node JS® Interactive

NORTH AMERICA

Austin, Texas
November 28–December 2, 2016

node JS Interactive

NORTH AMERICA

Austin, Texas

November 28–December 2, 2016

Quit writing everything yourself

Evan Lucas, Help.com

https://github.com/evanlucas

https://twitter.com/evanhlucas

About

- Node Engineer at Help.com
- Node.js Core Technical Committee (CTC) and release team member

Input Validation

- HTTP request, response, querystring param, and header validation
- Function argument validation
- Message format validation

How can we automate this?

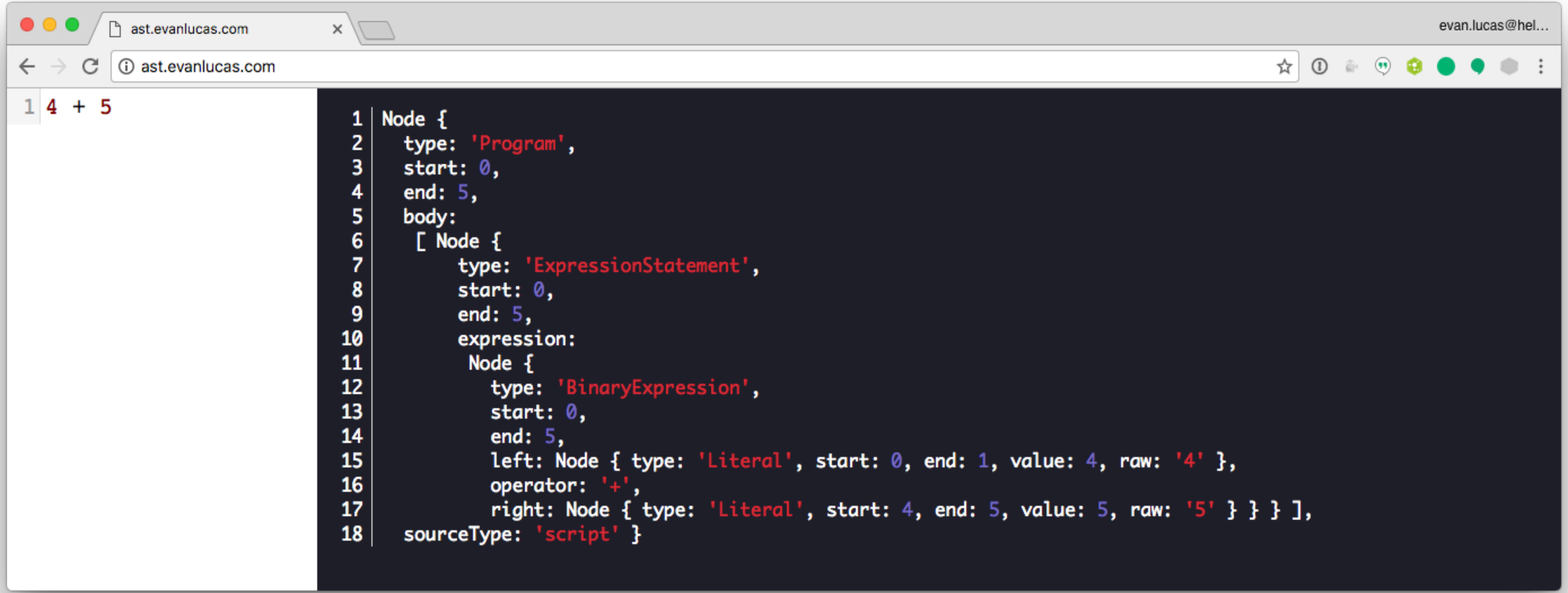
- Validate an object’s properties for different types
- Enforce how something is written (configuration vs code)

escodegen

“Escodegen (escodegen) is an ECMAScript (also popularly known as JavaScript) code generator from Mozilla's Parser API AST.”

https://github.com/estools/escodegen

Abstract Syntax Tree (AST)



```
4 + 5
```

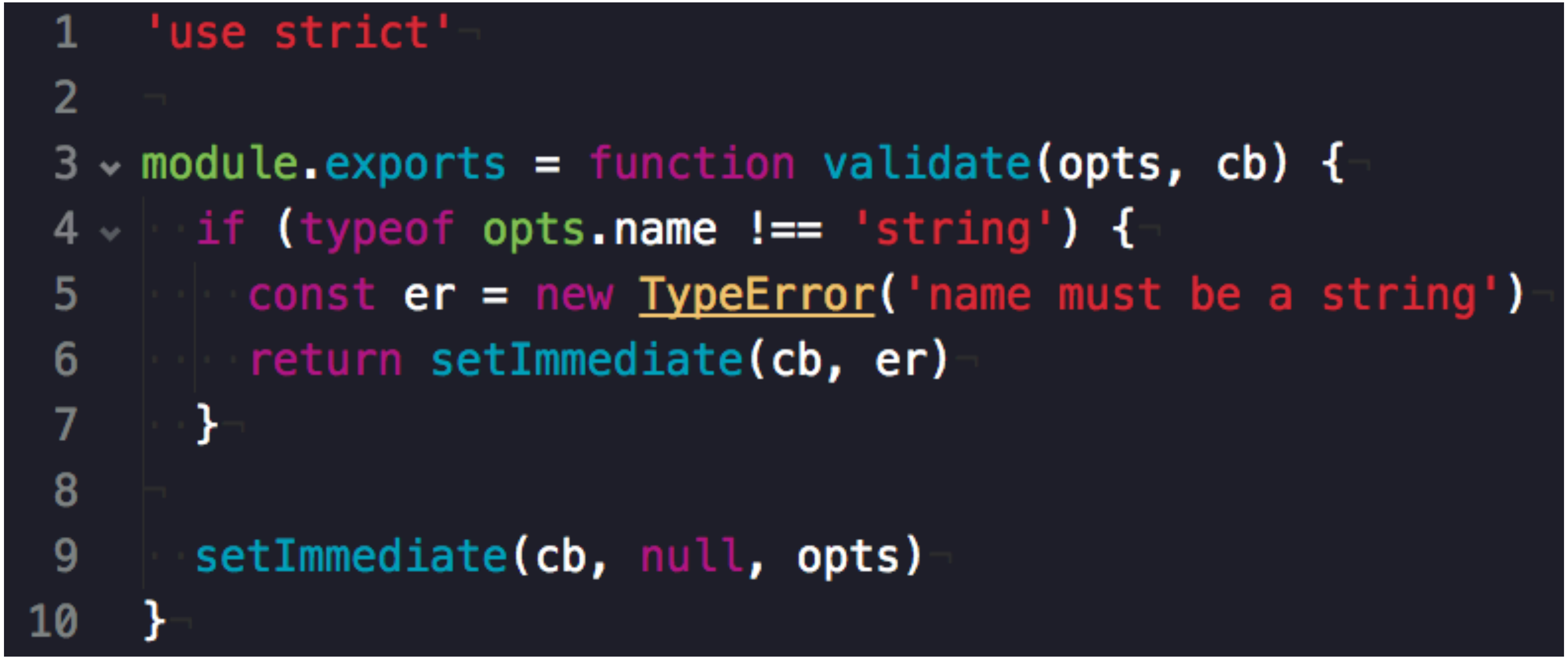
```
Node {
  type: 'Program',
  start: 0,
  end: 5,
  body:
   [ Node {
       type: 'ExpressionStatement',
       start: 0,
       end: 5,
       expression:
        Node {
          type: 'BinaryExpression',
          start: 0,
          end: 5,
          left: Node { type: 'Literal', start: 0, end: 1, value: 4, raw: '4' },
          operator: '+',
          right: Node { type: 'Literal', start: 4, end: 5, value: 5, raw: '5' } } } ],
  sourceType: 'script' }
```

How can we make this easier?

```
npm install @helpdotcom/build-ast
```

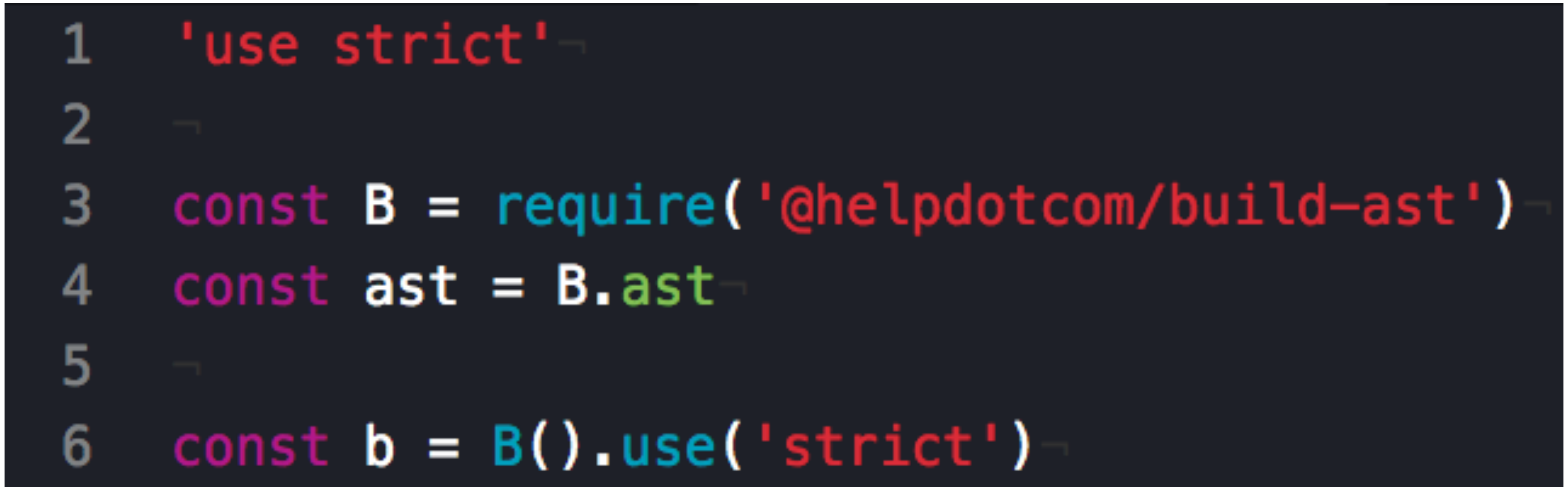
https://github.com/helpdotcom/build-ast

This is what we want



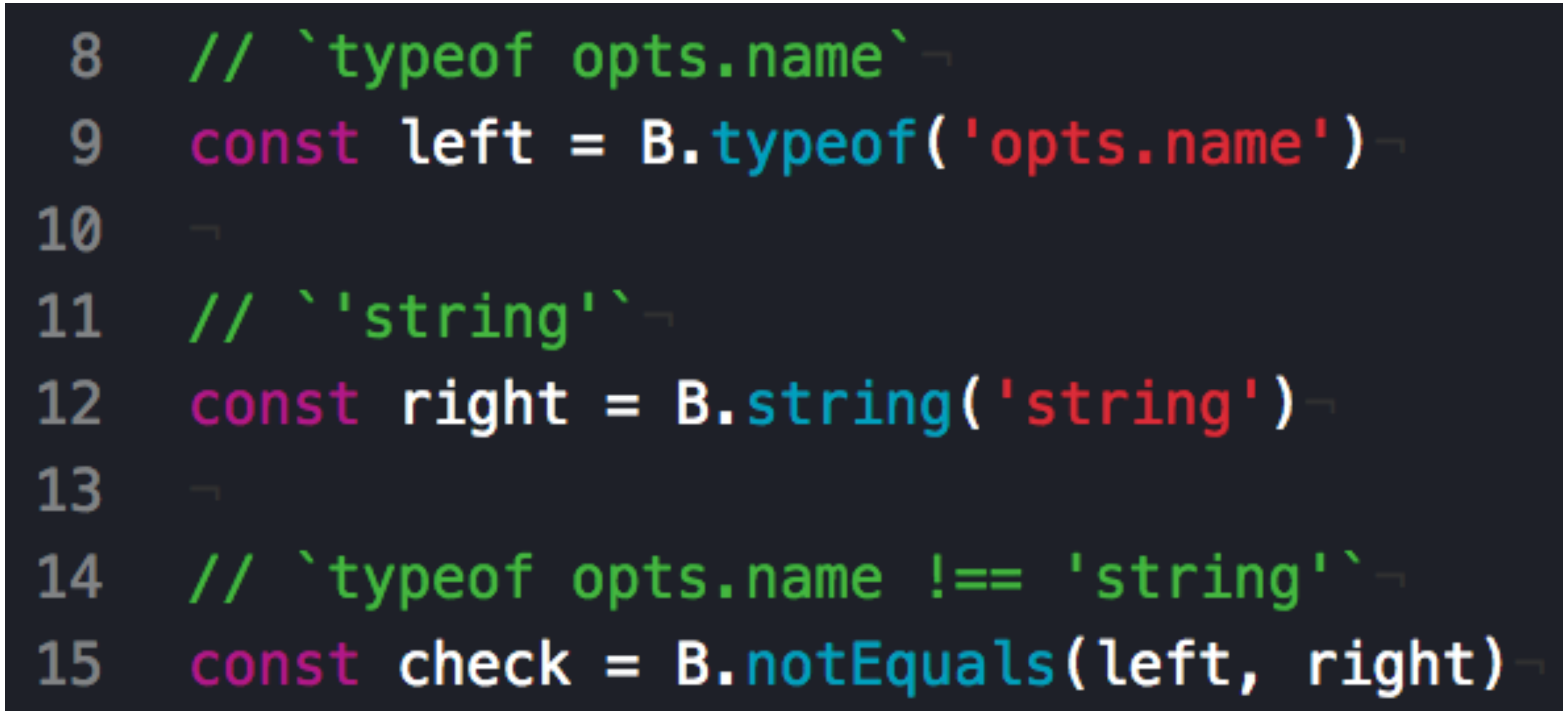
```
'use strict'

module.exports = function validate(opts, cb) {
  if (typeof opts.name !== 'string') {
    const er = new TypeError('name must be a string')
    return setImmediate(cb, er)
  }

  setImmediate(cb, null, opts)
}
```

@helpdotcom/build-ast



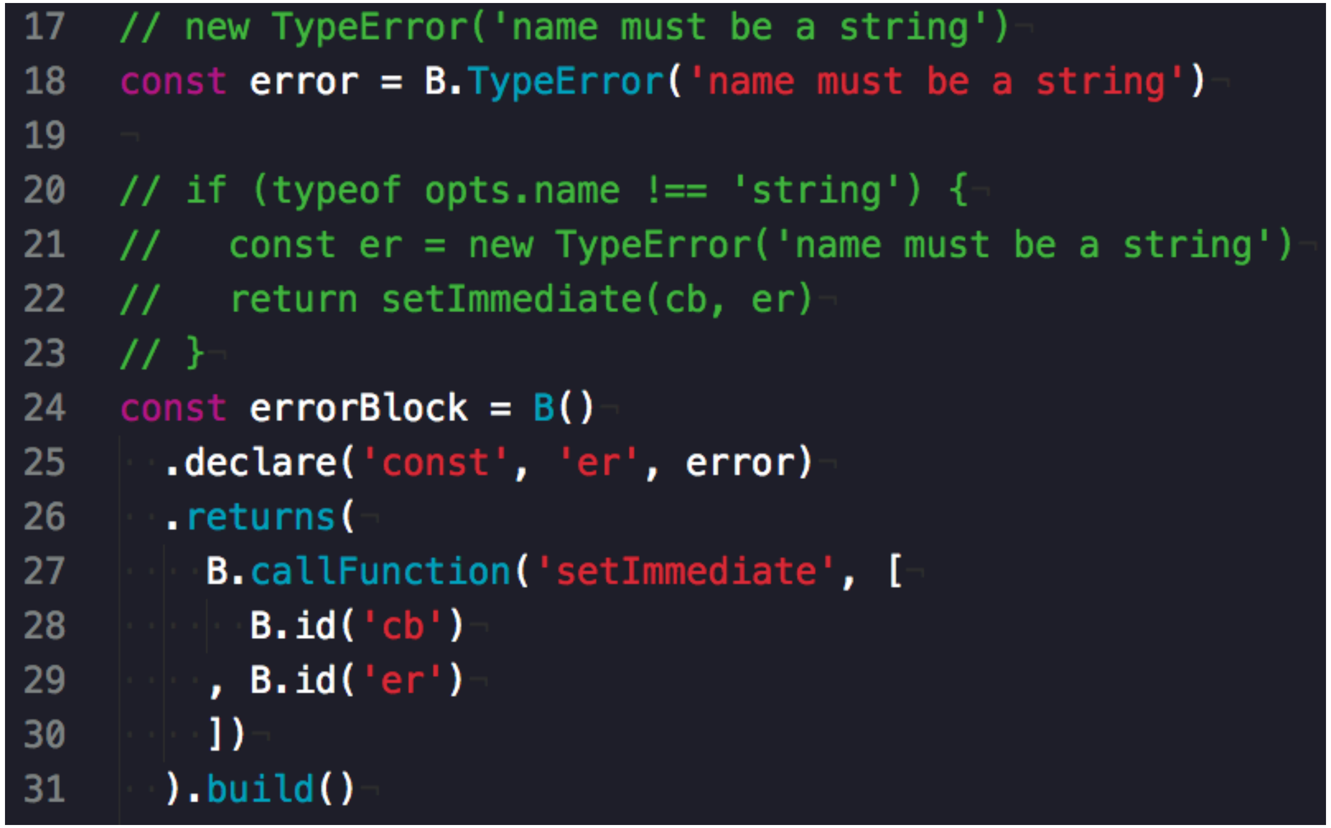
```
'use strict'

const B = require('@helpdotcom/build-ast')
const ast = B.ast

const b = B().use('strict')
```

Example



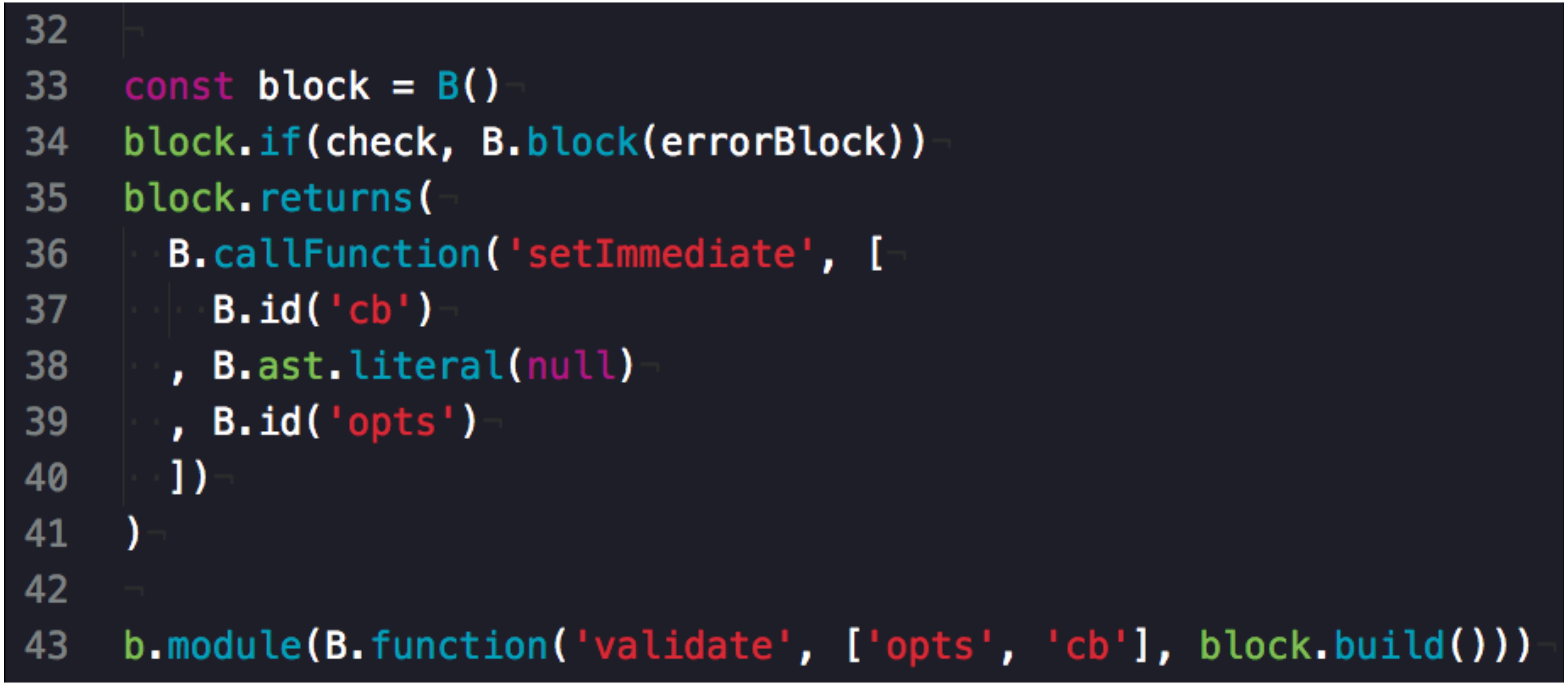
```
// `typeof opts.name`
const left = B.typeof('opts.name')

// `'string'`
const right = B.string('string')

// `typeof opts.name !== 'string'`
const check = B.notEquals(left, right)
```

Example



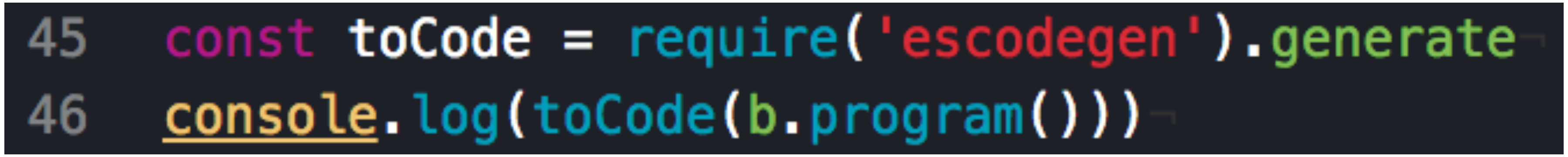
```
// new TypeError('name must be a string')
const error = B.TypeError('name must be a string')

// if (typeof opts.name !== 'string') {
//   const er = new TypeError('name must be a string')
//   return setImmediate(cb, er)
// }
const errorBlock = B()
  .declare('const', 'er', error)
  .returns(
    B.callFunction('setImmediate', [
      B.id('cb')
    , B.id('er')
    ])
  ).build()
```

Example



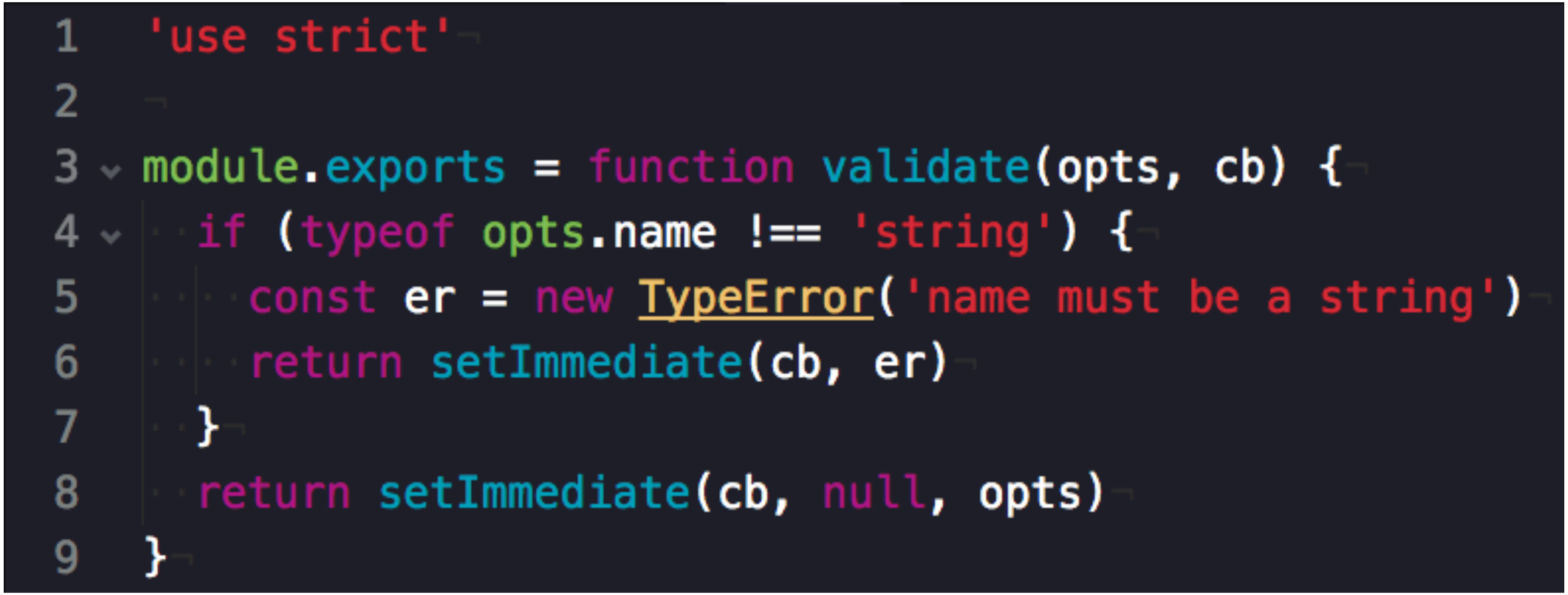
```

const block = B()
block.if(check, B.block(errorBlock))
block.returns(
  B.callFunction('setImmediate', [
    B.id('cb')
  , B.ast.literal(null)
  , B.id('opts')
  ])
)

b.module(B.function('validate', ['opts', 'cb'], block.build()))
```

Example



```javascript
const toCode = require('escodegen').generate
console.log(toCode(b.program()))
```

Example - Output



```
'use strict'

module.exports = function validate(opts, cb) {
  if (typeof opts.name !== 'string') {
    const er = new TypeError('name must be a string')
    return setImmediate(cb, er)
  }
  return setImmediate(cb, null, opts)
}
```

Thanks!

@evanhlucas

evanlucas