D3 (**Data-Driven Documents** 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
- Tutorials
- Plugins

Help & Community

- Stack Overflow
- Google Group
- Slack (Invite)
- Gitter
- IRC: #d3.js on irc.freenode.net

Translations (Unofficial)

- 한국어
- 日本語
- 中文手册
- 中文手册v4
- 简体中文
- 繁體中文
- Русский
- Türkçe
- Indonesian
- Português
- Español

Getting Started

Observable is the quickest way to start playing with D3. Read the introduction or browse the D3 example gallery for inspiration, and then fork a notebook!

Installing

See https://github.com/d3/d3/blob/main/README.md#installing

Supported Environments

D3 5+ supports recent browsers, such as Chrome, Edge, Firefox and Safari. D3 4 and below also supports IE 9+. Parts of D3 may work in older browsers, as many D3 modules have minimal requirements. For example, d3-selection uses the Selectors API Level 1, but you can preload Sizzle for compatibility. You'll need a modern browser to use SVG and CSS3 Transitions. D3 is not a compatibility layer, so if your browser doesn't support standards, you're out of luck. Sorry!

D3 also runs on Node and web workers. To use the DOM in Node, you must provide your own DOM implementation; JSDOM is recommended. To avoid defining a global `document`, pass a DOM element to d3.select or a NodeList to d3.selectAll, like so:

```
import {select} from "d3-selection";
import {JSDOM} from "jsdom";

const jsdom = new JSDOM(html);
const svg = select(jsdom.window.document.body).append("svg");
```

When using D3 in an environment that supports ES modules, you can import the default D3 bundle as a namespace:

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

If you want to import a D3 module that is not included in the default bundle, you must assign it a separate namespace:

```
import * as d3 from "d3";
import * as d3GeoProjection from "d3-geo-projection";
```

For this reason, the preferred pattern is to import symbols from the D3 modules directly, rather than using the default bundle:

```
import {select, selectAll} from "d3-selection";
import {geoPath} from "d3-geo";
import {geoPatterson} from "d3-geo-projection";
import "d3-transition";
```

If you are using a bundler, make sure your bundler is configured to consume the `modules` entry point in the package.json. See webpack's resolve.mainFields, for example.

Local Development

Browsers enforce strict security permissions to prevent you from reading files out of the local file system. To develop locally, you must run a local web server rather than using file://…. Node's http-server is recommended. To install:

```
npm install -g http-server
```

To run:

```
http-server &
```

This will start the server on http://localhost:8080 from the current working directory.