

Filtering and Sorting



Data Format

```
data = [row1, row2, ..., rowN]
```

Data Format

`data = [row1, row2, ..., rowN]`



```
{  
  Name: "John Doe",  
  Age: 25  
}
```

Filter

```
data = [row1, row2, ..., rowN]
```

JS API | *array.filter*

Syntax:

```
array.filter(func)
```

Example

```
data.filter(function(client) {  
  return client.age < 25  
})
```

Sorting

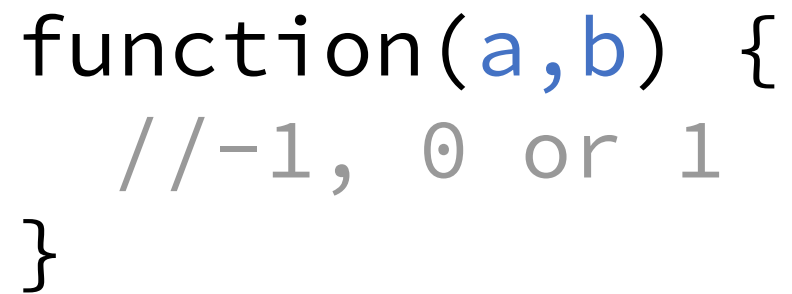
```
data = [row1, row2, ..., rowN]
```

order by age

JS API | *array.sort*

Syntax:

`array.sort(comparator)`



A diagram showing a function definition for the comparator parameter. A box contains the code `function(a,b) {` on the first line, `// -1, 0 or 1` on the second line, and `}` on the third line. An arrow points from the top of the box to the `comparator` parameter in the syntax line above.

```
function(a,b) {  
    // -1, 0 or 1  
}
```


API | *d3.ascending*
d3.descending

Syntax:

```
array.sort(d3.ascending)
```

API | *d3*.ascending *d3*.descending

Syntax:

```
array.sort(d3.ascending)
```

```
array.sort(function(a,b) {  
    return d3.descending(a,b)  
})
```

Example

```
data.sort(function(c1, c2) {  
    return d3.ascending(  
        c1.age,  
        c2.age  
    )  
})
```

Example

```
data.sort(function(c1, c2) {  
    return d3.ascending(  
        c1.age,  
        c2.age  
    )  
})
```

Filtering and Sorting

Native JavaScript Functions

D3 helpers