# An Introduction to *GenomeInfoDb*

Martin Morgan, Herve Pages, Marc Carlson, Sonali Arora

Modified: 17 January, 2014. Compiled: April 15, 2014

## Contents



## 1 Introduction

The *GenomeInfoDb* provides an interface to access seqlevelsStyles (such as UCSC, NCBI, Ensembl) and their supported mappings for organisms. For instance, for Homo sapiens, seqlevelsStyle "UCSC" maps to "chr1", "chr2", ..., "chrX","chrY". The section below introduces these functions with examples.

## 2 Functionality for all exsiting organisms

### 2.1 genomeStyles

The `genomeStyles` lists out for each organism, the seqlevelsStyles and their mappings.

```
seqmap <- genomeStyles()
head(seqmap, n = 2)
```

```
## $Arabidopsis_thaliana
##   circular  auto   sex NCBI TAIR10
## 1    FALSE  TRUE FALSE    1      1
## 2    FALSE  TRUE FALSE    2      2
## 3    FALSE  TRUE FALSE    3      3
## 4    FALSE  TRUE FALSE    4      4
```

```
## 5    FALSE  TRUE FALSE    5      5
## 6     TRUE FALSE FALSE   MT     Mt
## 7    FALSE FALSE  TRUE Pltd     Pt
##
## $Caenorhabditis_elegans
##   circular  auto   sex NCBI   UCSC Ensembl
## 1    FALSE  TRUE FALSE    I   chrI       I
## 2    FALSE  TRUE FALSE   II  chrII      II
## 3    FALSE  TRUE FALSE  III chrIII     III
## 4    FALSE  TRUE FALSE   IV  chrIV      IV
## 5    FALSE  TRUE FALSE    V   chrV       V
## 6    FALSE FALSE  TRUE    X   chrX       X
## 7     TRUE  TRUE FALSE   MT   chrM   MtDNA
```

Oragnism's supported by GenomeInfoDb can be found by :

```
names(genomeStyles())
```

```
## [1] "Arabidopsis_thaliana"     "Caenorhabditis_elegans"   "Cyanidioschyzon_merolae"
## [4] "Drosophila_melanogaster"  "Homo_sapiens"             "Oryza_sativa"
## [7] "Populus_trichocarpa"      "Saccharomyces_cerevisiae" "Zea_mays"
```

If one knows the organism one is interested in, then we can directly access the information for the given organism along. Each function accepts an argument called species which as "genus species", the default is "Homo sapiens". In the following example we list out only the first five entries returned by the code snippet.

```
head(genomeStyles("Homo_sapiens"), 5)
```

```
##   circular auto   sex NCBI UCSC
## 1    FALSE TRUE FALSE    1 chr1
## 2    FALSE TRUE FALSE    2 chr2
## 3    FALSE TRUE FALSE    3 chr3
## 4    FALSE TRUE FALSE    4 chr4
## 5    FALSE TRUE FALSE    5 chr5
```

We can also check if a given style is supported by GenomeInfoDb for a given species. For example, if we want to know if "UCSC" mapping is supported for "Homo sapiens" we can ask :

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

```
## [1] TRUE
```

## 2.2 extractSeqlevels

We can also extract the desired seqlevelsStyle from a given organism using the `extractSeqlevels`

```
extractSeqlevels(species = "Arabidopsis_thaliana", style = "NCBI")
```

```
## [1] "1"    "2"    "3"    "4"    "5"    "MT"   "Pltd"
```

## 2.3 extractSeqlevelsByGroup

We can also extract the desired seqlevelsStyle from a given organism based on a group ( Group - 'auto' denotes autosomes, 'circular' denotes circular chromosomes and 'sex' denotes sex chromosomes; the default is all chromosomes are returned).

```
extractSeqlevelsByGroup(species = "Arabidopsis_thaliana", style = "NCBI", group = "auto")
## [1] "1" "2" "3" "4" "5"
```

## 2.4 seqlevelsStyle

We can find the seqname Style for a given character vector by using the `seqlevelsStyle`

```
seqlevelsStyle(paste0("chr", c(1:30)))
## [1] "UCSC"
seqlevelsStyle(c("2L", "2R", "X", "Xhet"))
## [1] "NCBI"
```

## 2.5 seqlevelsInGroup

We can also subset a given character vector containing seqnames using the `seqlevelsInGroup`. We currently support 3 groups: 'auto' for autosomes, 'sex' for allosomes/sex chromosomes and circular for 'circular' chromosomes. The user can also prvoide the style and species they are working with. In the following examples, we extract the sex, auto and circular chromosomes for Homo sapiens :

```
newchr <- paste0("chr", c(1:22, "X", "Y", "M", "1_gl000192_random", "4_ctg9_hap1"))
seqlevelsInGroup(newchr, group = "sex")
## [1] "chrX" "chrY"
seqlevelsInGroup(newchr, group = "auto")
##  [1] "chr1"  "chr2"  "chr3"  "chr4"  "chr5"  "chr6"  "chr7"  "chr8"  "chr9"  "chr10"
## [11] "chr11" "chr12" "chr13" "chr14" "chr15" "chr16" "chr17" "chr18" "chr19" "chr20"
## [21] "chr21" "chr22"
seqlevelsInGroup(newchr, group = "circular")
## [1] "chrM"
seqlevelsInGroup(newchr, group = "sex", "Homo_sapiens", "UCSC")
## [1] "chrX" "chrY"
```

if we have a vector conatining seqnames and we want to verify the species and style for them , we can use:

```
seqnames <- c("chr1", "chr9", "chr2", "chr3", "chr10")
all(seqnames %in% extractSeqlevels("Homo_sapiens", "UCSC"))
## [1] TRUE
```

## 2.6 orderSeqlevels

The `orderSeqlevels` can return the order of a given character vector which contains seqnames.In the following example, we show how you can find the order for a given seqnames character vector.

```
seqnames <- c("chr1", "chr9", "chr2", "chr3", "chr10")
orderSeqlevels(seqnames)
## [1] 1 3 4 2 5
```

### 2.7 rankSeqlevels

The `rankSeqlevels` can return the rank of a given character vector which contains seqnames.In the following example, we show how you can find the rank for a given seqnames character vector.

```
seqnames <- c("chr1", "chr9", "chr2", "chr3", "chr10")
rankSeqlevels(seqnames)
```

```
## [1] 1 4 2 3 5
```

### 2.8 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.

```
mapSeqlevels(c("chrII", "chrIII", "chrM"), "NCBI")
```

```
##  chrII chrIII   chrM
##   "II"  "III"   "MT"
```

## 3 Examples

### 3.1 converting seqlevel styles (eg:UCSC to NCBI)

A quick example using Drosophila Melanogaster. The txdb object contains seqlevels in UCSC style, we want to convert them to NCBI

```
txdb <- TxDb.Dmelanogaster.UCSC.dm3.ensGene
seqlevels(txdb)
```

```
##  [1] "chr2L"     "chr2R"     "chr3L"     "chr3R"     "chr4"      "chrX"      "chrU"
##  [8] "chrM"      "chr2LHet"  "chr2RHet"  "chr3LHet"  "chr3RHet"  "chrXHet"   "chrYHet"
## [15] "chrUextra"
```

```
genomeStyles("Drosophila melanogaster")
```

```
##    circular   sex  auto  NCBI      UCSC                    Ensembl
## 1     FALSE FALSE  TRUE    2L     chr2L                         2L
## 2     FALSE FALSE  TRUE    2R     chr2R                         2R
## 3     FALSE FALSE  TRUE    3L     chr3L                         3L
## 4     FALSE FALSE  TRUE    3R     chr3R                         3R
## 5     FALSE FALSE  TRUE     4      chr4                          4
## 6     FALSE  TRUE FALSE     X      chrX                          X
## 7      TRUE FALSE FALSE    MT      chrM dmel_mitochondrion_genome
## 8     FALSE FALSE FALSE 2LHet  chr2LHet                       2LHet
## 9     FALSE FALSE FALSE 2Rhet  chr2RHet                       2RHet
## 10    FALSE FALSE FALSE 3LHet  chr3LHet                       3LHet
## 11    FALSE FALSE FALSE 3RHet  chr3RHet                       3RHet
## 12    FALSE FALSE FALSE  Xhet   chrXHet                        XHet
## 13    FALSE FALSE FALSE  Yhet   chrYHet                        YHet
## 14    FALSE FALSE FALSE    Un      chrU                           U
## 15    FALSE FALSE FALSE  <NA> chrUextra                      Uextra
```

```
mapSeqlevels(seqlevels(txdb), "NCBI")
```

```
##      chr2L      chr2R      chr3L      chr3R       chr4       chrX       chrU       chrM   chr2LHet
##       "2L"       "2R"       "3L"       "3R"        "4"        "X"       "Un"       "MT"    "2LHet"
##   chr2RHet   chr3LHet   chr3RHet    chrXHet    chrYHet chrUextra
##    "2Rhet"    "3LHet"    "3RHet"     "Xhet"     "Yhet"         NA
```

## 3.2 converting styles and removing unwanted seqlevels

Suppose we read in a Bam file or a BED file and the resulting GRanges have a lot of seqlevels which are not required by your analysis or you want to rename the seqlevels from the current style to your own style (eg:USCS to NCBI), we can use the functionality provided by GenomeInfoDb to do that.

Let us say that we have extracted the seqlevels of the Seqinfo object(say GRanges from a BED file) in a variable called "sequence".

```
sequence <- seqlevels(x)

## sequence is in UCSC format and we want NCBI style
newStyle <- mapSeqlevels(sequence, "NCBI")
newStyle <- newStyle[complete.cases(newStyle)]  # removing NA cases.

## rename the seqlevels
x <- renameSeqlevels(x, newStyle)

## keep only the seqlevels you want (say autosomes)
auto <- extractSeqlevelsByGroup(species = "Homo sapiens", style = "NCBI", group = "auto")
x <- keepSeqlevels(x, auto)
```