



BIONODE.IO

Introduction

github.com/bionode-hack/discussions

gitter.im/bionode-hack/discussions

WHAT

Modular and universal bioinformatics

- 👉 Tries to do one thing well
- ♻️ Provides highly reusable code and tools
- ↳ Scales by using Streams
- 🌐 Runs everywhere

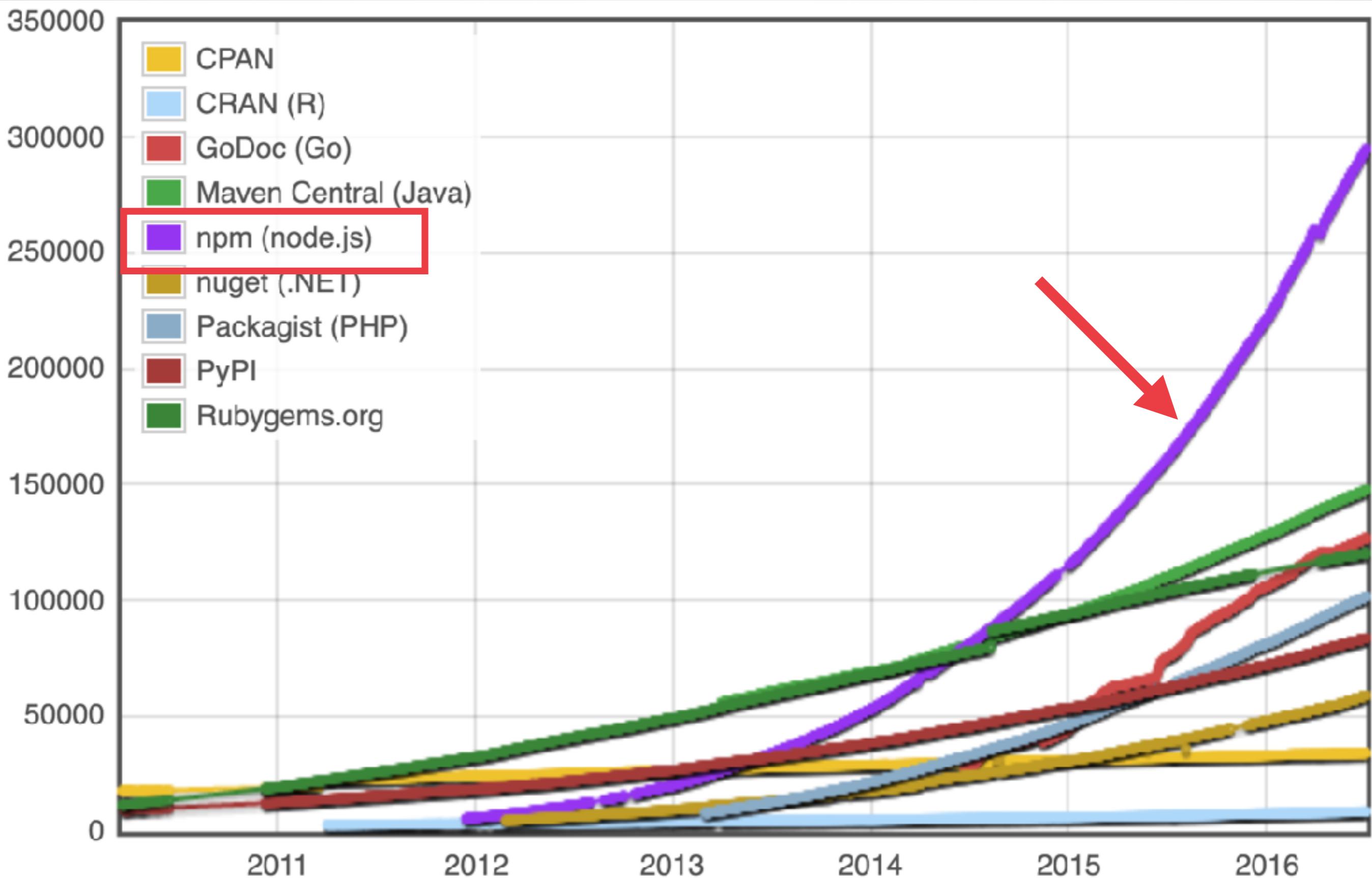
HOW

Using Node.js

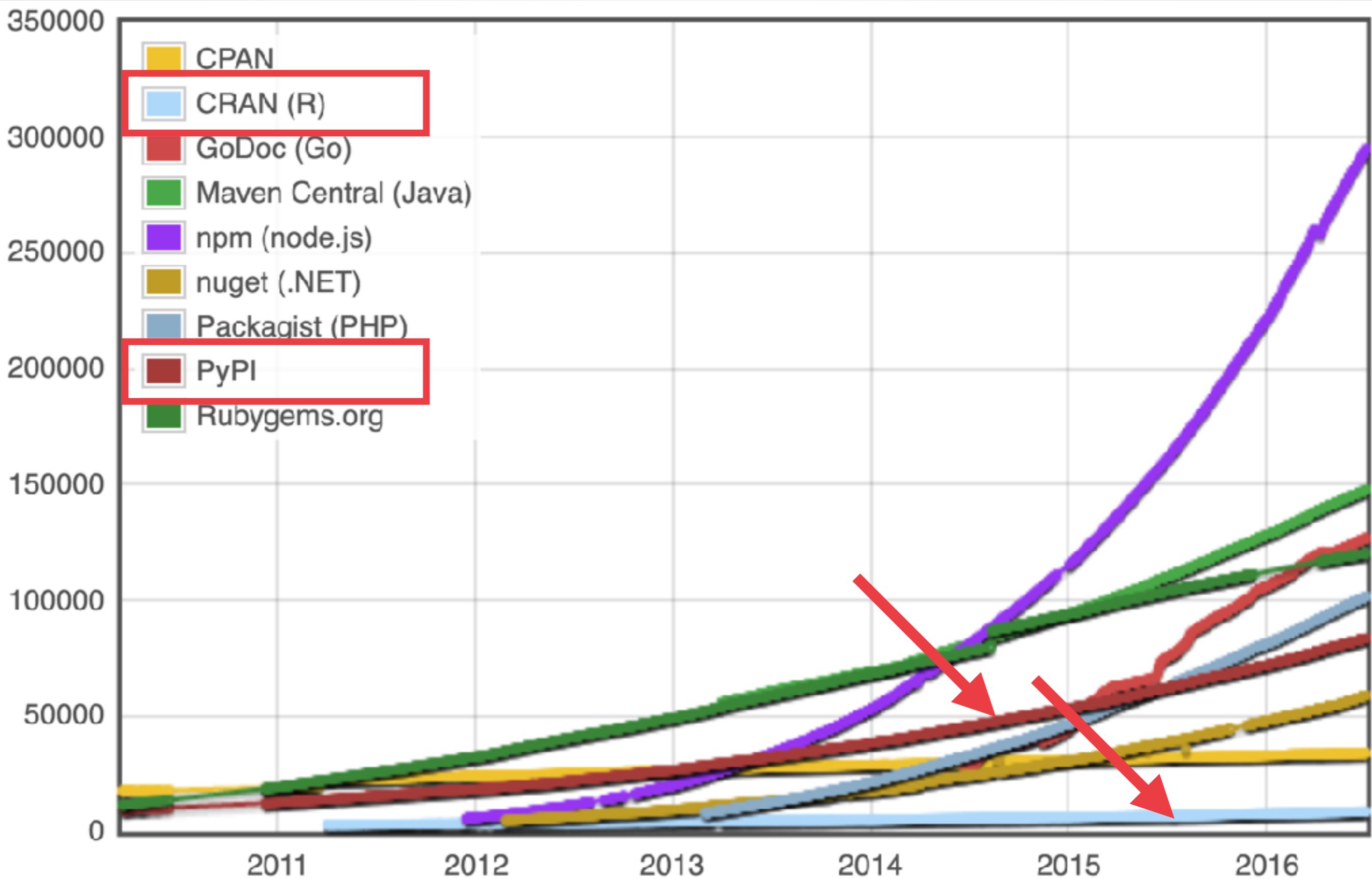
👉 Highly modular

♻️ Very open community on GitHub

Modules count



Modules count



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

FASTA file format

> X-gene

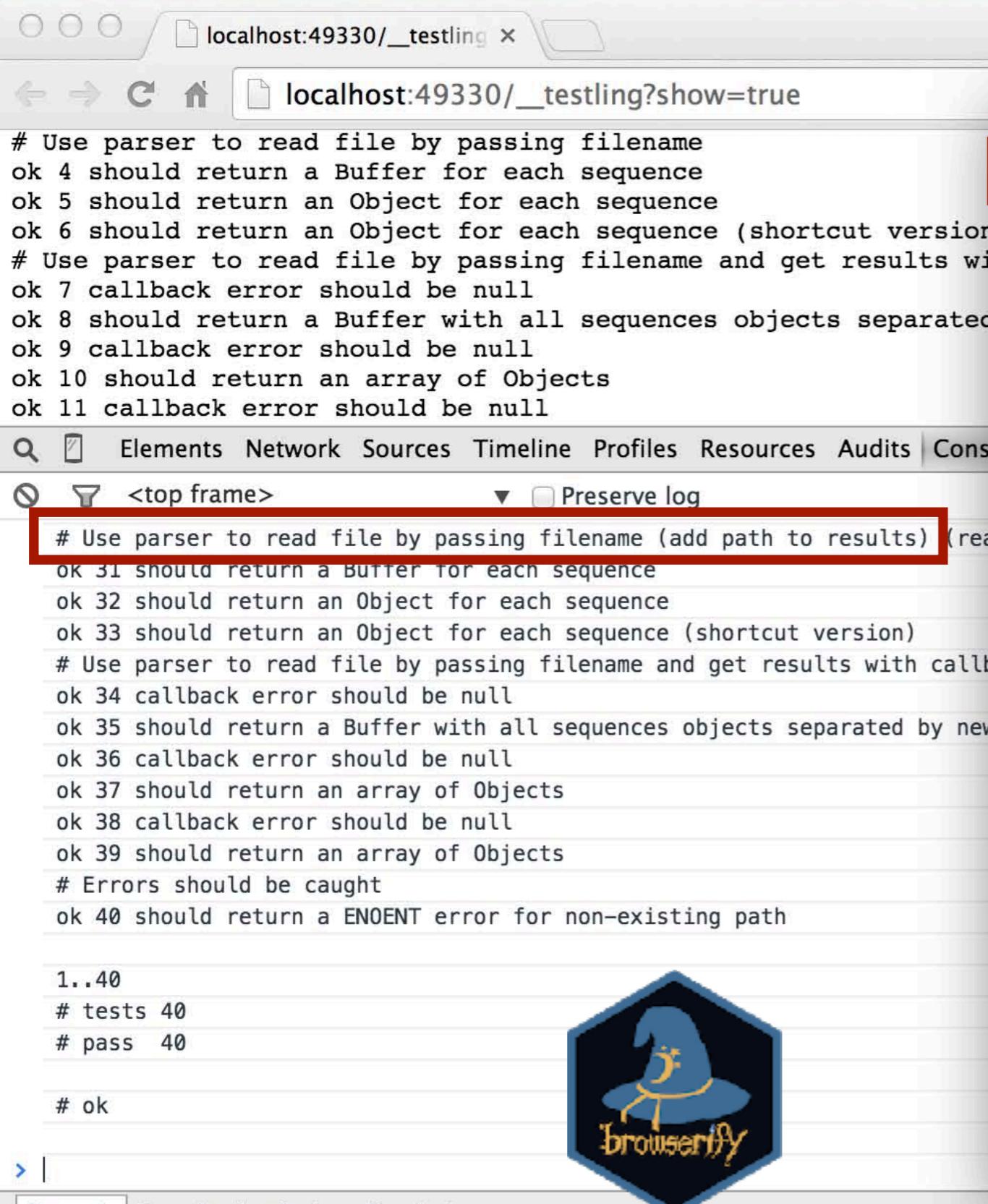
ATGCGTACTGCATCATG
ACGTACTGCATTCATGC
GTGCTAGGGGTTACGT

JSON (JavaScript Object Notation)

```
{  
  "id": "X-gene",  
  "seq": "ATGCGTACTGCATCAT  
GACGTACTGCATTCATGCGTG  
CTAGGGGTTACGT"  
}
```

```
3. ➔ fish: fish /Users/bmpvieira/Local/bionode-fasta  
✓ should return an array of Objects  
  Use parser to read file by passing filename (add path to results)  
    ✓ should return a Buffer for each sequence  
    ✓ should return an Object for each sequence  
    ✓ should return an Object for each sequence (shortcut version)  
  
  Use parser to read file by passing filename and get results with call  
  results)  
    ✓ callback error should be null  
    ✓ should return a Buffer with all sequences objects separated by ne  
    ✓ callback error should be null  
    ✓ should return an array of Objects  
    ✓ callback error should be null  
    ✓ should return an array of Objects  
  
  Use parser to read file by passing filename (add path to results) (re  
    ✓ should return a Buffer for each sequence  
    ✓ should return an Object for each sequence  
    ✓ should return an Object for each sequence (shortcut version)  
  
  Use parser to read file by passing filename and get results with call  
  results) (read gzipped file)  
    ✓ callback error should be null  
    ✓ should return a Buffer with all sequences objects separated by ne  
    ✓ callback error should be null  
    ✓ should return an array of Objects  
    ✓ callback error should be null  
    ✓ should return an array of Objects  
  
  Errors should be caught  
    ✓ should return a ENOENT error for non-existing path  
  
tests 40  
pass 40  
  
Pass!  
~/L/bionode-fasta ➤ feature/browser-testing-working ➤ npm test
```

Run same JavaScript code on browser or CLI

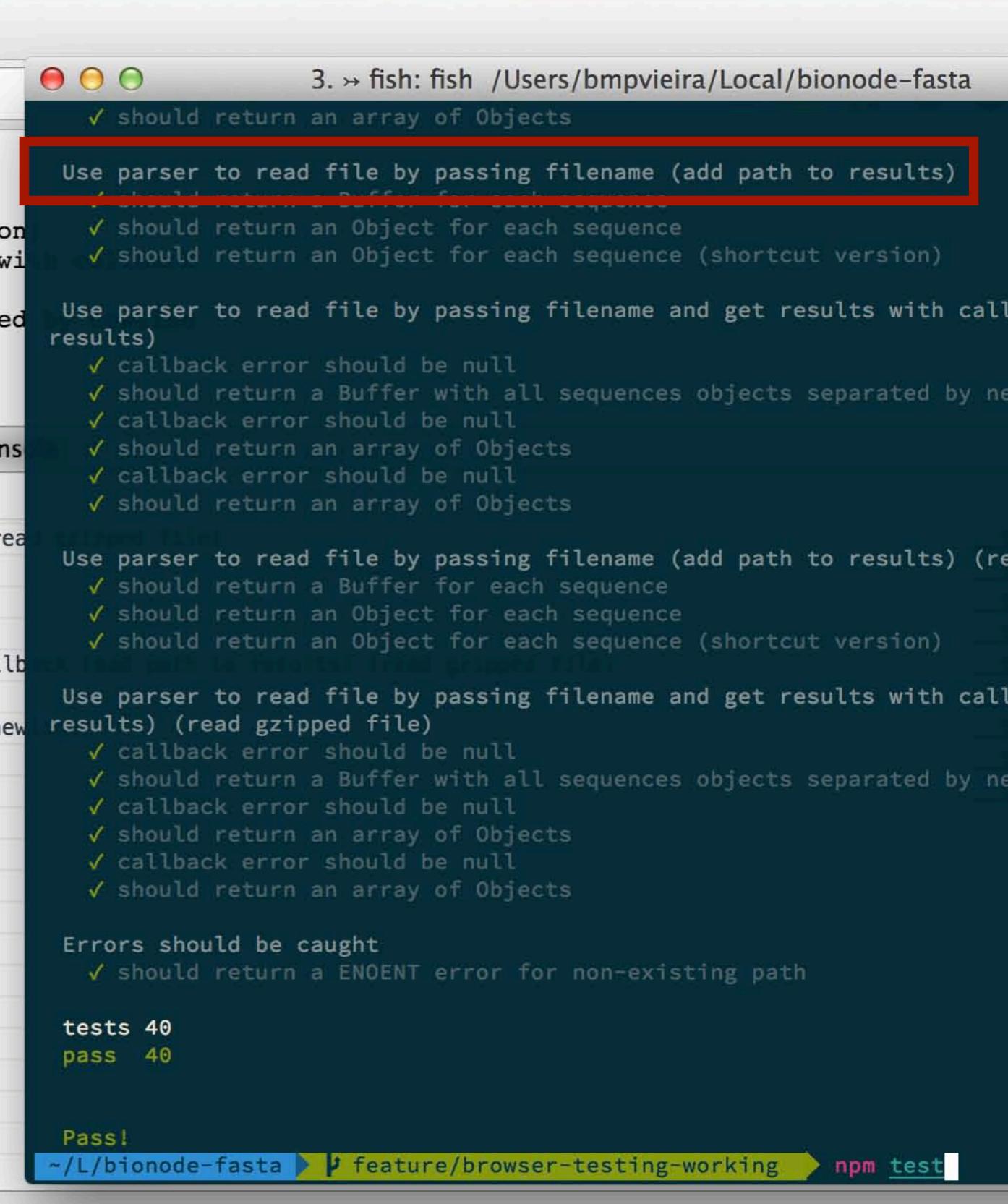
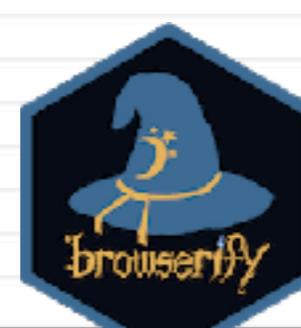


```
# Use parser to read file by passing filename
ok 4 should return a Buffer for each sequence
ok 5 should return an Object for each sequence
ok 6 should return an Object for each sequence (shortcut version)
# Use parser to read file by passing filename and get results with callback
ok 7 callback error should be null
ok 8 should return a Buffer with all sequences objects separated by new line
ok 9 callback error should be null
ok 10 should return an array of Objects
ok 11 callback error should be null

# Elements Network Sources Timeline Profiles Resources Audits Cons
<top frame> ▾ □ Preserve log

# Use parser to read file by passing filename (add path to results) (read)
ok 31 should return a Buffer for each sequence
ok 32 should return an Object for each sequence
ok 33 should return an Object for each sequence (shortcut version)
# Use parser to read file by passing filename and get results with callback
ok 34 callback error should be null
ok 35 should return a Buffer with all sequences objects separated by new line
ok 36 callback error should be null
ok 37 should return an array of Objects
ok 38 callback error should be null
ok 39 should return an array of Objects
# Errors should be caught
ok 40 should return a ENOENT error for non-existing path

1..40
# tests 40
# pass 40
# ok
```



```
3. ➔ fish: fish /Users/bmpvieira/Local/bionode-fasta
✓ should return an array of Objects

Use parser to read file by passing filename (add path to results)
✓ should return a Buffer for each sequence
✓ should return an Object for each sequence
✓ should return an Object for each sequence (shortcut version)

Use parser to read file by passing filename and get results with callback
✓ callback error should be null
✓ should return a Buffer with all sequences objects separated by new line
✓ callback error should be null
✓ should return an array of Objects
✓ callback error should be null
✓ should return an array of Objects

Use parser to read file by passing filename (add path to results) (read)
✓ should return a Buffer for each sequence
✓ should return an Object for each sequence
✓ should return an Object for each sequence (shortcut version)

Use parser to read file by passing filename and get results with callback (read gzipped file)
✓ callback error should be null
✓ should return a Buffer with all sequences objects separated by new line
✓ callback error should be null
✓ should return an array of Objects
✓ callback error should be null
✓ should return an array of Objects

Errors should be caught
✓ should return a ENOENT error for non-existing path

tests 40
pass 40

Pass!
~/L/bionode-fasta ➤ feature/browser-testing-working ➤ npm test
```

 Run same JavaScript code on browser or CLI

JavaScript not suitable for heavy scientific computation?

C++ preferred?

Run C++ from JavaScript



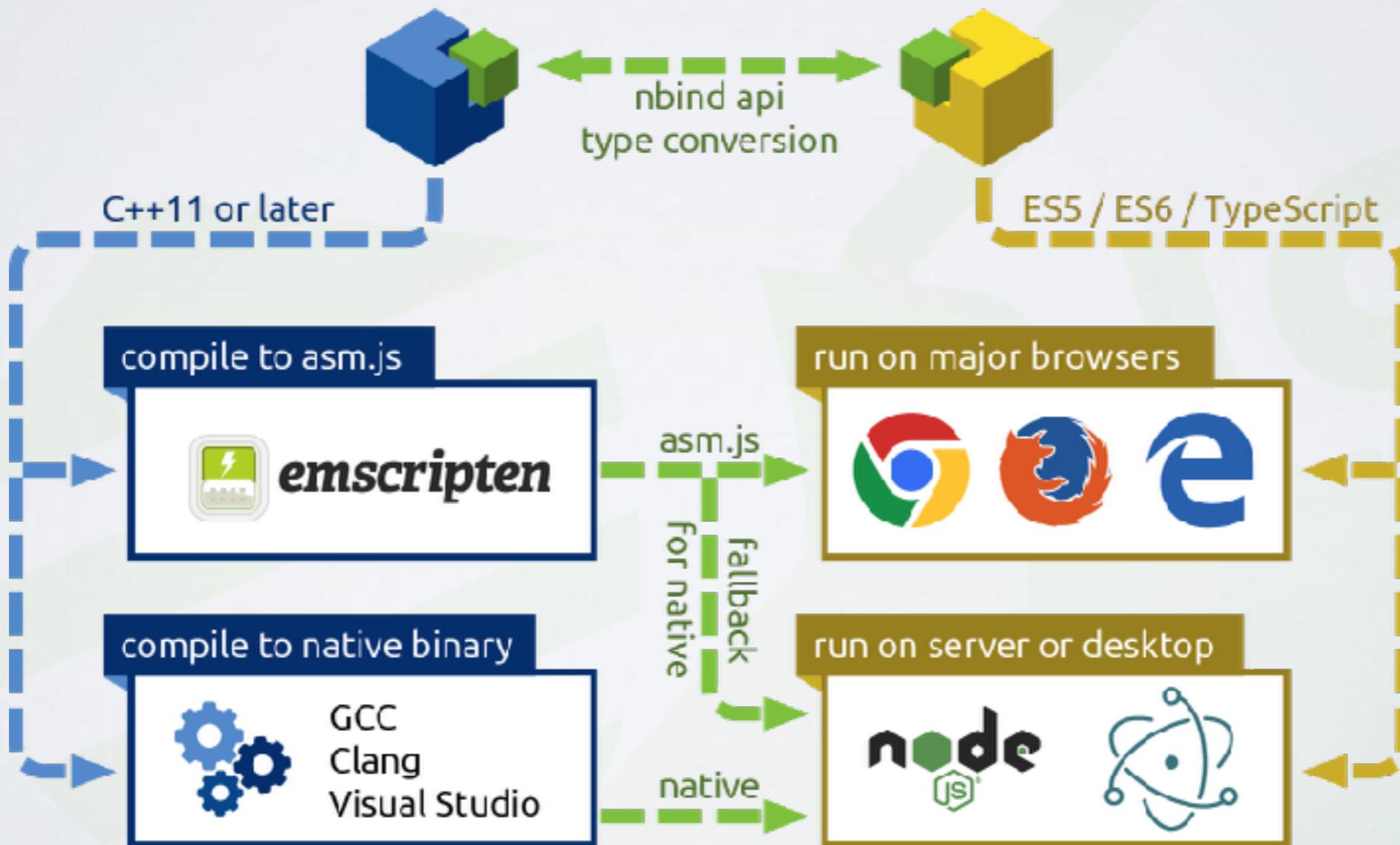
nbind

Icons made by Freepik and Icomoon from www.flaticon.com and this diagram made by BusFaster Ltd are licensed under



All code is under MIT.

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



ORIGIN

During my PhD at



- Involved in biological web projects that need JS

Involved in biological web projects that need JS

wurmlab.github.io

The screenshot shows the Afra web application interface. At the top, there's a menu bar with options like File, Edit, View, History, Bookmarks, Develop, Window, Help, and a toolbar with various icons. The title bar says "Afra". Below the title bar, the URL "afra.sbs.qmul.ac.uk/curate" is visible.

The main area displays genomic tracks for different sources:

- NAKER:** Shows three transcripts: "maker-SI_gnF%2Escaffold02694-augustus-gene-0.10-mRNA-1" (top), "maker-SI_gnF%2Escaffold02694-snap-gene-1.65-mRNA-1" (middle), and "maker-SI_gnF%2Escaffold02694-snap-gene-1.67-mRNA-1" (bottom).
- Augustus:** Shows one transcript: "augustus_masked-SI_gnF%2Escaffold02694-abinit-gene-1.0-mRNA-1".
- SNAP:** Shows two transcripts: "snap_masked-SI_gnF%2Escaffold02694-abinit-gene-0.2-mRNA-1" (top) and "snap_masked-SI_gnF%2Escaffold02694-abinit-gene-1.35-mRNA-1" (bottom).
- est2genome:** Shows five transcripts: "SI_estOR100817Isotig03162", "SI_estOR100817Isotig03160", "SI_estOR100817Isotig03161", "SI_estOR100817Isotig03163", and "SI_estOR100817Isotig03164". A new transcript "SI_estOR100817Isotig13200" is being added at the bottom right.
- protein2genome:** Shows two transcripts: "SI2.2.0_05723" and "NV17819-PA".

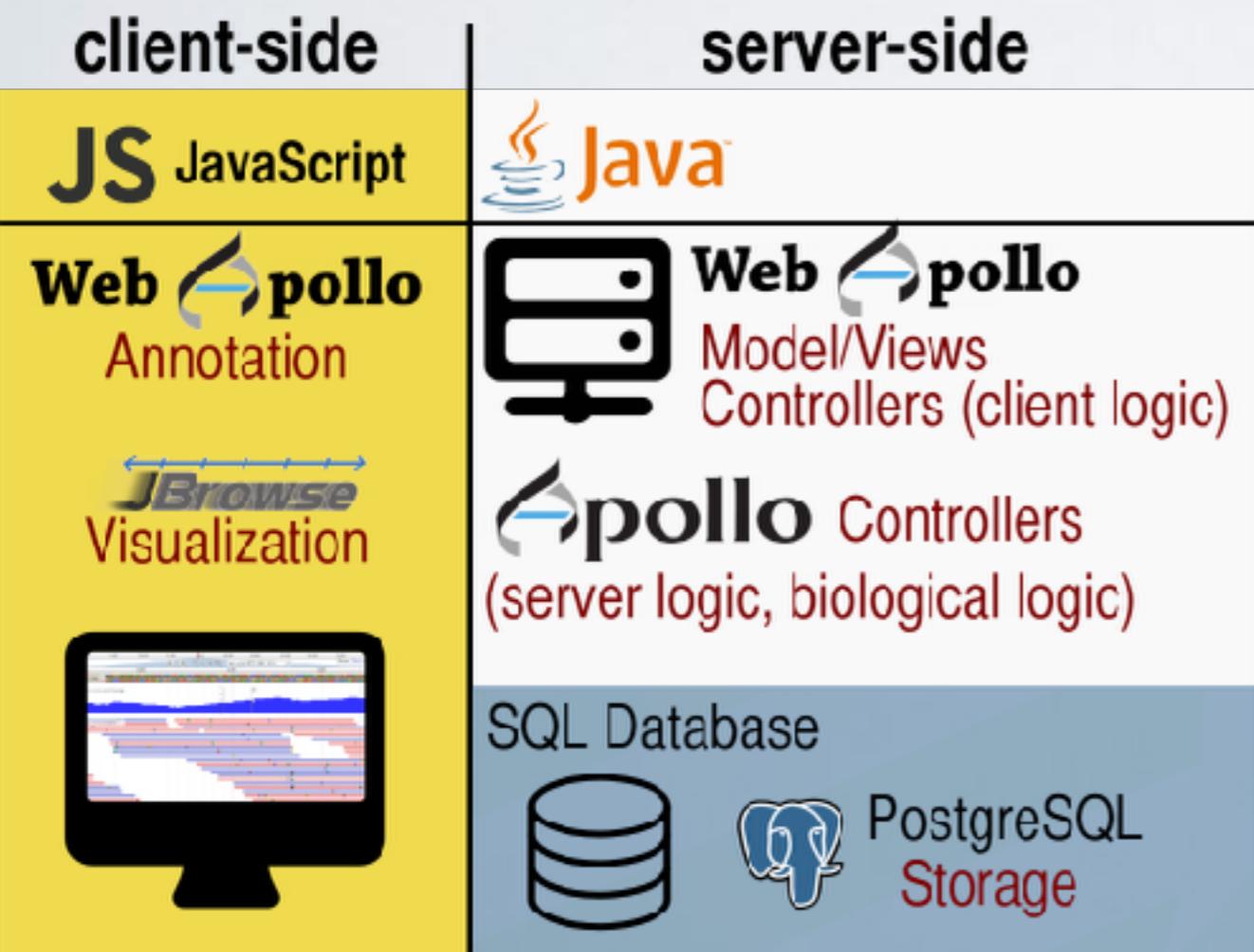
On the right side, there are several panels with analysis tools:

- Consistency with available EST data**
- Splice sites**
- Translation sites**
- Consistency with homologs and transcriptomes**

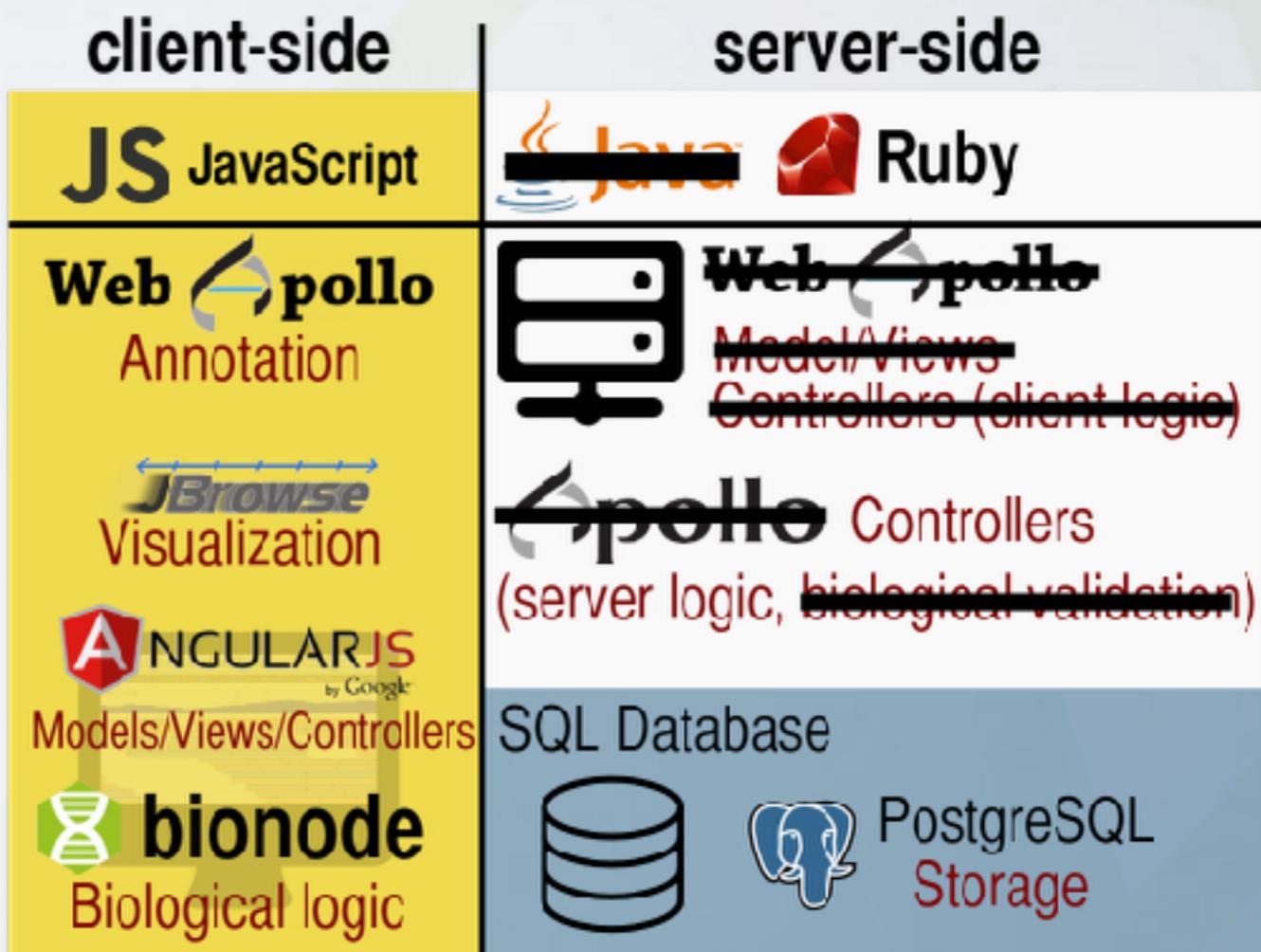
At the bottom right, there's a "Done" button with a checked checkbox.

Involved in biological web projects that need JS
wurmlab.github.io

GMOD



Afra



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(/[EFIJL0PQZX\*]/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("ATGACCTGAAGGTGAA");
=> "AAGTGGAAAGTCCCAGTA"
```

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

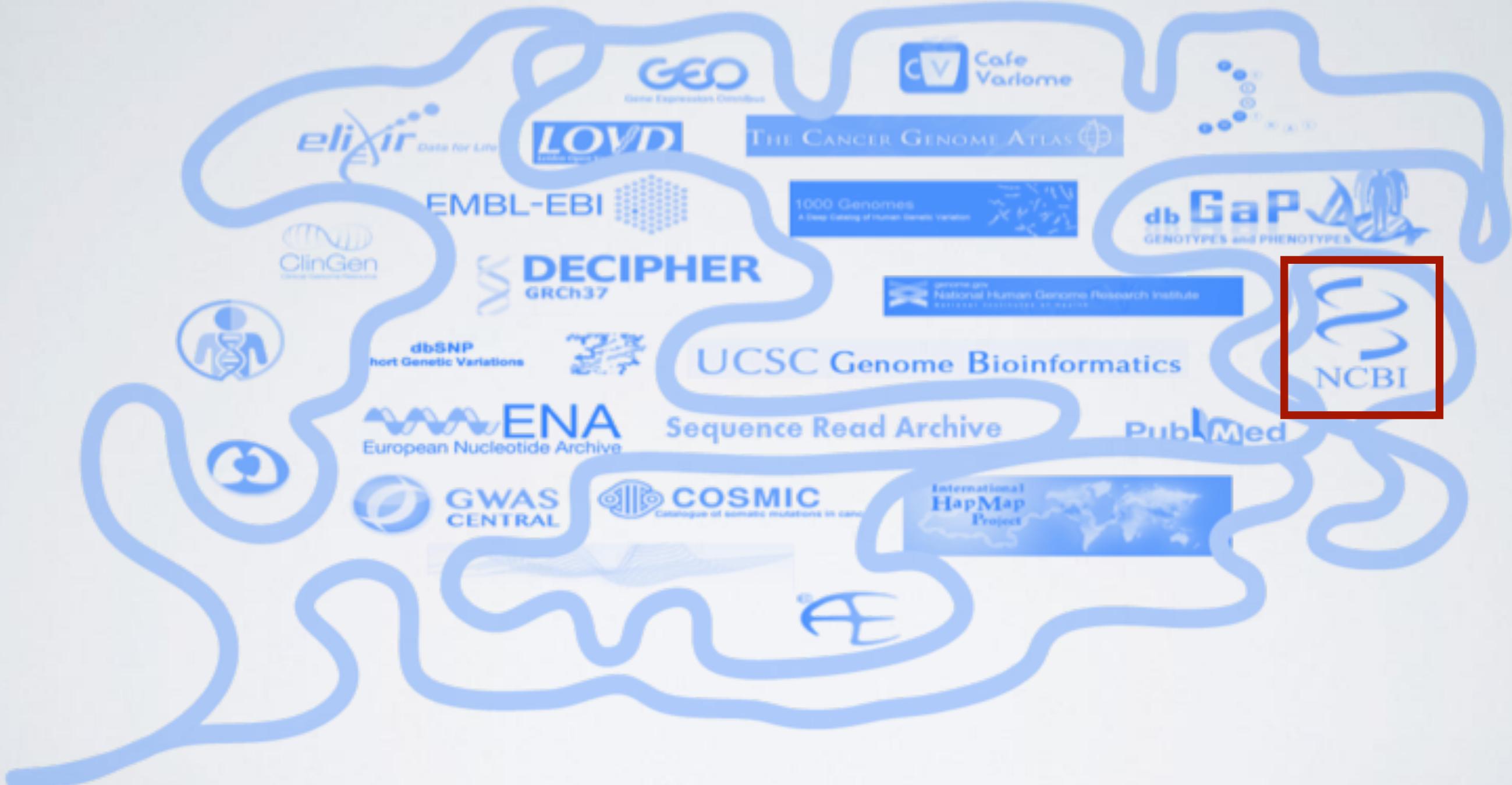


- Involved in biological web projects that need JS
- Had to find and get TB of data online

Compare genetic diversity of social vs solitary species

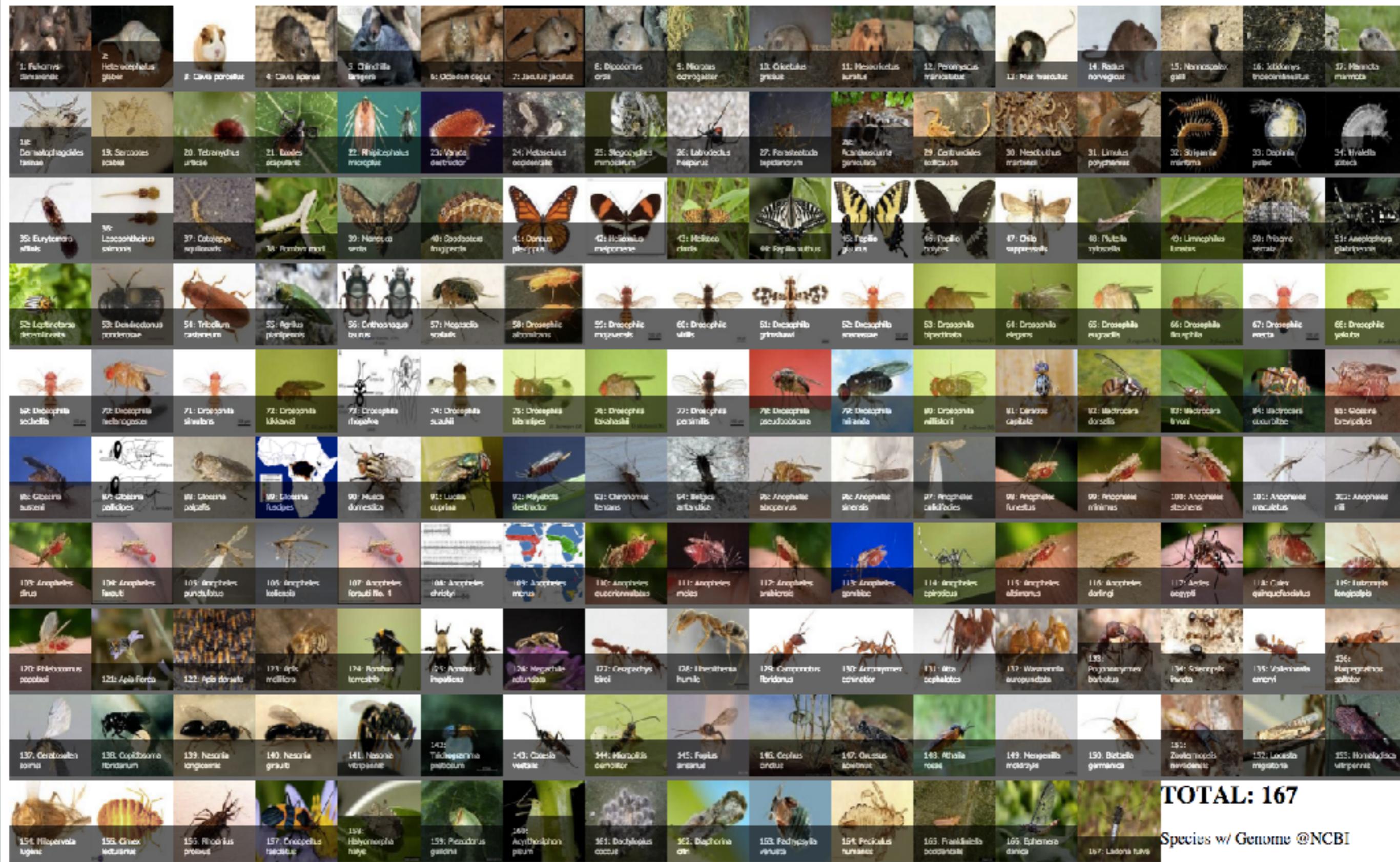


Maze of data sources



Had to find and get TB of data online

All Insects and Rodents



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.I_Si_gnG`

fish /home/bmpvieira/project-phd — docker - bash — 131x42

```
~/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",
    "gff": "http://ftp.ncbi.nlm.nih.gov/genomes/all/GCA_000611835.1_CerBir1.0/GCA_000611835.1_CerBir1.0_genomic.gff.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 ➤
```

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

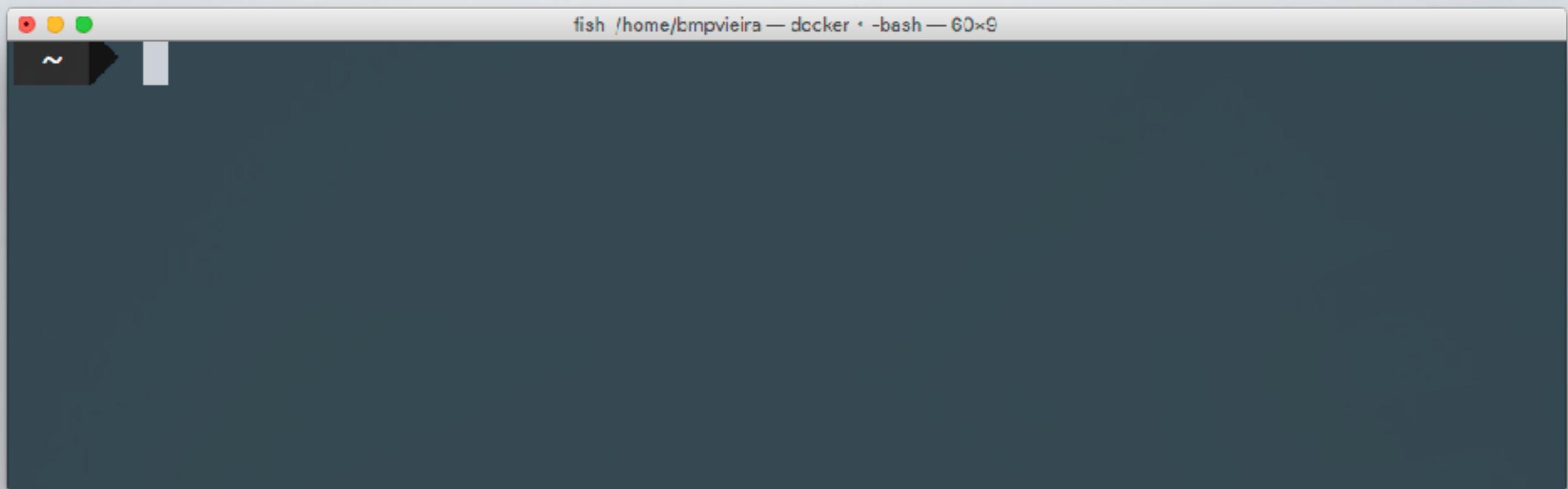
// Callback pattern
bio.ncbi.urls('assembly', 'ants', function(urls) {
  console.log(urls[0].genomic.fna)
})

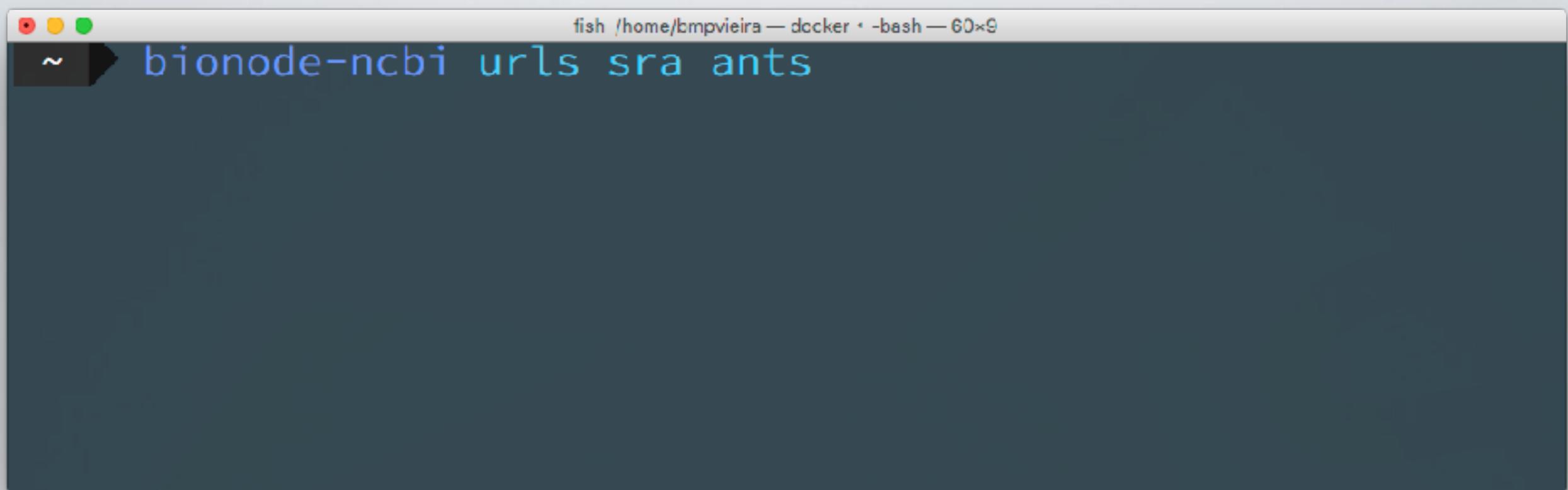
// Event pattern
bio.ncbi.urls('assembly', 'ants')
.on('data', printGenomeURL)

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

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

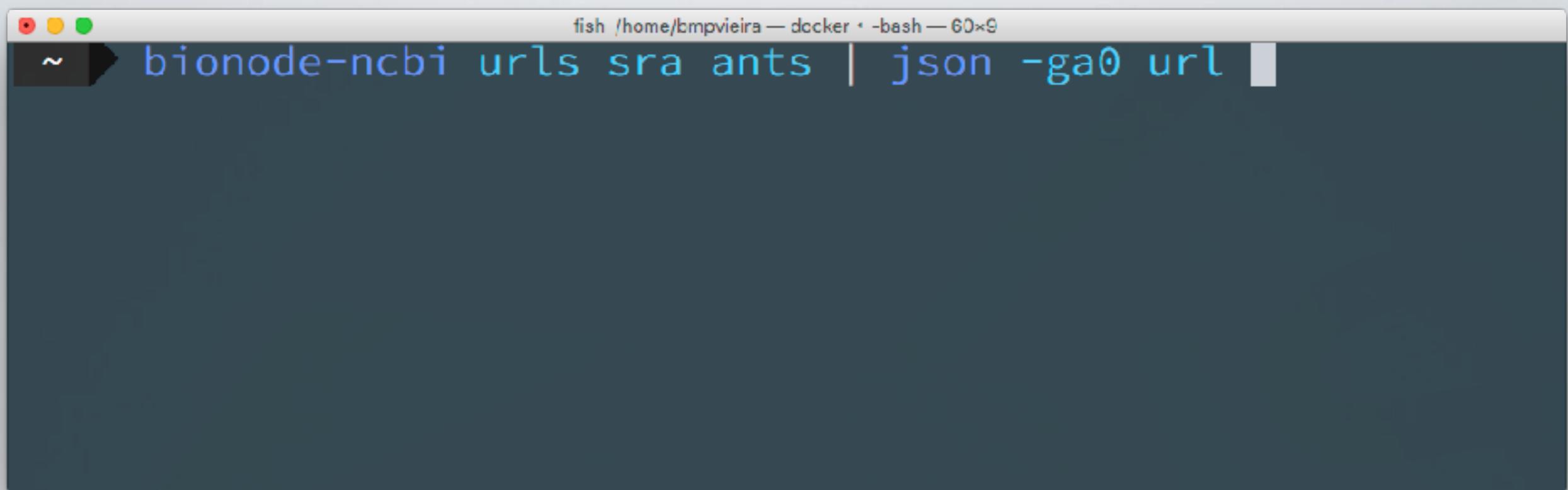
ncbi.urls('genomes', 'ants')
  .pipe(ndjson.stringify())
  .pipe(process.stdout)
```





A screenshot of a terminal window with a dark background and light-colored text. The window title bar shows "fish /home/bmpvieira — docker · -bash — 60x9". The command "bionode-ncbi urls sra ants" is being typed into the terminal. The terminal has a standard OS X-style window frame with red, yellow, and green buttons.

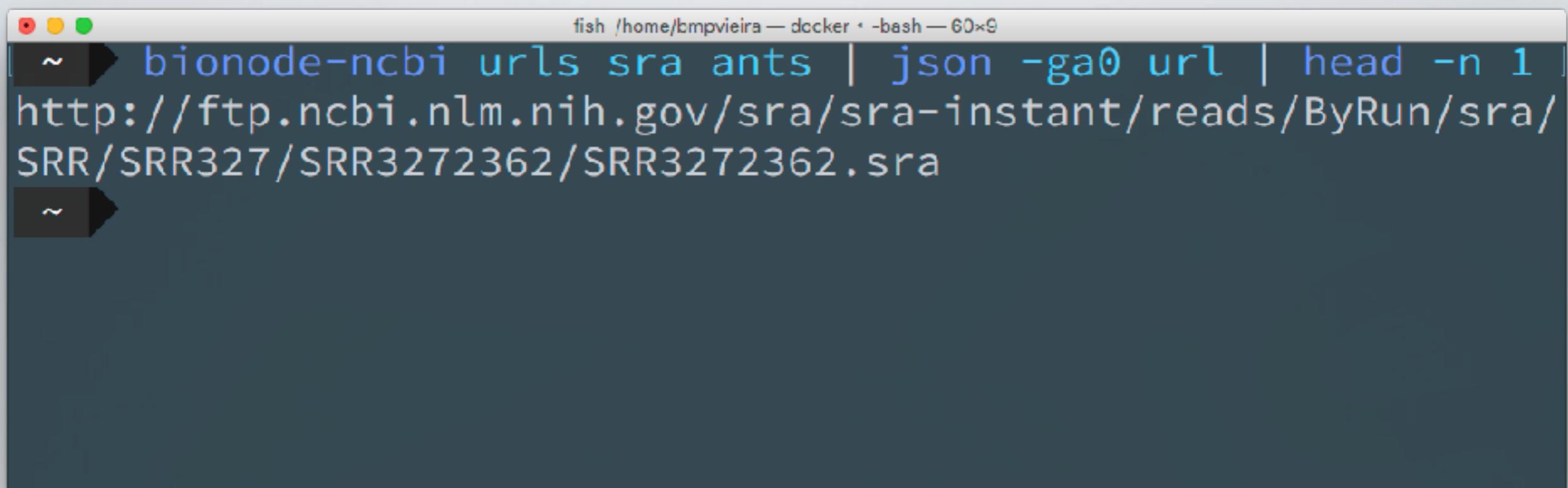
```
bionode-ncbi urls sra ants
```



A screenshot of a terminal window with a dark background and light-colored text. The window title bar shows "fish /home/bmpvieira — docker · -bash — 60x9". The command entered is:

```
bionode-ncbi urls sra ants | json -ga0 url
```

```
fish /home/bmpvieira — docker · -bash — 60x9
~ ➞ bionode-ncbi urls sra ants | json -ga0 url | head -n 1
```



```
fish /home/bmpvieira — docker — bash — 60x9
~ ➤ 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



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



Had to find and get TB of data online

Remove unusable for PSMC

1: <i>Folomyia</i> <i>discrepans</i>	2: <i>Heteroccephala</i> <i>gaster</i>	3: <i>Cnephomyia</i> <i>polysticta</i>	4: <i>Cnephomyia</i> <i>apicalis</i>	5: <i>Gimphila</i> <i>laevigata</i>	6: <i>Elachiptera</i> <i>delegat</i>	7: <i>Sarcophaga</i> <i>jeanneli</i>	8: <i>Dipteromyza</i> <i>antennalis</i>	9: <i>Miltogramma</i> <i>eximia</i>	10: <i>Ochetellus</i> <i>griseus</i>	11: <i>Neacanthella</i> <i>australis</i>	12: <i>Pseudaletia</i> <i>metamicta</i>	13: <i>Macrostemum</i> <i>heteropteron</i>	14: <i>Rhabdotellus</i> <i>longigena</i>	15: <i>Neacanthella</i> <i>pusilla</i>	16: <i>Tellulomyia</i> <i>intermittens</i>	17: <i>Neacanthella</i> <i>minima</i>
18: <i>Dipterophagidae</i> <i>fallax</i>	19: <i>Cnephomyia</i> <i>sootae</i>	20: <i>Tetanoclytus</i> <i>utriculus</i>	21: <i>baccha</i> <i>capitata</i>	22: <i>Miltogramma</i> <i>imperialis</i>	23: <i>Nemotelus</i> <i>decoratus</i>	24: <i>Neoteknoides</i> <i>orientalis</i>	25: <i>Diopeltides</i> <i>minutissima</i>	26: <i>Lethocerus</i> <i>hispinus</i>	27: <i>Forcipomyia</i> <i>repanda</i>	28: <i>Acanthocera</i> <i>genitalis</i>	29: <i>Catolaccida</i> <i>entomophaga</i>	30: <i>Macrostemum</i> <i>metamicta</i>	31: <i>Unkaka</i> <i>polyphemus</i>	32: <i>Sciaridae</i> <i>sootae</i>	33: <i>Cephalcia</i> <i>picea</i>	34: <i>Hydrellia</i> <i>sootae</i>
35: <i>Leptophlebiidae</i> <i>strobli</i>	36: <i>Cnephomyia</i> <i>equivalens</i>	37: <i>Forcipomyia</i> <i>metamicta</i>	38: <i>Forcipomyia</i> <i>metamicta</i>	39: <i>Macula</i> <i>sootae</i>	40: <i>Sciaridae</i> <i>sootae</i>	41: <i>Sciaridae</i> <i>sootae</i>	42: <i>Sciaridae</i> <i>sootae</i>	43: <i>Sciaridae</i> <i>sootae</i>	44: <i>Forcipomyia</i> <i>metamicta</i>	45: <i>Forcipomyia</i> <i>metamicta</i>	46: <i>Forcipomyia</i> <i>metamicta</i>	47: <i>Forcipomyia</i> <i>metamicta</i>	48: <i>Forcipomyia</i> <i>metamicta</i>	49: <i>Forcipomyia</i> <i>metamicta</i>	50: <i>Limnephilus</i> <i>lunatus</i>	51: <i>Sciaridae</i> <i>sootae</i>
52: <i>Leptophlebiidae</i> <i>decorata</i>	53: <i>Forcipomyia</i> <i>pondicola</i>	54: <i>Forcipomyia</i> <i>ostentum</i>	55: <i>Agathomyia</i> <i>pilosa</i>	56: <i>Onthophagus</i> <i>taeniatus</i>	57: <i>Megarilla</i> <i>solita</i>	58: <i>Forcipomyia</i> <i>decorata</i>	59: <i>Cnephomyia</i> <i>imperialis</i>	60: <i>Forcipomyia</i> <i>villo</i>	61: <i>Forcipomyia</i> <i>prinoleuca</i>	62: <i>Forcipomyia</i> <i>decorata</i>	63: <i>Forcipomyia</i> <i>decorata</i>	64: <i>Forcipomyia</i> <i>decorata</i>	65: <i>Forcipomyia</i> <i>decorata</i>	66: <i>Forcipomyia</i> <i>decorata</i>	67: <i>Forcipomyia</i> <i>decorata</i>	68: <i>Forcipomyia</i> <i>decorata</i>
69: <i>Diopsidae</i> <i>sootae</i>	70: <i>Diopsidae</i> <i>metamicta</i>	71: <i>Diopsidae</i> <i>decorata</i>	72: <i>Diopsidae</i> <i>sootae</i>	73: <i>Diopsidae</i> <i>sootae</i>	74: <i>Diopsidae</i> <i>sootae</i>	75: <i>Diopsidae</i> <i>sootae</i>	76: <i>Diopsidae</i> <i>sootae</i>	77: <i>Diopsidae</i> <i>sootae</i>	78: <i>Diopsidae</i> <i>sootae</i>	79: <i>Diopsidae</i> <i>sootae</i>	80: <i>Diopsidae</i> <i>sootae</i>	81: <i>Osettia</i> <i>capitata</i>	82: <i>Bacchurina</i> <i>decorata</i>	83: <i>Bacchurina</i> <i>tryoni</i>	84: <i>Bacchurina</i> <i>decorata</i>	85: <i>Bacchurina</i> <i>decorata</i>
86: <i>Closteromyces</i> <i>sootae</i>	87: <i>Closteromyces</i> <i>sootae</i>	88: <i>Closteromyces</i> <i>sootae</i>	89: <i>Closteromyces</i> <i>sootae</i>	90: <i>Musa</i> <i>constricta</i>	91: <i>Lucilia</i> <i>nummer</i>	92: <i>Neurobola</i> <i>decorata</i>	93: <i>Chironomus</i> <i>temerarius</i>	94: <i>Belobasis</i> <i>decorata</i>	95: <i>Anopheles</i> <i>stephensi</i>	96: <i>Anopheles</i> <i>strobli</i>	97: <i>Anopheles</i> <i>coluzzii</i>	98: <i>Anopheles</i> <i>coluzzii</i>	99: <i>Anopheles</i> <i>strobli</i>	100: <i>Anopheles</i> <i>strobli</i>	101: <i>Anopheles</i> <i>strobli</i>	102: <i>Anopheles</i> <i>strobli</i>
103: <i>Anopheles</i> <i>strobli</i>	104: <i>Anopheles</i> <i>strobli</i>	105: <i>Anopheles</i> <i>strobli</i>	106: <i>Anopheles</i> <i>strobli</i>	107: <i>Anopheles</i> <i>strobli</i>	108: <i>Anopheles</i> <i>strobli</i>	109: <i>Anopheles</i> <i>strobli</i>	110: <i>Anopheles</i> <i>strobli</i>	111: <i>Anopheles</i> <i>strobli</i>	112: <i>Anopheles</i> <i>strobli</i>	113: <i>Anopheles</i> <i>strobli</i>	114: <i>Anopheles</i> <i>strobli</i>	115: <i>Anopheles</i> <i>strobli</i>	116: <i>Anopheles</i> <i>strobli</i>	117: <i>Anopheles</i> <i>strobli</i>	118: <i>Anopheles</i> <i>strobli</i>	
119: <i>Anopheles</i> <i>strobli</i>	120: <i>Anopheles</i> <i>strobli</i>	121: <i>Aeta</i> <i>forficata</i>	122: <i>Aeta</i> <i>forficata</i>	123: <i>Agathomyia</i> <i>decorata</i>	124: <i>Amictus</i> <i>decorata</i>	125: <i>Amictus</i> <i>decorata</i>	126: <i>Amictus</i> <i>decorata</i>	127: <i>Amictus</i> <i>decorata</i>	128: <i>Amictus</i> <i>decorata</i>	129: <i>Amictus</i> <i>decorata</i>	130: <i>Amictus</i> <i>decorata</i>	131: <i>Amictus</i> <i>decorata</i>	132: <i>Amictus</i> <i>decorata</i>	133: <i>Amictus</i> <i>decorata</i>	134: <i>Amictus</i> <i>decorata</i>	
135: <i>Ceratobacillus</i> <i>sootae</i>	136: <i>Ceratobacillus</i> <i>sootae</i>	137: <i>Neurotoma</i> <i>longicornis</i>	138: <i>Neurotoma</i> <i>gracilis</i>	139: <i>Neurotoma</i> <i>virginica</i>	140: <i>Trichogramma</i> <i>pretiosa</i>	141: <i>Culicoides</i> <i>varipes</i>	142: <i>Haematopota</i> <i>decorata</i>	143: <i>Haematopota</i> <i>decorata</i>	144: <i>Haematopota</i> <i>decorata</i>	145: <i>Haematopota</i> <i>decorata</i>	146: <i>Haematopota</i> <i>decorata</i>	147: <i>Haematopota</i> <i>decorata</i>	148: <i>Haematopota</i> <i>decorata</i>	149: <i>Haematopota</i> <i>decorata</i>	150: <i>Haematopota</i> <i>decorata</i>	
151: <i>Haematopota</i> <i>decorata</i>	152: <i>Haematopota</i> <i>decorata</i>	153: <i>Haematopota</i> <i>decorata</i>	154: <i>Haematopota</i> <i>decorata</i>	155: <i>Haematopota</i> <i>decorata</i>	156: <i>Haematopota</i> <i>decorata</i>	157: <i>Haematopota</i> <i>decorata</i>	158: <i>Haematopota</i> <i>decorata</i>	159: <i>Haematopota</i> <i>decorata</i>	160: <i>Haematopota</i> <i>decorata</i>	161: <i>Haematopota</i> <i>decorata</i>	162: <i>Haematopota</i> <i>decorata</i>	163: <i>Haematopota</i> <i>decorata</i>	164: <i>Haematopota</i> <i>decorata</i>	165: <i>Haematopota</i> <i>decorata</i>	166: <i>Haematopota</i> <i>decorata</i>	
167: <i>Haematopota</i> <i>decorata</i>	168: <i>Haematopota</i> <i>decorata</i>	169: <i>Haematopota</i> <i>decorata</i>	170: <i>Haematopota</i> <i>decorata</i>	171: <i>Haematopota</i> <i>decorata</i>	172: <i>Haematopota</i> <i>decorata</i>	173: <i>Haematopota</i> <i>decorata</i>	174: <i>Haematopota</i> <i>decorata</i>	175: <i>Haematopota</i> <i>decorata</i>	176: <i>Haematopota</i> <i>decorata</i>	177: <i>Haematopota</i> <i>decorata</i>	178: <i>Haematopota</i> <i>decorata</i>	179: <i>Haematopota</i> <i>decorata</i>	180: <i>Haematopota</i> <i>decorata</i>	181: <i>Haematopota</i> <i>decorata</i>	182: <i>Haematopota</i> <i>decorata</i>	
183: <i>Haematopota</i> <i>decorata</i>	184: <i>Haematopota</i> <i>decorata</i>	185: <i>Haematopota</i> <i>decorata</i>	186: <i>Haematopota</i> <i>decorata</i>	187: <i>Haematopota</i> <i>decorata</i>	188: <i>Haematopota</i> <i>decorata</i>	189: <i>Haematopota</i> <i>decorata</i>	190: <i>Haematopota</i> <i>decorata</i>	191: <i>Haematopota</i> <i>decorata</i>	192: <i>Haematopota</i> <i>decorata</i>	193: <i>Haematopota</i> <i>decorata</i>	194: <i>Haematopota</i> <i>decorata</i>	195: <i>Haematopota</i> <i>decorata</i>	196: <i>Haematopota</i> <i>decorata</i>	197: <i>Haematopota</i> <i>decorata</i>	198: <i>Haematopota</i> <i>decorata</i>	199: <i>Haematopota</i> <i>decorata</i>
200: <i>Haematopota</i> <i>decorata</i>	201: <i>Haematopota</i> <i>decorata</i>	202: <i>Haematopota</i> <i>decorata</i>	203: <i>Haematopota</i> <i>decorata</i>	204: <i>Haematopota</i> <i>decorata</i>	205: <i>Haematopota</i> <i>decorata</i>	206: <i>Haematopota</i> <i>decorata</i>	207: <i>Haematopota</i> <i>decorata</i>	208: <i>Haematopota</i> <i>decorata</i>	209: <i>Haematopota</i> <i>decorata</i>	210: <i>Haematopota</i> <i>decorata</i>	211: <i>Haematopota</i> <i>decorata</i>	212: <i>Haematopota</i> <i>decorata</i>	213: <i>Haematopota</i> <i>decorata</i>	214: <i>Haematopota</i> <i>decorata</i>	2	

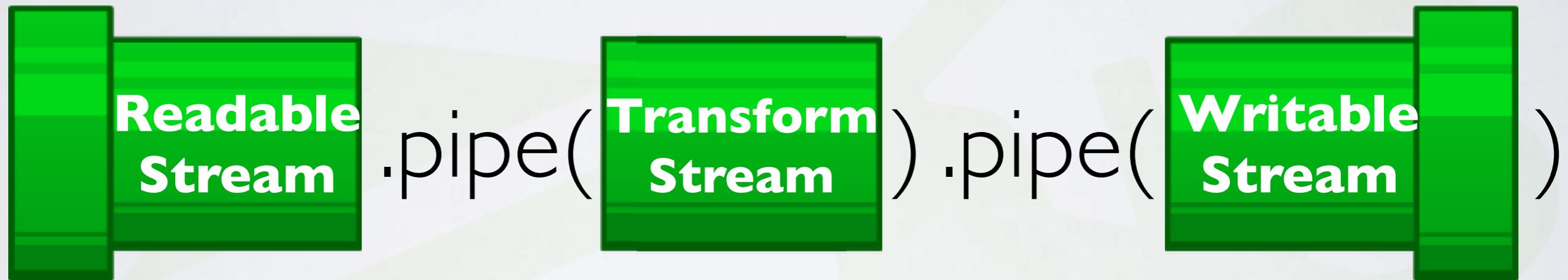
ORIGIN

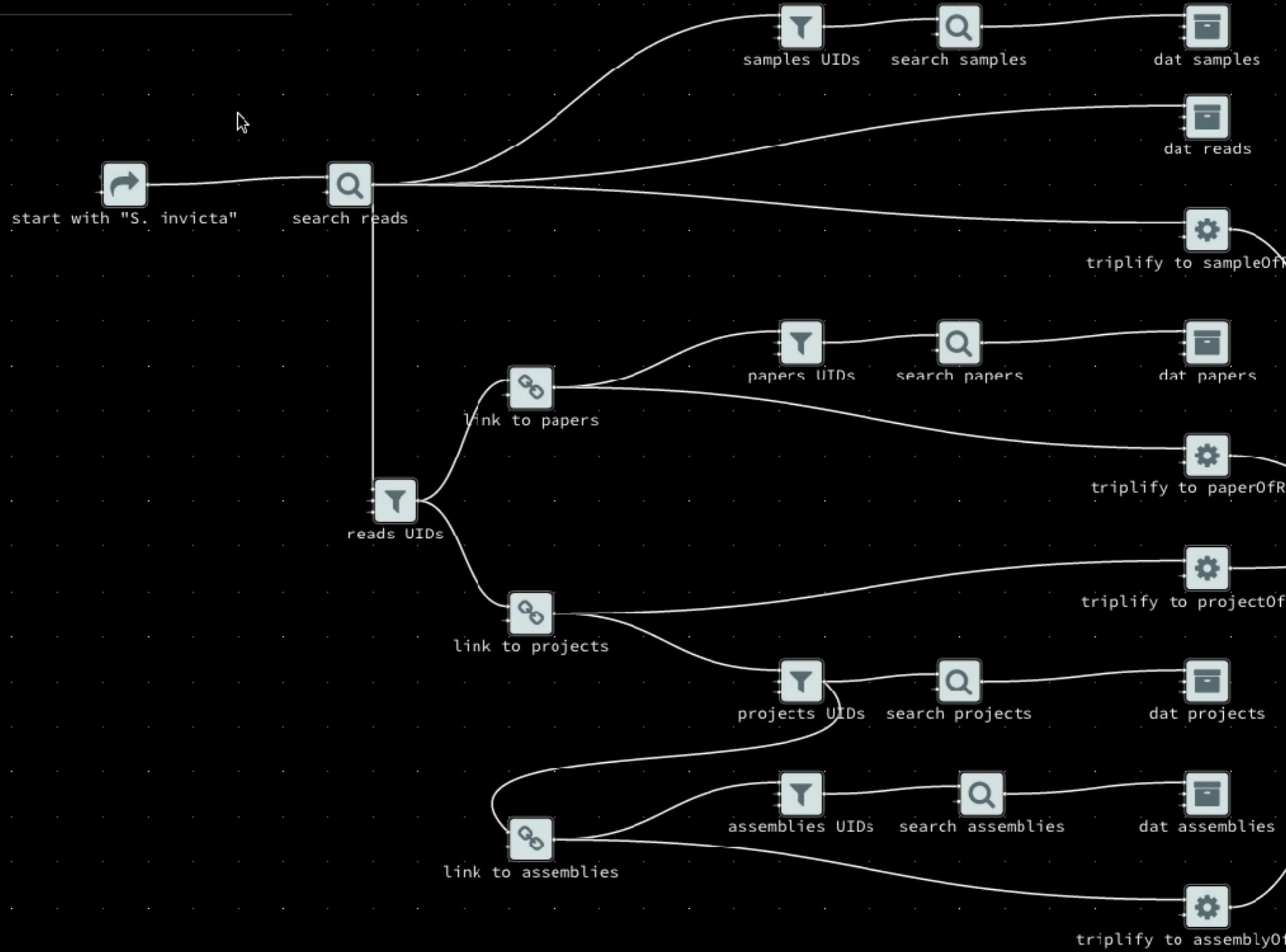
During my PhD at



- 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





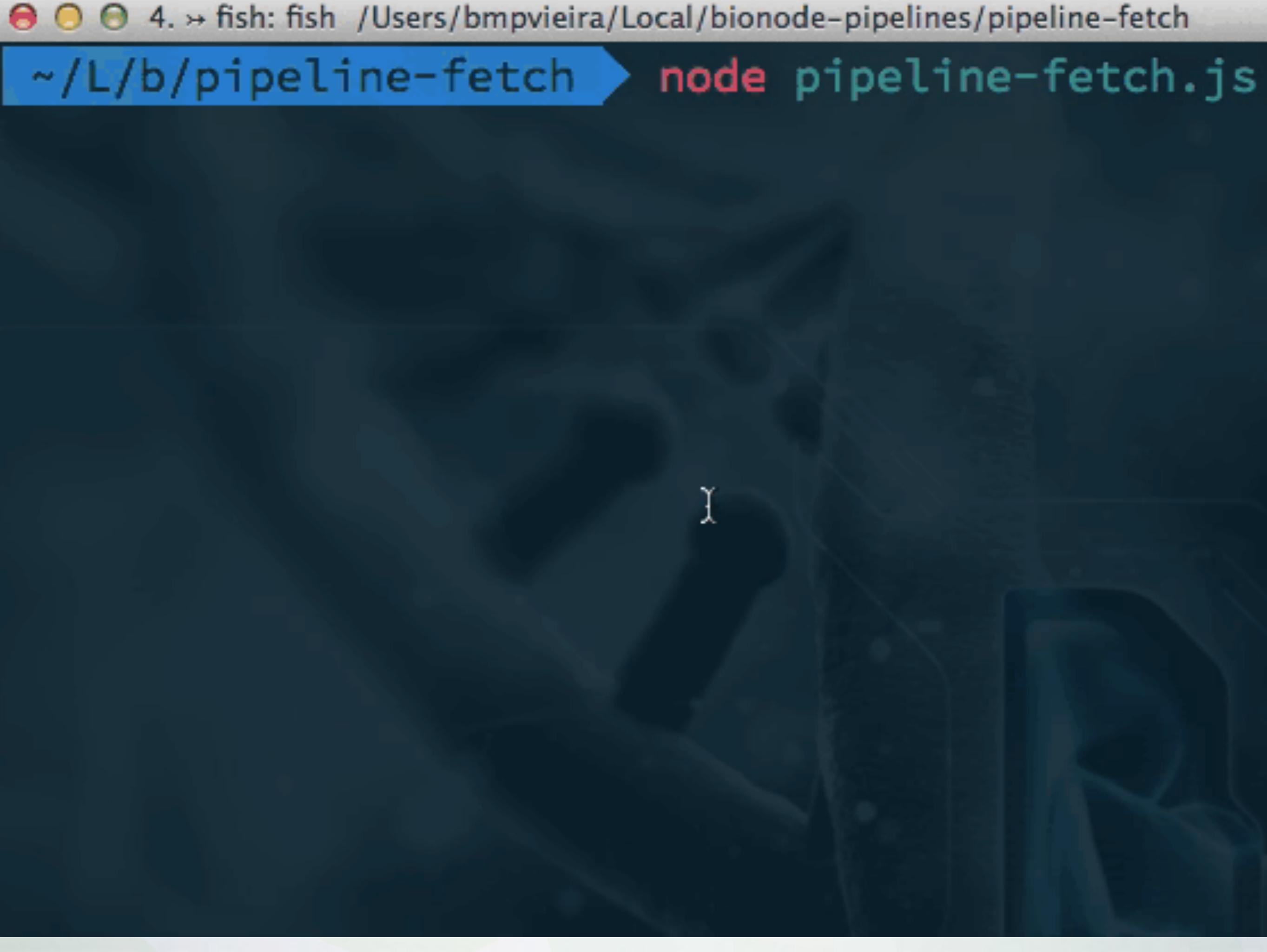
Data analysis pipeline



```
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)
```

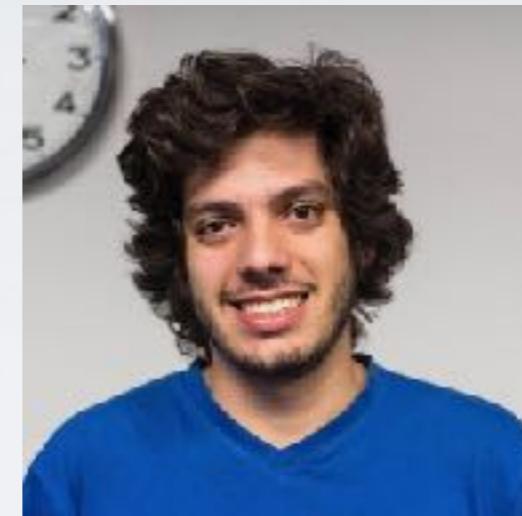


bionode-waterwheel

A Streaming Workflow Engine for bioinformatics and other big data explorations



Google Summer of Code 2016

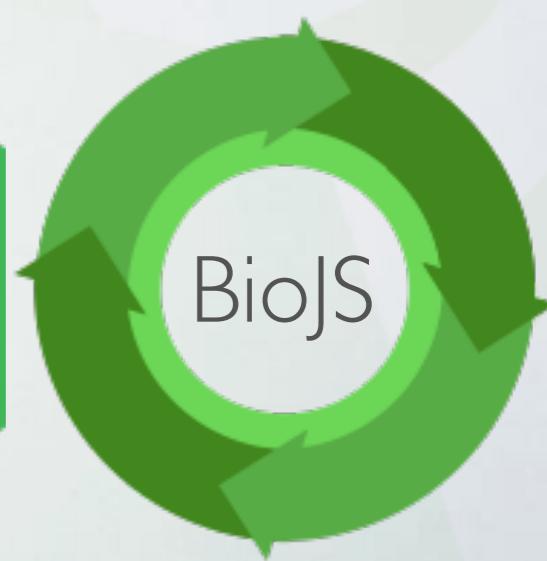


O|B|F

```
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



ACKNOWLEDGMENTS



Research group



Community



Friends

