D3.js

tutorialspoint

SIMPLY EASY LEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About the Tutorial

D3 stands for **Data-Driven Documents**. D3.js is a JavaScript library for manipulating documents based on data. D3.js is a dynamic, interactive, online data visualizations framework used in a large number of websites. D3.js is written by **Mike Bostock**, created as a successor to an earlier visualization toolkit called **Protovis**. This tutorial will give you a complete knowledge on D3.jsframework.

This is an introductory tutorial, which covers the basics of Data-Driven Documents and explains how to deal with its various components and sub-components.

Audience

This tutorial is prepared for professionals who are aspiring to make a career in online data visualization. This tutorial is intended to make you comfortable in getting started with the Data-Driven Documents and its various functions.

Prerequisites

Before proceeding with the various types of concepts given in this tutorial, it is being assumed that the readers are already aware about what a **Framework** is. In addition to this, it will be very helpful, if the readers have a sound knowledge on HTML, CSS and JavaScript.

Copyright and Disclaimer

Table of Contents

1. D3.JS – INTRODUCTION

Data visualization is the presentation of data in a pictorial or graphical format. The primary goal of data visualization is to communicate information clearly and efficiently via statistical graphics, plots and information graphics.

Data visualization helps us to communicate our insights quickly and effectively. Any type of data, which is represented by a visualization allows users to compare the data, generate analytic reports, understand patterns and thus helps them to take the decision. Data visualizations can be interactive, so that users analyze specific data in the chart. Well, Data visualizations can be developed and integrated in regular websites and even mobile applications using different JavaScript frameworks.

What is D3.js?

D3.js is a JavaScript library used to create interactive visualizations in the browser. The D3.js library allows us to manipulate elements of a webpage in the context of a data set. These elements can be **HTML**, **SVG**, or **Canvas elements** and can be introduced, removed, or edited according to the contents of the data set. It is a library for manipulating the DOM objects. D3.js can be a valuable aid in data exploration, it gives you control over your data's representation and lets you add interactivity.

Why Do We Need D3.js?

D3.js is one of the premier framework when compare to other libraries. This is because it works on the web and its data visualizations are par excellence. Another reason it has worked so well is owing to its flexibility. Since it works seamlessly with the existing web technologies and can manipulate any part of the document object model, it is as flexible as the **Client Side Web Technology Stack** (HTML, CSS, and SVG). It has a great community support and is easier to learn.

D3.js Features

D3.js is one of the best data visualization framework and it can be used to generate simple as well as complex visualizations along with user interaction and transition effects. Some of its salient features are listed below:

- Extremely flexible.
- Easy to use and fast.
- Supports large datasets.

- Declarative programming.
- Code reusability.
- Has wide variety of curve generating functions.
- Associates data to an element or group of elements in the html page.

D3.js Benefits

D3.js is an open source project and works without any plugin. It requires very less code and comes up with the following benefits:

- Great data visualization.
- It is modular. You can download a small piece of D3.js, which you want to use. No need to load the whole library every time.
- Easy to build a charting component.
- DOM manipulation.

In the next chapter, we will understand how to install D3.js on our system.

2.D3.JS – INSTALLATION

In this chapter, we will learn how to set up the D3.js development environment. Before we start, we need the following components:

- D3.js library
- Editor
- Web browser
- Web server

Let us go through the steps one by one in detail.

D3.js Library

We need to include the D3.js library into your HTML webpage in order to use D3.js to create data visualization. We can do it in the following two ways:

- Include the D3.js library from your project's folder.
- Include D3.js library from CDN (Content Delivery Network).

Download D3.js Library

D3.js is an open-source library and the source code of the library is freely available on the web at https://d3js.org/ website. Visit the D3.js website and download the latest version of D3.js (d3.zip). As of now, the latest version is 4.6.0.

After the download is complete, unzip the file and look for **d3.min.js**. This is the minified version of the D3.js source code. Copy the d3.min.js file and paste it into your project's root folder or any other folder, where you want to keep all the library files. Include the d3.min.js file in your HTML page as shown below.

Example: Let us consider the following example.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <script src="/path/to/d3.min.js"></script>
```

```
</head>
<body>
<script>
    // write your d3 code here..
</script>
</body>
</html>
```

D3.js is a JavaScript code, so we should write all our D3 code within "script" tag. We may need to manipulate the existing DOM elements, so it is advisable to write the D3 code just before the end of the "body" tag.

Include D3 Library from CDN

We can use the D3.js library by linking it directly into our HTML page from the Content Delivery Network (CDN). CDN is a network of servers where files are hosted and are delivered to a user based on their geographic location. If we use the CDN, we do not need to download the source code.

Include the D3.js library using the CDN URL https://d3js.org/d3.v4.min.js into our page as shown below.

Example: Let us consider the following example.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <script src="https://d3js.org/d3.v4.min.js"></script>
</head>
<body>

<script>
    // write your d3 code here..
</script>

</body>
</html>
```

D3.js Editor

We will need an editor to start writing your code. There are some great IDEs (Integrated Development Environment) with support for JavaScript like –

- Visual Studio Code
- WebStorm
- Eclipse
- Sublime Text

These IDEs provide intelligent code completion as well as support some of the modern JavaScript frameworks. If you do not have fancy IDE, you can always use a basic editor like Notepad, VI, etc.

Web Browser

D3.js works on all the browsers except IE8 and lower.

Web Server

Most browsers serve local HTML files directly from the local file system. However, there are certain restrictions when it comes to loading external data files. In the latter chapters of this tutorial, we will be loading data from external files like **CSV** and **JSON**. Therefore, it will be easier for us, if we set up the web server right from the beginning.

You can use any web server, which you are comfortable with – e.g. IIS, Apache, etc.

Viewing Your Page

In most cases, we can just open your HTML file in a web browser to view it. However, when loading external data sources, it is more reliable to run a local web server and view your page from the server (http://localhost:8080).

3.D3.JS – CONCEPTS

D3.js is an open source JavaScript library for –

- Data-driven manipulation of the Document Object Model (DOM).
- Working with data and shapes.
- Laying out visual elements for linear, hierarchical, network and geographic data.
- Enabling smooth transitions between user interface (UI) states.
- Enabling effective user interaction.

Web Standards

Before we can start using D3.js to create visualizations, we need to get familiar with web standards. The following web standards are heavily used in D3.js.

- HyperText Markup Language (HTML)
- Document Object Model (DOM)
- Cascading Style Sheets (CSS)
- Scalable Vector Graphics (SVG)
- JavaScript

Let us go through each of these web standards one by one in detail.

HyperText Markup Language (HTML)

As we know, HTML is used to structure the content of the webpage. It is stored in a text file with the extension ".html".

Example: A typical bare-bones HTML example looks like this

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title></title>
</head>
<body>


</body>
</html>
```

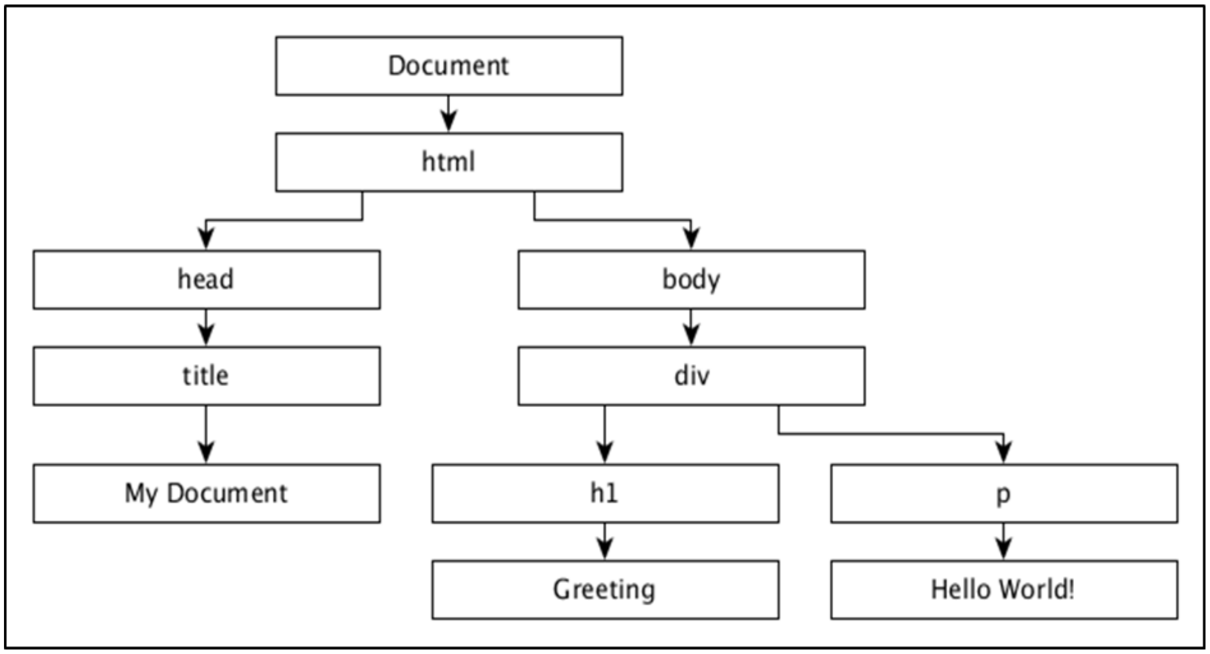
Document Object Model (DOM)

When a HTML page is loaded by a browser, it is converted to a hierarchical structure. Every tag in HTML is converted to an element / object in the DOM with a parent-child hierarchy. It makes our HTML more logically structured. Once the DOM is formed, it becomes easier to manipulate (add/modify/remove) the elements on the page.

Let us understand the DOM using the following HTML document:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>My Document</title>
    </head>
    <body>
        <div>
            <h1>Greeting</h1>
            <p>Hello World!</p>
        </div>
    </body>
</html>
```

The document object model of the above HTML document is as follows,



Cascading Style Sheets (CSS)

While HTML gives a structure to the webpage, CSS styles makes the webpage more pleasant to look at. CSS is a **Style Sheet Language** used to describe the presentation of a document written in HTML or XML (including XML dialects like SVG or XHTML). CSS describes how elements should be rendered on a webpage.

Scalable Vector Graphics (SVG)

SVG is a way to render images on the webpage. SVG is not a direct image, but is just a way to create images using text. As its name suggests, it is a **Scalable Vector**. It scales itself according to the size of the browser, so resizing your browser will not distort the image. All browsers support SVG except IE 8 and below. Data visualizations are visual representations and it is convenient to use SVG to render visualizations using the D3.js.

Think of SVG as a canvas on which we can paint different shapes. So to start with, let us create an SVG tag:

```
<svg width="500" height="500"></<svg>
```

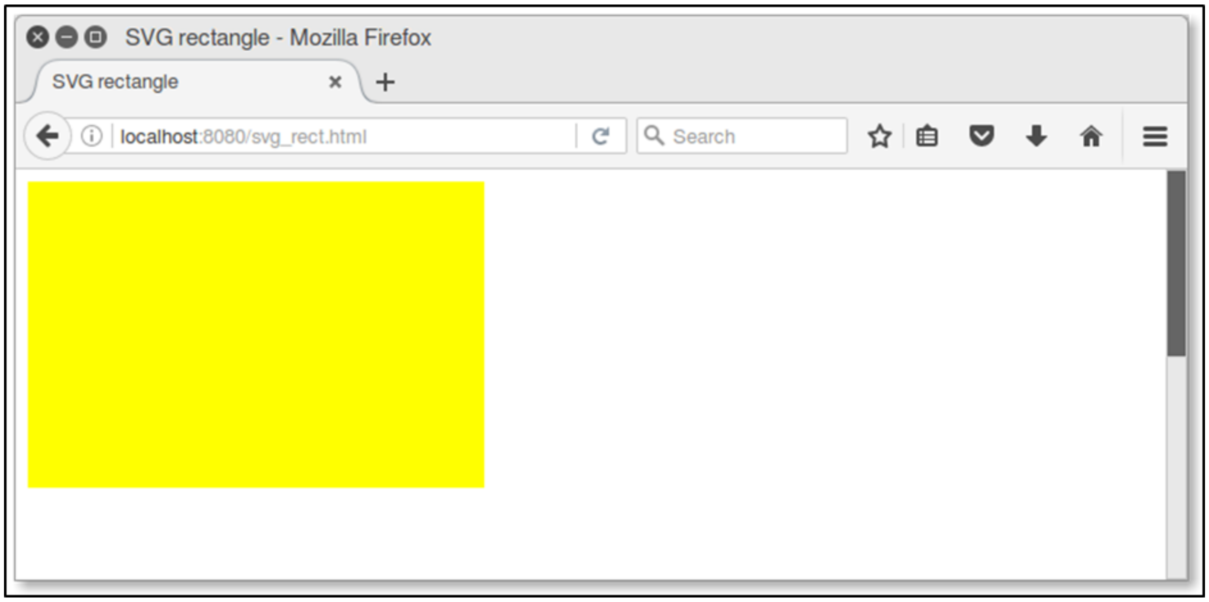
The default measurement for SVG is pixels, so we do not need to specify if our unit is pixel. Now, if we want to draw a rectangle, we can draw it using the code below:

```
<svg width="500" height="500">
    <rect x="0" y="0" width="300" height="200"></rect>
</svg>
```

We can draw other shapes in SVG such as – Line, Circle, Ellipse, Text and Path.

Just like styling HTML elements, styling SVG elements is simple. Let us set the background color of the rectangle to yellow. For that, we need to add an attribute "fill" and specify the value as **yellow** as shown below:

```
<svg width="500" height="500">
    <rect x="0" y="0" width="300" height="200" fill="yellow"></rect>
</svg>
```



JavaScript

JavaScript is a loosely typed client side scripting language that executes in the user's browser. JavaScript interacts with HTML elements (DOM elements) in order to make the web user interface interactive. JavaScript implements the **ECMAScript Standards**, which includes core features based on ECMA-262 specifications as well as other features, which are not based on the ECMAScript standards. JavaScript knowledge is a prerequisite for D3.js.

4. D3.JS – SELECTIONS

Selections is one of the core concepts in D3.js. It is based on CSS selectors. It allows us to select one or more elements in a webpage. In addition, it allows us to modify, append, or remove elements in a relation to the pre-defined dataset. In this chapter, we will see how to use selections to create data visualizations.

D3.js helps to select elements from the HTML page using the following two methods:

- **select() –** Selects only one DOM element by matching the given CSS selector. If there are more than one elements for the given CSS selector, it selects the first one only.
- **selectAll() –** Selects all DOM elements by matching the given CSS selector. If you are familiar with selecting elements with jQuery, D3.js selectors are almost the same.

Let us go through each of the methods in detail.

The select() method

The select() method selects the HTML element based on CSS Selectors. In CSS Selectors, you can define and access HTML-elements in the following three ways:

- Tag of a HTML element (e.g. div, h1, p, span, etc.,)
- Class name of a HTML element
- ID of a HTML element

Let us see it in action with examples.

Selection by Tag

You can select HTML elements using its TAG. The following syntax is used to select the “div” tag elements,

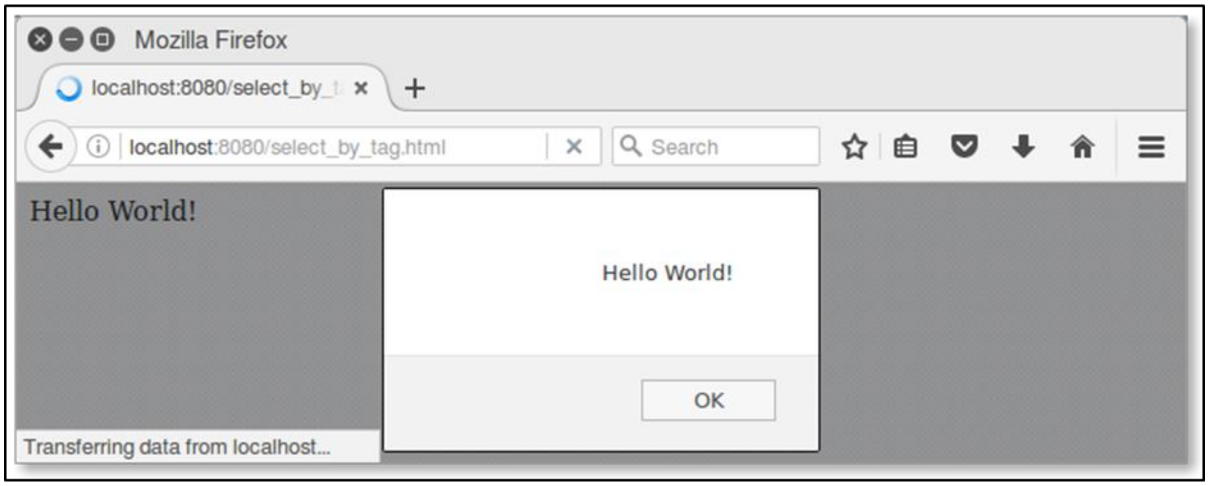
```
d3.select(“div”)
```

Example: Create a page “select_by_tag.html” and add the following changes,

```
<!DOCTYPE html>
<html>
    <head>
```

```
        <script type="text/javascript" src="d3/d3.min.js"></script>
    </head>
    <body>
        <div>
            Hello World!
        </div>
        <script>
alert(d3.select("div").text());
        </script>
    </body>
</html>
```

By requesting the webpage through the browser, you will see the following output on the screen:


Mozilla Firefox
localhost:8080/select_by_t
localhost:8080/select_by_tag.html
Search
Hello World!
OK
Transferring data from localhost...

Hello World!

Selection by Class name

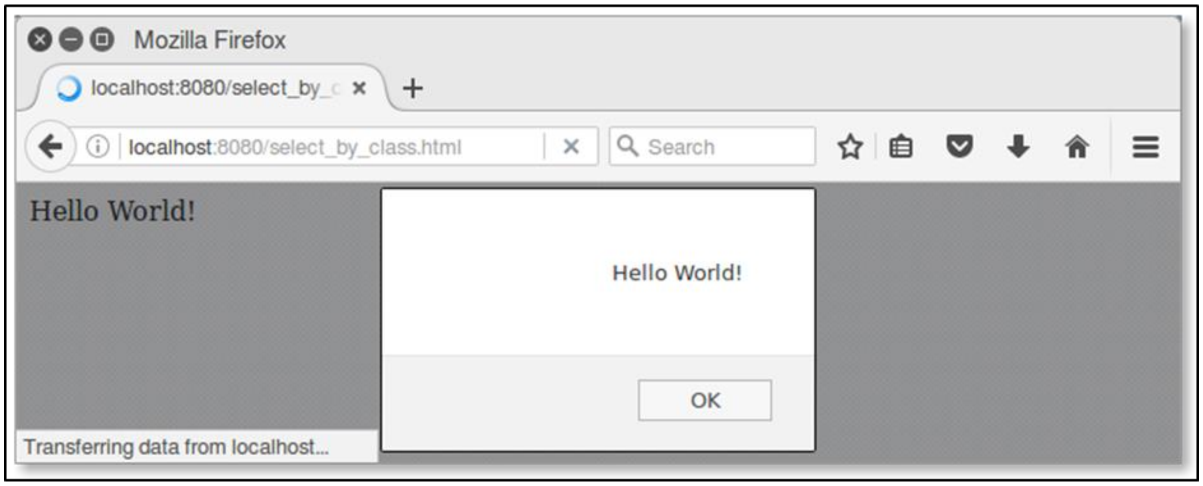
HTML elements styled using CSS classes can be selected by using the following syntax.

```
d3.select(“.<class name>”)
```

Create a webpage “select_by_class.html” and add the following changes:

```
<!DOCTYPE html>
<html>
    <head>
        <script type="text/javascript" src="d3/d3.min.js"></script>
    </head>
    <body>
        <div class="myclass">
            Hello World!
        </div>
        <script>
alert(d3.select(".myclass").text());
        </script>
    </body>
</html>
```

By requesting the webpage through the browser, you will see the following output on the screen.


Mozilla Firefox
localhost:8080/select_by_c
localhost:8080/select_by_class.html
Search
OK

Hello World!

Hello World!

Transferring data from localhost...

Selection by ID

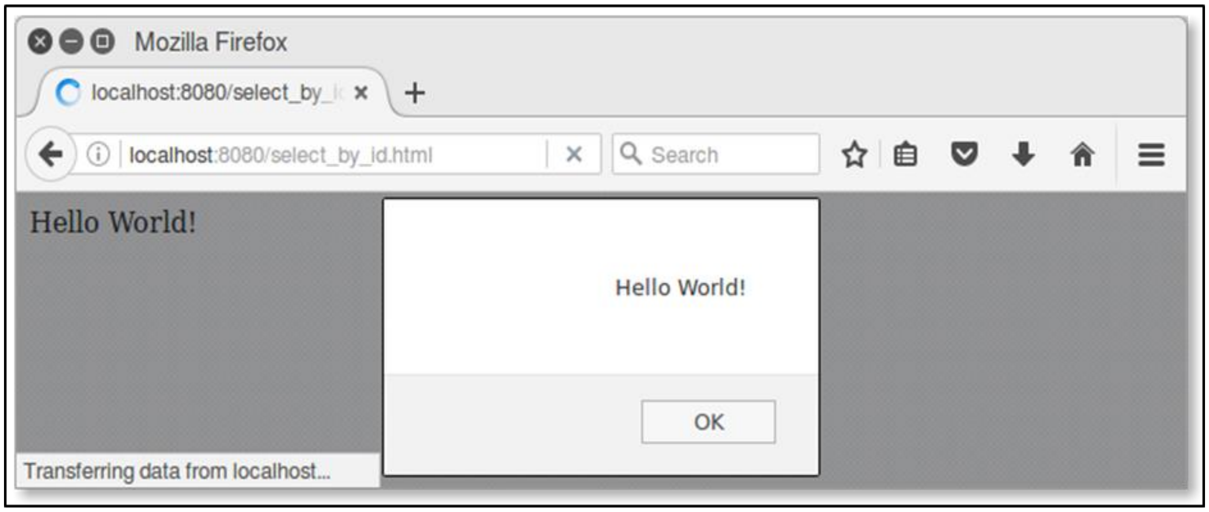
Every element in a HTML page should have a unique ID. We can use this unique ID of an element to access it using the select() method as specified below.

```
d3.select("#<id of an element>")
```

Create a webpage "select_by_id.html" and add the following changes.

```
<!DOCTYPE html>
<html>
    <head>
        <script type="text/javascript" src="d3/d3.min.js"></script>
    </head>
    <body>
        <div id="hello">
            Hello World!
        </div>
        <script>
alert(d3.select("#hello").text());
        </script>
    </body>
</html>
```

By requesting the webpage through the browser, you will see the following output on the screen.


Mozilla Firefox
localhost:8080/select_by_i
localhost:8080/select_by_id.html
Search
OK
Transferring data from localhost...

Hello World!

Hello World!

Adding DOM Elements

The D3.js selection provides the **append()** and the **text()** methods to append new elements into the existing HTML documents. This section explains about adding DOM elements in detail.

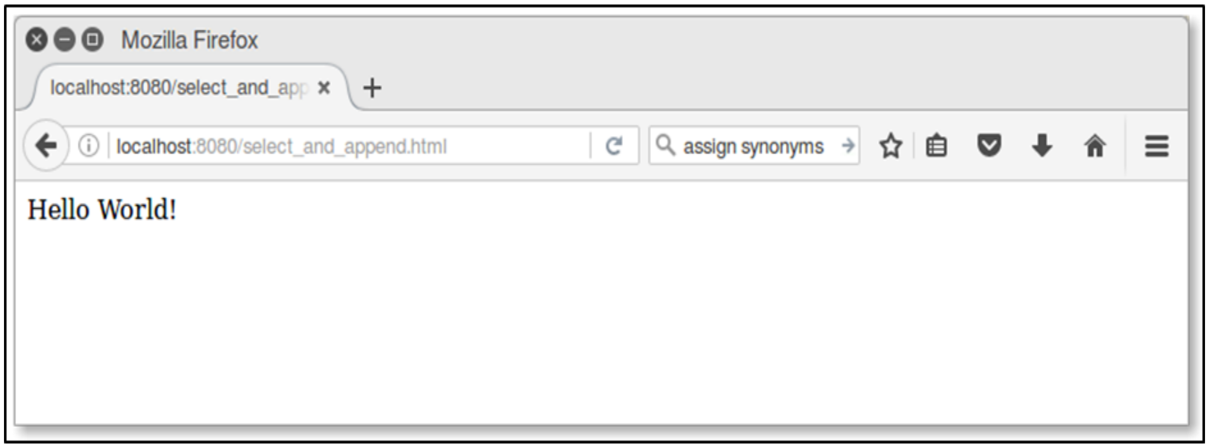
The append() Method

The append() method appends a new element as the last child of the element in the current selection. This method can also modify the style of the elements, their attributes, properties, HTML and text content.

Create a webpage "select_and_append.html" and add the following changes:

```
<!DOCTYPE html>
<html>
    <head>
        <script type="text/javascript" src="d3/d3.min.js"></script>
    </head>
    <body>
        <div class="myclass">
            Hello World!
        </div>
        <script>
d3.select("div.myclass").append("span");
        </script>
    </body>
</html>
```

Requesting the webpage through browser, you could see the following output on the screen,


Mozilla Firefox
localhost:8080/select_and_app
localhost:8080/select_and_append.html
assign synonyms

Hello World!

Here, the append() method adds a new tag span inside the div tag as shown below:

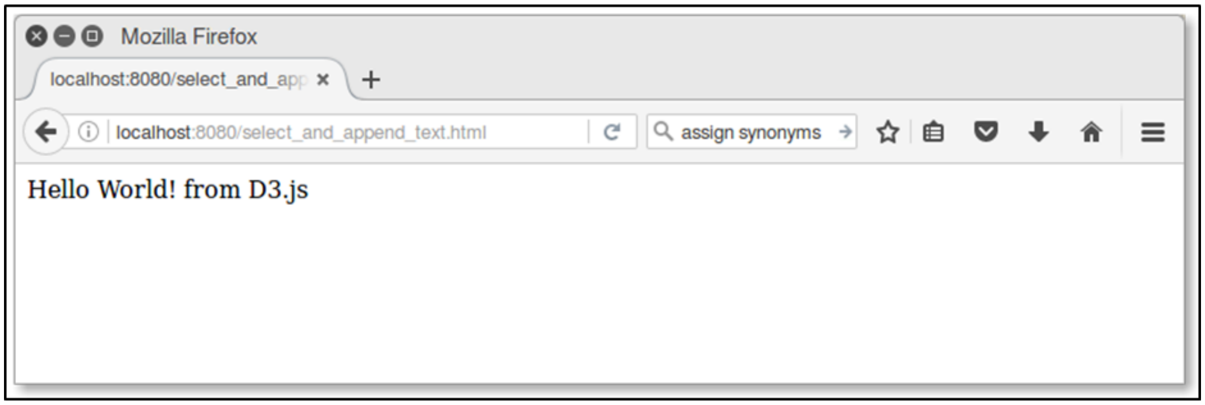
```
<div class="myclass">
    Hello World!<span></span>
</div>
```

The text() Method

The text() method is used to set the content of the selected / appended elements. Let us change the above example and add the text() method as shown below.

```
<!DOCTYPE html>
<html>
    <head>
        <script type="text/javascript" src="d3/d3.min.js"></script>
    </head>
    <body>
        <div class="myclass">
            Hello World!
        </div>
        <script>
d3.select("div.myclass").append("span").text("from D3.js");
        </script>
    </body>
</html>
```

Now refresh the webpage and you will see the following response.


Mozilla Firefox
localhost:8080/select_and_app
localhost:8080/select_and_append_text.html
assign synonyms

Hello World! from D3.js

Here, the above script performs a chaining operation. D3.js smartly employs a technique called the **chain syntax**, which you may recognize from **jQuery**. By chaining methods together with periods, you can perform several actions in a single line of code. It is fast and easy. The same script can also access without chain syntax as shown below.

```
var body = d3.select("div.myclass");
var span = body.append("span");
span.text("from D3.js");
```

Modifying Elements

D3.js provides various methods, **html()**, **attr()** and **style()** to modify the content and style of the selected elements. Let us see how to use modify methods in this chapter.

The html() Method

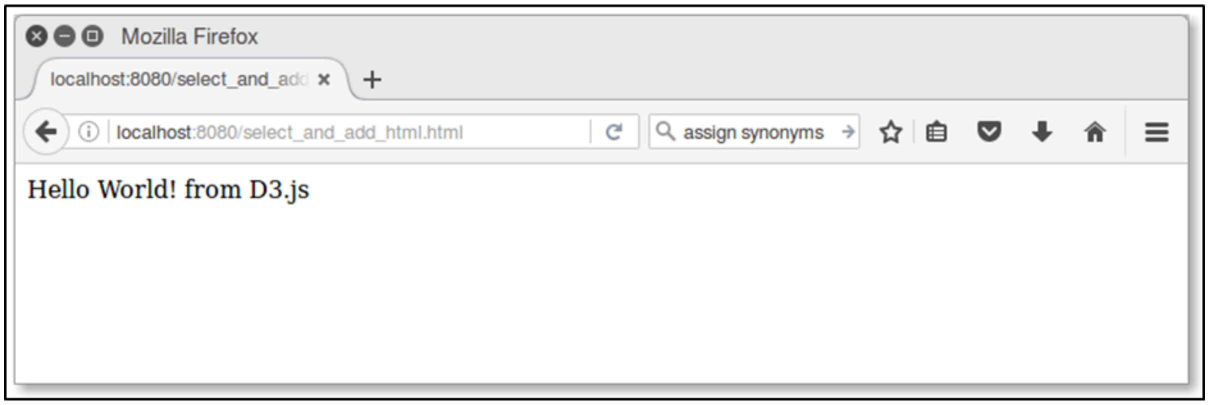
The html() method is used to set the html content of the selected / appended elements.

Create a webpage "select_and_add_html.html" and add the following code.

```
<!DOCTYPE html>
<html>
    <head>
        <script type="text/javascript" src="d3/d3.min.js"></script>
    </head>
    <body>
        <div class="myclass">
            Hello World!
        </div>
        <script>
```

```
d3.select(".myclass").html("Hello World! <span>from D3.js</span>");
        </script>
    </body>
</html>
```

By requesting the webpage through the browser, you will see the following output on the screen.


Mozilla Firefox
localhost:8080/select_and_add
localhost:8080/select_and_add_html.html
assign synonyms

Hello World! from D3.js

The attr() Method

The attr() method is used to add or update the attribute of the selected elements. Create a webpage "select_and_modify.html" and add the following code.

```
<!DOCTYPE html>
<html>
    <head>
        <script type="text/javascript" src="d3/d3.min.js"></script>
    </head>
    <body>
        <div class="myclass">
            Hello World!
        </div>
        <script>
d3.select(".myclass").attr("style", "color: red");
        </script>
    </body>
</html>
```