



# BIONODE.IO

Tutorial

# LIVE CODING!

Module to get data from European Variation Archive EMBL-EBI

```
fish /home/bmpvieira/bionode-eva — docker • -bash — 73x23
~/bionode-eva ▶ node index.js | json | head -n 21
{
  "apiVersion": "v1",
  "warning": "",
  "error": "",
  "queryOptions": {},
  "response": [
    {
      "time": 0,
      "dbTime": 9,
      "numResults": 15,
      "numTotalResults": 15,
      "resultType": "com.mongodb.BasicDBObject",
      "result": [
        {
          "studyId": "PRJEB8661",
          "studyName": "The Exome Aggregation Consortium (ExAC) v0.3"
        },
        {
          "studyId": "PRJEB6042",
          "studyName": "GEUVADIS: Genetic European Variation in Disease"
        }
      ]
    }
  ]
}
```

~/bionode-eva ▶

# PROCESS DATA IN CHUNKS

# PROCESS DATA IN CHUNKS





# PROCESS DATA IN CHUNKS

request()



# PROCESS DATA IN CHUNKS

request()



Binary data  
Buffer class  
chunks of 16kb

# PROCESS DATA IN CHUNKS

request()

A green icon for a Readable Stream, consisting of two overlapping rectangles. The front rectangle is horizontal and the back one is vertical, both in a vibrant green color with a slight gradient and a dark green border.

**Readable  
Stream**

A green icon for a Writable Stream, consisting of two overlapping rectangles. The front rectangle is vertical and the back one is horizontal, both in a vibrant green color with a slight gradient and a dark green border.

**Writable  
Stream**

Binary data  
Buffer class  
chunks of 16kb

# PROCESS DATA IN CHUNKS

`request()`



Binary data  
Buffer class  
chunks of 16kb

`process  
.stdout`





```
var url = require('url')
var request = require('request')

var urlObject = {
  protocol: 'http',
  host: 'www.ebi.ac.uk',
  pathname: '/eva/webservices/rest/v1/meta/studies/list',
  search: '?species=hsapiens_grch37'
}

var urlString = url.format(urlObject)

request(urlString).pipe(process.stdout)
```

## Live Coding Solution

### step 1 - get JSON

# PROCESS DATA IN CHUNKS

`request()`

**Readable  
Stream**

Binary data  
Buffer class  
chunks of 16kb

`process  
.stdout`

**Writable  
Stream**

# PROCESS DATA IN CHUNKS

`request()`



Binary data  
Buffer class  
chunks of 16kb

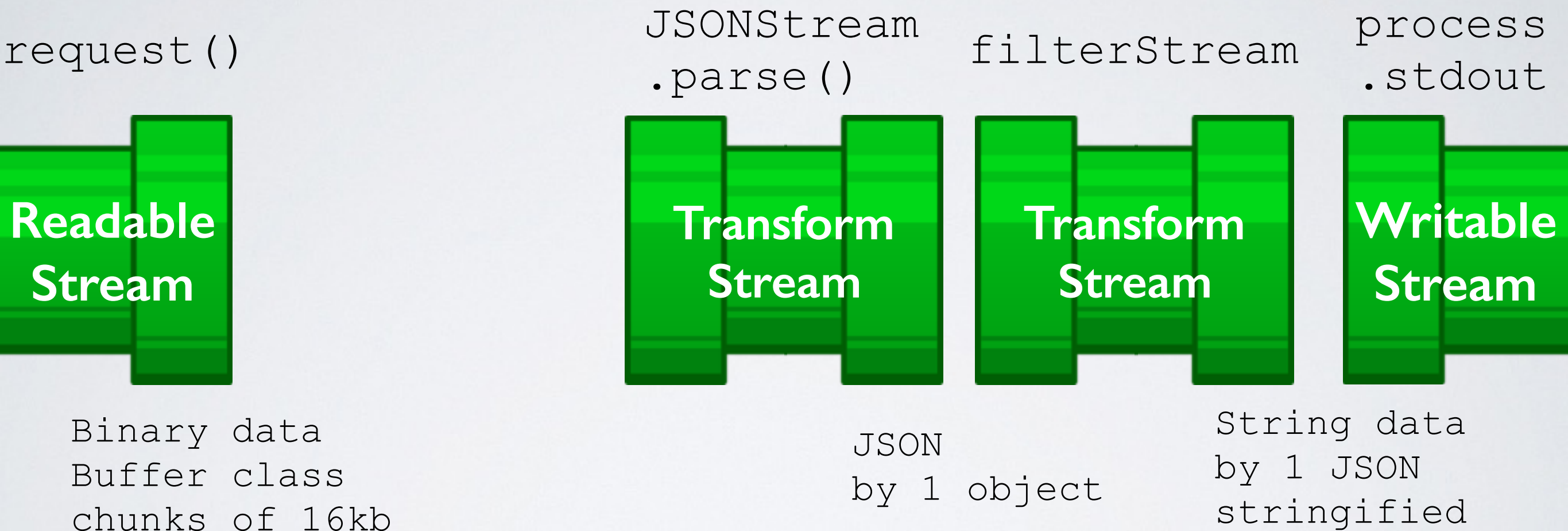
`process  
.stdout`



# PROCESS DATA IN CHUNKS

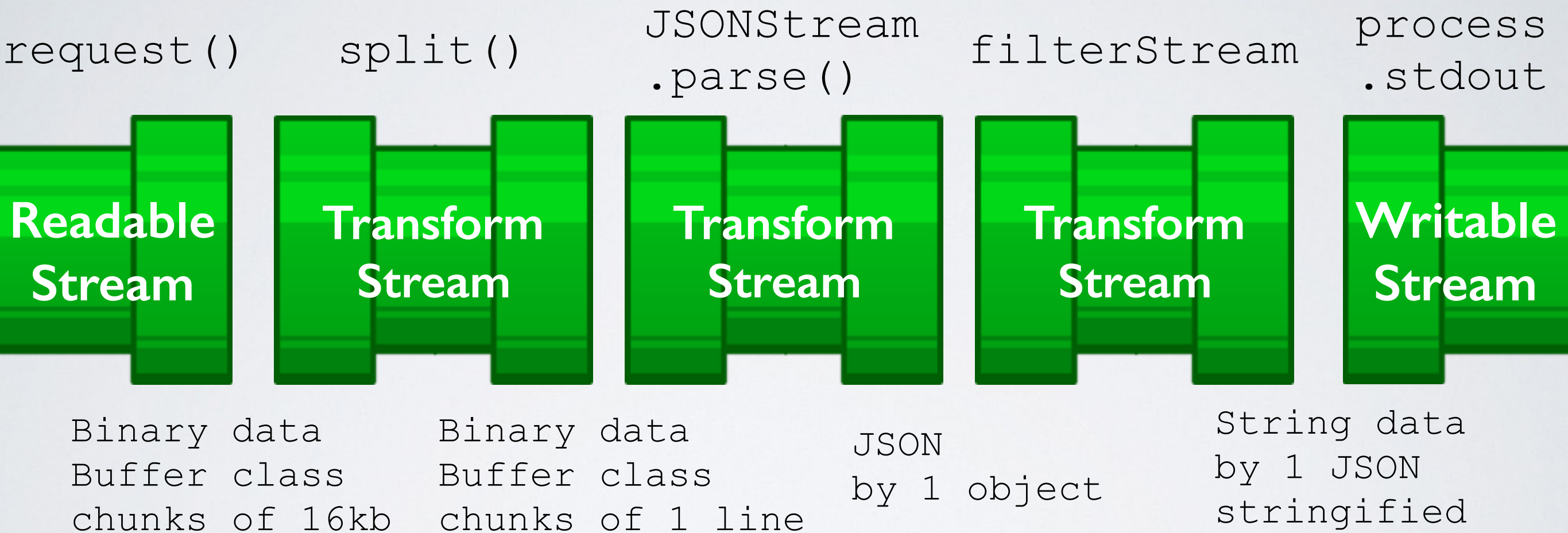


# PROCESS DATA IN CHUNKS





# PROCESS DATA IN CHUNKS



## Live Coding Solution step 2 - parse JSON

```
var url = require('url')
var split = require('split2')
var JSONstream = require('JSONstream')
var request = require('request')

var urlObject = {
  protocol: 'http',
  host: 'www.ebi.ac.uk',
  pathname: '/eva/webservices/rest/v1/meta/studies/list',
  search: '?species=hsapiens_grch37'
}

var filterStream = through.obj(filter)

function filter(object, encoding, callback) {
  this.push(JSON.stringify(object))
  callback()
}

var urlString = url.format(urlObject)

request(urlString)
  .pipe(split())
  .pipe(JSONstream.parse())
  .pipe(filterStream)
  .pipe(process.stdout)
```

## Live Coding Solution step 3 - filter JSON

```
var os = require('os')
var url = require('url')
var miss = require('mississippi')
var split = require('split2')
var JSONstream = require('JSONstream')
var request = require('request')

var urlObject = {
  protocol: 'http',
  host: 'www.ebi.ac.uk',
  pathname: '/eva/webservices/rest/v1/meta/studies/list',
  search: '?species=hsapiens_grch37'
}

var urlString = url.format(urlObject)

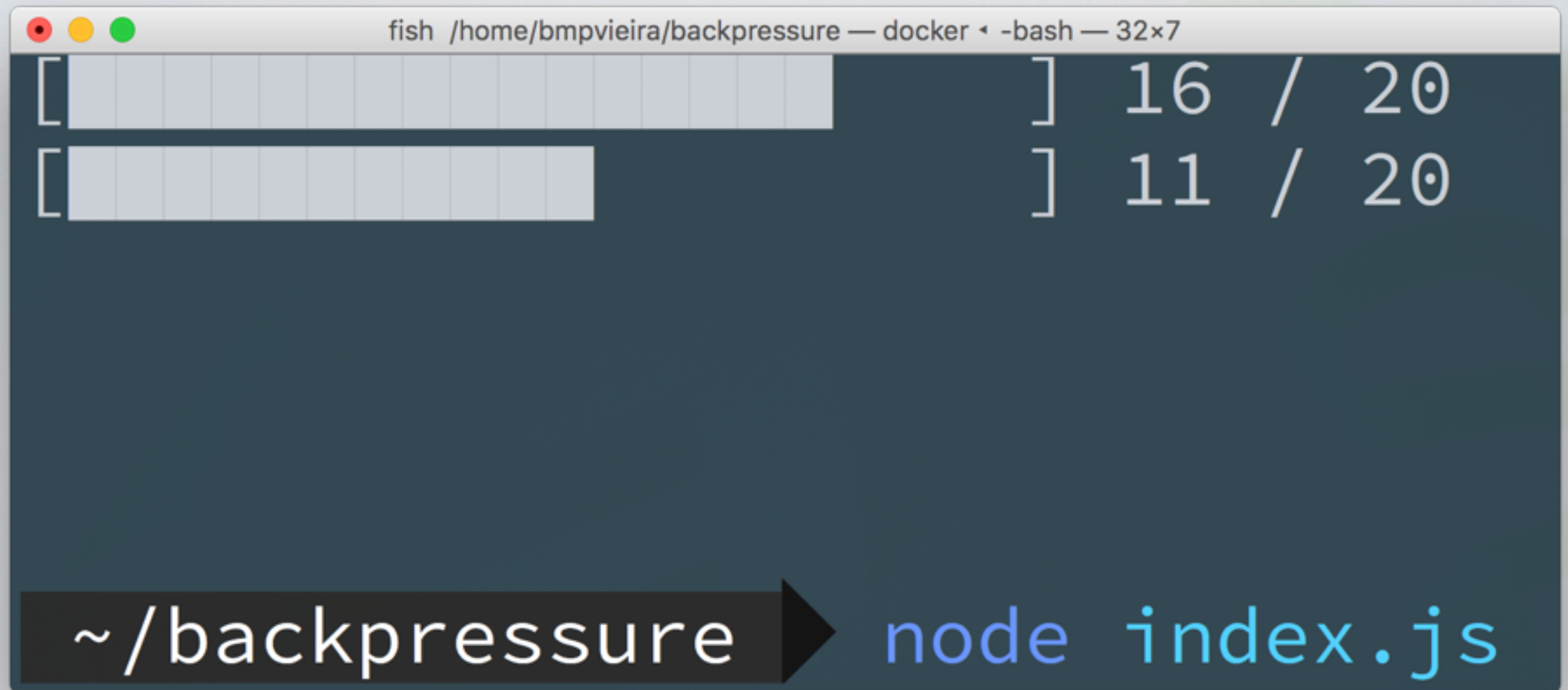
var filterStream = through.obj(filter)

function filter(object, encoding, callback) {
  var self = this
  var results = object.response[0].result
  results.forEach(filterAndPush())
  function filterAndPush(result) {
    if (result.studyName.match('1000 Genomes')) {
      self.push(JSON.stringify(result) + os.EOL)
    }
  }
  callback()
}

request(urlString)
  .pipe(split())
  .pipe(JSONstream.parse())
  .pipe(filterStream)
  .pipe(process.stdout)
```

# LIVE EXAMPLE

Show backpressure in action



A terminal window titled "fish /home/bmpvieira/backpressure — docker — -bash — 32x7" displays two data streams. The first stream is a long horizontal bar composed of 16 light blue segments, followed by a closing bracket and the text "16 / 20". The second stream is a shorter horizontal bar composed of 11 light blue segments, followed by a closing bracket and the text "11 / 20". At the bottom of the terminal, a dark blue banner contains the text "~ /backpressure" in white, followed by a black right-pointing arrow and the text "node index.js" in light blue.

```
fish /home/bmpvieira/backpressure — docker — -bash — 32x7  
[ 16 / 20  
[ 11 / 20  
~ /backpressure ➡ node index.js
```



```
var _ = require('lodash')
var through = require('through2')
var exec = require('child_process').exec;
var streamify = require('stream-array')
var multimeter = require('multimeter');
var multi = multimeter(process);

process.stdout.write('\033c'); // Clear the console

var jobs = 20

var progress = {
  fastStream: 1,
  slowStream: 1
}

var fastStreamProgressBar = multi(0, 1, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})

var slowStreamProgressBar = multi(0, 2, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```



```
var _ = require('lodash')
var through = require('through2')
var exec = require('child_process').exec;
var streamify = require('stream-array')
var multimeter = require('multimeter');
var multi = multimeter(process);
```

```
process.stdout.write('\033c'); // Clear the console
```

```
var jobs = 20
```

```
var progress = {
  fastStream: 1,
  slowStream: 1
}
```

```
var fastStreamProgressBar = multi(0, 1, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var slowStreamProgressBar = multi(0, 2, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var _ = require('lodash')
var through = require('through2')
var exec = require('child_process').exec;
var streamify = require('stream-array')
var multimeter = require('multimeter');
var multi = multimeter(process);
```

```
process.stdout.write('\033c'); // Clear the console
```

```
var jobs = 20
```

```
var progress = {
  fastStream: 1,
  slowStream: 1
}
```

```
var fastStreamProgressBar = multi(0, 1, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var slowStreamProgressBar = multi(0, 2, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var _ = require('lodash')
var through = require('through2')
var exec = require('child_process').exec;
var streamify = require('stream-array')
var multimeter = require('multimeter');
var multi = multimeter(process);
```

```
process.stdout.write('\033c'); // Clear the console
```

```
var jobs = 20
```

```
var progress = {
  fastStream: 1,
  slowStream: 1
}
```

```
var fastStreamProgressBar = multi(0, 1, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var slowStreamProgressBar = multi(0, 2, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```



```
var _ = require('lodash')
var through = require('through2')
var exec = require('child_process').exec;
var streamify = require('stream-array')
var multimeter = require('multimeter');
var multi = multimeter(process);
```

```
process.stdout.write('\033c'); // Clear the console
```

```
var jobs = 20
```

```
var progress = {
  fastStream: 1,
  slowStream: 1
}
```

```
var fastStreamProgressBar = multi(0, 1, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var slowStreamProgressBar = multi(0, 2, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var _ = require('lodash')
var through = require('through2')
var exec = require('child_process').exec;
var streamify = require('stream-array')
var multimeter = require('multimeter');
var multi = multimeter(process);
```

```
process.stdout.write('\033c'); // Clear the console
```

```
var jobs = 20
```

```
var progress = {
  fastStream: 1,
  slowStream: 1
}
```

```
var fastStreamProgressBar = multi(0, 1, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var slowStreamProgressBar = multi(0, 2, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```



```
var _ = require('lodash')
var through = require('through2')
var exec = require('child_process').exec;
var streamify = require('stream-array')
var multimeter = require('multimeter');
var multi = multimeter(process);
```

```
process.stdout.write('\033c'); // Clear the console
```

```
var jobs = 20
```

```
var progress = {
  fastStream: 1,
  slowStream: 1
}
```

```
var fastStreamProgressBar = multi(0, 1, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
var slowStreamProgressBar = multi(0, 2, {
  width : jobs,
  solid : { text : '█' },
  empty : { text : ' ' },
})
```

```
sourceStream = streamify(_.range(jobs))

var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 1', function (err, stdout, stderr) {
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)
    self.push(obj)
    next()
  })
})

var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 2', function (err, stdout, stderr) {
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)
    self.push(obj)
    next()
  })
})

sourceStream.pipe(fastStream).pipe(slowStream)

slowStream.resume()
```

```
sourceStream = streamify(_.range(jobs))
```

```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 1', function (err, stdout, stderr) {  
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 2', function (err, stdout, stderr) {  
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
sourceStream.pipe(fastStream).pipe(slowStream)
```

```
slowStream.resume()
```



```
sourceStream = streamify(_.range(jobs))
```

```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 1', function (err, stdout, stderr) {  
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 2', function (err, stdout, stderr) {  
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
sourceStream.pipe(fastStream).pipe(slowStream)
```

```
slowStream.resume()
```

```
sourceStream = streamify(_.range(jobs))

var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 1', function (err, stdout, stderr) {
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)
    self.push(obj)
    next()
  })
})

var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 2', function (err, stdout, stderr) {
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)
    self.push(obj)
    next()
  })
})

sourceStream.pipe(fastStream).pipe(slowStream)

slowStream.resume()
```



```
sourceStream = streamify(_.range(jobs))

var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 1', function (err, stdout, stderr) {
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)
    self.push(obj)
    next()
  })
})

var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 2', function (err, stdout, stderr) {
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)
    self.push(obj)
    next()
  })
})

sourceStream.pipe(fastStream).pipe(slowStream)

slowStream.resume()
```

```
sourceStream = streamify(_.range(jobs))
```

```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 1', function (err, stdout, stderr) {  
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 2', function (err, stdout, stderr) {  
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
sourceStream.pipe(fastStream).pipe(slowStream)
```

```
slowStream.resume()
```

```
sourceStream = streamify(_.range(jobs))

var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 1', function (err, stdout, stderr) {
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)
    self.push(obj)
    next()
  })
})

var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 2', function (err, stdout, stderr) {
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)
    self.push(obj)
    next()
  })
})

sourceStream.pipe(fastStream).pipe(slowStream)

slowStream.resume()
```



```
sourceStream = streamify(_.range(jobs))
```

```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 1', function (err, stdout, stderr) {  
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 2', function (err, stdout, stderr) {  
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
sourceStream.pipe(fastStream).pipe(slowStream)
```

```
slowStream.resume()
```



```
sourceStream = streamify(_.range(jobs))

var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 1', function (err, stdout, stderr) {
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)
    self.push(obj)
    next()
  })
})

var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 2', function (err, stdout, stderr) {
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)
    self.push(obj)
    next()
  })
})

sourceStream.pipe(fastStream).pipe(slowStream)

slowStream.resume()
```

```
sourceStream = streamify(_.range(jobs))
```

```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 1', function (err, stdout, stderr) {  
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 2', function (err, stdout, stderr) {  
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
sourceStream.pipe(fastStream).pipe(slowStream)
```

```
slowStream.resume()
```

```
sourceStream = streamify(_.range(jobs))
```

```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 1', function (err, stdout, stderr) {  
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 2', function (err, stdout, stderr) {  
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
sourceStream.pipe(fastStream).pipe(slowStream)
```

```
slowStream.resume()
```



```
sourceStream = streamify(_.range(jobs))
```

```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 1', function (err, stdout, stderr) {  
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 2', function (err, stdout, stderr) {  
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
sourceStream.pipe(fastStream).pipe(slowStream)
```

```
slowStream.resume()
```



```
sourceStream = streamify(_.range(jobs))
```

```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 1', function (err, stdout, stderr) {  
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

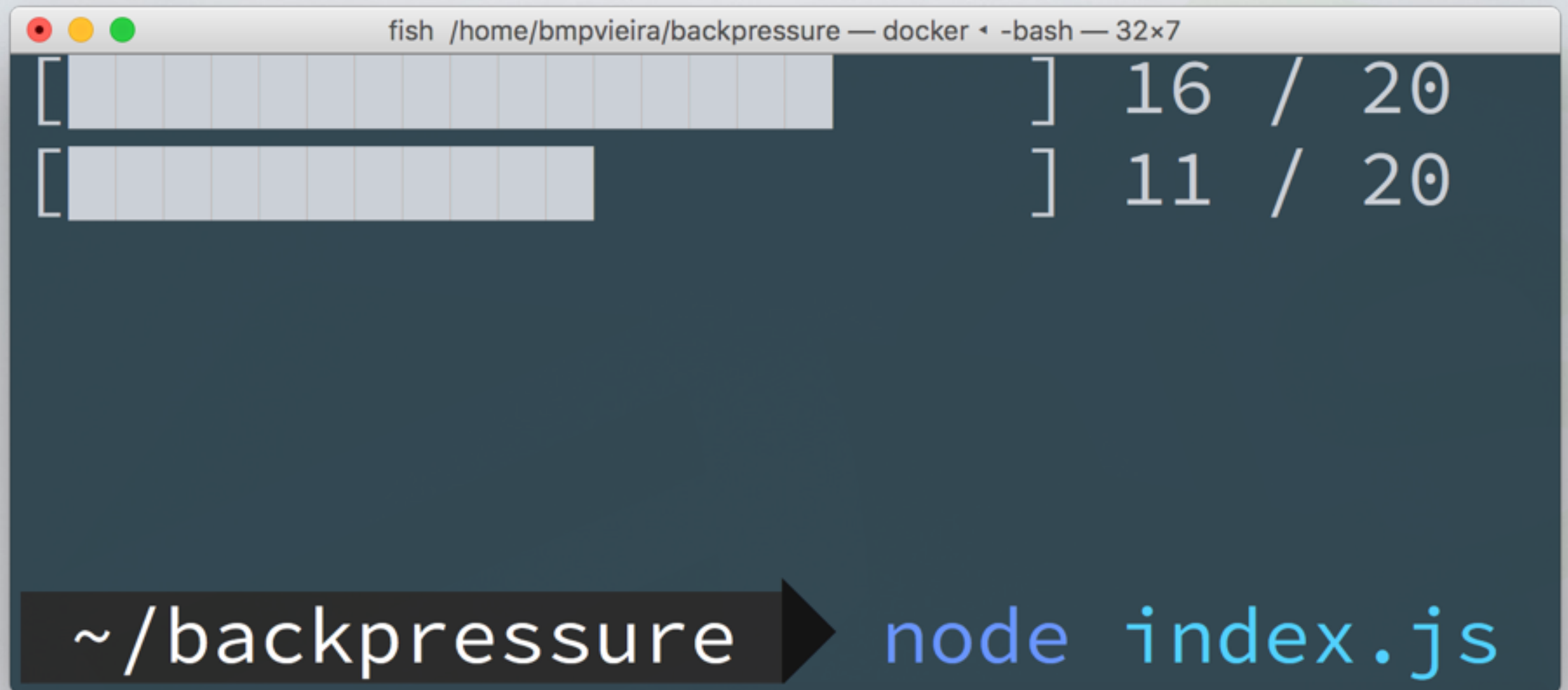
```
var slowStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {  
  var self = this  
  exec('sleep 2', function (err, stdout, stderr) {  
    slowStreamProgressBar.ratio(progress.slowStream++, jobs)  
    self.push(obj)  
    next()  
  })  
})
```

```
sourceStream.pipe(fastStream).pipe(slowStream)
```

```
slowStream.resume()
```

# LIVE EXAMPLE

Show backpressure in action



A terminal window titled "fish /home/bmpvieira/backpressure — docker — -bash — 32x7" displays two data streams. The first stream is represented by a row of 16 light blue squares, followed by a closing bracket and the text "16 / 20". The second stream is represented by a row of 11 light blue squares, followed by a closing bracket and the text "11 / 20". At the bottom of the terminal, a dark blue banner contains the text "~ /backpressure" in white, followed by a black right-pointing arrow and the text "node index.js" in light blue.

```
fish /home/bmpvieira/backpressure — docker — -bash — 32x7  
[ 16 / 20  
[ 11 / 20  
~ /backpressure ➡ node index.js
```

# ACKNOWLEDGMENTS

Organisers



Community



Sponsor



Research group



Venue



Friends

