MemUse

July 6, 2013

`memuse-package` *Core memuse Classes and Methods*

Description

The package gives the user the size (memory usage) of an in-core, dense matrix. The output is an S4 class object that can be manipulated to be presented in different ways (different units, short/long versions of those units, etc.). For more information, see the package vignette and the manual.

Details

Package:	memuse
Type:	Package
License:	GPL
LazyLoad:	yes

Author(s)

wrathematics

References

http://wrathematics.github.io

`memuse-class` *Class memuse*

Description

Memory usage class object.

Creating Objects

```
new('memuse', size = ..., unit = ..., unit.prefix = ...,  unit.names = ...)
```

Slots

`size`: Object of class `numeric`

`unit`: Object of class `character`

`unit.prefix`: Object of class `character`

`unit.names`: Object of class `character`

Prototype

numeric size 0

character unit "B"

character unit.prefix "IEC"

character unit.names "short"

Details

`memuse` is the container for memory usage data for an unallocated, dense, in-core R object. The `size` slot contains the memory usage in some unit of bytes. The `unit` slot contains the unit of bytes that `size` is stored in (e.g., kb, mb, gb, ...). The `unit.prefix` slot contains the unit prefix, either IEC or SI. The `unit.names` slot contains the unit names, either short (e.g., kb) or long (e.g., kilobyte).

See the memuse guide vignette for more details.

See Also

`Control`

`Control Variables` *Control Variables for the memuse Package.*

Description

A set of controls which provides default values for many functions in this package.

Details

.UNIT defaults to "best". The default choice will scale size values to the nearest (by scaling factor — 1024 or 1000 depending on unit prefix). Other acceptable choices are, for example, "kb" or "kib". If the user requests the wrong unit by prefix (e.g., "kb" instead of "kib" when the unit prefix is IEC), then the correct one will be chosen for the user.

.PREFIX defaults to "IEC". Acceptable values are "IEC" and "SI".

.NAMES defaults to "short". Acceptable values are "short" and "long".

All values are case insensitive. The correct case will be determined for the user if the incorrect case is supplied. For an explanation of what these values do, see memuse-class or the package user guide vignette.

See Also

memuse-class

Environment *Environment for the memuse Package*

Description

The environment for the memuse package.

Constructor *memuse Constructor*

Description

Constructor for objects of class memuse.

Usage

```
memuse(size=0, unit=.UNIT, unit.prefix=.PREFIX, unit.names=.NAMES)
mu(size=0, unit=.UNIT, unit.prefix=.PREFIX, unit.names=.NAMES)
```

Arguments

size	numeric; indicates the unit-multiple number of bytes used by the object.
unit	string; the unit of storage, such as "MiB" or "MB", depending on prefix. Case is ignored.
unit.prefix	string; the unit prefix, namely IEC or SI. Case is ignored.
unit.names	string; control for whether the unit names should be printed out or their abbreviation should be used. Options are "long" and "short", respectively. Case is ignored.

Details

deets

Value

Returns a memuse class object.

See Also

Constructor, memuse-class

Examples

```
x <- mu(100, unit="kb")
x

y <- mu(100, unit="kb", unit.prefix="SI")
y
```

Accessors *Accessors*

Description

Accessor methods for slots of objects of class memuse.

Usage

```
  ## S4 method for signature 'memuse'
size(x)
  ## S4 method for signature 'memuse'
unit(x)
  ## S4 method for signature 'memuse'
unit.prefix(x)
  ## S4 method for signature 'memuse'
unit.names(x)
```

Arguments

x	memuse object

Details

These methods are just syntactic sugar for ordinary S4 slot accessing. So for example, size(x) is no different semantically from calling x@size.

Value

Returns a numeric element in the case of size(), otherwise a string is returned.

Methods

```
signature(x = "memuse")
```

See Also

Replacers, memuse-class

Examples

```
x <- mu(1e6)

size(x)
unit(x)
unit.prefix(x)
unit.names(x)
```

Print *Printing*

Description

Print methods for memuse class objects.

Usage

```
  ## S4 method for signature 'memuse'
print(x, ..., unit=x@unit, unit.prefix=x@unit.prefix, unit.names=x@unit.names, digits=3)
  ## S4 method for signature 'memuse'
show(object)
```

Arguments

x, object	memuse class object
...	extra arguments
unit	the unit to be used in printing; defaults to x's unit
unit.prefix	the unit prefix to be used in printing; defaults to x's unit.prefix
unit.names	the unit names (short or long) to be used in printing; defaults to x's unit.names
digits	the number of decimal digits to print; default is 3

Details

deets

Value

Returns a string.

Methods

```
signature(x = "memuse")
```

See Also

Constructor, memuse-class

Examples

```
x <- mu(1e6)

print(x)
x # same as show(x)
```

Replacers *Replacers*

Description

Replacement methods for slots of objects of class memuse.

Usage

```
size(x) <- value
unit(x) <- value
unit.prefix(x) <- value
unit.names(x) <- value
```

Arguments

x	memuse object
value	replacement value

Details

These methods are syntactic sugar for assignment using ordinary S4 accessors. So for example, size(x) <- 10 is semantically no different from calling x@size <- 10

These methods are strict replacement methods; if you need to swap the units of a memuse class object, you should probably be using the Swaps methods. See example below for further details.

Value

Returns a numeric element in the case of size(), otherwise a string is returned.

Methods

```
signature(x = "memuse")
```

See Also

Accessors, memuse-class

Examples

```
x <- mu(2000, unit="bytes")
x

size(x) <- 1000
x
```

Arithmetic *memuse Arithmetic*

Description

Binary arithmetic operations for memuse objects.

Usage

```
x + y
x - y
x * y
x / y
x ^ y
```

Arguments

x, y	memuse class objects

Details

deets

Value

Returns a memuse class object.

Methods

```
signature(x = "memuse", y = "memuse")
signature(x = "numeric", y = "memuse")
signature(x = "memuse", y = "numeric")
```

See Also

Constructor, memuse-class

Examples

```
x <- mu(200)
y <- mu(100)

x+y
x-y
x*y
x/y
x^2
```

Swaps *Swaps*

Description

Binary arithmetic operations for memuse objects.

Usage

```
  ## S4 method for signature 'memuse'
swap.unit(x, unit)
  ## S4 method for signature 'memuse'
swap.prefix(x)
  ## S4 method for signature 'memuse'
swap.names(x)
```

Arguments

x	memuse object
unit	new unit for the memuse object after the swap occurs

Details

deets

Value

Returns a memuse class object.

Methods

```
signature(x = "memuse")
```

See Also

Constructor, memuse-class

Examples

```
x <- mu(1e6)

x
swap.prefix(x)
swap.names(x)
swap.unit(x, "bytes")
```

---

howbig *How Big in Memory is a Matrix with Known Rows/Cols*

Description

Determines the memory usage for a dense, in-core, numeric matrix.

Usage

```
howbig(nrow, ncol, unit=.UNIT, unit.prefix=.PREFIX, unit.names=.NAMES,
       ..., type="double", intsize=4)
howbig.par(nrow, ncol, cores, par="dmat", unit=.UNIT, unit.prefix=.PREFIX,
           unit.names=.NAMES, ..., type="double", intsize=4, ICTXT=0)
```

Arguments

nrow, ncol	Number of (global) rows/columns of the matrix.
cores	The number \itemunitstring; the unit of storage, such as "MiB" or "MB", depending on prefix. Case is ignored. \itemunit.prefixstring; the unit prefix, namely IEC or SI. Case is ignored. \itemunit.namesstring; control for whether the unit names should be printed out or their abbreviation should be used. Options are "long" and "short", respectively. Case is ignored. \item...Additional arguments. \itemtype"double" or "int"; the storage type of the data matrix. If you don't know the type, it is probably stored as a double, so the default value will suffice. \itemintsizeThe size (in bytes) of an integer. Default is 4, but this is platform dependent.
	These functions provide the memory usage of an unallocated, dense, in-core, numeric matrix. As the names suggest, howbig() simply returns the size (as a memuse object), while howbig.par() is the parallel, distributed analogue. The latter returns the memory usage of a *distributed*, object
	Returns a memuse class object.
	Constructor, memuse-class
	x <- mu(200) y <- mu(100)
	x+y x-y x*y x/y x^2
	Methods

howmany *How Many Rows/Cols of a Matrix for a Memory Size*

Description

Binary arithmetic operations for memuse objects.

Usage

```
x + y
x - y
x * y
x / y
x ^ y
```

Arguments

x, y	memuse class objects

Details

deets

Value

Returns a memuse class object.

Methods

signature(x = "memuse", y = "memuse")

signature(x = "numeric", y = "memuse")

signature(x = "memuse", y = "numeric")

See Also

Constructor, memuse-class

Examples

```
x <- mu(200)
y <- mu(100)

x+y
x-y
x*y
x/y
x^2
```

Index