D3 • d3js.org
Bring your data to life.

Force-Directed Graph

This network of character co-occurence in *Les Misérables* is positioned by simulated forces using d3-force. See also a disconnected graph, and compare to WebCoLa.

```
chart = {
  const links = data.links.map(d => Object.create(d));
  const nodes = data.nodes.map(d => Object.create(d));

  const simulation = d3.forceSimulation(nodes)
      .force("link", d3.forceLink(links).id(d => d.id))
      .force("charge", d3.forceManyBody())
      .force("center", d3.forceCenter(width / 2, height / 2));

  const svg = d3.create("svg")
      .attr("viewBox", [0, 0, width, height]);

  const link = svg.append("g")
      .attr("stroke", "#999")
      .attr("stroke-opacity", 0.6)
    .selectAll("line")
    .data(links)
    .join("line")
      .attr("stroke-width", d => Math.sqrt(d.value));

  const node = svg.append("g")
      .attr("stroke", "#fff")
      .attr("stroke-width", 1.5)
```

```
    .selectAll("circle")
    .data(nodes)
    .join("circle")
      .attr("r", 5)
      .attr("fill", color)
      .call(drag(simulation));

  node.append("title")
      .text(d => d.id);

  simulation.on("tick", () => {
    link
        .attr("x1", d => d.source.x)
        .attr("y1", d => d.source.y)
        .attr("x2", d => d.target.x)
        .attr("y2", d => d.target.y);

    node
        .attr("cx", d => d.x)
        .attr("cy", d => d.y);
  });

  invalidation.then(() => simulation.stop());

  return svg.node();
}
```

```
data = ▼ Object {
  nodes: ▶ Array(77) [Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Obje
  links: ▶ Array(254) [Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Obj
}
```

```
data = FileAttachment("miserables.json").json()
```

```
height = 600
```

```
height = 600
```

```
color = ƒ(d)
```

```
color = {
  const scale = d3.scaleOrdinal(d3.schemeCategory10);
  return d => scale(d.group);
}
```

```
drag = ƒ(simulation)
```

```
drag = simulation => {

  function dragstarted(event) {
    if (!event.active) simulation.alphaTarget(0.3).restart();
    event.subject.fx = event.subject.x;
    event.subject.fy = event.subject.y;
  }

  function dragged(event) {
    event.subject.fx = event.x;
```

Fork 4 unsaved edits

```
  function dragended(event) {
    if (!event.active) simulation.alphaTarget(0);
    event.subject.fx = null;
    event.subject.fy = null;
  }

  return d3.drag()
      .on("start", dragstarted)
      .on("drag", dragged)
```

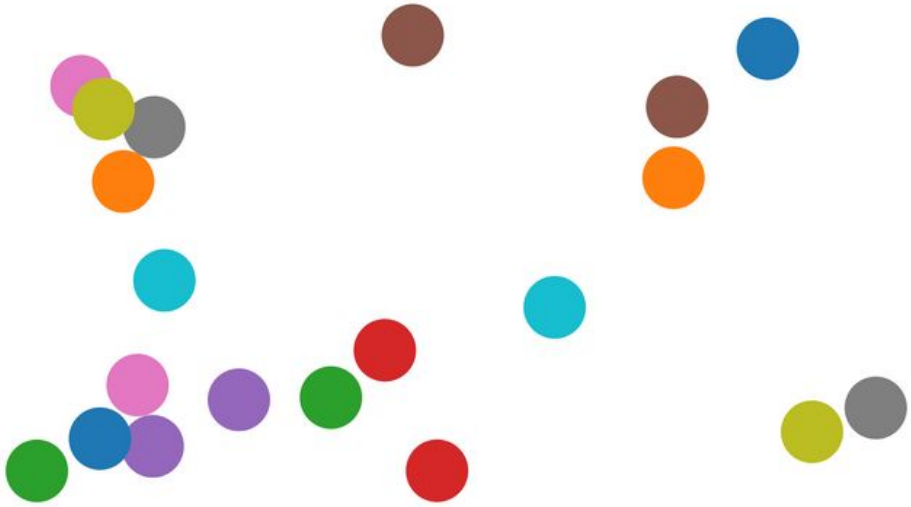
```
        .on("end", dragended);
}
```

```
d3 = ▸ Object {format: ƒ(t), formatPrefix: ƒ(t, n), timeFormat: ƒ(t), timeParse: ƒ(t), utcFormat: ƒ(t), utcParse:
```

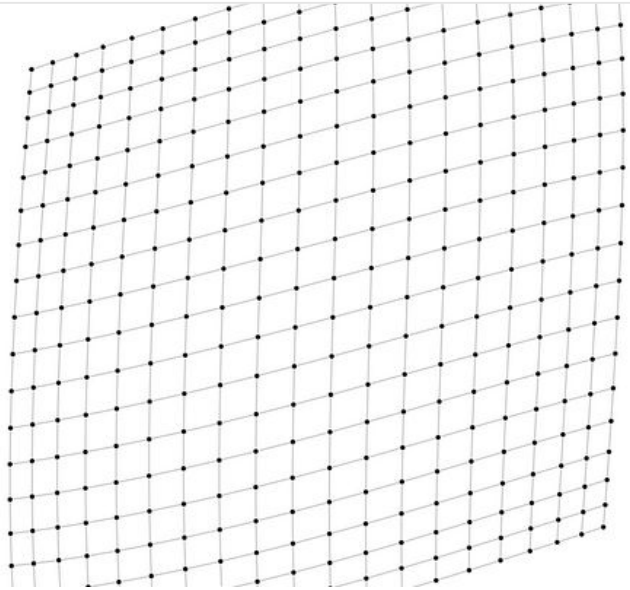
```
d3 = require("d3@6")
```

MORE FROM THIS COLLECTION

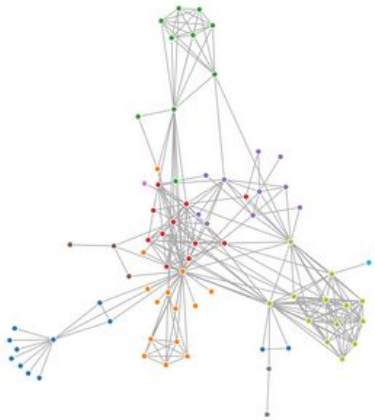
d3-drag



Drag (Canvas View)



Force-Directed Lattice



Force-Directed Graph (canvas)