Package 'GenomeInfoDb'

October 7, 2014

Type Package

Title Utilities for manipulating chromosome and other 'seqname' identifiers

Version 1.0.2

Author Sonali Arora, Martin Morgan, Marc Carlson, H. Pages

Maintainer Bioconductor Package Maintainer <maintainer@bioconductor.org>

Depends R (>= 3.1)

Suggests
BiocGenerics, IRanges, BSgenome,BSgenome.Scerevisiae.UCSC.sacCer2,BSgenome.Hsapiens.NCBI.GRCh38,TxDb.Dm
Style, knitr

Description The Seqnames package contains data and functions that
define and allow translation between different chromosome sequence naming conventions (e.g., ``chr1" versus ``1"),including a function that attempts to place sequence names in their natural, rather than lexicographic, order.

License Artistic-2.0

biocViews Genetics, DataRepresentation, Annotation

Collate assembly-utils.R fetchExtendedChromInfoFromUCSC.R seqlevels.R
seqlevels-helpers.R rankSeqlevels.R

VignetteBuilder knitr

R topics documented:

fetchExtendedChromInfoFromUCSC *Fetching chromosomes info for some of the UCSC genomes*

Description

Fetch the chromosomes info for some of the UCSC genomes. Only supports hg38, hg19, mm10, dm3, and sacCer3 at the moment.

Usage

```
fetchExtendedChromInfoFromUCSC(genome,
        goldenPath_url="http://hgdownload.cse.ucsc.edu/goldenPath")
```

Arguments

genome	A single string specifying the UCSC genome e.g. "sacCer3".
goldenPath_url	A single string specifying the URL to the UCSC goldenPath location. This URL is used internally to build the full URL to the 'chromInfo' MySQL dump containing chromosomes information for genome. See Details section below.

Details

Chromosomes information (e.g. names and lengths) for any UCSC genome is stored in the UCSC database in the 'chromInfo' table and is normally available as a MySQL dump at:

```
goldenPath_url/<genome>/database/chromInfo.txt.gz
```

fetchExtendedChromInfoFromUCSC downloads and imports that table into a data frame, and keeps only the UCSC_seqlevels and UCSC_seqlengths columns (after renaming them). Then it lookups the assembly report at NCBI for that genome corresponding (e.g. GRCh38 assembly for hg38), extracts the seqlevels and GenBank accession numbers from the report, matches them to each UCSC seqlevels (using some heuristic), and adds them to the returned data frame.

Value

A data frame with one row per seqlevel in the UCSC genome, and with the following columns:

- UCSC_seqlevels: Character vector with no NAs. This is the chrom field of the UCSC 'chromInfo' table for the genome. See Details section above.
- UCSC_seqlengths: Integer vector with no NAs. This is the size field of the UCSC 'chromInfo' table for the genome. See Details section above.
- NCBI_seqlevels: Character vector. This information is obtained from the NCBI assembly report for the genome. Will contain NAs for UCSC seqlevels with no corresponding NCBI seqlevels (e.g. for chrM in hg18 or chrUextra in dm3), in which case fetchExtendedChromInfoFromUCSC emits a warning.

- GenBank_accns: Character vector. This information is obtained from the NCBI assembly report for the genome. Can contain NAs but no warning is emitted in that case.

Note that the rows are not sorted in any particular order.

Note

Only supports the hg38, hg19, mm10, dm3, and sacCer3 genomes at the moment. More will come...

Author(s)

H. Pages

See Also

- The seqlevels getter and setter in the **GenomicRanges** package.
- The seqlevelsStyle getter and setter.
- The getBSgenome utility in the **BSgenome** package for searching the installed BSgenome data packages.

Examples

```
## ---------------------------------------------------------------------
## A. BASIC EXAMPLE
## ---------------------------------------------------------------------
chrominfo <- fetchExtendedChromInfoFromUCSC("sacCer3")
chrominfo

## ---------------------------------------------------------------------
## B. USING fetchExtendedChromInfoFromUCSC() TO PUT UCSC SEQLEVELS ON
##    THE GRCh38 GENOME
## ---------------------------------------------------------------------

## Load the BSgenome.Hsapiens.NCBI.GRCh38 package:
library(BSgenome)
genome <- getBSgenome("GRCh38")  # this loads the
                                 # BSgenome.Hsapiens.NCBI.GRCh38 package

## A quick look at the GRCh38 seqlevels:
length(seqlevels(genome))
head(seqlevels(genome), n=30)

## Fetch the extended chromosomes info for the hg38 genome:
hg38_chrominfo <- fetchExtendedChromInfoFromUCSC("hg38")
dim(hg38_chrominfo)
head(hg38_chrominfo, n=30)

## 2 sanity checks:
##   1. Check the NCBI seqlevels:
stopifnot(setequal(hg38_chrominfo$NCBI_seqlevels, seqlevels(genome)))
##   2. Check that the sequence lengths in hg38_chrominfo (which are
##      coming from the same chromInfo table as the UCSC seqlevels)
```

```
##      are the same as in genome:
stopifnot(
  identical(hg38_chrominfo$UCSC_seqlengths,
            unname(seqlengths(genome)[hg38_chrominfo$NCBI_seqlevels]))
)

## Extract the hg38 seqlevels and put the GRCh38 seqlevels on it as
## the names:
hg38_seqlevels <- setNames(hg38_chrominfo$UCSC_seqlevels,
                           hg38_chrominfo$NCBI_seqlevels)

## Set the hg38 seqlevels on genome:
seqlevels(genome) <- hg38_seqlevels[seqlevels(genome)]
head(seqlevels(genome), n=30)
```

GenomeInfoDb-helpers *List the supported seqname styles for all supported organisms*

Description

List the supported seqname styles for all supported organisms

Usage

```
genomeStyles(species)
extractSeqlevels(species, style)
extractSeqlevelsByGroup(species, style, group)
orderSeqlevels(seqnames, X.is.sexchrom = NA)
mapSeqlevels(seqnames, style, best.only=TRUE, drop=TRUE)
seqlevelsInGroup(seqnames, group, species, style)
seqlevelsStyle(x)
seqlevelsStyle(x) <- value
```

Arguments

species	The genus and species of the organism in question separated by a single space. Don't forget to capitalize the genus.
style	a character vector with a single element to specify the style.
group	Group can be 'auto' for autosomes, 'sex' for sex chromosomes/allosomes, 'circular' for circular chromosomes. The default is 'all' which returns all the chromosomes.
best.only	if TRUE (the default), then only the "best" sequence renaming maps (i.e. the rows with less NAs) are returned.
drop	if TRUE (the default), then a vector is returned instead of a matrix when the matrix has only 1 row.

seqnames	a character vector containing the labels attached to the chromosomes in a given genome for a given style. For example : For *Homo sapiens*, NCBI style - they are "1","2","3",...,"X","Y","MT"
X.is.sexchrom	A logical indicating whether X refers to the sexual chromosome or to chromosome with Roman Numeral X. If NA, makeSeqnameIds does its best to "guess".
x	The object from/on which to get/set the sequence information.
value	A single character string that sets the seqnameStyle for x.

Details

genomeStyles: Different organizations have different naming conventions for how they name the biologically defined sequence elements (usually chromosomes) for each organism they support. The Seqnames package contains a database that defines these different conventions.

genomeStyles() returns the list of all supported seqname mappings, one per supported organism. Each mapping is represented as a data frame with 1 column per seqname style and 1 row per chromosome name (not all chromosomes of a given organism necessarily belong to the mapping).

genomeStyles(species) returns a data.frame only for the given organism with all its supported seqname mappings.

extractSeqlevels: Returns a character vector of the seqnames for a single style and species.

extractSeqlevelsByGroup: Returns a character vector of the seqnames for a single style and species by group. Group can be 'auto' for autosomes, 'sex' for sex chromosomes/ allosomes, 'circular' for circular chromosomes. The default is 'all' which returns all the chromosomes.

orderSeqlevels: Returns an integer vector while attempting to provide the "natural" order of seqnames, e.g.,chr1, chr2, chr3, ...

mapSeqlevels: Returns a matrix with 1 column per supplied sequence name and 1 row per sequence renaming map compatible with the specified style. If best.only is TRUE (the default), only the "best" renaming maps (i.e. the rows with less NAs) are returned.

seqlevelsInGroup: It takes a character vector along with a group and optional style and species.If group is not specified , it returns "all" or standard/top level seqnames. Returns a character vector of seqnames after subsetting for the group specified by the user. See examples for more details.

seqlevelsStyle(x): finds the seqlevelsStyle for a given character vector.

Value

For extractSeqlevels , extractSeqlevelsByGroup and seqlevelsInGroup returns a character vector of seqlevels for given supported species and group.

For mapSeqlevels returns a matrix with 1 column per supplied sequence name and 1 row per sequence renaming map compatible with the specified style

For seqlevelsStyle returns a single character string containing the style of the seqlevels supplied. Note that this information is not stored in x but inferred by looking up a seqlevel style database stored inside GenomeInfoDb.

For genomeStyle : If species is specified returns a data.frame containg the seqlevel style and its mapping for a given organism. If species is not specified, a list is returned with one list per species containing the seqlevel style with the corresponding mappings.

For orderSeqlevels returns an integer vector with indices of seqlevels in their natural order.

Author(s)

Sonali Arora <sarora@fhcrc.org>, Martin Morgan , Marc Carlson, H. Pages

Examples

```
names(genomeStyles())
genomeStyles("Homo_sapiens")
"UCSC" %in% names(genomeStyles("Homo_sapiens"))

## List the supported seqname style for the given species and the given
## style
extractSeqlevels(species="Drosophila_melanogaster" , style="Ensembl")

## List all sex chromosomes for Homo sapiens using style UCSC
## 3 groups are supported: auto for autosomes, sex for allosomes
## and circular for circular chromosomes
extractSeqlevelsByGroup(species="Homo_sapiens", style="UCSC", group="sex")

## find whether the seqnames belong to a given group
newchr <- paste0("chr",c(1:22,"X","Y","M","1_gl000192_random","4_ctg9"))
seqlevelsInGroup(newchr, group="sex")

newchr <- as.character(c(1:22,"X","Y","MT"))
seqlevelsInGroup(newchr, group="all","Homo_sapiens","NCBI")

## find the seqname Style for a given character vector
seqlevelsStyle(paste0("chr",c(1:30)))

## order a character vector of seqnames
seqnames <- c("chr1","chr9", "chr2", "chr3", "chr10")
seqnames[orderSeqlevels(seqnames)]

## if we have a vector conatining seqnames and we want to verify the
## species and style for them , we can use:
all(seqnames %in% extractSeqlevels("Homo_sapiens", "UCSC"))

## find mapped seqlevelsStyles for exsiting seqnames
mapSeqlevels(c("chrII", "chrIII", "chrM"), "NCBI")
mapSeqlevels(c("chrII", "chrIII", "chrM"), "Ensembl")
```

`rankSeqlevels` *Assign sequence IDs to sequence names*

Description

`rankSeqlevels` assigns a unique ID to each unique sequence name in the input vector. The returned IDs span 1:N where N is the number of unique sequence names in the input vector.

Usage

```
rankSeqlevels(seqnames, X.is.sexchrom=NA)
```

Arguments

seqnames	A character vector or factor containing sequence names.
X.is.sexchrom	A logical indicating whether X refers to the sexual chromosome or to chromosome with Roman Numeral X. If NA, rankSeqlevels does its best to "guess".

Value

An integer vector of the same length as seqnames. The values in the vector span 1:N where N is the number of unique sequence names in the input vector.

Author(s)

H. Pages

See Also

- sortSeqlevels for sorting the sequence levels of an object in "natural" order.

Examples

```
library(BSgenome.Scerevisiae.UCSC.sacCer2)
rankSeqlevels(seqnames(Scerevisiae))
rankSeqlevels(seqnames(Scerevisiae)[c(1:5,5:1)])

newchr <- paste0("chr",c(1:3,6:15,4:5,16:22))
newchr
orderSeqlevels(newchr)
rankSeqlevels(newchr)
```

Index