Package 'rawrr'

October 18, 2022

Type Package

Title Direct Access to Orbitrap Data and Beyond

Version 1.4.0

Depends R (>= 4.1)

Imports grDevices, graphics, stats, utils

Suggests BiocStyle (>= 2.5), ExperimentHub, knitr, protViz (>= 0.7),
rmarkdown, tartare (>= 1.5), testthat

Description This package wraps the functionality
of the RawFileReader .NET assembly. Within the R environment,
spectra and chromatograms are represented by S3 objects
(Kockmann T. et al. (2020) <doi:10.1101/2020.10.30.362533>).
The package provides basic functions to download and install
the required third-party libraries.
The package is developed, tested, and used at the Functional
Genomics Center Zurich, Switzerland <https://fgcz.ch>.

License GPL-3

SystemRequirements mono-runtime 4.x or higher (including System.Data
library) on Linux/macOS, .Net Framework (>= 4.5.1) on Microsoft
Windows.

URL https://github.com/fgcz/rawrr/

BugReports https://github.com/fgcz/rawrr/issues

Encoding UTF-8

NeedsCompilation no

RoxygenNote 7.1.2

biocViews MassSpectrometry, Proteomics, Metabolomics

VignetteBuilder knitr

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

git_branch RELEASE_3_15

git_last_commit 3f84950

git_last_commit_date 2022-04-26

Author Christian Panse [aut, cre] (<<https://orcid.org/0000-0003-1975-3064>>),
	Tobias Kockmann [aut] (<<https://orcid.org/0000-0002-1847-885X>>)

Maintainer Christian Panse <cp@fgcz.ethz.ch>

R **topics documented:**

---

auc.rawrrChromatogram *Example snippet for deriving an AUC*

Description

Example snippet for deriving an AUC

Usage

```
auc.rawrrChromatogram(x)
```

Arguments

x   an rawrrChromatogram

Value

A numeric value.

---

basePeak *Base peak of a spectrum*

Description

Base peak of a spectrum

Usage

```
basePeak(x)
```

Arguments

x   A rawrrSpectrum object

Value

A double vector of length two. The first component is the base peak position (m/z). The second component is the base peak intensity.

Examples

```
S <- readSpectrum(rawfile = sampleFilePath(), 1)
basePeak(S[[1]])
```

---

buildRawrrExe   *Build* rawrr.exe *console application.*

Description

builds rawrr.exe file from C# source code requiring xbuild or msbuild tools. The console application rawrr.exe is used by the package's reader functions through a system2 call.

Usage

```
buildRawrrExe()
```

Details

The rawrr package implementation consists of two language layers, the top R layer and the hidden C# layer. Specifically, R functions requesting access to data stored in binary raw files invoke compiled C# wrapper methods using a system2 call. Calling a wrapper method typically results in the execution of methods defined in the RawFileReader dynamic link library provided by Thermo Fisher Scientific. Our precompiled wrapper methods are bundled in the rawrr.exe executable file (.NET assembly) and shipped with the released R package. Running rawrr.exe requires the https://www.mono-project.com/ environment on non-Microsoft operating systems. Mono is a cross platform, open source .NET framework. On Microsoft Windows the Microsoft .NET framework is typically already installed and sufficient. Our package also contains the C# source code rawrr.cs. In order to return extracted data back to the R layer we use file I/O. More specifically, the extracted information is written to a temporary location on the harddrive, read back into memory and parsed into R objects.

Value

the return value of the system2 command.

Author(s)

Tobias Kockmann, Christian Panse <cp@fgcz.ethz.ch>, 2021

References

- https://www.mono-project.com/docs/advanced/assemblies-and-the-gac/
- https://planetorbitrap.com/rawfilereader
- https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/compiler-options/advanced
- doi:10.1021/acs.jproteome.0c00866

See Also

installRawrrExe and installRawFileReaderDLLs

---

dependentScan   *Retrieve dependent scan(s) of a scan listed in scan index*

Description

Retrieve dependent scan(s) of a scan listed in scan index

Usage

```
dependentScan(x, scanNumber)
```

Arguments

x	A scan index returned by `readIndex`.
scanNumber	The scan number that should be inspected for dependent scans.

Value

The scan number of the dependent scan(s).

Examples

```
Idx <- readIndex(rawfile = sampleFilePath())
dependentScan(Idx, scanNumber = 1)
```

---

faimsVoltageOn   *Is FAIMS Voltage on?*

Description

Is FAIMS Voltage on?

Usage

```
faimsVoltageOn(x)
```

Arguments

x	A rawrrSpectrum object

Value

A boolean

Examples

```
S <- readSpectrum(rawfile = sampleFilePath(), 1:10)
try(faimsVoltageOn(S[[1]]))
```

installRawFileReaderDLLs

> *Download and install the New RawFileReader from Thermo Fisher Scientific .Net assemblies i*

Description

Download and install the New RawFileReader from Thermo Fisher Scientific .Net assemblies in the directory provided by rawrrAssemblyPath().

Usage

```
installRawFileReaderDLLs(sourceUrl = .thermofisherlsmsUrl(), ...)
```

Arguments

sourceUrl	url of New RawFileReader from Thermo Fisher Scientific assemblies.
...	other parameter for download.file

Details

The console application assembly rawrr.exe requires three assemplies:

- ThermoFisher.CommonCore.Data.dll,
- ThermoFisher.CommonCore.MassPrecisionEstimator.dll, and
- ThermoFisher.CommonCore.RawFileReader.dll

.

The rawrr.exe assembly can be built from C# source code by using the msbuild tool shipped by the https://www.mono-project.com or by Microsoft's .NET SDK https://dotnet.microsoft.com on Linux, Microsoft, and macOS.

If no build tool and C# compiler (csc or msc) are available or the build process fails, you can download rawrr.exe assembly from the authors' site.

Value

An (invisible) vector of integer code, 0 for success and non-zero for failure. For the "wget" and "curl" methods this is the status code returned by the external program.

Author(s)

Christian Panse <cp@fgcz.ethz.ch>, 2021

References

- https://www.mono-project.com/docs/advanced/assemblies-and-the-gac/
- https://planetorbitrap.com/rawfilereader
- doi:10.1021/acs.jproteome.0c00866

See Also

[buildRawrrExe](#) and [installRawrrExe](#)

Examples

```
# to install all assemblies

rawrr::installRawFileReaderDLLs()
rawrr::buildRawrrExe() || rawrr::installRawrrExe()
```

---

installRawrrExe   *Download and install the* rawrr.exe *console application*

Description

downloads and installs the rawrr.exe .Net assembly in the directory provided by rawrrAssemblyPath().

Usage

```
installRawrrExe(
  sourceUrl = "http://fgcz-ms.uzh.ch/~cpanse/rawrr/rawrr.1.3.7.exe",
  ...
)
```

Arguments

sourceUrl	url of rawrr.exe assembly.
...	other parameter for download.file.

Details

The console application rawrr.exe is used by the package's reader functions through a [system2](#) call.

Value

An integer code, 0 for success and non-zero for failure. For the "wget" and "curl" methods this is the status code returned by the external program.

References

[doi:10.1021/acs.jproteome.0c00866](#)

See Also

[buildRawrrExe](#)

---

is.rawrrChromatogram   *Function to check if an object is an instance of class* rawrrChromatogram

---

Description

Function to check if an object is an instance of class rawrrChromatogram

Usage

```
is.rawrrChromatogram(x)
```

Arguments

x   x any R object to be tested.

Value

TRUE or FALSE

Author(s)

Tobias Kockmann, 2020.

Examples

```
rawfile <- sampleFilePath()
C <- readChromatogram(rawfile, mass = 445.1181, tol = 10)
is.rawrrChromatogram(C[[1]])
```

---

is.rawrrSpectrum   *Function to check if an object is an instance of class* rawrrSpectrum

---

Description

Function to check if an object is an instance of class rawrrSpectrum

Usage

```
is.rawrrSpectrum(x)
```

Arguments

x   any R object to be tested.

Value

TRUE or FALSE

Examples

```
S <- rawrr::sampleFilePath() |> rawrr::readSpectrum(scan = 1:10)
rawrr::is.rawrrSpectrum(S[[1]])
```

---

is.rawrrSpectrumSet	*Function to check if an object is an instance of class* rawrrSpectrumSet

---

Description

Function to check if an object is an instance of class rawrrSpectrumSet

Usage

```
is.rawrrSpectrumSet(x)
```

Arguments

x   any R object to be tested.

Value

TRUE or FALSE

Examples

```
rawrr::sampleFilePath() |>
  rawrr::readSpectrum(scan = 1:10) |>
  rawrr::is.rawrrSpectrumSet()
```

---

makeAccessor	*Make accessor function for key value pair returned by RawFileReader*

Description

Make accessor function for key value pair returned by RawFileReader

Usage

```
makeAccessor(key, returnType = "integer")
```

Arguments

`key`	An object name found in instance of class `rawrrSpectrum`
`returnType`	The type used for casting of values

Details

This function factory creates accessor functions for class `rawrrSpectrum`.

Value

An accessor function

Author(s)

Tobias Kockmann, 2020

Examples

```
S <- readSpectrum(rawfile = sampleFilePath(), 1:10)
maxIonTime <- makeAccessor(key = "Max. Ion Time (ms):", returnType = "double")
maxIonTime(S[[1]])
```

---

massRange   *Acquisition/scan range of spectrum*

Description

Acquisition/scan range of spectrum

Usage

```
massRange(x)
```

Arguments

`x`	A rawrrSpectrum object

Value

A double vector of length two. The first component is the start m/z, the second is the stop m/z value used by the detector during data acquisition. Also referred to as scan range.

Examples

```
S <- readSpectrum(rawfile = sampleFilePath(), 1)
massRange(S[[1]])
```

---

masterScan   *Retrieve master scan of scan listed in scan index*

Description

Retrieve master scan of scan listed in scan index

Usage

```
masterScan(x, scanNumber)
```

Arguments

x   A scan index returned by `readIndex`.

scanNumber   The scan number that should be inspected for the presence of a master scan.

Value

Returns the scan number of the master scan or NA if no master scan exists.

Examples

```
rawrr::sampleFilePath() |> rawrr::readIndex() |> rawrr::masterScan(scanNumber = 1)
```

---

new_rawrrSpectrum   *Create instances of class* rawrrSpectrum

---

Description

Developer function.

Usage

```
new_rawrrSpectrum(
  scan = numeric(),
  massRange = numeric(),
  scanType = character(),
  rtinseconds = numeric(),
  centroidStream = logical(),
  mZ = numeric(),
  intensity = numeric()
)
```

Arguments

`scan`	scan number
`massRange`	Mass range covered by spectrum
`scanType`	Character string describing the scan type.
`rtinseconds`	Retention time in seconds
`centroidStream`	Logical indicating if centroided data is available
`mZ`	m/z values
`intensity`	Intensity values

Value

Object of class `rawrrSpectrum`

Author(s)

Tobias Kockmann, 2020.

`plot.rawrrChromatogram`

Plot `rawrrChromatogram` *objects*

Description

Plot `rawrrChromatogram` objects

Usage

```
## S3 method for class 'rawrrChromatogram'
plot(x, legend = TRUE, ...)
```

Arguments

`x`	A `rawrrChromatogram` object to be plotted.
`legend`	Should legend be printed?
`...`	Passes additional arguments.

Value

This function creates a plot.

Author(s)

Tobias Kockmann, 2020.

Examples

```
rawfile <- sampleFilePath()
C <- readChromatogram(rawfile, mass = 445.1181, tol = 10)
plot(C[[1]])
```

---

plot.rawrrChromatogramSet

Plot rawrrChromatogramSet *objects*

Description

Plot rawrrChromatogramSet objects

Usage

```
## S3 method for class 'rawrrChromatogramSet'
plot(x, diagnostic = FALSE, ...)
```

Arguments

x	A rawrrChromatogramSet object to be plotted.
diagnostic	Show diagnostic legend?
...	Passes additional arguments.

Value

This function creates a plot.

Author(s)

Tobias Kockmann, 2020.

---

plot.rawrrSpectrum   *Basic plotting function for instances of* rawrrSpectrum

---

Description

Plot method for objects of class rawrrSpectrum.

Usage

```
## S3 method for class 'rawrrSpectrum'
plot(
  x,
  relative = TRUE,
  centroid = FALSE,
  SN = FALSE,
  legend = TRUE,
  diagnostic = FALSE,
  ...
)
```

Arguments

x	an object of class `rawrrSpectrum`.
relative	If set to `TRUE` enforces plotting of relative intensities rather than absolute.
centroid	Should centroided data be used for plotting?
SN	Should Signal/Noise be used for plotting?
legend	Should legend be printed?
diagnostic	Should this option be applied? The default is `FALSE`.
...	function passes arbitrary additional arguments.

Details

`plot.rawrrSpectrum` is a low level function that calls `base::plot` for plotting `rawrrSpectrum` objects. It passes all additional arguments to `plot()`

Is usually called by method dispatch.

Value

This function creates a plot.

Author(s)

Tobias Kockmann, 2020.

---

print.rawrrSpectrum   *Print method imitate the look and feel of Thermo Fisher Scientific FreeStyle's output*

Description

Print method imitate the look and feel of Thermo Fisher Scientific FreeStyle's output

Usage

```
## S3 method for class 'rawrrSpectrum'
print(x, ...)
```

Arguments

x	an `rawrrSpectrum` object.
...	Arguments to be passed to methods.

Value

This function creates a print message.

Author(s)

Christian Panse and Tobias Kockmann, 2020.

---

rawrrAssemblyPath	*Derives the path where all .NET assemblies are stored.*

Description

Derives the path where all .NET assemblies are stored.

Usage

```
rawrrAssemblyPath()
```

Value

path

See Also

`installRawFileReaderDLLs` and `installRawrrExe`

Examples

```
rawrrAssemblyPath()
```

---

rawrrSpectrum   *Create* rawrrSpectrum *objects*

Description

High-level constructor for instances of class `rawrrSpectrum`, also named helper function. Currently, mainly to support testing and for demonstration.

Usage

```
rawrrSpectrum(sim = "TESTPEPTIDE")
```

Arguments

sim   Either `example_1` or `TESTPEPTIDE`

Value

Function returns a validated `rawrrSpectrum` object

Author(s)

Tobias Kockmann, 2020.

Examples

```
plot(rawrrSpectrum(sim = "TESTPEPTIDE"))
rawrrSpectrum(sim = "example_1")
```

---

readChromatogram   *Extracts chromatographic data from a raw file.*

Description

Extracts chromatographic data from a raw file.

Usage

```
readChromatogram(rawfile, mass = NULL, tol = 10, filter = "ms", type = "xic")
```

Arguments

`rawfile`	the file name.
`mass`	a vector of mass values iff `type = 'xic'`.
`tol`	mass tolerance in ppm iff `type = 'xic'`.
`filter`	defines the scan filter, default is `filter="ms"` if a wrong filter is set the function will return NULL and draws a warning.
`type`	c(xic, bpc, tic) for extracted ion , base peak or total ion chromatogram.

Details

Chromatograms come in different flavors but are always signal intensity values as a function of time. Signal intensities can be point estimates from scanning detectors or plain intensities from non-scanning detectors, e.g., UV trace. Scanning detector (mass analyzers) point estimates can be defined in different ways by, for instance, summing all signals of a given spectrum (total ion chromatogram or TIC), or by extracting signal around an expected value (extracted ion chromatogram = XIC), or by using the maximum signal contained in a spectrum (base peak chromatogram = BPC). On top, chromatograms can be computed from pre-filtered lists of scans. A total ion chromatogram (TIC), for instance, is typically generated by iterating over all MS1-level scans.

Value

chromatogram object(s) containing of a vector of `times` and a corresponding vector of `intensities`.

Author(s)

Christian Trachsel, Tobias Kockmann and Christian Panse <cp@fgz.ethz.ch> 2018, 2019, 2020.

References

Automated quality control sample 1 (autoQC01) analyzed across different Thermo Scientific mass spectrometers, MSV000086542.

See Also

- The Thermo Fisher Scientific RawFileReader C# code snippets `https://planetorbitrap.com/rawfilereader`.
- `https://CRAN.R-project.org/package=protViz`
- `https://massive.ucsd.edu/ProteoSAFe/dataset.jsp?accession=MSV000086542`

Examples

```
# Example 1: not meaningful but proof-of-concept
(rawfile <- rawrr::sampleFilePath())

rawrr::readChromatogram(rawfile, mass=c(669.8381, 726.8357), tol=1000) |> plot()
rawrr::readChromatogram(rawfile, type='bpc') |> plot()
rawrr::readChromatogram(rawfile, type='tic') |> plot()
```

```
# Example 2: extract iRT peptides
 if (require(ExperimentHub) & require(protViz)){
iRTpeptide <- c("LGGNEQVTR", "YILAGVENSK", "GTFIIDPGGVIR", "GTFIIDPAAVIR",
  "GAGSSEPVTGLDAK", "TPVISGGPYEYR", "VEATFGVDESNAK",
  "TPVITGAPYEYR", "DGLDAASYYAPVR", "ADVTPADFSEWSK",
  "LFLQFGAQGSPFLK")


# fetch via ExperimentHub
library(ExperimentHub)
eh <- ExperimentHub::ExperimentHub()
EH4547 <- normalizePath(eh[["EH4547"]])

(rawfile <- paste0(EH4547, ".raw"))
if (!file.exists(rawfile)){
    file.link(EH4547, rawfile)
}
op <- par(mfrow=c(2,1))
readChromatogram(rawfile, type='bpc') |> plot()
readChromatogram(rawfile, type='tic') |> plot()
par(op)

# derive [2H+] ions
((protViz::parentIonMass(iRTpeptide) + 1.008) / 2) |>
   readChromatogram(rawfile=rawfile) |>
   plot()
}
```

---

readFileHeader   *read file header Information*

Description

The function extracts meta information from a given rawfile.

Usage

```
readFileHeader(rawfile)
```

Arguments

rawfile   the name of the Thermo Fisher Scietific raw file

Value

A list object containing the following entries: RAW file version, Creation date, Operator, Number of instruments, Description, Instrument model, Instrument name, Serial number, Software version, Firmware version, Units, Mass resolution, Number of scans, Number of ms2 scans, Scan range, Time range, Mass range, Scan filter (first scan), Scan filter (last scan), Total number of filters, Sample name, Sample id, Sample type, Sample comment, Sample vial, Sample volume, Sample injection volume, Sample row number, Sample dilution factor, or Sample barcode.

Author(s)

Tobias Kockmann and Christian Panse 2018, 2019, 2020.

References

Thermo Fisher Scientific's NewRawfileReader C# code snippets https://planetorbitrap.com/rawfilereader.

Examples

```
rawrr::sampleFilePath() |> readFileHeader()
```

---

readIndex   *Read scan index*

Description

Read scan index

Usage

```
readIndex(rawfile, tmpdir = tempdir())
```

Arguments

rawfile	the name of the Thermo Fisher Scientific raw file.
tmpdir	defines the directory used to store temporary data generated by the .NET assembly `rawrr.exe`. The default uses `tempdir()`.

Value

returns a `data.frame` with the column names scan, scanType, rtinseconds, precursorMass, MSOrder, charge, masterScan, and dependencyType of all spectra.

Author(s)

Tobias Kockmann and Christian Panse <cp@fgz.ethz.ch>, 2020

Examples

```
Idx <- rawrr::sampleFilePath() |> rawrr::readIndex()
table(Idx$scanType)
plot(Idx$rtinseconds, Idx$precursorMass, col=as.factor(Idx$charge), pch=16)

table(Idx$MSOrder)
```

---

readSpectrum   *Reads spectral data from a raw file.*

Description

The function derives spectra of a given raw file and a given vector of scan numbers.

Usage

```
readSpectrum(
  rawfile,
  scan = NULL,
  tmpdir = tempdir(),
  validate = FALSE,
  mode = ""
)
```

Arguments

rawfile	the name of the Thermo Fisher Scientific raw file.
scan	a vector of requested scan numbers.
tmpdir	defines the directory used to store temporary data generated by the .NET assembly rawrr.exe. The default uses tempdir().
validate	boolean default is FALSE.
mode	if mode = "barebone" only mZ, intensity, pepmass, rtinseconds and charge state is returned. As default mode is "".

Details

All mass spectra are recorded by scanning detectors (mass analyzers) that log signal intensities for ranges of mass to charge ratios (m/z), also referred to as position. These recordings can be of continuous nature, so-called profile data (p), or appear centroided (c) in case discrete information (tuples of position and intensity values) are sufficient. This heavily compacted data structure is often called a peak list. In addition to signal intensities, a peak list can also cover additional peak attributes like peak resolution (R), charge (z), or local noise estimates. In short, the additional attributes further described the nature of the original profile signal or help to group peak lists with respect to their molecular nature or processing history. A well-known example is the assignment of peaks to peak groups that constitute isotope patterns (M, M+1, M+2, ...). The names of objects encapsulated within rawrrSpectrum instances are keys returned by the Thermo Fisher Scientific New RawFileReader API and the corresponding values become data parts of the objects, typically vectors.

Value

a nested list of rawrrSpectrum objects containing more than 50 values of scan information, e.g., the charge state, two vectors containing the mZ and its corresponding intensity values or the AGC information, mass calibration, ion optics . . .

Author(s)

Tobias Kockmann and Christian Panse <cp@fgz.ethz.ch> 2018, 2019, 2020, 2021

References

- C# code snippets of the NewRawfileReader library <https://planetorbitrap.com/rawfilereader>.
- rawrr: doi:10.1021/acs.jproteome.0c00866
- Universal Spectrum Explorer: <https://www.proteomicsdb.org/use/> doi:10.1021/acs.jproteome.1c00096

See Also

<https://massive.ucsd.edu/ProteoSAFe/dataset.jsp?accession=MSV000086542>

Examples

```
# Example 1
S <- rawrr::sampleFilePath() |> rawrr::readSpectrum(scan = 1:9)

S[[1]]

names(S[[1]])

plot(S[[1]])



# Example 2 - find best peptide spectrum match using the |> pipe operator
# fetch via ExperimentHub

if (require(ExperimentHub) & require(protViz)){
eh <- ExperimentHub::ExperimentHub()
EH4547 <- normalizePath(eh[["EH4547"]])

(rawfile <- paste0(EH4547, ".raw"))
if (!file.exists(rawfile)){
    file.link(EH4547, rawfile)
}

GAG <- "GAGSSEPVTGLDAK"

.bestPeptideSpectrumMatch <- function(rawfile,
    sequence="GAGSSEPVTGLDAK"){
    readIndex(rawfile) |>
        subset(abs((1.008 + (protViz::parentIonMass(sequence) - 1.008) / 2) -
            precursorMass) < 0.001, select = scan) |>
        unlist() |>
        readSpectrum(rawfile = rawfile) |>
        lapply(function(x) {
          y <- protViz::psm(sequence = GAG, spec=x, plot=FALSE);
          y$scan <- x$scan; y
        }) |>
```

```
        lapply(FUN= function(x){
          score <- sum(abs(x$mZ.Da.error) < 0.01);
          cbind(scan=x$scan, score=score)
        }) |>
        (function(x) as.data.frame(Reduce(rbind, x)))() |>
        subset(score > 0) |>
        (function(x) x[order(x$score, decreasing = TRUE),
            'scan'])() |>
        head(1)
}

start_time <- Sys.time()
bestMatch <- .bestPeptideSpectrumMatch(rawfile, GAG) |>
    rawrr::readSpectrum(rawfile=rawfile) |>
    lapply(function(x) protViz::peakplot(peptideSequence = GAG, x))

end_time <- Sys.time()
end_time - start_time

# Example 3
# using proteomicsdb \doi{10.1101/2020.09.08.287557}
# through https://www.proteomicsdb.org/use/

.UniversalSpectrumExplorer <- function(x, sequence){
    m <- protViz::psm( sequence, x)
    cat(paste(x$mZ[m$idx], "\t", x$intensity[m$idx]), sep = "\n")
}

rawrr::readSpectrum(rawfile=rawfile, 11091) |>
   lapply(function(x).UniversalSpectrumExplorer(x, sequence = GAG))
 }
```

---

sampleFilePath   *A small file size* sample.raw *BLOB*

Description

The binary example file sample.raw, shipped with the package, contains 574 Fourier-transformed Orbitrap spectra (FTMS) recorded on a Thermo Fisher Scientific Q Exactive HF-X. The mass spectrometer was operated in line with a nano electrospray source (NSI) in positive mode (+). All spectra were written to disk after applying centroiding (c) and lock mass correction.

Usage

```
sampleFilePath()
```

Details

Thermo Fisher Scientific Q Exactive HF-X raw file of size 1.5M bytes and checksum MD5 (sample.raw) = fe67058456c79af7442316c474d20e96. Additional raw data for demonstration and extended testing is available through MSV000086542 and the tartare package. **Lions love raw meat!**

Value

file path of the sample.raw location.

Author(s)

Tobias Kockmann, 2018, 2019.

References

- Bioconductor tartare package.
- Automated quality control sample 1 (autoQC01) analyzed across different Thermo Fisher Scientific mass spectrometers, MSV000086542.

Examples

```
sampleFilePath()
```

---

scanNumber	*Accessor function for scan number of* `rawrrSpectrum` *objects*

Description

Accessor function for scan number of `rawrrSpectrum` objects

Usage

```
scanNumber(x)
```

Arguments

x   A rawrrSpectrum object

Details

This accessor function returns the scan number of a mass spectrum stored as `rawrrSpectrum` object. Scan numbers are equal to the scan index j running from 1 to n with n being the last scan of a raw file.

Value

The scan number of type `integer`

Examples

```
S <- readSpectrum(rawfile = sampleFilePath(), 1:10)
scanNumber(S[[1]])
```

summary.rawrrChromatogram

Text summary of chromatogram

Description

Text summary of chromatogram

Usage

```
## S3 method for class 'rawrrChromatogram'
summary(object, ...)
```

Arguments

object   A rawrrChromatogram object

...   Function passes additional arguments.

Value

A rawrrChromatogram object

Examples

```
C <- readChromatogram(rawfile = sampleFilePath(),
mass = c(445.1181, 519.1367))
summary(C[[1]])
summary(C[[2]])
```

summary.rawrrSpectrum   *Basic summary function*

Description

Basic summary function

Usage

```
## S3 method for class 'rawrrSpectrum'
summary(object, ...)
```

Arguments

object   an rawrrSpectrum object.

...   Arguments to be passed to methods.

Value

This function creates a print message.

Author(s)

Christian Panse and Tobias Kockmann, 2020.

---

tic   *Total ion current of a spectrum*

Description

Total ion current of a spectrum

Usage

```
tic(x)
```

Arguments

x   A rawrrSpectrum object

Value

A double vector of length one.

Examples

```
S <- readSpectrum(rawfile = sampleFilePath(), 1)
tic(S[[1]])
```

---

validate_rawrrIndex   *Validate output of the readIndex function*

Description

Checks the validity of an readIndex returned object.

Usage

```
validate_rawrrIndex(x)
```

Arguments

x   object to be validated.

Value

Validated `data.frame` of `readIndex` object

Author(s)

Tobias Kockmann and Christian Panse, 2020-12-09.

Examples

```
rawrr::sampleFilePath() |> rawrr::readIndex() |> rawrr::validate_rawrrIndex()
```

---

validate_rawrrSpectrum

Validate instance of class rawrrSpectrum

Description

Checks the validity of `rawrrSpectrum` object attributes.

Usage

```
validate_rawrrSpectrum(x)
```

Arguments

x   object to be validated.

Value

Validated `rawrrSpectrum` object

Author(s)

Tobias Kockmann and Christian Panse, 2020.

Index