Package 'ulid'

June 4, 2024

Type Package

Title Generate Universally Unique 'Lexicographically' 'Sortable' Identifiers

Version 0.4.0

Date 2024-06-03

Description Universally unique identifiers ('UUIDs') can be sub-optimal for many uses-cases because they are not the most character efficient way of encoding 128 bits of randomness; v1/v2 versions are impractical in many environments, as they require access to a unique, stable MAC address; v3/v5 versions require a unique seed and produce randomly distributed IDs, which can cause fragmentation in many data structures; v4 provides no other information than randomness which can cause fragmentation in many data structures. Providing an alternative, 'ULIDs' (<https://github.com/ulid/spec>) have 128-bit compatibility with 'UUID', 1.21e+24 unique 'ULIDs' per millisecond, support standard (text) sorting, canonically encoded as a 26 character string, as opposed to the 36 character 'UUID', use 'base32' encoding for better efficiency and readability (5 bits per character), are case insensitive, have no special characters (i.e. are 'URL' safe) and have a monotonic sort order (correctly detects and handles the same millisecond).

URL https://github.com/eddelbuettel/ulid

BugReports https://github.com/eddelbuettel/ulid/issues

NeedsCompilation yes

Encoding UTF-8

License MIT + file LICENSE

Suggests tinytest

Imports Rcpp

LinkingTo Rcpp

RoxygenNote 7.3.1

Author Bob Rudis [aut] (<https://orcid.org/0000-0001-5670-2640>), Suyash Verma [aut] (ULID C++ <https://github.com/suyash/ulid/>), Dirk Eddelbuettel [cre] (<https://orcid.org/0000-0001-6419-907X>)

Maintainer Dirk Eddelbuettel <edd@debian.org>

Repository CRAN

Date/Publication 2024-06-04 09:52:31 UTC

R topics documented:

`generate` *Generate ULID*

Description

`generate()` generates a new Universally Unique Lexicographically Sortable Identifier. Several aliases are available for convience and backwards-compatibility.

This function generates a new Universally Unique Lexicographically Sortable Identifier from a vector of `POSIXct` timestamps.

As described in the ulid specification repo, and slightly edited here, UUID use can be suboptimal for many uses-cases because:(grifted from `https://github.com/ulid/spec`)

UUID can be suboptimal for many uses-cases because:

- It isn't the most character efficient way of encoding 128 bits of randomness
- UUID v1/v2 is impractical in many environments, as it requires access to a unique, stable MAC address
- UUID v3/v5 requires a unique seed and produces randomly distributed IDs, which can cause fragmentation in many data structures
- UUID v4 provides no other information than randomness which can cause fragmentation in many data structures

Instead, an alternative is proposed in ULID:

```
ulid() // 01ARZ3NDEKTSV4RRFFQ69G5FAV
```

with the following properties:

- 128-bit compatibility with UUID
- 1.21e+24 unique ULIDs per millisecond
- Lexicographically sortable!
- Canonically encoded as a 26 character string, as opposed to the 36 character UUID
- Uses Crockford's base32 for better efficiency and readability (5 bits per character)
- Case insensitive
- No special characters (URL safe)
- Monotonic sort order (correctly detects and handles the same millisecond)

```
01AN4Z07BY      79KA1307SR9X4MV3

|----------|    |----------------|
  Timestamp          Randomness
   48bits              80bits
```

Components

Timestamp

- 48 bit integer
- UNIX-time in milliseconds
- Will not run out of space until the year 10889 AD.

Randomness

- 80 bits
- Cryptographically secure source of randomness, if possible

Sorting

The left-most character must be sorted first, and the right-most character sorted last (lexical order). The default ASCII character set must be used. Within the same millisecond, sort order is not guaranteed.

Usage

```
generate(n = 1L)

unmarshal(ulids)

ts_generate(tsv)

ulid(n = 1L)

ulid_generate(n = 1L)

ULIDgenerate(n = 1L)
```

Arguments

`n`	number of id's to generate (default = 1)
`ulids`	character ULIDs (e.g. created with `generate()`)
`tsv`	vector of `POSIXct` values

Details

Note that up until release 0.3.1, the implementations had limitations that resulted in second rather than millisecond resolution. This has been addressed for release 0.4.0 and is now supported as expected.

Value

A `data.frame` with two columns `ts` and `rnd`.

Author(s)

Bob Rudis (bob@rud.is) wrote the package based on `ulid` C++ library by Suyash Verma.

Dirk Eddelbuettel now maintains the package.

See Also

The ulid specification provides the reference.

Examples

```
ULIDgenerate()
unmarshal(generate())
ts_generate(as.POSIXct("2017-11-01 15:00:00", origin="1970-01-01"))
```

Index