JBrowse

An AJAX Genome Browser

Mitchell Skinner
Ian Holmes Lab
UC Berkeley

“Genome Browser”

- Service
  - Software
  - Data
  - Servers
  - **People**
    - Software Development
    - Data management
    - Support
    - Outreach/Training

"Genome Browser"

- Service
  - Software $\longleftarrow$ JBrowse
  - Data
  - Servers
  - **People**
    - Data management
    - Support
    - Outreach/Training

Someone Else Does the Data Management

I have seen `kent/src/hg/makeDb/doc/*`

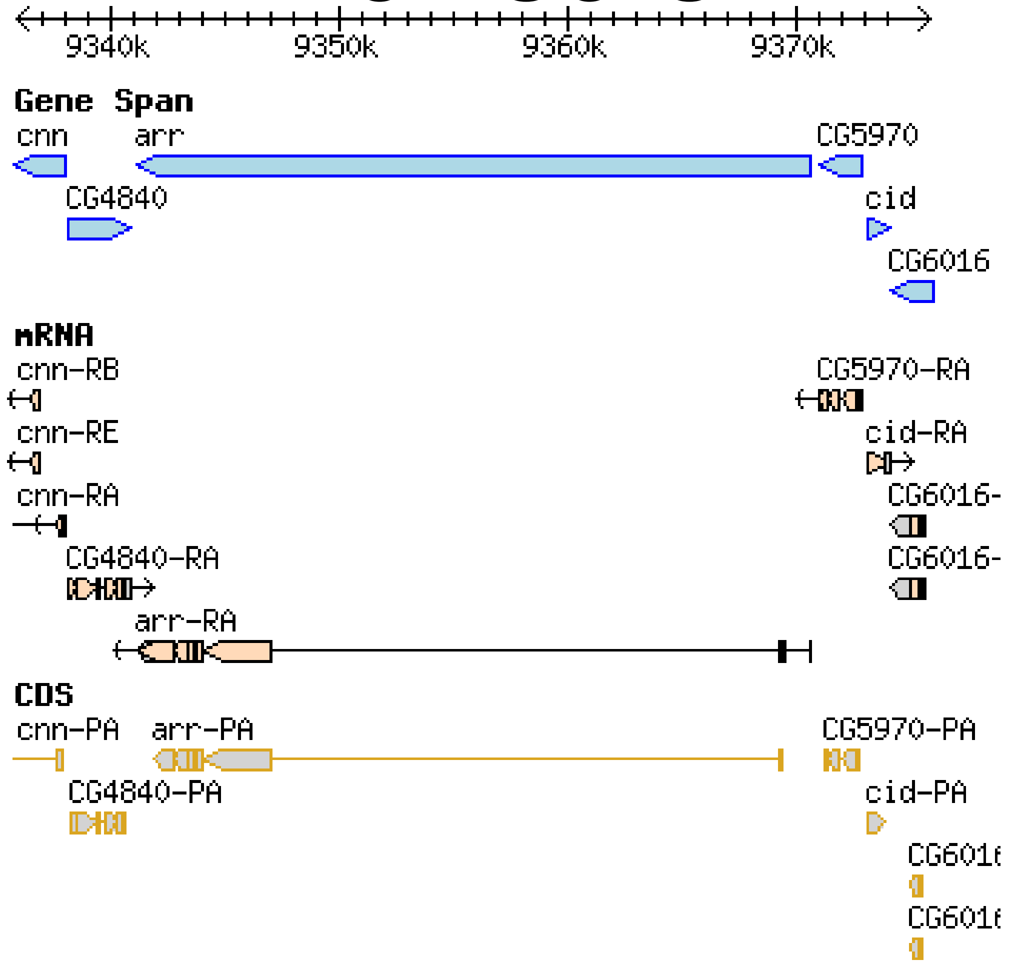
This talk

- Story of JBrowse
- Overview
- Demo

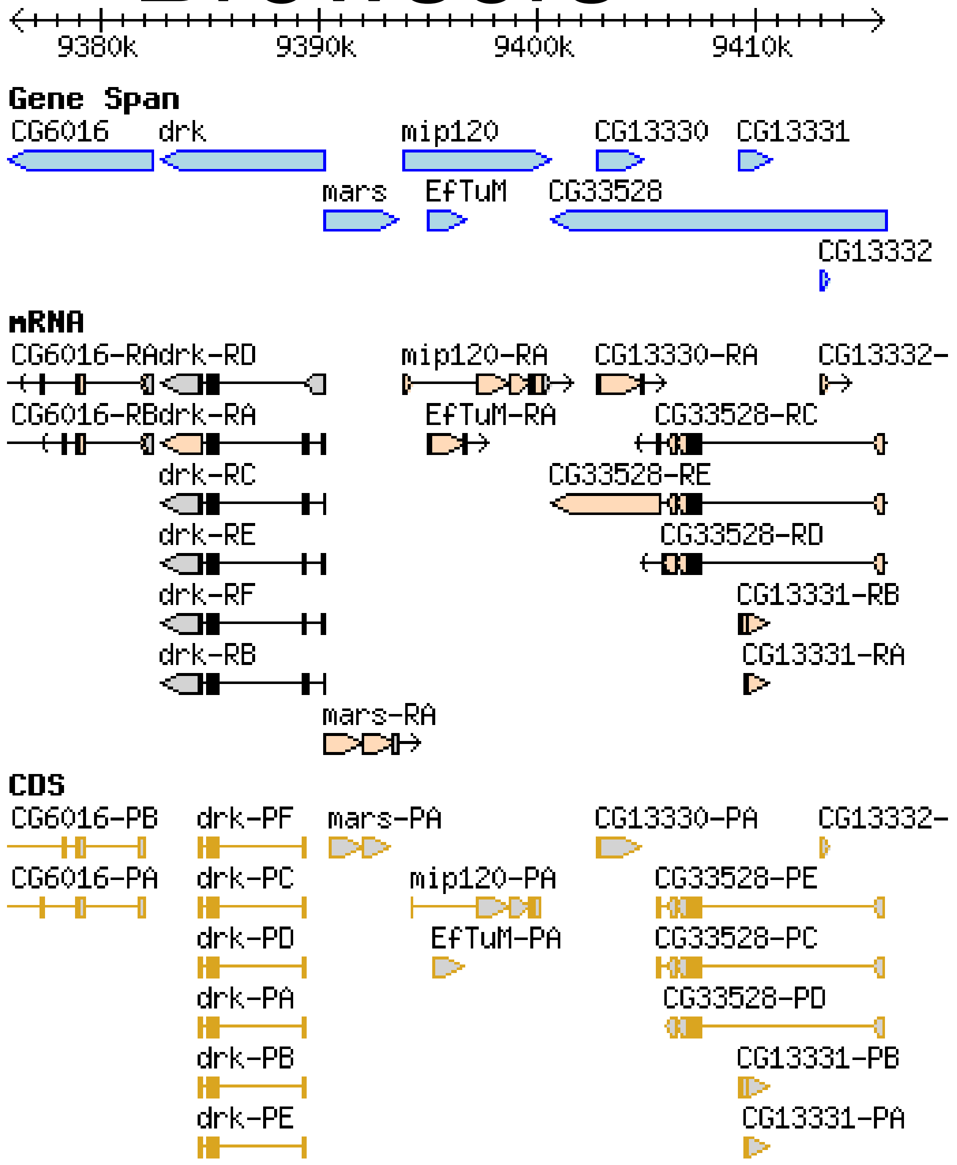
- - - - - - - - - - - - - - - - - -

- Implementation

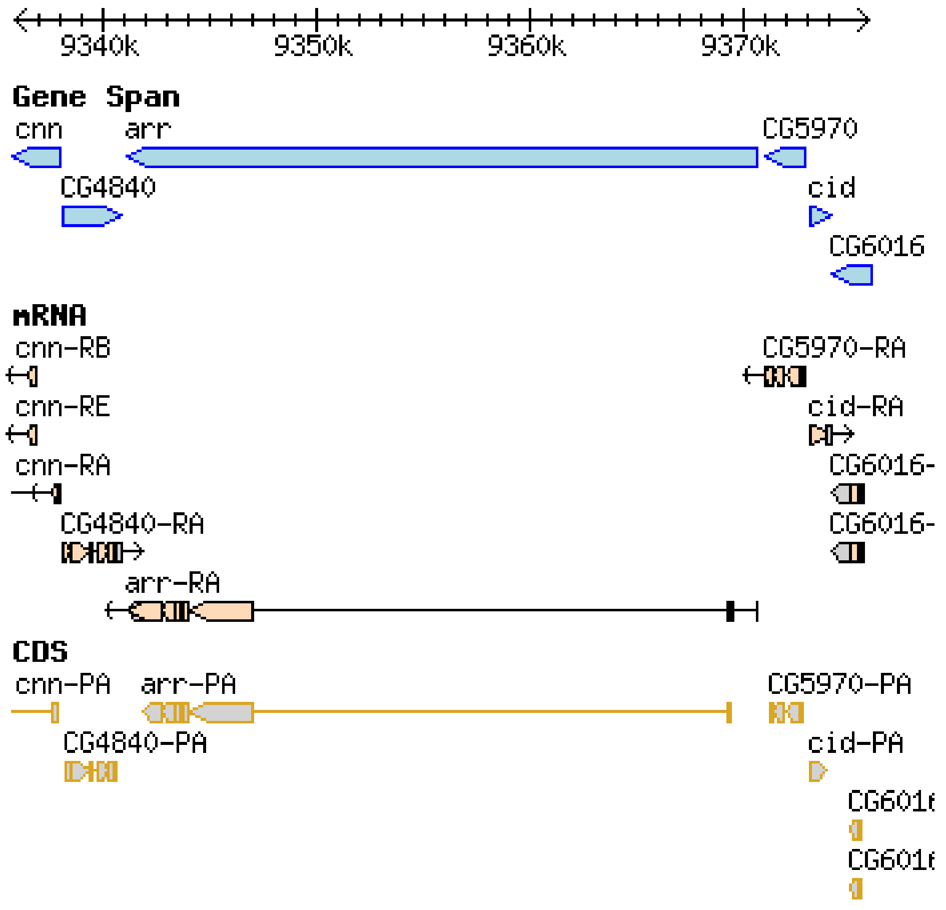
Traditional Web-based Genome Browsers



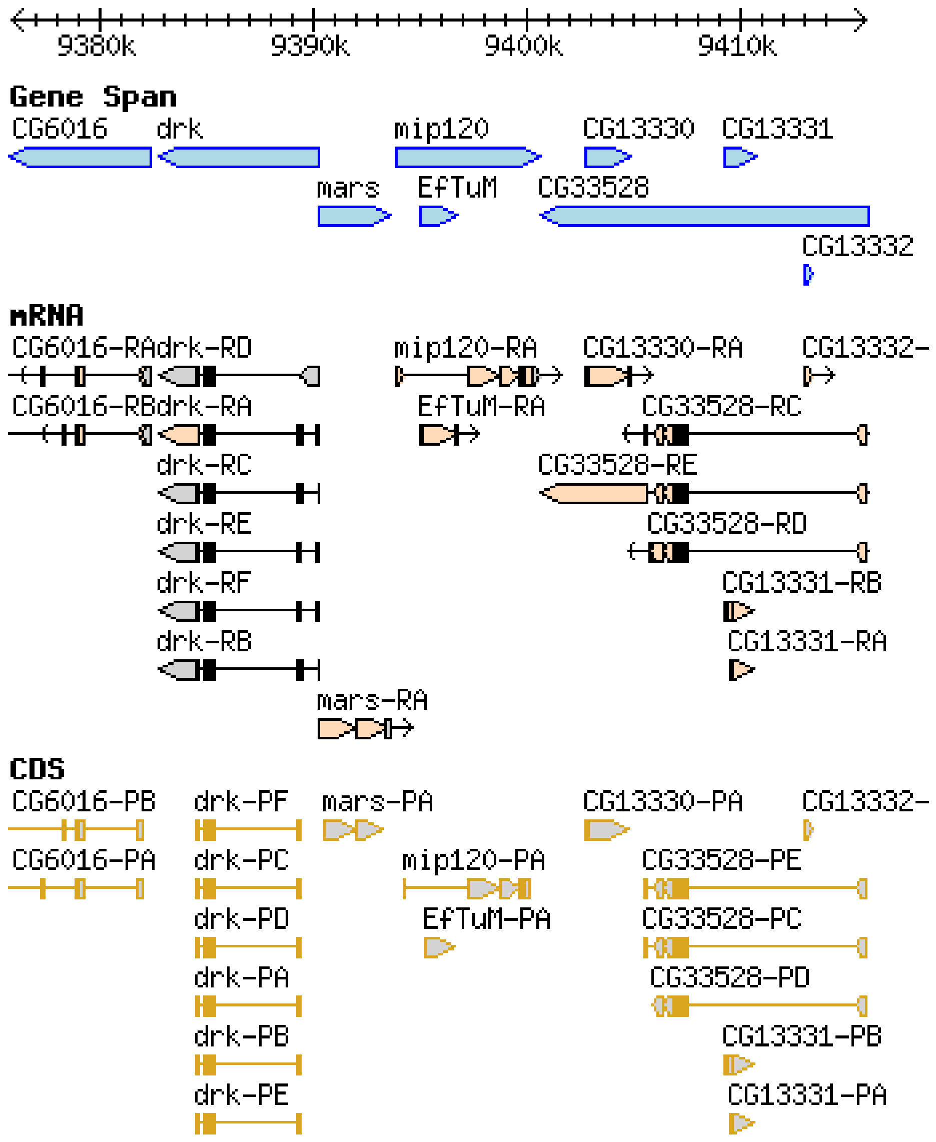
Traditional Web-based Genome Browsers



Tiling?

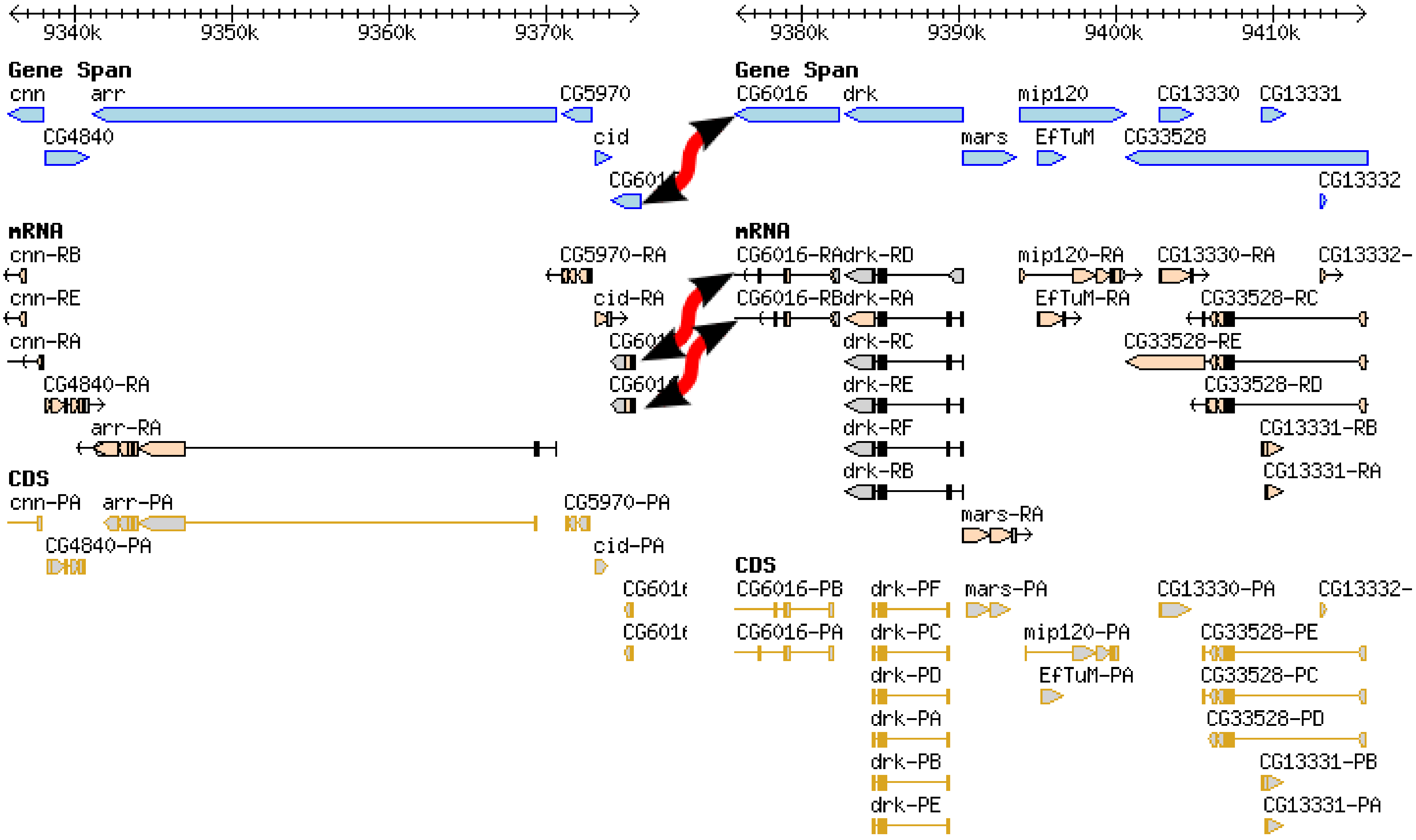


9340k
9350k
9360k
9370k
Gene Span
cnn
arr
CG5970
CG4840
cid
CG6016
mRNA
cnn-RB
CG5970-RA
cnn-RE
cid-RA
cnn-RA
CG6016-
CG4840-RA
CG6016-
arr-RA
CDS
cnn-PA
arr-PA
CG5970-PA
CG4840-PA
cid-PA
CG601
CG601



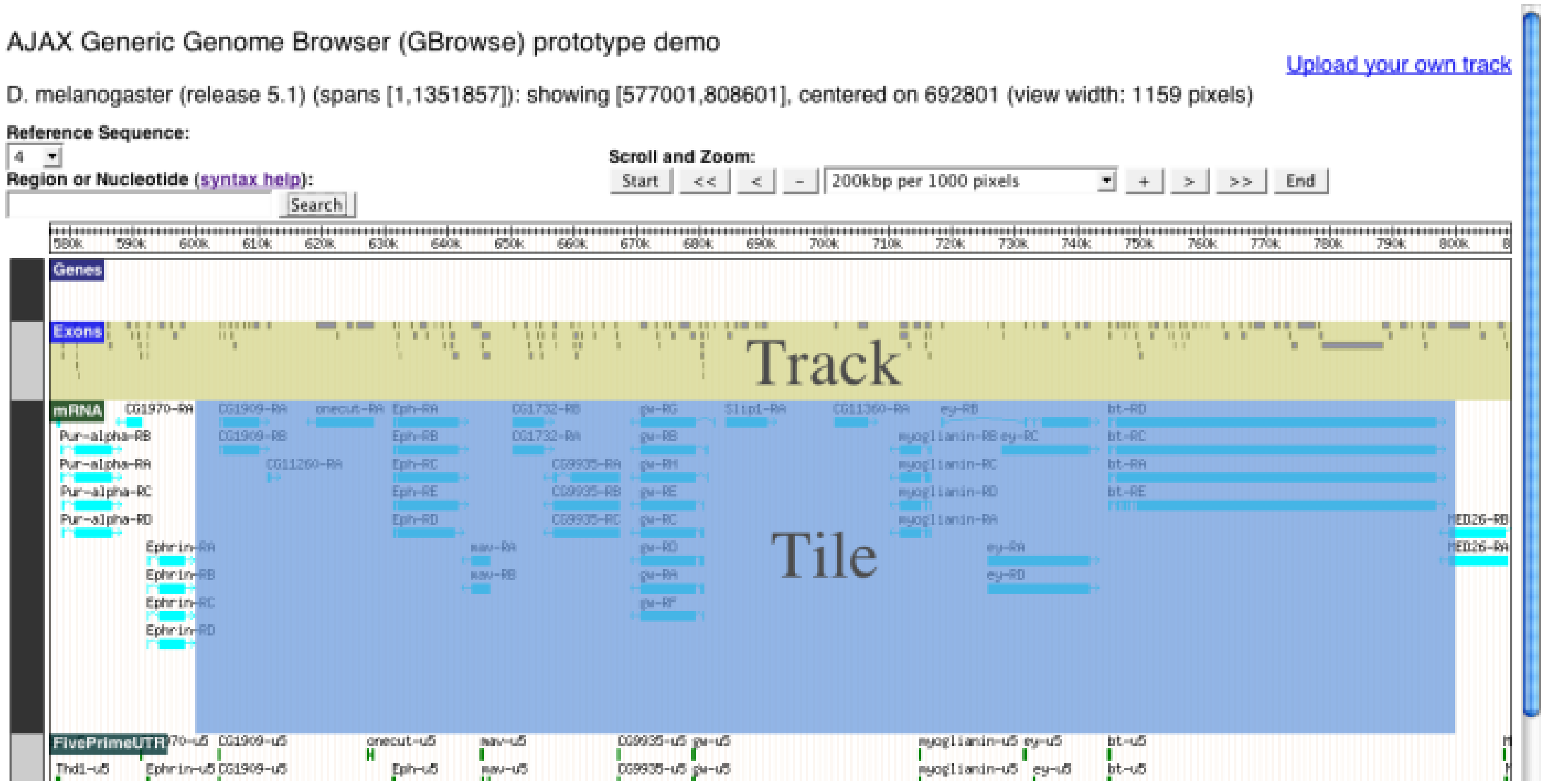
9380k
9390k
9400k
9410k
Gene Span
CG6016
drk
mip120
CG13330
CG13331
mars
EfTuM
CG33528
CG13332
mRNA
CG6016-RA
drk-RD
mip120-RA
CG13330-RA
CG13332-
CG6016-RB
drk-RA
EfTuM-RA
CG33528-RC
drk-RC
CG33528-RE
drk-RE
CG33528-RD
drk-RF
CG13331-RB
drk-RB
CG13331-RA
mars-RA
CDS
CG6016-PB
drk-PF
mars-PA
CG13330-PA
CG13332-
CG6016-PA
drk-PC
mip120-PA
CG33528-PE
drk-PD
EfTuM-PA
CG33528-PC
drk-PA
CG33528-PD
drk-PB
CG13331-PB
drk-PE
CG13331-PA

Tiling?



9340k
9350k
9360k
9370k
9380k
9390k
9400k
9410k
Gene Span
cnn
arr
CG5970
CG4840
cid
CG6016
drk
mip120
CG13330
CG13331
mars
EfTuM
CG33528
CG13332
mRNA
cnn-RB
cnn-RE
cnn-RA
CG4840-RA
arr-RA
CG5970-RA
cid-RA
CG6016-RA
CG6016-RB
drk-RD
drk-RA
drk-RC
drk-RE
drk-RF
drk-RB
mars-RA
mip120-RA
EfTuM-RA
CG13330-RA
CG33528-RC
CG33528-RE
CG33528-RD
CG13331-RB
CG13331-RA
CDS
cnn-PA
arr-PA
CG4840-PA
CG5970-PA
cid-PA
CG6016-PB
CG6016-PA
drk-PF
drk-PC
drk-PD
drk-PA
drk-PB
drk-PE
mars-PA
mip120-PA
EfTuM-PA
CG13330-PA
CG33528-PE
CG33528-PC
CG33528-PD
CG13331-PB
CG13331-PA

First prototype



AJAX Generic Genome Browser (GBrowse) prototype demo

Upload your own track

D. melanogaster (release 5.1) (spans [1,1351857]): showing [577001,808601], centered on 692801 (view width: 1159 pixels)

Reference Sequence: 4

Region or Nucleotide (syntax help):

Scroll and Zoom: 200kbp per 1000 pixels

Rendering Tiles - expensive

- D. melanogaster genome: ~120 million bases
- max zoom: 10 pixels/base
  - => 1.2 billion pixels wide, (US in google maps is only 35 million pixels wide)
  - 10s to 100s of pixels high,
- For the entire Drosophila genome, 10 tracks, all zoom levels
  - Space for tiles: 15gb
  - Rendering time: 15hours

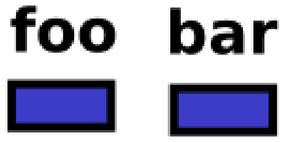
Rendering tiles - optimizations

- Space
  - Hard-linking identical tiles
- Time
  - GD:
    - Memset for filling rectangles
    - Memcpy for creating tiles from larger image
- Space & Time: Render on demand
  - Still requires substantial up-front work

Our server-rendered approach was inherently expensive.

Client-side rendering?

- SVG
  - Not in IE
  - Scalable?

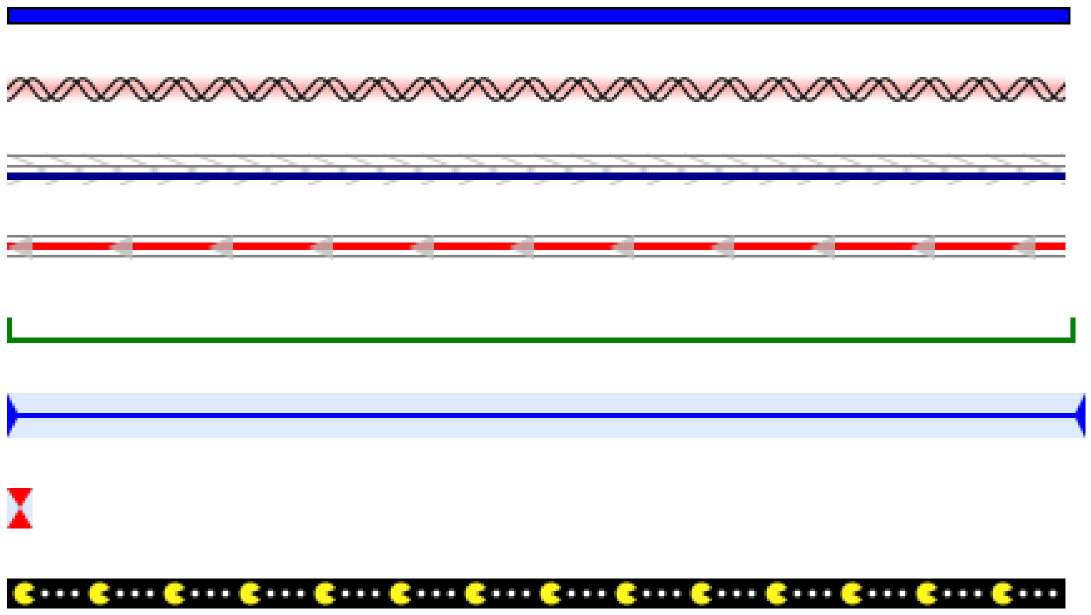


Client-side rendering?

- SVG
  - Not in IE
  - Scalable?

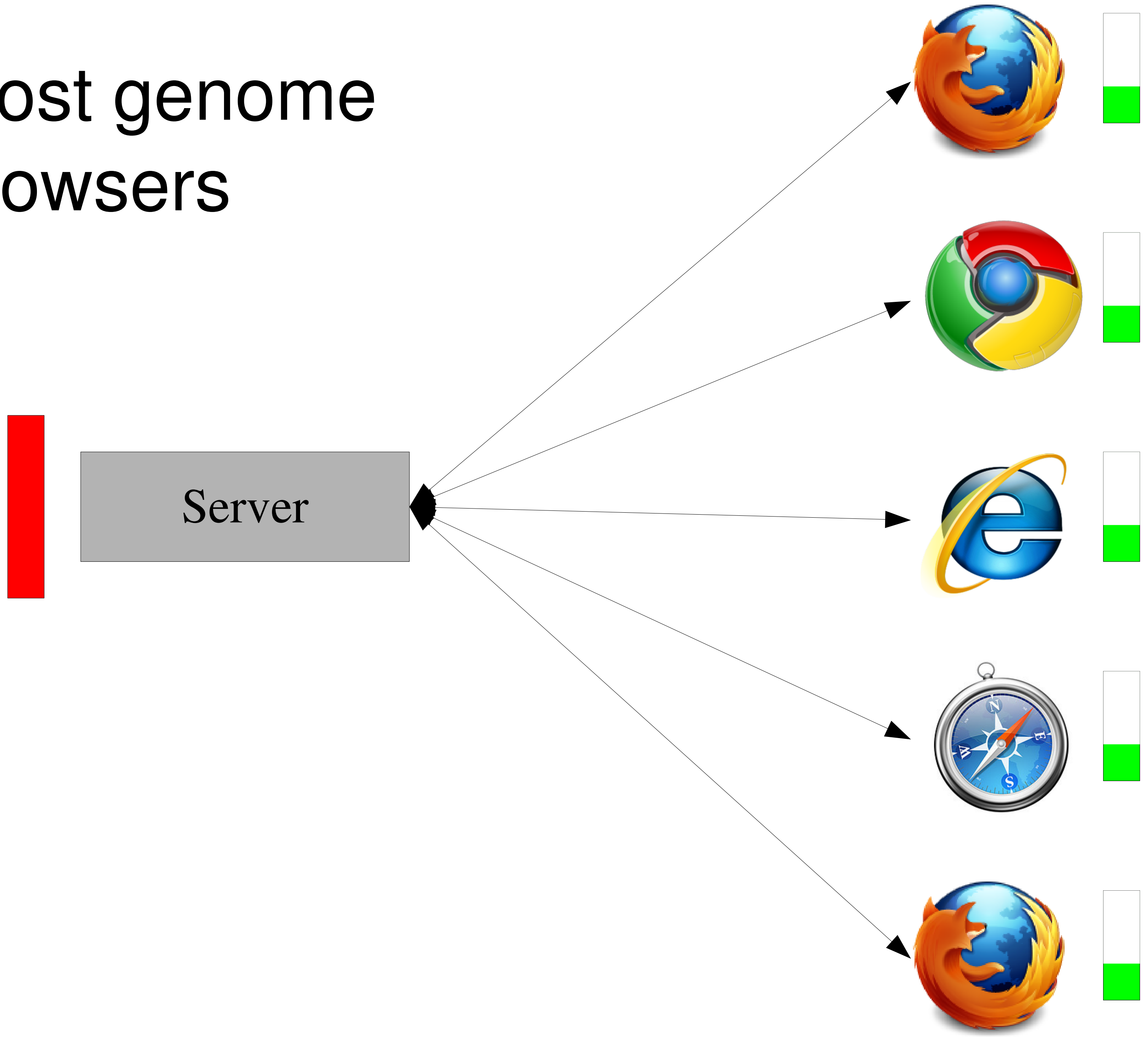
Client-side rendering: HTML

- Rectangles: work for genomic features
- HTML rectangles have fairly rich functionality
- Zooming: position/size in percentage units

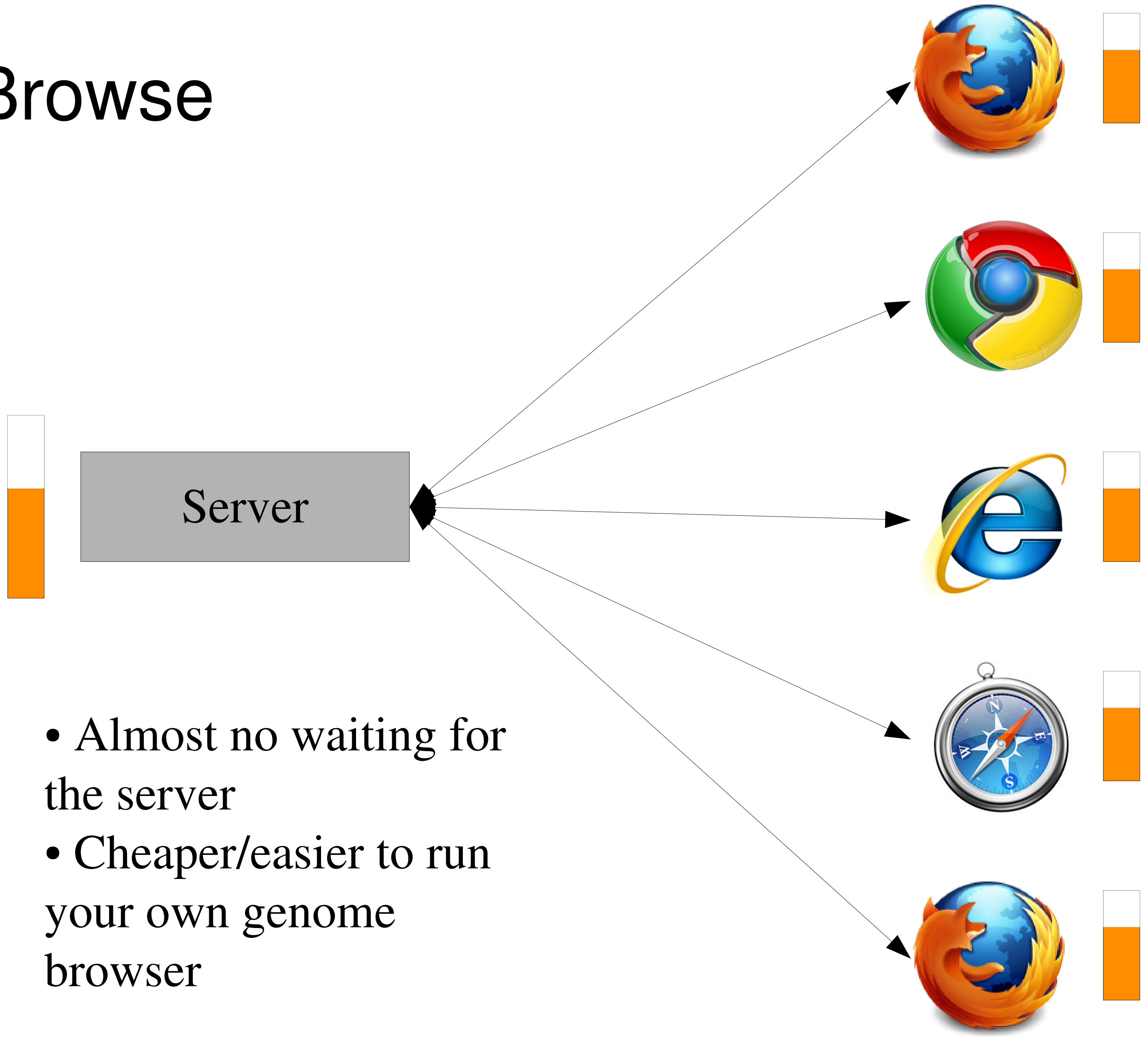


(demo)

Most genome browsers



JBrowse



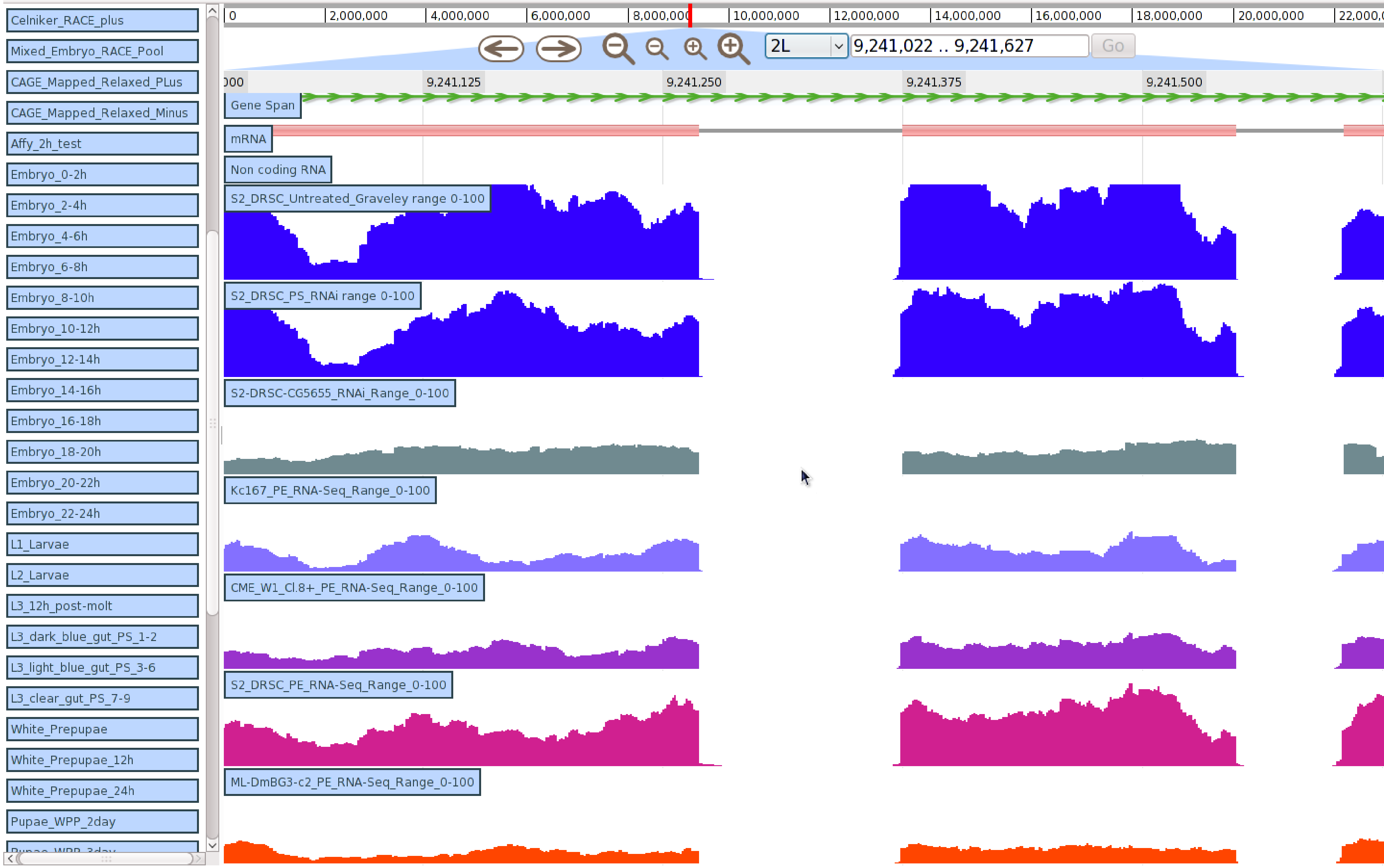
- Almost no waiting for the server
- Cheaper/easier to run your own genome browser

Client-side advantages
(relative to our earlier prototype)

- Much less storage/CPU usage on the server
- Useful amounts of data can be cached on the client
  - Maybe even enable off-line usage
- Client can do a lot more
  - Highlight features
  - Show subsets of features
  - Edit features? e.g., gene model curation
- Can combine image-based, pre-rendered tracks side-by-side with client-rendered tracks

JBrowse

- Works (and tested) in IE 6+, Firefox 2+, Safari 3+, Chrome
- Data sources:
  - Feature flatfiles: GFF2/3, BED
  - Quantitative data: WIG
  - Next-gen: BAM
  - BioPerl Bio::DB databases
    - Bio::DB::GFF, Bio::DB::SeqFeature::Store, chado, DAS/1


Celniker_RACE_plus
Mixed_Embryo_RACE_Pool
CAGE_Mapped_Relaxed_PLus
CAGE_Mapped_Relaxed_Minus
Affy_2h_test
Embryo_0-2h
Embryo_2-4h
Embryo_4-6h
Embryo_6-8h
Embryo_8-10h
Embryo_10-12h
Embryo_12-14h
Embryo_14-16h
Embryo_16-18h
Embryo_18-20h
Embryo_20-22h
Embryo_22-24h
L1_Larvae
L2_Larvae
L3_12h_post-molt
L3_dark_blue_gut_PS_1-2
L3_light_blue_gut_PS_3-6
L3_clear_gut_PS_7-9
White_Prepupae
White_Prepupae_12h
White_Prepupae_24h
Pupae_WPP_2day
0
2,000,000
4,000,000
6,000,000
8,000,000
10,000,000
12,000,000
14,000,000
16,000,000
18,000,000
20,000,000
22,000,0
2L
9,241,022 .. 9,241,627
Go
000
9,241,125
9,241,250
9,241,375
9,241,500
Gene Span
mRNA
Non coding RNA
S2_DRSC_Untreated_Graveley range 0-100
S2_DRSC_PS_RNAi range 0-100
S2-DRSC-CG5655_RNAi_Range_0-100
Kc167_PE_RNA-Seq_Range_0-100
CME_W1_Cl.8+_PE_RNA-Seq_Range_0-100
S2_DRSC_PE_RNA-Seq_Range_0-100
ML-DmBG3-c2_PE_RNA-Seq_Range_0-100

This talk

- Story of JBrowse
- Overview
- Demo

- - - - - - - - - - - - - - - - You are here

- Implementation

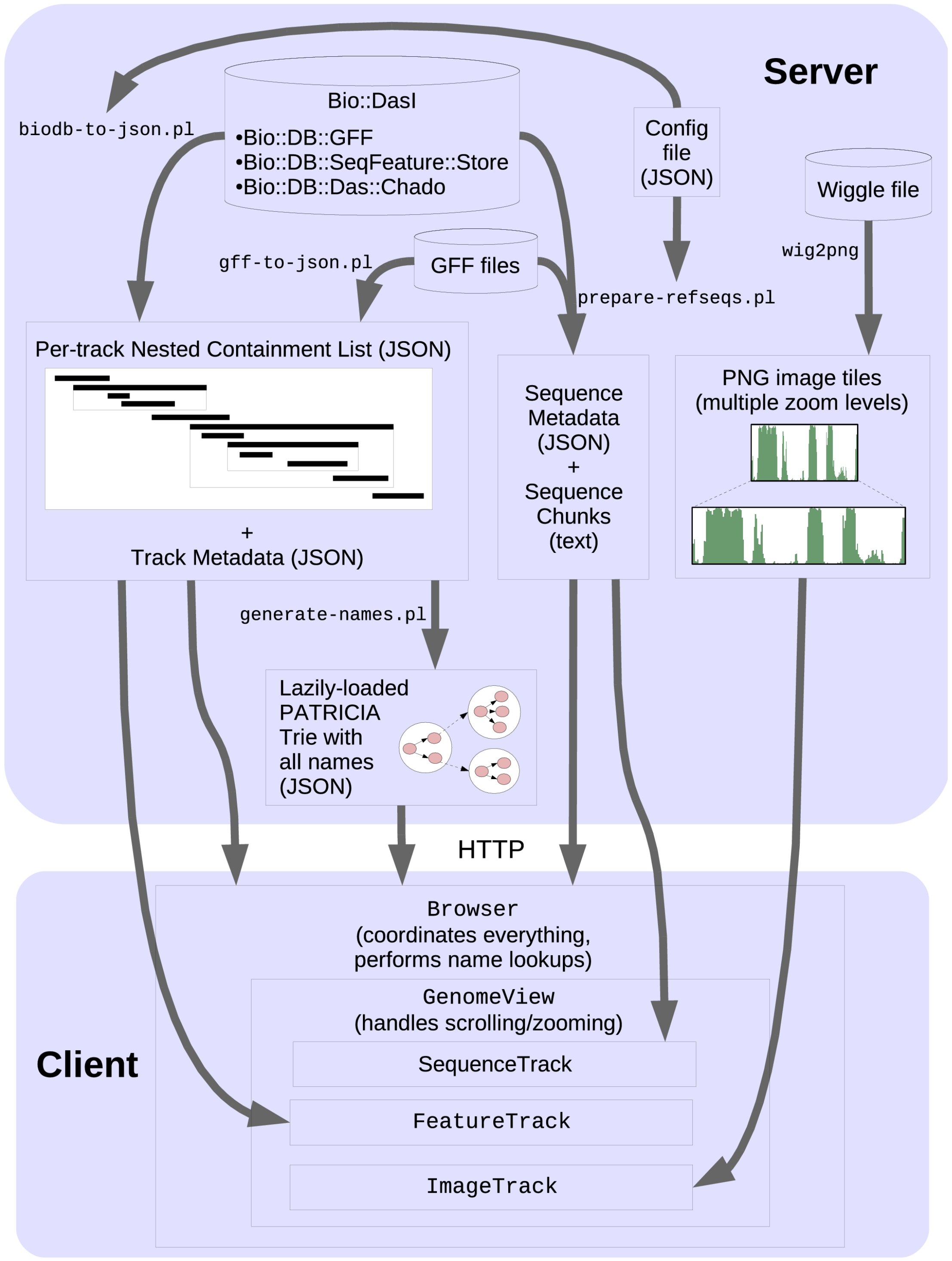
Implementation

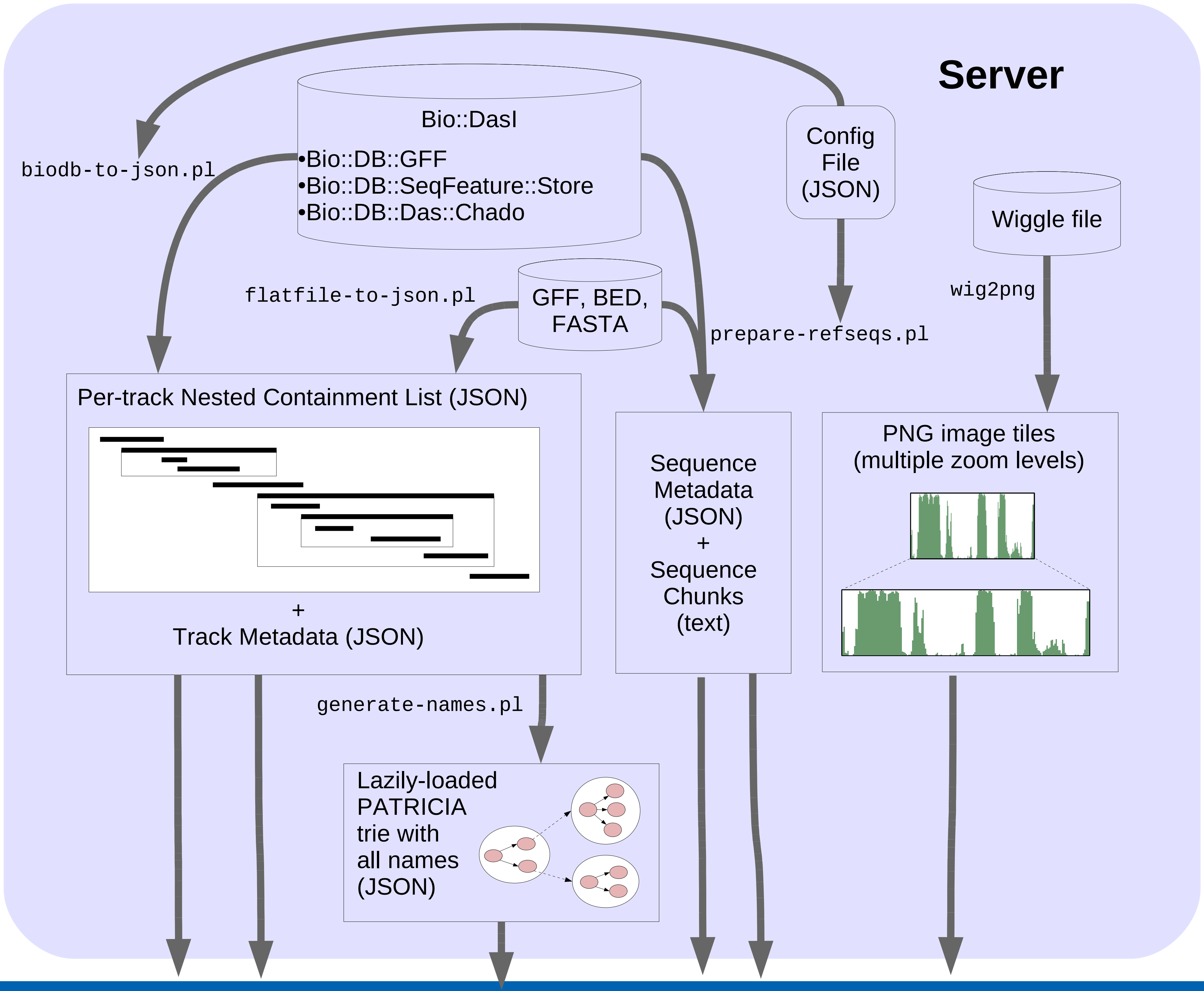
- Interbase!
- Caching
  - Useful amounts of data can be cached on the client
- (Relatively) simple installation
- Low CPU usage at request-time
  - HTTP server only serves static files, no CGI

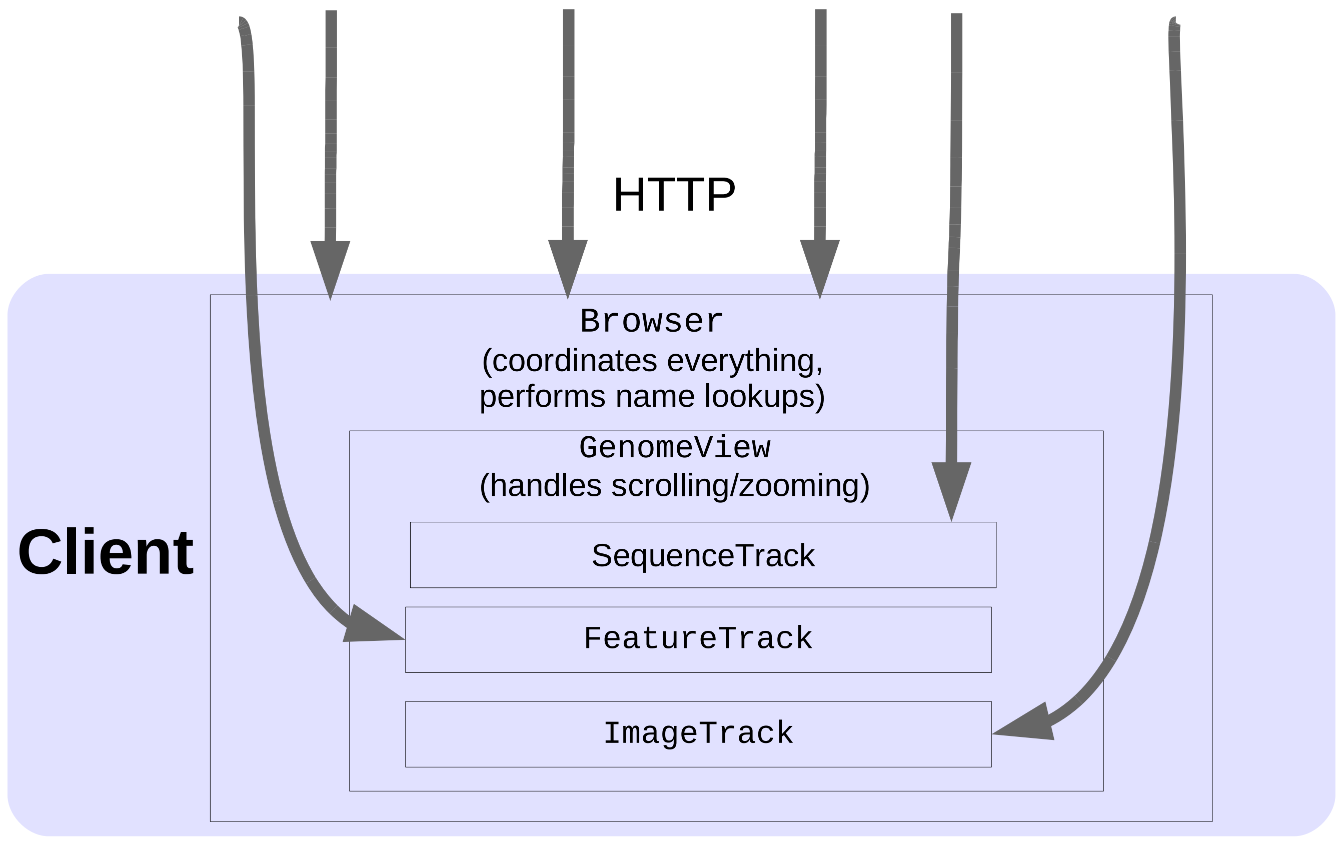
Storage vs. Computation

Do work at request (read) time	Do work at write time
Computation	Computation
Storage	Storage

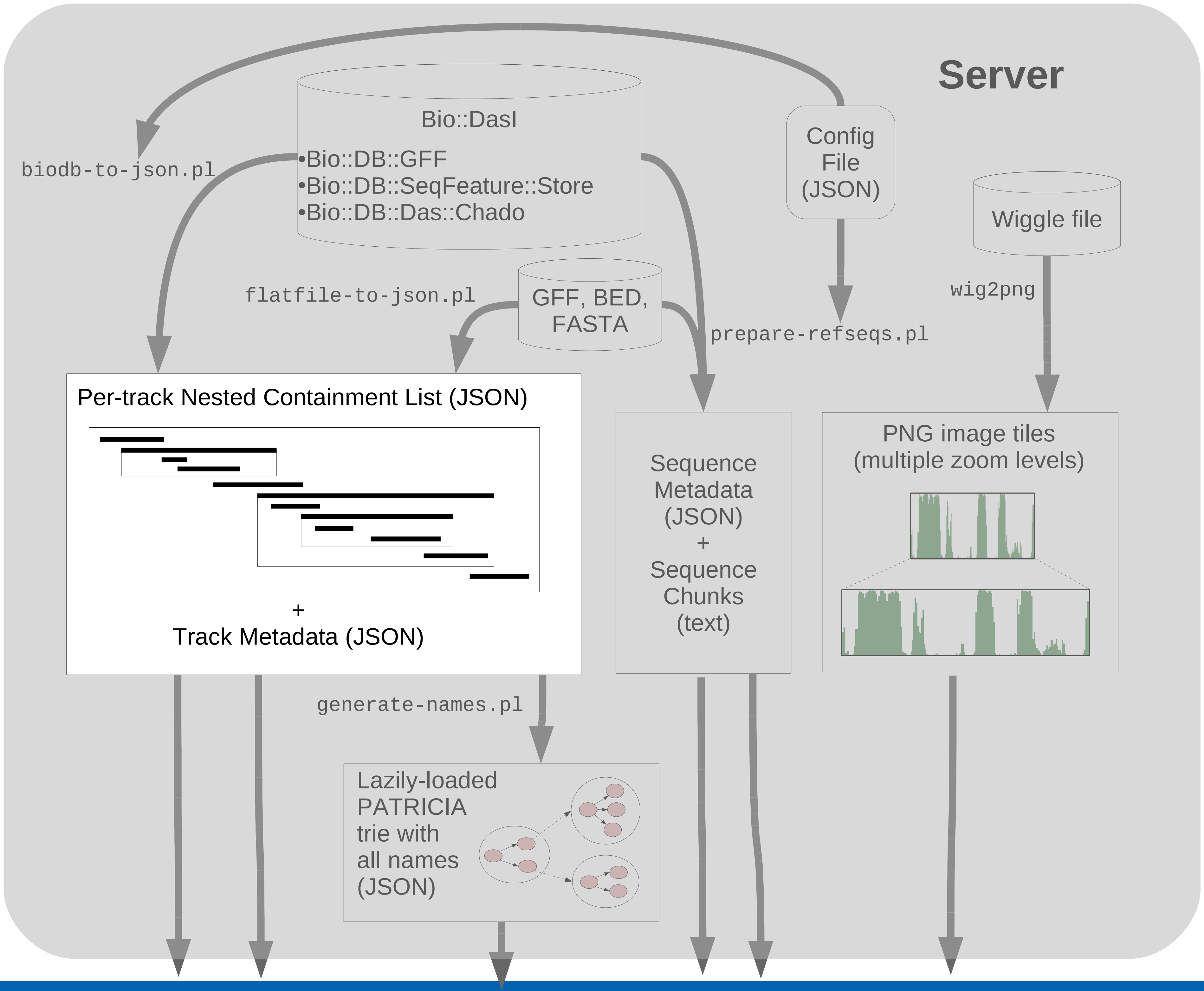
Assumption: read-heavy workload


Server
Bio::DasI
•Bio::DB::GFF
•Bio::DB::SeqFeature::Store
•Bio::DB::Das::Chado
biodb-to-json.pl
Config file (JSON)
Wiggle file
wig2png
GFF files
gff-to-json.pl
prepare-refseqs.pl
Per-track Nested Containment List (JSON)
+
Track Metadata (JSON)
Sequence Metadata (JSON) + Sequence Chunks (text)
PNG image tiles (multiple zoom levels)
generate-names.pl
Lazily-loaded PATRICIA Trie with all names (JSON)
HTTP
Client
Browser (coordinates everything, performs name lookups)
GenomeView (handles scrolling/zooming)
SequenceTrack
FeatureTrack
ImageTrack

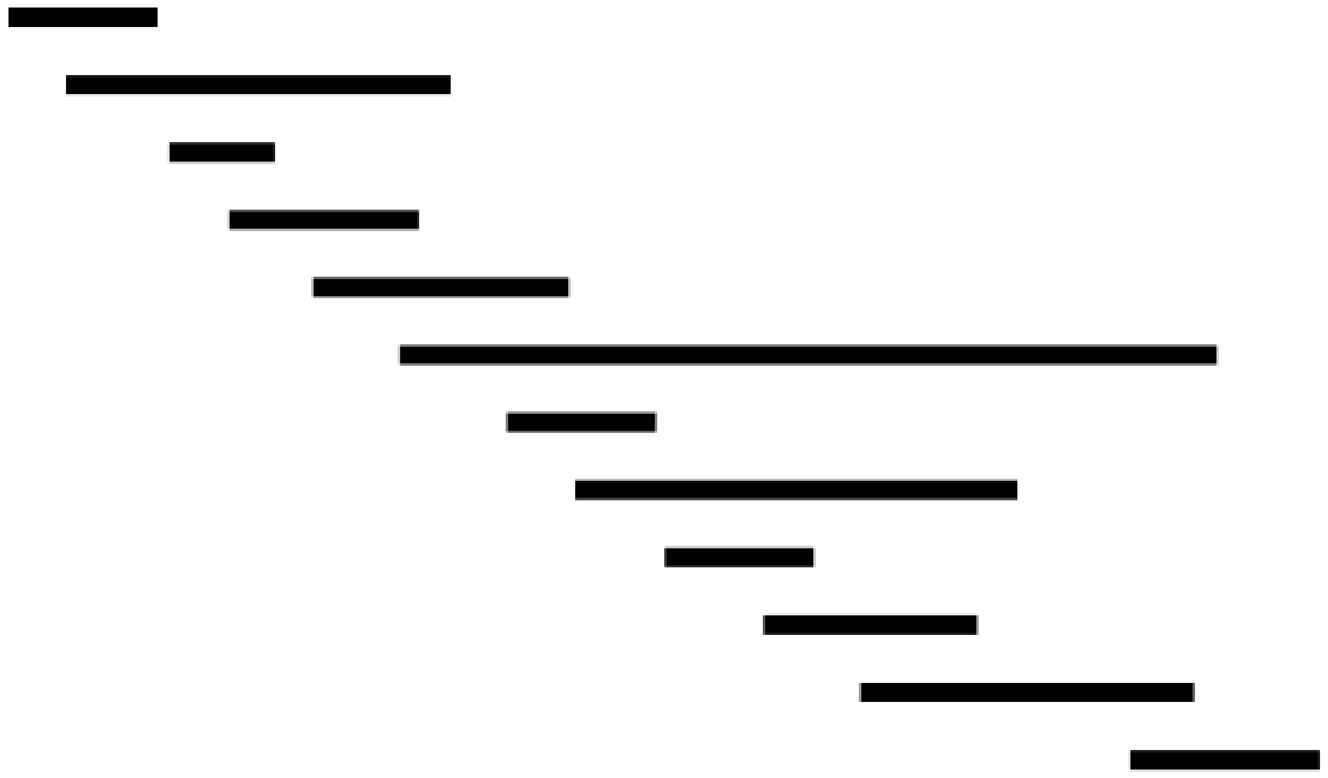

Server
Bio::DasI
•Bio::DB::GFF
•Bio::DB::SeqFeature::Store
•Bio::DB::Das::Chado
biodb-to-json.pl
Config File (JSON)
Wiggle file
flatfile-to-json.pl
GFF, BED, FASTA
prepare-refseqs.pl
wig2png
Per-track Nested Containment List (JSON)
+
Track Metadata (JSON)
Sequence Metadata (JSON)
+
Sequence Chunks (text)
PNG image tiles (multiple zoom levels)
generate-names.pl
Lazily-loaded PATRICIA trie with all names (JSON)



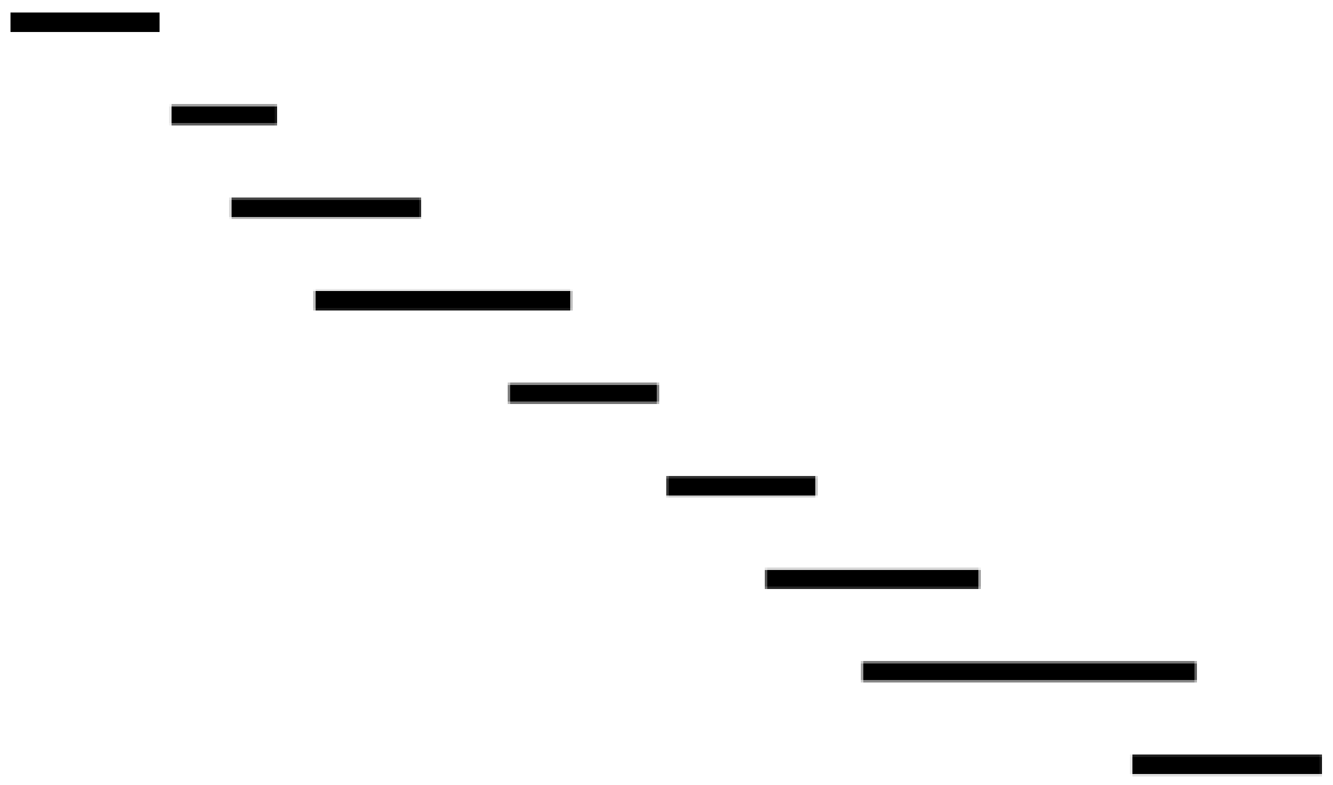
(88% of the code in JBrowse)


Server
Bio::DasI
•Bio::DB::GFF
•Bio::DB::SeqFeature::Store
•Bio::DB::Das::Chado
biodb-to-json.pl
Config File (JSON)
Wiggle file
flatfile-to-json.pl
GFF, BED, FASTA
prepare-refseqs.pl
wig2png
Per-track Nested Containment List (JSON)
+
Track Metadata (JSON)
Sequence Metadata (JSON) + Sequence Chunks (text)
PNG image tiles (multiple zoom levels)
generate-names.pl
Lazily-loaded PATRICIA trie with all names (JSON)

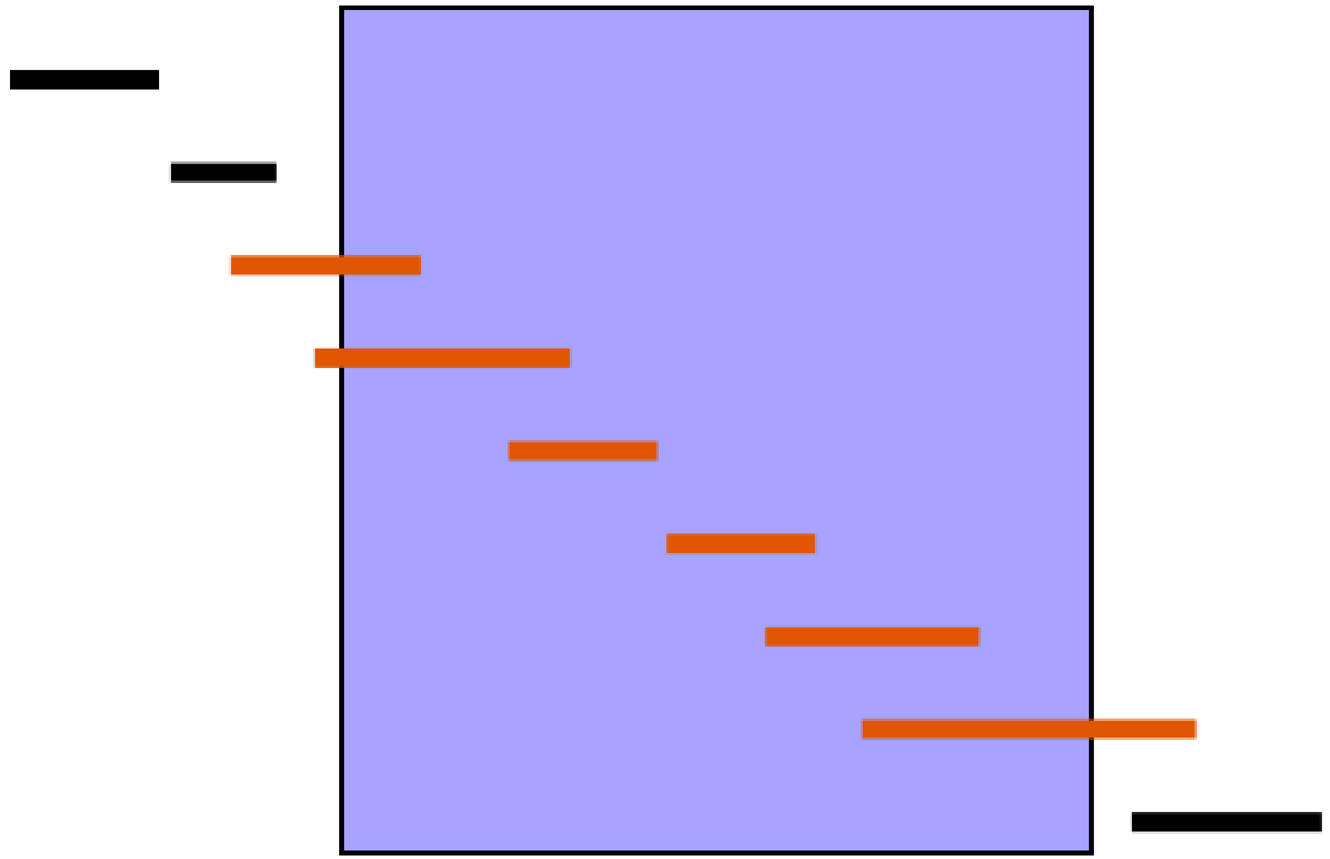
Nested Containment Lists



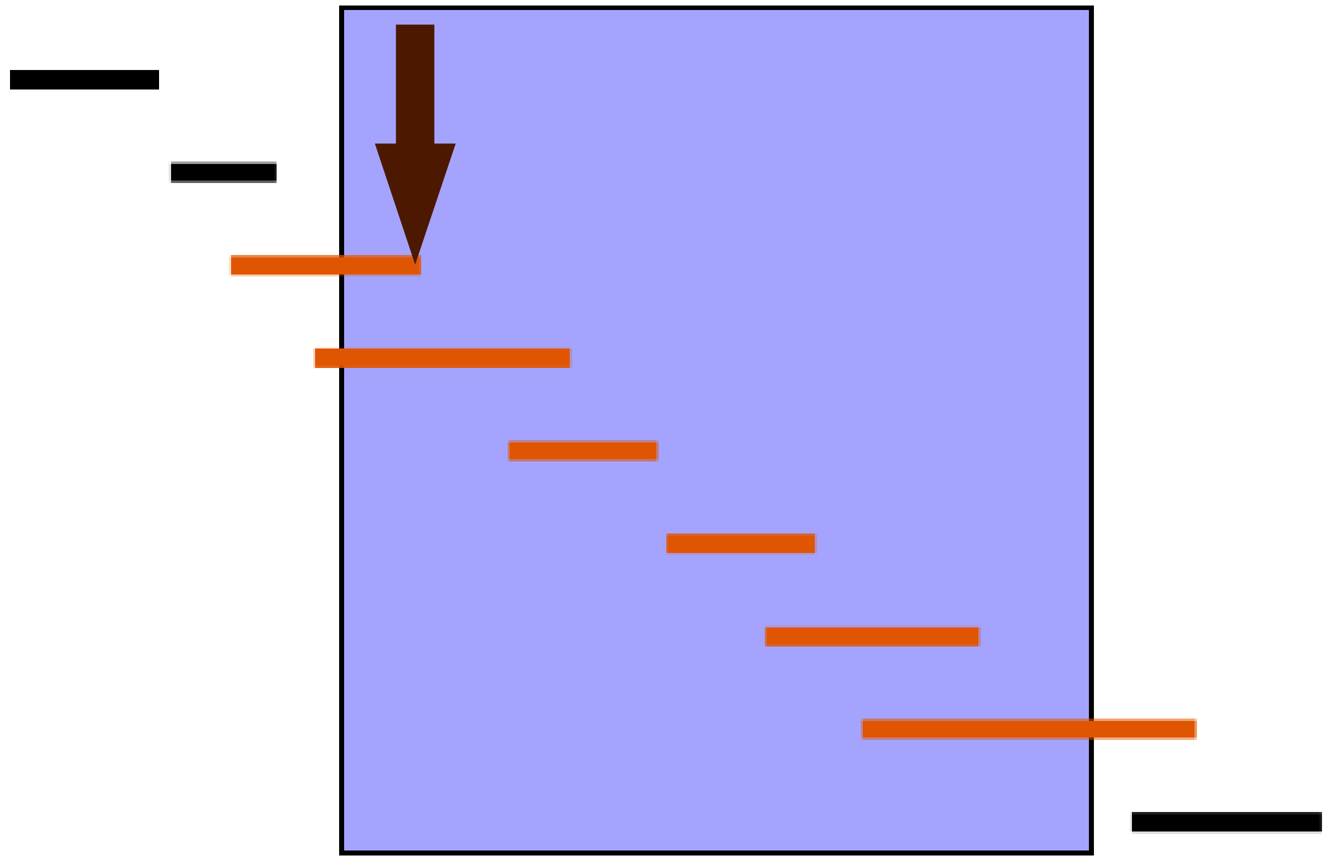
Sorted by both start and end



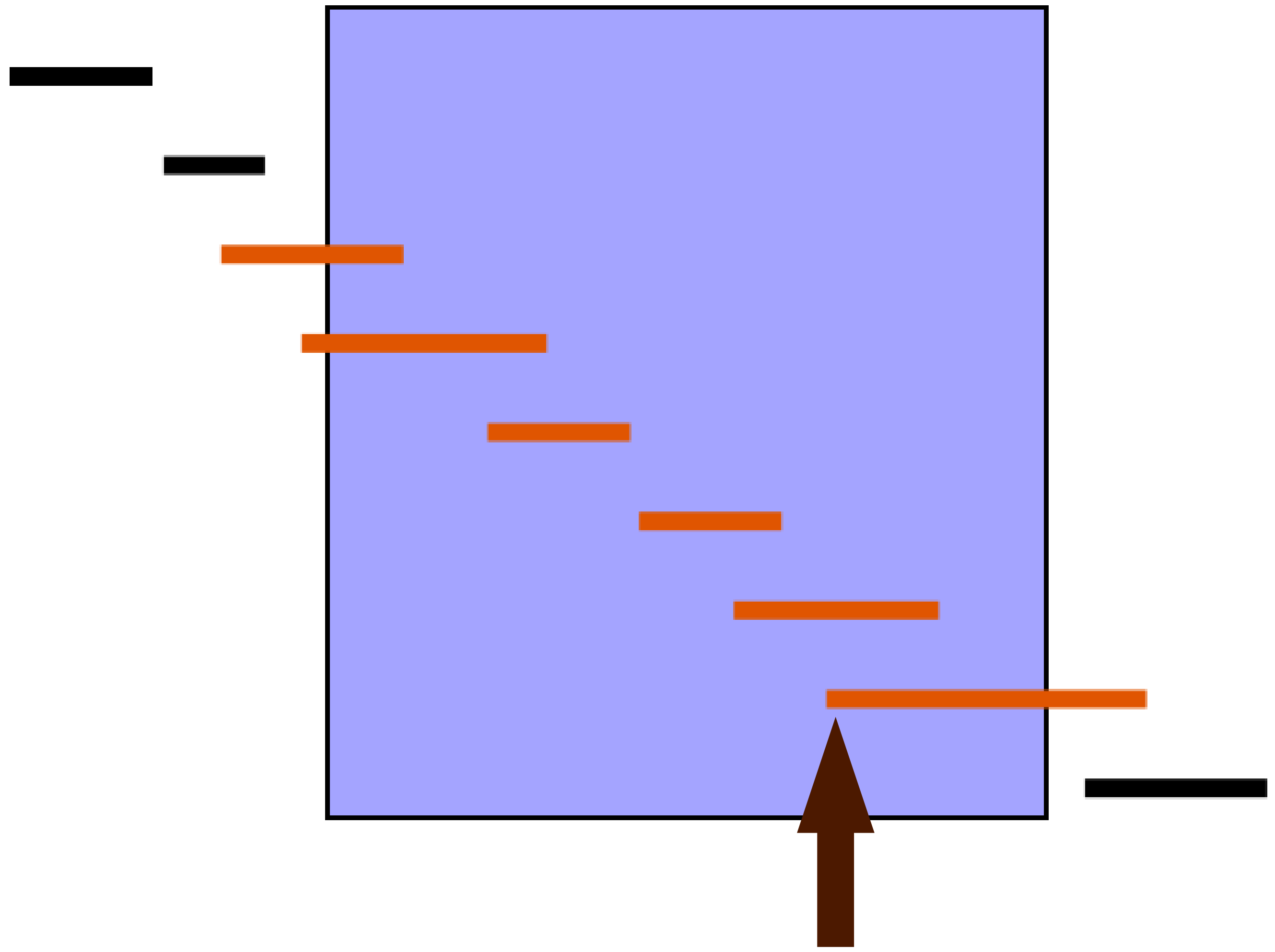
Range query



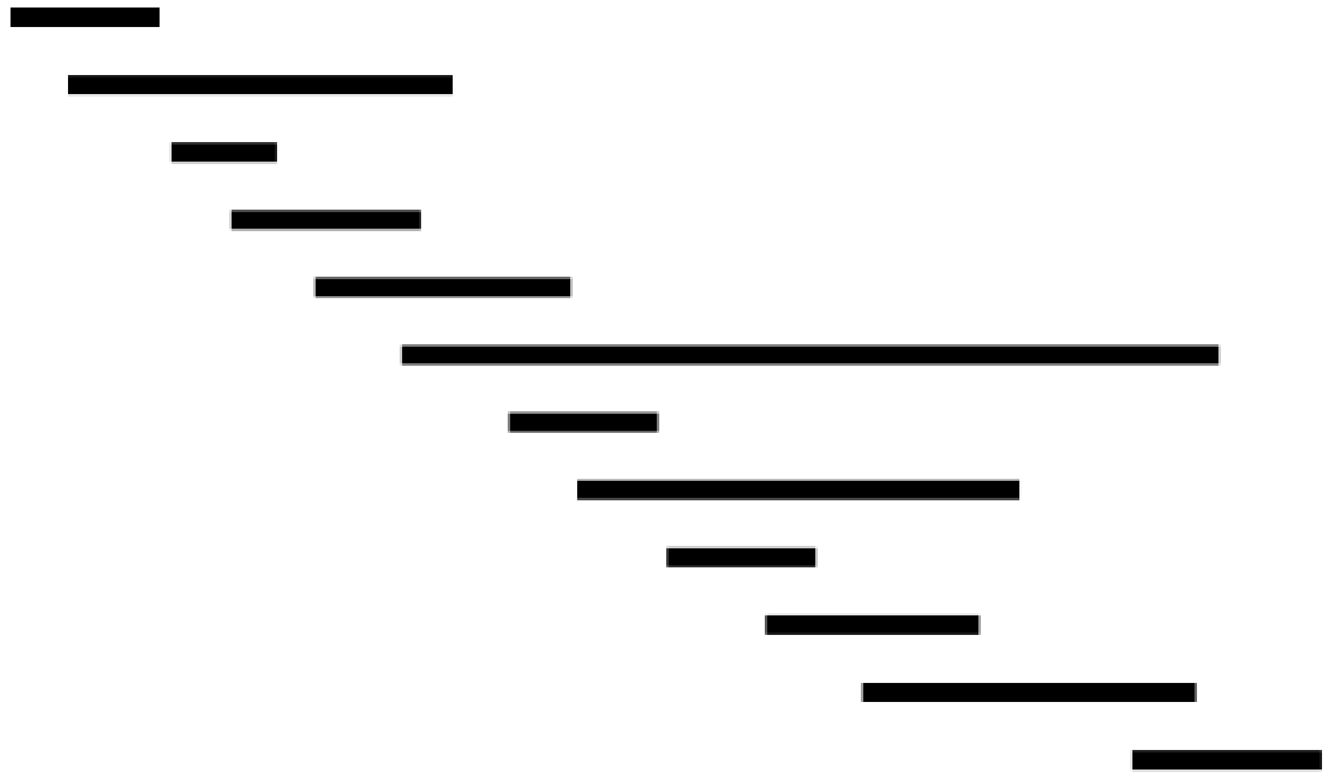
Range query



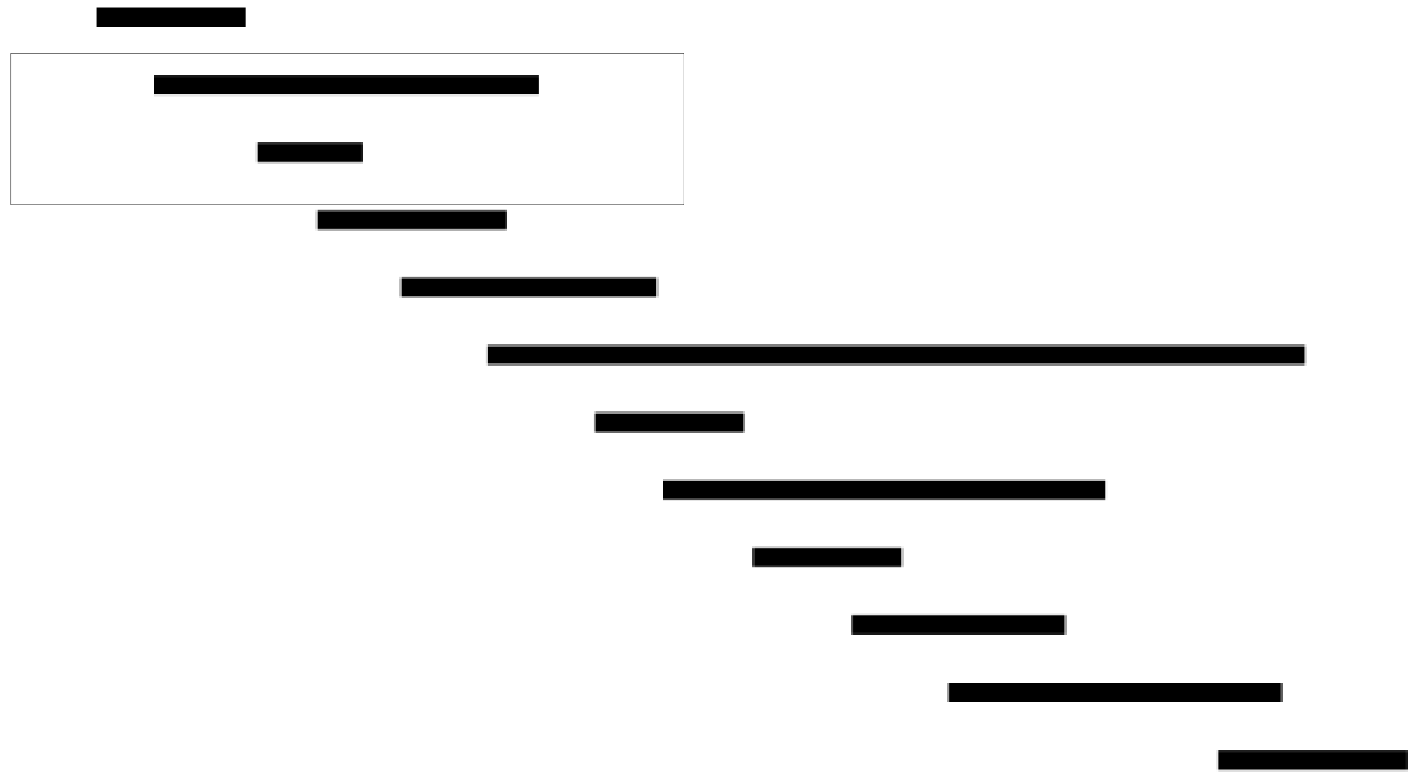
Range query



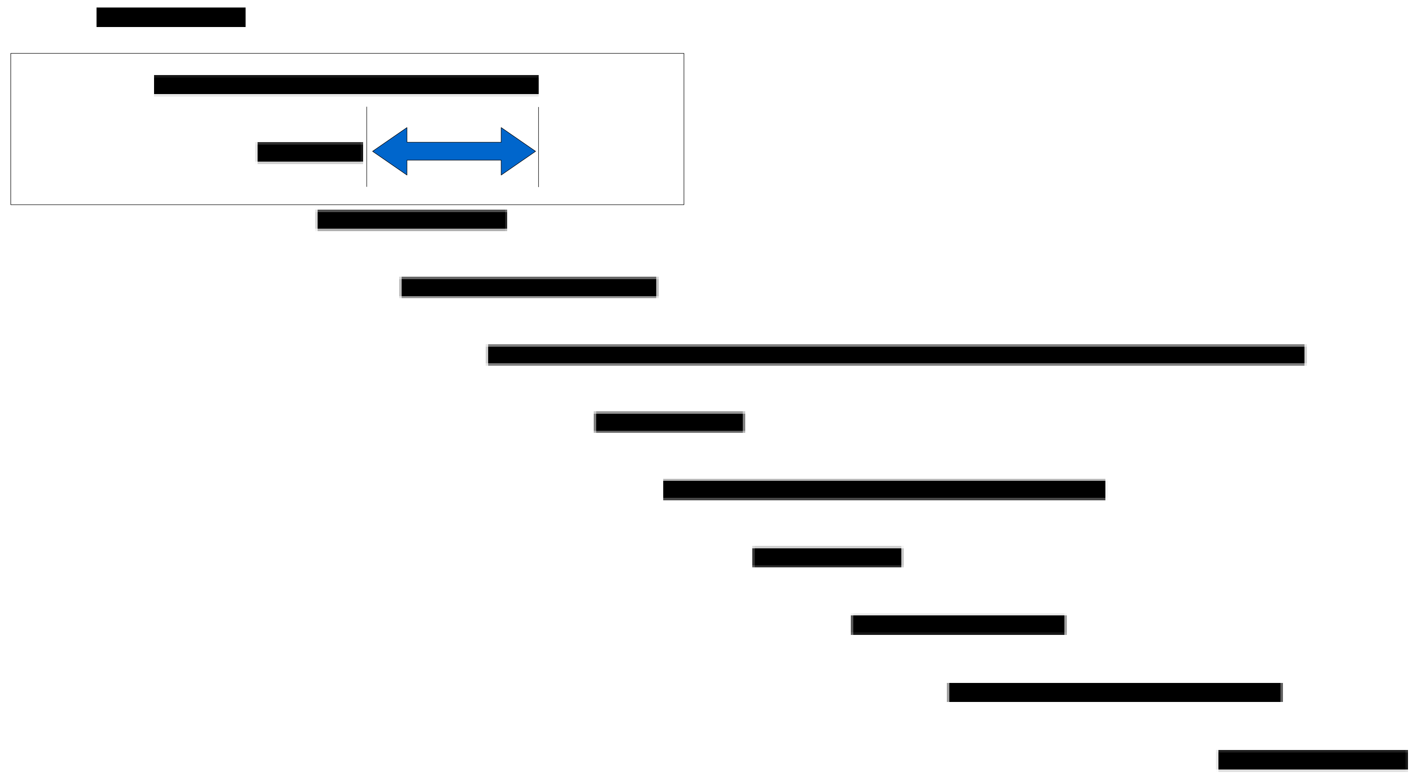
Real feature data isn't like that



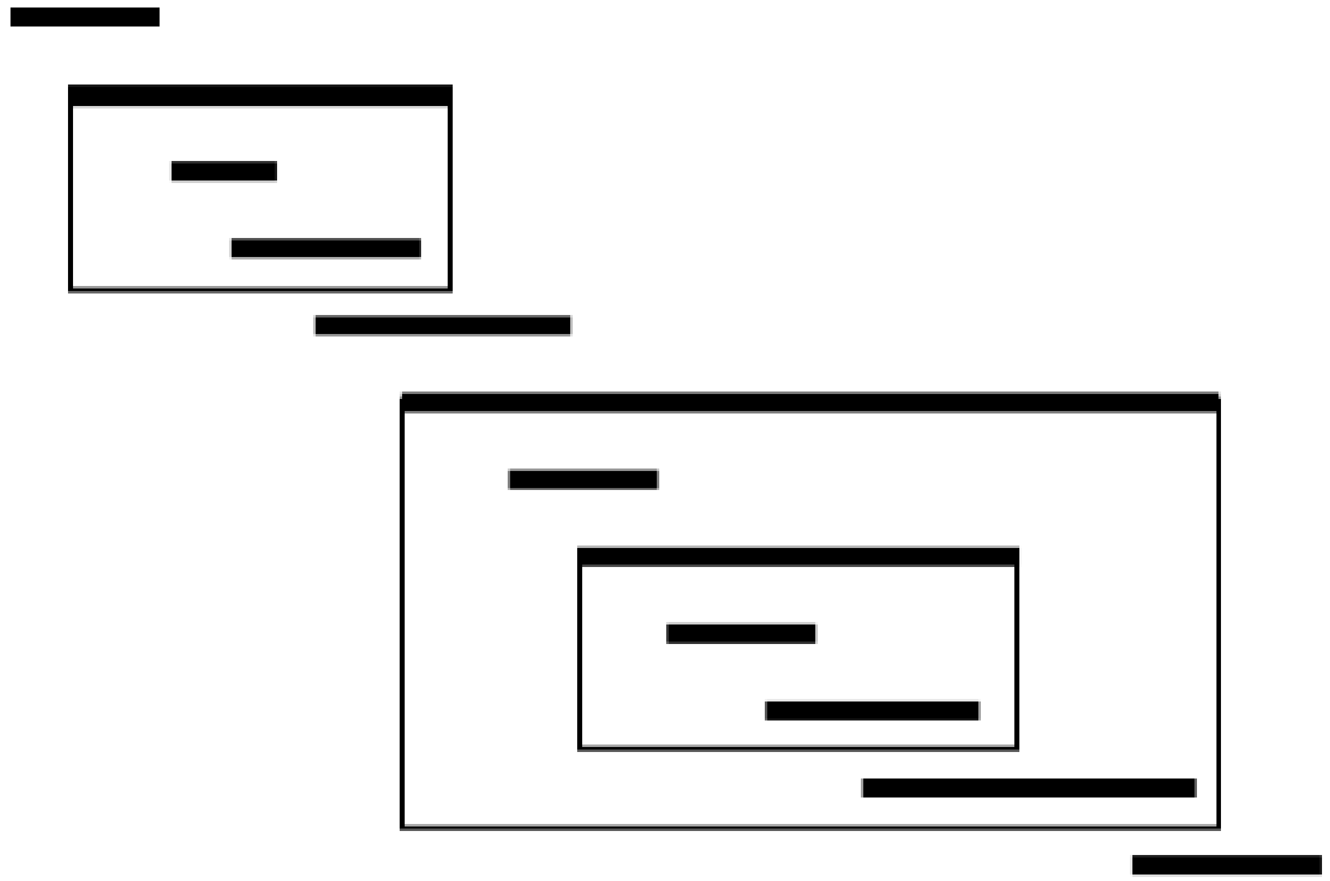
Sorted on start, but not on end



Containment



Nested Containment Lists



Alekseyenko A, Lee C (2007) Bioinformatics 23:1386–1393

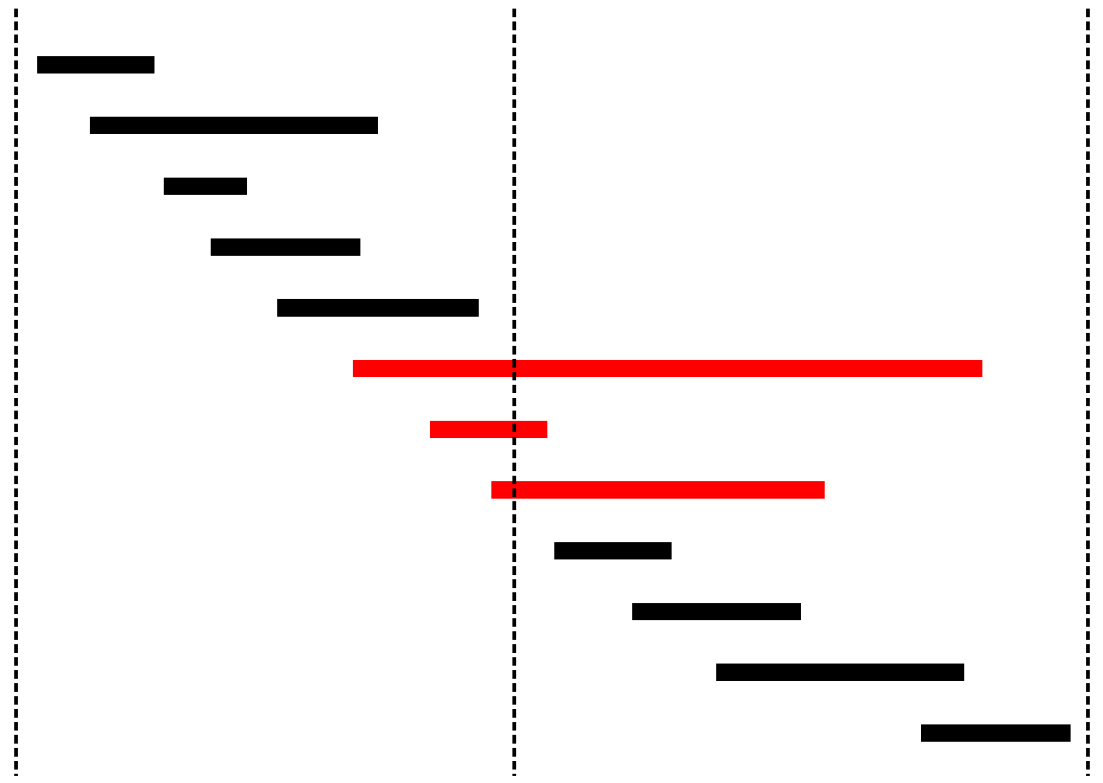
Nice Things About NCLists

- Simple to implement
  - Just a recursive binary search + iteration
- Tree structured
  - Just like JSON (!)
- Fast

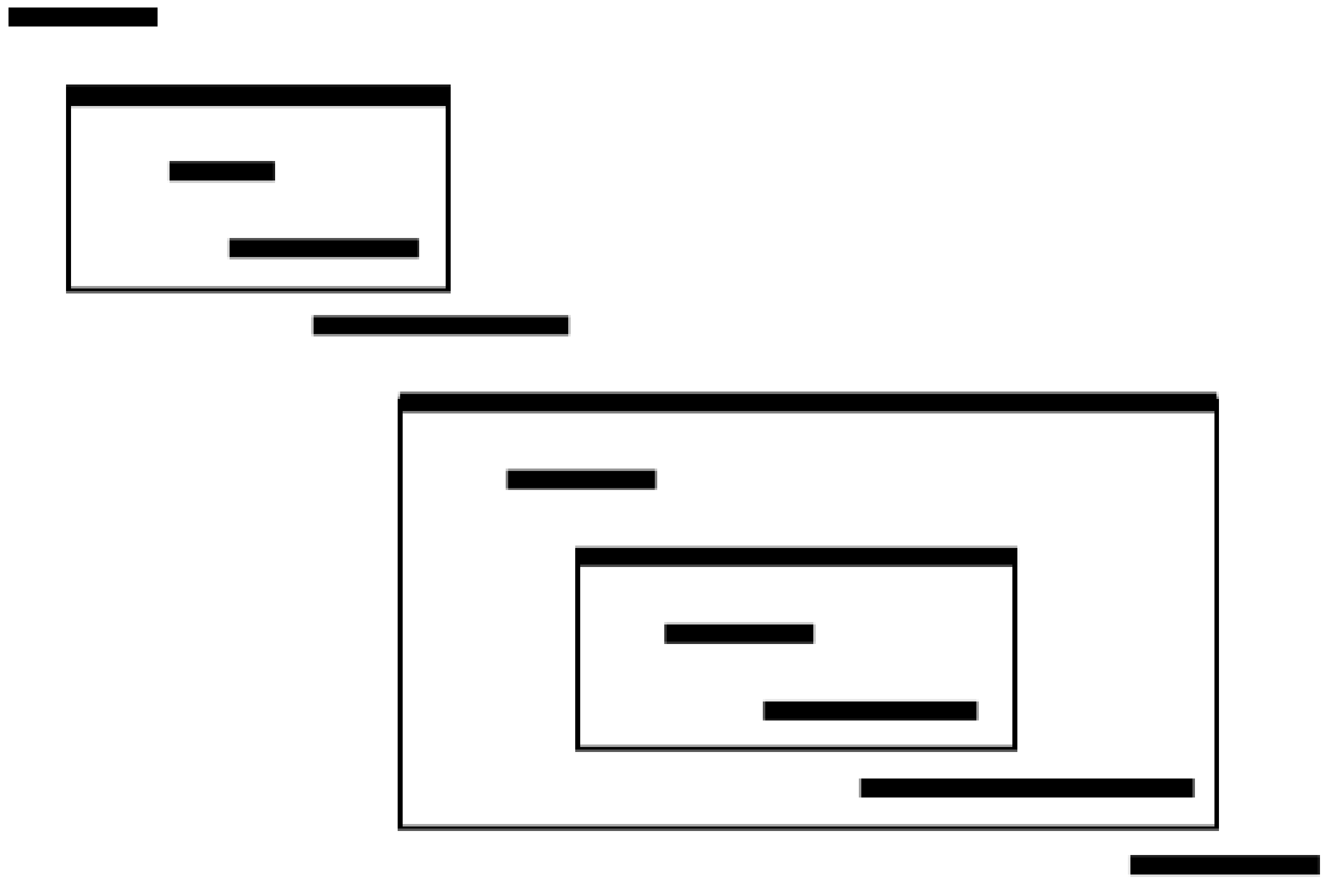
Scalability

Large numbers of features?

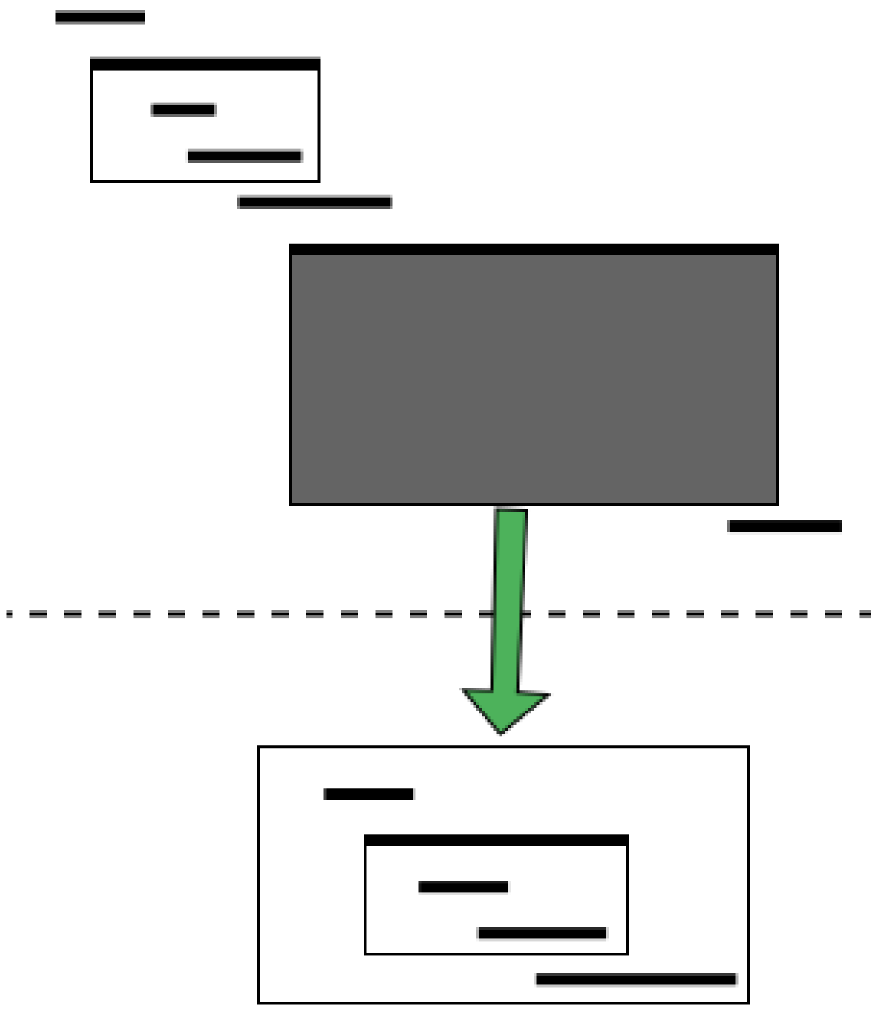
How to break up the data?



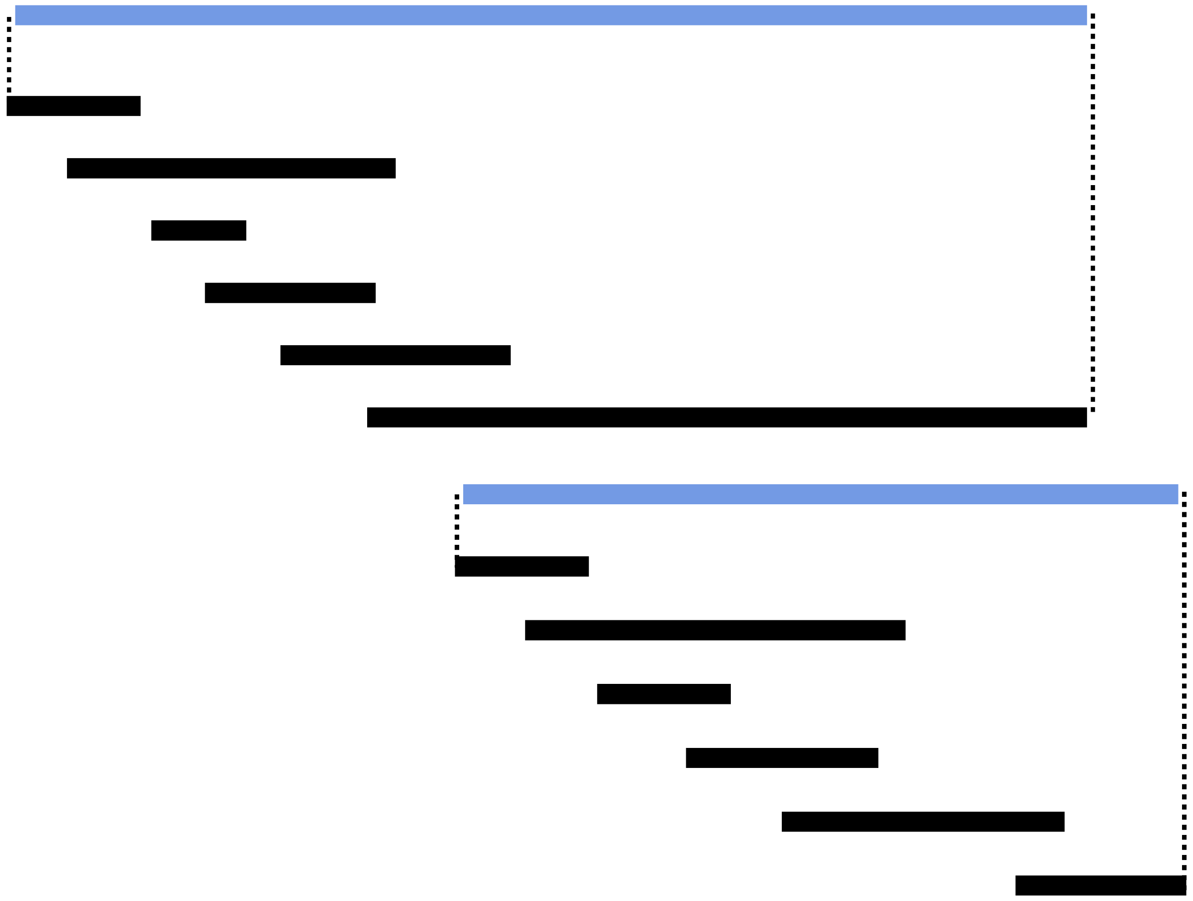
JBrowse uses NCLists



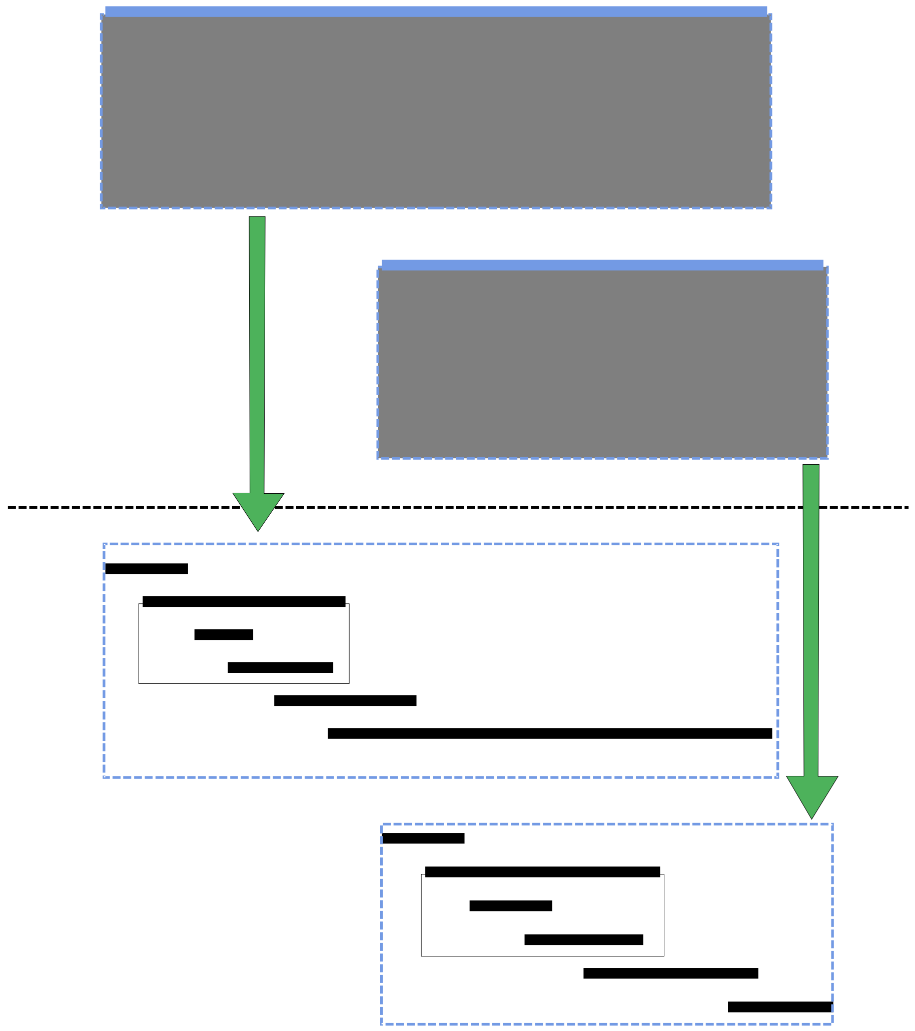
Lazy NCLists?



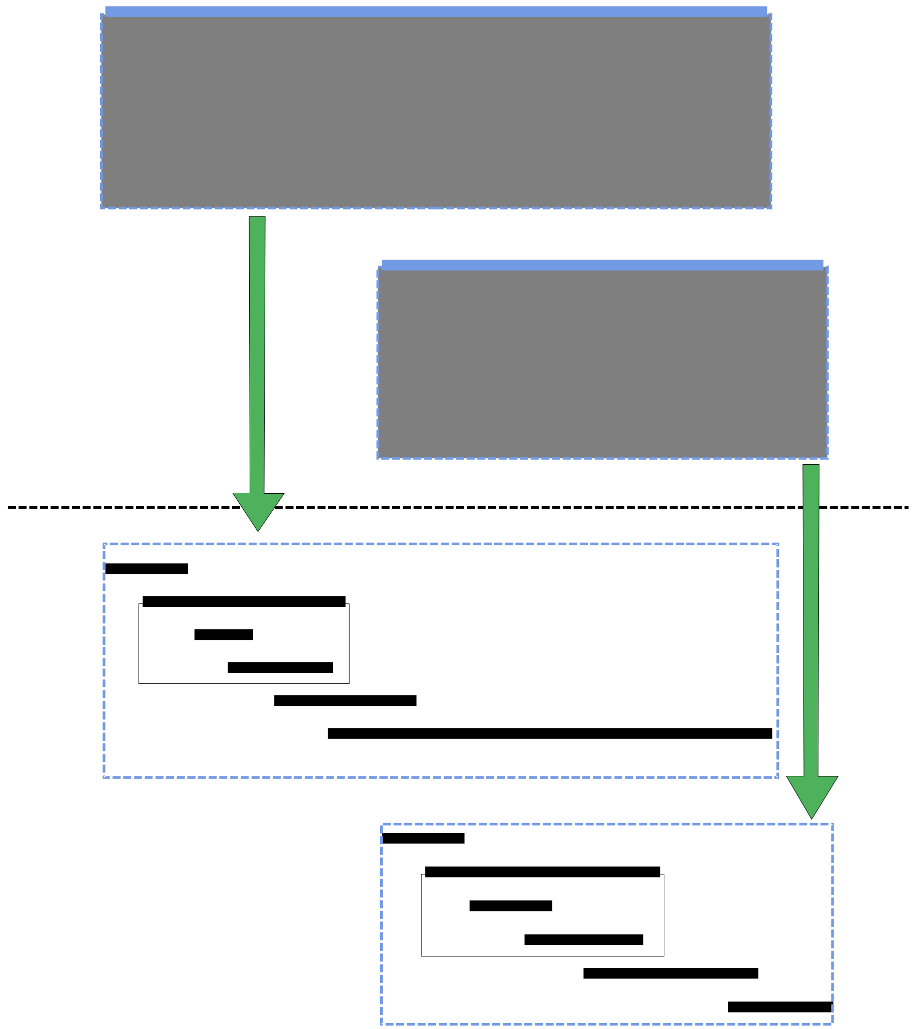
“fake” features



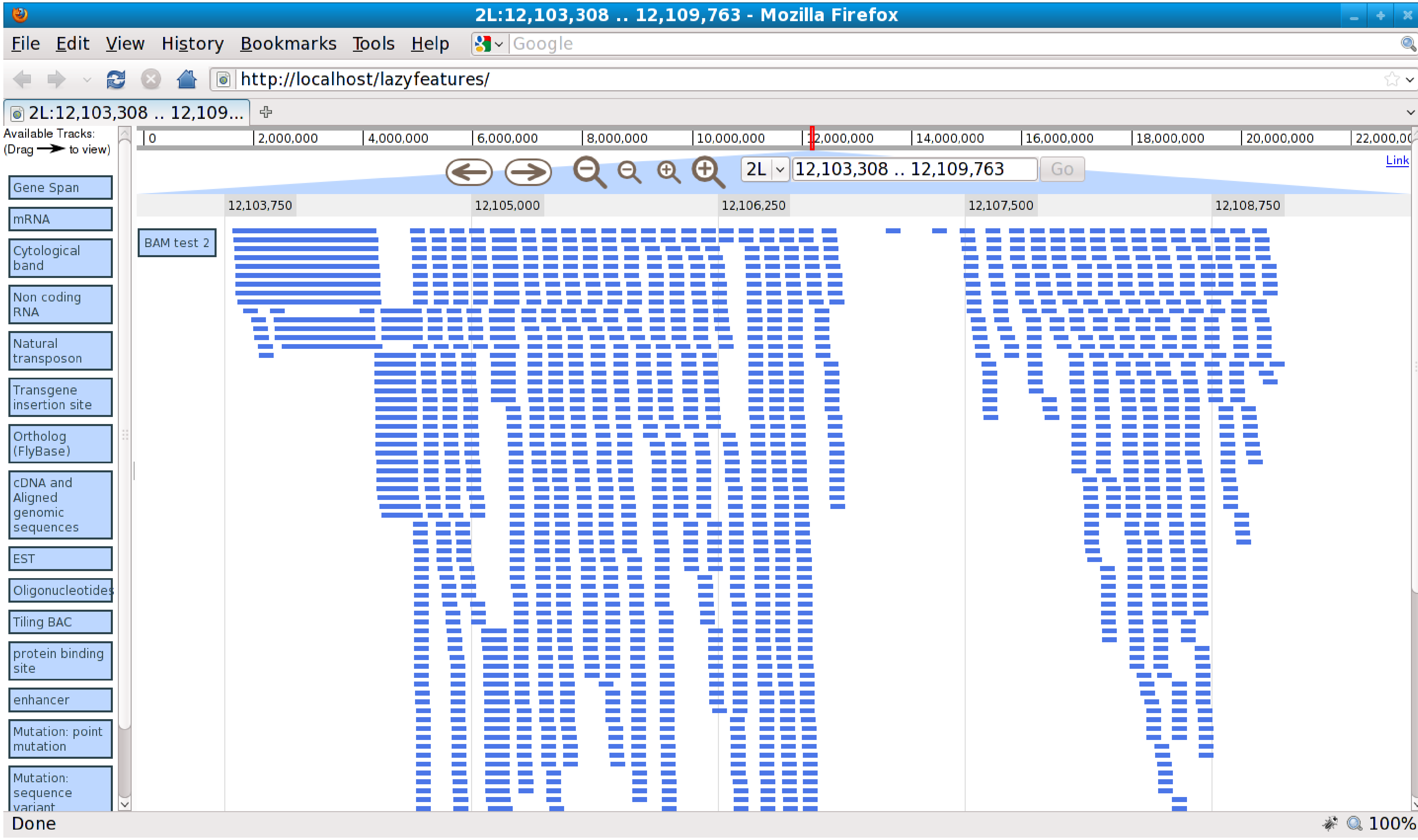
Lazy Loading in JBrowse



Lazy Loading in JBrowse



Alekseyenko A, Lee C (2007) Bioinformatics 23:1386–1393

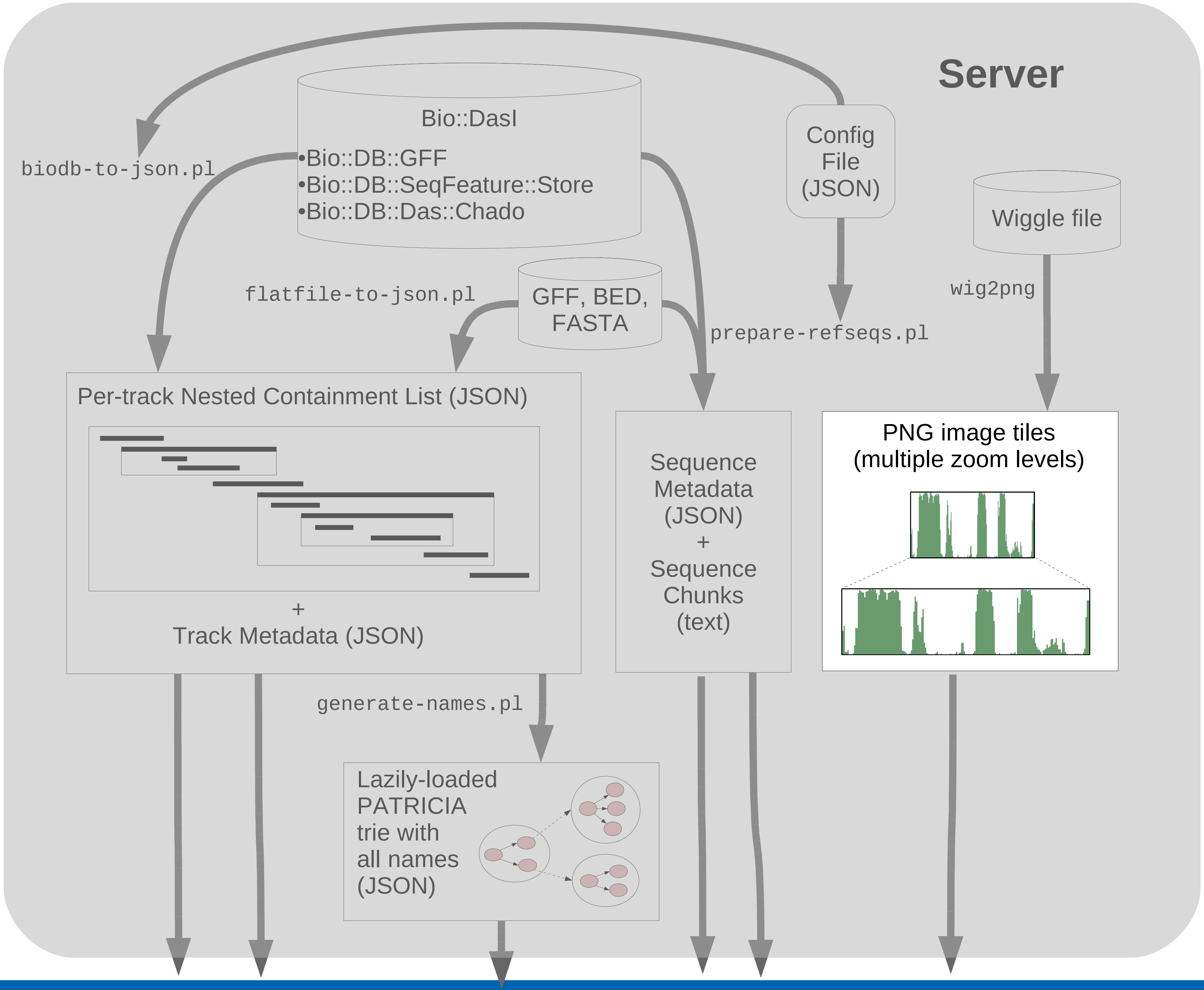

2L:12,103,308 .. 12,109,763 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Google
http://localhost/lazyfeatures/
2L:12,103,308 .. 12,109...
Available Tracks:
(Drag → to view)
Gene Span
mRNA
Cytological band
Non coding RNA
Natural transposon
Transgene insertion site
Ortholog (FlyBase)
cDNA and Aligned genomic sequences
EST
Oligonucleotides
Tiling BAC
protein binding site
enhancer
Mutation: point mutation
Mutation: sequence variant
0
2,000,000
4,000,000
6,000,000
8,000,000
10,000,000
12,000,000
14,000,000
16,000,000
18,000,000
20,000,000
22,000,000
2L
12,103,308 .. 12,109,763
Go
Link
12,103,750
12,105,000
12,106,250
12,107,500
12,108,750
BAM test 2
Done
100%

BAM example

- On one test data set:
  - 4.4 million features
  - 8 minutes to process
    - From 242 megabyte BAM file
    - Not paired-end
  - Used 400 megabytes of RAM
  - 330 megabytes on disk (without sequence)
  - Compresses down to 80 megabytes

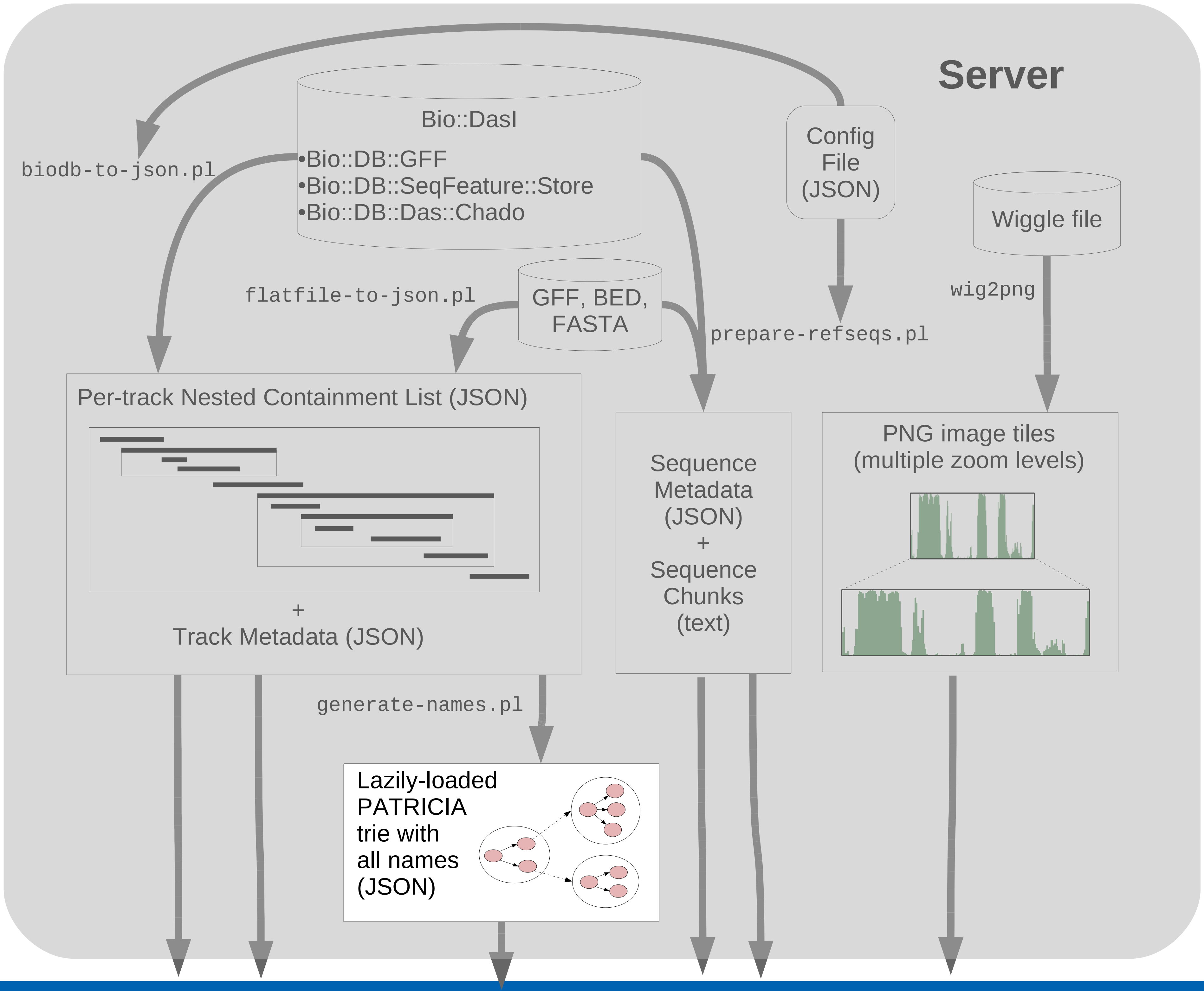
Other approaches to lazy loading

- Heng Li (SAMTools)
  - Binning, linear index
- Jim Kent (BigBed/BigWig)
  - R-Trees
- JBrowse javascript client can't use them directly
  - Can convert them to JSON
  - Or, potentially, access them through a proxy


Server
Bio::DasI
•Bio::DB::GFF
•Bio::DB::SeqFeature::Store
•Bio::DB::Das::Chado
biodb-to-json.pl
Config File (JSON)
Wiggle file
flatfile-to-json.pl
GFF, BED, FASTA
prepare-refseqs.pl
wig2png
Per-track Nested Containment List (JSON)
+
Track Metadata (JSON)
Sequence Metadata (JSON)
+
Sequence Chunks (text)
PNG image tiles (multiple zoom levels)
generate-names.pl
Lazily-loaded PATRICIA trie with all names (JSON)

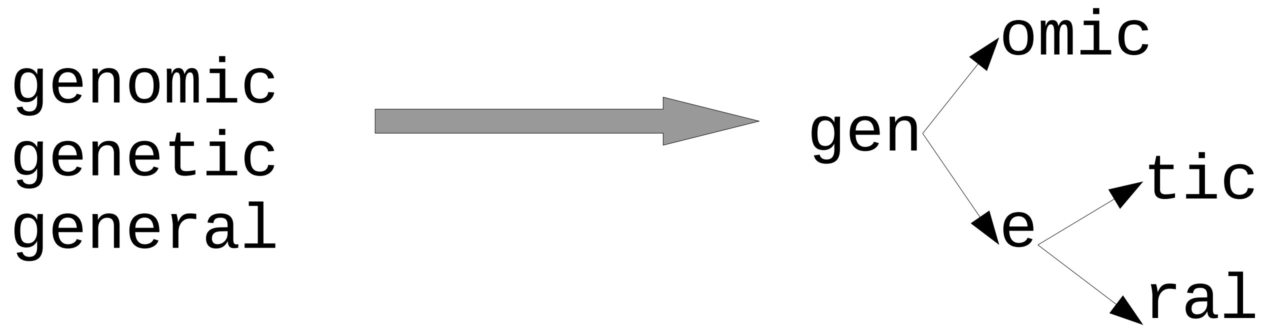
Wiggle tracks: pre-rendered

- Only rendered up to 1 base per pixel
- Implemented in C++
- ~12 min to generate tiles for Dmel conservation track (1 data point per base)
  - => ~1min per 10 million bases
- Wiggle tiles compress well
  - ~5 bytes/base, half of which is filesystem overhead
- They could also be rendered on the fly


Server
Bio::DasI
•Bio::DB::GFF
•Bio::DB::SeqFeature::Store
•Bio::DB::Das::Chado
biodb-to-json.pl
Config File (JSON)
Wiggle file
flatfile-to-json.pl
GFF, BED, FASTA
prepare-refseqs.pl
wig2png
Per-track Nested Containment List (JSON)
+
Track Metadata (JSON)
Sequence Metadata (JSON) + Sequence Chunks (text)
PNG image tiles (multiple zoom levels)
generate-names.pl
Lazily-loaded PATRICIA trie with all names (JSON)

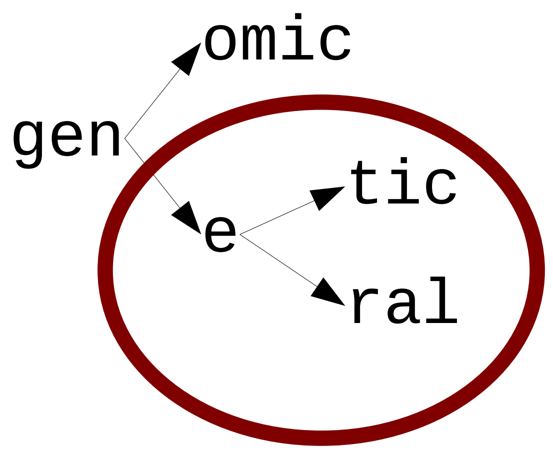
Name/ID searching: Trie

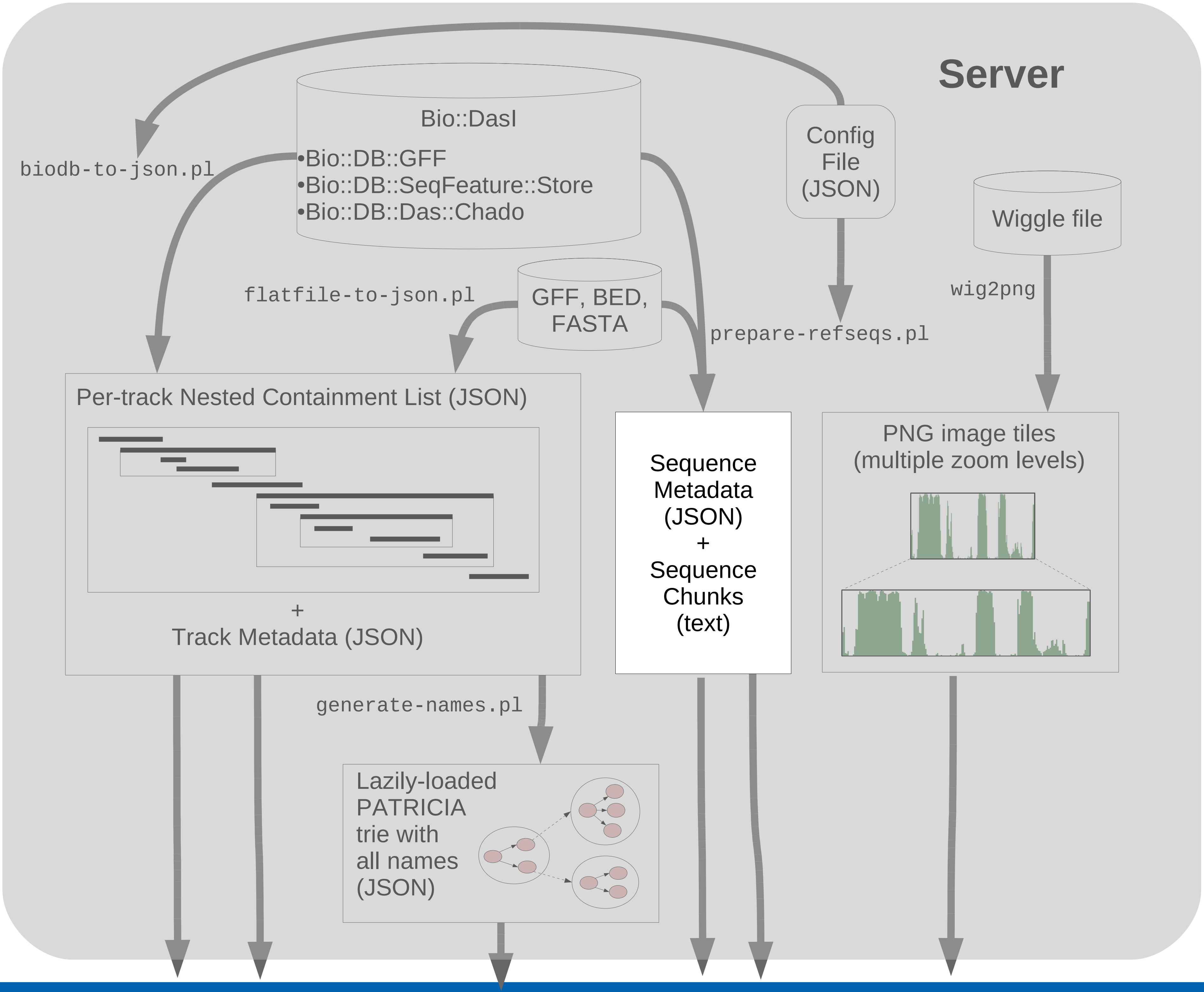
- Shares prefixes among a set of strings



Name/ID searching: Trie

- Subtries are lazily loaded




Server
Bio::DasI
•Bio::DB::GFF
•Bio::DB::SeqFeature::Store
•Bio::DB::Das::Chado
biodb-to-json.pl
Config File (JSON)
Wiggle file
flatfile-to-json.pl
GFF, BED, FASTA
prepare-refseqs.pl
wig2png
Per-track Nested Containment List (JSON)
+
Track Metadata (JSON)
Sequence Metadata (JSON) + Sequence Chunks (text)
PNG image tiles (multiple zoom levels)
generate-names.pl
Lazily-loaded PATRICIA trie with all names (JSON)

Summary

- Compared to existing web-based genome browsers, JBrowse:
  - Moves work from server to client
  - Moves work from read-time to write-time
- Caching
  - Offline usage?
- Scalability: it works, still some bugs
- Intended to fit in as a component of a larger system

Thanks

- Ian Holmes
- Andrew Uzilov
- Lincoln Stein
- Chris Mungall
- GMOD
  - Scott Cain
  - Dave Clements
- BioPerl
- NHGRI
- Users
  - Brenton Graveley

jbrowse.org

Pre-rendering isn't totally crazy

```
000010000     000010000
010011010     010001010
111111011     101100001
111111111     000000100
111111111     000000000
```