

# Package ‘Rcollectl’

October 31, 2024

**Title** Help use collectl with R in Linux, to measure resource consumption in R processes

**Version** 1.6.0

**Date** 2023-03-08

**Description** Provide functions to obtain instrumentation data on processes in a unix environment. Parse output of a collectl run. Vizualize aspects of system usage over time, with annotation.

**Imports** utils, ggplot2, lubridate, processx

**License** Artistic-2.0

**URL** <https://github.com/vjcitn/Rcollectl>

**BugReports** <https://support.bioconductor.org/t/Rcollectl>

**biocViews** Software, Infrastructure

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**SystemRequirements** collectl

**Suggests** knitr, BiocStyle, knitcitations, sessioninfo, rmarkdown, testthat, covr

**VignetteBuilder** knitr

**git\_url** <https://git.bioconductor.org/packages/Rcollectl>

**git\_branch** RELEASE\_3\_20

**git\_last\_commit** 75a1d4c

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2024-10-30

**Author** Vincent Carey [aut, cre] (<<https://orcid.org/0000-0003-4046-0063>>), Yubo Cheng [aut]

**Maintainer** Vincent Carey <stvjc@channing.harvard.edu>

## Contents

browse_units . . . . .	2
cl_exists . . . . .	2
cl_parse . . . . .	3
cl_result_path . . . . .	4
cl_start . . . . .	4
cl_stop . . . . .	5
cl_timestamp . . . . .	5
plot_usage . . . . .	6
print.Rcollectl_process . . . . .	7

<b>Index</b>	<b>8</b>
--------------	----------

---

browse_units	<i>browse a page defining units of collectl reporting</i>
--------------	---

---

### Description

browse a page defining units of collectl reporting

### Usage

```
browse_units(...)
```

### Arguments

... passed to [browseURL](#)

### Value

side effect is running browseURL

### Examples

```
if (interactive()) {
  browse_units()
}
```

---

cl_exists	<i>check for collectl availability</i>
-----------	--

---

### Description

check for collectl availability

### Usage

```
cl_exists()
```

**Value**

logical(1)

**Examples**

```
cl_exists()
```

---

cl_parse	<i>parse a collectl output – could be conditional on discovered call</i>
----------	--

---

**Description**

parse a collectl output – could be conditional on discovered call

**Usage**

```
cl_parse(path, tz = "EST", rescale_mem = TRUE)
```

**Arguments**

path	character(1) path to (possibly gzipped) collectl output
tz	character(1) POSIXct time zone code, defaults to "EST"
rescale_mem	logical(1) if TRUE, divide reported MEM_Used by 1000

**Value**

a data.frame

**Note**

A lubridate datetime is added as a column. The test file `demo_1123.tab.gz` is a collectl-generated report for a session ranging over 10 minutes, analyzing RNA-seq data on a multicore machine.

**Examples**

```
lk = cl_parse(system.file("demotab/demo_1123.tab.gz", package="Rcollectl"))
head(lk)
```

---

cl_result_path	<i>get full path to collectl report</i>
----------------	---

---

**Description**

get full path to collectl report

**Usage**

```
cl_result_path(proc)
```

**Arguments**

proc            an entity inheriting from "Rcollectl\_process" S3 class

**Value**

character(1) path to report

**Examples**

```
example(cl_start)
```

---

cl_start	<i>start collectl if possible</i>
----------	-----------------------------------

---

**Description**

start collectl if possible

**Usage**

```
cl_start(target = tempfile())
```

**Arguments**

target            character(1) path; destination of collectl report

**Value**

instance of Rcollectl\_process with components process (a processx R6 instance) and target (a file path where collectl results will be written)

**Examples**

```

if (cl_exists()) {
  zz = cl_start()
  Sys.sleep(2)
  print(zz)
  Sys.sleep(2)
  print(cl_result_path(zz))
  cl_stop(zz)
  Sys.sleep(2)
  zz$process$is_alive()
}

```

---

cl_stop	<i>stop collectl via processx interrupt</i>
---------	---

---

**Description**

stop collectl via processx interrupt

**Usage**

```
cl_stop(proc)
```

**Arguments**

proc                    an entity inheriting from "Rcollectl\_process" S3 class

**Value**

invisibly returns the input

**Examples**

```
example(cl_start)
```

---

cl_timestamp	<i>Functions to add time stamps to collectl output</i>
--------------	--

---

**Description**

Functions to add time stamps to collectl output

**Usage**

```
cl_timestamp(proc, step)
```

```
cl_timestamp_layer(arg)
```

```
cl_timestamp_label(arg, tz = "EST")
```

**Arguments**

proc	an entity inheriting from "Rcollectl_process" S3 class
step	character(1) name of step within a workflow
arg	proc (an entity inheriting from "Rcollectl_process" S3 class) or path to collectl output
tz	character(1) time zone code

**Value**

cl\_timestamp() returns a tab delimited text file

cl\_timestamp\_layer() and cl\_timestamp\_label() return objects that can be combined with ggplot.

**Examples**

```
id <- cl_start()
Sys.sleep(2)
cl_timestamp(id, "step1")
Sys.sleep(2)
Sys.sleep(2)
cl_timestamp(id, "step2")
Sys.sleep(2)
Sys.sleep(2)
cl_timestamp(id, "step3")
Sys.sleep(2)
cl_stop(id)
path <- cl_result_path(id)
plot_usage(cl_parse(path)) +
  cl_timestamp_layer(path) +
  cl_timestamp_label(path) +
  ggplot2::theme(axis.text.x = ggplot2::element_text(angle = 90, vjust = 0.5, hjust=1))
```

---

plot\_usage

*elementary display of usage data from collectl*

---

**Description**

elementary display of usage data from collectl

**Usage**

```
plot_usage(x)
```

**Arguments**

x	output of cl_parse
---	--------------------

**Value**

ggplot with geom\_point and facet\_grid

### **Examples**

```
lk = cl_parse(system.file("demotab/demo_1123.tab.gz", package="Rcollectl"))
plot_usage(lk)
```

---

`print.Rcollectl_process`

*print method for Rcollectl process*

---

### **Description**

print method for Rcollectl process

### **Usage**

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

### **Arguments**

x	an entity inheriting from "Rcollectl_process" S3 class
...	not used

### **Value**

invisibly returns the input

### **Examples**

```
example(cl_start)
```

# Index

`browse_units`, 2  
`browseURL`, 2

`cl_exists`, 2  
`cl_parse`, 3  
`cl_result_path`, 4  
`cl_start`, 4  
`cl_stop`, 5  
`cl_timestamp`, 5  
`cl_timestamp_label (cl_timestamp)`, 5  
`cl_timestamp_layer (cl_timestamp)`, 5

`plot_usage`, 6  
`print.Rcollectl_process`, 7