Package 'rDotNet'

September 1, 2017

Type Package

Title Low-Level Interface to the '.NET' Virtual Machine Along the Lines of the R C/Call API

Version 0.9.1

Date 2017-08-30

Author Jonathan Shore <jonathan.shore@gmail.com>

Maintainer Jonathan Shore <jonathan.shore@gmail.com>

Description Low-level interface to '.NET' virtual machine along the lines of the R C .call interface. Can create '.NET' object, call methods, get or set properties, call static functions, etc.

License Apache License (== 2.0)

URL https://github.com/tr8dr/.Net-Bridge/tree/master/src/R/rDotNet

Imports Rcpp (>= 0.12.3), testthat

LinkingTo Rcpp

ByteCompile true

SystemRequirements mono 4.x or higher on OSX / Linux, .NET 4.x or higher on Windows, 'msbuild' and 'nuget' available in the path

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-09-01 09:27:10 UTC

R topics documented:

`.ccall` *call a method on a .NET object*

Description

This function calls a method on a previously created object.

Usage

```
.ccall(obj, methodname, ...)
```

Arguments

`obj`	An object previously created with `.cnew()`
`methodname`	name of method to be called on object
`...`	a sequence of arguments to the method. The arguments can either be fundamental types, vectors, matrices, or other objects. Fuzzy matching will be applied, allowing imperfectly matched argument types to be converted, as may be needed given R's more basic type system.

Examples

```
## Not run:
series <- rnorm(100)

## create instance of an object, using fully qualified class name
obj <- .cnew ("com.stg.math.Model", 1.0, 3.5)

## call method F(3) on object
val <- .ccall (obj, "F", 3)

## call overload method F of 2 arguments F(3,4) on object
val <- .ccall (obj, "F", 3, 4)

## call method Estimate(series) on object
series <- rnorm(100)
val <- .ccall (obj, "Estimate", series)

## End(Not run)
```

.cget — *Get a property value on a .NET object*

Description

This function gets the value of a property on a previously created .NET object.

Usage

```
.cget(obj, propertyname)
```

Arguments

obj	Previously created .NET object
propertyname	name of the property to retrieve

Examples

```
## Not run:
obj <- .cnew ("com.stg.math.statistics.AugmentedDickeyFuller", series, 3, 'Driftless')
lag <- obj$Get("Lag")

## End(Not run)
```

.cinit — *Initialize R <-> .NET bridge*

Description

The function either connects to an existing running CLR bridge process at the given host:port or instantiates a new CLR server on the local computer with the given port and server arguments.

Usage

```
.cinit(host='localhost', port=56789, dll=NULL, server.args=NULL)
```

Arguments

host	The host machine on which the CLR bridge server is running; generally this is the localhost, which is the default.
port	The port on which the CLR bridge is listening (default: 56789)
dll	The path to an optional library dll to be loaded by the server. This dll would contain .NET classes and functions one wants to call from R.
server.args	Optional parameters to the CLRServer process (CLRServer.exe -help to list the options).

Details

If the .NET libraries are to be changed, the CLRServer process and R should be restarted. CLR references in the R session are only valid for the current CLR server instance.

Instead of calling `.cinit(dll="~/mydll.dll")` explictly one can set an environment variable `Sys.setenv(rDotNet_DLL="~/mydll.dll")` and use `.cnew()` and other functions after loading the package as opposed to first calling `.cinit`. One can also run the `CLRServer` from the command line or an IDE with the appropriate DLL.

Examples

```
## Not run:

## create .NET bridge server, loading personal library to be referenced in the R session
.cinit (dll="~/Dev/MyLibrary.dll")
obj <- .cnew("NormalDistribution1D", 0.0, 1.0)

## alternative without explicit initialization
Sys.setenv(rDotNet_DLL="~/Dev/models.dll")
...
obj <- .cnew("NormalDistribution1D", 0.0, 1.0)


## End(Not run)
```

.cnew *Create .NET object for class of given name and arguments*

Description

A .NET object will be created and a reference / R object will be returned, providing access to all public methods and properties.

Usage

```
.cnew(classname, ...)
```

Arguments

`classname`	The name of the .NET based class to be created. Can either be a fully qualified name like "com.stg.models.DickeyFuller" or "DickeyFuller" if the classname is unique in the VM.
`...`	a sequence of arguments to the constructor for this class. The arguments can either be fundamental types, vectors, matrices, or other objects. Fuzzy matching will be applied, allowing imperfectly matched argument types to be converted, as may be needed given R's more basic type system.

Details

The type name to be created need not be fully qualified if it is unique in the VM. Constructor arguments are matched with a fuzzy approach so that the lack of type precision within R does not present a problem for strong typing on the .NET side. Structurally equivalent types are casted or ortherwise converted.

For example if a C# ctor signature was the following: `AugmentedDickeyFuller (Vector<double> series, int lag, ADF` where ADFType is an enumeration. A call to `.cnew("com.stg.math.statistics.AugmentedDickeyFuller", c(1,2,3,4` would convert the R vector to a Vector<double> and `'Driftless'` to the enum value `ADFType.Driftless`.

Examples

```
## Not run:

series <- rnorm(100)

## create instance of an object, using fully qualified class name
obj <- .cnew ("com.stg.math.statistics.AugmentedDickeyFuller", series, 3, 'Driftless')

## or create without fully qualified name
obj <- .cnew ("AugmentedDickeyFuller", series, 3, 'Driftless')

## End(Not run)
```

.cset *Set a property value on a .NET object*

Description

This function sets the value of a property on previously created .NET object.

Usage

```
.cset(obj, propertyname, value)
```

Arguments

`obj`	Previously created .NET object
`propertyname`	name of the property to set
`value`	property value to set to

Examples

```
## Not run:
obj <- .cnew ("com.stg.models.FastMFT")
obj$Set("Alpha", 0.90)
obj$Set("Gamma", 1e-4)
```

```
gamma <- obj$Get("Gamma")

## End(Not run)
```

`.cstatic` *call a static method of a .NET class*

Description

This function calls a static method on a .NET type (class).

Usage

```
.cstatic(classname, methodname, ...)
```

Arguments

`classname`	The name of the .NET based class to be created. Can either be a fully qualified name like "com.stg.models.DickeyFuller" or "DickeyFuller" if the classname is unique in the VM.
`methodname`	name of the static method to be called on type
`...`	a sequence of arguments to the method. The arguments can either be fundamental types, vectors, matrices, or other objects. Fuzzy matching will be applied, allowing imperfectly matched argument types to be converted, as may be needed given R's more basic type system.

Examples

```
## Not run:

## call MtM static method on MatrixUtils
matrix <- diag(c(1,2,3,4,5,6,7))
newmat <- .cstatic ("com.stg.math.MatrixUtils", "MtM", matrix)

## End(Not run)
```

print.rDotNet *print .NET object*

Description

This operator returns the object as a string

Usage

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

Arguments

x	An object previously created with .cnew()
...	other arguments, not used

Examples

```
## Not run:

## create instance of an object, using fully qualified class name
obj <- .cnew ("com.stg.math.Model", 1.0, 3.5)

## print the object as a string
print (obj)


## End(Not run)
```

[.rDotNet *return ith element on a .NET object (such as an array, list, etc)*

Description

This operator returns the ith element of an an indexable object.

Usage

```
## S3 method for class 'rDotNet'
obj[ith]
```

Arguments

obj	An object previously created with .cnew()
ith	numerical index into array, list, etc

Examples

```
## Not run:

## create instance of an object, using fully qualified class name
obj <- .cnew ("com.stg.math.Model", 1.0, 3.5)

## get the 34th element from this indexable object
element <- obj[34]


## End(Not run)
```

Index