

# Package: ratelimitr (via r-universe)

September 17, 2024

**Type** Package

**Title** Rate Limiting for R

**Version** 0.4.1

**Author** Tarak Shah

**Maintainer** Tarak Shah <tarak.shah@gmail.com>

**Description** Allows to limit the rate at which one or more functions can be called.

**License** MIT + file LICENSE

**LazyData** TRUE

**RoxygenNote** 6.1.0

**Suggests** testthat, microbenchmark, knitr, rmarkdown, covr

**Imports** assertthat

**VignetteBuilder** knitr

**URL** <https://github.com/tarakc02/ratelimitr>

**BugReports** <https://github.com/tarakc02/ratelimitr/issues>

**Repository** <https://tarakc02.r-universe.dev>

**RemoteUrl** <https://github.com/tarakc02/ratelimitr>

**RemoteRef** HEAD

**RemoteSha** e0d3a1e7b72288ba0916448d49b5ab8d2a2a5bd9

## Contents

get_function . . . . .	2
get_precision . . . . .	2
get_rates . . . . .	2
limit_rate . . . . .	3
rate . . . . .	4
reset . . . . .	4
UPDATE_RATE . . . . .	5

<b>Index</b>	<b>6</b>
--------------	----------

---

get_function	<i>Access the original function from a rate limited function</i>
--------------	--

---

**Description**

Access the original function from a rate limited function

**Usage**

```
get_function(f)
```

**Arguments**

f	A rate limited function or group of functions
---	---

---

get_precision	<i>Access the rate limit precision</i>
---------------	--

---

**Description**

Access the rate limit precision

**Usage**

```
get_precision(f)
```

**Arguments**

f	A rate limited function or group of functions
---	---

---

get_rates	<i>Access the rate limit(s) of a rate limited function</i>
-----------	--

---

**Description**

Access the rate limit(s) of a rate limited function

**Usage**

```
get_rates(f)
```

**Arguments**

f	A rate limited function or group of functions
---	---

---

limit_rate	<i>Limit the rate at which a function will execute</i>
------------	--

---

## Description

Limit the rate at which a function will execute

## Usage

```
limit_rate(f, ..., precision = 60)

## S3 method for class 'list'
limit_rate(f, ..., precision = 60)

## S3 method for class 'function_list'
limit_rate(f, ..., precision = 60)

## S3 method for class 'function'
limit_rate(f, ..., precision = 60)
```

## Arguments

f	A single function to be rate-limited, or a named list of functions
...	One or more rates, created using <a href="#">rate</a>
precision	The precision with which time intervals can be measured, in hertz

## Value

If f is a single function, then a new function with the same signature and (eventual) behavior as the original function, but rate limited. If f is a named list of functions, then a new list of functions with the same names and signatures, but collectively bound by a shared rate limit.

## See Also

[rate](#), [UPDATE\\_RATE](#)

## Examples

```
## limiting a single function
f <- limit_rate(Sys.time, rate(n = 5, period = .1))
res <- replicate(10, f())
## show the elapsed time between each function call:
round(res[-1] - head(res, -1), 3)

## for multiple functions, make sure the list is named:
f <- function() 1
g <- function() 2
limited <- limit_rate(list(f = f, g = g), rate(n = 1, period = .1))
```

```
system.time({limited$f(); limited$g()})
```

---

rate	<i>Create a new rate</i>
------	--------------------------

---

### Description

Create a new rate

### Usage

```
rate(n, period)
```

### Arguments

n	Number of allowed events within a period
period	Length (in seconds) of measurement period

### See Also

[limit\\_rate](#)

### Examples

```
## a function
f <- function() NULL

## limit f to 10 calls per second
limited_f <- limit_rate(f, rate(n = 10, period = 1))
```

---

reset	<i>Re-create a rate-limited function</i>
-------	--

---

### Description

This function does not modify the original rate-limited function, instead it returns a new function with the same rate limits (but no memory of prior function calls).

### Usage

```
reset(f)
```

### Arguments

f	A rate-limited function or group of functions
---	---

**Examples**

```
f <- function() NULL
f_lim <- limit_rate(f, rate(n = 1, period = .1))
f_lim() ## the next call to f_lim will trigger the rate limit

f_lim2 <- reset(f_lim) ## but f_lim2 has a fresh start

## f_lim2 behaves as though no calls have been made
system.time(f_lim2())

## while f_lim is still constrained
system.time(f_lim())
```

---

UPDATE\_RATE

*Update the rate limit of an existing rate limited function*


---

**Description**

UPDATE\_RATE modifies an existing rate-limited function in place, changing the rate limits without otherwise altering the function's behavior. When a rate limited function has its rate limits updated, the previous rate limits and any calls that would have counted against those rate limits are immediately forgotten, and only the new rate limits are obeyed going forward.

**Usage**

```
UPDATE_RATE(lf, ..., precision = 60)
```

**Arguments**

lf	A rate-limited function or group of functions
...	One or more rates, created using <a href="#">rate</a>
precision	The precision with which time intervals can be measured, in hertz

**Examples**

```
f <- function() NULL
f_lim <- limit_rate(f, rate(n = 1, period = .1))

# update the rate limits to 2 calls per .1 second
UPDATE_RATE(f_lim, rate(n = 2, period = .1))
```

# Index

`get_function`, 2  
`get_precision`, 2  
`get_rates`, 2

`limit_rate`, 3, 4

`rate`, 3, 4, 5  
`reset`, 4

`UPDATE_RATE`, 3, 5