

# Package ‘phonics’

October 14, 2022

**Type** Package

**Title** Phonetic Spelling Algorithms

**Version** 1.3.10

**Date** 2021-7-11

**Encoding** UTF-8

**URL** <https://jameshoward.us/phonics-in-r/>

**BugReports** <https://github.com/k3jph/phonics-in-r/issues>

**Description** Provides a collection of phonetic algorithms including Soundex, Metaphone, NYSIIS, Caverphone, and others. The package is documented in <[doi:10.18637/jss.v095.i08](https://doi.org/10.18637/jss.v095.i08)>.

**License** BSD\_2\_clause + file LICENSE

**Imports** Rcpp (>= 0.12.1), data.table

**Suggests** testthat, knitr, markdown, rmarkdown, devtools

**LinkingTo** Rcpp, BH

**RoxygenNote** 7.1.1

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** James Howard [aut, cre] (<<https://orcid.org/0000-0003-4530-1547>>),  
Kyle Haynes [ctb],  
Amanda Hood [ctb],  
Os Keyes [ctb]

**Maintainer** James Howard <jh@jameshoward.us>

**Repository** CRAN

**Date/Publication** 2021-07-11 21:30:02 UTC

## R topics documented:

caverphone	2
cologne	3

lein . . . . .	4
metaphone . . . . .	5
mra_encode . . . . .	6
nysiis . . . . .	8
onca . . . . .	9
phonex . . . . .	10
phonics . . . . .	11
rogerroot . . . . .	12
soundex . . . . .	14
statcan . . . . .	15

<b>Index</b>	<b>17</b>
--------------	-----------

---

caverphone	<i>Caverphone</i>
------------	-------------------

---

## Description

The Caverphone family of phonetic algorithms

## Usage

```
caverphone(word, maxCodeLen = NULL, modified = FALSE, clean = TRUE)
```

## Arguments

word	string or vector of strings to encode
maxCodeLen	maximum length of the resulting encodings, in characters
modified	if TRUE, use the Caverphone 2 algorithm
clean	if TRUE, return NA for unknown alphabetical characters

## Details

The variable `maxCodeLen` is the limit on how long the returned Caverphone code should be. The default is 6, unless `modified` is set to TRUE, then the default is 10.

The variable `modified` directs `caverphone` to use the `Caverphone2` method, instead of the original.

The `caverphone` algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z.". Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Û," may be permissible in the current locale but are unknown to `caverphone`. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, `caverphone` attempts to process the strings. The default is TRUE.

## Value

the Caverphone encoded character vector

## References

David Hood, "Caverphone: Phonetic matching algorithm," Technical Paper CTP060902, University of Otago, New Zealand, 2002.

David Hood, "Caverphone Revisited," Technical Paper CTP150804 University of Otago, New Zealand, 2004.

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

## See Also

Other phonics: [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

## Examples

```
caverphone("William")
caverphone(c("Peter", "Peady"), modified = TRUE)
caverphone("Stevenson", maxCodeLen = 4)
```

---

cologne

*Cologne Phonetic Name Coding*

---

## Description

The Cologne phonetic name coding procedure.

## Usage

```
cologne(word, maxCodeLen = NULL, clean = TRUE)
```

## Arguments

word	string or vector of strings to encode
maxCodeLen	maximum length of the resulting encodings, in characters
clean	if TRUE, return NA for unknown alphabetical characters

## Details

The variable `word` is the name to be encoded. The variable `maxCodeLen` is the limit on how long the returned name code should be. The default is 4.

The `cologne` algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z," "Ä," "Ö," "Ü," and "ß." Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "ç," may be permissible in the current locale but are unknown to `cologne`. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, `cologne` attempts to process the strings. The default is TRUE.

**Value**

the Cologne encoded character vector

**References**

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

Hans Joachim Postel. "Die Koelner Phonetik. Ein Verfahren zur Identifizierung von Personennamen auf der Grundlage der Gestaltanalyse." *IBM-Nachrichten* 19. Jahrgang, 1969, p. 925-931.

**See Also**

Other phonics: [caverphone\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

**Examples**

```
cologne("William")
cologne(c("Peter", "Peedy"))
cologne("Stevenson", maxCodeLen = 8)
```

---

lein

*Lein Name Coding*


---

**Description**

The Lein name coding procedure.

**Usage**

```
lein(word, maxCodeLen = 4, clean = TRUE)
```

**Arguments**

word	string or vector of strings to encode
maxCodeLen	maximum length of the resulting encodings, in characters
clean	if TRUE, return NA for unknown alphabetical characters

**Details**

The variable `word` is the name to be encoded. The variable `maxCodeLen` is the limit on how long the returned name code should be. The default is 4.

The `lein` algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z.". Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Ü," may be permissible in the current locale but are unknown to `lein`. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, `lein` attempts to process the strings. The default is TRUE.

**Value**

the Lein encoded character vector

**References**

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

Billy T. Lynch and William L. Arends. "Selection of surname coding procedure for the SRS record linkage system." United States Department of Agriculture, Sample Survey Research Branch, Research Division, Washington, 1977.

**See Also**

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

**Examples**

```
lein("William")
lein(c("Peter", "Peady"))
lein("Stevenson", maxCodeLen = 8)
```

---

metaphone

*Generate phonetic versions of strings with Metaphone*

---

**Description**

The function `metaphone` phonetically encodes the given string using the metaphone algorithm.

**Usage**

```
metaphone(word, maxCodeLen = 10L, clean = TRUE)
```

**Arguments**

<code>word</code>	string or vector of strings to encode
<code>maxCodeLen</code>	maximum length of the resulting encodings, in characters
<code>clean</code>	if TRUE, return NA for unknown alphabetical characters

## Details

There is some discrepancy with respect to how the metaphone algorithm actually works. For instance, there is a version in the Java Apache Commons library. There is a version provided within PHP. These do not provide the same results. On the questionable theory that the implementation in PHP is probably more well known, this code should match it in output.

This implementation is based on a Javascript implementation which is itself based on the PHP internal implementation.

The variable `maxCodeLen` is the limit on how long the returned metaphone should be.

The metaphone algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z". Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Ü," may be permissible in the current locale but are unknown to metaphone. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, metaphone attempts to process the strings. The default is TRUE.

## Value

a character vector containing the metaphones of word, or an NA if the word value is NA

## References

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

## See Also

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

## Examples

```
metaphone("wheel")
metaphone(c("school", "benji"))
```

---

mra\_encode

*Match Rating Approach Encoder*

---

## Description

The Western Airlines matching rating approach name encoder

## Usage

```
mra_encode(word, clean = TRUE)
```

```
mra_compare(x, y)
```

## Arguments

word	string or vector of strings to encode
clean	if TRUE, return NA for unknown alphabetical characters
x	MRA-encoded character vector
y	MRA-encoded character vector

## Details

The variable `word` is the name to be encoded. The variable `maxCodeLen` is *not* supported in this algorithm encoder because the algorithm itself is dependent upon its six-character length. The variables `x` and `y` are MRA-encoded and are compared to each other using the MRA comparison specification.

The `mra_encode` algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z". Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Û," may be permissible in the current locale but are unknown to `mra_encode`. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, `mra_encode` attempts to process the strings. The default is TRUE.

## Value

The `mra_encode` function returns match rating approach encoded character vector. The `mra_compare` returns a boolean vector which is TRUE if `x` and `y` pass the MRA comparison test.

## References

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

G.B. Moore, J.L. Kuhns, J.L. Treffzs, and C.A. Montgomery, *Accessing Individual Records from Personal Data Files Using Nonunique Identifiers*, US National Institute of Standards and Technology, SP-500-2 (1977), p. 17.

## See Also

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

## Examples

```
mra_encode("William")
mra_encode(c("Peter", "Peady"))
mra_encode("Stevenson")
```

---

`nysiis`*New York State Identification and Intelligence System*

---

**Description**

The NYSIIS phonetic algorithm

**Usage**

```
nysiis(word, maxCodeLen = 6, modified = FALSE, clean = TRUE)
```

**Arguments**

<code>word</code>	string or vector of strings to encode
<code>maxCodeLen</code>	maximum length of the resulting encodings, in characters
<code>modified</code>	if TRUE, use the modified NYSIIS algorithm
<code>clean</code>	if TRUE, return NA for unknown alphabetical characters

**Details**

The `nysiis` function phonetically encodes the given string using the New York State Identification and Intelligence System (NYSIIS) algorithm. The algorithm is based on the implementation provided by Wikipedia and is implemented in pure R using regular expressions.

The variable `maxCodeLen` is the limit on how long the returned NYSIIS code should be. The default is 6.

The variable `modified` directs `nysiis` to use the modified method instead of the original.

The `nysiis` algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z". Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Ü," may be permissible in the current locale but are unknown to `nysiis`. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, `nysiis` attempts to process the strings. The default is TRUE.

**Value**

the NYSIIS encoded character vector

**References**

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

Robert L. Taft, *Name search techniques*, Bureau of Systems Development, Albany, New York, 1970.



**See Also**

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

**Examples**

```
nysiis("Robert")
nysiis("rupert")
nysiis(c("Alabama", "Alaska"), modified = TRUE)
nysiis("mississippi", 4)
```

---

onca

*Oxford Name Compression Algorithm*


---

**Description**

The Oxford Name Compression Algorithm name coding procedure

**Usage**

```
onca(word, maxCodeLen = 4, clean = TRUE, modified = FALSE, refined = FALSE)
```

**Arguments**

word	string or vector of strings to encode
maxCodeLen	maximum length of the resulting encodings, in characters
clean	if TRUE, return NA for unknown alphabetical characters
modified	if TRUE, use the modified nysiis function
refined	if TRUE, use the refinedSoundex function

**Details**

The variable `word` is the name to be encoded. The variable `maxCodeLen` is the limit on how long the returned name code should be. The default is 4.

The `onca` algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z". Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Ü," may be permissible in the current locale but are unknown to `onca`. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, `onca` attempts to process the strings. The default is TRUE.

**Value**

the ONCA encoded character vector

**References**

Gill, Leicester. "OX-LINK: the Oxford medical record linkage system." (1997).

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

**See Also**

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerrroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

**Examples**

```
onca("William")
onca(c("Peter", "Peady"))
onca("Stevenson", maxCodeLen = 8)
```

---

 phonex

*Phonex Name Coding*


---

**Description**

The Phonex name coding procedure.

**Usage**

```
phonex(word, maxCodeLen = 4, clean = TRUE)
```

**Arguments**

word	string or vector of strings to encode
maxCodeLen	maximum length of the resulting encodings, in characters
clean	if TRUE, return NA for unknown alphabetical characters

**Details**

The variable `word` is the name to be encoded. The variable `maxCodeLen` is the limit on how long the returned name code should be. The default is 4.

The phonex algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z," "Ä," "Ö," "Ü," and "ß." Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "ç," may be permissible in the current locale but are unknown to phonex. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, phonex attempts to process the strings. The default is TRUE.

**Value**

the Phonex encoded character vector

**References**

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

A.J. Lait and Brian Randell. "An assessment of name matching algorithms." Technical Report Series-University of Newcastle Upon Tyne Computing Science (1996).

**See Also**

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

**Examples**

```
phonex("William")
phonex(c("Peter", "Peady"))
phonex("Stevenson", maxCodeLen = 8)
```

---

phonics

*Phonetic Spelling Algorithms*

---

**Description**

The phonics package for R is designed to provide a variety of phonetic indexing algorithms in common and not-so-common use today. The algorithms generally reduce a string to a symbolic representation approximating the sound made by pronouncing the string. They can be used to match names, strings, and as a proxy for assorted string distance algorithms. The algorithm reduces a string to a symbolic representation approximating the sound. It can be used to match names, strings, and as a proxy for assorted string distance algorithms.

**Usage**

```
phonics(word, method, clean = TRUE)
```

**Arguments**

word	string or vector of strings to encode
method	vector of method names to use
clean	if TRUE, return NA for unknown alphabetical characters

## Details

The phonics package for R is designed to provide a variety of phonetic indexing algorithms in common and not-so-common use today. The algorithms generally reduce a string to a symbolic representation approximating the sound made by pronouncing the string. They can be used to match names, strings, and as a proxy for assorted string distance algorithms. The algorithm reduces a string to a symbolic representation approximating the sound. It can be used to match names, strings, and as a proxy for assorted string distance algorithms.

The variable `word` is a character string or a vector of character strings to be encoded.

Different phonetic algorithm are only defined for inputs over the limited alphabets, Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, phonics attempts to process the strings. The default is TRUE.

The method parameter should be a character vector containing one or more methods that should be used. The available list of methods is "caverphone", "caverphone.modified", "cologne", "lein", "metaphone", "nysiis", "nysiis.modified", "onca", "onca.modified", "onca.refined", "onca.modified.refined", "phonex", "rogerroot", "soundex", "soundex.refined", and "statcan".

## Value

Returns a data frame containing the phonetic spellings of the input for each method applied.

## References

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

## See Also

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

## Examples

```
phonics(c("Peter", "Peady"), c("soundex", "soundex.refined"))
```

---

rogerroot

*Roger Root Name Coding Procedure*

---

## Description

Provides the Roger Root name coding system

## Usage

```
rogerroot(word, maxCodeLen = 5, clean = TRUE)
```

## Arguments

word	string or vector of strings to encode
maxCodeLen	maximum length of the resulting encodings, in characters
clean	if TRUE, return NA for unknown alphabetical characters

## Details

The `rogerroot` function phonetically encodes the given string using the Roger Root algorithm. The variable `word` is a string or vector of strings to encode.

The variable `maxCodeLen` is the limit on how long the returned code should be. The default is 5.

The `rogerroot` algorithm is only defined for inputs over the standard English alphabet, *i.e.*, "A-Z". Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Ü," may be permissible in the current locale but are unknown to `rogerroot`. For inputs outside of its known range, the output is undefined and NA is returned and a warning is thrown. If `clean` is FALSE, `rogerroot` attempts to process the strings. The default is TRUE.

## Value

the Roger Root encoded character vector

## References

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

Robert L. Taft, *Name search techniques*, Bureau of Systems Development, Albany, New York, 1970.

## See Also

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [soundex\(\)](#), [statcan\(\)](#)

## Examples

```
rogerroot("William")
rogerroot(c("Peter", "Peady"))
rogerroot("Stevenson")
```

---

soundex	<i>Soundex</i>
---------	----------------

---

### Description

The Soundex phonetic algorithms

### Usage

```
soundex(word, maxCodeLen = 4L, clean = TRUE)
```

```
refinedSoundex(word, maxCodeLen = 10L, clean = TRUE)
```

### Arguments

word	string or vector of strings to encode
maxCodeLen	maximum length of the resulting encodings, in characters
clean	if TRUE, return NA for unknown alphabetical characters

### Details

The function `soundex` phonetically encodes the given string using the soundex algorithm. The function `refinedSoundex` uses Apache's refined soundex algorithm. Both implementations are loosely based on the Apache Commons Java editons.

The variable `maxCodeLen` is the limit on how long the returned soundex should be.

The `soundex` and `revisedSoundex` algorithms are only defined for inputs over the standard English alphabet, *i.e.*, "A-Z." Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Ü," may be permissible in the current locale but are unknown to `soundex` and `revisedSoundex`. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, `soundex` and `revisedSoundex` attempts to process the strings. The default is TRUE.

### Value

soundex encoded character vector

### Caveats

The `soundex` and `refinedSoundex` algorithms are only defined for inputs over the standard English alphabet, *i.e.*, "A-Z." For inputs outside this range, the output is undefined.

## References

Charles P. Bourne and Donald F. Ford, "A study of methods for systematically abbreviating English words and names," *Journal of the ACM*, vol. 8, no. 4 (1961), p. 538-552.

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

Howard B. Newcombe, James M. Kennedy, "Record linkage: making maximum use of the discriminating power of identifying information," *Communications of the ACM*, vol. 5, no. 11 (1962), p. 563-566.

## See Also

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [statcan\(\)](#)

## Examples

```
soundex("wheel")
soundex(c("school", "benji"))
```

---

 statcan

*Statistics Canada Name Coding*


---

## Description

The modified Statistics Canada name coding procedure

## Usage

```
statcan(word, maxCodeLen = 4, clean = TRUE)
```

## Arguments

word	string or vector of strings to encode
maxCodeLen	maximum length of the resulting encodings, in characters
clean	if TRUE, return NA for unknown alphabetical characters

## Details

The variable `word` is the name to be encoded. The variable `maxCodeLen` is the limit on how long the returned name code should be. The default is 4.

The `statcan` algorithm is only defined for inputs over the standard French alphabet. Non-alphabetical characters are removed from the string in a locale-dependent fashion. This strips spaces, hyphens, and numbers. Other letters, such as "Ü," may be permissible in the current locale but are unknown to `statcan`. For inputs outside of its known range, the output is undefined and NA is returned and a warning this thrown. If `clean` is FALSE, `statcan` attempts to process the strings. The default is TRUE.

**Value**

the Statistics Canada encoded character vector

**References**

James P. Howard, II, "Phonetic Spelling Algorithm Implementations for R," *Journal of Statistical Software*, vol. 25, no. 8, (2020), p. 1–21, <10.18637/jss.v095.i08>.

Billy T. Lynch and William L. Arends. "Selection of surname coding procedure for the SRS record linkage system." United States Department of Agriculture, Sample Survey Research Branch, Research Division, Washington, 1977.

**See Also**

Other phonics: [caverphone\(\)](#), [cologne\(\)](#), [lein\(\)](#), [metaphone\(\)](#), [mra\\_encode\(\)](#), [nysiis\(\)](#), [onca\(\)](#), [phonex\(\)](#), [phonics\(\)](#), [rogerroot\(\)](#), [soundex\(\)](#)

**Examples**

```
statcan("William")
statcan(c("Peter", "Peady"))
statcan("Stevenson", maxCodeLen = 8)
```



# Index

## \* **phonics**

- caverphone, [2](#)
- cologne, [3](#)
- lein, [4](#)
- metaphone, [5](#)
- mra\_encode, [6](#)
- nysiis, [8](#)
- onca, [9](#)
- phonex, [10](#)
- phonics, [11](#)
- rogerroot, [12](#)
- soundex, [14](#)
- statcan, [15](#)

caverphone, [2](#), [4-7](#), [9-13](#), [15](#), [16](#)

cologne, [3](#), [3](#), [5-7](#), [9-13](#), [15](#), [16](#)

lein, [3](#), [4](#), [4](#), [6](#), [7](#), [9-13](#), [15](#), [16](#)

metaphone, [3-5](#), [5](#), [7](#), [9-13](#), [15](#), [16](#)

mra\_compare (mra\_encode), [6](#)

mra\_encode, [3-6](#), [6](#), [9-13](#), [15](#), [16](#)

nysiis, [3-7](#), [8](#), [10-13](#), [15](#), [16](#)

onca, [3-7](#), [9](#), [9](#), [11-13](#), [15](#), [16](#)

phonex, [3-7](#), [9](#), [10](#), [10](#), [12](#), [13](#), [15](#), [16](#)

phonics, [3-7](#), [9-11](#), [11](#), [13](#), [15](#), [16](#)

refinedSoundex (soundex), [14](#)

rogerroot, [3-7](#), [9-12](#), [12](#), [15](#), [16](#)

soundex, [3-7](#), [9-13](#), [14](#), [16](#)

statcan, [3-7](#), [9-13](#), [15](#), [15](#)