
EasyConversion Documentation

Release master

May 16, 2021

Contents

1	Setup	3
2	EasyConversion.convert	5
2.1	Decimal To Binary	5
2.2	Binary to Decimal	6
2.3	Decimal to Letter	6
2.4	Letter to Decimal	7
2.5	Letter (string) to Ascii	8
2.6	Ascii binary to Letter (string)	9
2.7	Morse to String	9
2.8	String to Morse	10
2.9	Fahrenheit to celsius	11
2.10	Celsius to fahrenheit	11
3	EasyConversion.convert.detect	13
3.1	String and asciibinary	13
3.2	Decimal and Binary	14
3.3	Morse and String	15
3.4	Celsius and Fahrenheit	15
4	EasyConversion.textformat	17
4.1	.color	17
5	EasyConversion.docs	19
5.1	Documentation fetch format	19
6	EasyConversion.info	21
6.1	.version	21
7	Version history	23
7.1	0.6.3 : 16 May 2021	23
7.2	0.6.1 : 12 July 2020	23
7.3	0.6.0 : 12 July 2020	24
7.4	0.5.5 : 3 July 2020	24
7.5	0.5.4 : 3 July 2020	24
7.6	0.5.2 : 1 July 2020	24
7.7	0.5.1 : 30 June 2020	24

7.8	0.5.0 : 30 June 2020	25
7.9	0.4.1 : 28 June 2020	25
7.10	0.4.0 : 28 June 2020	25
7.11	0.3.1 : 28 June 2020	26
7.12	0.3 : 28 June 2020	26
7.13	0.2 : 27 June 2020	26
7.14	0.1 : 27 June 2020	26

[PyPi page](#) | [GitHub page](#)

EasyConversion is a library for easily converting in python. It is mostly made for testing, but can be used. It is very early so don't expect much from it. For code examples, please see [here](#)

CHAPTER 1

Setup

Installation:

```
pip install EasyConversion
```

Or download it from the [PyPi page](#)

Importing:

Importing main conversion: `from EasyConversion import convert`

Importing Documentation in python: `from EasyConversion import docs`

Importing Info: `from EasyConversion import info`

Importing Print Formatting: `from EasyConversion import textformat`

Importing all: `from EasyConversion import convert, docs, info, textformat`

EasyConversion.convert

Section for converting. There will be input, output, aliases and usage Documented.

2.1 Decimal To Binary

Usage:

```
.decimal.binary(decimal : [str, int, list], return_type=bin)
```

Full example:

```
from EasyConversion import convert
print(convert.decimal.binary("21", return_type=bin))
```

Arguments:

decimal the decimal number to input. Type: str, int, list

Optional: return_type the output type. Options: bin, str, int Defaults to bin

Output:

Output can be in a bin, str, int, or [list](if input type is list)

Output type defaults to bin

Output type can be changed with argument return_type=[str, int, bin]

If input type is list, it returns all sections converted in the same order

List form returns '0' in error.

Aliases:

- dec.Bin
- Dec.Bin
- Decimal.Binary

2.2 Binary to Decimal

Usage:

```
.binary.decimal(binary : [bin, int, str, list], return_type=int)
```

Full example:

```
from EasyConversion import convert
print(convert.binary.decimal("10101", return_type=str))
```

Arguments:

binary the binary number to input. Type: str, int, bin, list
Optional: return_type the output type. Options: str, int Defaults to int

Output:

Output can be in a str, int, or [list](if input type is list)
Output type defaults to int
Output type can be changed with argument return_type=[str, int]
If input type is list, it returns all sections converted in the same order
List form returns '0' in error.

Aliases:

- Bin.Dec
- Bin.dec
- Binary.Decimal

2.3 Decimal to Letter

Usage:

```
.decimal.letter(input_number : [int, str, list], repeat=False)
```

Arguments:

`input_number` the number to input to be converted
`repeat` if it should repeat the alphabet for converting (defaults to False)

Full example:

```
from EasyConversion import convert
print(convert.decimal.letter(["100", "3", "4", "not_number"]))
print(convert.decimal.letter(["100", "3", "4", "not_number"], repeat=True))
```

Output:

The full example would output

```
[None, 'c', 'd', None]
['v', 'c', 'd', None]
```

Output is the input number in letters (based on alphabet)
Output is in `str`

Aliases:

- `Dec.letter`
- `Dec.let`
- `Decimal.Let`
- `Decimal.Letter`
- `decimal.Letter`

2.4 Letter to Decimal

Usage:

```
.letter.decimal(input_letter : [str, list], return_type=int)
```

Arguments:

`input_letter` the letter to input and be converted
`return_type` the type for a return. Defaults to `int`

Full example:

```
from EasyConversion import convert
print(convert.letter.decimal(["a", "b", "g", "100number"]))
print(convert.letter.decimal("abcdefgh", return_type=str))
```

Output:

The full example would output

```
[1, 2, 7, None]
['1', '2', '3', '4', '5', '6', '7', '8']
```

Output is the input letter(s) in numbers (based on alphabet)
Output is in `int` by default, or `return_type=[option]`
Output is a list unless it's a single letter

Aliases:

- Letter.dec
- Let.dec
- Letter.Dec
- Letter.Decimal
- letter.Decimal

2.5 Letter (string) to Ascii

Usage:

```
.string.asciibinary(input_string)
```

Arguments:

`input_string` the string to input and be converted into an asciibinary list

Full example:

```
from EasyConversion import convert
print(convert.string.asciibinary("string"))
```

Output:

The full example would output

```
['011110011', '01110100', '01110010', '01101001', '01101110',
 '01100111']
```

Output is the input letter(s) in ascii binary

Output is in `str-list` by default

Aliases:

- Letter.Ascii
- Let.Asc
- Str.Asc
- Letter.asc
- letter.asc

2.6 Ascii binary to Letter (string)

Usage:

```
.asciibinary.string(input_string)
```

Arguments:

`input_ascii` the ascii to input and be converted to a string

Full example:

```
from EasyConversion import convert
print(convert.asciibinary.string("01110011 01110100 01110010 01101001 01101110_
↪01100111"))
```

Output:

The full example would output

string

Output is the input ascii binary in a string

Output is in `str` by default

Aliases:

- `Ascii.Letter`
- `Asc.Let`
- `Asc.Str`
- `asc.Letter`
- `Asciibinary.String`

2.7 Morse to String

Usage:

```
.morse.string(morse_code)
```

Arguments:

`input` the morse to be converted into a string

Full example:

```
from EasyConversion import convert
print(convert.morse.string("... - -. . - -. -"))
```

Output:

The full example would output:

STRING

Output is the input morse converted into a string.

Output is in `str`

Aliases:

- `Morse.String`
- `Morse.string`
- `morse.String`
- `morse.letter`
- `Morse.Letter`
- `morse.Letter`

2.8 String to Morse

Usage:

```
.string.morse(input_text)
```

Arguments:

`input` the text to be converted into morse

Full example:

```
from EasyConversion import convert
print(convert.string.morse("String"))
```

Output:

The full example would output:

STRING

Output is the input text converted into morse

Output is `... - .- . . - .`

Aliases:

- `String.Morse`
- `string.Morse`
- `String.morse`
- `letter.morse`
- `Letter.Morse`
- `Letter.morse`

2.9 Farenheit to celsius

Usage:

```
.fahrenheit.celsius(fahrenheit)
```

Arguments:

fahrenheit the fahrenheit to be converted into celsius

Full example:

```
from EasyConversion import convert
print(convert.fahrenheit.celsius("50"))
```

Output:

The full example would output:

```
10.0
```

Output is the input fahrenheit converted into celsius

Aliases:

- f.c
- fahrenheit.celsius
- Farenheit.celsius
- fahrenheit.c
- f.celsius
- fahrenheit.Celsius

2.10 Celsius to fahrenheit

Usage:

```
.celsius.fahrenheit(celsius)
```

Arguments:

celsius the celsius to be converted into fahrenheit

Full example:

```
from EasyConversion import convert
print(convert.celsius.fahrenheit("10"))
```

Output:

The full example would output:

50.0

Output is the input celsius converted into fahrenheit

Aliases:

- c.f
- celsius.fahrenheit
- Celsius.fahrenheit
- celcius.f
- c.fahrenheit
- celsius.Fahrenheit

EasyConversion.convert.detect

Detect input type and create output based on that

3.1 String and asciibinary

Usage:

```
.asciistring(input, return_type=list)
```

Arguments:

input the string to be converted
return_type the type to return, list, str. Defaults to list

Full example:

```
from EasyConversion import convert

print(convert.detect.asciistring("a string", return_type=str))

print(convert.detect.asciistring("01100001 00100000 01110011 01110100 01110010_
↪01101001 01101110 01100111"))
```

Output:

The full example would output:

```
01100001 00100000 01110011 01110100 01110010 01101001 01101110
01100111
a string
```

Output is the input converted, after detecting if it a string or ascii

Output is in `str-list` by default

Aliases:

- `Stringascii`
- `stringascii`
- `StringAscii`
- `Asciistring`
- `AsciiString`

3.2 Decimal and Binary

Usage:

```
.binarydecimal(input)
```

Arguments:

`input` the binary or decimal to be converted

Full example:

```
from EasyConversion import convert

print(convert.detect.decimalbinary(21))
print(convert.detect.decimalbinary("10101"))
```

Output:

The full example would output:

```
10101
21
```

Output is the input converted, after detecting if it a binary number or a normal decimal number

Output is in `str`

Aliases:

- `Decimalbinary`
- `DecimalBinary`
- `decimalbinary`
- `Binarydecimal`
- `BinaryDecimal`

3.3 Morse and String

Usage:

```
.morsestring(input)
```

Arguments:

input the morse or string to be converted

Full example:

```
from EasyConversion import convert

print(convert.detect.morsestring("string"))
print(convert.detect.morsestring("... - ... .. - .-"))
```

Output:

The full example would output:

```
... - ... .. - .-
```

```
STRING
```

Output is the input converted, after detecting if it morse code or a string

Output is in str

Aliases:

- MorseString
- Morsestring
- Stringmorse
- stringmorse
- StringMorse

3.4 Celsius and Farenheit

Usage:

```
.celsiusfahrenheit(input)
```

Arguments:

input the celsius or fahrenheit to be converted

Full example:

```
from EasyConversion import convert

print(convert.detect.celsiusfahrenheit("50f"))
print(convert.detect.celsiusfahrenheit(["10c", "50f"]))
```

Output:

The full example would output:

```
10.0
```

```
[50.0, 10.0]
```

Output is the input converted, after detecting if it is celsius or fahrenheit (requires a c or f)

Output is in `float`

Aliases:

- `celsiusfahrenheit`
- `FahrenheitCelsius`
- `CelsiusFahrenheit`
- `Fahrenheitcelsius`
- `Celsiusfahrenheit`

Formatting print text in python

4.1 .color

Main options:

These are the options for using colors, and how to use them

- .color.purple
- .color.cyan
- .color.darkcyan
- .color.blue
- .color.green
- .color.yellow
- .color.red
- .color.bold
- .color.underline
- .color.end

To start a color use .color.[color name from above] To end a color use .color.end

Full example:

```
from EasyConversion import textformat

print(f"""
This text is {textformat.color.green} Green {textformat.color.end}
```

(continues on next page)

(continued from previous page)

```
This text is /textformat.color.underline/ /textformat.color.bold/ Underlined and bold
↪ /textformat.color.end/ /textformat.color.end/
" " ")
```

Full example output

Get the docs for a function, in the python script (less detailed, easier to find)

5.1 Documentation fetch format

Usage:

```
[from].[to]
```

Example:

```
.letter.decimal
```

Full example:

```
from EasyConversion import docs
print(docs.decimal.letter)
```

Output:

Docs for the section in `str`

Aliases:

- See aliases for the section you want to see the documentation for

Aliases for `.docs`

- `.docfetch`

- `.fetch_docs`
- `.documentation`

6.1 .version

.current

Current version of the package with different Options:

- `.name` Current version name/number
- `.release_date` Current version release date

.get_release(version_number : str)

Get a version of the package with different Options:

- `.name` Version name/number
- `.release_date` Version release date

Returns error in invalid version

Full example:

```
from EasyConversion import info

print("We are version " + info.version.current.name)
```

(continues on next page)

(continued from previous page)

```
chosen_version = info.version.get_version("0.2.0")
print("Version " + chosen_version.name + " was released on " + chosen_version.release_
↪date + ".")
```

CHAPTER 7

Version history

7.1 0.6.3 : 16 May 2021

Minor changes to morse
Small bug fixes

7.2 0.6.1 : 12 July 2020

Fixed more issues with documentation
Added GitHub examples for the last 5 updates
Fixed bugs and tweaked small things with conversion
Rewrote inbuilt docs to return links instead of text (it's easier to maintain this way)
Added a few missing versions for getting versions with `EasyConversion.info`
Minor performance improvements

7.3 0.6.0 : 12 July 2020

Fixed bugs with inputting lists on *detect* for binarydecimal
Added support for / on morse
Fixed bugs with morse and .
Some bugs with getting versions and incorrect version names fixed
GitHub updates
Added conversion between celsius and fahrenheit (with detect option)
Fixed a few errors in documentation

7.4 0.5.5 : 3 July 2020

Fixed many bugs

7.5 0.5.4 : 3 July 2020

Fixed some issues with detection
Detection is now out of beta

7.6 0.5.2 : 1 July 2020

- Changed how getting current version works; smaller code
- Fixed a few things in the documentation and examples
- Changed default return type for `detect.asciistring` to `str`
- Added better error messages to morse
- Fixed detection errors

7.7 0.5.1 : 30 June 2020

- Fixed some bugs with release 0.5.0

- Added more examples to the GitHub

7.8 0.5.0 : 30 June 2020

- Added conversions between string and Ascii Binary
- Fixed some output type bugs with other conversions
- File size changes
- Changed the way version info is fetched, allowing for custom version searches
- Added `EasyConversion.convert.detect` for detecting input type (alpha)
- Documented text formatting options (print colors)
- Added morse and text conversions
- Added some better section descriptions

7.9 0.4.1 : 28 June 2020

- Fixed major bug causing letter conversions to freeze
- Added PyPi description
- Updated GitHub page

7.10 0.4.0 : 28 June 2020

- Re-ordered sections to make converting easier to read
- Fixed more aliases
- Improved (this) documentation page
- New convert option: letter (convert between number and letter)
- Fixed bugs with binary with decimal errors
- New file system, seperated sections convert and doc
- New section, info (get version info, release date etc)
- General fixes and improvements all-round

7.11 0.3.1 : 28 June 2020

- Fixed docs function
- Fixed most aliases

7.12 0.3 : 28 June 2020

- Changed the file system so imports are smaller and easier
- Fixed inputting binary in type `bin`

7.13 0.2 : 27 June 2020

- Added in-built docs

7.14 0.1 : 27 June 2020

- Initial release (`.Convert.BinToDec` and `.Convert.DecToBin`) [after **0.3** these do not work.]