
The Art
of the AST

//TODO

Write linting rules to enforce
your team's code conventions

Write your own javascript
transpiling code

Write powerful "code-mods"
to automatically refactor
thousands of legacy scripts
from ES5 to ES6

//TODO

Write linting rules to enforce your team's code conventions

Write your own javascript transpiling code

Write powerful "code-mods" to automatically refactor thousands of legacy scripts from ES5 to ES6

`jscodeshift`

//TODO

Write linting rules to enforce your team's code conventions

Write your own javascript transpiling code

Write powerful "code-mods" to automatically refactor thousands of legacy scripts from ES5 to ES6

`jscodeshift`

ESTree-based AST

+

Visitor Pattern

Yonatan Mevorach

- Javascript infrastructure at sears.israel
- code, blog, speak, repeat

blog.cowchimp.com
github.com/cowchimp
@cowchimp

Webpack

Babel

Uglifyjs

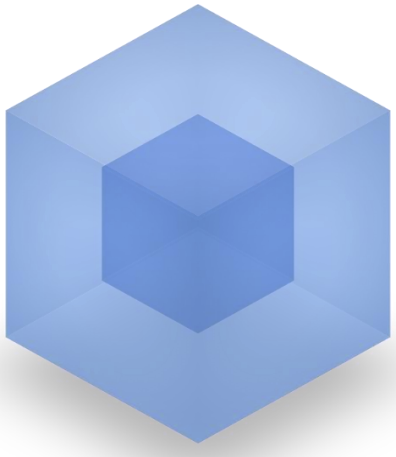
Rollup

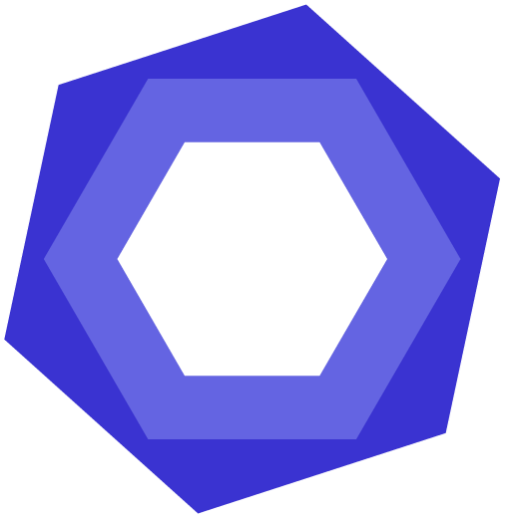
ESLint

Istanbul

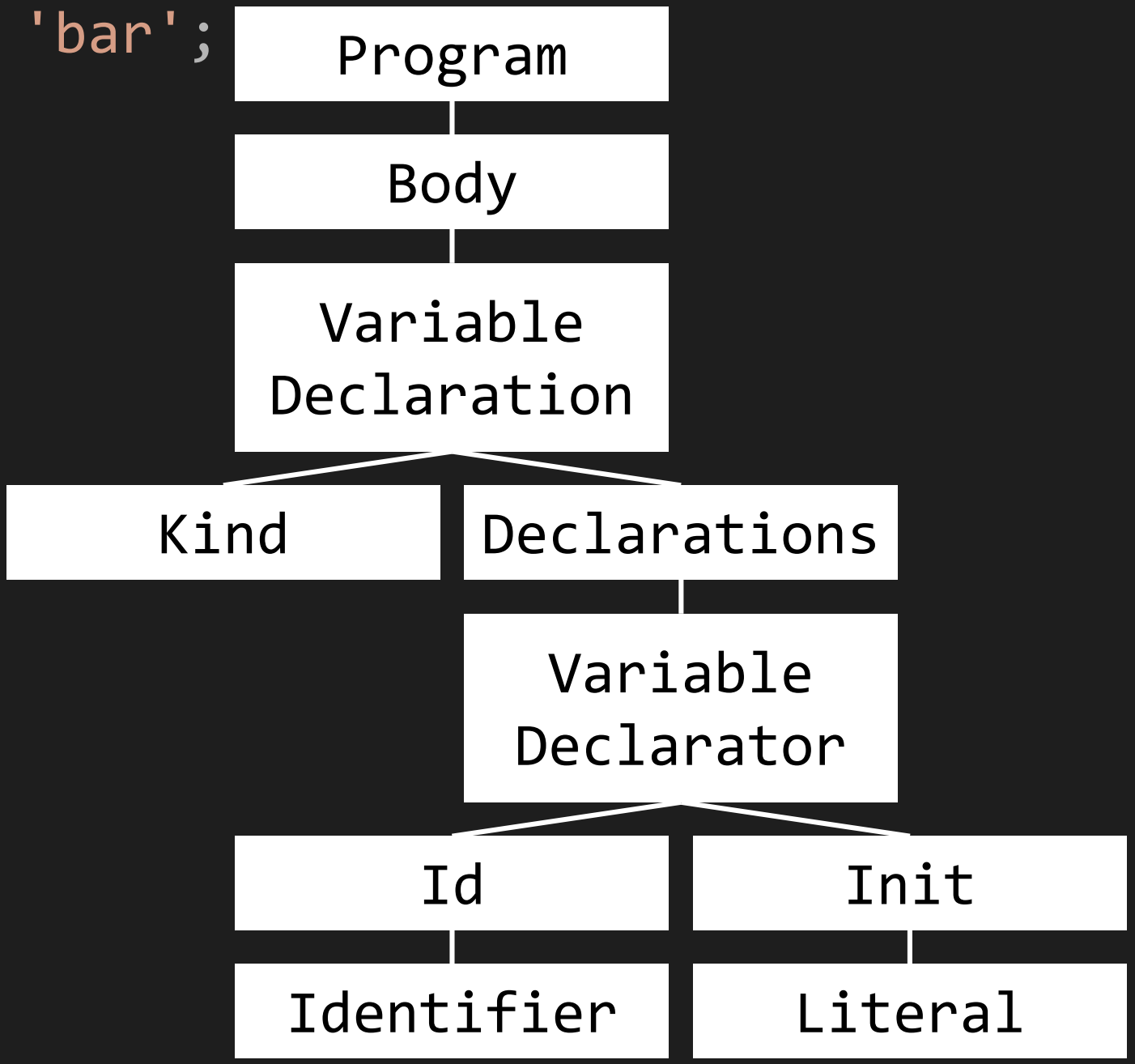
Ternjs

jscodeshift





```
var foo = 'bar';
```


Program
Body
Variable Declaration
Kind
Declarations
Variable Declarator
Id
Init
Identifier
Literal

```
var foo = 'bar';
```

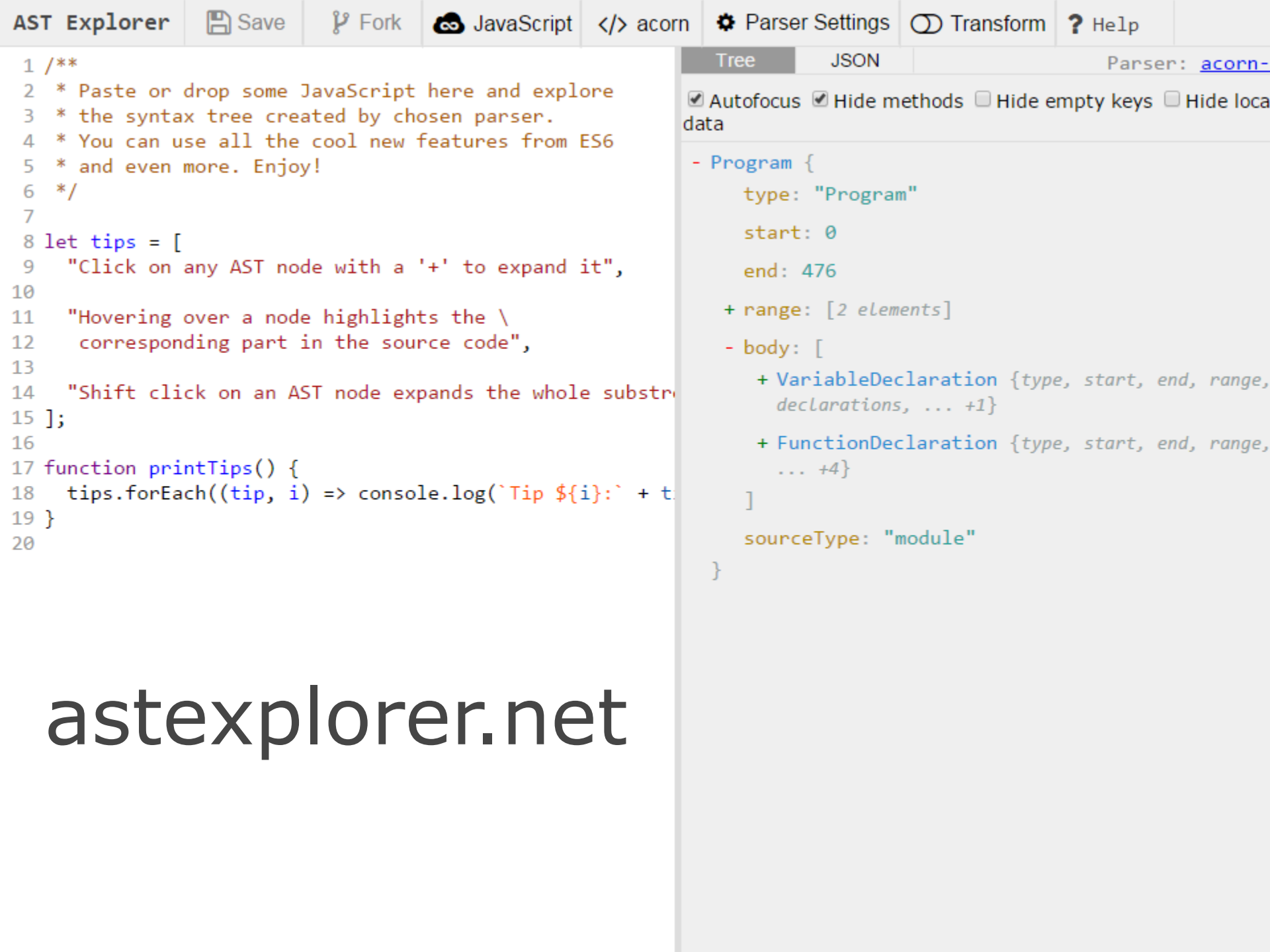

```
{
  "type": "Program",
  "body": [
    {
      "type": "VariableDeclaration",
      "declarations": [
      {
        "type": "VariableDeclarator",
        "id": {
          "type": "Identifier",
          "name": "foo"
        },
        "init": {
          "type": "Literal",
          "value": "bar"
        }
      }
      ],
      "kind": "var"
    }
  ]
}
```

```
var foo = 'bar';
```


```
{
  "type": "Program",
  "body": [
    {
      "type": "VariableDeclaration",
      "declarations": [
      {
        "type": "VariableDeclarator",
        "id": {
          "type": "Identifier",
          "name": "foo"
        },
        "init": {
          "type": "Literal",
          "value": "bar"
        }
      }
      ],
      "kind": "var"
    }
  ]
}
```

I/ { Parse
Traverse

O { Manipulate
Generate code


AST Explorer
Save
Fork
JavaScript
acorn
Parser Settings
Transform
Help
/**
* Paste or drop some JavaScript here and explore
* the syntax tree created by chosen parser.
* You can use all the cool new features from ES6
* and even more. Enjoy!
*/
let tips = [
"Click on any AST node with a '+' to expand it",
"Hovering over a node highlights the \
corresponding part in the source code",
"Shift click on an AST node expands the whole substr
];
function printTips() {
tips.forEach((tip, i) => console.log(`Tip ${i}:` + t
}
Tree
JSON
Parser: acorn-
Autofocus
Hide methods
Hide empty keys
Hide loca
data
- Program {
type: "Program"
start: 0
end: 476
+ range: [2 elements]
- body: [
+ VariableDeclaration {type, start, end, range,
declarations, ... +1}
+ FunctionDeclaration {type, start, end, range,
... +4}
]
sourceType: "module"
}
astexplorer.net

I/ { Traverse

O {


ESLint

Administrator: C:\Windows\System32\cmd.exe

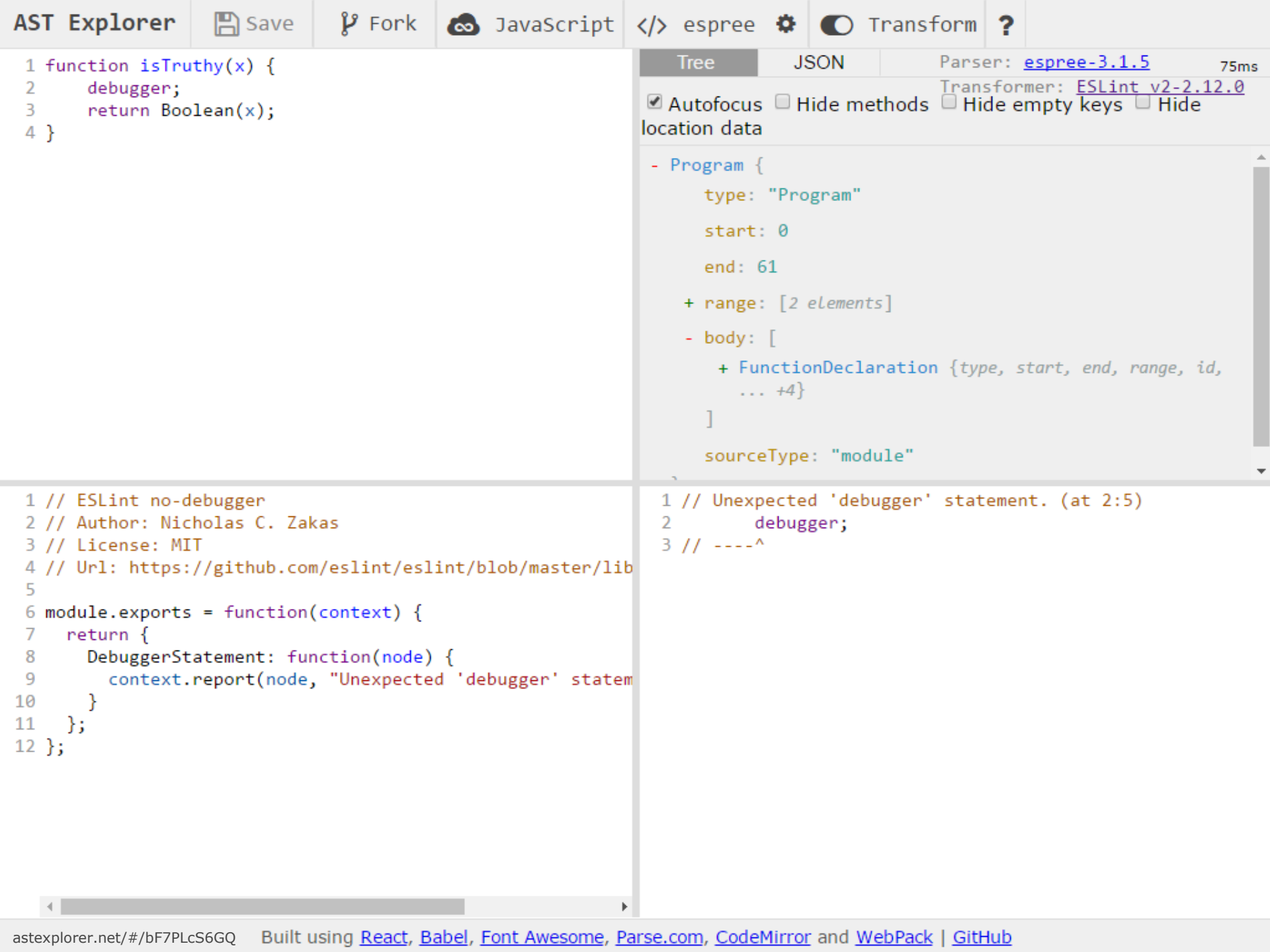
```
C:\dev\temp>eslint react.js

C:\dev\temp\react.js
     4:10   error  Missing space before function parentheses                space-before-function-paren
     4:13   error  Requires a space after '{'                               block-spacing
     4:13   error  Missing space before opening brace                       space-before-blocks
     4:14   error  Expected space(s) after "if"                             keyword-spacing
     4:17   error  Strings must use singlequote                             quotes
     4:17   error  Expected literal to be on the right side of ==           yoda
     4:27   error  Expected '===' and instead saw '=='                      eqeqeq
     4:27   error  Infix operators must be spaced                           space-infix-ops
     4:46   error  'bootstrap' is not defined                               no-undef
     4:56   error  Strings must use singlequote                             quotes
     4:63   error  A space is required after ','                            comma-spacing
     4:66   error  Missing whitespace after semicolon                       semi-spacing
     4:72   error  Expected space(s) after "if"                             keyword-spacing
     4:75   error  Expected literal to be on the right side of ==           yoda
     4:75   error  Strings must use singlequote                             quotes
     4:83   error  Expected '===' and instead saw '=='                      eqeqeq
     4:83   error  Infix operators must be spaced                           space-infix-ops
     4:114  error  Infix operators must be spaced                           space-infix-ops
     4:118  error  Missing whitespace after semicolon                       semi-spacing
     4:124  error  Expected space(s) after "if"                             keyword-spacing
     4:127  error  Strings must use singlequote                             quotes
     4:127  error  Expected literal to be on the right side of ==           yoda
     4:137  error  Infix operators must be spaced                           space-infix-ops
     4:137  error  Expected '===' and instead saw '=='                      eqeqeq
     4:152  error  Infix operators must be spaced                           space-infix-ops
     4:154  error  'define' is not defined                                  no-undef
     4:165  error  'define' is not defined                                  no-undef
     4:174  error  Missing whitespace after semicolon                       semi-spacing
     4:180  error  Expected space(s) after "if"                             keyword-spacing
     4:183  error  Strings must use singlequote                             quotes
     4:183  error  Expected literal to be on the right side of !=           yoda
     4:194  error  Infix operators must be spaced                           space-infix-ops
     4:194  error  Expected '!==' and instead saw '!='                      eqeqeq
     4:207  error  Requires a space after '{'                               block-spacing
     4:207  error  Missing space before opening brace                       space-before-blocks
     4:208  error  Expected space(s) after "if"                             keyword-spacing
     4:212  error  'ses' is not defined                                     no-undef
     4:221  error  Expected space(s) before "return"                        keyword-spacing
     4:227  error  Missing whitespace after semicolon                       semi-spacing
     4:228  error  'ses' is not defined                                     no-undef
     4:241  error  Infix operators must be spaced                           space-infix-ops
     4:243  error  Requires a space before '}'                              block-spacing
     4:244  error  Expected space(s) before "else"                          keyword-spacing
     4:244  error  Expected space(s) after "else"                           keyword-spacing
     4:248  error  Strings must use singlequote                             quotes
     4:248  error  Expected literal to be on the right side of !=           yoda
     4:259  error  Infix operators must be spaced                           space-infix-ops
     4:259  error  Expected '!==' and instead saw '!='                      eqeqeq
     4:274  error  Infix operators must be spaced                           space-infix-ops
     4:287  error  Infix operators must be spaced                           space-infix-ops
     4:304  error  Infix operators must be spaced                           space-infix-ops
     4:308  error  Requires a space before '}'                              block-spacing
     4:319  error  Missing space before function parentheses                space-before-function-paren
     4:321  error  Missing space before opening brace                       space-before-blocks
```

DEMO 1\3

ESLint

and then...

Questions?


AST Explorer
Save
Fork
JavaScript
espree
Transform
function isTruthy(x) {
debugger;
return Boolean(x);
}
Tree
JSON
Parser: espree-3.1.5
75ms
Transformer: ESLint v2-2.12.0
Autofocus
Hide methods
Hide empty keys
Hide location data
- Program {
type: "Program"
start: 0
end: 61
+ range: [2 elements]
- body: [
+ FunctionDeclaration {type, start, end, range, id, ... +4}
]
sourceType: "module"
// ESLint no-debugger
// Author: Nicholas C. Zakas
// License: MIT
// Url: https://github.com/eslint/eslint/blob/master/lib
module.exports = function(context) {
return {
DebuggerStatement: function(node) {
context.report(node, "Unexpected 'debugger' statem
}
};
};
// Unexpected 'debugger' statement. (at 2:5)
debugger;
// ----^
astexplorer.net/#/bF7PLcS6GQ
Built using React, Babel, Font Awesome, Parse.com, CodeMirror and WebPack | GitHub

```
/*eslint-env browser */

console.log('holy metal batman!');

DCComics.Characters.Batman.speak();
```

Tree | JSON | Parser: espree-3.1.5 | 76ms | Transformer: ESLint v2-2.12.0

Autofocus | Hide methods | Hide empty keys | Hide location data

```
- Program {
    type: "Program"
    start: 0
    end: 96
  + range: [2 elements]
  - body: [
    + ExpressionStatement {type, start, end, range, expression}
    + ExpressionStatement {type, start, end, range, expression}
    ]
```

```
// ESLint no-undef
// Author: Mark Macdonald
// License: MIT
// Url: https://github.com/eslint/eslint/blob/master/l

module.exports = function(context) {
  var options = context.options[0];
  var considerTypeOf = options && options.typeof === t

  return {
    "Program:exit": function(/* node */) {
      var globalScope = context.getScope();

      globalScope.through.forEach(function(ref) {
        var identifier = ref.identifier;

        if (!considerTypeOf && hasTypeOfOperator(ident
          return;
```

```
// 'DCComics' is not defined. (at 5:1)
   DCComics.Characters.Batman.speak();
// ^
```

AST Explorer | Save | Fork | JavaScript | espree | Transform | ?

```
1 /*eslint-env browser */
2
3 console.log('holy metal batman!');
4
5 window.DCComics.Characters.Batman.speak();
```

Tree | JSON | Parser: espree-3.1.5 | 93ms | Transformer: ESLint v2-2.12.0

Autofocus | Hide methods | Hide empty keys | Hide location data

```
- ExpressionStatement {
    type: "ExpressionStatement"
    start: 61
    end: 103
```

```
1  // Custom ESLint no-restricted-global-extend
2  // Author: Yonatan Mevorach
3  // License: ISC
4  // Url: https://github.com/cowchimp/eslint-plugin-piggyb
5
6  module.exports = function(context) {
7    var globalScope;
8
9    return {
10     "Program": function() {
11       globalScope = context.getScope();
12     },
13
14     "MemberExpression": function(node) {
15       if(node.object.type === "Identifier" && node.objec
16         context.report(node, "'{{propertyName}}' piggyba
17       }
18     }
19   };
20
21   function isGlobalProperty(node) {
22     return globalScope.variables.some(function(variable)
23       return variable.name == node.name;
24     });
25   }
26 };
```

```
1 // 'DCComics' piggybacks on 'window' to extend the globa
2    window.DCComics.Characters.Batman.speak();
3 // ^
```

I/ {

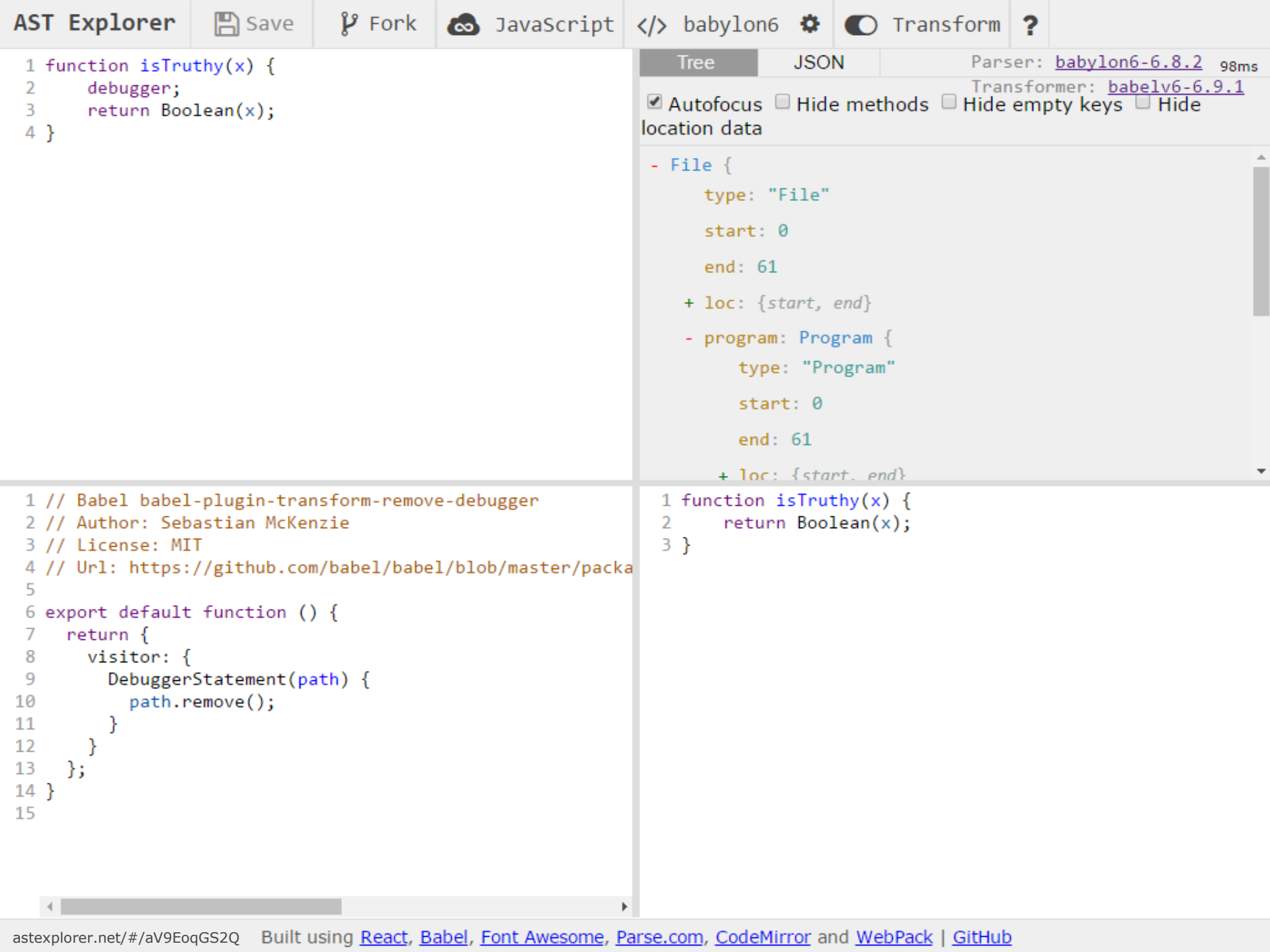
O { Manipulate

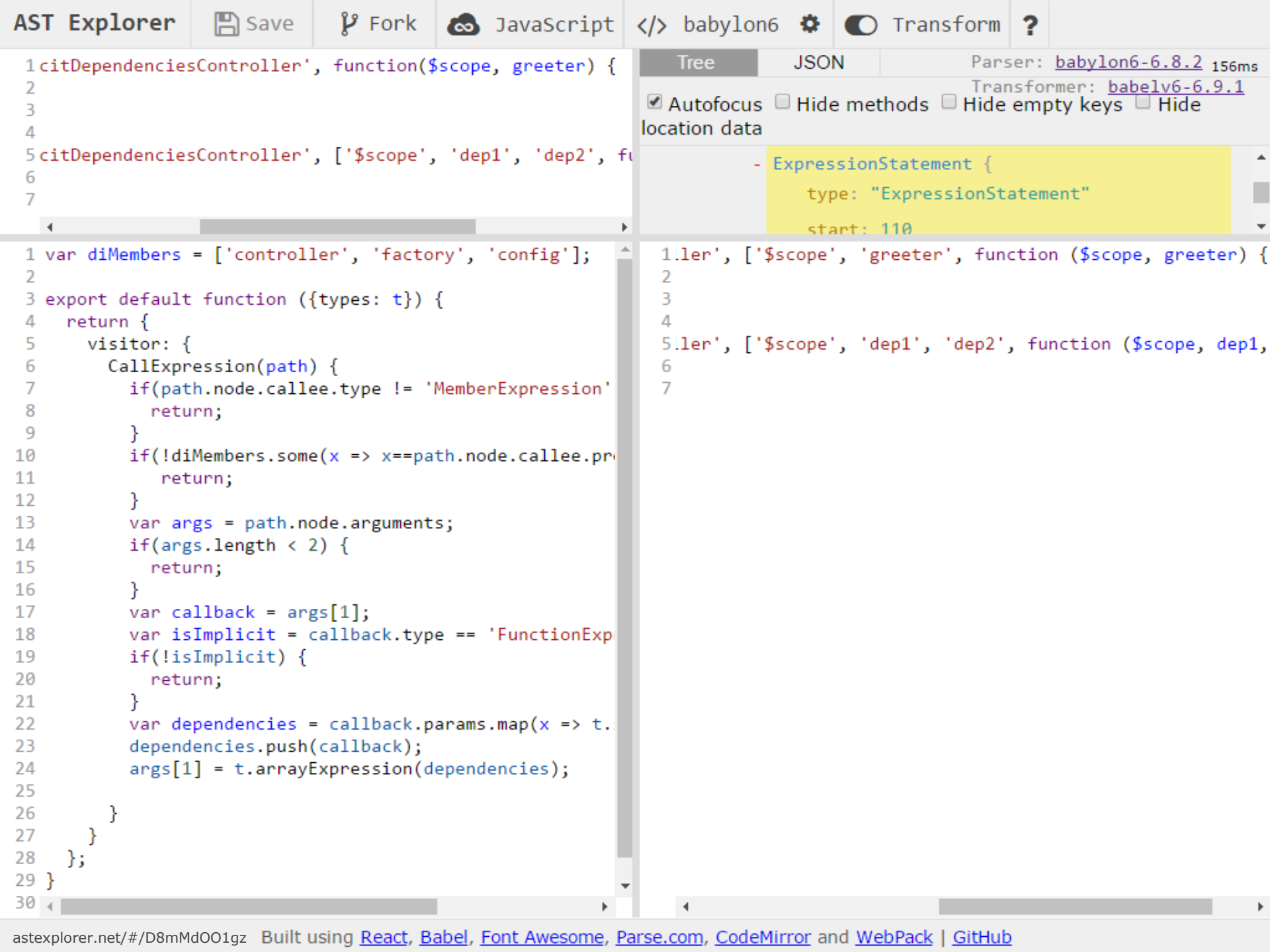

BABEL

DEMO 2\3

and then...

Questions?


AST Explorer
Save
Fork
JavaScript
babylon6
Transform
function isTruthy(x) {
debugger;
return Boolean(x);
}
Tree
JSON
Parser: babylon6-6.8.2 98ms
Transformer: babelv6-6.9.1
Autofocus
Hide methods
Hide empty keys
Hide location data
- File {
type: "File"
start: 0
end: 61
+ loc: {start, end}
- program: Program {
type: "Program"
start: 0
end: 61
+ loc: {start, end}
// Babel babel-plugin-transform-remove-debugger
// Author: Sebastian McKenzie
// License: MIT
// Url: https://github.com/babel/babel/blob/master/packa
export default function () {
return {
visitor: {
DebuggerStatement(path) {
path.remove();
}
}
};
}
function isTruthy(x) {
return Boolean(x);
}
astexplorer.net/#/aV9EoqGS2Q
Built using React, Babel, Font Awesome, Parse.com, CodeMirror and WebPack | GitHub


AST Explorer
Save
Fork
JavaScript
babylon6
Transform
citDependenciesController', function($scope, greeter) {
citDependenciesController', ['$scope', 'dep1', 'dep2', fu
Tree
JSON
Parser: babylon6-6.8.2 156ms
Transformer: babelv6-6.9.1
Autofocus
Hide methods
Hide empty keys
Hide location data
- ExpressionStatement {
type: "ExpressionStatement"
start: 110
var diMembers = ['controller', 'factory', 'config'];
export default function ({types: t}) {
return {
visitor: {
CallExpression(path) {
if(path.node.callee.type != 'MemberExpression'
return;
}
if(!diMembers.some(x => x==path.node.callee.pr
return;
}
var args = path.node.arguments;
if(args.length < 2) {
return;
}
var callback = args[1];
var isImplicit = callback.type == 'FunctionExp
if(!isImplicit) {
return;
}
var dependencies = callback.params.map(x => t.
dependencies.push(callback);
args[1] = t.arrayExpression(dependencies);
}
}
};
}
ler', ['$scope', 'greeter', function ($scope, greeter) {
ler', ['$scope', 'dep1', 'dep2', function ($scope, dep1,
astexplorer.net/#/D8mMdOO1gz
Built using React, Babel, Font Awesome, Parse.com, CodeMirror and WebPack | GitHub

Generate code

jscodeshift

	Babel	jscodeshift
Input ➞ Output	Javascript ➞ Javascript	Javascript ➞ Javascript
Plugin-based	Yes	Yes
Building Block	AST-to-AST Transform function	AST-to-AST Transform function
Endorsed by	Facebook	Facebook

	Babel	jscodeshift
Input ➞ Output	Javascript ➞ Javascript	Javascript ➞ Javascript
Plugin-based	Yes	Yes
Building Block	AST-to-AST Transform function	AST-to-AST Transform function
Endorsed by	Facebook	Facebook
Goal	Compiler	"Codemod" (refactoring)
Example Use	ES6 ➞ ES5	ES5 ➞ ES6
Output level	Lower level than source	Same level as source
Output readability	Best-effort (boilerplate code)	High
Usage frequency	All the time	One-time

DEMO 3\3

`jscodeshift`

and then...

Questions?

```
(function() {
  if(true) {
    var colorOfSky = 'blue';
  }

  console.log(colorOfSky);
})();
```

Tree | JSON | Parser: recast-0.11.6 440ms | Transformer: jscodeshift-0.3.20

Autofocus | Hide methods | Hide empty keys | Hide location data

```
- File {
  - program: Program {
      type: "Program"
      start: 0
      end: 95
    + loc: {start, end, lines, indent}
      sourceType: "module"
    - body: [
      - ExpressionStatement {
          type: "ExpressionStatement"
```

```
// WARNING: Contrived example of a transformation
// that will produce faulty code

module.exports = function(file, api) {
    var j = api.jscodeshift;
    var root = j(file.source);

    root.find(j.VariableDeclaration, { kind: 'var' }).fo
        var letStatement = j.variableDeclaration('let',
        return j(p).replaceWith(letStatement);
    });

    return root.toSource();
};
```

```
(function() {
  if(true) {
    let colorOfSky = 'blue';
  }

  console.log(colorOfSky);
})();
```

AST Explorer | Save | Fork | JavaScript | recast | Transform | ?

```
(function() {
  if(true) {
    var colorOfSky = 'blue';
  }

  console.log(colorOfSky);
})();

(function() {
  if(true) {
    var colorOfSky = 'blue';
    console.log(colorOfSky);
  }
})();
```

Tree | JSON | Parser: recast-0.11.6 487ms | Transformer: jscodeshift-0.3.20

Autofocus | Hide methods | Hide empty keys | Hide location data

```
- ExpressionStatement {
    type: "ExpressionStatement"
    start: 0
    end: 95
  + loc: {start, end, lines, indent}
  - expression: CallExpression {
      type: "CallExpression"
      start: 0
      end: 94
    + loc: {start, end, lines, indent}
```

```
// js-codemod no-vars
// License: MIT
// Url: https://github.com/cpojer/js-codemod/blob/mast

export default function(file, api, options) {
  const j = api.jscodeshift;

  const root = j(file.source);

  const TOP_LEVEL_TYPES = [
    'Function',
    'FunctionDeclaration',
    'FunctionExpression',
    'ArrowFunctionExpression',
    'Program',
  ];
  const FOR_STATEMENTS = [
    'ForStatement',
```

```
(function() {
  if(true) {
    var colorOfSky = 'blue';
  }

  console.log(colorOfSky);
})();

(function() {
  if(true) {
    const colorOfSky = 'blue';
    console.log(colorOfSky);
  }
})();
```

github.com/cowchimp/awesome-ast

blog.cowchimp.com

@cowchimp

yonatan@sears.co.il