Who am I?

Nick Janetakis

nickjanetakis.com

Consultant by day

Open source by night

@nickjanetakis

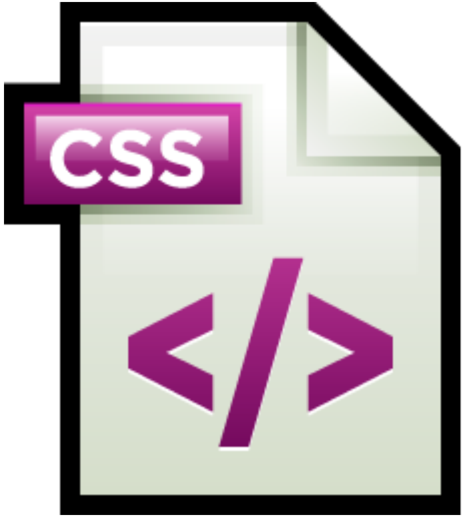
GitHub: nickjj

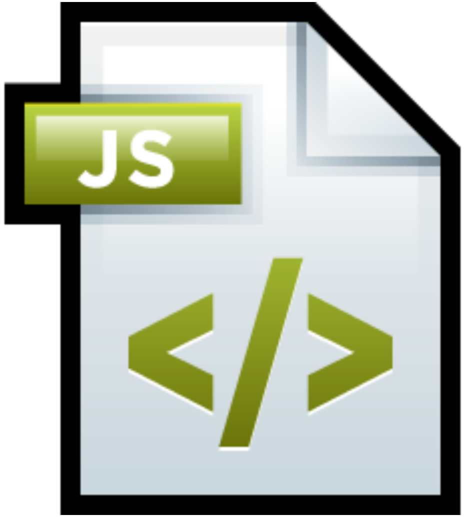
Untangle the mess of asset management

Combine Webpack with Flask-Webpack

What is an asset?

- Static file such as an image, CSS or JS





6 asset management steps

1. Minify and concatenate files

2. Pre-process HTML, CSS and Javascript

3. Attach vendor prefixes to all CSS

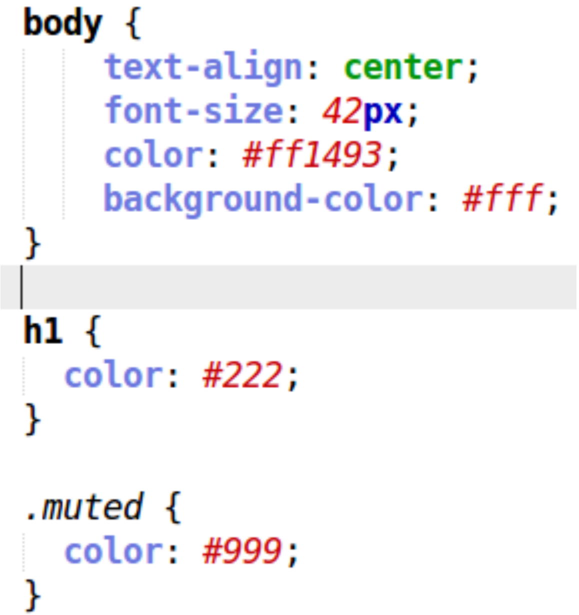
6 asset management steps

4. Optimize images (lossless compression)

5. Deal with module dependencies

6. Automate all of this with Webpack

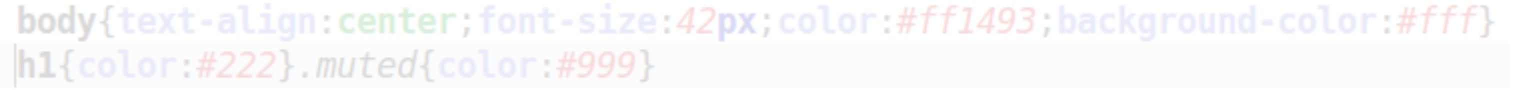
1. Minify and concatenate files



```
body {
    text-align: center;
    font-size: 42px;
    color: #ff1493;
    background-color: #fff;
}

h1 {
  color: #222;
}

.muted {
  color: #999;
}
```

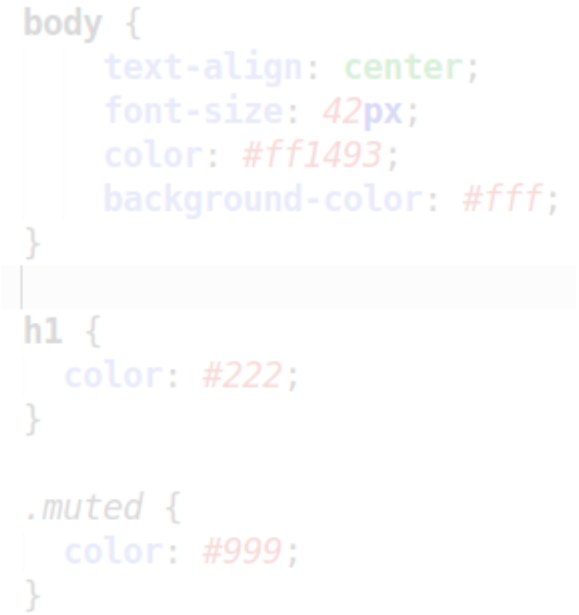
Before:

151 bytes



```
body{text-align:center;font-size:42px;color:#ff1493;background-color:#fff}
h1{color:#222}.muted{color:#999}
```

1. Minify and concatenate files

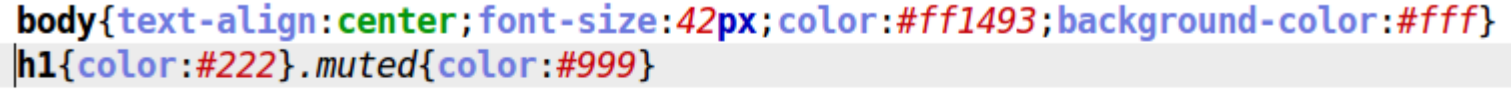


```
body {
    text-align: center;
    font-size: 42px;
    color: #ff1493;
    background-color: #fff;
}

h1 {
  color: #222;
}

.muted {
  color: #999;
}
```

After:

107 bytes

~50% smaller



```
body{text-align:center;font-size:42px;color:#ff1493;background-color:#fff}
h1{color:#222}.muted{color:#999}
```

1. **Minify and concatenate files**

- 200-300% reduction in real apps
- Less files served over the wire
- Faster page loads for users

2. Pre-process HTML, CSS and JS

- Markdown -> HTML
- LESS, SASS and Stylus -> CSS
- Typescript, Coffeescript, and ES6 -> JS
- React JSX templates -> JS

2. Pre-process HTML, CSS and JS

- Markdown -> HTML
- LESS, SASS and Stylus -> CSS
- Typescript, Coffeescript, and ES6 -> JS
- React JSX templates -> JS

2. Pre-process HTML, CSS and JS

- Markdown -> HTML
- LESS, SASS and Stylus -> CSS
- Typescript, Coffeescript, and ES6 -> JS
- React JSX templates -> JS

2. Pre-process HTML, CSS and JS

- Markdown -> HTML

- LESS, SASS and Stylus -> CSS

- Typescript, Coffeescript, and ES6 -> JS

- React JSX templates -> JS

3. Attach vendor prefixes to all CSS

Before: (15 LINES)

```
.a {
  display: -webkit-box;
  display: -webkit-flex;
  display: -ms-flexbox;
  display: flex;
  background: -webkit-linear-gradient(#fff, #000);
  background: linear-gradient(#fff, #000);
  -webkit-column-count: 2;
     -moz-column-count: 2;
          column-count: 2;
  -webkit-column-gap: 10px;
     -moz-column-gap: 10px;
          column-gap: 10px;
  -webkit-column-rule: 1px solid #000;
     -moz-column-rule: 1px solid #000;
          column-rule: 1px solid #000;
}
```

Zoomed in:

```
display: -webkit-box;
display: -webkit-flex;
display: -ms-flexbox;
display: flex;

-webkit-column-count: 2;
   -moz-column-count: 2;
        column-count: 2;
```

3. Attach vendor prefixes to all CSS

After: (5 LINES)

```css
.a {
  display: flex;
  background: linear-gradient(#fff, #000);
  column-count: 2;
  column-gap: 10px;
  column-rule: 1px solid #000;
}
```

3. Attach vendor prefixes to all CSS

- Never duplicate vendor prefixes again
- Works with plain old CSS
- Compiles once at build time, so it’s fast

4. Optimize images

- Create smaller images
- Avoid ugly compression artifacts
- Serve less bandwidth and save money

5. Deal with module dependencies

Define a module `App.js`:

```
import React, { Component } from 'react';

export default class App extends Component {
  render() {
    return (
      <h1>Rendered with React</h1>
    );
  }
}
```

5. Deal with module dependencies

Import it in another file `index.js`:

```
import React from 'react';
import App from './App';
```

- Browsers cannot understand this yet
- Webpack converts it so they can

5. Deal with module dependencies

- Write modular code similar to Python

- Future proofed because it is ES6

- Compiles once at build time,
  so it’s fast

6. Automate all of this with Webpack

- What is Webpack?
- How do I make it work with Flask?
- How do I make it work anywhere?

What is Webpack?

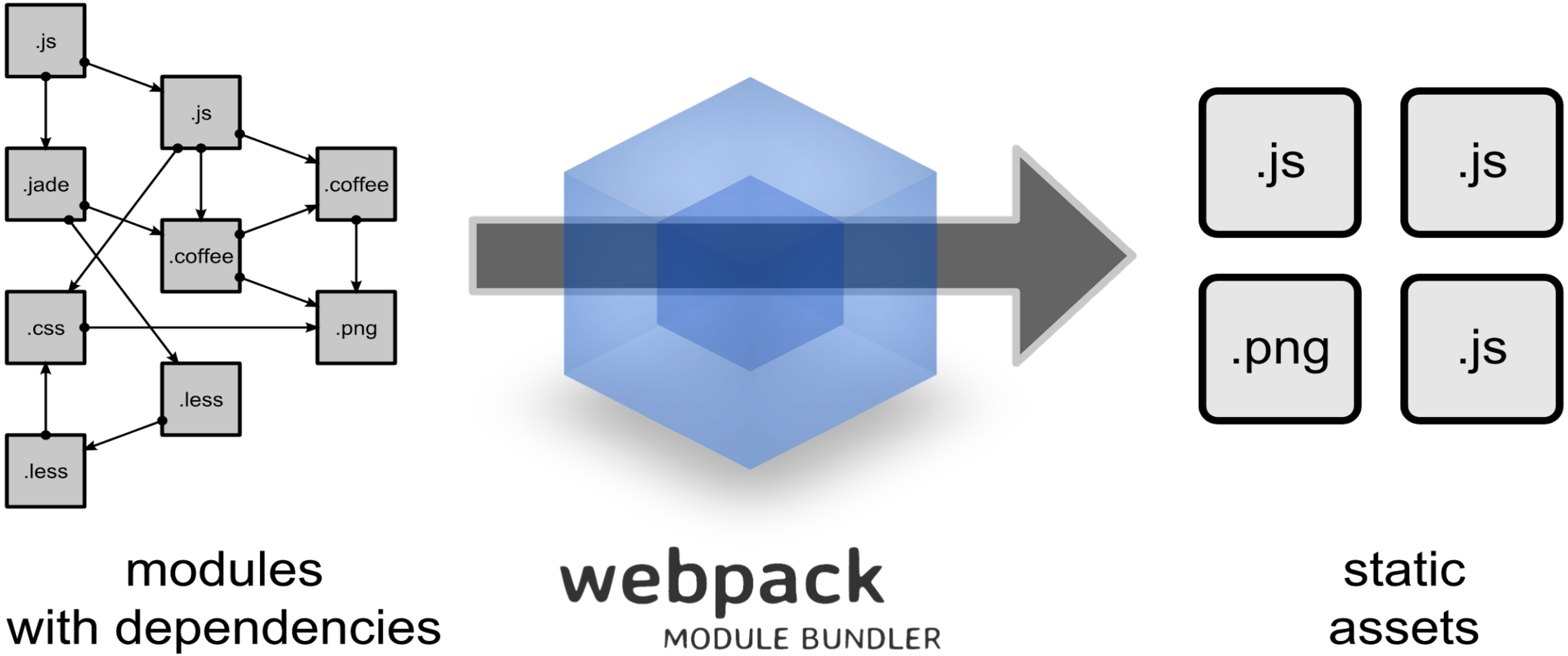


image supplied by webpack

How do I make it work with Flask?

- Auto-generated `manifest.json` file

- Global Jinja2 (template) tags

Manifest.json

```
{
  "assets": {
    "images/cats.jpg": "images/cats.b9252d.jpg",
    "app_js.js": "app_js.8b7c0d.js",
    "app_css.css": "app_css.d5cbf1.css"
  },
  "publicPath": "http://localhost:2992/assets/"
}
```

Manifest.json

```
{
  "assets": {
    "images/cats.jpg": "images/cats.b9252d.jpg",
    "app_js.js": "app_js.8b7c0d.js",
    "app_css.css": "app_css.d5cbf1.css"
  },
  "publicPath": "http://localhost:2992/assets/"
}
```

Manifest.json

```
{
  "assets": {
    "images/cats.jpg": "images/cats.b9252d.jpg",
    "app_js.js": "app_js.8b7c0d.js",
    "app_css.css": "app_css.d5cbf1.css"
  },
  "publicPath": "http://localhost:2992/assets/"
}
```

Manifest.json

```
{
  "assets": {
    "images/cats.jpg": "images/cats.b9252d.jpg",
    "app_js.js": "app_js.8b7c0d.js",
    "app_css.css": "app_css.d5cbf1.css"
  },
  "publicPath": "http://localhost:2992/assets/"
}
```

Manifest.json

```
{
  "assets": {
    "images/cats.jpg": "images/cats.b9252d.jpg",
    "app_js.js": "app_js.8b7c0d.js",
    "app_css.css": "app_css.d5cbf1.css"
  },
  "publicPath": "http://localhost:2992/assets/"
}
```

Global Jinja2 (template) tags

asset_url_for

```
<img src="{{ asset_url_for('images/cats.jpg') }}" />
```

After the template tag is compiled

```
<img src="http://localhost:2992/assets/images/cats.b9252d.jpg" />
```

Global Jinja2 (template) tags

stylesheet_tag / javascript_tag

```
<head>
    {{ stylesheet_tag('app_css') | safe }}
    {{ javascript_tag('app_js') | safe }}
</head>
```

After the template tags are compiled

```
<link rel="stylesheet" href="http://localhost:2992/assets/app_css.d5cbf1.css" />
<script src="http://localhost:2992/assets/app_js.8b7c0d.js" />
```

How do I make it work anywhere?

- ManifestRevisionPlugin (webpack plugin)
- Customize the `manifest.json` output
- Works with Flask and Ruby on Rails
- Custom format functions are possible

https://github.com/nickjj/demo-for-chattanooga-python-user-group

Build a SAAS app with Flask

Become a Flask master

- Users
- Recurring billing
- Websockets
- Third party APIs
- Custom admin
- Asset management
- Deployment
- General Flask tips

http://buildasaaswithflask.com

* successfully funded on Kickstarter