bmpvieira.com

#bionodehack

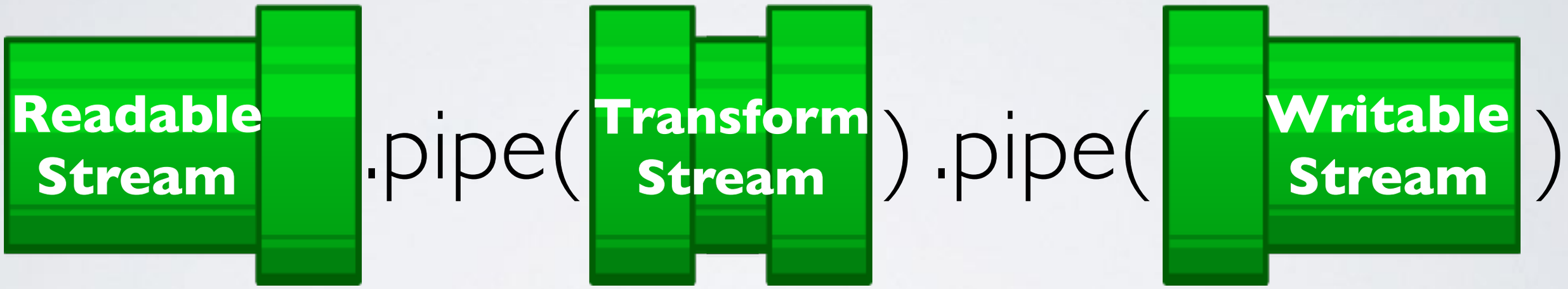
BIONODE.IO

Tutorial

STREAMS

Streams are a first-class construct in Node.js for handling data.

PROCESS DATA IN CHUNKS



fs.createReadStream(file)

request(url)

process.stdin()

JSONStream.parse()

filterFunction()

multithreadAnalysis()

fs.createWriteStream(file)

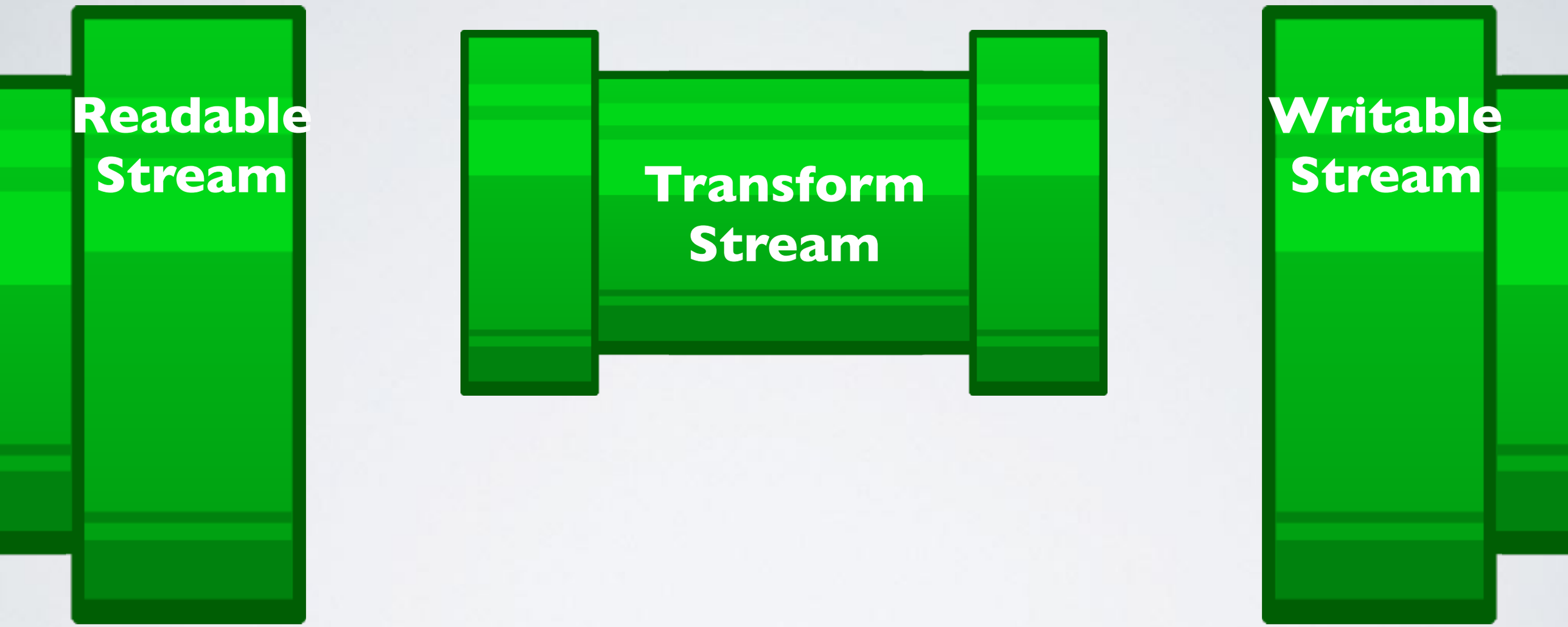
process.stdout()

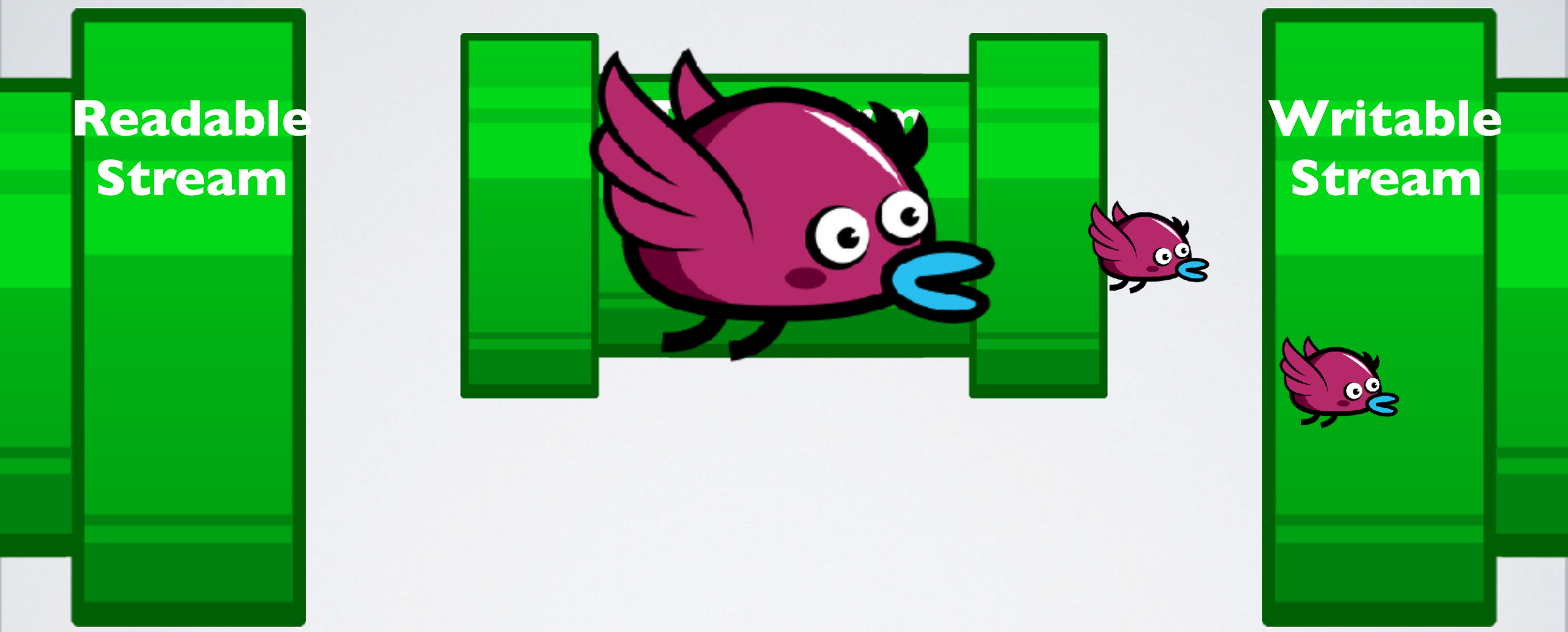
PROCESS DATA IN CHUNKS

Readable Stream

Transform Stream

Writable Stream


Readable Stream
Transform Stream
Writable Stream


Readable Stream
Writable Stream

this.push(data)
FALSE

this.push(data)
TRUE

Readable Stream

Writable Stream

callback()

Buffer

this.push(data)
TRUE

Readable Stream

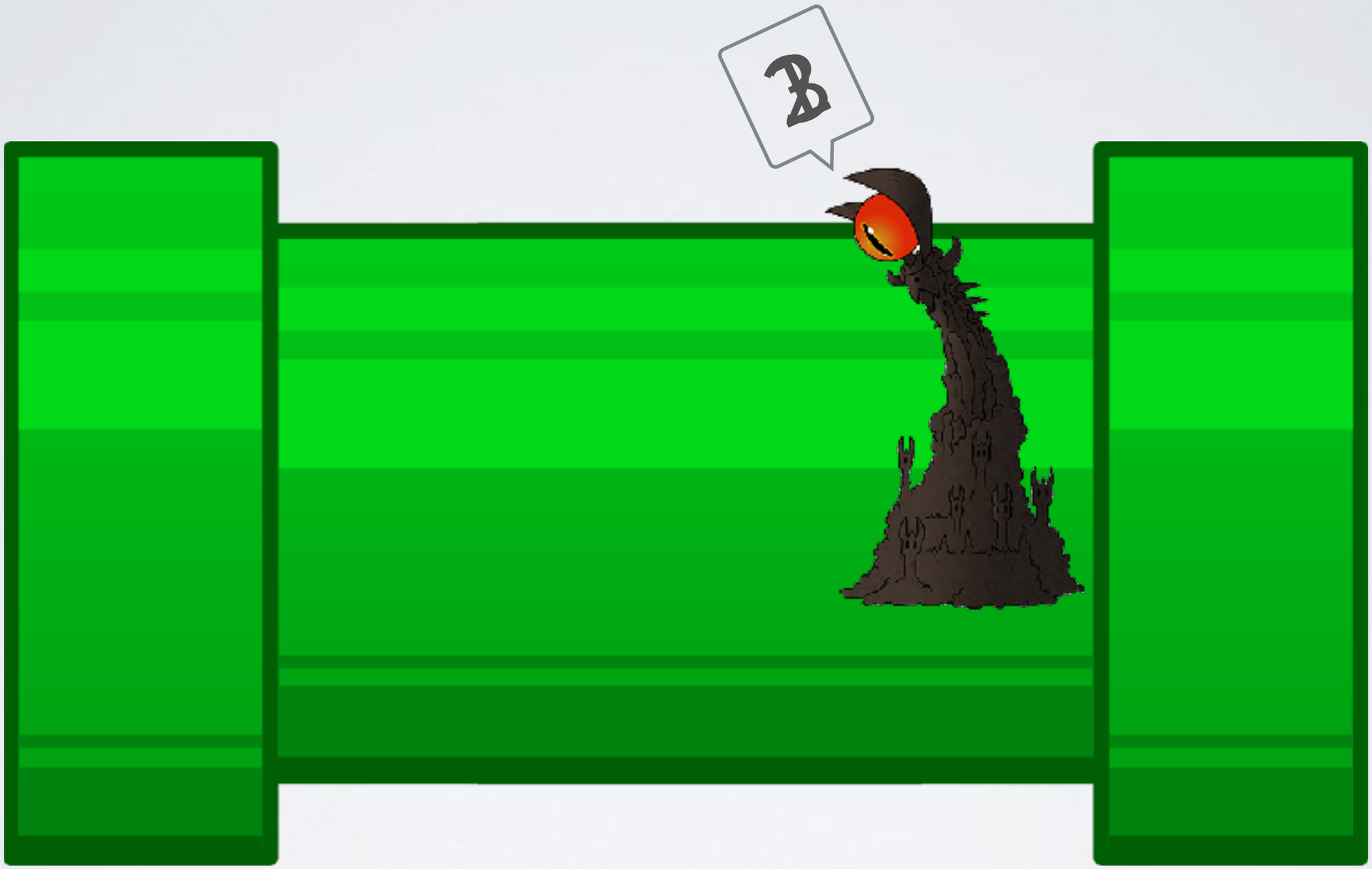
Writable Stream

callback()

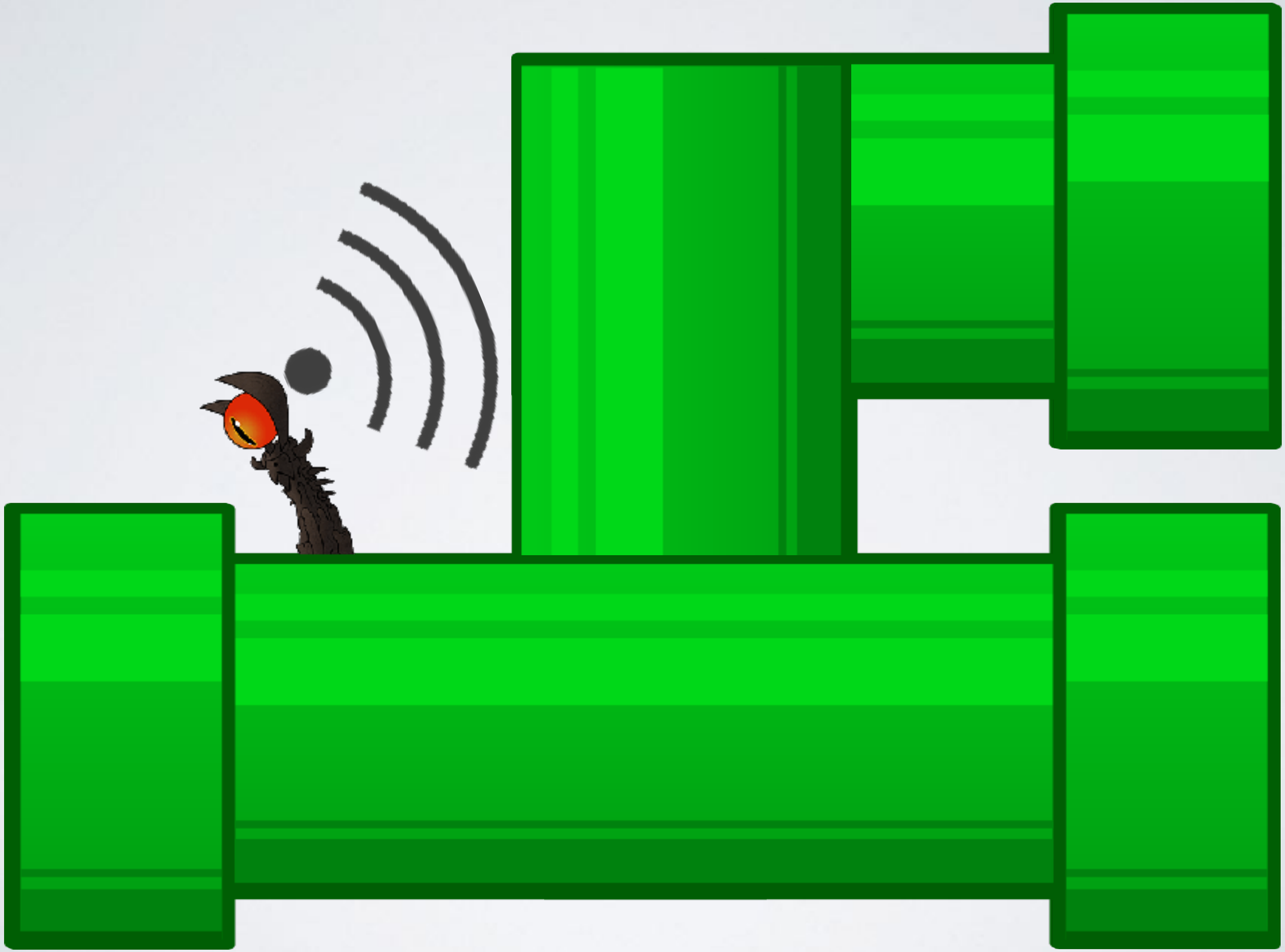
Buffer

DUPLEX STREAM

PASSTHROUGH STREAM



PASSTHROUGH STREAM



STREAM BENEFITS

- Lazily produce or consume data in buffered chunks.
- Evented and non-blocking
- Low memory footprint
- Automatically handle back-pressure
- Buffers allow you to work around the V8 heap memory limit
- Most core Node.js content sources/sinks are streams already!

STREAMS CLASSES

- Readable -- Data Sources
- Writable -- Data Sinks
- Duplex -- Both a Source and a Sink
- Transform -- In-flight stream operations
- Passthrough -- Stream spy

HOW TO IMPLEMENT

- Use handy abstractions like mississippi module (easy way)
- Subclass appropriate Stream Class and implement required methods, i.e., _read(), _write(), etc (hard way)

github.com/maxogden/mississippi

a collection of useful stream utility modules

```
var miss = require('mississippi')
```

- **from** - Make a custom readable stream

- **to** - Make a custom writable stream

- **through** - Make a custom transform stream.

- **duplex** - Take two separate streams, a writable and a readable, and turn them into a single duplex (readable and writable) stream.

- **pipeline** - Combine streams together

check github for example code

LIVE CODING!

Module to get data from European Variation Archive EMBL-EBI

fish /home/bmpvieira/bionode-eva — docker • -bash — 73×23

```
~/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

```
request()
```

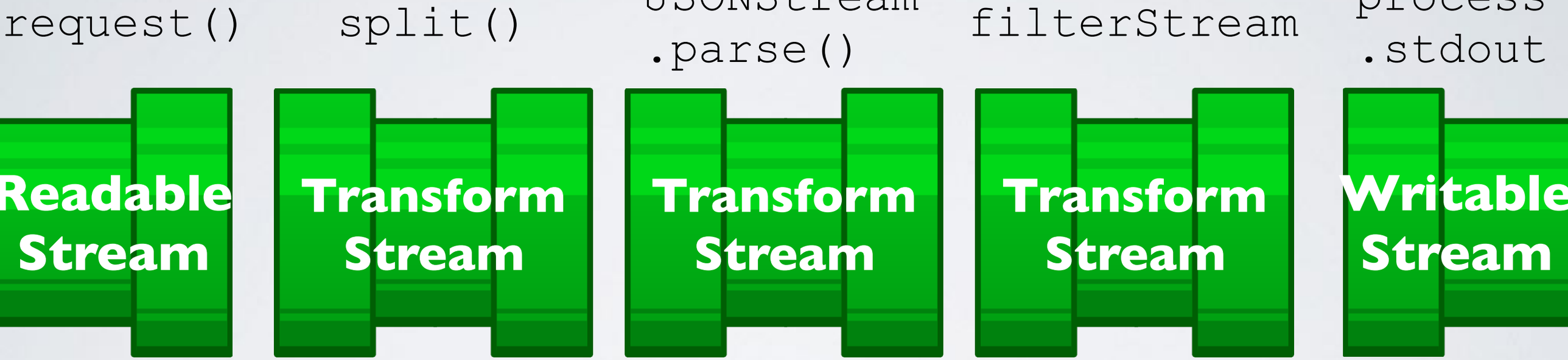
```
process
.stdout
```

```
Binary data
Buffer class
chunks of 16kb
```

Live Coding Solution step 1 - get JSON

```
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)
```

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 through = require('through2')


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)
```

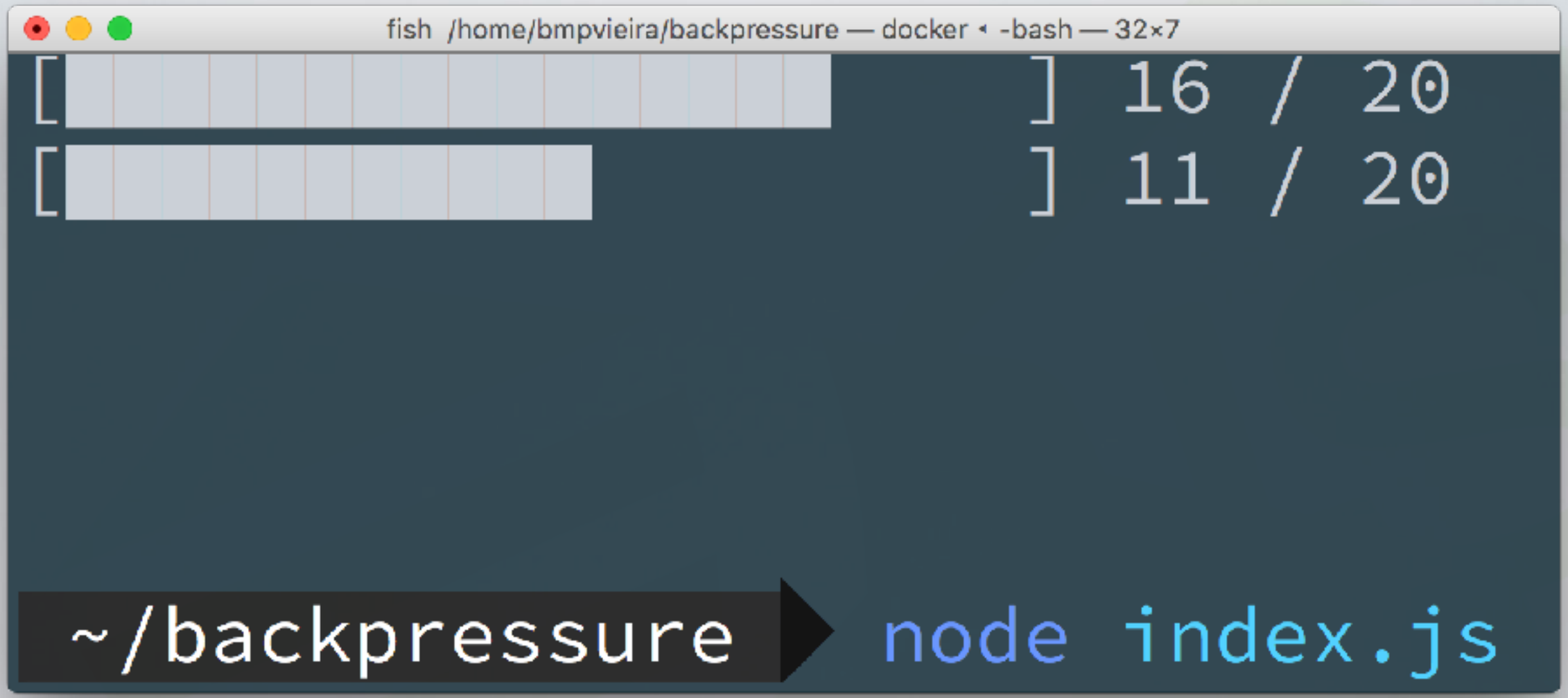
Live Coding Solution step 3 - filter JSON

```
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 through = require('through2')


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



```
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))
```

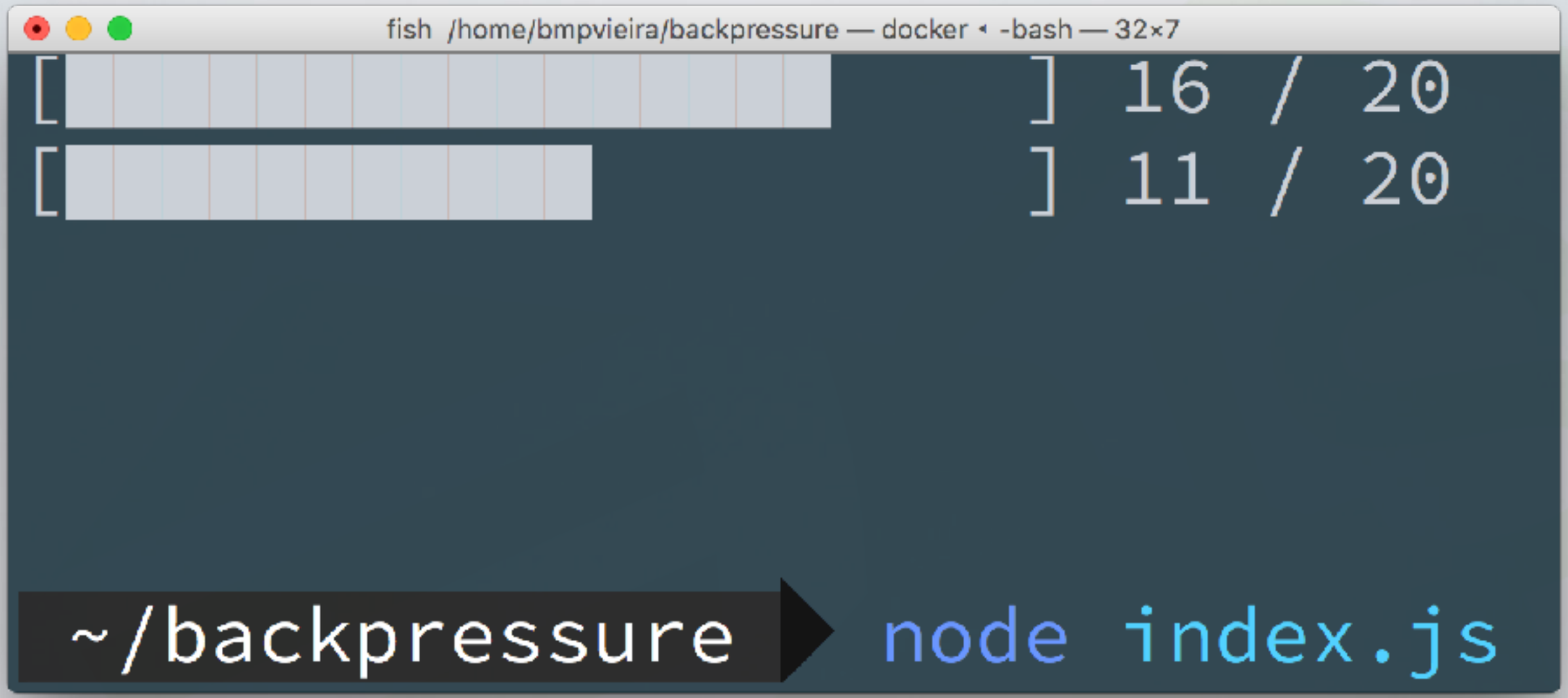
```
var fastStream = through.obj({highWaterMark: 16}, function (obj, enc, next) {
  var self = this
  exec('sleep 1', function (err, stdout, stderr) {
    fastStreamProgressBar.ratio(progress.fastStream++, jobs)
    self.push(obj)
    next()
  })
})

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()
```

LIVE EXAMPLE

Show backpressure in action



LINKS

https://github.com/bionode-hack/discussions#useful-nodejs-modules

https://github.com/bionode-hack/discussions#presentation-slides

Follow @maxogden, @mafintosh and @substack

ACKNOWLEDGMENTS

mozilla Science Lab

Research group

WurmLab .github.io

Queen Mary University of London

Community

bionode.io

Friends

dat DATA

IMAGES SOURCES

http://opengameart.org/content/bevouliin-green-flappy-bird-sprite-sheets

http://neoriceisgood.deviantart.com/art/100-furniture-sprites-405058884

http://android272.deviantart.com/art/Teeworlds-Teleport-570298308

http://www.how-to-draw-cartoons-online.com/eye-of-sauron.html