12. D3JS Selection API.

Selections are powerful data-driven transformation of the document object model (DOM). It is used to set Attributes, Styles, Properties, HTML or Text Content and much more. This chapter explains the selections API in detail.

12.1. Configuring API

You can configure the API directly using the script below.

Example:

```
<script src = "https://d3js.org/d3-selection.v1.min.js"></script>
<body>
   <script>
   </script>
</body>
```

12.2. Selection API Methods

Following are the most important methods in selection API.

- **d3.selection()**
- **d3.select(selector)**
- **d3.selectAll(selector)**
- **selection.selectAll(selector)**
- **selection.filter(filter)**
- **selection.merge(other)**
- **d3.matcher(selector)**
- **d3.creator(name)**
- **selection.each(function)**

- **selection.call(function[, arguments…])**
- **d3.local()**
- **local.set(node, value)**
- **local.get(node)**
- **local.remove(node)**

Let us go through each of these API in detail.

d3.selection()

This method is used to select the root element. This function can also be used to test for selections or to extend the selection d3js.

d3.select(selector)

This method is used to select the first element that matches the specified selector string.

Example: Consider the following script.

```
var body = d3.select("body");
```

If the selector is not a string, then it selects the specified node, which is defined below.

Example: Consider the following script.

```
d3.select("p").style("color", "red");
```

d3.selectAll(selector)

This method selects all the elements that match the specified selector string.

Example: Consider the following script.

```
var body = d3.selectAll("body");
```

If the selector is not a string, then it selects the specified array of nodes, which is defined below.

Example: Consider the following script.

```
d3.selectAll("body").style("color", "red");
```

selection.selectAll(selector)

This method is used to select an element. It selects the descendant elements that match the specified selector string. The elements in the returned selection are grouped by their corresponding parent node in this selection. If no element matches the specified selector for the current element, or if the selector is null, the group at the current index will be empty.

Example: Consider the following script.

```
var b = d3.selectAll("p").selectAll("b");
```

selection.filter(filter)

This method is used to filter the selection, returning a new selection that contains only the elements for which the specified filter is true.

Example: Consider the following script.

```
var even = d3.selectAll("tr").filter(":nth-child(odd)");
```

Here, filter a selection of table rows returns only odd.

selection.merge(other)

This method is used to return a new selection merging with the specified other selection.

Example: Consider the following script.

```
var rect = svg.selectAll("rect").data(data);
rect.enter().append("rect").merge(rect);
```

d3.matcher(selector)

This method is used to assign the specified selector. It returns a function, which returns true.

Example: Consider the following script.

```
var p = selection.filter(d3.matcher("p"));
```

d3.creator(name)

This method is used to assign the specified element name It returns a function, which creates an element of the given name, assuming that this is the parent element.

Example: Consider the following script.

```
selection.append(d3.creator("p"));
```

d3.each(function)

This method is used to invoke the specified function for each selected element, in the order passed by the current datum (d), the current index (i) and the current group (nodes) with this as the current DOM element (nodes[i]). It is explained below.

Example: Consider the following script.

```
parent.each(function(p, j) {
   d3.select(this)
      .selectAll(".child")
      .text(function(d, i) {
               return "child " + d.name + " of " + p.name;
       });
});
```

selection.call(function[, arguments…])

It is used to invoke the specified function exactly once. The syntax is shown below.

Example: Consider the following script.

```
function name(selection, first, last) {
   selection.attr("first-name", first).attr("last-name", last);
}
```

This method can be specified as shown below.

Example: Consider the following script.

```
d3.selectAll("p").call(name, "Adam", "David");
```

d3.local()

D3 local allows you to define the local state that is independent of data.

Example: Consider the following script.

```
var data = d3.local();
```

Unlike var, the value of each local is also scoped by the DOM.

local.set(node, value)

This method sets the value of this local on the specified node to the value.

Example: Consider the following script.

```
selection.each(function(d)
   { data.set(this, d.value); });
local.get(node)
```

This method returns the value of this local on the specified node. If the node does not define this local, then it returns the value from the nearest ancestor that defines it.

local.remove(node)

This method deletes this local's value from the specified node. It returns true, if the node defined, otherwise returns false.