Package 'RGalaxy'

April 10, 2015

Title Make an R function available in the Galaxy web platform

Description Given an R function and its manual page, make the documented function available in Galaxy.

Version 1.10.0

Author Dan Tenenbaum

Maintainer Bioconductor Package Maintainer <maintainer@bioconductor.org>

biocViews Infrastructure

Depends XML, methods, tools, optparse, digest,

Imports BiocGenerics, Biobase, roxygen2

Suggests RUnit, hgu95av2.db, knitr, formatR, Rserve

Enhances RSclient

License Artistic-2.0

KeepSource TRUE

Collate 'AllClasses.R' 'galaxy.R' 'man.R' 'test_functions.R' 'utils.R'

VignetteBuilder knitr

R topics documented:

`addTwoNumbers` *Add two numbers*

Description

An example function that can be made into a Galaxy tool. Takes two numbers, adds them, and returns a file containing the result.

Usage

```
addTwoNumbers(number1=GalaxyNumericParam(required=TRUE),
        number2=GalaxyNumericParam(required=TRUE),
        sum=GalaxyOutput("sum", "txt"))
```

Arguments

`number1`	The first number to add.
`number2`	The second number to add.
`sum`	Where the result of the addition should be written.

Value

invisible(NULL)

See Also

`galaxy`, `GalaxyConfig`, `GalaxyOutput`, `addTwoNumbers`

Examples

```
t <- tempfile()
addTwoNumbers(2, 2, t)
readLines(t, warn=FALSE)
```

addTwoNumbersWithTest *Add two numbers (with functional test)*

Description

An example function that can be made into a Galaxy tool. Takes two numbers, adds them, and returns a file containing the result. This version demonstrates a functional test.

Usage

```
addTwoNumbersWithTest(number1=GalaxyNumericParam(required=TRUE, testValues=5L),
        number2=GalaxyNumericParam(required=TRUE, testValues=5L),
        sum=GalaxyOutput("sum", "txt"))
```

Arguments

number1	The first number to add.
number2	The second number to add.
sum	Where the result of the addition should be written.

Value

invisible(NULL)

See Also

galaxy, GalaxyConfig, GalaxyOutput, addTwoNumbers

Examples

```
t <- tempfile()
addTwoNumbersWithTest(2, 2, t)
readLines(t, warn=FALSE)
runFunctionalTest(addTwoNumbersWithTest) ## should return TRUE
```

anotherTestFunction *Add two matrices*

Description

An example function that can be made into a Galaxy tool. Reads matrices from two tab-delimited files, adds them, and writes the result to a comma-separated file and a PDF plot.

Usage

```
anotherTestFunction(inputfile1=GalaxyInputFile(),
    inputfile2=GalaxyInputFile(),
    plotTitle=GalaxyCharacterParam(c("TitleA"="A", "TitleB"="B")),
    plotSubTitle=GalaxyCharacterParam("My subtitle"),
    outputfile1=GalaxyOutput("mydata", "csv"),
    outputfile2=GalaxyOutput("myplot", "pdf"))
```

Arguments

inputfile1	The filename of the first tab-separated matrix.
inputfile2	The filename of the second tab-separated matrix.
plotTitle	The title of the plot to create.
plotSubTitle	The subtitle of the plot to create.
outputfile1	The filename of the comma-separated output file to generate.
outputfile2	The filename of the PDF plot file to create.

Value

invisible(NULL)

See Also

galaxy, GalaxyConfig, GalaxyOutput

Examples

```
anotherTestFunction(system.file("extdata", "a.tsv", package="RGalaxy"),
    system.file("extdata", "b.tsv", package="RGalaxy"),
    "My Plot Title", "My Plot Subtitle",
    "output.csv", "output.pdf")
```

---

foo *a foo function*

Description

a foo function

Usage

```
foo(input = GalaxyInputFile(), x = GalaxyNumericParam(),
  y = TRUE,
  z = GalaxyCharacterParam(c("Seattle", "Tacoma", "Olympia")),
  output = GalaxyOutput("pdf"))
```

Arguments

input	An input dataset
x	the x param
y	the y param
z	the z param
output	the output

Details

nothing

`functionToGalaxify` *Add two matrices*

Description

An example function that can be made into a Galaxy tool. Reads matrices from two tab-delimited files, adds them, and writes the result to a comma-separated file and a PDF plot.

Usage

```
functionToGalaxify(inputfile1=GalaxyInputFile(),
    inputfile2=GalaxyInputFile(),
    plotTitle=GalaxyCharacterParam(testValues="test plot title"),
    plotSubTitle=GalaxyCharacterParam("My subtitle",
        testValues="test plot subtitle"),
    outputfile1=GalaxyOutput("mydata", "csv"),
    outputfile2=GalaxyOutput("myplot", "png"))
```

Arguments

inputfile1	The filename of the `first` tab-separated matrix.
inputfile2	The filename of the second tab-separated matrix.
plotTitle	The title of the plot to create.
plotSubTitle	The subtitle of the plot to create.
outputfile1	The filename of the comma-separated output file to generate.
outputfile2	The filename of the PNG plot file to create.

Details

This trivial method illustrates some best practices to use when writing functions to be adapted as Galaxy tools. For example, any error conditions should be handled with `stop` with a useful/informative error message. The Galaxy user will see these messages if an error occurs.

Functions which take datasets as input should accept as arguments the filenames pointing to those datasets. The Galaxy user interface will allow the user to chose the dataset graphically.

Return values of functions are ignored. Function output should be written to one or more files, and the names of these files should be passed into the function as arguments.

Functions should be documented with a manual page. `RGalaxy` will use this manual page to fill in relevant sections of the Galaxy XML file.

Value

invisible(NULL)

See Also

`galaxy`, `GalaxyConfig`, `GalaxyOutput`

Examples

```
functionToGalaxify(system.file("extdata", "a.tsv", package="RGalaxy"),
    system.file("extdata", "b.tsv", package="RGalaxy"),
    "My Plot Title", "My Plot Subtitle",
    "output.csv", "output.pdf")
```

`galaxy` *Make a function available in Galaxy*

Description

Makes an R function available in the Galaxy web platform. Automates all the work necessary to expose an R function in Galaxy. See the vignette for more information

Usage

```
galaxy(func,
        package=getPackage(func),
        manpage=func,
        name=getFriendlyName(func),
        version=getVersion(func),
        galaxyConfig,
        dirToRoxygenize,
        RserveConnection=NULL,
        path.to.R="",
```

```
functionalTestDirectory)
```

Arguments

func	**Required.** TA character vector naming the function to make available in Galaxy. This function should be entirely-self contained, and should accept as arguments the full paths to its input and output file(s).
package	The name of the package where func lives, or NULL if it is not in a package. By default, RGalaxy will try to determine which package the code lives in and set the parameter to NULL if it cannot.
manpage	The full path to the Rd-formatted manual page for the function, or if the func is in a package, an alias that will pull up that manpage.
name	Text describe the action this function performs. Becomes a clickable link in Galaxy. By default, if your function is called "fooBar", name is set to "Foo Bar".
version	The version of this function. If func lives in a package, defaults to the version of package.
galaxyConfig	**Required.** A link{GalaxyConfig} object describing your Galaxy configuration.
dirToRoxygenize	If present, points to the directory of a source package upon which to run roxygenize() from the roxygen2 package, creating manual pages from source code comments.
RserveConnection	If set (by calling RserveConnection), sends the function to a running Rserve for evaluation. This can make functions run faster since dependencies have already been loaded. If NULL (the default), function is run normally. See vignette for more information.
path.to.R	If not empty, Galaxy will look for R in this directory. This is useful if there are multiple versions of R on your system and your Galaxy configuration prevents you from setting your PATH to point to the right one.
functionalTestDirectory	If package is not specified, this directory is searched for functional test fixtures.

Value

invisible(NULL)

See Also

GalaxyConfig, GalaxyOutput, GalaxyParam

Examples

```
## set up galaxyHome (you dont need to do this if
## you really have Galaxy installed)
galaxyHome <- tempdir()
dir.create(galaxyHome, recursive=TRUE, showWarnings=FALSE)
```

```
file.copy(system.file("galaxy", "tool_conf.xml", package="RGalaxy"),
    file.path(galaxyHome, "tool_conf.xml"), overwrite=FALSE)

## Not run:
galaxy("functionToGalaxify",
    galaxyConfig=GalaxyConfig(galaxyHome, "mytool",
      "Test Section", "testSectionId"),
    RserveConnection=RserveConnection())

## End(Not run)
```

GalaxyClasses *Galaxy Parameter Classes*

Description

Galaxy Parameter Classes These classes encapsulate parameters to be passed to functions exposed in Galaxy.

Usage

```
GalaxyIntegerParam(...)

GalaxyNumericParam(...)

GalaxyCharacterParam(...)

GalaxyLogicalParam(...)

GalaxySelectParam(...)
```

Arguments

...	Arguments can be a single unnamed argument, corresponding to the name of the class, e.g., integer, numeric, character, or logical. The first argument can be missing, can be of length 1 or can be a vector of a greater length, in which case it is rendered as a dropdown (select) list. **testValues:** (optional) Either an integer(1), numeric(1), character(1), or logical(1), depending on the class used. A value to be used in functional testing (running your function with a fixed set of inputs and checking that the outputs match what is expected). Additional parameters are as follows: **label:** A character(1). The friendly name for this parameter. By default, RGalaxy will use the parameter's name ("fooBar" is changed to "Foo Bar") if this is not supplied.

min: (optional) An integer(1). If the parameter is integer or numeric, this specifies a minimum value for the parameter. Galaxy will not allow lower values.

max: (optional) An integer(1). If the parameter is integer or numeric, this specifies a maximum value for the parameter. Galaxy will not allow higher values.

force_select: (default: FALSE) A logical(1). If this parameter is a dropdown list (i.e., the first argument is a vector of length > 1), a TRUE value forces the user to select an item.

display: (optional) A character(1). If parameter is a dropdown list, this can be set to "checkboxes" or "radio" which renders the list as a set of check boxes or radio buttons. Defaults to a drop-down menu select list.

checked: (default: FALSE) A logical(1). Whether the rendered checkbox should be checked. Only applicable if the parameter is a logical of length < 2.

size: (default: 60L) An integer(1) The width of the field, in characters. For character parameters only.

required: (default: FALSE) A logical(1). Whether Galaxy will require this field be filled in.

requiredMsg: (default: "This field is required.") A character(1) If `required` is TRUE, the message to display to the user when the leave the field empty.

Details

The arguments to these classes are based on http://wiki.g2.bx.psu.edu/Admin/Tools/Tool%20Config%20Syntax#A.3Cparam.3E_tag_set

Examples

```
GalaxyIntegerParam()
GalaxyIntegerParam(1L)
GalaxySelectParam(c(a=1L, b=2L))

GalaxyNumericParam(2.0, required=TRUE)

GalaxySelectParam(c("First Choice"="one",
"Second Choice"="two")) # a select list
GalaxyLogicalParam(checked=TRUE)
```

GalaxyConfig-class *Class* "GalaxyConfig"

Description

Represents information about the configuration of Galaxy.

Usage

```
GalaxyConfig(galaxyHome, toolDir, sectionName, sectionId)
```

Arguments

galaxyHome	**Required.** The full path to your Galaxy distribution.
toolDir	**Required.** The directory where the files associated with your function will reside.
sectionName	**Required** A friendly name for the section of Galaxy tools that will contain your function. Multiple tools can be in a single section; section names are headings in the left-hand side of the Galaxy window. If multiple tools are to reside in the same section, they must have identical values for sectionName and sectionId.
sectionId	An internal identifier for the section of Galaxy tools that will contain your function. If multiple tools are to reside in the same section, they must have identical values for sectionName and sectionId.

Methods

No methods defined with class "GalaxyConfig" in the signature.

See Also

link{galaxy}, GalaxyOutput

Examples

```
GalaxyConfig(galaxyHome=getwd(), toolDir="mytool", sectionName="Test Section",
    sectionId="testSectionId")
```

GalaxyInputFile-class *Class* "GalaxyInputFile"

Description

Represents a dataset hosted on Galaxy, to be passed to an R function that you expose in Galaxy.

Usage

```
GalaxyInputFile(required=TRUE, formatFilter=character(0))
```

Arguments

required	Whether to require that the user provide this dataset.
formatFilter	(optional) A file type for filtering the list of possible input data sets. Should be one of the file types listed in the datatypes_conf.xml file in the root of your Galaxy distribution.

Methods

No methods defined with class "GalaxyInputFile" in the signature.

See Also

link{galaxy}, GalaxyConfig

Examples

```
inputfile1=GalaxyInputFile()
## The user can only choose csv files:
inputfile2=GalaxyInputFile(formatFilter="csv")
```

GalaxyOutput-class *Class* "GalaxyOutput"

Description

Represents an output file generated by an R function in Galaxy.

Usage

```
GalaxyOutput(basename, format)
```

Arguments

basename	**Required.** The name of the output file, minus the extension.
format	**Required.** The file type of the output file. For suggested extensions, see GALAXY_HOME/datatypes_co

Methods

No methods defined with class "GalaxyOutput" in the signature.

See Also

link{galaxy}, GalaxyConfig

Examples

```
params <- list(
    outputfile1=GalaxyOutput("plot", "pdf"),
    outputfile2=GalaxyOutput("data", "csv"))
```

getFriendlyName *Change a camelCase name to a friendlier version*

Description

Takes a string like "fooBarBaz" and returns "Foo Bar Baz".

Usage

```
getFriendlyName(camelName)
```

Arguments

camelName	The "camelCased" name to make friendly.

Details

Used by `galaxy()` to create default labels based on function and parameter names.

Value

The friendly version of the camel-cased name.

See Also

galaxy, GalaxyConfig, GalaxyOutput

Examples

```
getFriendlyName("fooBarBaz")
```

gmessage *Sends informational, warning, and error messages to the user.*

Description

Send error, warning, and informational messages to the user. Use these instead of message, warning, and stop. Output is wrapped consistently and passed through sprintf so you can use inline formatting (see examples). Output of `gstop` will appear in Galaxy user's web browser.

Usage

```
gmessage(..., appendLF = TRUE)

gstop(..., call. = FALSE)

gwarning(..., call. = FALSE, immediate. = FALSE)
```

Arguments

...	Passed to sprintf.
appendLF	Passed to message.
call.	Passed to stop or warning.
immediate.	Passed to warning.

Value

NULL

See Also

message, warning, stop, sprintf

Examples

```
gmessage("This is an %s message.", "example")
## Not run:
gstop("Encountered a %s error.", "serious")

## End(Not run)
## Not run:
 gwarning("Something is not quite right.")

## End(Not run)
```

---

probeLookup *Get the PFAM and SYMBOL names for a set of Affymetrix probe IDs*

Description

Given a space-separated list of Affymetrix probe IDs, return the PFAM and SYMBOL names. Uses hgu95av2.db and the select method from the AnnotationDbi package.

Usage

```
probeLookup(probe_ids=GalaxyCharacterParam(
  required=TRUE,
  testValues="1002_f_at 1003_s_at"),
    outputfile=GalaxyOutput("probeLookup", "csv"))
```

Arguments

probe_ids	A space-separated list of Affymetrix probe IDs.
outputfile	The name of a .csv file where the returned output is to be written.

Details

Given one or more Affymetrix probe IDs separated by spaces, this function will return the PFAM and SYMBOL names associated with them in human. The function uses the `hgu95av2.db` package and the `select` function in `AnnotationDbi`.

Value

invisible(NULL)

Examples

```
t <- tempfile()
probeLookup("1002_f_at 1003_s_at", t)
readLines(t)
```

RserveConnection-class

Class "RserveConnection"

Description

Represents a connection to Rserve.

Usage

```
RserveConnection(...)
```

Arguments

...	**host**	character(1) The name of the host. Defaults to "localhost".
	port	integer(1) The port. Defaults to 6311L.

Methods

No methods defined with class "RserveConnection" in the signature.

See Also

link{galaxy}, GalaxyOutput

Examples

```
RserveConnection(host="myhost", port=2012L)
```

`runFunctionalTest` *Run the functional test associated with a function.*

Description

FIXME

Usage

```
runFunctionalTest(func, functionalTestDirectory)
```

Arguments

`func`	A function to be exposed in Galaxy.
`functionalTestDirectory`	Where to find functional tests, if func is not in a package.

Value

Whether the test passes.

`testFunctionWithGalaxySelectParam` *A variation on functionToGalaxify that takes a multiple-choice option using the GalaxySelectParam class.*

Description

A variation on functionToGalaxify that takes a multiple-choice option using the GalaxySelectParam class.

Usage

```
testFunctionWithGalaxySelectParam(inputfile1 = GalaxyInputFile(),
  inputfile2 = GalaxyInputFile(),
  plotTitle = GalaxySelectParam(c(TitleA = "A"), force_select = TRUE),
  plotSubTitle = GalaxyCharacterParam("My subtitle"),
  outputfile1 = GalaxyOutput("mydata", "csv"),
  outputfile2 = GalaxyOutput("myplot", "pdf"))
```

Arguments

inputfile1	the first matrix
inputfile2	the second matrix
plotTitle	the plot title
plotSubTitle	the plot subtitle
outputfile1	the csv output file
outputfile2	the pdf output file

Details

There are no details.

---

testFunctionWithSelect

A variation on functionToGalaxify that takes a multiple-choice option.

Description

A variation on functionToGalaxify that takes a multiple-choice option.

Usage

```
testFunctionWithSelect(inputfile1 = GalaxyInputFile(),
  inputfile2 = GalaxyInputFile(),
 plotTitle = GalaxyCharacterParam(c(TitleA = "A", TitleB = "B"), force_select = TRUE),
  plotSubTitle = GalaxyCharacterParam("My subtitle"),
  outputfile1 = GalaxyOutput("mydata", "csv"),
  outputfile2 = GalaxyOutput("myplot", "pdf"))
```

Arguments

inputfile1	the first matrix
inputfile2	the second matrix
plotTitle	the plot title
plotSubTitle	the plot subtitle
outputfile1	the csv output file
outputfile2	the pdf output file

Details

There are no details.

Index