Latexify.jl

and how Julia's metaprogramming makes it useful

Niklas Korsbo NKorsbo Korsbo

```
Julia> using Latexify
Julia> ex = :(x^n/(k^n + x^n))
Julia> latexify(ex)
```

$$\frac{x^{n}}{k^{n}+x^{n}}$$

UNIVERSITY OF CAMBRIDGE
Department of Applied Mathematics and Theoretical Physics

THE SAINSBURY LABORATORY
UNIVERSITY OF CAMBRIDGE

Some supported inputs

Floats

```
latexify( 1.2 )
```

1.2

Rationals

```
latexify( 3//4 )
```

$\frac{3}{4}$

Symbols

```
latexify( :X_active )
```

X_{active}

...

Some supported inputs

Expressions (key functionality)

```
latexify( :(a/(b+c)) )
```

$\frac{a}{b+c}$

Some supported inputs

Container types (Arrays, Dicts, etc.)

```
M = [:(c_1*e^(-c_2*t)) 3//4
     "x/(k+x)"          :X_inactive]
latexify(M)
```

$$\begin{bmatrix} c_1 \cdot e^{-c_2 \cdot t} & \frac{3}{4} \\ \frac{x}{k+x} & X_{inactive} \end{bmatrix}$$

```
D = Dict(:a=>1.0, :b=>"a^c", :c=>3//4)
latexify( D )
```

$$\begin{bmatrix} a & 1.0 \\ b & a^c \\ c & \frac{3}{4} \end{bmatrix}$$

Some supported outputs

```julia
D = Dict(:a=>1.0, :b=>"a^c", :c=>3//4)
latexify( D )
```

$$\begin{bmatrix} a & 1.0 \\ b & a^{c} \\ c & \frac{3}{4} \end{bmatrix}$$

```julia
latexify( D; env=:align )
```

$$\begin{aligned} a =& 1.0 \\ b =& a^{c} \\ c =& \frac{3}{4} \end{aligned}$$

```julia
latexify( D; env=:tabular )
```

a	1.0
b	a^{c}
c	$\frac{3}{4}$

```julia
latexify( D; env=:mdtable )
```

```
| $a$ |          $1.0$ |
| ---:| --------------:|
| $b$ |        $a^{c}$ |
| $c$ | $\frac{3}{4}$ |
```

Print vs. Display (default)

- Outputs a LaTeXString.

```
typeof( latexify("x/y") )

 LaTeXStrings.LaTeXString
```

- Print:

```
Julia> print( latexify("x/y") )
$\frac{x}{y}$
```

- Display (when supported):

```
Julia> display( latexify("x/y") )
```

$$\frac{x}{y}$$

Cedits: LaTeXStrings.jl

Julia makes this useful

- Homoiconicity (Expressions)
    - Convertible to $\LaTeX$
    - Information availability:

```
Julia> f(x) = x/2
Julia> @code_lowered f(1.)
CodeInfo(:(begin
    nothing
    return x / 2
  end))
```

- Expressions → Macros → Domain-specific languages (DSLs)

DiffEqBiological.jl

```julia
reactions = @reaction_network ReactionDemo begin
    r_1, A + B --> C
    r_2, C --> A
end r_1 r_2
```

DiffEqBiological.jl

```
reactions = @reaction_network ReactionDemo begin
    r_1, A + B --> C
    r_2, C --> A
end r_1 r_2
```

- Information is available:

```
Julia> reactions.f_func
3-element Array{Expr,1}:
   :(-1 * r_1 * A * B + r_2 * C)
   :(+(-1 * r_1 * A * B))
   :(r_1 * A * B + -1 * r_2 * C)
```

DiffEqBiological.jl

```julia
reactions = @reaction_network ReactionDemo begin
    r_1, A + B --> C
    r_2, C --> A
end r_1 r_2
```

```julia
latexify(reactions)
```

```julia
latexify(reactions; env=:arrow)
```

$$
\begin{aligned}
\frac{dA}{dt} &= - r_{1} \cdot A \cdot B + r_{2} \cdot C \\
\frac{dB}{dt} &= - r_{1} \cdot A \cdot B \\
\frac{dC}{dt} &= r_{1} \cdot A \cdot B - r_{2} \cdot C
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{A} + \mathrm{B} &\xrightarrow{r_{1}} \mathrm{C} \\
\mathrm{C} &\xrightarrow{r_{2}} \mathrm{A}
\end{aligned}
$$

ParameterizedFunctions.jl

```julia
ode = @ode_def PositiveFeedback begin
  dx = v_x * y^n_x/(k_x^n_x + y^n_x) - d_x*x
  dy = v_y * x^n_y/(k_y^n_y + x^n_y) - d_y*y
end v_x k_x n_x d_x v_y k_y n_y d_y
```

```julia
latexify(ode)
```

$$
\begin{aligned}
\frac{dx}{dt} &= \frac{v_x \cdot y^{n_x}}{k_x^{n_x} + y^{n_x}} - d_x \cdot x \\
\frac{dy}{dt} &= \frac{v_y \cdot x^{n_y}}{k_y^{n_y} + x^{n_y}} - d_y \cdot y
\end{aligned}
$$

ParameterizedFunctions.jl

```
ode = @ode_def PositiveFeedback begin
  dx = v_x * y^n_x/(k_x^n_x + y^n_x) - d_x*x
  dy = v_y * x^n_y/(k_y^n_y + x^n_y) - d_y*y
end v_x k_x n_x d_x v_y k_y n_y d_y
```

```
latexify(ode.symjac)
```

$$\begin{bmatrix} -d_{x} & \frac{y^{-1 + n_{x}} \cdot v_{x} \cdot n_{x}}{k_{x}^{n_{x}} + y^{n_{x}}} - \frac{y^{-1 + 2 \cdot n_{x}} \cdot v_{x} \cdot n_{x}}{\left(k_{x}^{n_{x}} + y^{n_{x}} \right)^{2}} \\ \frac{x^{-1 + n_{y}} \cdot v_{y} \cdot n_{y}}{k_{y}^{n_{y}} + x^{n_{y}}} - \frac{x^{-1 + 2 \cdot n_{y}} \cdot v_{y} \cdot n_{y}}{\left(k_{y}^{n_{y}} + x^{n_{y}} \right)^{2}} & -d_{y} \end{bmatrix}$$

Final words

- Information access in Julia
    - Look for it,
    - Use it (and do cool stuff),
    - Pass it forward in your own packages.

Acknowledgements

PhD supervisor:

Prof. Henrik Jönsson

Fellow Julia enthusiast:

Torkel Loman

Reasons why I started with Julia:

Chris Rackauckas

David P. Sanders

Thank you!