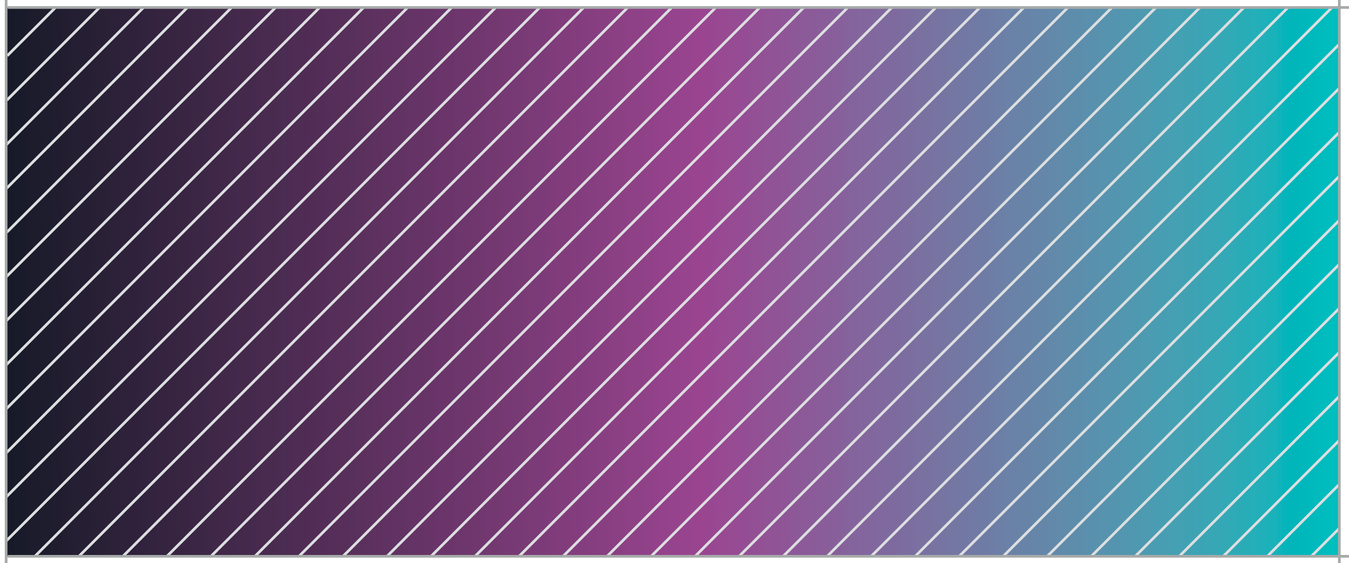
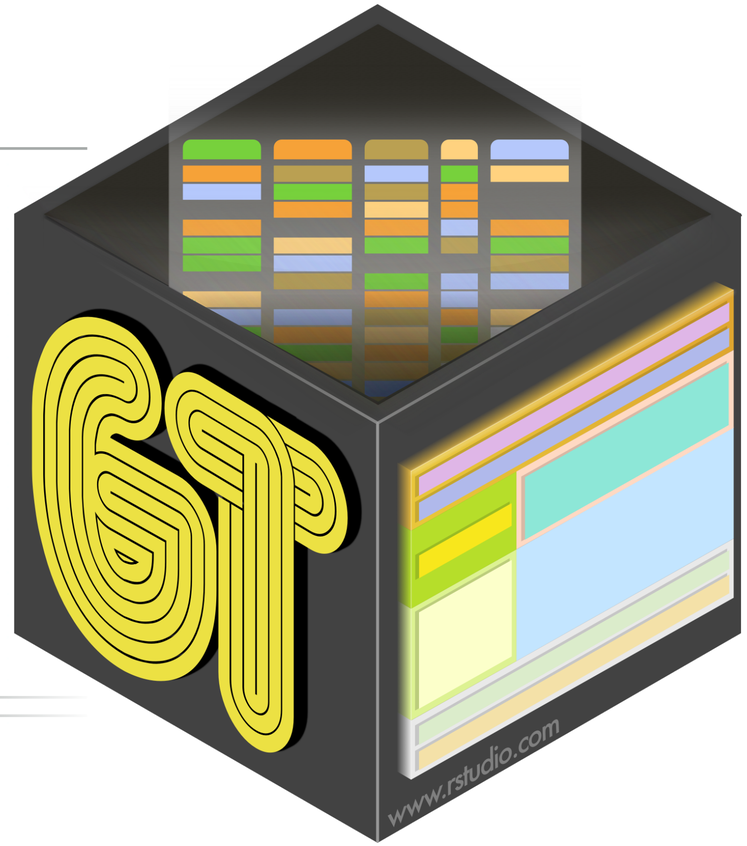
The **gt** Package:
Past, Present, and Future



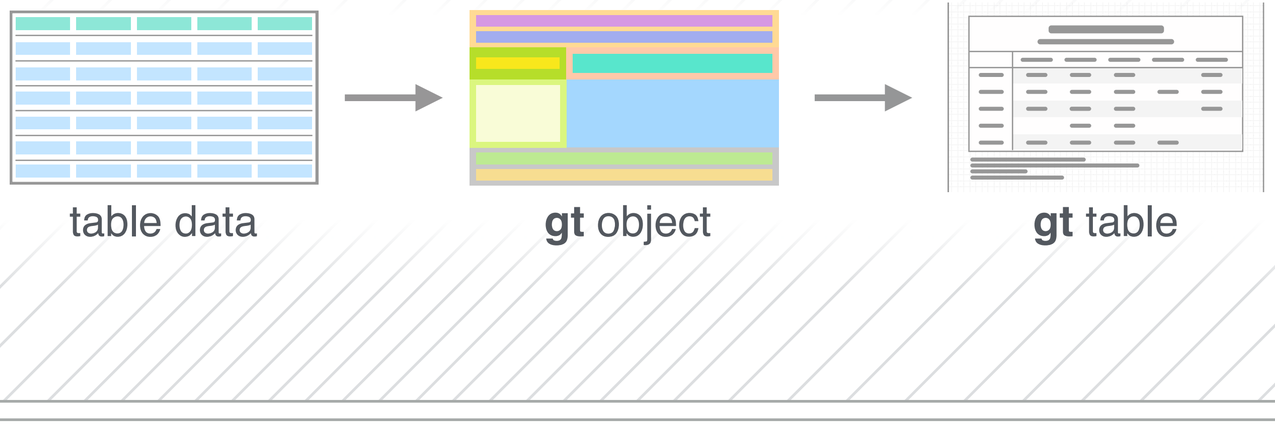


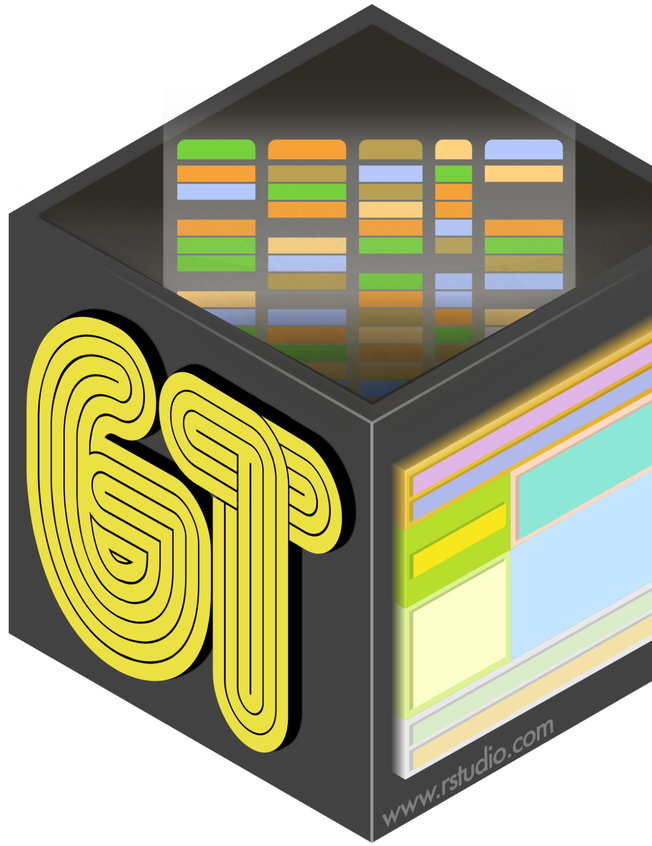
rich-iannone

@riannone

rich@rstudio.com

The **gt** Package:
Past, Present, and Future





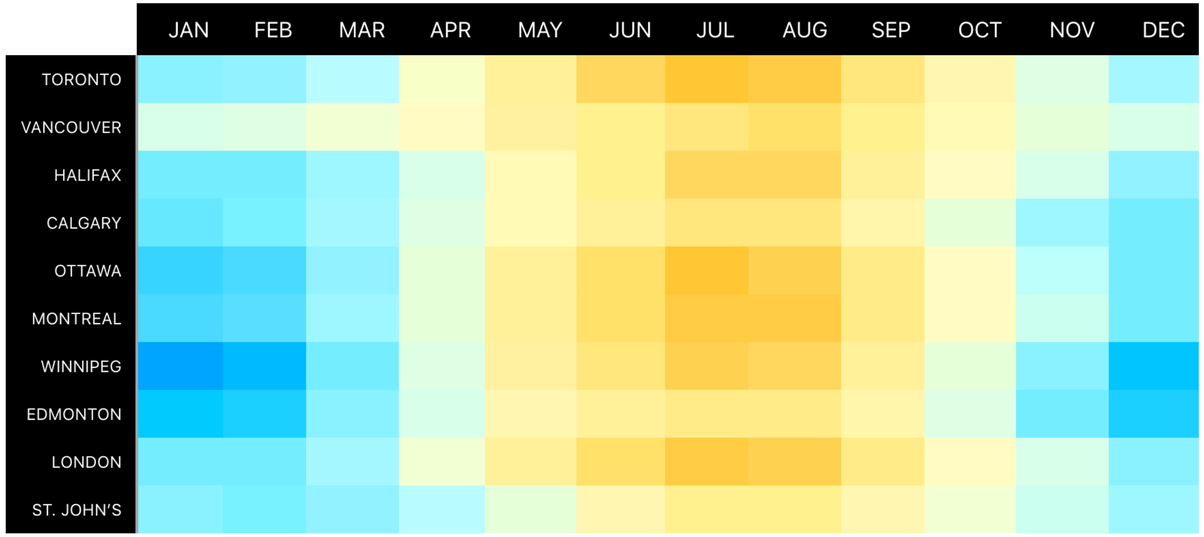
The **gt** package lets us create *display tables* with a declarative interface, allowing us to fine-tune the final appearance.

We can integrate the tables in **R Markdown** docs and **Shiny** applications.

Solar Zenith Angles

A.M. Hours

20°N	True Solar Time																
	0400	0430	0500	0530	0600	0630	0700	0730	0800	0830	0900	0930	1000	1030	1100	1130	1200
jan							84.9	78.7	72.7	66.1	61.5	56.5	52.1	48.3	45.5	43.6	43.0
feb						88.9	82.5	75.8	69.6	63.3	57.7	52.2	47.4	43.1	40.0	37.8	37.2
mar						85.7	78.8	72.0	65.2	58.6	52.3	46.2	40.5	35.5	31.4	28.6	27.7
apr					88.5	81.5	74.4	67.4	60.3	53.4	46.5	39.7	33.2	26.9	21.3	17.2	15.5
may					85.0	78.2	71.2	64.3	57.2	50.2	43.2	36.1	29.1	26.1	15.2	8.8	5.0
jun				89.2	82.7	76.0	69.3	62.5	55.7	48.8	41.9	35.0	28.1	21.1	14.2	7.3	2.0
jul				88.8	82.3	75.7	69.1	62.3	55.5	48.7	41.8	35.0	28.1	21.2	14.3	7.7	3.1
aug					83.8	77.1	70.2	63.3	56.4	49.4	42.4	35.4	28.3	21.3	14.3	7.3	1.9
sep					87.2	80.2	73.2	66.1	59.1	52.1	45.1	38.1	31.3	24.7	18.6	13.7	11.6
oct						84.1	77.1	70.2	63.3	56.5	49.9	43.5	37.5	32.0	27.4	24.3	23.1
nov						87.8	81.3	74.5	68.3	61.8	56.0	50.2	45.3	40.7	37.4	35.1	34.4
dec							84.3	78.0	71.8	66.1	60.5	55.6	50.9	47.2	44.2	42.4	41.8



The Cars of **gtcars**

These are some fine automobiles[1]

	Year	Trim	Transmission	*Performance* MPG	*Performance* HP	*Performance* Torque	MSRP[2]
Germany							
Audi R8	2015	4.2 (Manual) Coupe	6 Speed *Manual*	11c 20h	430 @7900rpm	317 @4500rpm	$115,900
Audi S8	2016	Base Sedan	8 Speed *Automatic/Manual*	15c 25h	520 @5800rpm	481 @1700rpm	$114,900
Italy							
Ferrari LaFerrari[3]	2015	Base Coupe	7 Speed *Automatic*	12c 16h	949 @9000rpm[5]	664 @6750rpm	$1,416,362
Ferrari F12Berlinetta[3]	2015	Base Coupe	7 Speed *Automatic*	11c 16h	731 @8250rpm	509 @6000rpm	$319,995
Lamborghini Gallardo[3]	2014	LP 550-2 Coupe	6 Speed *Automatic*	12c 20h	550 @8000rpm	398 @6500rpm	$191,900
Maserati Granturismo	2016	Sport Coupe	6 Speed *Automatic/Manual*	13c 21h	454 @7600rpm	384 @4750rpm	$132,825
United States							
Chevrolet Corvette	2016	Z06 Coupe	7 Speed *Manual*	15c 22h	650 @6400rpm	650 @3600rpm	$88,345
Dodge Viper[3]	2017	GT Coupe	6 Speed *Manual*	12c 19h	645 @5000rpm	600 @5000rpm	$95,895
Ford GT[3]	2017	Base Coupe	7 Speed *Automatic*	11c 18h	647 @6250rpm	550 @5900rpm	$447,000
Tesla Model S	2017	75D	1 Speed *Direct Drive*	—[6]	259 @6100rpm	243	$74,500
Japan							
Acura NSX	2017	Base Coupe	9 Speed *Automatic*	21c 22h	573 @6500rpm	476 @2000rpm	$156,000
Nissan GT-R	2016	Premium Coupe	6 Speed *Automatic*	16c 22h	545 @6400rpm	436 @3200rpm	$101,770

[1] Grand tourers are distinguished with a blue background.
[2] All prices in U.S. dollars (USD).
[3] Model no longer in production (as of 2018).
[4] Best gas mileage (city) of all the **gtcars**.
[5] The highest horsepower of all the **gtcars**.
[6] This is an all-electric vehicle and fuel economy is based on a miles per gallon gasonline equivalent. More information on this is available at fueleconomy.gov.

Source: Various pages within edmunds.com.

New York Air Quality Measurements

Daily measurements in New York City (May 1-10, 1973)

Time			Measurement			
Year	Month	Day	Ozone, ppbV	Solar R., cal/m^2	Wind, mph	Temp, °F
1973	5	1	41	190	7.4	67
1973	5	2	36	118	8.0	72
1973	5	3	12	149	12.6	74
1973	5	4	18	313	11.5	62
1973	5	5	—	—	14.3	56
1973	5	6	28	—	14.9	66
1973	5	7	23	299	8.6	65
1973	5	8	19	99	13.8	59
1973	5	9	8	19	20.1	61
1973	5	10	—	194	8.6	69

temperature	pressure
0	2.00×10^{-4}
20	1.20×10^{-3}
40	6.00×10^{-3}
60	3.00×10^{-2}
80	9.00×10^{-2}
100	2.70×10^{-1}
120	7.50×10^{-1}
140	1.85
160	4.20
180	8.80
200	1.73×10^{1}
220	3.21×10^{1}
240	5.70×10^{1}
260	9.60×10^{1}
280	1.57×10^{2}
300	2.47×10^{2}
320	3.76×10^{2}
340	5.58×10^{2}
360	8.06×10^{2}

	pies	income
chicken		
L	4932	$102,339.00
M	3894	$65,224.50
S	2224	$28,356.00
TOTAL	11050	$195,919.50
classic		
L	4057	$74,518.50
M	4112	$60,581.75
S	6139	$69,870.25
XL	552	$14,076.00
XXL	28	$1,006.60
TOTAL	14888	$220,053.10
supreme		
L	4564	$94,258.50
M	4046	$66,475.00
S	3377	$47,463.50
TOTAL	11987	$208,197.00
veggie		
L	5403	$104,202.70
M	3583	$57,101.00
S	2663	$32,386.75
TOTAL	11649	$193,690.45

OUTLINE

PAST
What is the **gt** package and what went into it?

PRESENT
What can **gt** do today?

FUTURE
What's in store for **gt** and *beyond*.

PAST
What is the **gt** package and what went into it?

PRESENT
What can **gt** do today?

FUTURE
What's in store for **gt** and *beyond*.

PAST

The **gt** prehistory: things we noted about table packages in **R**.

knitr	2011
kableExtra	2015
formattable	2015
DT	2014
pander	2012
huxtable	2017
flextable	2016
pixiedust	2015
tangram	2016
ztable	2014
condformat	2015
stargazer	2012
xtable	2000

← Here's all the packages we looked at in early 2018.

They had strengths in different areas, some were noted to be domain-specific while others were general.

Most handled one, maybe two, sometimes a few output formats.

We wanted to make something **ggplot**-like, focusing on non-interactive tables that deliver camera-ready output (in different formats).

PAST

We developed our core requirements for **gt**.

---

1 // A declarative but forgiving API.

```
library(gt)

exibble %>%
  gt(rowname_col = "row", groupname_col = "group") %>%
  tab_source_note(source_note = "Source note.") %>%
  tab_footnote(
    footnote = "This is a footnote.",
    locations = cells_body(columns = 1, rows = 1)
  ) %>%
  tab_header(
    title = "The title of the table",
    subtitle = "The table's subtitle"
  )
```

Should always start with `gt()`.

Subsequent statements could be expressed in any order.

Each function acts as instructions. **gt** should then figure out what to render.

PAST

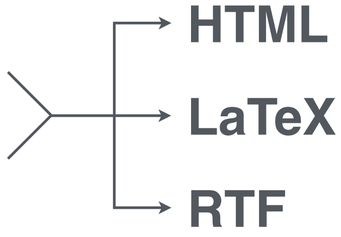
We developed our core requirements for **gt**.

---

2 // Table generation in multiple output types with the same API.

```r
library(gt)

exibble %>%
  gt(rowname_col = "row", groupname_col = "group") %>%
  tab_source_note(source_note = "Source note.") %>%
  tab_footnote(
    footnote = "This is a footnote.",
    locations = cells_body(columns = 1, rows = 1)
  ) %>%
  tab_header(
    title = "The title of the table",
    subtitle = "The table's subtitle"
  )
```



The set of statements supplied to **gt** should not change depending on the rendering context.

gt should always work when taking the same code used for HTML table over to a LaTeX **R Markdown** document.

PAST

We developed our core requirements for **gt**.

3 // Powerful formatters for cell values.

We ought to be able to format numbers, dates, strings, etc., with very flexible functions.

UNFORMATTED

Value									
1.2	1.20	1	1.20	+1.2%	0.0012	£1.20	1 B	<1 B>
30.3	30.30	30	3.03×10^{1}	+30.3%	0.030	£30.30	30 B	<30 B>
1023	1,023.00	1.023	1.02×10^{3}	+1,023%	1023	£1023.00	1 kB	<1 kB>
34502.4	34,502.40	34.502	3.45×10^{4}	+34,502.4%	34502.4	£34502.40	34.5 kB	<34.5 kB>
-7900345	–7,900,345.00	–7.900.345	-7.90×10^{6}	–7,900,345%	–7900345	(£7900345.00)	–7.9 MB	<–7.9 MB>
9.23	9.23	9	9.23	+9.2%	9.23	£9.23	9 B	<9 B>

- format as integers
- change separators via locale codes

- use % signs
- don't scale by 100
- force '+' sign

- use currency codes to format
- remove separators
- accounting notation

- decorate formatted values using a pattern

- format to an exact number of decimal places

- use scientific notation

- scale values manually
- use sig. digits

- express values in bytes

PAST

We developed our core requirements for **gt**.

4 // Methods for restructuring table data.

We ought to be able to express how tables are structured within the **gt** API. Some should be automatic, some manual.

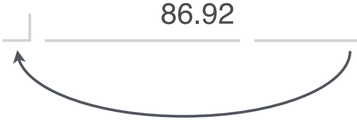
- columns gathered together when placed under a column spanner

Column 1	Column 2	Column 3
23.42	–	15.24
63.90	21.34	43.70
—	61.93	26.00
1.29	17.60	15.58
–28.02	–10.55	–5.23
86.92	65.23	47.25

	Column Spanner	
Column 1	Column 3	Column 2
23.42	15.24	–
63.90	43.70	21.34
—	26.00	61.93
1.29	15.58	17.60
–28.02	–5.23	–10.55
86.92	47.25	65.23

- move columns manually

Column 1	Column 2
23.42	–
63.90	21.34
—	61.93
1.29	17.60
–28.02	–10.55
86.92	65.23



PAST

We developed our core requirements for **gt**.

---

5 // Pain-free footnotes.

BASIC FEATURES

It should be straightforward to define table footnotes in **gt**.

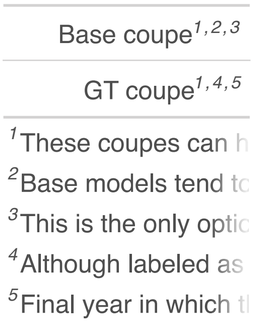
gt should always express the ordering of footnotes automatically.

MORE ADVANCED FEATURES

We should be able to apply the same footnote to multiple locations.

gt should preserve the same footnote mark

Multiple footnotes should be allowed at the same location.



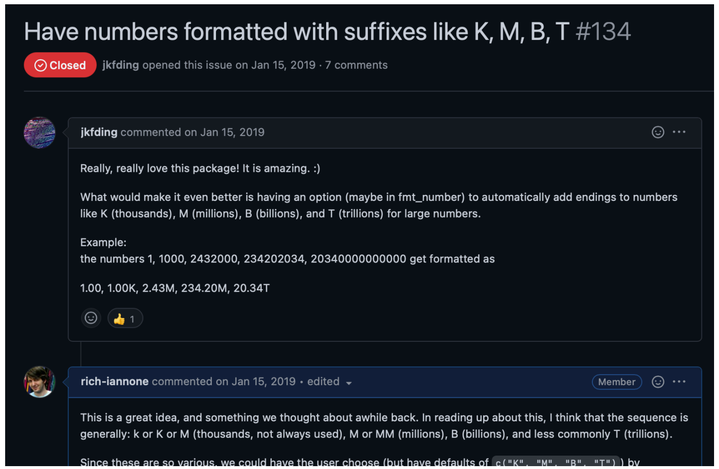
gt should handle complex footnote marks in a logical manner

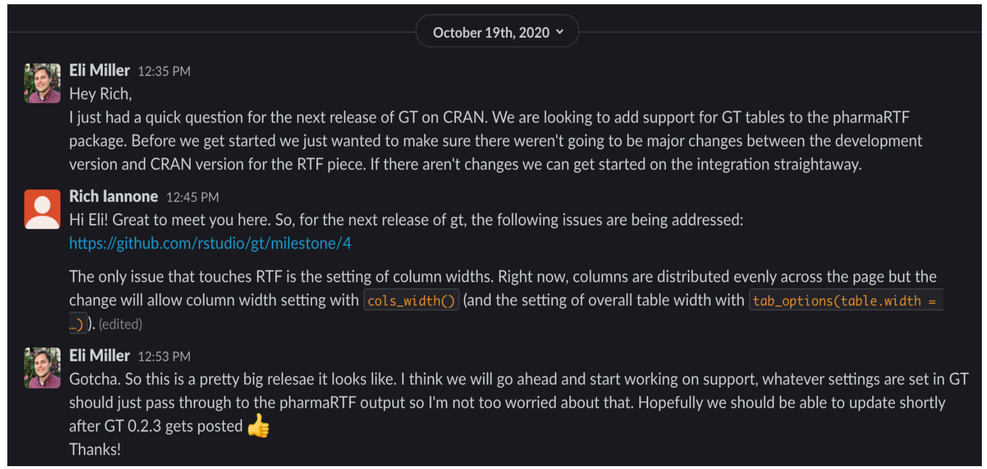
PAST

We made **gt** public and listened to our users.

We publicly released **gt** in late 2018 (having LaTeX and RTF outputs as *experimental* output types).

Since then, we've had numerous discussions on social media, in **GitHub** issues, and through customer conversations.





Through all of that, we learned very quickly what was successful and, more importantly, what wasn't so great about the initial release.

PAST

We made **gt** public and listened to our users.

We fixed a lot of the initial technical problems and learned through the first 1–2 years of public release what users wanted most.

Improved RTF output

Improved LaTeX output

Additional cell formatters

- `fmt_engineering()`
- `fmt_bytes()`
- `fmt_integer()`
- `fmt_markdown()`
- `fmt_missing()`

More ways to style cells

- `data_color()`
- `cell_text()`
- `cell_fill()`
- `cell_borders()`
- `google_font()`

Improvements to table structuring

API enhancements

More!

PAST

We made **gt** public and listened to our users.

In terms of customer conversations with people in Pharma we discovered what the biggest requirements were.

WE HAD GREAT CONVERSATIONS WITH:

Merck: Yilong Zhang

Pfizer: Tom Tensfeldt and Mike K. Smith

Atorus Research: Mike Stackhouse

Metrum Research Group: Devin Pastoor and Anna Nevison

Roche: Adrian Waddell and Gabe Becker

PAST

We made **gt** public and listened to our users.

In terms of customer conversations with people in Pharma we discovered what the biggest requirements were.

We fixed **gt**'s RTF and LaTeX output support, making it more reliable.

To get more dialogue going about what Pharma needs, a working group was started to discuss development of **R** packages suited to these needs.

RTRS // R Tables for Regulatory Submissions Working Group

R consortium

"Creating standards to guide the development of R tools for creating tables that meet the requirements of the FDA and other regulatory agencies"

Meets every six weeks via Zoom to discuss and plan improvements to the R table ecosystem for benefit of Pharma. All are welcome to join.

CONTACT

Joe Rickert
R Community Ambassador
joseph.rickert@rstudio.com

PAST
What is the **gt** package and what went into it?

PRESENT
What can **gt** do today?

FUTURE
What's in store for **gt** and *beyond*.

PAST

What is the **gt** package and what went into it?

PRESENT

What can **gt** do today?

FUTURE

What's in store for **gt** and *beyond*.

PRESENT

You can create really nice presentation tables in three output formats.

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	€49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	€17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	€1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	€65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	€1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	€13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	€0.44

HTML

PRESENT

You can create really nice presentation tables in three output formats.

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	EUR49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	EUR17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	EUR1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	EUR65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	EUR1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	EUR13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	EUR0.44

LaTeX

PRESENT

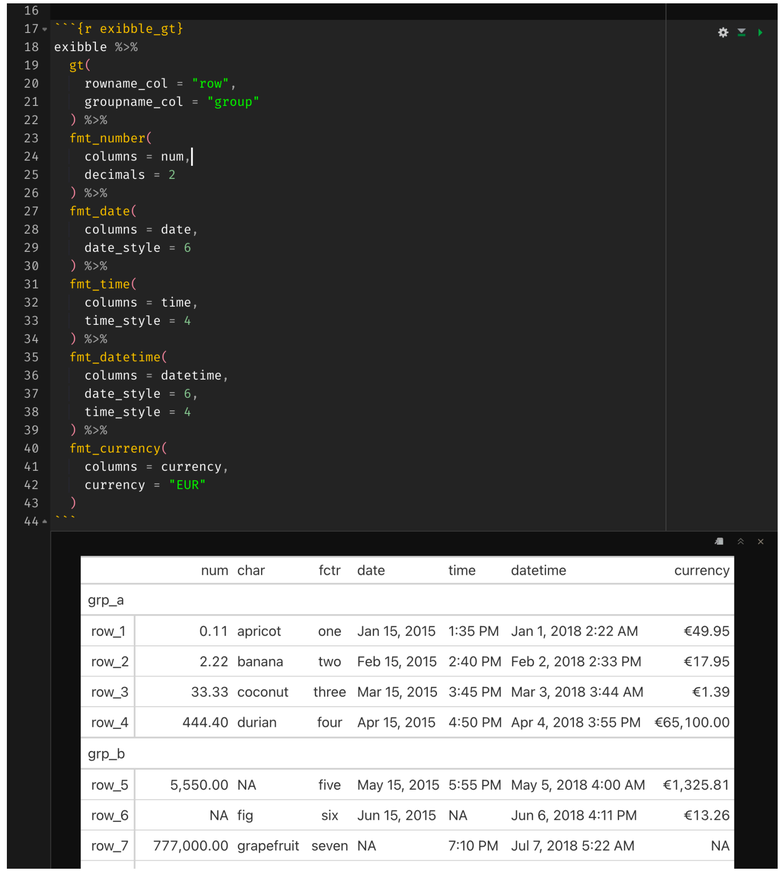
You can create really nice presentation tables in three output formats.

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	EUR49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	EUR17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	EUR1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	EUR65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	EUR1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	EUR13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	EUR0.44

RTF

PRESENT

gt works nicely within **R Markdown** and **Quarto**.



```{r exibble_gt}
exibble %>%
  gt(
    rowname_col = "row",
    groupname_col = "group"
  ) %>%
  fmt_number(
    columns = num,
    decimals = 2
  ) %>%
  fmt_date(
    columns = date,
    date_style = 6
  ) %>%
  fmt_time(
    columns = time,
    time_style = 4
  ) %>%
  fmt_datetime(
    columns = datetime,
    date_style = 6,
    time_style = 4
  ) %>%
  fmt_currency(
    columns = currency,
    currency = "EUR"
  )
```

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	€49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	€17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	€1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	€65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	€1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	€13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA

.Rmd / .qmd

code chunks within the document. You can embed an R code chunk like this:

```
exibble %>%
  gt(
    rowname_col = "row",
    groupname_col = "group"
  ) %>%
  fmt_number(
    columns = num,
    decimals = 2
  ) %>%
  fmt_date(
    columns = date,
    date_style = 6
  ) %>%
  fmt_time(
    columns = time,
    time_style = 4
  ) %>%
  fmt_datetime(
    columns = datetime,
    date_style = 6,
    time_style = 4
  ) %>%
  fmt_currency(
    columns = currency,
    currency = "EUR"
  )
```

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	€49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	€17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	€1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	€65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	€1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	€13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	€0.44

R Markdown HTML

document. You can embed an R code chunk like this:

```
exibble %>%
  gt(
    rowname_col = "row",
    groupname_col = "group"
  ) %>%
  fmt_number(
    columns = num,
    decimals = 2
  ) %>%
  fmt_date(
    columns = date,
    date_style = 6
  ) %>%
  fmt_time(
    columns = time,
    time_style = 4
  ) %>%
  fmt_datetime(
    columns = datetime,
    date_style = 6,
    time_style = 4
  ) %>%
  fmt_currency(
    columns = currency,
    currency = "EUR"
  )
```

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	€49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	€17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	€1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	€65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	€1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	€13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	€0.44

Quarto HTML

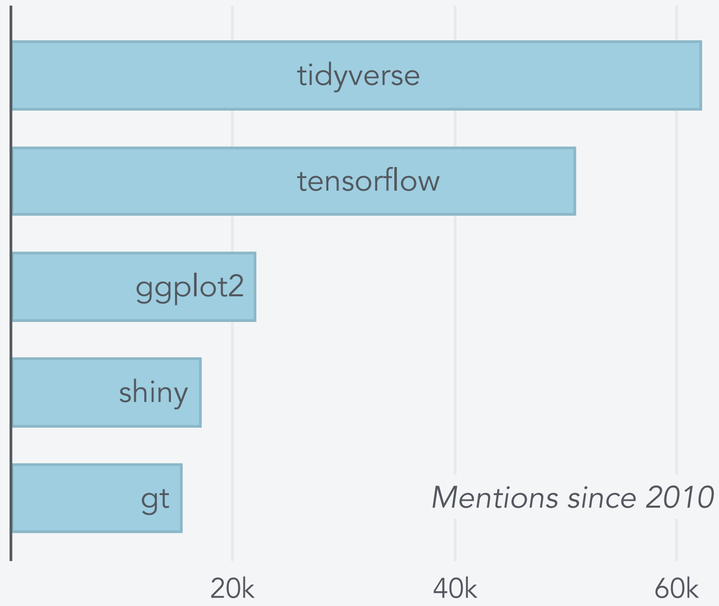
PRESENT

Users across industry and academia have been using **gt** quite a lot.

We are seeing evidence of a fairly large and enthusiastic user base.

Here are some of the metrics that show signs of success:

Highly mentioned **R** pkgs on **Twitter**



GitHub stars

1.4k⭐

Closed **GitHub** issues

350+

Presentations given

8

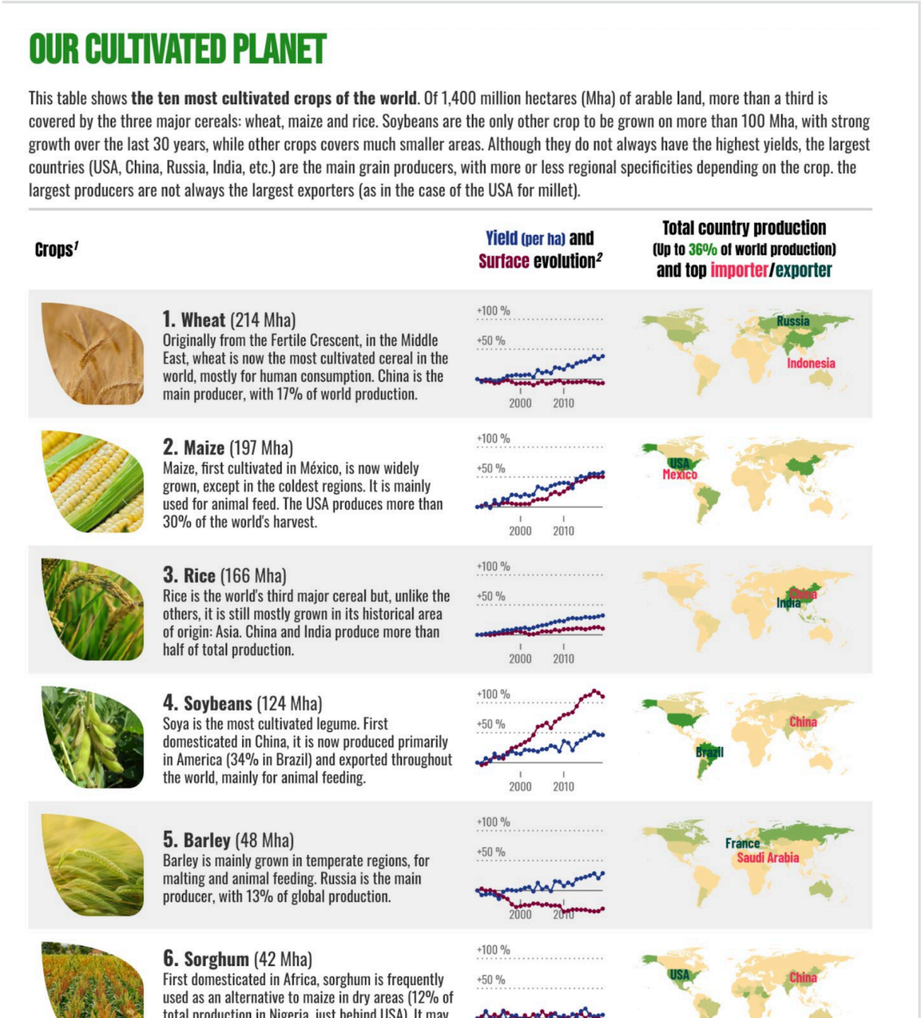
Workshops delivered

6

PRESENT

Users across industry and academia have been using **gt** quite a lot.

People on **Twitter** are excited to share their made-with-**gt** tables.



OUR CULTIVATED PLANET

This table shows **the ten most cultivated crops of the world**. Of 1,400 million hectares (Mha) of arable land, more than a third is covered by the three major cereals: wheat, maize and rice. Soybeans are the only other crop to be grown on more than 100 Mha, with strong growth over the last 30 years, while other crops covers much smaller areas. Although they do not always have the highest yields, the largest countries (USA, China, Russia, India, etc.) are the main grain producers, with more or less regional specificities depending on the crop. the largest producers are not always the largest exporters (as in the case of the USA for millet).

SEATTLE SEAHAWKS

	EPA/play	SR	1st%	Plays
All plays	0.16	45	28	60
Rush	-0.20	27	20	30
Pass	0.52	63	37	30
Early downs (1st & 2nd)	0.09	42	20	45
Early rush	-0.12	26	19	27
Early pass	0.40	67	22	18
Late downs (3rd & 4th)	0.35	53	53	15
Late rush	-0.99	33	33	3
Late pass	0.69	58	58	12

BRON
D E N

	EPA/
All plays	
Rush	
Pass	
Early downs (1st & 2nd)	
Early rush	
Early pass	
Late downs (3rd & 4th)	
Late rush	
Late pass	

PRESENT

Users across industry and academia have been using **gt** quite a lot.

We held the **RStudio Table Contest** in late 2020 and saw a lot of **gt** tables.

What do I binge next? A detailed overview of the best TV documentaries as rated by IMDb users

		Title (Start Year)	Runtime	Genres	Rating	Votes	Overview	Top-Rated	Rating Trends: Are episodes worse or better than the show's average?[1]
2	↑+2	Planet Earth (2006)	49	Documentary	9.4 ★	164,122	1 season 11 episodes	S01E01	
8	↓−2	Our Planet (2019)	50	Documentary	9.3 ★	29,878	1 season 8 episodes	S01E02	
9	•	Cosmos: A Spacetime Odyssey (2014)	43	Documentary	9.2 ★	108,623	1 season 13 episodes	S01E13	
10	•	Cosmos (1980)	60	Documentary	9.2 ★	35,828	1 season 13 episodes	S01E08	
20	↓−2	The Vietnam War (2017)	41	Documentary History War	9.1 ★	19,387	1 season 10 episodes	S01E06 S01E10	
23	↓−3	Human Planet (2011)	50	Documentary	9.0 ★	23,284	1 season 8 episodes	S01E04	
25	↑+5	The Blue Planet (2001)	39	Documentary	9.0 ★	33,917	1 season 10 episodes	S01E02	
26	NEW!	The Beatles Anthology (1995)	75	Documentary Music	9.0 ★	7,270	1 season 8 episodes	S01E04 S01E05 S01E06 S01E07 S01E08	
28	↑+5	Frozen Planet (2011)	33	Documentary	9.0 ★	26,205	1 season 10 episodes	S01E01	
33	↓−11	The Civil War (1990)	76	Documentary History War	8.9 ★	13,272	1 season 9 episodes	S01E05 S01E08 S01E09	
34	↑+7	Der Krieg (2009) (Apocalypse: La 2ème guerre mondiale)	52	Documentary History War	8.9 ★	9,484	1 season 6 episodes	S01E06	
37	↓−2	Africa (2013)	60	Documentary	8.9 ★	13,779	1 season 6 episodes	S01E01	
52	↑+16	Nathan for You (2013)	30	Comedy Documentary	8.8 ★	18,074	4 seasons 32 episodes	S04E08	
54	NEW!	How the Universe Works (2010)	60	Documentary	8.8 ★	5,593	8 seasons 71 episodes	S07E10	

2020 RStudio Table Contest Submission by A. Calatroni, S. Lussier & R. Krouse Repo

Burgernomics: The Big Mac Index for G20 compared to the US Dollar[1]

Measuring the purchasing power parity (PPP) between two currencies[2]

	Country	Currency (ABBR)[4]	Latest Price[3] Local[5]	Latest Price[3] Dollar[6]	USD Under/Over Raw[7]	USD Under/Over Adjusted[8]	Historic 2000 → 2011 → 2020 Raw[7]
	Switzerland	Franc (CHF)	SFr.6.50	$6.91	20.9%	4.3%	
	United States	Dollar (USD)	$5.71[1]	$5.71[1]			
	Canada	Dollar (CAD)	$6.88	$5.08	–11.1%	2.3%	
	Euro area	Euro (EUR)	€4.21	$4.79	–16.2%	2.0%	
	Australia	Dollar (AUD)	$6.55	$4.58	–19.8%	–15.5%	
	Brazil	Real (BRL)	R$20.90	$3.91	–31.5%	19.1%	
	South Korea	Won (KRW)	₩4,500	$3.75	–34.4%	–14.5%	
	Saudi Arabia	Riyal (SAR)	SR14.00	$3.73	–34.6%	–5.3%	

PRESENT

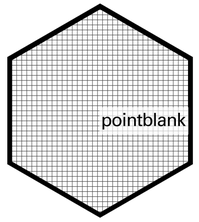
gt can be used by other **R** packages.

Just like **dplyr** and **ggplot2**, we can program with **gt** to prepare tabular outputs.

Characteristic	Tumor Response			Time to Death		
	OR[1]	95% CI[1]	p-value	HR[1]	95% CI[1]	p-value
Chemotherapy Treatment						
Drug A	—	—		—	—	
Drug B	1.13	0.60, 2.13	0.7	1.30	0.88, 1.92	0.2
Age	1.02	1.00, 1.04	0.10	1.01	0.99, 1.02	0.3
Grade						
I	—	—		—	—	
II	0.85	0.39, 1.85	0.7	1.21	0.73, 1.99	0.5
III	1.01	0.47, 2.15	>0.9	1.79	1.12, 2.86	0.014

[1] OR = Odds Ratio, CI = Confidence Interval, HR = Hazard Ratio

gtsummary



Pointblank Validation

A local table test.

TIBBLE pointblank::small_table | WARN 0.10 | STOP 0.20 | NOTIFY —

STEP	COLUMNS	VALUES	TBL	EVAL	UNITS	PASS	FAIL	W	S	N	EXT
1 col_vals_gt()	date_time	date	↪	⚠	13	13 1.00	0 0.00	○	○	—	—
2 col_vals_gt()	b	g	↪	💥	–	–	–	–	–	–	–
3 col_vals_regex()	b	[1-9]-[a-z]{3}-…	↪	✓	13	13 1.00	0 0.00	○	○	—	—
4 rows_distinct()	—	—	↪	✓	13	11 0.85	2 0.15	●	○	—	CSV
5 col_vals_gt()	d	100	↪	✓	13	13 1.00	0 0.00	○	○	—	—
6 col_vals_equal()	d	d	↪	✓	13	13 1.00	0 0.00	○	○	—	—
7 col_vals_between()	c	[a, d]	↪	✓	13	9 0.69	4 0.31	●	●	—	CSV
8 col_vals_not_between()	c	[10, 20]	↪	✓	13	13 1.00	0 0.00	○	○	—	—
9 rows_distinct()	d, e, f	—	↪	✓	13	11 0.85	2 0.15	●	○	—	CSV
10 col_is_integer()	a	—	↪	✓	1	1 1.00	0 0.00	○	○	—	—

pointblank

gtExt

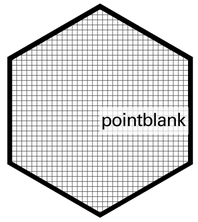
PLAYER	DUNKS	DU
Evan Mobley	40	
Sandro Mamukelashvili	48	
Charles Bassey	50	
Luke Garza	50	
Moses Wright	51	
Neemias Queta	55	
Isaiah Jackson	60	
Day'Ron Sharpe	66	

gtExt

PRESENT

gt can be used by other **R** packages.

Just like **dplyr** and **ggplot2**, we can program with **gt** to prepare tabular outputs.



Pointblank Validation

A local table test.

TIBBLE	pointblank::small_table	WARN	0.10	STOP	0.20	NOTIFY	—

	STEP	COLUMNS	VALUES	TBL	EVAL	UNITS	PASS	FAIL	W	S	N	EXT
1	col_vals_gt()	date_time	date	↪	⚠	13	13 1.00	0 0.00	○	○	—	—
2	col_vals_gt()	b	g	↪	💥	—	—	—	—	—	—	—
3	col_vals_regex()	b	[1-9]-[a-z]{3}-…	↪	✓	13	13 1.00	0 0.00	○	○	—	—
4	rows_distinct()	—	—	↪	✓	13	11 0.85	2 0.15	●	○	—	CSV
5	col_vals_gt()	d	100	↪	✓	13	13 1.00	0 0.00	○	○	—	—
6	col_vals_equal()	d	d	↪	✓	13	13 1.00	0 0.00	○	○	—	—
7	col_vals_between()	c	[a, d]	↪	✓	13	9 0.69	4 0.31	●	●	—	CSV
8	col_vals_not_between()	c	[10, 20]	↪	✓	13	13 1.00	0 0.00	○	○	—	—
9	rows_distinct()	d, e, f	—	↪	✓	13	11 0.85	2 0.15	●	○	—	CSV
10	col_is_integer()	a	—	↪	✓	1	1 1.00	0 0.00	○	○	—	—

pointblank

gtExtras

PLAYER	DUNKS	DUNKS‖HOOKS/FLOATS‖JUMPERS
Evan Mobley	40	40 | 26 | 33
Sandro Mamukelashvili	48	48 | 10 | 42
Charles Bassey	50	50 | 19 | 31
Luke Garza	50	50 | 15 | 35
Moses Wright	51	51 | 25 | 25
Neemias Queta	55	55 | 27 | 18
Isaiah Jackson	60	60 | 15 | 25
Day'Ron Sharpe	66	66 | 24 | 10

gtExtras

PRESENT

Where does **gt** fall flat?

We learned that several users in Pharma don't have their table requirements fulfilled by **gt**.

Some table typologies that are needed by Pharma are difficult to make with **gt**:

	Column 1	Column 2	Column 3	Column 4	Column 5
numeric values	7.24	15.24	2.78	73.23	3.26
	12.75	36.87	13.40	–	73.38
	43.28	–	15.24	2.12	8.36
	63.90	21.34	43.70	75.87	47.55
	—	61.93	26.00	41.06	35.93
character values	true	true	false	true	false
	primary	secondary	primary	unknown	primary
	false	true	true	NA	true

Heterogeneous columns

PRESENT

Where does **gt** fall flat?

We learned that several users in Pharma don't have their table requirements fulfilled by **gt**.

Some table typologies that are needed by Pharma are difficult to make with **gt**:

level 1 spanners

level 2 spanner

	Column Spanner Over Top				
	Column Spanner 1		Column Spanner 2		
	Column 1	Column 2	Column 3	Column 4	Column 5
	23.42	–	15.24	2.12	8.36
	63.90	21.34	43.70	75.23	47.55
	—	61.93	26.00	43.07	35.93
	1.29	17.60	15.58	13.47	11.78
	–28.02	–10.55	–5.23	–0.49	3.52
	86.92	65.23	47.25	35.29	47.20

Multiple Spanner Levels

PRESENT

Where does **gt** fall flat?

We learned that several users in Pharma don't have their table requirements fulfilled by **gt**.

Some table typologies that are needed by Pharma are difficult to make with **gt**:

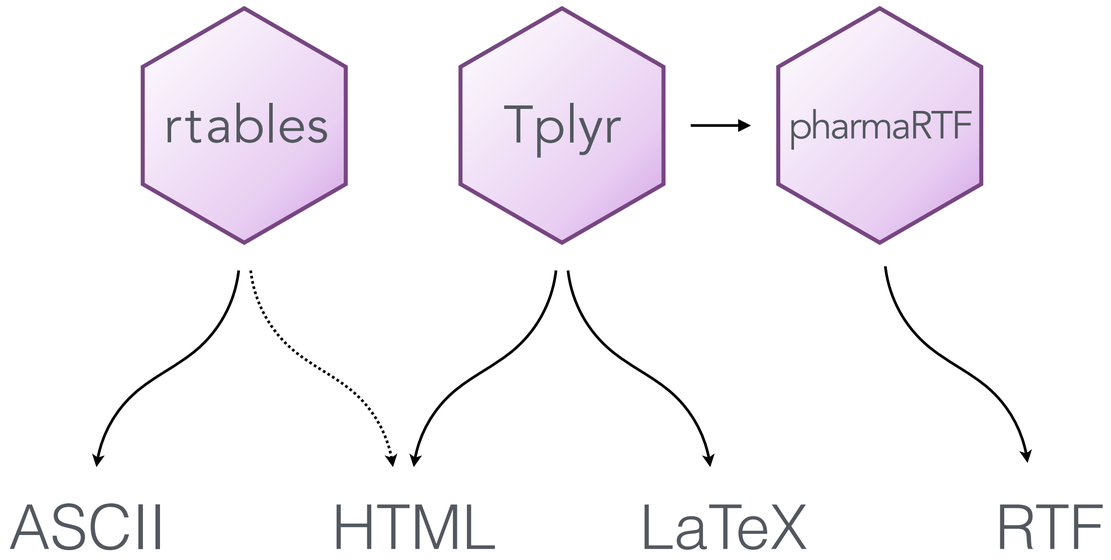
		Column 1	Column 2	Column 3	Column 4
	Location 1	7.24	15.24	2.78	73.23
indents	Alt 1	12.75	36.87	13.40	–
indents	Alt 2	43.28	–	15.24	2.12
indents	Alt 3	63.90	21.34	43.70	75.87
	Secondary	—	61.93	26.00	41.06
indents	Alt 1	36.47	83.72	37.82	29.00
indents	Alt 2	2.63	5.21	9.92	4.87
indents	Alt 3	4.03	5.29	5.21	5.83

Indentation Levels in the Table Stub

PRESENT

Also, there are several great Pharma-focused table packages.

We also see many really great table generation packages with APIs tuned directly into Pharma's needs. However, output formats are limited.



These packages are ideal for Pharma but rendering support is a mixed bag.

PAST

What is the **gt** package and what went into it?

PRESENT

What can **gt** do today?

FUTURE

What's in store for **gt** and *beyond*.

PAST

What is the **gt** package and what went into it?

PRESENT

What can **gt** do today?

FUTURE

What's in store for **gt** and *beyond*.

FUTURE

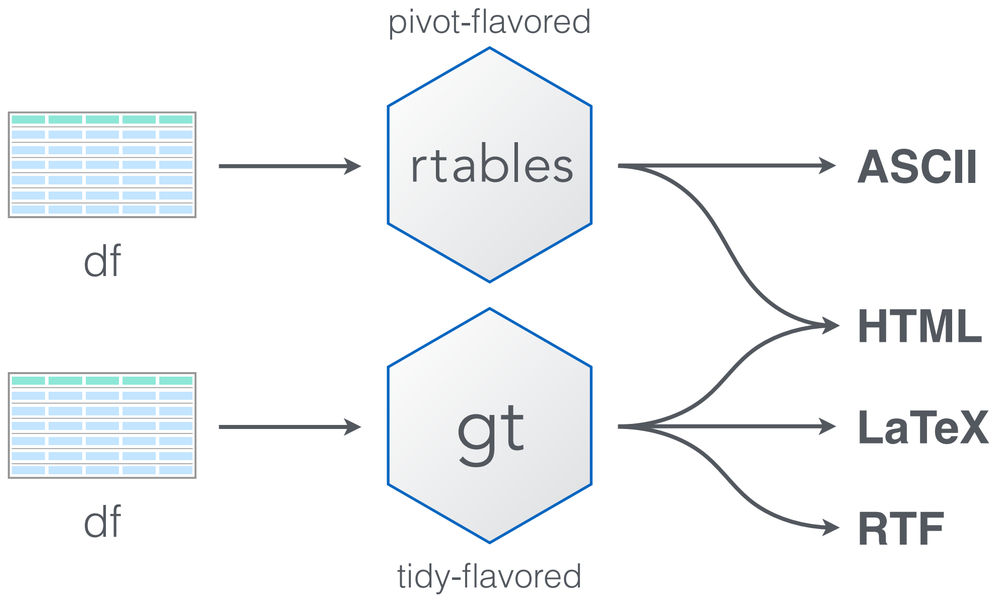
What are we going to do next? Break things apart.

While **gt** is general, a large amount of effort was put into rendering for multiple formats (HTML, LaTeX, RTF).

FUTURE

What are we going to do next? Break things apart.

While **gt** is general, a large amount of effort was put into rendering for multiple formats (HTML, LaTeX, RTF).

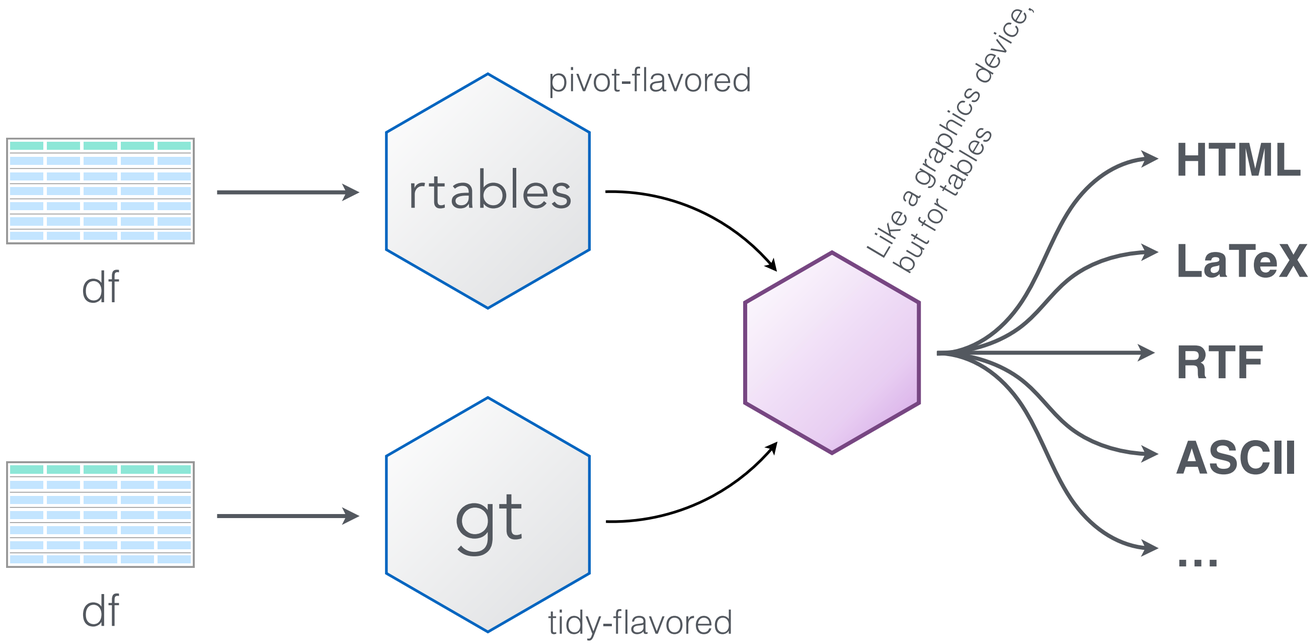


Adrian Waddell and Gabe Becker at Roche have made **rtables**; it's perfect for what Pharma needs but has different support for table outputs.

FUTURE

What are we going to do next? Break things apart.

While **gt** is general, a large amount of effort was put into rendering for multiple formats (HTML, LaTeX, RTF).



What if we made a separate library for rendering that others can use?

FUTURE

What are we going to do next? Break things apart.

It turns out that we think this is a good idea, worth the development investment, and is underway now as **tgen**.

We are keeping **tgen** under wraps for a bit longer, mostly to save developers from what is alpha-grade software at this point.

This is the first mention of **tgen** at any conference and so think of this an an early look.

FUTURE

Goals for **tgen**.

tgen isn't meant to be used directly by **R** users, but through packages.

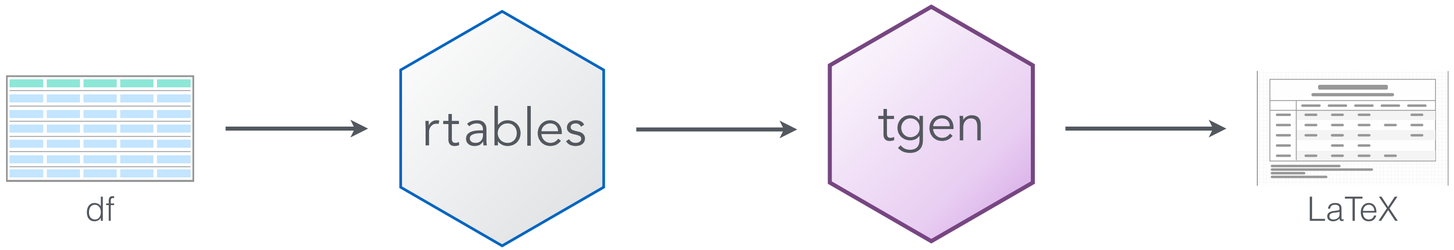
tgen will contain **R** functions for describing tables using a low-level API (e.g., `row()` and `cell()` constructor functions).

tgen is designed to be a good fit for any table packages making static tables (present and future).

FUTURE

tgen in Action! (**tgen** + **rtables**)

rtables makes pharma-focused tables. **tgen** renders. Let's combine them!



The **gt** team created a fork of **rtables** that interfaces with the **tgen R** package, providing support for LaTeX rendering (which **rtables** doesn't currently have).

Demo

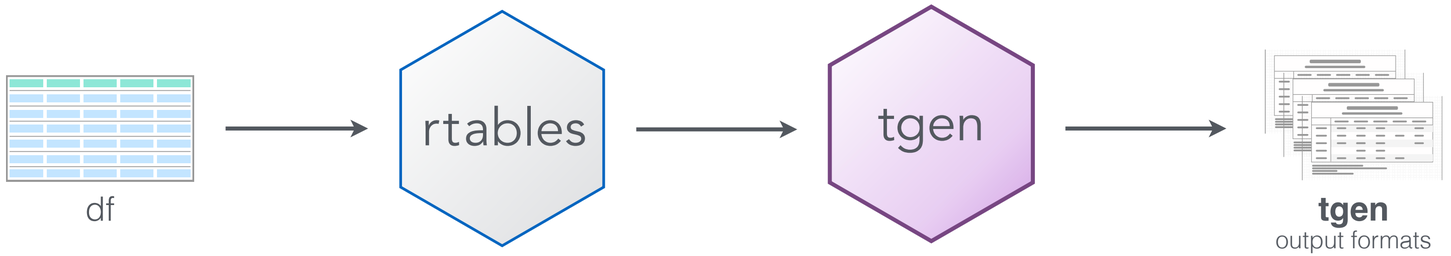
FUTURE

tgen in Action! (**tgen** + **rtables**)

Demo

Because **tgen** handled the rendering, the **rtables** team needn't focus much on the rendering aspect. They only need to use **tgen** for the rendering.

They can focus their efforts on making great table APIs for the Pharma industry.



FUTURE

Extending **tgen** for Pharma.

We plan to extend **tgen** with features that Pharma can make use of.

Call to Developers of Table Packages

The **gt/tgen** team wants to talk to you about what you want and need from **tgen**. Please get in touch with us!

We recognize there are table requirements that are particular to regulatory submissions and we will address those in **tgen**.

All **GitHub** issues will be seen by the **gt/tgen** team and we'll actively participate in discussions with developers in Pharma.

We will continue to prioritize any action items arising from such engagement.

FUTURE

Roche can extend their stack with **tgen**.

A logical endpoint: access to an eventual package that makes regulatory table creation as simple as a single, standardized function call.

Roche's vision of this

rtables: Non-Pharma-specific package for turning data frames into formatted summary tables (hierarchical, multi-dimensional grouping operations that **dplyr** isn't great at). Open source and on CRAN.

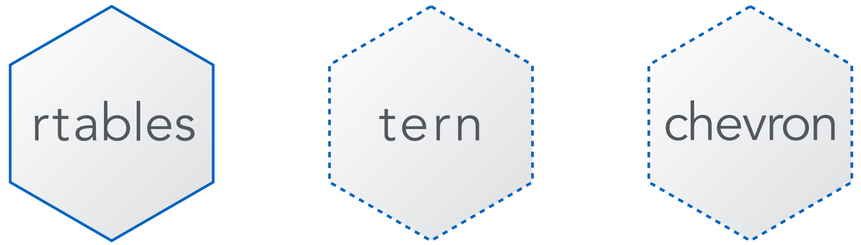
tern: Builds on **rtables** by providing many helper functions, for analyses that are common for Pharma (adverse event tables, etc.). Currently closed source and a WIP.

chevron: Builds on **rtables** + **tern**, provides functions that implement specific FDA submission tables—one function per table. Currently closed source and a WIP.

FUTURE

Roche can extend their stack with **tgen**.

A logical endpoint: access to an eventual package that makes regulatory table creation as simple as a single, standardized function call.

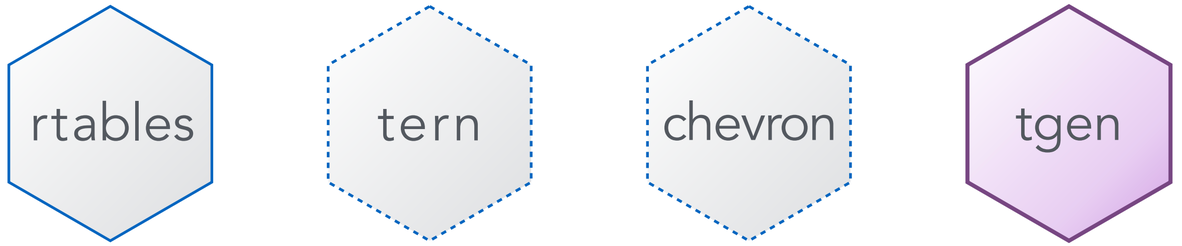


Our friends at Roche are working to get permission to open source **tern** and **chevron**, but don't have it yet.

FUTURE

Roche can extend their stack with **tgen**.

A logical endpoint: access to an eventual package that makes regulatory table creation as simple as a single, standardized function call.



Our friends at Roche are working to get permission to open source **tern** and **chevron**, but don't have it yet.

By combining Roche's stack with **tgen**, it will be possible for you to take advantage of these tools regardless of whether your workflow is based on ASCII, HTML, RTF, or PDF.

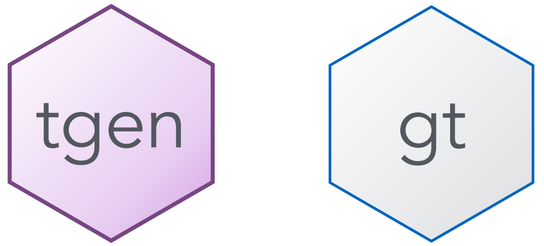
FUTURE

What's in store for the **gt**/**tgen** team?

We want to focus for a bit on **tgen** and make it work for **gt** and help others make it work for their own packages.

We want to have support for HTML, RTF, and LaTeX tables and migrate the rendering in **gt** to be **tgen**-mediated.

CORE PRIORITIES



FUTURE

That handles the *beyond* part, what about **gt** proper?

We are still keeping on top all the things that are important for users of **gt**!

We won't do much on table rendering in **gt**, of course, since **tgen** will eventually take full responsibility of that.

---

gt still needs a lot of work and many more things can (and should) be added.

TO WORK ON:

Better formatters; more of them.

Ability to easily make manual changes (e.g., adding rows, merging cells)

Unlimited levels for spanner labels.

Enhancements for footnotes.

Ways to access values from a **gt** table (for **R Markdown** and **Quarto**).

EDUCATION & OUTREACH

Learning about how to use **gt**.

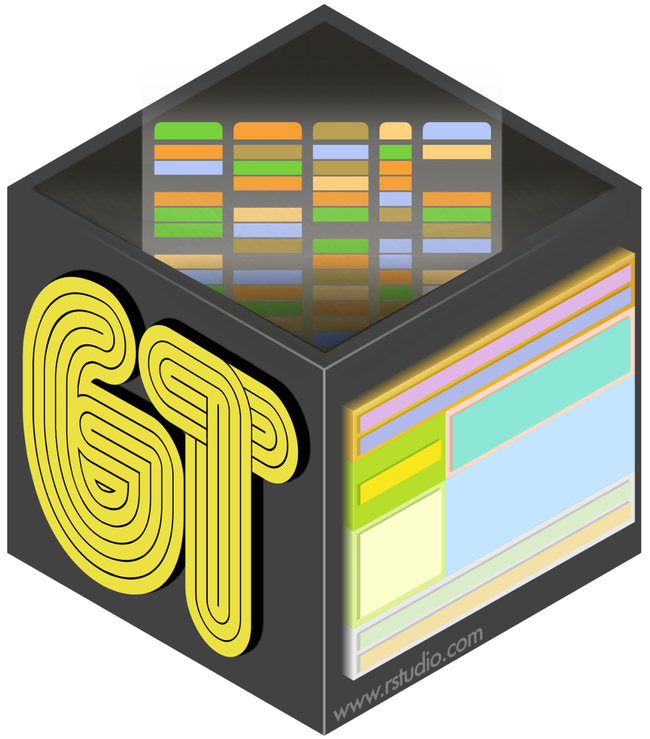
I've provided a workshop here at **R/Pharma**:

Workshop Clinical Tables in **gt**	Thursday, October 28, 2021 10:00 AM – 12:30 PM EDT	Workshop was recorded.

I'll be putting my streaming hat on and teaching **gt** (and maybe some other **R** things) in the far more causal setting of **Twitch**.

I'm always available to conduct more workshops, just ask if you should need it.

Have any questions about **gt**? Contact me, Phil Bowsher, or Jason Milnes.



https://github.com/rich-iannone/presentations

rich-iannone @riannone rich@rstudio.com