Chapter 2 JavaScript, jMath 와 jqPlot

통계 학습을 시작하기 전에 이 교재에서 사용될 통계 툴에 대한 사용법을 익히도록 하겠습니다. 통계는 SAS, SPSS, R 과 같은 통계전용 프로그램이 있고, Microsoft Excel 을 이용하여도 통계처리가 가능합니다. 하지만 Web 으로 정보를 주고받는 시대에서 이러한 방식들은 PC 에 프로그램을 설치하고 자료값에 대한 통계 결과를 갖고 Web 에 나타낼 수 있도록 작업을 해야 되기 때문에 작업이 번거롭고, 자료값이 변화 되거나 계산 방법이 바뀌게 된다면 처음 부터 다시 해야되기 때문에 불편합니다. 이를 보완하고자 Web 에서 바로 통계처리가 가능하도록 JavaScript 를 기반으로 하는 jMath 라는통계 패키지를 교재 개발을 하면서 별도로 제작을 하게 되었습니다. JavaScript 으로 이러한 처리를 하게한 이유는 서버는 단순히 자료만 주면 되고 이를 받은 Web browser 에서 계산을 하면 되기 때문에 서버 부담이 줄일 수 있기 때문입니다.

도표를 그리기 위해서 사용된 jpPlot 은 Excel 에 차트와 같은 기능들을 Web browser 에서 바로 사용할 수 있게 해 줍니다. jqPlot 에는 다양한 형태의 차트를 생성할 수 있도록 많은 plugin 들을 제공하기 때문에 필요한 plugin 들에 대한 사용법을 알아야 합니다.

설명의 이해를 위해서는 JavaScript 언어에 대한 기초가 있어야 이해를 하실 수 있습니다. 하지만 이 교재에서는 JavaScript 의 기초부분에 대한 간단한 설명은 하지만 자세하게 다루지 않기 때문에 다른 교재를 보고 학습을 별도로 하셔야 합니다. 또한 jqPlot 은 jQuery 라는 JavaScript 기반의 fraemwork 으로 설계된 것이기 때문에 jqPlot 을 이해하시려면 jQuery 를 내용을 아셔야 하지만, 하지만 통계학습이 주 목적인 분들에게는 JavaScript 과 jQuery 를 모두 이해할 필요가 없이 간단한 JavaScript 문법만 이해하시면 됩니다.

1. JavaScript 기초

JavaScript 언어는 C, Java, C#과 같은 컴퓨터 언어와 유사한 점이 많기 때문에 다른 컴퓨터 언어를 알고 계시는 분들이면 학습하는데 큰 어려움이 없겠지만 깊숙히 내용을 다루다 보면 이 언어만의 특징들이 있어 별도의 학습이 필요합니다. 하지만 이 교재에서 사용될 JavaScript 는 일반적인 컴퓨터 언어의 기능들만 이용하기 때문에 다른 컴퓨터 언어를 알고 계시는 분들은 대충 읽어 보시면 됩니다.

이 교재의 예제들을 실행하기 위해서 Google chrome 을 설치해 주시기 바랍니다. 프로그램은 구글 홈페이지나 다른 검색 사이트에서 "구글 크롬"이라고 검색하시면 해당 사이트로 연결을 해 줄 것입니다. 아니면 주소창에 "http://www.google.com/chrome" 기입하셔서 설치를 하시면 됩니다.

1.1. Data

컴퓨터 프로그램은 자료를 입력받아 원하는 형태도 가공을 하여 결과를 만드는 것으로 컴퓨터 언어를 학습하기 위해서 자료의 형태와 쓰임세를 우선 이해를 해야 합니다.
JavaScript 에서 제공된는 기본 자료 유형들로 숫자(Number), 문자열(String), Boolean(불린), Array(배열), Object(객체)로 분류할 수 있습니다.
숫자는 정수와 실수 모두를 포함하고 JavaScript 은 모든 숫자를 실수로 처리합니다. 문자열은 "abc"와 같은 글을 말하는 것이고, Boolean 은 true 와 false 값입니다.
배열은 자료들의 집합으로 몇 번째 위치한 값을 읽고 쓰기를 하는 것으로 예를 들어 "a", "b", "c", "d", 1, 2, 3 와 같이 숫자와 문자열을 나열한 것을 말하고 2 번째 값은 "b"이다라고 표시합니다. 마지막으로 객체는 저장된 값들을 몇번째로 표기하는 것이 아니라 이름으로 표시하여 값을 읽고 씁니다. 예를 들어 a.address 는 a 라는 객체에 address 라는 이름으로 저장된 값을 말합니다.

Data 유형	예	typeof

Number(숫자)	> var a = 1 1	> typeof(a) "number"
String(문자열)	> a = "문자열" "문자열"	> typeof(a) "string"
Boolean(불린)	> a = true true	> typeof(a) "boolean"
Array(배열)	> a = [1,2,3] [1, 2, 3]	> typeof(a) "object"
Object(객체)	> a = { name: "홍길동", 학년: 3 } *Object {name: "홍길동", 학년: 3}*	> typeof(a) "object"
Function(함수)	> a = function(a1,a2) { return a1+a2; } function (a1,a2) { return a1+a2; }	> typeof(a) "function"
Null(널)	> a = null null	> typeof(a) "object"

예를 실행하기 위해서 google chrome 을 열고 Ctrl+Shift+J 를 누르시게 되면 화면 아래에 Console 이라는 창이 나타나고 거기에 위의 내용을 작성하시면 됩니다.

1) 숫자

```
var pos = 2
var neg = -2
var real = 3.141592
var pexp = 2e3      // 2x10^3 = 2000
var nexp = 2e-3    // 2x10^-3 = 0.002
```

숫자는 정수(Integer), 실수(Real, Float)을 받을 수 있습니다. 음수를 표현한 때는 숫자 앞에 –(minus)를 붙이기만 하면 됩니다. 양수는 별도로 표시가 없어도 되지만 숫자 앞에 +(plus)을 붙일 수 있습니다. 숫자에 사용되는 e 는 큰 숫자를 표시할 때 사용되는데 의미는 ex 는 10^X와 같습니다.

- 특수 값

값	설명

NaN	Not a Number 의 약자로 숫자가 아니라는 뜻입니다. 예를 들어 숫자와 숫자가 아닌 값과 곱셈과 같이 처리 할 수 없는 연산을 할 경우 이 값을 얻게 됩니다
Infinity	수학 기호 ∞에 해당하는 값으로 숫자를 0 으로 나눌 때 발생되는 값입니다.

2) Boolean

```
var t = true
var f = false
```

참(true)와 거짓(false) 두 개의 값만을 갖고 있습니다. true 는 숫자로 1 이고 false 는 0 에 해당 합니다. 주로 사용되는 곳은 나중에 배우게 될 조건식에서 조건을 만족하지를 나타날 때 사용되는 됩니다.

3) String(문자열)

```
var s1 = "문자열.  Abc"
var s2 = '다른 문자열 123'
```

문자(character)들이 모여서 만들어지는 데이터입니다. 문자열을 만들기 위해서는 " "(double quotes) 또는 ' '(single quotes)아무거나 사용이 가능합니다. 주의 할 점은 같은 quotation 으로 시작하여 끝나야 합니다. 만일 예를 들어 "aaa'와 같이 하면 오류가 발생 합니다.

- 특수 문자

문자열에서 new line(새줄), tab(탭)등과 같은 특수 문자를 넣기 위해서는 \(backslash)을 사용 해야 합니다.

코드	값	예
\n	다음 라인으로 이동하기 입니다.	> "abc\n123" "abc 123"

\r	현재 라인 맨 앞으로 이동하기입니다. 보통 "₩n"만 쓰던지 "₩r₩n"로 같이 사용합니다.	> "abc\r\n123" "abc 123"
\t	Tab 으로 문자 사이에 수개의 스페이스를 넣습니다.	> "abc\t123" "abc 123"
\"	"를 문자열에 넣습니다. ' '(single quotes)안에서는 그냥 "만 쓰면 됩니다.	> "abc\"123" "abc"123" > 'abc"123' "abc"123"
\'	'를 문자열에 넣습니다. " "(double quotes)안에서는 그냥 '만 쓰면 됩니다.	> "abc\'123" "abc'123" > "abc'123" "abc'123"
\u	Unicode 넣기로 반드시 4 자리의 16 진수 값을 넣어야 합니다.	> "\u0041" "A" > "\u4F03" "伃" > "abc\u0041efg" "abcAefg"
\b	Back space 를 넣습니다.	
\f	프린터에 Form feed 를 말하는 것으로 다음 페이지로 이동하라는 명령입니다.	

- 길이(length)

문자열(String)의 문자(Character)의 개수는 변수명.length, " ".length 또는 ' '.length 라고 하면 됩니다.

> var a = "문자열 a" undefined > a.length 5	> "문자열 a".length 5	> '문자열 a'.length 5

- String(문자열)에서 몇 번째 character(문자)읽기

변수명[index] 또는 " "[index]

Index 는 몇 번째 인가를 알려주는 값입니다. 하지만 index 는 0 부터 시작을 합니다. 즉 첫번째는 1 이 아니 0 입니다. 그래서 마지막 문자의 index 는 길이-1 입니다.

https://github.com/handuck/jMath

```
> var a = "문자열 12"
undefined
> a[0]
"문"
> a[2]
"열"
> a[4]
"1"
> a[a.length-1]
"2"
```

4) 배열(Array)

```
//숫자로만 구성된 array
var nlist = [ 1, 2, 3 ]
// 문자열로만 구성된 array
var slist = [ "apple", "kiwi", "orange" ]
// 숫자, 문자열 및 변수를 직접 넣을 수 있습니다.
var a = 1
var mlist = [ 1, "apple", 2, 3, "kiwi", true, a, nlist]
```

[](square brackets)안에 값들을 ","(comma)로 구분하여 표시하며 여러 개의 값을 순서대로 저장하고 있는 data 유형으로, array 에 있는 값을 element(요소)라고 말합니다. 원칙적으로 array 에 element 는 같은 data 유형 즉 데이터 크기가 동일해야 하지만 JavaScript 은 실제로 모든 data 유형을 다음에 배울 object 로 취급하기 때문을 typeof 로 나타나는 data 유형에 상관 없이 섞어서 사용이 가능합니다.

- Zero-based index : [index]

Array 에서 몇 번째 element 의 값을 얻기 위해서는 [](square brackets)안에 정수 값 index(인텍스)로 얻게 됩니다.

```
> var mlist = [ 1, "apple", 2, 3, "kiwi", true];
undefined
> mlist[1]
"apple"
```

https://github.com/handuck/jMath

```
> mlist[0]
1
```

보시는 것과 같이 문자열이 문자들이 모여서 생성된 것이기 때문에 배열과 유사합니다.

- 길이(length)

문자열과 같게 element(요소)의 개수는 변수명.length, [].length 라고 하면 됩니다.

```
> var mlist = [ 1, "apple", 2, 3];
undefined
> mlist.length
4
```

```
[ 1, "apple", 2, 3].length
4
```

- Element 값 수정

[index]를 하고 변수 값을 바꾸는 것과 같이 하면 됩니다.

```
> var mlist = [ 1, "apple", 2, 3, "kiwi", true];
undefined
> mlist[0] = 9
9
> mlist
[9, "apple", 2, 3, "kiwi", true]
```

하지만 문자열(string)의 경우는 이러한 방식으로는 수정이 안됩니다.

만약 index 를 현재 array 에 최대 값보다 크게 하여 값을 저장할 때 중간에 없는 값들은 undefined 로 자동으로 저장됩니다.

```
> var mlist = [ 'a', 'b', 1, 2 ];
undefined
> mlist[6] = 'c';
"c"
> mlist
["a", "b", 1, 2, undefined × 2, "c"]
```

변수 mlist 는 4 개의 element 들을 갖고 있는 array 로 index 최대 값은 3 입니다. 하지만 index 가 6 인 곳에 문자 "c"를 넣으면 index 4 와 index 5 에는 아무런 값이 저장되지 않았기에 undefined 가 자동으로 들어갑니다.

- n-dimensional array(n 차원 배열)

배열(array)내에 배열(array)을 갖고 있는 형태로 만일 array 내에 다른 array 가 없는 것은 1 차원 배열(1-dimensional array)라고 합니다.

```
> var dim2 = [ [1,2, "a"], [3,4, "b"] ]
Undefined
> dim2[0]
```

https://github.com/handuck/jMath

```
[1, 2, "a"]
> dim2[0][1]
2
> dim2.length
2
> dim2[1].length
3
```

위의 예는 2 차원 배열로 크기는 2X3 입니다. Array(배열)의 element(요소)값을 읽기 위해서는 [index]을 두 번 연속으로 사용 해야 합니다.

5) Object(객체): {}

Object 의 목적은 데이터 유형들을 갖고 조합을 하여 새로운 데이터 유형을 만드는데 목적이 있습니다. 예를 들어 회원 정보라는 데이터 유형이 필요한데 JavaScript 는 이러한 유형을 제공해주지 않습니다. 그래서 회원 정보 유형을 직접 만들어야 합니다.
그럼 회원 정보를 위해 이름(name), 태어난 날짜(birthDate), 휴대전화번호(mobile), 이메일(email)4 가지가 필요하다고 하면 다음과 같이 표현하여 값을 넣을 수 있습니다.

```
var acc1 = { name: "홍길동", birthdate: "1986-01-03"
          , mobile: "010-123-1234", email:"gildong@gmail.com" }
또는
var acc2 = { name: "홍길동", "birth Date": "1986-01-03"
          , mobile: "010-123-1234", "e-mail":"gildong@gmail.com" }
```

보시는 것과 같이 object 는 {}(curly brackets, braces)안에 key:값을 ","(comma)로 구분하여 값을 저장하는 data 유형입니다. Array 가 index 를 사용하듯이 object 는 key 를 사용합니다. 그래서 key 는 object 내에서 중복이 되어서는 안됩니다. 만일 key 에 이름에 공백이나 특수 문자가 있을 경우는 acc2 에 birth Date 과 e-mail key 처럼 ""나 ''로 표시하면 됩니다.

1.2. 연산자(Operator)

자료값을 받아 가공을 하기 위해서 필요한 것이 연산자 입니다. 연산자의 종류는 덧셈, 뺄셈, 곱셈, 나눗셈을 하는 산술연산자(arithmetic operator)들과 내용을 비교하는 비교연산자(comparison operator), 논리를 판단하는 논리연산자(logical operator)가 있고 그 외에 비트를 다루는 비트연산자(bit operator)가 있는데 통계를 학습하기 위해서 중요한 내용은 아닙니다.

1) 산술 연산자

연산자	설명	예제 a=5, b=7, c="2", d=-3
x+y (addition)	두 값을 받아서 더한 결과를 만듭니다.	3+1 = 4 a+3 = 8 a+b = 12 c+b = "27"
x-y (subtraction)	두 값을 받아서 첫 번째 값에 두 번째 값의 차에 대한 결과를 만듭니다.	3-1=2 a-3=2 a-b=-2 c-b=-5 "str1"-"str2" = NaN
-x (negative)	값에 음수처리를 하여 결과를 만듭니다.	-a = -5 -c = -2 -d = 3 -"test" = NaN
x*y (multiplication)	두 값을 받아서 곱셈한 결과값을 만듭니다.	a*3=15 a*b=35 a*c= 10 "str" * a = NaN
x/y (division)	첫 번째 값을 두 번째 값에 나눈 결과값을 만듭니다.	b/a = 1.4 a/c = 2.5 “str”/a = NaN
x%y (modulus)	첫 번째 값을 두 번째 값으로 나누었을 때 나머지 값을 결과 값으로 만듭니다.	7%3=1 // 7 = 3x2 + 1 a%b = 5 b%a = 2 // 7 = 5*1 + 2
x++ ++x (increment)	값에 1 을 증가 시킴이다.. 이것은 x=x+1 과 동일합니다.	a++ // a ➔ 6 a+b ➔ 13 // 6 + 7
x-- --x	값에 1 을 감소 시킴이다. 이것은 x=x-	a-- // a ➔ 4 a+b ➔ 11 // 4 + 7

(decrement)	1 과 동일 합니다.	

수학에서 배운것과 같아서 이해하시는데 큰 어려움이 없을 것입니다.

2) 비교 연산자

연산자	설명	예제 a=2, b=7, c="2", d="a"

x==y	두 값이 같으면 true 그렇지 않으면 false	a == c ➔ true // c 값이 숫자로 변환
		"a" == d ➔ true // String 비교
		a == b ➔ false
		var list = [1,2,3];
		var list2 = list
		list == list2 ➔ true

x!=y	두 값이 같다면 false 그렇지 않으면 true	a != c ➔ false "a" != d ➔ false
x > y	x 가 y 보다 크면 true	b > a ➔ true a > c ➔ false d > 10 ➔ false // 문자와 숫자 비교 불가 "c" > d ➔ true (“c”가 “a” 보다는 크다)
x >= y	x 가 y 보다 크거나 같으면 true	a >= c => true
x < y	x 가 y 보다 작으면 true	
x <= y	x 가 y 보다 작거나 같으면 true	
x===y	x 와 y 가 data 유형도 같아야 하고 값도 같아야 true	a === c ➔ false a === 2 ➔ true c === "2" ➔ true
x!==y	x 와 y 가 data 유형이 같지 않거나 값이 다르면 true	a === c ➔ true

같다를 표시할 때는 ==와 ===의 차이점을 이해하셔야 하고 다른 주의 사항은 "2 < a < 5"와 같은 연속된 비교연산자를 사용할 수 없습니다. 이를 위해서 논리 연산자를 사용해야 합니다.

3) 논리 연산자

https://github.com/handuck/jMath

연산자	설명	예제 a=0, b=7, c="2", d=-3
x&&y (and)	x 과 y 둘다 false, 0, undefined 또는 null 이 아닐 때만 true	a && b ➔ 0 (false) b && c ➔ "2" (true) c && b ➔ 7 c && b&& a ➔ 0
x||y (or)	x 또는 y 둘 중 하나라도 false, 0, undefined 또는 null 이 아니면 true	a||b ➔ 7 b||c ➔ 7 c||b ➔ "2" a||c||b ➔ "2"
!x (not)	x 이 false, 0, undefined 또는 null 이면 true	!a ➔ true !b ➔ false

앞서 설명드린 "2 < a < 5"를 표현하기 위해서는 "2<a && a<5"와 같이 표기를 해야 합니다.

4) 연산자 우선순위

연산자를 다루다 보면 여러 연산자들을 함께 사용할 경우가 많은데 어떤 연산자를 우선 처리할 것인가에 대한 내용을 알지 못하면 잘 못된 결과를 만들어 냅니다. 예를 들어 2+3*4 를 계산 할 때 우리는 당연히 3*4 를 우선하고 결과 12 에 2 를 더하는 것으로 이해를 하지만 2+3 을 먼저하고 여기에 4 를 곱하게 한다면 (2+3)*4 와 같이 명확하게 표시를 해줘야 합니다. 일반적으로 연산자 우선 순위는 산술연산자가 가장 우선 순위가 높고 다음 비교 연산자 마지막으로 논리 연산자 입니다. 이를 위해서 연산자 우선순위 표를 만들어 설명하는 경우가 많은데 가장 간단한 방법은 괄호로 명확하게 표기를 하면 됩니다. "2<a && a < 5"와 같은 경우는"(2<a) && (a < 5)"와 같이 하게 되면 명확하게 처리 순서를 결정 지을 수 있습니다.

1.3. 조건문과 반복문

1) 조건문

조건을 검색하여 조건에 맞으면 실행하고 그렇지 않으면 실행하지 않는 것으로 if, else 를 갖고 만듭니다.

유형 1	유형 2	유형 3
If 만 있을 경우	If 와 else	If, else if 와 else

`If ( 조건식 ){`
 	
`}`
	`If (조건식){`
	
	`}`
	`else{`
	
	`}`
		`If(조건식) {`
		`}`
		`else if (조건식){`
		`}`
		`else if (조건식) {`
		`}`
		…
		`else {`
		`}`

조건식이 true 라면 if 다음에 { }내에 있는 일을 하고, 그렇지 않으면 그 다음으로 넘어 가라는 뜻입니다.	조건식이 true 라면 if 다음 { }을 실행하고 false 라면 else 다음 { }을 실행합니다.	조건식이 true 이면 if 다음 { }을 실행, false 이면 다음 else if 에 조건식으로 어떤 block 을 처리할까를 결정합니다. 만일 모두가 false 이면 else 의 {}내를 실행 합니다.

`If ( a < 3 )`
`{`
`  a = 6;`
`}`
만일 변수 a 가 3 보다 작으면 a 에 6 을 입력합니다.
	`If ( a < 3 ) {`
	`  a = 6;`
	`}`
	`else {`
	`  a = 10;`
	`}`
	만일 변수 a 가 3 보다 작으면 a 에 6 을 입력 그렇지 않으면 a 에 10 을 입력합니다.
		`If ( a < 3 ) a = 4;`
		`else if ( a < 5 ) a = 6;`
		`else if ( a < 7 ) a = 8;`
		
		a 가 3 보다 작으면 a 에 4 를 입력 그렇지 않고 5 보다 작으면 a 에 6 입력 그렇지

https://github.com/handuck/jMath

		않고 7 보다 작으면 a 에 8 을 입력합니다. 만일 모두 false 이면 아무 일도 하지 않습니다.

2) 반복문

반복적인 일을 처리할 때 사용되는 구문으로 while 과 for 문을 알아 보겠습니다.

구문	예	결과

while(조건문)
{
  반복
}
	var a = 0;
	while(a < 4)
	{
	  console.log(a);
	  a++;
	}
		0
		1
		2
		3

구문	예	결과

for(시작;조건문;후처리)
{
  반복
}
	for(var a=0; a < 4 ; a++)
	{
	  console.log(a);
	}
		0
		1
		2
		3

반복문에서 사용 가능한 두개의 명령어가 있습니다.

- **break**: 반복문을 빠져 나올 때 사용되는 명령어
- **continue**: 반복문 속에 반복처리를 하는데 이 명령 다음 부터는 실행을 하지 않고 다음 반복일을 처리하는 명령어

지금까지 배우 내용들은 JavaScript 에 기초가 되는 부분으로 자세하게 다루지는 않았지만 이 부분들만 이해를 한다면 앞으로 설명할 통계 예제들을 이해하시는데 큰 어려움은 없을 것입니다. 더 자세한 내용은 JavaScript & jQuery 교재를 참조하시면 됩니다.

https://github.com/handuck/jMath

2. jMath

jMath 는 통계프로그램들이 제공되는 기능들을 JavaScript 으로 구현한 통계 library 로 이 교재에서 다루는 기업 통계 내용들을 처리할 수 있습니다.

jMath 를 이용하기 위해서 우선 jMath 객체를 만드는 방법 부터 알아야 합니다. chapter02/2_1.html 화일을 Google Chrome 으로 열면 아무내용도 나타나지 않고 HTML 화일을 열어 보시면 단순히 JavaScripot 소스 코드만 연결되어 있음을 알 수 있습니다.

```
<html>
<meta charset="utf-8">
    <script type="text/javascript" src="../js/jMath.js"></script>
    <script type="text/javascript" src="2_1.js"></script>
<body></body>
</html>
```

실행되는 내용을 보기 위해서 chapter02/2_1.js 화일을 열어 보면 다음과 같이 기술 되어 있습니다.

```
var data  = jMath([49,30,40,50,41,58]);
var mat = jMath([[1,2],[3,4]]);
var matstr = jMath('1 2;3 4');
var step = jMath('1:2:10');
```

이 코드를 보시면 단순히 jMath 객체들을 만드는 방법들만 나타납니다. 이를 확인 하기 위해서 Google Chrome 에 개발자 툴에 Console(Ctrl+Shift+J)를 열고 ">"표시된 다음 글들을 작성해서 Enter 치면 다음과 같은 결과를 볼 수 있습니다.

```
> data
[ Array[6]
0: 49
1: 30
2: 40
3: 50
4: 41
5: 58
length: 6
__proto__: Array[0] ]

> mat
 [ Array[2]            ,   Array[2]            ]
  0: 1                      0: 3
```

https://github.com/handuck/jMath

```
1: 2                    1: 4
length: 2               length: 2
__proto__: Array[0]     __proto__: Array[0]
```

변수 data 는 1 차원 배열로 행렬(matrix)로 표기하면 1x6 크기 입니다. 변수 mat 은 2 차원 배열로 2x2 로 [1,2]와 [3,4]각각은 행(row)에 해당합니다.

jMath 라는 함수를 통해서 나타난 값들은 배열인 것으로 나타나지만 실제로는 배열은 아닙니다. 그래서 JavaScript 에서 기본으로 제공되는 배열관련 함수들을 모두 지원하지 않습니다.

```
> data.map(function(v){ return v*v;} )
[ Array[6]
0: 2401
1: 900
2: 1600
3: 2500
4: 1681
5: 3364
length: 6
__proto__: Array[0]

> data.filter( function(v) { return v > 1 ; } )
TypeError: undefined is not a function
```

정리하면 jMath 는 jMath 객체를 만드는 것이지 JavaScript 의 배열을 생성하는 것이 아닙니다.

jMath 객체를 생성을 하는 다른 방법으로 입력 값으로 문자를 받아서 처리도 가능합니다. 변수 matstr 은 문자 '1 2; 3 4'로 변수 mat 과 동일한 값을 갖고 있는 객체를 만듭니다. 배열의 값들은 공백으로 구분되면 semicolon ';'으로 행(row)을 구분합니다.

마지막으로 변수 step 은 jMath 에 '1:2:10'으로 표기해 1,3,5,7,9 라는 1x5 크기의 행렬을 얻게 됩니다. 이 표기법의 의미는 '시작값:증가값:종료값'으로 시작값을 계속 증가하여 종료값 보다 크면 중단하는 것입니다. 그래서 1:2:10'에서 10 이 포함이 되지 않고 '1:2:9'와 같은 결과를 만듭니다. 만일 생성 순서를 반대로 하고 싶다면 '9:-2:1'과 같이 하면 '9,7,5,3,1'로 생성이 됩니다.

`jMath([49,30,40,50,41,58])`	$[49 \quad 30 \quad 40 \quad 50 \quad 41 \quad 58]$
`jMath([[1,2],[3,4]])`	$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$
`jMath('1 2;3 4')`	
`jMath('1:2:10')`	$[1 \quad 3 \quad 5 \quad 7 \quad 9]$

2.1. 크기

Method	설명
m$().rows	jMath object 의 row(행)의 개수를 얻게 됩니다. Ex) m$([[1,2],[3,4]]).rows 2
m$().cols	jMath object 의 column(열)의 개수를 얻게 됩니다. Ex) m$([[1,2,5],[3,4,6]]).cols 3 만일 모든 row 에 column 의 개수가 다르다면 최대 column 수를 결과값으로 돌려 줍니다. Ex) m$([[1,2,5],[3,4,6,7]]).cols 4
m$().length	m$().rows 와 같습니다.

2.2. 생성

Method	설명
jMath.fill(rows,cols,fn)	rows X cols 크기의 matrix 를 생성하고 각 element 값은 fn 함수의 결과 값이 됩니다. Ex) jMath.fill(2,2, function(r,c) { return (r+1)* (c+1); }) [[1,2], [2, 4]]
jMath.zeros(rows,cols)	rows X cols 크기의 matrix 를 생성하고 모든 element 값을 0 으로 채웁니다.

https://github.com/handuck/jMath

Method	설명
	Ex) jMath.zeros(2,2); [[0,0],[0,0]]
jMath.ones(rows,cols)	rows X cols 크기의 matrix 를 생성하고 모든 element 값을 1 으로 채웁니다. Ex) jMath.zeros(2,2); [[1,1],[1,1]]
j$().eye(row[,col])	rows X cols 크기의 matrix 를 생성하고 대각 element 값만 1 으로 채우고 나머지 값들은 모두 0 이 됩니다. Ex) jMath.eye(2,2); [[1,0],[0,1]]
j$().rand(row[,col])	rows X cols 크기의 matrix 를 생성하고 element 값을 Math.rand()으로 채웁니다. Ex) jMath.rand(2,2) [[0.8008511301595718, 0.9400642642285675] ,[0.0031995337922126055,0.53631179057993]]

2.3. 값얻기 및 변환

Method	설명
m$().diag()	jMath object 의 diagonal 값을 담은 새로운 jMath object 를 얻게 됩니다. 예) m$([[1,2],[3,4]]).diag() [1,4]
m$().transpose()	jMath object 의 transpose 값을 담은 새로운 jMath object 를 얻게 됩니다. 에) m$([[1,2],[3,4]]). transpose() [[1,3],[2,4]]

https://github.com/handuck/jMath

Method	설명
m$().map(fn)	jMath object 의 모든 element 값을 fn 에 입력값으로 하여 돌려준 값을 갖고 새로운 jMath object 생성합니다. 예) m$([[1,2],[3,4]]).map(function(x) { return x*x; }) [[1,4],[9,16]]

m$().forEach(fn)	jMath object 에 모든 element 값을 fn 에 입력값으로 하여 실행을 합니다.
	m$([[1,2],[3,4]]).forEach(function(v,r,c){
	console.log(v, r, c);
	});
	1 0 0
	2 0 1
	3 1 0
	4 1 1

m$().repeatColumn(len)	Column 방향으로 반복되는 시켜 새로운 jMath object 를 얻습니다. m$([[1,2],[3,4]]).repeatColumn(3) [[1 2 1 2 1 2 1 2],[3 4 3 4 3 4 3 4]]
m$().repeatRow(len)	Column 방향으로 반복되는 시켜 새로운 jMath object 를 얻습니다. m$([[1,2],[3,4]]). repeatRow (3) [[1 2], [3, 4], [1 2], [3, 4], [1 2], [3, 4], [1 2], [3, 4]]

2.4. 산술 연산자들

JavaScript 에서 제공되는 기본 연산자들을 사용할 수 없기 때문에 jMath 에 함수로 등록하여 기본연산처럼 다룰 수 있도록 했습니다.

https://github.com/handuck/jMath

연산자들	설명
덧셈 A['+'](B)	두 jMath 객체를 덧셈을 하는데 더해질 두 jMath 의 크기는 동일해야 합니다. Ex) a = jMath([[1,2],[3,4]]); b = jMath([[1,2],[3,4]]); a['+'](b) [[2,4], [6 8]] 만일 입력값이 숫자일 경우에는 모든 element 에 각 값을 더하게 됩니다. Ex) a['+'](1) [[2,3], [4 5]]
뺄셈 A['-'](B)	덧셈과 내용은 같습니다. Ex) a = jMath([[1,2],[3,4]]); b = jMath.ones(2,2); a['-'](b) [[0,1], [2 3]] a['-'](1) [[0,1], [2 3]]
곱셈 A['*'](B)	행렬의 곱을 실행합니다. 두 jMath 의 크기는 행렬곱이 가능해야 합니다. Ex) a = jMath([[1,2],[3,4]]); b = jMath([[1],[3]]); a['*'](b) [[7],[15]] 만일 입력값이 숫자을 경우에는 모든 element 에 각 값을 곱하게 됩니다. a['*'](2) [[2,4],[6,8]]
나눗셈 A['/'](B)	A/B 에서 B 는 역행렬로 바뀌어서 A 와 곱을한 결과를 돌려줍니다. a = jMath([[1,2],[3,4]]); a['/'](a)

	[[1,0],[0,1]] 만일 입력값이 숫자을 경우에는 모든 element 에 각 값을 나누게 됩니다. a['/'](2) [[0.5, 1],[1.5, 2]]
제곱 A['^'](x)	A 에 x 제곱한 결과 값을 돌려 줍니다. a = jMath([[1,2],[3,4]]); a['^'](2) [[7 10],[15 22]] 이 결과는 A['*'](A)와 같습니다.

앞서 설명드린 연산자들은 모두 Matrix 를 연산하기 위한 방식으로 만일 곱셈, 나눗셈, 제곱을 element 별로 하고 싶다면 다음과 같이 연산자를 사용해야 합니다.

연산자	설명

곱셈 A['.*'](B)	Element 별로 곱셈을 합니다.
	a = jMath([[1,2],[3,4]]);
	b = jMath([[3,4],[5,6]]);
	a['.*'](b).toString()
	[[3,8], [15,24]]

나눗셈 A['./'](B)	Element 별로 나눗셈을 합니다.
	a = jMath([[6,8],[10,12]]);
	b = jMath([[3,4],[5,6]]);
	a['./'](b)
	[[2,2],[2,2]]

제곱 A['.^'](x)	Element 별로 x 제곱을 합니다. a = jMath([[1,2],[3,4]]); a['.^'](2) [[1,4],[9,16]]

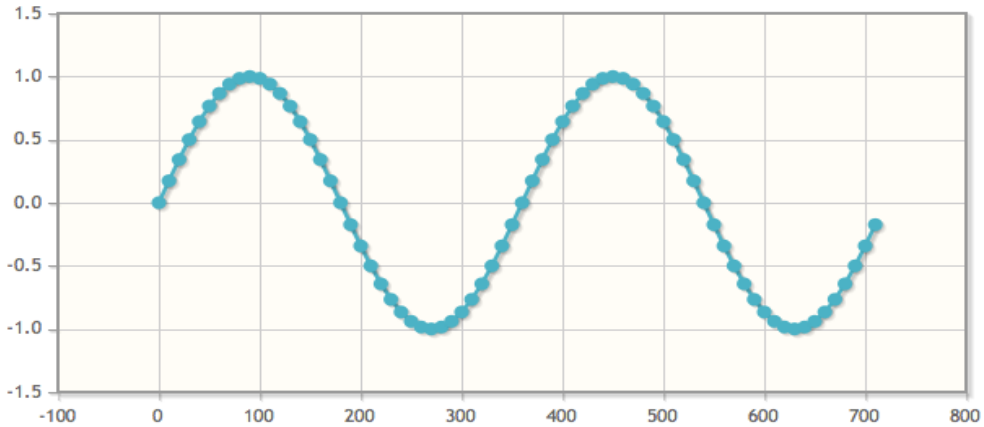
3. jqPlot

jqPlot 은 2D 로 표현되는 chart 를 그릴 때 사용되는 jQuery plugin 입니다. jqPlot 의 동작은 원하는 형태의 chart 에 해당하는 Renderer 를 선택해서 결과물 여러가지 형태로 표현할 수 있습니다. 이 교재에서는 jqPlot 으로 그릴 수 있는 대표적인 몇 가지를 소개하겠습니다.

종류	설명
Line chart	선을 연결하여 보여주는 것으로 renderer 가 선택되지 않으면 기본으로 이 chart 를 사용합니다.
Scatter plot	점들만 있는 chart
Bar chart(Histogram)	막대 그래프: Clustered 와 Stack 을 지원합니다.
Pie chart	파이 차트
Bubble chart	x,y 에 반지름 r 만큼 원을 그려서 값을 표시하는 차트
Open Hi Low Close chart	한국 주식에 사용될 수 있는 차트

이러한 chart 들을 생성할 때 부가적으로 사용 가능한 renderer 들이 있는데 기능은 x 축 y 축에 그릴 때 사용되는 renderer 들, legend 관련 renderer 들이 있습니다.

3.1. Line Chart



```
<html>
	<head>
	  <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
	  <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
	  <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
	  <script type="text/javascript">
$(function(){
	var values = [];
	for ( var d = 0, i=0 ; d < 720 ; d+= 10, i++ )
	{
	  values[i] = [ d, Math.sin( d * Math.PI/180 ) ];
	}
	$.jqplot('chart', [values] );
});
	  </script>
	</head>
	<body>
	  <div id="chart" style="width:600px;height:300px"></div>
	</body>
</html>
```

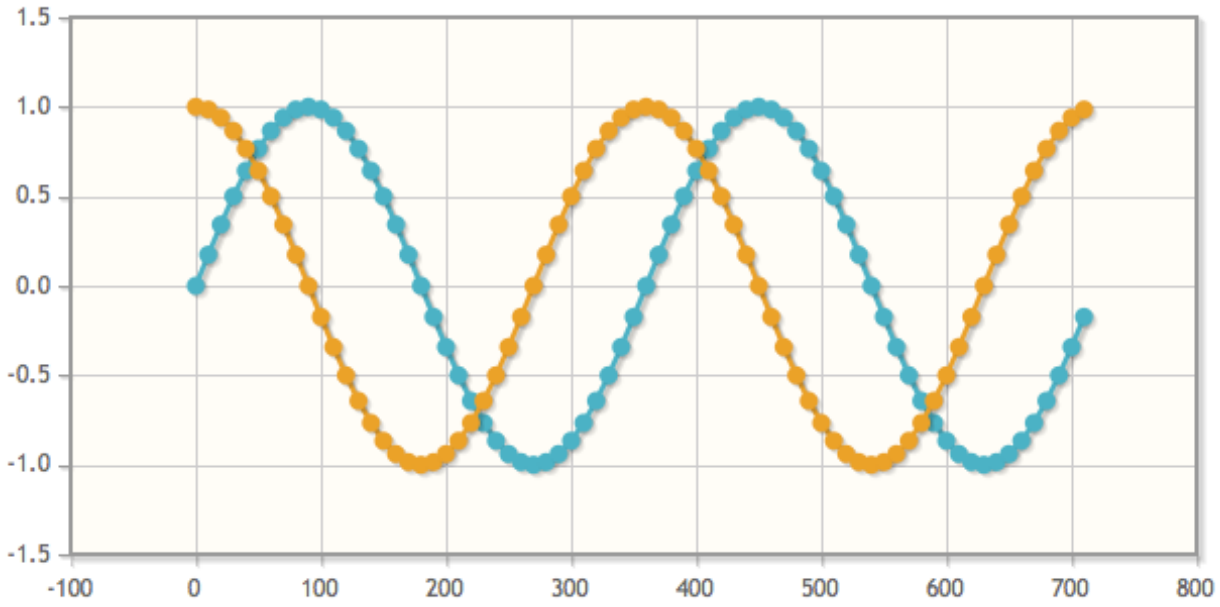
2_3_1.html 은 sin 곡선을 그리는 예제입니다. 예제에서 보시는 것과 같이 아무런 option 도 없이 jqplot 을 그린 예로 $.jqplot(id, [xys])과 같이 하면 line chart 가 그려지게 됩니다. 여기서 id 는 chart 를 그릴 대상에 id 값입니다.

여기서 중요한 것은 $.jqplot()함수의 두번째 값은 array 로 이 array 안에는 x,y 가 쌍으로 있는 array 입니다. 예를 들어 (1,1), (2,2), (3,3), (4,4)와 같이 x,y 가 쌍으로 4 개가 존재 한다면 이값은 다음과 같아야 합니다.

[[[1,1], [2,2],[3,3],[4,4]]]

다음 $.jqplot()에 두 번째 값인 x,y 값들이 왜 array 로 싸여 있는가를 알아 보겠습니다.

```
<html>
	<head>
	  <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
	  <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
	  <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
	  <script type="text/javascript">
$(function(){
	var values1 = [];
```




```
      var values2 = [];
      for ( var d = 0, i=0 ; d < 720 ; d+= 10, i++ )
      {
        values1[i] = [ d, Math.sin( d * Math.PI/180 ) ];
        values2[i] = [ d, Math.cos( d * Math.PI/180 ) ];
      }
      $.jqplot('chart', [values1, values2] );
});
        </script>
      </head>
      <body>
        <div id="chart" style="width:600px;height:300px"></div>
      </body>
</html>
```

2_3_2.html 을 보시면 values1 과 value2 가 있는데 values1 은 sin 을 value2 는 cos 을 넣었고 $.jqplot()함수의 두 번째 값에 [value1, value2]와 같이 넣었습니다. 다시 말해 두 번째 argument 가 []로 받는 이유는 여러 개의 line 을 한 그래프에 동시에 그리기 위해서 였습니다.

3.2. Chart 꾸미기.

Chart 의 구성은 그리는 방법(series), 제목(title), x 와 y 축, legend 이 4 가지로 나뉘울 수 있습니다. 즉 우리는 이 4 가지를 어떻게 조정하고 어떤 plugin 을 추가로 사용하는 가에 따라서 출력되는 그래프의 형태가 달라지게 됩니다. 그럼 그리는 방법인 series 는 chart 의 전체를 바꾸는 것이 때문에 여기서는 line chart 만 사용할 것이고, 나머지 3 가지 내용은 바로 $.jqplot()함수의 3 번재 argument 값들 입니다.

```
{
    title: {  …  },
    seriesDefaults: { … },
    series : [  …  ],
    axesDefaults: {  … }
    axes: {
       xaxis: {  },
       yaxis: {  }
    },
    legend: { … }
}
```

이외에 다른 몇 가지가 더 있지만 이것 하나 하나 학습 하면서 후에 추가 되는 사항을 배우도록 하겠습니다.

3.2.1. Title

title 의 구성은 { text:'' , show: true, fontFamily:, fontSize:, textAlign:, textColor: } 의 값을 조정 할 수 있습니다. fontSize 는 숫자로 pt 값을 넣으면 되고, textAlign 은 'left', 'right', ' center'중 하나면 됩니다. 그리고 textColor 는 css 와 같이 '#rrggbb'형태로 넣으시면 됩니다.

3.2.2. x, y 축 (axes)

axesDefaults 값은 x,y axis 에 공통으로 적용될 값을 넣는 곳이고 axes 에 xaxis 와 yaxis 는 각 축에 해당하는 값을 넣으면 됩니다. 값으로는 { min, max, pad, ticks: [], numberTick,

showTicks, showTickMarks, renderer, rendererOptions: {}, tickOpetions: {} }으로 내용이 많이 있습니다. 몇 가지 용어만 알아보고 사용법은 예를 통해서 알아 보겠습니다.

min, max 는 각 축에 최대 최소값입니다. pad 는 입력된 값의 최대 최소에 몇 배의 값을 더한 값으로 min,max 를 사용합니다. ticks 는 좌표에 표시될 숫자값들의 위치이고 없다면 자동으로 생성됩니다. renderer 는 축을 그리는데 도움을 주는 사용할 renderer 입니다. renderer 에 따라서 추가로 필요한 option 들이 더 생길 수 있습니다.

3.2.3. legend

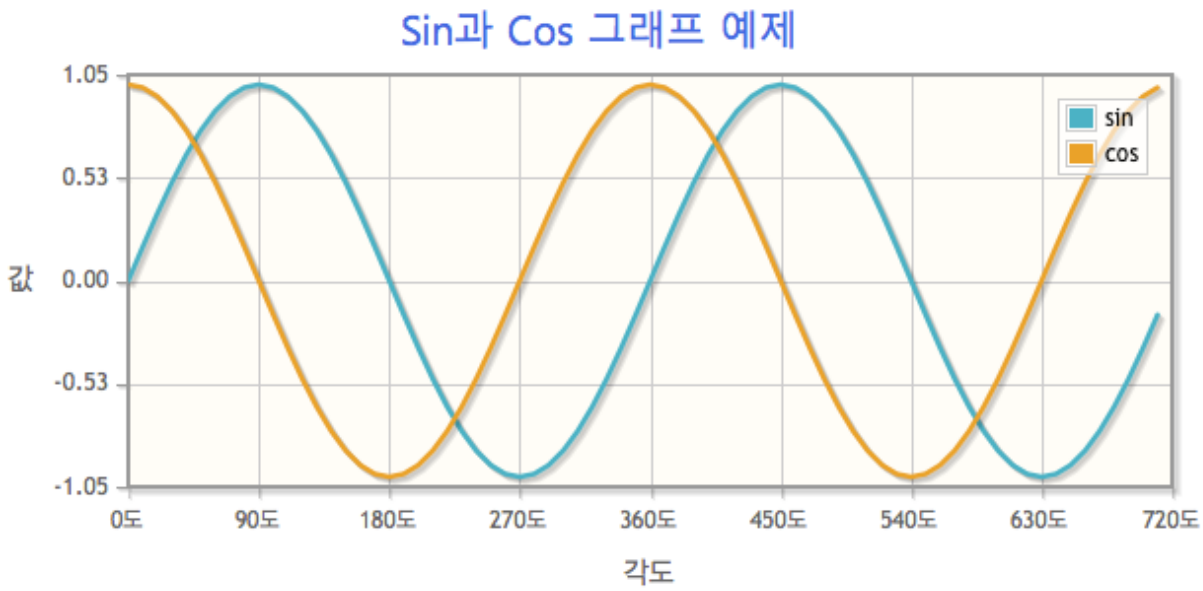
여러개의 line 을 동시에 그릴 때 각 선이 무엇을 의미하는지 알아야 합니다. 이러한 표시를 해주는 것이 legend 입니다. 값으로는 { show: false, location: 'ne', xoffset: 12, yoffset: 12, renderer: null }의 기본값을 갖고 있습니다. location 은 표시할 위치로 nw, n, ne, e, se, s, sw, w 의 8 가지 방향중 하나를 선택하시면 됩니다. xoffset 과 yoffset 은 축으로 부터 떨어질 폭과 높이 입니다.

내용이 많아 예를 통해서 배우면서 차근차근 알아 보도록 하겠습니다. 그럼 sin 과 cos 을 동시에 그래프의 예제에 내용을 추가해 보도록 하겠습니다.

```
<html>
      <head>
        <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
        <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
        <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
        <script type="text/javascript"
                src="../js/plugins/jqplot.EnhancedLegendRenderer.min.js"></script>

        <script type="text/javascript">
$(function(){
      var values1 = [];
      var values2 = [];
      for ( var d = 0, i=0 ; d < 720 ; d+= 10, i++ )
      {
        values1[i] = [ d, Math.sin( d * Math.PI/180 ) ];
        values2[i] = [ d, Math.cos( d * Math.PI/180 ) ];
      }

      var title = {
        text: 'Sin과 Cos 그래프 예제',
        fontSize: 20,
        fontFamily: 'malgungothic',
```

https://github.com/handuck/jMath

```
        textColor: 'royalblue'
    };

    var xaxis = {
      label : '각도',
      min: 0,
      max: 720,
      tickOptions: {
            formatString: '%d도'
      }
    };

    var yaxis = {
      label : '값',
      max: 1.05,
      min: -1.05,
      tickOptions: {
            formatString: '%3.2f'
      }
    };

    var series = {
      renderer: $.jqplot.LineRenderer,
      showMarker: false
    }

    var legend = {
      renderer: $.jqplot.EnhancedLegendRenderer,
      show: true,
      labels: [ 'sin', 'cos' ]
    }

    $.jqplot('chart', [values1, values2], {
      title: title,
      axes: {
          xaxis: xaxis,
          yaxis: yaxis,
      },
      legend: legend,
      seriesDefaults: series
    });
});
      </script>
    </head>
    <body>
      <div id="chart" style="width:600px;height:300px"></div>
    </body>
</html>
```



첫번째로 title 에서는 폰트(fontFamily), 글자크기(fontSize), 색깔(textColor)을 수정했고, x 와 y 축은 각각 최대(max) 최소값(min)을 직접 조정을 하였고 label 값을 넣었습니다. 그리고 tickOptions 에 formatString 이라는 것으로 통해서 화면에 각 축마다 보여주는 값을 어떠한 형태로 보여 줄 것인가를 조정합니다. Format 은 C 언어에서 사용하는 것과 동일합니다.

Legend 는 labels 를 통해서 이름을 넣었는데 이것이 없으면 Series1, Series2 와 같이 나타납니다. 다른 값은 없기 때문에 위치는 기본 위치인 "ne"측 화면의 오른쪽 위에 축으로부터 12 만큼 떨어진 위치에서 나타납니다. 이 예제에서 legend 를 위한 enhancedLegendRenderer 를 사용했습니다. 이것이 없어도 상관은 없지만 좋은 점은 여러 series 들을 그릴 때 각각의 series 를 끄고 켜줍니다. Legend 에 sin 을 누르시면 sin 곡선이 화면에서 없어 질것입니다.

마지막으로 seriesDefaults 에 입력되는 값은 (x,y)값을 갖고 있는 두 array 를 그릴 때 기본적으로 사용할 사용될 정보로 renderer 는 기본값인 lineRenderer 를 사용했고, showMarker 를 false 로 해서 점이 찍히는 것을 방지 했습니다. 더 많은 기능 들이 있지만 다양한 series 를 그리는 방법을 학습을 하면서 하나씩 배워 가겠습니다.

3.3. Scatter chart

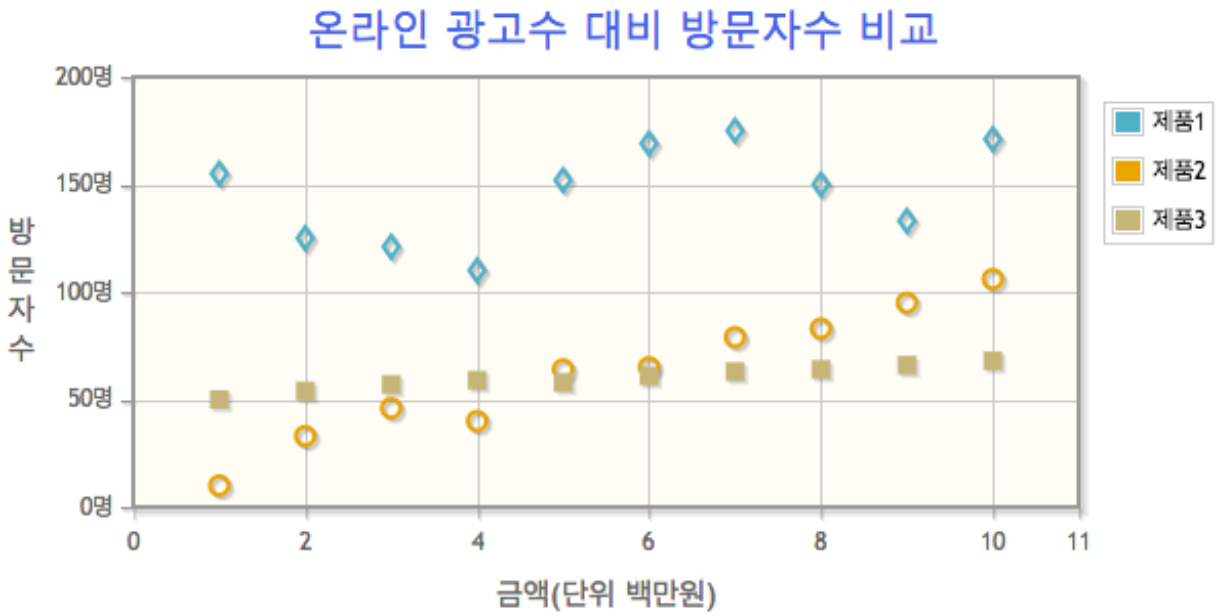
line 은 없고 점들만 있는 그래프입니다. 예제를 만들면서 보겠습니다. 한 업체에서 3 개의 상품에 광고비용에 대비 판매량을 비교하여 scatter plot 을 그린다고 할 때 광고 비용 대비 판매량을 가상으로 만들어 보겠습니다.

https://github.com/handuck/jMath

```
<html>
      <head>
        <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
        <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
        <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
        <script type="text/javascript"
                src="../js/plugins/jqplot.EnhancedLegendRenderer.min.js"></script>
        <script type="text/javascript">
$(function(){

// 값 생성
      var values1 = [];
      var values2 = [];
      var values3 = [];

      for ( var i = 0 ; i < 10 ; i++ )
      {
        values1[i] = [ i+1, parseInt(100 + Math.random() * 100) ];
        values2[i] = [ i+1, parseInt( 10 + Math.random() * 20 + i * 10) ];
        values3[i] = [ i+1, parseInt( 50 + Math.random() * 5 + i * 2) ];
      }

// 타이틀 꾸미기
      var title = {
        text: '온라인 광고수 대비 방문자수 비교',
        fontSize: 20,
        fontFamily: 'malgungothic',
        textColor: 'royalblue'
      };

// ticks을 사용해서 x좌표에 표시될 값을 직접 알려줍니다.
      var xaxis = {
        label : '금액(단위 백만원)',
        ticks: [ 0, 2, 4, 6, 8, 10, 11 ],
        tickOptions: {
             // 기본이 실수형태이기 때문에 정수로 나타내주기 사용됩니다.
               formatString: '%d'
        }
      };
// pad를 이용해서 y의 최대, 최소범위에 1.05배로 확장합니다.
// 만일 max, min보다 이 값이 작다면 max, min으로 대신 사용됩니다.
      var yaxis = {
        label : '방문자수',
        pad: 1.05,
        tickOptions: {
               formatString: '%d명'
        }
      };

// marker를 어떤 형태로 사용할 것인가를 알려 줍니다.
```

https://github.com/handuck/jMath

```
// series별로 값을 따로 지정해야 합니다.
      var series = [
        { markerOptions: { style: 'diamond' } },
        { markerOptions: { style: 'circle' } },
        { markerOptions: { style: 'filledSquare' } }
      ];

// legend를 placement를 이용 그래프 밖으로 보냅니다.
      var legend = {
        renderer: $.jqplot.EnhancedLegendRenderer,
        show: true,
        labels: [ '제품1', '제품2', '제품3' ],
        placement: 'outsideGrid'
      }

      $.jqplot('chart', [values1, values2, values3], {
        title: title,
               axes: {
               xaxis: xaxis,
               yaxis: yaxis,
        },
        legend: legend,
// 모든 series에 공통으로 적용되는 값입니다.
        seriesDefaults: {
               showLine: false,
               showMarker: true
        },
        series: series
      });
});
        </script>
      </head>
      <body>
        <div id="chart" style="width:600px;height:300px"></div>
      </body>
</html>
```



values1 인 제품 1 은 100 에서 200 사이에 값이 나오도록 했고, values2 와 values3 는 광고수가 늘어나면 방문자수가 늘어 나도록 했습니다. 차이점은 values2 가 온라인 광고수가 많을 수록 values3 보다 더 빠르게 증가합니다.

그럼 Marker 에 대해서 자세히 알아 보겠습니다. Marker 를 표시하기 위해서 예제 $.jqplot()함수에 serieseDefaults 에 showLine 을 false 로 line 을 그리지 못하도록 했습니다. 이유는 기본으로 사용되는 renderer 가 LineRenderer 이기 때문에 이 값의 기본값이 true 이기 때문입니다. 이렇게 seriesDefaults 에 값은 모든 series 에 동시에 적용되는 값들입니다. 그리고 series 에 array(배열)로 각 series 별 option 값을 받게 됩니다. 이 예제에서는 각 series 마다 다른 모양의 marker 를 사용하기 위해서 markerOptions 에 style 에 각기 다른 값을 넣었습니다. Style 로 사용될 수 있는 값은 다음과 같습니다.

```
circle, diamond, square,
filledCircle,filledDiamond,filledSquare.
```

xaxis 에 보시면 ticks 에 array(배열)로 값을 0 부터 10 까지 짝수와 마지막에 11 을 넣었는데, 이는 결과 그래프에 x 축을 보시면 ticks 에서 명시된 것만 화면에 나타납니다.
Series 에 사용되는 option 들에 대해 보시면 다음 bar 차트를 학습 하도록 하겠습니다.

```
seriesDefaults: {
    show: true,   // 입력된 series들을 보여 줄것인가
    xaxis: 'xaxis', // x축의 속성으로 사용될 key로 'xaxis' 또는 'x2axis'.
```

https://github.com/handuck/jMath

```
    yaxis: 'yaxis', // y축의 속성으로 사용될 key로 'yaxis' 또는 'y2axis'.
    label: '',      // legend에 사용될 label. 없으면 Series숫자 형태로 나타납니다.
    color: '',      // 사용될 색깔로 없으면 자동으로 선택 됩니다.
    lineWidth: 2.5, // 선 두께
    shadow: true,   // 그림자를 보여 줄 것인가.
    shadowAngle: 45,    // 그림자 각도, x축에서 아래로 몇도
    shadowOffset: 1.25, // 선으로 부터 그림자 거리
    shadowDepth: 3,     // 그림자를 몇번 그릴 것인가 그릴 때 마다 shadowOffset만큼 떨어 짐니다.
    shadowAlpha: 0.1,   // 그림자 불투명도 0 ~ 1.
    showLine: true,     // 선을 보여 줄 것인가
    showMarker: true,   // x,y점을 보여줄 것인가
    fill: false,        // 선 아래 부분을 채울 것인가
    fillAndStroke: false,       // 선 아래를 채우면서 선을 그릴 것인가
    fillColor: undefined,       // 선 아래 채울 색깔( 없으면 선 색깔을 사용합니다)
    fillAlpha: undefined,       // 선 아래 채울때 불투명도 0 ~ 1
    renderer: $.jqplot.LineRenderer],    // 사용될 renderer
    rendererOptions: {}, // 사용될 renderer에 필요한 옵션들로 lineRenderer는 필요 없습니다.
    markerRenderer: $.jqplot.MarkerRenderer,    // (x,y)점을 그리기 위한 renderer
    markerOptions: {
        show: true,             // marker를 보여 줄 것인가
        style: 'filledCircle',  // circle, diamond, square, filledCircle, filledDiamond, filledSquare.
        lineWidth: 2,       // marker 두께
        size: 9,            // marker크기
        color: '#666666'    // marker색깔. 기본으로 line색깔을 사용합니다.
        shadow: true,       // marker의 그림자를 사용할 것인가
        shadowAngle: 45,    //그림자 각도, x축에서 아래로 몇도
        shadowOffset: 1,    // marker로 부터 그림자 거리
        shadowDepth: 3,    //그림자를 몇번 그릴 것인가 그릴 때 마다 shadowOffset만큼 떨어
짐니다.
        shadowAlpha: 0.07   // 그림자 불투명도 0 ~ 1
    }
},
series:[  { seriesDefaults와 동일}, { seriesDefaults와 동일},  ....   ]
```

내용이 좀 많아 보이시겠지만 지금까지의 예제에 위의 값들을 적용하면서 보시면 무엇을 의미하시는지 이해하시기 쉬울 것입니다.

3.4. Bar chart

막대 그래프라고 하는 qualitative data 와 같은 category 별로 값을 표시하기 유용합니다. 표시 방법은 수직, 수평으로 표시가 가능하며 여러개의 막대를 동시에 표시는 clustered 방식과 stack 방식 둘 다를 지원 합니다.

Bar chart 를 그리기 위해서 필요한 plugin 들은 다음과 같습니다.

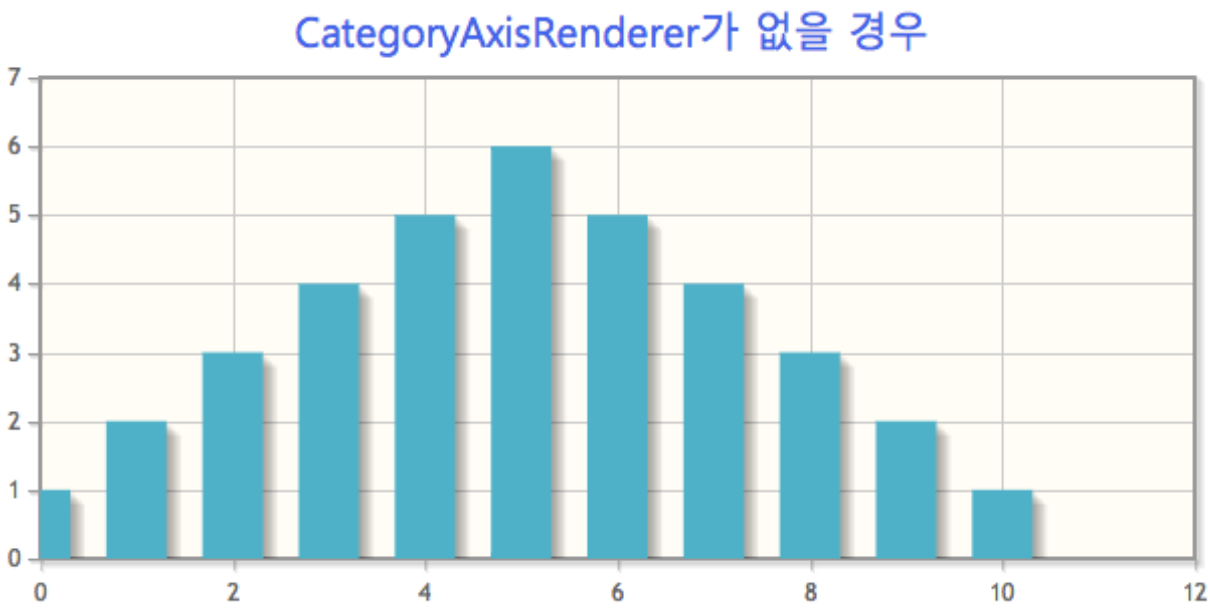
plugins	설명
BarRenderer	Bar Chart 를 그려주는 renderer 입니다.
CategoryAxisRenderer	축의 값이 category 일 경우 반드시 있어야 합니다. 만일 이것이 없고 해당 축값이 숫자이면 tick 에 맞추어 bar 가 생성 됩니다.

다음 예는 barRenderer 만을 추가하여 그래프를 그린 결과 입니다.

```
<html>
      <head>
        <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
        <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
        <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
        <script type="text/javascript"
                 src="../js/plugins/jqplot.barRenderer.min.js"></script>
        <script type="text/javascript">
$(function(){
      var values = [1,2,3,4,5,6,5,4,3,2,1].map( function(v,idx){
        return [idx,v];
      });

      var title = {
        text: 'CategoryAxisRenderer가 없을 경우',
        fontSize: 20,
        fontFamily: 'malgungothic',
        textColor: 'royalblue'
      };

     // Bar chart를 위해 jqplot.barRenderer.min.js가 필요
      $.jqplot('chart', [values], {
        title: title,
        seriesDefaults: {
                renderer: $.jqplot.BarRenderer,
                rendererOptions: {
                        barWidth: 30
```

https://github.com/handuck/jMath

```
                }
            }
        });
});
        </script>
    </head>
    <body>
        <div id="chart" style="width:600px;height:300px"></div>
    </body>
</html>
```



보시는 것과 같이 category 로 처리를 하지 않았기 때문에 tick 에 맞춰서 값이 나타납니다. 이러한 방식이 필요할 때도 있지만 histogramcj 처럼 category 별로 보여 주는 경우가 대부분이라 x 축에 CategoryAxisRenderer 를 사용하여 원하는 결과를 얻을 수 있습니다.

영화진흥 위원회의 2011 년 영화소비자 조사 보고서에서 1 년 동안 남여의 연령대별 평균 영화 관람수를 비교한 자료입니다.

	15-18 세	19-23 세	24-29 세	30-34 세	35-39 세	40-49 세	50-59 세
남자	13.3	14.7	16.9	11.9	9.0	7.5	7.2
여자	13.8	12.6	14.0	12.0	9.6	8.9	8.1

3.4.1. Vertical Bar Chart

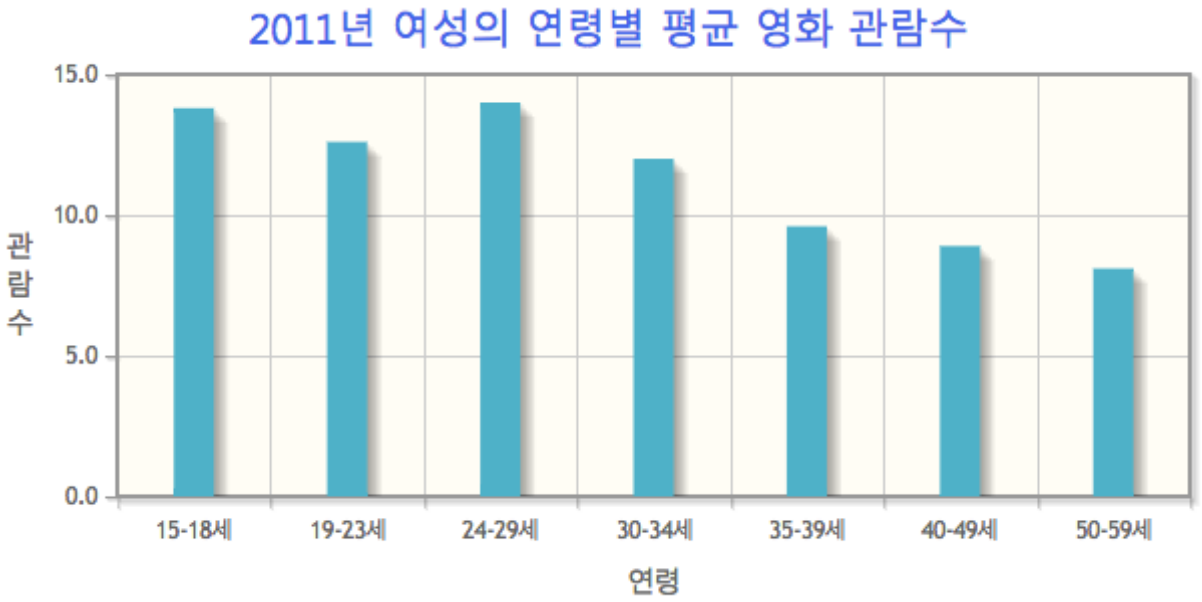
https://github.com/handuck/jMath

여성의 평균 영화 관람수 만을 그려 보도록 하겠습니다.

```
<html>
     <head>
       <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
       <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
       <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
       <script type="text/javascript"
               src="../js/plugins/jqplot.barRenderer.min.js"></script>
       <script type="text/javascript"
               src="../js/plugins/jqplot.categoryAxisRenderer.min.js"></script>
       <script type="text/javascript">
$(function(){
    // x축에 사용될 category들
     var xlabels = ['15-18세', '19-23세', '24-29세', '30-34세', '35-39세', '40-49세', '50-
59세'];
    // 여성의 평균 관람수 (x,y)생성
     var females = [13.8,12.6,14,12,9.6,8.9,8.1].map( function(v,idx){
       return [ xlabels[idx],v];
     });

     var title = {
       text: '2011년 여성의 연령별 평균 영화 관람수',
       fontSize: 20,
       fontFamily: 'malgungothic',
       textColor: 'royalblue'
     };

    // x축의 값이 category이므로 jqplot.categoryAxisRenderer.min.js가 필요
     var xaxis = {
       label: '연령',
       renderer: $.jqplot.CategoryAxisRenderer
     };

     var yaxis = {
       label : '관람수',
       tickOptions: {
              formatString: '%3.1f'
       }
     };

    // Bar chart를 위해 jqplot.barRenderer.min.js가 필요
     var series = {
       renderer: $.jqplot.BarRenderer,
        // BarRenderer를 위한 option값들
       rendererOptions: {
              barWidth: 20,     // bar의 폭
              fillToZero: true  // series에 y축 최소값을 무조건 0으로 합니다.
```

https://github.com/handuck/jMath

```
        }
      }

      $.jqplot('chart', [females], {
        title: title,
               axes: {
               xaxis: xaxis,
               yaxis: yaxis,
        },
        seriesDefaults: series
      });
});
        </script>
      </head>
      <body>
        <div id="chart" style="width:600px;height:300px"></div>
      </body>
</html>
```



값을 생성하는 과정에서 (x,y)값에서 x 는 category 이름이고, y 는 해당 숫자값이 들어갑니다. X 축을 보시면 값을 그리기 위해서 renderer 로 $.jqplot.CategoryAxisRenderer 를 사용했는데 이 때문에 tick 사이에 bar 와 tick 값이 그려지게 됩니다.

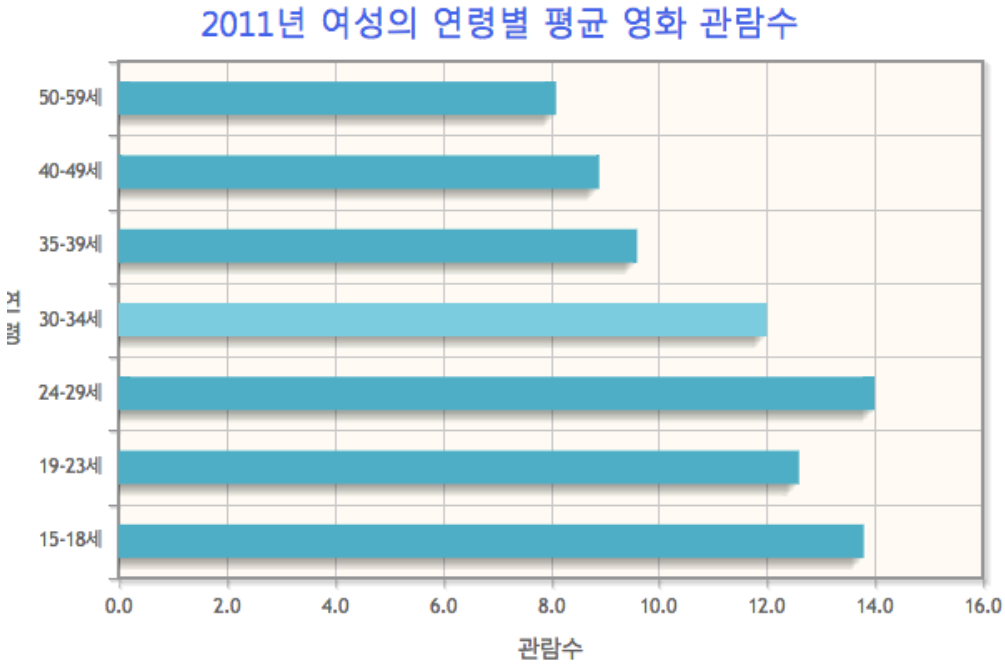
다음으로 SeriesDefaults 에서 renderer 를 BarRenderer 로 사용하면서 rendererOptions 에 fillToZero 를 추가 했는데 이 값이 없다면 관람자 수 범위에서 계산된 min 부터 bar 가 그려지기 때문에 fillToZero 를 true 로 해서 강제로 min 을 0 으로 만듭니다.

3.4.2. Horizontal Bar Chart

https://github.com/handuck/jMath

여성 연령별 평균 영화 관람수를 수직으로 나타내기 위해서 몇가지 처리가 필요합니다.

- (x,y)값을 (y,x)로 뒤집고,
- x 축과 y 축에 옵션도 바꿔야 합니다.
- Series 에 rendererOptions 에 barDirection 값에 horizontal 을 추가 해야합니다.




```
<html>
      <head>
        <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
        <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
        <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
        <script type="text/javascript"
                src="../js/plugins/jqplot.barRenderer.min.js"></script>
        <script type="text/javascript"
                src="../js/plugins/jqplot.categoryAxisRenderer.min.js"></script>
        <script type="text/javascript">
$(function(){
    // x축에 사용될 category들
var xlabels = ['15-18세', '19-23세', '24-29세', '30-34세', '35-39세', '40-49세', '50-59세'];
    // 여성의 평균 관람수 (x,y)생성
     // 수평으로 그래기 우해 수직에 x,y값은 y,x형태로 있으어 합니다.
     var females = [13.8,12.6,14,12,9.6,8.9,8.1].map( function(v,idx){
       return [ v, xlabels[idx] ];
     });

     var title = {
       text: '2011년 여성의 연령별 평균 영화 관람수',
       fontSize: 20,
       fontFamily: 'malgungothic',
       textColor: 'royalblue'
```

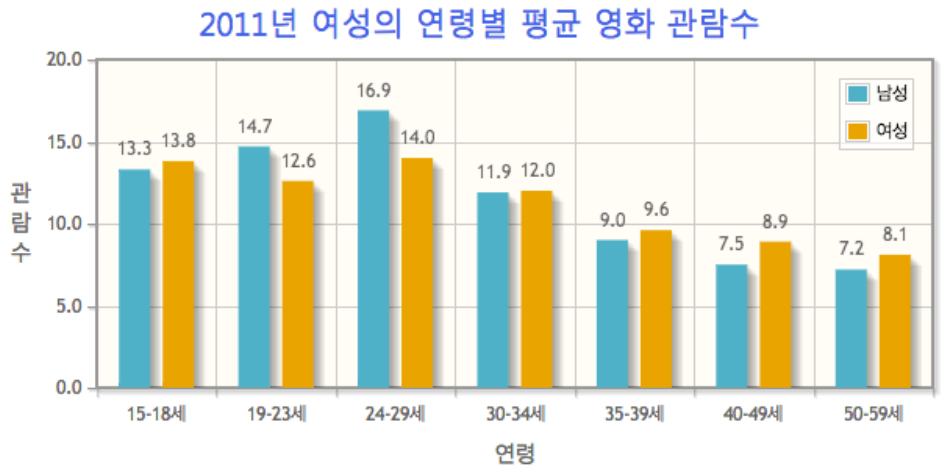
https://github.com/handuck/jMath

```
    };

  // y축의 값이 category이므로 jqplot.categoryAxisRenderer.min.js가 필요
    var yaxis = {
      label: '연령',
      renderer: $.jqplot.CategoryAxisRenderer
    };

    var xaxis = {
      label : '관람수',
      tickOptions: {
            formatString: '%3.1f'
      }
    };

  // Bar chart를 위해 jqplot.barRenderer.min.js가 필요
    var series = {
      renderer: $.jqplot.BarRenderer,
      // 수평으로 그릴 때 그림자가 45도면 bar보다 크게 나와서 135도로 합니다
      shadowAngle: 135,
       // BarRenderer를 위한 option값들
      rendererOptions: {
            barWidth: 20,     // bar의 폭
            fillToZero: true, // series에 y축 최소값을 무조건 0으로 합니다.
            barDirection: 'horizontal' // 수평으로 그리게 합니다
      }
    }

    $.jqplot('chart', [females], {
      title: title,
            axes: {
            xaxis: xaxis,
            yaxis: yaxis,
      },
      seriesDefaults: series
    });
});
      </script>
    </head>
    <body>
      <div id="chart" style="width:600px;height:400px"></div>
    </body>
</html>
```

3.4.3. Clustered Bar Chart 와 Stacked Bar Chart

https://github.com/handuck/jMath

남성과 여성의 연령대별 비교를 위해서 두개의 bar chart 를 합쳐 보도록 하겠습니다. 첫번째로 나란히 bar 들을 보여주는 clustered bar chart 입니다.



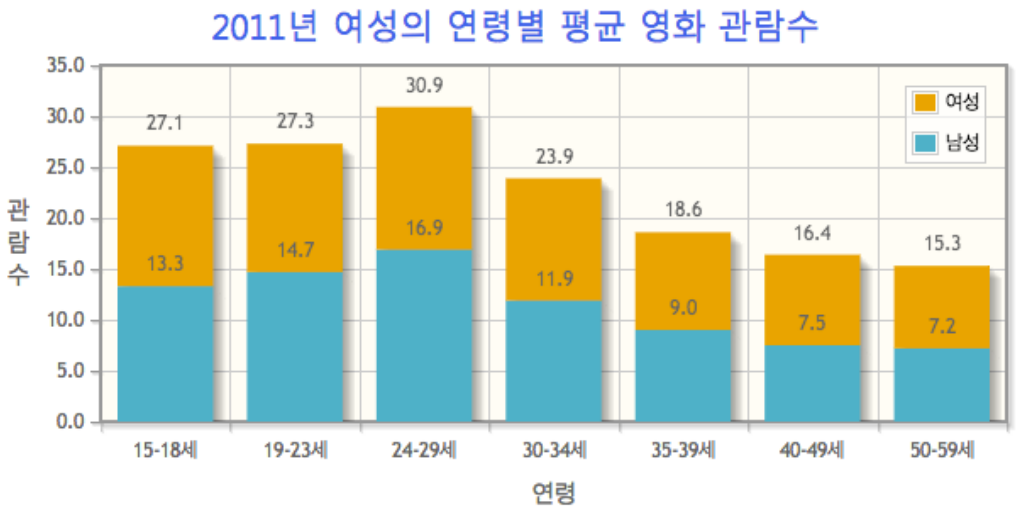

```
<html>
      <head>
        <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
        <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
        <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
        <script type="text/javascript"
               src="../js/plugins/jqplot.enhancedLegendRenderer.min.js"></script>
        <script type="text/javascript"
               src="../js/plugins/jqplot.barRenderer.min.js"></script>
        <script type="text/javascript"
               src="../js/plugins/jqplot.categoryAxisRenderer.min.js"></script>
        <script type="text/javascript"
               src="../js/plugins/jqplot.pointLabels.min.js"></script>
        <script type="text/javascript">
$(function(){

// 남성과 여성의 (x,y)값을 생성합니다.
      var xlabels = ['15-18세', '19-23세', '24-29세', '30-34세', '35-39세', '40-49세', '50-
59세'];
      var males = [13.3,14.7,16.9,11.9,9,7.5,7.2].map( function(v, idx){
        return [ xlabels[idx],v];
      });
      var females = [13.8,12.6,14,12,9.6,8.9,8.1].map( function(v,idx){
        return [ xlabels[idx],v];
      });

      var title = {
        text: '2011년 여성의 연령별 평균 영화 관람수',
        fontSize: 20,
        fontFamily: 'malgungothic',
        textColor: 'royalblue'
```

https://github.com/handuck/jMath

```
      };

// x축이 category이므로 categryAxisRender.min.js가 필요합니다.
      var xaxis = {
        label: '연령',
        renderer: $.jqplot.CategoryAxisRenderer
      };

      var yaxis = {
        label : '관람수',
        tickOptions: {
// bar위의 값은 yaxis.tickOptions.formatString을 이용합니다.
                formatString: '%3.1f'
        }
      };

      var series = {
        renderer: $.jqplot.BarRenderer,
        rendererOptions: {
                barWidth: 20,
                fillToZero: true
        },
// bar위에 값을 추가 합니다.
// pointLabels.min.js가 필요합니다.
        pointLabels: { show: true }
      }

      var legend = {
        renderer: $.jqplot.EnhancedLegendRenderer,
        show: true,
        labels: [ '남성', '여성' ]
      }

      $.jqplot('chart', [males,females], {
        title: title,
                axes: {
                xaxis: xaxis,
                yaxis: yaxis,
        },
        seriesDefaults: series,
        legend: legend
      });
});
        </script>
      </head>
      <body>
        <div id="chart" style="width:600px;height:300px"></div>
      </body>
</html>
```

여러개의 series 들을 bar 로 그리는 것은 여러 line 을 그리는 것과 같이 $.jqplot()함수의 series 값을 여러개 넣으면 됩니다. 이번 예제가 전과 다른 것은 y 값을 각 bar 위에 넣었습니다. 이를 위해서 pointLabels.min.js 를 읽어 오고 series 옵션에 pointLabels 옵션을 추가 해야 합니다.

Clustered bar chart 를 Stacked bar chart 로 변환하는 것은 위의 코드에서 다음 부분만 수정하면 됩니다. 첫 번째로 series 데이타는 y 값만이 필요합니다. 그리고 x 축 tick 값들은 기본으로 1,2,3 과 같이 숫자가 사용되기 때문에 xaxis.ticks 에 값을 넣어야 합니다. $.jqplot()의 옵션에 stackSeries:true 가 있어야 하고 pointLabels 에 stackedValue:true 를 넣어서 각 bar 의 값보다는 증가되는 값을 보여 줍니다.




```
$(function(){

      var xlabels = ['15-18세', '19-23세', '24-29세', '30-34세', '35-39세', '40-49세', '50-
59세'];

      var males = [13.3,14.7,16.9,11.9,9,7.5,7.2];
      var females = [13.8,12.6,14,12,9.6,8.9,8.1];

      var title = {
        text: '2011년 여성의 연령별 평균 영화 관람수',
        fontSize: 20,
        fontFamily: 'malgungothic',
        textColor: 'royalblue'
      };

// x축이 category이므로 categryAxisRender.min.js가 필요합니다.
      var xaxis = {
        label: '연령',
```

https://github.com/handuck/jMath

```
        renderer: $.jqplot.CategoryAxisRenderer,
        ticks: xlabels
      };

      var yaxis = {
        label : '관람수',
        tickOptions: {
// bar위의 값은 yaxis.tickOptions.formatString을 이용합니다.
                formatString: '%3.1f'
        }
      };

      var series = {
        renderer: $.jqplot.BarRenderer,
        rendererOptions: {
                barMargin: 20,
                fillToZero: true,
        },
// bar위에 값을 추가 합니다.
// pointLabels.min.js가 필요합니다.
// stackedValue를 true로 해서 bar값이 증가되는 값을 보여 줍니다.
        pointLabels: { show: true, stackedValue: true }
      }

      var legend = {
        renderer: $.jqplot.EnhancedLegendRenderer,
        show: true,
        labels: [ '남성', '여성' ]
      }

      $.jqplot('chart', [males,females], {
        title: title,
                axes: {
                xaxis: xaxis,
                yaxis: yaxis,
        },
// stacked bar라는 것을 알려줍니다.
        stackSeries: true,
        seriesDefaults: series,
        legend: legend
      });
});
```

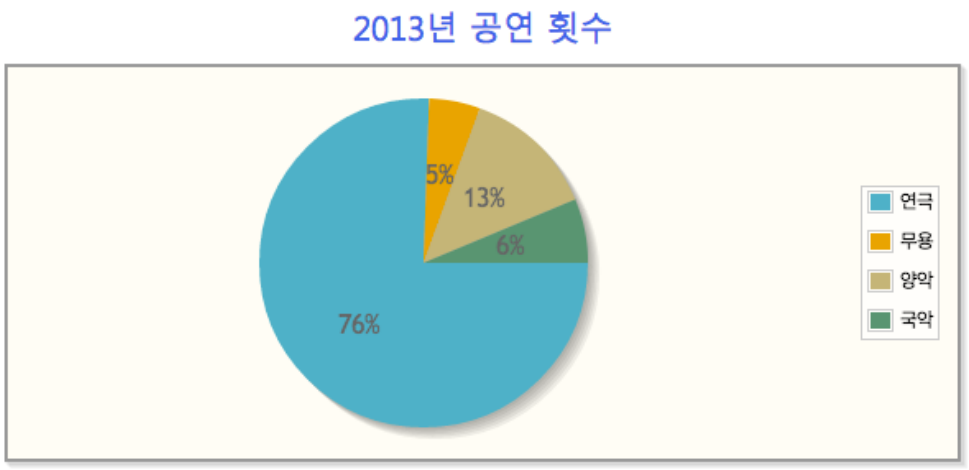
3.5. Pie Chart

막대 그래프와 같이 qualitative data 를 표현하기 유용하며 특히 비율을 쉽게 알 수 있습니다. 다음의 예는 한국문화예술위원회 문예연감에 2013 공연 횟수 정보 입니다.

https://github.com/handuck/jMath

연극	무용	양악	국악
47,781	3,188	8,298	3,984

이 내용을 Pie chart 로 표시하기 위해서 PieRenderer 가 필요하고, 축에 대한 정보는 필요가 없습니다.




```
<html>
      <head>
        <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
        <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
        <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
        <script type="text/javascript"
                 src="../js/plugins/jqplot.pieRenderer.min.js"></script>
        <script type="text/javascript">
$(function(){

      var xlabels = [ '연극', '무용', '양악', '국악' ];
      var list = [ 47781, 3188, 8298, 3984 ].map( function(v, idx){
        return [ xlabels[idx],v];
      });
      var title = {
        text: '2013년 공연 횟수',
        fontSize: 20,
        fontFamily: 'malgungothic',
        textColor: 'royalblue'
      };

      var series = {
        renderer: $.jqplot.PieRenderer,
        rendererOptions: {
            showDataLabels: true
        },
```

https://github.com/handuck/jMath

```
    }

    var legend = {
      show: true,
      location: 'e'
    }

    $.jqplot('chart', [list], {
      title: title,
      seriesDefaults: series,
      legend: legend
    });
});
      </script>
    </head>
    <body>
      <div id="chart" style="width:600px;height:300px"></div>
    </body>
</html>
```

Pie chart 를 그리게 되면 보시는 것과 같이 자동으로 비율이 계산되어 나타나고 rendererOptions 에 showDataLables 를 true 로 하여 백분율을 같이 보여 줄 수 있습니다.

3.6. Bubble Chart

Scatter chart 이면서 값의 크기를 원의 크기로 보여 줄 수 있어서 3 가지 정보를 동시에 보여 줄 수 있습니다. 예를 들어 자동차 생산량과 판매액을 표현 할 때 scatter chart 로 점으로 표현 되지만 여기에 점유율으로 원의 크기를 다르게 한다면 3 가지 정보를 동시에 볼 수 있습니다.

다음 예는 통계청 자료로 2012 년 5 월에 5 개의 규모가 작은 지방 공항별 수송현황 입니다.

지역	무안동	군산시	원주시	사천시	포항시
운항수(편)	72	124	60	156	262
여행자수(명)	7187	16207	8555	14034	23747
화물량(톤)	70	137	46	64	78

이를 jqPlot 으로 표현하기 위해서 series 의 element 는 [x,y,radius,label]로 존해 해야 합니다. jqPlot 이 radius 에 있는 값을 전체 데이타와 비교하여 원의 크기를 자동으로 잡아주기 때문에 정확한 pixel 값을 넣지 않아도 되지만, 만일 series 옵션에 rendererOptions 에 autoScaleBubbles 값을 false 로 하면 정확한 pixel 값을 넣어야 합니다.
https://github.com/handuck/jMath

여깃 x,y 는 여행자수, 운항수(편)으로 각각 사용되며 원의 크기는 편당 평균 무게에 비례하여 나타내 보도록 하겠습니다.

```
<html>
      <meta charset="utf-8">
      <head>
        <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
        <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
        <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
        <script type="text/javascript"
src="../js/plugins/jqplot.bubbleRenderer.min.js"></script>
        <script type="text/javascript">
$(function(){

      var list = [ [7187,72,70, '무안동'], [16207,124,137, '군산시'],
                 [14034,156,64, '사천시'], [8555,60, 46, '원주시'],
                 [23747,262,78, '포항시']];

      for ( var i = 0  ; i < list.length; i++ )
      {
         //승객수 1000명 단위로 조정
        list[i][0] = Math.round(list[i][0] / 1000);
         // 비행기 편당 화물(톤)
         list[i][2] = list[i][2] / list[i][1];
      }

      var title = {
        text: '2012년 5월 지방 공항별 수송현황',
        fontSize: 20,
        fontFamily: 'malgungothic',
        textColor: 'royalblue'
      };

      var xaxis = {
        tickOptions: {
                formatString: '%d천명'
        }
      };

      var yaxis = {
        min: 0,
        tickOptions: {
                formatString: '%d편'
        }
      };

      var series = {
        renderer: $.jqplot.BubbleRenderer,
        rendererOptions: {
                bubbleAlpha: 0.6,
                highlightAlpha: 0.8,
                bubbleGradients: true
```

https://github.com/handuck/jMath

```
        },
        shadow: true,
        shadowAlpha: 0.05
      };

      $.jqplot('chart', [list], {
        title: title,
        seriesDefaults: series,
                axes: {
                xaxis: xaxis,
                yaxis: yaxis,
        }
      });
});
        </script>
      </head>
      <body>
        <div id="chart" style="width:600px;height:300px"></div>
      </body>
</html>
```

3.7. Candle Chart: Open Hi Low Close chart

한국 주식의 값을 표현하기에 적합한 chart 로 series 의 element 구성은 [날짜 ,시작가(open), 최대가(hi), 최저가(low), 종가(close)]입니다. 필요한 plugin 들은

https://github.com/handuck/jMath

캔들차트를 그리기 위한 ohlcRenderer.min.js, 축에 날짜를 표시하기 위한 dateAxisRenderer.min.js, 캔들에 마우스를 올려 놓았을 때 툴팁을 보여주는 highlighter.min.js 입니다.

```
<html>
	<meta charset="utf-8">
	<head>
	  <link rel="stylesheet" type="text/css" href="../css/jquery.jqplot.min.css">
	  <script type="text/javascript" src="../js/jquery-2.1.0.min.js"></script>
	  <script type="text/javascript" src="../js/jquery.jqplot.min.js"></script>
	  <script type="text/javascript"
	          src="../js/plugins/jqplot.ohlcRenderer.min.js"></script>
	  <script type="text/javascript"
	          src="../js/plugins/jqplot.dateAxisRenderer.min.js"></script>
	  <script type="text/javascript"
	          src="../js/plugins/jqplot.highlighter.min.js"></script>
	  <script type="text/javascript">
$(function(){

	// 주가 :  [날짜,시가,최고,최저,종가]
	var list = [
	  ['2013/02/07',200500,203000,199000,200000],
	  ['2013/02/08',200500,210500,200000,209500],
	  ['2013/02/12',211500,212500,209000,211500],
	  ['2013/02/13',213500,217000,211500,216500],
	  ['2013/02/14',216500,218000,214500,216500],
	  ['2013/02/15',215000,217500,211500,213500],
	  ['2013/02/18',210000,212500,208000,209500],
	  ['2013/02/19',210000,213000,209500,210500],
	  ['2013/02/20',213500,217500,213000,217000],
	  ['2013/02/21',216000,218000,215000,216000],
	  ['2013/02/22',217500,220000,215500,219000],
	  ['2013/02/25',218000,218000,214500,214500],
	  ['2013/02/26',213500,214500,212000,213500],
	  ['2013/02/27',214500,217500,214000,215000],
	  ['2013/02/28',216500,219000,215000,218000],
	  ['2013/03/04',219500,222500,217000,219500],
	  ['2013/03/05',222500,224500,220000,221000],
	  ['2013/03/06',222000,223000,215500,217500],
	  ['2013/03/07',219000,219000,213500,216000],
	  ['2013/03/08',213000,216000,211500,213500],
	  ['2013/03/11',210000,210000,206500,208500],
	  ['2013/03/12',210500,212000,208500,210500],
	  ['2013/03/13',209500,211500,209000,210000],
	  ['2013/03/14',210000,212500,209000,212000],
	  ['2013/03/15',213000,221500,213000,220000],
	  ['2013/03/18',223000,223500,218000,219000],
	  ['2013/03/19',219500,221500,216000,218000],
	  ['2013/03/20',217000,220500,215000,217500],
	  ['2013/03/21',219000,220000,215000,215000],
	  ['2013/03/22',214000,217000,213500,214500] ];

	var title = {
	  text: '2013/02/07 ~ 2013/03/22 현대차 주식',
	  fontSize: 20,
```

https://github.com/handuck/jMath

```
        fontFamily: 'malgungothic',
        textColor: 'royalblue'
      };

      // 날짜 표시를 위해 DateAxisRendere 사용
      var xaxis = {
        label: '날짜',
        renderer: $.jqplot.DateAxisRenderer,
        min: '2013/02/06',
        max: '2013/03/23',
        tickOptions: {
                formatString: '%m/%d',
                tickInterval: '1 week'
        }
      };

      var yaxis = {
        label: '가격',
        tickOptions: {
                // y축의 tick값 조정
                formatter: function(fmt,v)
                {
                        return "₩" + (v/1000).toLocaleString() + '천원'
                },
        }
      };

      // 캔들차트 그리기
      var series = {
        renderer: $.jqplot.OHLCRenderer,
        rendererOptions: {
                candleStick: true,
                // 상가일때 색
                upBodyColor: 'red',
                // 하가일때 색
                downBodyColor: 'blue',
                // 몸통색깔 채우기
                fillUpBody: true,
                fillDownBody: true,
        },
        // 시가와 종가가 같을 때 색깔을 검정색으로
        color: 'black'
      };

      // 마우스를 캔들위로 이동하였을때 정보 표시
      var hl = {
        show: true,
        // 해당 캔들에 점을 보여줄 것인가
        showMarker: false,
        // 보여줄 값들
        tooltipAxes: 'xy',
```

```
        yvalues: 4,
        formatString: '<table class="jqplot-highlighter">\
                <tr><td>날짜:</td><td>%s</td>\
                <tr><td>시가:</td><td>%s</td>\
                <tr><td>최대:</td><td>%s</td>\
                <tr><td>최소:</td><td>%s</td>\
                <tr><td>종가:</td><td>%s</td>'
      };

      $.jqplot('chart', [list], {
        title: title,
        seriesDefaults: series,
                axes: {
                xaxis: xaxis,
                yaxis: yaxis,
        },
        highlighter: hl,
        animate: true,
        animateReplot: true
      });
});
        </script>
      </head>
      <body>
        <div id="chart" style="width:650px;height:300px"></div>
      </body>
</html>
```

여기서 새로 적용된 것들은 yaxis.tickOptions.formatter 입니다. 이것은 callback 함수를 등록받아서 tick 그릴 때 tickOptions.formatString 값과 y 값을 입력값으로 넣어 주어 표현될 값을 받아 y 축에 그립니다. 다음 highlighter 는 마우스를 캔들 위로 이동하면 나타나는

툴팁을 어떻게 그릴 것인가에 대한 정보 입니다. 마지막으로 animate 은 그릴 때 효과를 주기 위한 option 입니다.

jqPlot 은 이 외에 여러 가지 renderer 들을 제공해주며, renderer 에 option 을 주는 방식에 따라 다양한 형태의 결과를 볼 수 있습니다. 또한 zoom 기능, 마우스 event 처리, 축에 값 회전해서 보여주기등에 기능들을 제공해 줍니다.