# D3: Data-Driven Documents

**D3** (or **D3.js**) is a JavaScript library for visualizing data using web standards. D3 helps you bring data to life using SVG, Canvas and HTML. D3 combines powerful visualization and interaction techniques with a data-driven approach to DOM manipulation, giving you the full capabilities of modern browsers and the freedom to design the right visual interface for your data.

## Resources

- Introduction
- API Reference
- Releases
- Examples
- Wiki

## Installing

If you use npm, `npm install d3`. You can also download the latest release on GitHub. For vanilla HTML in modern browsers, import D3 from Skypack:

```
<script type="module">

import * as d3 from "https://cdn.skypack.dev/d3@7";

const div = d3.selectAll("div");

</script>
```

For legacy environments, you can load D3's UMD bundle from an npm-based CDN such as jsDelivr; a `d3` global is exported:

```
<script src="https://cdn.jsdelivr.net/npm/d3@7"></script>
<script>

const div = d3.selectAll("div");

</script>
```

You can also use the standalone D3 microlibraries. For example, d3-selection:

```
<script type="module">

import {selectAll} from "https://cdn.skypack.dev/d3-selection@3";

const div = selectAll("div");

</script>
```

D3 is written using ES2015 modules. Create a custom bundle using Rollup, Webpack, or your preferred bundler. To import D3 into an ES2015 application, either import specific symbols from specific D3 modules:

```
import {scaleLinear} from "d3-scale";
```

Or import everything into a namespace (here, `d3`):

```
import * as d3 from "d3";
```

Or using dynamic import:

```
const d3 = await import("d3");
```

You can also import individual modules and combine them into a `d3` object using Object.assign:

```
const d3 = await Promise.all([
  import("d3-format"),
  import("d3-geo"),
  import("d3-geo-projection")
]).then(d3 => Object.assign({}, ...d3));
```