Mathmaker Lib Documentation

Release 0.6.4

Nicolas Hainaux <nh.techn@gmail.com>

May 05, 2018

Contents

CHAPTER 1

User's guide

Contents:

1.1 Overview

Mathmaker Lib offers python objects to create mathematical expressions or geometric figures to print as LaTeX. These objects can also be used to create detailed calculations, like equations resolutions or expressions expansions.

License

1.2 Quickstart

1.2.1 Install

OS requirement: Linux, FreeBSD or Windows.

The required python version to use Mathmaker Lib is python>=3.6. You'll need to install it if it's not already on your system.

```
$ pip3 install mathmakerlib
```

1.2.2 Basic use

```
>>> from mathmakerlib.calculus import Number
>>> Number('5.807')
Number('5.807')
>>> Number('5.807').atomized()
[Number('5'), Number('0.8'), Number('0.007')]
>>> Number('150').is_power_of_10()
```

```
False
>>> Number('0.001').is_power_of_10()
True
>>>
```

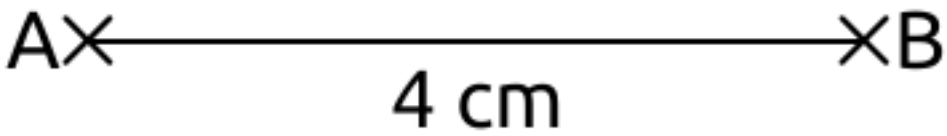
You can also play with units. Basic operations are available:

```
>>> n = Number(4, unit='cm')
>>> n
Number('4 cm')
>>> str(n)
'4 cm'
>>> n.unit
Unit('cm')
>>> n.printed
'\\SI{4}{cm}'
>>> n + Number(6, unit='kg')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/home/nico/dev/mathmaker/mathmakerlib/mathmakerlib/calculus/number.py", line␣
↪177, in __add__
    other_unit))
ValueError: Cannot add two Numbers having different Units (cm and kg).
>>> n * Number(6, unit='cm')
Number('24 cm^2')
>>>
```

Basic geometric shapes can also be created:

```
>>> from mathmakerlib.geometry import Point, LineSegment
>>> ls = LineSegment(Point(0, 0, 'A'), Point(4, 0, 'B'), label='4 cm')
>>> ls.drawn
\begin{tikzpicture}
% Declare Points
\coordinate (A) at (0,0);
\coordinate (B) at (4,0);

% Draw Points
\draw (A) node {$\times$};
\draw (B) node {$\times$};

% Draw Line Segment
\draw[thick] (A) -- (B) node[midway, below, sloped] {4 cm};

% Label Points
\draw (A) node[left] {A};
\draw (B) node[right] {B};
\end{tikzpicture}
>>>
```

Once compiled (here using Ubuntu font):



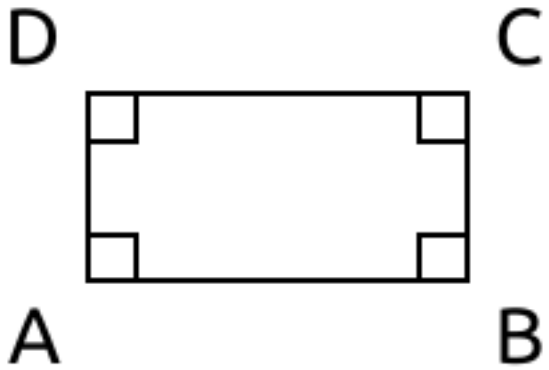
```
>>> from mathmakerlib.geometry import Point, DividedLineSegment
>>> A = Point(0, 0, 'A')
>>> B = Point(10, 0, 'B')
>>> ls = DividedLineSegment(A, B, n=5, fill=3, fillcolor='Cerulean')
>>> ls.drawn
\begin{tikzpicture}
% Declare Points
\coordinate (A) at (0,0);
\coordinate (B) at (10,0);
\coordinate (a3) at (6,0);

% Draw Divided Line Segment
\draw[ultra thick] (A) -- (B);
\draw[ultra thick, Cerulean] (A) -- (a3);
\draw[ultra thick, opacity=0] (A) -- (B) node[opacity=1, pos=0, sloped] {|}␣
↪node[opacity=1, pos=0.2, sloped] {|} node[opacity=1, pos=0.4, sloped] {|}␣
↪node[opacity=1, pos=0.6, sloped] {|} node[opacity=1, pos=0.8, sloped] {|}␣
↪node[opacity=1, pos=1, sloped] {|};

% Label Points

\end{tikzpicture}
```

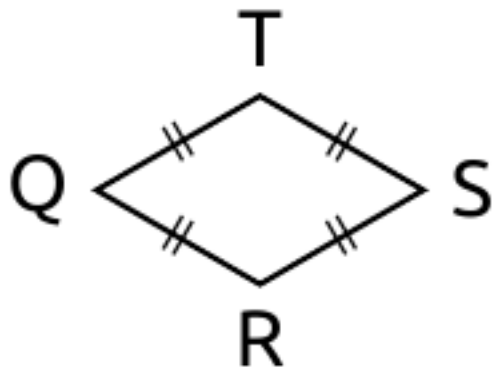
Once compiled (here using Ubuntu font):

```
>>> from mathmakerlib.geometry import Rectangle, Rhombus
>>> import mathmakerlib.required
>>> mathmakerlib.required.init()
>>> print(Rectangle().drawn)

\begin{tikzpicture}
% Declare Points
\coordinate (A) at (0,0);
\coordinate (B) at (2,0);
\coordinate (C) at (2,1);
\coordinate (D) at (0,1);

% Draw Rectangle
\draw[thick] (A)
-- (B)
-- (C)
-- (D)
-- cycle;

% Mark right angles
\draw[thick, cm={cos(0), sin(0), -sin(0), cos(0), (A)}] (0.25 cm, 0) -- (0.25 cm, 0.
↪25 cm) -- (0, 0.25 cm);
\draw[thick, cm={cos(90), sin(90), -sin(90), cos(90), (B)}] (0.25 cm, 0) -- (0.25 cm,␣
↪0.25 cm) -- (0, 0.25 cm);
\draw[thick, cm={cos(180), sin(180), -sin(180), cos(180), (C)}] (0.25 cm, 0) -- (0.25␣
↪cm, 0.25 cm) -- (0, 0.25 cm);
\draw[thick, cm={cos(-90), sin(-90), -sin(-90), cos(-90), (D)}] (0.25 cm, 0) -- (0.25␣
↪cm, 0.25 cm) -- (0, 0.25 cm);
```

```
% Label Points
\draw (A) node[below left] {A};
\draw (B) node[below right] {B};
\draw (C) node[above right] {C};
\draw (D) node[above left] {D};
\end{tikzpicture}

>>> print(Rhombus().drawn)

\begin{tikzpicture}
% Declare Points
\coordinate (Q) at (0,0);
\coordinate (R) at (0.866,-0.5);
\coordinate (S) at (1.732,0);
\coordinate (T) at (0.866,0.5);

% Draw Rhombus
\draw[thick] (Q)
-- (R) node[midway, sloped, scale=0.5] {||}
-- (S) node[midway, sloped, scale=0.5] {||}
-- (T) node[midway, sloped, scale=0.5] {||}
-- cycle node[midway, sloped, scale=0.5] {||};

% Label Points
\draw (Q) node[left] {Q};
\draw (R) node[below] {R};
\draw (S) node[right] {S};
\draw (T) node[above] {T};
\end{tikzpicture}
```

Once compiled (still using Ubuntu font):

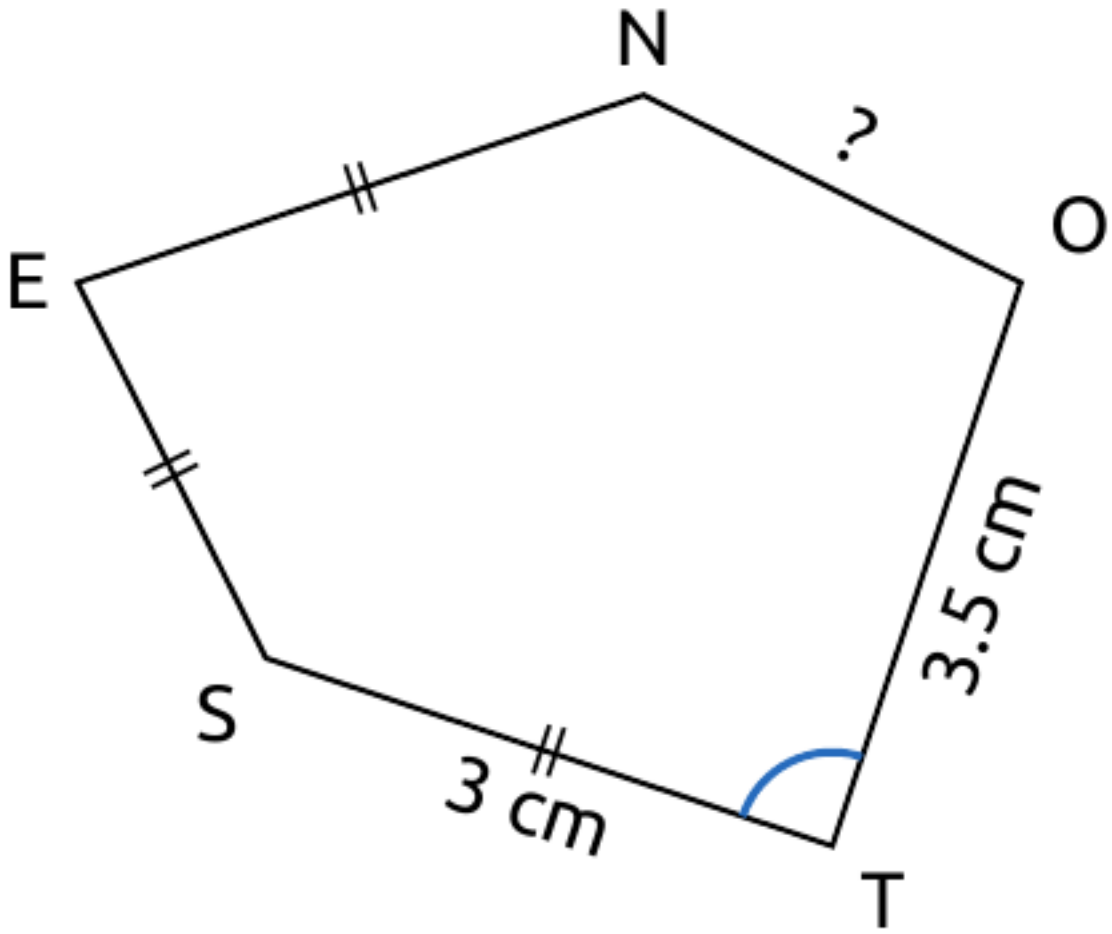





```
>>> import mathmakerlib.required
>>> from mathmakerlib.calculus import Number
>>> from mathmakerlib.geometry import Polygon, Point, AngleMark
>>> mathmakerlib.required.init()
>>> p = Polygon(Point(0, 0), Point(3, -1), Point(4, 2), Point(2, 3), Point(-1, 2),␣
↪name='STONE')
>>> p.setup_labels([Number(3, unit='cm'), Number('3.5', unit='cm'), Number('2.5',␣
↪unit='cm'), Number(3, unit='cm'), Number(3, unit='cm')], masks=[None, None, '?', '
↪', ' '])
>>> p.sides[0].mark = p.sides[3].mark = p.sides[4].mark = '||'
>>> p.sides[0].mark_scale = p.sides[3].mark_scale = p.sides[4].mark_scale = Number('0.
↪67')
>>> p.angles[1].mark = AngleMark(color='NavyBlue', thickness='very thick',␣
↪radius=Number('0.5', unit='cm'))
>>> print(p.drawn)

\begin{tikzpicture}
% Declare Points
\coordinate (S) at (0,0);
\coordinate (T) at (3,-1);
\coordinate (O) at (4,2);
\coordinate (N) at (2,3);
\coordinate (E) at (-1,2);

% Draw Pentagon
\draw[thick] (S)
-- (T) node[midway, below, sloped] {3 cm} node[midway, sloped, scale=0.67] {||}
-- (O) node[midway, below, sloped] {3.5 cm}
-- (N) node[midway, above, sloped] {?}
-- (E) node[midway, sloped, scale=0.67] {||}
-- cycle node[midway, sloped, scale=0.67] {||}
pic [draw, NavyBlue, very thick, angle radius = 0.5 cm] {angle = O--T--S};

% Label Points
\draw (S) node[below left] {S};
\draw (T) node[below right] {T};
\draw (O) node[above right] {O};
\draw (N) node[above] {N};
\draw (E) node[left] {E};
\end{tikzpicture}

>>> p.lbl_perimeter
Number('15.0 cm')
>>> p.type
```

```
'Pentagon'
>>> print(mathmakerlib.required.tikz_library)
{'angles': True}
>>>
```

Once compiled (still using Ubuntu font, take care to include angles tikz library):



1.3 Contribute

Before submitting a PR, please ensure you've had a look at the mathmaker's writing rules.

So far, more details can be found in the documentation for developers of mathmaker.

Any question can be sent to nh dot techn (hosted at gmail dot com).

1.4 Contributors

1.4.1 Development

- Lead developer: Nicolas Hainaux

- Clever advices: Olivier Cecillon

1.4.2 Patience and chocolate cakes

Sophie Reboud

1.5 Changelog

1.5.1 Version 0.6 (2018-04-12)

- A standalone Angle or AnglesSet can be drawn. Enrich Angles' decorations (hatch marks, labeling, second decoration etc.).
- An integer Number can be split as a sum of integers $\pm$ 0.5 (or $\pm$ 0.25)
- Add Number.lowest_nonzero_digit_index()
- Patch Number.split() to get a consistent behaviour for integers too (default split will be done at lowest non zero digit place: 500 will be split as 100 + 400, or 200 + 300 etc. and with dig=1, it will be split as 10 + 490, or 20 + 480 etc.).
- Fix: Numbers with an angle's unit should be displayed as \ang{...} rather than \SI{...}{\textdegree}.
- Add basic classes to handle LaTeX commands and options' lists.

Patch 0.6.1 (2018-04-13)

- Add a constant in LaTeX module

Patch 0.6.2 (2018-04-30)

- Add Number.digits_sum()

Patch 0.6.3 (2018-05-02)

- Add Number.digits and Number.digit()

Patch 0.6.4 (2018-05-05)

- Add some amsmath symbols.

1.5.2 Version 0.5 (2018-01-10)

- Add Number.quantize().
- A Number can be converted into a another unit of the same physical quantity.
- Accept int as exponent (or even content) of an Exponented.
- Fractions can be created from a decimal Number.
- Fractions become Evaluable and can be compared to other numbers.

- Standalone Units will be printed using siunitx (e.g. as '\si{cm}').
- Fix bug: current locale is ignored when printing a number having a unit.
- Do not automatically remove possible trailing zeros when printing a Number.
- Add the tonne (t) as mass unit.
- An optional patch allow Polygons to be drawn to the first vertex again instead of only cycling (default behaviour).

1.5.3 Version 0.4 (2017-12-19)

- Add more complex geometric objects: Polygon, Triangle, RightTriangle, EquilateralTriangle, IsoscelesTriangle, Quadrilateral, Rhombus, Rectangle, Square.
- Numbers can be "copied" using copy.copy() or copy.deepcopy().
- Add Point.rotate().
- Add the ability to change the size of Point's drawn shape (using Point.shape_scale).
- Add LineSegment.mark and the ability to change its size (using LineSegment.mark_scale).
- mathmakerlib.requires_pkg becomes mathmakerlib.required and will also handle required options and hacks.
- Add module mathmakerlib.mmlib_setup to configure the behaviour (default values etc.).

Patch 0.4.1 (2018-01-01)

- Fix the locale monkey patch.

1.5.4 Version 0.3 (2017-11-17)

- Add basic geometric objects: Point, LineSegment and DividedLineSegment.
- Add Fraction.
- Add module mathmakerlib.requires_pkg that tells which LaTeX packages will be required to compile the document (like tikz, xcolor, siunitx...).

1.5.5 Version 0.2 (2017-11-01)

- Add Sign, Exponented and Unit classes.
- Numbers are now Signed objects and may be assigned a Unit.

Patches 0.2.1 and 0.2.2 (2017-11-02)

- Bring back Numbers' hashability.
- Add physical_quantity() in unit module.

1.5.6 Initial version 0.1 (2017-10-24)

- Number class and decimal numbers' related functions.

CHAPTER 2

Developer's documentation

2.1 Start Working

Note: `python 3.6` as least, and `git` are required.

Install mathmakerlib in dev mode in a venv:

- Linux:

```
$ cd to/your/dev/directory/
$ python3 -m venv dev0
$ source dev0/bin/activate
(dev0) $ pip3 install pytest tox flake8 pydocstyle sphinx sphinx-autodoc-
↪annotation sphinx-rtd-theme
(dev0) $ mkdir mathmakerlib
(dev0) $ cd mathmakerlib/
(dev0) $ git clone https://github.com/nicolashainaux/mathmakerlib.git
(dev0) $ python3 setup.py develop
```

- FreeBSD:

```
$ cd to/your/dev/directory/
$ python3 -m venv dev0
$ source dev0/bin/activate.csh
[dev0] $ sudo pip3 install pytest tox flake8 pydocstyle sphinx sphinx-autodoc-
↪annotation sphinx-rtd-theme
[dev0] $ mkdir mathmakerlib
[dev0] $ cd mathmakerlib/
[dev0] $ git clone https://github.com/nicolashainaux/mathmakerlib.git
[dev0] $ python3 setup.py develop
```

- Windows (PowerShell):

```
It is strongly advised to develop on a Linux or FreeBSD box.
Nevertheless, for testing purposes at least, this is useful to know how to start␣
↪a virtual environment on Windows (PowerShell), so here it is:

First open a PowerShell:
Windows 7: start menu > search for "PowerShell" > right-click > run as␣
↪administrator.
Windows 10: hit the Windows key + X to open a PowerShell with admin rights.


    PS C:\Windows\system32> cd ../..
    PS C:\> cd .\Users\username\
    PS C:\Users\username> Set-ExecutionPolicy -ExecutionPolicy RemoteSigned
    ...
    PS C:\Users\username\venv> & 'C:\Program Files\Python36\python.exe' -m venv␣
↪dev0
    PS C:\Users\username\venv> .\dev0\Scripts\Activate.ps1
    (dev0) PS C:\Users\username\venv>

then follow the same steps as under Linux or FreeBSD: pip install the␣
↪dependencies, clone the git repo and run ``setup.py`` with develop option.

Or, if this is just for testing: ``pip install mathmakerlib``
```

The tests are stored under `tests/`.

Run the tests:

```
(dev0) $ py.test
```

So far, more details can be found in the documentation for developers of mathmaker.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search