

The Front End Stack

As of 2019

A brief history of web tech

HTML, CSS, and JavaScript throughout the years

HTML (Hypertext Markup Language)

Plain ol' HTML files

Preprocessors (PHP, Jinja2, Handlebars)

HTML-in-JS (JSX, Angular, Vue.js)

CSS (Cascading Style Sheets)

CSS-in-HTML (Inline styles)

External CSS files

Preprocessors (Sass, Less, Stylus)

CSS-in-JS (styled-components, glamor, emotion)

JavaScript (the old parts)

One global scope

Namespacing

IIFE (Immediately Invoked Function Expression)

CommonJS modules (Node.js)

JavaScript (the new parts)

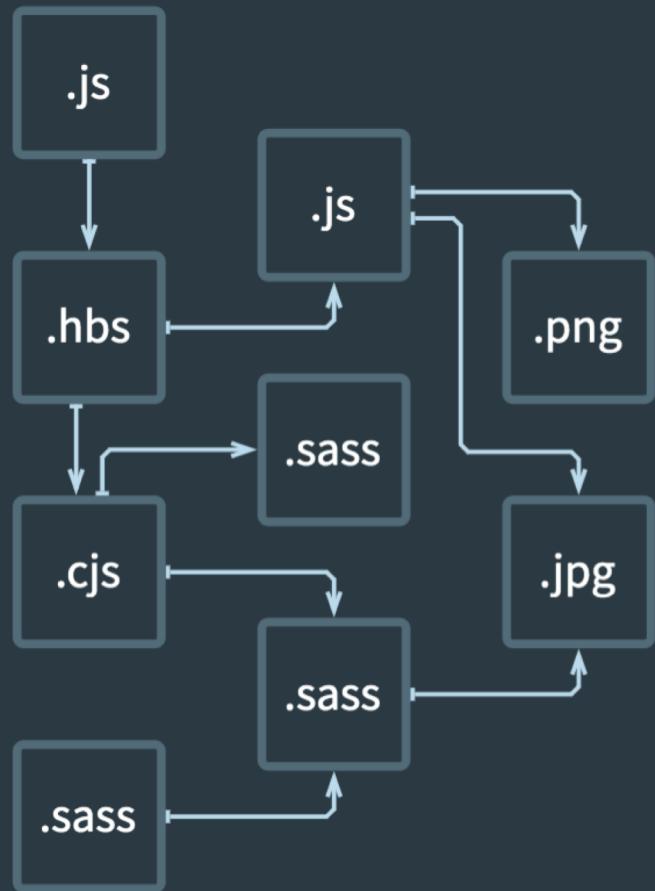
ES6 Modules

Dynamic imports

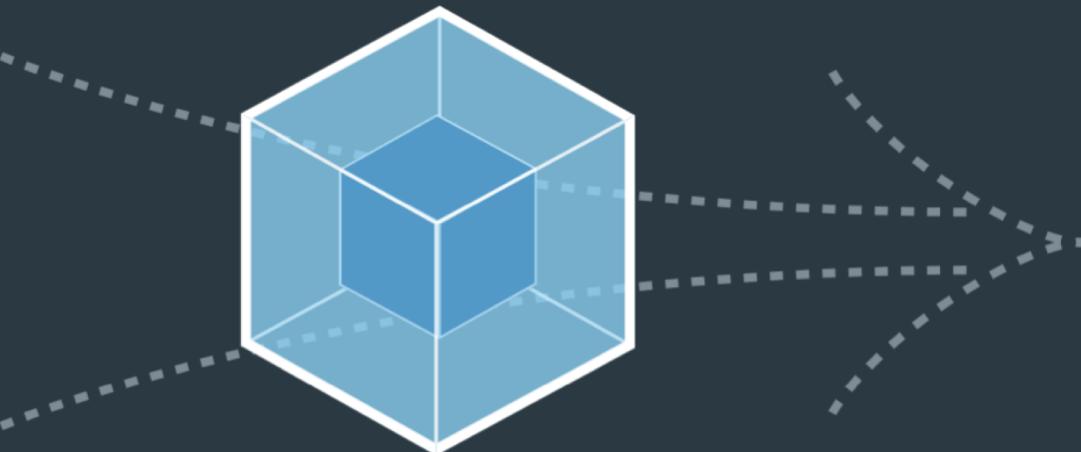
Entering the almighty *webpack*

webpack

Throwing stuff together (aka module bundler)



MODULES WITH DEPENDENCIES



STATIC ASSETS

Everything is a PLUGING



Mastering  **webpack**
from the **inside out**

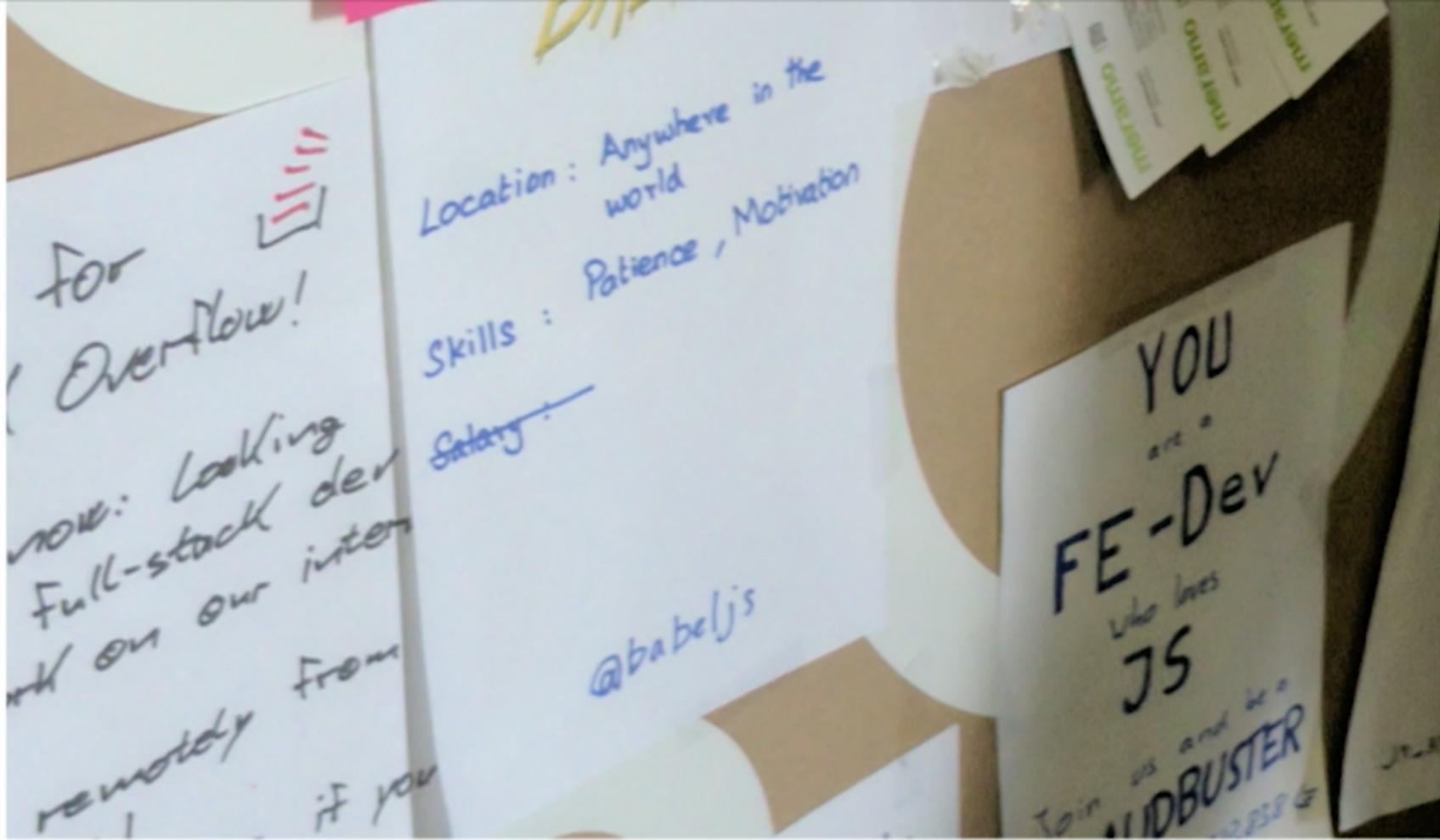


<https://www.youtube.com/watch?v=jFc898NTh5Q>

Babel

Transform something into usable JavaScript (aka compiler)

```
1 function foo () {           1 "use strict";
2   return 'foo'              2
3 }                         3 function _classCallCheck(instance, Constructor) {
4                           4   if (!(instance instanceof Constructor)) {
5 class Bar {                5     throw new TypeError("Cannot call a class as a function");
6   constructor () {          6   }
7     this.value = foo()      7   }
8   }                         8
9 }                         9 function foo() {
10                          10   return "foo";
11 new Bar()                 11 }
12                           12
13 var Bar = function Bar() { 13
14   _classCallCheck(this, Bar); 14
15                           15
16   this.value = foo();       16
17 };                      17
18                           18
19 new Bar();                 19
20                           20
```



So how does Babel even work?

Henry Zhu

<https://www.youtube.com/watch?v=fntd0sPMOrQ>

Vue.js

Getting fancy with Single File Components

The image shows a screenshot of a code editor window titled "Hello.vue". The window has a dark theme with light-colored text. The code is a Vue.js component with three main sections: template, script, and style.

```
<template>
  <p>{{ greeting }} World!</p>
</template>

<script>
module.exports = {
  data: function () {
    return {
      greeting: 'Hello'
    }
  }
}
</script>

<style scoped>
p {
  font-size: 2em;
  text-align: center;
}
</style>
```

At the bottom of the editor, there are status indicators: "Line 21, Column 1", "Spaces: 2", and "Vue Component".

Tree Shaking

Dead-code elimination 

Tree Shaking

- Relies on static “import” statements
- Smaller overall production code
- Only bundle stuff that’s actually used

Code Splitting

Create smaller chunks of JavaScript

Code Splitting

- On demand or parallel loading
- Smaller bundles (aka “chunks”)
- Shorter load and compute times
- “Prefetching” and “Preloading” with resource hints

webpack configs

Putting it all together in an (sometimes) ugly config file

Some examples

Basic

- <https://github.com/webpack/webpack/blob/master/examples/two-explicit-vendor-chunks/webpack.config.js>

Advanced

- <https://github.com/pangolinjs/core/tree/master/lib/config>

Insane

- <https://github.com/vuejs/vue-cli/tree/dev/packages/%40vue/cli-service/lib/config>