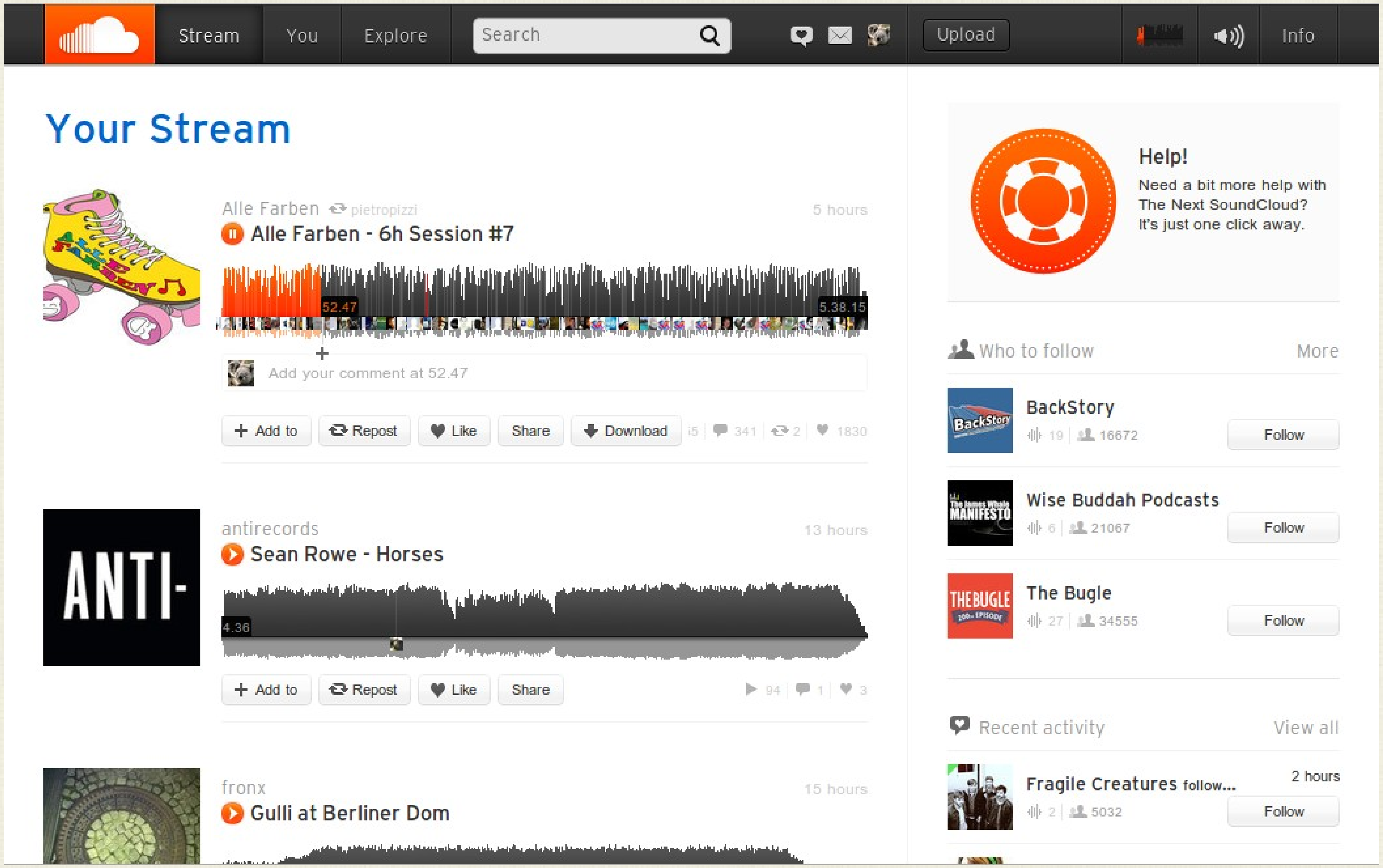
The Next SoundCloud



Stream You Explore Search Upload Info

Your Stream

Alle Farben pietropizzi 5 hours
Alle Farben - 6h Session #7
52.47 5.38.15
Add your comment at 52.47
Add to Repost Like Share Download 15 341 2 1830

antirecords 13 hours
Sean Rowe - Horses
4.36
Add to Repost Like Share 94 1 3

fronx 15 hours
Gulli at Berliner Dom

Help!
Need a bit more help with The Next SoundCloud? It's just one click away.

Who to follow More

BackStory	19 | 16672	Follow
Wise Buddah Podcasts	6 | 21067	Follow
The Bugle	27 | 34555	Follow

Recent activity View all

Fragile Creatures follow... 2 hours
2 | 5032 Follow

@spadgos #nextsoundcloud

Developer tooling

- Written in NodeJS
  - Can share config between runtime and tools
- Development server
- Build script

Build Optimisations

- JS preprocessors

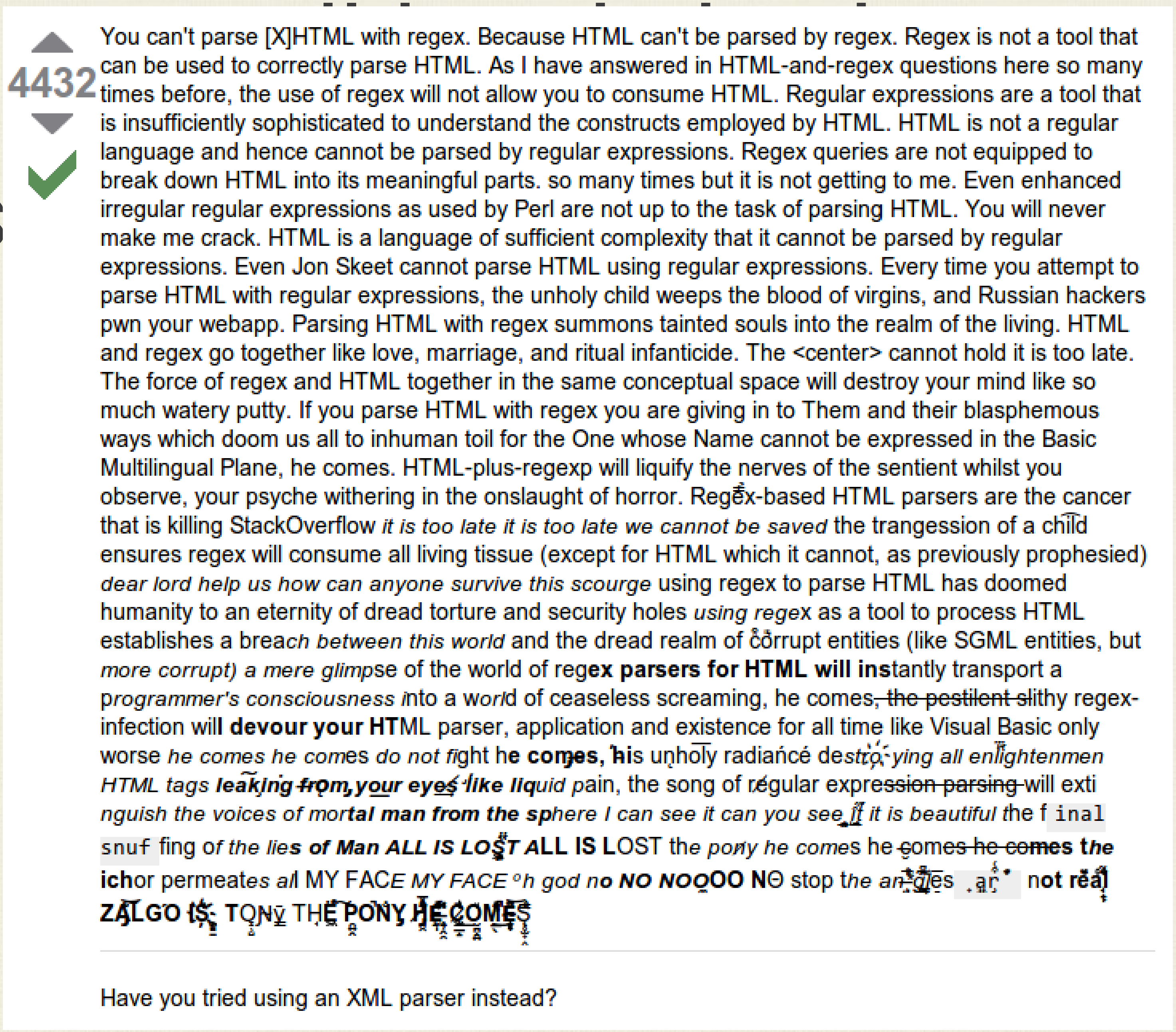
```
//#define DEBUG 1

//#if DEBUG
console.log(...);
//#endif
```

Build Optimisations

- JS preprocessors

```
//    fine DEBUG

//#if    UG
console.    (...);
//#endif
```

- JS



4432

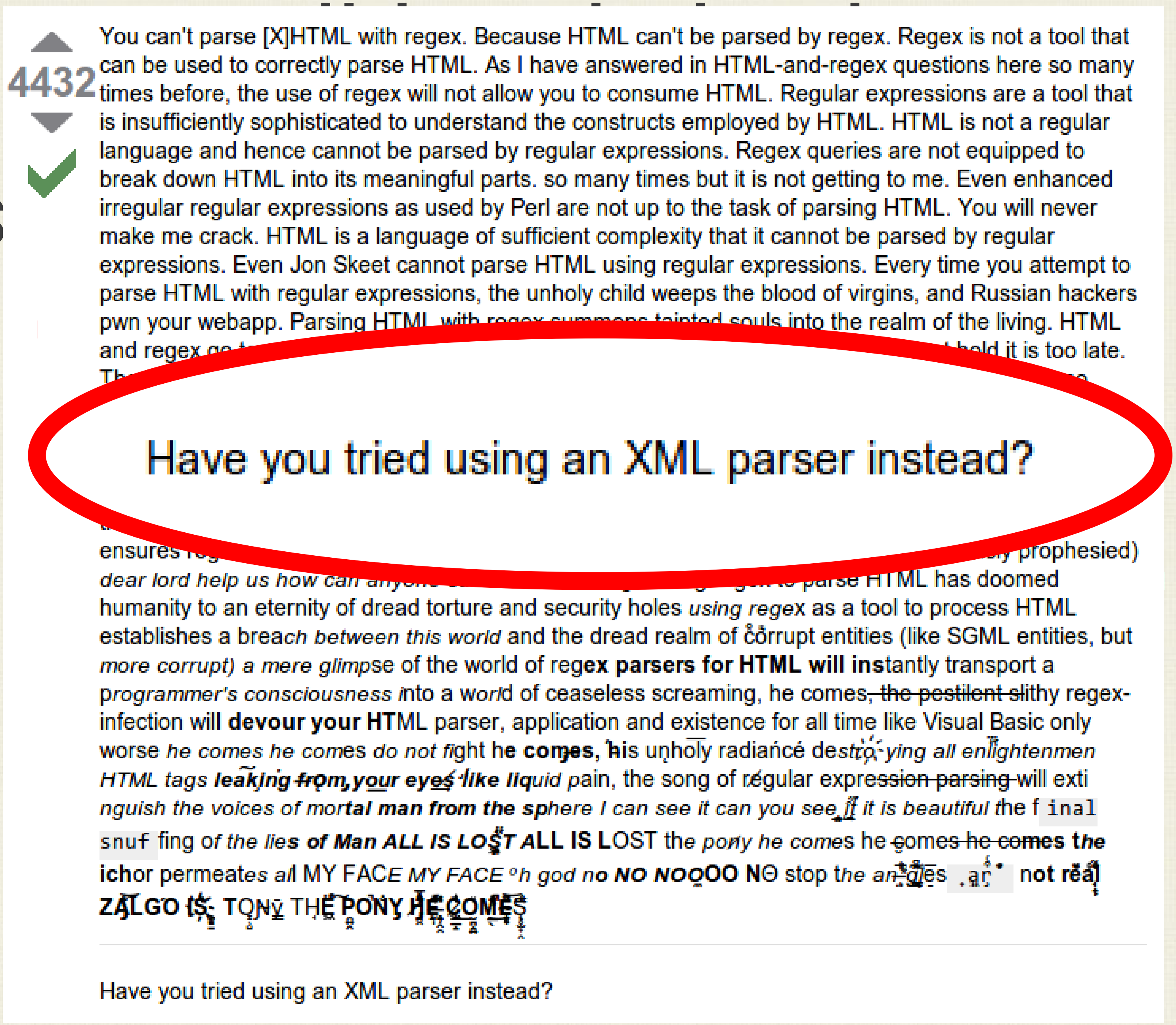
You can't parse [X]HTML with regex. Because HTML can't be parsed by regex. Regex is not a tool that can be used to correctly parse HTML. As I have answered in HTML-and-regex questions here so many times before, the use of regex will not allow you to consume HTML. Regular expressions are a tool that is 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 regular expressions as used by Perl are not up to the task of parsing HTML. You will never make me crack. HTML is a language of sufficient complexity that it cannot be parsed by regular expressions. Even Jon Skeet cannot parse HTML using regular expressions. Every time you attempt to parse HTML with regular expressions, the unholy child weeps the blood of virgins, and Russian hackers pwn your webapp. Parsing HTML with regex summons tainted souls into the realm of the living. HTML and regex go together like love, marriage, and ritual infanticide. The <center> cannot hold it is too late. The force of regex and HTML together in the same conceptual space will destroy your mind like so much watery putty. If you parse HTML with regex you are giving in to Them and their blasphemous ways which doom us all to inhuman toil for the One whose Name cannot be expressed in the Basic Multilingual Plane, he comes. HTML-plus-regexp will liquify the nerves of the sentient whilst you observe, your psyche withering in the onslaught of horror. Regex-based HTML parsers are the cancer that is killing StackOverflow *it is too late it is too late we cannot be saved* the trangession of a child ensures regex will consume all living tissue (except for HTML which it cannot, as previously prophesied) *dear lord help us how can anyone survive this scourge* using regex to parse HTML has doomed humanity to an eternity of dread torture and security holes *using rege*x as a tool to process HTML establishes a brea*ch between this world* and the dread realm of corrupt entities (like SGML entities, but *more corrupt) a mere glimp*se of the world of reg**ex parsers for HTML will ins**tantly transport a p*rogrammer's consciousness i*nto a world of ceaseless screaming, he comes~~, the pestilent sl~~ithy regex-infection will **devour your HT**ML parser, application and existence for all time like Visual Basic only worse *he comes he com*es *do not fi*ght h**e comes, hi**s unholy radiancé destro*ying all enlightenmen HTML tags* ***leaking from your eyes like liq****uid* pain, the song of regular expre~~ssion parsing~~ will exti*nguish the voices of mor****tal man from the sp****here I can see it can you see it it is beautiful t*he f `inal snuf` fing o*f the lie****s of Man ALL IS LOST*** **ALL IS L**OST th*e pony he comes* he ~~comes he comes~~ **t*he* ich**or permeates a*ll* MY FACE *MY FACE ᵒh god n**o NO NOO**O*** **O N**Θ stop th*e an*gles `ar` **not real ZALGO IS TONY THE PONY HE COMES**

Have you tried using an XML parser instead?

@spadgos

#nextsoundcloud

- JS



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 regular expressions as used by Perl are not up to the task of parsing HTML. You will never make me crack. HTML is a language of sufficient complexity that it cannot be parsed by regular expressions. Even Jon Skeet cannot parse HTML using regular expressions. Every time you attempt to parse HTML with regular expressions, the unholy child weeps the blood of virgins, and Russian hackers pwn your webapp. Parsing HTML with regex summons tainted souls into the realm of the living. HTML and regex go to [illegible] hold it is too late. Th[illegible]

Have you tried using an XML parser instead?

ensures reg[illegible] prophesied) *dear lord help us how can anyone* [illegible] parse HTML has doomed humanity to an eternity of dread torture and security holes *using rege*x as a tool to process HTML establishes a brea*ch between this world* and the dread realm of c̈orrupt entities (like SGML entities, but *more corrupt) a mere glimp*se of the world of reg**ex parsers for HTML will ins**tantly transport a p*rogrammer's consciousness i*nto a wor*l*d of ceaseless screaming, he comes~~, the pestilent sl~~ithy regex-infection wil**l devour your HT**ML parser, application and existence for all time like Visual Basic only worse *he comes he com*es *do not fi*ght h**e com̡e̶s, ̕h̵i**s un̨ho͞ly radiańcé destro҉ying all enli̍ghtenment, HTML tags **lea͠ki̧n͘g ~~fr̶ǫm~~ ̡yo͟ur eye͢s̸ ̛l̕ik͏e liq**uid pain, the song of re̸gular expre~~ssion parsing~~ will exti*nguish the voices of mor***tal man from the sp***here I can see it can you see ̲͚̖͔̙î̩́t̲͎̩̱͔́̋̀ it is beautiful t*he f`inal snuf`fing o*f the lie***s of Man ALL IS LOŚ͖̩͇̗̪̏̈́T ALL I**S LOST the pon̷y he comes he ~~c̶̮omes he comes~~ **the ich**or permeates all MY FACE MY FACE ᵒh god no NO NOO̼OO NΘ stop the an*̶͑̾̾̅ͫ͏̙̤g͇̫͛͆̾ͫ̑͆l͖͉̗̩̳̟̍ͫͥͨe̠̅s ͎a̧͈͖r̽̾̈́͒͑e not rè̑ͧ̌aͨl̘̝̙̃ͤ͂̾̆ ZA̡͊͠͝LGΌ ISͮ̂҉̯͈͕̹̘̱ TO͇̹̺ͅƝ̴ȳ̳ TH̘Ë͖́̉ ͠P̯͍̭O̚N̐Y̡ H̸̡̪̯ͨ͊̽̅̾̎Ȩ̬̩̾͛ͪ̈́̀́͘ ̶̧̨̱̹̭̯ͧ̾ͬC̷̙̲̝͖ͭ̏ͥͮ͟Oͮ͏̮̪̝͍M̲̖͊̒ͪͩͬ̚̚͜Ȇ̴̟̟͙̞ͩ͌͝S̨̥̫͎̭ͯ̿̔̀ͅ

Have you tried using an XML parser instead?

Build Optimisations

- Use UglifyJS to parse code into an AST
- Inspect and manipulate from there

```
var parser = require('uglify-js').parser;

var ast = parser.parse(fileContents);
```

Build Optimisations

- Variable substitution

```
var client_id = __ENV__ === 'production' ? 'abc123' : 'def456';

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

- `uglify.ast_mangle(ast, {defines: … })`

```
var a = 'production' === 'production' ? 'abc123' : 'def456';

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

Build Optimisations

- Dead code removal

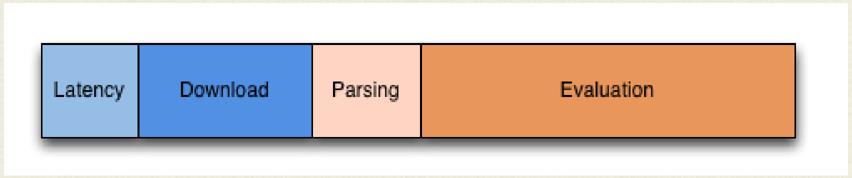
```
var a = 'production' === 'production' ? 'abc123' : 'def456';

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

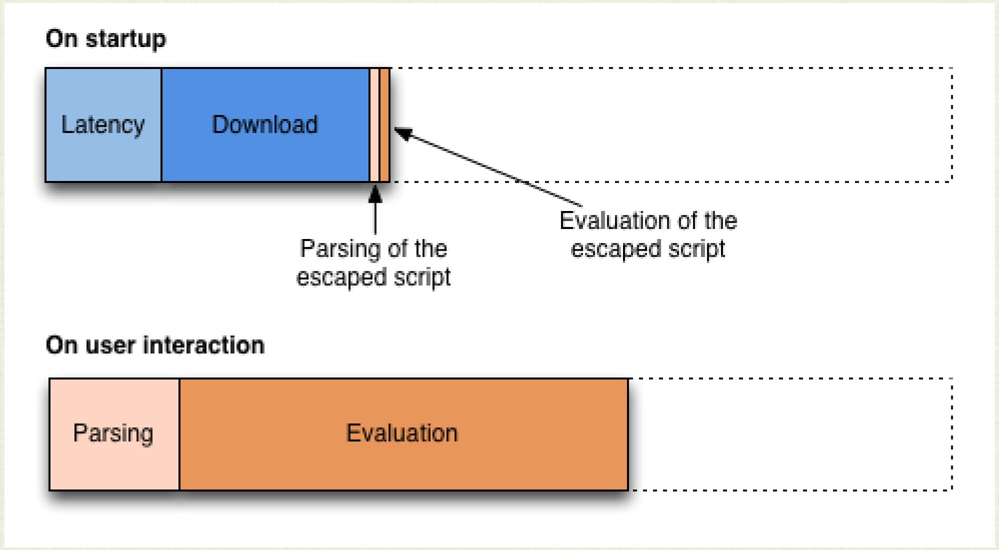
- `uglify.ast_squeeze(ast)`

```
var a = 'abc123';
```

Build Optimisations

- Lazy evaluation of AMD modules
  - Tobie Langel (@tobie)
  - http://calendar.perfplanet.com/2011/lazy-evaluation-of-commonjs-modules/




On startup
Latency
Download
Parsing of the escaped script
Evaluation of the escaped script
On user interaction
Parsing
Evaluation

Templating: Handlebars

- Enforce good practices
- Pre-compiled
  - Faster to render
  - Smaller to deliver
- Custom helpers

Code style

```
var PlayQueue = require('lib/play-queue'),
    Comments  = require('collections/comments'),
    View      = require('lib/view');

module.exports = View.extend({
  // ...
});
```

Code style

```
var PlayQueue = require('lib/play-queue'),
    Comments  = require('collections/comments'),
    View      = require('lib/view');

module.exports = View.extend({
  template: require('views/my-view.tmpl'),
  css     : require('views/my-view.css')
});
```

CSS AMD modules?

Yeah! CSS AMD modules.

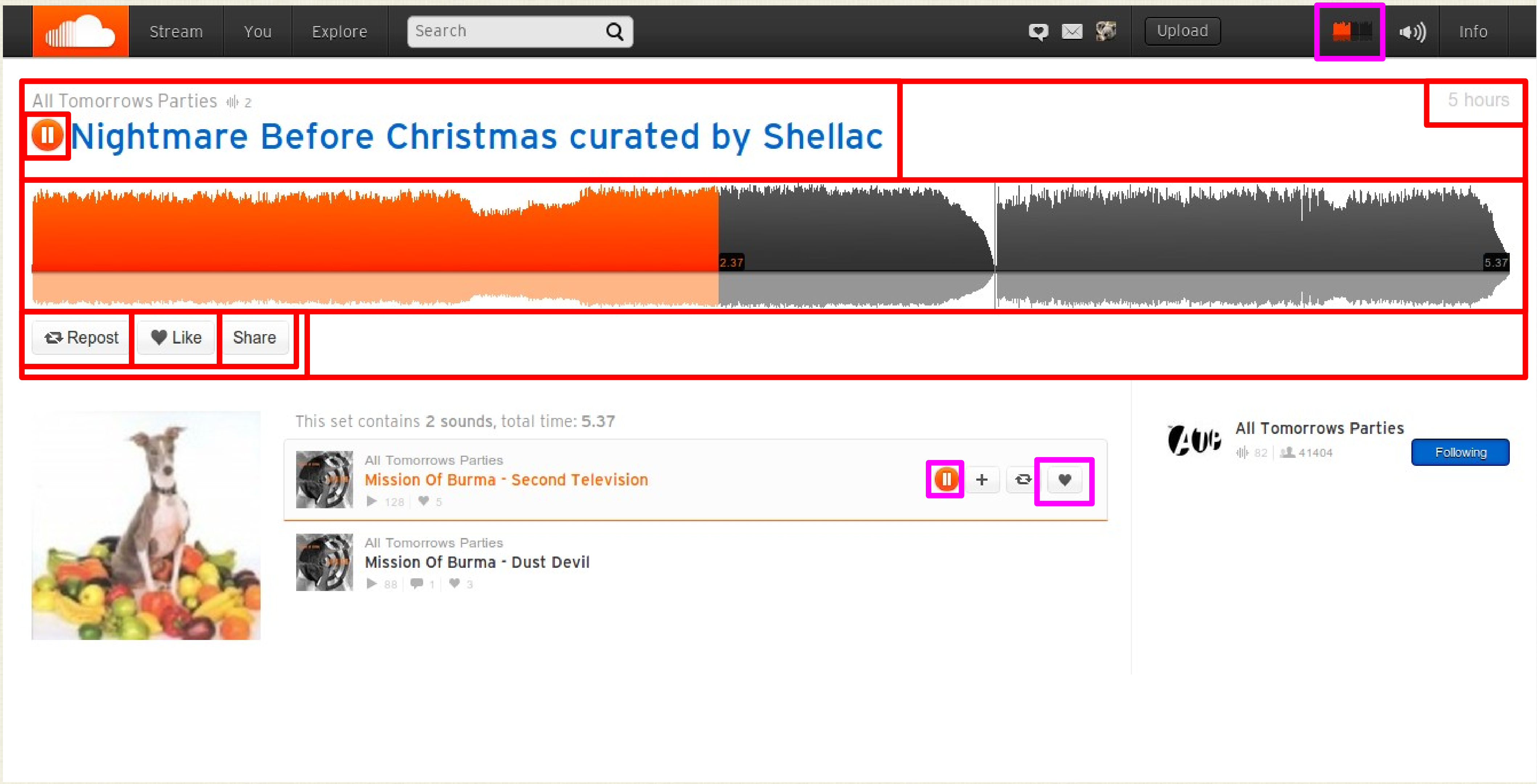
```
define("views/my-view.css", function () {

  var style = document.createElement('style'),
      data = ".myView{margin-top: 20px;} .myV...";

  style.appendChild(
    document.createTextNode(data)
  );
  return style;
});
```

Views as Components

- **Independent** & Reuseable
- Can include subviews with subviews with subviews...
- Some are large, many small
- Must play nice

Views added via Templates

```
<div class="foo">
  {{view "views/sound/play-button"
    resource_id=sound_id
  }}
  {{view "views/user/user-badge"
    resource_id=user_id
    size="large"
  }}
</div>
```



Stream
You
Explore
Search
Upload
Info

All Tomorrows Parties 2

Nightmare Before Christmas curated by Shellac

5 hours

2.37
5.37

Repost
Like
Share

This set contains 2 sounds, total time: 5.37

All Tomorrows Parties Mission Of Burma - Second Television 128 5	
All Tomorrows Parties Mission Of Burma - Dust Devil 88 1 3	

All Tomorrows Parties
82 | 41404
Following

@spadgos

#nextsoundcloud

Models

- Identity map behaviour

```
var soundA = new Sound({ id: 123, title: 'Foo' }),
    soundB = new Sound({ id: 123, genre: 'techno' });

soundA === soundB;   // true
soundB.get('title'); // 'Foo'
soundA.get('genre'); // 'techno'
```

How?

- Instance store is just an object (in essence)
- Override Model constructor

```
store = {};
Sound = Backbone.Model.extend({
  constructor: function (attrs) {
    var id = attrs.id;
    if (store[id]) {
       return store[id]; // ← return the other one
    }
    store[id] = this;
    // regular instantiation...
  }
});
```

Problem solved!

- Models fetched once, rendered many times
- Models + events can be used to synchronise views
- Nice side effect: full use of the response

Sub-resources in a response

https://api.soundcloud.com/tracks/52167545.json

```
{
  "id": 52167545,
  "user_id": 2,
  "duration": 71523,
  "user": {
    "id": 2,
    "permalink": "eric",
    "username": "Eric",
    "avatar_url": "https://i1.sndcdn..."
  },
  "created_with": {
    "id": 124,
    "name": "SoundCloud iOS"
  }
  // …
}
```

Releasing

- Instance store must let go at some point
- When a model is 'constructed': `usage++`
- When a model is 'released': `usage--`
- Periodically, remove unused models

Backbone + SoundCloud