2장

d3js 의 시작

cdn파일 호출

```
 <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
    <script src="http://d3js.org/d3.v3.min.js" charset="UTF-8"></script>
```

myGraph svg는 초기값으로 width 300px height 150px이 적용되는듯함.
그래프를 이 바깥 테두리로 그리면 나타나지 않기 때문에
myGraph 범위 밖으로 그릴려면
svg id=myGraph 인 것을 강제로 크게 키워야함.
예제)

```
<svg id="myGraph" style="width:600px; height:600px;">
</svg>
```

2.1

예제
sample01.js

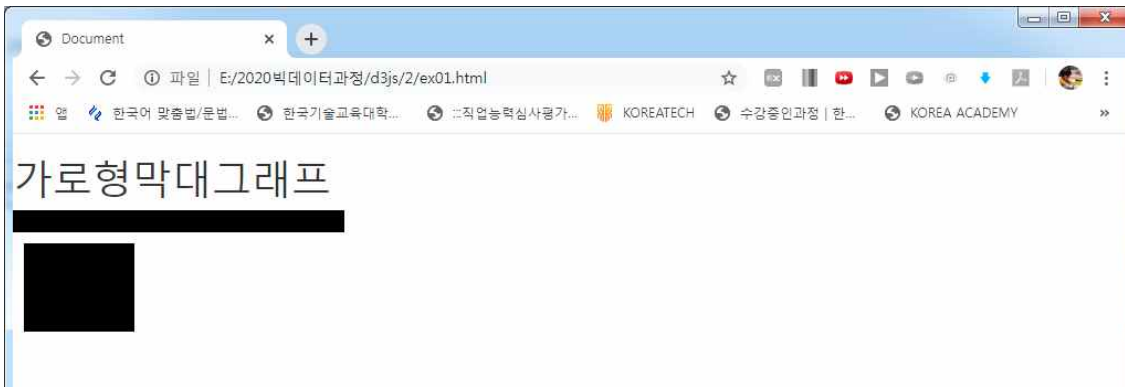
```
$(document).ready(function(){
// 막대그래프 데이터(데이터 세트)
var dataSet = [300, 130, 5, 60, 240];
// 데이터를 기반으로 그리기
d3.select("#myGraph")    // SVG 요소를 지정
        .append("rect")  // SVG 사각형 생성
        .attr("x", 0)       // 가로형 막대그래프이므로 X 좌표를 0으로 설정
        .attr("y", 0)       // Y 좌표를 0으로 설정
        .attr("width", dataSet[0]) // 최초 데이터를 기반으로 넓이 설정
        .attr("height", "20px")     // 막대그래프의 높이는 20px로 지정

});
```

예제
ex01.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
    <script src="http://d3js.org/d3.v3.min.js" charset="UTF-8"></script>
    <script src="sample.js"></script>
</head>
<body>
    <h1>가로형막대그래프</h1>
    <svg id="myGraph">
        <rect x="10" y="30" width="100" height="80"/>
    </svg>
</body>
</html>
```



2.2

예제
여러 개의 그래프 표시

sample.js

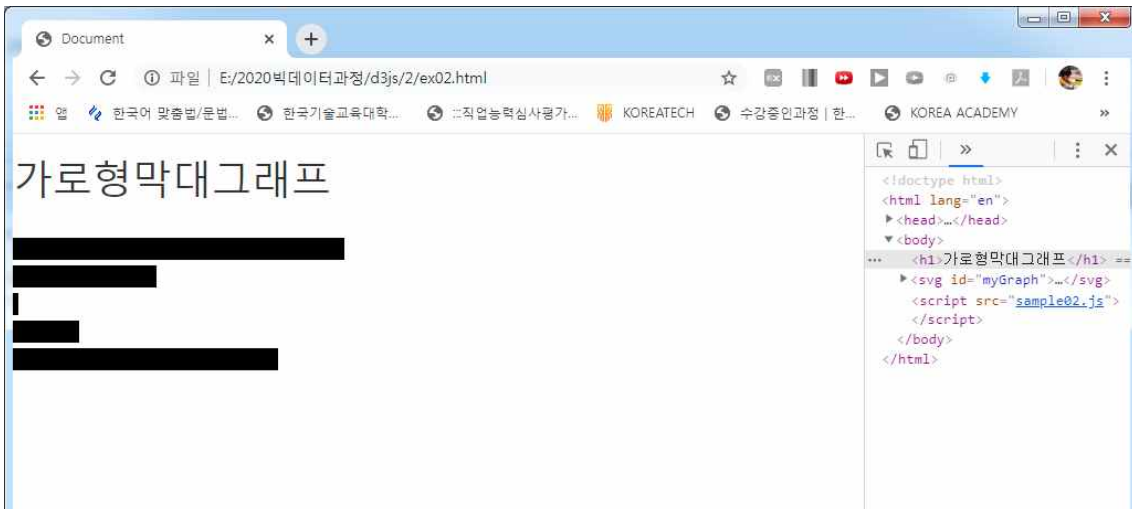
```
// 막대그래프 데이터(데이터 세트)
var dataSet = [300, 130, 5, 60, 240];
// 데이터를 기반으로 그리기
d3.select("#myGraph")
        .append("rect")
        .attr("x", 0)
        .attr("y", 25)
        .attr("width", dataSet[0])
        .attr("height", "20px")
d3.select("#myGraph")
        .append("rect")
        .attr("x", 0)
        .attr("y", 50)
        .attr("width", dataSet[1])
        .attr("height", "20px")
d3.select("#myGraph")
        .append("rect")
        .attr("x", 0)
        .attr("y", 75)
        .attr("width", dataSet[2])
        .attr("height", "20px")
d3.select("#myGraph")
        .append("rect")
        .attr("x", 0)
        .attr("y", 100)
        .attr("width", dataSet[3])
        .attr("height", "20px")
d3.select("#myGraph")
        .append("rect")
        .attr("x", 0)
        .attr("y", 125)
        .attr("width", dataSet[4])
        .attr("height", "20px")
```

예제
ex02.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
    <script src="http://d3js.org/d3.v3.min.js" charset="UTF-8"></script>

</head>
<body>
    <h1>가로형막대그래프</h1>
    <svg id="myGraph">
        <!-- <rect x="10" y="30" width="100" height="80"/> -->
    </svg>
    <script src="sample02.js"></script>
</body>
</html>
```



지금까지는 append 함수로 rect를 만들어 넣었는데 조금 더 편하게 하기 위해서 enter() 함수를 이용하에 data수에 따라 rect 요소를 생성 하도록 한다.

속성을 줄때에 y의 값은

attr(“y”, function(d,i){ retrun i*25;})

넣어서 긴 막대 그래프가 y 좌표가 25씩 증가하여 나타나도록 한다.

그리고 width 의 값은
data 값 안에 있는 것을 주기 위하여 함수 인자인 d를 이용하도록 한다.

.attr("width", function(d, i){ return d +"px“})

예제

sample03.js

```
// 막대그래프 데이터(데이터 세트)
var dataSet = [300, 130, 5, 60, 240];
// 데이터를 기반으로 그리기
d3.select("#myGraph")                          // SVG 요소 지정
        .selectAll("rect")                             // SVG로 사각형을 표시할
요소를 지정
        .data(dataSet)                                 // 데이터 설정
        .enter()                                       // 데이터 수에 따라
rect 요소 생성
        .append("rect")                                // SVG 사각형 생성
        .attr("x", 0)                                  // 가로형
막대그래프이므로 X좌표를 0으로 함
        .attr("y", function(d,i){                      // Y 좌표를 배열의 순서에 따라 계산
                return i * 25;                         // 막대그래프의
높이를 25px 단위로 계산
        })
        .attr("width", function(d, i){       // 넓이를 배열의 내용에 따라 계산
                return d +"px";                        // 데이터의 값을
그대로 넓이로 함
        })
        .attr("height", "20px")                        // 막대그래프의 높이를 20px로 지정
```

ex03.html

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
    <script src="http://d3js.org/d3.v3.min.js" charset="UTF-8"></script>

</head>
<body>
    <h1>가로형막대그래프</h1>
    <svg id="myGraph" style="width: 500px; height: 600px;">
    </svg>
    <script src="sample03.js"></script>
</body>
</html>
```

2.2 막대그래프의 스타일 지정

스타일 지정 종류
fill 도형을 채울 색을 지정 red나 rgb(0,255,0),등을 지정할수 있음.
stroke 도형의 선 색을 지정 red나 rgb(0,255,0)등을 지정할 수 있음.
stroke-width 도형의 선 넓이를 지정

예제

sample04.js

```
// 막대그래프 데이터(데이터 세트)
var dataSet = [300, 130, 5, 60, 240];
// 데이터를 기반으로 그리기
d3.select("#myGraph")                                   // SVG 요소 지정
        .selectAll("rect")                                      // SVG로 사각형을 표시할
요소를 지정
        .data(dataSet)                                                  // 데이터 설정
        .enter()                                                                //
```

```
데이터 수에 따라 rect 요소 생성
        .append("rect")                                         // SVG 사각형 생성
        .attr("x", 0)                                                   // 가로형
막대그래프이므로 X좌료를 0으로 함
        .attr("y", function(d,i){                       // Y 좌표를 배열의 순서에 따라 계산
                return i * 25;                                          // 막대그래프의
높이를 25px 단위로 계산
        })
        .attr("width", function(d, i){          // 넓이를 배열의 내용에 따라 계산
                return d +"px";                                         // 데이터의 값을
그대로 넓이로 함
        })
        .attr("height", "20px")                         // 막대그래프의 높이를 20px로 지정
```

ex04.html

```
<!DOCTYPE html>
<html>
        <head>
                <meta charset="utf-8">
                <title>가로형 막대그래프</title>
                <style>
                        svg { width: 320px; height: 240px; border: 1px solid black; }
                        #myGraph rect {
                                stroke : rgb( 160, 0, 0 );
                                stroke-width : 1px;
                                fill : rgb( 255, 0, 0 );
                        }
                </style>
                <script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
        </head>
        <body>
                <h1>가로형 막대그래프</h1>
                <svg id="myGraph"></svg>
                <script src="sample04.js"></script>
        </body>
</html>
```

2.3 막대그리프의 데이터 값 변경하기

sample05.js

```
// 막대 그래프 데이터(데이터셋)
var dataSet = [300, 130, 5, 60, 240];
// 데이터를 기반으로 그리기
d3.select("#myGraph")              // SVG 요소 지정
        .selectAll("rect")         // SVG로 사각형을 표시할 요소를 지정
        .data(dataSet)             // 데이터 설정
        .enter()                   // 데이터 수에 따라 rect 요소 생성
        .append("rect")            // SVG 사각형 생성
        .attr("x", 0)              // 가로형 막대그래프이므로 X좌표를 0으로 함
        .attr("y", function(d,i){  // Y 좌표를 배열의 순서에 따라 계산
                return i * 25;     // 막대그래프의 높이를 25px 단위로 계산
        })
        .attr("width", function(d, i){      // 넓이를 배열의 내용에 따라 계산
                return d +"px";             // 데이터의 값을 그대로 넓이로 함
        })
        .attr("height", "20px")    // 막대그래프의 높이를 20px로 지정

// 버튼 클릭 시의 처리
d3.select("#updateButton")
        .on("click", function(){
                dataSet = [20, 230, 150, 10, 20]; // 새로운 데이터로 변경
                d3.select("#myGraph")              // SVG 요소 지정
                        .selectAll("rect")         // SVG로 사각형을 표시할 요소를 지정
                        .data(dataSet)             // 데이터 설정
                        .attr("width", function(d, i){ // 넓이를 배열의 내용에 따라 계산
                                return d +"px"; // 데이터의 값을 그대로 넓이로 함
                        })
})
```

ex05.html

```
<!DOCTYPE html>
<html>
        <head>
                <meta charset="utf-8">
                <title>가로형 막대그래프</title>
                <style>
```

```
                #myGraph rect {
                        stroke : rgb( 160, 0, 0 );
                        stroke-width : 1px;
                        fill : rgb( 255, 0, 0 );
                }
        </style>
        <script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
</head>
<body>
        <h1>가로형 막대그래프</h1>
        <button id="updateButton">  데이터 업데이트  </button>
        <svg id="myGraph"></svg>
        <script src="sample05.js"></script>
</body>
</html>
```

2.4 그래프 애니메이션

sample06.js

```
// 막대 그래프 데이터(데이터셋)
var dataSet = [300, 130, 5, 60, 240];
// 데이터를 기반으로 그리기
d3.select("#myGraph")                   // SVG 요소 지정
        .selectAll("rect")              // SVG로 사각형을 표시할 요소를 지정
        .data(dataSet)                  // 데이터 설정
        .enter()                        // 데이터 수에 따라 rect 요소 생성
        .append("rect")                 // SVG 사각형 생성
        .attr("x", 0)                   // 가로형 막대그래프이므로 X좌표를 0으로 함
        .attr("y", function(d,i){       // Y 좌표를 배열의 순서에 따라 계산
                return i * 25;          // 막대그래프의 높이를 25px 단위로 계산
        })
        .attr("width", function(d, i){  // 넓이를 배열의 내용에 따라 계산
                return d +"px";         // 데이터의 값을 그대로 넓이로 함
        })
        .attr("height", "20px")         // 막대그래프의 높이를 20px로 지정

// 버튼 클릭 시의 처리
d3.select("#updateButton").on("click", function(){
        for(var i=0; i<dataSet.length; i++){
                dataSet[i] = Math.floor(Math.random() * 320);    // 0~320 미만의
```

```
값을 생성
        }
        d3.select("#myGraph")            // SVG 요소 지정
                .selectAll("rect")                  // SVG로 사각형을 표시할 요소를 지정
                .data(dataSet)                      // 데이터 설정
                .transition()                       // 변환 표시
                .attr("width", function(d, i){      // 넓이를 배열의 내용에 따라 계산
                        return d +"px";             // 데이터의 값을 그대로 넓이로 함
                })
})
```

ex06.html

```
<!DOCTYPE html>
<html>
        <head>
                <meta charset="utf-8">
                <title>가로형 막대그래프</title>
                <style>
                        #myGraph rect {
                                stroke : rgb( 160, 0, 0 );
                                stroke-width : 1px;
                                fill : rgb( 255, 0, 0 );
                        }
                </style>
                <script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
        </head>
        <body>
                <h1>가로형 막대그래프</h1>
                <button id="updateButton">  데이터 업데이트  </button>
                <svg id="myGraph"></svg>
                <script src="sample06.js"></script>
        </body>
</html>
```

그래프 효과시 딜레이시간 다르게 하기

sample07.js

```
// 막대 그래프 데이터(데이터셋)
var dataSet = [300, 130, 5, 60, 240];
```

```
// 데이터를 기반으로 그리기
d3.select("#myGraph")                          // SVG 요소 지정
        .selectAll("rect")                     // SVG로 사각형을 표시할 요소를 지정
        .data(dataSet)                         // 데이터 설정
        .enter()                               // 데이터 수에 따라 rect 요소 생성
        .append("rect")                        // SVG 사각형 생성
        .attr("x", 0)                          // 가로형 막대그래프이므로 X좌표를
0으로 함
        .attr("y", function(d,i){              // Y 좌표를 배열의 순서에 따라 계산
                return i * 25;                 // 막대그래프의 높이를 25px 단위로 계산
        })
        .attr("height", "20px")                // 막대그래프의 높이를 20px로 지정
        .attr("width", "0px")                  // 최초 막대그래프의 넓이를 0px로 지정
        .transition()                          // 그래프 출력 시 애니메이션 효과 적용
        .delay(function(d, i){
                return i * 500;                // 0.5초마다 그리도록 대기 시간을 설정
        })
        .duration(2500)                        // 2.5초에 걸쳐 애니메이션화 함
        .attr("width", function(d, i){         // 넓이를 배열 내용에 따라 계산
                return d +"px";                // 데이터의 수를 그대로 넓이로 함
        })
```

위와 같이 마지막에 width를 표시해서 0부터 각각의 데이터의 width 까지 2.5초시간동안 변하게 한다.
duration 함수 다음에 attr로 width값을 표시 해서 순서대로 적는게 중요하다
width값이 초기값이 없으면 transition 효과가 적용되지 않는 것을 볼 수 있다.

```
// 버튼 클릭 시의 처리
d3.select("#updateButton").on("click", function(){
        for(var i=0; i<dataSet.length; i++){
                dataSet[i] = Math.floor(Math.random() * 320);    // 0~320 미만의
값을 생성
        }
        d3.select("#myGraph")                  // SVG 요소 지정
                .selectAll("rect")                     // SVG로 사각형을 표시할 요소를 지정
                .data(dataSet)                                 // 데이터 설정
                .transition()                                          // 변환 표시
                .attr("width", function(d, i){         // 넓이를 배열의 내용에 따라 계산
                        return d +"px";                                        // 데이터의
값을 그대로 넓이로 함
                })
```

})

ex07.html

```
<!DOCTYPE html>
<html>
	<head>
		<meta charset="utf-8">
		<title>가로형 막대그래프</title>
		<style>
			#myGraph rect {
				stroke : rgb( 160, 0, 0 );
				stroke-width : 1px;
				fill : rgb( 255, 0, 0 );
			}
		</style>
		<script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
	</head>
	<body>
		<h1>가로형 막대그래프</h1>
		<button id="updateButton">  데이터 업데이트  </button>
		<svg id="myGraph"></svg>
		<script src="sample07.js"></script>
	</body>
</html>
```

마우스 이벤트로 그래프 색 바꾸기

transition()뒤에는 delay()속성 duration()속성등이 올 수 있고 on(“click”,function(){});은 올 수 없다.

sample08.js

```
// 막대 그래프 데이터(데이터셋)
var dataSet = [300, 130, 5, 60, 240];
// 데이터를 기반으로 그리기
d3.select("#myGraph")                                   // SVG 요소 지정
	.selectAll("rect")                                      // SVG로 사각형을 표시할 요소를 지정
	.data(dataSet)                                          // 데이터 설정
	.enter()                                                //
데이터 수에 따라 rect 요소 생성
	.append("rect")                                         // SVG 사각형 생성
```

```
        .attr("x", 0)                                                    // 가로형
막대그래프이므로 X좌표를 0으로 함
        .attr("y", function(d,i){                          // Y 좌표를 배열의 순서에 따라 계산
                return i * 25;                                          // 막대그래프의
높이를 25px 단위로 계산
    })
    .attr("height", "20px")                                // 막대그래프의 높이를 20px로 지정
    .attr("width", "0px")
    .on("click",function(){
        // alert(10);
        d3.select(this)
        .style("fill","cyan")
    })
    .transition()
    .delay(function(d,i){
        return i*500;
    })
    .duration(function(d,i){
        return 2500;
    })
    .attr("width", function(d,i){
        return d+"px";
    })



// 버튼 클릭 시의 처리
d3.select("#updateButton").on("click", function(){
        for(var i=0; i<dataSet.length; i++){
                dataSet[i] = Math.floor(Math.random() * 320);    // 0～320 미만의
값을 생성
        }
        d3.select("#myGraph")              // SVG 요소 지정
                .selectAll("rect")                    // SVG로 사각형을 표시할 요소를 지정
                .data(dataSet)                                  // 데이터 설정
        .transition()                                           // 변환 표시
        .delay(function(d,i){
            return i*500;
        })
        .duration(function(d,i){
            return 2500;
```

```
        })
                .attr("width", function(d, i){       // 넓이를 배열의 내용에 따라 계산
                        return d +"px";                                    // 데이터의
값을 그대로 넓이로 함
        })

})
```

ex08.html

```
<!DOCTYPE html>
<html>
        <head>
                <meta charset="utf-8">
                <title>가로형 막대그래프</title>
                <style>
                        #myGraph rect {
                                stroke : rgb( 160, 0, 0 );
                                stroke-width : 1px;
                                fill : rgb( 255, 0, 0 );
                        }
                </style>
                <script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
        </head>
        <body>
                <h1>가로형 막대그래프</h1>
                <button id="updateButton">  데이터 업데이트  </button>
                <svg id="myGraph"></svg>
                <script src="sample08.js"></script>
        </body>
</html>
```

외부 데이터 불러오기

크롬브라우저에서 보안 때문에 파일을 못 불러오게 되어 있는 경우가 있는데

```
C:\Program Files (x86)\Google\Chrome\Application\
chrome.exe -disable-web-security -allow-file-access-from-files
```

이렇게 실행하게 되면 보안 끄고 실행함으로써 잘 작동되는 것을 볼 수 있다.

예제

sample09.js

```
// 막대 그래프 데이터(데이터셋)
var dataSet = [];

d3.csv("./mydata.csv", function(error,data){
    for (let i = 0; i < data.length; i++) {
        dataSet.push(data[i].item1);
        console.log("test");
    }
    // 데이터를 기반으로 그리기
d3.select("#myGraph")                          // SVG 요소 지정
.selectAll("rect")                             // SVG로 사각형을 표시할 요소를 지정
.data(dataSet)                                       // 데이터 설정
.enter()                                                          // 데이터
수에 따라 rect 요소 생성
.append("rect")                                      // SVG 사각형 생성
.attr("x", 0)                                               // 가로형
막대그래프이므로 X좌표를 0으로 함
.attr("y", function(d,i){                  // Y 좌표를 배열의 순서에 따라 계산
    return i * 25;                                   // 막대그래프의 높이를 25px
단위로 계산
})
.attr("height", "20px")                    // 막대그래프의 높이를 20px로 지정
.attr("width", "0px")
.on("click",function(){
    // alert(10);
    d3.select(this)
    .style("fill","cyan")
})
.transition()
.delay(function(d,i){
    return i*500;
})
.duration(function(d,i){
    return 2500;
})
```

```
.attr("width", function(d,i){
    return d+"px";
})



// 버튼 클릭 시의 처리
d3.select("#updateButton").on("click", function(){
for(var i=0; i<dataSet.length; i++){
    dataSet[i] = Math.floor(Math.random() * 320);        // 0~320 미만의 값을 생성
}
d3.select("#myGraph")              // SVG 요소 지정
    .selectAll("rect")                    // SVG로 사각형을 표시할 요소를 지정
    .data(dataSet)                        // 데이터 설정
    .transition()                         // 변환 표시
    .delay(function(d,i){
        return i*500;
    })
    .duration(function(d,i){
        return 2500;
    })
    .attr("width", function(d, i){  // 넓이를 배열의 내용에 따라 계산
        return d +"px";                   // 데이터의 값을 그대로
넓이로 함
    })

})
});
```

ex09.html

```
<!DOCTYPE html>
<html>
        <head>
                <meta charset="utf-8">
                <title>가로형 막대그래프</title>
                <style>
                        #myGraph rect {
                                stroke : rgb( 160, 0, 0 );
                                stroke-width : 1px;
                                fill : rgb( 255, 0, 0 );
```

```
                }
            </style>
            <script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
    </head>
    <body>
            <h1>가로형 막대그래프</h1>
            <button id="updateButton">  데이터 업데이트  </button>
            <svg id="myGraph"></svg>
            <script src="sample09.js"></script>
    </body>
</html>
```

눈금표시

눈금스타일 설정

우선 눈금의 수치나 레이블, 선 스타일을 설정합니다. 코드에서 직접 설정할 수도 있습니다만, 디자이너 등과의 분업을 고려하면 스타일은 분리해두는 것이 나중을 위해 좋습니다.
스타일은 HTML 파일안에 작성합니다. 물론 별도의 css파일로 작성한 다음 이를 불러오는 방식을 택해도 됩니다.

눈금의 종류와 스케일 설정

눈금을 표시하려면 몇가지 과정이 필요합니다. 우선, 사용할 눈금의 종류를 결정해야 합니다. 직선이나 로그 등이 있으나 이번 막대그래프는 간단하고 직선 눈금입니다.
이것은 선형 스케일이므로 D3.js에서는 d3.scale.linear()를 이용하여 생성할 수 있습니다.

다음 설정해야 하는 것은 스케일/척도입니다. 이것은 데이터셋에 있는 데이터의 수치범위(원래 데이터의 범위)와 실제로 표시할 표시 범위를 조정하는 것이 됩니다. 원래 데이터 범위는 domain() 메서드로, 표시할 범위는 range()메서드로 지정합니다. 두 메서드 모두 파라미터로는 [시작값,종료값]과 같은 형식의 배열을 지정합니다. 예를 들어 원래 데이터 범위가 0~10일 때 그대로를 넓이로 사용한다면 10픽셀 정도로 아주 작게 표시될 것입니다. 그러므로 표시할 범위를 0~100으로 조정하면 크게 표시할 수가 있습니다.

이번의 그래프는 데이터 값과 표시할 범위가 1:1이므로 domain(),range() 모두 같은 범위를 지정합니다. 이번의 그래프 데이터는 가장 큰값이 300까지 이므로 다음과 같이 작성하여 눈금과 대응하도록 합니다.

sample10.js

```
// CSV 파일을 불러와 그래프 그리기
```

```
d3.csv("mydata.csv", function(error, data){
        var dataSet = [ ];        // 데이터를 저장할 배열 준비
        for(var i=0; i<data.length; i++){  // 데이터 줄 수만큼 반복
                dataSet.push(data[i].item1);       // item1 레이블의 데이터만 추출
        }
        // 데이터를 이용하여 그래프 그리기
        d3.select("#myGraph")   // SVG 요소 지정
                .selectAll("rect") // SVG에서 사각형을 나타낼 요소를 지정
                .data(dataSet)   // 데이터 설정
                .enter() // 데이터 개수에 때라 Rect 요소 생성
                .append("rect")  // SVG 사각형 생성
                .attr("x", 10)     // 가로형 막대그래프이므로 X 좌표를 10으로 조정
                .attr("y", function(d,i){   // Y 좌표를 배열 순서에 따라 계산
                        return i * 25;  // 막대그래프의 Y 좌표를 25px 단위로 계산
                })
                .attr("height", "20px")     // 막대그래프의 높이를 20px로 지정
                .attr("width", function(d, i){       // 막대그래프의 넓이를 배열의 내용에
따라 계산
                        return d +"px"; // 데이터의 값을 그대로 넓이로 함
                })
                // 눈금을 표시하고자 선형 스케일을 설정
                var xScale = d3.scale.linear()  // 선형 스케일 설정
                        .domain([0, 300])   // 원래 데이터 범위
                        .range([0, 300])  // 실제 출력 크기
                // 눈금을 설정하고 표시
                d3.select("#myGraph")
                        .append("g")      // 그룹화함
                        .attr("class", "axis")       // 스타일시트 클래스 설정
                        .attr("transform", "translate(10, "+((1+dataSet.length) *
20+5)+")")       // 표시 위치 조정
                        .call(d3.svg.axis()         // call()로 눈금을 표시할 함수를 호출
                                .scale(xScale)  // 스케일을 적용
                                .orient("bottom")   // 눈금의 표시 위치를 아래쪽으로
지정
                        )
});
```

ex10.html

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

```
		<meta charset="utf-8">
		<title>가로형 막대그래프</title>
		<style>
			#myGraph rect {
				stroke : rgb( 160, 0, 0 );
				stroke-width : 1px;
				fill : rgb( 255, 0, 0 );
			}
			.axis text {
				font-family: sans-serif;
				font-size: 11px;
			}
			.axis path,
			.axis line {
				fill: none;
				stroke: black;
			}
		</style>
		<script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
	</head>
	<body>
		<h1>가로형 막대그래프</h1>
		<svg id="myGraph"></svg>
		<script src="sample10.js"></script>
	</body>
</html>
```

눈금 표시

눈금 설정이 끝났으므로 실제로 화면에 표시하는 코드를 작성해보겠습니다. 눈금을 표시하려면 SVG 요소를 그룹화하는 기능을 사용하여 눈금의 선, 숫자와 그래프를 하나의 세트로 처리합니다. 그룹화 하면 눈금을 한 번에 이동하는 등을 간단하게 처리할 수 있게 됩니다. 그룹화하려면 SVG의 g요소를 사용합니다. g요소를 사용하면 g요소안에 포함된 여러 개의 요소를 하나의 그룹으로 다룰 수 있게 됩니다.

D3.js로 SVG의 g요소를 추가하려면 append(“g”)라고 하면 됩니다. 막대그래프의 사각형인 rect 요소를 추가할때도 마찬가지입니다.

다음으로 g요소에 스타일을 추가합니다만, 앞의 ‘눈금 스타일 설정’에서 설명한 axis라는 이름의 스타일시트 정의를 사용합니다. 정의한 스타일 시트는 g요소의 class 속성에 aixs 문자를 설정하면 반영할 수 있습니다. 즉 attr(“class”,“axis”)라고 하면 g요소에 스타일

시트를 적용할 수 있습니다.

눈금을 표시하려면 다음과 같이 call()메서드를 사용합니다. call()메서드는 파라미터에 설정한 함수를 호출하는 메서드입니다. call() 메서드로 지정하는 함수에는 처리 대상이 되는 모든 요소를 전달합니다. 여기서는 g요소가 대상이므로 call() 메서드로 호출할 함수에는 g요소 하나만 전달하게 됩니다.

```
.call(d3.svg.axis()          // call()로 눈금을 표시할 함수를 호출
    .scale(xScale)  // 스케일을 적용
    .orient("bottom")    // 눈금의 표시 위치를 아래쪽으로 지정
)
```

d3.svg.axis()가 눈금을 처리하는 메서드입니다. 다음에 오는 scale() 메서드의 파라미터에 앞서 설정한 선형 스케일(d3.scale.linear())의 객체를 지정합니다.
이것만으로도 눈금을 표시할 수 있습니다만, 추가로 orient() 메서드를 사용하여 눈금의 표시 위치를 지정해두었습니다.(여기서는 bottom)

실제로 이 단계에서 표시해보면 눈금 위치가 조금 어긋난 상태로 표시됩니다. 그러므로 막대그래프와 눈금을 오른쪽으로 조금 이동하도록 합니다. 막대그래프일 때는 rect요소의 x좌표를 대상으로 attr(“x”,10)이라고 하면 됩니다.

눈금은 x,y 속성을 설정하면 원하는 위치로 이동합니다. 눈금의 위치를 이동하려면 transform 속성에 이동할 거리를 지정해야 합니다. 예를 들어 X방향으로 10픽셀 Y방향으로 20필셀 이동하려면 .attr(“transform”,“translate(10,20)”)과 같이 지정하면 됩니다. translate()는 파라미터에 지정한 만큼 XY좌표를 이동시킵니다. 처음 값이 X좌표, 두 번째 값이 Y좌표의 이동거리입니다. 또한 transform 속성을 조작하는데에는 확대 비율을 지정하는 scale(), 회전 각도를 지정하는 roteta(), 기울기를 지정하는 skewX(), skewY() 등이 있습니다.

모든 기능 정리

예제

sample11.js

```
// 데이터를 저장할 배열 준비
var dataSet = [ ];
// CSV 파일을 불러와 그래프 그리기
d3.csv("mydata.csv", function(error, data){
        for(var i=0; i<data.length; i++){  // 데이터 줄 수만큼 반복
                dataSet.push(data[i].item1);        // item1 레이블의 데이터만 추출
```

```
        }
        // 데이터를 이용하여 그래프 그리기
        d3.select("#myGraph")   // SVG 요소 지정
                .selectAll("rect") // SVG에서 사각형을 나타낼 요소를 지정
                .data(dataSet)   // 데이터 설정
                .enter() // 데이터 개수에 따라 Rect 요소 생성
                .append("rect")  // SVG 사각형 생성
                .attr("x", 10)     // 가로형 막대그래프이므로 X 좌표를 10으로 조정
                .attr("y", function(d,i){   // Y 좌표를 배열 순서에 따라 계산
                        return i * 25;  // 막대그래프의 Y 좌표를 25px 단위로 계산
                })
                .on("click", function(){
                        d3.select(this)   // 클릭한 요소를 지정
                                .style("fill", "cyan")        // 채우기 스타일을 하늘색으로
지정
                })
                .attr("height", "20px")     // 막대그래프의 높이를 20px로 지정
                .attr("width", "0px")       // 처음에는 막대그래프의 넓이를 0px로 지정
                .transition()      // 그래프에 애니메이션 적용
                .delay(function(d, i){
                        return i * 500; // 0.5초마다 그리도록 대기 시간을 설정
                })
                .duration(2500) // 2.5초 동안 애니메이션이 진행되도록 함
                .attr("width", function(d, i){       // 넓이를 배열의 내용에 따라 계산
                        return d +"px"; // 데이터의 값을 그대로 넓이로 함
                })
        // 눈금을 표시하기 위한 선형 스케일 설정
        var xScale = d3.scale.linear()  // 선형 스케일 설정
                .domain([0, 300])   // 원래 데이터 범위
                .range([0, 300]);  // 실제 출력 크기
        // 눈금을 설정하고 표시함
        d3.select("#myGraph")
                .append("g")      // 그룹화함
                .attr("class", "axis")       // 스타일시트 클래스를 설정
                .attr("transform", "translate(10, "+((1+dataSet.length) * 20+5)+")") //
표시 위치 조정
                .call(d3.svg.axis()         // call()로 눈금을 표시할 함수를 호출
                        .scale(xScale)  // 스케일을 적용
                        .orient("bottom")   // 눈금의 표시 위치를 아래쪽으로 지정
                )
});
```

```
// 버튼을 클릭했을 때의 처리
d3.select("#updateButton").on("click", function(){
	for(var i=0; i<dataSet.length; i++){
		dataSet[i] = Math.floor(Math.random() * 320);    // 0～320미만의 값을
생성
	}
	d3.select("#myGraph")    // SVG 요소 지정
		.selectAll("rect") // SVG에서 사각형을 나타낼 요소를 지정
		.data(dataSet)    // 데이터 설정
		.transition()      // 변환 표시
		.attr("width", function(d, i){       // 넓이를 배열의 내용에 따라 계산
			return d +"px"; // 데이터의 값을 그대로 넓이로 함
		})
})
```

ex11.html

```
<!DOCTYPE html>
<html>
	<head>
		<meta charset="utf-8">
		<title>가로형 막대그래프</title>
		<style>
			#myGraph rect {
				stroke : rgb( 160, 0, 0 );
				stroke-width : 1px;
				fill : rgb( 255, 0, 0 );
			}
			.axis text {
				font-family: sans-serif;
				font-size: 11px;
			}
			.axis path,
			.axis line {
				fill: none;
				stroke: black;
			}
		</style>
		<script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
	</head>
```

```
        <body>
                <h1>가로형 막대그래프</h1>
                <button id="updateButton">데이터 업데이트</button>
                <svg id="myGraph"></svg>
                <script src="sample11.js"></script>
        </body>
</html>
```