

# Package ‘GenomicCoordinates’

May 9, 2026

**Title** Enhanced string parsing for genomic coordinates

**Version** 1.1.0

**Description** Extends string parsing capabilities for genomic coordinates, supporting various formats including comma-separated numbers, space-delimited coordinates, and automatic detection of GRanges, GPos, and GInteractions objects.

**License** Artistic-2.0

**Encoding** UTF-8

**URL** <https://github.com/js2264/GenomicCoordinates>

**BugReports** <https://github.com/js2264/GenomicCoordinates/issues>

**Depends** R (>= 4.5), GenomicRanges, IRanges

**Imports** S4Vectors, Seqinfo, InteractionSet, methods, plyranges, plyinteractions

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, BiocStyle

**VignetteBuilder** knitr

**biocViews** Infrastructure, DataRepresentation, GenomeAnnotation

**RoxygenNote** 7.3.3

**git\_url** <https://git.bioconductor.org/packages/GenomicCoordinates>

**git\_branch** devel

**git\_last\_commit** 0a24b7e

**git\_last\_commit\_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-05-08

**Author** Jacques Serizay [aut, cre] (ORCID:  
<<https://orcid.org/0000-0002-4295-0624>>)

**Maintainer** Jacques Serizay <jacquesserizay@gmail.com>

## Contents

GenomicCoordinates-package . . . . .	2
.clean_numeric_string . . . . .	3
.handle_special_formats . . . . .	3
.is_single_position . . . . .	4

as_gpos	4
coercion	5
detect_genomic_class	6
GenomicCoordinates	7
reexports	8

<b>Index</b>	<b>9</b>
--------------	----------

---

GenomicCoordinates-package

*GenomicCoordinates: Enhanced string parsing for genomic coordinates*

---

## Description

The GenomicCoordinates package extends the string parsing capabilities for genomic coordinates in Bioconductor. It supports various string formats including comma-separated numbers, space-delimited coordinates, and automatically detects whether to return GRanges, GPos, or GInteractions objects.

## Supported formats

- Standard format: "chr1:1000-2000", "chr1:1000-2000:+"
- Comma-separated: "chr1:1,000-2,000", "chr1:1,000,000-2,000,000"
- Space-delimited: "chr1 1000 2000"
- Single positions: "chr1:1000" (returns GPos)
- Interactions: "chr1:1000-2000|chr2:3000-4000" (returns GInteractions)

## Main functions

- GenomicCoordinates(x): Main function - auto-detect and convert to appropriate type
- detect\_genomic\_class(x): Detect appropriate class without parsing
- as\_granges(x): Convert character to GRanges
- as\_gpos(x): Convert character to GPos
- as\_iranges(x): Convert character to IRanges
- as\_ginteractions(x): Convert character to GInteractions

## Author(s)

**Maintainer:** Jacques Serizay <jacquesserizay@gmail.com> ([ORCID](#))

## See Also

Useful links:

- <https://github.com/js2264/GenomicCoordinates>
- Report bugs at <https://github.com/js2264/GenomicCoordinates/issues>

---

`.clean_numeric_string` *Parse genomic coordinate strings with enhanced format support*

---

### **Description**

Internal utility functions to parse various genomic coordinate string formats including comma-separated numbers, space-delimited coordinates, and different separators.

### **Usage**

```
.clean_numeric_string(x)
```

### **Arguments**

`x`                    A character string representing genomic coordinates

### **Value**

Parsed components as a list

---

`.handle_special_formats`  
*Handle special genomic string formats*

---

### **Description**

Handle special genomic string formats

### **Usage**

```
.handle_special_formats(x)
```

### **Arguments**

`x`                    Character string

### **Value**

Parsed genomic information

`.is_single_position`     *Additional helper functions for corner cases*

---

### Description

Handle various edge cases and special formats in genomic coordinate parsing. Detect if a string represents a single genomic position.

### Usage

```
.is_single_position(x)
```

### Arguments

`x`                      Character string

### Value

Logical indicating if it's a single position.

---

`as_gpos`                      *Convert to GPos object*

---

### Description

Converts character strings representing single genomic positions to GPos objects.

### Usage

```
as_gpos(.data, ...)
```

### Arguments

`.data`                      A character vector of genomic position strings  
`...`                      Additional arguments (unused)

### Value

A GPos object

### Examples

```
as_gpos("chr1:1000")  
as_gpos("chr1:1,000+")  
as_gpos(c("chr1:1000", "chr2:2000", "chr3:3000"))
```

---

coercion

*Conversion methods for genomic coordinates*


---

## Description

Methods to convert character strings to GRanges, GPos, and GInteractions objects with support for various string formats including comma-separated numbers and space-delimited coordinates.

Extensions to IRanges parsing to handle comma-separated numbers and space-delimited coordinates.

## Usage

```
## S4 method for signature 'character'
as_granges(.data, ..., keep_mcols = TRUE)

## S4 method for signature 'character'
as_gpos(.data, ...)

## S4 method for signature 'character'
as_ginteractions(
  .data,
  ...,
  keep.extra.columns = TRUE,
  starts.in.df.are.0based = FALSE
)

## S4 method for signature 'character'
as_iranges(.data, ..., keep_mcols = TRUE)
```

## Arguments

<code>.data</code>	A character vector of coordinate strings
<code>...</code>	Additional arguments (unused)
<code>keep_mcols</code>	Ignored for character input (included for generic compatibility with plyranges)
<code>keep.extra.columns</code>	Ignored for character input (included for generic compatibility with plyinteractions)
<code>starts.in.df.are.0based</code>	Ignored for character input (included for generic compatibility with plyinteractions)

## Value

The appropriate Bioconductor object type

An IRanges object

## Examples

```
# GRanges conversion
as_granges("chr1:1000-2000")
as_granges("chr1:1,000-2,000:+")
as_granges(c("chr1:1000-2000", "chr2:3000-4000"))

# GPos conversion
as_gpos("chr1:1000")
as_gpos(c("chr1:1000", "chr2:2000"))

# GInteractions conversion
as_ginteractions("chr1:1-10|chr2:20-30")

as_iranges("1000-2000")
as_iranges("1,000-2,000")
as_iranges(c("100-200", "300-400"))
```

---

detect\_genomic\_class *Detect the appropriate class for genomic strings*

---

## Description

Utility function to determine what class a genomic string should be parsed as, without actually performing the parsing.

## Usage

```
detect_genomic_class(x)
```

## Arguments

x                    Character string or vector

## Value

Character vector of predicted classes

## Examples

```
detect_genomic_class("chr1:1000-2000")
detect_genomic_class("chr1:1000")
detect_genomic_class(c("chr1:1-10|chr2:20-30", "1000-2000"))
```

---

GenomicCoordinates      *GenomicCoordinates: Main parsing function*

---

## Description

Automatically parse genomic coordinate strings into the most appropriate Bioconductor object type (GRanges, GPos, GInteractions, or IRanges). Parse strings into appropriate genomic objects

## Usage

```
GenomicCoordinates(x, force_class = NULL)
```

## Arguments

x	Character string or vector of genomic coordinates
force_class	Optional class to force ("GRanges", "GPos", "GInteractions", "IRanges")

## Details

This is the main function of the GenomicCoordinates package. It automatically detects the most appropriate object type based on the input string format and returns the corresponding Bioconductor object.

## Value

GRanges, GPos, GInteractions, or IRanges object

## Examples

```
# Auto-detection examples
GenomicCoordinates("chr1:1000-2000")           # Returns GRanges
GenomicCoordinates("chr1:1000")                # Returns GPos
GenomicCoordinates("chr1:1-10|chr2:4-40")      # Returns GInteractions
GenomicCoordinates("1000-2000")                # Returns IRanges

# Force specific class
GenomicCoordinates("chr1:1000", force_class = "GRanges")

# Enhanced format support
GenomicCoordinates("chr1:100,000-200,000")     # Comma-separated
GenomicCoordinates("chr1 1000 2000")          # Space-delimited
```

---

reexports

*Re-exported functions from plyranges and plyinteractions*

---

## Description

These generics are re-exported from plyranges and plyinteractions to provide conversion functions for character strings.

## Usage

```
as_granges(.data, ..., keep_mcols = TRUE)

as_iranges(.data, ..., keep_mcols = TRUE)

as_ginteractions(
  .data,
  ...,
  keep.extra.columns = TRUE,
  starts.in.df.are.0based = FALSE
)
```

## Arguments

<code>.data</code>	Object to convert
<code>...</code>	Additional arguments passed to methods
<code>keep_mcols</code>	Logical; whether to keep metadata columns (plyranges)
<code>keep.extra.columns</code>	Logical; whether to keep extra columns (plyinteractions)
<code>starts.in.df.are.0based</code>	Logical; whether starts are 0-based (plyinteractions)

## Value

A Bioconductor object

## Examples

```
as_granges("chr1:1000-2000")
as_iranges("1000-2000")
as_ginteractions("chr1:1-10|chr2:20-30")
```

# Index

## \* internal

- [.clean\\_numeric\\_string](#), 3
  - [GenomicCoordinates-package](#), 2
- [.clean\\_numeric\\_string](#), 3
- [.handle\\_special\\_formats](#), 3
- [.is\\_single\\_position](#), 4
  
- [as\\_ginteractions](#) (reexports), 8
- [as\\_ginteractions](#), character-method (coercion), 5
- [as\\_gpos](#), 4
- [as\\_gpos](#), character-method (coercion), 5
- [as\\_granges](#) (reexports), 8
- [as\\_granges](#), character-method (coercion), 5
- [as\\_iranges](#) (reexports), 8
- [as\\_iranges](#), character-method (coercion), 5
  
- [coercion](#), 5
  
- [detect\\_genomic\\_class](#), 6
  
- [GenomicCoordinates](#), 7
- [GenomicCoordinates-package](#), 2
  
- [reexports](#), 8