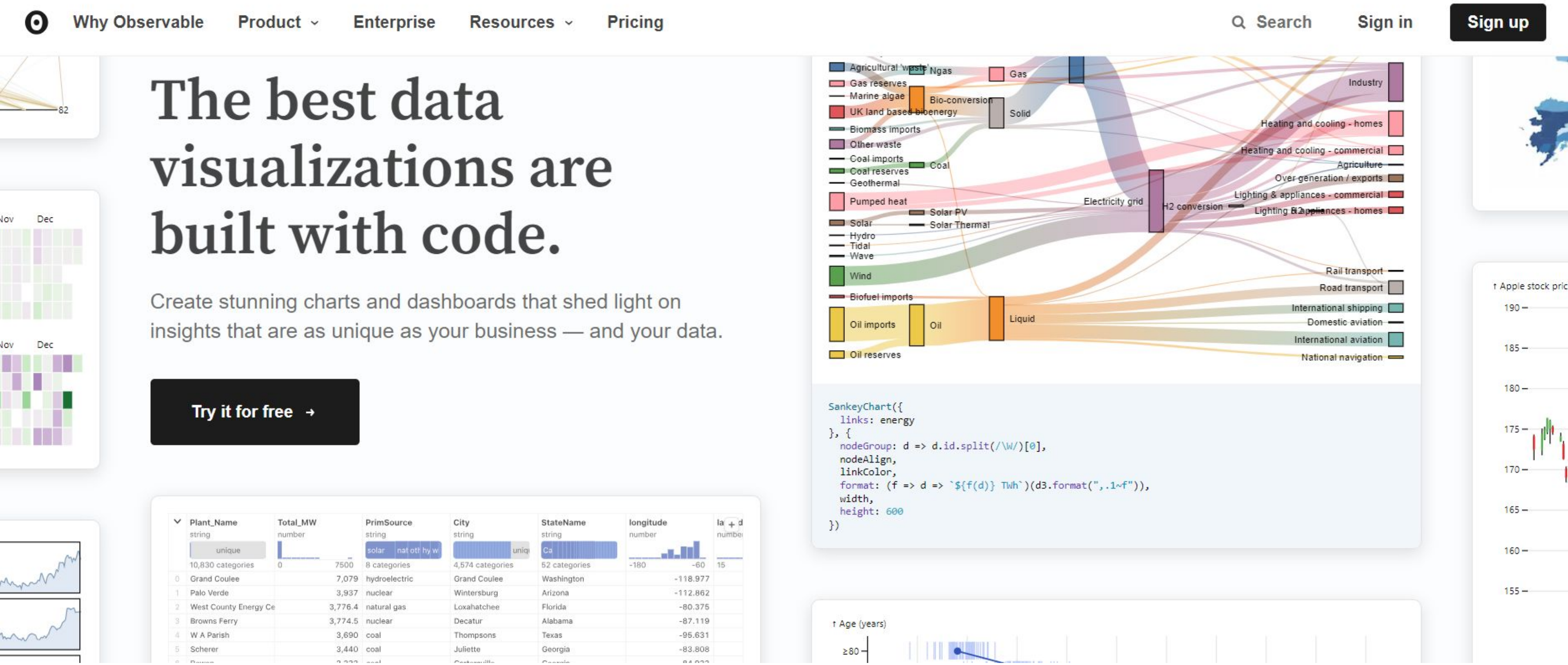
Introduction to ObservableHQ

D3.js and Plot

Tutorial for Interactive Information Visualization Course

10/02/2025

Katerina Batziakoudi
aikaterini.batziakoudi@inria.fr



The best data visualizations are built with code.

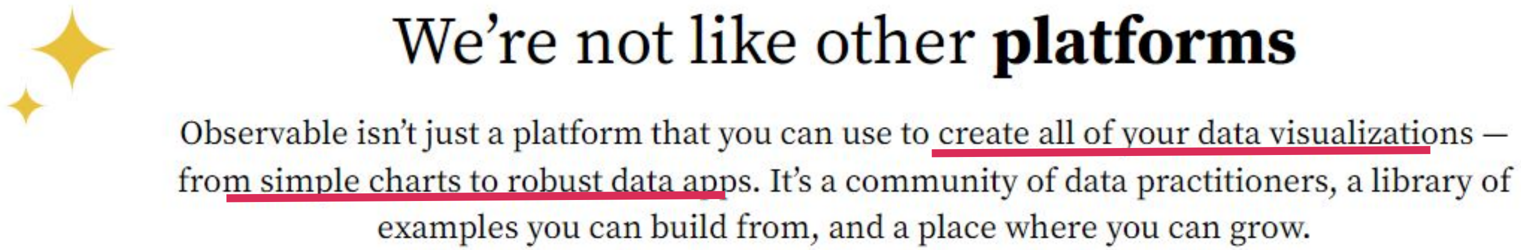
Create stunning charts and dashboards that shed light on insights that are as unique as your business — and your data.

	Plant_Name	Total_MW	PrimSource	City	StateName	longitude
	string	number	string	string	string	number
	10,830 categories	0 7500	8 categories	4,574 categories	52 categories	-180 -60
0	Grand Coulee	7,079	hydroelectric	Grand Coulee	Washington	-118.977
1	Palo Verde	3,937	nuclear	Wintersburg	Arizona	-112.862
2	West County Energy Ce	3,776.4	natural gas	Loxahatchee	Florida	-80.375
3	Browns Ferry	3,774.5	nuclear	Decatur	Alabama	-87.119
4	W A Parish	3,690	coal	Thompsons	Texas	-95.631
5	Scherer	3,440	coal	Juliette	Georgia	-83.808

```
SankeyChart({
  links: energy
}, {
  nodeGroup: d => d.id.split(/\W/)[0],
  nodeAlign,
  linkColor,
  format: (f => d => `${f(d)} TWh`)(d3.format(",.1~f")),
  width,
  height: 600
})
```

https://observablehq.com/

Why Observable?

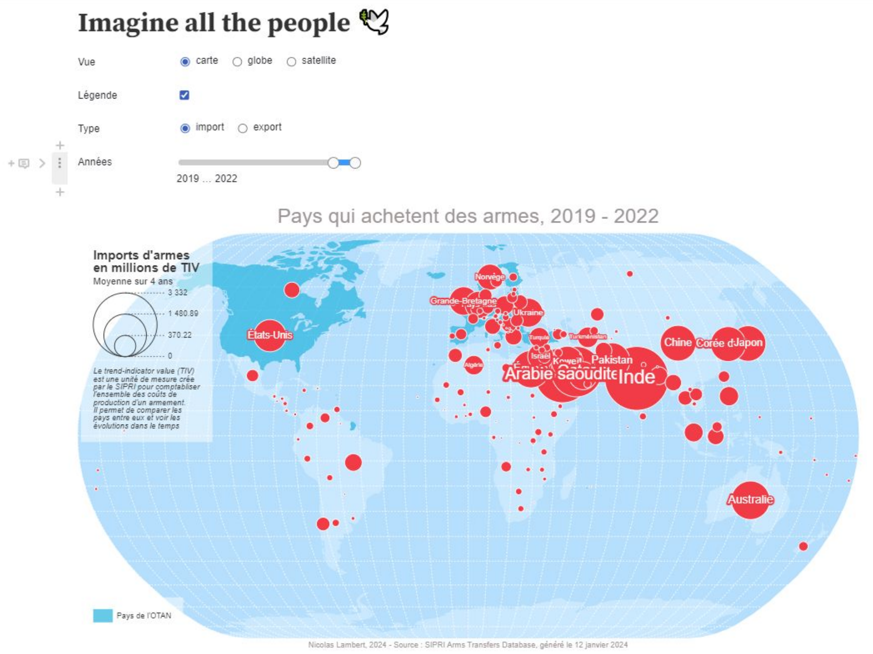


We're not like other **platforms**

Observable isn't just a platform that you can use to create all of your data visualizations — from simple charts to robust data apps. It's a community of data practitioners, a library of examples you can build from, and a place where you can grow.

https://observablehq.com/why-observable

Examples: D3 in Observable



Imagine all the people 🕊

Vue [✓ carte] globe satellite

Légende ✓

Type [✓ import] export

Années 2019 … 2022

Pays qui achetent des armes, 2019 - 2022

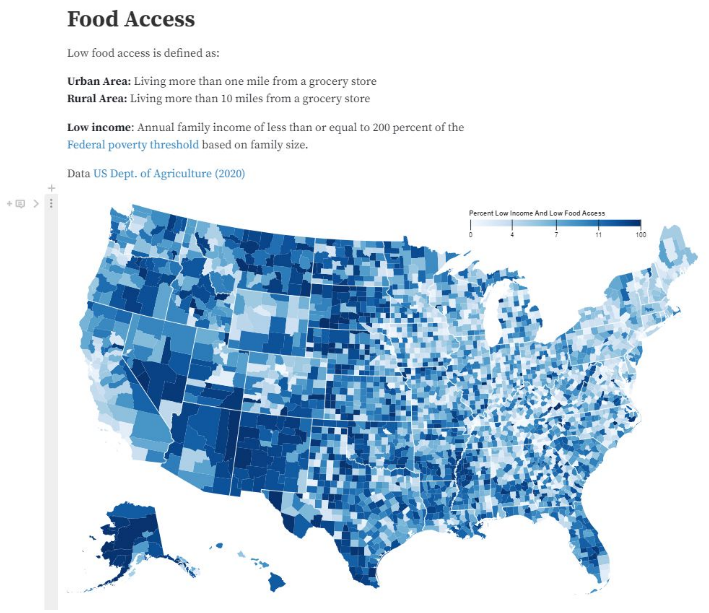
Imports d'armes en millions de TIV

Moyenne sur 4 ans

Le trend-indicator value (TIV) est une unité de mesure créée par le SIPRI pour comptabiliser l'ensemble des coûts de production d'un armement. Il permet de comparer les pays entre eux et voir les évolutions dans le temps

Nicolas Lambert, 2024 - Source : SIPRI Arms Transfers Database, généré le 12 janvier 2024

https://observablehq.com/@neocartocnrs/quelle-connerie-la-guerre



Food Access

Low food access is defined as:

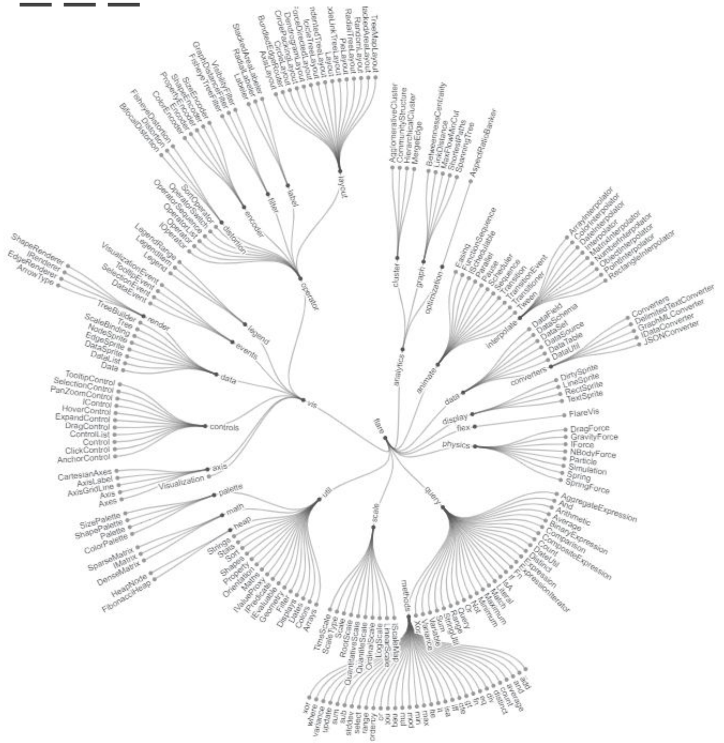
Urban Area: Living more than one mile from a grocery store
Rural Area: Living more than 10 miles from a grocery store

Low income: Annual family income of less than or equal to 200 percent of the Federal poverty threshold based on family size.

Data US Dept. of Agriculture (2020)

https://observablehq.com/@strzepek/food-access

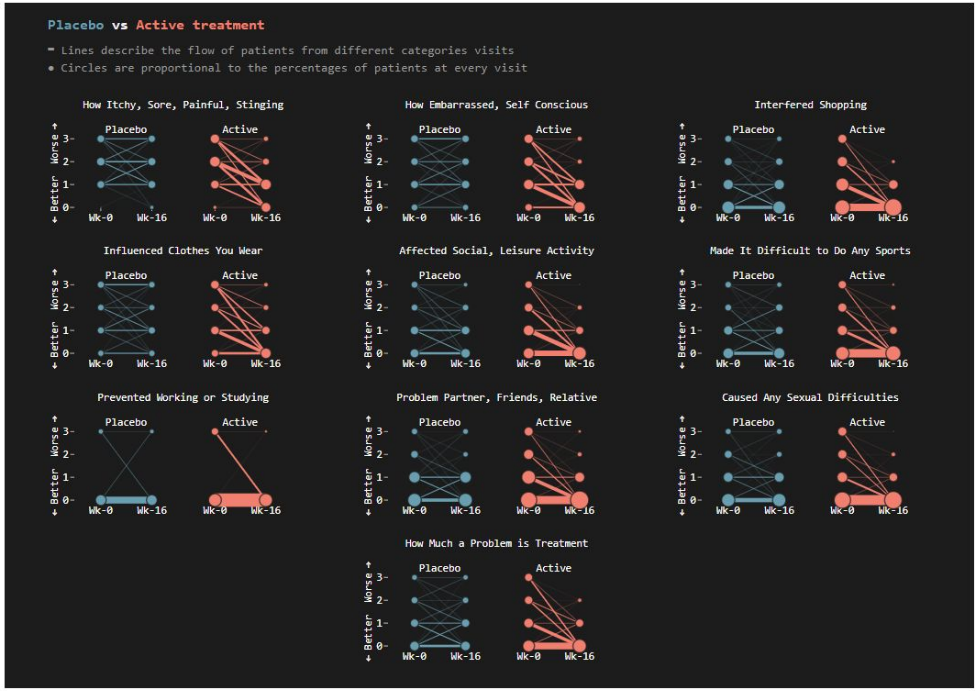
Examples: D3 in Observable



https://observablehq.com/@d3/radial-tree/2?intent=fork

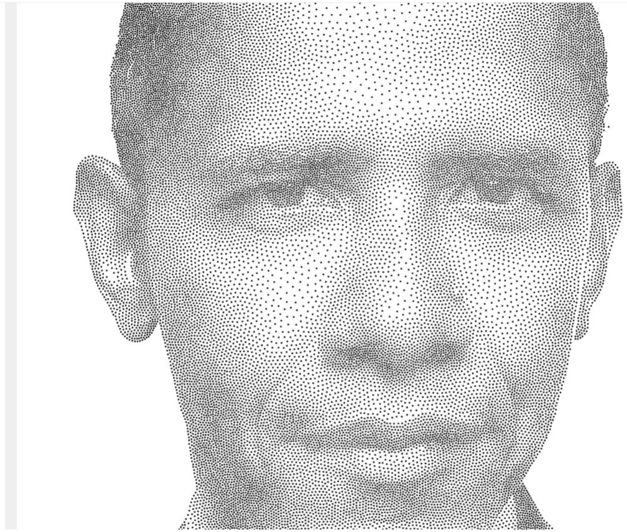
Comparing Placebo vs. Active treatments

Wonderful Wednesday DLQI data set



https://observablehq.com/@irenedelatorre/comparing-placebo-vs-active-treatments

Examples: D3 in Observable



https://observablehq.com/@mbostock/voronoi-stippling

https://observablehq.com/@mbostock/owls-to-the-max

Observable Plot

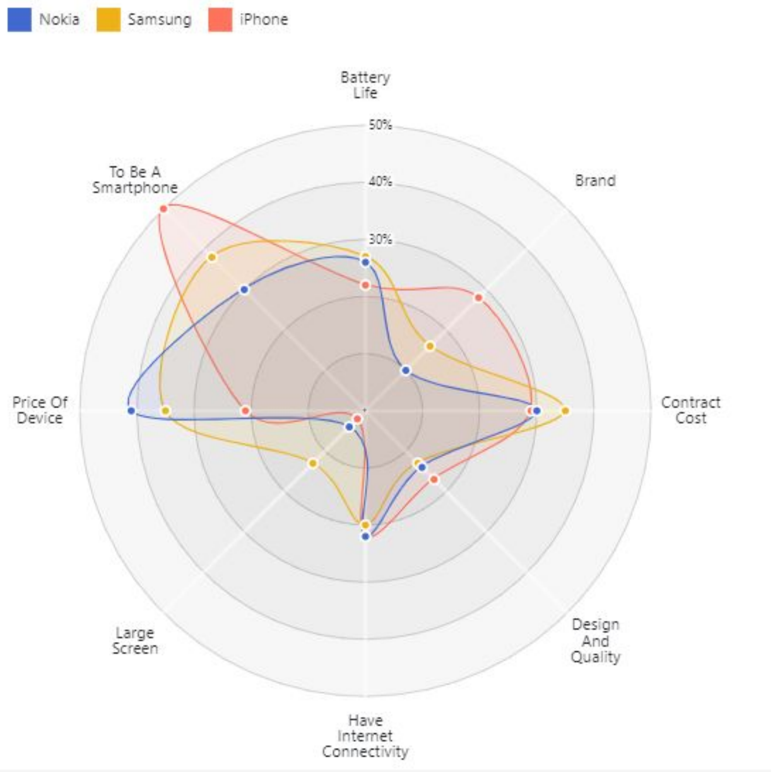
Observable Plot is a free, open-source, Javascript library for data visualization, built by the team that made D3.js.

“Plot’s goal is to make the easy things easy, and fast.”

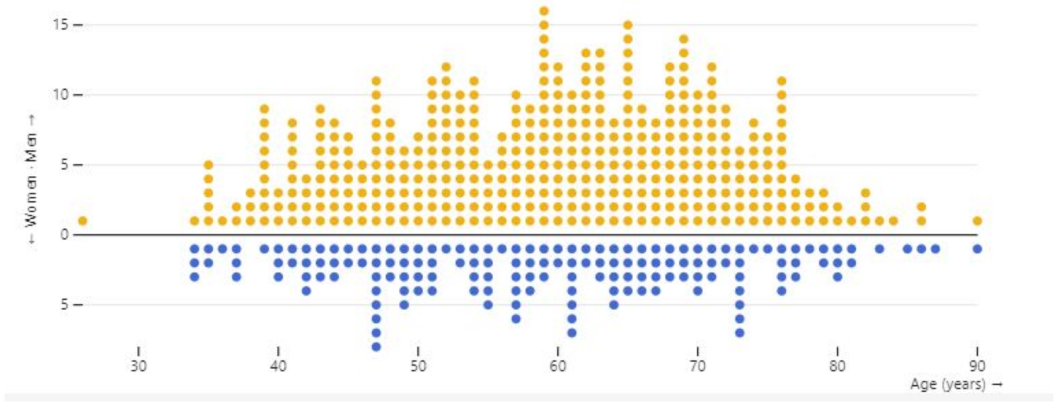
…for complex data visualization D3.js is better than Plot.

https://observablehq.com/plot/why-plot

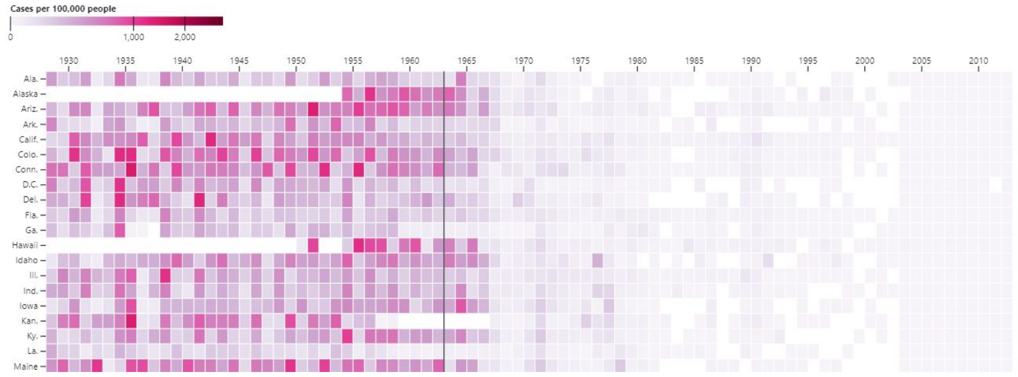
Examples: Plot



https://observablehq.com/@observablehq/plot-radar-chart?intent=fork



https://observablehq.com/@observablehq/plot-stacked-dots?intent=fork



https://observablehq.com/@observablehq/plot-impact-of-vaccines?intent=fork

Let's try it!

Plot

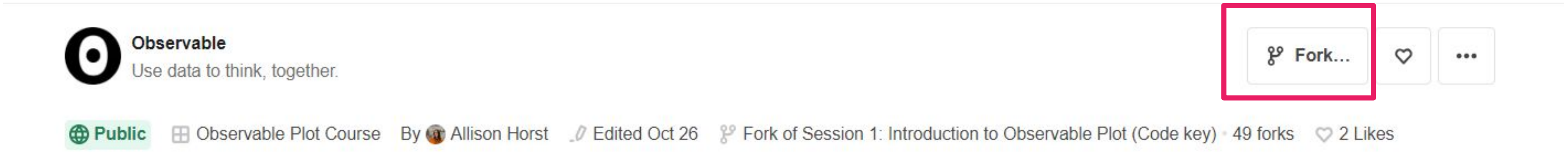
Tutorial based on "Session 1: Introduction to Observable Plot" by Observable team.

https://observablehq.com/@observablehq/plot-session-1-follow-along

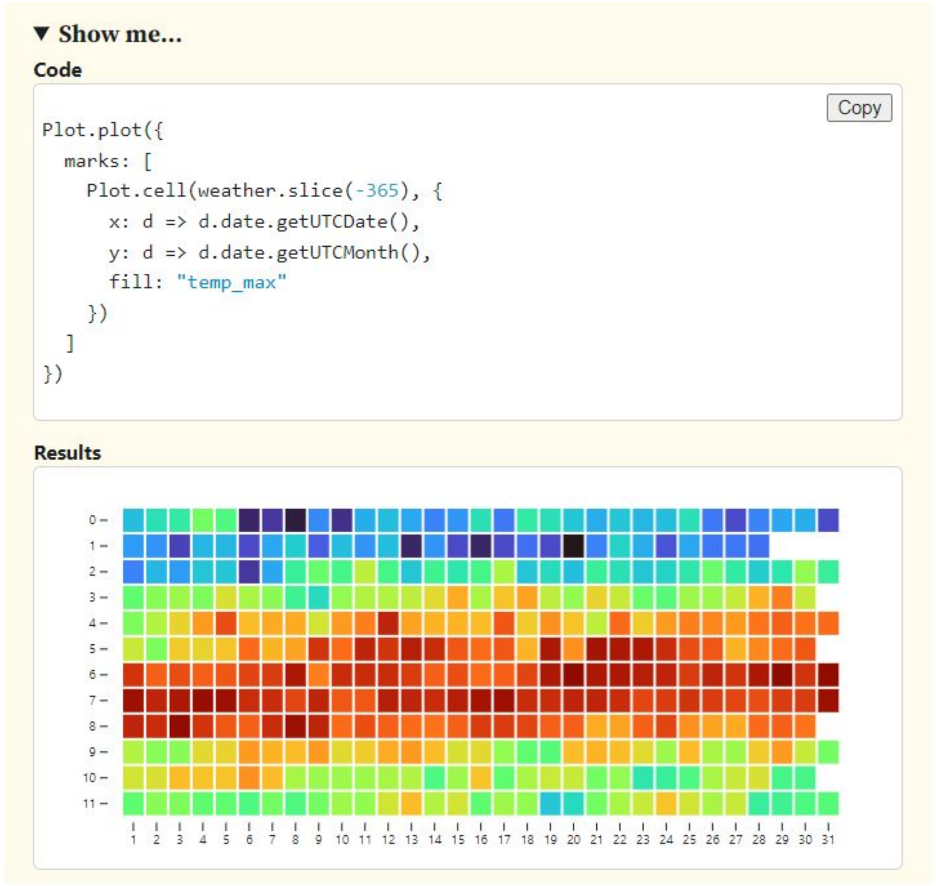
Setting up

Create an observable account: https://observablehq.com/

Go to the tutorial following this link and click at Fork button to create a copy of the tutorial i unin your account.


Observable
Use data to think, together.
Fork...
Public Observable Plot Course By Allison Horst Edited Oct 26 Fork of Session 1: Introduction to Observable Plot (Code key) · 49 forks 2 Likes

Help code



▼ **Show me...**

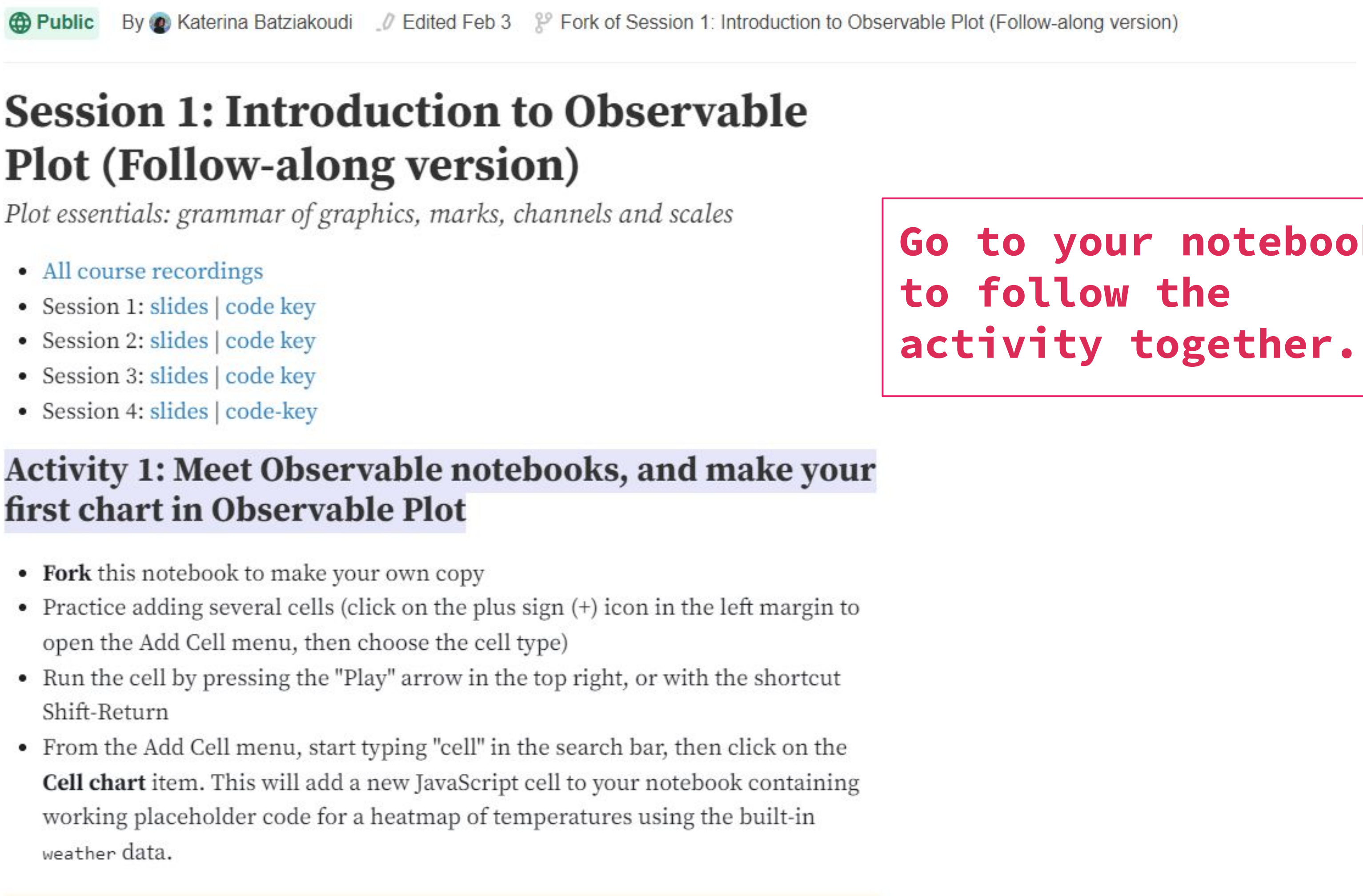
Code

Copy

```
Plot.plot({
  marks: [
    Plot.cell(weather.slice(-365), {
      x: d => d.date.getUTCDate(),
      y: d => d.date.getUTCMonth(),
      fill: "temp_max"
    })
  ]
})
```

Results

1. Meet Observable Notebooks and make our first chart in Observable Plot!



Public By Katerina Batziakoudi Edited Feb 3 Fork of Session 1: Introduction to Observable Plot (Follow-along version)

Session 1: Introduction to Observable Plot (Follow-along version)

Plot essentials: grammar of graphics, marks, channels and scales

- All course recordings
- Session 1: slides | code key
- Session 2: slides | code key
- Session 3: slides | code key
- Session 4: slides | code-key

Activity 1: Meet Observable notebooks, and make your first chart in Observable Plot

- **Fork** this notebook to make your own copy
- Practice adding several cells (click on the plus sign (+) icon in the left margin to open the Add Cell menu, then choose the cell type)
- Run the cell by pressing the "Play" arrow in the top right, or with the shortcut Shift-Return
- From the Add Cell menu, start typing "cell" in the search bar, then click on the **Cell chart** item. This will add a new JavaScript cell to your notebook containing working placeholder code for a heatmap of temperatures using the built-in `weather` data.

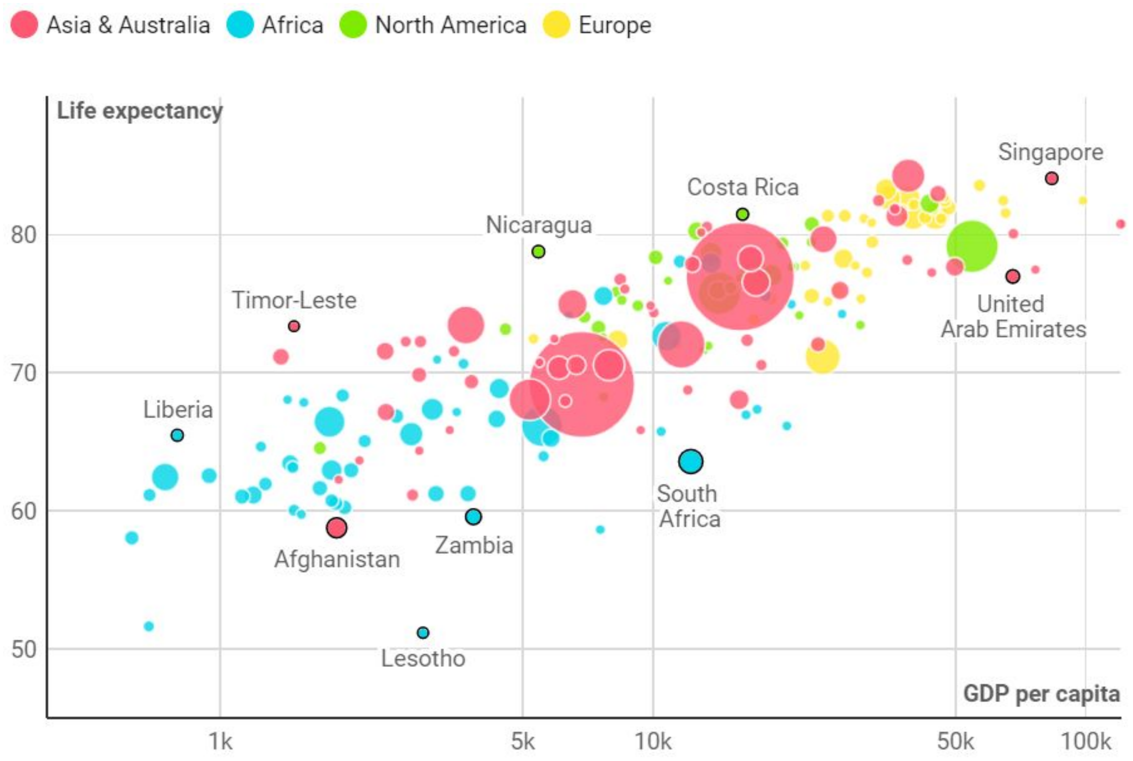
Go to your notebook to follow the activity together.

2. The grammar of graphics in Observable Plot!

Grammar of graphics

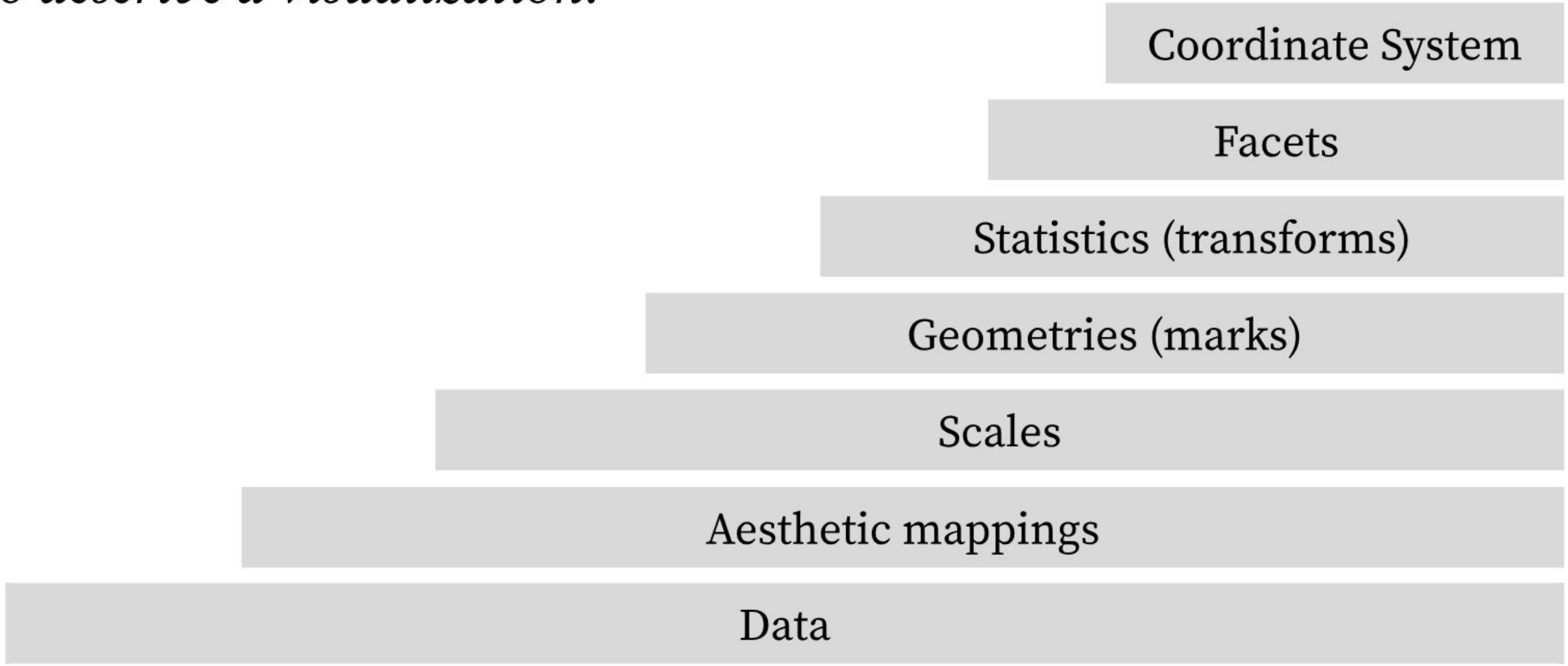
"A **grammar of graphics** is a tool that enables us to concisely describe the components of a graphic. Such a grammar allows us to move beyond named graphics (e.g., the "scatterplot") and gain insight into the deep structure that underlies statistical graphics."

Wickam 2010, *A Layered Grammar of Graphics*



This is a scatterplot.

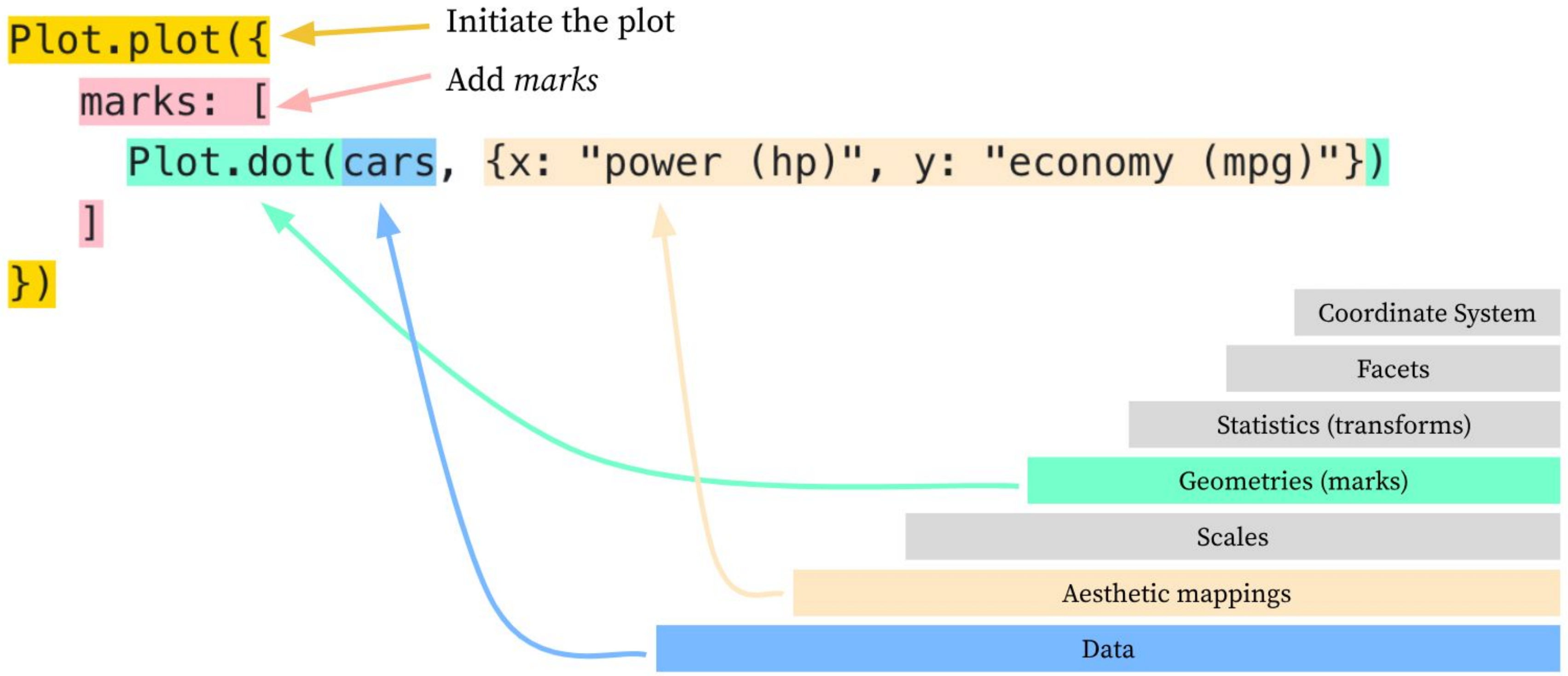
Following the grammar of graphics, we'll be able to efficiently build charts layer-by-layer. *What **layers** do we use to describe a visualization?*



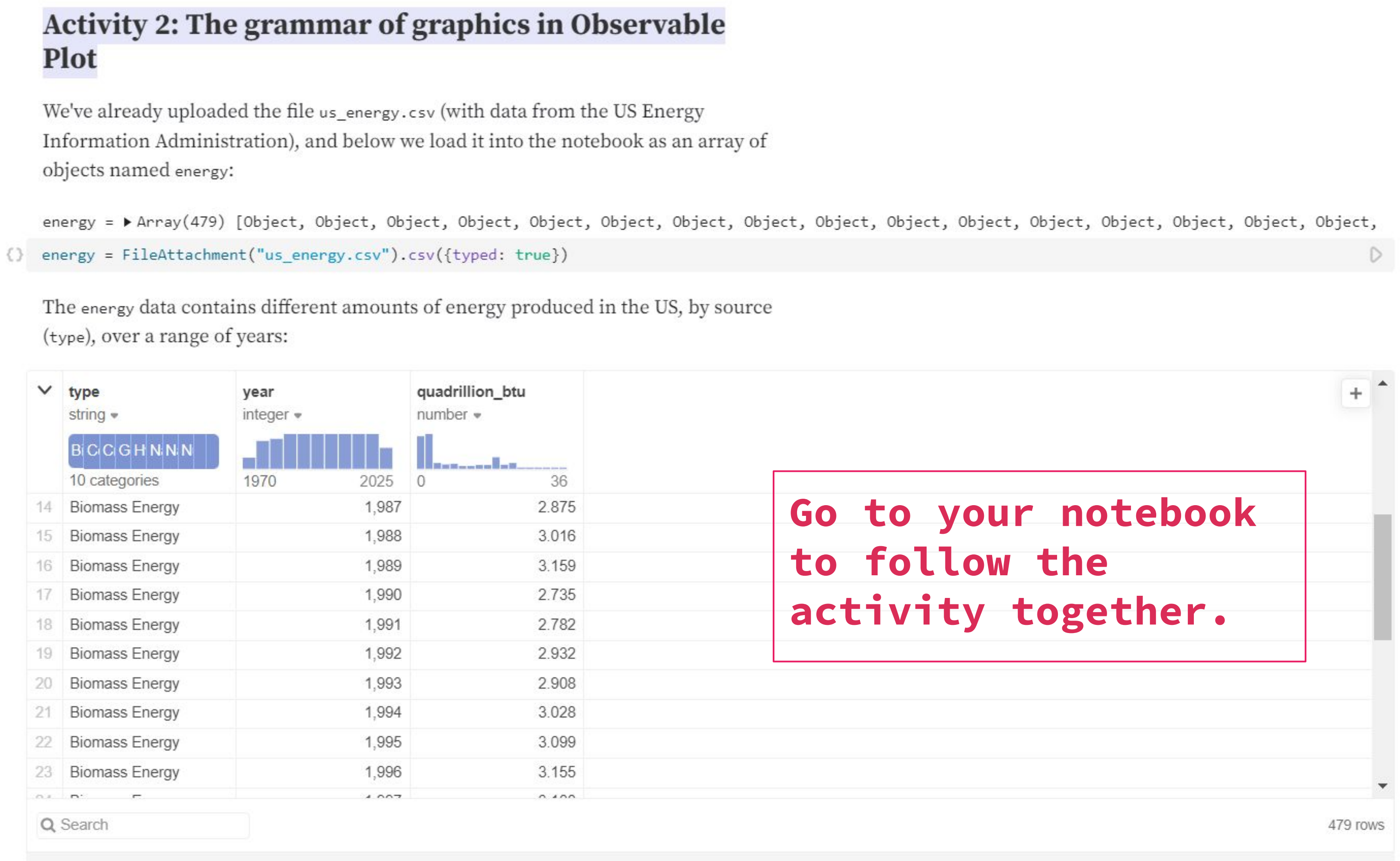
Slide made by Observable team

In Observable Plot

```
Plot.plot({
    marks: [
      Plot.dot(cars, {x: "power (hp)", y: "economy (mpg)"})
    ]
})
```



Slide made by Observable team



Activity 2: The grammar of graphics in Observable Plot

We've already uploaded the file `us_energy.csv` (with data from the US Energy Information Administration), and below we load it into the notebook as an array of objects named `energy`:

```
energy = ▸ Array(479) [Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object,
```

```
energy = FileAttachment("us_energy.csv").csv({typed: true})
```

The `energy` data contains different amounts of energy produced in the US, by source (`type`), over a range of years:

	type	year	quadrillion_btu
	string	integer	number
	10 categories	1970 – 2025	0 – 36
14	Biomass Energy	1,987	2.875
15	Biomass Energy	1,988	3.016
16	Biomass Energy	1,989	3.159
17	Biomass Energy	1,990	2.735
18	Biomass Energy	1,991	2.782
19	Biomass Energy	1,992	2.932
20	Biomass Energy	1,993	2.908
21	Biomass Energy	1,994	3.028
22	Biomass Energy	1,995	3.099
23	Biomass Energy	1,996	3.155

Go to your notebook to follow the activity together.

3. Chart customization with marks, channels and scales

Marks, channels and scales

With our base plot built, we can start customizing! We'll learn a bit more about *marks*, *channels*, and *scales*, and how we can update them to customize our charts.

Marks

From Observable Plot documentation:

“Think of marks as the “visual vocabulary” — the painter’s palette 🎨, but of shapes instead of colors — that you pull from when composing a chart. Each mark type produces a certain type of geometric shape…Mark constructors take two arguments: **data** and **options**. ”

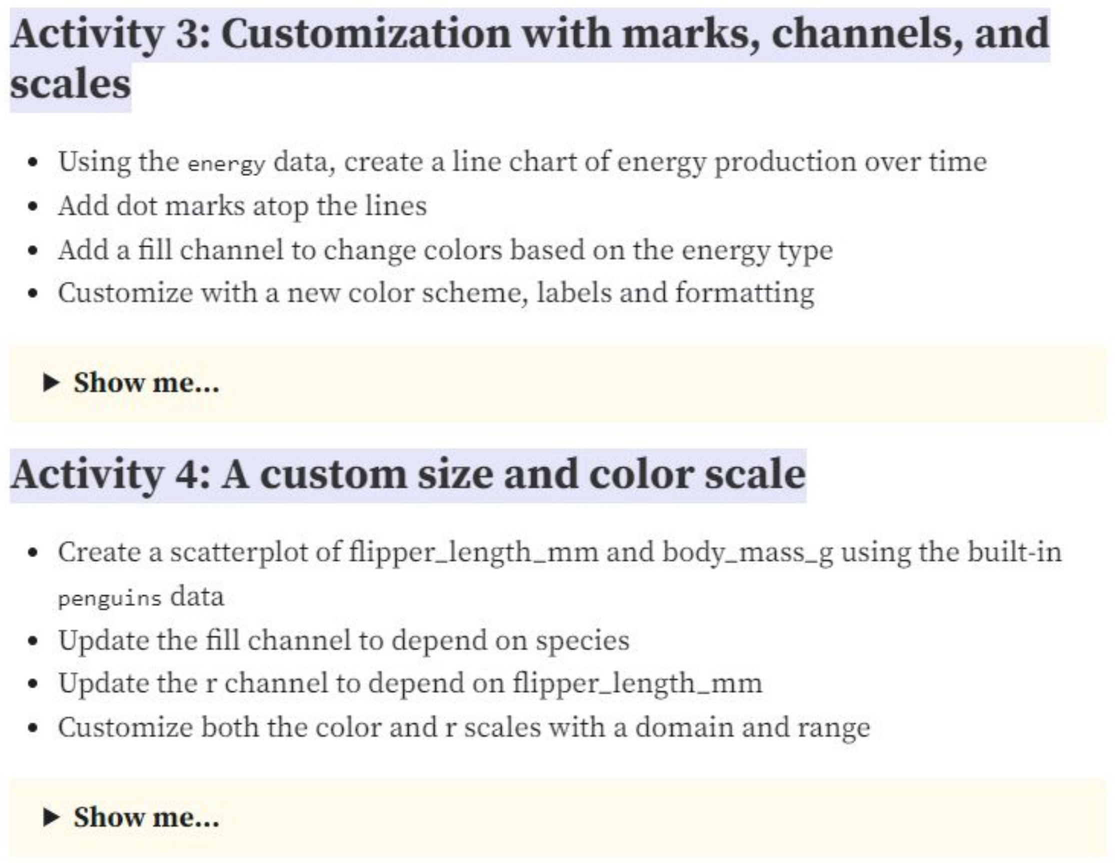
We use **marks** instead of specific **chart types**.

Scales

From Observable Plot documentation:

“**Scales** convert an abstract value such as time or temperature to a visual value such as *x*→ or *y*↑ position or color.”

Scales are defined by their **domain** (input values) and **range** (the visual values that the scale generates as output).



Activity 3: Customization with marks, channels, and scales

- Using the `energy` data, create a line chart of energy production over time
- Add dot marks atop the lines
- Add a fill channel to change colors based on the energy type
- Customize with a new color scheme, labels and formatting

▶ **Show me...**

Activity 4: A custom size and color scale

- Create a scatterplot of flipper_length_mm and body_mass_g using the built-in `penguins` data
- Update the fill channel to depend on species
- Update the r channel to depend on flipper_length_mm
- Customize both the color and r scales with a domain and range

▶ **Show me...**

Go to your notebook to follow the activity together.