Package 'genbankr'

October 16, 2020

Version 1.16.0

Title Parsing GenBank files into semantically useful objects

Description Reads Genbank files.

Copyright Genentech, Inc.

Maintainer Gabriel Becker <becker.gabriel@gene.com>

License Artistic-2.0

Depends methods

Imports BiocGenerics, IRanges (>= 2.13.15), GenomicRanges (>= 1.31.10), GenomicFeatures (>= 1.31.5), Biostrings, VariantAnnotation, rtracklayer, S4Vectors (>= 0.17.28), GenomeInfoDb, Biobase

Suggests RUnit, rentrez, knitr, rmarkdown, BiocStyle

VignetteBuilder knitr

RoxygenNote 5.0.1.9000

biocViews Infrastructure, DataImport

NeedsCompilation no

git_url https://git.bioconductor.org/packages/genbankr

git_branch RELEASE_3_11

git_last_commit 349d18f

git_last_commit_date 2020-04-27

Date/Publication 2020-10-15

Author Gabriel Becker [aut, cre], Michael Lawrence [aut]

R topics documented:

`cds,GenBankRecord-method`

Annotation extraction api

Description

Accessor functions shared with the larger Bioconductor ecosystem.

Usage

```
## S4 method for signature 'GenBankRecord'
cds(x)

## S4 method for signature 'GenBankRecord'
exons(x)

## S4 method for signature 'GenBankRecord'
genes(x)

## S4 method for signature 'GenBankRecord'
transcripts(x)

## S4 method for signature 'GenBankRecord'
getSeq(x, ...)

## S4 method for signature 'GenBankFile'
getSeq(x, ...)

## S4 method for signature 'GBAccession'
getSeq(x, ...)

## S4 method for signature 'GenBankRecord'
cdsBy(x, by = c("tx", "gene"))

## S4 method for signature 'GenBankRecord'
exonsBy(x, by = c("tx", "gene"))

## S4 method for signature 'GenBankRecord'
isCircular(x)

## S4 method for signature 'GenBankRecord'
seqinfo(x)
```

Arguments

x	The object containing the annotations
...	unused.
by	character. Factor to group the resulting GRanges by.

Value

The expected types, GenomicRanges for most functions, a DNAStrimgSet for getSeq

Examples

```
gb = readGenBank(system.file("sample.gbk", package="genbankr"))
cds(gb)
exons(gb)
genes(gb)
```

GBAccession-class *GBAccession ID class*

Description

A class representing the (versioned) GenBank accession

Usage

```
GBAccession(id)
```

Arguments

id	A versioned GenBank Accession id

Value

a GBAccession object.

Examples

```
id = GBAccession("U49845.1")
## Not run: gb = readGenBank(id)
```

gbk-specific-api *genbankr specific api*

Description

Accessor functions specific to genbankr objects.

Usage

```
accession(x, ...)

## S4 method for signature 'GenBankRecord'
accession(x)

definition(x, ...)

## S4 method for signature 'GenBankRecord'
definition(x)

locus(x, ...)

## S4 method for signature 'GenBankRecord'
locus(x)

vers(x, ...)

## S4 method for signature 'GenBankRecord'
vers(x)

sources(x, ...)

## S4 method for signature 'GenBankRecord'
sources(x)
```

Arguments

x	A genbank annotation object
...	unused.

Value

Character vectors for accession and vers

Examples

```
gb = readGenBank(system.file("sample.gbk", package="genbankr"))
accession(gb)
vers(gb)
```

GenBankFile-class *GenBank File*

Description

A resource class for use within the rtracklayer framework

Create a GenBankFile object.

Usage

```
GenBankFile(fil)
```

Arguments

fil	character. Path to the genbank file

Value

A GenBankFile object

Examples

```
fil = GenBankFile(system.file("sample.gbk", package="genbankr"))
gb = import(fil)
```

GenBankRecord-class *GenBank data objects*

Description

These objects represent GenBank annotations

Examples

```
gb = readGenBank(system.file("sample.gbk", package="genbankr"))
gb
```

`import,GenBankFile,ANY,ANY-method` *Import genbank file*

Description

Import a genbank file using the rtracklayer API.

Usage

```
## S4 method for signature 'GenBankFile,ANY,ANY'
import(con, format, text, ...)
```

Arguments

`con`	See import docs.
`format`	See import docs.
`text`	See import docs.
`...`	Arguments passed to `readGenBank`

Value

A `GenBankRecord` object.

`intergenic` *Extract intergenic regions from processed GenBank annotations*

Description

Extract the intergenic regions from a set of GenBank annotations.

Usage

```
## S4 method for signature 'GenBankRecord'
intergenic(x)
```

Arguments

`x`	A GenBankRecord object

Value

A GRanges for the intergenic regions, defined as regions not overlapping any genes defined in the annotations on either strand.

Examples

```
gb = readGenBank(system.file("sample.gbk", package="genbankr"))
intergenic(gb)
```

makeTxDbFromGenBank *Create a TxDb from a GenBank record*

Description

Create a TxDb object from a GenBankRecord.

Usage

```
makeTxDbFromGenBank(gbr, reassign.ids = FALSE)

## S4 method for signature 'GenBankRecord'
makeTxDbFromGenBank(gbr, reassign.ids = FALSE)

## S4 method for signature 'GBAccession'
makeTxDbFromGenBank(gbr, reassign.ids = FALSE)
```

Arguments

gbr	A GenBankRecord or GBAccession object
reassign.ids	logical. Passed down to makeTxDb

Value

A TxDb object

Examples

```
thing = readGenBank(system.file("unitTests/compjoin.gbk", package="genbankr"))
tx = makeTxDbFromGenBank(thing)
```

make_gbrecord *GenBank object constructors*

Description

Constructors for GenBankRecord objects.

Usage

```
make_gbrecord(rawgbk, verbose = FALSE)
```

Arguments

rawgbk	list. The output of parseGenBank
verbose	logical. Should informative messages be shown

Value

A GenBankRecord object

Examples

```
prsed = parseGenBank(system.file("sample.gbk", package="genbankr"))
gb = make_gbrecord(prsed)
```

---

otherFeatures *Retrieve 'other' features*

Description

Retrieve the other features (not covered by a different accessor) from the set of annotations

Usage

```
otherFeatures(x)

## S4 method for signature 'GenBankRecord'
otherFeatures(x)
```

Arguments

x	a GenBankRecord object

Value

A GRanges containing the features which don't fall into another category (ie not gene, exon, transcript, cds, or variant) annotated in the source file

Examples

```
gb = readGenBank(system.file("sample.gbk", package="genbankr"))
otherFeatures(gb)
```

---

parseGenBank *Parse raw genbank file content*

Description

Parse genbank content and return a low-level list object containing each component of the file.

Usage

```
parseGenBank(file, text = readLines(file), partial = NA, verbose = FALSE,
  ret.anno = TRUE, ret.seq = TRUE)
```

Arguments

file	character. The file to be parsed. Ignored if text is specified
text	character. The text to be parsed.
partial	logical. If TRUE, features with non-exact boundaries will be included. Otherwise, non-exact features are excluded, with a warning if partial is NA (the default).
verbose	logical. Should informative messages be printed to the console as the file is being processed.
ret.anno	logical. Should the annotations in the GenBank file be parsed and included in the returned object. (Defaults to TRUE)
ret.seq	logical. Should the origin sequence (if present) in the GenBank file be included in the returned object. (Defaults to TRUE)

Value

if ret.anno is TRUE, a list containing the parsed contents of the file, suitable for passing to make_gbrecord. If ret.anno is FALSE, a DNAStringSet object containing the origin sequence.

Note

This is a low level function not intended for common end-user use. In nearly all cases, end-users (and most developers) should call readGenBank or create a GenBankFile object and call import instead.

Examples

```
prsd = parseGenBank(system.file("sample.gbk", package="genbankr"))
```

---

readGenBank *Read a GenBank File*

Description

Read a GenBank file from a local file, or retrieve and read one based on an accession number. See Details for exact behavior.

Usage

```
readGenBank(file, text = readLines(file), partial = NA, ret.seq = TRUE,
  verbose = FALSE)
```

Arguments

file	character or GBAccession. The path to the file, or a GBAccession object containing Nuccore versioned accession numbers. Ignored if text is specified.
text	character. The text of the file. Defaults to text within file
partial	logical. If TRUE, features with non-exact boundaries will be included. Otherwise, non-exact features are excluded, with a warning if partial is NA (the default).

`ret.seq` logical. Should an object containing the raw ORIGIN sequence be created and returned. Defaults to `TRUE`. If `FALSE`, the sequence slot is set to `NULL`. See NOTE.

`verbose` logical. Should informative messages be printed to the console as the file is processed. Defaults to `FALSE`.

Details

If a a `GBAccession` object is passed to `file`, the rentrez package is used to attempt to fetch full GenBank records for all ids in the

Often times, GenBank files don't contain exhaustive annotations. For example, files including CDS annotations often do not have separate transcript features. Furthermore, chromosomes are not always named, particularly in organisms that have only one. The details of how genbankr handles such cases are as follows:

In files where CDSs are annotated but individual exons are not, 'approximate exons' are defined as the individual contiguous elements within each CDS. Currently, no mixing of approximate and explicitly annotated exons is performed, even in cases where, e.g., exons are not annotated for some genes with CDS annotations.

In files where transcripts are not present, 'approximate transcripts' defined by the ranges spanned by groups of exons are used. Currently, we do not support generating approximate transcripts from CDSs in files that contain actual transcript annotations, even if those annotations do not cover all genes with CDS/exon annotations.

Features (gene, cds, variant, etc) are assumed to be contained within the most recent previous source feature (chromosome/physical piece of DNA). Chromosome name for source features (seqnames in the resulting `GRanges/VRanges` is determined as follows:

1. The 'chromosome' attribute, as is (e.g., "chr1");
2. the 'strain' attribute, combined with auto-generated count (e.g., "VR1814:1");
3. the 'organism' attribute, combined with auto-generated count (e.g. "Human herpesvirus 5:1".

In files where no origin sequence is present, importing varation features is not currently supported, as there is no easy/ self-contained way of determining the reference in those situations and the features themselves list only alt. If variation features are present in a file without origin sequence, those features are ignored with a warning.

Currently some information about from the header of a GenBank file, primarily reference and author based information, is not captured and returned. Please contact the maintainer if you have a direct use-case for this type of information.

Value

A `GenBankRecord` object containing (most, see detaisl) of the information within `file/text` Or a list of `GenBankRecord` objects in cases where a `GBAccession` vector with more than one ID in it is passed to `file`

Note

We have endeavored to make this parser as effcient as easily possible. On our local machines, a 19MB genbank file takes 2-3 minutes to be parsed. That said, this function is not tested and likely not suitable for processing extremely large genbank files.

The origin sequence is always parsed when calling readGenBank, because it is necessary to generate a VRanges from variant features. So currently `ret.seq=FALSE` will not reduce parsing time, or maximum memory usage, though it will reduce memory usage by the final GenBankRecord object.

The lower-level parseGenBank does skil parsing the sequence at all via `ret.seq=FALSE`, but variant annotations will be excluded if make_gbrecord is called on the resulting parsed list.

Examples

```
gb = readGenBank(system.file("sample.gbk", package="genbankr"))
```

`variants` *Retrieve variantion features*

Description

Extract the annotated variants from a GenBankRecord object

Usage

```
variants(x)

## S4 method for signature 'GenBankRecord'
variants(x)
```

Arguments

x	a GenBankRecord object

Value

A VRanges containing the variations annotated in the source file

Examples

```
gb = readGenBank(system.file("sample.gbk", package="genbankr"))
variants(gb)
```

Index