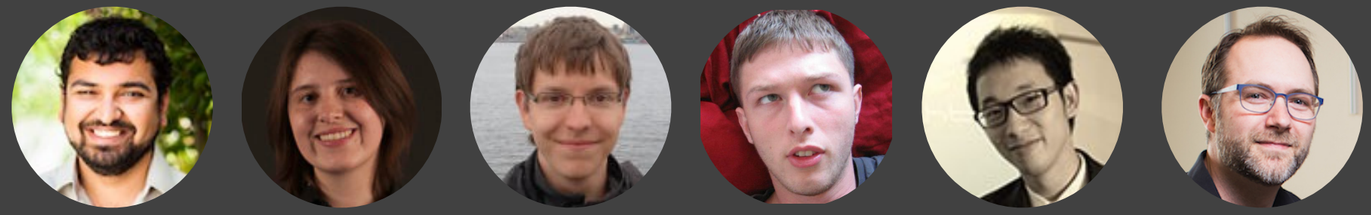
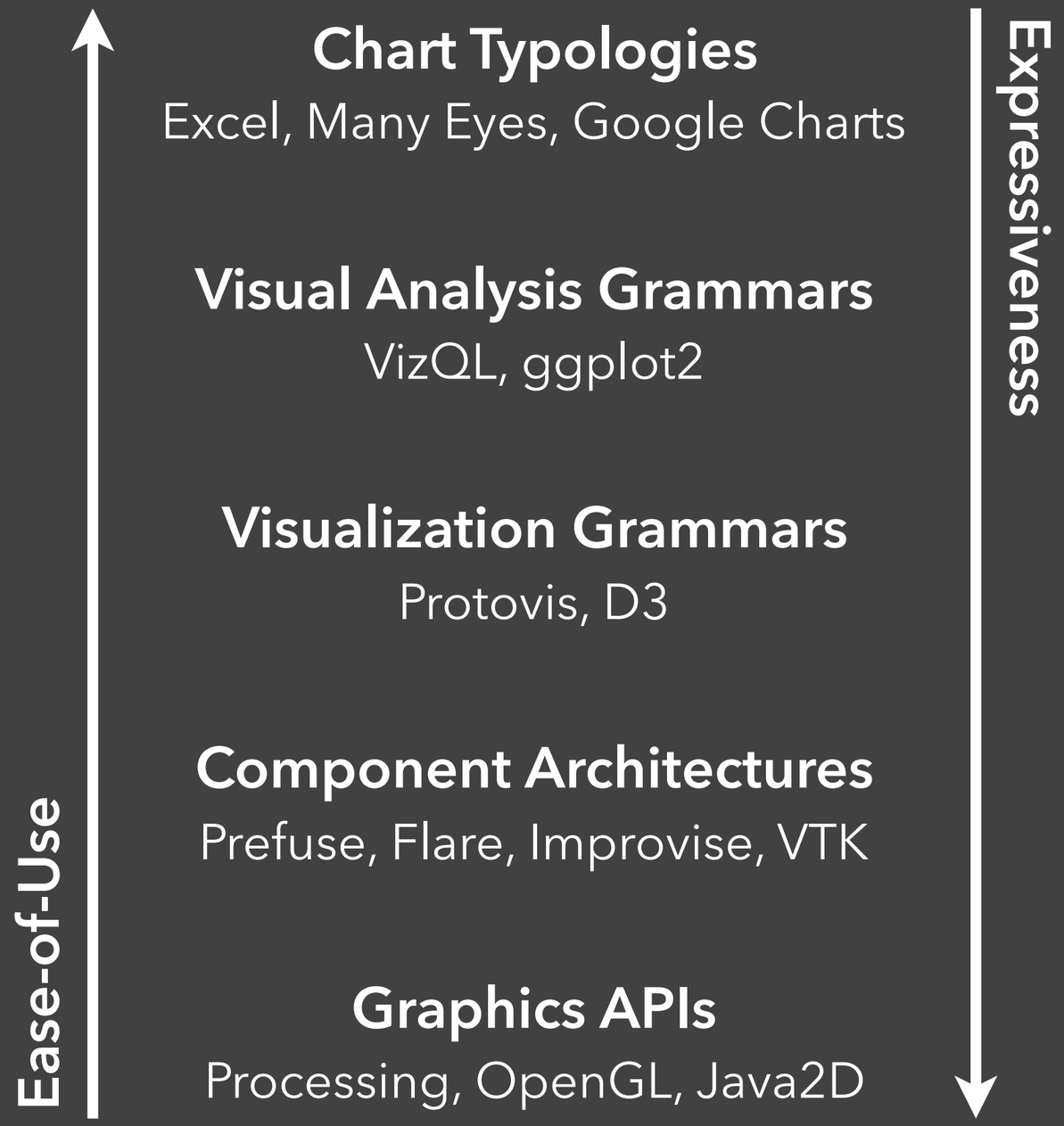
The Vega Ecosystem

Visualization in Practice



Arvind Satyanarayan @arvindsatya1
Stanford University

Jane Hoffswell
Dominik Moritz @domoritz
Ryan Russell
Kanit "Ham" Wongsuphasawat @kanitw
Jeffrey Heer @jeffrey_heer
University of Washington


Ease-of-Use
Chart Typologies
Excel, Many Eyes, Google Charts
Visual Analysis Grammars
VizQL, ggplot2
Visualization Grammars
Protovis, D3
Component Architectures
Prefuse, Flare, Improvise, VTK
Graphics APIs
Processing, OpenGL, Java2D
Expressiveness

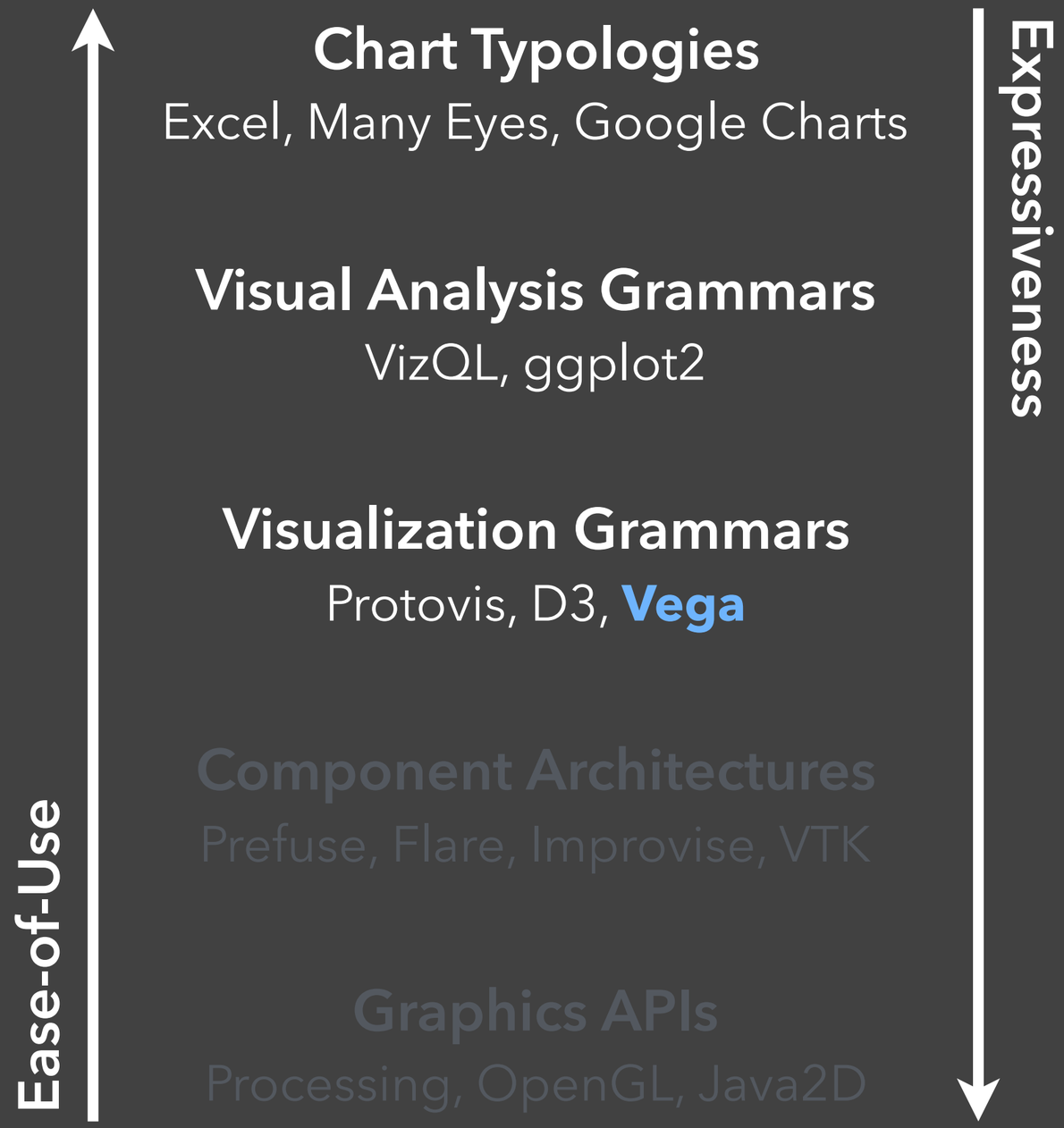

Ease-of-Use
Chart Typologies
Excel, Many Eyes, Google Charts
Visual Analysis Grammars
VizQL, ggplot2
Visualization Grammars
Protovis, D3, Vega
Component Architectures
Prefuse, Flare, Improvise, VTK
Graphics APIs
Processing, OpenGL, Java2D
Expressiveness

Chart Typologies
Excel, Many Eyes, Google Charts

Visual Analysis Grammars
VizQL, ggplot2

Visualization Grammars
Protovis, D3, **Vega**

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D


```
{
  "width": 400, "height": 200,
  "data": [
    {"name": "table", "url": "/data/sample.json"}
  ],
  "scales": [
    {
      "name": "xs","type": "ordinal",
      "range": "width",
      "domain": {"data": "table", "field": "u"}
    }, {
      "name": "ys","type": "ordinal",
      "range": "height",
      "domain": {"data": "table", "field": "v"}
    }
  ],
  "axes": [
    {"type": "x", "scale": "xs"},
    {"type": "y", "scale": "ys"}
  ],
  "marks": [{
    "type": "rect",
    "from": {"data": "table"},
    "properties": {
      "enter": {
        "x": {"scale": "xs", "field": "u"},
        "width": {"scale": "xs", "band": true},
        "y": {"scale": "ys", "field": "v"},
        "y2": {"scale": "ys", "value": 0},
        "fill": {"value": “steelblue"}
      }
    }
  ]
}
```

Ease-of-Use

Chart Typologies
Excel, Many Eyes, Google Charts

Visual Analysis Grammars
VizQL, ggplot2

Visualization Grammars
Protovis, D3, **Vega**

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D

Expressiveness

```
{
  "width": 400, "height": 200,
  "data": [
    {"name": "table", "url": "/data/sample.json"}
  ],
  "scales": [
    {
      "name": "xs","type": "ordinal",
      "range": "width",
      "domain": {"data": "table", "field": "u"}
    }, {
      "name": "ys","type": "ordinal",
      "range": "height",
      "domain": {"data": "table", "field": "v"}
    }
  ],
  "axes": [
    {"type": "x", "scale": "xs"},
    {"type": "y", "scale": "ys"}
  ],
  "marks": [{
    "type": "rect",
    "from": {"data": "table"},
    "properties": {
      "enter": {
        "x": {"scale": "xs", "field": "u"},
        "width": {"scale": "xs", "band": true},
        "y": {"scale": "ys", "field": "v"},
        "y2": {"scale": "ys", "value": 0},
        "fill": {"value": “steelblue"}
      }
    }
  ]
}
```

Chart Typologies
Excel, Many Eyes, Google Charts

Visual Analysis Grammars
VizQL, ggplot2

Visualization Grammars
Protovis, D3, **Vega**

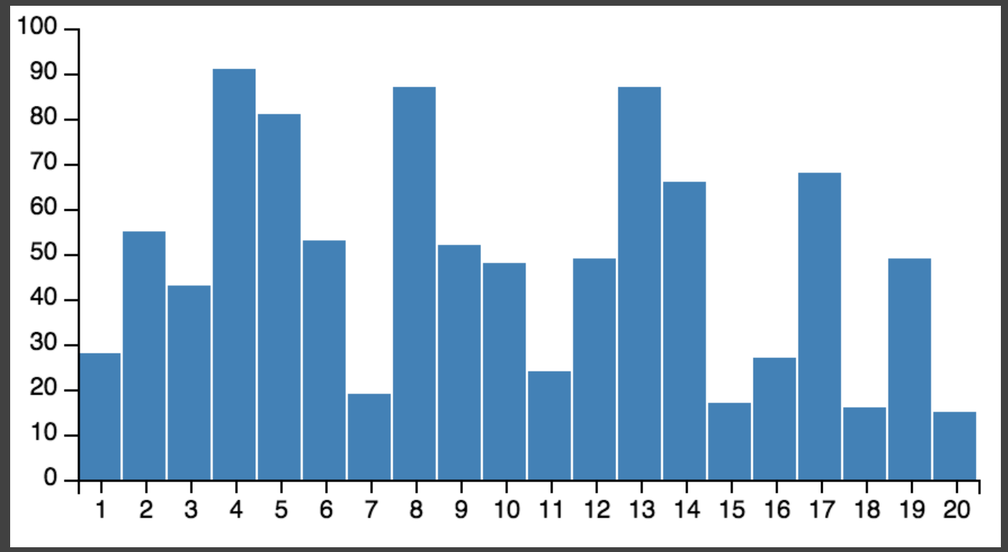
Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D

Ease-of-Use

Expressiveness

```
    }
  ],
  "axes": [
    {"type": "x", "scale": "xs"},
    {"type": "y", "scale": "ys"}
  ],
  "marks": [{
    "type": "rect",
    "from": {"data": "table"},
    "properties": {
      "enter": {
        "x": {"scale": "xs", "field": "u"},
        "width": {"scale": "xs", "band": true},
        "y": {"scale": "ys", "field": "v"},
        "y2": {"scale": "ys", "value": 0},
        "fill": {"value": “steelblue"}
      }
    }
  ]
}
```



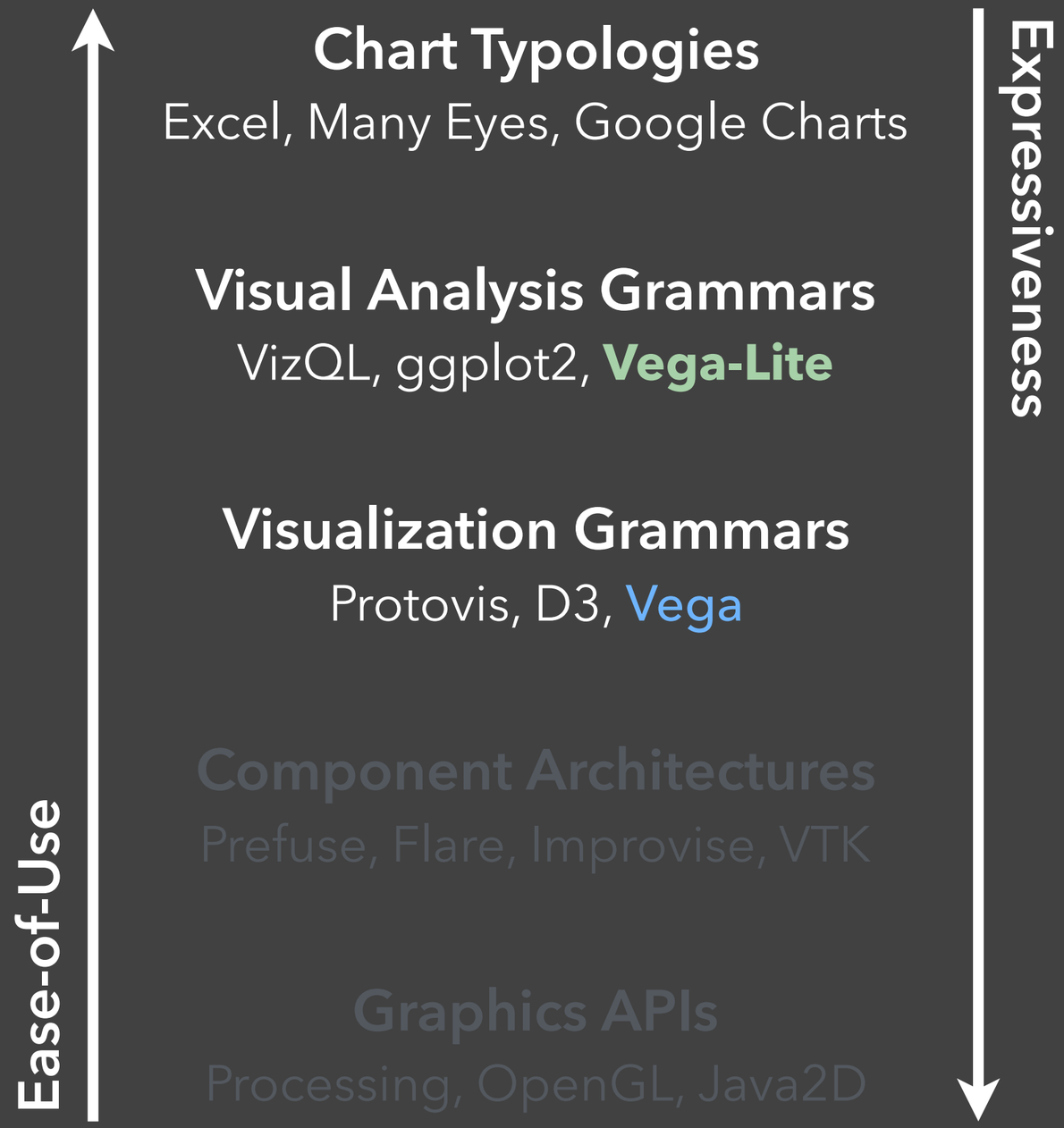

Ease-of-Use
Chart Typologies
Excel, Many Eyes, Google Charts
Visual Analysis Grammars
VizQL, ggplot2, Vega-Lite
Visualization Grammars
Protovis, D3, Vega
Component Architectures
Prefuse, Flare, Improvise, VTK
Graphics APIs
Processing, OpenGL, Java2D
Expressiveness

Chart Typologies
Excel, Many Eyes, Google Charts

Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

Visualization Grammars
Protovis, D3, Vega

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D


```
{
  "data": {"url": "data/cars.json"},
  "mark": "circle",
  "encoding": {
    "x": {
      "field": "Horsepower",
      "type": "quantitative"
    },
    "y": {
      "field": "MPG",
      "type": "quantitative"
    }
  }
}
```

Chart Typologies
Excel, Many Eyes, Google Charts

Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

Visualization Grammars
Protovis, D3, Vega

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D


```
{
  "data": {"url": "data/cars.json"},
  "mark": "circle",
  "encoding": {
    "x": {
      "field": "Horsepower",
      "type": "quantitative"
    },
    "y": {
      "field": "MPG",
      "type": "quantitative"
    }
  }
}
```

Ease-of-Use

Chart Typologies
Excel, Many Eyes, Google Charts

Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

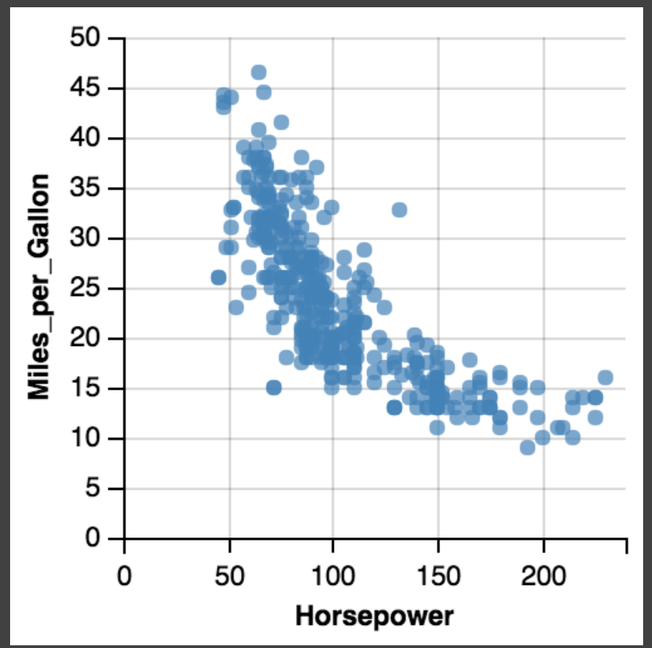
Visualization Grammars
Protovis, D3, Vega

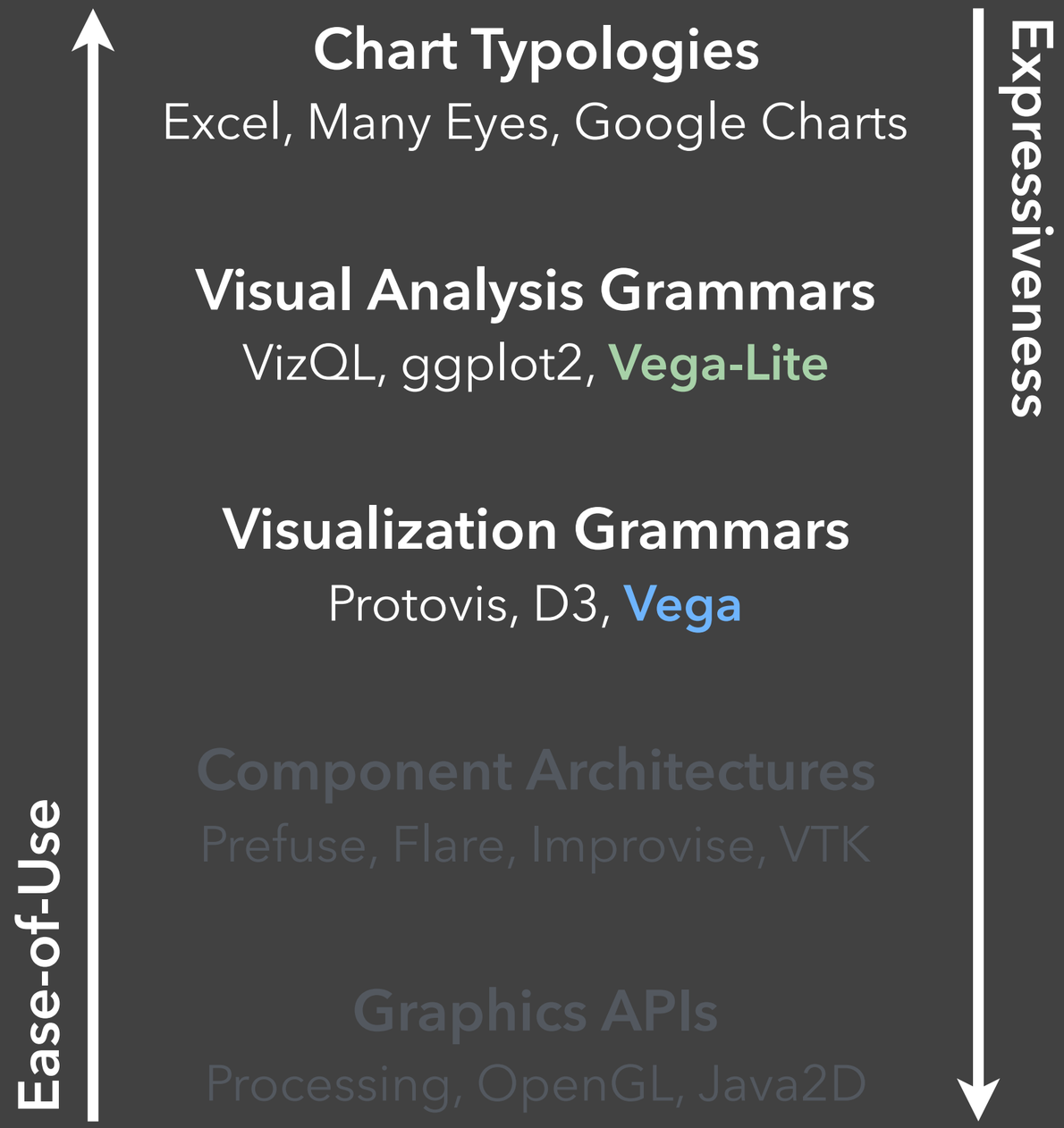
Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D

Expressiveness

```
{
  "data": {"url": "data/cars.json"},
  "mark": "circle",
  "encoding": {
    "x": {
      "field": "Horsepower",
      "type": "quantitative"
    },
    "y": {
      "field": "MPG",
      "type": "quantitative"
    }
  }
}
```





Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.

Ease-of-Use

Expressiveness

Chart Typologies
Excel, Many Eyes, Google Charts

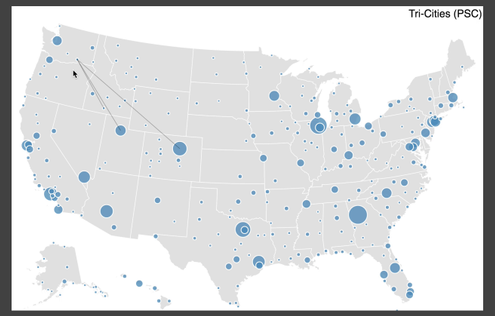
Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

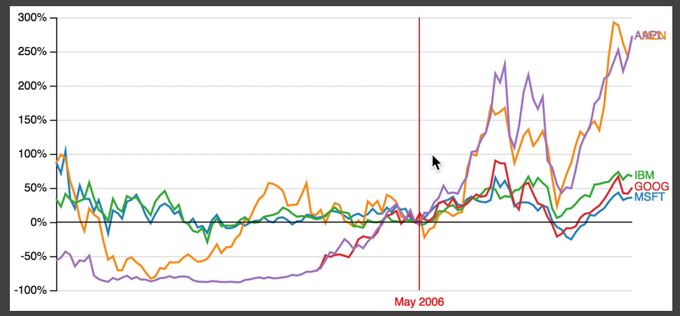
Visualization Grammars
Protovis, D3, **Vega**

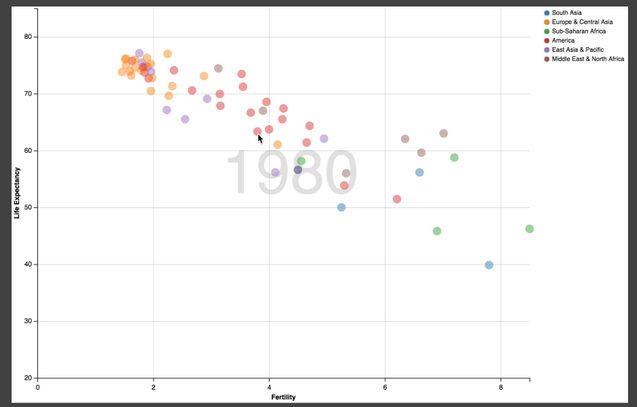
Component Architectures
Prefuse, Flare, Improvise, VTK

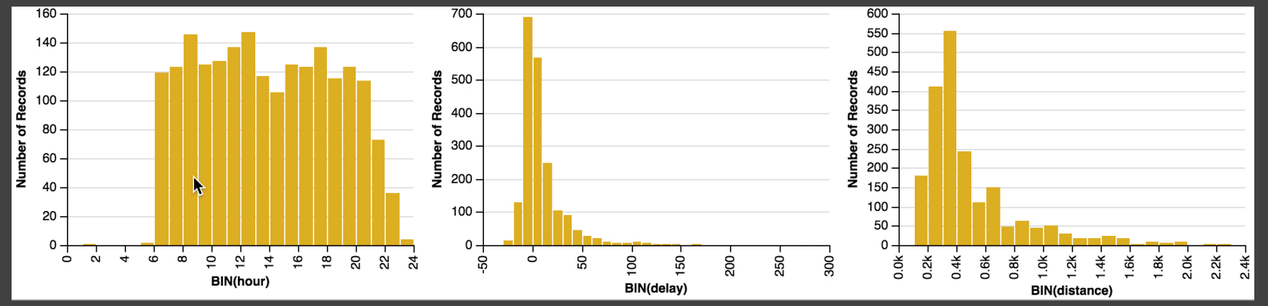
Graphics APIs
Processing, OpenGL, Java2D

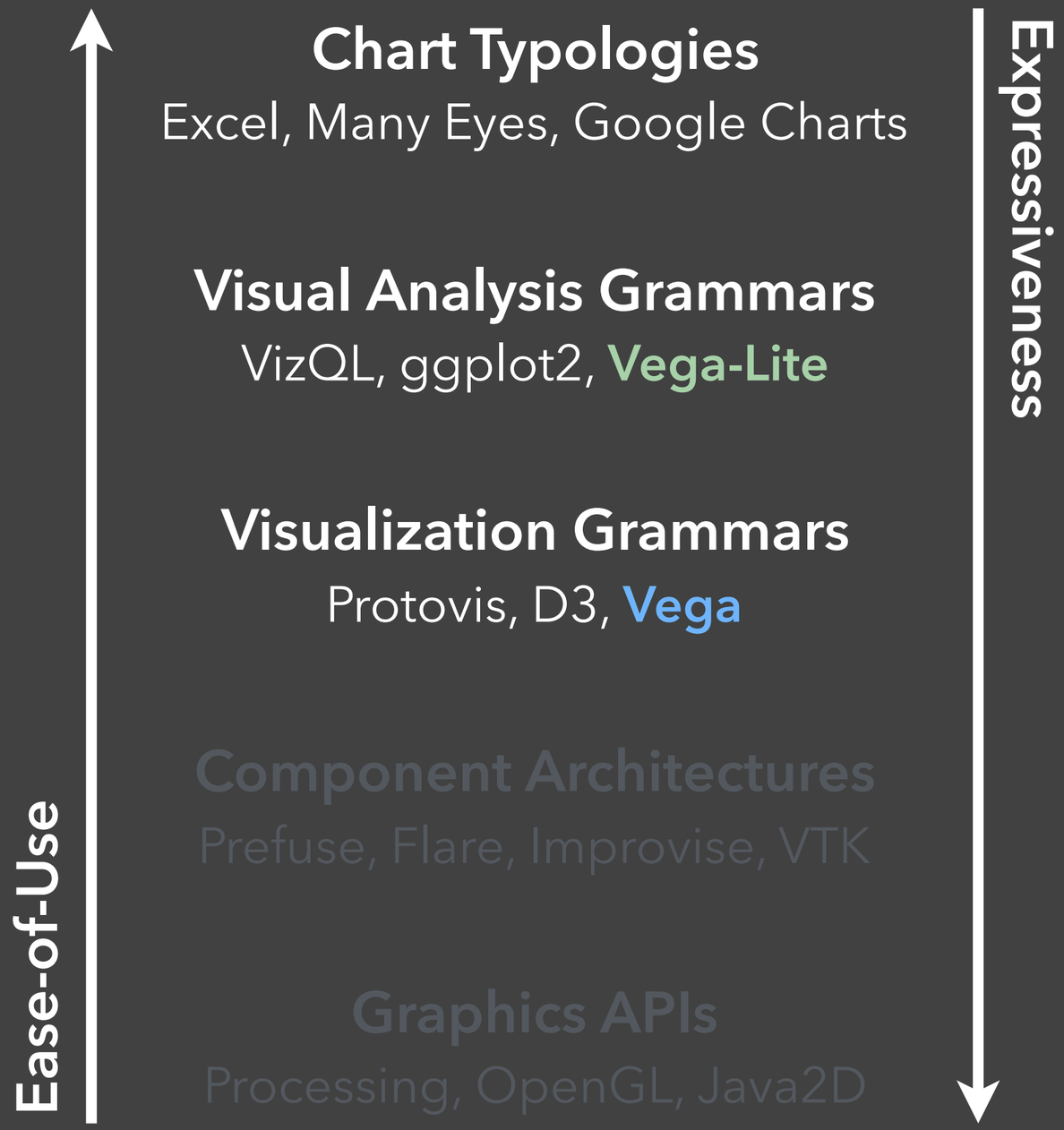
Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.



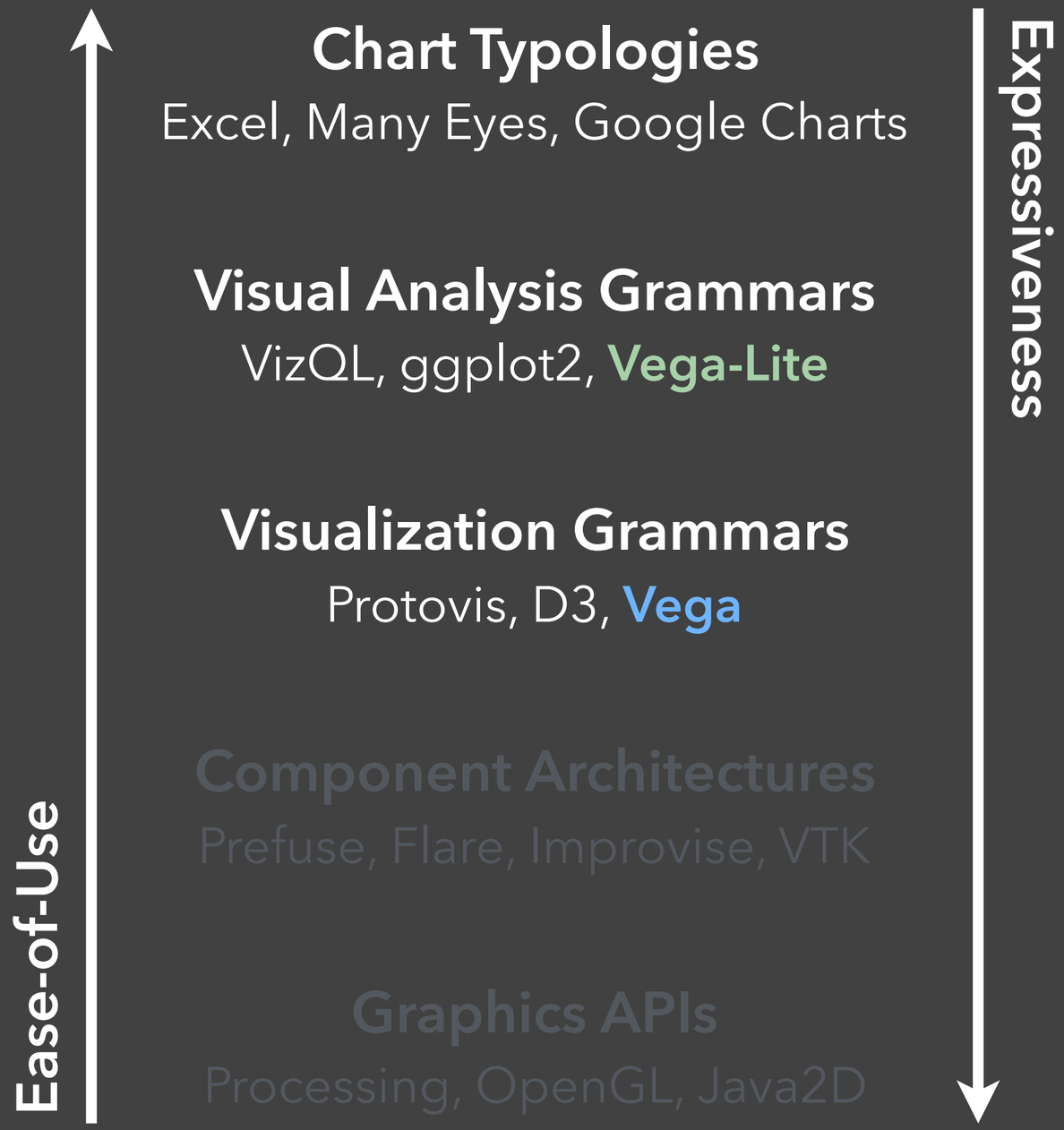








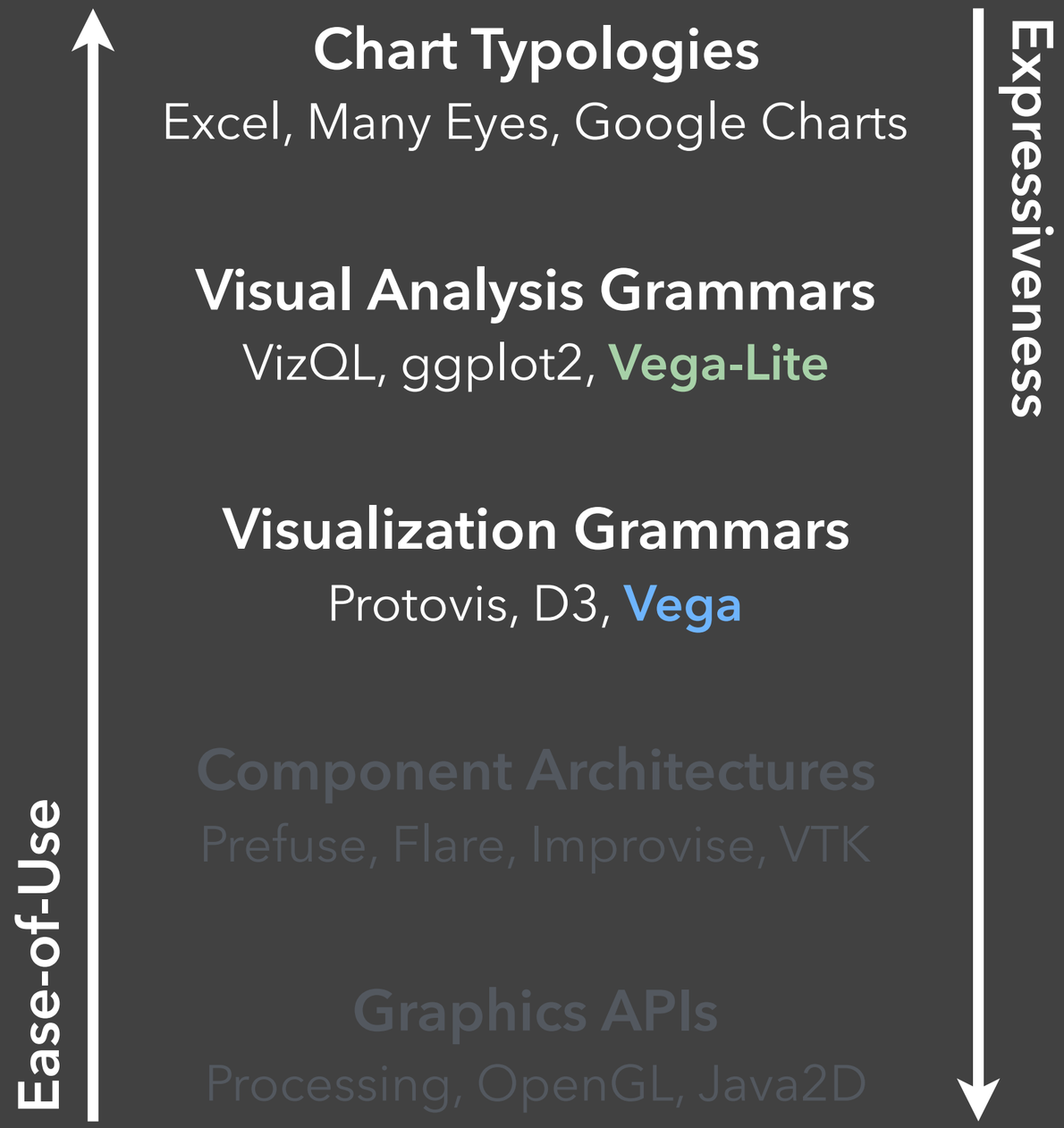
Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.



Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.

✓ Less code + faster iteration.

Accessible to a larger audience.



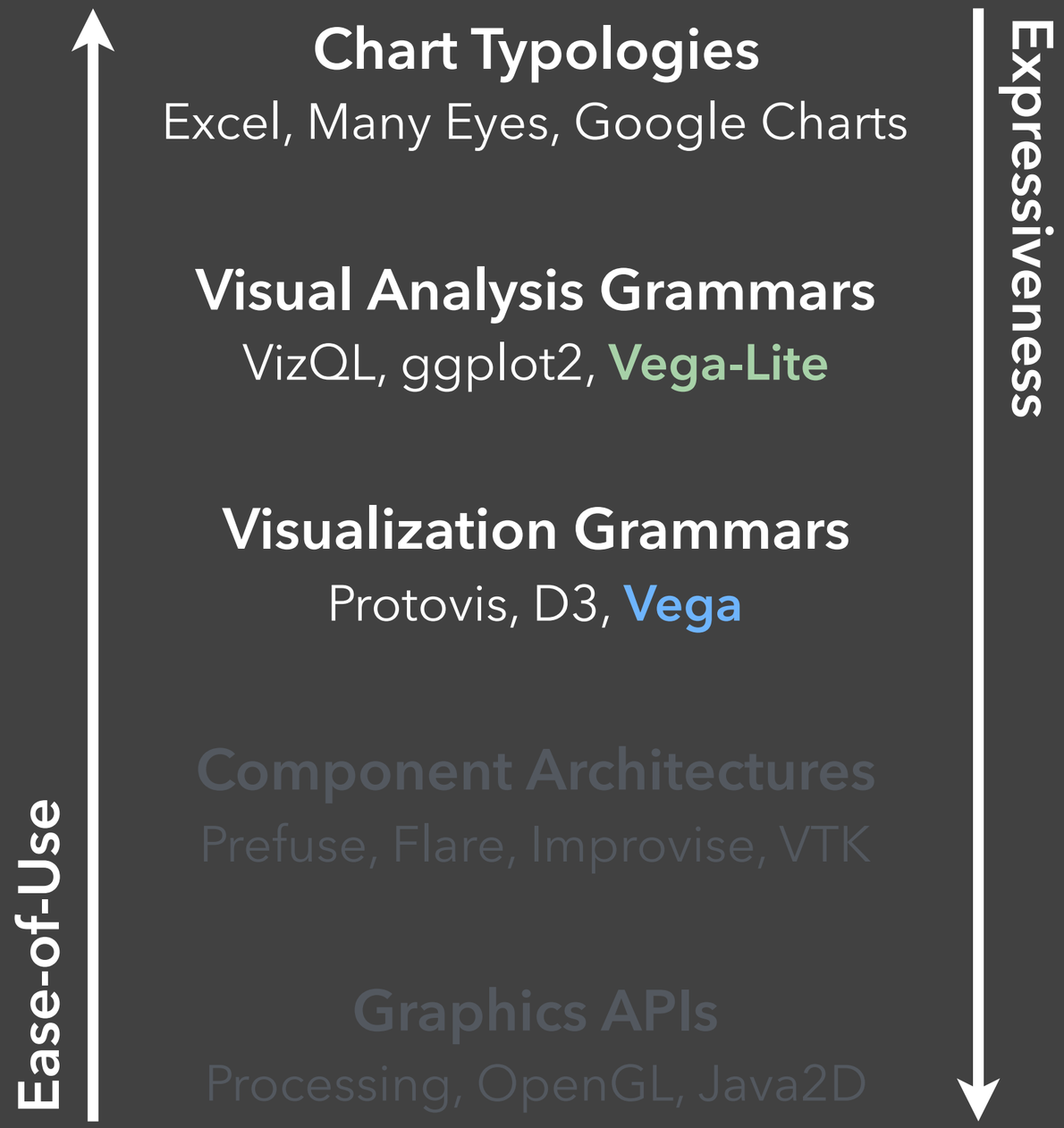
Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.

✓ Less code + faster iteration.

Accessible to a larger audience.

✓ Performance + scalability.

Vega is at least 2x faster than D3.



Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.

✓ Less code + faster iteration.

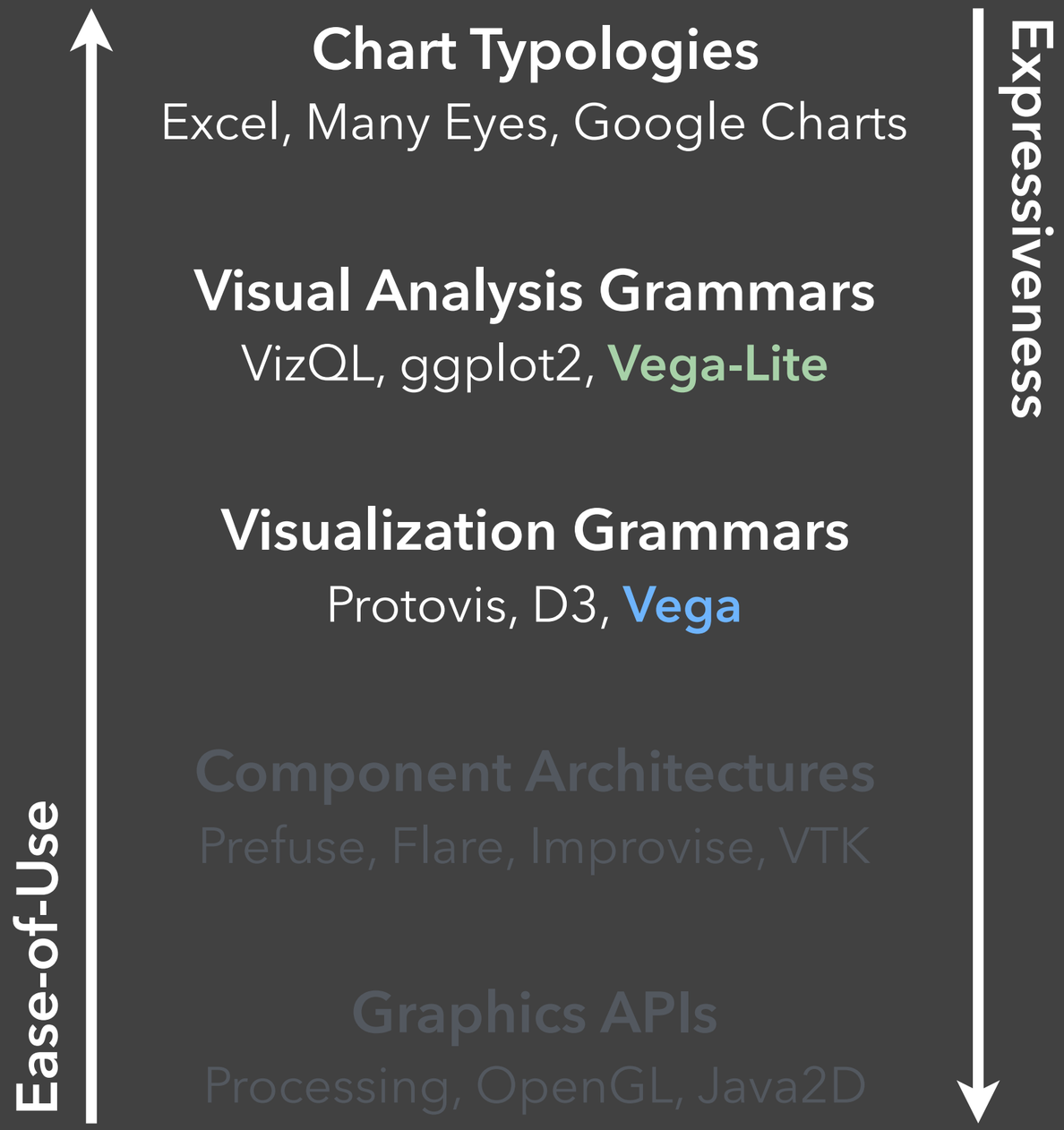
Accessible to a larger audience.

✓ Performance + scalability.

Vega is at least 2x faster than D3.

✓ Reuse + portability.

Write once. Re-apply with different input data. Re-target to multiple devices, renderers, or modalities.



Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.

✓ Less code + faster iteration.

Accessible to a larger audience.

✓ Performance + scalability.

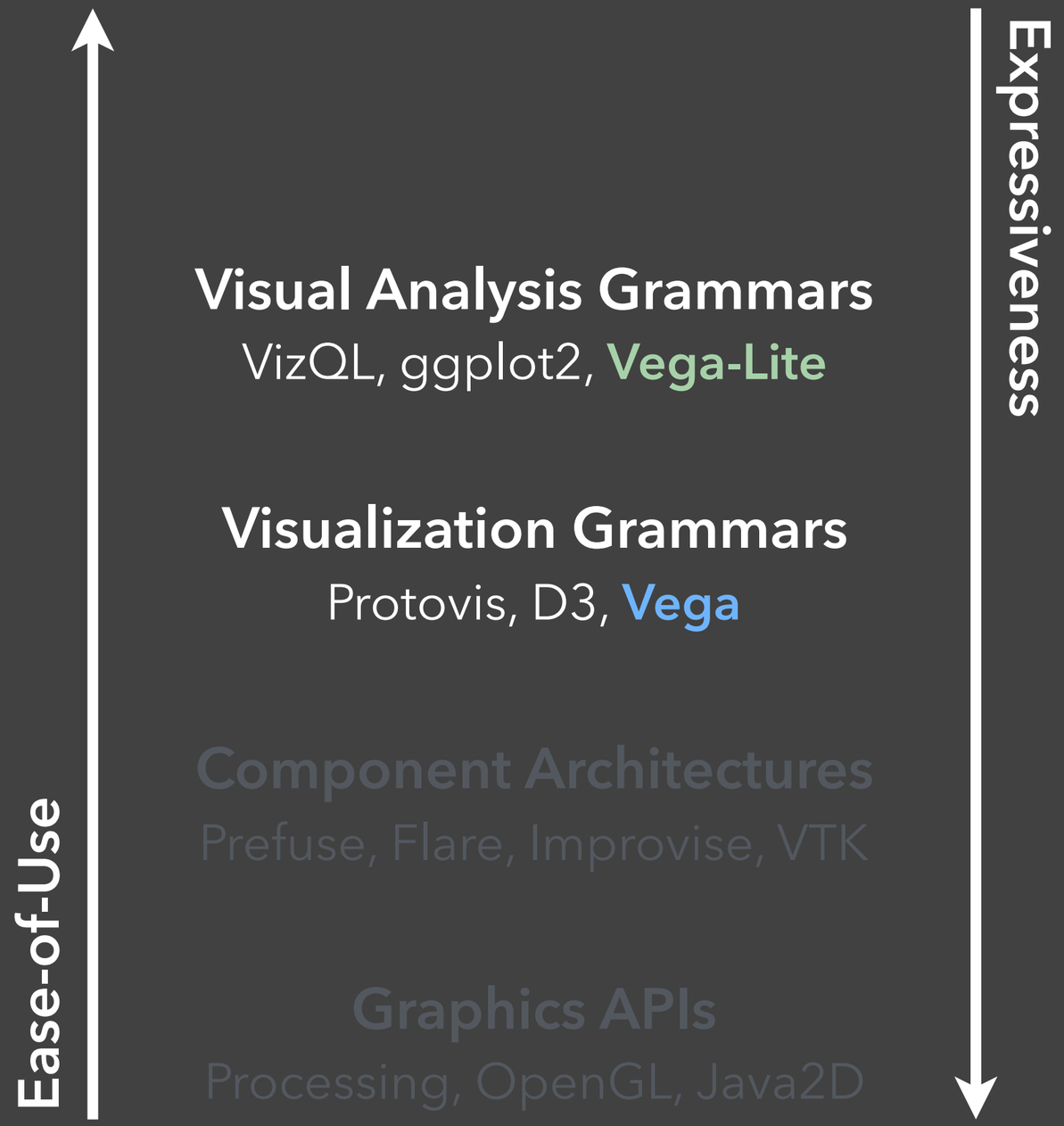
Vega is at least 2x faster than D3.

✓ Reuse + portability.

Write once. Re-apply with different input data. Re-target to multiple devices, renderers, or modalities.

✓ **Programmatic Generation.**

Higher-level software for creating and recommending visualizations.



Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.

✓ Less code + faster iteration.

Accessible to a larger audience.

✓ Performance + scalability.

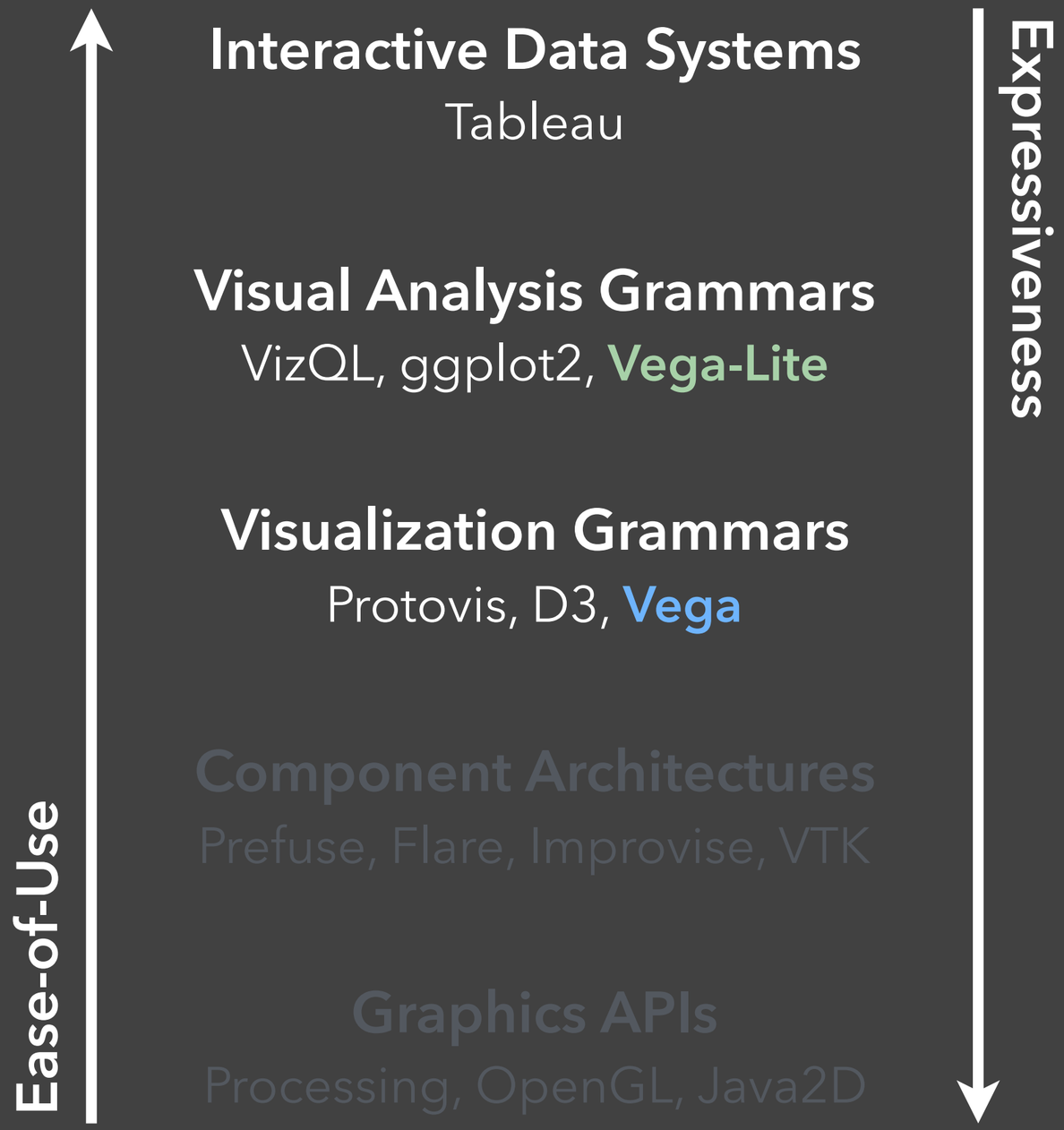
Vega is at least 2x faster than D3.

✓ Reuse + portability.

Write once. Re-apply with different input data. Re-target to multiple devices, renderers, or modalities.

✓ **Programmatic Generation.**

Higher-level software for creating and recommending visualizations.



Declarative specification: describes *what* the visualization should look like vs. *how* it should be computed.

✓ Less code + faster iteration.

Accessible to a larger audience.

✓ Performance + scalability.

Vega is at least 2x faster than D3.

✓ Reuse + portability.

Write once. Re-apply with different input data. Re-target to multiple devices, renderers, or modalities.

✓ **Programmatic Generation.**

Higher-level software for creating and recommending visualizations.

Ease-of-Use

Expressiveness

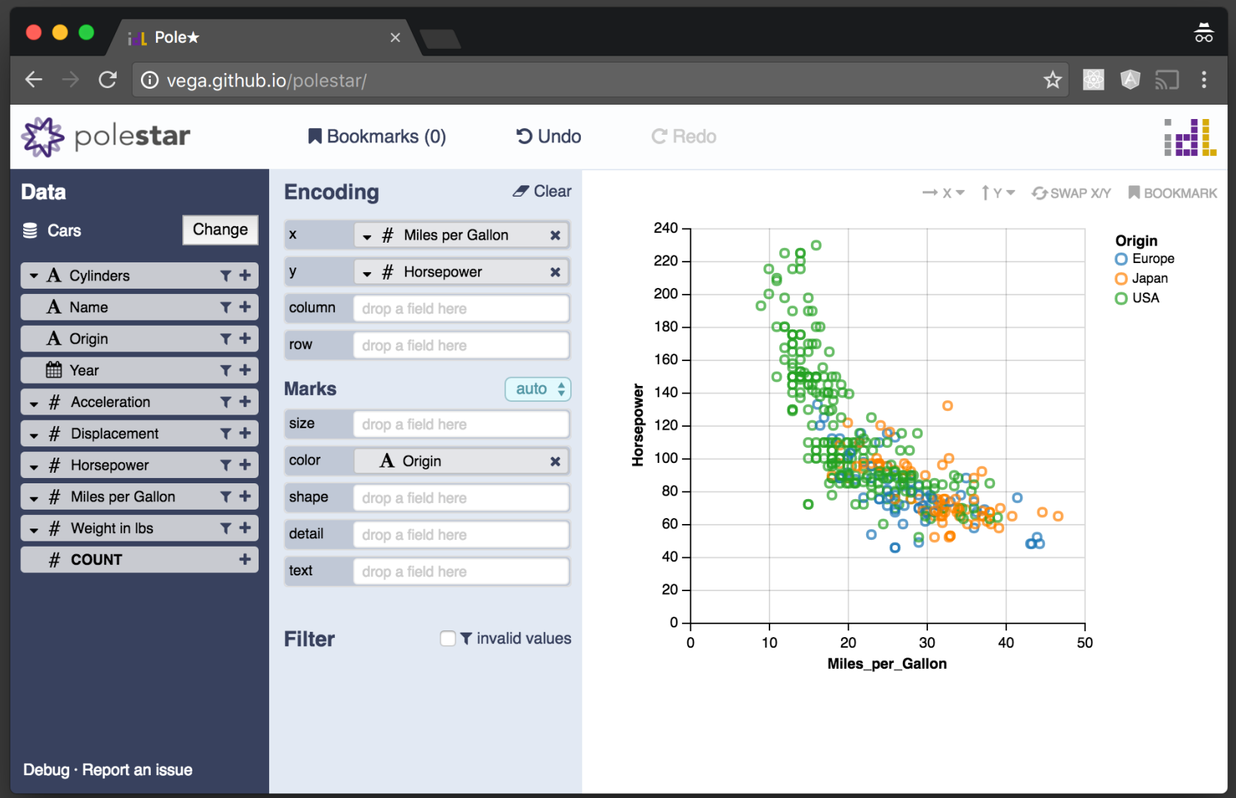
Interactive Data Systems
Tableau, PoleStar

Visual Analysis Grammars
VizQL, ggplot2, Vega-Lite

Visualization Grammars
Protovis, D3, Vega

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D



Ease-of-Use

Expressiveness

Interactive Data Systems

Tableau, PoleStar, Voyager

Visual Analysis Grammars

VizQL, ggplot2, **Vega-Lite**

Visualization Grammars

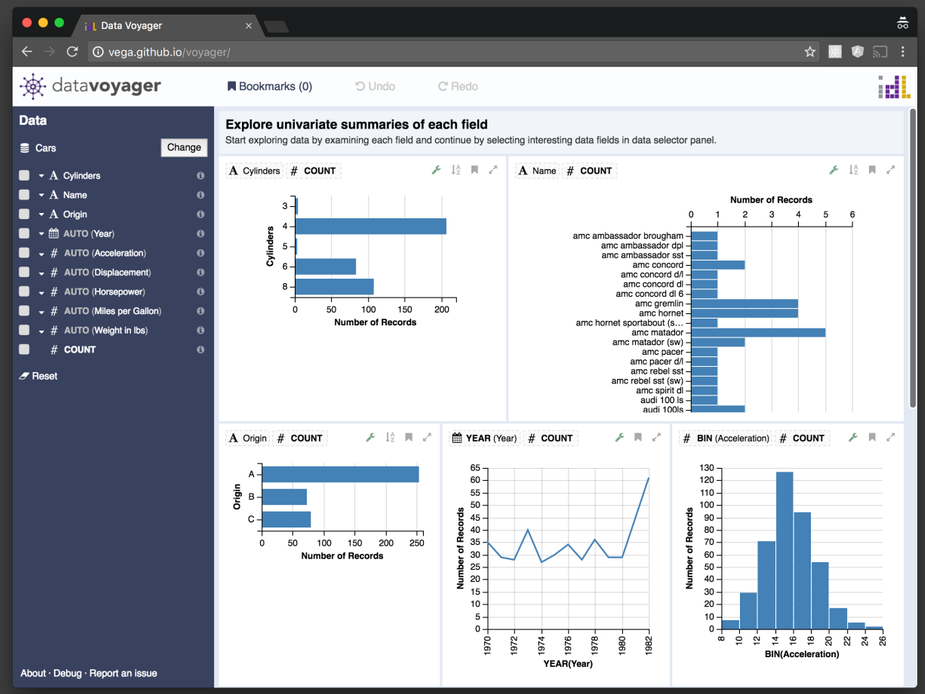
Protovis, D3, **Vega**

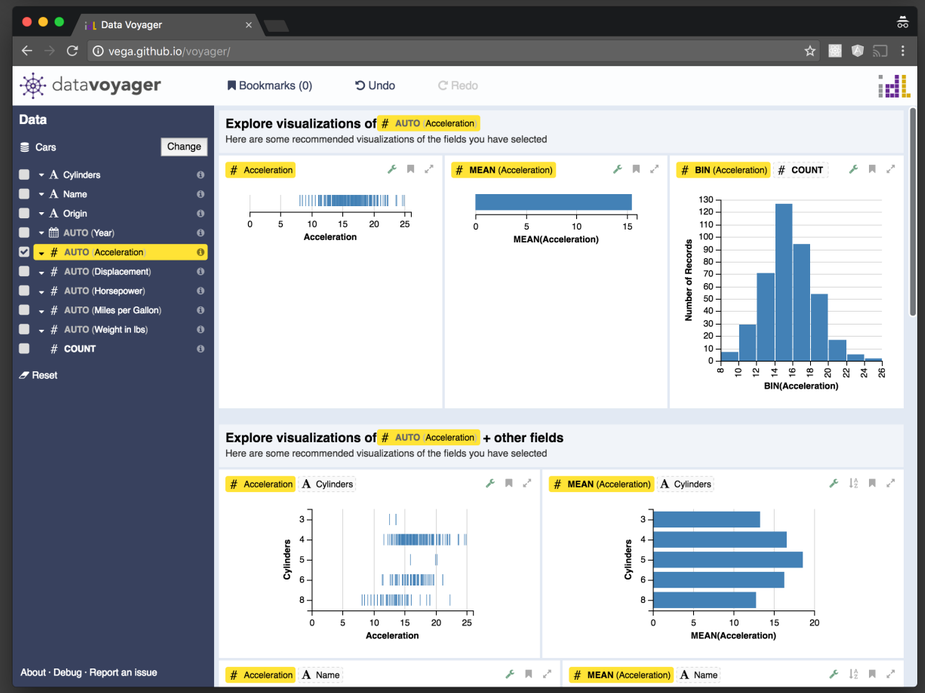
Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Processing, OpenGL, Java2D





Ease-of-Use

Expressiveness

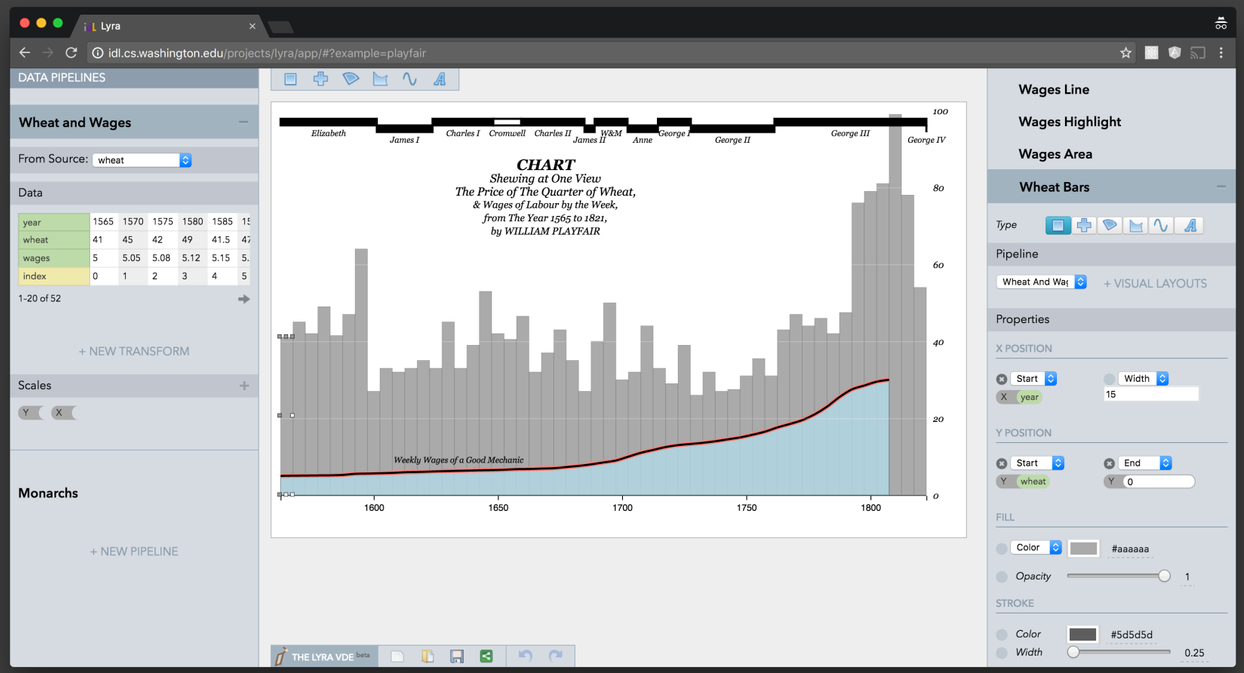
Interactive Data Systems
Tableau, PoleStar, Voyager, Lyra

Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

Visualization Grammars
Protovis, D3, **Vega**

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D



Ease-of-Use

Interactive Data Systems
Tableau, PoleStar, Voyager, Lyra

Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

Visualization Grammars
Protovis, D3, **Vega**

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D

Expressiveness

Ease-of-Use

Interactive Data Systems
Tableau, PoleStar, Voyager, Lyra

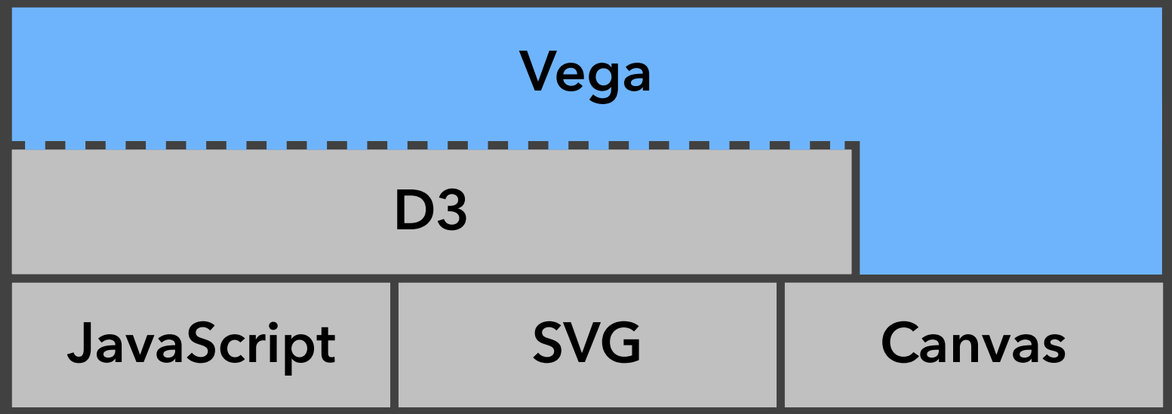
Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

Visualization Grammars
Protovis, D3, **Vega**

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D

Expressiveness



Ease-of-Use

Interactive Data Systems
Tableau, PoleStar, Voyager, Lyra

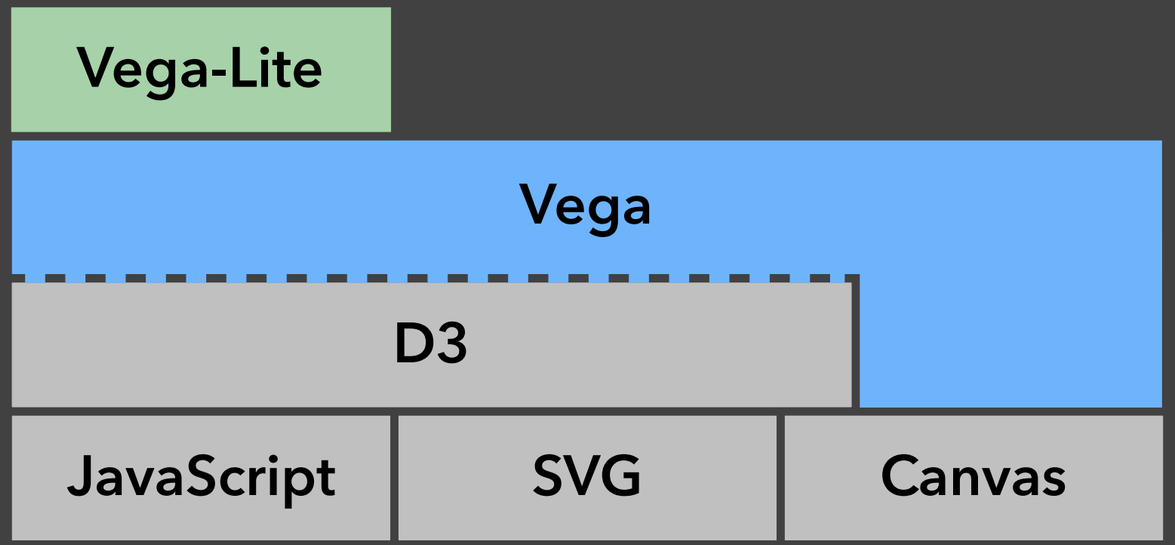
Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

Visualization Grammars
Protovis, D3, **Vega**

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D

Expressiveness



Ease-of-Use

Interactive Data Systems
Tableau, PoleStar, Voyager, Lyra

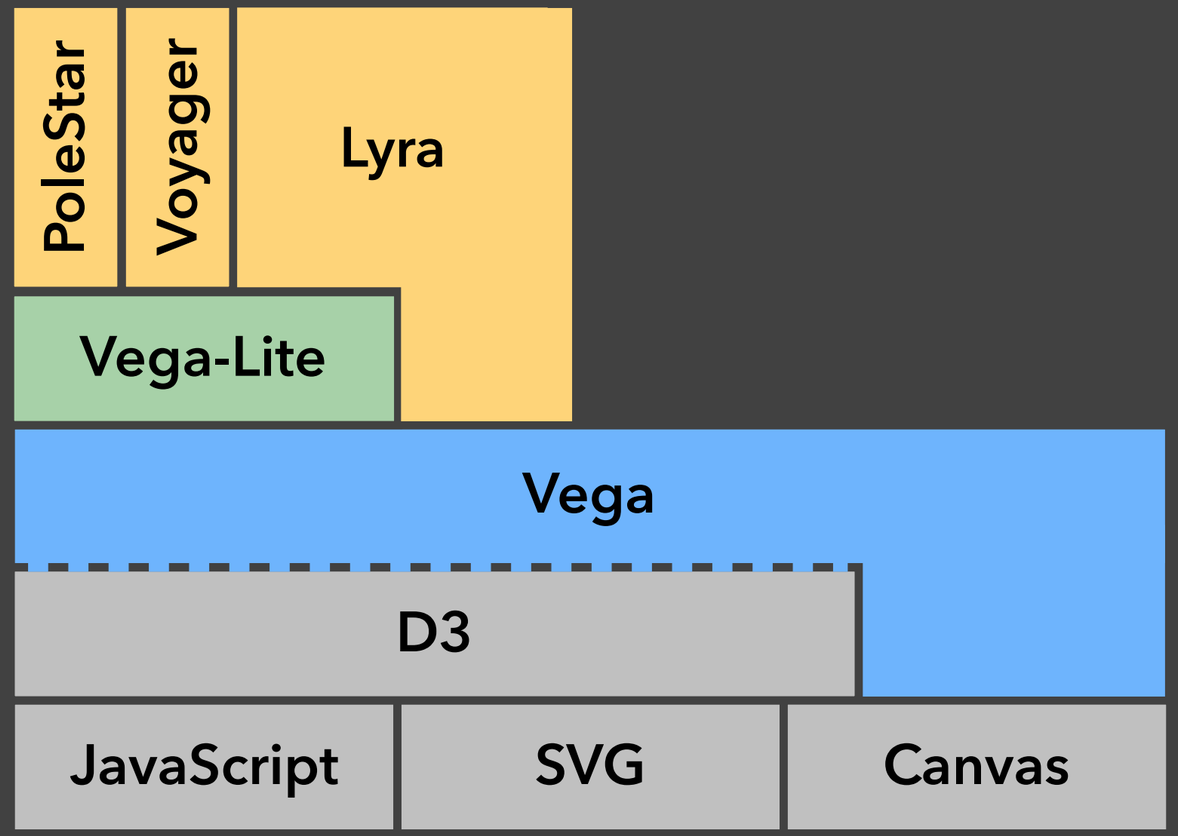
Visual Analysis Grammars
VizQL, ggplot2, Vega-Lite

Visualization Grammars
Protovis, D3, Vega

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D

Expressiveness



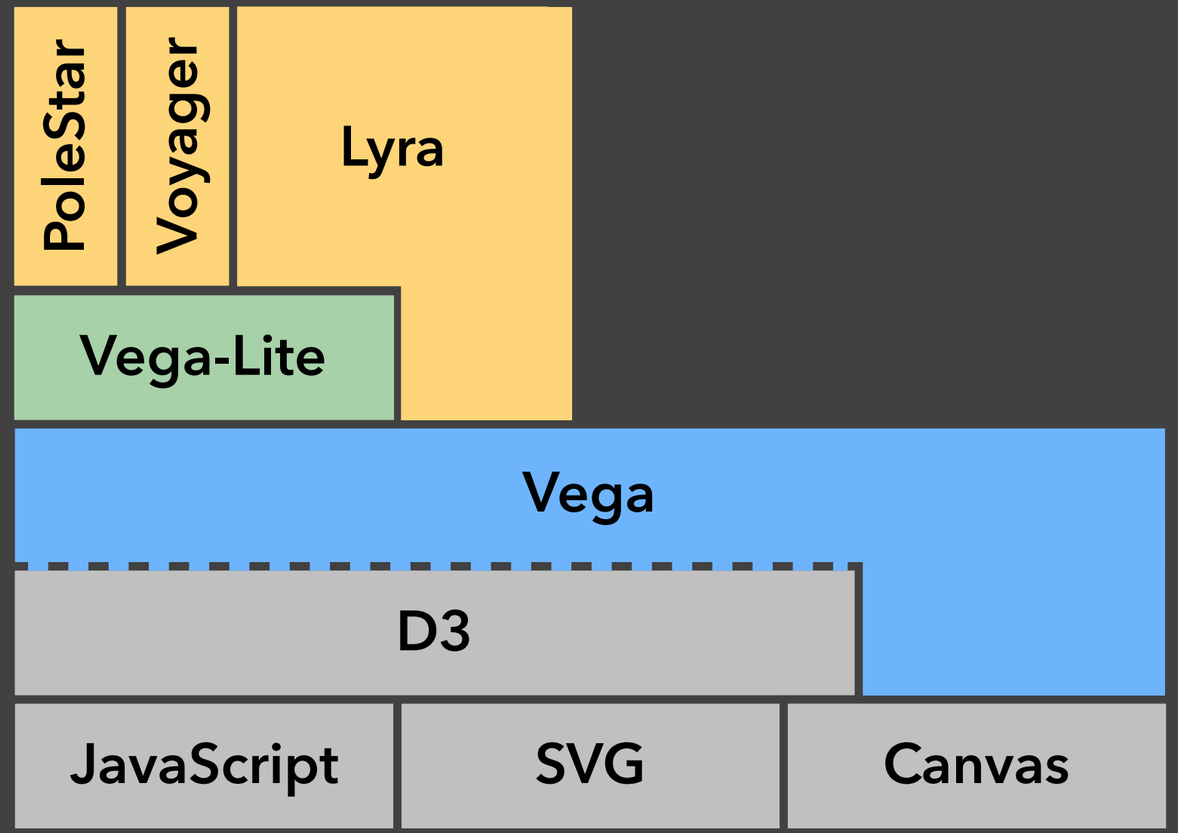
Interactive Data Systems
Tableau, PoleStar, Voyager, Lyra

Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

Visualization Grammars
Protovis, D3, **Vega**

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D



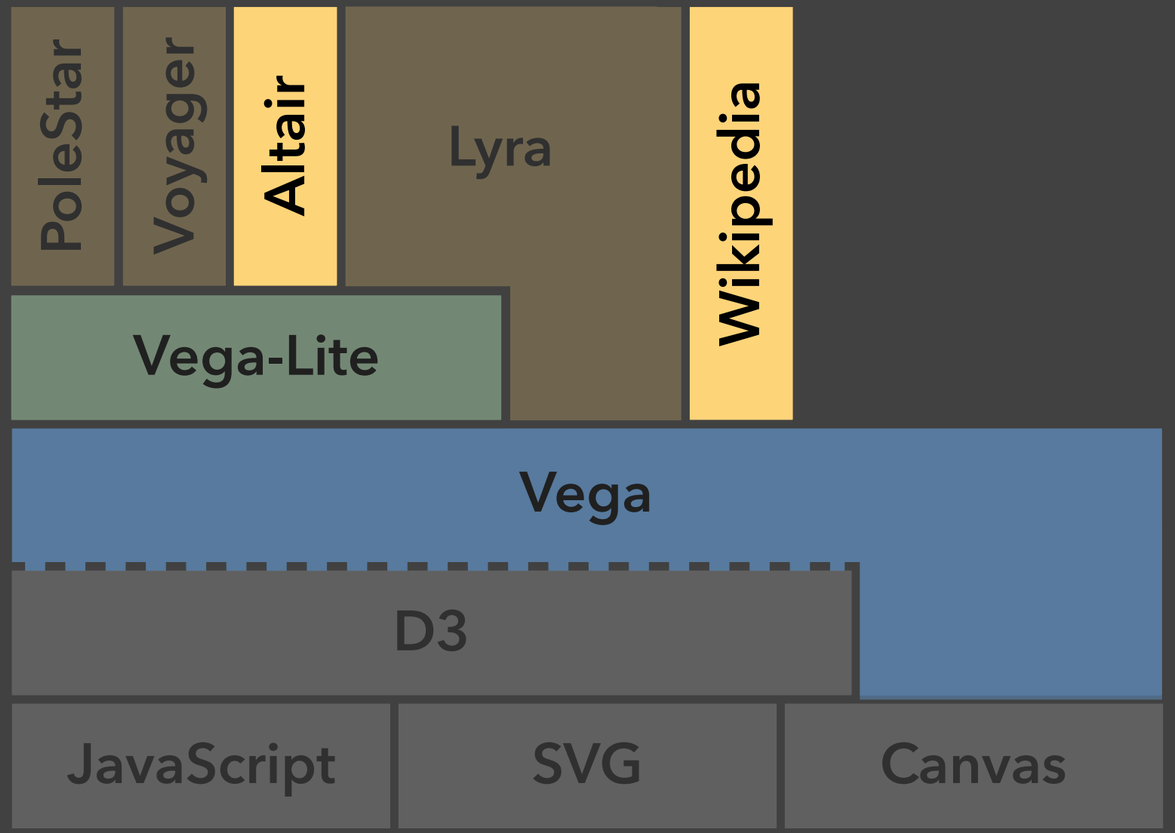
Interactive Data Systems
Tableau, PoleStar, Voyager, Lyra

Visual Analysis Grammars
VizQL, ggplot2, **Vega-Lite**

Visualization Grammars
Protovis, D3, **Vega**

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D



Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Wikipedia: Graph Extension

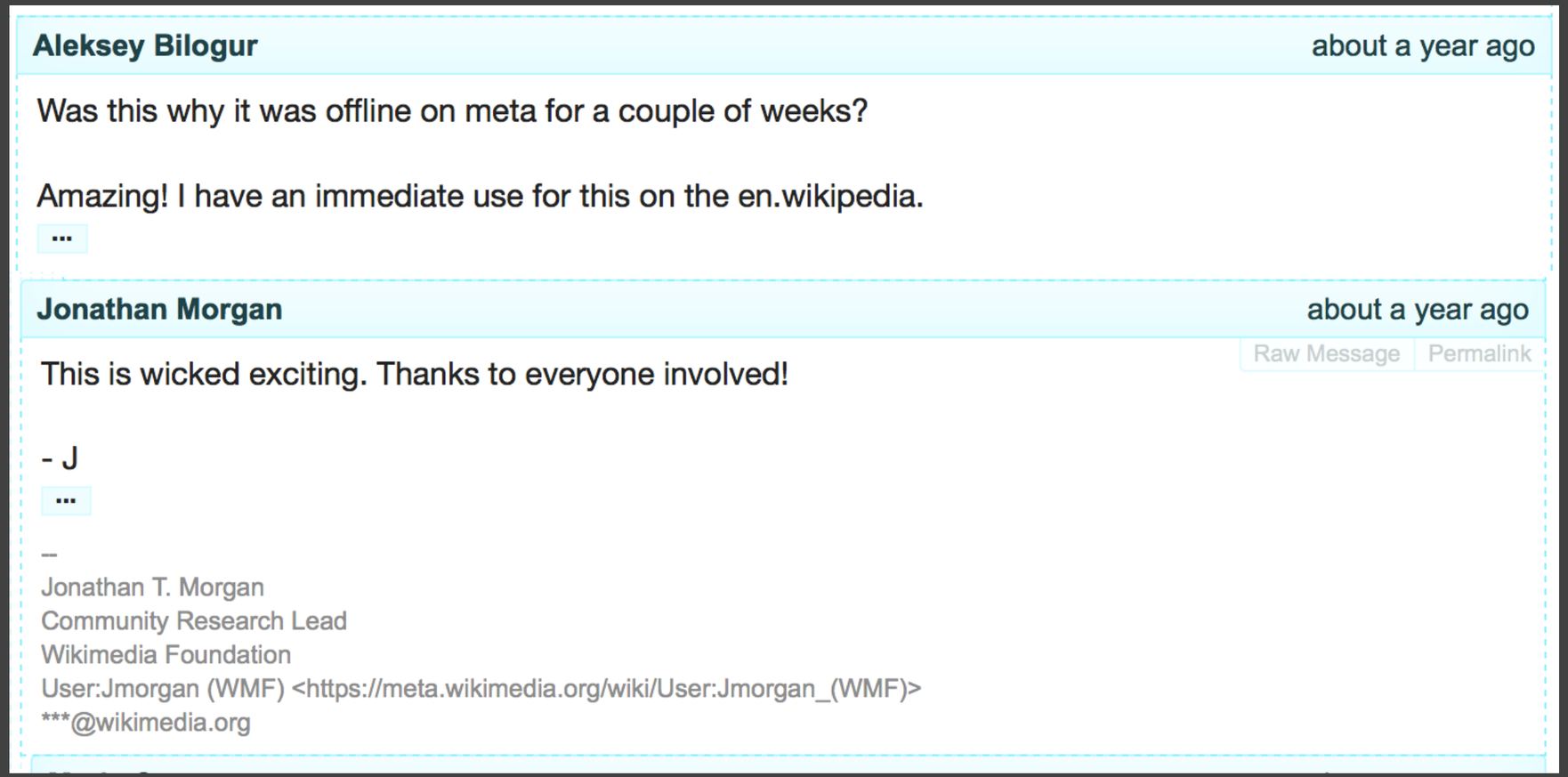
Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.



Aleksey Bilogur about a year ago

Was this why it was offline on meta for a couple of weeks?

Amazing! I have an immediate use for this on the en.wikipedia.

...

Jonathan Morgan about a year ago

Raw Message Permalink

This is wicked exciting. Thanks to everyone involved!

- J

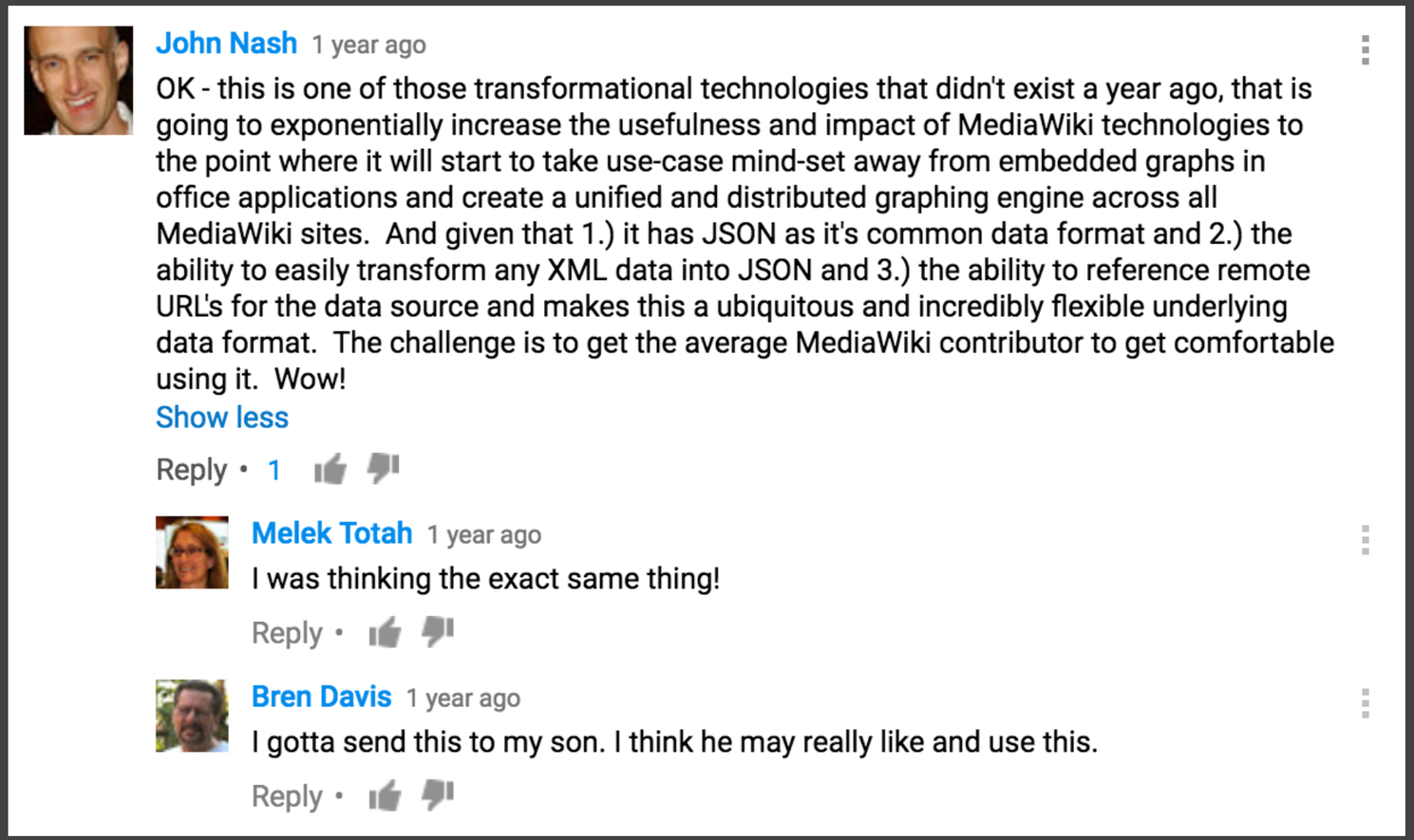
...

–
Jonathan T. Morgan
Community Research Lead
Wikimedia Foundation
User:Jmorgan (WMF) <https://meta.wikimedia.org/wiki/User:Jmorgan_(WMF)>
***@wikimedia.org

Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.



John Nash 1 year ago

OK - this is one of those transformational technologies that didn't exist a year ago, that is going to exponentially increase the usefulness and impact of MediaWiki technologies to the point where it will start to take use-case mind-set away from embedded graphs in office applications and create a unified and distributed graphing engine across all MediaWiki sites. And given that 1.) it has JSON as it's common data format and 2.) the ability to easily transform any XML data into JSON and 3.) the ability to reference remote URL's for the data source and makes this a ubiquitous and incredibly flexible underlying data format. The challenge is to get the average MediaWiki contributor to get comfortable using it. Wow!

Show less

Reply • 1

Melek Totah 1 year ago

I was thinking the exact same thing!

Reply •

Bren Davis 1 year ago

I gotta send this to my son. I think he may really like and use this.

Reply •

Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

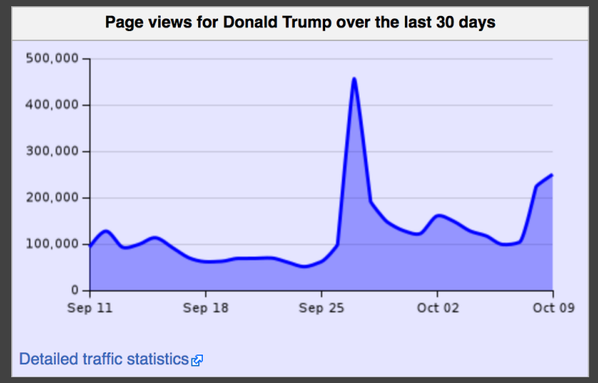
Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.

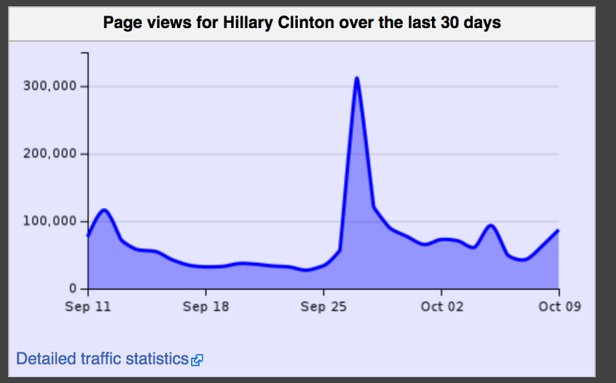
Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.



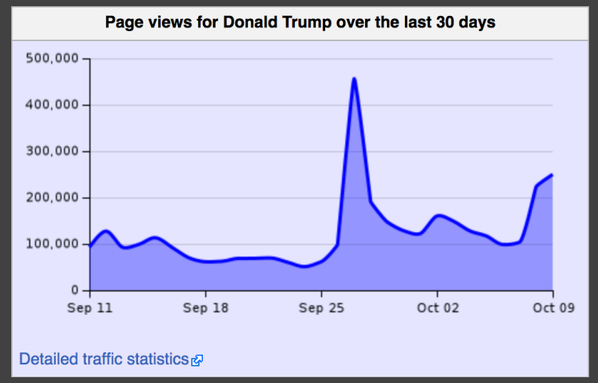


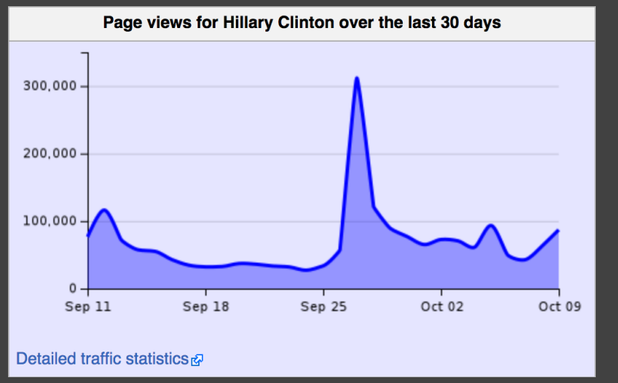
Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.





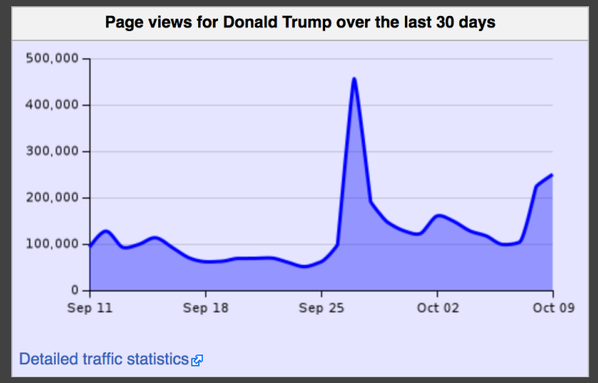

```
{{Graph:PageViews}}
```

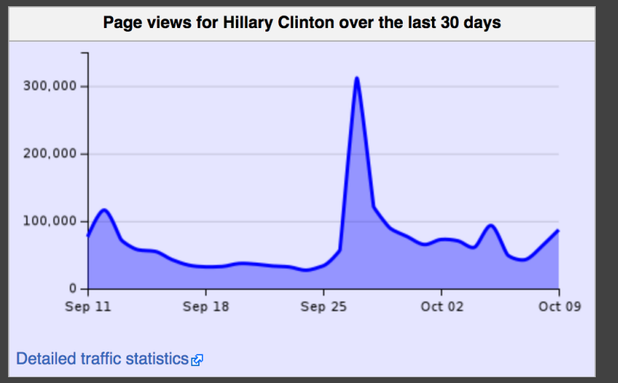
Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.





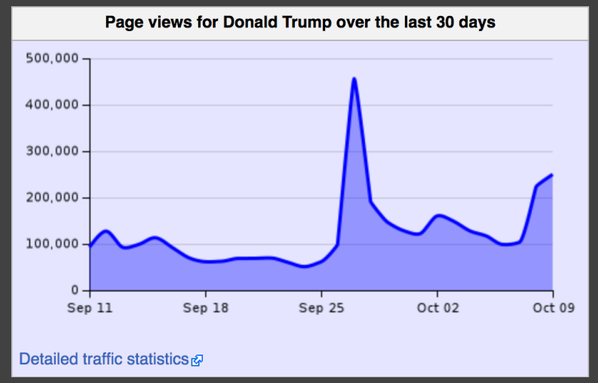

```
{{Graph:PageViews}}
{{Graph:PageViews | width = 200}}
```

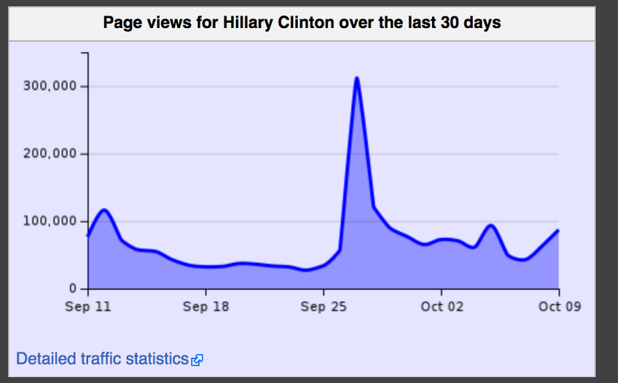
Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.





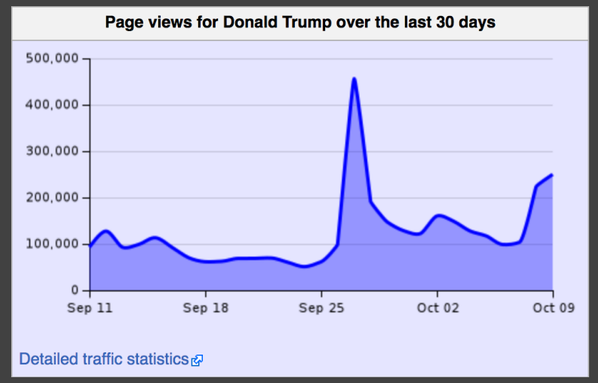

```
{{Graph:PageViews}}
{{Graph:PageViews | width = 200}}
{{Graph:PageViews | width = 200 | days = 90}}
```

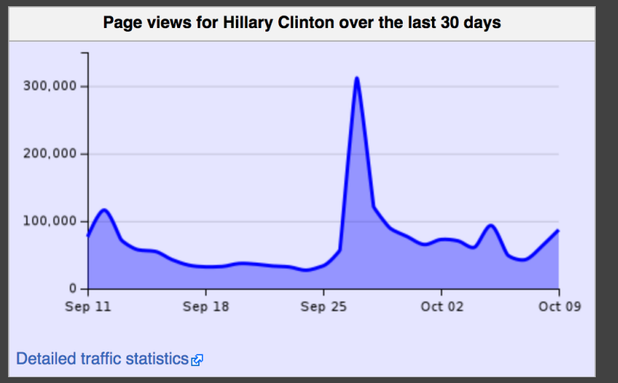
Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

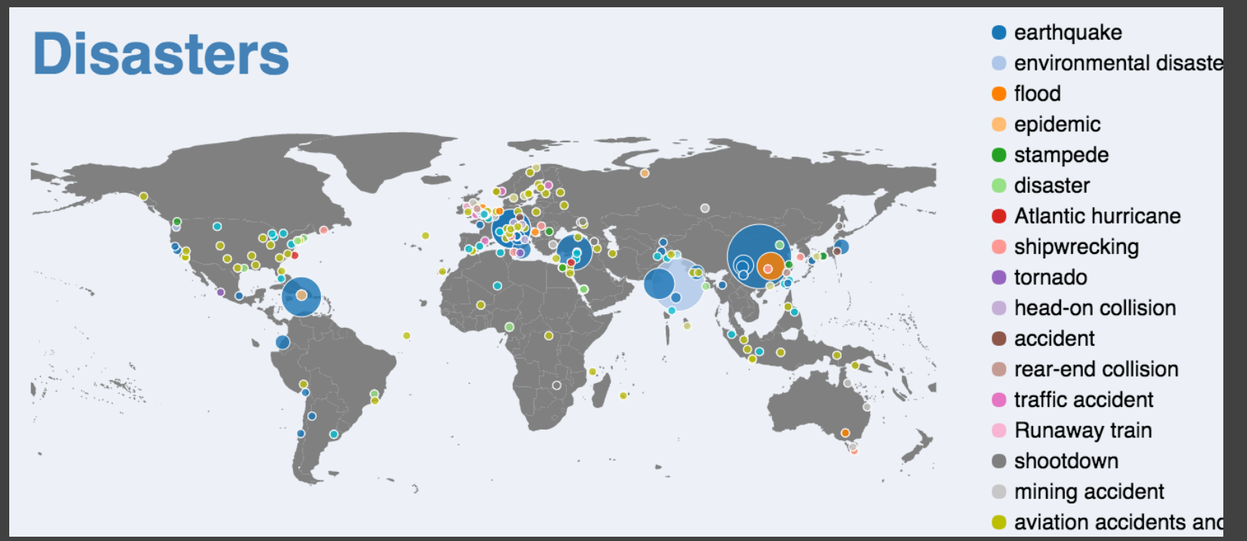
Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.






```
{{Graph:PageViews}}
{{Graph:PageViews | width = 200}}
{{Graph:PageViews | width = 200 | days = 90}}
```



Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.

For performance reasons, visualizations are rendered server-side and displayed as a **png**.

Click to enable interactivity: the Vega library is loaded, and the visualization re-rendered.

Wikipedia: Graph Extension

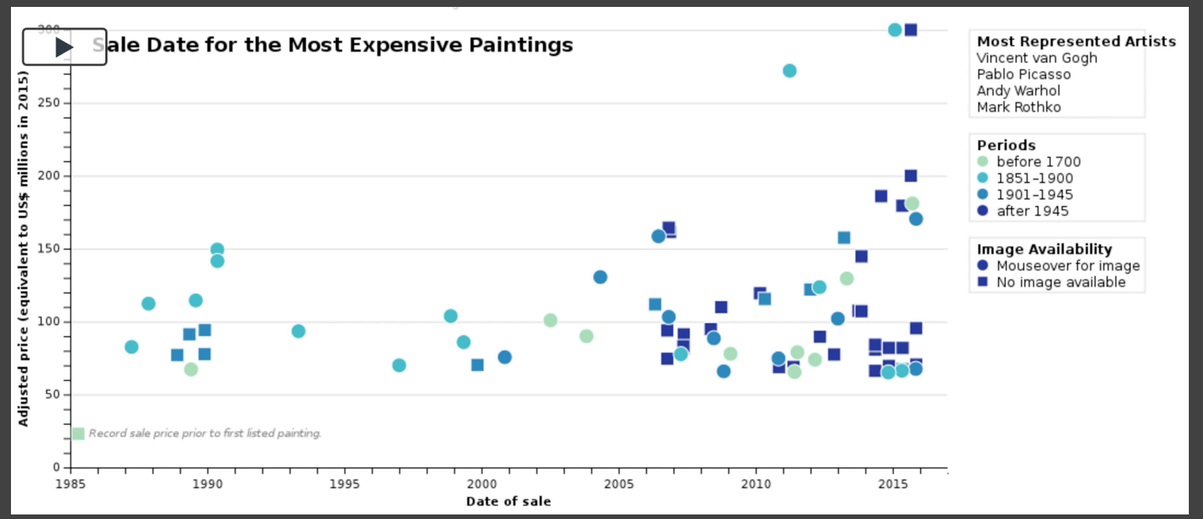
Led by Yuri Astrakhan from WikiMedia.

Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.

For performance reasons, visualizations are rendered server-side and displayed as a **png**.

Click to enable interactivity: the Vega library is loaded, and the visualization re-rendered.



Wikipedia: Graph Extension

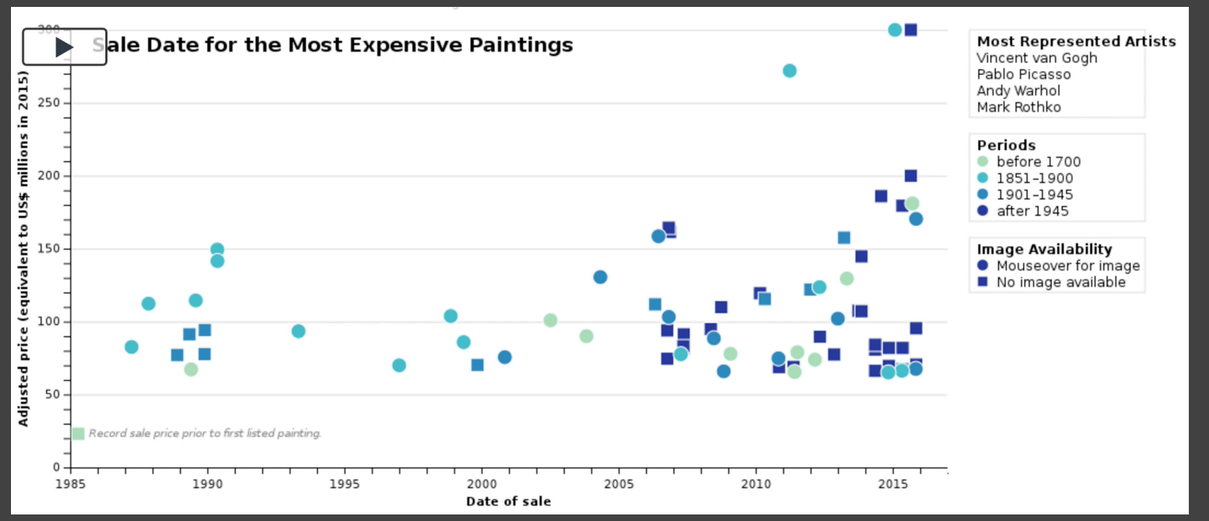
Led by Yuri Astrakhan from WikiMedia.

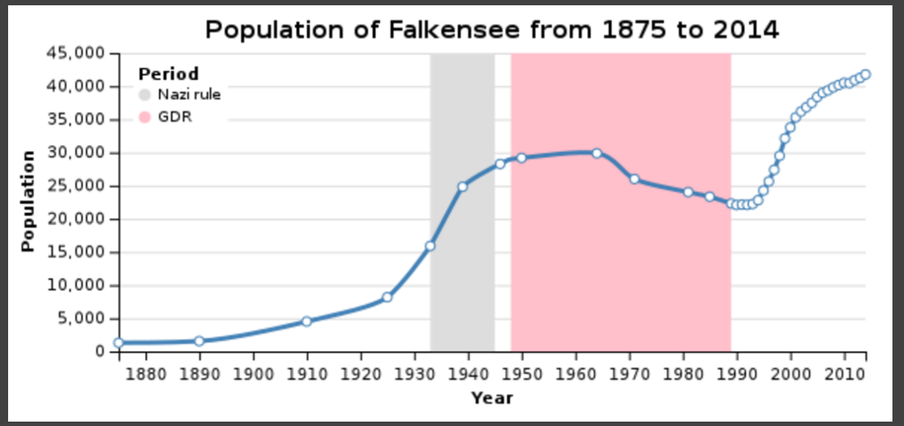
Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.

For performance reasons, visualizations are rendered server-side and displayed as a **png**.
Click to enable interactivity: the Vega library is loaded, and the visualization re-rendered.

374 pages currently have Vega visualizations on English Wikipedia. Millions across all languages.





Wikipedia: Graph Extension

Led by Yuri Astrakhan from WikiMedia.

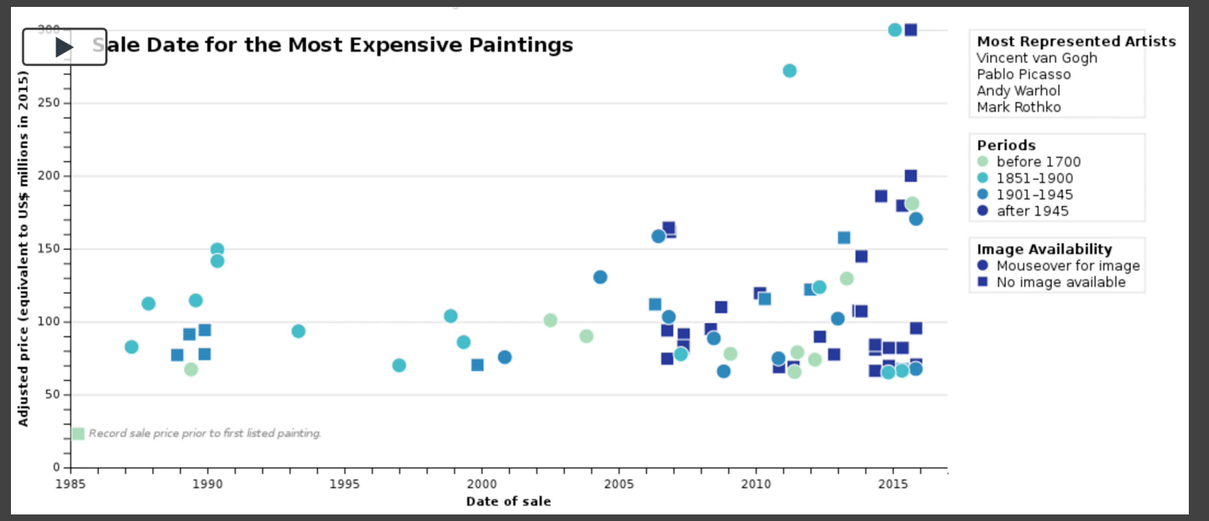
Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

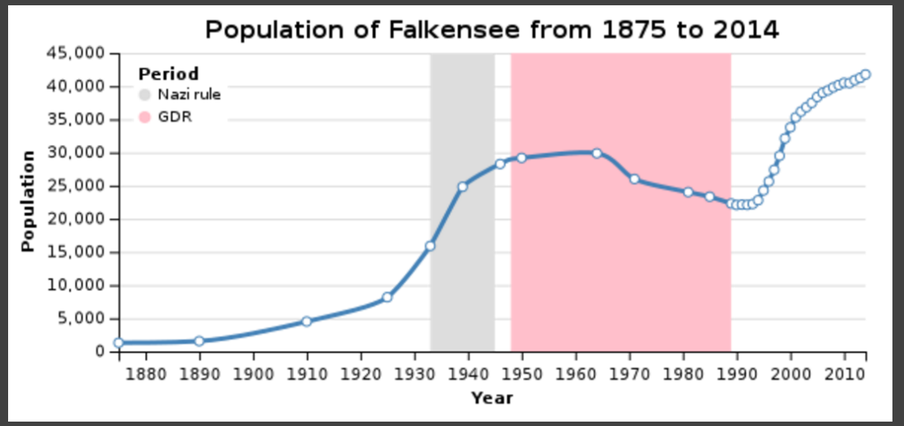
Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.

For performance reasons, visualizations are rendered server-side and displayed as a **png**.
Click to enable interactivity: the Vega library is loaded, and the visualization re-rendered.

374 pages currently have Vega visualizations on English Wikipedia. Millions across all languages.

Limitations: (1) Data still inaccessible.





Wikipedia: Graph Extension

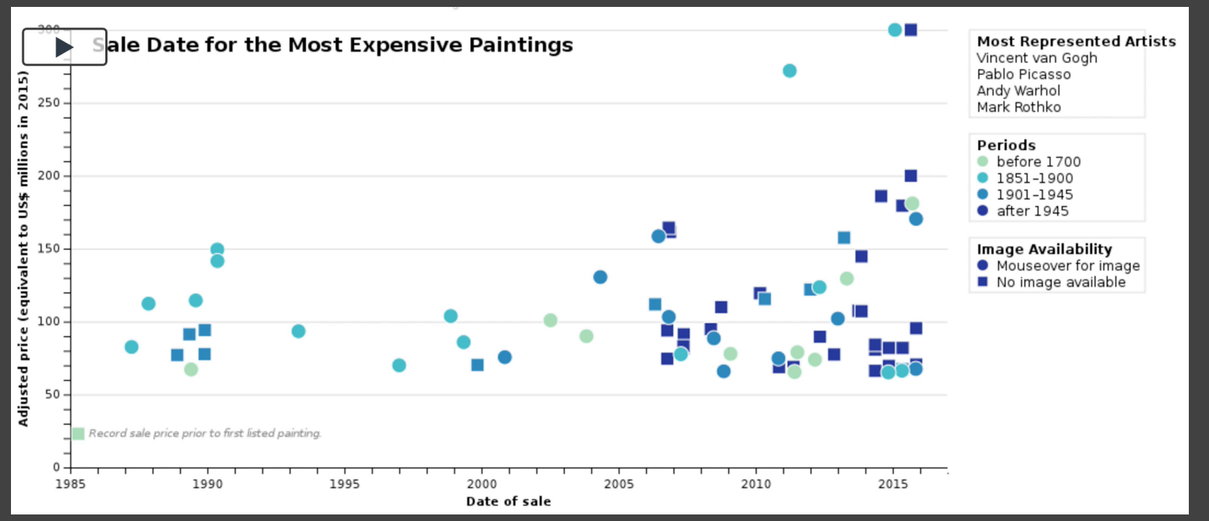
Led by Yuri Astrakhan from WikiMedia.

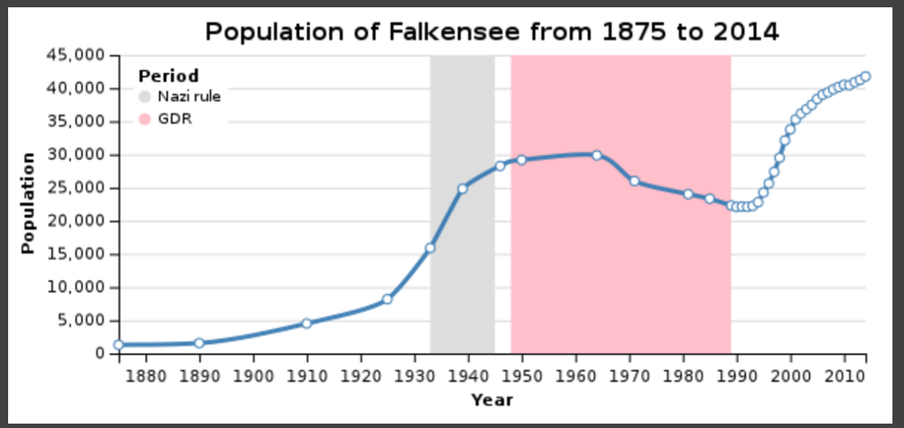
Static visualizations (Vega 1) in May 2015.
Interactive visualizations (Vega 2) in December.

Integrated with Wikipedia APIs including page analytics, Lua templating, and Wikidata.

For performance reasons, visualizations are rendered server-side and displayed as a **png**.
Click to enable interactivity: the Vega library is loaded, and the visualization re-rendered.

374 pages currently have Vega visualizations on English Wikipedia. Millions across all languages.





Limitations: (1) Data still inaccessible. (2) Vega/JSON difficult to write.

Wikipedia: Graph Extension

Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

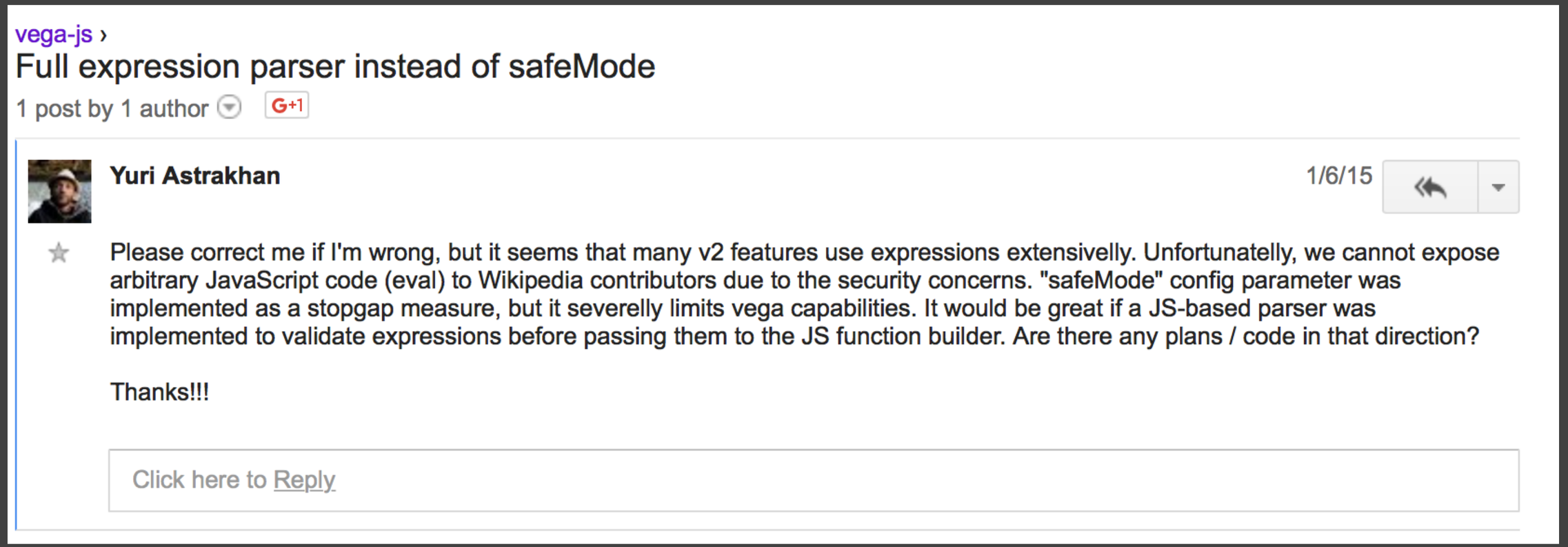
Pass expression string to Function constructor.

Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

Pass expression string to Function constructor.



Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

~~Pass expression string to Function constructor.~~

Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

~~Pass expression string to Function constructor.~~

Built in JavaScript parser (Esprima) that parses a limited subset of JavaScript expressions.

Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

~~Pass expression string to Function constructor.~~

Built in JavaScript parser (Esprima) that parses a limited subset of JavaScript expressions.

Has ultimately increased Vega's expressivity while reducing codebase surface area.

Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

~~Pass expression string to Function constructor.~~

Built in JavaScript parser (Esprima) that parses a limited subset of JavaScript expressions.

Has ultimately increased Vega's expressivity while reducing codebase surface area.

Data Loading

Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

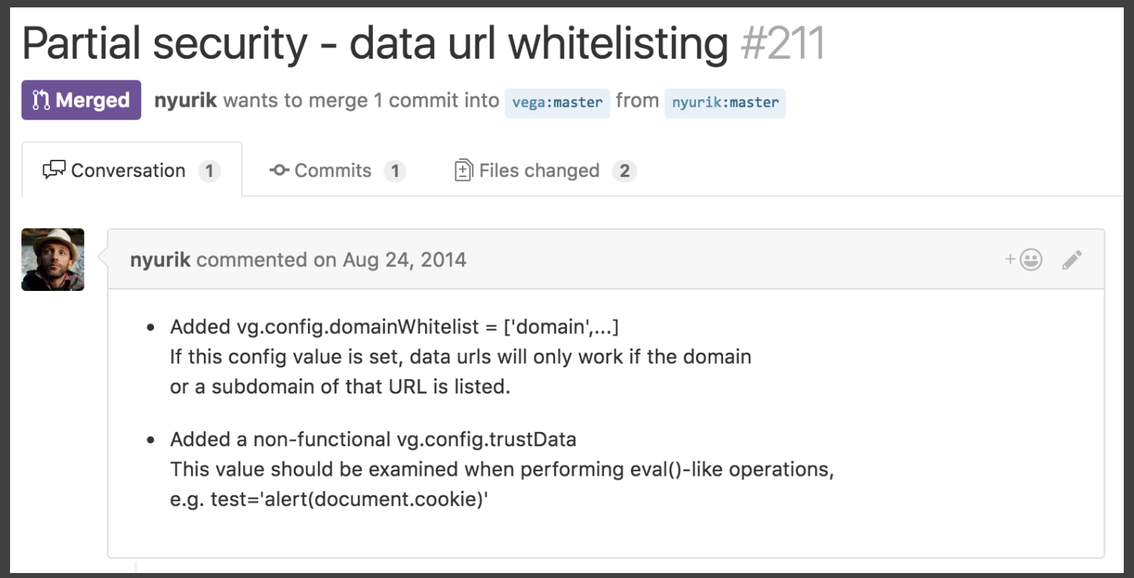
Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

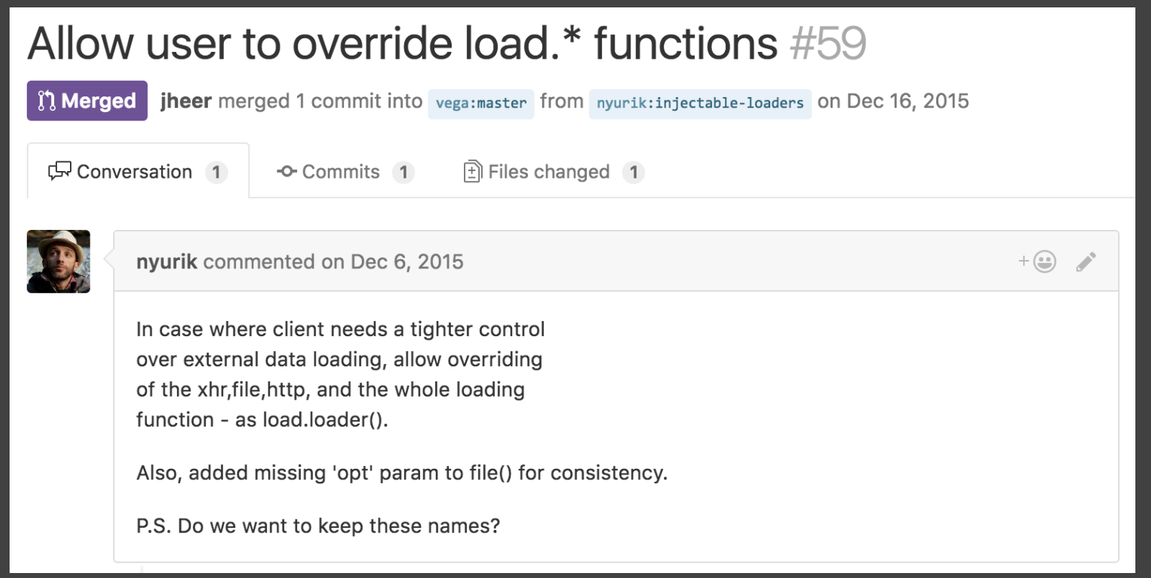
~~Pass expression string to Function constructor.~~

Built in JavaScript parser (Esprima) that parses a limited subset of JavaScript expressions.

Has ultimately increased Vega's expressivity while reducing codebase surface area.

Data Loading





Wikipedia: Graph Extension

Two major priorities which influenced Vega's design: **Security** and **Deployment**.

Vega Expression Parser (`"datum.quantity > 500"`, `"datum.price / 100"`, … → `Function`)

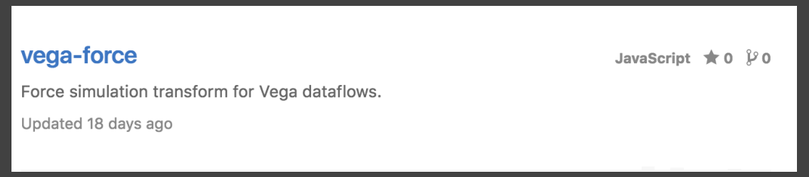
~~Pass expression string to Function constructor.~~

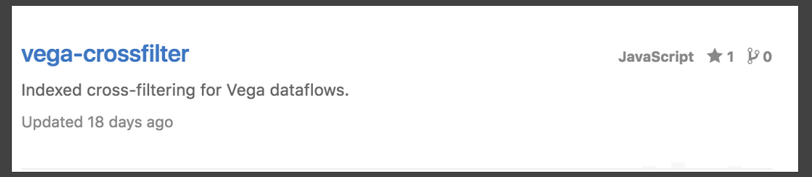
Built in JavaScript parser (Esprima) that parses a limited subset of JavaScript expressions.

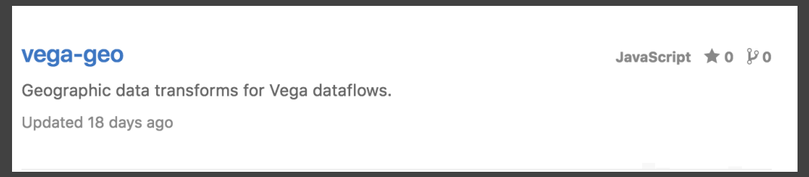
Has ultimately increased Vega's expressivity while simplifying codebase.

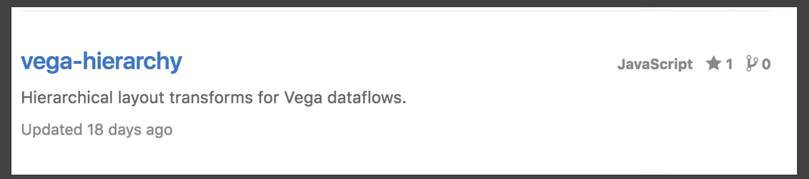
Data Loading – URL black/white-listing and custom handling (e.g., for WikiMedia protocols).

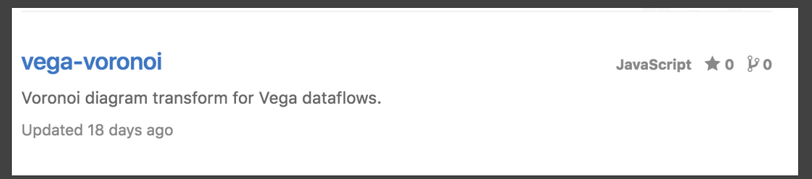
Modularity – only include needed functionality on a per-page basis.

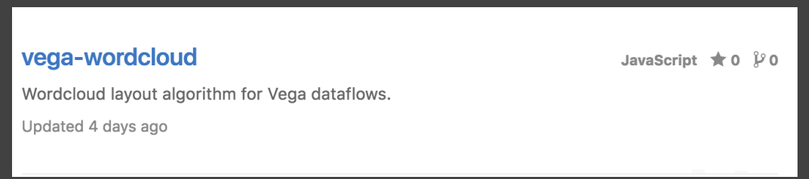








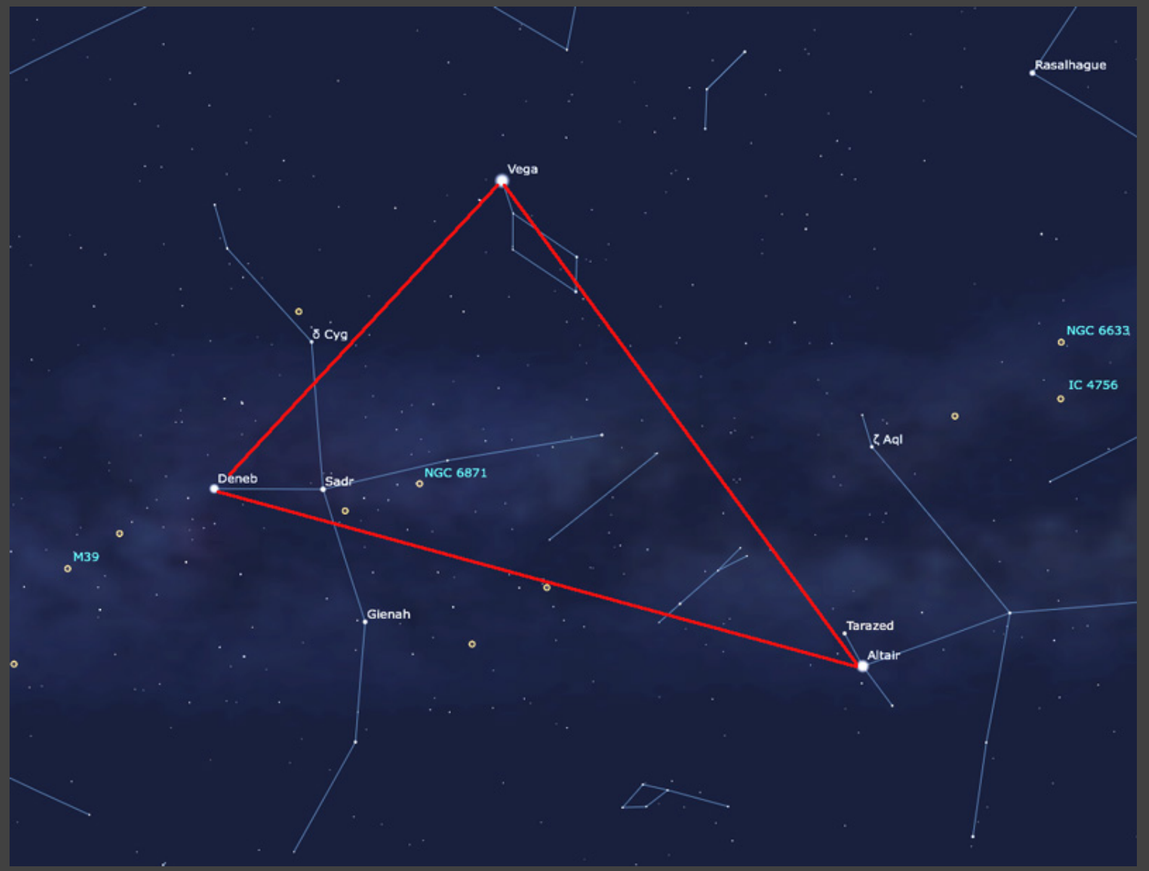




Altair: Vega-Lite in Python

Altair: Vega-Lite in Python

Led by Brian Granger and Jake VanDerPlas.



James Guilford, https://cuyastro.org/2011/07/15/a-summery-triangle/

Altair: Vega-Lite in Python

Led by Brian Granger and Jake VanDerPlas.

Python API automatically generated from the Vega-Lite JSON schema, with minor changes to make the it more *pythonic*.

Altair: Vega-Lite in Python

Led by Brian Granger and Jake VanDerPlas.

Python API automatically generated from the Vega-Lite JSON schema, with minor changes to make the it more *pythonic*.

> *“It is this type of 1:1:1 mapping between thinking, code, and visualization that is my favorite thing about [Altair]”* – Dan Saber.
>
> https://dansaber.wordpress.com/2016/10/02/a-dramatic-tour-through-pythons-data-visualization-landscape-including-ggplot-and-altair/

Altair: Vega-Lite in Python

Led by Brian Granger and Jake VanderPlas.

Python API automatically generated from the Vega-Lite JSON schema, with minor changes to make the it more *pythonic*.

Altair: Vega-Lite in Python

Led by Brian Granger and Jake VanderPlas.

Python API automatically generated from the Vega-Lite JSON schema, with minor changes to make the it more *pythonic*.

Interest from other Python visualization vendors (Matplotlib, Bokeh, Plotly) to make their own Vega-Lite renderers.

Altair: Vega-Lite in Python

Led by Brian Granger and Jake VanderPlas.

Python API automatically generated from the Vega-Lite JSON schema, with minor changes to make the it more *pythonic*.

Interest from other Python visualization vendors (Matplotlib, Bokeh, Plotly) to make their own Vega-Lite renderers.

> *"We see this portion of the effort as much bigger than Altair itself: the Vega and Vega-Lite specifications are perhaps the best existing candidates for a principled* lingua franca *of data visualization"* – Altair Team.

Altair: Vega-Lite in Python

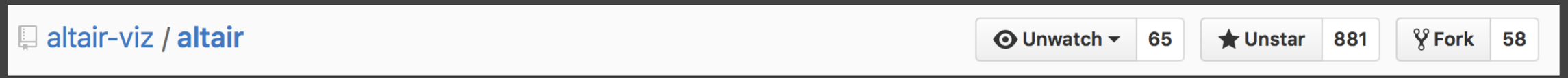
Led by Brian Granger and Jake VanderPlas.

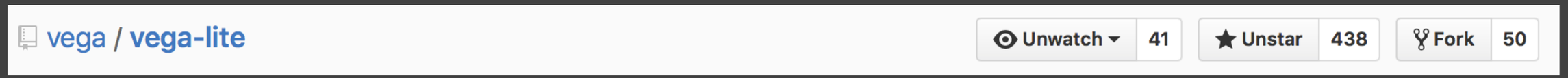
Python API automatically generated from the Vega-Lite JSON schema, with minor changes to make the it more *pythonic*.

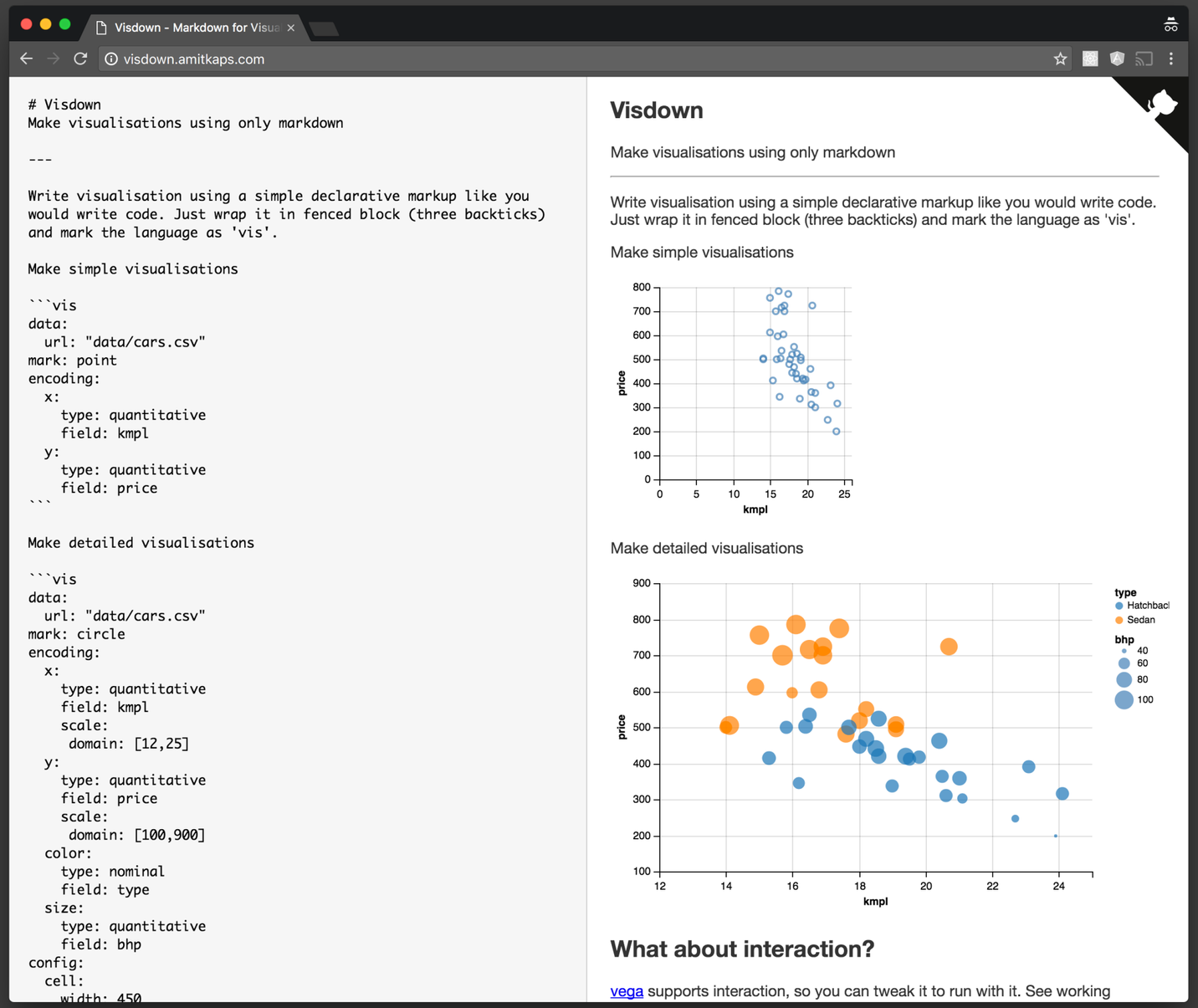
Interest from other Python visualization vendors (Matplotlib, Bokeh, Plotly) to make their own Vega-Lite renderers.

> *"We see this portion of the effort as much bigger than Altair itself: the Vega and Vega-Lite specifications are perhaps the best existing candidates for a principled* lingua franca *of data visualization"* – Altair Team.

Has helped prioritize features on the roadmap: box plots and more responsive layout.







Visdown, Amit Kapoor. http://visdown.amitkaps.com/

Visdown - Markdown for Visua

visdown.amitkaps.com

```
# Visdown
Make visualisations using only markdown

---

Write visualisation using a simple declarative markup like you
would write code. Just wrap it in fenced block (three backticks)
and mark the language as 'vis'.

Make simple visualisations

```vis
data:
  url: "data/cars.csv"
mark: point
encoding:
  x:
    type: quantitative
    field: kmpl
  y:
    type: quantitative
    field: price
```

Make detailed visualisations

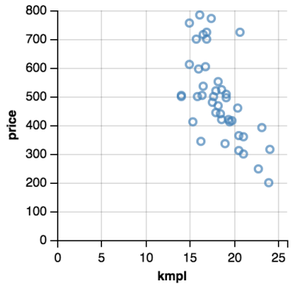
```vis
data:
  url: "data/cars.csv"
mark: circle
encoding:
  x:
    type: quantitative
    field: kmpl
    scale:
     domain: [12,25]
  y:
    type: quantitative
    field: price
    scale:
     domain: [100,900]
  color:
    type: nominal
    field: type
  size:
    type: quantitative
    field: bhp
config:
  cell:
    width: 450
```

Visdown

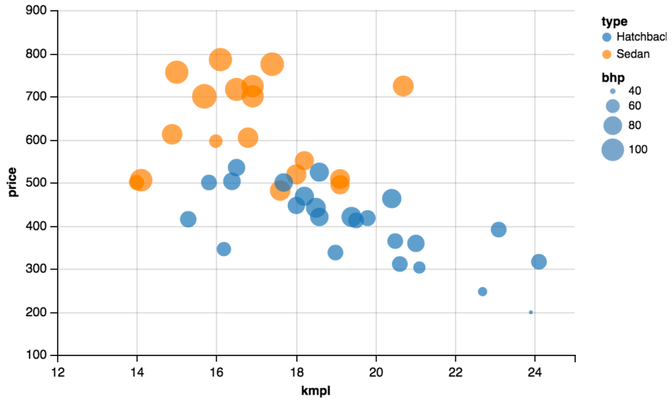
Make visualisations using only markdown

Write visualisation using a simple declarative markup like you would write code. Just wrap it in fenced block (three backticks) and mark the language as 'vis'.

Make simple visualisations



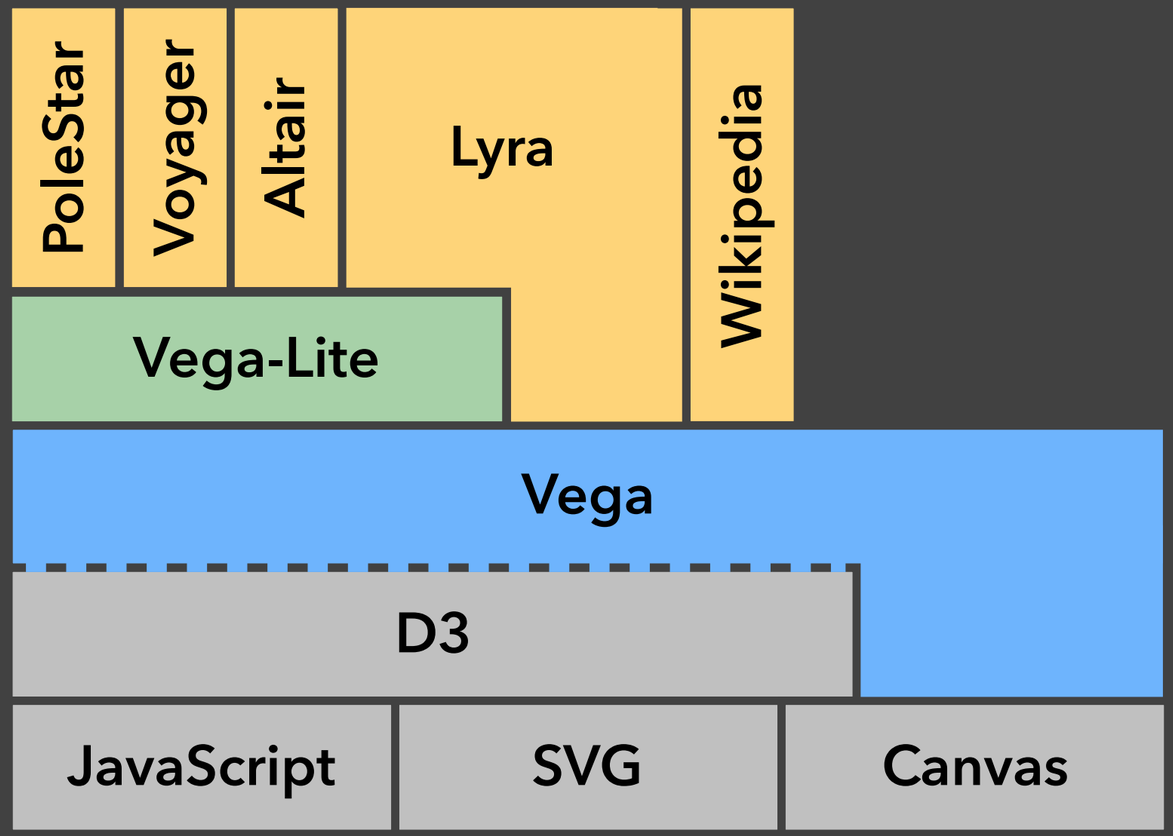
Make detailed visualisations

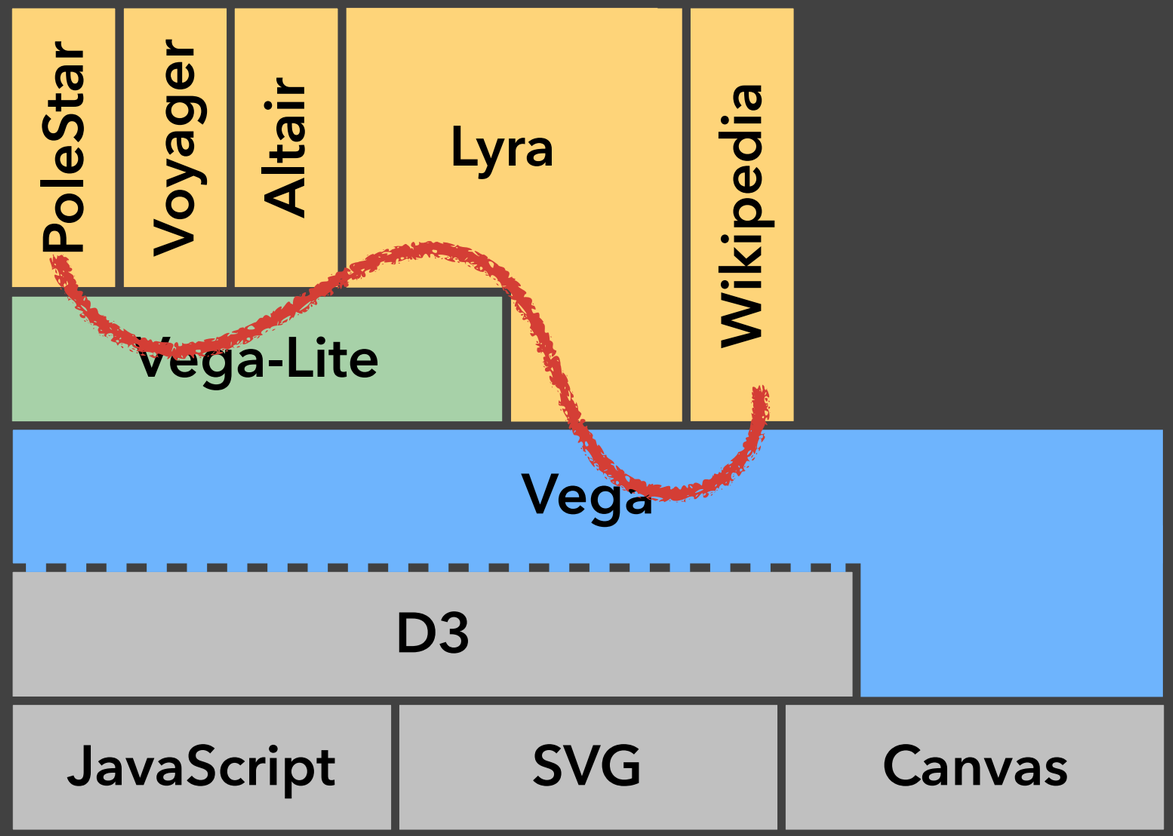


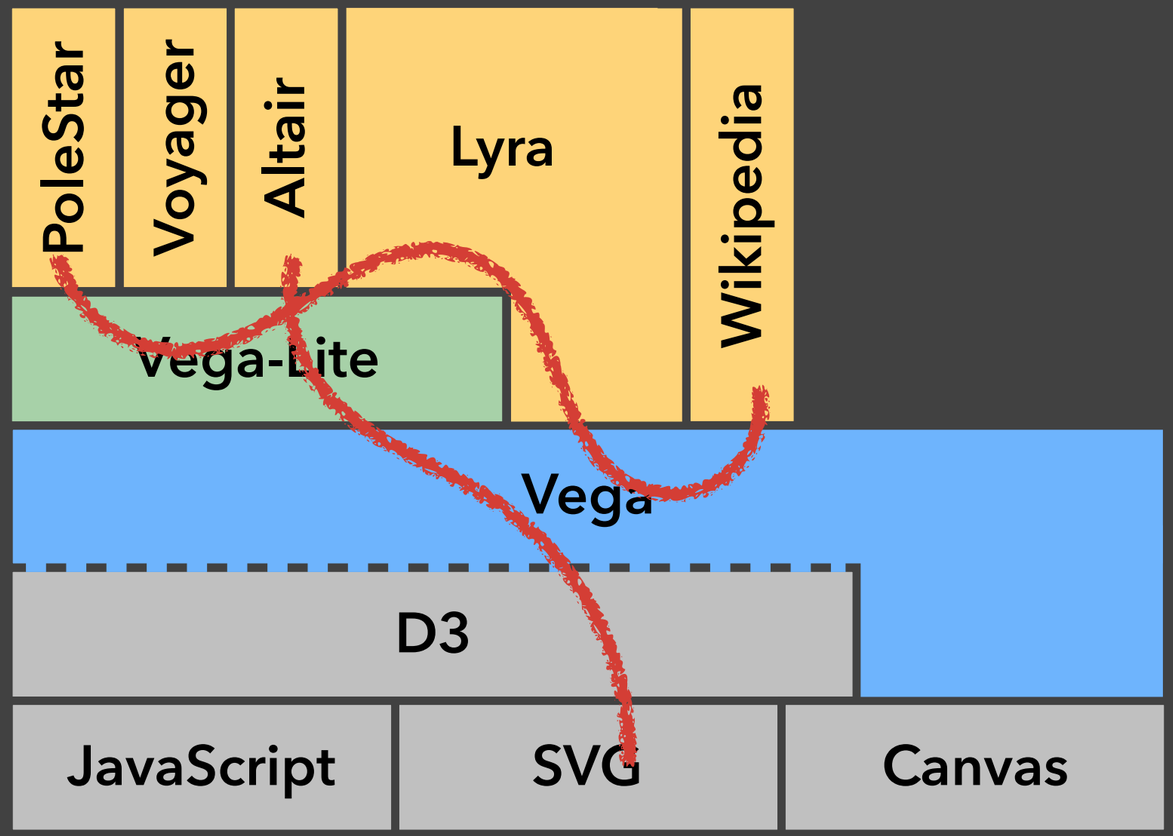
What about interaction?

vega supports interaction, so you can tweak it to run with it. See working

Visdown, Amit Kapoor. http://visdown.amitkaps.com/
```


PoleStar
Voyager
Altair
Lyra
Wikipedia
Vega-Lite
Vega
D3
JavaScript
SVG
Canvas


PoleStar
Voyager
Altair
Lyra
Wikipedia
Vega-Lite
Vega
D3
JavaScript
SVG
Canvas


PoleStar
Voyager
Altair
Lyra
Wikipedia
Vega-Lite
Vega
D3
JavaScript
SVG
Canvas

Marketing

Interactive Data Lab Retweeted

Arvind Satyanarayan @arvindsatya1 · Aug 9

Honored our work on interactive Vega-Lite won Best Paper. Excited to present at InfoVis '16. vimeo.com/177767802 idl.cs.washington.edu/papers/vega-li…

Vega-Lite: A Grammar of Interactive Graphics

We present Vega-Lite, a high-level grammar that enables rapid specification of interactive data visualizations. Vega-Lite combines a traditional grammar of graphics,…

vimeo.com

23 68

You Retweeted

Interactive Data Lab @uwdata · 14 Jul 2015

Announcing the release of Vega 2.0! Now with rich interaction, streaming data, improved rendering and more: vega.github.io/vega

17 12

Arvind Satyanarayan @arvindsatya1 · 14 Jul 2015

After 500+ commits, proud to announce the release of Vega 2.0 featuring streaming data and declarative interaction. vega.github.io/vega/

19 38

Interactive Data Lab @uwdata · Feb 23

Announcing the 1.0 release of Vega-Lite: a concise, high-level visualization language for analysis and presentation!

Introducing Vega-Lite – HCI & Design at UW

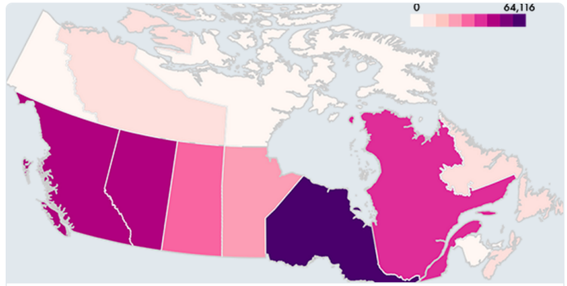
Today we are excited to announce the official 1.0 release of Vega-Lite, a high-level format for rapidly creating visualizations for…

medium.com

97 103

Interactive Data Lab @uwdata · Sep 27

Bayesian Surprise Maps to show the unexpected. New @uwdata blog post by Michael Correll!



Surprise Maps: Showing the Unexpected

In 1977, Jerry Ehman — an astronomer working with the SETI project to seek out alien life — came across an interesting radio signal, one…

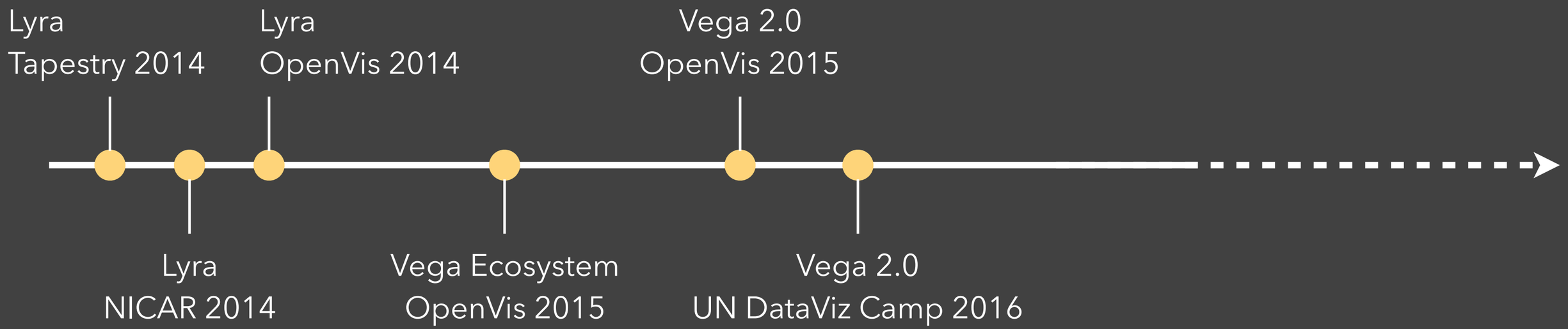
medium.com

17 26

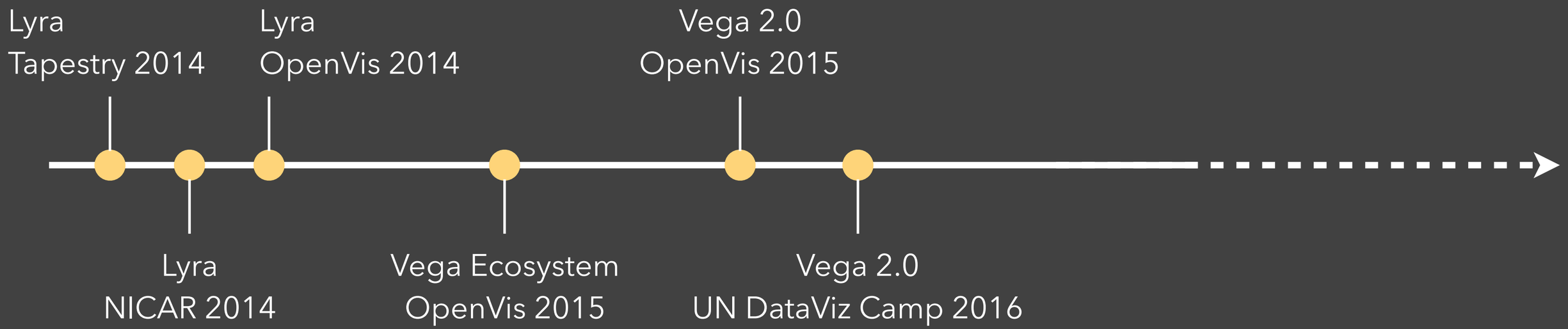
Stories Responses

Date ↑	Views	Reads	Read ratio	Recommends
Surprise Maps: Showing the Unexpected 9 min read · View story · Referrers	1.8K	392	22%	17
Atlas of Me: Personalized Spatial Analogy Maps for Unf... 10 min read · In HCI & Design at UW · View story · Referrers	298	93	31%	8
Author's Response to Stephen Few's critique of Hypoth... 7 min read · View story · Referrers	579	332	57%	10
Introducing Vega-Lite 5 min read · In HCI & Design at UW · View story · Referrers	20K	9.6K	48%	140
Hypothetical Outcome Plots: Experiencing the Uncert... 11 min read · In HCI & Design at UW · View story · Referrers	3.1K	1.2K	40%	22
Next Steps for Data Visualization Research 7 min read · View story · Referrers	3.6K	1.8K	51%	42

Marketing & Outreach

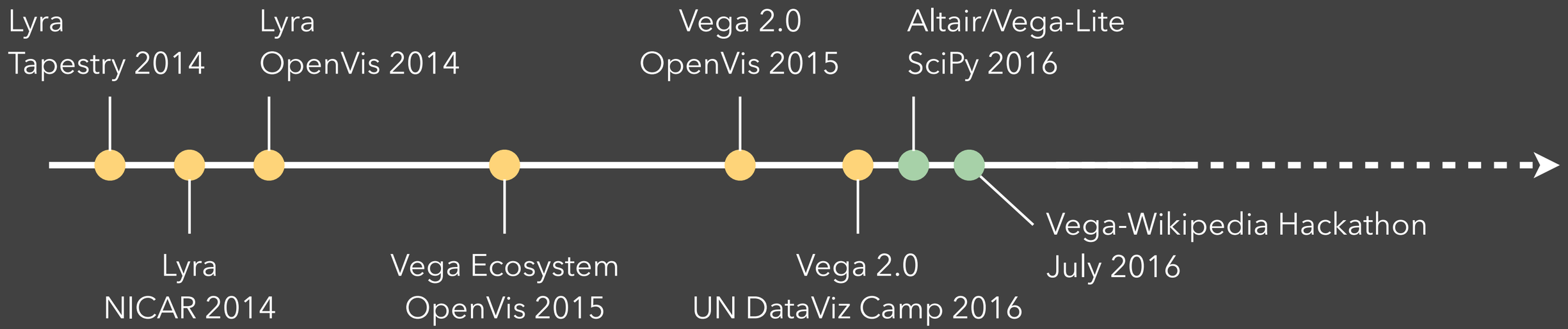


Marketing & Outreach



Talk about your work to diverse audiences, but craft the message for them specifically.

Marketing & Outreach



Talk about your work to diverse audiences, but craft the message for them specifically.

Challenges

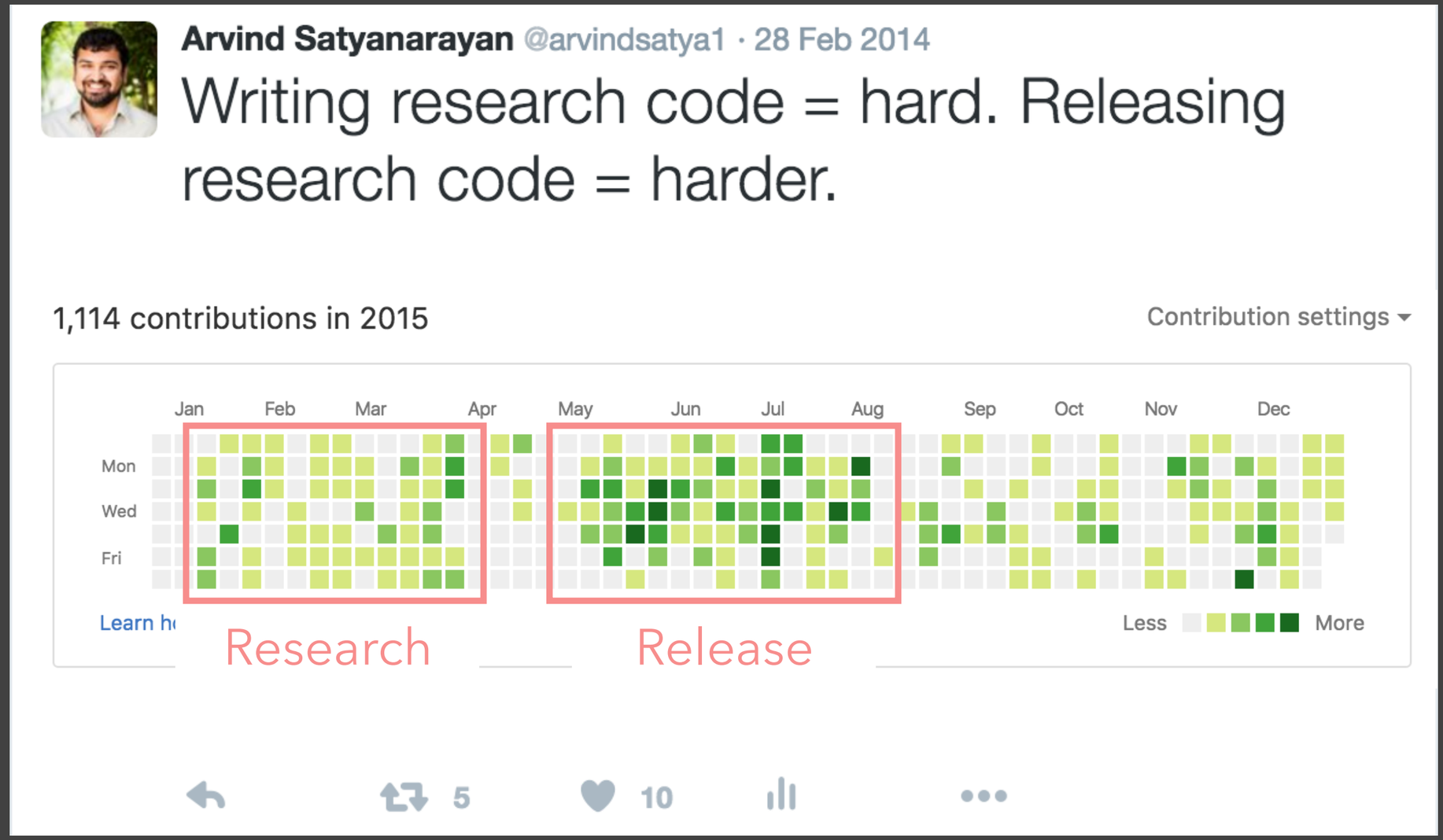
Challenges

Arvind Satyanarayan @arvindsatya1 · 28 Feb 2014

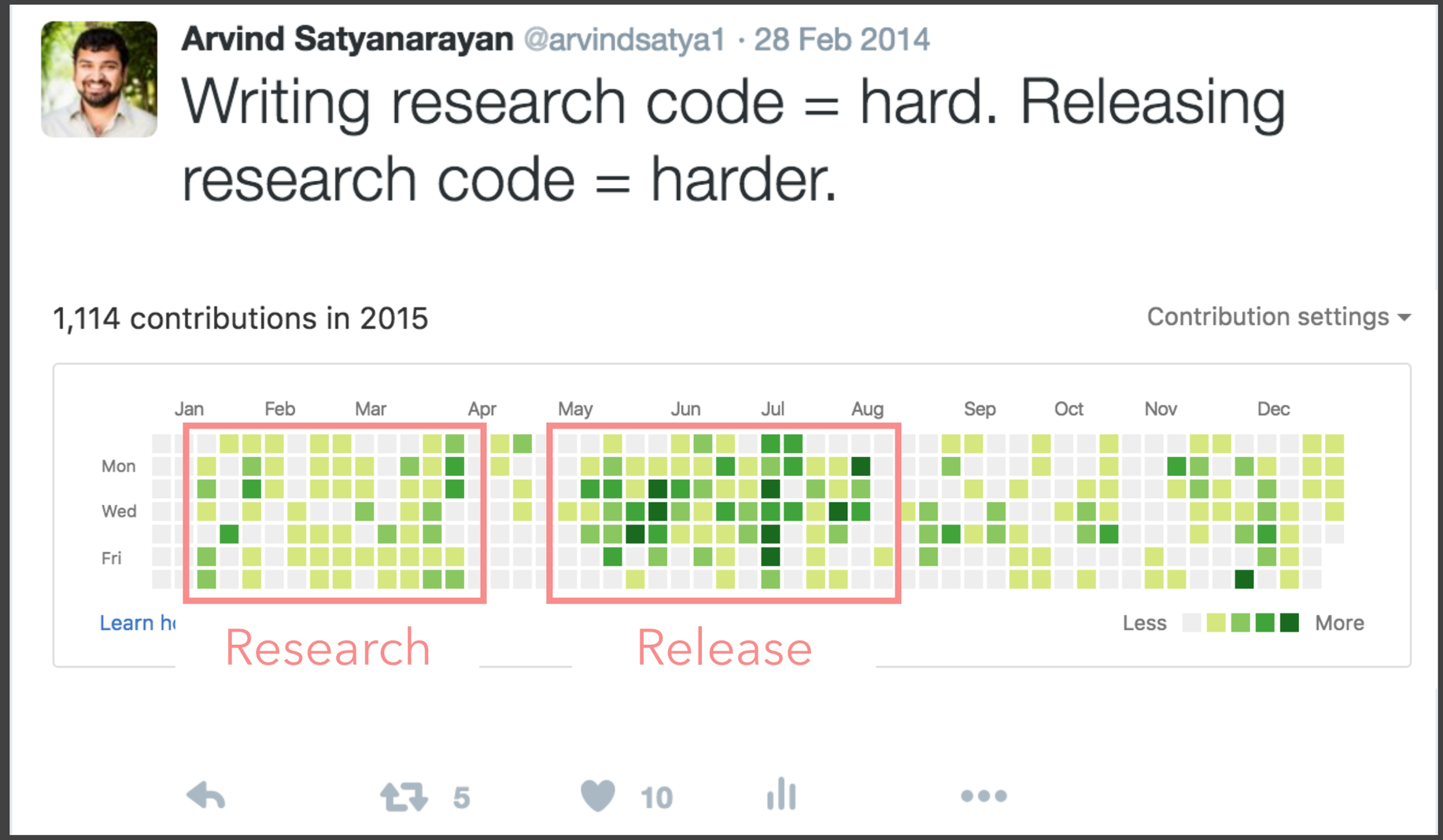
Writing research code = hard. Releasing research code = harder.

5 10

Challenges

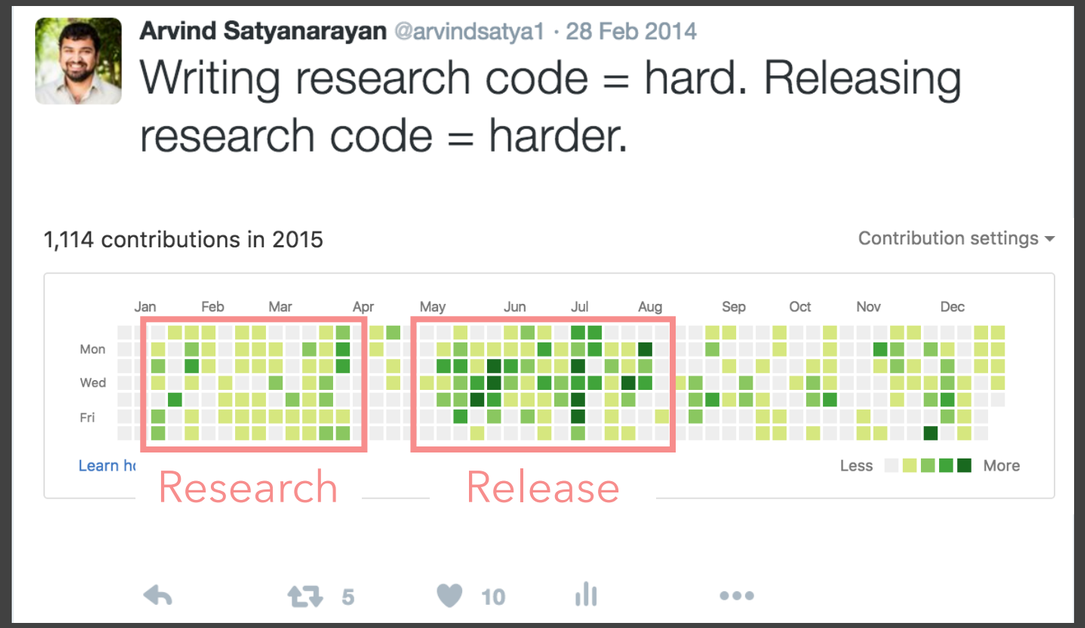

Arvind Satyanarayan @arvindsatya1 · 28 Feb 2014
Writing research code = hard. Releasing research code = harder.
1,114 contributions in 2015
Contribution settings
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
Mon
Wed
Fri
Learn h
Less More
Research
Release
5
10

Challenges


Arvind Satyanarayan @arvindsatya1 · 28 Feb 2014
Writing research code = hard. Releasing research code = harder.
1,114 contributions in 2015
Contribution settings
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
Mon
Wed
Fri
Learn h
Less More
Research
Release
5
10

Challenges

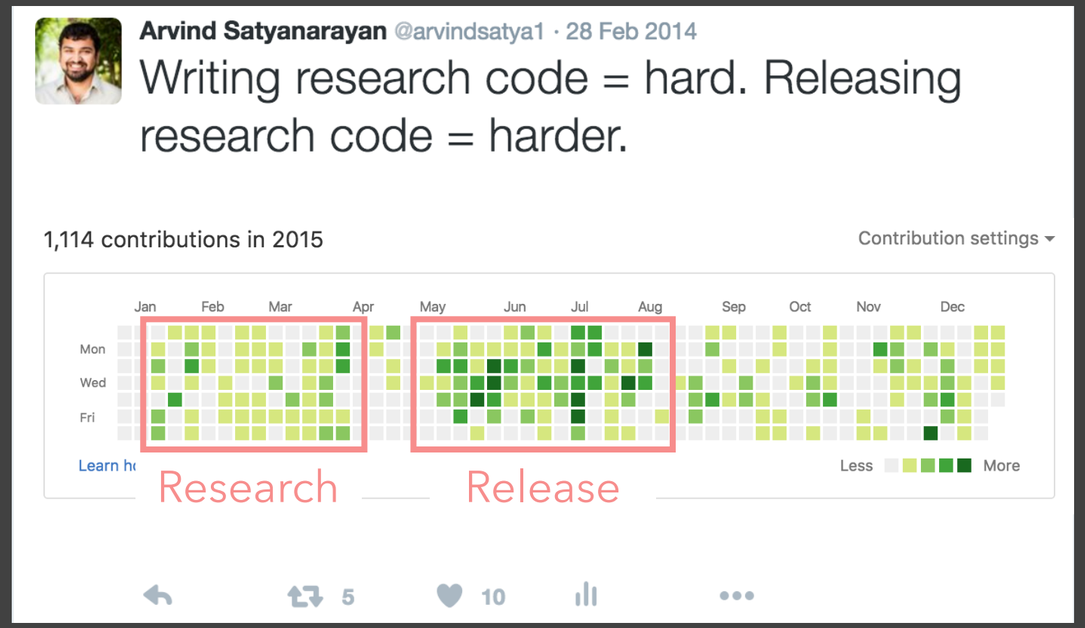
Technical debt builds up with research.



Challenges

Technical debt builds up with research.

Hacking for "proof-of-concept" deadline.

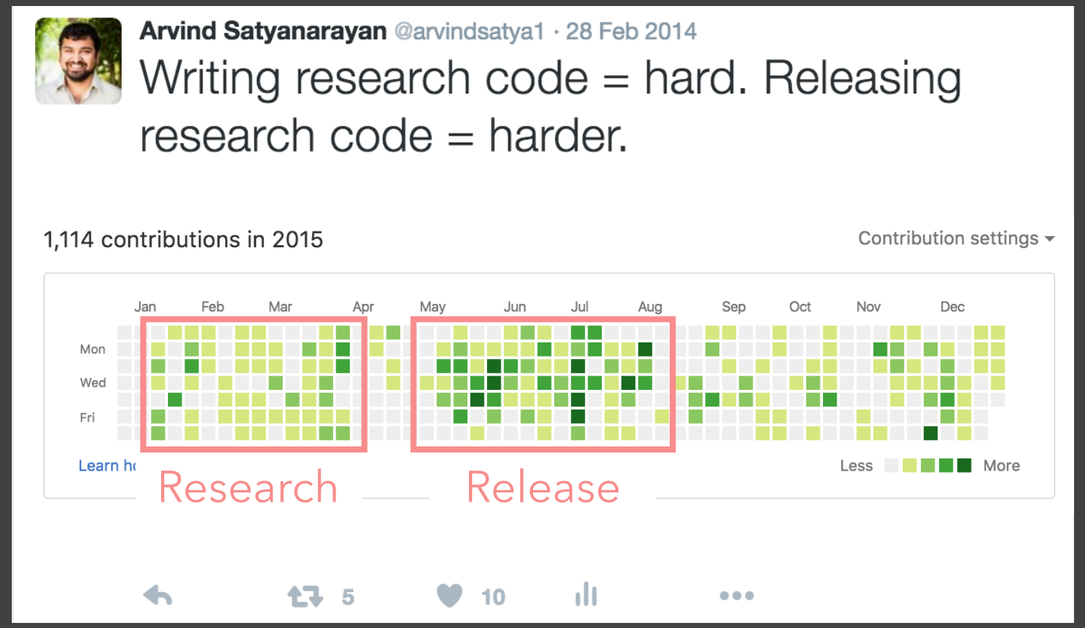


Challenges

Technical debt builds up with research.

Hacking for "proof-of-concept" deadline.

How do we get where we want to go?



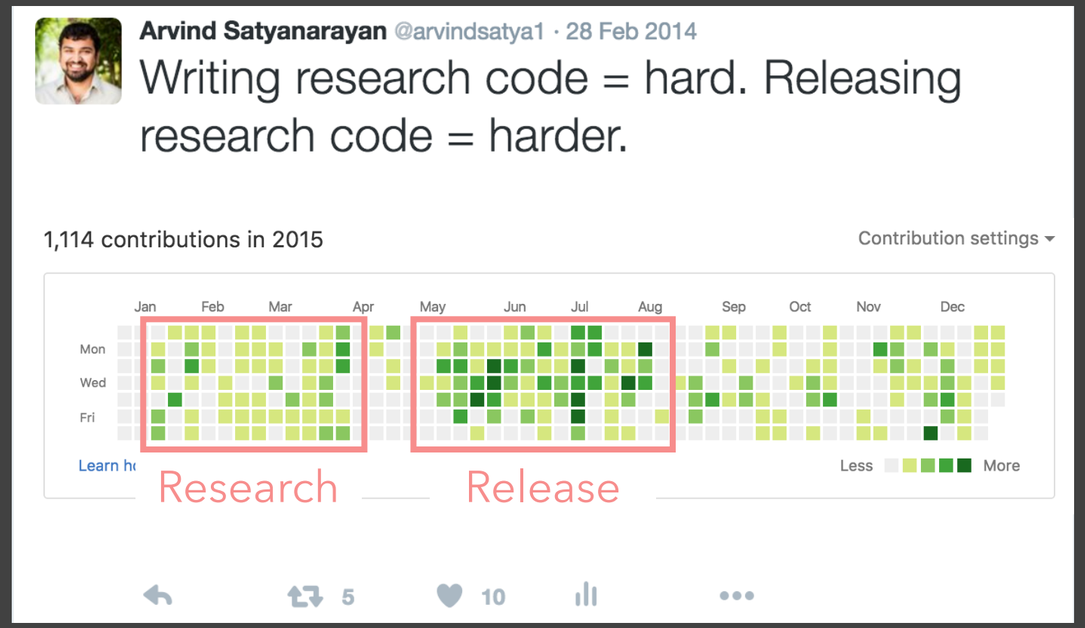
Challenges

Technical debt builds up with research.

Hacking for "proof-of-concept" deadline.

How do we get where we want to go?

Eat your own dog food.



Challenges

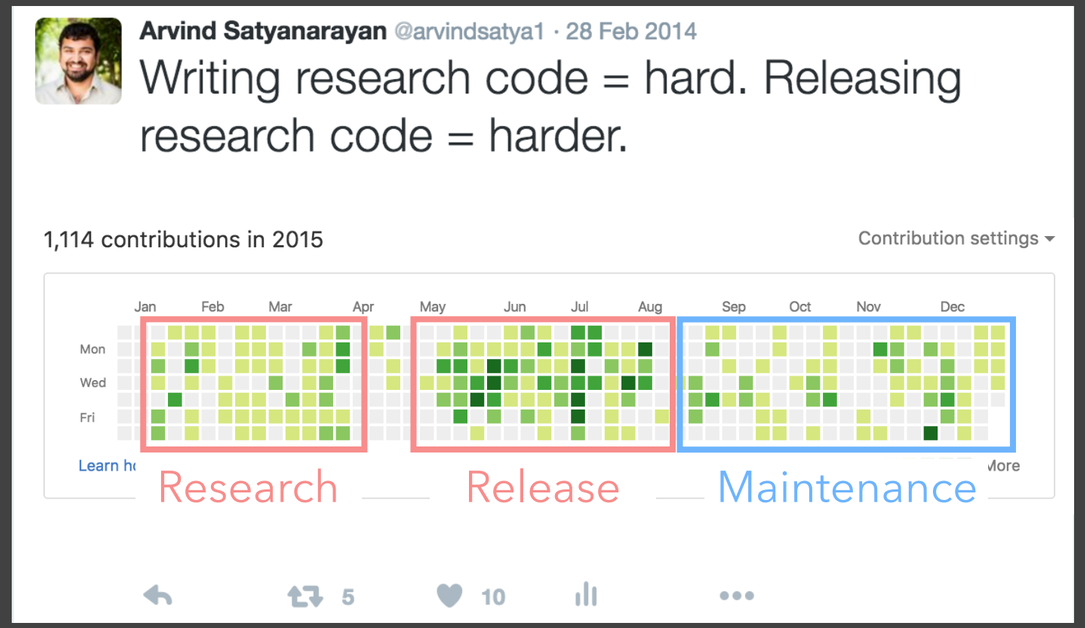
Technical debt builds up with research.

Hacking for "proof-of-concept" deadline.

How do we get where we want to go?

Eat your own dog food.

Maintenance commitment.



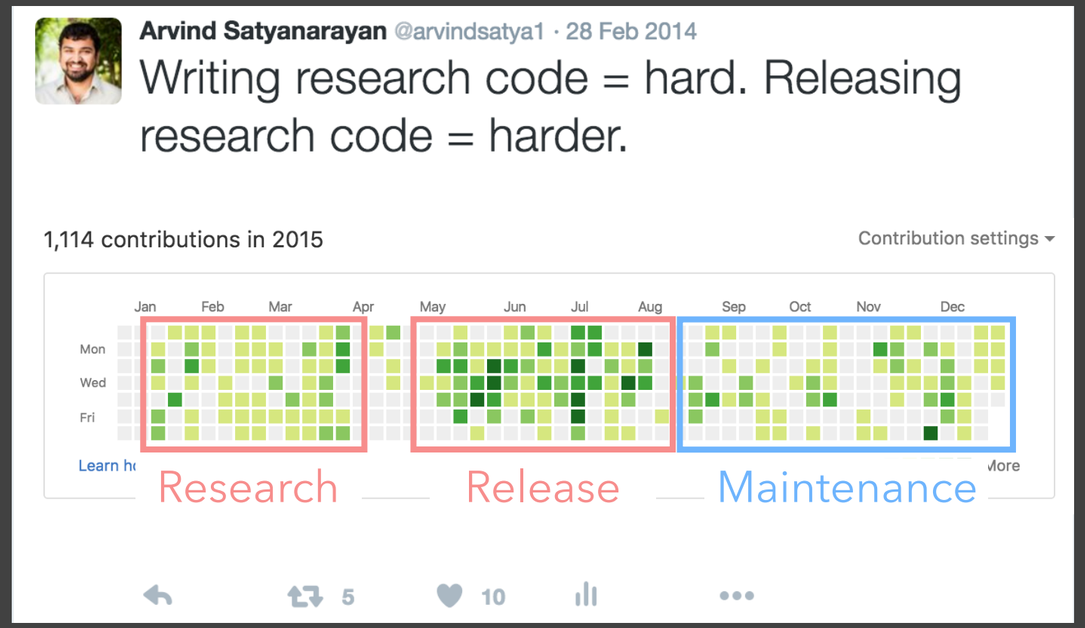
Challenges

Technical debt builds up with research.

- Hacking for "proof-of-concept" deadline.
- How do we get where we want to go?
- Eat your own dog food.

Maintenance commitment.

- Building out support infrastructure (e.g., docs).



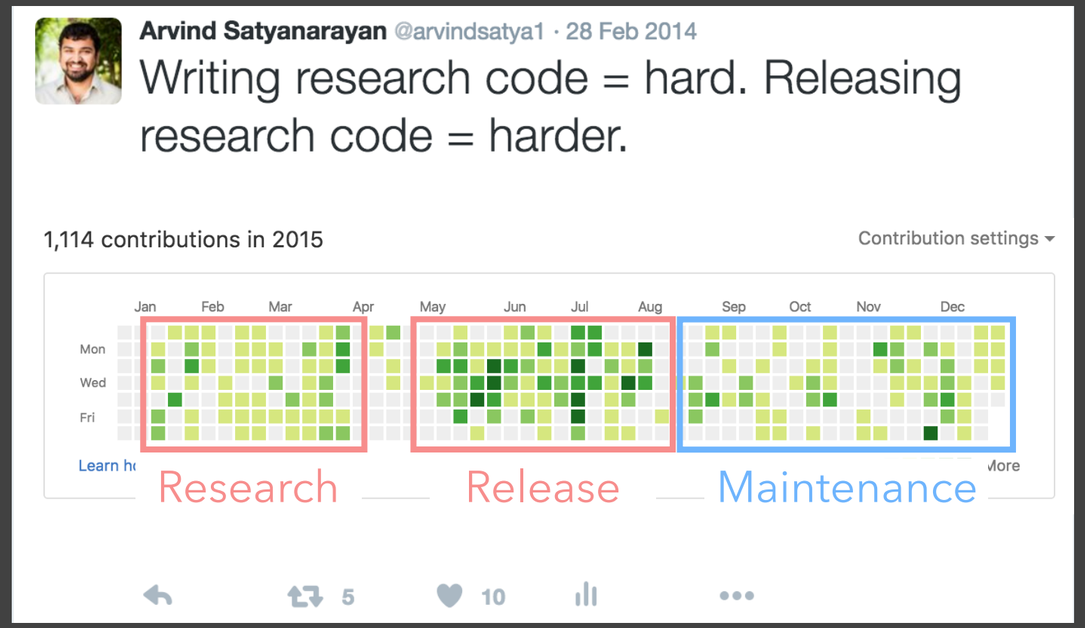
Challenges

Technical debt builds up with research.

- Hacking for "proof-of-concept" deadline.
- How do we get where we want to go?
- Eat your own dog food.

Maintenance commitment.

- Building out support infrastructure (e.g., docs).
- Not only bug fixes, but feature releases to demonstrate active + healthy projects.



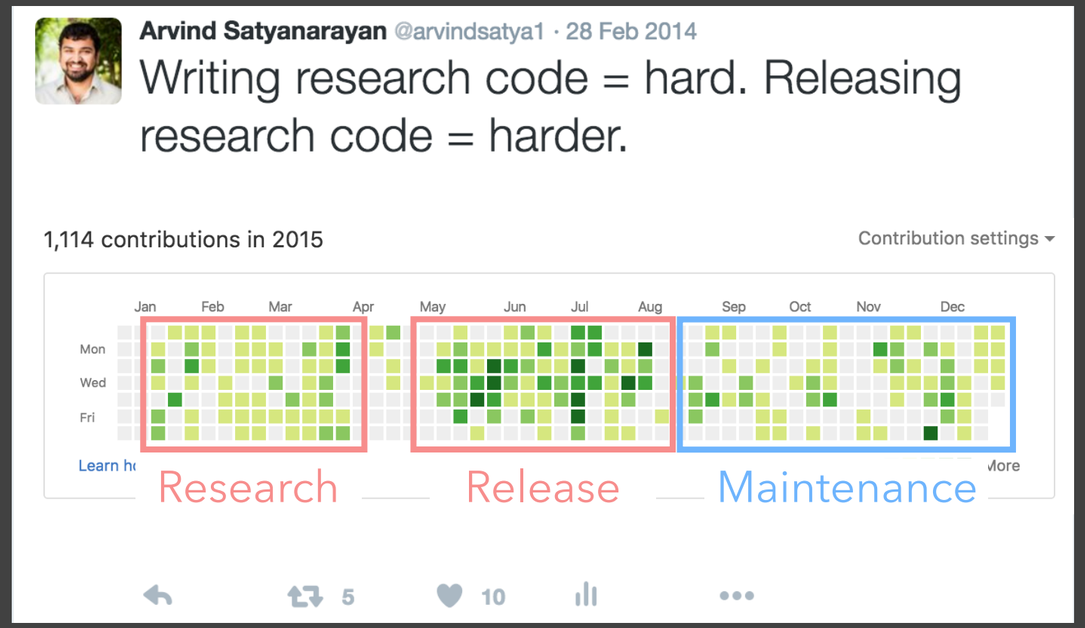
Challenges

Technical debt builds up with research.

- Hacking for "proof-of-concept" deadline.
- How do we get where we want to go?
- Eat your own dog food.

Maintenance commitment.

- Building out support infrastructure (e.g., docs).
- Not only bug fixes, but feature releases to demonstrate active + healthy projects.



Challenges

Technical debt builds up with research.

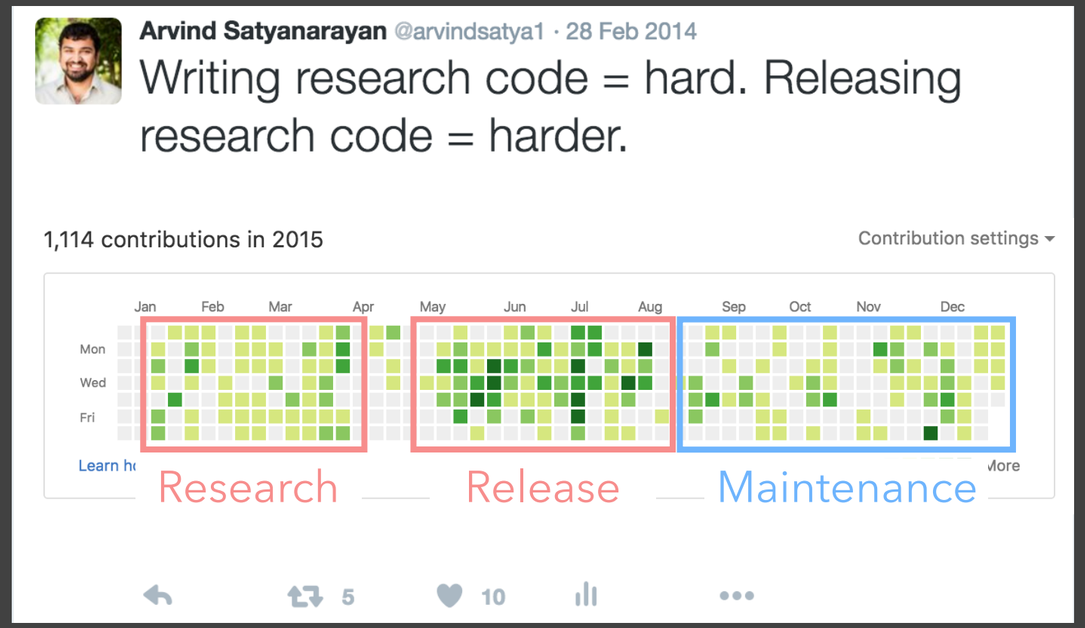
- Hacking for "proof-of-concept" deadline.
- How do we get where we want to go?
- Eat your own dog food.

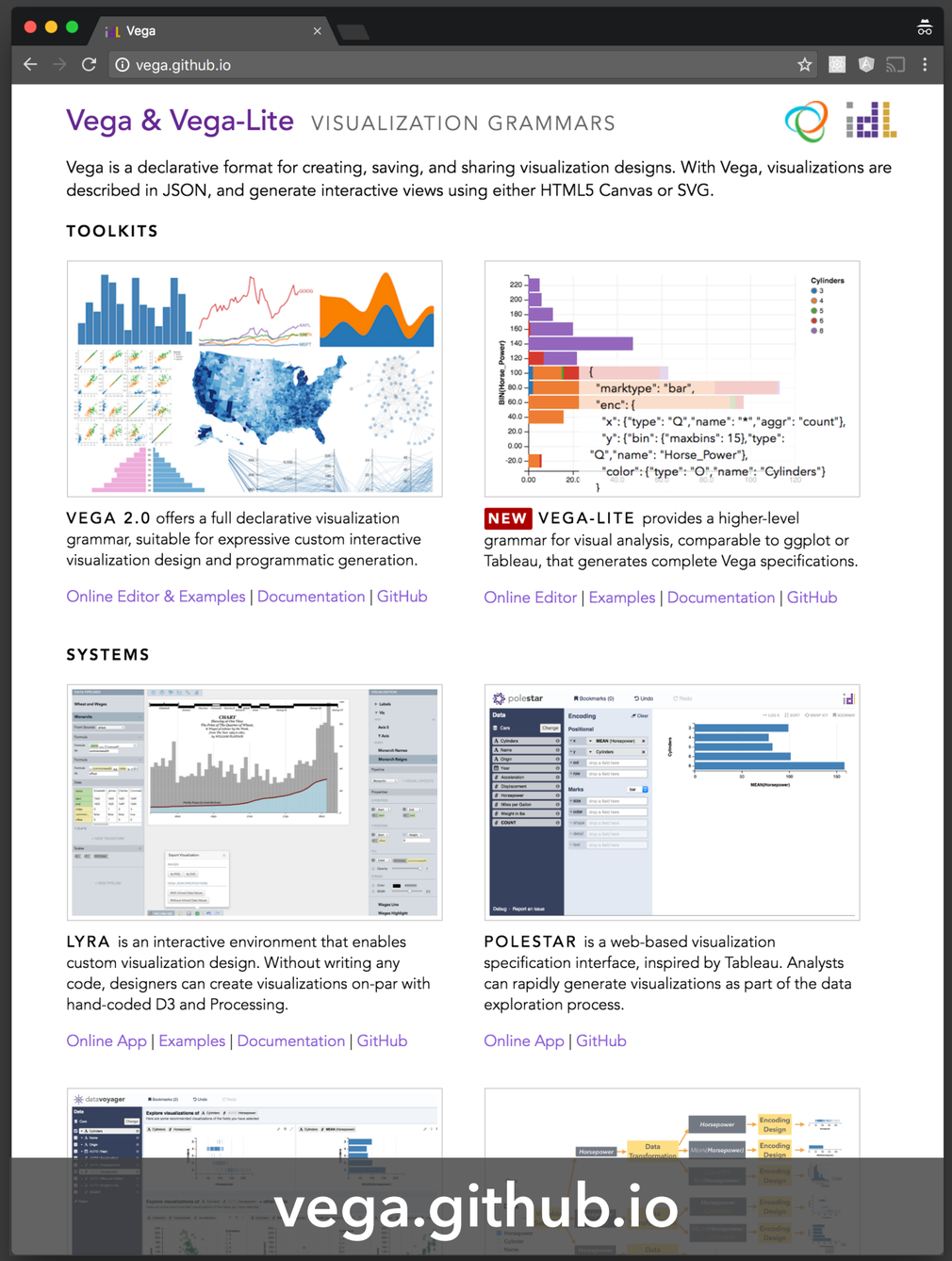
Maintenance commitment.

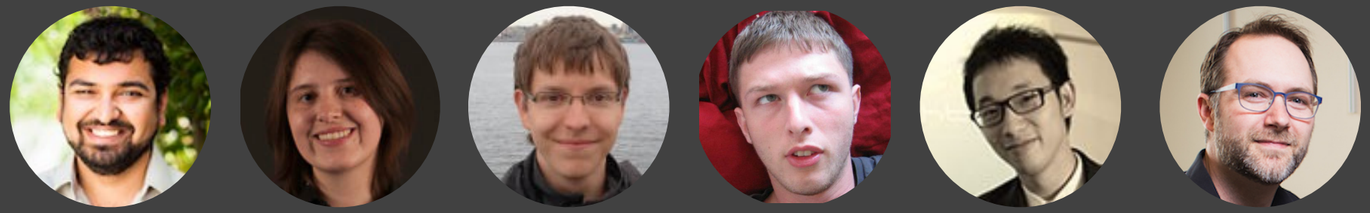
- Building out support infrastructure (e.g., docs).
- Not only bug fixes, but feature releases to demonstrate active + healthy projects.

Developer experience.

- How to integrate Vega into your own system?
- How to contribute?







Arvind Satyanarayan @arvindsatya1
Stanford University

Jane Hoffswell, Dominik Moritz @domoritz, Ryan Russell,
Kanit Wongsuphasawat @kanitw, Jeffrey Heer @jeffrey_heer
University of Washington

ISTC BIG DATA
DARPA
GORDON AND BETTY MOORE FOUNDATION