Sven Sauleau

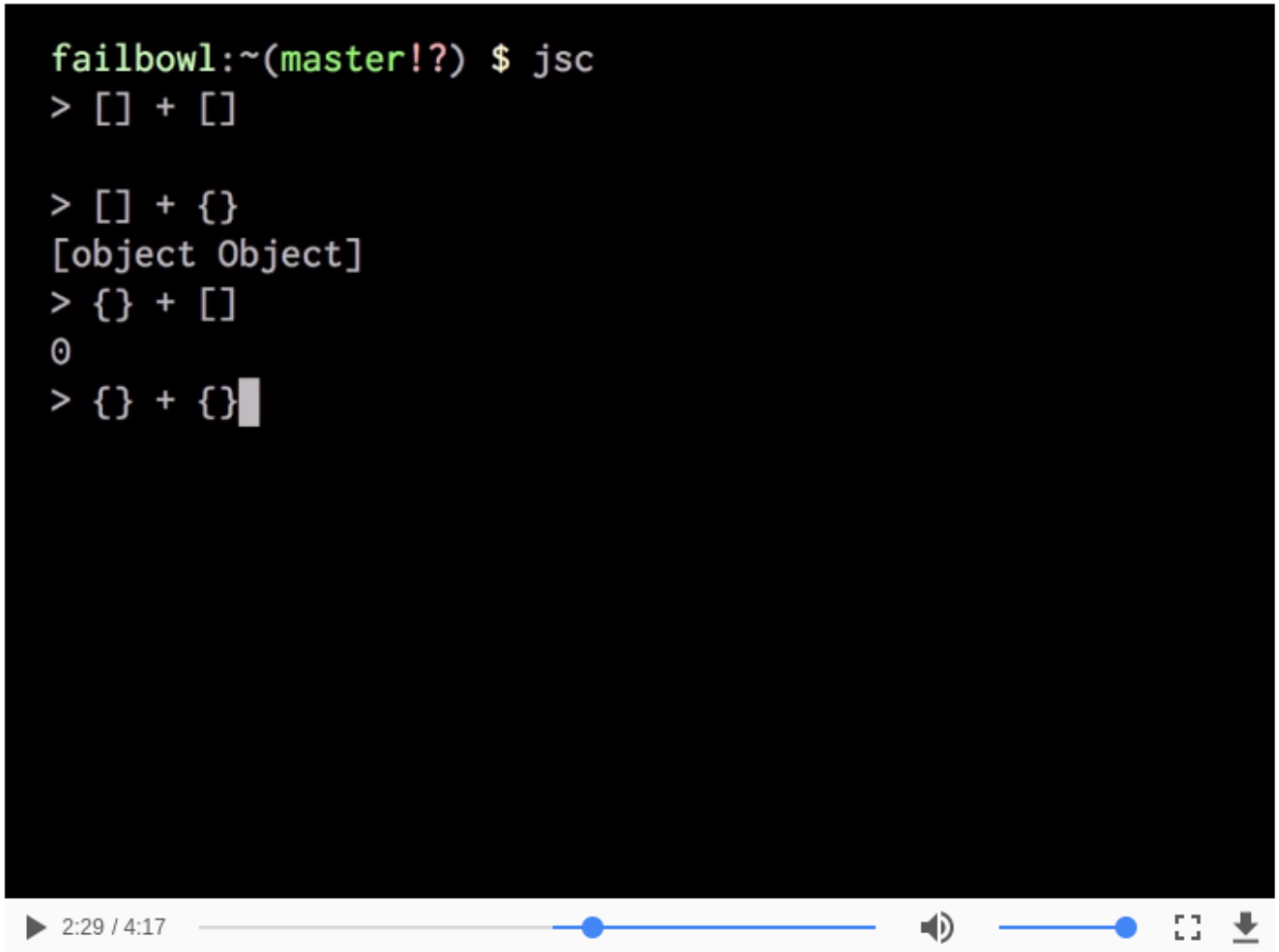
@svensauleau

Understanding the Differences Is Accepting

Prior art

Wat

A lightning talk by Gary Bernhardt from CodeMash 2012



Question

```
switch (0) {
    case 0:
        a();
        break;

    case 1:
        function a() { console.log("foo"); }
        break;
}
```

What's the output?

1. "foo"
2. a is not defined

ECMAScript Spec

```
switch (0) {
    case 0:
        a();
        break;

    case 1:
        function a() { console.log("foo"); }
        break;
}
```

1. the switch body creates a new scope
2. cases are not creating a new scope
3. function declarations are hoisted

Fix

```
switch (0) {
    case 0:
        a();
        break;

    case 1: { // <--
        function a() { console.log("foo"); }
        break;
    } // <--
}
```

Question

```
1 "\n\t\r\n\t\r" == false
```

What's the output?

1. true
2. false
3. $x > \infty$

ECMAScript Spec

```
1 "\n\t\r\n\t\r" == false
```

1. \t, \n and \r are expanded to empty string
2. empty string is falsy

Question

```
1 NaN == NaN
```

What's the output?

1. true
2. true (because it can't be false)

ECMAScript Spec

Unrepresentable/broken value

6.1.6 The Number Type

"[...] all NaN values are indistinguishable from each other."

7.2.15 Strict Equality Comparison

"2.a If x is NaN, return false or 2.b If y is NaN, return false."

Question

```
Math.pow(2, 53) + 1 === Math.pow(2, 53)
```

What's the output?

1. true
2. false
3. quantum state

ECMAScript Spec

```
1 Math.pow(2, 53) + 1 === Math.pow(2, 53)
```

6.1.6 The Number Type

1. Number are 64 bits float
2. 11 bits are for the exponent
3. -2^{53} to $+2^{53}$

Question

```
1 [1, 2] == '1,2'
```

What's the output?

1. true
2. false

ECMAScript Spec

```
1 [1 ,  2] == '1,2'
```

7.2.14 Abstract Equality Comparison

"9. If Type(x) is Object and Type(y) is either [...], return the result of the comparison ToPrimitive(x) == y."

and then basically [1, 2].toString().

Question

```
1 <!-- console.log("foo") -->
```

What's the output?

1. "foo"
2. parse error (because I'm a JS engine?)
3. no output?

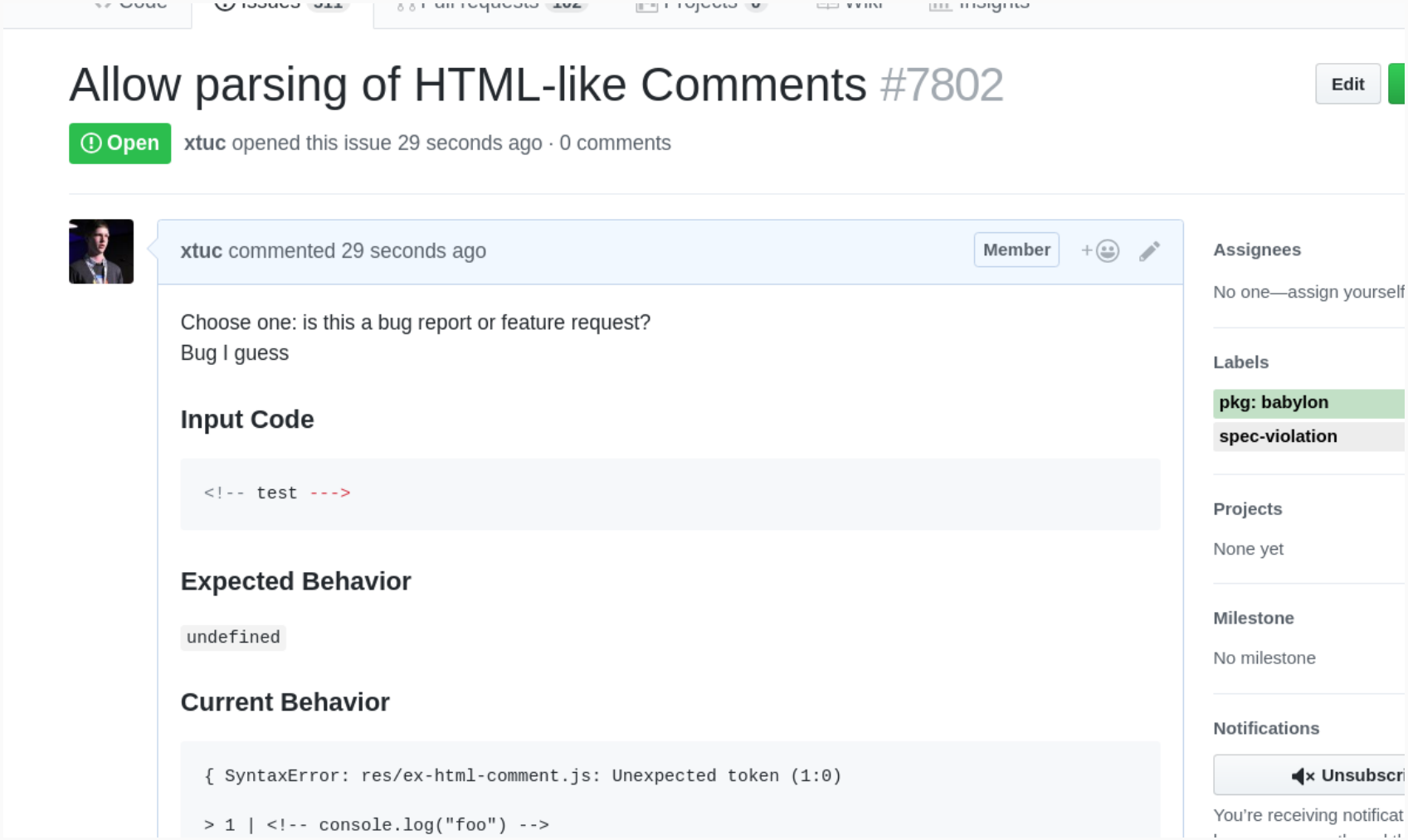
ECMAScript Spec

Parsing is defined at B.1.3 HTML-like Comments.

Allow browsers that didn't understand the script tag to degrade gracefully

ex Netscape 1

Babel



Allow parsing of HTML-like Comments #7802

Edit

Open xtuc opened this issue 29 seconds ago · 0 comments

xtuc commented 29 seconds ago Member

Choose one: is this a bug report or feature request?
Bug I guess

Input Code

```
<!-- test --->
```

Expected Behavior

```
undefined
```

Current Behavior

```
{ SyntaxError: res/ex-html-comment.js: Unexpected token (1:0)

> 1 | <!-- console.log("foo") -->
```

Assignees

No one—assign yourself

Labels

pkg: babylon

spec-violation

Projects

None yet

Milestone

No milestone

Notifications

Unsubscri

You're receiving notificat

Question

```
1 var t
2
3 t = 0
4 (1 + 1)
```

What's the output?

1. 1
2. 0 is not a function (because that's the truth)

ECMAScript Spec

```
1 var t
2
3 t = 0
4 (1 + 1)
```

11.9 Automatic Semicolon Insertion

No ASI because “[...] the parenthesized expression that begins the second line can be interpreted as an argument list for a function call.”

Fix

```
var t;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;t = 0;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;(1 + 1);;;;;;;;;;;;;;;;;;;;
```

Clearer Fix

```
var t;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

<!----------------------------------------------------->

;;;;;;;;;;;;;;;;;;;;t = 0;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;(1 + 1);;;;;;;;;;;;;;;;;;;;

<!---------- I hope that make sense ----------->
```

Question

```
1 bitcoin: {}{}{}{}{}{}{}{}{}{}
```

What's the output?

1. Mining bitcoins?

Spec

```
1 bitcoin: {}{}{}{}{}{}{}{}{}{}
```

bitcoin is a label forming a LabeledStatement

Following by BlockStatements

aka the blockchain

Conclusion

Everything is specified!

Don't let me commit on your project

Thanks