

Package ‘hashids’

May 8, 2026

Title Generate Short Unique YouTube-Like IDs (Hashes) from Integers

Version 0.9.0

Description An R port of the hashids library. hashids generates YouTube-like hashes from integers or vector of integers. Hashes generated from integers are relatively short, unique and non-sequential. hashids can be used to generate unique ids for URLs and hide database row numbers from the user. By default hashids will avoid generating common English curse-words by preventing certain letters being next to each other. hashids are not one-way: it is easy to encode an integer to a hashid and decode a hashid back into an integer.

URL <https://github.com/ALShum/hashids-r/>, <http://hashids.org>

BugReports <https://github.com/ALShum/hashids-r/issues>

Depends R (>= 3.2.2)

License MIT + file LICENSE

LazyData true

Suggests testthat

NeedsCompilation no

Author Alex Shum [aut, cre],
Ivan Akimov [aut] (original author of hashids -- implemented in javascript),
David Aurelio [ctb] (implemented hashids in python 2 and 3)

Maintainer Alex Shum <Alex@ALShum.com>

Repository CRAN

Date/Publication 2015-09-11 10:10:26

Contents

ascii_val	2
base16_to_dec	2
decode	3
decode_hex	3
dec_to_base16	4
encode	4

encode_hex	5
enforce_min_length	5
hash	6
hashid_defaults	6
hashid_settings	7
shuffle	7
split	8
unhash	8

Index	9
--------------	----------

ascii_val	<i>Calculate the ascii value number of a character</i>
-----------	--

Description

Calculate the ascii value number of a character

Usage

```
ascii_val(char)
```

Arguments

char	character
------	-----------

Value

ascii value integer

base16_to_dec	<i>Converts a base 16 string to a base 10 number. Because I couldn't get base R functions to work for big hex numbers.</i>
---------------	--

Description

Converts a base 16 string to a base 10 number. Because I couldn't get base R functions to work for big hex numbers.

Usage

```
base16_to_dec(str_16)
```

Arguments

str_16	base 16 number as a string.
--------	-----------------------------

Value

base 10 integer.

decode	<i>Decodes a hashid into the original integer or integer vector</i>
--------	---

Description

Decodes a hashid into the original integer or integer vector

Usage

```
decode(hash_str, settings)
```

Arguments

hash_str	hashid string to decode into integer or integer vector
settings	Settings list generated by hashid_settings

Value

integer or integer vector

decode_hex	<i>Decodes a hashid into the original hexadecimal number</i>
------------	--

Description

Decodes a hashid into the original hexadecimal number

Usage

```
decode_hex(hashid, settings)
```

Arguments

hashid	hashid to decode
settings	Settings list generated by hashid_settings

Value

hexadecimal number as a string

dec_to_base16	<i>Converts a base 10 number to base 16 number. Because I couldn't get R's as.hexmode() to work for big integers.</i>
---------------	---

Description

Converts a base 10 number to base 16 number. Because I couldn't get R's as.hexmode() to work for big integers.

Usage

```
dec_to_base16(dec)
```

Arguments

dec	base 10 integer
-----	-----------------

Value

base 16 number as a string

encode	<i>Encodes an integer or integer vector into a hashid string. All numbers must be non-negative integers.</i>
--------	--

Description

Encodes an integer or integer vector into a hashid string. All numbers must be non-negative integers.

Usage

```
encode(int, settings)
```

Arguments

int	Integer or integer vector to encode
settings	Settings list generated by hashid_settings

Value

hashid string

encode_hex	<i>Encodes a hexadecimal number into a hashid</i>
------------	---

Description

Encodes a hexadecimal number into a hashid

Usage

```
encode_hex(hex_str, settings)
```

Arguments

hex_str	Hexadecimal number as string
settings	Settings list generated by hashid_settings

Value

hashid string

enforce_min_length	<i>Enforces hashid minimum length by padding the hashid with additional characters.</i>
--------------------	---

Description

Enforces hashid minimum length by padding the hashid with additional characters.

Usage

```
enforce_min_length(encoded, min_length, alphabet, guards, values_hash)
```

Arguments

encoded	encoded hashid
min_length	minimum length required for hashid
alphabet	set of letters used to generate hashid
guards	set of guards used to generate hashid
values_hash	value hashed used to select guard characters

Value

hashid with padded characters to insure minimum length

hash	<i>Maps an integer to a string. Generated string will be inversely proportional to alphabet length.</i>
------	---

Description

Maps an integer to a string. Generated string will be inversely proportional to alphabet length.

Usage

```
hash(number, alphabet)
```

Arguments

number	Integer to hash
alphabet	Possible letters for string.

Value

hashed string

hashid_defaults	<i>Default Values for hashid settings</i>
-----------------	---

Description

Default alphabet, separators, and ratio of character separators and guards for hashid

Usage

```
DEFAULT_ALPHABET
```

```
DEFAULT_SEPS
```

```
RATIO_SEPARATORS
```

```
RATIO_GUARDS
```

Format

```
chr "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890"
```

Source

<http://www.hashids.org>

hashid_settings	<i>A function to create a hashid settings list.</i>
-----------------	---

Description

A function to create a hashid settings list.

Usage

```
hashid_settings(salt, min_length = 0, alphabet = DEFAULT_ALPHABET,  
                sep = DEFAULT_SEPS)
```

Arguments

salt	An additional string to make hashids more unique.
min_length	Minimum length for hashid.
alphabet	String of characters for hashid.
sep	String of characters to use as separators.

Value

A list of parameters used in encoding and decoding.

shuffle	<i>Permutes the characters in a string based on an inputted salt string.</i>
---------	--

Description

Permutes the characters in a string based on an inputted salt string.

Usage

```
shuffle(string, salt)
```

Arguments

string	String to be permuted
salt	cryptograph salt string that is used to permute strings

Value

shuffled string

split	<i>Splits a string based on a set of splitting characters</i>
-------	---

Description

Splits a string based on a set of splitting characters

Usage

```
split(string, splitters)
```

Arguments

string	String to split
splitters	set of splitting characters as a string

Value

split vector of characters

unhash	<i>Unhashes a string to an integer based on alphabet.</i>
--------	---

Description

Unhashes a string to an integer based on alphabet.

Usage

```
unhash(hash, alphabet)
```

Arguments

hashed	String to unhash
alphabet	Set of letters used for hashing

Value

Unhashed integer

Index

* datasets

- hashid_defaults, 6
- ascii_val, 2
- base16_to_dec, 2
- dec_to_base16, 4
- decode, 3
- decode_hex, 3
- DEFAULT_ALPHABET (hashid_defaults), 6
- DEFAULT_SEPS (hashid_defaults), 6
- encode, 4
- encode_hex, 5
- enforce_min_length, 5
- hash, 6
- hashid_defaults, 6
- hashid_settings, 7
- RATIO_GUARDS (hashid_defaults), 6
- RATIO_SEPARATORS (hashid_defaults), 6
- shuffle, 7
- split, 8
- unhash, 8