

Package: JBrowseR (via r-universe)

October 10, 2026

Title An R Interface to the JBrowse 2 Genome Browser

Version 0.14.0

Description Provides an R interface to the JBrowse 2 genome browser. Enables embedding a JB2 genome browser in a Shiny app or R Markdown document. The browser can also be launched from an interactive R console. The browser can be loaded with a variety of common genomics data types, and can be used with a custom theme.

License Apache License (>= 2)

URL <https://gmod.github.io/JBrowseR/> <https://github.com/GMOD/JBrowseR>

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports htmlwidgets

Suggests jsonlite, testthat (>= 3.0.0), knitr, rmarkdown, shiny

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake make libuv1-dev

Repository <https://gmod.r-universe.dev>

Date/Publication 2026-10-10 19:19:07 UTC

RemoteUrl <https://github.com/gmod/jbrowser>

RemoteRef HEAD

RemoteSha b7e4bc46a20e75356ce753f1a5326ba754f97a21

Contents

JBrowseR	2
JBrowseR-shiny	3

JBrowseRApp	4
JBrowseRApp-shiny	5
track_data_frame	6
update_jbrowse	7
Index	9

JBrowseR	<i>Embed a JBrowse 2 linear genome view</i>
----------	---

Description

Renders an interactive, GPU-accelerated JBrowse 2 linear genome view as an htmlwidget for R Markdown documents, Shiny apps, or the R console.

Usage

```
JBrowseR(  
  ...,  
  local_files = NULL,  
  width = NULL,  
  height = NULL,  
  elementId = NULL  
)
```

Arguments

- ... createLinearGenomeView options, each named.
- local_files Files on this machine to open without a web server: a path, a vector of paths, or a list mixing paths with raw vectors. Each registers under its basename (or its list name), and a track refers to that name as if it were a URL. A sibling index (.tbi, .csi, .bai, .crai, .fai, .gzi) next to a path comes along.
- width, height, elementId Standard htmlwidget sizing arguments.

Details

Every named argument in ... is an option of JBrowse’s **createLinearGenomeView**, sent verbatim under JBrowse’s own camelCase name: assembly, tracks, location, session, aggregateTextSearchAdapters, internetAccounts, plugins, configuration, and whatever JBrowse adds next. This package names none of them, so a whole options object is do.call(JBrowseR, jsonlite::read_json("options.json")).

assembly takes a hub name ("hg38", a GenArk accession), a sequence-file URL, or an assembly config. A tracks entry takes a bare data-file URL, a list(uri =), a [track_data_frame\(\)](#) result, or a full track config. plugins entries are list(name = , url =).

A length-1 vector serializes to a JSON scalar, so fields JBrowse reads as arrays take list(): assemblyNames = list("hg38").

Value

an htmlwidget

Examples

```
JBrowseR(assembly = "hg38", location = "BRCA1")
```

JBrowseR-shiny

Shiny bindings for JBrowseR

Description

Output and render functions for [JBrowseR\(\)](#) in Shiny apps and interactive Rmd documents.

Usage

```
JBrowseROutput(outputId, width = "100%", height = "400px")

renderJBrowseR(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

outputId	output variable to read from
width, height	a valid CSS unit, or a number coerced to pixels
expr	An expression that generates a JBrowseR
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())?

Details

The widget reports three inputs, namespaced by output id: `input[[paste0(outputId, "_selected_feature")]]` is the clicked feature, `_location` the visible region as the location box prints it (thousands separators included), and `_session` the layout in the shape `session = takes`. Each settles after a gesture rather than firing per frame.

A re-render whose options differ only in `tracks`, `location` or `local_files` is applied to the browser on the page, keeping the user's zoom and track order; any other changed option builds a new browser. Reading `_location` or `_session` in the reactive that feeds `renderJBrowseR()` builds a loop: navigate from an observer with [update_jbrowse\(\)](#) instead.

Value

the Shiny UI bindings for a JBrowseR htmlwidget
the Shiny server bindings for a JBrowseR htmlwidget

JBrowseRApp

Embed the full JBrowse 2 app (multiple views of any type)

Description

Where `JBrowseR()` shows one linear genome view, `JBrowseRApp()` runs the whole app, so views can mix a linear view, a synteny view, a dotplot, and more.

Usage

```
JBrowseRApp(
  ...,
  local_files = NULL,
  width = NULL,
  height = NULL,
  elementId = NULL
)
```

Arguments

... createApp options, each named.

local_files Files on this machine to open without a web server: a path, a vector of paths, or a list mixing paths with raw vectors. Each registers under its basename (or its list name), and a track refers to that name as if it were a URL. A sibling index (`.tbi`, `.csi`, `.bai`, `.crai`, `.fai`, `.gzi`) next to a path comes along.

width, height, elementId Standard htmlwidget sizing arguments.

Details

Every named argument in ... is an option of JBrowse's `createApp`, sent verbatim under JBrowse's own camelCase name: `assemblies`, `tracks`, `views`, `session`, `connections`, `internetAccounts`, `aggregateTextSearchAdapters`, `configuration`, `plugins`, and whatever JBrowse adds next. A whole options object is `do.call(JBrowseRApp, jsonlite::read_json("options.json"))`.

A views entry is `list(type = , ...)` with the view's settings beside type, the object a `config.json`'s `defaultSession.views` holds. An `assemblies` entry may be a hub name or sequence-file URL, as `JBrowseR()`'s assembly takes.

Value

an htmlwidget

Examples

```
## Not run:
JBrowseRApp(
  assemblies = list(
    list(name = "hg38", uri = hg38_fa),
    list(name = "mm39", uri = mm39_fa)
  ),
  tracks = list(list(
    type = "SyntenyTrack",
    trackId = "hg38_mm39",
    name = "hg38 vs mm39",
    assemblyNames = list("hg38", "mm39"),
    adapter = list(
      type = "PAFAdapter",
      targetAssembly = "hg38",
      queryAssembly = "mm39",
      uri = paf_url
    )
  )),
  views = list(list(
    type = "LinearSyntenyView",
    views = list(list(assembly = "hg38"), list(assembly = "mm39")),
    tracks = list("hg38_mm39")
  ))
)

## End(Not run)
```

JBrowseRApp-shiny

Shiny bindings for JBrowseRApp

Description

Output and render functions for `JBrowseRApp()`. The app is a separate widget with its own bundle, so it cannot render into a `JBrowseROutput()`.

Usage

```
JBrowseRAppOutput(outputId, width = "100%", height = "400px")
```

```
renderJBrowseRApp(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

<code>outputId</code>	output variable to read from
<code>width, height</code>	a valid CSS unit, or a number coerced to pixels
<code>expr</code>	An expression that generates a <code>JBrowseRApp</code>
<code>env</code>	The environment in which to evaluate <code>expr</code> .
<code>quoted</code>	Is <code>expr</code> a quoted expression (with <code>quote()</code>)?

Details

The app reports `input[[paste0(outputId, "_session")]]` in the shape `session = takes, _selected_feature` for the clicked feature, and `_location` as a list with one entry per open view. A re-render that changes only `session` restores it in the app on the page; any other changed option builds a new app.

Value

- the Shiny UI bindings for a `JBrowseRApp` `htmlwidget`
- the Shiny server bindings for a `JBrowseRApp` `htmlwidget`

track_data_frame	<i>Build a track from an R data frame</i>
------------------	---

Description

Turns a data frame of features into an in-browser track (a `FromConfigAdapter`), no files or web server required. This is the natural way to view results you computed in R — peaks, windows, hits — directly on the genome.

Usage

```
track_data_frame(data, name, assembly_name = NULL, ...)
```

Arguments

- | | |
|---------------|---|
| data | A data frame with columns <code>chrom</code> (or <code>chr</code> , or <code>refName</code>), <code>start</code> and <code>end</code> (0-based, half-open). An optional <code>score</code> column makes it a quantitative track; every other column rides along as a feature attribute, shown in the feature details and readable by a display’s encoding. |
| name | Track display name. |
| assembly_name | Assembly the track belongs to. Usually left <code>NULL</code> — <code>JBrowseR()</code> backfills it from the loaded assembly. |
| ... | Extra config merged into the track. A displays list plots the columns the way a grammar of graphics does: a <code>LinearMarkDisplay</code> maps a column to <code>y</code> and another to a colour scale. |

Value

- a track config list

Examples

```
df <- data.frame(
  chrom = c("1", "2"),
  start = c(123, 456),
  end = c(789, 101112),
  name = c("feature1", "feature2")
)
track_data_frame(df, "my_features")

# a point per row at its log2 fold-change, coloured by call
de <- data.frame(
  chrom = "7", start = c(1e6, 2e6), end = c(1e6, 2e6) + 6000,
  log2fc = c(2.1, -1.7), sig = c("up", "down")
)
track_data_frame(de, "de", displays = list(list(
  type = "LinearMarkDisplay",
  marks = list(list(
    mark = "point",
    encoding = list(y = "log2fc", color = list(field = "sig", scale = "categorical"))
  ))
)))
```

update_jbrowse

Change options of a rendered browser

Description

Sends option changes to a [JBrowseR\(\)](#) or [JBrowseRApp\(\)](#) already on the page, without re-running its render expression. The changes are named the way the widget's own options are.

Usage

```
update_jbrowse(outputId, ..., domain = shiny_domain())
```

Arguments

outputId	The browser's output id. Inside a Shiny module, pass the bare id, as you would to output\$.
...	Options to change, each named.
domain	The Shiny session, defaulting to the current one. Not named session, which is an option.

Details

```
observeEvent(input$gene, update_jbrowse("browser", location = input$gene))
```

A [JBrowseR\(\)](#) applies tracks and location in place and a [JBrowseRApp\(\)](#) applies session in place; any other option builds a new browser from the rendered options plus the changes. A later re-render compares against what it rendered last, so a change sent here survives a re-render that does not restate that option.

8

update_jbrowse

Value

outputId, invisibly.

Index

JBrowseR, [2](#)
JBrowseR(), [3](#), [4](#), [6](#), [7](#)
JBrowseR-shiny, [3](#)
JBrowseRApp, [4](#)
JBrowseRApp(), [5](#), [7](#)
JBrowseRApp-shiny, [5](#)
JBrowseRAppOutput (JBrowseRApp-shiny), [5](#)
JBrowseROutput (JBrowseR-shiny), [3](#)
JBrowseROutput(), [5](#)

renderJBrowseR (JBrowseR-shiny), [3](#)
renderJBrowseRApp (JBrowseRApp-shiny), [5](#)

track_data_frame, [6](#)
track_data_frame(), [2](#)

update_jbrowse, [7](#)
update_jbrowse(), [3](#)