

tt()

Schedule

1. Dataset ophalen (show and tell)
2. Review (scales, joins)
3. Events
4. All together!



Schedule

- 1. Dataset ophalen (show and tell)**
2. Review (scales, joins)
3. Events
4. All together!



Schedule

1. Dataset ophalen (show and tell)
- 2. Review (scales, joins)**
3. Events
4. All together!



Schedule

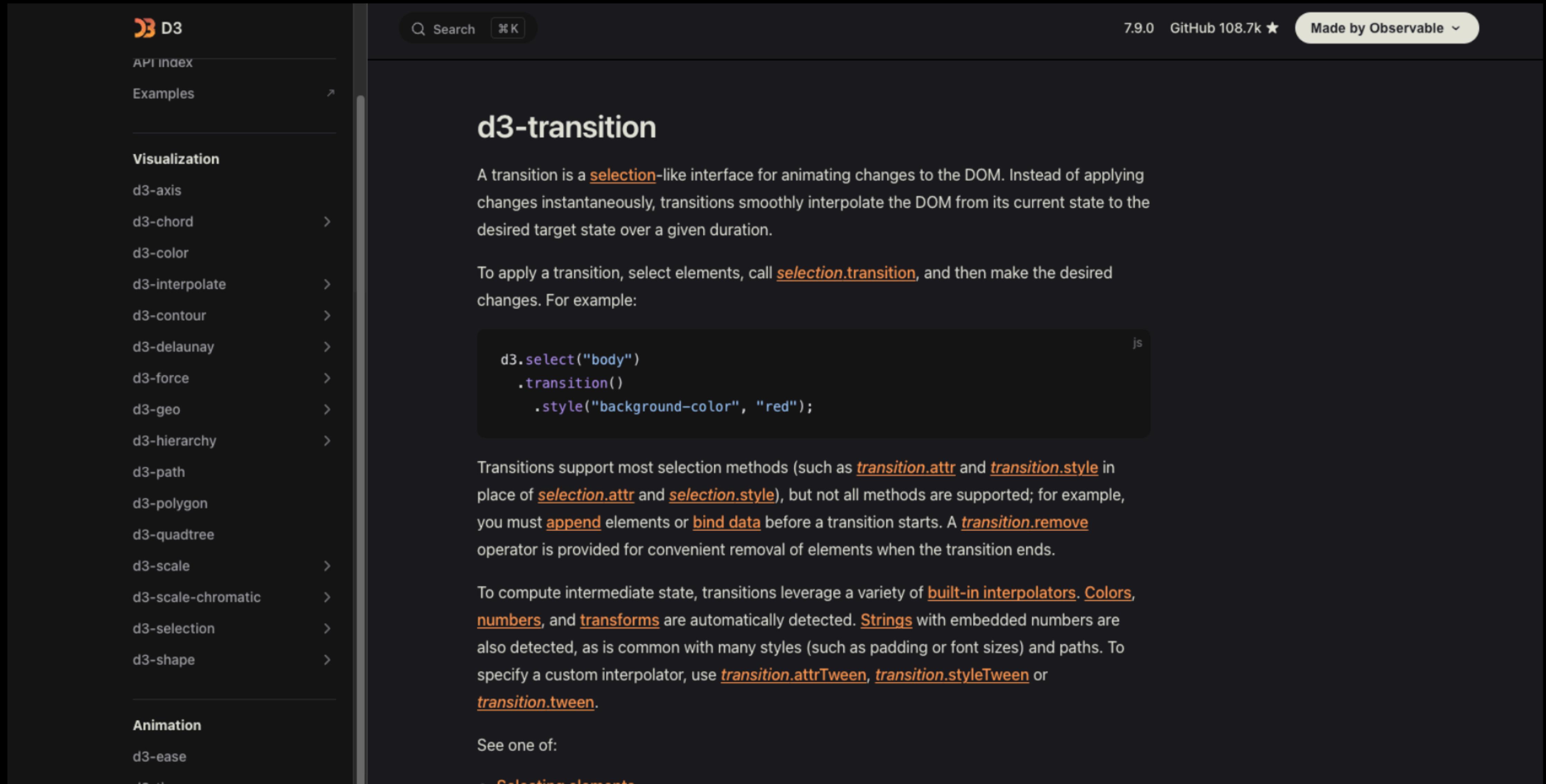
1. Dataset ophalen (show and tell)
2. Review (scales, joins)
- 3. Events**
4. All together!



Events

1. Mouse events (clicks, mouseover etc.)
2. Keyboard events (keypress etc.)
3. Touch events (longpress etc.)
4. Drag & Zoom events (drag etc.)

Events (transitions)



The screenshot shows the d3.js API documentation for transitions. The left sidebar has a dark theme with a navigation menu. The main content area has a light background and displays the following information:

d3-transition

A transition is a [selection](#)-like interface for animating changes to the DOM. Instead of applying changes instantaneously, transitions smoothly interpolate the DOM from its current state to the desired target state over a given duration.

To apply a transition, select elements, call [selection.transition](#), and then make the desired changes. For example:

```
js
d3.select("body")
  .transition()
  .style("background-color", "red");
```

Transitions support most selection methods (such as [transition.attr](#) and [transition.style](#) in place of [selection.attr](#) and [selection.style](#)), but not all methods are supported; for example, you must [append](#) elements or [bind data](#) before a transition starts. A [transition.remove](#) operator is provided for convenient removal of elements when the transition ends.

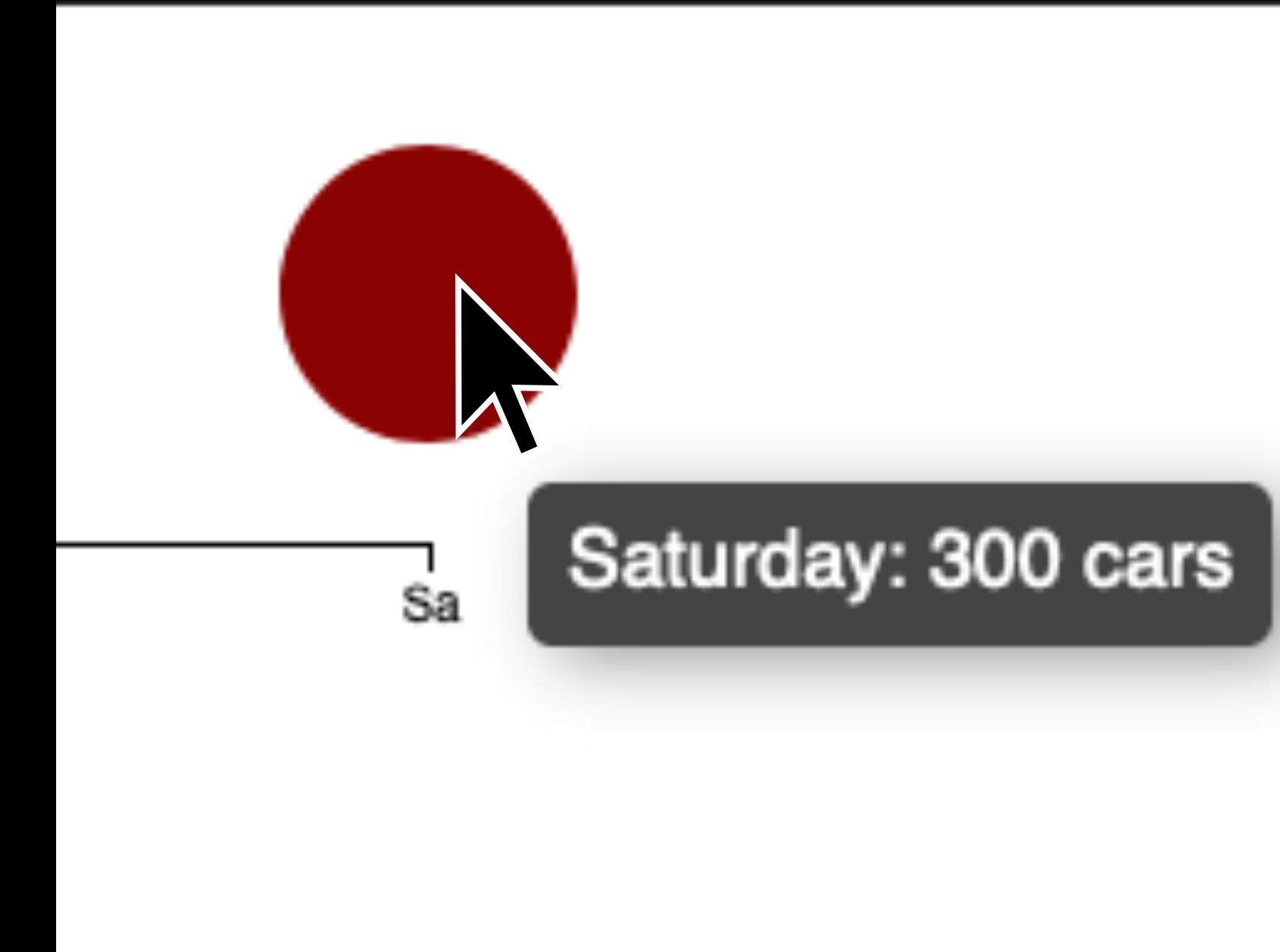
To compute intermediate state, transitions leverage a variety of [built-in interpolators](#). [Colors](#), [numbers](#), and [transforms](#) are automatically detected. [Strings](#) with embedded numbers are also detected, as is common with many styles (such as padding or font sizes) and paths. To specify a custom interpolator, use [transition.attrTween](#), [transition.styleTween](#) or [transition.tween](#).

See one of:

- [Selecting elements](#)

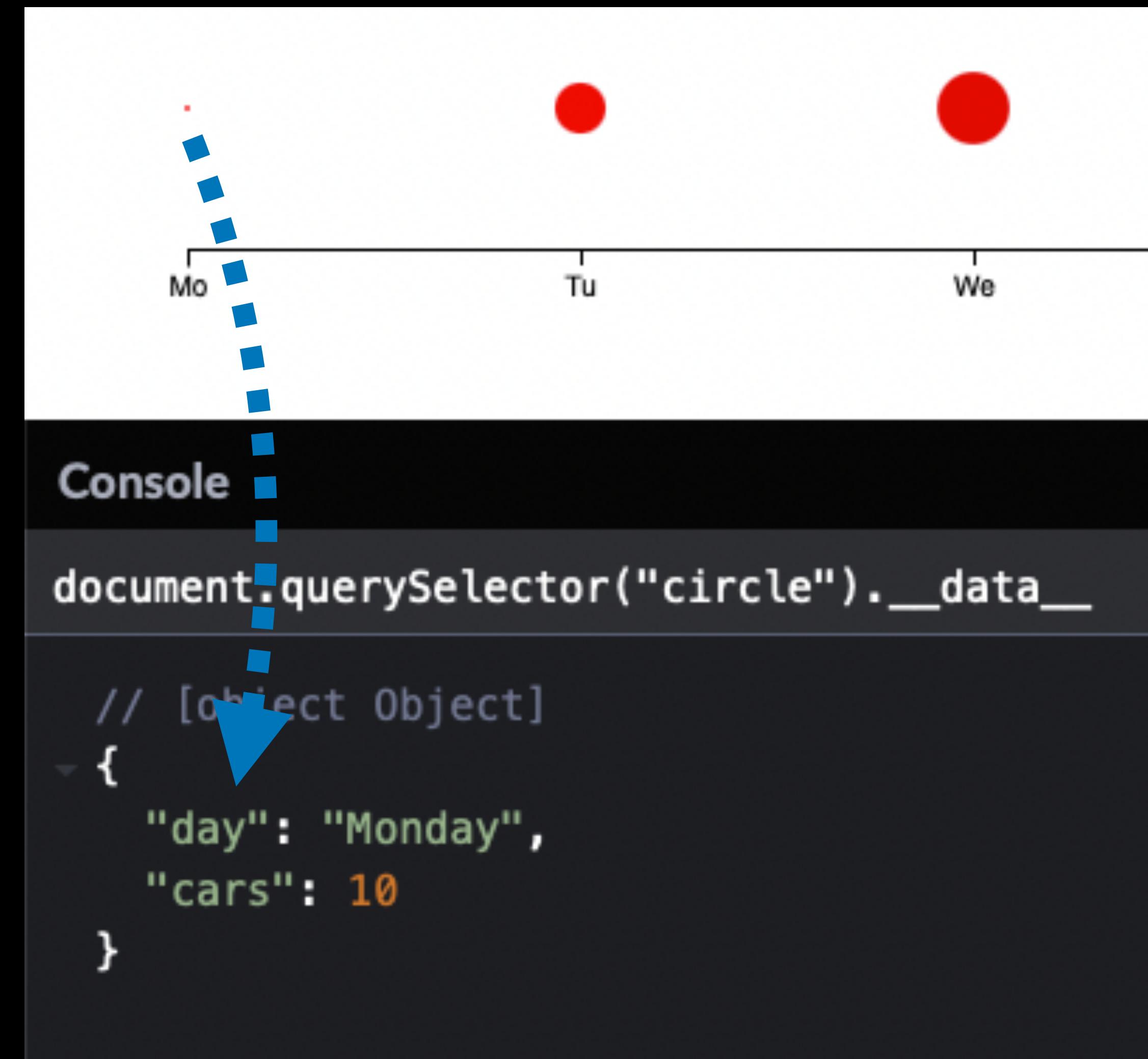
Events

Events help you to add interactivity to your graphs. For example you can add a tooltip or a side panel showing additional details.



Event data

D3 has a magical feature: it adds a `__data__` object to all DOM elements you created so you have access to the original data in your event.



```
Mo Tu We
Console
document.querySelector("circle").__data__
// [object Object]
{
  "day": "Monday",
  "cars": 10
}
```

Event binding



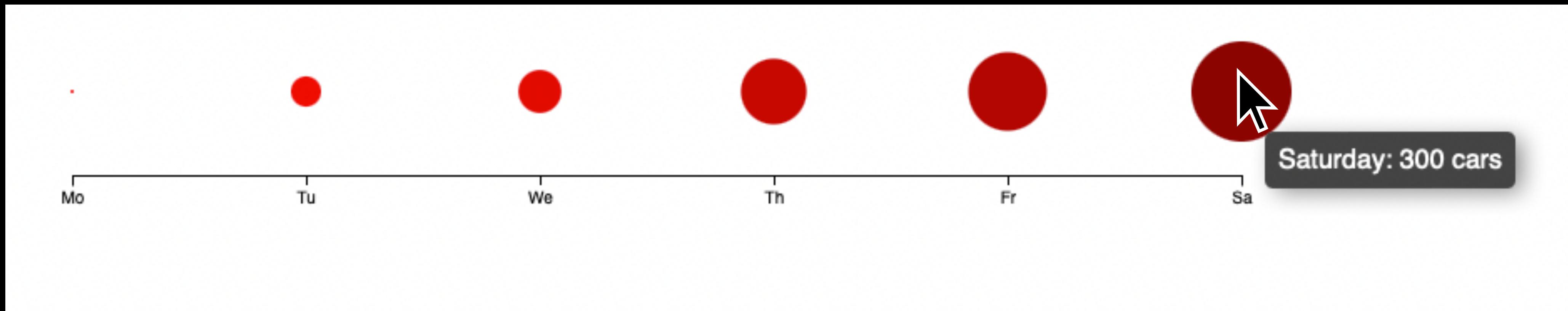
```
d3.select("#scale1")
  .selectAll("circle")
  .data(dataSet)
  .join("circle")
  .on("mouseover", (e, d) =>
    d3.select("#tooltip")
      .style("opacity", 1)
      .text(` ${d.day}: ${d.cars} cars`)
  )
  .on("mousemove", (e) =>
    d3
      .select("#tooltip")
      .style("left", e.pageX + 15 + "px")
      .style("top", e.pageY + 15 + "px")
  )
)
```

You add events by calling `d3.on()`. D3 will call your event function with two parameters:

1. Event data
2. Object data used during `d3.join()`

Tooltip demo

<https://codepen.io/dandevri/pen/azdrEQb>



Huiswerk

- Maak je visualisatie

**Uncaught SyntaxError
Unexpected end of input**