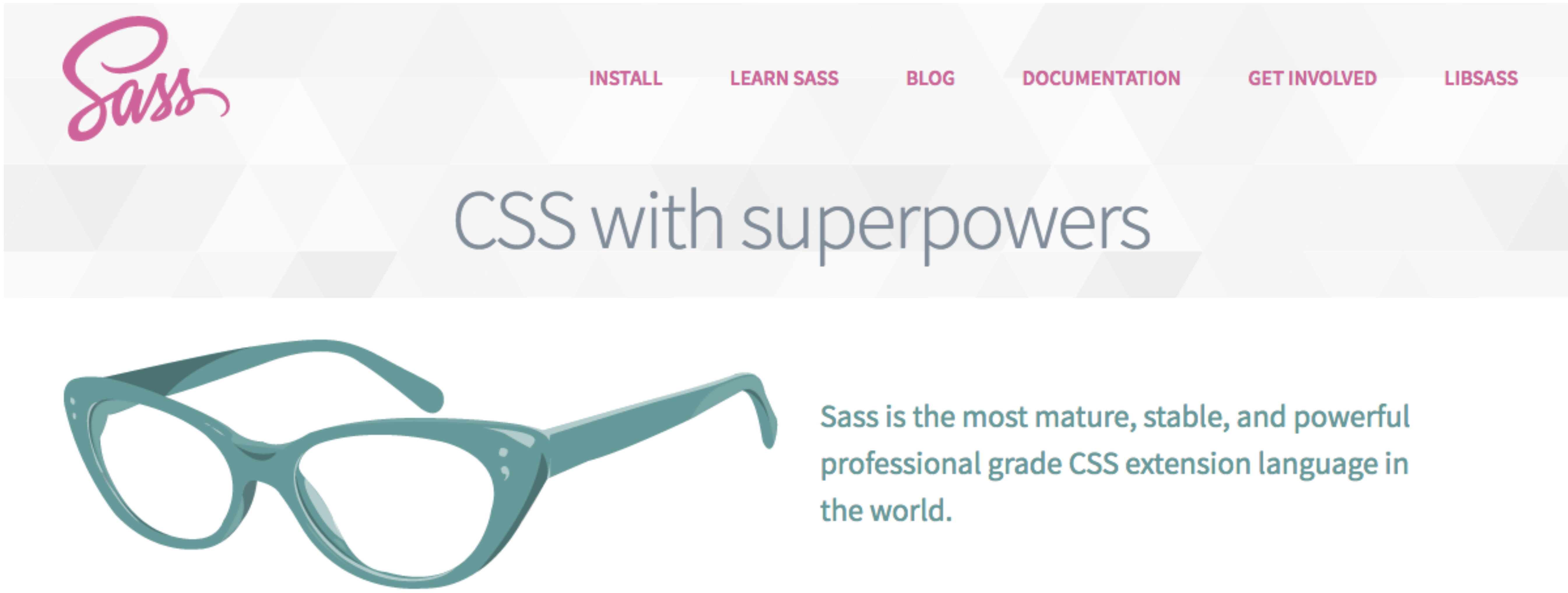
CSS Preprocessors & Semantic UI

CSS Preprocessing languages

- A CSS preprocessor helps write maintainable, future-proof code and it will seriously reduce the amount of CSS you have to write.
- Where these tools shine best are in large-scale user interfaces that require huge stylesheets and many style rules.
- Two candidates:
  - SASS
  - LESS

SASS



CSS with superpowers

Sass is the most mature, stable, and powerful professional grade CSS extension language in the world.

Preprocessing

Variables

Nesting

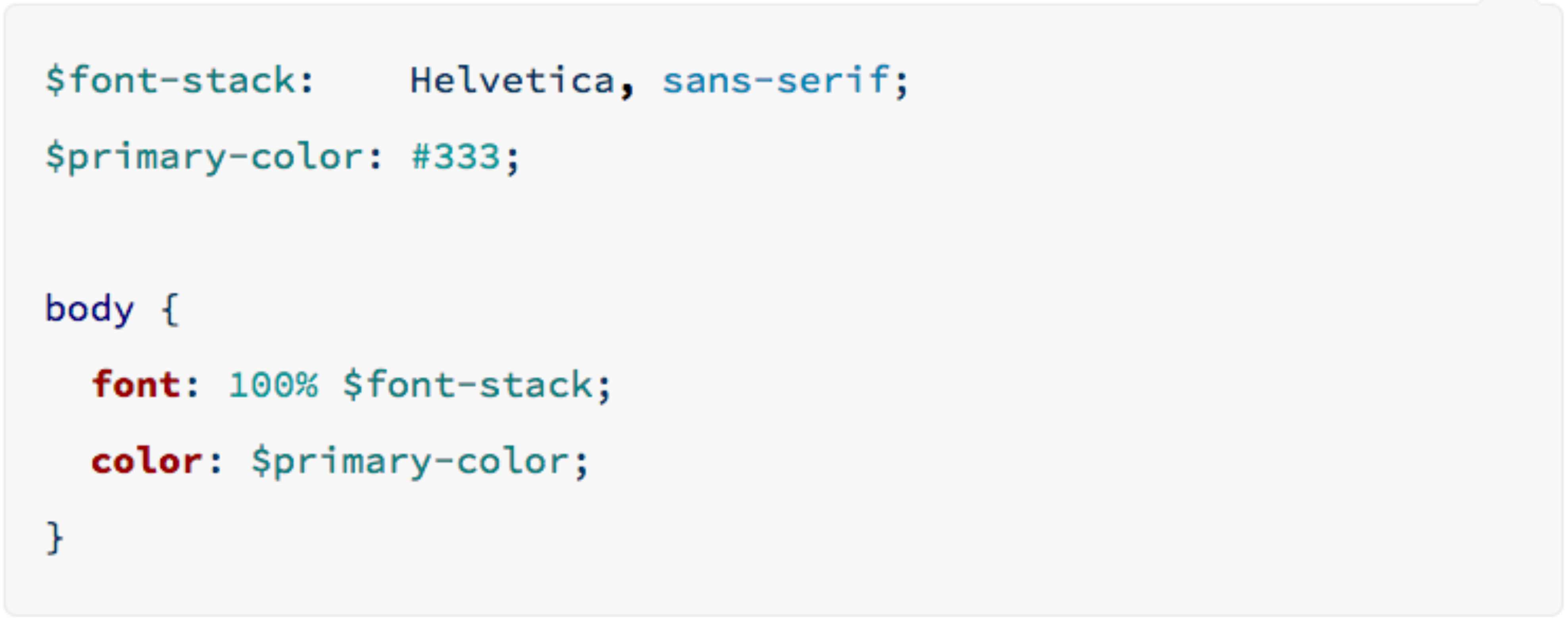
Partials

Import

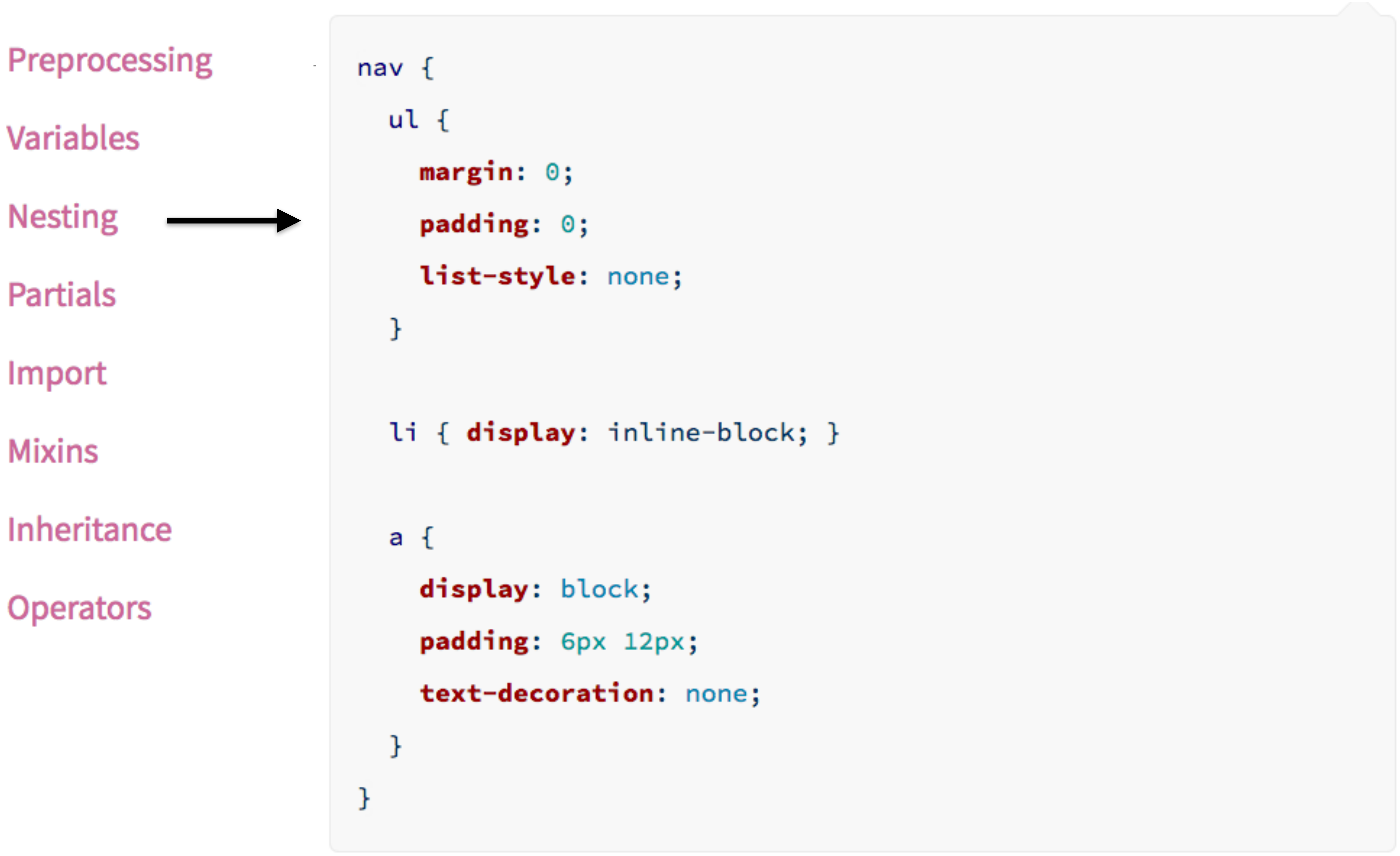
Mixins

Inheritance

Operators



```scss
$font-stack:    Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```



Preprocessing

Variables

Nesting

Partials

Import

Mixins

Inheritance

Operators

```
nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }

  li { display: inline-block; }

  a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
  }
}
```

Preprocessing

Variables

Nesting

Partials

Import

Mixins

Inheritance

Operators

```scss
// _reset.scss

html,
body,
ul,
ol {
   margin: 0;
  padding: 0;
}
```

```scss
/* base.scss */

@import 'reset';

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef;
}
```

Preprocessing

Variables

Nesting

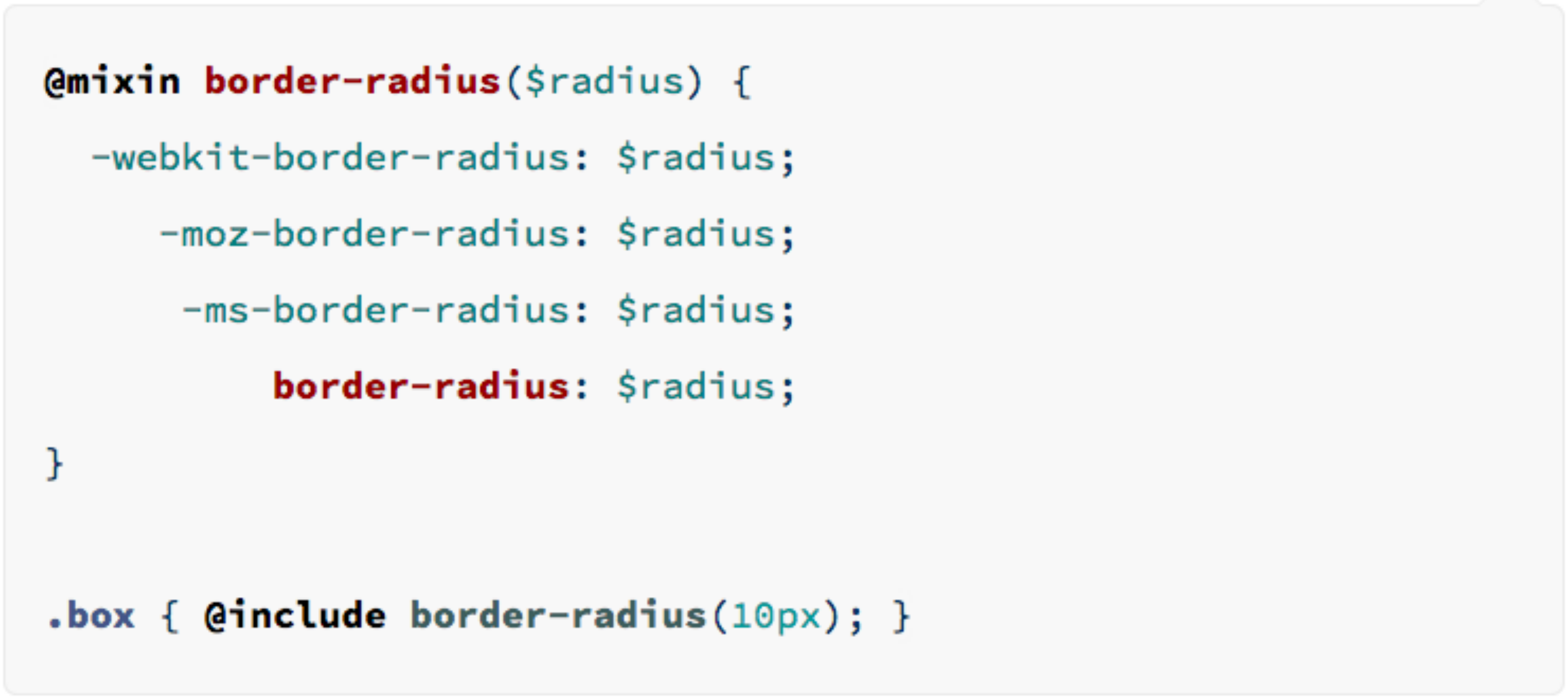
Partials

Import

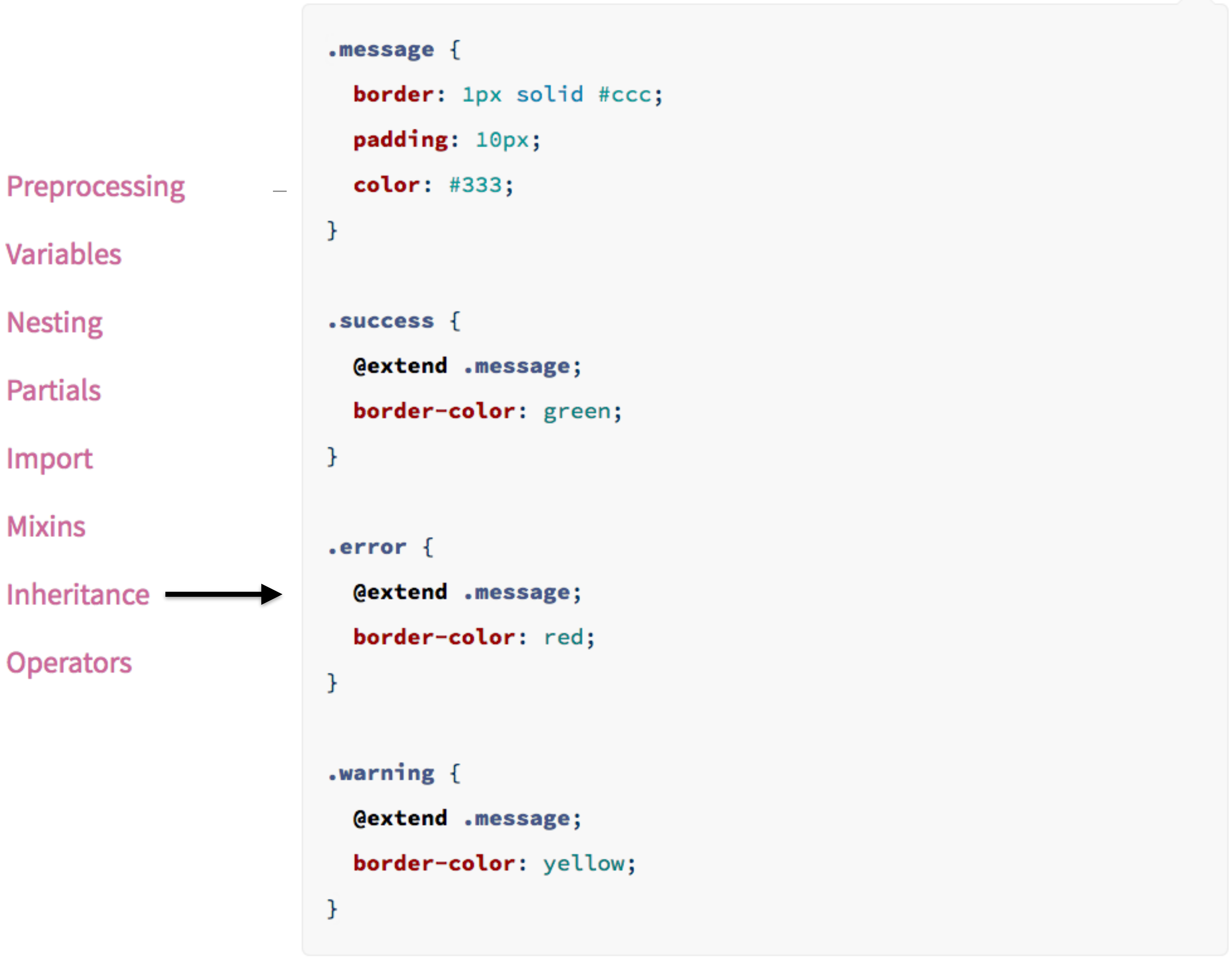
Mixins

Inheritance

Operators



```scss
@mixin border-radius($radius) {
  -webkit-border-radius: $radius;
     -moz-border-radius: $radius;
      -ms-border-radius: $radius;
          border-radius: $radius;
}

.box { @include border-radius(10px); }
```



Preprocessing

Variables

Nesting

Partials

Import

Mixins

Inheritance

Operators

```
.message {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}

.success {
  @extend .message;
  border-color: green;
}

.error {
  @extend .message;
  border-color: red;
}

.warning {
  @extend .message;
  border-color: yellow;
}
```

Preprocessing

Variables

Nesting

Partials

Import

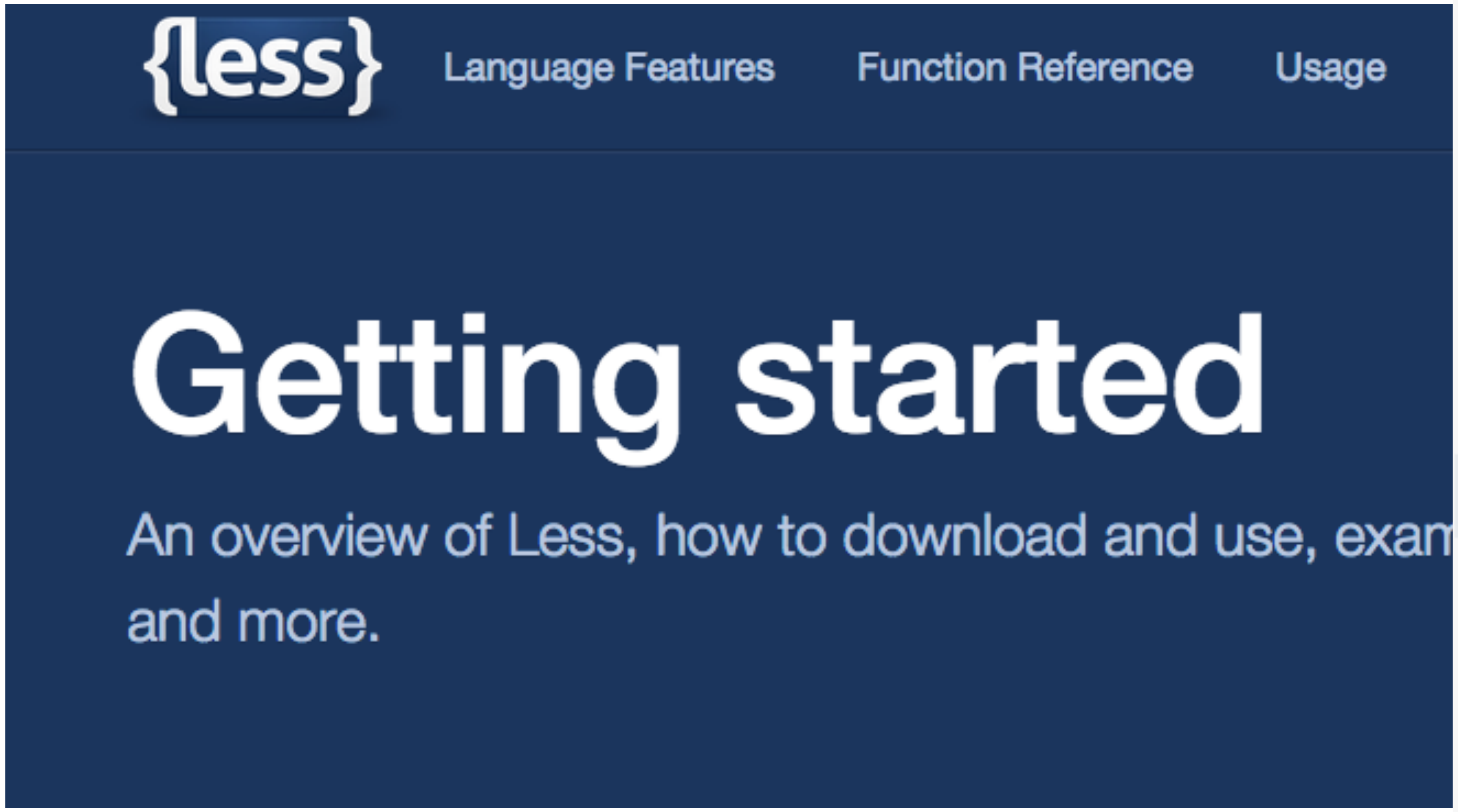
Mixins

Inheritance

Operators →

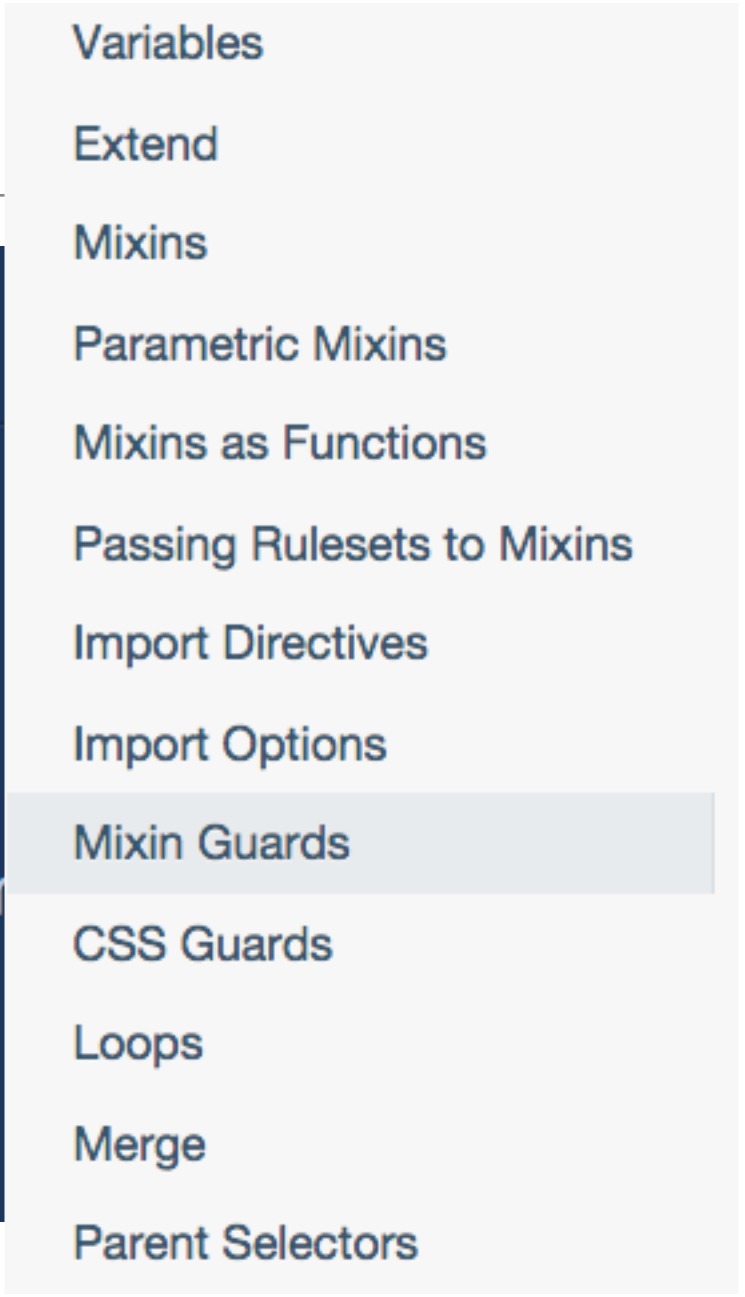
```
.container { width: 100%; }


article[role="main"] {
  float: left;
  width: 600px / 960px * 100%;
}


aside[role="complimentary"] {
  float: right;
  width: 300px / 960px * 100%;
}
```

LESS


{less} Language Features Function Reference Usage

Getting started

An overview of Less, how to download and use, exam
and more.



- Less is a CSS pre-processor, meaning that it extends the CSS language, adding features that allow variables, mixins, functions and many other techniques that allow you to make CSS that is more maintainable, themable and extendable.

Less Variables

Less

```
@color : #33333;

p {
  color : @color;
}

.demo {
  background : @color;
}
```

Generated Css

```
p {
  color : #333333;
}

.demo {
  background : #333333;
}
```

Less Scope

Less

```
@var: red;

#page {
  #header {
    @var: white;
    color: @var; // white
  }
  color : @var;
}
```

Generated Css

```
#page {
  color: #ff0000;
}
#page #header {
  color: #ffffff;
}
```

Less Mixins

Less

```
.demo-class {
  color : #aaa;
  font-size : 20px;
}

.class-A {
  .demo-class;
  background : #000;
}
```

Generated Css

```
demo-class {
  color: #aaa;
  font-size: 20px;
}

.class-A {
  color : #aaa;
  font-size : 20px;
  background : #000;
}
```

Less Parametric Mixins

Less

```
demo-class(@padding){
  padding : @padding;
}

.class-A {
  .demo-class(5px);
  background : #000;
}

.class-B {
  .demo-class(8px);
}
```

Generated Css

```
.class-A {
  padding : 5px;
  background : #000;
}

.class-B{
  padding : 8px;
}
```

Less Operations

Less

```
@base: 5%;
@filler: @base * 2;
@other: @base + @filler;
@base-color : #aaa;

#class {
  color: #888 / 4;
  background-color: @base-color + #111;
  height: 100% / 2 + @filler;
  width : @other + 50%;
}
```

Generated Css

```
#class {
  color: #222222;
  background-color: #bbbbbb;
  height: 60%;
  width: 65%;
}
```

Less Functions & Loops

Less

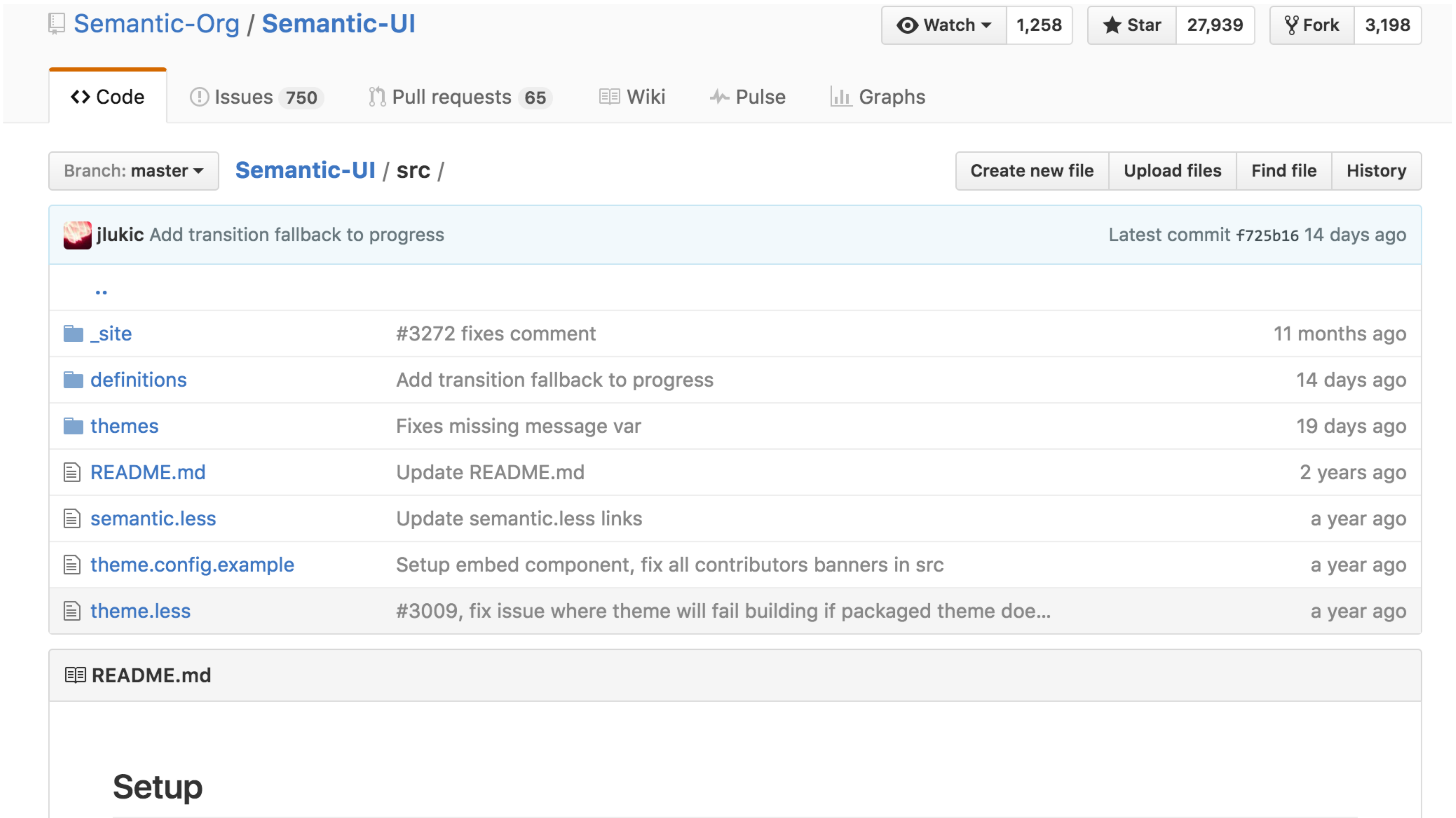
```
.generate-columns(@n, @i: 1) when (@i =< @n) {
  .column-@{i} {
    width: (@i * 100% / @n);
  }
  .generate-columns(@n, (@i + 1));
}
.generate-columns(2);
```

```
.column-1 {
  width: 50%;
}
.column-2 {
  width: 100%;
}
```

Generated Css

Semantic UI Sources

- Written in Less



Semantic-Org / Semantic-UI

Semantic-UI / src /

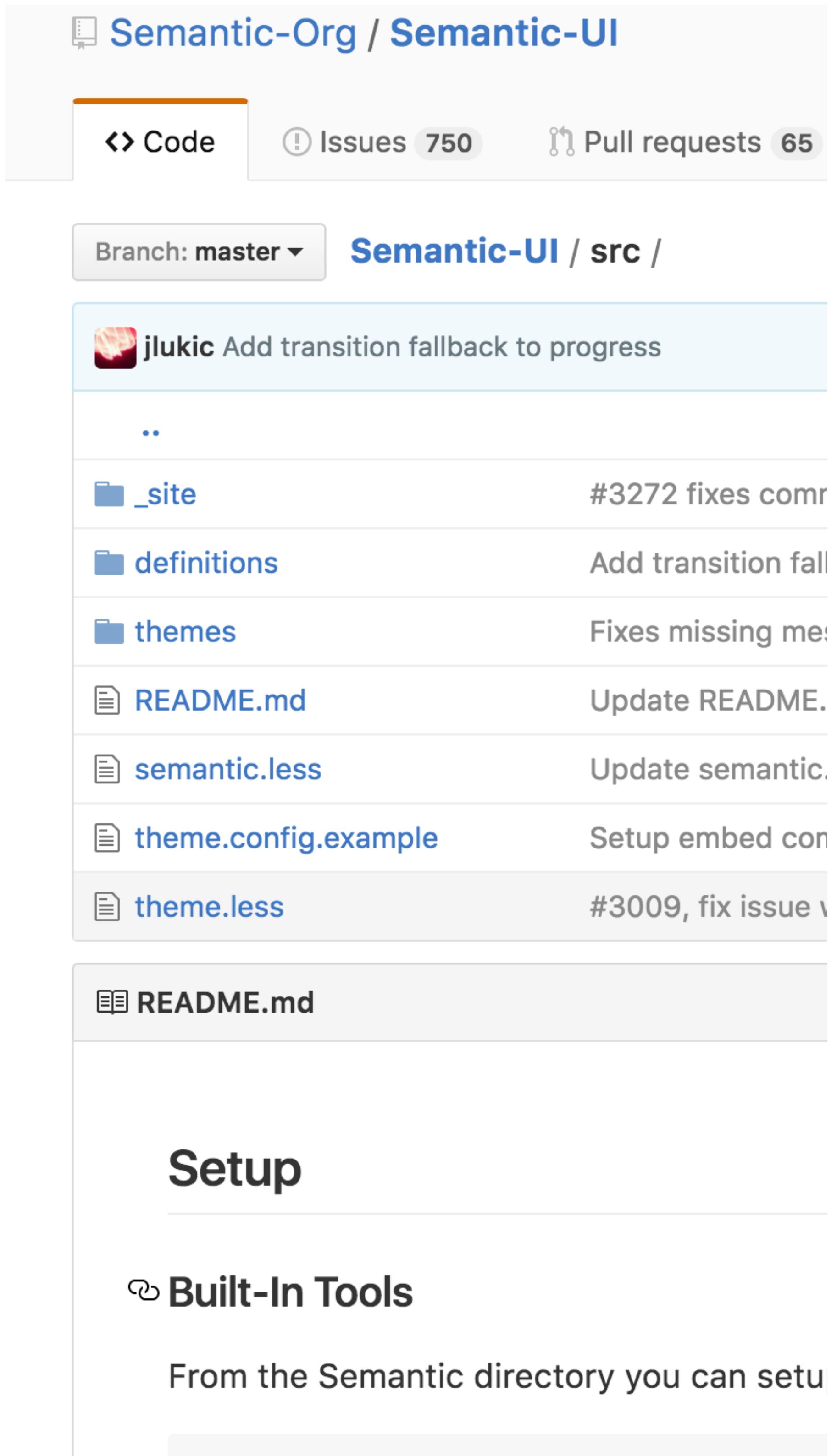
jlukic Add transition fallback to progress — Latest commit f725b16 14 days ago

..		
_site	#3272 fixes comment	11 months ago
definitions	Add transition fallback to progress	14 days ago
themes	Fixes missing message var	19 days ago
README.md	Update README.md	2 years ago
semantic.less	Update semantic.less links	a year ago
theme.config.example	Setup embed component, fix all contributors banners in src	a year ago
theme.less	#3009, fix issue where theme will fail building if packaged theme doe...	a year ago

README.md

Setup

Source Structure


Semantic-Org / Semantic-UI

Code | Issues 750 | Pull requests 65

Branch: master ▾ Semantic-UI / src /

jlukic Add transition fallback to progress

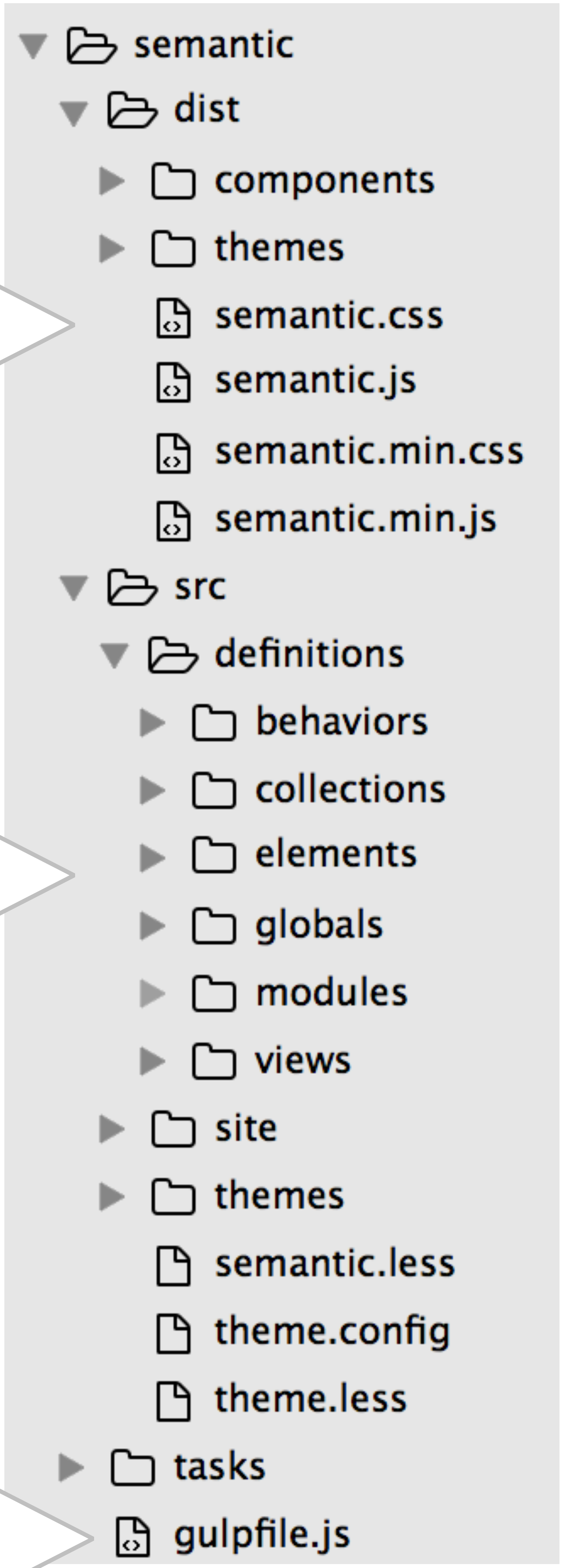
..	
_site	#3272 fixes comr
definitions	Add transition fall
themes	Fixes missing me:
README.md	Update README.
semantic.less	Update semantic.
theme.config.example	Setup embed con
theme.less	#3009, fix issue v

README.md

Setup

Built-In Tools

From the Semantic directory you can setu


```
semantic
  dist
    components
    themes
    semantic.css
    semantic.js
    semantic.min.css
    semantic.min.js
  src
    definitions
      behaviors
      collections
      elements
      globals
      modules
      views
    site
    themes
    semantic.less
    theme.config
    theme.less
  tasks
  gulpfile.js
```

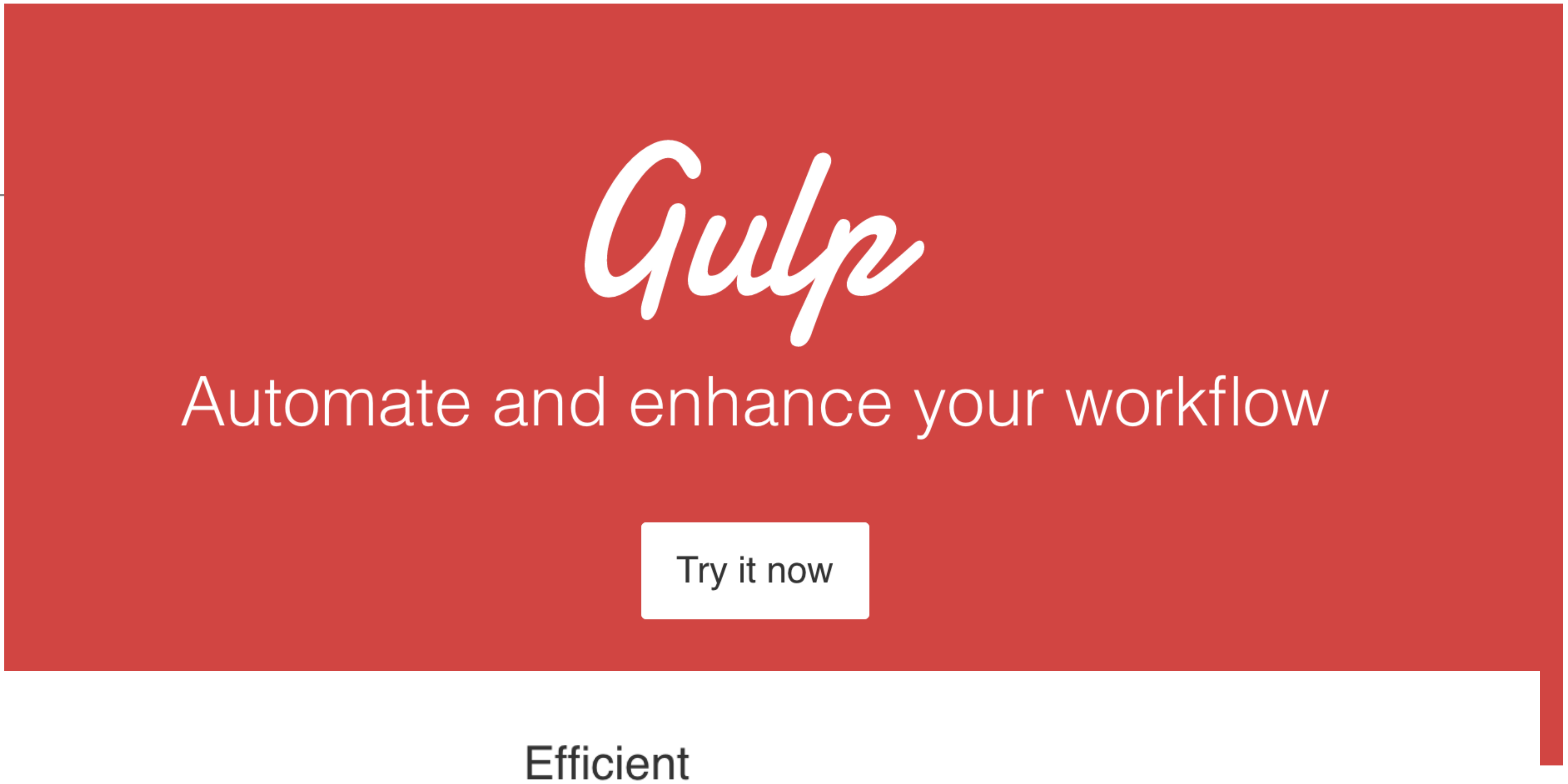
Generated css/js

Sources

Build script

Gulp

- A task runner/ build tool for front end developers



Easy to use

By preferring code over configuration, gulp keeps things simple and makes complex tasks manageable.

Efficient

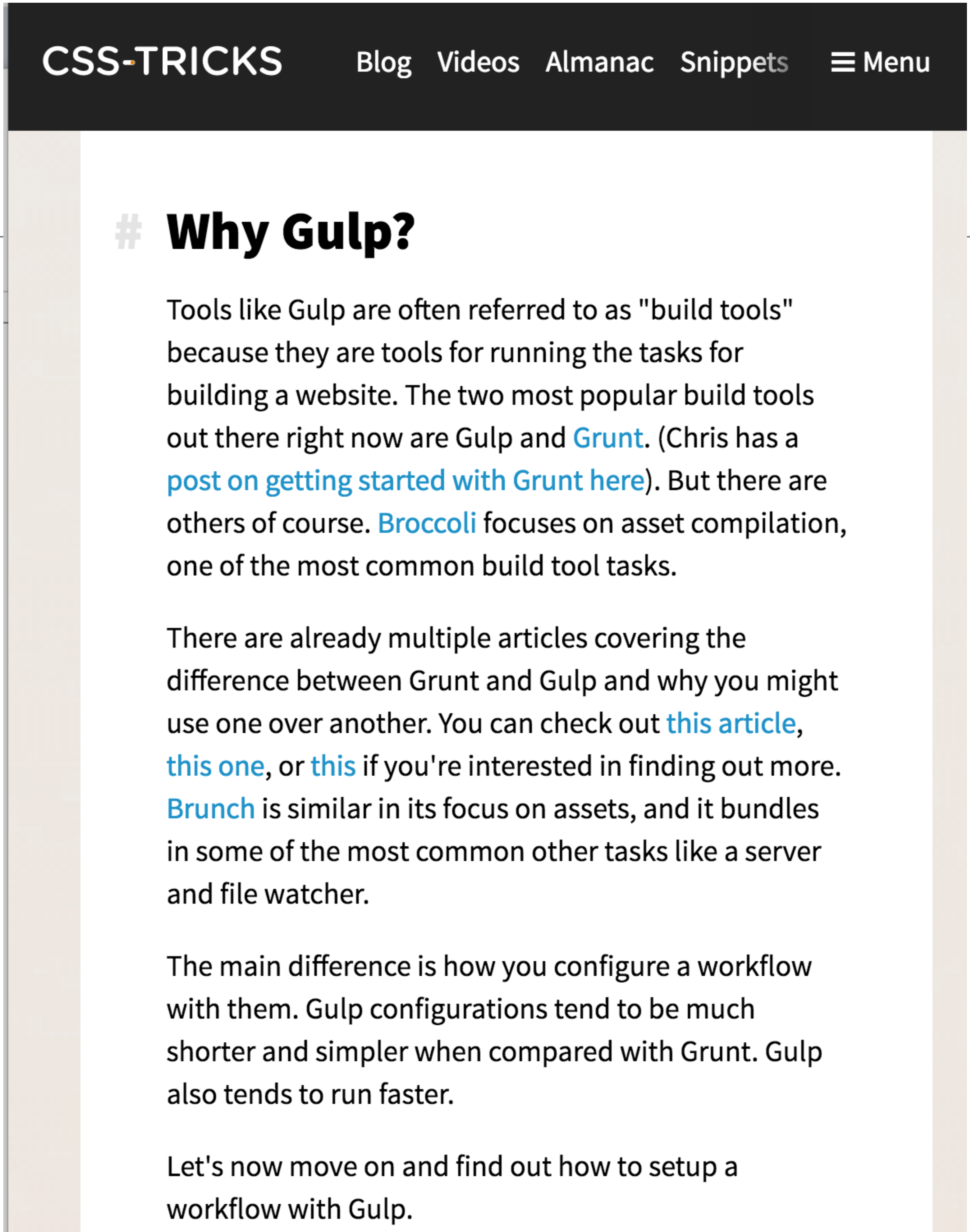
Using the power of node streams, gulp gives you fast builds that don't write intermediary files to disk.

High Quality

By enforcing strict plugin guidelines, we ensure that plugins stay simple and work as expected.

Easy to Learn

Using node best practices and maintaining a minimal API surface, your build works exactly as you would imagine.



Why Gulp?

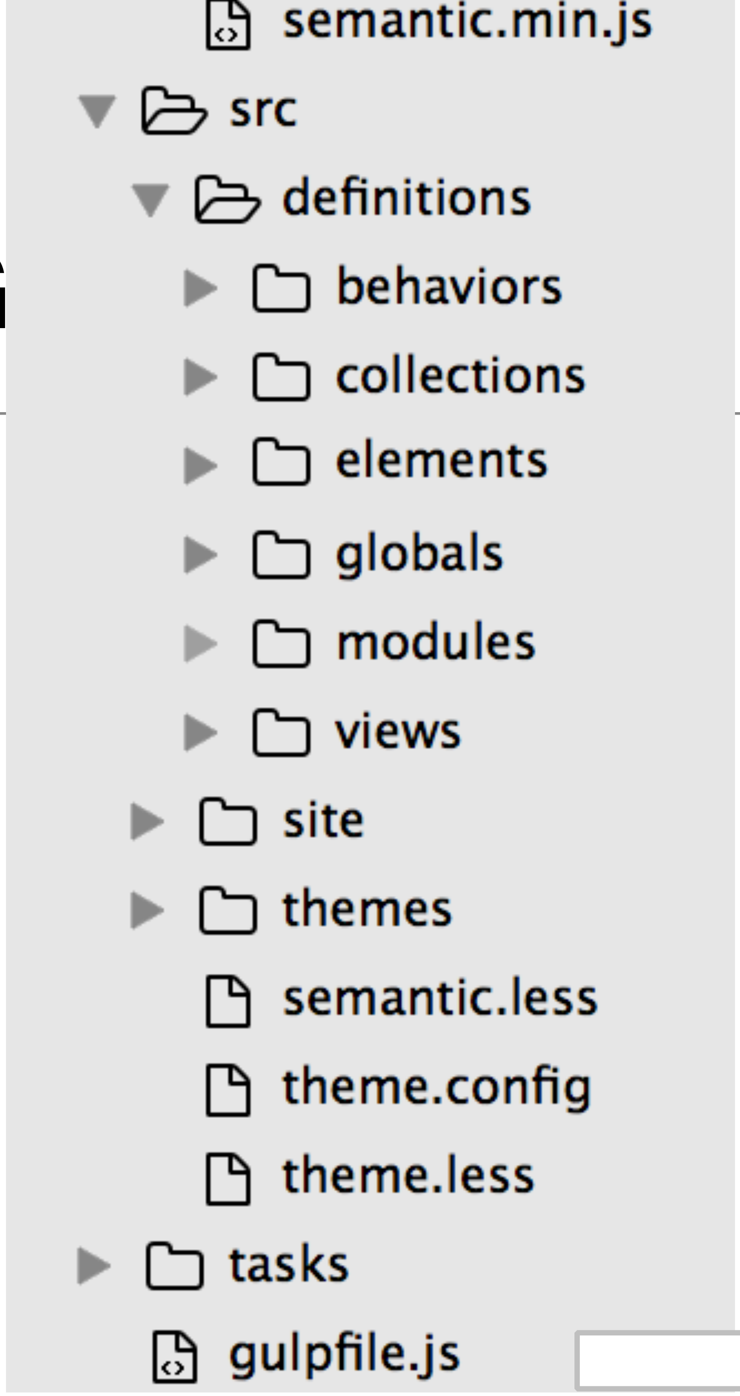
Tools like Gulp are often referred to as "build tools" because they are tools for running the tasks for building a website. The two most popular build tools out there right now are Gulp and Grunt. (Chris has a post on getting started with Grunt here). But there are others of course. Broccoli focuses on asset compilation, one of the most common build tool tasks.

There are already multiple articles covering the difference between Grunt and Gulp and why you might use one over another. You can check out this article, this one, or this if you're interested in finding out more. Brunch is similar in its focus on assets, and it bundles in some of the most common other tasks like a server and file watcher.

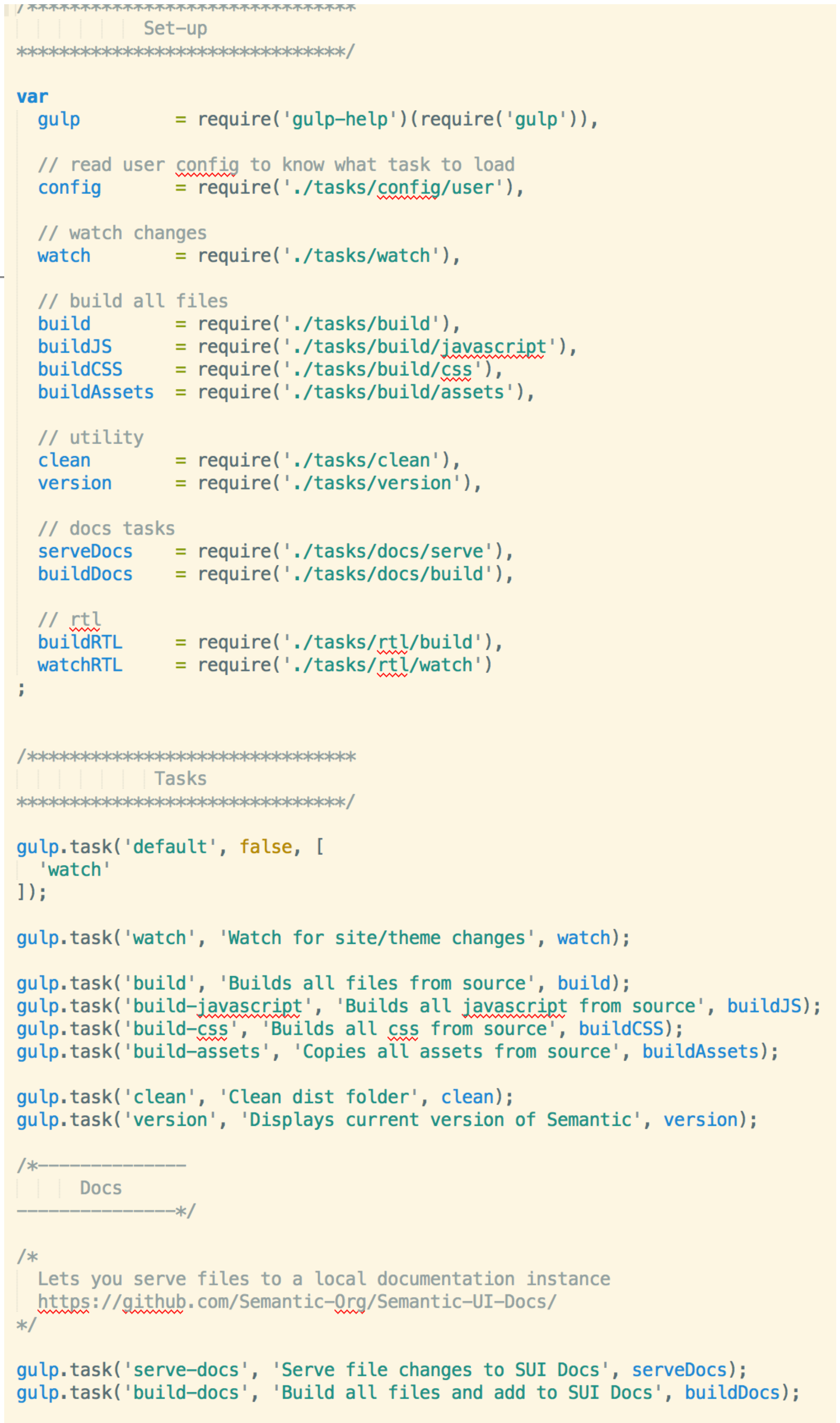
The main difference is how you configure a workflow with them. Gulp configurations tend to be much shorter and simpler when compared with Grunt. Gulp also tends to run faster.

Let's now move on and find out how to setup a workflow with Gulp.

G



```
    semantic.min.js
▼ src
  ▼ definitions
    ▶ behaviors
    ▶ collections
    ▶ elements
    ▶ globals
    ▶ modules
    ▶ views
  ▶ site
  ▶ themes
    semantic.less
    theme.config
    theme.less
▶ tasks
  gulpfile.js
```



```
/*******************************
         Set-up
*******************************/

var
  gulp         = require('gulp-help')(require('gulp')),

  // read user config to know what task to load
  config       = require('./tasks/config/user'),

  // watch changes
  watch        = require('./tasks/watch'),

  // build all files
  build        = require('./tasks/build'),
  buildJS      = require('./tasks/build/javascript'),
  buildCSS     = require('./tasks/build/css'),
  buildAssets  = require('./tasks/build/assets'),

  // utility
  clean        = require('./tasks/clean'),
  version      = require('./tasks/version'),

  // docs tasks
  serveDocs    = require('./tasks/docs/serve'),
  buildDocs    = require('./tasks/docs/build'),

  // rtl
  buildRTL     = require('./tasks/rtl/build'),
  watchRTL     = require('./tasks/rtl/watch')
;


/*******************************
          Tasks
*******************************/

gulp.task('default', false, [
  'watch'
]);

gulp.task('watch', 'Watch for site/theme changes', watch);

gulp.task('build', 'Builds all files from source', build);
gulp.task('build-javascript', 'Builds all javascript from source', buildJS);
gulp.task('build-css', 'Builds all css from source', buildCSS);
gulp.task('build-assets', 'Copies all assets from source', buildAssets);

gulp.task('clean', 'Clean dist folder', clean);
gulp.task('version', 'Displays current version of Semantic', version);

/*--------------
      Docs
---------------*/

/*
  Lets you serve files to a local documentation instance
  https://github.com/Semantic-Org/Semantic-UI-Docs/
*/

gulp.task('serve-docs', 'Serve file changes to SUI Docs', serveDocs);
gulp.task('build-docs', 'Build all files and add to SUI Docs', buildDocs);
```

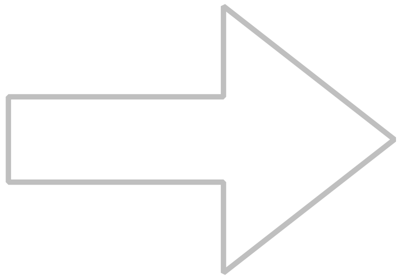
- Gulp script is Javascript!

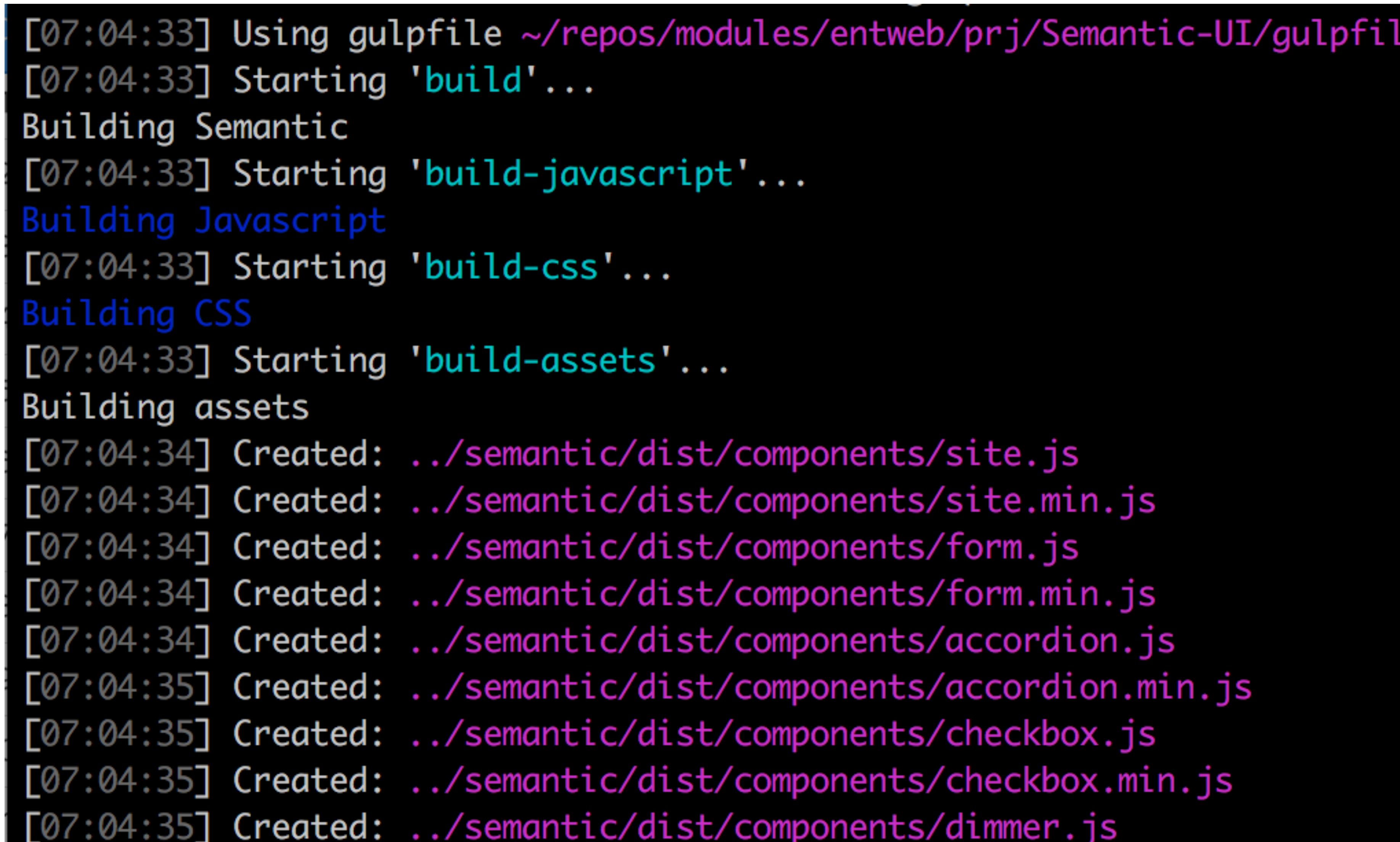
- List files to be compiled/ transformed + suitable steps

Building Semantic UI

```
$ npm install gulp -g
$ git clone https://github.com/Semantic-Org/Semantic-UI.git
$ cd Semantic-UI
$ gulp build
```

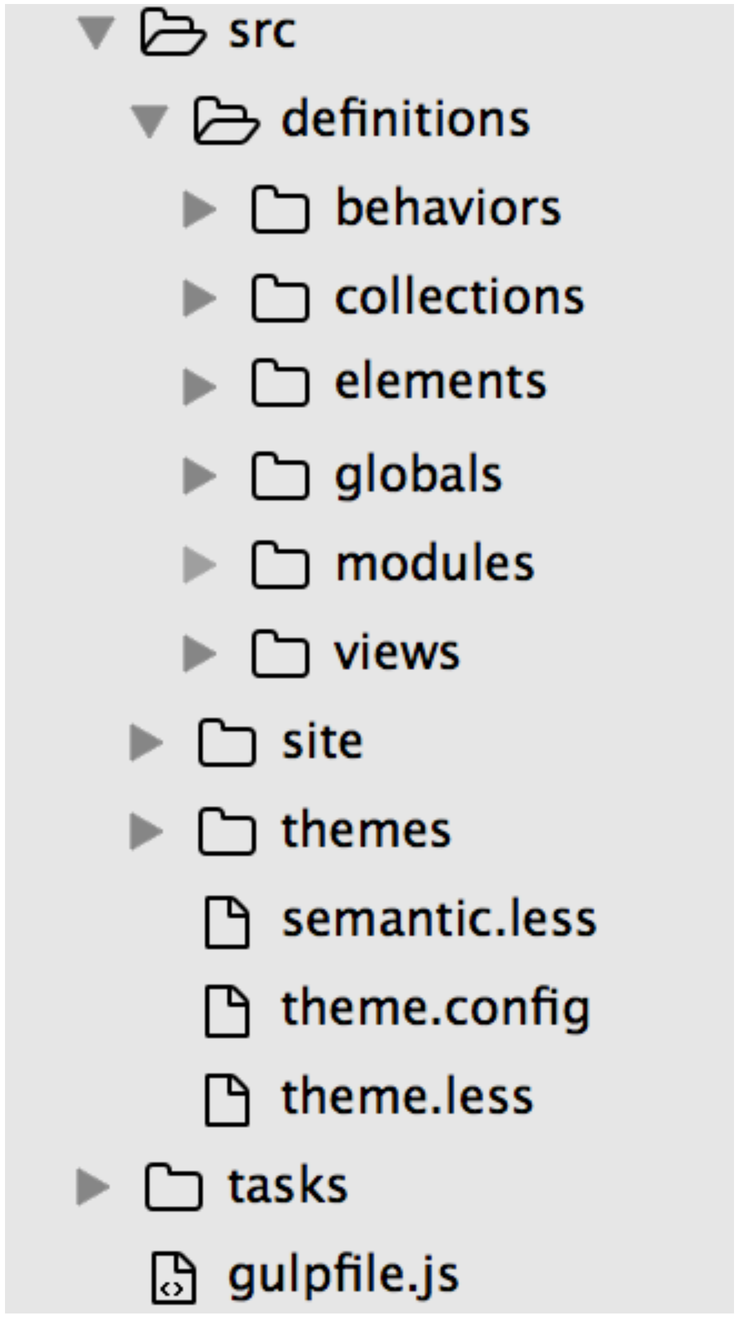
- gulp compiles the less + js sources to `dist` folder ready for use



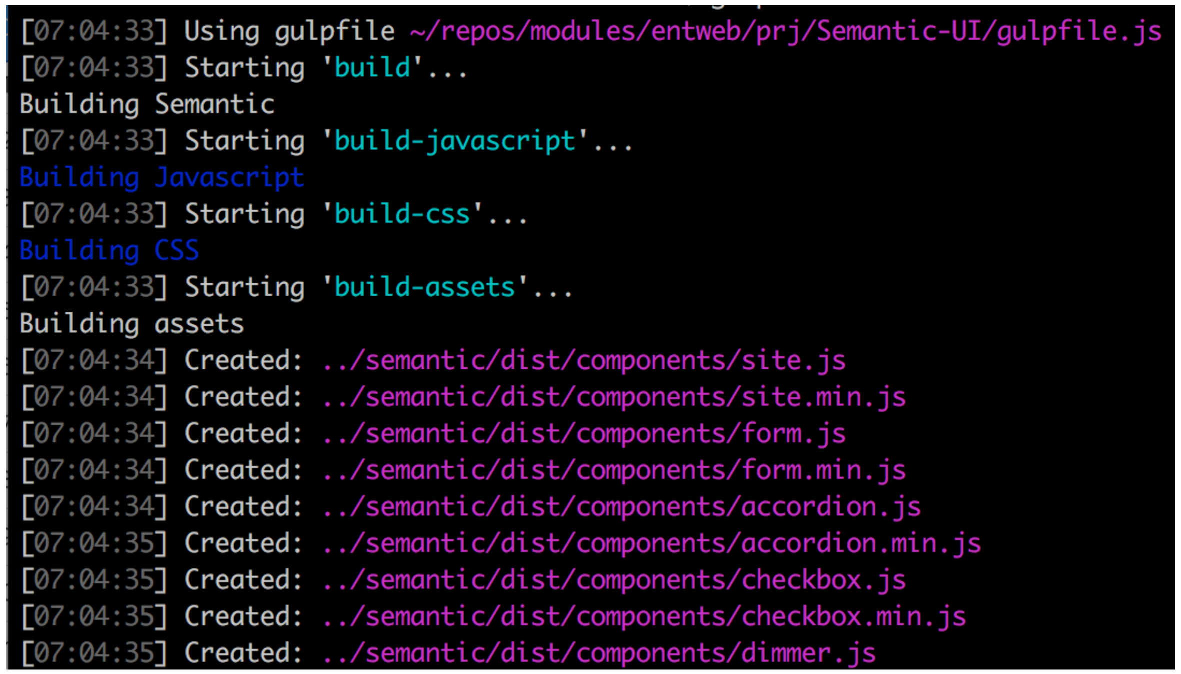


```
[07:04:33] Using gulpfile ~/repos/modules/entweb/prj/Semantic-UI/gulpfil
[07:04:33] Starting 'build'...
Building Semantic
[07:04:33] Starting 'build-javascript'...
Building Javascript
[07:04:33] Starting 'build-css'...
Building CSS
[07:04:33] Starting 'build-assets'...
Building assets
[07:04:34] Created: ../semantic/dist/components/site.js
[07:04:34] Created: ../semantic/dist/components/site.min.js
[07:04:34] Created: ../semantic/dist/components/form.js
[07:04:34] Created: ../semantic/dist/components/form.min.js
[07:04:34] Created: ../semantic/dist/components/accordion.js
[07:04:35] Created: ../semantic/dist/components/accordion.min.js
[07:04:35] Created: ../semantic/dist/components/checkbox.js
[07:04:35] Created: ../semantic/dist/components/checkbox.min.js
[07:04:35] Created: ../semantic/dist/components/dimmer.js
```

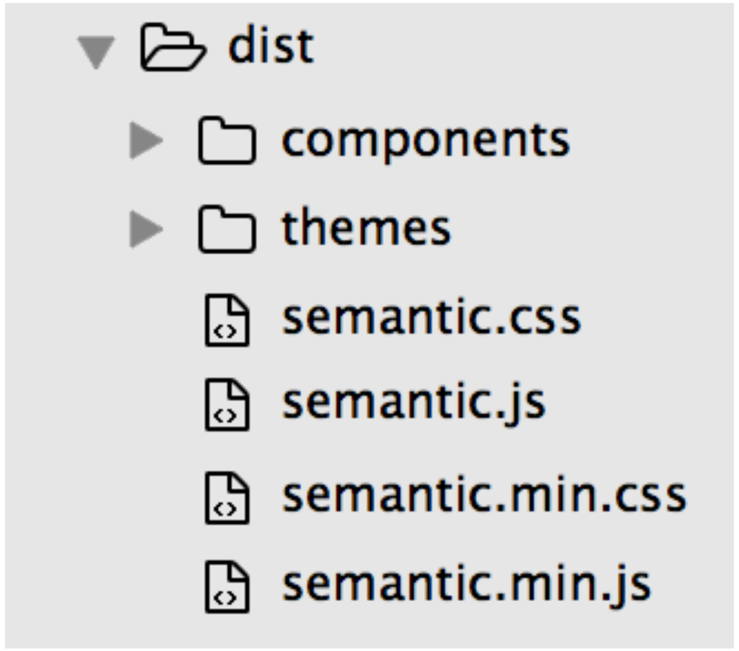
Gulp Build …



```
▼ src
  ▼ definitions
    ▶ behaviors
    ▶ collections
    ▶ elements
    ▶ globals
    ▶ modules
    ▶ views
  ▶ site
  ▶ themes
    semantic.less
    theme.config
    theme.less
▶ tasks
  gulpfile.js
```

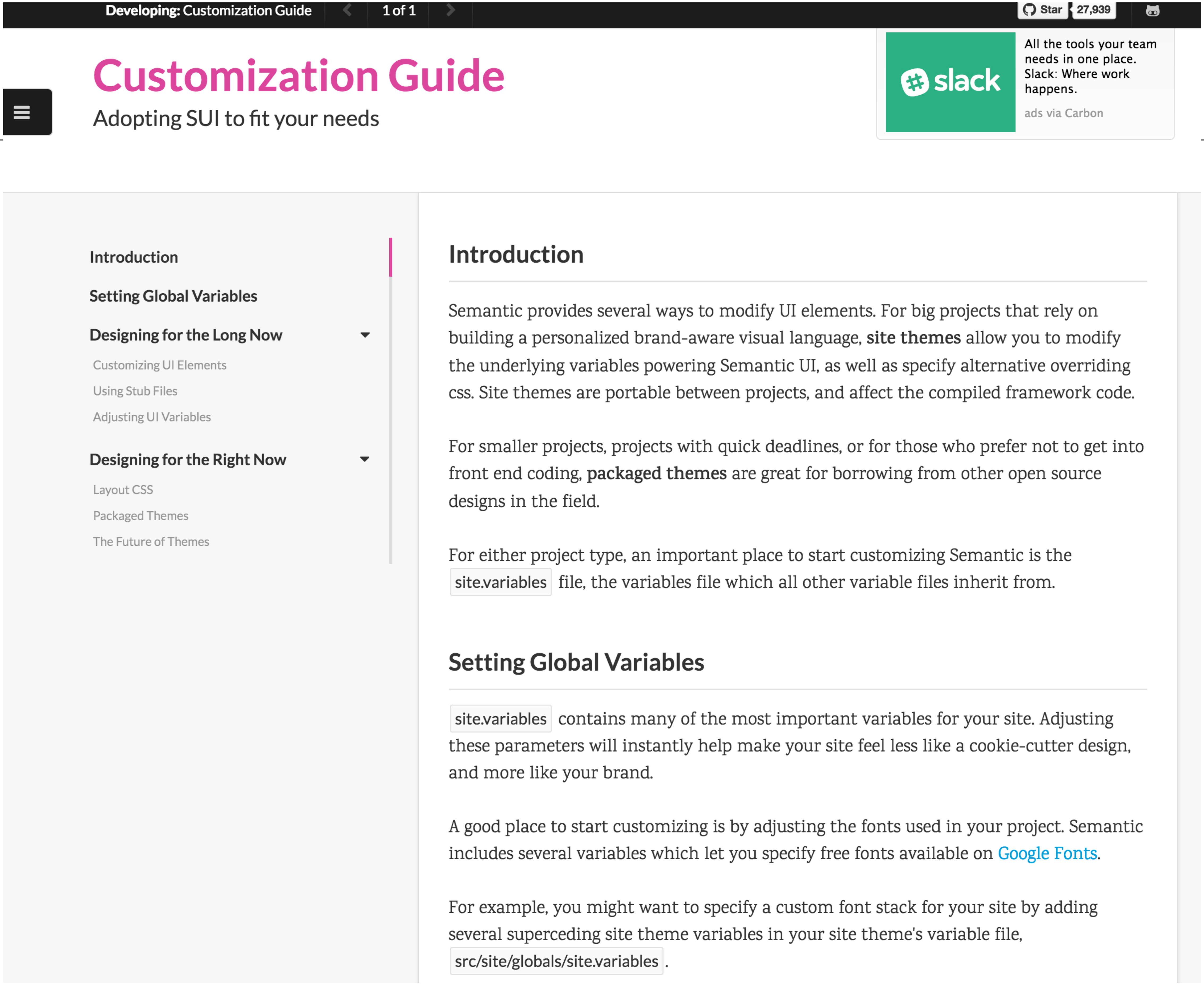


```
[07:04:33] Using gulpfile ~/repos/modules/entweb/prj/Semantic-UI/gulpfile.js
[07:04:33] Starting 'build'...
Building Semantic
[07:04:33] Starting 'build-javascript'...
Building Javascript
[07:04:33] Starting 'build-css'...
Building CSS
[07:04:33] Starting 'build-assets'...
Building assets
[07:04:34] Created: ../semantic/dist/components/site.js
[07:04:34] Created: ../semantic/dist/components/site.min.js
[07:04:34] Created: ../semantic/dist/components/form.js
[07:04:34] Created: ../semantic/dist/components/form.min.js
[07:04:34] Created: ../semantic/dist/components/accordion.js
[07:04:35] Created: ../semantic/dist/components/accordion.min.js
[07:04:35] Created: ../semantic/dist/components/checkbox.js
[07:04:35] Created: ../semantic/dist/components/checkbox.min.js
[07:04:35] Created: ../semantic/dist/components/dimmer.js
```



```
▼ dist
  ▶ components
  ▶ themes
    semantic.css
    semantic.js
    semantic.min.css
    semantic.min.js
```

Customising Semantic UI


Developing: Customization Guide
1 of 1
Star
27,939

Customization Guide

Adopting SUI to fit your needs

slack
All the tools your team needs in one place. Slack: Where work happens.
ads via Carbon

Introduction

Semantic provides several ways to modify UI elements. For big projects that rely on building a personalized brand-aware visual language, **site themes** allow you to modify the underlying variables powering Semantic UI, as well as specify alternative overriding css. Site themes are portable between projects, and affect the compiled framework code.

For smaller projects, projects with quick deadlines, or for those who prefer not to get into front end coding, **packaged themes** are great for borrowing from other open source designs in the field.

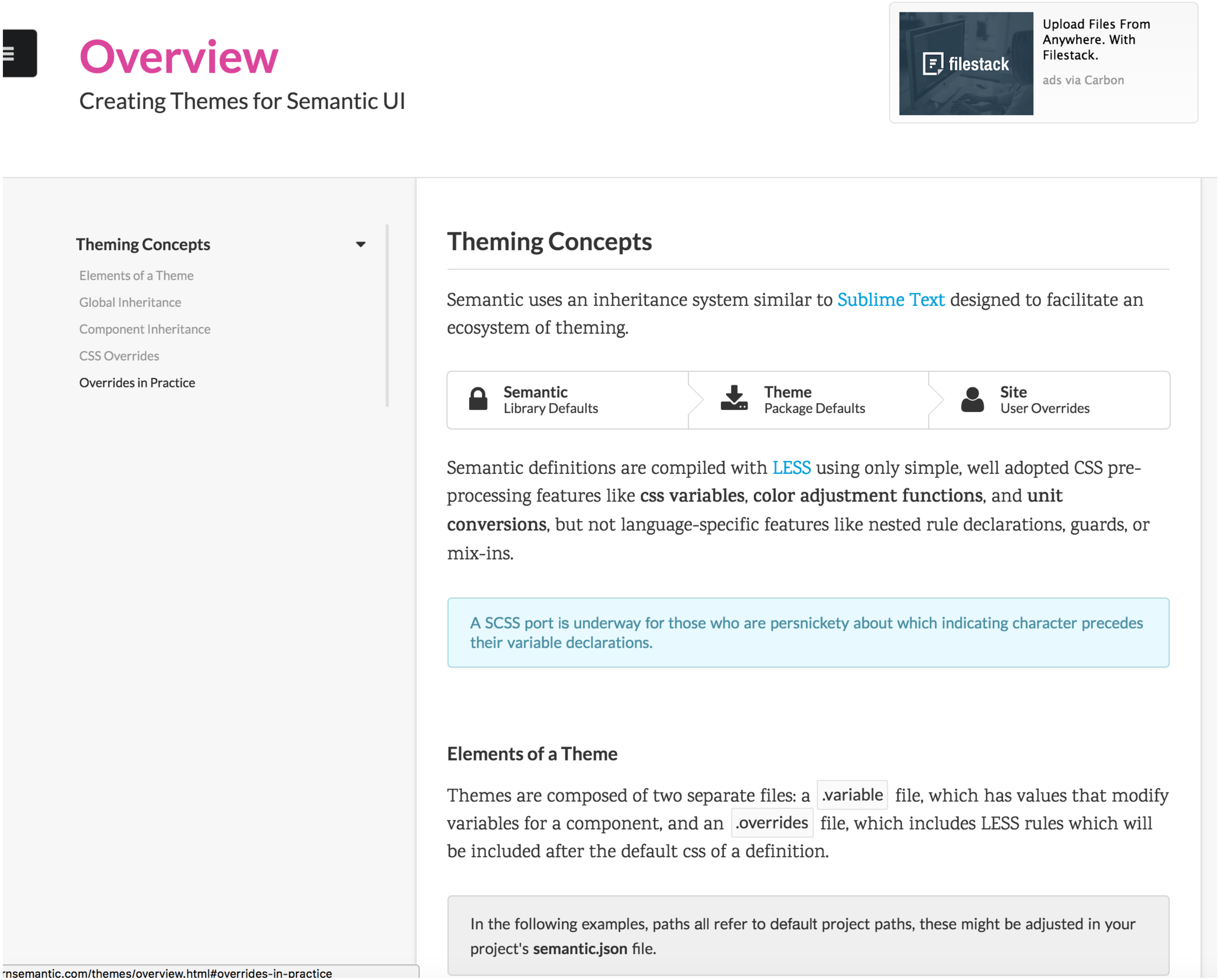
For either project type, an important place to start customizing Semantic is the `site.variables` file, the variables file which all other variable files inherit from.

Setting Global Variables

`site.variables` contains many of the most important variables for your site. Adjusting these parameters will instantly help make your site feel less like a cookie-cutter design, and more like your brand.

A good place to start customizing is by adjusting the fonts used in your project. Semantic includes several variables which let you specify free fonts available on Google Fonts.

For example, you might want to specify a custom font stack for your site by adding several superceding site theme variables in your site theme's variable file, `src/site/globals/site.variables`.



Overview

Creating Themes for Semantic UI

Upload Files From Anywhere. With Filestack.

ads via Carbon

Theming Concepts

Semantic uses an inheritance system similar to Sublime Text designed to facilitate an ecosystem of theming.

Semantic	Theme	Site
Library Defaults	Package Defaults	User Overrides

Semantic definitions are compiled with LESS using only simple, well adopted CSS pre-processing features like **css variables**, **color adjustment functions**, and **unit conversions**, but not language-specific features like nested rule declarations, guards, or mix-ins.

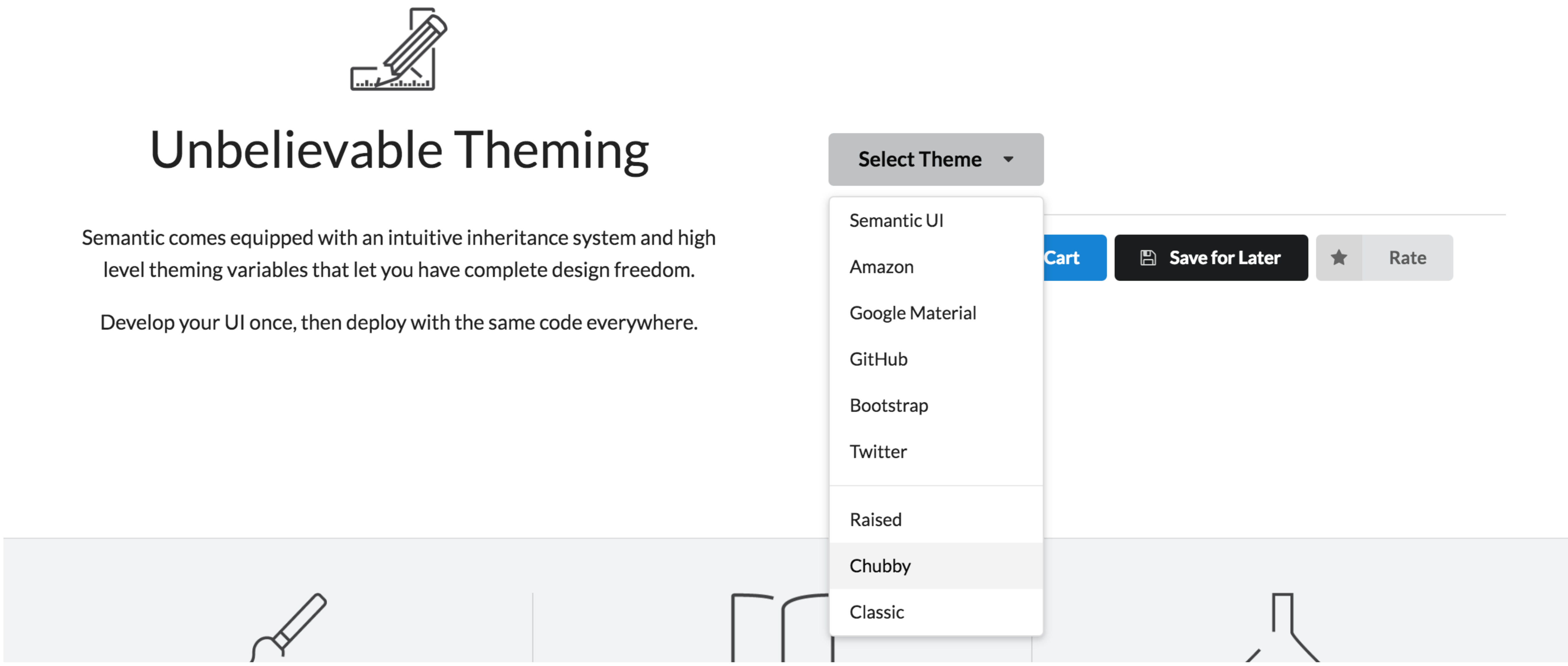
A SCSS port is underway for those who are persnickety about which indicating character precedes their variable declarations.

Elements of a Theme

Themes are composed of two separate files: a `.variable` file, which has values that modify variables for a component, and an `.overrides` file, which includes LESS rules which will be included after the default css of a definition.

In the following examples, paths all refer to default project paths, these might be adjusted in your project's **semantic.json** file.

rnsemantic.com/themes/overview.html#overrides-in-practice



Unbelievable Theming

Semantic comes equipped with an intuitive inheritance system and high level theming variables that let you have complete design freedom.

Develop your UI once, then deploy with the same code everywhere.