Chapter 4. Setup

Getting set up with D3 is pretty straightforward—a simple matter of downloading the latest version, creating an empty page in which to write your code, and finally setting up a local web server.

Downloading D3

Start by creating a new folder for your project. Call it whatever you like, but maybe something like *project-folder*.

Within that folder, I recommend creating a subfolder called *d3*. Then download the latest version of D3 into that subfolder. Download the latest version of D3 as a ZIP file, and then decompress the ZIP file. As of this writing, the current version of D3 is 3.0.6.

D3 is also provided in a “minified” version, *d3.v3.min.js*, from which whitespace has been removed for smaller file sizes and faster load times. The functionality is the same, but typically you’d use the regular version while working on a project (for friendlier debugging), and then switch to the minified version once you’ve launched the project publicly (for optimized load times). The choice is up to you, but in this book, I use the standard version.

A third option is to download the entire D3 repository, which gives you not just the JavaScript files, but also all of the component source code. You can browse the repository contents first, or just download the whole thing as a compressed ZIP file. Of course, once you’ve downloaded everything, make a copy of the file *d3.v3.js* and move that into *project-folder/d3/*.

Referencing D3

Now create a simple HTML page within your project folder named *index.html*. Remember, HTML documents are just plain text files, so you can use the text editor of your choice. Free editors like TextEdit and Notepad are fine, but your life might be easier if you use an editor designed specifically for working with code, like Coda, Espresso, or Sublime Text (among many, many others).

If your editor gives you the option to set the file encoding, choose Unicode (UTF-8).

Your folder structure should now look something like this:

```
project-folder/
```

```
    d3/
        d3.v3.js
        d3.v3.min.js (optional)
    index.html
```

Paste the following into your new HTML file:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>D3 Page Template</title>
        <script type="text/javascript" src="d3/d3.v3.js"></script>
    </head>
    <body>
        <script type="text/javascript">
            // Your beautiful D3 code will go here
        </script>
    </body>
</html>
```

Or, rather than go through all that manual labor, you could just download the sample code files (see Chapter 1 for instructions), and take a look at *01_empty_page_template.html*, which contains almost exactly the same code. (The `src` path to D3 is different in my version.)

Here are a few things to note about this template:

- The `meta` tag identifies the encoding for this file as `utf-8`, which is needed to ensure that the browser can parse D3's functions and data properly.
- The first `script` tag sets the reference to *d3.v3.js*. You should edit this file path as needed if you're using the minified version of D3 or decided to locate *d3.v3.js* somewhere other than the *d3* directory.
- The second `script` tag, in the `body`, is where you will soon key in all your beautiful code. And it will be beautiful.

Done! Your D3 template files and folders are all set up. You might want to make a copy of this template for each new project down the line.

Setting Up a Web Server

In some cases, you can view local HTML files directly in your web browser. However, some browsers have restrictions that prevent them from loading local files via JavaScript, for security reasons. That means if your D3 code is trying to pull in any external datafiles (like CSVs or JSON), it will fail with no good explanation. This isn't D3's fault; it's a browser feature that prevents loading of scripts and other external files from third-party, untrusted websites.

For this reason, it is much more reliable to load your page via a web server. Although you *could* use a remote web server, it is much, much faster to store and host everything locally (meaning, on the same computer, the one right in front of you). It is a strange idea, to use your local computer to host and serve files to itself, but you can think about it as the different programs talking to each other: the browser program requests files from the server program, which responds by serving them back.

The good news is that it's quite easy to get a local server up and running. Here are a couple of ways to do that.

Terminal with Python

If you're using Mac OS X or Linux, then you already have Python installed. As long as you're comfortable entering commands in the terminal, then running a miniserver with Python is definitely the quickest option. (If you're on Windows, you'll need to install Python first.)

To use Python, you'll need to open up a terminal window on your system. On a Mac, open the Terminal application. You can find it in the Utilities folder, or by typing **`Terminal`** into Spotlight (the magnifying glass menu item in the upper-right corner of your screen). Linux users are born knowing how to open a terminal window, so I won't waste your time explaining it here.

To run a Python web server:

1. Open up a new terminal window.
2. Via the command line, navigate into the directory that you want served. For example, if your project folder is in your Desktop folder on your Mac, you could type: **`cd ~/Desktop/project-folder`**.
3. Enter **`python -m SimpleHTTPServer 8888 &`**.

(This will work with Python version 2.x, but in Python versions 3.0 and newer, `SimpleHTTPServer` has been removed. For Python 3.x, just replace `SimpleHTTPServer` with `http.server` in the command.)

This will activate the server on port `8888`. Switch back to your web browser and visit the following URL: http://localhost:8888/. Yes, instead of *www.something.com*, you just use *localhost*, which tells the browser to request a page from *this machine*.

You should see the blank “D3 Page Template” page. Given that `body` of the page is empty, it won’t look like much. Select View source, and you should see the contents of our HTML template page.

MAMP, WAMP, and LAMP

This option takes longer, but is best if you like dragging and dropping to install things, and want to avoid scary things like the terminal.

All of the AMPs in this section stand for Apache (the web server software), MySQL (popular database software), and PHP (a popular web scripting language). We are really interested only in Apache, the web server, but it usually comes bundled with the other two, as they all work well together.

On a Mac, you can download and install MAMP or XAMPP for Mac.

The Windows equivalents are WampServer and XAMPP for Windows.

If you use Linux, then all of this is probably already installed on your machine, but you could still download XAMPP for Linux.

Installation for each of these packages varies somewhat, so follow the documentation carefully. (I’ve found MAMP to be the easiest to install.)

Each package will designate one folder as the web server directory, so only the files *within that folder* will be served. You should find out what that folder is, and move your D3 *project-folder* into it.

Once the local server is up and running, you can view any pages within the server directory by opening a browser (on the same computer, of course) and pointing it to *localhost*, as in the following: http://localhost/. Depending on your AMP configuration, you might need to append a port number to the URL, as in the following: http://localhost:8888/.

If the server’s port number is `8888` and your project folder is called *project-folder*, then you could view your D3 template page by going to http://localhost:8888/project-folder/.

Diving In

All set? Great! Let’s start working with data.