JBrowse
How JBrowse is different
Most Web-based Genome Browsers

Server Code (e.g., CGI)

HTTPD (e.g., Apache)

Web Browser

User

exec

image

request

image

click

view

exec

image

request

image

view

exec

image

view
JBrowse

Server Code (e.g., CGI)

HTTPD (e.g., Apache)

Web Browser (with javascript)

User

Request

JSON

View

Click

JSON
HTTPD (e.g., Apache) → Web Browser (with javascript) → User

Server Code (e.g., CGI) → JBrowse

JSON

request

view

click

view

JSON
Most genome browsers

Server
JBrowse moves work:

Server -> client

On the server:
Read-time -> write-time
Most Web-based Genome Browsers

**CPU**

**Disk Space**

**Write time**

**Read time**

**JBrowse**

**CPU**

**Disk Space**
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)
    - Broken into ~40 kilobyte chunks
  - Compresses down to 80 megabytes
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)
  - => ~1 min 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
Most Web-based Genome Browsers

JBrowse

CPU

Disk Space

Write time

Read time
Most Web-based Genome Browsers

CPU

Disk Space

JBrowse

CPU

Disk Space

Write time

Read time
JBrowse

Server Code (e.g., CGI)

HTTPD (e.g., Apache)

Web Browser (with javascript)

User

HTTP

JSON

request

JSON

User

click

view

User

click

view

User

click

view
JBrowse

Server Code (e.g., CGI)

HTTPD (e.g., Apache)

Web Browser (with javascript)

User

JSON

click

view

request

JSON

HTTP
Summary: JBrowse...

Moves work from server to client

Moves work from read-time to write-time

HTTPD (e.g., Apache)  Web Browser (with javascript)  User

HTTP

JSON

request

JSON

? 

Allows the client to cache useful amounts of data
Lower server load means:

• The user waits much less

• Cheaper/easier to host a genome browser
Client-side approach

- Richer interaction
  - Smooth, continuous transitions
    - Help the user build an intuitive sense of where things are relative to one another
  - Client-rendered graphics: client can filter, highlight, etc.
- Web Apollo
Why it wasn't done earlier

• Getting the web browser to do the work is non-trivial
  • Some web browsers have mechanisms intended to enable the browser to render graphics (e.g., SVG, canvas)
  • None of those mechanisms work in all browsers (Internet explorer doesn't have SVG or canvas)
HTML element
<table>
<thead>
<tr>
<th>GBrowse</th>
<th>JBrowse</th>
</tr>
</thead>
<tbody>
<tr>
<td>More</td>
<td>Faster, smoother</td>
</tr>
<tr>
<td>functionality</td>
<td>UI</td>
</tr>
<tr>
<td>Does work on server</td>
<td>Moves work to web browser</td>
</tr>
<tr>
<td>GBrowse</td>
<td>JBrowse</td>
</tr>
<tr>
<td>-----------</td>
<td>------------</td>
</tr>
<tr>
<td>Same underlying perl machinery</td>
<td></td>
</tr>
<tr>
<td>Same data sources (GFF, BED, WIG, SAM/BAM...)</td>
<td></td>
</tr>
</tbody>
</table>