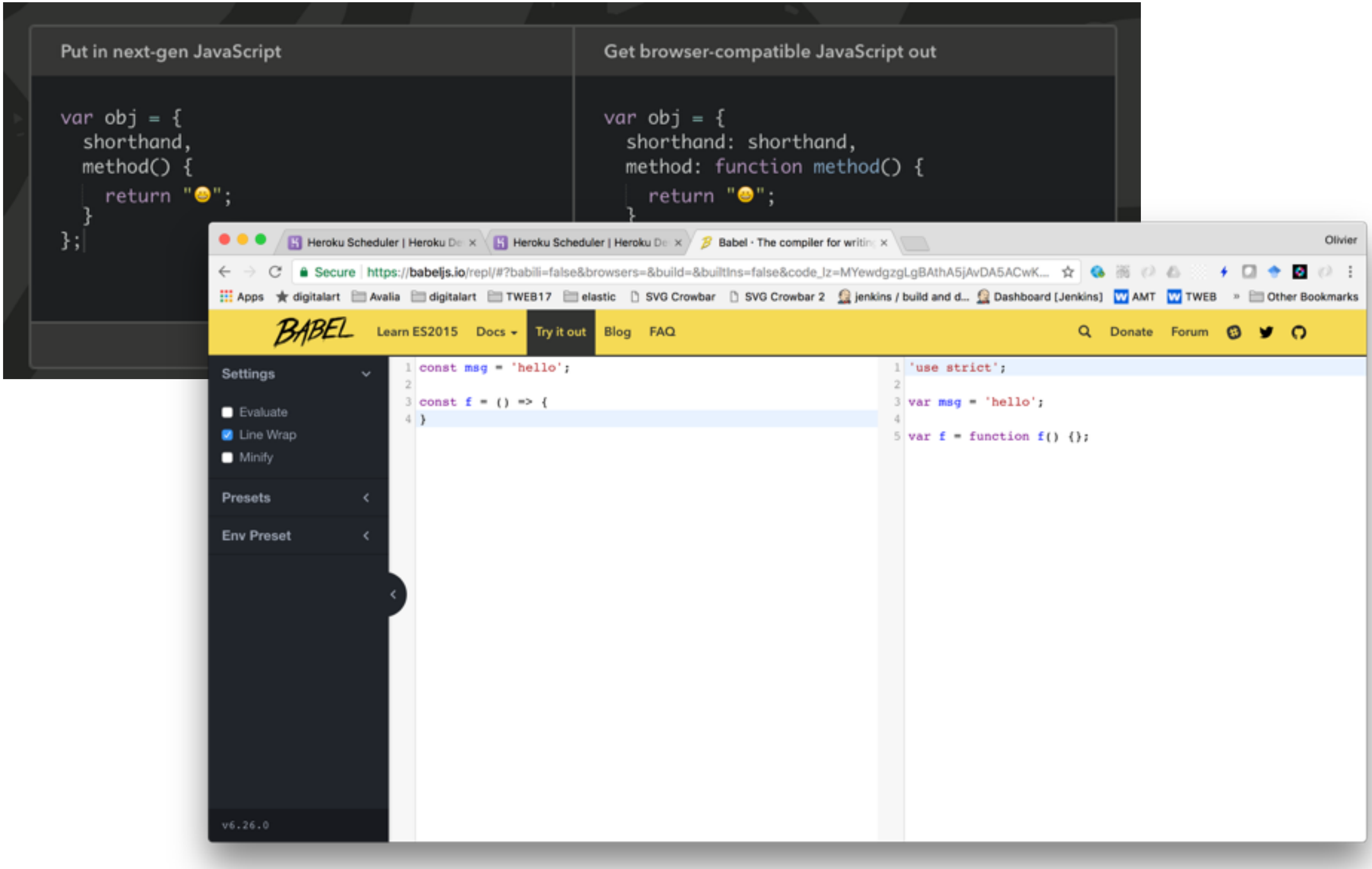
# Forces

- We have seen that there are different generations of Javascript (es5, es6, etc.).
- You may also have heard about other languages “related” to Javascript: Typescript, CoffeeScript
- As a developer, you may want to use the last features of the language.
- On the server side, you have control on the runtime environment. On the client side, you don’t.

# Solution

- There are tools that transform (transpile) different dialects of Javascript in code that is understood by older engines.
- Today, the most popular is **Babel.js**.
- It gives you a command line tool, which takes “modern” Javascript files as input and generates “universal” Javascript as output.

# Solution (2)


Put in next-gen JavaScript
Get browser-compatible JavaScript out
Heroku Scheduler | Heroku De
Heroku Scheduler | Heroku De
Babel · The compiler for writin
Olivier
Secure
https://babeljs.io/repl/#?babili=false&browsers=&build=&builtIns=false&code_lz=MYewdgzgLgBAthA5jAvDA5ACwK...
Apps
digitalart
Avalia
digitalart
TWEB17
elastic
SVG Crowbar
SVG Crowbar 2
jenkins / build and d...
Dashboard [Jenkins]
AMT
TWEB
Other Bookmarks
BABEL
Learn ES2015
Docs
Try it out
Blog
FAQ
Donate
Forum
Settings
Presets
Env Preset
v6.26.0

Put in next-gen JavaScript

```
var obj = {
  shorthand,
  method() {
    return "😁";
  }
};
```

Get browser-compatible JavaScript out

```
var obj = {
  shorthand: shorthand,
  method: function method() {
    return "😁";
  }
```

Settings

☐ Evaluate
☑ Line Wrap
☐ Minify

```
const msg = 'hello';

const f = () => {
}
```

```
'use strict';

var msg = 'hello';

var f = function f() {};
```

# Problem

What are asset pipelines, task managers and module bundlers?

# Forces

- The Javascript ecosystem is full of tools that seem to be “similar” things.
- It is confusing, because they have overlapping functionality. There are many ways to combine them...
- You may have heard about grunt, gulp, browserify, webpack and others...
- Even npm, which is a package manager, has some overlap because (scripts...)

# Solution

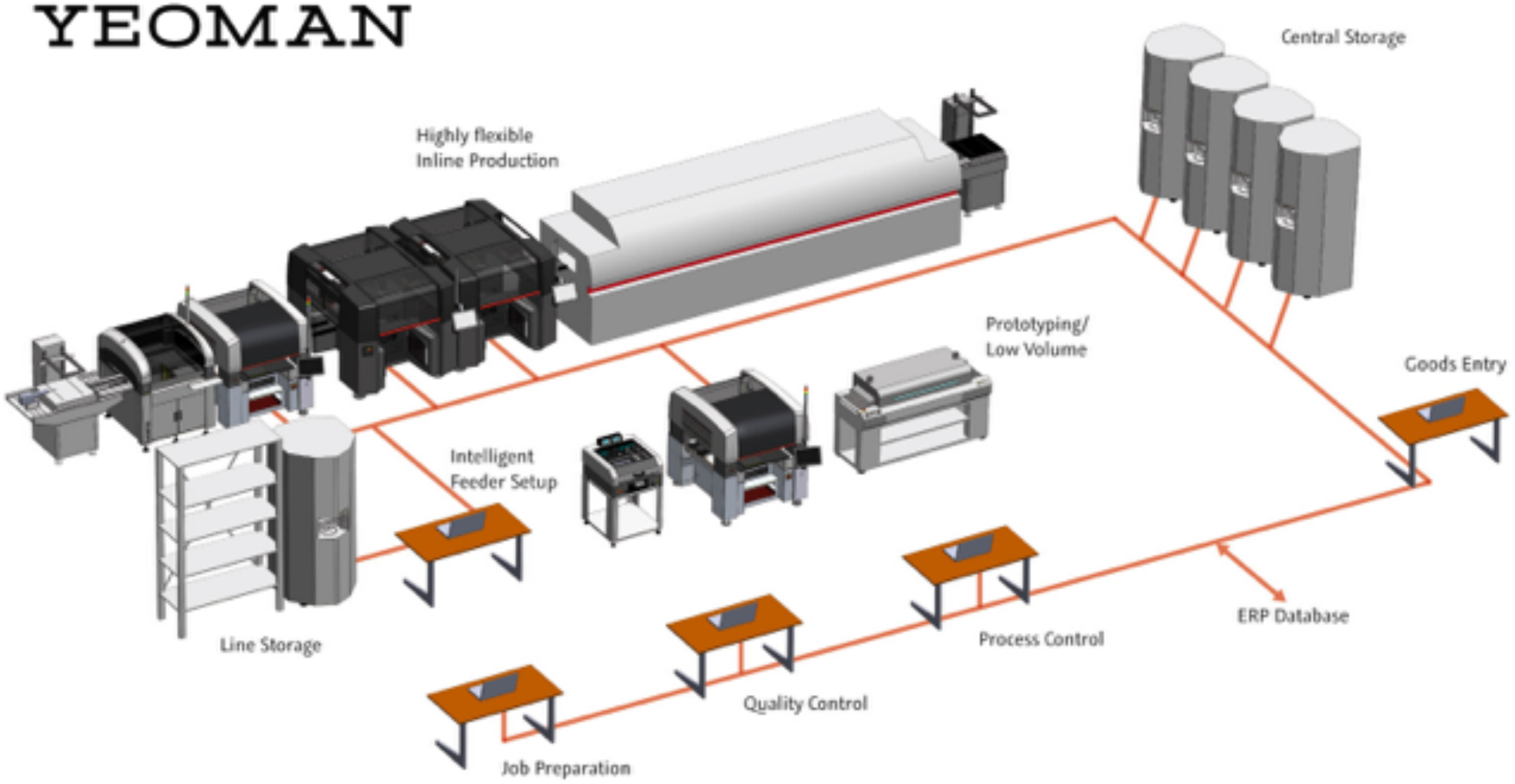
- It is important to know about “first generation” tools that are still used a lot: Grunt and Gulp, which are primarily task managers
- For about 1 year, webpack is becoming the most popular tool. It is primarily a bundle manager, but can also be used as a task manager...
- Today, we start with Grunt and Gulp

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

YEOMAN

GRUNT

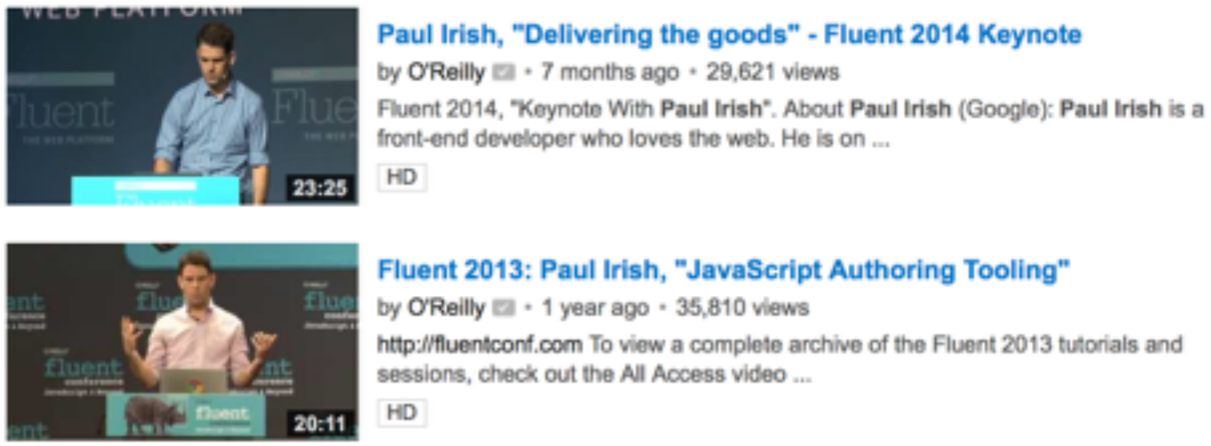
Gulp



# Development tools & pipelines

# How do I **bootstrap** and **structure** my project?

- Based on the specifications, we know that we will develop components **both on the client and on the server side**. We also want to **automate the build process** for our project.
- What should we do? **Start from scratch** or use some kind of **skeleton**? What are our **options**? What are the **professional** front-end developers doing?



Paul Irish, "Delivering the goods" - Fluent 2014 Keynote

by O'Reilly · 7 months ago · 29,621 views

Fluent 2014, "Keynote With Paul Irish". About Paul Irish (Google): Paul Irish is a front-end developer who loves the web. He is on ...

HD

Fluent 2013: Paul Irish, "JavaScript Authoring Tooling"

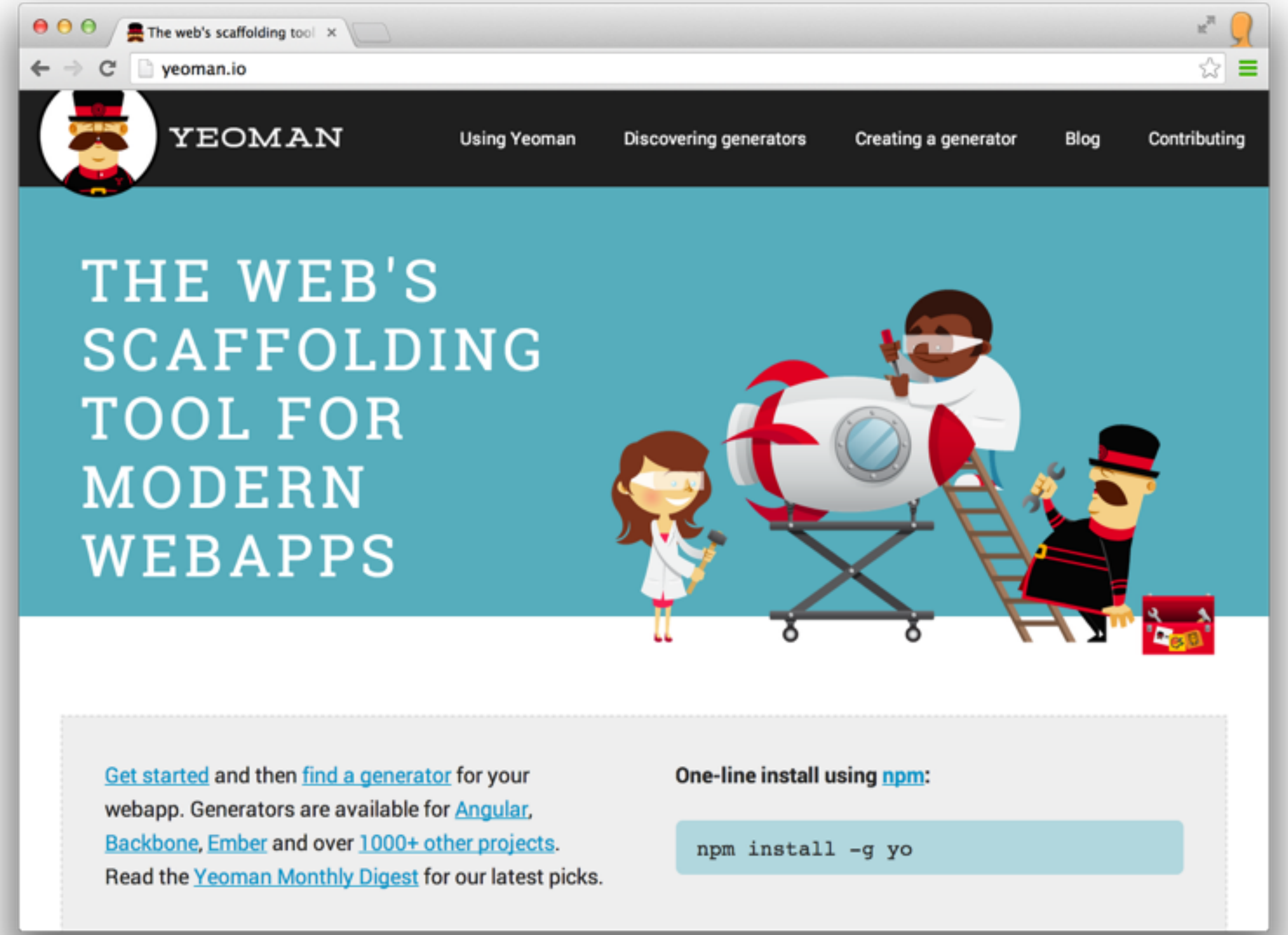
by O'Reilly · 1 year ago · 35,810 views

http://fluentconf.com To view a complete archive of the Fluent 2013 tutorials and sessions, check out the All Access video ...

HD

https://www.youtube.com/watch?v=vDbbz-BdyYc

# Meet **Yeoman**.



THE WEB'S SCAFFOLDING TOOL FOR MODERN WEBAPPS

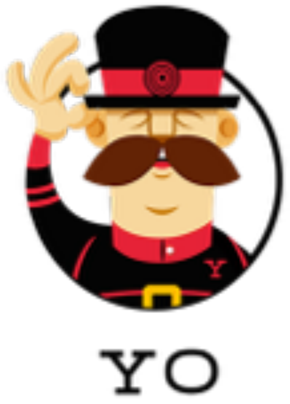
Get started and then find a generator for your webapp. Generators are available for Angular, Backbone, Ember and over 1000+ other projects. Read the Yeoman Monthly Digest for our latest picks.

One-line install using npm:

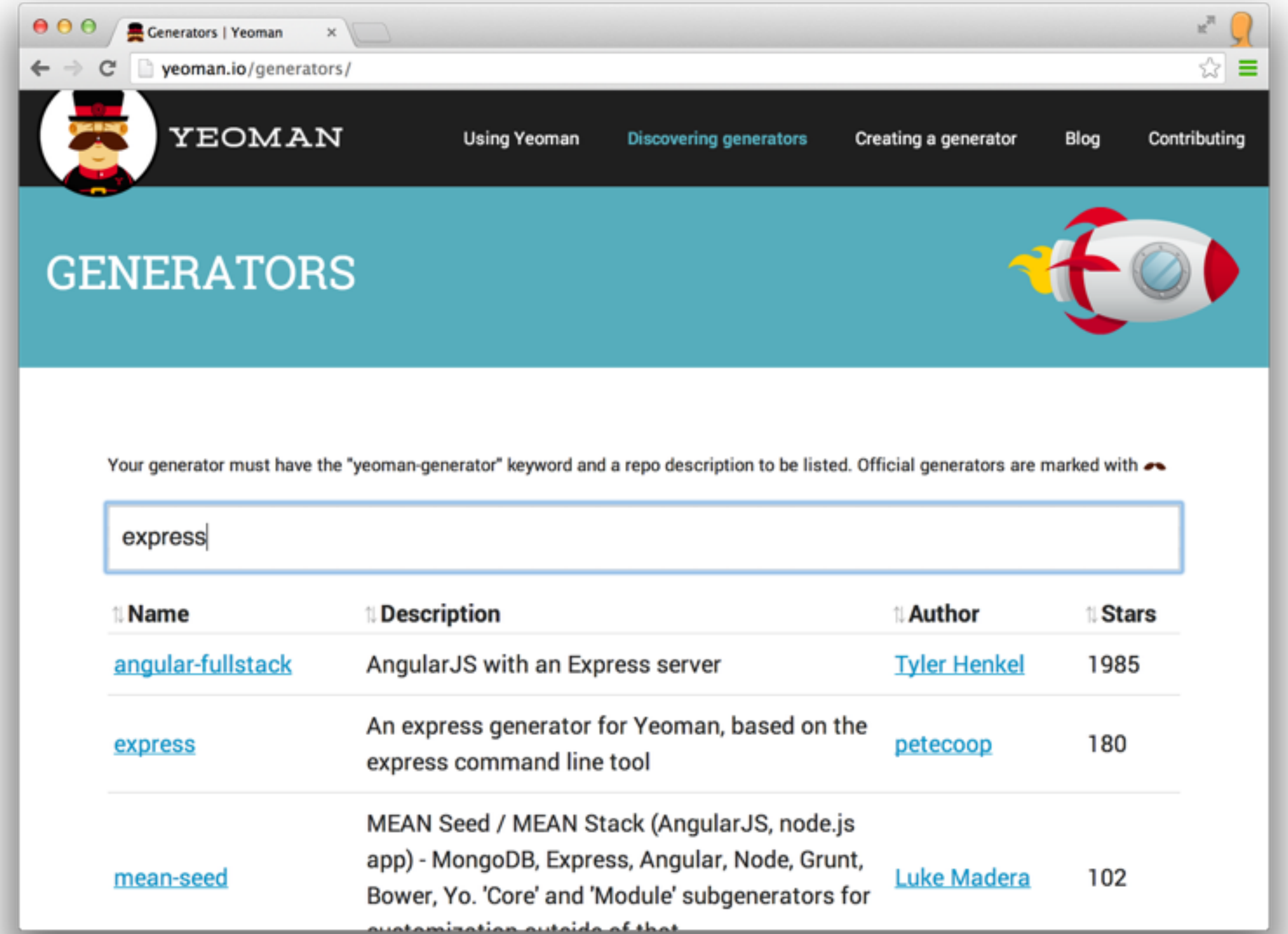
```
npm install -g yo
```

# What is **Yeoman**?

- Yeoman is a **combination of tools**, which allows to you to setup a **complete**, **automated**, **efficient** and **reliable development workflow**.
- **Yo** is a tool for generating project skeletons (**scaffolding**). You can create and share your skeletons. **Yo generators are npm modules** and you can find one for most popular web frameworks.
- **Bower** is a tool for managing "**web dependencies**". Not only javascript modules, but also CSS files, images, etc.
- **Grunt** is a **task runner**. It is the tool that drives your automated process, by executing a series of tasks. There are lots of grunt plugins provided by the community for all aspects of your project.



Ok... generators look cool. But how should I **pick the "right" one**?



## GENERATORS

Your generator must have the "yeoman-generator" keyword and a repo description to be listed. Official generators are marked with

express

| Name | Description | Author | Stars |
|---|---|---|---|
| angular-fullstack | AngularJS with an Express server | Tyler Henkel | 1985 |
| express | An express generator for Yeoman, based on the express command line tool | petecoop | 180 |
| mean-seed | MEAN Seed / MEAN Stack (AngularJS, node.js app) - MongoDB, Express, Angular, Node, Grunt, Bower, Yo. 'Core' and 'Module' subgenerators for customization outside of that | Luke Madera | 102 |

# How do you **pick a generator** for your project?

- You probably **have an idea of the framework(s) you want to use** on the server and or client side (express, angular, backbone, etc.). You will use this as a first filter.

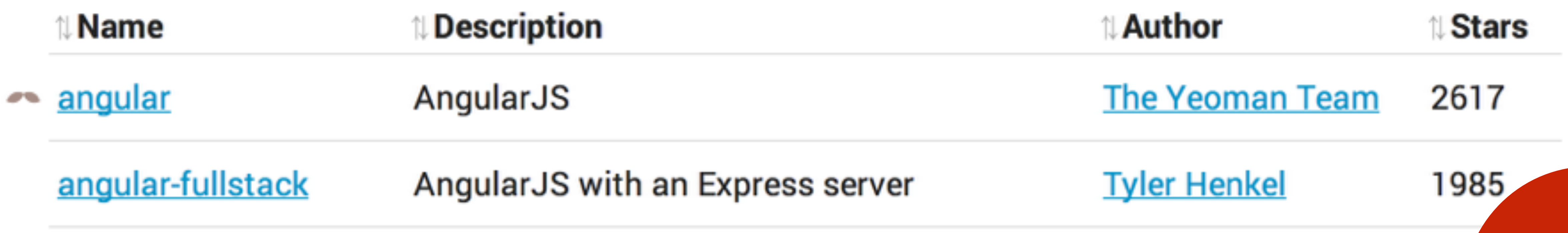
- Some of the generators are **supported by the Yeoman Team**. That is probably a good indication about the quality and support over time (evolution).

- Developers who use generators can **"star"** those they like. **Sorting by popularity** is also an interesting indication. If the community is big, you can expect issues to be reported and fixed, to see new features, etc.

- After you have identified **promising candidates**, you need to get a **first impression**. Generate and build a project with each candidate. Look at their Github repository. Do you like what you see? Do you like the documentation?

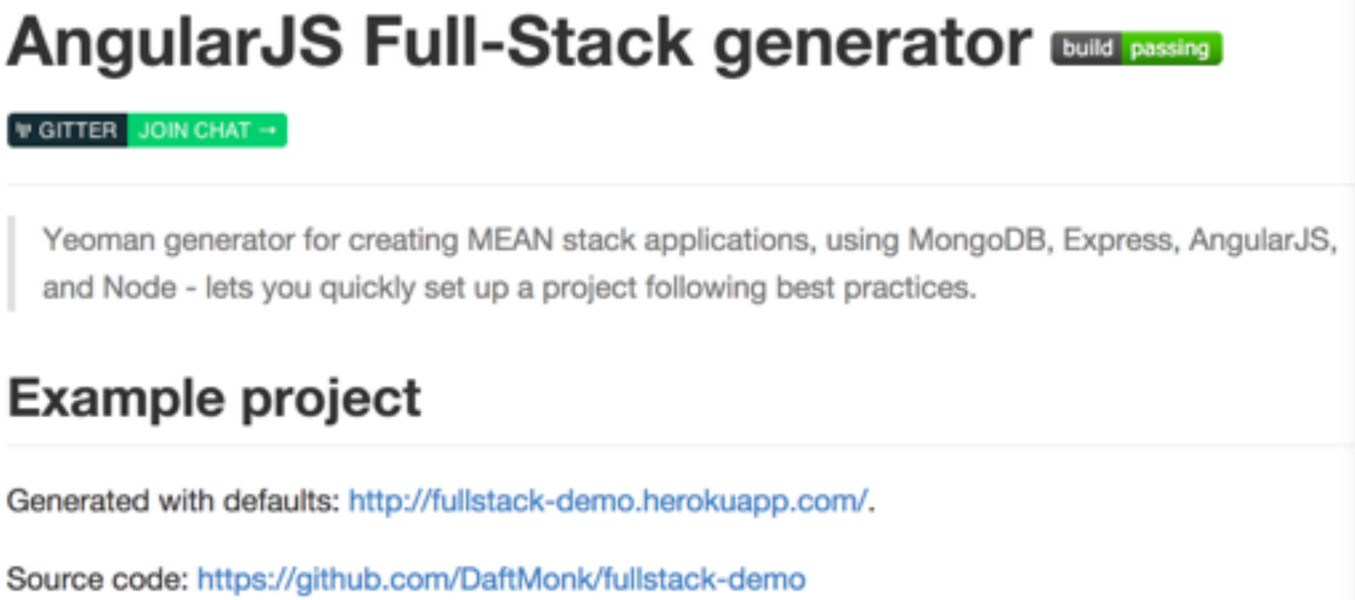
- Often, you will need to choose between "**lightweight**" and very "**rich**" generators. Lightweight generators are easier to learn and give you more control (but more work). Rich generators do a lot of things out-of-the-box but can be intimidating at first (learning curve to understand the skeleton).

Meet the **angular-fullstack** generator.



| Name | Description | Author | Stars |
|---|---|---|---|
| angular | AngularJS | The Yeoman Team | 2617 |
| angular-fullstack | AngularJS with an Express server | Tyler Henkel | 1985 |

2015



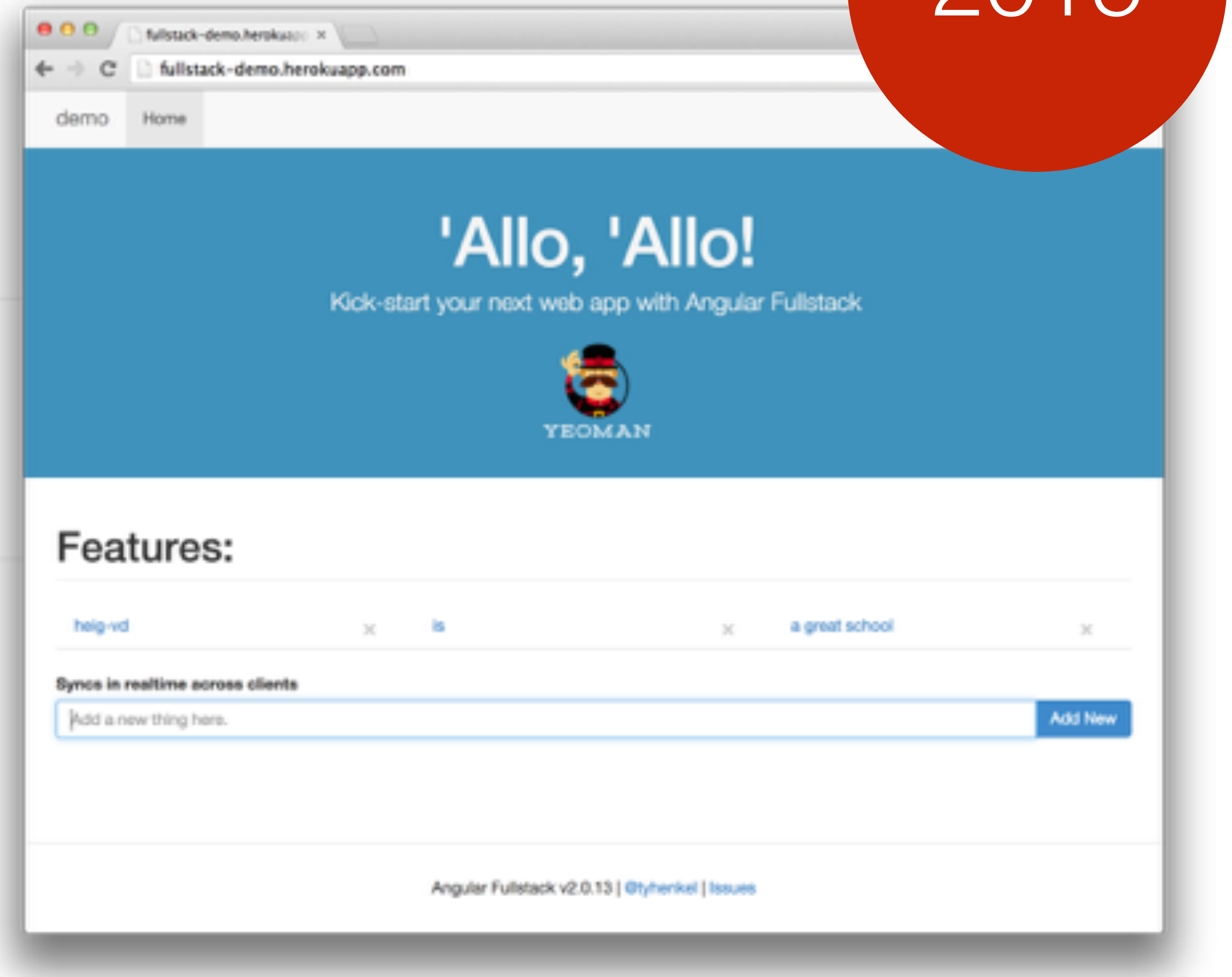
## AngularJS Full-Stack generator

> Yeoman generator for creating MEAN stack applications, using MongoDB, Express, AngularJS, and Node - lets you quickly set up a project following best practices.

### Example project

Generated with defaults: http://fullstack-demo.herokuapp.com/.

Source code: https://github.com/DaftMonk/fullstack-demo



'Allo, 'Allo!

Kick-start your next web app with Angular Fullstack

Features:

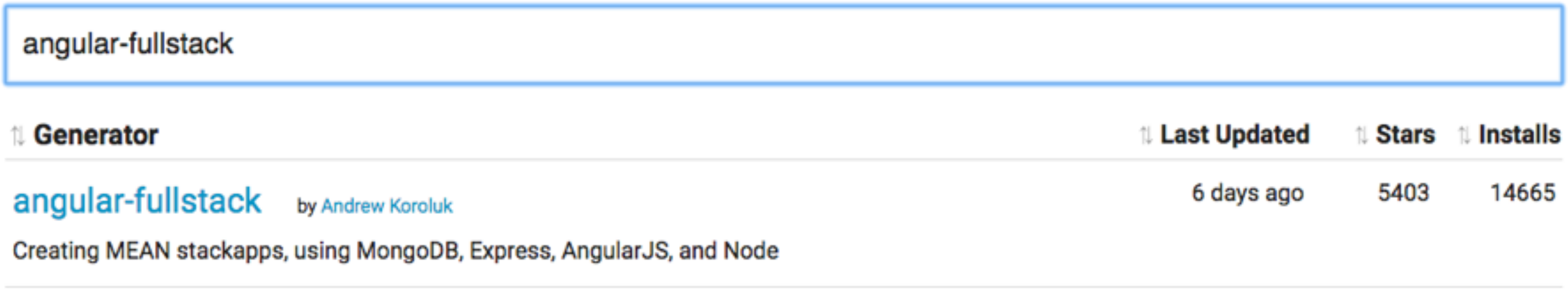
heig-vd ×    is ×    a great school ×

Syncs in realtime across clients

Add a new thing here.

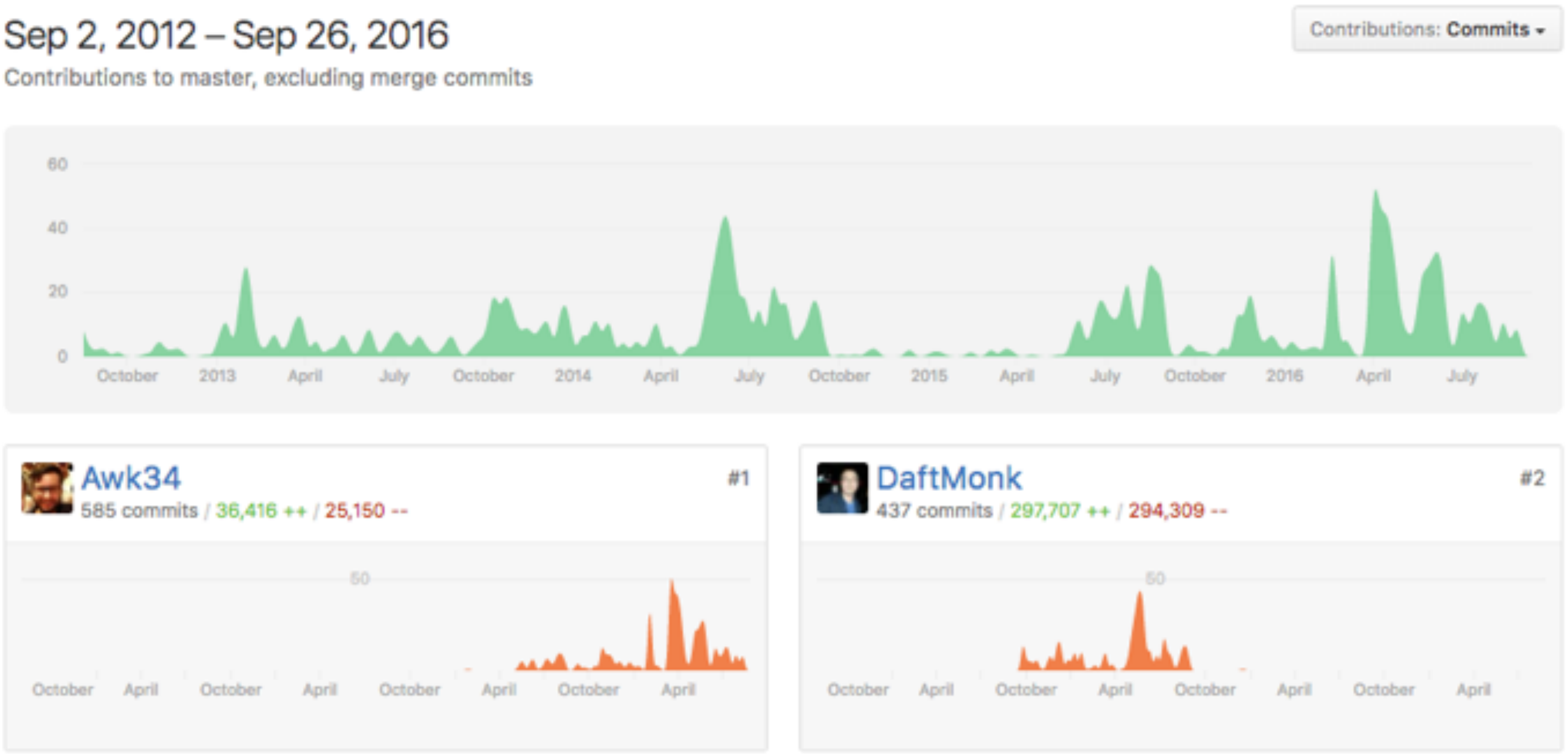
Angular Fullstack v2.0.13 | @tyhenkel | Issues

# Meet the **angular-fullstack** generator.



angular-fullstack

| ⇅ Generator | ⇅ Last Updated | ⇅ Stars | ⇅ Installs |
|---|---|---|---|
| angular-fullstack by Andrew Koroluk<br>Creating MEAN stackapps, using MongoDB, Express, AngularJS, and Node | 6 days ago | 5403 | 14665 |



Sep 2, 2012 – Sep 26, 2016

Contributions to master, excluding merge commits

Contributions: **Commits** ▾

Awk34 #1

585 commits / 36,416 ++ / 25,150 --

DaftMonk #2

437 commits / 297,707 ++ / 294,309 --

# **Why** and **when** is this generator interesting?

- It uses **Express.js** on the server side, **AngularJS** and **Socket.IO** on the client side and the glue between the frameworks.

- It has **sub-generators** to iteratively add new server-side and client-side components (new entities, new REST endpoints, new UI pages, etc.)

- The **structure of the AngularJS components** (folders and files were the UI elements are coded) follows best practices.

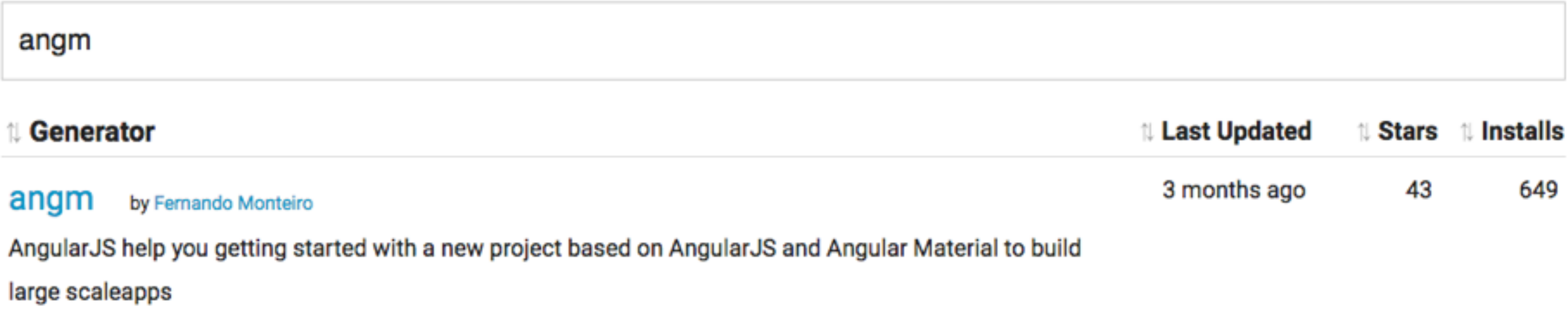
- It comes with support for **persistence** (with MongoDB), for **push notifications** (with Socket.IO) and for **authentication** (with Passport.js). The framework implements an interesting pattern for notifying CRUD operations to all connected users in realtime.

- Out-of-the-box, it provides a **complete and functional application** (which you can test on-line with their demo app).

- It supports **deployment on heroku** (and other cloud providers).

- The **drawback** is that the generated code and the build file is much more complex, compared to other generators. There is a much steeper learning curve and if you are starting with JavaScript, npm and Gulp, you can get lost and intimated.

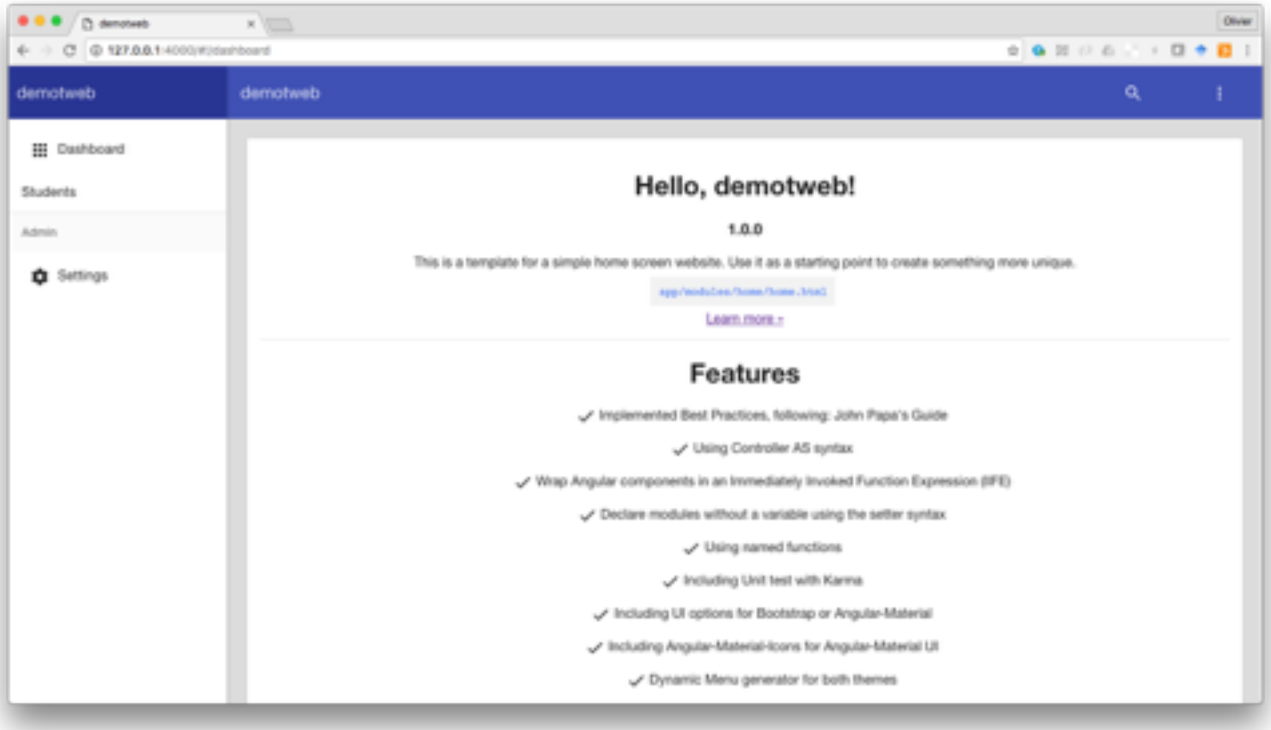
# Meet the **angm** generator.



angm

| ⇅ Generator | ⇅ Last Updated | ⇅ Stars | ⇅ Installs |
|---|---|---|---|
| angm by Fernando Monteiro<br>AngularJS help you getting started with a new project based on AngularJS and Angular Material to build large scaleapps | 3 months ago | 43 | 649 |

- Much simpler
- Only client-side
- Uses angular material
- **Used in the TWEB webcasts**



**Hello, demotweb!**

**1.0.0**

This is a template for a simple home screen website. Use it as a starting point to create something more unique.

**Features**

✓ Implemented Best Practices, following: John Papa's Guide

✓ Using Controller AS syntax

✓ Wrap Angular components in an Immediately Invoked Function Expression (IIFE)

✓ Declare modules without a variable using the setter syntax

✓ Using named functions

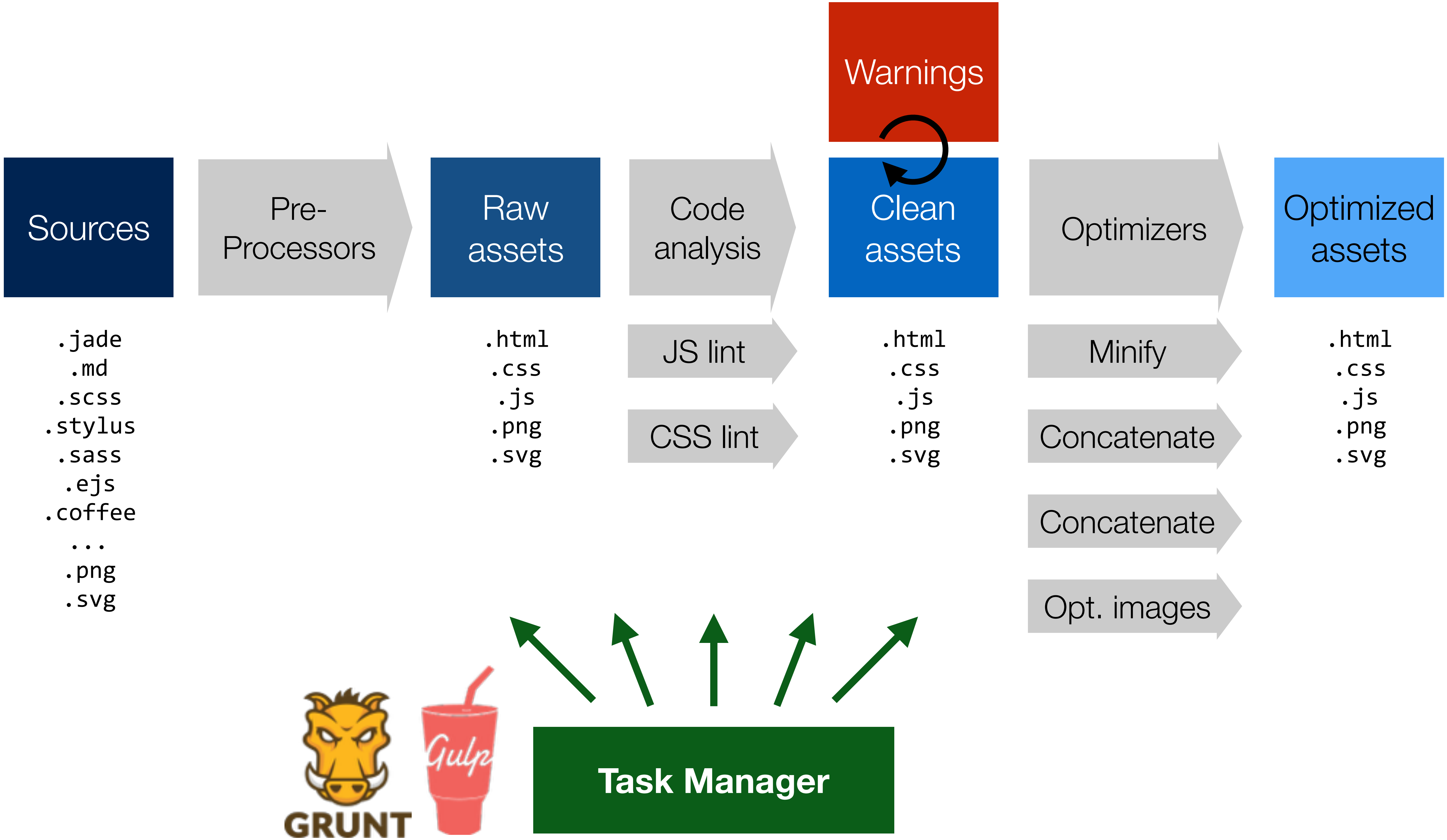
✓ Including Unit test with Karma

✓ Including UI options for Bootstrap or Angular-Material

✓ Including Angular-Material-Icons for Angular-Material UI

✓ Dynamic Menu generator for both themes

# Build Pipeline



Sources

.jade
.md
.scss
.stylus
.sass
.ejs
.coffee
...
.png
.svg

Pre-Processors

Raw assets

.html
.css
.js
.png
.svg

Code analysis

JS lint

CSS lint

Warnings

Clean assets

.html
.css
.js
.png
.svg

Optimizers

Minify

Concatenate

Concatenate

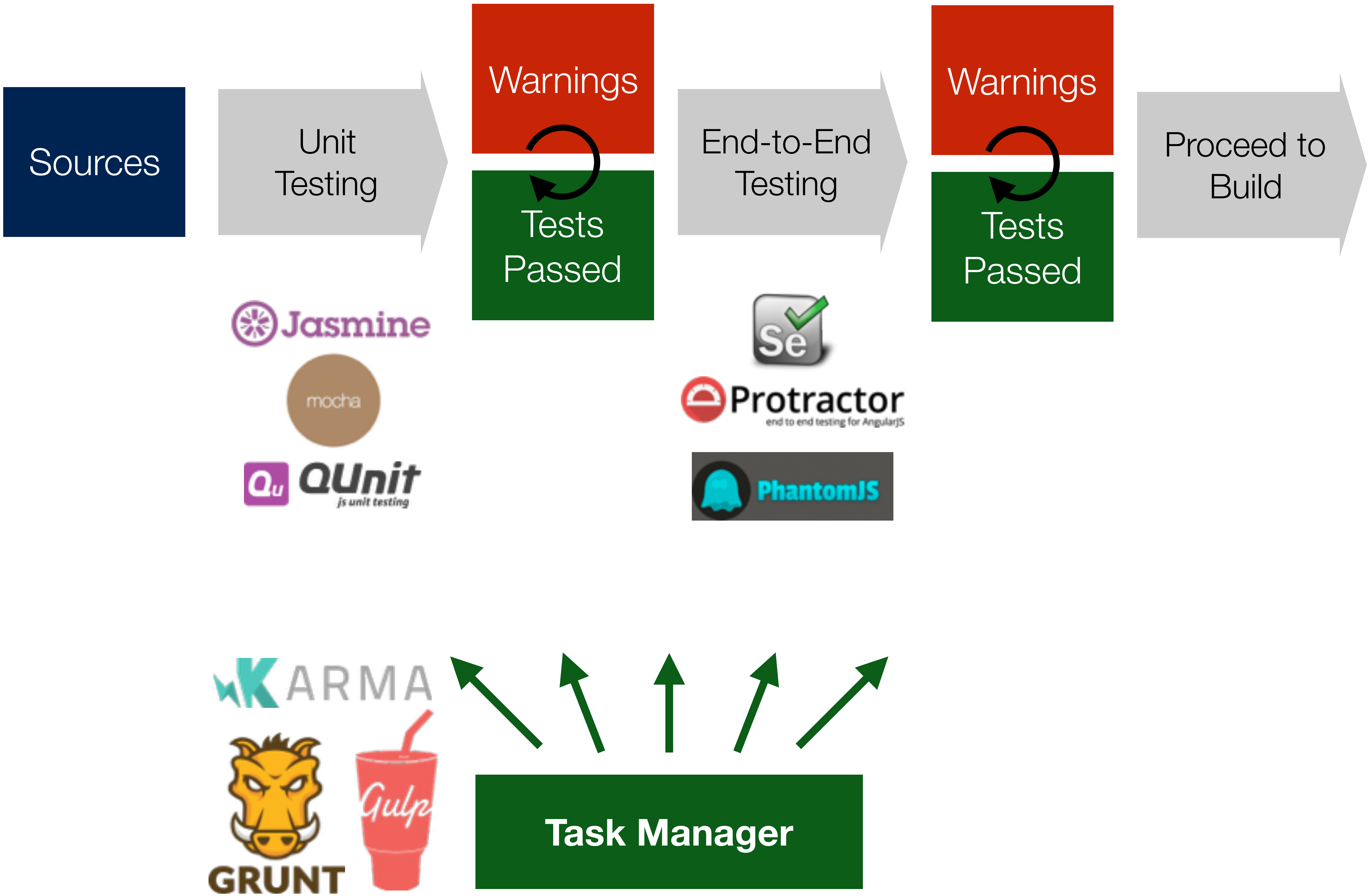
Opt. images

Optimized assets

.html
.css
.js
.png
.svg

Task Manager

# Test Pipeline



Sources

Unit Testing

Warnings

Tests Passed

End-to-End Testing

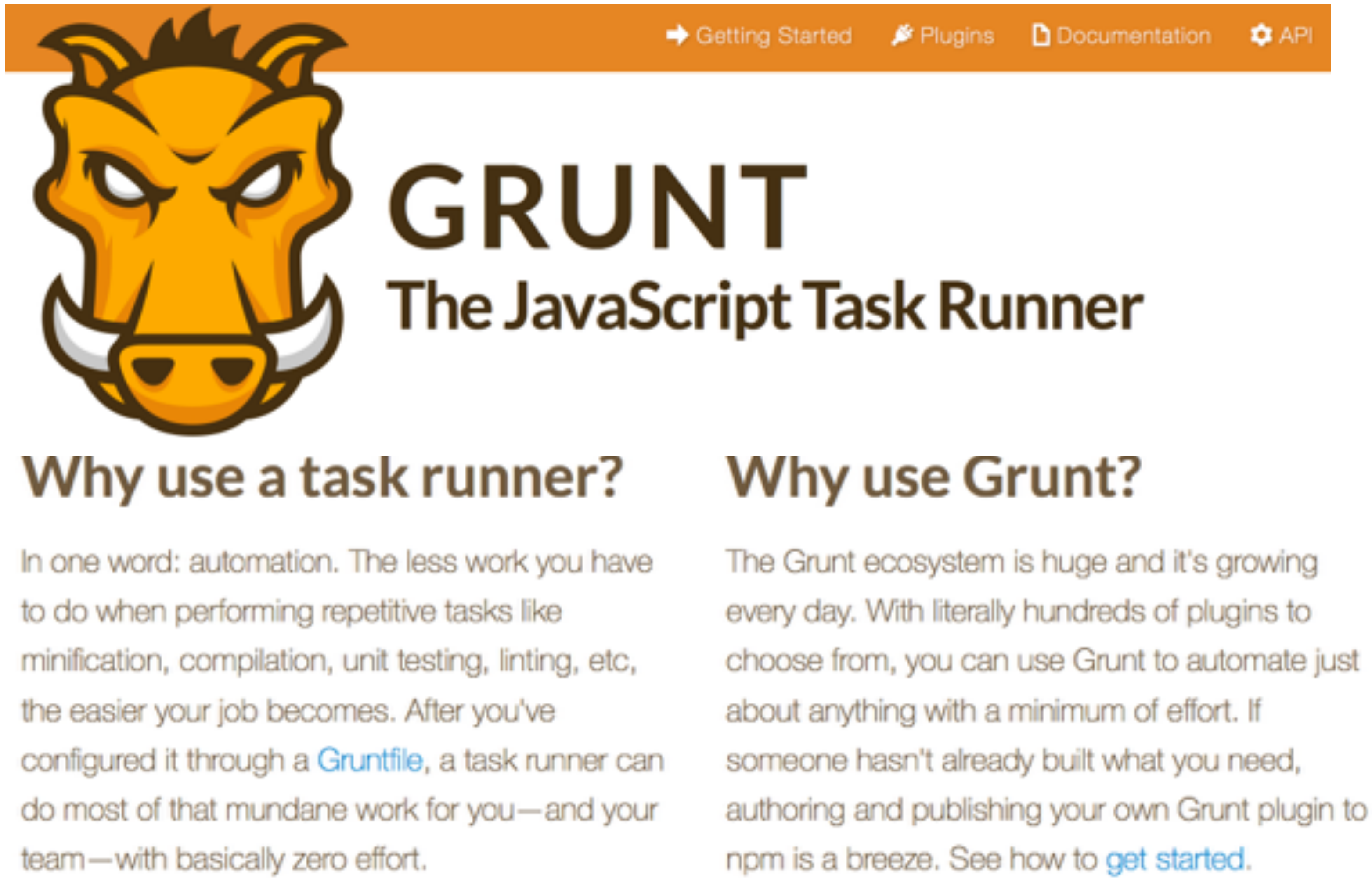
Warnings

Tests Passed

Proceed to Build

Task Manager

# Grunt



## GRUNT

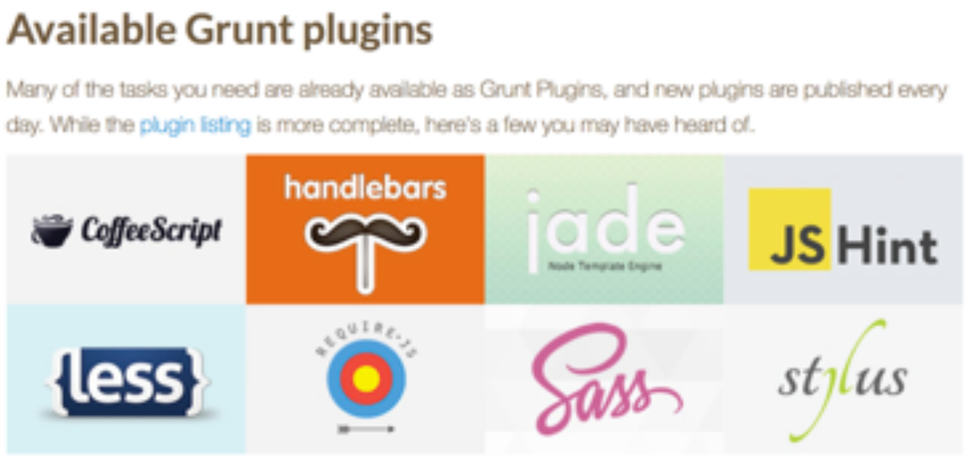
### The JavaScript Task Runner

## Why use a task runner?

In one word: automation. The less work you have to do when performing repetitive tasks like minification, compilation, unit testing, linting, etc, the easier your job becomes. After you've configured it through a Gruntfile, a task runner can do most of that mundane work for you—and your team—with basically zero effort.

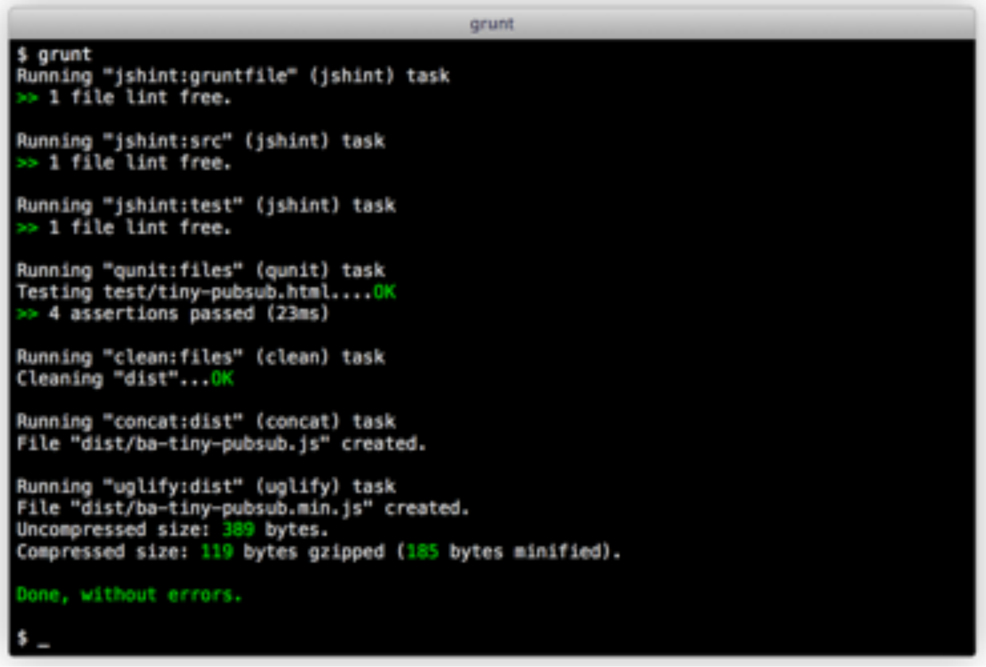
## Why use Grunt?

The Grunt ecosystem is huge and it's growing every day. With literally hundreds of plugins to choose from, you can use Grunt to automate just about anything with a minimum of effort. If someone hasn't already built what you need, authoring and publishing your own Grunt plugin to npm is a breeze. See how to get started.



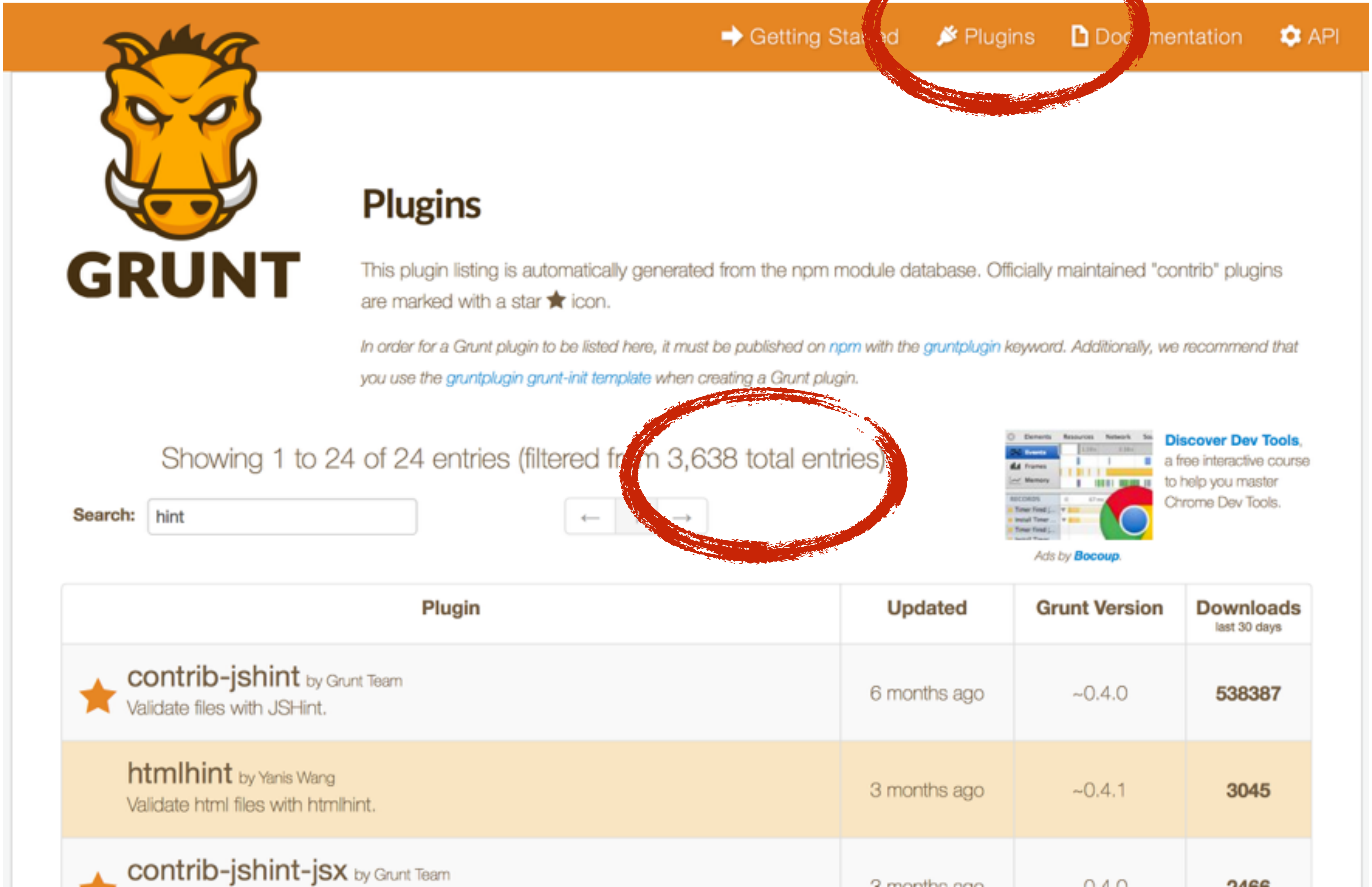
## Available Grunt plugins

Many of the tasks you need are already available as Grunt Plugins, and new plugins are published every day. While the plugin listing is more complete, here's a few you may have heard of.



```
$ grunt
Running "jshint:gruntfile" (jshint) task
>> 1 file lint free.

Running "jshint:src" (jshint) task
>> 1 file lint free.

Running "jshint:test" (jshint) task
>> 1 file lint free.

Running "qunit:files" (qunit) task
Testing test/tiny-pubsub.html....OK
>> 4 assertions passed (23ms)

Running "clean:files" (clean) task
Cleaning "dist"...OK

Running "concat:dist" (concat) task
File "dist/ba-tiny-pubsub.js" created.

Running "uglify:dist" (uglify) task
File "dist/ba-tiny-pubsub.min.js" created.
Uncompressed size: 389 bytes.
Compressed size: 119 bytes gzipped (185 bytes minified).

Done, without errors.

$ _
```

# Grunt



## Plugins

This plugin listing is automatically generated from the npm module database. Officially maintained "contrib" plugins are marked with a star ★ icon.

*In order for a Grunt plugin to be listed here, it must be published on npm with the gruntplugin keyword. Additionally, we recommend that you use the gruntplugin grunt-init template when creating a Grunt plugin.*

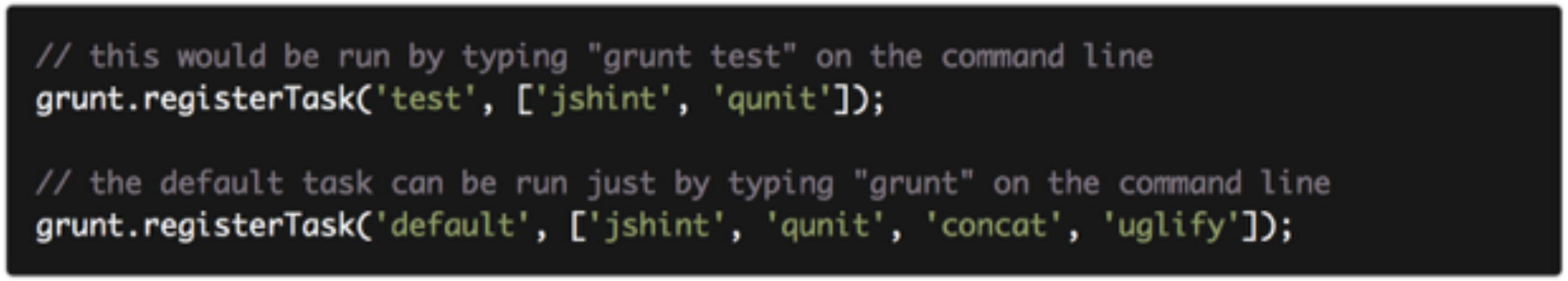
Showing 1 to 24 of 24 entries (filtered from 3,638 total entries)

Search: hint

| | Plugin | Updated | Grunt Version | Downloads<br>last 30 days |
|---|---|---|---|---|
| ★ | contrib-jshint by Grunt Team<br>Validate files with JSHint. | 6 months ago | ~0.4.0 | 538387 |
| | htmlhint by Yanis Wang<br>Validate html files with htmlhint. | 3 months ago | ~0.4.1 | 3045 |
| ★ | contrib-jshint-jsx by Grunt Team | 3 months ago | ~0.4.0 | 2466 |

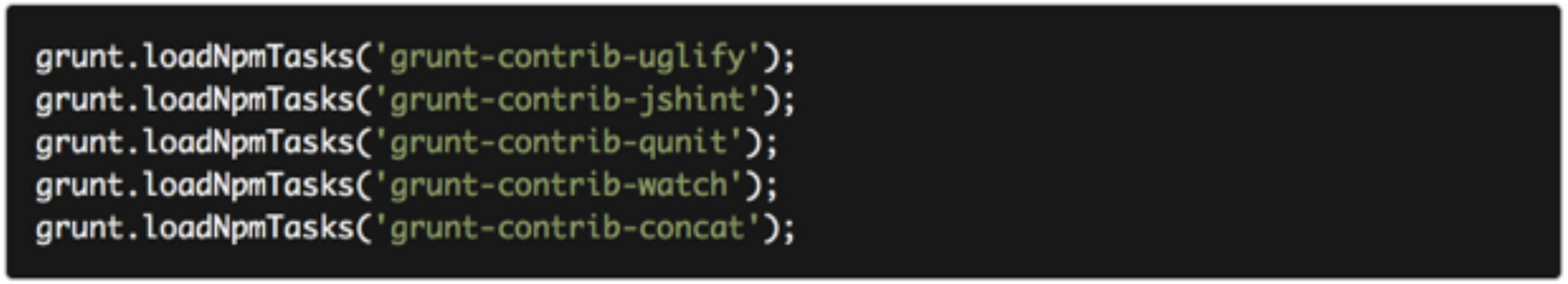
# Telling grunt what to do: **Gruntfile.js**

- Let's define **two workflows**: a "test" workflow and a "default" workflow. I will be able to type "grunt test" and a "grunt" on the command line to run them.



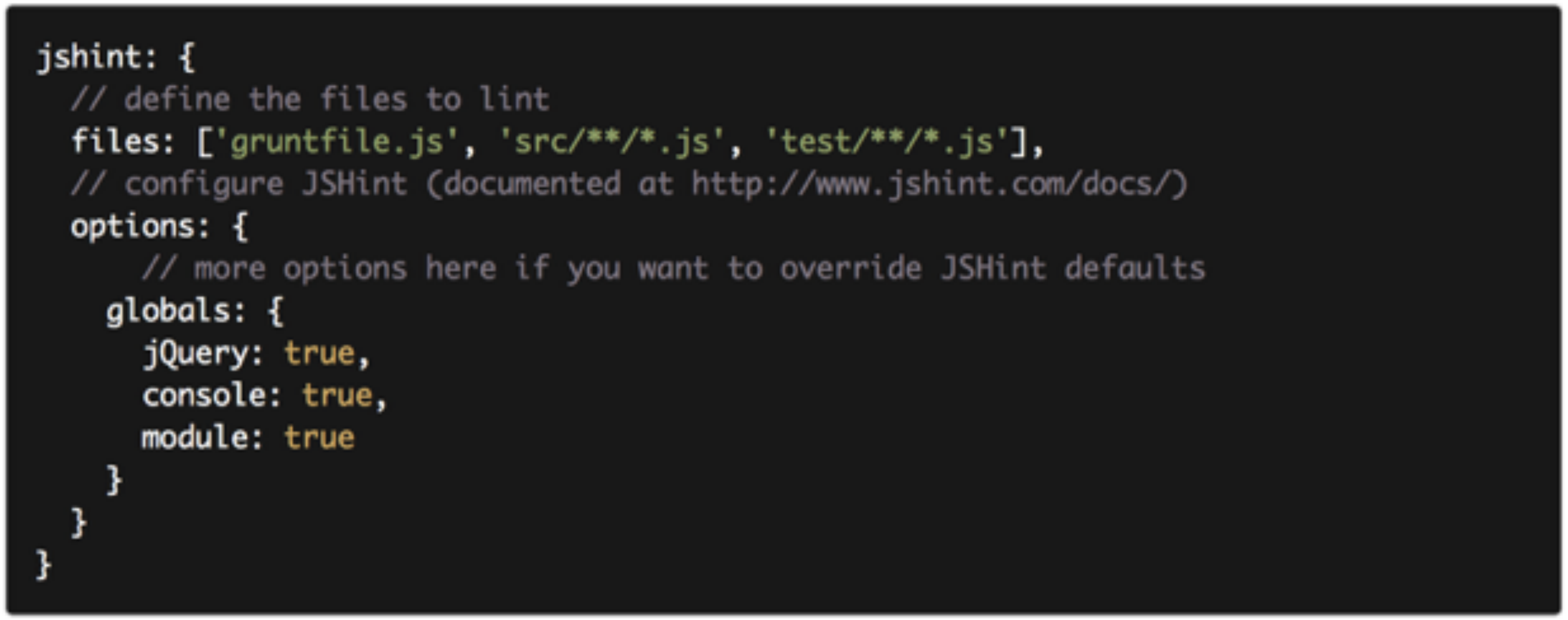
```
// this would be run by typing "grunt test" on the command line
grunt.registerTask('test', ['jshint', 'qunit']);

// the default task can be run just by typing "grunt" on the command line
grunt.registerTask('default', ['jshint', 'qunit', 'concat', 'uglify']);
```

- The workflows use a few standard grunt plugins. Let's load them in the Gruntfile.js.



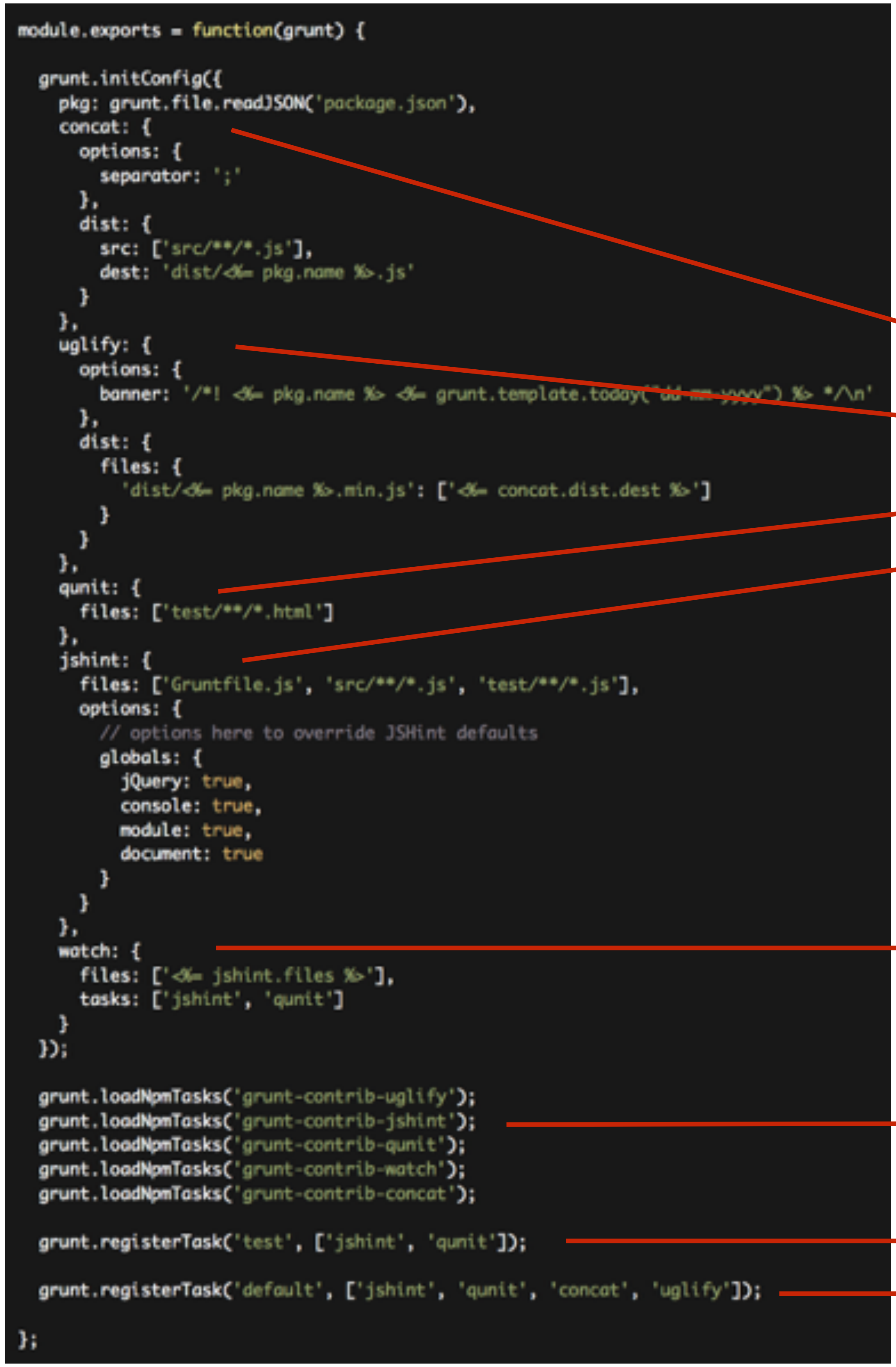
```
grunt.loadNpmTasks('grunt-contrib-uglify');
grunt.loadNpmTasks('grunt-contrib-jshint');
grunt.loadNpmTasks('grunt-contrib-qunit');
grunt.loadNpmTasks('grunt-contrib-watch');
grunt.loadNpmTasks('grunt-contrib-concat');
```

- Each grunt plugin can be configured. Here, we specify what files to lint and how.



```
jshint: {
  // define the files to lint
  files: ['gruntfile.js', 'src/**/*.js', 'test/**/*.js'],
  // configure JSHint (documented at http://www.jshint.com/docs/)
  options: {
      // more options here if you want to override JSHint defaults
    globals: {
      jQuery: true,
      console: true,
      module: true
    }
  }
}
```

# A simple, complete Gruntfile.js



```
module.exports = function(grunt) {

  grunt.initConfig({
    pkg: grunt.file.readJSON('package.json'),
    concat: {
      options: {
        separator: ';'
      },
      dist: {
        src: ['src/**/*.js'],
        dest: 'dist/<%= pkg.name %>.js'
      }
    },
    uglify: {
      options: {
        banner: '/*! <%= pkg.name %> <%= grunt.template.today("dd-mm-yyyy") %> */\n'
      },
      dist: {
        files: {
          'dist/<%= pkg.name %>.min.js': ['<%= concat.dist.dest %>']
        }
      }
    },
    qunit: {
      files: ['test/**/*.html']
    },
    jshint: {
      files: ['Gruntfile.js', 'src/**/*.js', 'test/**/*.js'],
      options: {
        // options here to override JSHint defaults
        globals: {
          jQuery: true,
          console: true,
          module: true,
          document: true
        }
      }
    },
    watch: {
      files: ['<%= jshint.files %>'],
      tasks: ['jshint', 'qunit']
    }
  });

  grunt.loadNpmTasks('grunt-contrib-uglify');
  grunt.loadNpmTasks('grunt-contrib-jshint');
  grunt.loadNpmTasks('grunt-contrib-qunit');
  grunt.loadNpmTasks('grunt-contrib-watch');
  grunt.loadNpmTasks('grunt-contrib-concat');

  grunt.registerTask('test', ['jshint', 'qunit']);

  grunt.registerTask('default', ['jshint', 'qunit', 'concat', 'uglify']);

};
```

http://gruntjs.com/sample-gruntfile

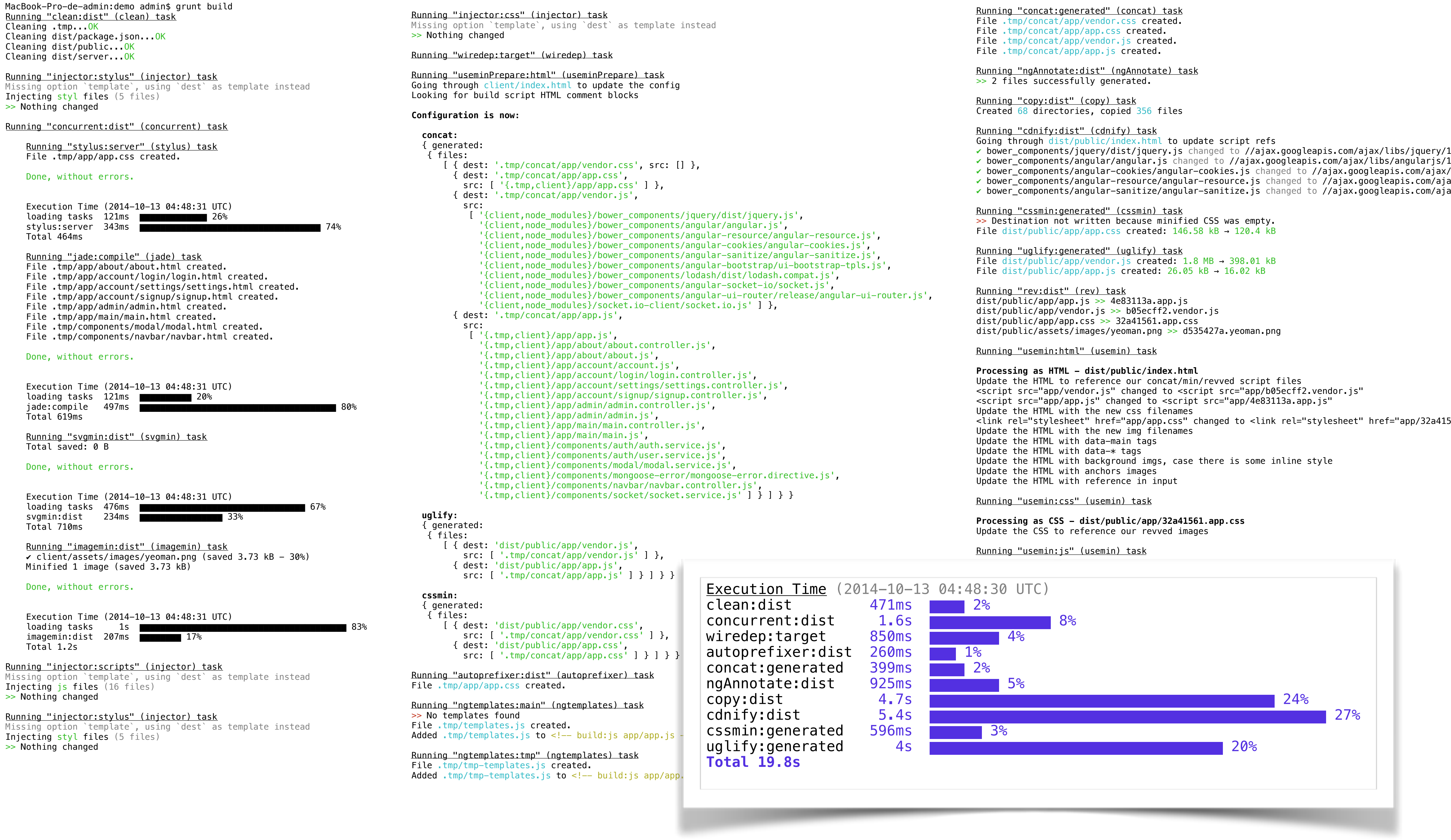
Configure the grunt plugins.

Watch is a special plugin. It allows to execute tasks when files are changed in the file system.

Load the grunt plugins.

Define the two workflows.

# Executing a more complex workflow...



```
MacBook-Pro-de-admin:demo admin$ grunt build
Running "clean:dist" (clean) task
Cleaning .tmp...OK
Cleaning dist/package.json...OK
Cleaning dist/public...OK
Cleaning dist/server...OK

Running "injector:stylus" (injector) task
Missing option `template`, using `dest` as template instead
Injecting styl files (5 files)
>> Nothing changed

Running "concurrent:dist" (concurrent) task

    Running "stylus:server" (stylus) task
    File .tmp/app/app.css created.

    Done, without errors.


    Execution Time (2014-10-13 04:48:31 UTC)
    loading tasks  121ms  █████ 26%
    stylus:server  343ms  ██████████████ 74%
    Total 464ms

    Running "jade:compile" (jade) task
    File .tmp/app/about/about.html created.
    File .tmp/app/account/login/login.html created.
    File .tmp/app/account/settings/settings.html created.
    File .tmp/app/account/signup/signup.html created.
    File .tmp/app/admin/admin.html created.
    File .tmp/app/main/main.html created.
    File .tmp/components/modal/modal.html created.
    File .tmp/components/navbar/navbar.html created.

    Done, without errors.


    Execution Time (2014-10-13 04:48:31 UTC)
    loading tasks  121ms  ████ 20%
    jade:compile   497ms  ███████████████ 80%
    Total 619ms

    Running "svgmin:dist" (svgmin) task
    Total saved: 0 B

    Done, without errors.


    Execution Time (2014-10-13 04:48:31 UTC)
    loading tasks  476ms  █████████████ 67%
    svgmin:dist    234ms  ██████ 33%
    Total 710ms

    Running "imagemin:dist" (imagemin) task
    ✔ client/assets/images/yeoman.png (saved 3.73 kB - 30%)
    Minified 1 image (saved 3.73 kB)

    Done, without errors.


    Execution Time (2014-10-13 04:48:31 UTC)
    loading tasks     1s  ████████████████ 83%
    imagemin:dist  207ms  ███ 17%
    Total 1.2s

Running "injector:scripts" (injector) task
Missing option `template`, using `dest` as template instead
Injecting js files (16 files)
>> Nothing changed

Running "injector:stylus" (injector) task
Missing option `template`, using `dest` as template instead
Injecting styl files (5 files)
>> Nothing changed
```

```
Running "injector:css" (injector) task
Missing option `template`, using `dest` as template instead
>> Nothing changed

Running "wiredep:target" (wiredep) task

Running "useminPrepare:html" (useminPrepare) task
Going through client/index.html to update the config
Looking for build script HTML comment blocks

Configuration is now:

  concat:
  { generated:
   { files:
      [ { dest: '.tmp/concat/app/vendor.css', src: [] },
        { dest: '.tmp/concat/app/app.css',
          src: [ '{.tmp,client}/app/app.css' ] },
        { dest: '.tmp/concat/app/vendor.js',
          src:
           [ '{client,node_modules}/bower_components/jquery/dist/jquery.js',
             '{client,node_modules}/bower_components/angular/angular.js',
             '{client,node_modules}/bower_components/angular-resource/angular-resource.js',
             '{client,node_modules}/bower_components/angular-cookies/angular-cookies.js',
             '{client,node_modules}/bower_components/angular-sanitize/angular-sanitize.js',
             '{client,node_modules}/bower_components/angular-bootstrap/ui-bootstrap-tpls.js',
             '{client,node_modules}/bower_components/lodash/dist/lodash.compat.js',
             '{client,node_modules}/bower_components/angular-socket-io/socket.js',
             '{client,node_modules}/bower_components/angular-ui-router/release/angular-ui-router.js',
             '{client,node_modules}/socket.io-client/socket.io.js' ] },
        { dest: '.tmp/concat/app/app.js',
          src:
           [ '{.tmp,client}/app/app.js',
             '{.tmp,client}/app/about/about.controller.js',
             '{.tmp,client}/app/about/about.js',
             '{.tmp,client}/app/account/account.js',
             '{.tmp,client}/app/account/login/login.controller.js',
             '{.tmp,client}/app/account/settings/settings.controller.js',
             '{.tmp,client}/app/account/signup/signup.controller.js',
             '{.tmp,client}/app/admin/admin.controller.js',
             '{.tmp,client}/app/admin/admin.js',
             '{.tmp,client}/app/main/main.controller.js',
             '{.tmp,client}/app/main/main.js',
             '{.tmp,client}/components/auth/auth.service.js',
             '{.tmp,client}/components/auth/user.service.js',
             '{.tmp,client}/components/modal/modal.service.js',
             '{.tmp,client}/components/mongoose-error/mongoose-error.directive.js',
             '{.tmp,client}/components/navbar/navbar.controller.js',
             '{.tmp,client}/components/socket/socket.service.js' ] } ] } }

  uglify:
  { generated:
   { files:
      [ { dest: 'dist/public/app/vendor.js',
          src: [ '.tmp/concat/app/vendor.js' ] },
        { dest: 'dist/public/app/app.js',
          src: [ '.tmp/concat/app/app.js' ] } ] } }

  cssmin:
  { generated:
   { files:
      [ { dest: 'dist/public/app/vendor.css',
          src: [ '.tmp/concat/app/vendor.css' ] },
        { dest: 'dist/public/app/app.css',
          src: [ '.tmp/concat/app/app.css' ] } ] } }

Running "autoprefixer:dist" (autoprefixer) task
File .tmp/app/app.css created.

Running "ngtemplates:main" (ngtemplates) task
>> No templates found
File .tmp/templates.js created.
Added .tmp/templates.js to <!-- build:js app/app.js -

Running "ngtemplates:tmp" (ngtemplates) task
File .tmp/tmp-templates.js created.
Added .tmp/tmp-templates.js to <!-- build:js app/app.
```

```
Running "concat:generated" (concat) task
File .tmp/concat/app/vendor.css created.
File .tmp/concat/app/app.css created.
File .tmp/concat/app/vendor.js created.
File .tmp/concat/app/app.js created.

Running "ngAnnotate:dist" (ngAnnotate) task
>> 2 files successfully generated.

Running "copy:dist" (copy) task
Created 68 directories, copied 356 files

Running "cdnify:dist" (cdnify) task
Going through dist/public/index.html to update script refs
✔ bower_components/jquery/dist/jquery.js changed to //ajax.googleapis.com/ajax/libs/jquery/1
✔ bower_components/angular/angular.js changed to //ajax.googleapis.com/ajax/libs/angularjs/1
✔ bower_components/angular-cookies/angular-cookies.js changed to //ajax.googleapis.com/ajax/
✔ bower_components/angular-resource/angular-resource.js changed to //ajax.googleapis.com/aja
✔ bower_components/angular-sanitize/angular-sanitize.js changed to //ajax.googleapis.com/aja

Running "cssmin:generated" (cssmin) task
>> Destination not written because minified CSS was empty.
File dist/public/app/app.css created: 146.58 kB → 120.4 kB

Running "uglify:generated" (uglify) task
File dist/public/app/vendor.js created: 1.8 MB → 398.01 kB
File dist/public/app/app.js created: 26.05 kB → 16.02 kB

Running "rev:dist" (rev) task
dist/public/app/app.js >> 4e83113a.app.js
dist/public/app/vendor.js >> b05ecff2.vendor.js
dist/public/app/app.css >> 32a41561.app.css
dist/public/assets/images/yeoman.png >> d535427a.yeoman.png

Running "usemin:html" (usemin) task

Processing as HTML - dist/public/index.html
Update the HTML to reference our concat/min/revved script files
<script src="app/vendor.js" changed to <script src="app/b05ecff2.vendor.js"
<script src="app/app.js" changed to <script src="app/4e83113a.app.js"
Update the HTML with the new css filenames
<link rel="stylesheet" href="app/app.css" changed to <link rel="stylesheet" href="app/32a415
Update the HTML with the new img filenames
Update the HTML with data-main tags
Update the HTML with data-* tags
Update the HTML with background imgs, case there is some inline style
Update the HTML with anchors images
Update the HTML with reference in input

Running "usemin:css" (usemin) task

Processing as CSS - dist/public/app/32a41561.app.css
Update the CSS to reference our revved images

Running "usemin:js" (usemin) task
```

```
Execution Time (2014-10-13 04:48:30 UTC)
clean:dist          471ms  ██ 2%
concurrent:dist      1.6s  ██████ 8%
wiredep:target      850ms  ████ 4%
autoprefixer:dist   260ms  █ 1%
concat:generated    399ms  ██ 2%
ngAnnotate:dist     925ms  ████ 5%
copy:dist            4.7s  █████████████████████ 24%
cdnify:dist          5.4s  ████████████████████████ 27%
cssmin:generated    596ms  ███ 3%
uglify:generated       4s  ██████████████████ 20%
Total 19.8s
```

# Problem

How can we deploy our services "to the cloud"?

# Forces

- Our app consists of multiple components (services): front-end assets, back-end API, crawler.
- We want to make them publicly available, so we need to “deploy” them somewhere.
- There are many ways to do that, how do we pick one? In our particular case, money and ease of use are 2 constraints.

# Solution

- We can use a combination of "providers" to deploy our different components.
- Serving the client-side assets is very easy with GitHub Pages.
- For the back-end API and the crawler, Heroku is a PaaS provider that we can use for free.

# Solution (2)

- But what is a PaaS?

- It means Platform as a Service

- It is one type of "cloud provider", which allows you to deploy applications (you don't worry about the OS, the DB, etc.)

- Other types of "cloud providers" include SaaS (e.g. Google Docs) and IaaS (e.g. Amazon Web Services EC2)

# In practice (1)

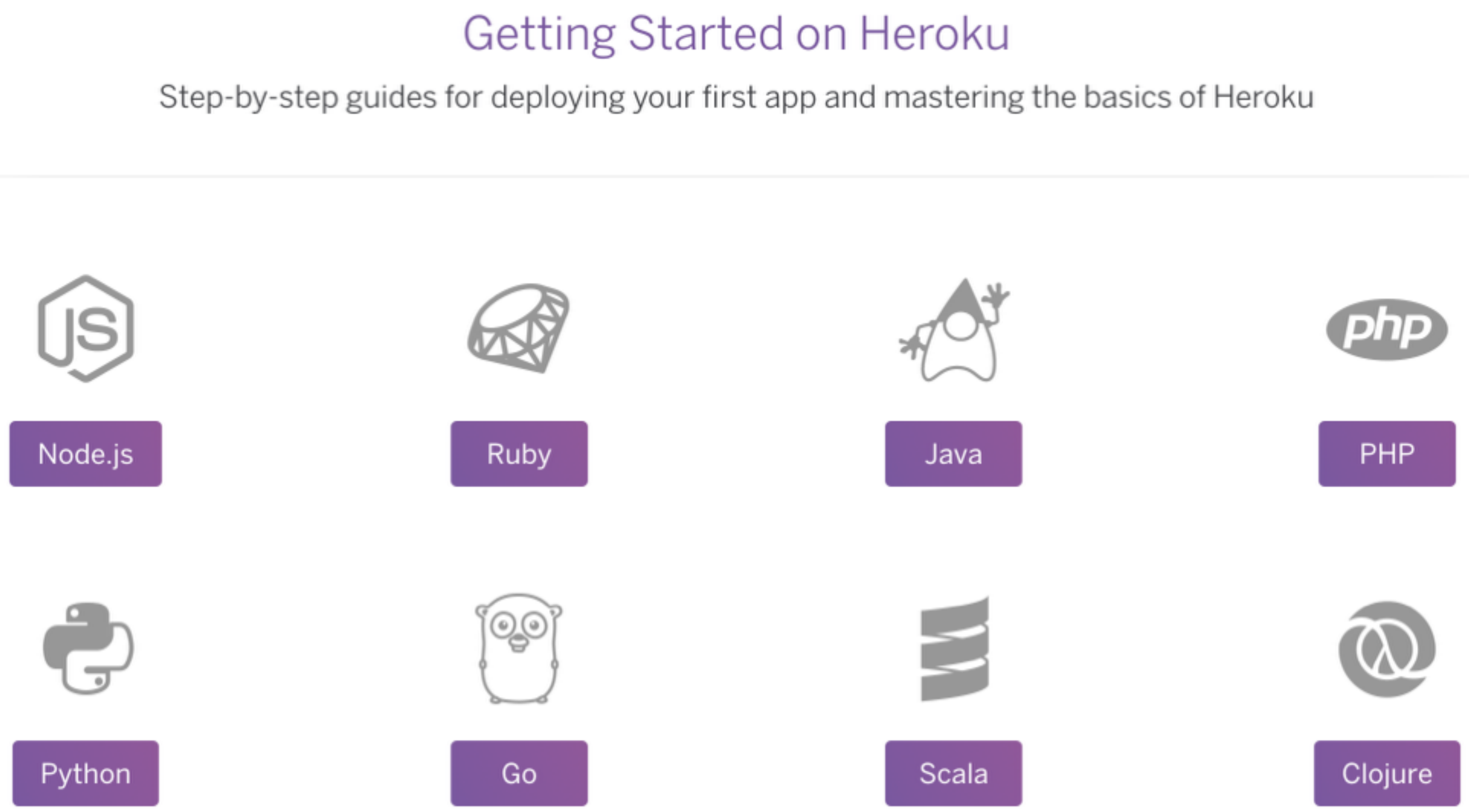
- Experiment with GitHub Pages:
  - Create a repo, configure Pages
  - Push assets, view web site
- Understand that for this project, you will use “bare bones” GitHub Pages. Later on, we will look at Jekyll.

https://pages.github.com/

# In practice (2)

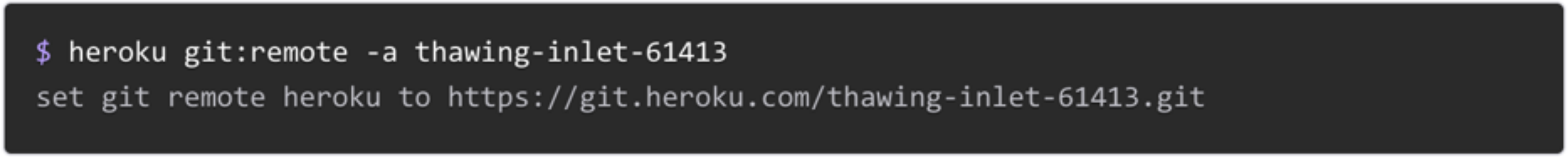
- A quick tour of Heroku
  - What is heroku?
  - How to deploy the backend? Node.js vs Docker
  - Add-ons
- About

# https://devcenter.heroku.com/start

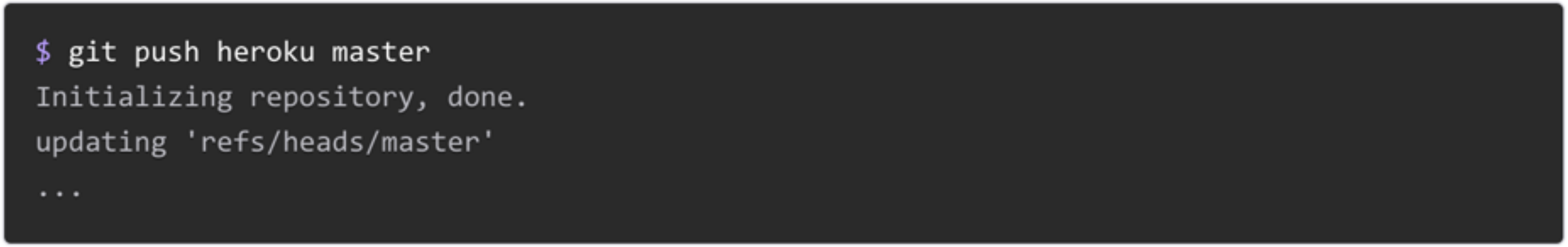


## Getting Started on Heroku

Step-by-step guides for deploying your first app and mastering the basics of Heroku

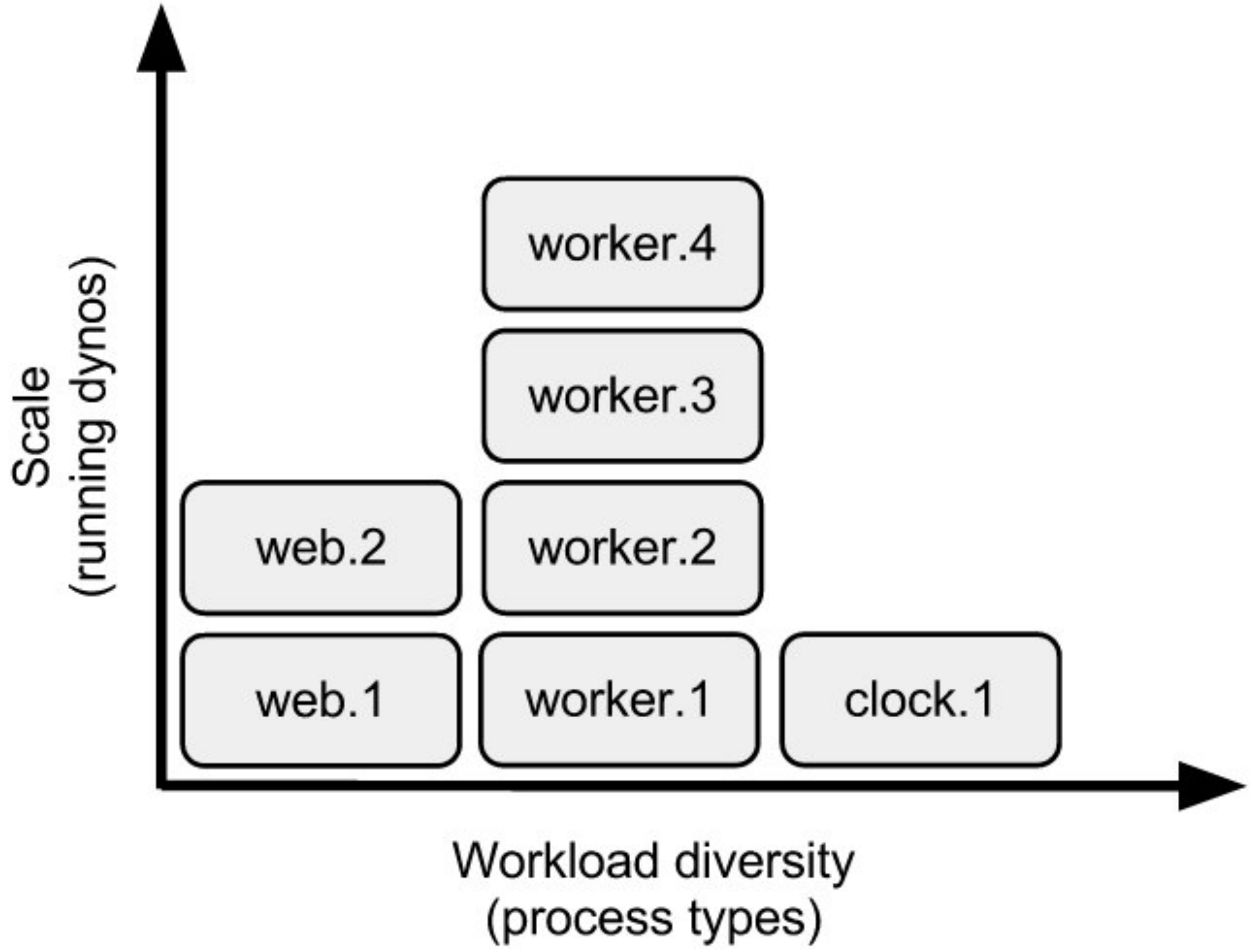


```
$ heroku git:remote -a thawing-inlet-61413
set git remote heroku to https://git.heroku.com/thawing-inlet-61413.git
```



```
$ git push heroku master
Initializing repository, done.
updating 'refs/heads/master'
...
```

# https://devcenter.heroku.com/start



Many apps need to run jobs at scheduled times. For example, polling a remote API every 5 minutes, or sending email reports every night at midnight.

Scheduler runs one-off dynos that will count towards your usage for the month. Dyno-hours from Scheduler tasks are counted just like those from heroku run or from scaled dynos.

https://devcenter.heroku.com/articles/scheduled-jobs-custom-clock-processes
https://devcenter.heroku.com/articles/scheduled-jobs-custom-clock-processes#simple-job-scheduling
https://devcenter.heroku.com/articles/scheduler

# ToDo List

- Validate that we are able to serve our client assets (HTML + CSS + Javascript) with GitHub Pages.
- Follow the Heroku tutorial and learn how to deploy a Node.js application (alternative: look at docker deployment)
- Validate that you are able to deploy your back-end API server to Heroku.
- Experiment with "one-off" dynos and the scheduler add-on.
- validate that you can execute your script on a period basis.