Package 'Rcwl'

October 17, 2020

Title Wrap Command Tools and Pipelines Using CWL

Version 1.4.8

Description The package can be a simple and user-friendly way to manage command line tools and build data analysis pipelines in R using Common Workflow Language (CWL).

Depends R (>= 3.6), yaml, methods, S4Vectors

Imports utils, stats, BiocParallel, batchtools, DiagrammeR, shiny, R.utils, codetools

License GPL-2 | file LICENSE

Encoding UTF-8

LazyData true

Suggests testthat, knitr, rmarkdown, BiocStyle

VignetteBuilder knitr

RoxygenNote 7.1.1

biocViews Software, WorkflowStep, ImmunoOncology

SystemRequirements python (>= 2.7), cwltool (>= 1.0.2018)

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

git_branch RELEASE_3_11

git_last_commit 0845e17

git_last_commit_date 2020-08-10

Date/Publication 2020-10-16

Author Qiang Hu [aut, cre], Qian Liu [aut]

Maintainer Qiang Hu <qiang.hu@roswellpark.org>

R topics documented:

`+,cwlStepParam,stepParam-method`
Pipeline

Description

To build a pipeline by connecting multiple ‘stepParam‘ to a ‘cwlStepParam‘ object.

Usage

```
## S4 method for signature 'cwlStepParam,stepParam'
e1 + e2
```

Arguments

`e1`	A ‘cwlStepParam‘ object.
`e2`	A ‘stepParam‘ object.

Value

A ‘cwlStepParam‘ object.

See Also

`cwlStepParam`

cwlParam-class *Parameters for CWL*

Description

The main CWL parameter class and constructor for command tools. More details: https://www.commonwl.org/v1.0/Com

Usage

```
cwlParam(
  cwlVersion = "v1.0",
  cwlClass = "CommandLineTool",
  baseCommand = character(),
  requirements = list(),
  hints = list(),
  arguments = list(),
  id = character(),
  label = character(),
  inputs = InputParamList(),
  outputs = OutputParamList(),
  stdout = character(),
  expression = character(),
  extensions = list()
)
```

Arguments

cwlVersion	CWL version
cwlClass	"CommandLineTool"
baseCommand	Specifies the program or R function to execute
requirements	A list of Requirement lists that apply to either the runtime environment or the workflow engine.
hints	Any or a list for the workflow engine.
arguments	Command line bindings which are not directly associated with input parameters.
id	The unique identifier for this process object.
label	A short, human-readable label of this process object.
inputs	A object of 'InputParamList'.
outputs	A object of 'OutputParamList'.
stdout	Capture the command's standard output stream to a file written to the designated output directory.
expression	Javascripts for ExpressionTool class.
extensions	A list of extensions and metadata

Details

https://www.commonwl.org/v1.0/CommandLineTool.html

Value

A ‘cwlParam‘ class object.

Examples

```
input1 <- InputParam(id = "sth")
echo <- cwlParam(baseCommand = "echo", inputs = InputParamList(input1))
```

`cwlShiny` *cwlShiny*

Description

Function to generate shiny app automaticlly for a ‘cwlParam‘ object.

Usage

```
cwlShiny(cwl, inputList = list(), upload = FALSE, ...)
```

Arguments

`cwl`	A cwlParam object.
`inputList`	a list of choices for the inputs of cwl object. The name of the list must match the inputs of the cwl object.
`upload`	Whether to upload file. If FALSE, the upload field will be text input (file path) instead of file input.
`...`	More options for ‘runCWL‘.

Value

A shiny webapp.

Examples

```
input1 <- InputParam(id = "sth")
echo <- cwlParam(baseCommand = "echo", inputs = InputParamList(input1))
echoApp <- cwlShiny(echo)
```

cwlStepParam-class *cwlStepParam*

Description

A workflow steps paramter, which connect multiple command line steps into a workflow. More details: stepInParamList.

Usage

```
cwlStepParam(
  cwlVersion = "v1.0",
  cwlClass = "Workflow",
  requirements = list(),
  id = character(),
  hints = list(),
  arguments = list(),
  extensions = list(),
  inputs = InputParamList(),
  outputs = OutputParamList(),
  stdout = character(),
  steps = stepParamList()
)
```

Arguments

cwlVersion	CWL version
cwlClass	"Workflow".
requirements	Requirements that apply to either the runtime environment or the workflow engine.
id	The unique identifier for this process object.
hints	Any or a list for the workflow engine.
arguments	Command line bindings which are not directly associated with input parameters.
extensions	A list of extensions and metadata.
inputs	A object of 'InputParamList'.
outputs	A object of 'OutputParamList'.
stdout	Capture the command's standard output stream to a file written to the designated output directory.
steps	A list of 'stepParamList'.

Value

An object of class 'cwlStepParam'.

Examples

```
input1 <- InputParam(id = "sth")
echo1 <- cwlParam(baseCommand = "echo",
                  inputs = InputParamList(input1))
input2 <- InputParam(id = "sthout", type = "File")
echo2 <- cwlParam(baseCommand = "echo",
                  inputs = InputParamList(input2),
                  stdout = "out.txt")
i1 <- InputParam(id = "sth")
o1 <- OutputParam(id = "out", type = "File", outputSource = "echo2/output")
wf <- cwlStepParam(inputs = InputParamList(i1),
                   outputs = OutputParamList(o1))
s1 <- Step(id = "echo1", run = echo1, In = list(sth = "sth"))
s2 <- Step(id = "echo2", run = echo2, In = list(sthout = "echo1/output"))
wf <- wf + s1 + s2
```

`cwlVersion` *cwlParam methods*

Description

cwlParam methods

cwlVersion CWL document version

cwlClass

cwlClass

baseCommand

baseCommand

arguments

arguments

hints

hints

requirements

requirements

stdout of cwlParam

stdout of cwlParam

Extensions and metadata of cwlParam

Usage

```
cwlVersion(cwl)

cwlVersion(cwl) <- value

cwlClass(cwl)

cwlClass(cwl) <- value
```

```
baseCommand(cwl)

baseCommand(cwl) <- value

arguments(cwl, step = NULL)

arguments(cwl, step = NULL) <- value

hints(cwl)

hints(cwl) <- value

requirements(cwl)

requirements(cwl) <- value

stdOut(cwl)

stdOut(cwl) <- value

extensions(cwl)

extensions(cwl) <- value
```

Arguments

cwl	A ‘cwlParam‘ object.
value	Assign value to the ‘cwlParam‘ object.
step	To specifiy a step ID when ‘cwl‘ is a workflow. It can be multiple levels of steps separated by "/" for nested workflow.

Value

cwlVersion: cwl version

cwlClass: CWL Class

baseCommand: CWL baseCommand

arguments: CWL arguments

hints: CWL hints

requirements: CWL requirments

stdOut: CWL stdout

extensions: A list of extensions or metadata

InputArrayParam-class *InputArrayParam*

Description

Parameters for array inputs. To specify an array parameter, the array definition is nested under the type field with 'type: array' and items defining the valid data types that may appear in the array. More details: https://www.commonwl.org/v1.0/CommandLineTool.html#CommandInputArraySchema

Usage

```
InputArrayParam(
  label = "",
  type = "array",
  items = character(),
  prefix = "",
  separate = TRUE,
  itemSeparator = character(),
  valueFrom = character()
)
```

Arguments

label	A short description for this object
type	Must be "array".
items	Defines the type of the array elements.
prefix	Command line prefix to add before the value.
separate	If true (default), then the prefix and value must be added as separate command line arguments; if false, prefix and value must be concatenated into a single command line argument.
itemSeparator	Join the array elements into a single string with separator.
valueFrom	String or Expression.

Value

An object of class 'InputArrayParam'.

Examples

```
InputArrayParam(items = "string", prefix="-B=", separate = FALSE)
```

InputParam-class *Input parameters InputParam*

Description

parameter for a command tool. More details: https://www.commonwl.org/v1.0/CommandLineTool.html#CommandInput

Usage

```
InputParam(
  id,
  label = "",
  type = "string",
  doc = character(),
  secondaryFiles = character(),
  streamable = logical(),
  format = character(),
  loadListing = character(),
```

```
  loadContents = logical(),
  position = 0L,
  prefix = "",
  separate = TRUE,
  itemSeparator = character(),
  valueFrom = character(),
  shellQuote = logical(),
  default = character(),
  value = character()
)

## S4 method for signature 'cwlParam'
x$name

## S4 replacement method for signature 'cwlParam'
x$name <- value
```

Arguments

id	The unique identifier for this parameter object.
label	A short, human-readable label of this object.
type	valid types of data that may be assigned to this parameter.
doc	Optional. This argument takes an arbitrary documentation as a note for this object.
secondaryFiles	Only valid when type: File or is an array of items: File. Provides a pattern or expression specifying files or directories that must be included alongside the primary file.
streamable	Only valid when type: File or is an array of items: File. A value of true indicates that the file is read or written sequentially without seeking.
format	Only valid when type: File or is an array of items: File.
loadListing	Only valid when type: Directory or is an array of items: Directory.
loadContents	Only valid when type: File or is an array of items: File.
position	The position for this parameter.
prefix	Command line prefix to add before the value.
separate	If true (default), then the prefix and value must be added as separate command line arguments; if false, prefix and value must be concatenated into a single command line argument.
itemSeparator	Join the array elements into a single string with the elements separated by by itemSeparator.
valueFrom	String or Expression.
shellQuote	If ShellCommandRequirement is in the requirements for the current command, this controls whether the value is quoted on the command line (default is true).
default	The default value for this parameter
value	Assigned value for this parameter
x	A ‘cwlParam‘ object.
name	One one of input list

Value

An object of class 'InputParam'.

Examples

```
input1 <- InputParam(id = "sth")
```

`InputParamList-class` *InputParamList*

Description

InputParamList

InputParamList A list of InputParam

inputs

Usage

```
InputParamList(...)

inputs(cwl)
```

Arguments

`...`	The InputParam objects.
`cwl`	A cwlParam object

Value

An object of class 'InputParamList'.

inputs: A list of 'InputParam'.

Examples

```
input1 <- InputParam(id = "sth")
InputParamList(input1)
## Inputs
input1 <- InputParam(id = "sth")
echo <- cwlParam(baseCommand = "echo", inputs = InputParamList(input1))
inputs(echo)
```

OutputArrayParam-class *Output array parameters*

Description

Parameters for array outputs. More details: https://www.commonwl.org/v1.0/CommandLineTool.html#CommandOutput

Usage

```
OutputArrayParam(
  label = character(),
  type = "array",
  items = character(),
  glob = character(),
  loadContents = logical(),
  outputEval = character()
)
```

Arguments

label	A short, human-readable label of this object.
type	Must be "array".
items	Defines the type of the array elements.
glob	Pattern to find files relative to the output directory.
loadContents	Read text from globbed file.
outputEval	Evaluate an expression to generate the output value.

Value

An object of class 'OutputArrayParam'.

Examples

```
b <- OutputParam(id = "b", type = OutputArrayParam(items = "File"), glob = "*.txt")
```

OutputParam-class *Output parameters*

Description

An output parameter for a Command Line Tool. More details: https://www.commonwl.org/v1.0/CommandLineTool.html

Usage

```
OutputParam(
  id = "output",
  label = character(),
  doc = character(),
  type = "stdout",
  format = character(),
  secondaryFiles = character(),
  streamable = logical(),
  glob = character(),
  loadContents = logical(),
  outputEval = character(),
  outputSource = character()
)
```

Arguments

id	The unique identifier for this parameter object.
label	A short, human-readable label of this object.
doc	A documentation string for this object, or an array of strings which should be concatenated.
type	Specify valid types of data that may be assigned to this parameter.
format	Only valid when type: File or is an array of items: File. This is the file format that will be assigned to the output File object.
secondaryFiles	Provides a pattern or expression specifying files or directories. Only valid when type: File or is an array of items: File.
streamable	A value of true indicates that the file is read or written sequentially without seeking. Only valid when type: File or is an array of items: File.
glob	Pattern to find files relative to the output directory.
loadContents	Read text from globbed file.
outputEval	Evaluate an expression to generate the output value.
outputSource	Specifies one or more workflow parameters that supply the value of to the output parameter.

Value

An object of class ‘OutputParam‘.

Examples

```
o1 <- OutputParam(id = "file", type = "File", glob = "*.txt")
```

OutputParamList-class *OutputParamList*

Description

OutputParamList

OutputParamList #' A list of InputParam

outputs The outputs of a cwlParam object

Usage

```
OutputParamList(out = OutputParam(), ...)

outputs(cwl)
```

Arguments

out	The default stdout parameter.
...	The InputParam objects.
cwl	A cwlParam object

Value

An object of class 'OutputParamList'.

outputs: A list of 'OutputParam'.

Examples

```
o1 <- OutputParam(id = "file", type = "File", glob = "*.txt")
OutputParamList(o1)
input1 <- InputParam(id = "sth")
echo <- cwlParam(baseCommand = "echo", inputs = InputParamList(input1))
outputs(echo)
```

plotCWL *plotCWL*

Description

Function to plot cwlStepParam object.

Usage

```
plotCWL(cwl, output = "graph", layout = "tree", ...)
```

Arguments

cwl	A cwlStepParam object to plot
output	A string specifying the output type. An option inherits from ‘render_graph‘ and can also be "mermaid".
layout	Layout from ‘render_graph‘.
...	other parameters from ‘mermaid‘ or ‘render_graph‘ function

Value

A workflow plot.

Examples

```
input1 <- InputParam(id = "sth")
echo1 <- cwlParam(baseCommand = "echo",
                  inputs = InputParamList(input1))
input2 <- InputParam(id = "sthout", type = "File")
echo2 <- cwlParam(baseCommand = "echo",
                  inputs = InputParamList(input2),
                  stdout = "out.txt")
i1 <- InputParam(id = "sth")
o1 <- OutputParam(id = "out", type = "File", outputSource = "echo2/output")
wf <- cwlStepParam(inputs = InputParamList(i1),
                   outputs = OutputParamList(o1))
s1 <- Step(id = "echo1", run = echo1, In = list(sth = "sth"))
s2 <- Step(id = "echo2", run = echo2, In = list(sthout = "echo1/output"))
wf <- wf + s1 + s2
plotCWL(wf)
```

Rcwl *Rcwl*

Description

An R package to wrap command line tools and build pipelines with Common Workflow Language.

See Also

cwlParam

runCWL

readCWL *Read CWL Function to read CWL command or workflow files.*

Description

Read CWL Function to read CWL command or workflow files.

Usage

```
readCWL(cwlfile)
```

Arguments

cwlfile	The cwl file to read.

Value

A object of class ‘cwlParam‘ or ‘cwlStepParam‘.

Examples

```
input1 <- InputParam(id = "sth")
echo <- cwlParam(baseCommand = "echo",
                 inputs = InputParamList(input1))
tf <- tempfile()
writeCWL(echo, tf)
readCWL(paste0(tf, ".cwl"))
```

requireDocker *requireDocker*

Description

requireDocker

requireJS

requireSoftware

InitialWorkDirRequirement

SubworkflowFeatureRequirement

ScatterFeatureRequirement

MultipleInputFeatureRequirement

StepInputExpressionRequirement

Usage

```
requireDocker(
  docker = NULL,
  Load = NULL,
  File = NULL,
  Import = NULL,
  ImageId = NULL,
  OutputDir = NULL
)

requireJS(expressionLib = list())

requireSoftware(packages = list())

requireInitialWorkDir(listing = list())

requireSubworkflow()

requireScatter()

requireMultipleInput()

requireStepInputExpression()
```

Arguments

docker	The docker pull address.
Load	dockerLoad
File	dockerFile
Import	dockerImport
ImageId	dockerImageId
OutputDir	dockerOutputDirectory
expressionLib	optional code
packages	The list of software to be configured.
listing	The list of files or directories.

Value

A DockerRequirement list

A InlineJavascriptRequirement list

A SoftwareRequirement list

A InitialWorkDirRequirement list

A SubworkflowFeatureRequirement list

A ScatterFeatureRequirement list

A MultipleInputFeatureRequirement list

A StepInputExpressionRequirement list

runCWL *run cwlParam*

Description

Execute a cwlParam object with assigned inputs.

Usage

```
runCWL(
  cwl,
  prefix = tempfile(),
  cwlRunner = "cwltool",
  cwlTemp = NULL,
  outdir = ".",
  cwlArgs = character(),
  stdout = TRUE,
  stderr = TRUE,
  showLog = FALSE,
  docker = TRUE,
  ...
)
```

Arguments

cwl	A ‘cwlParam‘ or ‘cwlStepParam‘ object.
prefix	The prefix of ‘cwl‘ and ‘yml‘ file to write.
cwlRunner	The path to the ‘cwltool‘ or ‘cwl-runner‘. If not exists, the cwltool package will be installed by ‘reticulate‘.
cwlTemp	Path to keep temporary files. If a directory path is given, the temporary files will be kept in the directory.
outdir	Output directory, default current directory.
cwlArgs	The arguments for ‘cwltool‘ or ‘cwl-runner‘. For example, "–debug" can work with ‘cwltool‘ to show debug information.
stdout	standard output from ‘system2‘.
stderr	standard error from ‘system2‘. By setting it to "", the detailed running logs will return directly.
showLog	Whether to show log details to standard out. i.e. stderr = "".
docker	Whether to use docker, or "sigularity" if use Singularity runtime to run container.
...	The other options from ‘writeCWL‘ and ‘system2‘.

Value

A list of outputs from tools and logs from cwltool.

Examples

```
input1 <- InputParam(id = "sth")
echo <- cwlParam(baseCommand = "echo",
                 inputs = InputParamList(input1))
echo$sth <- "Hello World!"
## res <- runCWL(echo)
```

---

runCWLBatch *run CWL with batchtools*

Description

run CWL with batchtools

Usage

```
runCWLBatch(
  cwl,
  outdir = getwd(),
  inputList,
  paramList = list(),
  BPPARAM = BatchtoolsParam(workers = lengths(inputList)[1]),
  ...
)
```

Arguments

cwl	A 'cwlParam' or 'cwlStepParam' object.
outdir	Directory to output results
inputList	An input list to run in parallel. The list names must be in the inputs of cwl. Jobs will be submitted in parallel for each element in the list. The output directory of each job will be made using the name of each element under the 'outdir'.
paramList	A parameter list for the cwl. The list names must be in the inputs of cwl.
BPPARAM	The options for 'BiocParallelParam'.
...	The options from runCWL.

Value

Results from computing nodes and logs from cwltool.

runs *runs*

Description

The function to access all runs of a cwlStepParam object

Usage

```
runs(object)
```

Arguments

object	A cwlStepParam object.

Value

cwlParam objects or paths of CWL file.

Examples

```
s1 <- cwlStepParam()
runs(s1)
```

short *short*

Description

The function to show short summary of cwlParam or cwlStepParam

Usage

```
short(object)
```

Arguments

object	An cwlParam or cwlStepParam object

Value

A short summary of an object of cwlParam or cwlStepParam.

Examples

```
s1 <- cwlStepParam()
short(s1)
```

Step *Step function*

Description

Function to assign value to ‘stepParam‘ object.

Usage

```
Step(
  id,
  run = cwlParam(),
  In = list(),
  scatter = character(),
  scatterMethod = character()
)
```

Arguments

id	The id of ‘stepParam‘ object.
run	A ‘cwlParam‘ object for command tool, or path to a CWL file.
In	one or two layes of list.
scatter	character or a list. The inputs to be scattered.
scatterMethod	required if scatter is an array of more than one element. It can be one of "dot-product", "nested_crossproduct" and "flat_crossproduct". Details: https://www.commonwl.org/v1.0/W

Value

An object of ‘stepParam‘.

See Also

cwlStepParam

stepInParam-class *stepInParam*

Description

The input parameter of a workflow step. More details: https://www.commonwl.org/v1.0/Workflow.html#WorkflowStepIn

Usage

```
stepInParam(
  id,
  source = character(),
  linkMerge = character(),
  default = character(),
  valueFrom = character()
)
```

Arguments

id	A unique identifier for this workflow input parameter.
source	Specifies one or more workflow parameters that will provide input to the underlying step parameter.
linkMerge	The method to use to merge multiple inbound links into a single array.
default	The default value for this parameter to use if either there is no source field, or the value produced by the source is null.
valueFrom	value from string or expression.

Value

An object of class 'stepInParam'.

Examples

```
s1 <- stepInParam(id = "s1")
```

---

stepInParamList-class *stepInParamList*

Description

stepInParamList

stepInParamList

Usage

```
stepInParamList(...)
```

Arguments

...	A list of 'stepInParam' objects.

Value

An object of class 'stepInParamList'.

Examples

```
s1 <- stepInParam(id = "s1")
stepInParamList(s1)
```

stepParam-class *stepParam*

Description

A workflow step parameters. More details: https://www.commonwl.org/v1.0/Workflow.html#WorkflowStep

Usage

```
stepParam(
  id,
  run = cwlParam(),
  In = stepInParamList(),
  Out = list(),
  scatter = character(),
  scatterMethod = character()
)
```

Arguments

id	The unique identifier for this workflow step.
run	A ‘cwlParam‘ object or the path of a cwl file.
In	A ‘stepInParamList‘.
Out	A list of outputs
scatter	character or a list. The inputs to be scattered.
scatterMethod	required if scatter is an array of more than one element. It can be one of "dot-product", "nested_crossproduct" and "flat_crossproduct". Details: https://www.commonwl.org/v1.0/W

Value

An object of class ‘stepParam‘.

Examples

```
s1 <- stepParam(id = "s1")
```

stepParamList-class *stepParamList*

Description

stepParamList

stepParamList

Usage

```
stepParamList(...)
```

Arguments

...	A list of ‘stepParam‘.

Value

An object of class ‘stepParamList‘.

Examples

```
s1 <- stepParam(id = "s1")
stepParamList(s1)
```

---

steps *Steps*

Description

Function to extract step slots

Usage

```
steps(cwl)

steps(cwl) <- value
```

Arguments

cwl	A cwlStepParam object.
value	A list of steps.

Value

steps: A list of stepParam objects.

See Also

cwlStepParam

writeCWL *Write CWL*

Description

write ‘cwlParam‘ to cwl and yml.

Usage

```
writeCWL(cwl, prefix, docker = TRUE, ...)
```

Arguments

cwl	A ‘cwlParam‘ or ‘cwlStepParam‘ object.
prefix	The prefix of ‘cwl‘ and ‘yml‘ file to write.
docker	Whether to use docker.
...	Other options from ‘yaml::write_yaml‘.

Value

A CWL file and A YML file.

Examples

```
input1 <- InputParam(id = "sth")
echo <- cwlParam(baseCommand = "echo",
                 inputs = InputParamList(input1))
tf <- tempfile()
writeCWL(echo, tf)
```

Index