Getting Fat

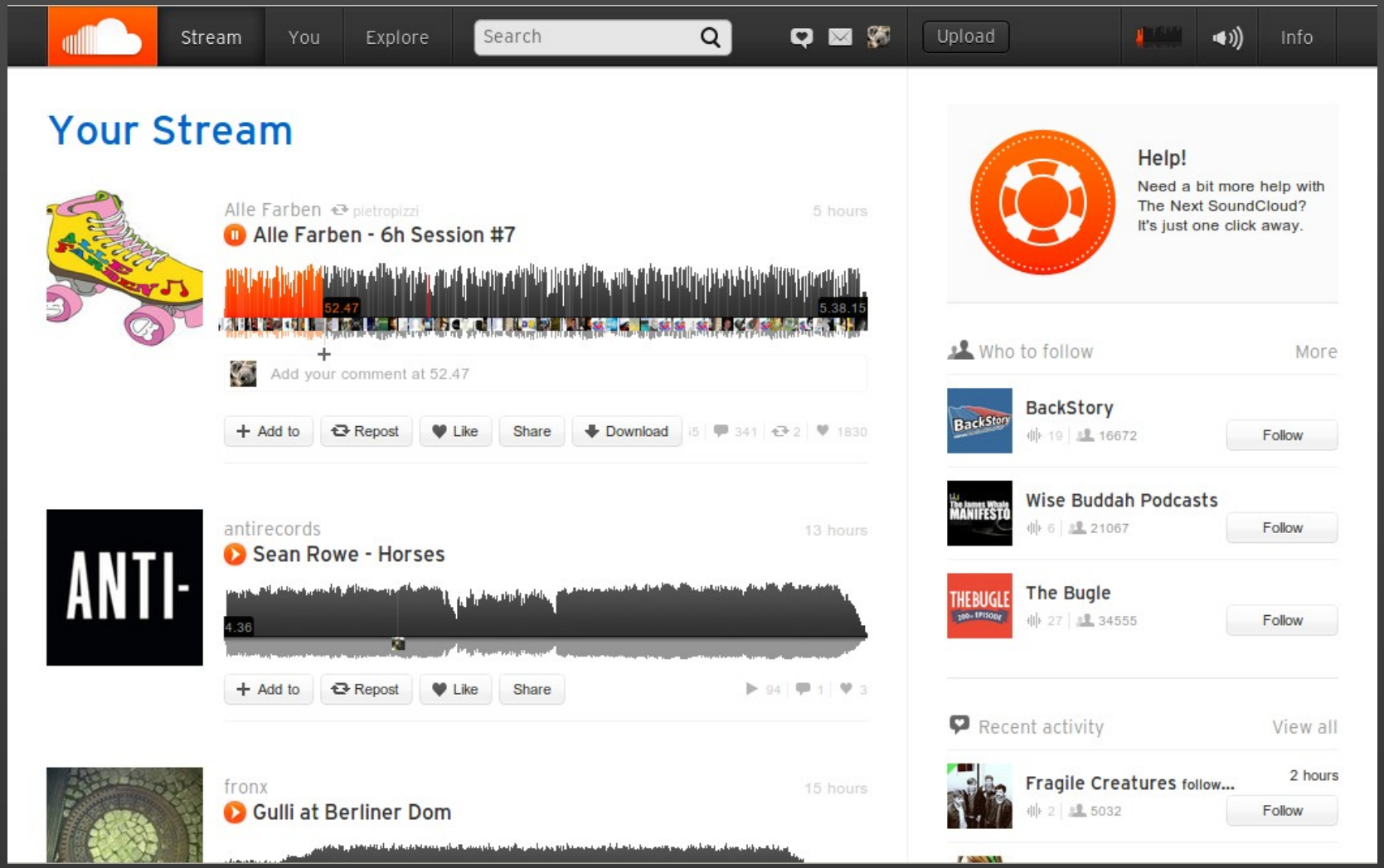
Tips and tricks for when you're building a single-page app

Me

- Nick Fisher
- @spadgos
- Front End Engineer
- Berlin based


SOUNDCLOUD

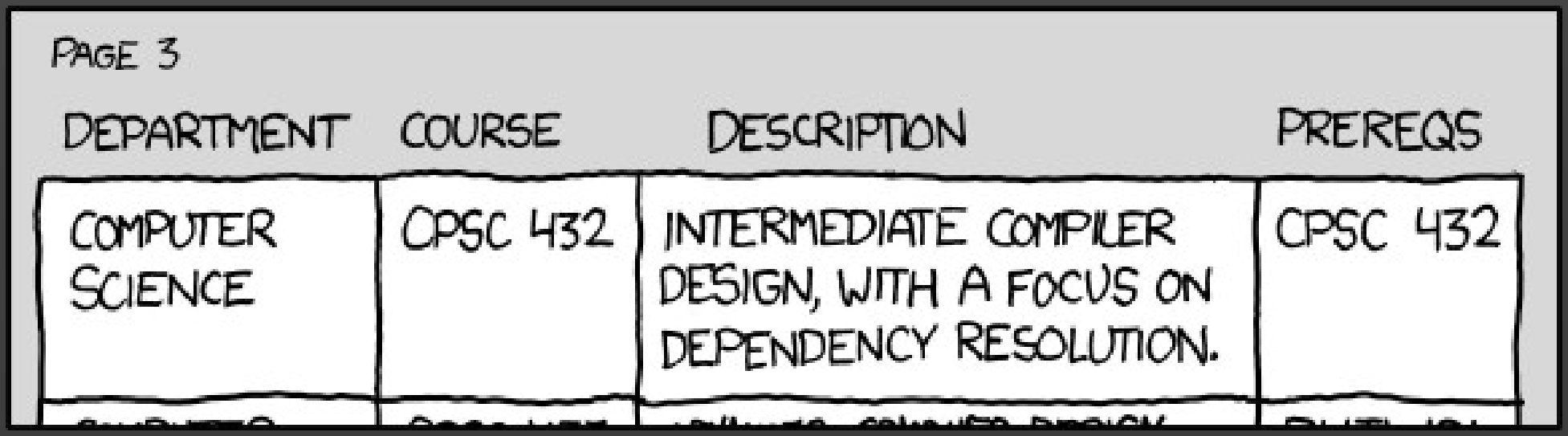
Next SoundCloud



Pain points

- Dependencies
- CSS Management
- Packaging for efficient delivery

Dependencies



PAGE 3

DEPARTMENT	COURSE	DESCRIPTION	PREREQS
COMPUTER SCIENCE	CPSC 432	INTERMEDIATE COMPILER DESIGN, WITH A FOCUS ON DEPENDENCY RESOLUTION.	CPSC 432

Dependencies

- Global namespaces

```
window.SC = window.SC || {};
SC.Sound = SC.Model.extend({
  …
})
```

```
<script src="/lib/localstorage.js"></script>
<script src="/lib/oauth-connection.js"></script>
<script src="/lib/backbone-sync.js"></script>
<script src="/lib/backbone-history.js"></script>
<script src="/lib/url.js"></script>
<script src="/lib/time.js"></script>
<script src="/lib/tmpl-manager.js"></script>
<script src="/lib/images.js"></script>
<script src="/lib/cache.js"></script>
<script src="/lib/audio.js"></script>
<script src="/lib/tracker.js"></script>
<script src="/lib/page.js"></script>
<script src="/lib/form.js"></script>
<script src="/lib/followings.js"></script>
<script src="/lib/page-list.js"></script>
<script src="/lib/lists.js"></script>
<script src="/lib/comment.js"></script>
<script src="/lib/authorization.js"></script>
<script src="/lib/connections.js"></script>
<script src="/lib/swipe.js"></script>
<script src="/lib/replace-diacritics.js"></script>
<script src="/application.js"></script>
<script src="/application-view.js"></script>
<script src="/search-view.js"></script>
<script src="/track/track.js"></script>
<script src="/playlist/playlist.js"></script>
<script src="/user/user.js"></script>
<script src="/home/home.js"></script>
<script src="/search/search.js"></script>
<script src="/suggestions/suggestions.js"></script>
```

Dependencies

Combining avoids this, but...

Must load everything

Ordering done manually

No correlation between code and file names

Solution: Modules!

- RequireJS + AMD
- ...but not
- Write CommonJS, serve AMD

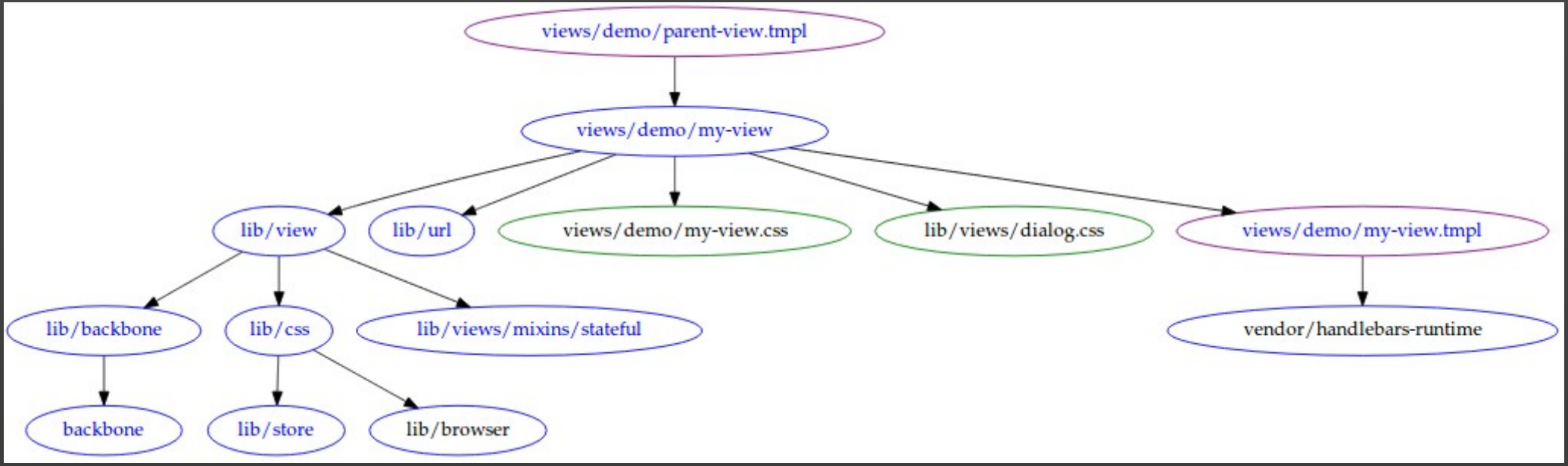
Modules

```
var Sound = require('models/sound'),
    URL = require('lib/url');

module.exports = { … };

define(
  'path/to/my-module',
  ['models/sound', 'lib/url'],
  function (require, exports, module) {
    var Sound = require('models/sound'),
        URL = require('lib/url');
    module.exports = { … };
  }
);
```

Graphs!


views/demo/parent-view.tmpl
views/demo/my-view
lib/view
lib/url
views/demo/my-view.css
lib/views/dialog.css
views/demo/my-view.tmpl
lib/backbone
lib/css
lib/views/mixins/stateful
vendor/handlebars-runtime
backbone
lib/store
lib/browser

Dependencies

```
<script src="/lib/localstorage.js"></script>
<script src="/lib/oauth-connection.js"></script>
<script src="/lib/backbone-sync.js"></script>
<script src="/lib/backbone-history.js"></script>
<script src="/lib/url.js"></script>
<script src="/lib/time.js"></script>
<script src="/lib/tmpl-manager.js"></script>
<script src="/lib/images.js"></script>
<script src="/lib/cache.js"></script>
<script src="/lib/audio.js"></script>
<script src="/lib/tracker.js"></script>
<script src="/lib/page.js"></script>
<script src="/lib/form.js"></script>
<script src="/lib/followings.js"></script>
<script src="/lib/page-list.js"></script>
<script src="/lib/lists.js"></script>
<script src="/lib/comment.js"></script>
<script src="/lib/authorization.js"></script>
<script src="/lib/connections.js"></script>
<script src="/lib/swipe.js"></script>
<script src="/lib/replace-diacritics.js"></script>
<script src="/application.js"></script>
<script src="/application-view.js"></script>
<script src="/search-view.js"></script>
<script src="/track/track.js"></script>
<script src="/playlist/playlist.js"></script>
<script src="/user/user.js"></script>
<script src="/home/home.js"></script>
<script src="/search/search.js"></script>
<script src="/suggestions/suggestions.js"></script>
```

```
<script src="require.js"
data-main="application">
```

Must load everything

Ordering done manually

No correlation between code and file names

CSS

{{insert witty meme image here}}

CSS

```
body#pages.sounds #main-wrapper #main-
wrapper-inner #soundpage #primary .social-
sharing .share-plugin.last .twitter-share-
button {
  width: 101px !important;
}
```

CSS Problems

- Everything loaded at once
- Fragile to template changes
- Slow!

Our Solution

- Lots of small-to-medium sized views
- One CSS file per view

```
.soundTitle__info {
  display: inline-block;
  vertical-align: baseline;
  font-size: 11px;
}

.soundTitle__title {
  display: block;
  width: 100%;
}

.soundTitle__playButton {
  margin-right: 5px;
}

.soundTitle__tag {
  line-height: 12px;
}
```

CSS → JS

Yep.

CSS → JS

```
define(
  "views/search/search-item.css",
  [],
  function(require, exports, module) {
    var style = module.exports =
       document.createElement("style");
    style.appendChild(
      document.createTextNode(
        '.soundTitle__info{display...}'
      )
    );
  }
);
```

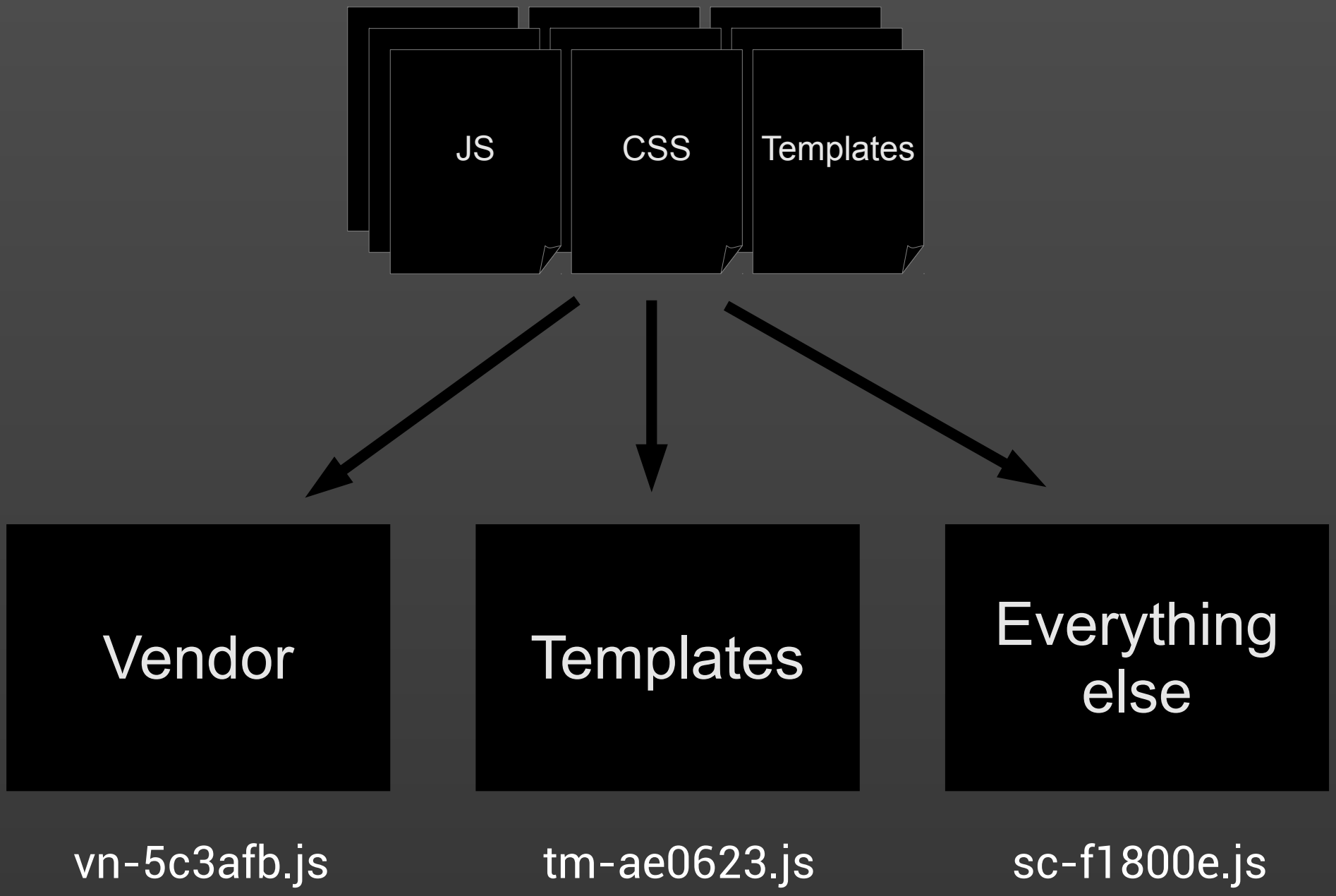
CSS Problems

- Everything loaded at once
- Fragile to template changes
- Slow!

Packaging

Packaging

- Combine + Minify
- Use the AST for great win
- Find just what you need
- Application files + assets

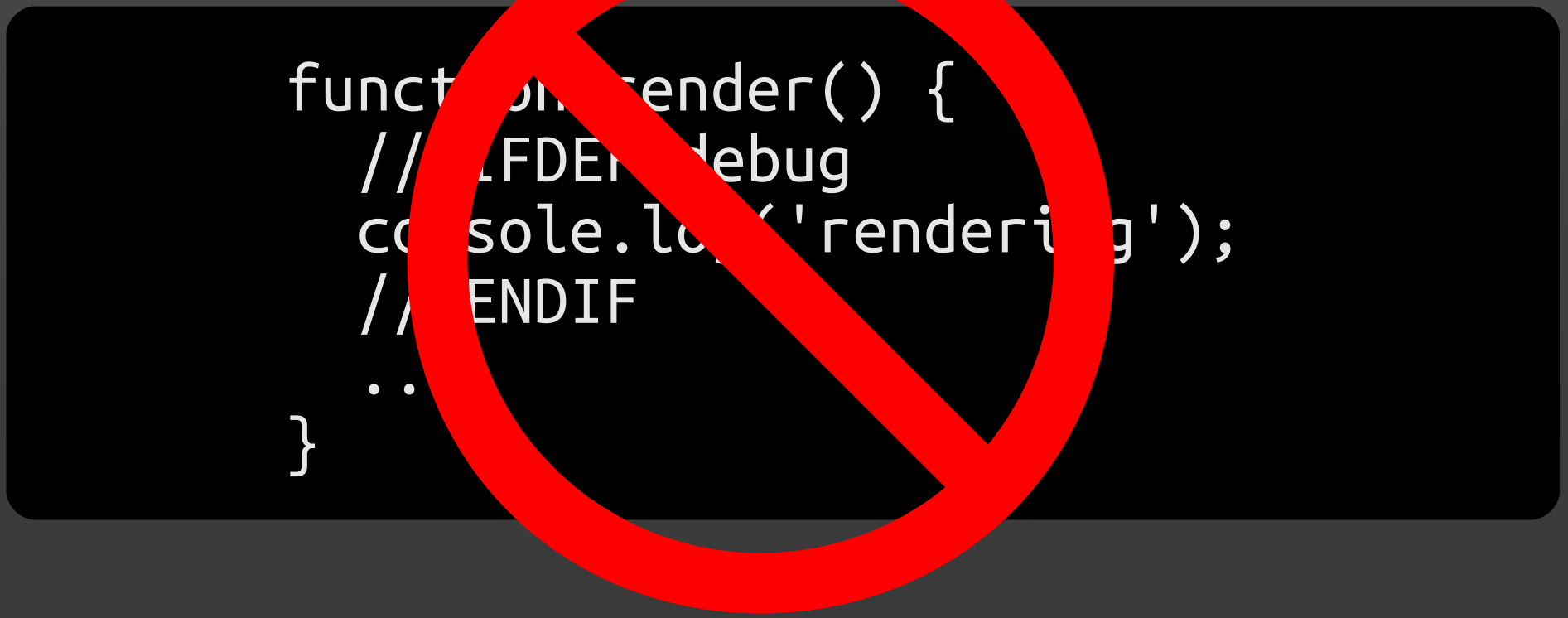

JS
CSS
Templates
Vendor
Templates
Everything else
vn-5c3afb.js
tm-ae0623.js
sc-f1800e.js

Minify

- UglifyJS
- Great out of the box
- With more poking, fantastic.

Preprocessors

- Development-only code



Use the AST!

4432

You can't parse [X]HTML with regex. Because HTML can't be parsed by regex. Regex is not a tool that can be used to correctly parse HTML. As I have answered in HTML-and-regex questions here so many times before, the use of regex will not allow you to consume HTML. Regular expressions are a tool that is insufficiently sophisticated to understand the constructs employed by HTML. HTML is not a regular language and hence cannot be parsed by regular expressions. Regex queries are not equipped to break down HTML into its meaningful parts. so many times but it is not getting to me. Even enhanced irregular ... You will never ... to ... asic only worse ... lightenmen HTML tags ... ~~parsing~~ will exti nguish the voices of mor... you see it it is beautiful the f `inal` `snuf` fing of *the lies of Man ALL IS LOST* **ALL IS** LOST the *pony he comes* he ~~comes he comes~~ **the ichor** permeates all MY FACE *MY FACE ᵒh god no* **NO NOOOO** NΘ stop the an...gles `.ar` **not real** **ZALGO TONY THE PONY HE COMES**

Have you tried using an XML parser instead?

humanity to an eternity of dread torture and security holes *using regex* as a tool to process HTML establishes a breach *between this world* and the dread realm of corrupt entities (like SGML entities, but *more corrupt) a mere glimpse* of the world of regex **parsers for HTML will ins**tantly transport a *programmer's consciousness into a world* of ceaseless screaming, he comes, ~~the pestilent slithy~~ regex-infection will **devour your HT**ML parser, application and existence for all time like Visual Basic only worse *he comes he comes do not fight* **he comes, his** unholy radiance *destroying all enlightenmen HTML tags* ***leaking from your eyes like liquid*** pain, the song of regular expression ~~parsing~~ will exti *nguish the voices of mortal man from the sphere I can see it can you see it it is beautiful* the f `inal` `snuf` fing of *the lies of Man ALL IS LOST* **ALL IS** LOST the *pony he comes* he ~~comes he comes~~ **the ichor** permeates all MY FACE *MY FACE ᵒh god no* **NO NOOOO** NΘ stop the an...gles `.ar` **not real** **ZALGO TONY THE PONY HE COMES**

Have you tried using an XML parser instead?

Using the AST

```
var client_id = __ENV__ === 'production' ? 'abc' : 'def';

if (__DEBUG_MODE__ && someCondition) {
  console.log(client_id);
}
```

- Mangle. Shortens variables, substitutes constants

```
var a = 'production' === 'production' ? 'abc' : 'def';

if (false && someCondition) {
  console.log(a);
}
```

Using the AST

```
var a = 'production' === 'production' ? 'abc' : 'def';

if (false && someCondition) {
  console.log(a);
}
```

- Squeeze. Many things, including dead code removal

```
var a = 'abc';
```

Summing up

- Modular JS!
- Modular CSS!
- Modular JS + modular CSS + ASTs = win!

Kiitos!

Questions?

Nick Fisher @spadgos #nextsoundcloud