Stencila

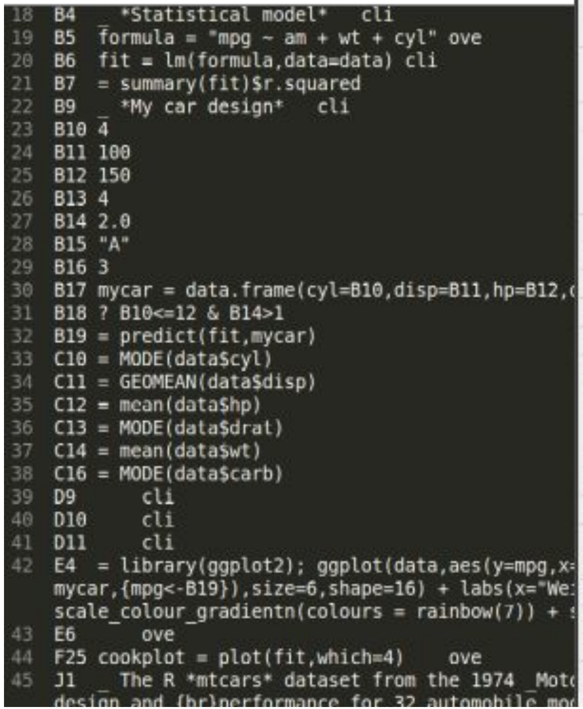
at Library Carpentry Community Call

29 April 2019 @ 20:00 UTC

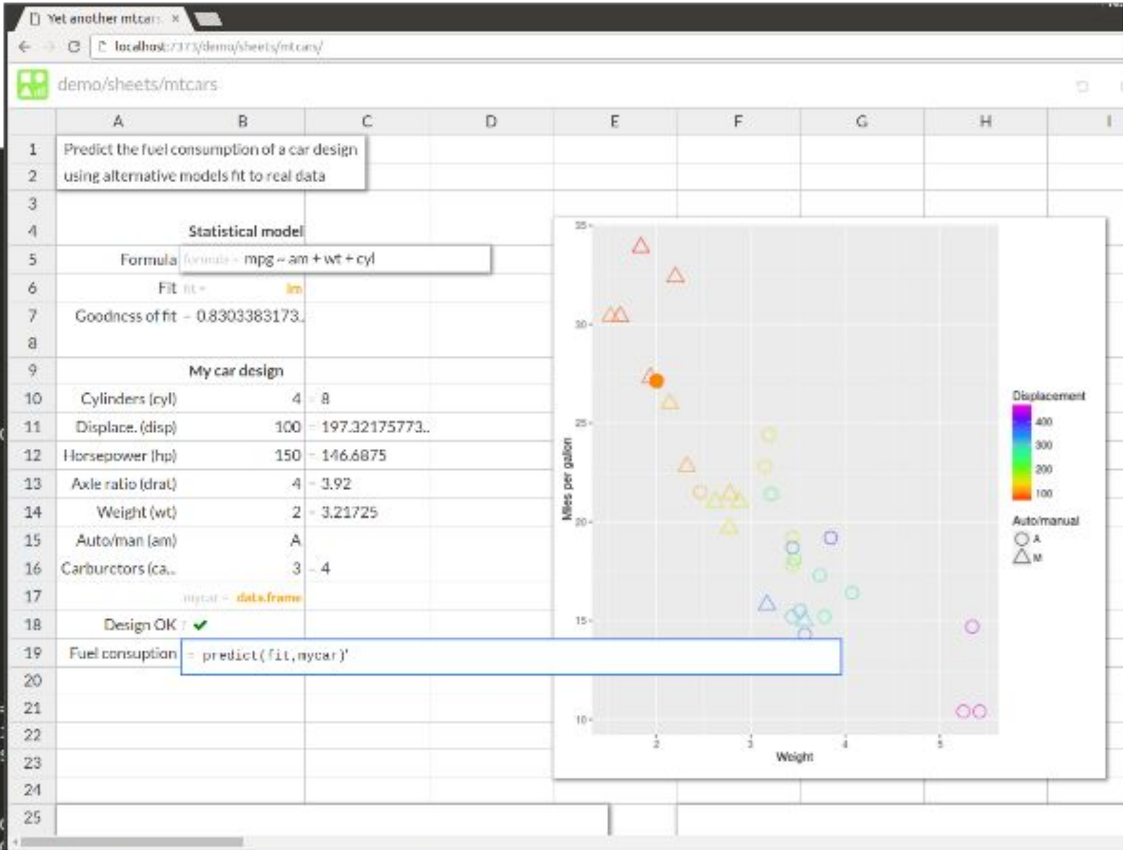
Living documents

- Interactive code with prose
- Fully reproducible and transparent
- Interoperable with existing research tools
- Easy to use (WSIWYG) yet powerful

Stencila



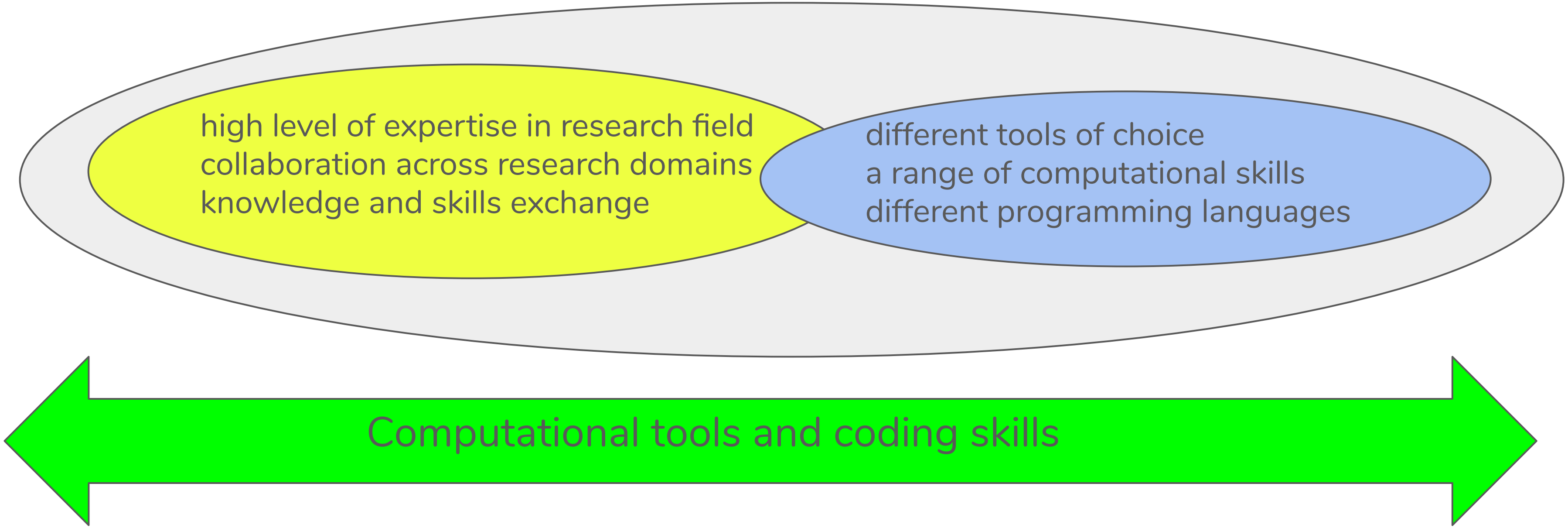
```
B4   _ *Statistical model*   cli
B5  formula = "mpg ~ am + wt + cyl" ove
B6  fit = lm(formula,data=data) cli
B7  = summary(fit)$r.squared
B9   _ *My car design*   cli
B10 4
B11 100
B12 150
B13 4
B14 2.0
B15 "A"
B16 3
B17 mycar = data.frame(cyl=B10,disp=B11,hp=B12,c
B18 ? B10<=12 & B14>1
B19 = predict(fit,mycar)
C10 = MODE(data$cyl)
C11 = GEOMEAN(data$disp)
C12 = mean(data$hp)
C13 = MODE(data$drat)
C14 = mean(data$wt)
C16 = MODE(data$carb)
D9       cli
D10      cli
D11      cli
E4  = library(ggplot2); ggplot(data,aes(y=mpg,x=
mycar,{mpg<-B19}),size=6,shape=16) + labs(x="We:
scale_colour_gradientn(colours = rainbow(7)) + s
E6       ove
F25 cookplot = plot(fit,which=4)    ove
J1   _ The R *mtcars* dataset from the 1974 _Motc
design and {br}performance for 32 automobile moc
```



	A	B	C
1	Predict the fuel consumption of a car design		
2	using alternative models fit to real data		
3			
4		Statistical model	
5	Formula	formula = mpg ~ am + wt + cyl	
6	Fit	fit = lm	
7	Goodness of fit	= 0.8303383173..	
8			
9		My car design	
10	Cylinders (cyl)	4	= 8
11	Displace. (disp)	100	= 197.32175773..
12	Horsepower (hp)	150	= 146.6875
13	Axle ratio (drat)	4	= 3.92
14	Weight (wt)	2	= 3.21725
15	Auto/man (am)	A	
16	Carburetors (ca..	3	= 4
17		mycar = data.frame	
18	Design OK	? ✓	
19	Fuel consuption	= predict(fit,mycar)'	
20			
21			
22			
23			
24			
25			

Stencila

Enabling collaboration



high level of expertise in research field
collaboration across research domains
knowledge and skills exchange

different tools of choice
a range of computational skills
different programming languages

Computational tools and coding skills

Spreadsheets, Excel, Word, Google Drive, Dropbox

Scripts, “hacking” existing code, R code, Word

Advanced code, R, Python, LaTex, version control

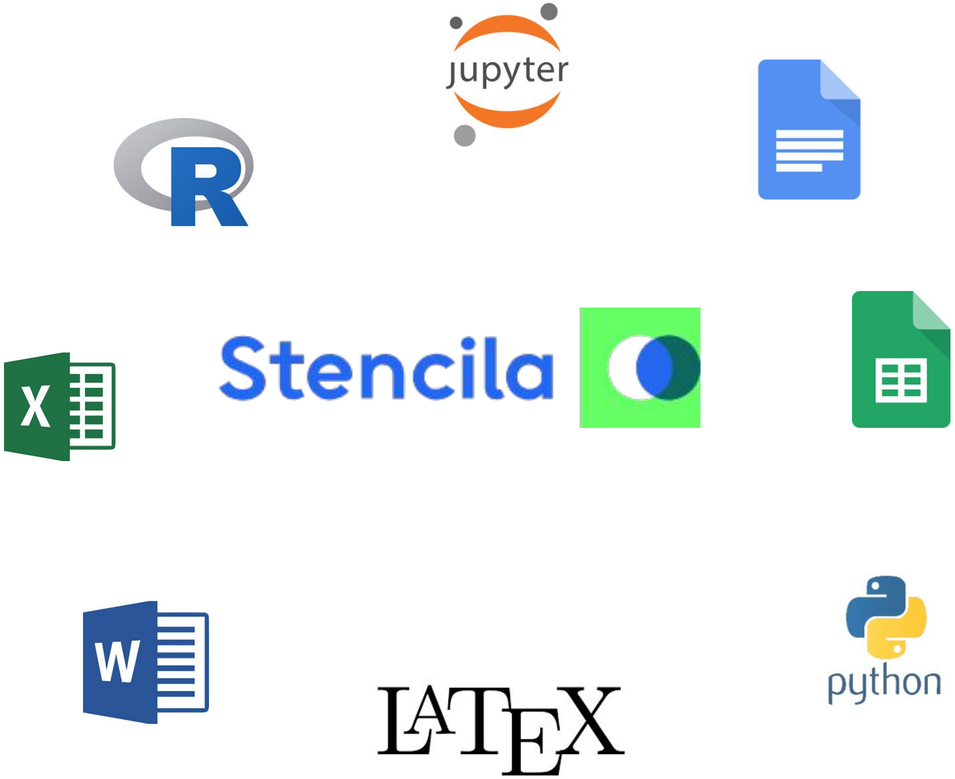
Reproducible journal articles

- smart and computationally reproducible articles
- continuous reproducibility
- support for JATS - publishers’ format
- authoring and reviewing via Stencila Hub
- interactive code within the prose

eLIFE

Fitting into ecosystem

- Complementing not competing
- Interoperable
- Format agnostic (WiP)
- Seamlessly moving between tools
- Retaining interactive and reproducible elements
- Reducing friction

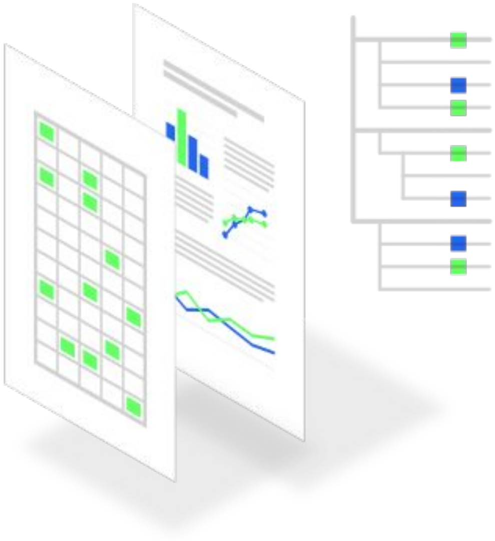


Stencila

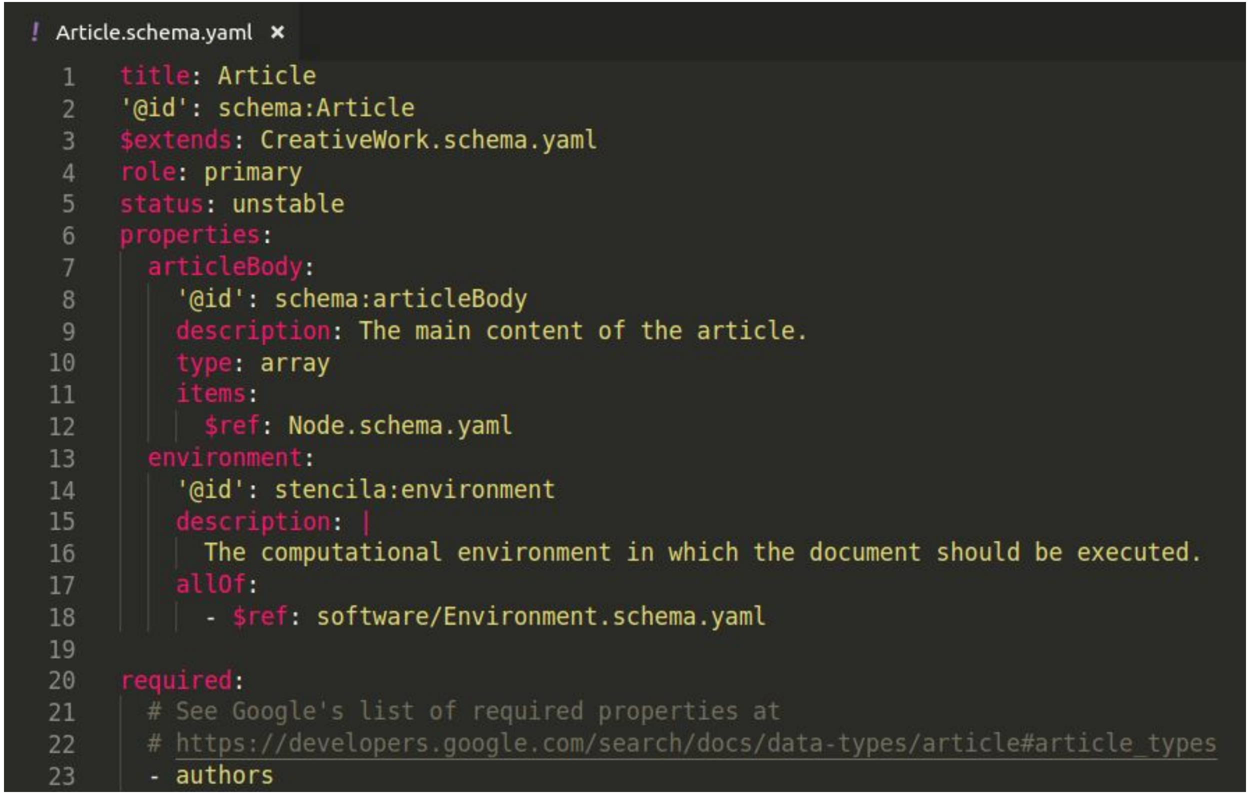
Stencila Schema

- Documents are represented as a **tree of JSON** nodes.

- Each node in the document conforms to a **JSON Schema**.
  - Text content nodes: e.g. Paragraph, Figure, Table
  - Executable nodes: e.g. Expressions, CodeChunks
  - Data nodes e.g. Numbers, Arrays, Datatables



Reusing existing **schemas and vocabularies** e.g. schema.org, codemeta.github.io

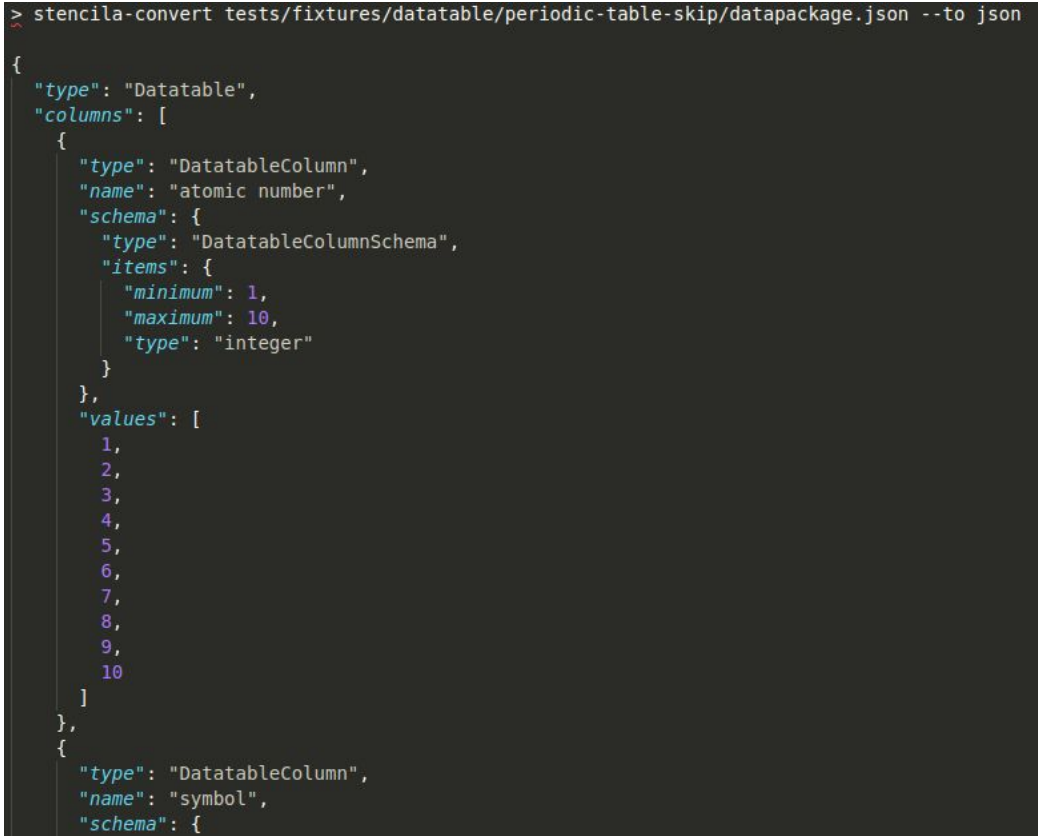


```yaml
title: Article
'@id': schema:Article
$extends: CreativeWork.schema.yaml
role: primary
status: unstable
properties:
  articleBody:
    '@id': schema:articleBody
    description: The main content of the article.
    type: array
    items:
      $ref: Node.schema.yaml
  environment:
    '@id': stencila:environment
    description: |
      The computational environment in which the document should be executed.
    allOf:
      - $ref: software/Environment.schema.yaml

required:
  # See Google's list of required properties at
  # https://developers.google.com/search/docs/data-types/article#article_types
  - authors
```

e.g. **JSON Schema** for an executable **Article**

Integrating with existing **formats** via converters to/from JSON e.g RMarkdown, Excel, JATS



```
> stencila-convert tests/fixtures/datatable/periodic-table-skip/datapackage.json --to json

{
  "type": "Datatable",
  "columns": [
    {
      "type": "DatatableColumn",
      "name": "atomic number",
      "schema": {
        "type": "DatatableColumnSchema",
        "items": {
          "minimum": 1,
          "maximum": 10,
          "type": "integer"
        }
      },
      "values": [
        1,
        2,
        3,
        4,
        5,
        6,
        7,
        8,
        9,
        10
      ]
    },
    {
      "type": "DatatableColumn",
      "name": "symbol",
      "schema": {
```

e.g. converting a FrictionlessData **datapackage.json** to a **Datatable**

Work in progress

- Development of JSON schemas to v1.0
    - https://github.com/stencila/schema
- Converters for other formats (e.g RMarkdown, Excel, JATS) to/from JSON
    - https://github.com/stencila/convert
- Plugins for Google Docs and Sheets and integration with Stencila Hub
    - https://github.com/stencila/hub
- User testing!

Thank you for your attention!

https://stenci.la/

https://gitter.im/stencila/stencila

@stencila

✉ hello@stenci.la