Reactive Vega

A Streaming Dataflow Architecture for Declarative Interactive Visualization

Arvind Satyanarayan, Ryan Russell, Jane Hoffswell, Jeffrey Heer

Presented by Zipeng Liu
Dec 3 2015
CSPSC 547 Information Visualization

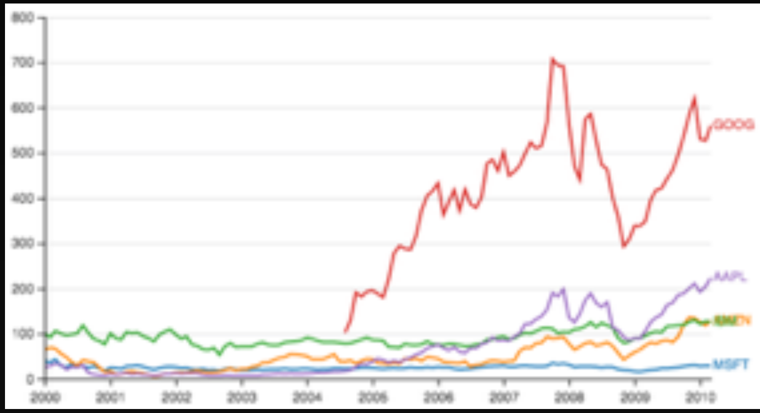
Based on Arvind's slides presented in InfoVis'15. http://arvindsatya.com/slides/ReactiveVega-InfoVis2015.pdf

Reactive Vega

“Talk is cheap. Show me the code”

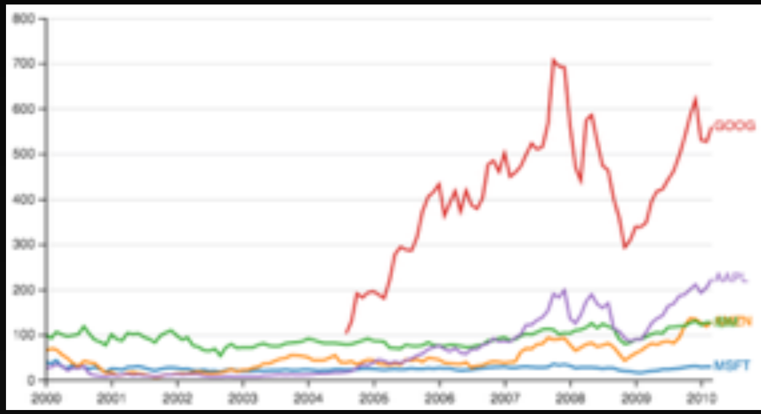
–Linus Torvalds

```
{
  "width": 650, "height": 300,
  "data": [
    {"name": "stocks", "url": “data/stocks.json”}
  ],
  "scales": [{
    "name": "sx", "type": "ordinal",
    "domain": {"data": "stocks", "field": "date"}
    "range": "width"
  }, ... ],
  "axes": [
     {"type": "x", "scale": "sx"}, ...
  ],
  "marks": [{
    "type": "group",
    "from": {
        "data": "stocks",
        "transform": [
          {"type": "facet", "groupby": ["symbol"]}
        ]
    },
    "marks": [{
      "type": "line",
      "properties": { "enter": {
        "x": {"scale": "sx", "field": "date"},
        "y": {"scale": "sy", "field": "price"},
        "stroke": {"scale": "sc", "field": "symbol"}
      }}
    }, {
        "type": "text",... }]
```



Data + Transforms

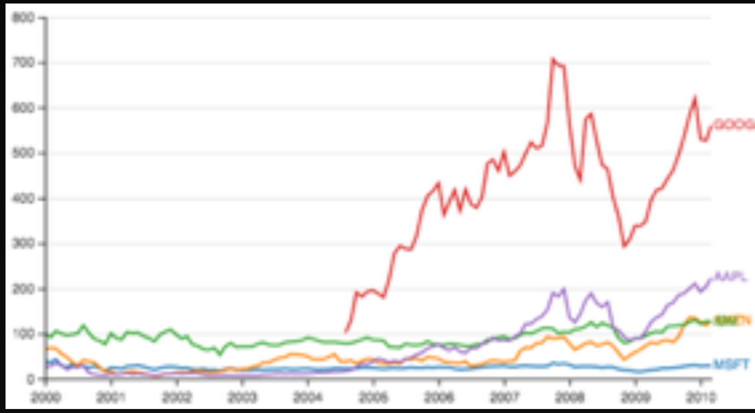
```
{
  "width": 650, "height": 300,
  "data": [
    {"name": "stocks", "url": “data/stocks.json”}
  ],
  "scales": [{
    "name": "sx", "type": "ordinal",
    "domain": {"data": "stocks", "field": "date"}
    "range": "width"
  }, ... ],
  "axes": [
     {"type": "x", "scale": "sx"}, ...
  ],
  "marks": [{
    "type": "group",
    "from": {
        "data": "stocks",
        "transform": [
          {"type": "facet", "groupby": ["symbol"]}
        ]
    },
    "marks": [{
      "type": "line",
      "properties": { "enter": {
        "x": {"scale": "sx", "field": "date"},
        "y": {"scale": "sy", "field": "price"},
        "stroke": {"scale": "sc", "field": "symbol"}
      }}
    }, {
        "type": "text",... }]
```



```
{
  "width": 650, "height": 300,
  "data": [
    {"name": "stocks", "url": “data/stocks.json”}
  ],
  "scales": [{
    "name": "sx", "type": "ordinal",
    "domain": {"data": "stocks", "field": "date"}
    "range": "width"
  }, ... ],
  "axes": [
     {"type": "x", "scale": "sx"}, ...
  ],
  "marks": [{
    "type": "group",
    "from": {
        "data": "stocks",
        "transform": [
          {"type": "facet", "groupby": ["symbol"]}
        ]
    },
    "marks": [{
      "type": "line",
      "properties": { "enter": {
        "x": {"scale": "sx", "field": "date"},
        "y": {"scale": "sy", "field": "price"},
        "stroke": {"scale": "sc", "field": "symbol"}
      }}
    }, {
        "type": "text",... }]
```

Data + Transforms

Scales

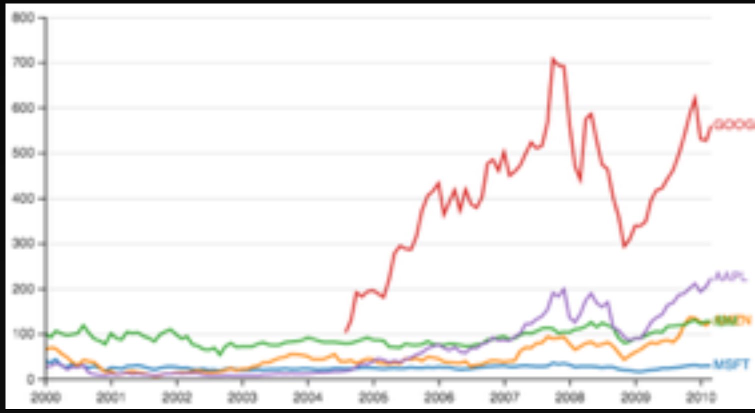


```
{
  "width": 650, "height": 300,
  "data": [
    {"name": "stocks", "url": “data/stocks.json”}
  ],
  "scales": [{
    "name": "sx", "type": "ordinal",
    "domain": {"data": "stocks", "field": "date"}
    "range": "width"
  }, ... ],
  "axes": [
     {"type": "x", "scale": "sx"}, ...
  ],
  "marks": [{
    "type": "group",
    "from": {
        "data": "stocks",
        "transform": [
          {"type": "facet", "groupby": ["symbol"]}
        ]
    },
    "marks": [{
      "type": "line",
      "properties": { "enter": {
        "x": {"scale": "sx", "field": "date"},
        "y": {"scale": "sy", "field": "price"},
        "stroke": {"scale": "sc", "field": "symbol"}
      }}
    }, {
        "type": "text",... }]
```

Data + Transforms

Scales

Guides

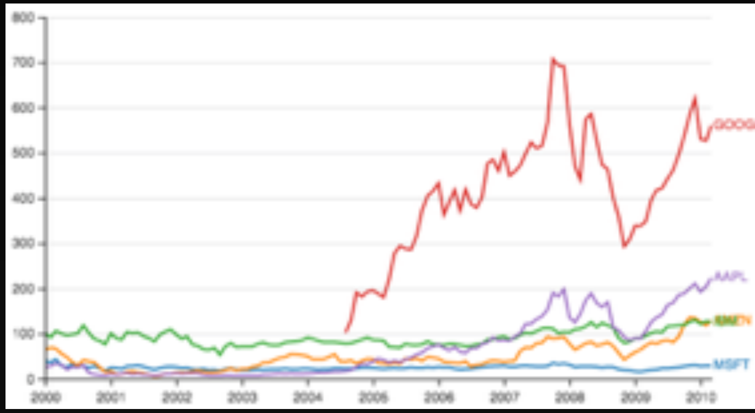


```
{
  "width": 650, "height": 300,
  "data": [
    {"name": "stocks", "url": “data/stocks.json”}
  ],
  "scales": [{
    "name": "sx", "type": "ordinal",
    "domain": {"data": "stocks", "field": "date"}
    "range": "width"
  }, ... ],
  "axes": [
     {"type": "x", "scale": "sx"}, ...
  ],
  "marks": [{
    "type": "group",
    "from": {
       "data": "stocks",
       "transform": [
         {"type": "facet", "groupby": ["symbol"]}
       ]
    },
    "marks": [{
      "type": "line",
      "properties": { "enter": {
        "x": {"scale": "sx", "field": "date"},
        "y": {"scale": "sy", "field": "price"},
        "stroke": {"scale": "sc", "field": "symbol"}
      }}
    }, {
       "type": "text",... }]
```

Data +
Transforms

Scales

Guides



Marks

Why Declarative

- Less code + faster iteration
- Performance + scalability
- **Reuse + portability (flexibility)**
- **Programmatic generation**

Interaction?

Reactive Vega

Imperative Interaction

```
var dragging = false;
d3.selectAll(“rect")
  .on("mousedown", function() {
     dragging = true;
  })
  .on("mouseup", function() {
     dragging = false;
     d3.event.stopPropagation();
  })
  .on("mousemove", function() {
     var e = d3.event;
     if (!dragging) return;
     d3.select(this)
       .attr(“x”, e.pageX)
       .attr(“y”, e.pageY);
});
```

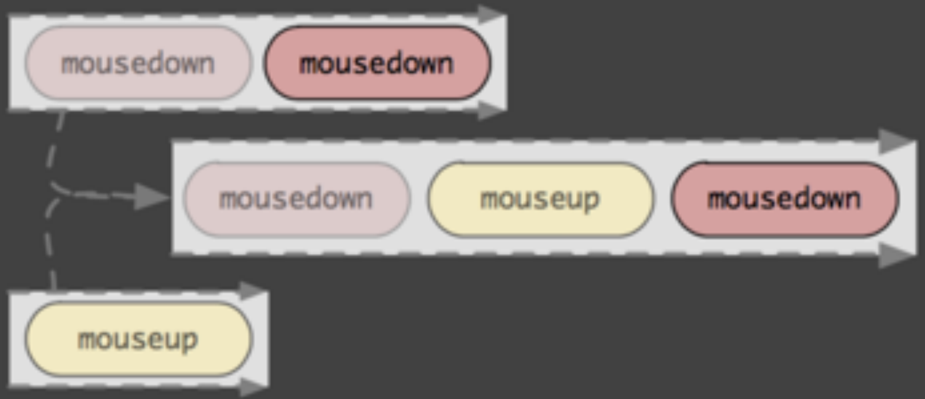
1. Manually maintain state and dependencies
2. Side-effects
3. “Callback hell”

Declarative Interaction

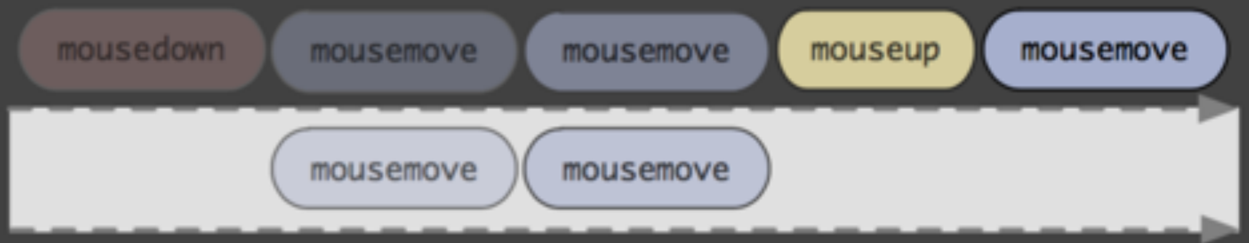
- Event-driven Functional Reactive Programming (E-FRP)
  - mutable values as time-varying **data streams**
  - event triggers **propagation** through **dataflow graph**
  - but only for scalar values
- Streaming Database

Event Streams

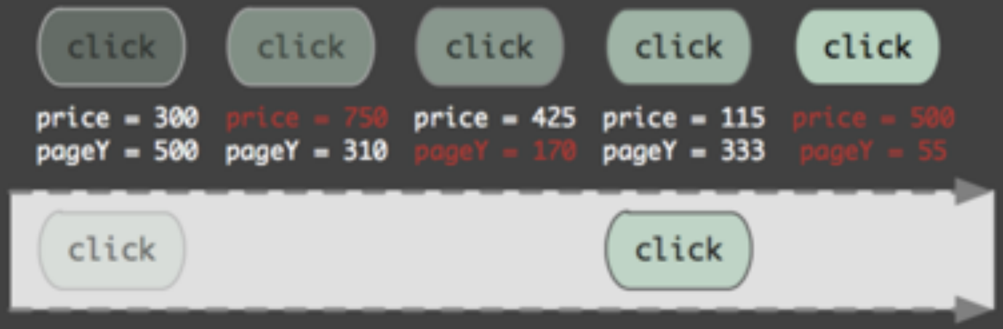
`*:mousedown, *:mouseup`
a single stream merges `mousedown` and `mouseup` streams.



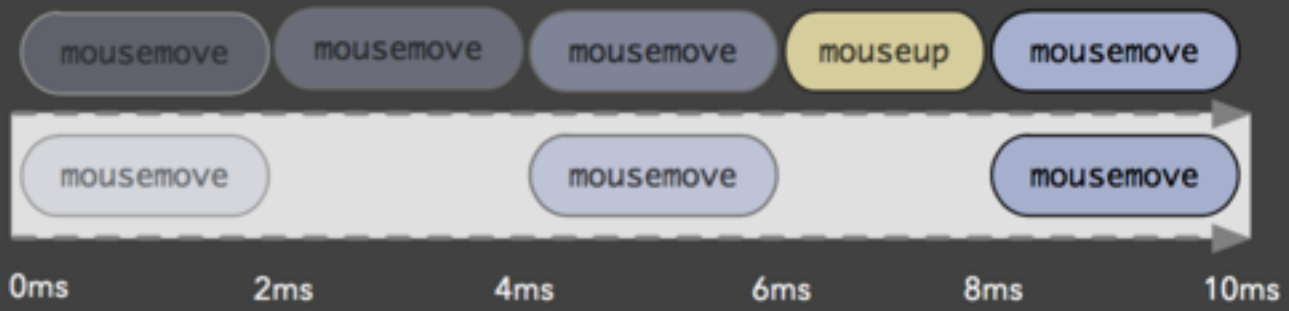
`[*:mousedown, *:mouseup] > *:mousemove`
drag



```
*:click[event.pageY >= 300]
       [data.price < 500]
```
filtered stream of `click` events

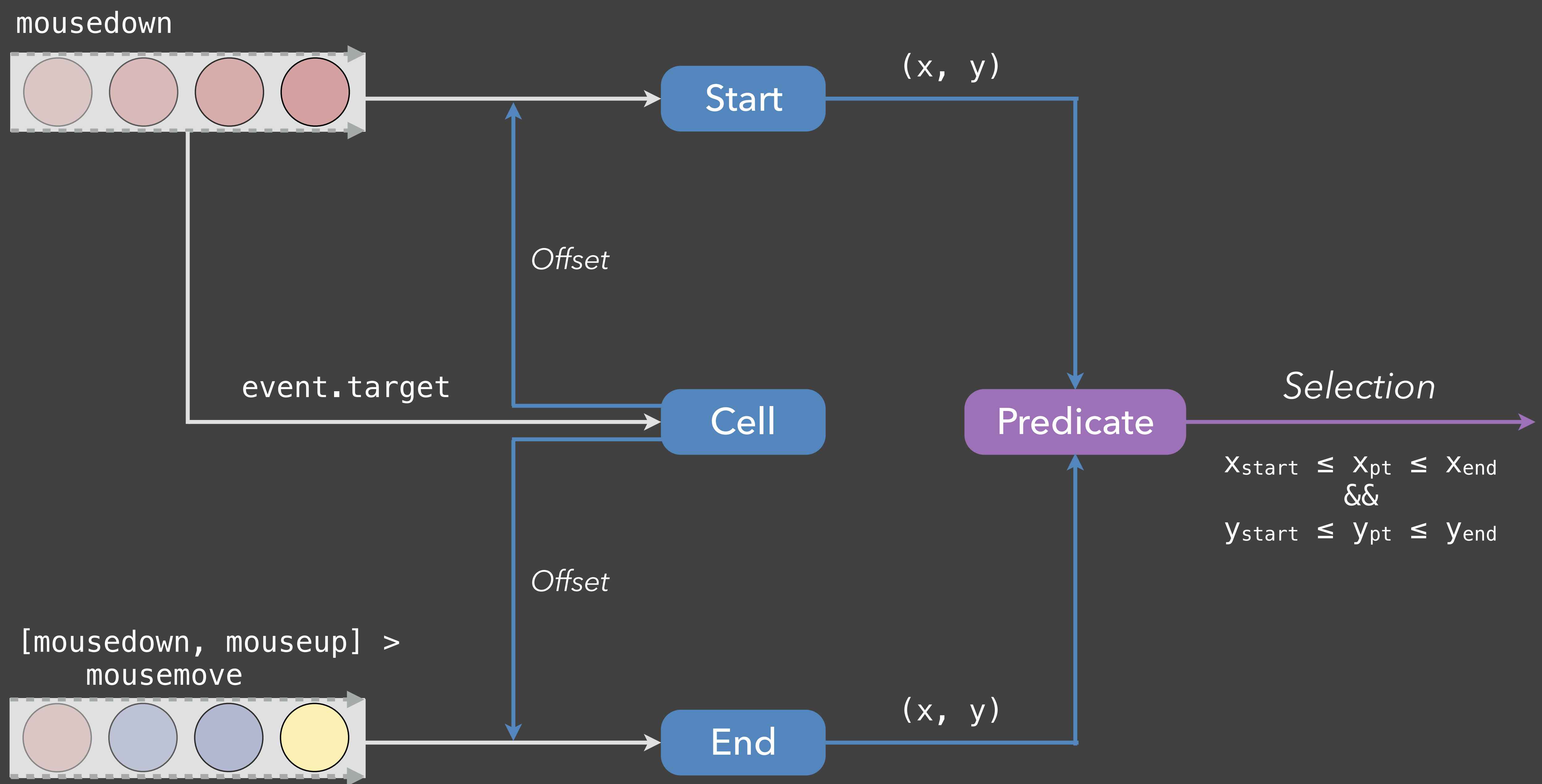


`*:mousemove{3ms,5ms}`
stream of `mousemove` events that occur at least 3ms, and at most 5ms, apart

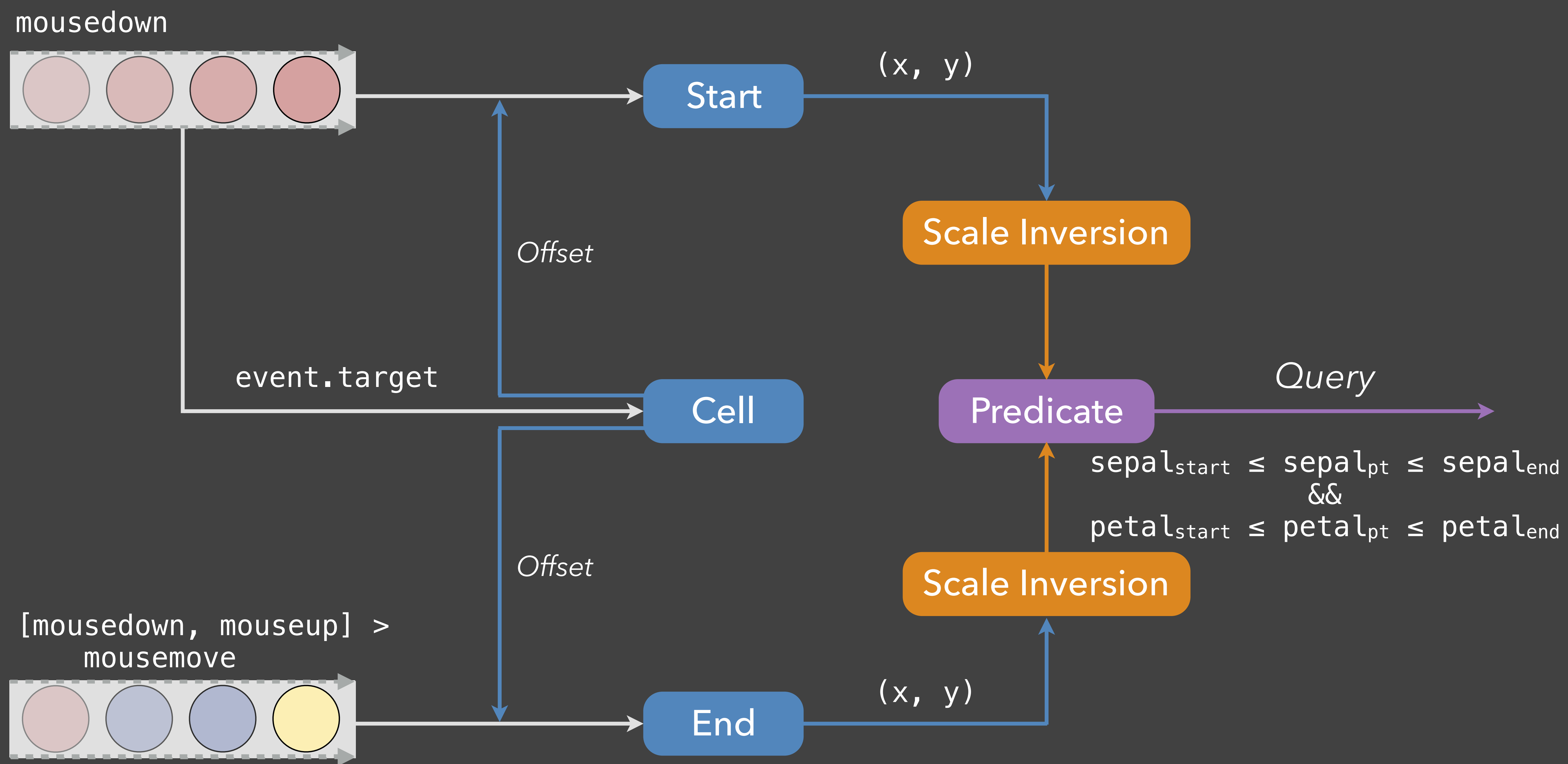


Demo: SPLOM of Iris

http://vega.github.io/vega-editor/index.html?spec=linking

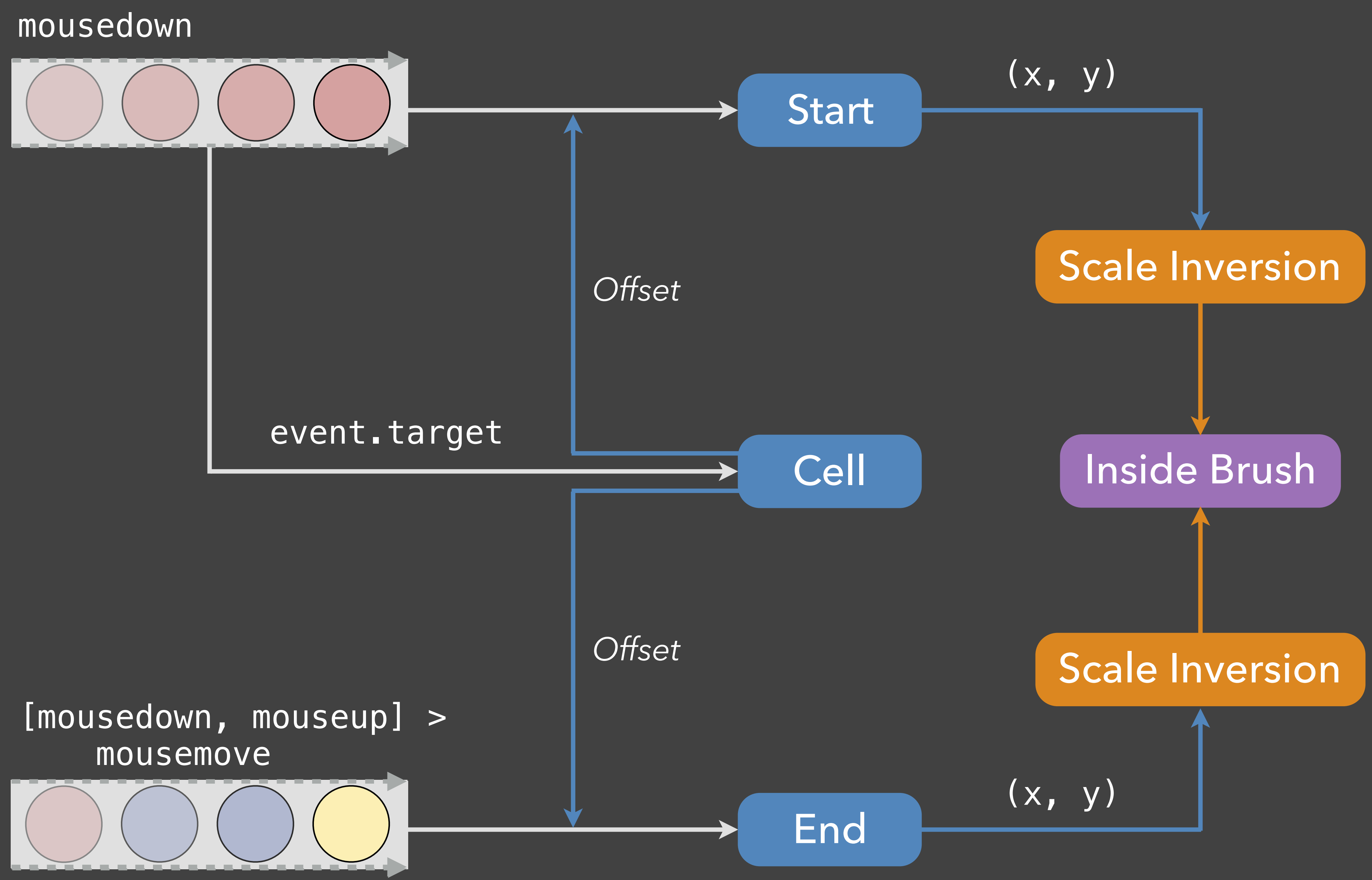


$$x_{start} \leq x_{pt} \leq x_{end}$$
$$\&\&$$
$$y_{start} \leq y_{pt} \leq y_{end}$$

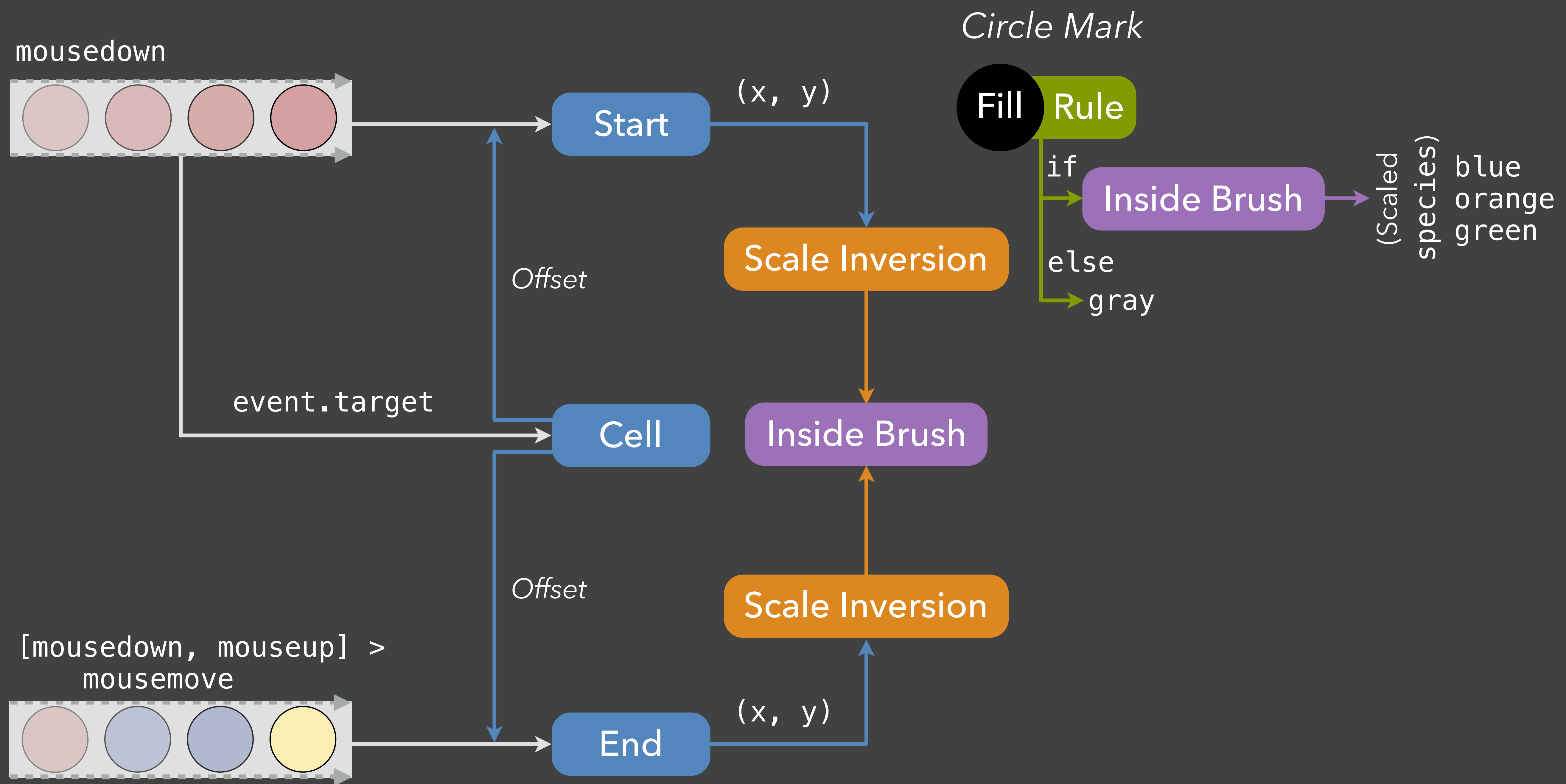


$\mathtt{sepal}_{\mathtt{start}} \le \mathtt{sepal}_{\mathtt{pt}} \le \mathtt{sepal}_{\mathtt{end}}$
&&
$\mathtt{petal}_{\mathtt{start}} \le \mathtt{petal}_{\mathtt{pt}} \le \mathtt{petal}_{\mathtt{end}}$

21


mousedown
Start
(x, y)
Scale Inversion
Offset
event.target
Cell
Inside Brush
Offset
Scale Inversion
[mousedown, mouseup] > mousemove
End
(x, y)

22

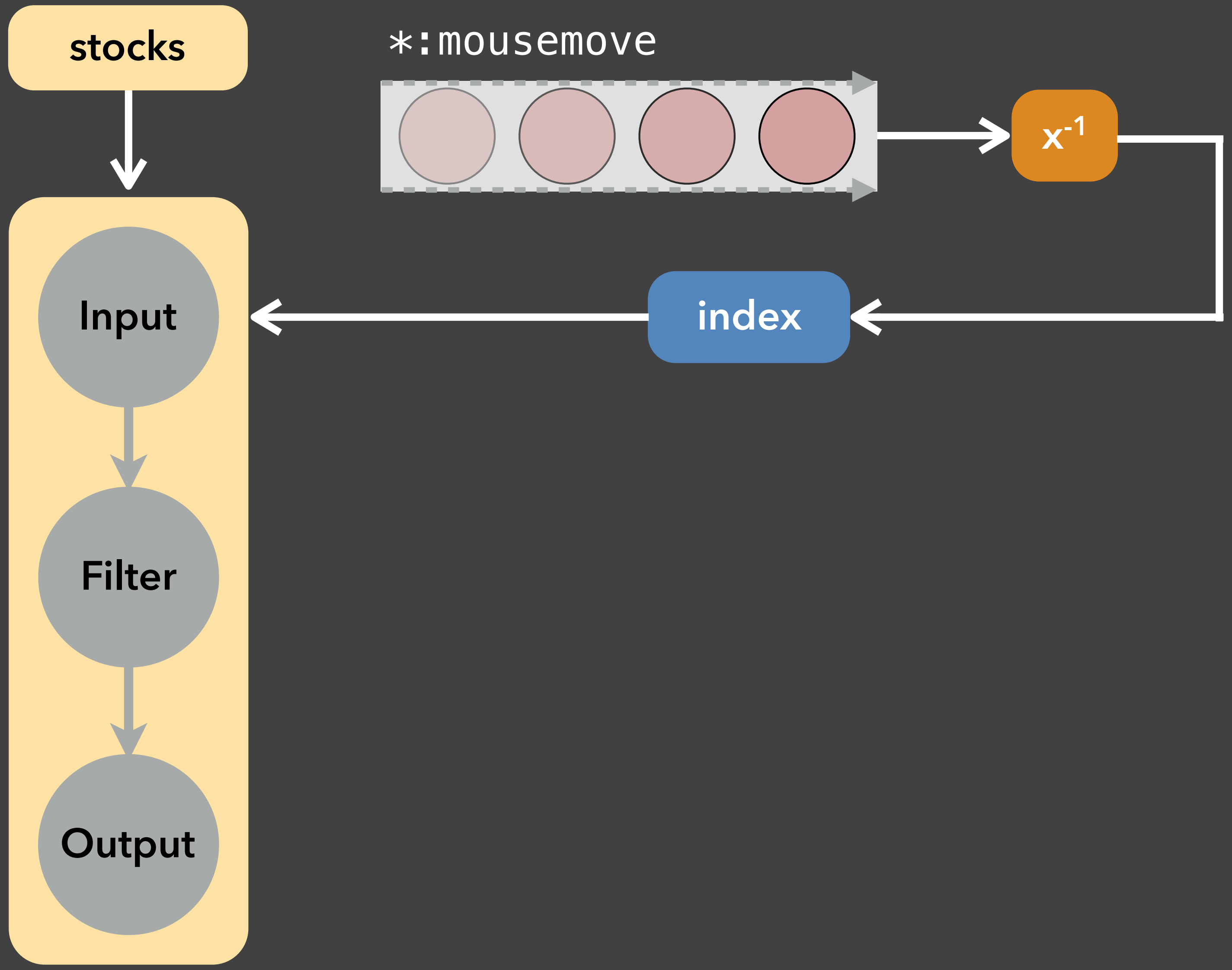

Circle Mark
mousedown
Start
(x, y)
Fill
Rule
if
Inside Brush
(Scaled species)
blue
orange
green
Offset
Scale Inversion
else
gray
event.target
Cell
Inside Brush
Offset
Scale Inversion
[mousedown, mouseup] > mousemove
(x, y)
End

Architecture: Dataflow Graph

optional

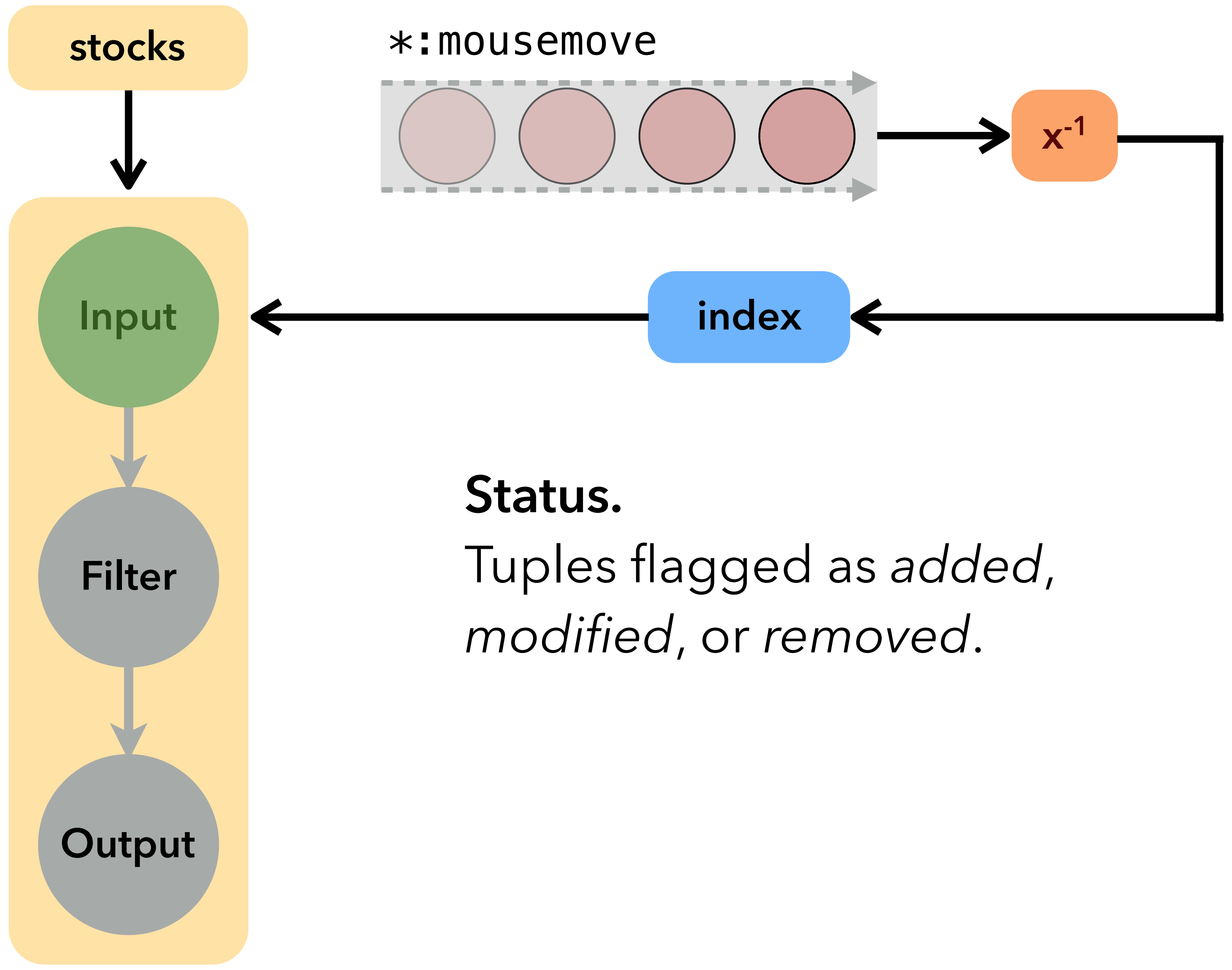
Compile Time

```
{
  "data": [
    {...},
    {
      "name": "index_pts",
      "source": "stocks",
      "transform": [{
        "type": "filter",
        "test": "month(datum) ==
month(index) && year(datum) ==
year(index)"
      }]
    },
    {...}
  ]
}
```



Run Time

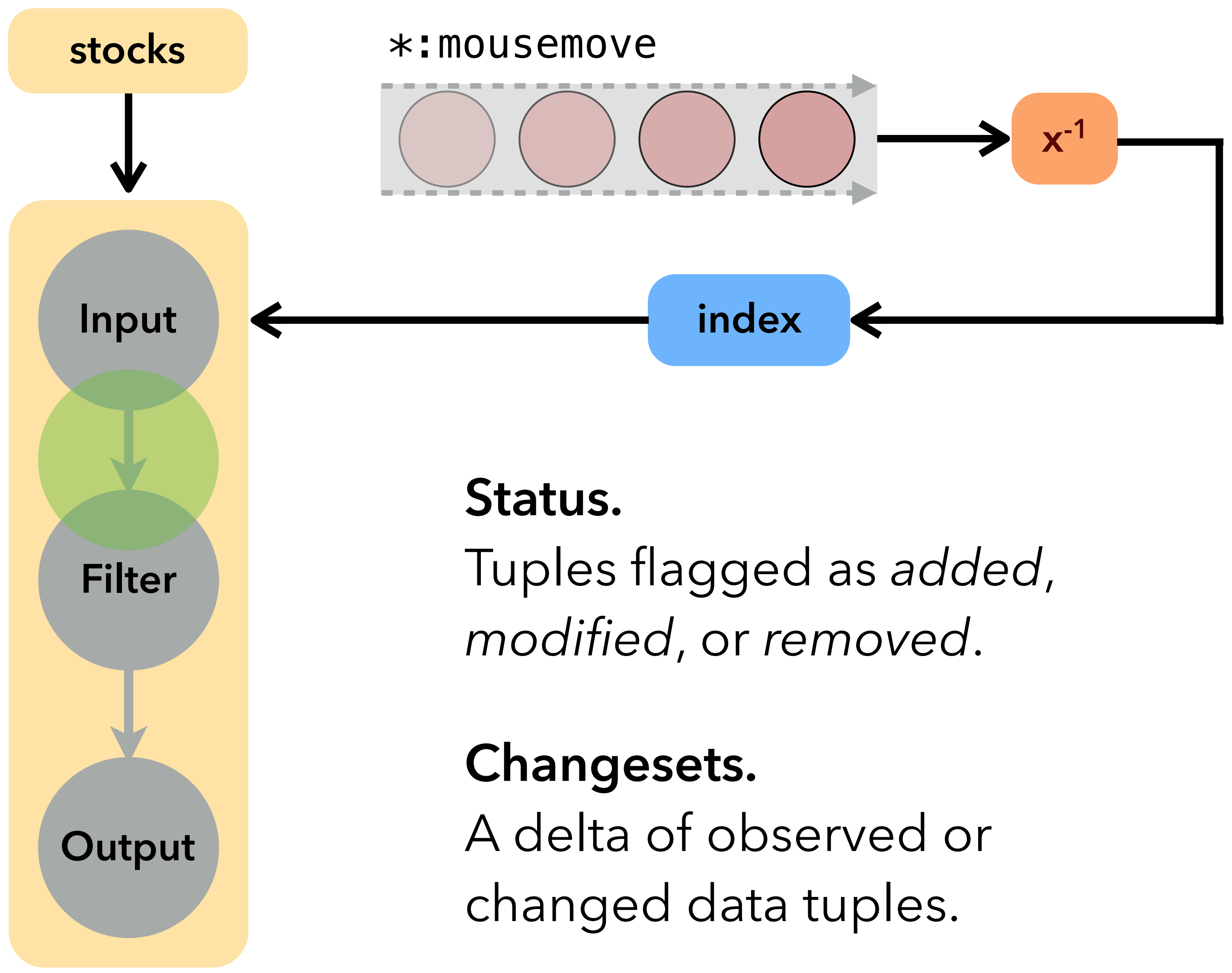
```
{
  "data": [
    {...},
    {...
      "name": "index_pts",
      "source": "stocks",
      "transform": [{
        "type": "filter",
        "test": "month(datum) ==
month(index) && year(datum) ==
year(index)"
      }]
    },
    {...}
  ]
}
```



Status.
Tuples flagged as *added*, *modified*, or *removed*.

Run Time

```
{
  "data": [
    {...},
    {...
      "name": "index_pts",
      "source": "stocks",
      "transform": [{
        "type": "filter",
        "test": "month(datum) ==
month(index) && year(datum) ==
year(index)"
      }]
    },
    {...}
  ]
}
```

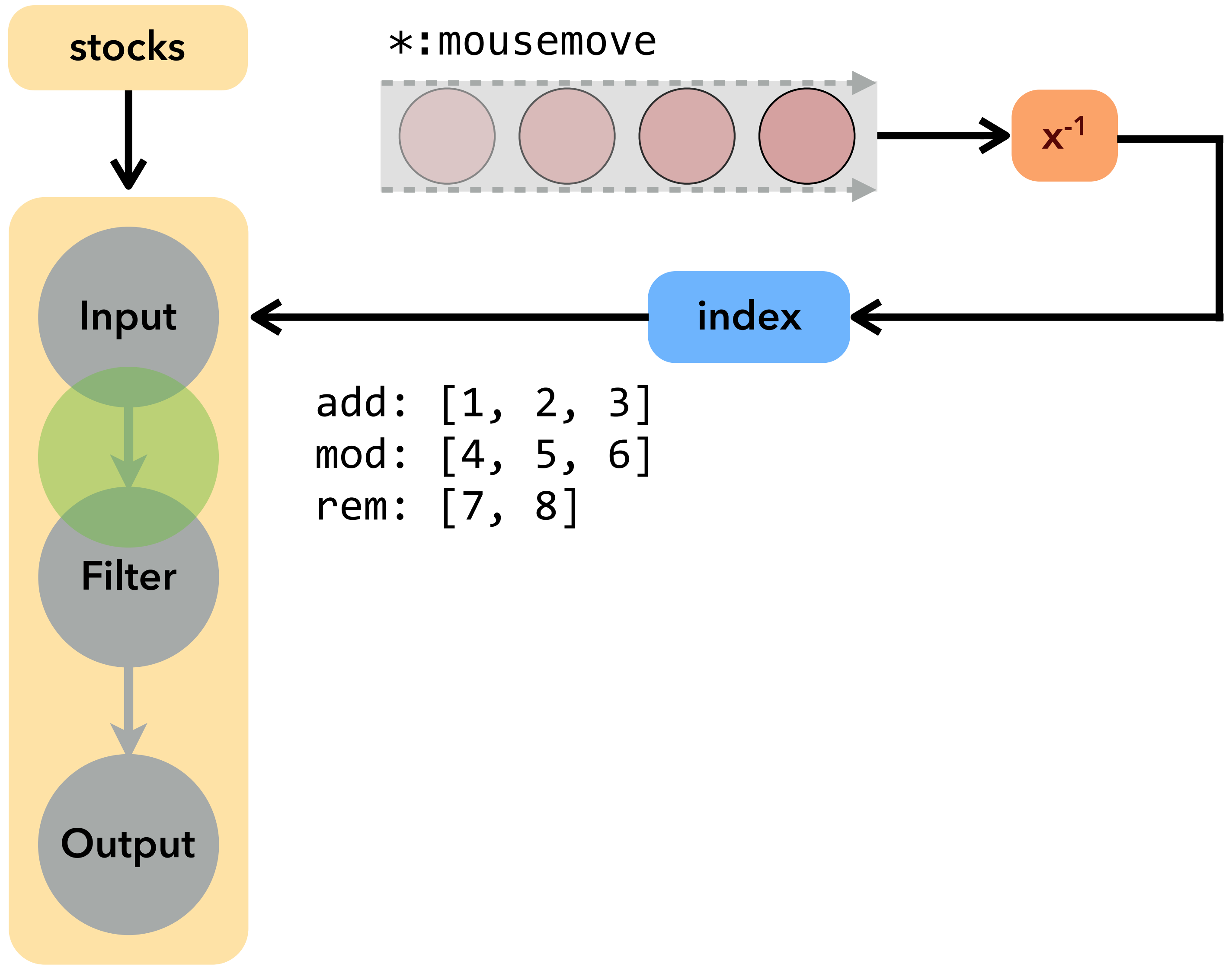


Status.
Tuples flagged as *added*, *modified*, or *removed*.

Changesets.
A delta of observed or changed data tuples.

Run Time

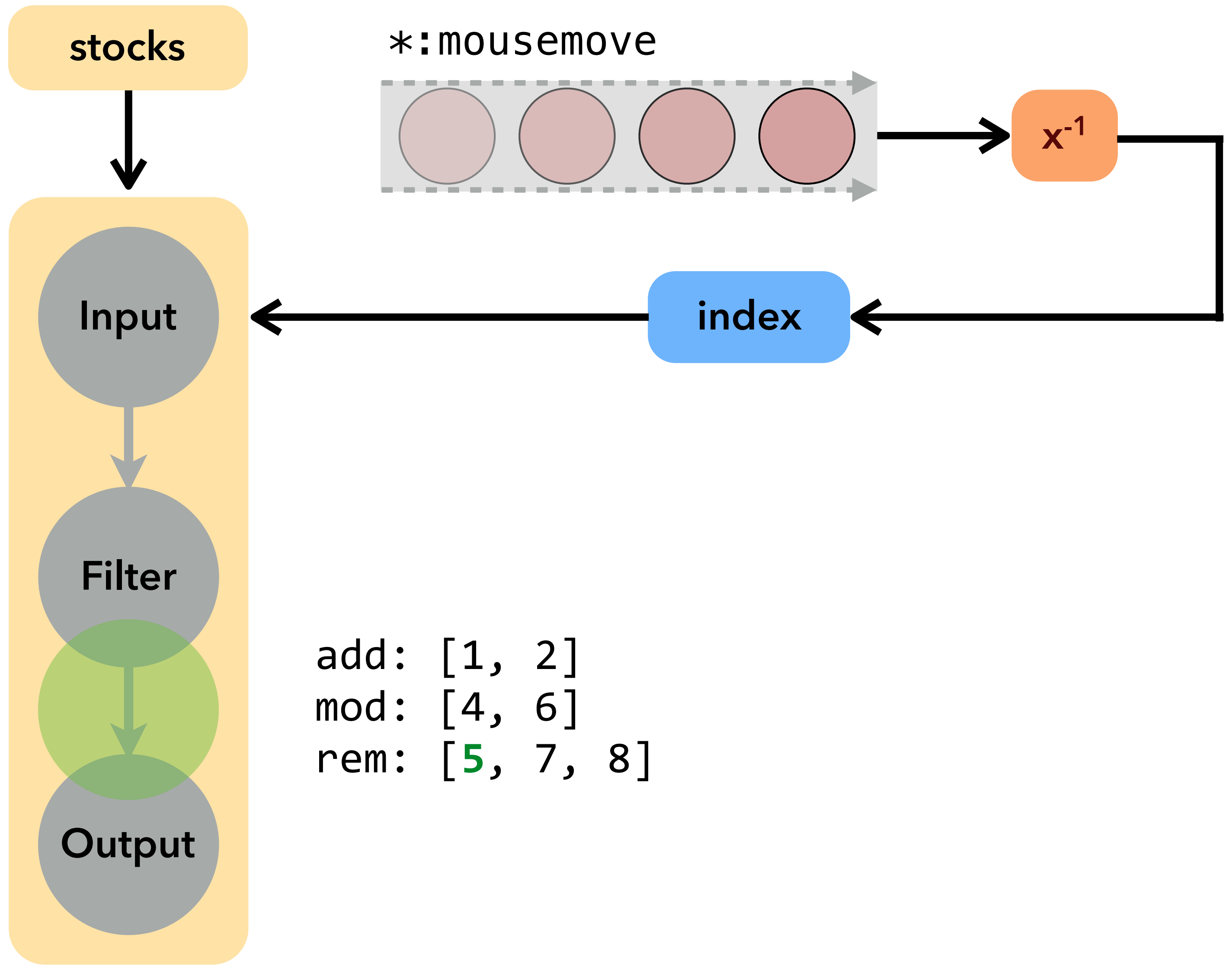
```
{
  "data": [
    {...},
    {...
      "name": "index_pts",
      "source": "stocks",
      "transform": [{
        "type": "filter",
        "test": "month(datum) ==
month(index) && year(datum) ==
year(index)"
      }]
    },
    {...}
  ]
}
```



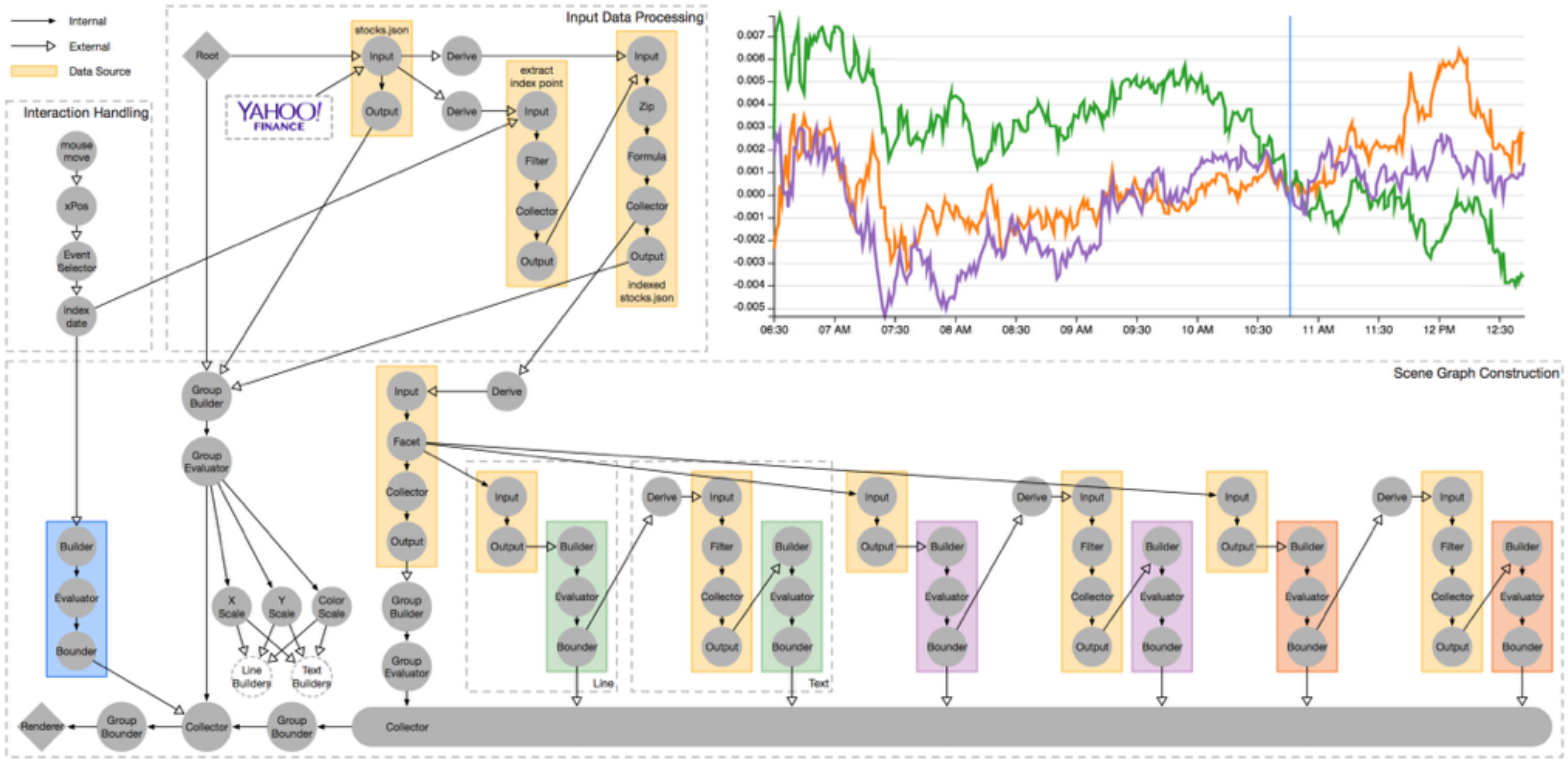
```
add: [1, 2, 3]
mod: [4, 5, 6]
rem: [7, 8]
```

Run Time

```
{
  "data": [
    {...},
    {...
      "name": "index_pts",
      "source": "stocks",
      "transform": [{
        "type": "filter",
        "test": "month(datum) ==
month(index) && year(datum) ==
year(index)"
      }]
    },
    {...}
  ]
}
```



```
add: [1, 2]
mod: [4, 6]
rem: [5, 7, 8]
```

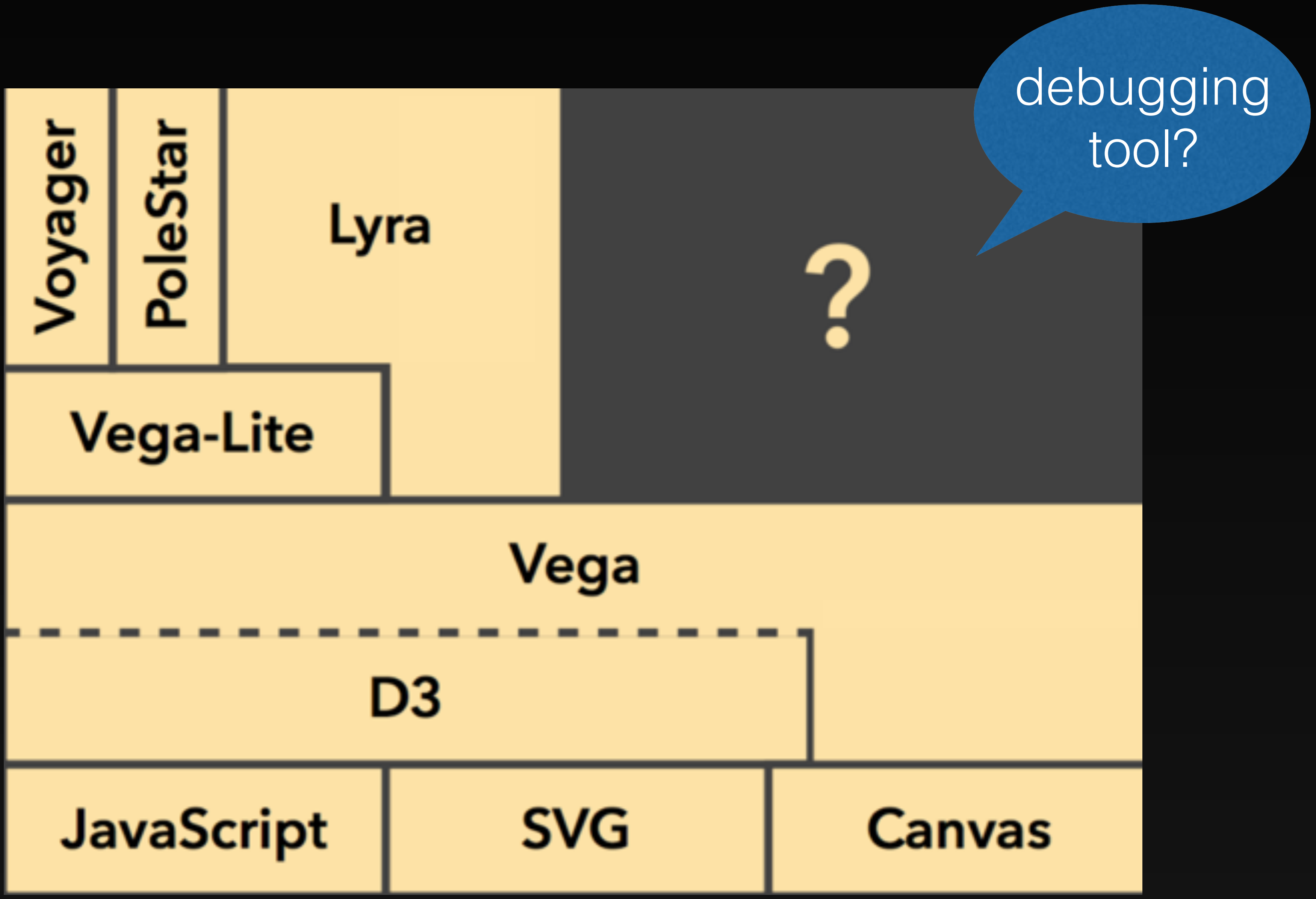


Dataflow graph for index chart

~2x faster than D3

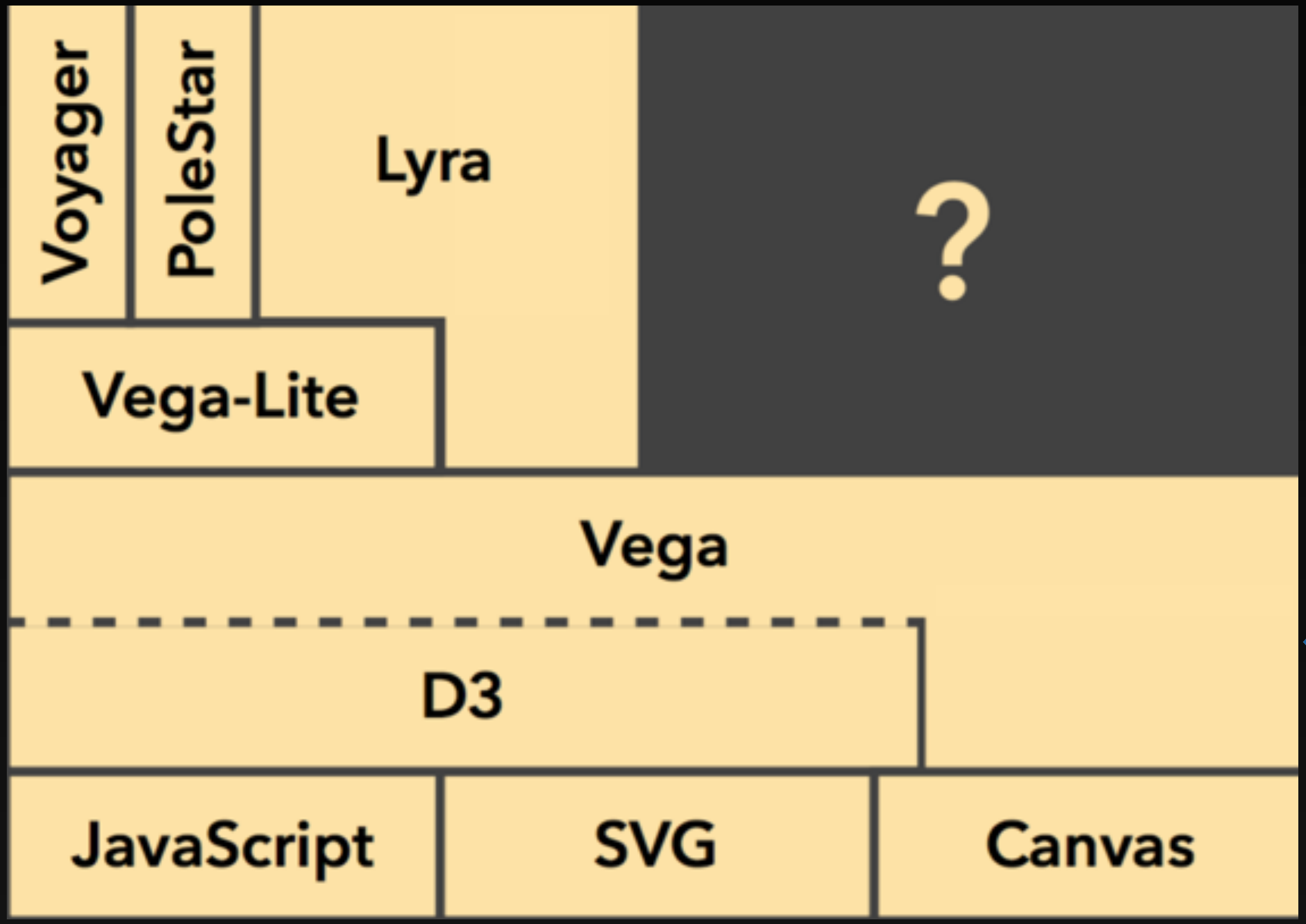
Full benchmark studies in the paper and online:
http://github.com/vega/vega-benchmarks

Future Work



Eco-system

Future Work



Eco-system

Server-side computation

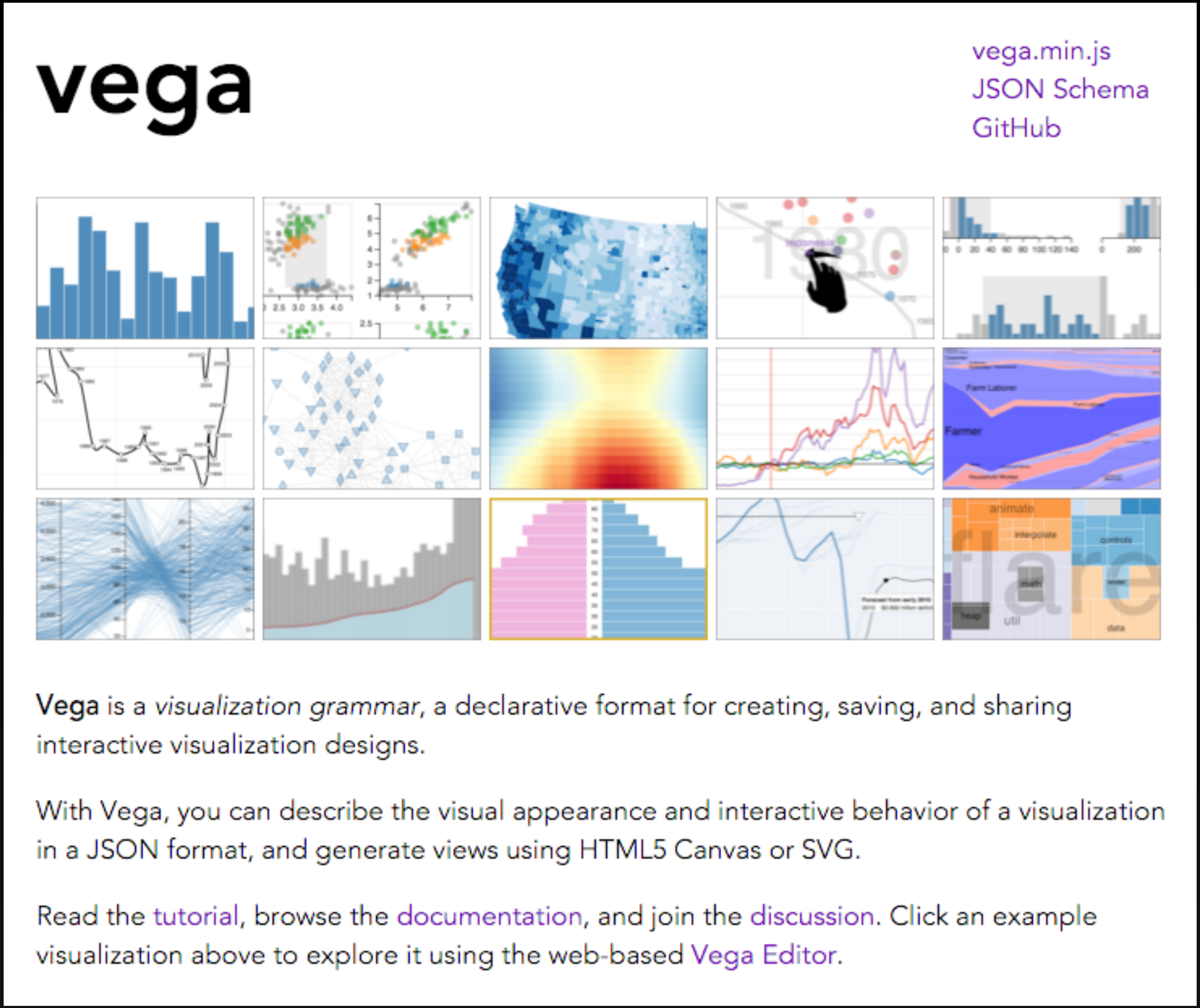
workload partition?

Comments

- Declarative specification rocks
  - reusable, shareable (also iVisDesigner, …)
  - elegant! (once learning curve is climbed)
- E-FRP could be the next hotspot
  - Similar as ReactJS
  - FP also
- Eco-system that speaks Vega
  - but Vega is not enough
- Open source

Comments

- Requires clear and well-ordered data
  - Same as Tableau
- No way to debug
  - Language-level optimisation & runtime evaluation
  - Tradeoff: Cognitive Dimensions of Notation
- Learning curve is quite steep
  - Lack of community
  - Foreign to FRP



vega

vega.min.js
JSON Schema
GitHub

Vega is a *visualization grammar*, a declarative format for creating, saving, and sharing interactive visualization designs.

With Vega, you can describe the visual appearance and interactive behavior of a visualization in a JSON format, and generate views using HTML5 Canvas or SVG.

Read the tutorial, browse the documentation, and join the discussion. Click an example visualization above to explore it using the web-based Vega Editor.

vega.github.io/vega/