bmpvieira.com

#bionodehack

BIONODE.IO

Introduction

github.com/bionode-hack/discussions

gitter.im/bionode-hack/discussions

WHAT

Modular and universal bioinformatics

WHAT

Modular and universal bioinformatics

- Tries to do one thing well

WHAT

Modular and universal bioinformatics


Tries to do one thing well


Provides highly reusable code and tools

WHAT

Modular and universal bioinformatics

- Tries to do one thing well
- Provides highly reusable code and tools
- Scales by using Streams

WHAT

Modular and universal bioinformatics


Tries to do one thing well


Provides highly reusable code and tools


Scales by using Streams


Runs everywhere

HOW

Using Node.js

HOW

Using Node.js

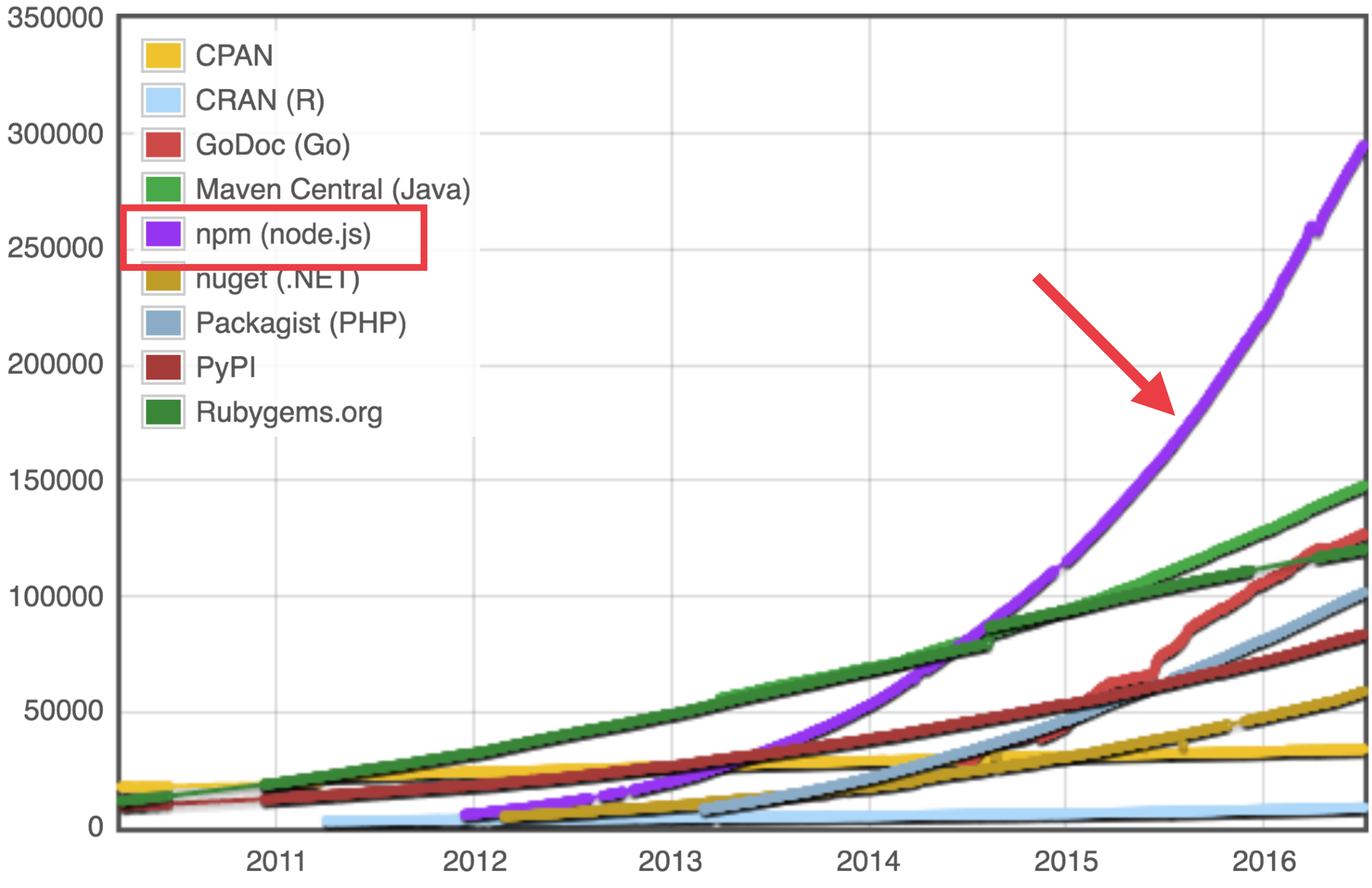
- Highly modular

HOW

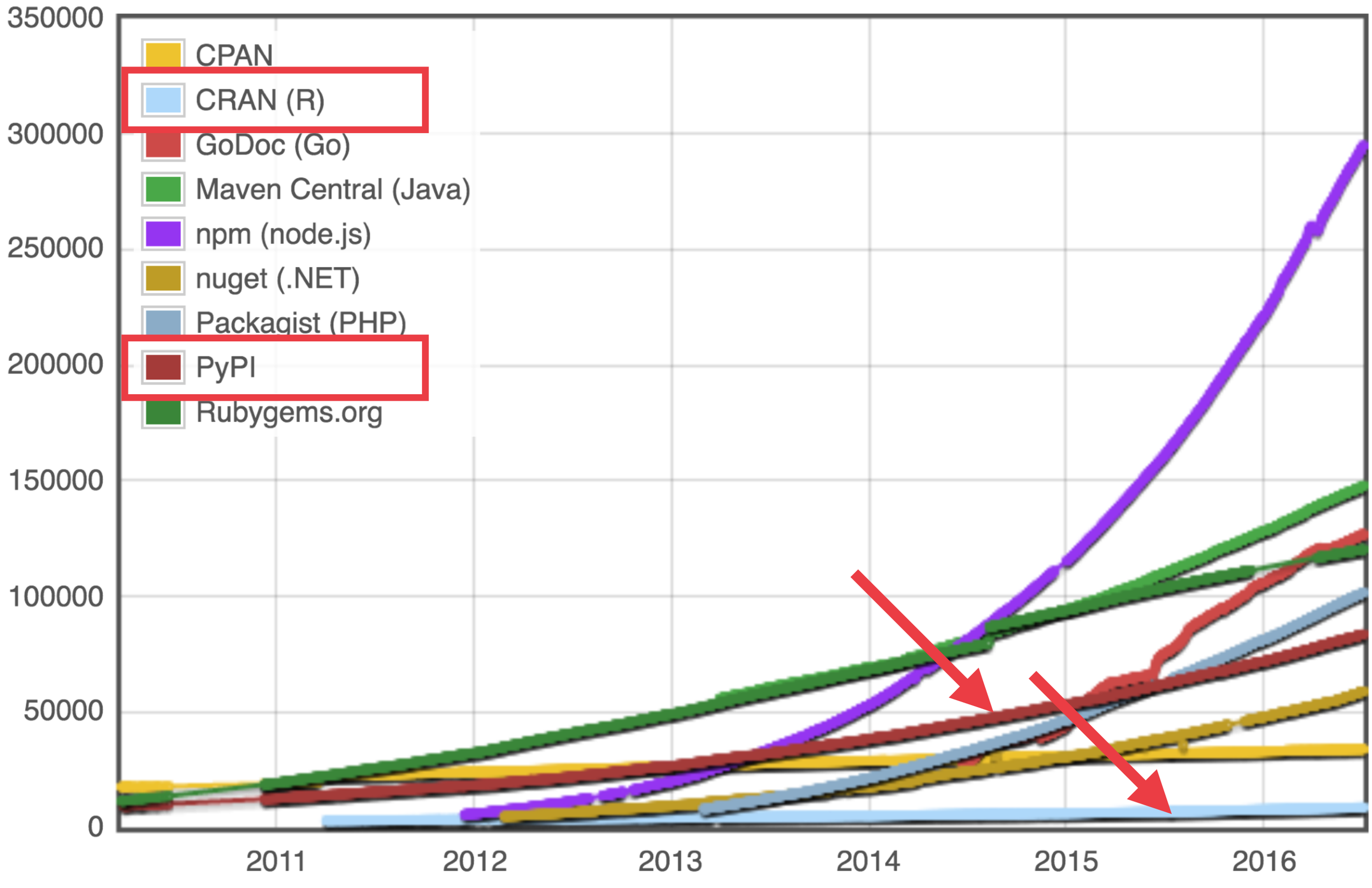
Using Node.js

- Highly modular
- Very open community on GitHub

Modules count



Modules count



HOW

Using Node.js

- Highly modular
- Very open community on GitHub

HOW

Using Node.js

- Highly modular
- Very open community on GitHub
- Provides native implementation of Streams

HOW

Using Node.js

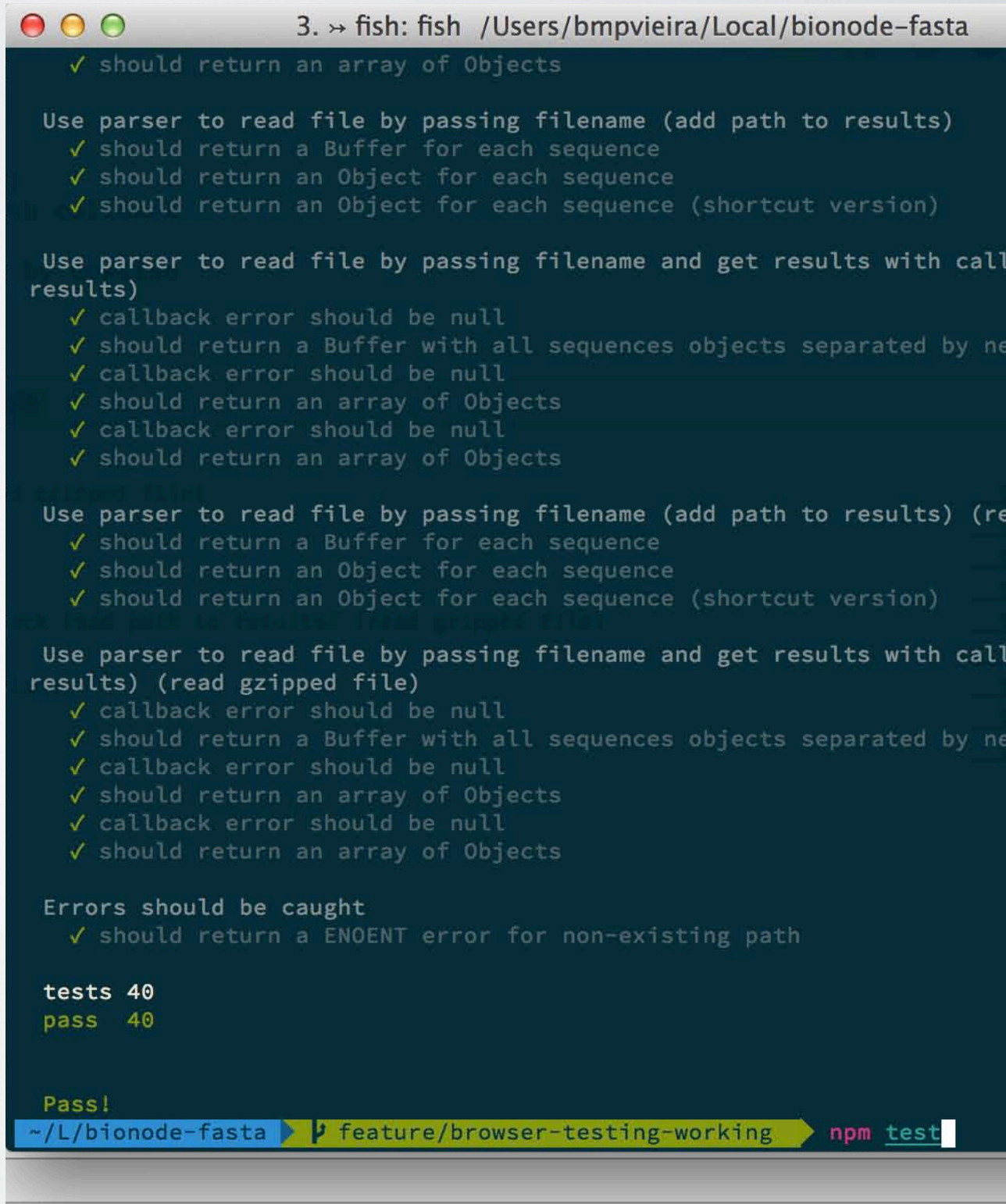

Highly modular


Very open community on GitHub

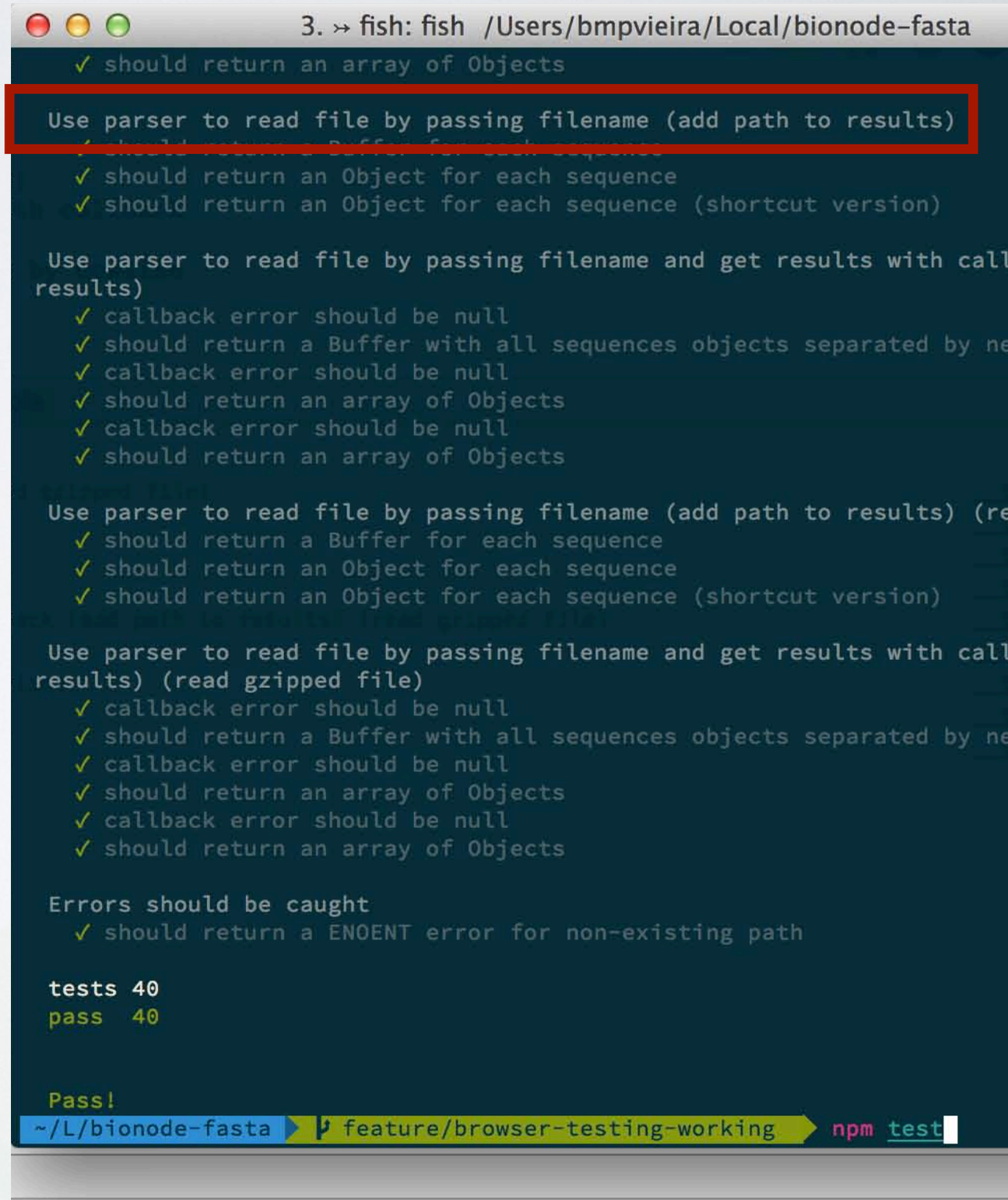

Provides native implementation of Streams


Run same JavaScript code on browser or CLI

Run same JavaScript code on browser or CLI

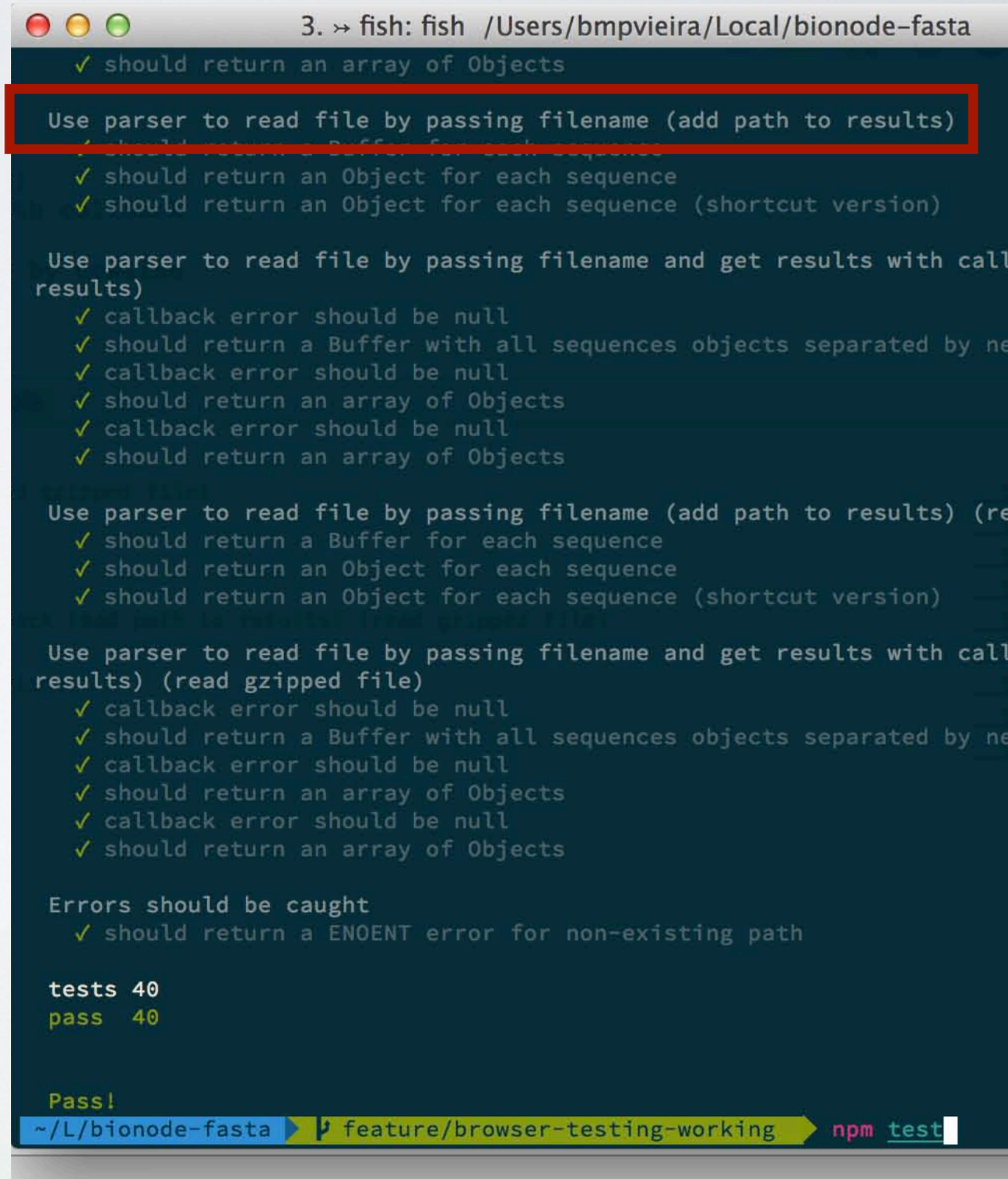


Run same JavaScript code on browser or CLI



Run same JavaScript code on browser or CLI

FASTA file format
> X-gene
ATGCGTACTGCATCATG
ACGTACTGCATTCATGC
GTGCTAGGGGTTTACGT



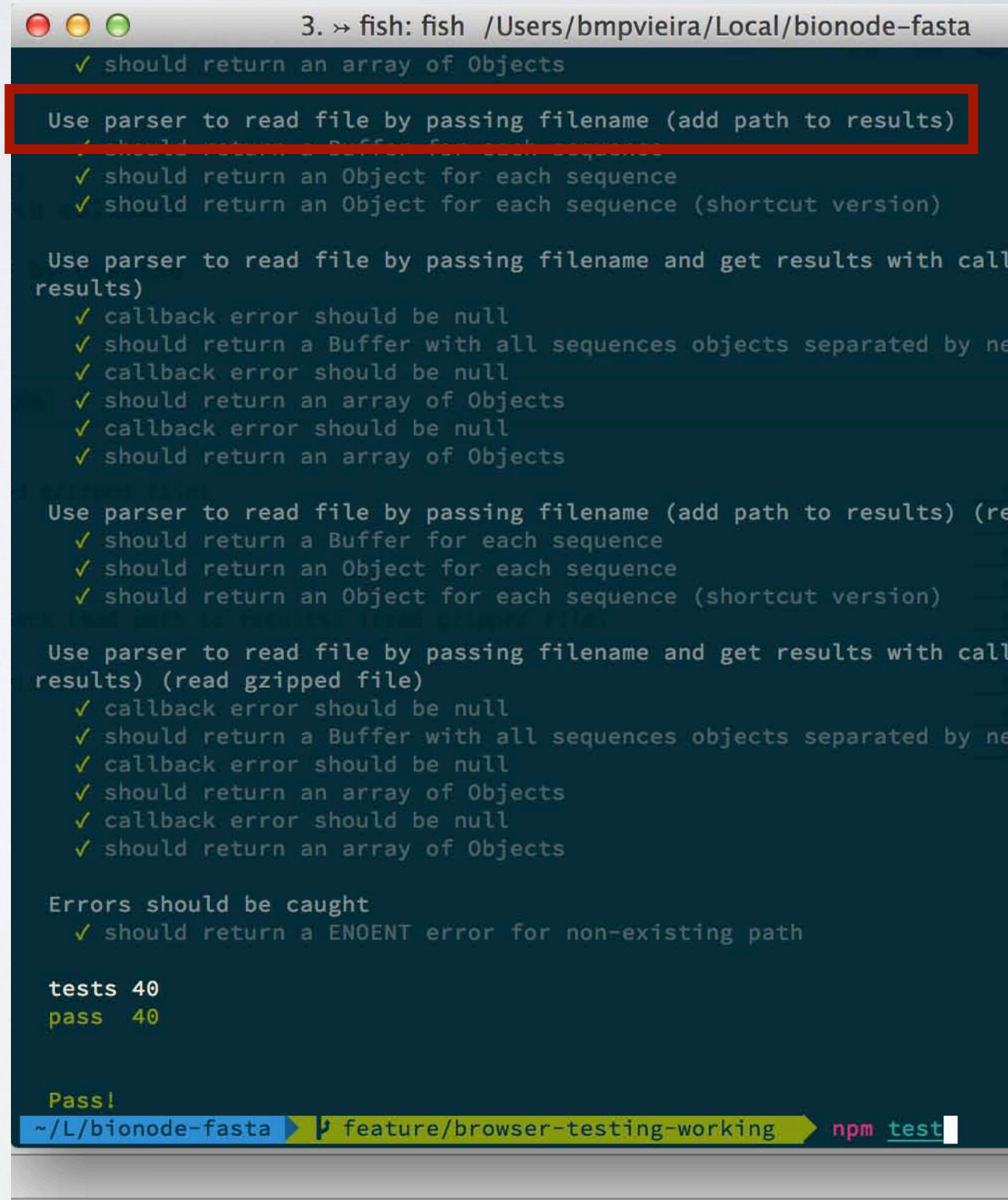
Run same JavaScript code on browser or CLI

FASTA file format

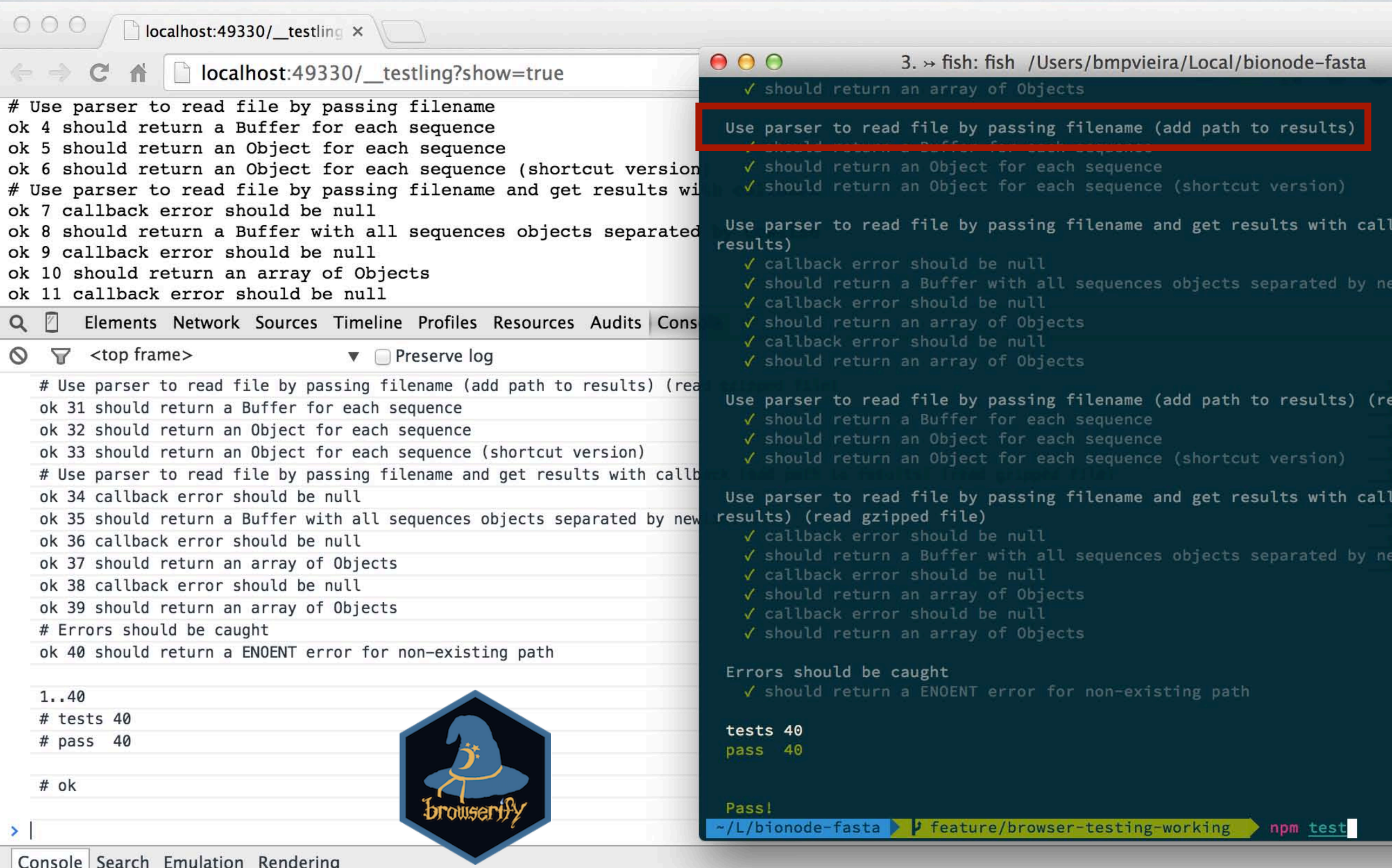
```
> X-gene
ATGCGTACTGCATCATG
ACGTACTGCATTCATGC
GTGCTAGGGGTTTACGT
```

JSON (JavaScript Object Notation)

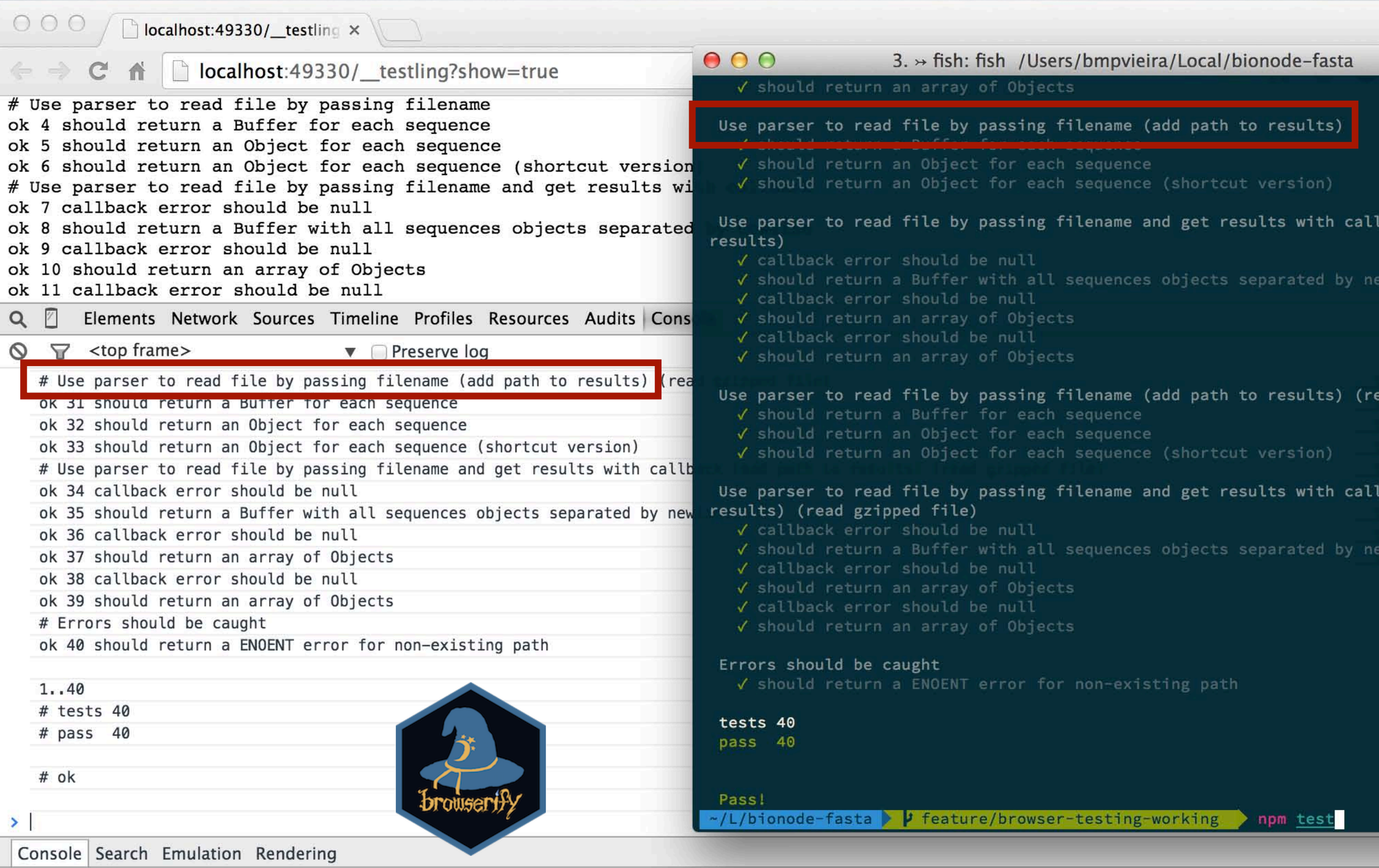
```
{
  “id”:“X-gene”,
  “seq”:“ATGCGTACTGCATCAT
GACGTACTGCATTCATGCGTG
CTAGGGGTTTACGT”
}
```



Run same JavaScript code on browser or CLI



Run same JavaScript code on browser or CLI



- Run same JavaScript code on browser or CLI

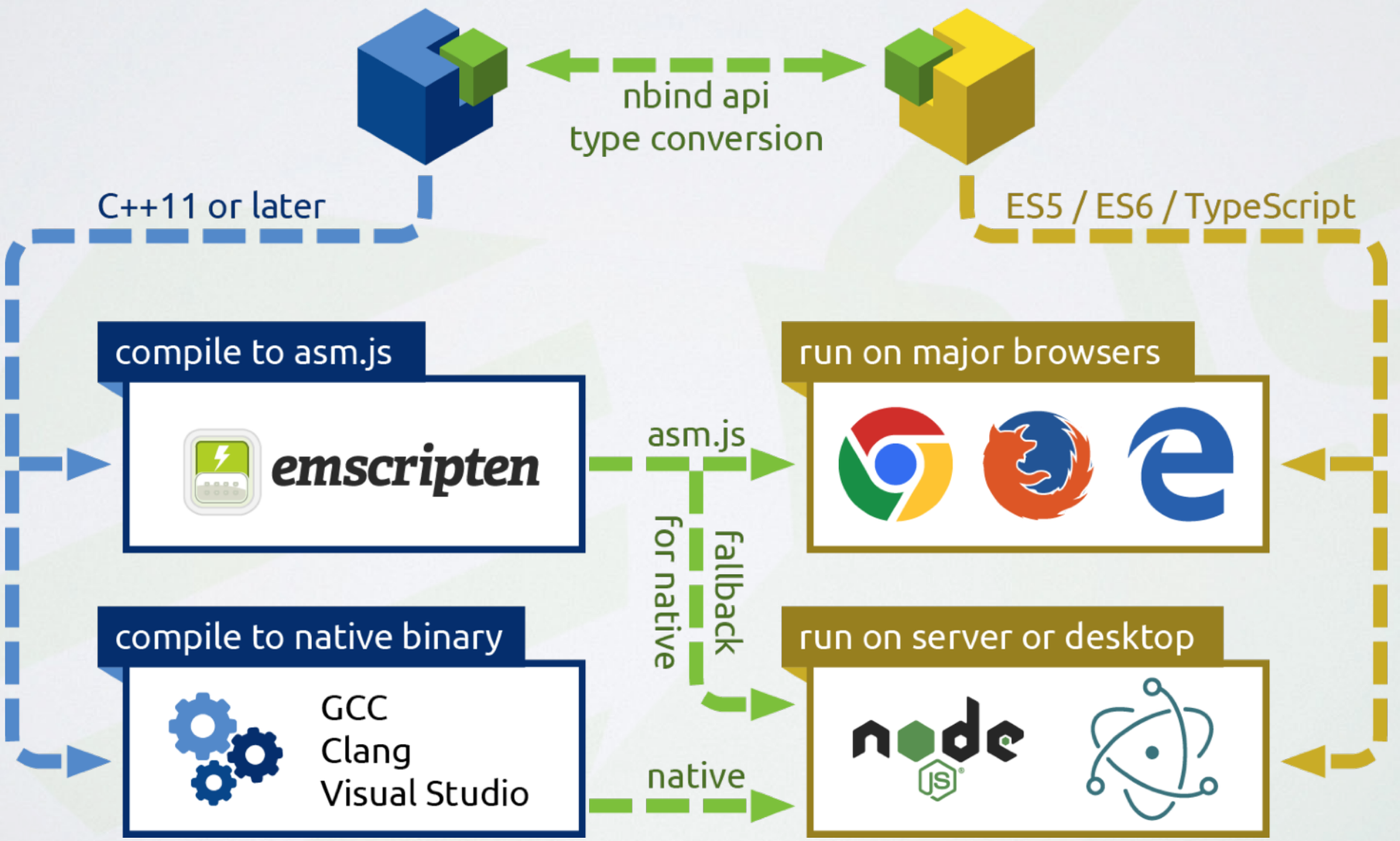
JavaScript not suitable for heavy scientific computation?

C++ preferred?

Run C++ from JavaScript

nbind

#include nbind in C++ & call effortlessly from JavaScript without changes.



ORIGIN

During my PhD at

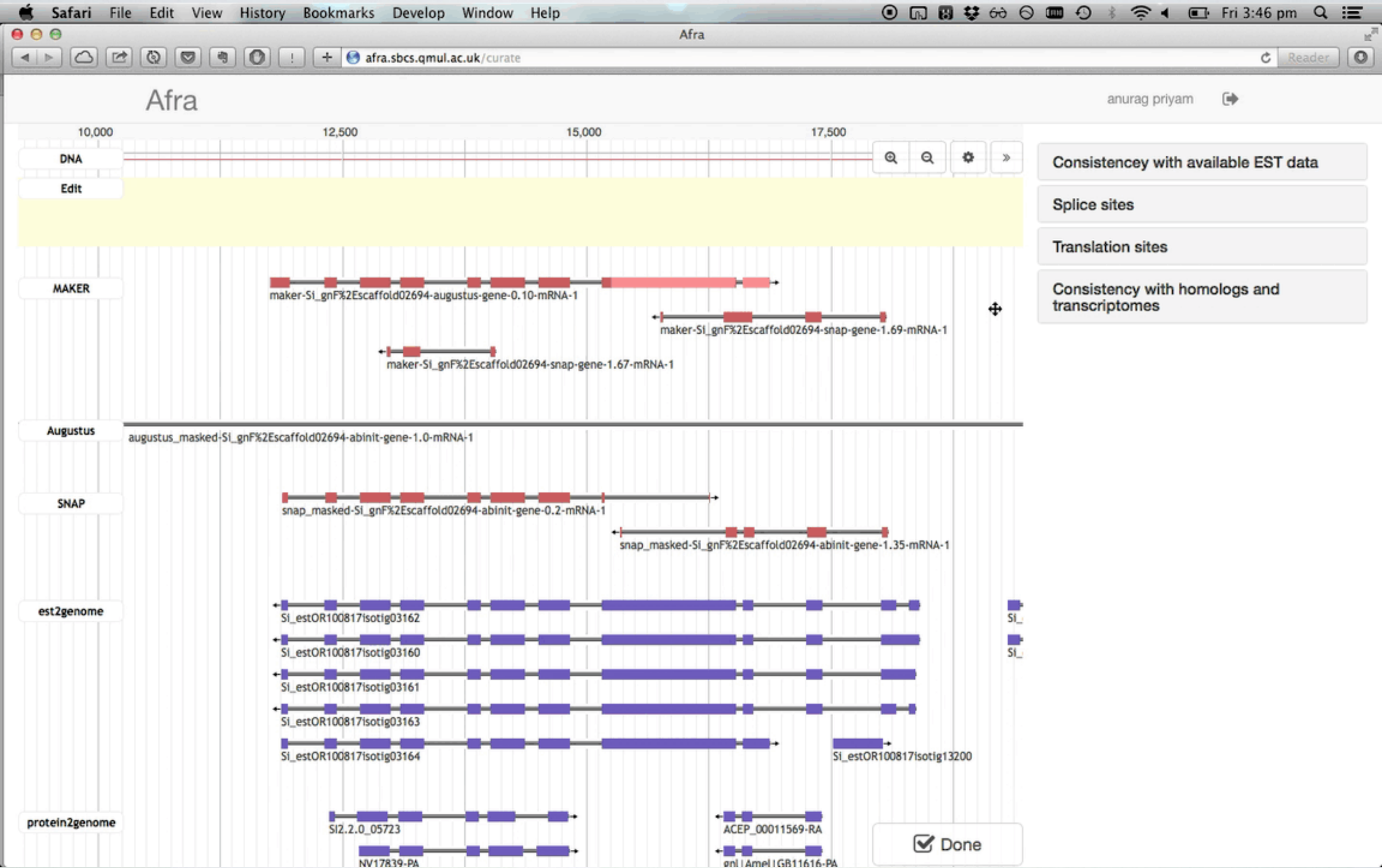
ORIGIN

During my PhD at

- Involved in biological web projects that need JS

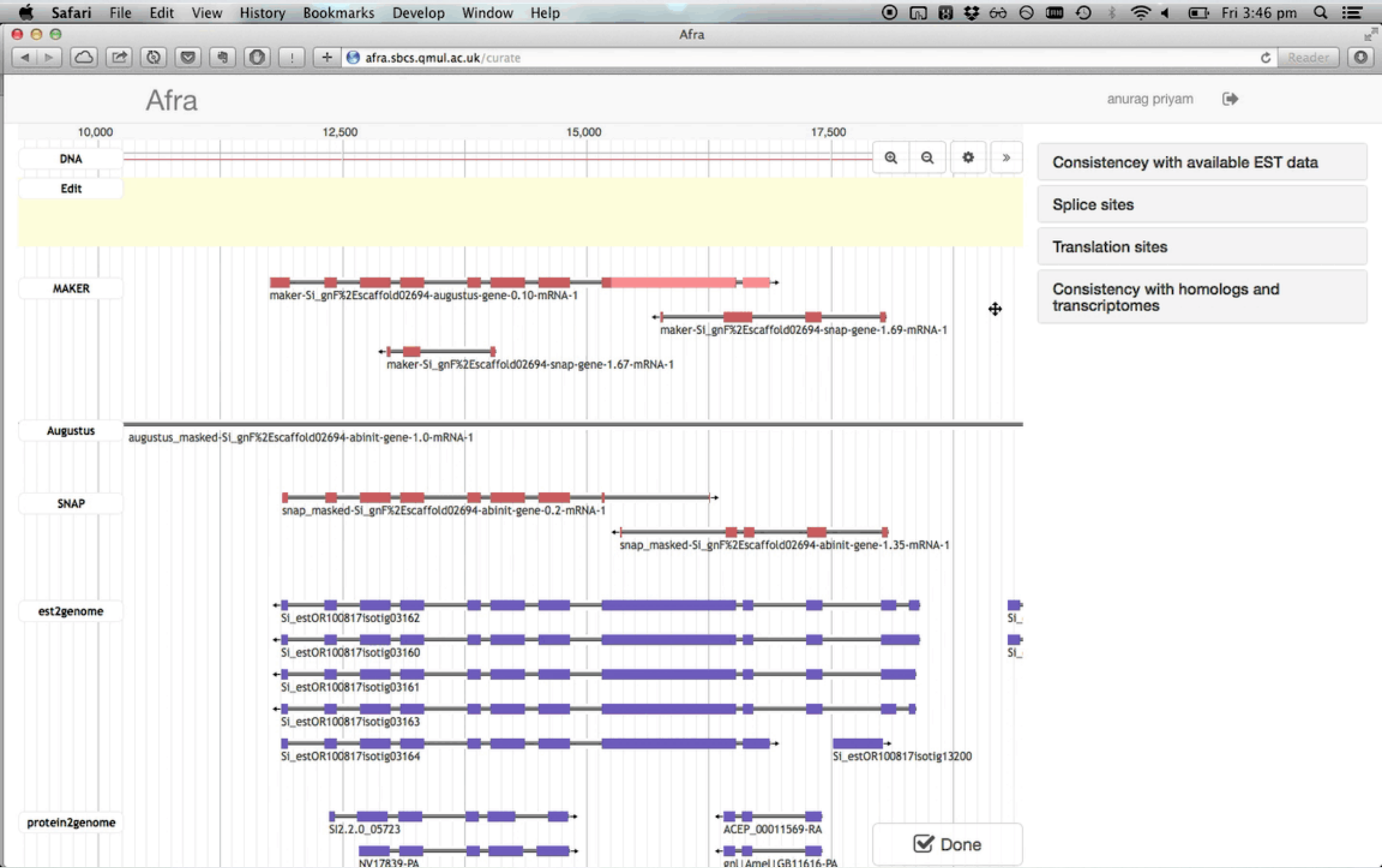
Involved in biological web projects that need JS

wurmlab.github.io



Involved in biological web projects that need JS

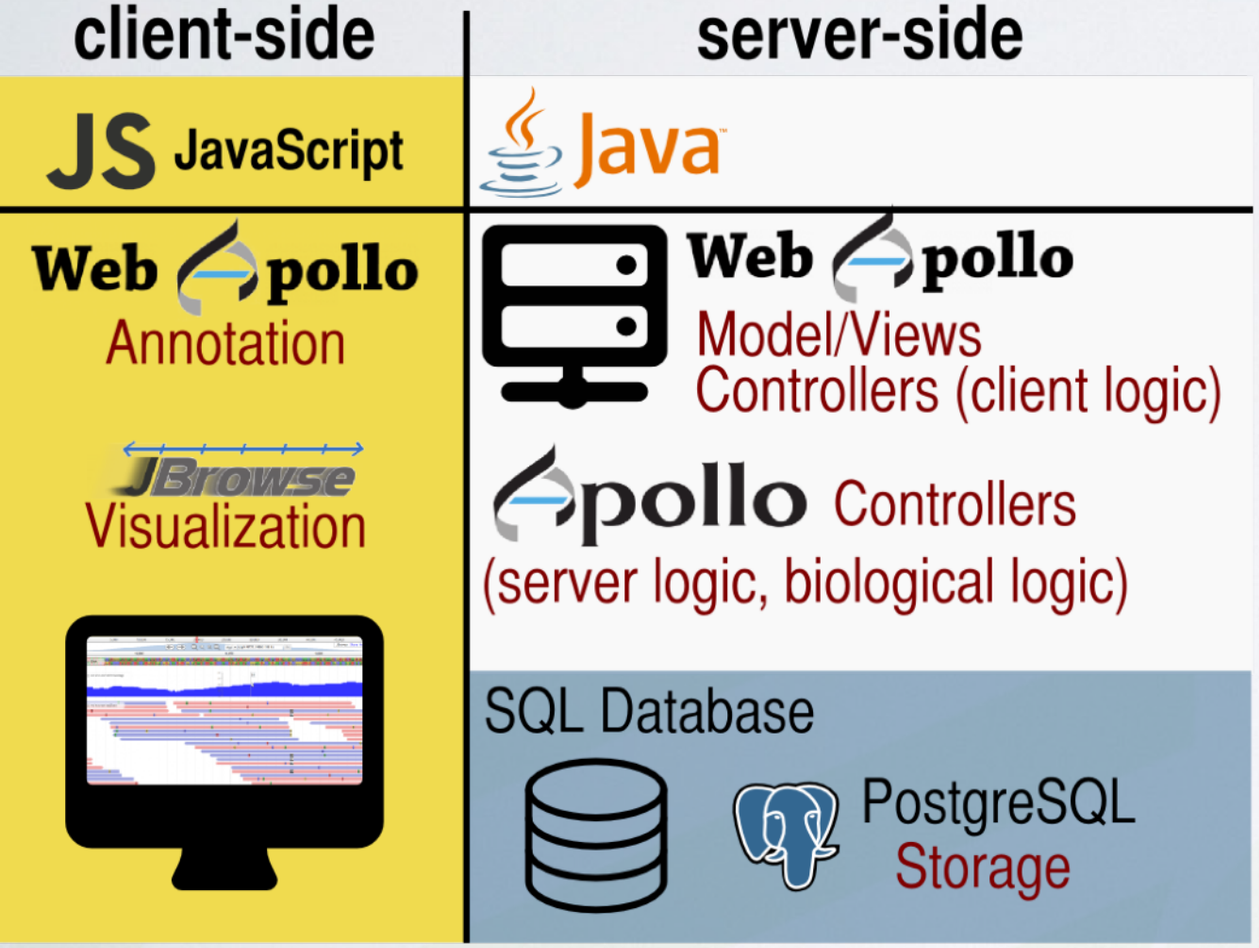
wurmlab.github.io



Involved in biological web projects that need JS

wurmlab.github.io

GMOD



Involved in biological web projects that need JS

wurmlab.github.io

GMOD

client-side	server-side
JS JavaScript	Java
Web Apollo Annotation	Web Apollo Model/Views Controllers (client logic)
JBrowse Visualization	Apollo Controllers (server logic, biological logic)
	SQL Database PostgreSQL Storage

Afra

client-side	server-side
JS JavaScript	~~Java~~ Ruby
Web Apollo Annotation	~~Web Apollo Model/Views Controllers (client logic)~~
JBrowse Visualization	~~Apollo~~ Controllers (server logic, ~~biological validation~~)
AngularJS by Google Models/Views/Controllers	SQL Database PostgreSQL Storage
bionode Biological logic	

CHECK SEQUENCE TYPE

Takes a sequence string and checks if it's DNA, RNA or protein. Follows IUPAC notation which allows ambiguous sequence notation. In this case the sequence is labelled as ambiguous nucleotide rather than amino acid sequence.

```
seq.checkType("ATGACCCTGAGAAGAGCACCG");
=> "dna"
seq.checkType("AUGACCCUGAAGGUGAAUGAA");
=> "rna"
seq.checkType("MAYKSGKRPTFFEVFKAHCSDS");
=> "protein"
seq.checkType("AMTGACCCTGAGAAGAGCACCG");
=> "ambiguousDna"
seq.checkType("AMUGACCCUGAAGGUGAAUGAA");
=> "ambiguousRna"
```

```
seq.checkType = function(sequence) {
  var acgMatch = sequence.match(/[ACG]/i);
  var tMatch = sequence.match(/[T]/i);
  var nMatch = sequence.match(/[N]/i);
  var uMatch = sequence.match(/[U]/i);
  var potentialNucleotideMatch = sequence.match(/[WSMKRYBDHV]/i);
  var proteinMatch = sequence.match(/[EFIJLOPQZX\*]/i);
  if (proteinMatch) {
    return "protein";
  } else if (acgMatch && !potentialNucleotideMatch && !uMatch) {
    return "dna";
  } else if (acgMatch && potentialNucleotideMatch && !uMatch) {
    return "ambiguousDna";
  } else if (acgMatch && !potentialNucleotideMatch && uMatch && !tMatch) {
    return "rna";
  } else if (acgMatch && potentialNucleotideMatch && uMatch && !tMatch) {
    return "ambiguousRna";
  }
}
```

REVERSE SEQUENCE

Takes sequence string and returns the reverse sequence.

```
seq.reverse("ATGACCCTGAAGGTGAA");
=> "AAGTGGAAGTCCCAGTA"
```

```
seq.reverse = function(sequence) {
  return sequence.split('').reverse().join('')
}
```

(REVERSE) COMPLEMENT SEQUENCE

Takes a sequence string and optional boolean for reverse, and returns its complement.

```
seq.complement("ATGACCCTGAAGGTGAA");
=> "TACTGGGACTTCCACTT"
seq.complement("ATGACCCTGAAGGTGAA", true);
=> "TTCACCTTCAGGGTCAT"
//Alias
seq.reverseComplement("ATGACCCTGAAGGTGAA");
=> "TTCACCTTCAGGGTCAT"
```

```
seq.complement = function(sequence, reverse) {
  var reverse = reverse || false
  var sequenceType = seq.checkType(sequence)
  var getComplementBase = seq.createComplementBase(sequenceType)
  if (reverse) {
    return sequence.split('').reverse().map(getComplementBase).join('')
  }
  else {
    return sequence.split('').map(getComplementBase).join('')
  }
}
```

ORIGIN

During my PhD at

- Involved in biological web projects that need JS

ORIGIN

During my PhD at

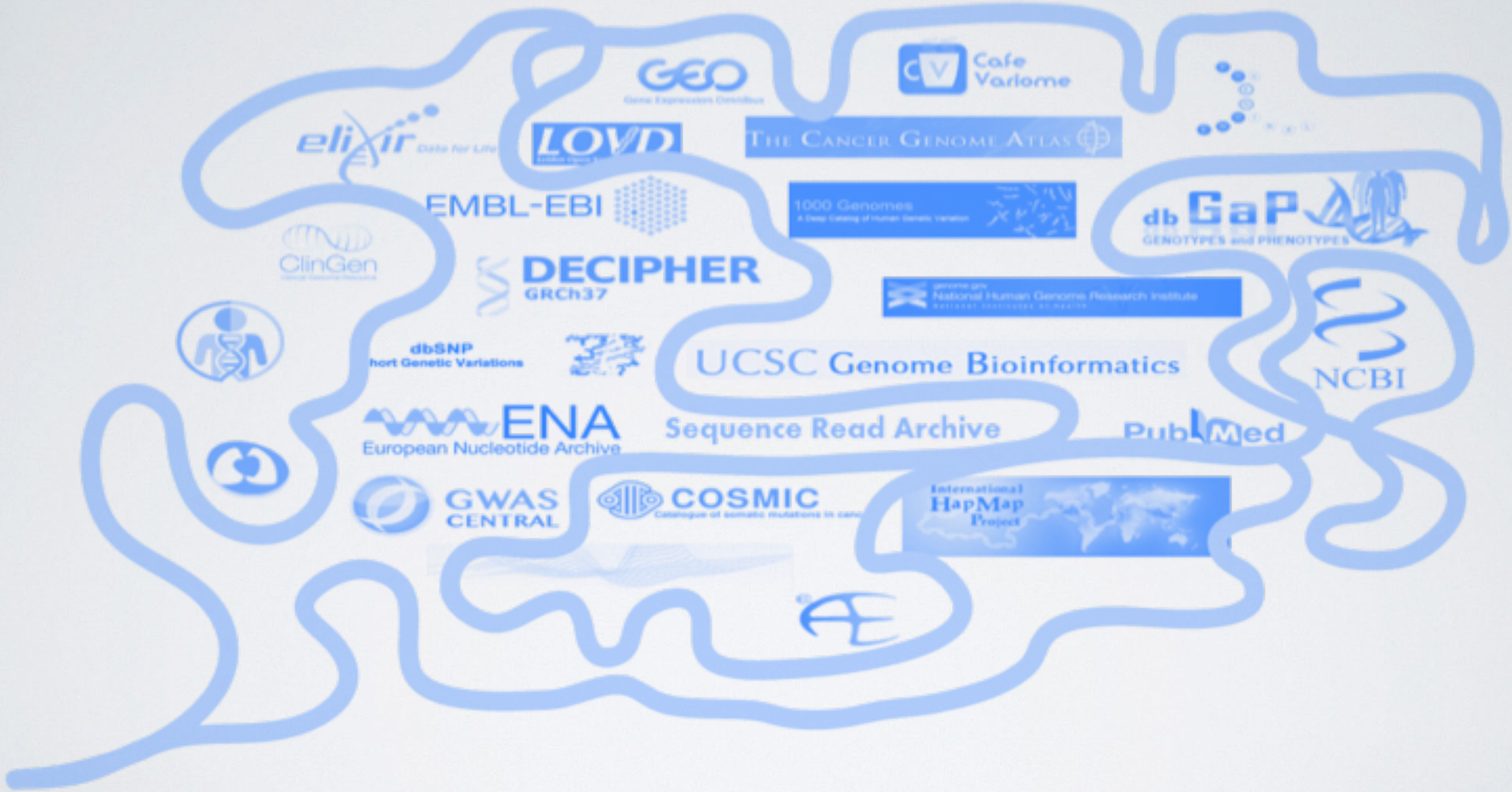
- Involved in biological web projects that need JS
- Had to find and get TB of data online

Compare genetic diversity of social vs solitary species

Compare genetic diversity of social vs solitary species

Compare genetic diversity of social vs solitary species

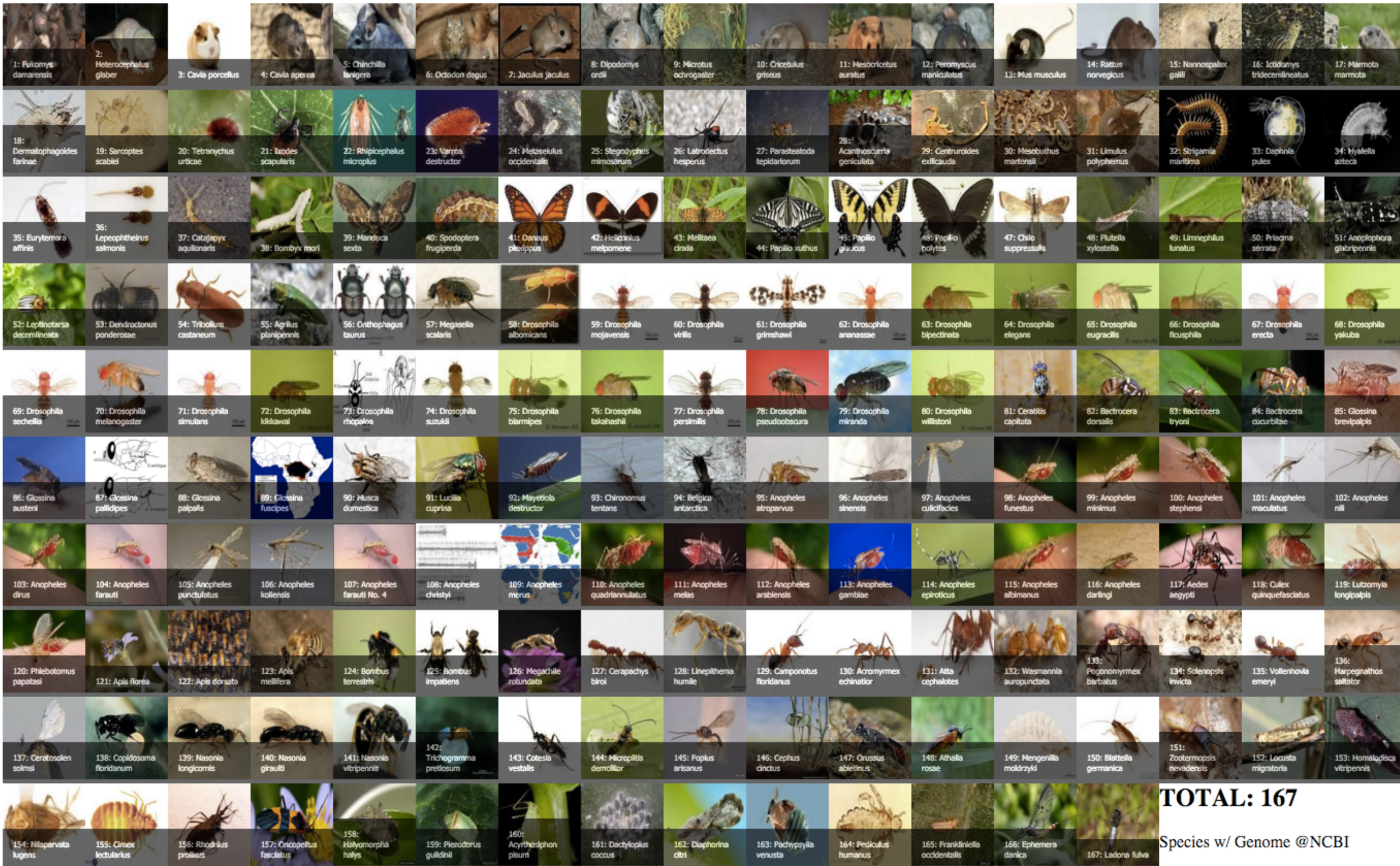
Maze of data sources



Maze of data sources

Had to find and get TB of data online

All Insects and Rodents



TOTAL: 167

Species w/ Genome @NCBI

Had to find and get TB of data online

Social (red) - Solitary (blue)

TOTAL: 167

Blue solitary, Red social

How to get the URLs for genomic dataset?

for example raw data of genome assembly:

ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000188075.1_Si_gnG

```
~/project-phd ▶ bionode-ncbi urls assembly ants | head -n 1 | json
{
  "uid": "140471",
  "structure": {
    "dir": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_assembly_structure/"
  },
  "report": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_assembly_report.txt"
  },
  "stats": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_assembly_stats.txt"
  },
  "genomic": {
    "fna": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_rna_from_genomic.fna.gz",
    "gbff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_genomic.gbff.gz",
    "gff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_genomic.gff.gz"
  },
  "table": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_feature_table.txt.gz"
  },
  "protein": {
    "faa": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_protein.faa.gz",
    "gpff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_protein.gpff.gz"
  },
  "rm": {
    "out": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_rm.out.gz",
    "run": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_rm.run"
  },
  "wgsmaster": {
    "gbff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_wgsmaster.gbff.gz"
  },
  "README": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/README.txt"
  },
  "hashes": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/annotation_hashes.txt"
  },
  "md5checksums": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/md5checksums.txt"
  }
}
~/project-phd ▶
```

fish /home/bmpvieira/project-phd — docker ◂ -bash — 131×42

```
~/project-phd bionode-ncbi urls assembly ants | head -n 1 | json
{
  "uid": "140471",
  "structure": {
    "dir": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_assembly_structure/"
  },
  "report": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_assembly_report.txt"
  },
  "stats": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_assembly_stats.txt"
  },
  "genomic": {
    "fna": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_rna_from_genomic.fna.gz",
    "gbff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_genomic.gbff.gz",
    "gff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_genomic.gff.gz"
  },
  "table": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_feature_table.txt.gz"
  },
  "protein": {
    "faa": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_protein.faa.gz",
    "gpff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_protein.gpff.gz"
  },
  "rm": {
    "out": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_rm.out.gz",
    "run": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_rm.run"
  },
  "wgsmaster": {
    "gbff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_wgsmaster.gbff.gz"
  },
  "README": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/README.txt"
  },
  "hashes": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/annotation_hashes.txt"
  },
  "md5checksums": {
    "txt": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/md5checksums.txt"
  }
}
~/project-phd
```

bmpvieira.com

bionode-ncbi

#bionodehack

```
var bio = require('bionode')
```

```
var bio = require('bionode')
```

```
var bio = require('bionode')

// Callback pattern
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})

// Event pattern
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})

// Event pattern
bio.ncbi.urls('assembly', ‘ants’)
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})

// Event pattern
bio.ncbi.urls('assembly', ‘ants’)
.on('data', printGenomeURL)
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})

// Event pattern
bio.ncbi.urls('assembly', ‘ants’)
.on('data', printGenomeURL)
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})

// Event pattern
bio.ncbi.urls('assembly', ‘ants’)
.on('data', printGenomeURL)

function printGenomeURL(urls) {
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})
```

```
// Event pattern
bio.ncbi.urls('assembly', ‘ants’)
.on('data', printGenomeURL)

function printGenomeURL(urls) {
  console.log(urls[0].genomic.fna)
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})
```

```
// Event pattern
bio.ncbi.urls('assembly', ‘ants’)
.on('data', printGenomeURL)

function printGenomeURL(urls) {
  console.log(urls[0].genomic.fna)
})
```

```
var bio = require('bionode')

// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})
```

```
// Event pattern
bio.ncbi.urls('assembly', ‘ants’)
.on('data', printGenomeURL)

function printGenomeURL(urls) {
  console.log(urls[0].genomic.fna)
})
```

bmpvieira.com

bionode-ncbi

#bionodehack

```
// Pipe pattern
```

```
// Pipe pattern
var ncbi = require('bionode-ncbi')
```

```
// Pipe pattern
var ncbi = require('bionode-ncbi')
var ndjson = require(‘ndjson')
```

```
// Pipe pattern
var ncbi = require('bionode-ncbi')
var ndjson = require(‘ndjson')
```

```
// Pipe pattern
var ncbi = require('bionode-ncbi')
var ndjson = require(‘ndjson')

ncbi.urls('genomes', 'ants')
```

```
// Pipe pattern
var ncbi = require('bionode-ncbi')
var ndjson = require('ndjson')

ncbi.urls('genomes', 'ants')
.pipe(ndjson.stringify())
```

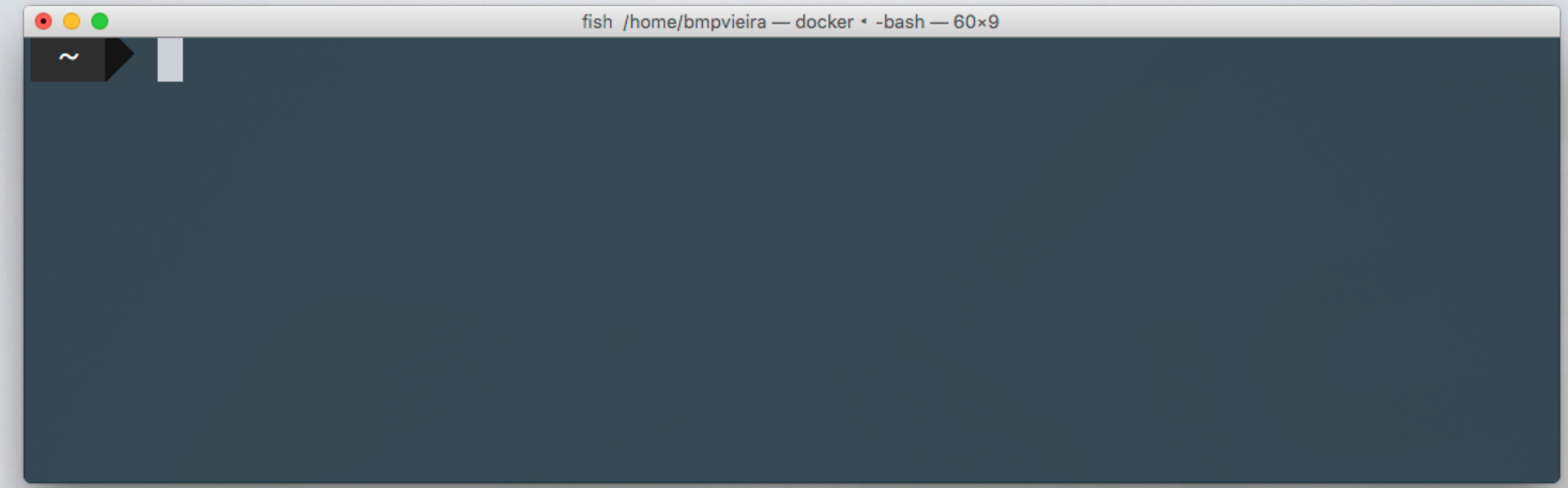
```
// Pipe pattern
var ncbi = require('bionode-ncbi')
var ndjson = require('ndjson')

ncbi.urls('genomes', 'ants')
.pipe(ndjson.stringify())
.pipe(process.stdout)
```

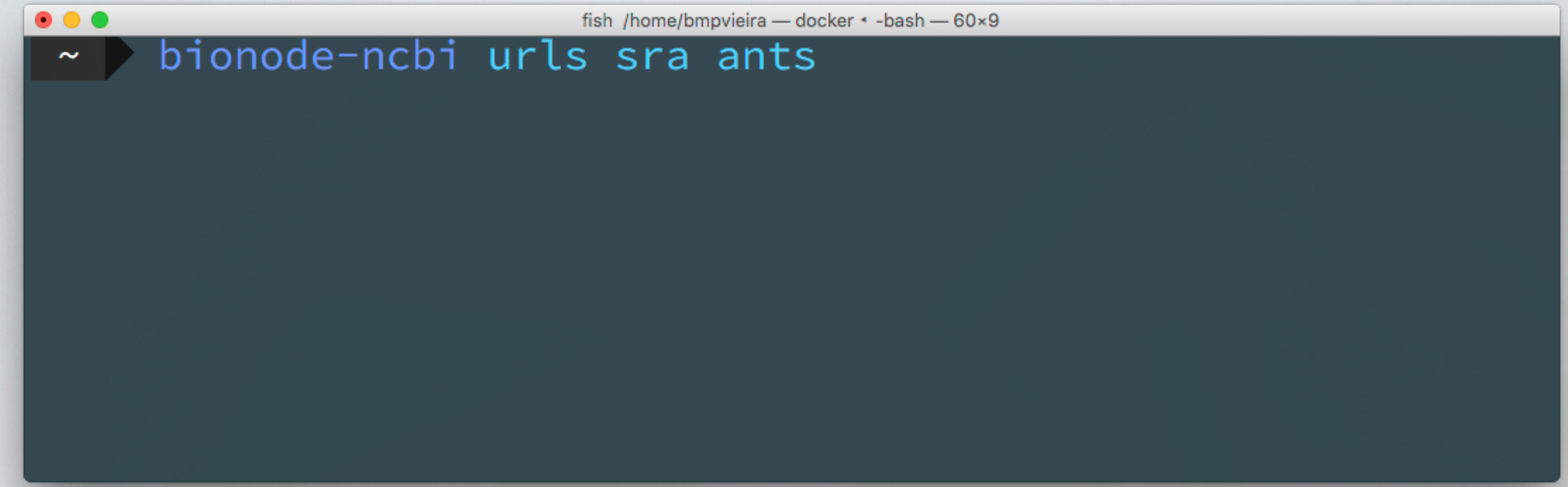
```
// Pipe pattern
var ncbi = require('bionode-ncbi')
var ndjson = require('ndjson')

ncbi.urls('genomes', 'ants')
.pipe(ndjson.stringify())
.pipe(process.stdout)
```

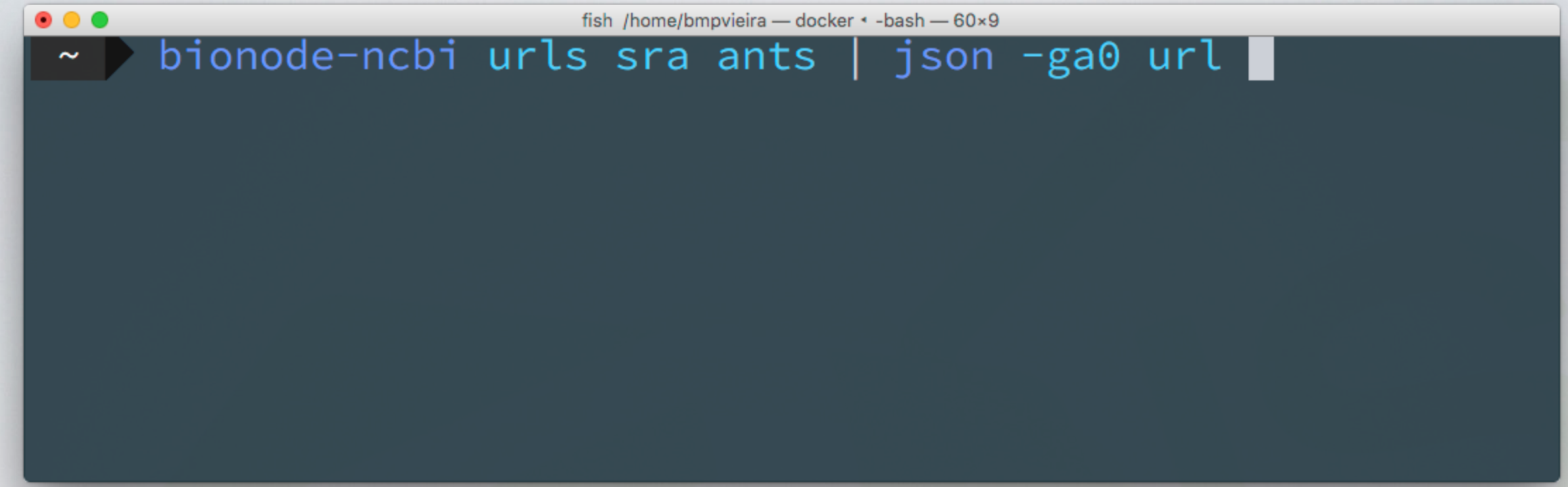
bmpvieira.com

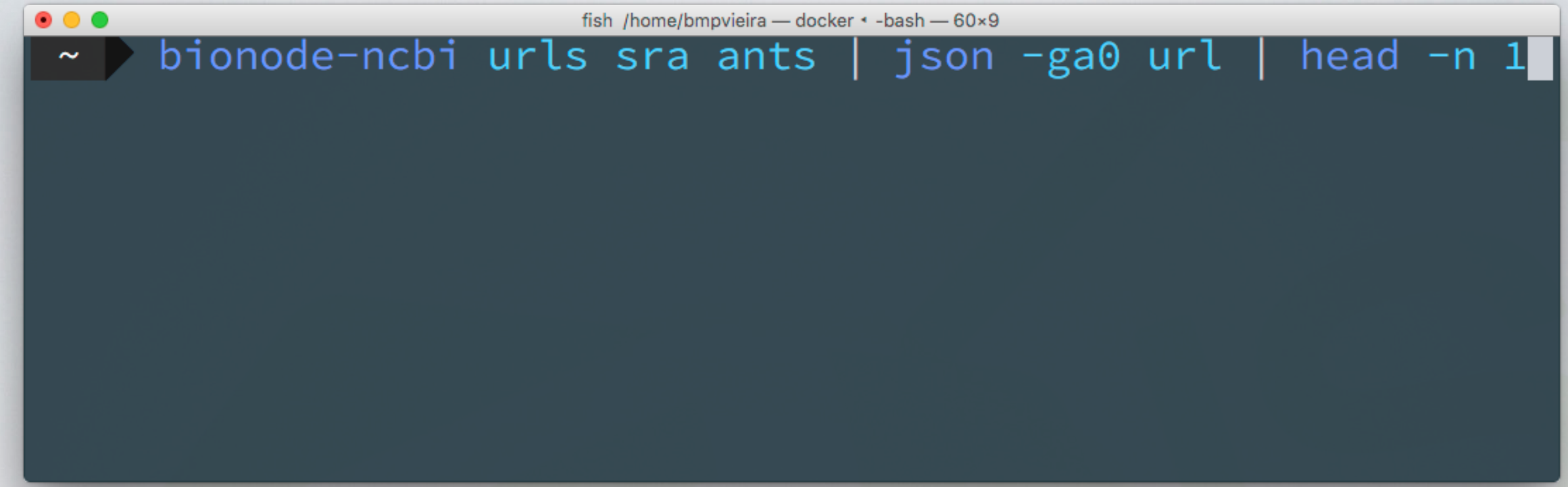
bionode-ncbi

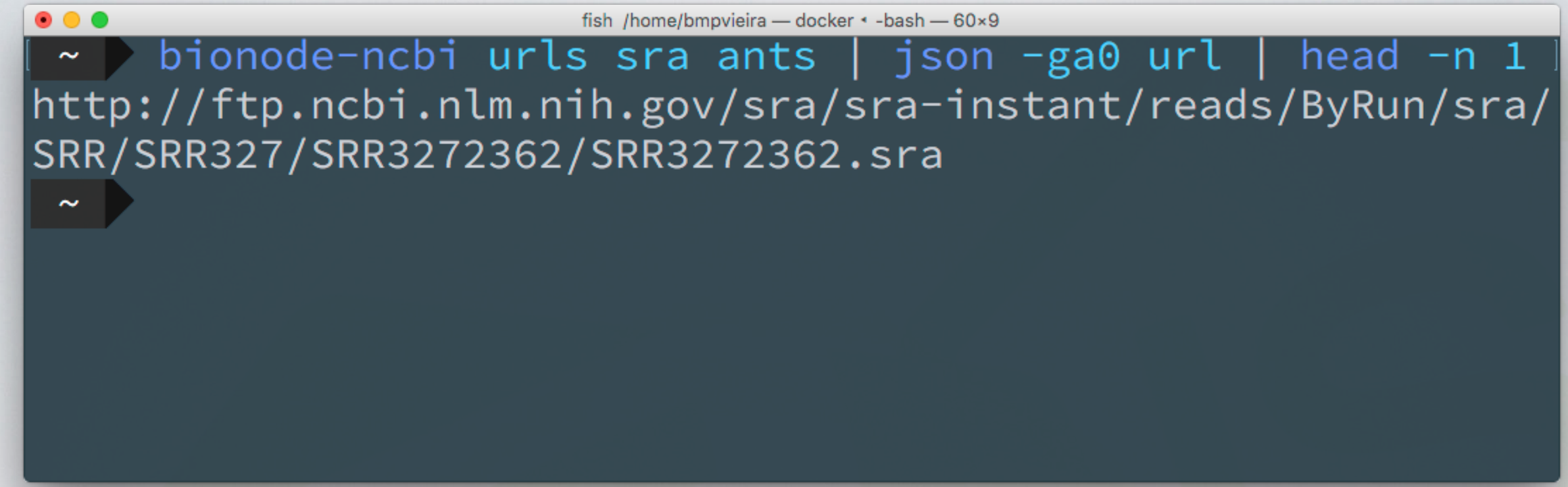
#bionodehack




fish /home/bmpvieira — docker • -bash — 60×9
~
bionode-ncbi urls sra ants


fish /home/bmpvieira — docker • -bash — 60×9
~ bionode-ncbi urls sra ants | json -ga0 url


fish /home/bmpvieira — docker • -bash — 60×9
~ bionode-ncbi urls sra ants | json -ga0 url | head -n 1


fish /home/bmpvieira — docker • -bash — 60×9
~ bionode-ncbi urls sra ants | json -ga0 url | head -n 1
http://ftp.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR327/SRR3272362/SRR3272362.sra
~

Had to find and get TB of data online

Social (red) - Solitary (blue)

TOTAL: 167

Blue solitary, Red social

Had to find and get TB of data online

Remove non-wild samples

TOTAL: 119

No Droso., Aedes, Anoph., Mus or R

Had to find and get TB of data online
Remove missing WGS

TOTAL: 88

Usable reads (genomic, wgs, etc)

Had to find and get TB of data online

Remove low coverage

TOTAL: 51

30x coverage

Had to find and get TB of data online
Remove unusable for PSMC

TOTAL: 45

30x manually curated (no haploids, p
inbreds, etc)

ORIGIN

During my PhD at

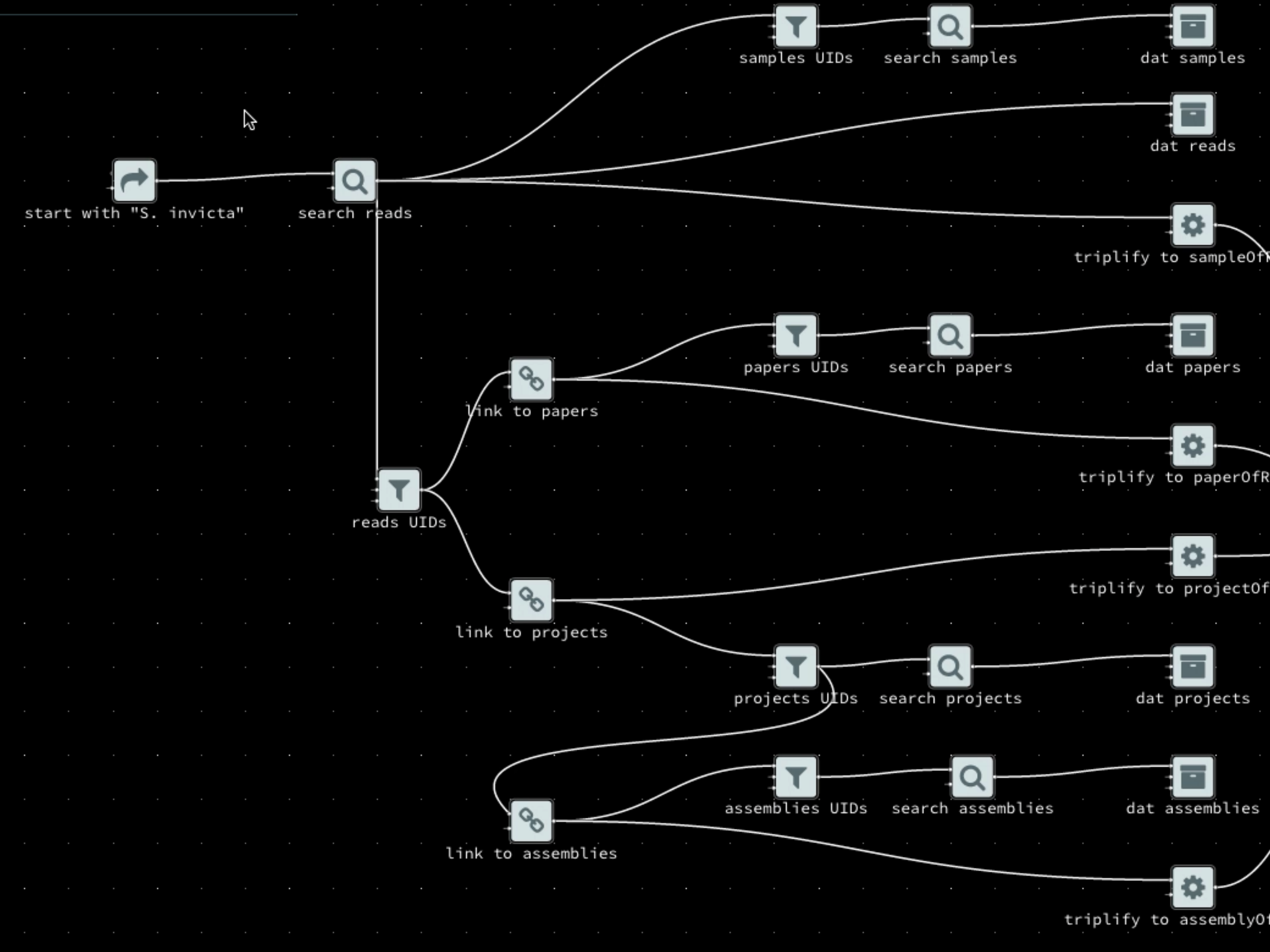
- Involved in biological web projects that need JS
- Had to find and get TB of data online

ORIGIN

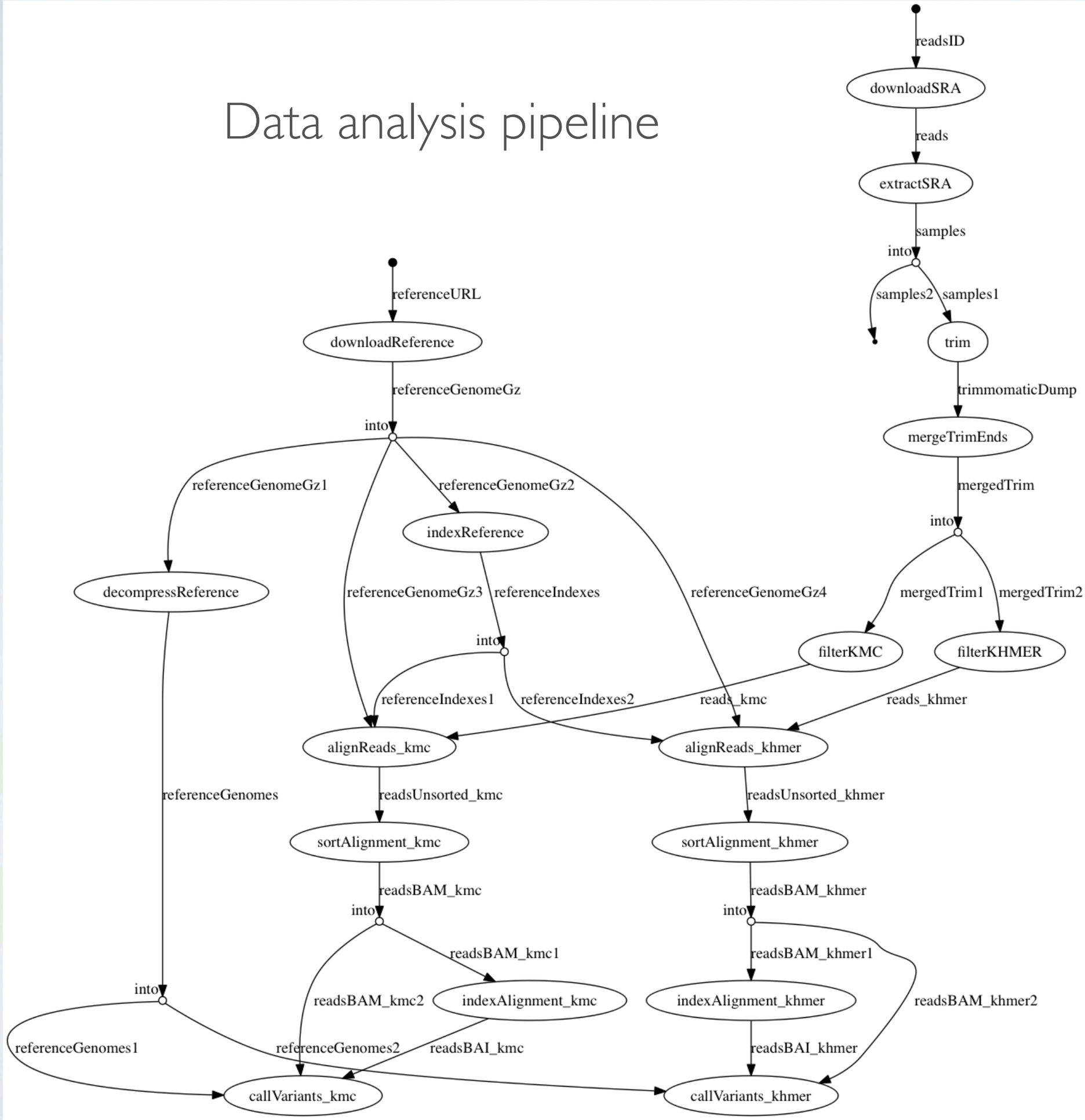
During my PhD at

- Involved in biological web projects that need JS
- Had to find and get TB of data online
- Had to build complicated bioinformatic pipelines

DATA PROCESSING


samples UIDs
search samples
dat samples
dat reads
start with "S. invicta"
search reads
triplify to sampleOf
papers UIDs
search papers
dat papers
link to papers
triplify to paperOfR
reads UIDs
triplify to projectOf
link to projects
projects UIDs
search projects
dat projects
assemblies UIDs
search assemblies
dat assemblies
link to assemblies
triplify to assemblyOf

Data analysis pipeline



Metadata pipeline

Metadata pipeline


```
ncbi
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)

fork1
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)

fork1
.pipe(tool.extractProperty('uid'))
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)

fork1
.pipe(tool.extractProperty('uid'))
.pipe(ncbi.link('sra', 'pubmed'))
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)

fork1
.pipe(tool.extractProperty('uid'))
.pipe(ncbi.link('sra', 'pubmed'))
.pipe(ncbi.search('pubmed'))
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)

fork1
.pipe(tool.extractProperty('uid'))
.pipe(ncbi.link('sra', 'pubmed'))
.pipe(ncbi.search('pubmed'))
.pipe(fork2)
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)

fork1
.pipe(tool.extractProperty('uid'))
.pipe(ncbi.link('sra', 'pubmed'))
.pipe(ncbi.search('pubmed'))
.pipe(fork2)
.pipe(dat.papers)
```

Metadata pipeline


```
ncbi
.search('sra', 'Solenopsis invicta')
.pipe(fork1)
.pipe(dat.reads)

fork1
.pipe(tool.extractProperty('expxml.Biosample.id'))
.pipe(ncbi.search('biosample'))
.pipe(dat.samples)

fork1
.pipe(tool.extractProperty('uid'))
.pipe(ncbi.link('sra', 'pubmed'))
.pipe(ncbi.search('pubmed'))
.pipe(fork2)
.pipe(dat.papers)
```

4. ↣ fish: fish /Users/bmpvieira/Local/bionode-pipelines/pipeline-fetch

```
~/L/b/pipeline-fetch node pipeline-fetch.js
```

4. ↣ fish: fish /Users/bmpvieira/Local/bionode-pipelines/pipeline-fetch

```
~/L/b/pipeline-fetch node pipeline-fetch.js
```

bionode-waterwheel

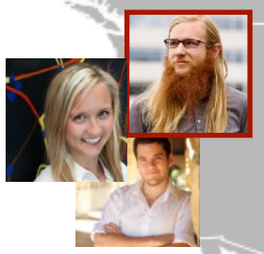
A Streaming Workflow Engine for bioinformatics and other big data explorations

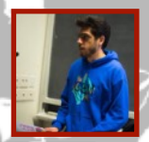
Google Summer of Code 2016

OIBIF

```
const samples = task({
  input: {
    db: 'sra',
    accession: config.sraAccession
  },
  output: '**/*.sra'
}, ({ input }) => ncbi.download(input.db, input.accession) )

const fastqDump = task({
  input: new File('**/*.sra'),
  output: [1, 2].map(n => new File(`*_${n}.fastq.gz`))
}, ({ input }) => shell(`fastq-dump --split-files --skip-technical --gzip ${input}`) )
```

BIONODE COMMUNITY






WurmLab
.github.io

Queen Mary
University of London

DNA
digest

dat
DATA

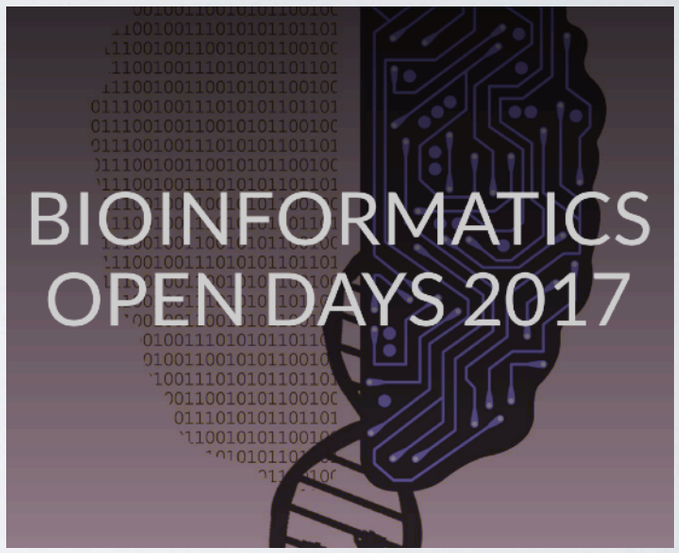
BioJS

ACKNOWLEDGMENTS

Research group

Community

BOD2017



Mozilla Science Lab

Fellowship Founder

THE LEONA M. AND HARRY B.
HELMSLEY
CHARITABLE TRUST

Friends