알메이트-차트 사용 설명서

버전 7.0

리아모어 소프트

목차

1. 처음 시작하기

1.1 처음 차트 만들기

먼저 알메이트 차트를 이용해 주셔서 감사드립니다. 이제부터 여러분은 다양한 종류와 풍부한 데이터 분석 기능을 제공하는 HTML5 차트 제품을 경험하게 되실 것입니다. 다음은 알메이트 차트를 처음 접하시는 분들을 위해서 불과 몇 분이면 적용이 가능한 차트 생성 방법을 단계적으로 설명한 내용입니다.

1. HTML 파일의 <head> 섹션 내에 알메이트 라이센스 파일, 라이브러리 파일, CSS 파일을 include 합니다.

```
<!DOCTYPE html>
<html>
<head>
  <script
        src="https://www.riamore.net/HTML5demo/chart/LicenseKey/rMateChartH5License.j
        s"></script>
  <script
        src="https://www.riamore.net/HTML5demo/chart/rMateChartH5/JS/rMateChartH5.js">
        </script>
  <link rel="stylesheet"
        href="https://www.riamore.net/HTML5demo/chart/rMateChartH5/Assets/Css/rMateCh
        artH5.css"/>
</head>
</html>
```

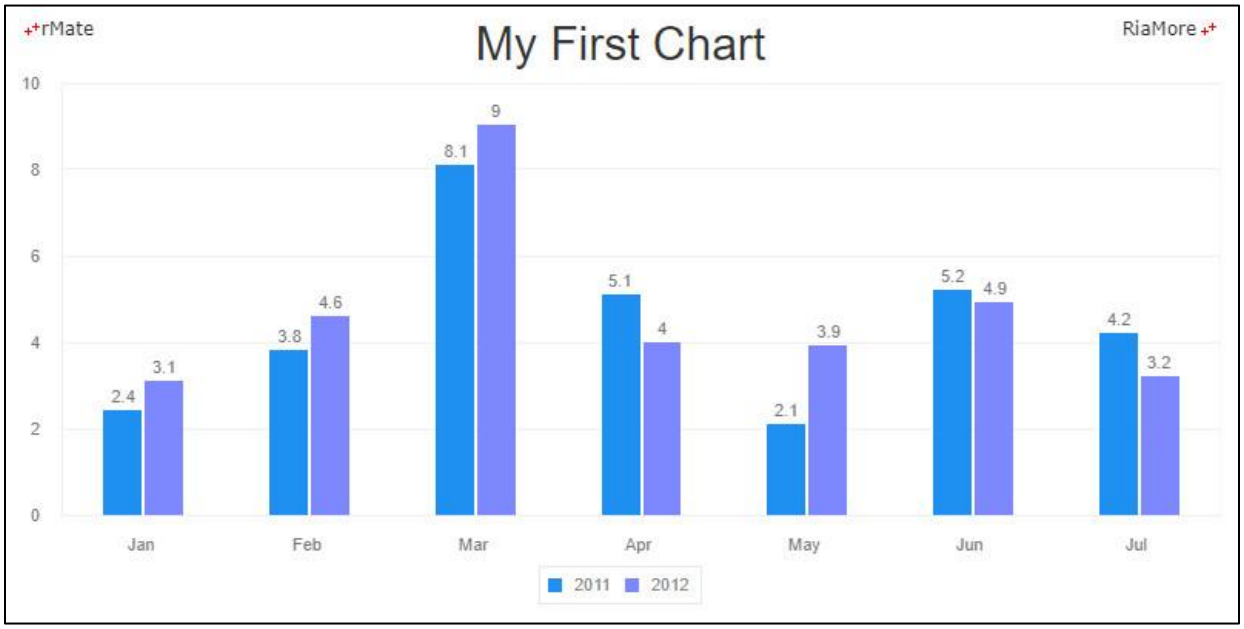
2. 차트의 레이아웃(Layout)과 데이터셋(Dataset)을 설정하고 차트 생성을 실행하는 자바스크립트 코드를 구현합니다. 레이아웃은 차트의 종류, 모양, 기능 등을 설정하고 데이터셋은 차트에 표현될 데이터를 설정합니다.

```
<!DOCTYPE html>
<html>
<head>
  <script
       src="https://www.riamore.net/HTML5demo/chart/LicenseKey/rMateChartH5License.
       js"></script>
  <script
       src="https://www.riamore.net/HTML5demo/chart/rMateChartH5/JS/rMateChartH5.js
       "></script>
  <link rel="stylesheet"
       href="https://www.riamore.net/HTML5demo/chart/rMateChartH5/Assets/Css/rMateC
       hartH5.css"/>

  <script type="text/javascript">
    rMateChartH5.create("chart1", "chartHolder", "", "100%", "100%");
    var layoutStr =
      '<rMateChart backgroundColor="#FFFFFF" borderStyle="none">'
       +'<Options>'
         +'<Caption text="My First Chart" fontSize="20"/>'
         +'<Legend/>'
       +'</Options>'
       +'<Column2DChart showDataTips="true">'
         +'<horizontalAxis>'
           +'<CategoryAxis categoryField="Month"/>'
         +'</horizontalAxis>'
         +'<verticalAxis>'
           +'<LinearAxis maximum="10"/>'
         +'</verticalAxis>'
         +'<series>'
           +'<Column2DSeries labelPosition="outside" yField="2011"
      displayName="2011"/>'
         +'</series>'
       +'</Column2DChart>'
     +'</rMateChart>';

    var chartData =
      [{"Month":"Jan", "2011":2.4},
      {"Month":"Feb", "2011":3.8},
      {"Month":"Mar", "2011":8.1},
      {"Month":"Apr", "2011":5.1},
      {"Month":"May", "2011":2.1},
      {"Month":"Jun", "2011":5.2},
      {"Month":"Jul", "2011":4.2}];

    rMateChartH5.calls("chart1", {
      "setLayout" : layoutStr,
      "setData" : chartData
    });
  </script>
```

3. 차트가 생성될 를 HTML 파일의 <body> 섹션 내에 지정합니다.

```
<!DOCTYPE html>
<html>
<head>
  <script
       src="https://www.riamore.net/HTML5demo/chart/LicenseKey/rMateChartH5
       License.js"></script>
  <script
       src="https://www.riamore.net/HTML5demo/chart/rMateChartH5/JS/rMateCh
       artH5.js"></script>
  <link rel="stylesheet"
       href="https://www.riamore.net/HTML5demo/chart/rMateChartH5/Assets/Cs
       s/rMateChartH5.css"/>
  <script type="text/javascript">
    rMateChartH5.create("chart1", "chartHolder", "", "100%", "100%");
    var layoutStr =
      '<rMateChart backgroundColor="#FFFFFF" borderStyle="none">'
       +'<Options>'
         +'<Caption text="My First Chart" fontSize="20"/>'
         +'<Legend/>'
       +'</Options>'
       +'<Column2DChart showDataTips="true">'
         +'<horizontalAxis>'
           +'<CategoryAxis categoryField="Month"/>'
         +'</horizontalAxis>'
         +'<verticalAxis>'
           +'<LinearAxis maximum="10"/>'
         +'</verticalAxis>'
         +'<series>'
           +'<Column2DSeries labelPosition="outside" yField="2011"
      displayName="2011"/>'
         +'</series>'
       +'</Column2DChart>'
     +'</rMateChart>';

    var chartData =
      [{"Month":"Jan", "2011":2.4, "2012":3.1},
       {"Month":"Feb", "2011":3.8, "2012":4.6},
       {"Month":"Mar", "2011":8.1, "2012":9},
       {"Month":"Apr", "2011":5.1, "2012":4},
       {"Month":"May", "2011":2.1, "2012":3.9},
       {"Month":"Jun", "2011":5.2, "2012":4.9},
       {"Month":"Jul", "2011":4.2, "2012":3.2}];

    rMateChartH5.calls("chart1", {
      "setLayout" : layoutStr,
      "setData" : chartData
    });
  </script>
</head>
<body>
  <div id="chartHolder" style="width:600px; height:400px;">
</body>
</html>
```

다음은 위에서 설명된 내용 중, 마지막 단계의 코드를 그대로 적용하여 만들어진 차트의 결과입니다:



See the CodePen 알메이트 차트 - My First Chart

1.2 시스템 특징 및 요구사항

알메이트 차트는 HTML5 Canvas 기술을 기반으로 개발된 순수 자바스크립트 라이브러리입니다. 알메이트 차트의 강력한 분석 기능과 세련된 비주얼 효과(그라데이션, 애니메이션, 이벤트 등)를 이용하면 누구라도 깊이 있는 데이터 분석을 가능하게 하는 다양한 종류의 2D, ㅋ차트를 웹페이지에 쉽게 표현할 수 있습니다.

알메이트 차트는 모든 서버 환경에서 동작하며, 서버 스크립트 언어에 의존적이지 않습니다. 한 가지 요구되는 것은 클라이언트 환경이 HTML5 Canvas 를 지원하는 브러우저여야 한다는 것입니다. 다음은 HTML5 Canvas 를 지원하는 브라우저와 해당 브라우저의 버전이 표시된 표입니다.

IE	Firefox	Safari	Chrome	iPhone	Android
9.0+	3.0+	3.0+	3.0+	1.0+	1.0+

1.3 다운로드 및 설치

알메이트 차트 평가판은 리아모어 홈페이지에서 다운로드 받으실 수 있습니다. 평가판 다운로드를 원하시면 다음 링크에서 다운로드 버튼을 클릭하십시요.

http://www.riamore.net/component/chart.html

제품을 다운로드 받으신 후 압축을 풀면 다음과 같은 구성 요소(하위 디렉토리)들을 확인하실 수 있습니다.

1. rMateChartH5 : 제품의 엔진인 자바스크립트 라이브러리 파일들(rMateChartH5.js 등)과 제품에서 사용되는 이미지 파일과 CSS 파일이 존재합니다. 이 디렉토리에는 다음과 같은 하위 디렉토리들이 존재합니다.

    a. Assets: CSS 파일, 아이콘, 이미지, 패턴 이미지, 테마 (Theme) 라이브러리
    b. JS: 엔진 라이브러리, 익스플로러 캔버스 라이브러리, PDF 변환 라이브러리, 테이블 변환 라이브러리

2. LicenseKey : 알메이트 차트 라이센스 파일(rMateChartH5License.js)이 존재하는 디렉토리입니다.

3. Docs: 알메이트 차트 사용자 설명서와 API 문서 파일이 존재하는 디렉토리입니다. 문서는 HTML 파일들로 구성되어 있습니다.

    a. api : API 문서들이 존재하는 디렉토리
    b. User Manual – html : 사용자 설명 문서들이 존재하는 디렉토리

4. Samples : 각종 예제 파일들이 존재하는 디렉토리입니다.

5. index.html : 제품에 포함된 샘플 차트들을 실행할 수 있는 시작 페이지입니다. 이 파일을 웹브라우저를 통해서 오픈하면 400 여개의 즉시 활용이 가능한 차트들을 실행할 수 있습니다.

설치를 원하는 서버나 사용자 PC 의 특정 디렉토리에 설치 파일의 압축을 해제하시면 바로 알메이트 차트를 사용하실 수 있습니다. 압축 해제 후 어떠한 설정 작업도 필요없으며, 차트 생성을 원하는 웹페이지 파일에 처음 차트 만들기 에서 설명드린 3 가지 파일(라이센스, 라이브러리, CSS)에 대한 URL 을 설정하시면 됩니다.

2. 차트 만들기

2.1 레이아웃 생성

알메이트 차트의 레이아웃에는 차트의 종류, 기능, 모양, 색상 등 차트의 외관과 기능에 관련된 모든 것이 설정됩니다. 알메이트 차트에서 레이아웃은 XML 형식으로 만들어져야 하며, 작성된 레이아웃은 자바스크립트의 스트링 변수에 저장되거나 파일에 저장되어 알메이트 차트의 레이아웃 설정함수 호출을 통해서 차트에 설정됩니다. 다음은 처음 차트 만들기에서 예제에 적용된 2011 년도의 월별 데이터 값을 컬럼 차트로 표현하는 레이아웃입니다. 레이아웃은 반드시 <rMateChart> 노드로 시작(최상위 XML 노드)해야 합니다. 아래 예제 코드에는 차트의 제목과 범례를 생성하는 <Option> 노드, 컬럼 차트를 생성하는 <Column2DChart> 노드가 <rMateChart> 노드의 자식 노드로 설정되었습니다. <Column2DChart> 노드에는 컬럼 차트의 가로축 (<horizontalAxis> 노드), 세로축(<verticalAxis> 노드) 그리고 표현되는 데이터 (<series> 노드)가 자식 노드로 설정되었습니다. 레이아웃을 구성하는 각 노드에 대한 자세한 설명은 차트 구성 요소를 참조하십시요.

```
<rMateChart backgroundColor="#FFFFFF" borderStyle="none">
  <Options>
    <Caption text="My First Chart" fontSize="20"/>
    <Legend/>
  </Options>
  <Column2DChart showDataTips="true">
    <horizontalAxis>
      <CategoryAxis categoryField="Month"/>
    </horizontalAxis>
    <verticalAxis>
      <LinearAxis maximum="10"/>
    </verticalAxis>
    <series>
      <Column2DSeries labelPosition="outside" yField="2011" displayName="2011"/>
    </series>
  </Column2DChart>
</rMateChart>
```

2.2 데이터셋 생성

알메이트 차트의 데이터셋의 형식은 XML, JSON(JavaScript Object Notification) 형식의 배열 그리고 CSV 형식이 지원됩니다.

다음은 처음 차트 만들기에서 예제에 적용된 데이터셋을 XML 형식으로 표현한 것입니다. XML 형식의 데이터셋은 반드시 <items> 노드로 시작해야 하고, 차트에서 데이터 포인트로 표현되는 하나의 데이터는 <item> 노드의 자식 노드로 정의되어야 합니다. 아래 예제에서는 X 축 카테고리에 표시되는 데이터는 <Month> 노드에, 2011 년 데이터 값은 <2011> 노드에 정의되었습니다.

```
<items>
  <item>
    <Month>Jan</Month>
    <2011>2.4</2011>
  </item>
  <item>
    <Month>Feb</Month>
    <2011>3.8</2011>
  </item>
  <item>
    <Month>Mar</Month>
    <2011>8.1</2011>
  </item>
  <item>
    <Month>Apr</Month>
    <2011>5.1</2011>
  </item>
  <item>
    <Month>May</Month>
    <2011>2.1</2011>
  </item>
  <item>
    <Month>Jun</Month>
    <2011>5.2</2011>
  </item>
  <item>
    <Month>Jul</Month>
    <2011>4.2</2011>
  </item>
</items>
```

다음은 위에서 설명한 XML 데이터셋을 JSON 형식의 배열로 표시한 것입니다. 차트에서 데이터 포인트로 표현되는 하나의 데이터가 하나의 객체에 해당됩니다. 예제에서는 X 축 카테고리에 표시되는 데이터는 객체의 “Month” 필드에, 2011 년 데이터 값은 객체의 “2011” 필드에 정의되었습니다.

```
[{"Month":"Jan", "2011":2.4},
{"Month":"Feb", "2011":3.8},
{"Month":"Mar", "2011":8.1},
{"Month":"Apr", "2011":5.1},
{"Month":"May", "2011":2.1},
{"Month":"Jun", "2011":5.2},
{"Month":"Jul", "2011":4.2}]
```

다음은 위에서 설명한 데이터를 CSV 형식으로 표현한 것입니다.

```
Jan,2.4
Feb,3.8
Mar,8.1
Apr,5.1
May,2.1
Jun,5.2
Jul,4.2
```

CSV 형식의 데이터셋에는 XML 과 JSON 형식과는 다르게 필드명(“Month”, “2011”)이 지정되지 않습니다. 하지만 레이아웃에는 어떤 데이터가 X 축에 표현될 데이터이고, 어떤 데이터가 차트의 데이터 포인트에 표현될 데이터인지 지정을 해야하기 때문에 필드명 대신 필드 번호를 이용합니다. 필드 번호의 지정 규칙은 좌측 첫 번째 필드(Jan, Feb, …)가 “F0” 이고 우측 필드로 갈수록 1 씩 증가됩니다. (F0, F1, F2, …) 아래 레이아웃에서 <CategoryAxis> 와 <Column2DSeries>노드를 참조하십시요.

```
<rMateChart backgroundColor="#FFFFFF" borderStyle="none">
  <Options>
    <Caption text="My First Chart" fontSize="20"/>
    <Legend/>
  </Options>
  <Column2DChart showDataTips="true">
    <horizontalAxis>
      <CategoryAxis categoryField="F0"/>
    </horizontalAxis>
    <verticalAxis>
      <LinearAxis maximum="10"/>
    </verticalAxis>
    <series>
      <Column2DSeries labelPosition="outside" yField="F1" displayName="2011"/>
    </series>
  </Column2DChart>
</rMateChart>
```

2.3 차트 생성 함수

rMateOnLoadCallFunction 함수를 이용한 차트 생성

차트를 생성하기 위한 레이아웃과 데이터셋의 작성이 완료되면 알메이트 차트에서 제공하는 API 함수를 이용하여 차트를 생성할 수 있습니다. 차트를 생성하기 위해서 호출하는 함수는 rMateChartH5.create() 입니다. 다음은 rMateOnLoadCallFunction 함수를 이용하여 차트를 생성하는 자바스크립트 문장입니다.

```
var chartVars = "rMateOnLoadCallFunction=chartReadyHandler";

rMateChartH5.create("chart1", "chartHolder", chartVars, "100%", "100%");

function chartReadyHandler(id) {
  document.getElementById(id).setLayout(layoutStr);
  document.getElementById(id).setData(chartData);
}
```

1. chartVars 변수 : 알메이트 차트를 생성할 때 반드시 필요한 변수이며, 다음 설명을 참조하십시요.

   - 스트링 변수이며 변수의 이름은 변경하여도 관계없습니다. 만약 변수 이름을 다르게 사용할 경우, rMateChartH5.create() 함수의 파라미터에도 변경된 이름의 변수명을 사용해야 합니다.
   - 변수에 저장되는 값은 “name=value” 형식의 쌍으로 구성되며, 각 “name=value” 쌍의 구분은 “&”을 구분자로 이용합니다. 다음은 레이아웃 파일의 URL 과 XML 형식의 데이터셋의 URL 을 “&” 구분자를 지정하여 설정한 예제입니다.

```
var layoutURL = "./Column_3D_Layout.xml";

var chartVars = "layoutURL="+layoutURL;

var dataURL = "./singleData.xml";

chartVars += "&dataURL="+dataURL;
```

   - 차트 생성 준비가 완료되면 실행될 함수명을 지정하는 rMateOnLoadCallFunction 값은 반드시 chartVars 변수에 저장되어야 합니다.

```
var chartVars = "rMateOnLoadCallFunction=chartReadyHandler";
```

다음은 chartVars 변수에 설정이 가능한 값들과 이에 대한 설명입니다.

속성명	유효값 (*: 기본값)	Description
layoutURL	URL	외부 파일을 레이아웃으로 이용할 경우, 이 파일에 대한 URL 을 설정합니다.
dataURL	URL	외부 파일을 데이터셋으로 이용할 경우, 이 파일에 대한 URL 을 설정합니다
rMateOnLoadCallFunction	자바스크립트 함수명	차트를 생성할 준비가 완료된 후 호출할 함수를 설정합니다. (1 회만 호출됨) 파라미터: id – 차트 생성시 사용자가 지정한 id. (rMateChartH5.create() 함수의 첫 번째 파라미터)
displayCompleteCallFunction	자바스크립트 함수명	차트에 데이터가 모두 렌더링된 후 호출할 함수를 설정합니다. 파라미터: id – 차트 생성시 사용자가 지정한 id. (rMateChartH5.create() 함수의 첫 번째 파라미터)
useDataEditor	true, false(*)	생성되는 차트에 데이터 에디터를 사용할지 여부를 설정합니다.
usePattern	true, false(*)	생성되는 차트에 패턴 이미지를 적용할지 여부를 설정합니다.

주의

rMateOnLoadCallFunction 과 displayCompleteCallFunction 의 차이점:

rMateOnLoadCallFunction 에 설정되는 함수는 차트를 생성하기 위한 준비가 완료된 후 호출되는 함수 입니다. 이 함수가 실행될 때 레이아웃과 데이터셋을 차트에 설정하는 작업을 하게됩니다. 그러나 displayCompleteCallFunction 에 설정되는 함수는 차트에 레이아웃과 데이터셋이 설정되어 차트 생성이 완료되었을 경우에 호출됩니다. 차트 생성이 마무리된 시점에 어떤 작업을 해야 할 경우 유용하게 활용되는 함수입니다.

2. rMateChartH5.create() : 차트 생성 작업을 시작하는 함수입니다. 다음은 rMateChartH5.create() 함수를 호출한 예입니다. 함수에 사용되는 파라미터에 대한 설명은 아래 표와 같습니다.

```
rMateChartH5.create("chart1", "chartHolder", chartVars, "100%", "100%");
...
<body>
  <div id="chartHolder" style="width:600px; height:400px;"></div>
</body>
```

파라미터	예제에서 적용된 값	설명
첫 번째 파라미터	"chart1"	차트의 식별자(id). 임의의 id 를 지정할 수 있습니다.
두 번째 파라미터	"chartHolder"	차트가 위치할 의 식별자(id) 반드시 해당 id 의 가 HTML 문서 내에 존재해야 합니다.
세 번째 파라미터	chartVars	chartVars 변수명
네 번째 파라미터	"100%"	차트의 가로 크기 차트가 위치할 에 지정된 width 스타일에 대한 비율입니다. 생략시 100% 가 적용됩니다.
다섯 번째 파라미터	"100%"	차트의 세로 크기 차트가 위치할 에 지정된 height 스타일에 대한 비율입니다. 생략시 100% 가 적용됩니다.

주의

위 예제에서 차트가 위치할 의 크기는 가로 600px, 세로 400px 로 지정이 되었고, rMateChartH5.create() 함수에서는 가로, 세로 각각 100% 로 설정이 되었으므로 결과적으로 생성되는 차트의 크기는 가로 600px, 세로 400px 가 됩니다. rMateChartH5.create() 함수에서

> 크기를 100% 로 지정하더라도 차트가 위치할 에 크기를 지정하지 않았거나 차트를 표현하기에 충분한 크기가 지정되지 않으면 차트가 정상적으로 표시되지 않을 수 있으니 주의하십시요.

3. rMateOnLoadCallFunction 에 지정되는 함수

rMateOnLoadCallFunction 에 지정되는 함수는 차트를 생성할 준비가 완료된 시점에서 호출됩니다. 함수 내에서는 보통 다음과 같이 레이아웃과 데이터셋을 설정하는 알메이트 차트가 제공하는 API 함수를 호출합니다.

```
function chartReadyHandler(id) {
  document.getElementById(id).setLayout(layoutStr);
  document.getElementById(id).setData(chartData);
}
```

위 예제는 아래와 같이 스트링 변수에 저장된 레이아웃과 JSON 형식의 배열 변수에 저장된 데이터셋을 차트에 설정하는 방식의 코드입니다.

```
var layoutStr =
  '<rMateChart backgroundColor="#FFFFFF" borderStyle="none">'
    +'<Options>'
      +'<Caption text="My First Chart" fontSize="20"/>'
      +'<Legend/>'
    +'</Options>'
    +'<Column2DChart showDataTips="true">'
      +'<horizontalAxis>'
        +'<CategoryAxis categoryField="Month"/>'
      +'</horizontalAxis>'
      +'<verticalAxis>'
        +'<LinearAxis maximum="10"/>'
      +'</verticalAxis>'
      +'<series>'
        +'<Column2DSeries labelPosition="outside" yField="2011"
       displayName="2011"/>'
      +'</series>'
    +'</Column2DChart>'
  +'</rMateChart>';

var chartData =
  [{"Month":"Jan", "2011":2.4},
  {"Month":"Feb", "2011":3.8},
  {"Month":"Mar", "2011":8.1},
  {"Month":"Apr", "2011":5.1},
  {"Month":"May", "2011":2.1},
  {"Month":"Jun", "2011":5.2},
  {"Month":"Jul", "2011":4.2}];
```

레이아웃과 데이터셋을 차트에 설정하는 함수는 두 가지 유형이 있습니다. 한 가지는 자바스크립트 내의 변수에 저장된 값을 설정하는 것이고(setLayout(), setData()) 다른 한 가지는 자바스크립트 외부에 저장된 파일을 설정하는 것입니다. (setLayoutURL(), setDataURL()) 데이터셋을 외부 파일을 이용해서 설정할 경우 기본 형식은 XML 입니다. 만약 JSON 형식의 배열이나 CSV 형식을 이용해야 할 경우에는 setDataType() 함수를 setDataURL() 함수가 호출되기 전에 먼저 호출해야 합니다. 이 때 setDataType() 함수의 파라미터 값은 JSON 형식의 배열일 경우에는 “json” (setDataType(“json”)), CSV 형식일 경우에는 “csv” (setDataType(“csv”)) 입니다. 다음은 레이아웃과 데이터셋을 호출하는 함수의 가능한 조합을 표로 나타낸 것입니다.

방법	레이아웃	데이터셋	사용 함수
1	XML 형식의 파일이 존재하는 경로	XML 형식의 파일이 존재하는 경로	setLayoutURL(url), setDataURL(url)
2	XML 형식의 파일이 존재하는 경로	배열 변수	setLayoutURL(url), setData(array variable)
3	XML 형식의 파일이 존재하는 경로	XML 형식의 스트링 변수	setLayoutURL(url), setData(string variable)
4	XML 형식의 스트링 변수	XML 형식의 파일이 존재하는 경로	setLayout(string variable), setDataURL(url)
5	XML 형식의 스트링 변수	배열 변수	setLayout(string variable), setData(array variable)
6	XML 형식의 스트링 변수	XML 형식의 스트링 변수	setLayout(string variable), setData(string variable)
7	XML 형식의 스트링 변수	CSV (혹은 배열) 형식의 파일이 존재하는 경로	setDataType(“csv”) (혹은 setDataType(“json”)), setLayoutURL(url), setDataURL(url)

rMateChartH5.call 함수를 이용한 차트 생성

rMateChartH5.call 함수를 이용하여 좀 더 간단한 방식으로 차트를 생성할 수 있습니다. 이 경우에는 rMateChartH5.create() 함수의 세번째 파라미터에 chartVars 변수를 지정할 필요가 없습니다. rMateOnLoadCallFunction 함수에 의해서 차트의 레이아웃과 데이터셋을 설정하지 않고 rMateChartH5.call 함수를 실행하여 레이아웃과 데이터셋을 설정하기 때문입니다.

```
rMateChartH5.create("chart1", "chartHolder", "", "100%", "100%");
```

다음은 rMateChartH5.call 함수를 통해서 레이아웃과 데이터셋을 설정하는 코드입니다.

```
rMateChartH5.call("chart1", "setLayout", layoutStr);

rMateChartH5.call("chart1", "setData", chartData);
```

rMateChartH5.calls 함수는 한번에 실행될 자바스트립트 함수를 여러 개 등록할 수 있습니다. 위 예제에서는 레이아웃(setLayout)과 데이터셋(setData)을 설정하기 위한 자바스크립트 함수를 rMateChartH5.call 함수를 이용하여 각각 등록하였지만 다음 예제에서는 rMateChartH5.calls 함수를 이용하여 레이아웃(setLayout)과 데이터셋(setData)을 설정하기 위한 자바스크립트 함수를 한번에 등록하는 것을 보여줍니다. 처음 차트 만들기에서 예제 코드의 실행을 확인하실 수 있습니다.

```
rMateChartH5.calls("chart1", {
  "setLayout" : layoutStr,
  "setData" : chartData
});
```

2.4 버전 조회

현재 사용중인 알메이트 차트의 버전 정보는 기술지원 요청시 혹은 제품의 업그레이드를 위해서 알고계실 필요가 있습니다. 현재 운영중인 제품의 버전 정보는 rMateChartH5.version 을 조회하여 확인할 수 있습니다. 다음과 같이 자바스크립트 alert() 함수를 이용하여 알메이트 차트의 버전 정보를 조회할 수 있습니다.

```
alert(rMateChartH5.version);
```

2.5 API 함수

알메이트 차트는 자바스크립트 API 를 제공함으로써 차트 개발자에게 좀 더 유연한 개발 환경을 제공합니다. 제공되는 API 함수 리스트는 아래 표와 같습니다.

API 함수	설명
setData(value)	차트 생성시 적용될 데이터셋을 설정합니다. value: 배열, XML 형식의 스트링
setDataDrillDown(value)	설정된 데이터셋을 이용하여 드릴 다운을 실행합니다. value: 배열, XML 형식의 스트링
setDataDrillDownURL(url)	설정된 경로의 데이터셋을 이용하여 드릴 다운을 실행합니다. value: 경로값
setDataType(value)	차트 생성시 적용될 데이터셋의 형식을 설정합니다. value: csv, json, xml
setDataURL(value)	차트 생성시 적용될 데이터셋에 대한 경로(URL)를 설정합니다. value: 경로값
setLayout(value)	차트 생성시 적용될 레이아웃을 설정합니다. value: XML 형식의 스트링
setLayoutURL(value)	차트 생성시 적용될 레이아웃에 대한 경로(URL)를 설정합니다. value: 경로값
setSlideDataSet(value)	슬라이드 차트 생성시 데이터셋의 배열을 설정합니다. value: 배열
setSlideLayoutSet(value)	슬라이드 차트 생성시 레이아웃의 배열을 설정합니다. value: 배열

setTheme(value)	rMateChartH5.registerTheme() 함수를 통해서 설정된 디자인 테마를 적용합니다. value: 테마명
getCSVData()	차트의 데이터를 CSV 형식으로 리턴합니다.
getDrillDownDepth()	현재 드릴 다운된 단계(depth)를 리턴합니다.
getSnapshot()	base64 로 인코딩된 차트의 스냅샷을 리턴합니다.
saveAsImage()	차트의 스냅샷을 이미지 형식으로 저장합니다.
showAdditionalPreloader()	프리로더 이미지를 표시합니다.
showDataEditor()	숨겨진 데이터 편집기를 표시합니다.
hideDataEditor()	표시된 데이터 편집기를 감추기합니다.
removeAdditionalPreloader()	프리로더 이미지를 표시하지 않습니다.
sliderPause()	모션 차트에서 슬라이드 실행을 멈추기합니다.
sliderPlay()	모션 차트에서 슬라이드를 실행합니다.
sliderReset()	모션 차트에서 슬라이드를 초기화합니다.
legendCheck(value)	<Legend> 노드의 useVisibleCheck 속성값이 “true” 인 경우, 파라미터값의 인덱스의 시리즈에 대한 선택/해제를 실행합니다. value: 숫자
legendAllCheck(value)	<Legend> 노드의 useVisibleCheck 속성값이 “true” 인 경우 전체 범례 아이템에 대한 선택(true) 혹은 해제(false)를 실행합니다. value: true, false
changeScrollBarSize(value)	스크롤 차트에서 스크롤바의 크기를 설정합니다. value: 객체 (arrowSize, scrollHeight, thumbHeight, thumbMinSize)
hasNoData()	데이터가 존재하지 않는다는(null) 창을 차트에 표시합니다.
resize()	차트의 크기를 변경합니다. resize() 함수를 실행하기 전에 차트가 표시되는 의 width, height 스타일에 변경될 크기값을 지정해야 합니다.

visibleItemSize(value)	스크롤 차트의 화면에 표시될 데이터의 개수를 설정합니다. value: 숫자

3. 차트 구성 요소

3.1 데이터 시리즈

차트의 레이아웃을 구성하는 여러가지 요소들 중에서 데이터 시리즈는 가장 중요한 요소입니다. 적용되는 데이터셋과의 연결과 데이터 포인트들이 차트 상에 표현되는 다양한 방식은 데이터 시리즈의 설정을 통해서 이루어집니다. 알메이트 차트의 레이아웃에서 데이터 시리즈는 <“차트의 종류명” + “Series”> 로 표시되고, 차트 노드(<“차트의 종류명” + “Chart”>)의 series 속성에 정의됩니다. 예들 들면, 레이아웃에서 영역 차트의 데이터 시리즈는 <Area2DSeries> 로 표시되고 차트 노드는 <Area2DChart> 로 표시됩니다. 다음은 처음 차트 만들기에서 예제에 적용된 컬럼 차트 생성을 위한 데이터 시리즈 설정 부분입니다.

```
<Column2DChart showDataTips="true">
...
  <series>
    <Column2DSeries labelPosition="outside" yField="2011" displayName="2011"/>
  </series>
</Column2DChart>
```

데이터 필드 설정

차트에 데이터 값을 표현하기 위해서는 데이터셋의 특정 데이터 필드를 데이터 시리즈의 특정한 속성값에 지정해야 합니다. 지정할 속성은 차트의 유형에 따라서 달라지는데, 데이터 값의 크기가 세로 축(Y 축) 상에 표현된다면 yField(예, 컬럼 차트, 영역 차트 등) 속성에, 가로 축(X 축) 상에 표현된다면 xField(예, 바 차트) 속성에 그리고 기타 차트 유형에 따라서(예, 파이 차트: field, 피라미드 차트: weightField) 정해진 속성에 지정해야만 합니다. 위 컬럼 차트 예제에서는 yField 속성에 데이터 필드명 “2011” 이 지정되었습니다.

다중 데이터 시리즈 설정

차트에 표현될 데이터 시리즈가 여러 개일 경우에는 표현하기 원하는 개수만큼의 데이터 시리즈 노드를 정의하면 됩니다. 다음 코드 예제는 컬럼 차트에서 3 개의 데이터 시리즈를 표현하기 위한

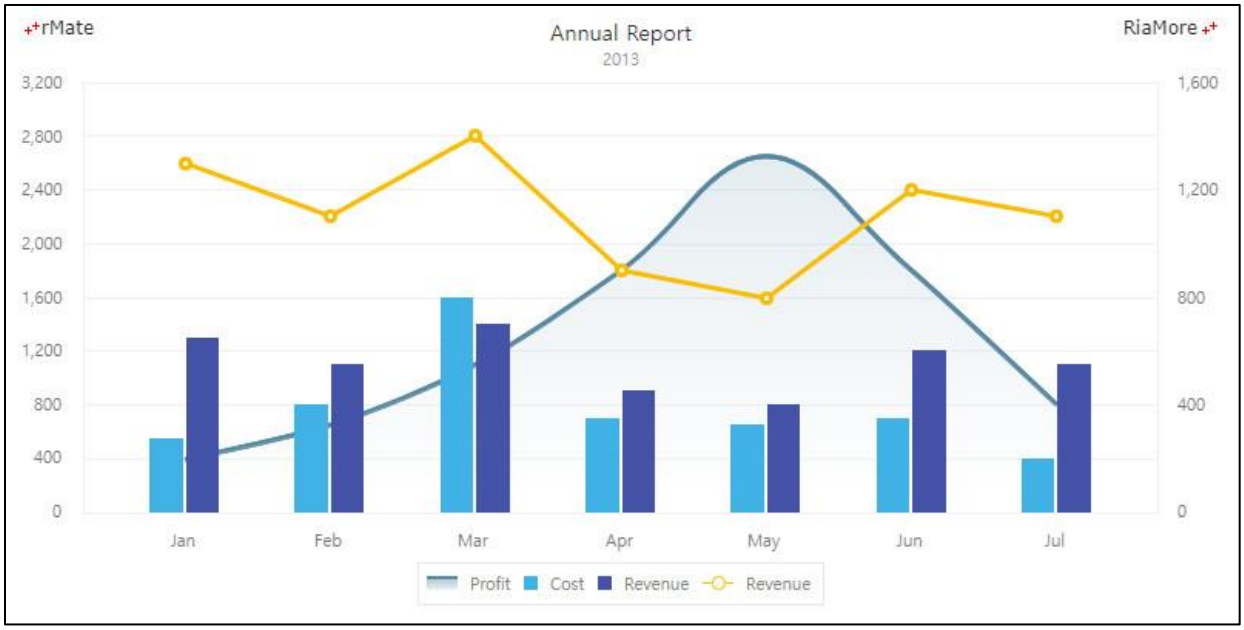
레이아웃과 데이터셋을 나타낸 것입니다. 위에 설명된 예제에서 “2012”, “2013” 데이터 필드가 추가되어 추가된 2 개의 데이터 시리즈에 지정되었습니다.

```
<Column2DChart showDataTips="true">
...
  <series>
    <Column2DSeries labelPosition="outside" yField="2011" displayName="2011"/>
    <Column2DSeries labelPosition="outside" yField="2012" displayName="2012"/>
    <Column2DSeries labelPosition="outside" yField="2013" displayName="2013"/>
  </series>
</Column2DChart>

var chartData =
  [{"Month":"Jan", "2011":2.4, "2012":3.1, "2013":2.5},
  {"Month":"Feb", "2011":3.8, "2012":4.6, "2013":1.3},
  {"Month":"Mar", "2011":8.1, "2012":9, "2013":6.0},
  {"Month":"Apr", "2011":5.1, "2012":4, "2013":7.7},
  {"Month":"May", "2011":2.1, "2012":3.9, "2013":1.9},
  {"Month":"Jun", "2011":5.2, "2012":4.9, "2013":1.9},
  {"Month":"Jul", "2011":4.2, "2012":3.2, "2013":1.9}]
```

차트의 유형 중 콤비네이션 차트에서는 두 가지 이상의 서로 다른 종류의 데이터 시리즈가 차트 노드의 series 속성에 정의됩니다. 콤비네이션 차트에서 특정 데이터 시리즈를 다중 데이터 시리즈로 표현하고자 할 경우에는 차트 노드의 <series> 속성에 해당 데이터 시리즈들를 바로 설정하지 않고 반드시 <“차트의 종류명” + “Set”>(예, <Column2DSet>) 노드를 정의 하고 정의된 노드의 <series> 속성에 데이터 시리즈들을 정의해야 합니다. 다음은 1 개의 영역 시리즈, 2 개의 컬럼 시리즈, 1 개의 라인 시리즈를 정의한 레이아웃과 이를 적용해서 출력한 차트의 예제입니다. 2 개의 컬럼 시리즈를

```
<series>
  <Area2DSeries yField="Profit" form="curve" displayName="Profit">
    ...
  </Area2DSeries>
  <Column2DSet type="clustered">
    <series>
      <Column2DSeries yField="Cost" displayName="Cost">
        ...
      </Column2DSeries>
      <Column2DSeries yField="Revenue" displayName="Revenue">
        ...
      </Column2DSeries>
    </series>
  </Column2DSet>
  <Line2DSeries yField="Revenue" radius="4.5" itemRenderer="CircleItemRenderer"
       displayName="Revenue">
    ...
    <verticalAxis>
      <LinearAxis id="vAxis2" interval="400" formatter="{numFmt}"/>
    </verticalAxis>
    ...
  </Line2DSeries>
</series>
```

표현하기 위해서 <Column2DSet> 노드가 정의되었고 이 노드의 <series>
속성에 <Column2DSeries> 노드 2 개가 설정되었습니다.



See the CodePen 알메이트 차트 - 다중 데이터 시리즈 설정

데이터 시리즈와 차트 유형

다음은 알메이트 차트에서 사용 가능한 데이터 시리즈에 대한 리스트입니다. 데이터 시리즈가 적용되는 차트 유형과 데이터셋의 필드명을 지정해야 하는 속성도 함께 표시되었습니다.

데이터 시리즈	차트 유형	데이터 필드 설정 속성
<Area2DSeries> (<Area2DSet>)	<Area2DChart>	yField
<Bar2DSeries> (<Bar2DSet>)	<Bar2DChart>	xField
<Bar2DWingSeries>	<Bar2DWingChart>	xField, xFieldOpp
<Bar3DSeries> (<Bar3DSet>)	<Bar3DChart>	xField

<BoxPlotSeries>	<BoxPlotChart>	yField
<Bubble2DSeries>	<Bubble2DChart>	xField, yField, radiusField
<Bubble3DSeries>	<Bubble3DChart>	xField, yField, radiusField
<CandleArea2DSeries>	<CandleArea2DChart>	yField
<CandleLine2DSeries>	<CandleLine2DChart>	yField
<Candlestick2DSeries>	<Candlestick2DChart>	openField, closeField, highField, lowField
<Column2DSeries> (<Column2DSet>)	<Column2DChart>	yField
<Column2DWingSeries>	<Column2DWingChart>	yField, yFieldOpp
<Column3DSeries> (<Column3DSet>)	<Column3DChart>	yField
<Equalizer2DSeries>	<Equalizer2DChart>	yField
<Histogram2DSeries>	<Histogram2DChart>	yField
<Histogram3DSeries>	<Histogram3DChart>	yField
<HTarget2DGoalSeries>	<Combination2DChart>	xField
<HTarget2DResultSeries>	<Combination2DChart>	xField
<HTarget3DGoalSeries>	<Combination3DChart>	xField
<HTarget3DResultSeries>	<Combination3DChart>	xField
<ImageMatrixSeries> (<ImageMatrixSet>)	<ImageMatrixChart>	field
<ImageSeries>	<ImageChart>	yField
<Line2DSeries>	<Line2DChart>	yField
<Matrix2DSeries>	<Matrix2DChart>	xField, yField, zField
<MotionBubbleSeries>	<MotionChart>	yField, radiusField
<MotionColumnSeries>	<MotionChart>	yField
<MotionLineSeries>	<MotionChart>	yField
<OverlayBubbleSeries>	<OverlayBubbleChart>	field, nameField
<Pie2DSeries>	<Pie2DChart>	field, nameField
<Pie3DSeries>	<Pie3DChart>	field, nameField
<Pyramid2DSeries>	<Pyramid2DChart>	nameField, weightField
<RadarSeries>	<RadarChart>	field

<TreeMapSeries>	<TreeMapChart>	nameField, groupField, groupNameField, weightField
<Vector2DSeries>	<Vector2DChart>	xField , degreeField, meterField, velocityField
<VTarget2DGoalSeries>	<Combination2DChart>	yField
<VTarget2DResultSeries>	<Combination2DChart>	yField
<VTarget3DGoalSeries>	<Combination3DChart>	yField
<VTarget3DResultSeries>	<Combination3DChart>	yField
<WindRoseSeries> (<WindRoseSet>)	<WindRoseChart>	field
<WordCloudSeries>	<WordCloudChart>	textField, weightField

3.2 축과 스케일

알메이트 차트에서 제공되는 축의 종류는 다음과 같습니다.

- 숫자 축(Numeric Axis) : 숫자를 표현하는 축입니다. <LinearAxis>, <LogAxis>, <DateTimeAxis>, <BrokenAxis> 유형이 지원됩니다.
- 카테고리 축(Category Axis) : 수치화되지 않은 카테고리를 표현하는 축입니다. <CategoryAxis>, <CategoryLinearAxis>, <HistogramCategoryAxis> 유형이 지원됩니다.
- 컬러 축(Color Axis) : 트리맵 차트에서 색과 데이터 값을 연관시켜 표현하는 축의 유형입니다. <ColorAxis> 유형이 지원됩니다.

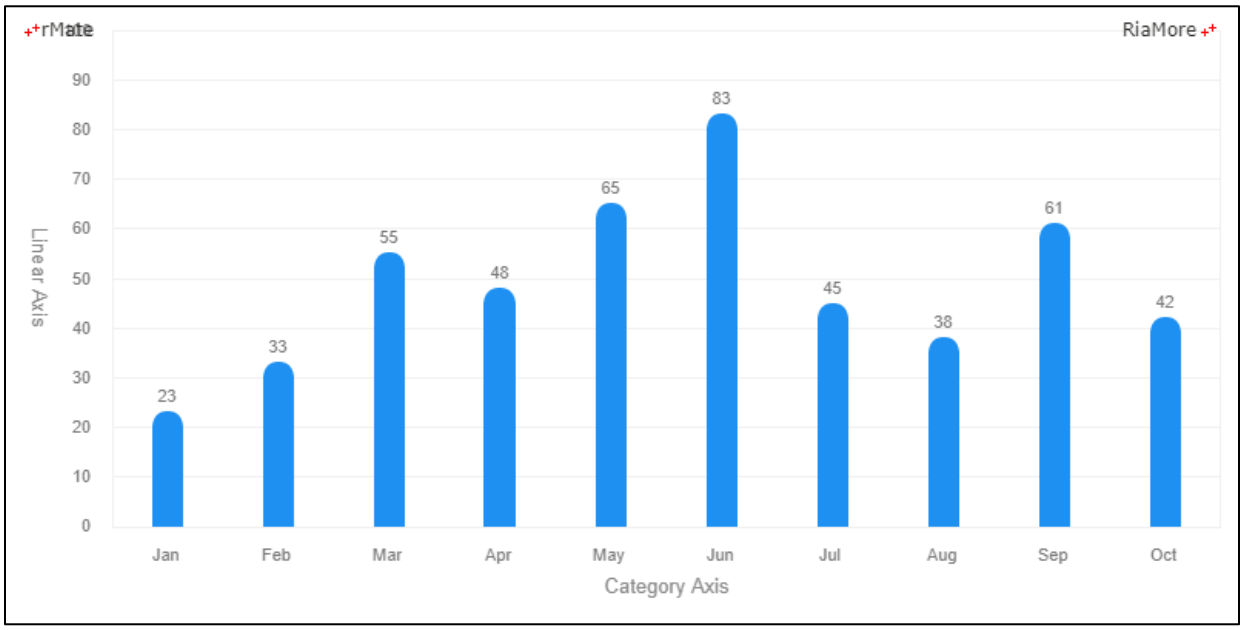
숫자 축

1. <LinearAxis> - 숫자 축에서 가장 활용도가 높은 축의 유형입니다. 숫자 레이블의 간격, 최대값, 최소값은 적용되는 데이터의 값에 따라서 자동으로 설정되고, 제공되는 속성을 이용하여 사용자가 원하는 숫자 레이블의 간격, 최대값, 최소값 등을 설정할 수 있습니다. 다음 표는 <LinearAxis> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
interval	숫자	축에 표시되는 레이블의 간격을 지정합니다.
labelJsFunction	자바스크립트 함수명	사용자 정의 함수를 이용하여 축의 레이블을 표시할 경우, 해당 자바스크립트 함수명을 지정합니다.
maximum	숫자	축의 최대값을 지정합니다.
minimum	숫자	축의 최소값을 지정합니다.
title	텍스트	축의 제목을 지정합니다.

다음은 세로 축(<verticalAxis> 노드)에 <LinearAxis> 노드를 설정하여 컬럼 차트를 생성하는 예제입니다. 이 예제에서는 가로 축(<horizontalAxis> 노드)에 <CategoryAxis> 노드가 설정되었습니다.

```
<verticalAxis>
  <LinearAxis minimum="0" maximum="100" interval="10" title="Linear Axis"/>
</verticalAxis>
<horizontalAxis>
  <CategoryAxis categoryField="Month" title="Category Axis"/>
</horizontalAxis>
```



See the CodePen 알메이트 차트 - Y 축에 LinearAxis, X 축에 CategoryAxis 설정

2. <LogAxis> - 로그 축은 축 레이블의 간격이 설정된 값의 10 의 지수로 계산된 숫자 축으로써 큰 숫자의 데이터를 차트에 표현할 때 유용하게 활용됩니다. 다음 표는 <LogAxis> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
interval	숫자	축에 표시되는 레이블의 간격을 지정합니다. 지정된 값의 로그값(상용 로그)을 10 의 지수로 계산한 값이 레이블 간격이 됩니다.
labelJsFunction	자바스크립트 함수명	사용자 정의 함수를 이용하여 축의 레이블을 표시할 경우, 해당 자바스크립트 함수명을 지정합니다.

maximum	숫자	축의 최대값을 지정합니다. 지정된 값의 로그값(상용 로그)을 10 의 지수로 계산한 값이 레이블 간격이 됩니다.
minimum	숫자	축의 최소값을 지정합니다. 지정된 값의 로그값(상용 로그)을 10 의 지수로 계산한 값이 레이블 간격이 됩니다.
title	텍스트	축의 제목을 지정합니다.

주의

로그 축에 설정된 간격, 최대값, 최소값이 축에 표현될 때는 해당값의 로그값(상용 로그)을 10 의 지수로 계산한 값이 실제 표시되는 값이 됩니다. 예를 들면, minimum="5" 와 같이 설정이 되었다면, 5 에 대한 로그값은 0.7 이고, 이 값의 반올림인 1 을 10 의 지수로 계산한 10 이 축에 표현됩니다. interval="100"으로 설정되었다면, 100 의 로그값인 2 를 10 의 지수로 계산한 값의 단위에 의해서 간격이 표시됩니다. (10, 1,000, 100,000, …)

3. <DateTimeAxis> - 날짜와 시간을 표현하는 축의 유형입니다. 다음 표는 <DateTimeAxis> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
dataInterval	숫자	dataUnits 속성에 지정된 단위로 차트에 표시할 데이터의 간격을 지정합니다. 예를 들어, dataUnits="seconds" 이고, 3 초의 데이터 단위로 차트를 표현하고자 한다면 dataInterval="4" 로 지정합니다. (차트의 종류에 따라서 이 속성은 무시됩니다.)
dataUnits	milliseconds, seconds, minutes, hours, days,	데이터의 표시 단위를 지정합니다.

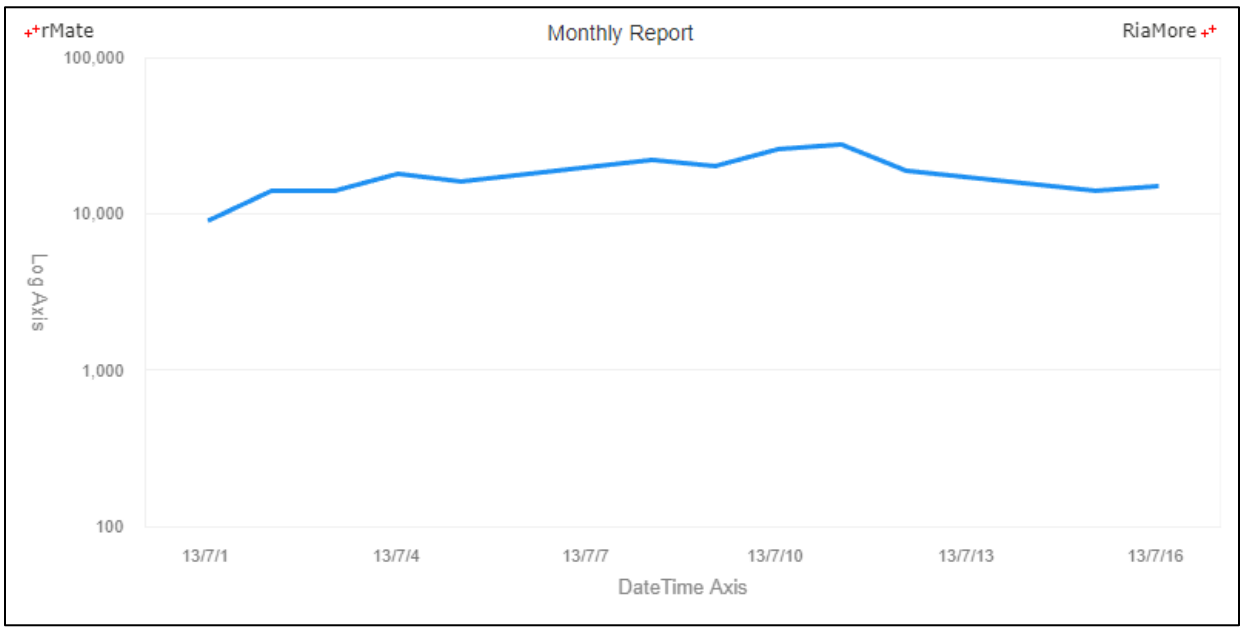
	weeks, months, years	
displayLocalTime	true, false(*)	true: 클라이언트 PC 의 타임 존을 적용합니다. false: 세계 표준시 (그리니지 표준시)를 적용합니다.
interval	숫자	축에 표시되는 레이블의 간격을 지정합니다. (레이블이 표시될 공간이 충분하지 않으면 이 값은 무시됩니다.)
labelJsFunction	자바스크립트 함수명	사용자 정의 함수를 이용하여 축의 레이블을 표시할 경우, 해당 자바스크립트 함수명을 지정합니다.
labelUnits	milliseconds, seconds, minutes, hours, days, weeks, months, years	축 레이블의 표시 단위를 지정합니다.
title	텍스트	축의 제목을 지정합니다.

주의

<DateTimeAxis>를 적용하기 위해서는 해당 데이터 시리즈의 field 속성 (예, 라인 차트의 수평축에 <DateTimeAxis> 를 적용할 경우, <Line2DSeries> 노드의 xField 속성)에 반드시 축의 값으로 적용할 데이터 필드명을 지정해야 합니다. 이 때 데이터 필드의 값은 자바스크립트 Date 함수로 변환 가능한 포맷을 가지고 있어야 합니다. (예, YYYY-MM-DD, YYYY/MM/DD, MM-DD-YYYY, etc)

다음은 수평축에 <DateTimeAxis> 를 수직축에 <LogAxis> 를 적용한 예제입니다. 아래 예제에서는 데이터 시리즈 (<Line2DSeries> 노드)의 xField 속성에 날짜/시간 축에 표시될 데이터 필드(Date)가 지정되었습니다.

```
<horizontalAxis>
  <DateTimeAxis title="DateTime Axis" formatter="{dateFmt}" dataUnits="days"
        labelUnits="days" interval="3" alignLabelsToUnits="false"
        displayLocalTime="true" padding="1"/>
</horizontalAxis>
<verticalAxis>
  <LogAxis title="Log Axis" formatter="{numFmt}" interval="10" minimum="100"/>
</verticalAxis>
…
<Line2DSeries xField="Date" yField="Profit" displayName="Profit"
        formatter="{numFmt}">
…
var chartData = [
{
  "Date" : "2013/07/01",
  "Profit" : 9000,
  "Cost" : 15000,
  "Revenue" : 23000
},{
  "Date" : "2013/07/02",
  "Profit" : 14000,
  "Cost" : 14000,
  "Revenue" : 12000
},{
  "Date" : "2013/07/03",
  "Profit" : 14000,
  "Cost" : 12000,
  "Revenue" : 16000
…
}];
```



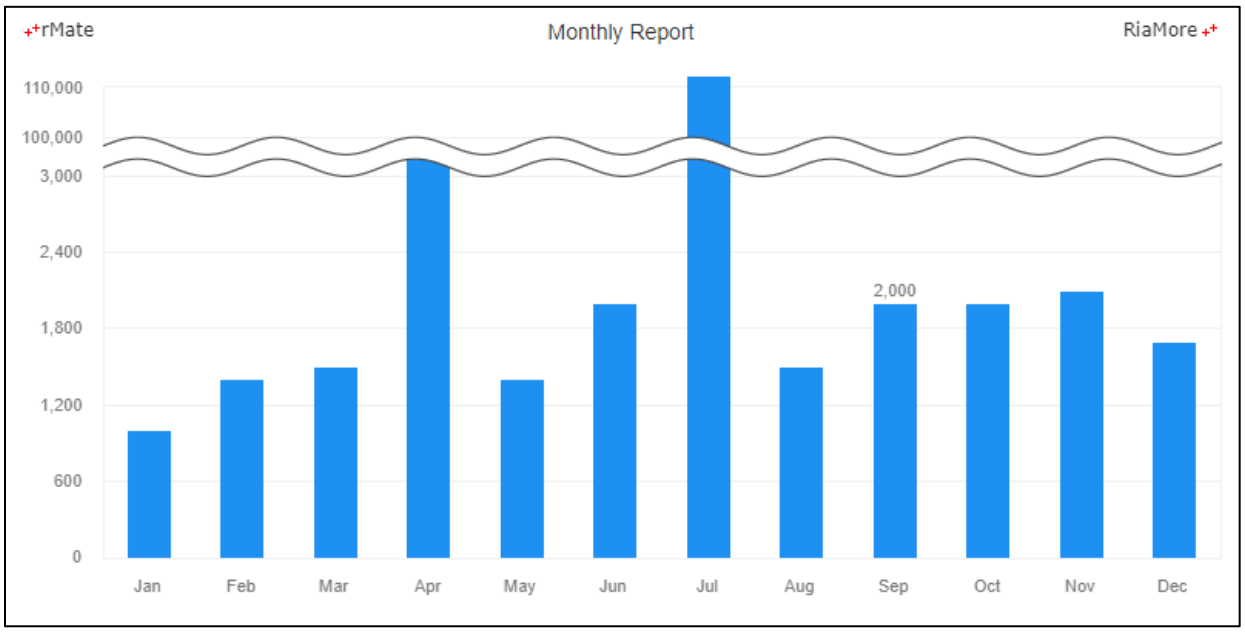
See the CodePen 알메이트 차트 - X 축에 DateTimeAxis 설정

4. <BrokenAxis> - 브로큰 축 차트에서 브로큰 축을 표현하는 축의 유형입니다. 다음 표는 <BrokenAxis> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
brokenMaximum	숫자 기본값: NaN	브로큰 축의 최대값(두 번째 물결무늬가 표시되는 축 상의 지점)을 지정합니다.
brokenMaximumInterval	숫자 기본값: NaN	브로큰 축의 최대값(brokenMaximum)보다 큰 값들이 축에 표시되는 간격을 지정합니다.
brokenMinimum	숫자 기본값: NaN	브로큰 축의 최소값(첫 번째 물결무늬가 표시되는 축 상의 지점)을 지정합니다.
brokenMinimumInterval	숫자 기본값: NaN	브로큰 축의 최소값(brokenMinimum)보다 작은 값들이 축에 표시되는 간격을 지정합니다.
brokenOffset	숫자 기본값: 14	두 물결무늬 사이의 간격을 지정합니다.
brokenRatio	0 과 1 사이의 숫자 기본값: 0.5	축의 시작 지점에서 브로큰 축의 최소값까지의 크기와 브로큰 축의 최대값에서 축의 끝 지점까지의 크기의 비율을 지정합니다.

다음은 수평축에 <CategoryAxis> 를 수직축에 <BrokenAxis> 를 적용한 예제입니다.

```
<horizontalAxis>
  <CategoryAxis categoryField="Month"/>
</horizontalAxis>
<verticalAxis>
  <BrokenAxis id="vAxis" brokenMinimum="3000" brokenMaximum="100000"
       maximum="116000" brokenRatio="0.8" formatter="{numfmt}" />
</verticalAxis>
<verticalAxisRenderers>
  <BrokenAxis2DRenderer axis="{vAxis}"/>
</verticalAxisRenderers>
```



See the CodePen 알메이트 차트 - Y 축에 BrokenAxis 설정

카테고리 축

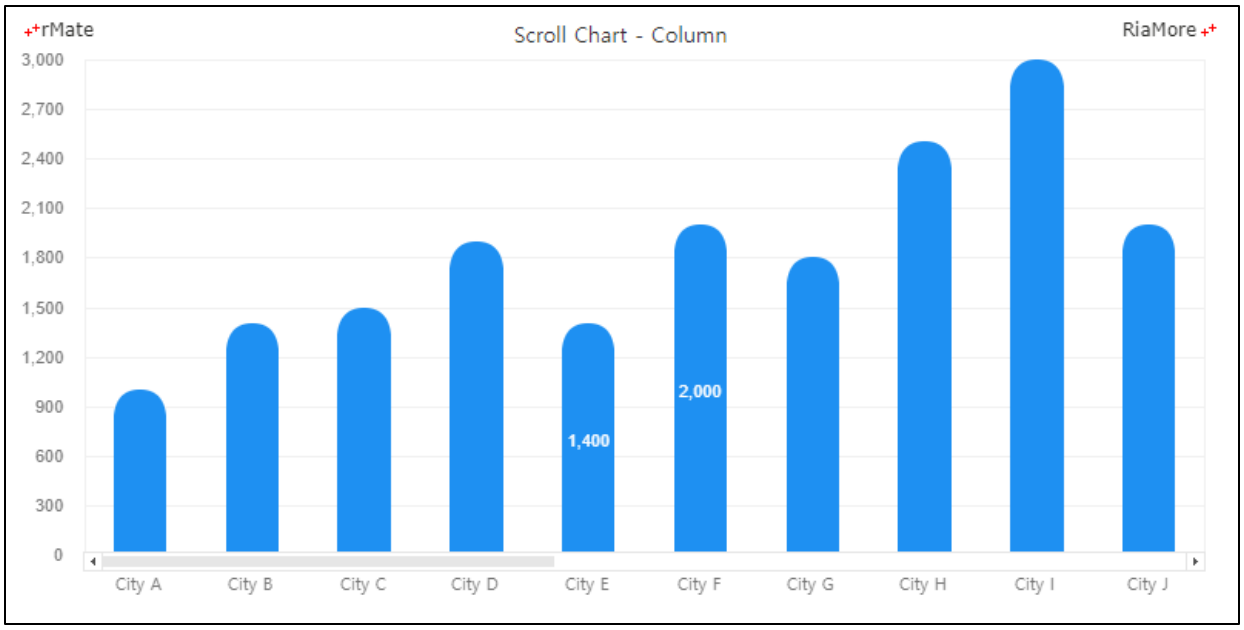
1. <CategoryAxis> - 숫자가 아닌 카테고리를 표현하는 축의 유형입니다. 예를 들면, 연도별, 월별, 부서별, 국가별 등과 같이 수치로 표현되지 않고 카테고리의 항목들이 열거되는 형태입니다. <CategoryAxis> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
categoryField	텍스트	카테고리 값이 저장된 데이터 필드명을 지정합니다.
labelJsFunction	자바스크립트 함수명	사용자 정의 함수를 이용하여 축의 레이블을 표시할 경우, 해당 자바스크립트 함수명을 지정합니다.
title	텍스트	축의 제목을 지정합니다.

카테고리 축에 대한 예제는 위에서 설명된 <BrokenAxis> 예제를 참조하십시요. 이 예제에서는 수직축에 카테고리 축이 적용되었고, categoryField 속성값으로 “Month” 필드가 지정되었습니다.

2. <CategoryLinearAxis> - 카테고리 축을 스크롤 차트에 표현해야 할 경우 <CategoryAxis> 노드를 사용하지 않고 <CategoryLinearAxis> 노드를 사용해야 합니다. 다음은 스크롤 차트의 수평축에 <CategoryLinearAxis> 노드가 적용된 예제입니다

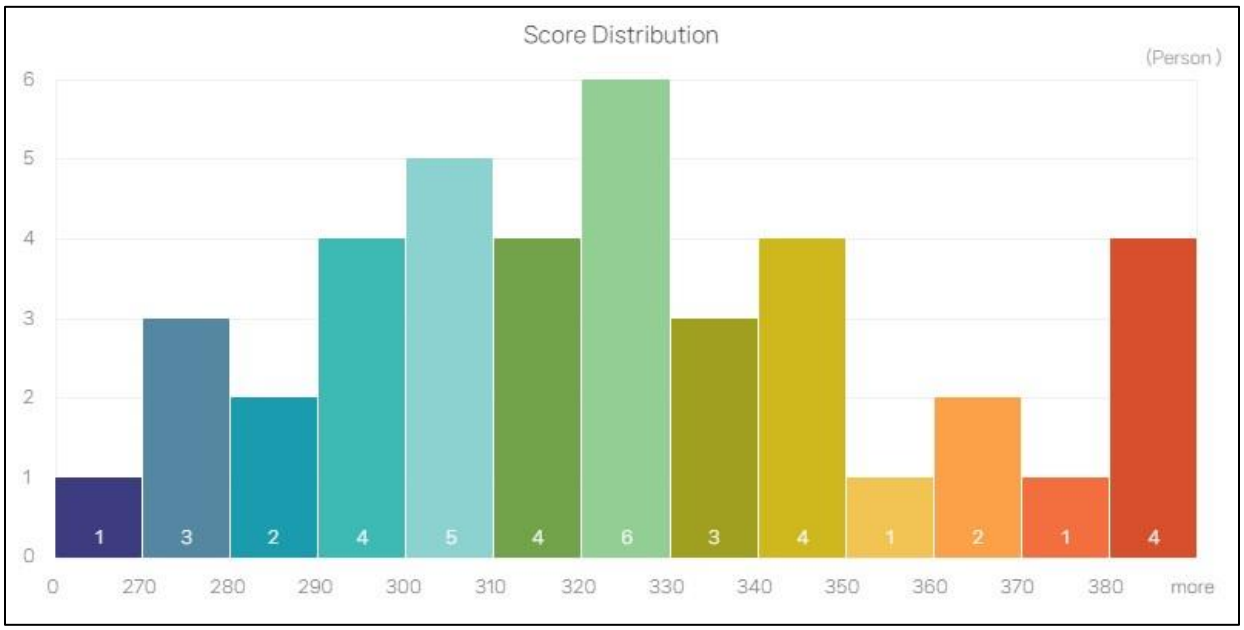
```
<verticalAxis>
  <LinearAxis interval="300" formatter="{numfmt}"/>
</verticalAxis>
<horizontalAxis>
  <CategoryLinearAxis id="hAxis" categoryField="City"/>
</horizontalAxis>
<horizontalAxisRenderers>
  <ScrollableAxisRenderer axis="{hAxis}" visibleItemSize="10" />
</horizontalAxisRenderers>
```



See the CodePen 알메이트 차트 - X 축에 CategoryLinearAxis 설정

3. <HistogramCategoryAxis> - 히스토그램 차트의 X 축을 표현하는 축의 유형입니다. 다음은 히스토그램 차트에 <HistogramCategoryAxis> 노드가 적용된 예제입니다.

```
<horizontalAxis>
  <HistogramCategoryAxis id="hAxis" categoryField="histogramXField"/>
</horizontalAxis>
<horizontalAxisRenderers>
  <HistogramAxis2DRenderer axis="{hAxis}"/>
</horizontalAxisRenderers>
```

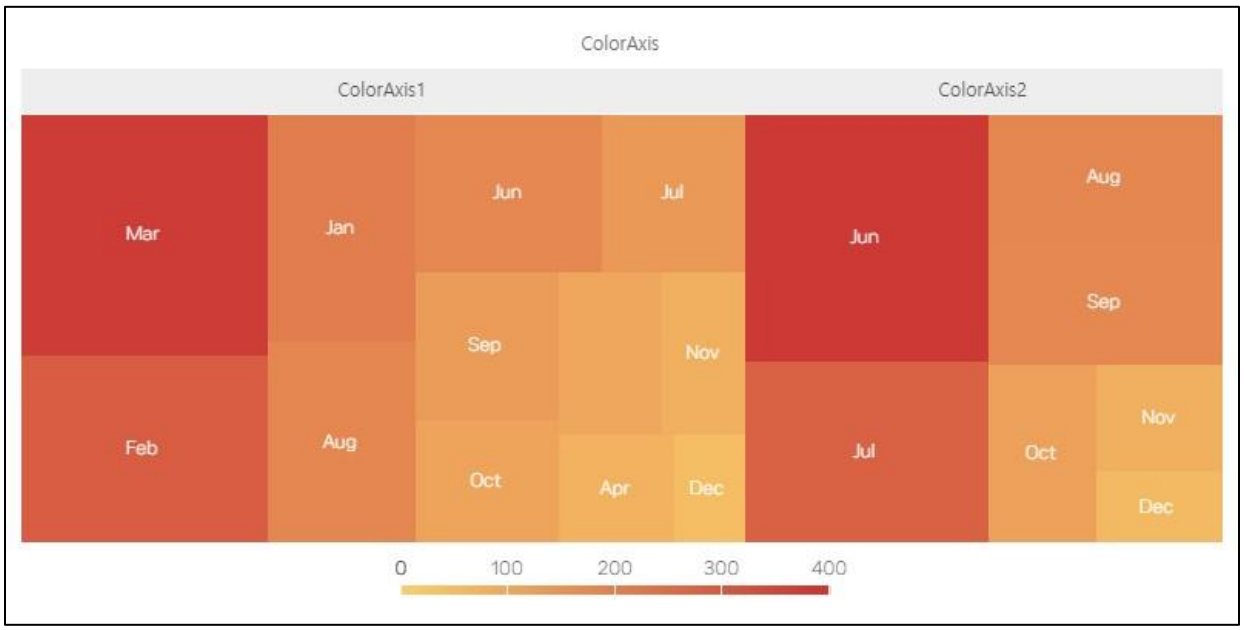


See the CodePen 알메이트 차트 - X 축에 HistogramCategoryAxis 설정

컬러 축

컬러 축은 트리맵 차트에서 색과 데이터 값을 연관시켜 표현하는 축의 유형입니다. 다음은 트리맵 차트에서 컬러 축이 표현된 예제입니다. 이 예제에서는 색의 범위를 colors 속성에 “[#fcd26b,#cc3635]” 으로 지정하였습니다. 최소값(minimum=“0”)의 색은 #fcd26b 이고, 최대값(maximum=“400”)의 색은 “#cc3635” 이며, 컬러축에는 100 (interval=“100”) 단위로 레이블이 표시됩니다. 사각형에 표시되는 색은 컬러축과 데이터 값이 연동되어 자동으로 표시됩니다.

```
<ColorAxis id="colorAxis" maximum="400" minimum="0" interval="100"
       colors="[#fcd26b,#cc3635]"/>
```



See the CodePen 알메이트 차트 - 컬러 축

축의 위치 속성

위에서 설명한 축의 유형들은 차트 노드의 축의 위치 속성의 자식 노드에 정의되어야 합니다. 다음은 축의 위치 속성들을 열거한 표입니다.

속성명	적용되는 차트 유형	설명
<horizontalAxis>	카테시안 계열의 차트	카테시안 계열의 차트에서 수평축 (X 축)
<verticalAxis>	카테시안 계열의 차트	카테시안 계열의 차트에서 수직축 (Y 축)
<angularAxis>	방사형 차트, 윈드 로즈 차트	방사형 차트, 윈드 로즈 차트에서 바깥쪽 원호에 표시되는 카테고리 축
<radialAxis>	방사형 차트, 윈드 로즈 차트	방사형 차트, 윈드 로즈 차트에서 중심에서 원호까지 직선으로 표시되는 숫자 축
<colorAxis>	트리맵 차트	a href="treemap-chart.html">트리맵 차트에 표시되는 컬러 축

축 렌더러

알메이트 차트에서 축의 스타일은 사용자의 기호에 따라서 다양하게 표현이 가능합니다. 축의 색상, 틱, 보조틱, 축의 레이블, 축 제목의 폰트 등 모든 축의 표현 방식은 축 렌더리 노드에 설정됩니다. 다음은 알메이트 차트에서 제공하는 축 렌더러의 종류들을 열거한 표입니다.

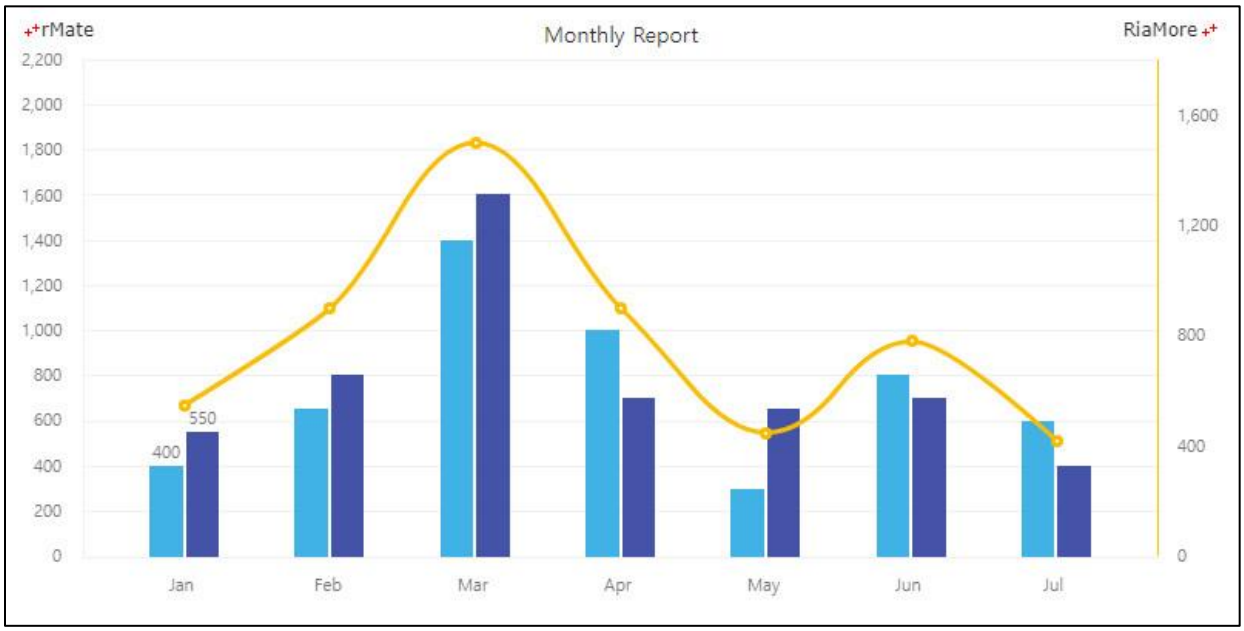
축 렌더러	설명
<AngularAxisRenderer>	방사형 차트, 윈드 로즈 차트에서 <angularAxis>를 표현하는 렌더러입니다.
<Axis2DRenderer>	2 차원 축을 표현하는 렌더러입니다.
<Axis3DRenderer>	3 차원 축을 표현하는 렌더러입니다.
<Axis2DWingRenderer>	윙 차트에서 2 차원 축을 표현하는 렌더러입니다.
<BrokenAxis2DRenderer>	브로큰 축 차트에서 2 차원 축을 표현하는 렌더러입니다.
<BrokenAxis3DRenderer>	브로큰 축 차트에서 3 차원 축을 표현하는 렌더러입니다.
<HistogramAxis2DRenderer>	히스토그램 차트에서 2 차원 축을 표현하는 렌더러입니다.
<ScrollableAxisRenderer>	스크롤 차트에서 스크롤이 가능한 축을 표현하는 렌더러입니다.

축 렌더러의 활용 방식과 예제에 대한 자세한 설명은 축 스타일링과 축 레이블을 참조하십시요.

다중 축 표시

한 차트에서 두 개 이상의 축을 표시해서 데이터 시리즈별로 서로 다른 특정한 축을 기준으로 데이터를 표현하도록 지정할 수 있습니다. 콤비네이션 차트와 같이 서로 다른 유형의 데이터 시리즈가 적용되거나, 동일한 차트에서 데이터 시리즈별로 다른 스케일의 축을 적용하고 싶은 경우에 활용이 가능합니다. <series> 노드의 형제 노드에 정의된 축이 기본적으로 모든 데이터 시리즈에 적용되고, 특정 데이터 시리즈는 기본 축이 아닌 다른 축을 적용하고자 할 경우에는 해당 데이터 시리즈의 자식 노드에 원하는 축을 정의하면 됩니다. 이 때 축 렌더러 노드에서 어떤 축을 표현할지에 대한 구별은 축 정의시에 설정하는 id 속성을 이용합니다. 축의 id 속성값을 중괄호(브레이스, {})로 묶은 값을 렌더러의 axis 속성값으로 설정합니다. 다음은 컬럼 시리즈는 기본축(vAxis1)을 적용하고 라인 시리즈는 레이블 간격이 400 인 별도의 축을 해당 데이터 시리즈의 자식 노드로 정의하여 적용한 예제입니다.

```
<horizontalAxis>
  <CategoryAxis categoryField="Month" padding="0.7"/>
</horizontalAxis>
<verticalAxis>
  <LinearAxis id="vAxis1" formatter="{numFmt}" maximum="2200"/>
</verticalAxis>
<series>
  ...
  <Line2DSeries selectable="true" yField="Revenue" radius="4" form="curve"
       displayName="Revenue" itemRenderer="CircleItemRenderer">
    ...
    <verticalAxis>
      <LinearAxis id="vAxis2" interval="400" formatter="{numFmt}" maximum="1800"/>
    </verticalAxis>
  </Line2DSeries>
</series>
<verticalAxisRenderers>
  <Axis2DRenderer axis="{vAxis1}" showLine="false"/>
  <Axis2DRenderer axis="{vAxis2}" showLine="true">
    <axisStroke>
      <Stroke color="#f9bd03"/>
    </axisStroke>
  </Axis2DRenderer>
</verticalAxisRenderers>
```



See the CodePen 알메이트 차트 - 다중 축 표시

역 Y 축

수직 Y 축에서 최소값이 시작되는 위치를 차트의 하단이 아닌 차트의 상단에 둘 수 있습니다. 이를 역 Y 축이라고 합니다. 역 Y 축을 차트에 표시하기 위해서는 <LinearAxis> 노드의 direction 속성값을 “inverted” 로 지정합니다. 이 때 수평 X 축도 차트의 상단에 위치시키는 것이 좋습니다. 이를 위해서는 축 렌더러 노드의 placement 속성값을 “top” 으로 지정해야 합니다.
다음은 <Axis2DRenderer> 노드의 placement 속성값을 “top” 으로 지정한 역 Y 축에 대한 예제입니다.

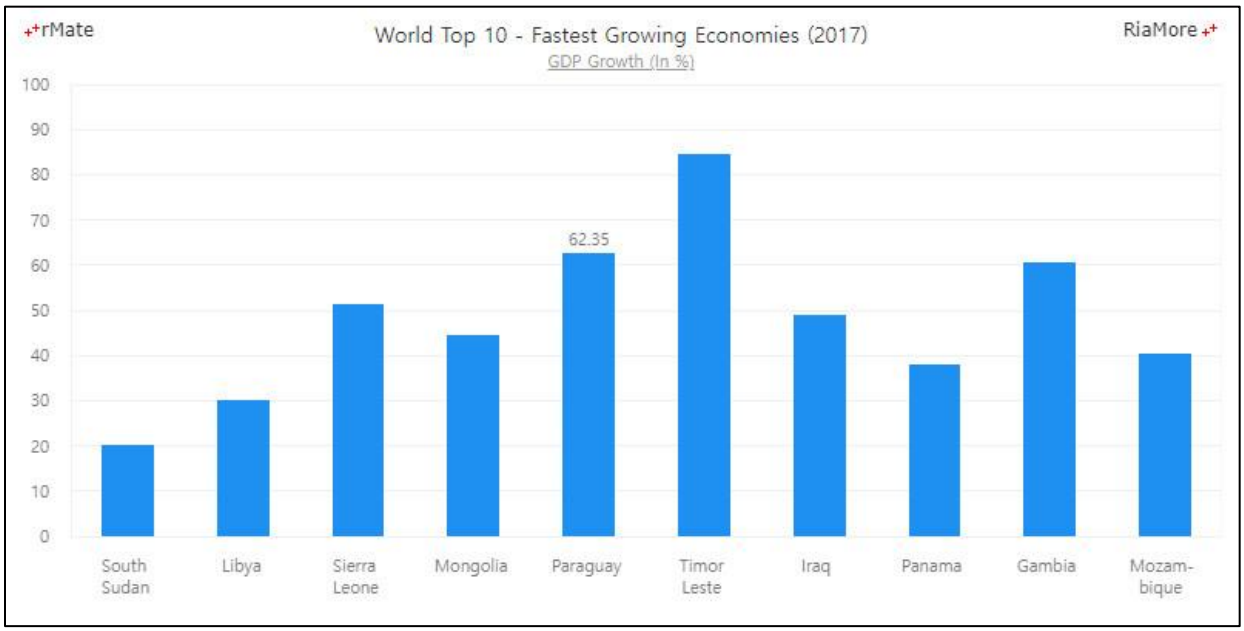
```
<horizontalAxis>
  <CategoryAxis id="hAxis" categoryField="Month" padding="0.5"/>
</horizontalAxis>
<horizontalAxisRenderers>
  <Axis2DRenderer axis="{hAxis}" placement="top" showLine="true"/>
</horizontalAxisRenderers>
<verticalAxis>
  <LinearAxis direction="inverted" interval="400" formatter="{numfmt}"/>
</verticalAxis>
```

See the CodePen 알메이트 차트 - 역 Y 축

3.3 제목

차트의 제목은 <Caption> 노드에 부제목은 <SubCaption> 노드에 각각 정의됩니다. 두 노드 모두 <Options> 노드의 자식 노드로 정의되어야 합니다. 다음은 제목과 부제목을 컬럼 차트에 설정하는 예제입니다.

```
<Options>
<Caption text="World Top 10 - Fastest Growing Economies (2017)" />
  <SubCaption text="GDP Growth (In %)" textAlign="center"
       textDecoration="underline" />
</Options>
```



See the CodePen 알메이트 차트 - 제목과 부제목

제목과 부제목 설정에 대한 자세한 내용은 <Caption> 노드와 <SubCaption> 노드의 아래 주요 속성에 관한 설명을 참조하십시요.

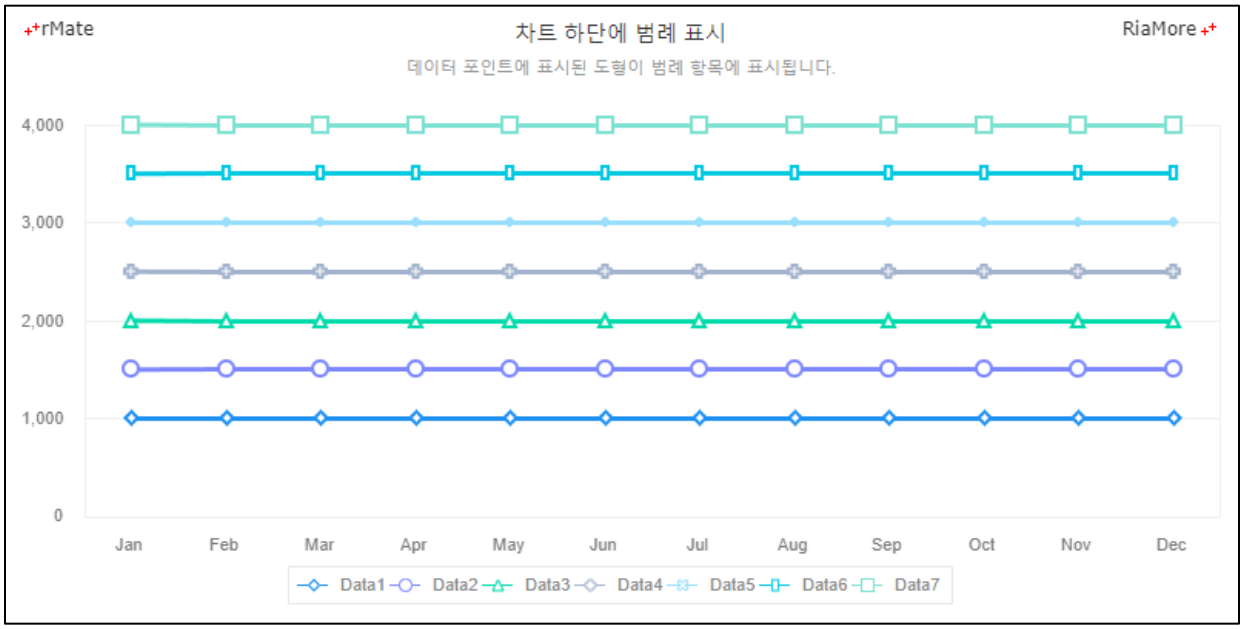
속성명	유효값 (*: 기본값)	설명
borderColor	#16 진수 컬러 코드 표기 기본값: #000000	(부)제목 테두리 선의 색을 지정합니다.

borderStyle	none(*), solid	(부)제목 테두리 선의 스타일을 설정합니다.
color	#16 진수 컬러 코드 표기 기본값: #000000	(부)제목 텍스트의 색을 지정합니다.
fontFamily	폰트명	(부)제목 텍스트에 적용될 폰트명을 지정합니다.
fontSize	숫자	(부)제목 텍스트에 적용될 폰트의 크기를 지정합니다.
fontWeight	normal(*), bold	(부)제목 텍스트를 볼드체로 표현할지 여부를 설정합니다.
text	텍스트	(부)제목을 설정합니다.
textAlign	left, center(*), right	(부)제목 텍스트의 정렬 방식을 설정합니다.
textDecoration	none(*), underline	(부)제목 텍스트에 밑줄을 표시할지 여부를 설정합니다.

3.4 범례

차트에 표현되는 데이터 시리즈가 여러 개일 경우 범례를 사용하여 차트의 이해도를 높일 수 있습니다. 알메이트 차트에서 범례는 <Options> 노드의 자식으로 <Legend> 노드를 정의하여 생성할 수 있습니다. 다음은 범례를 차트의 하단(기본값)에 표시하는 예제입니다.

```
<Options>
  <Legend useVisibleCheck="true" horizontalGap="0" position="bottom"
        horizontalScrollPolicy="off"/>
</Options>
...
<Line2DSeries yField="Data1" radius="6" displayName="Data1"
        itemRenderer="DiamondItemRenderer" fill="#ffffff">
...
```



See the CodePen 알메이트 차트 - 차트 하단에 범례 표시

다음은 <Legend> 노드의 주요 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명

borderColor	#16 진수 컬러 코드 표기 기본값: #000000	범례 테두리 선의 색을 지정합니다.
borderStyle	none(*), solid	범례 테두리 선의 스타일을 설정합니다.
defaultMouseOverAction	true, false(*)	사용자가 마우스를 특정 범례 항목에 올려두었을 때, 해당 데이터 시리즈 이외의 다른 데이터 시리즈는 흐리게 표시할지 여부를 설정합니다.
direction	horizontal(*), vertical	범례 항목의 표시 방향을 설정합니다.
hAlign	left, center(*), right	범례 항목의 수평 정렬 방식을 설정합니다. Position 속성값이 “top” 혹은 “bottom” 일 경우에만 유효합니다.
horizontalScrollPolicy	on, off(*)	범례에 수평 스크롤바를 표시할지 여부를 설정합니다.
itemClickJsFunction	자바스크립트 함수명	사용자가 범례 항목을 클릭할 경우 실행될 자바스크립트 함수명을 지정합니다.
position	left, right, top, bottom(*)	차트에서 범례의 위치를 지정합니다.
useVisibleCheck	true, false(*)	사용자가 범례 항목을 클릭할 경우 해당 항목의 데이터 시리즈에 대한 보이기 / 감추기를 설정합니다.
vAlign	top, middle(*), bottom	범례 항목의 수직 정렬 방식을 설정합니다. Position 속성값이 “left” 혹은 “right” 일 경우에만 유효합니다.
verticalScrollPolicy	on, off(*)	범례에 수직 스크롤바를 표시할지 여부를 설정합니다.
width	숫자	범례의 가로 크기를 지정합니다.

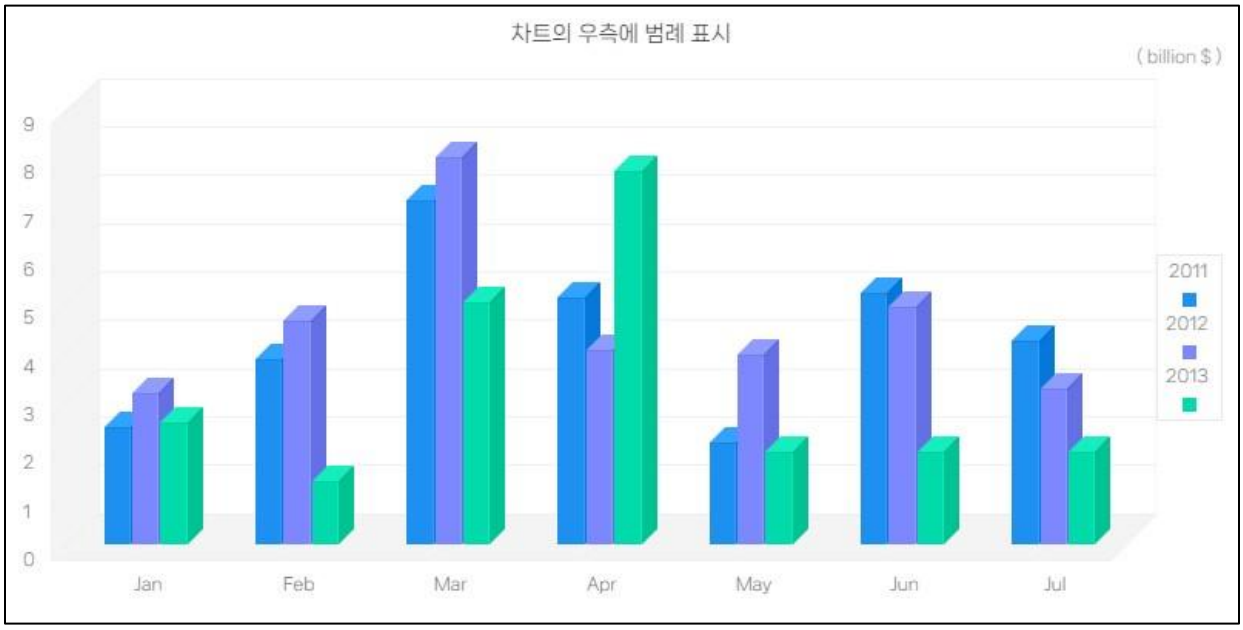
주의

범례의 항목으로 표시되는 텍스트는 데이터 시리즈의(예, <Line2DSeries> 노드) displayName 속성에 지정된 값입니다. 따라서 반드시 데이터 시리즈의 displayName 속성을 지정하여야 범례에서 항목명이 표시됩니다. 위 예제의 첫 번째 데이터 시리즈(yField="Data1")에서 displayName="Data1" 과 같이 설정되었습니다.

범례의 위치

범례의 기본 위치는 차트의 하단입니다. 따라서 범례의 position 속성값을 설정하지 않으면 자동으로 차트의 하단에 범례가 표시됩니다. 따라서 범례의 position 속성값을 설정하지 않으면 자동으로 차트의 하단에 범례가 표시됩니다. 다음은 범례의 위치를 차트의 우측에(position="right") 위치하게 하는 예제입니다. 이 예제에서는 항목들이 수직으로(direction="vertical") 표시되고 항목의 텍스트가 항목의 마커 위에(labelPlacement="top") 표시되도록 설정되었습니다.

```
<Options>
  <Legend position="right" direction="vertical" labelPlacement="top"
       useVisibleCheck="true"/>
</Options>
```



See the CodePen 알메이트 차트 - 차트 우측에 범례 표시

데이터 시리즈 보이기 / 숨기기

범례의 항목을 클릭했을 때 해당 항목의 데이터 시리즈에 대한 보이기 / 숨기기를 할 수 있습니다. 이 기능을 적용하기 위해서는 <Legend> 노드의 useVisibleCheck 속성을 “true” 로 설정해야 합니다. 다음은 차트가 처음 생성되는 시점에서는 두 개의 데이터 시리즈가 감추어진 상태이고, 이후 사용자가 범례 항목을 클릭하여 보이기를 할 수 있도록 설정된 차트의 예제입니다. 차트가 처음 생성되는 시점에 특정 데이터 시리즈에 대한 숨기기를 하기 위해서는 데이터 시리즈의 visible 속성을 “false” 로 설정해야 합니다.

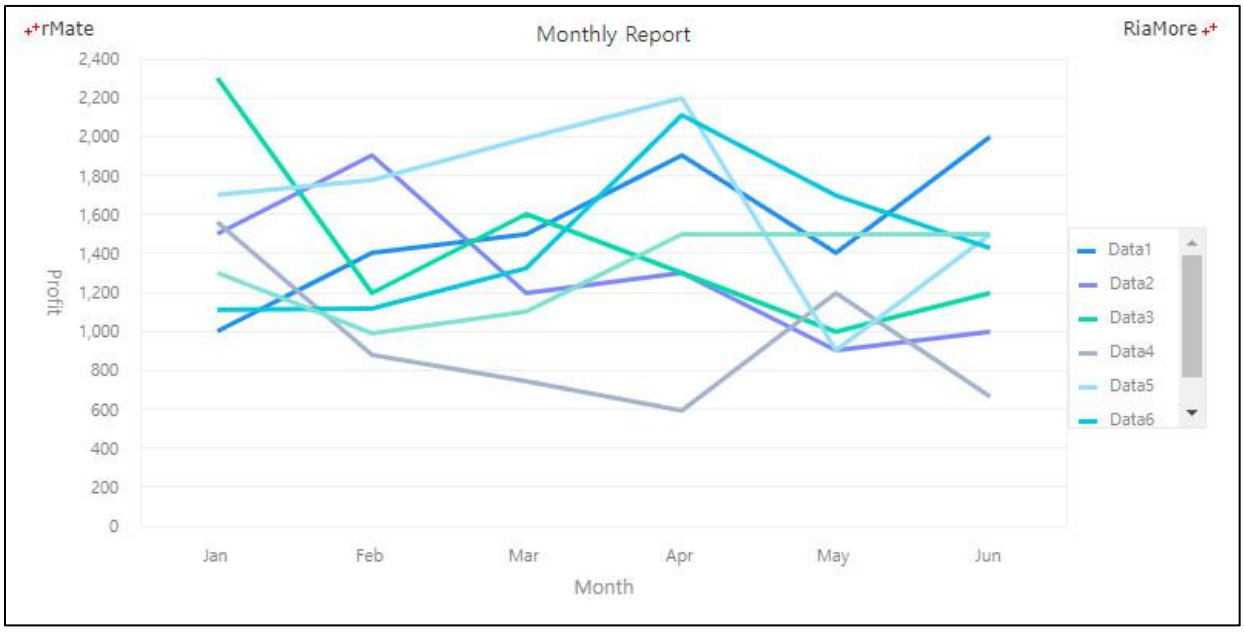
```
<Options>
  <Legend useVisibleCheck="true" markerWidth="12" markerHeight="12"/>
</Options>
...
<series>
  <Column2DSeries labelPosition="none" yField="goods" visible="false"
       displayName="goods"/>
  <Column2DSeries labelPosition="none" yField="income" visible="false"
       displayName="income"/>
  <Column2DSeries labelPosition="none" yField="service" displayName="service"/>
</series>
```

See the CodePen 알메이트 차트 - 범례 항목 보이기 / 숨기기

범례 스크롤바

범례의 항목이 많아서 주어진 공간에 모두 표시하지 못할 경우에는 스크롤바를 이용할 수 있습니다. 범례에 스크롤바를 표시하기 위해서는 해당 속성값을(verticalScrollPolicy, horizontalScrollPolicy) “on” 으로 지정해야 합니다. 다음은 수직 스크롤바를 범례에 표시한 예제입니다.

```
<Options>
  <Legend position="right" useVisibleCheck="true" direction="horizontal"
       labelPlacement="right" markerWidth="12" markerHeight="12" width="90"
       height="130" verticalScrollPolicy="on"/>
</Options>
```



See the CodePen 알메이트 차트 - 범례 스크롤바

범례 항목 툴팁

사용자가 마우스를 범례 항목에 올려두었을 때 해당 항목의 툴팁을 표시할 수 있습니다. 이 기능은 <Legend> 노드의 titleJsFunction 속성에 실행될 자바스크립트 함수명을 지정하고 해당 함수를 구현함으로써 가능합니다. 다음은 이에 대한 예제입니다.

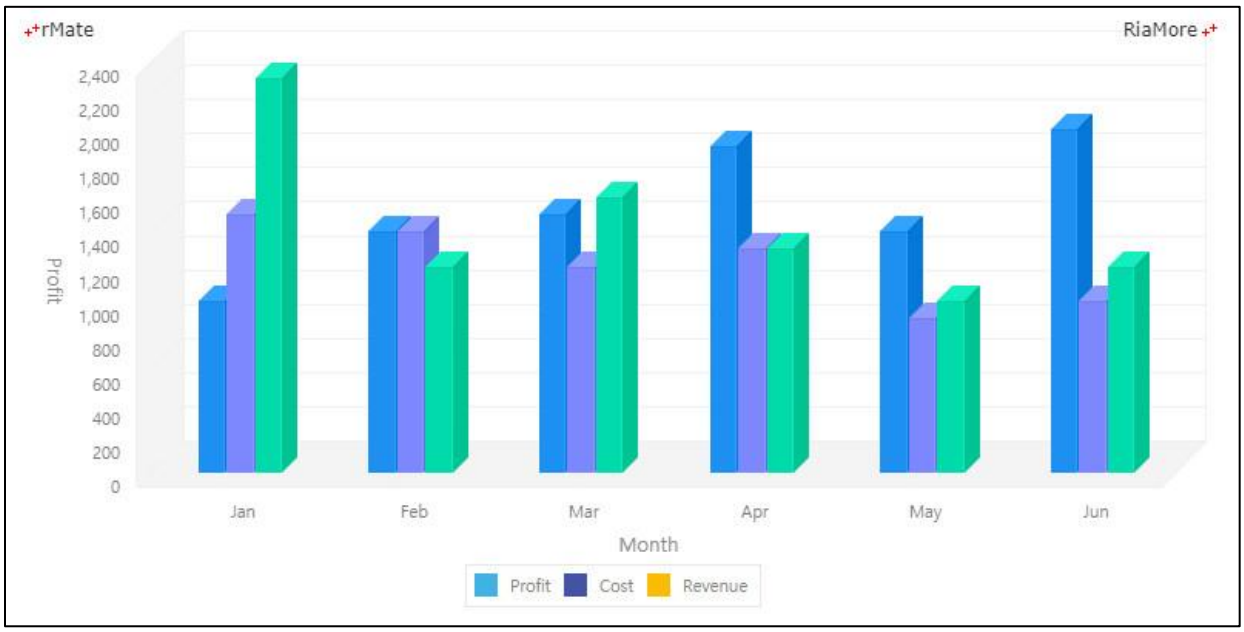
```
<Options>
  <Legend titleJsFunction="legendTitleFunc" .../>
</Options>
...
function legendTitleFunc(index, value) {
  return value + "입니다.";
}
```

See the CodePen 알메이트 차트 - 범례 항목 툴팁 표시

사용자 정의 범례

<Legend> 노드를 이용하지 않고 <SubLegend> 노드를 <Box> 노드의 자식으로 정의하여 범례를 생성할 수 있습니다. 이 방법은 <Legend> 노드를 이용하여 범례를 만드는 것보다 차트 개발자에게 더 유연한 개발 환경을 제공합니다. 다음은 <SubLegend> 노드와 <CheckableLegendItem> 노드를 이용하며 범례를 만드는 예제입니다.

```
<Box horizontalAlign="center" width="100%">
  <SubLegend useVisibleCheck="true" direction="horizontal" markerHeight="15"
       markerWidth="15">
    <CheckableLegendItem targetSeries="{series1}" fill="#40b2e6" label="Profit"/>
    <CheckableLegendItem targetSeries="{series2}" fill="#4453a8" label="Cost"/>
    <CheckableLegendItem targetSeries="{series3}" fill="#fabc05" label="Revenue"/>
  </SubLegend>
</Box>
```



See the CodePen 알메이트 차트 - 사용자 정의 범례 (Sub-legend)

주의

위 예제에서는 <Legend> 노드의 useVisibleCheck 속성을 설정한 (useVisibleCheck="true") 것과 동일한 기능을 제공하는 <CheckableLegendItem> 노드를 이용하였습니다. 범례 항목 클릭시 데이터 시리즈에 대한 보이기 / 숨기기 기능을 사용자 정의 범례(Sub-legend)에 적용하기 위해서는 <LegendItem> 노드가 아닌 <CheckableLegendItem> 노드를 이용해야 합니다.

범례 항목 클릭시 강조

범례 항목을 클랙했을 때 기본적인 차트의 동작은 클릭한 항목의 시리즈를 보이기/감추기 하는 것입니다. itemClickEmphasizeType (기본값 : ”appearance”) 속성값을 ”focus” 로 설정하면 클릭한 항목에 대한 시리즈를 보이기/감추기 하지않고 해당 시리즈를 강조하여 표현하도록 할 수 있습니다. 다음은 <Legend> 노드의 itemClickEmphasizeType 속성값을 ”focus” 설정한 스택 컬럼 차트의 예제입니다.

```
<Options>
...
<Legend useVisibleCheck="true" markerWidth="12" markerHeight="12"
        itemClickEmphasizeType="focus"/>
</Options>
```

See the CodePen 알메이트 차트 - 범례 항목 클릭시 시리즈 강조

3.5 툴팁

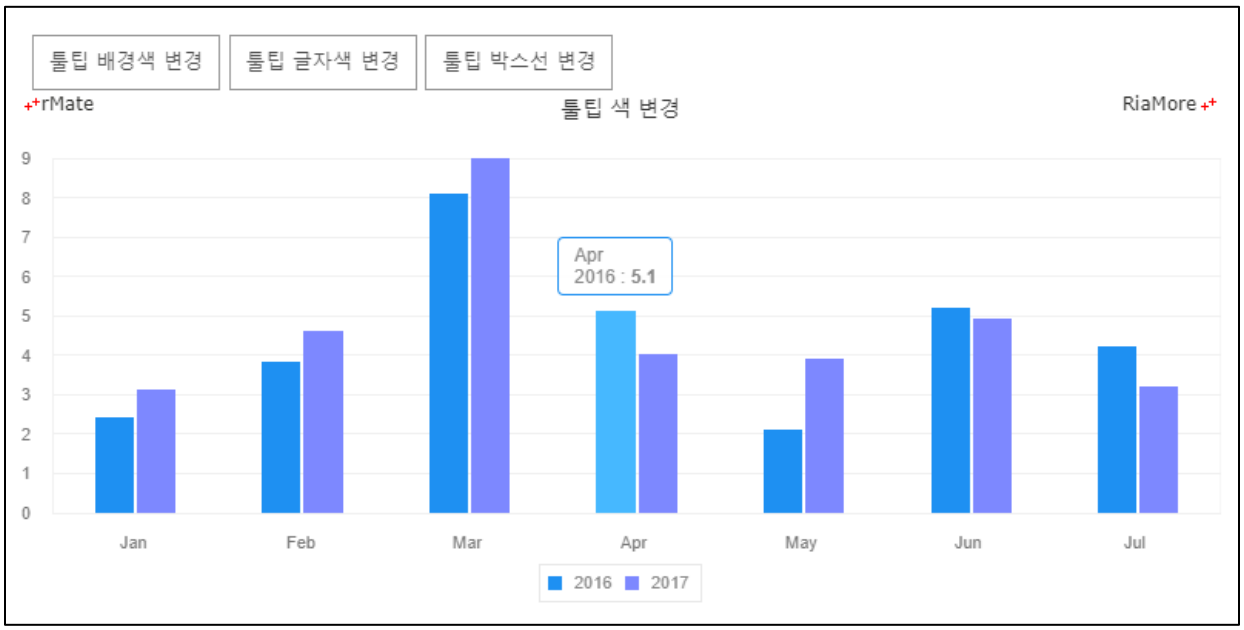
모든 종류의 차트에서 사용자가 차트의 데이터 포인트에 마우스를 올려놓았을 때 툴팁을 표시하도록 할 수 있습니다. 툴팁을 표시하기 위해서는 차트 노드 (예, <Column2DChart>)의 showDataTips 속성을 “true” 로 설정해야 합니다. 다음은 툴팁과 관련된 차트 노드의 주요 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명
dataTipBackgroundColorOnSeries	true, false(*)	툴팁의 배경색을 데이터 시리즈의 색과 동일한 색으로 표시할지 여부를 설정합니다.
dataTipBorderColor	#16 진수 컬러 코드 표기 기본값: #000000	툴팁 테두리 선의 색을 지정합니다.
dataTipBorderColorOnSeries	true, false(*)	툴팁의 테두리 선의 색을 데이터 시리즈의 색과 동일한 색으로 표시할지 여부를 설정합니다.
dataTipBorderStyle	none(*), solid	툴팁 테두리 선의 스타일을 설정합니다.
dataTipDisplayMode	none(*), mouse, axis	none: 데이터 포인트에 툴팁이 표시됩니다. mouse: 툴팁이 표시된 후 마우스 포인터를 이동하면, 이동한마우스 포인터를 따라서 툴팁이 표시됩니다. axis: 같은 X 축의 좌표에 툴팁이 표시됩니다. (바 차트의 경우에는 같은 Y 축 좌표)

dataTipFontColorOnSeries	true, false(*)	툴팁의 텍스트 색을 데이터 시리즈의 색과 동일한 색으로 표시할지 여부를 설정합니다.
dataTipFormatter	CurrencyFormatter, DateFormatter, NumberFormatter default: null	툴팁에 표시되는 숫자 텍스트의 포맷터를 설정합니다.
dataTipJsFunction	자바스크립트 함수명	사용자 정의 툴팁을 표시하기 위해서 실행될 자바스크립트 함수명을 지정합니다.
dataTipMode	single(*), multiple	single: 마우스 포인터에 가장 가까운 한 개의 데이터 포인트에 툴팁이 표시됩니다. multiple: 마우스 포인터에 가까운 여러 개의 데이터 포인트에 툴팁이 표시됩니다.
showDataTips	true, false(*)	사용자가 차트의 데이터 포인트에 마우스를 올려놓았을 때 툴팁을 표시할지 여부를 지정합니다.

다음은 컬럼 차트에서 툴팁을 표시하는 예제입니다. 처음 차트를 실행하고 마우스를 데이터 포인트에 올리면 기본색(배경, 테두리 선, 텍스트 색)이 표시됩니다. 버튼을 클릭하면 해당 속성이 설정되고 툴팁의 색이 변경되는 것을 확인할 수 있습니다.

```
<Column2DChart showDataTips="true">
```

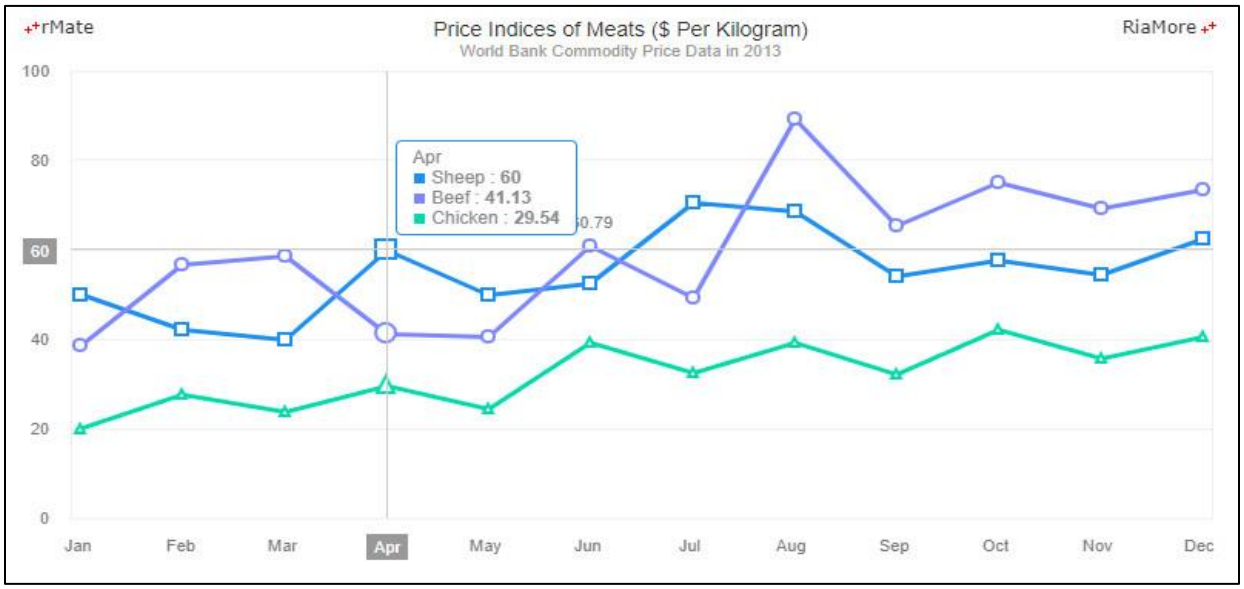


See the CodePen 알메이트 차트 - 툴팁의 배경, 외곽선, 텍스트 설정

축의 좌표에 따라서 툴팁 표시

다음은 dataTipDisplayMode 속성을 “axis” 로 설정한 예제입니다. 세 개의 데이터 시리즈의 툴팁이 X 축과 동일한 좌표에서 같이 표시되는 것을 확인하실 수 있습니다.

```
<Column2DChart showDataTips="true" dataTipDisplayMode="axis">
```

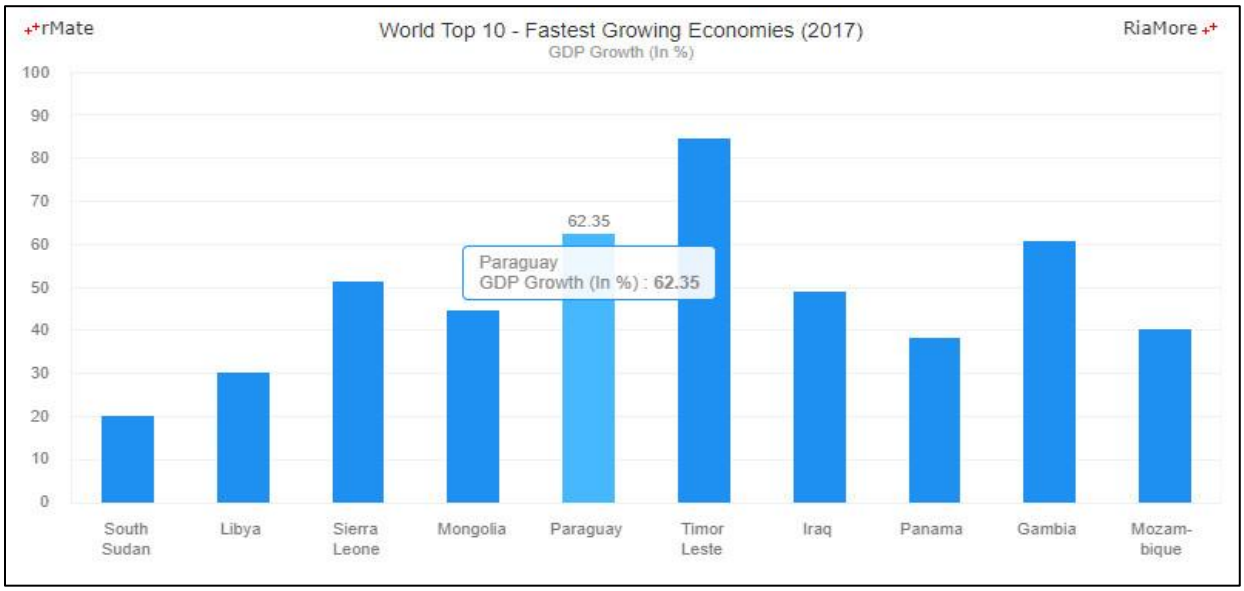


See the CodePen 알메이트 차트 - 축의 좌표에 따라서 툴팁 표시

마우스 포인터에 따라서 툴팁 표시

다음은 dataTipDisplayMode 속성을 “mouse” 로 설정한 예제입니다. 컬럼을 따라서 마우스를 이동하면 이동한 경로를 따라서 툴팁이 표시되는 것을 확인하실 수 있습니다.

```
<Column2DChart showDataTips="true" dataTipDisplayMode="mouse">
```

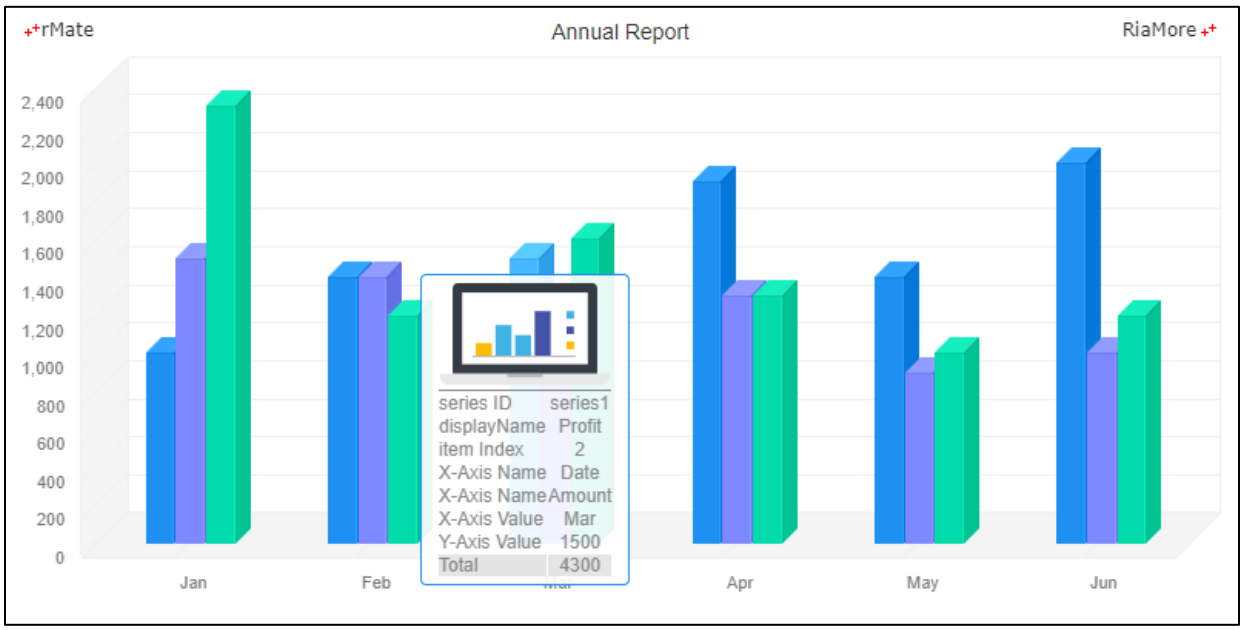


See the CodePen 알메이트 차트 - 마우스 포인터에 따라서 툴팁 표시

사용자 정의 툴팁

다음은 기본 형식으로 툴팁을 표시하지 않고, 자바스크립트 함수를 이용하여 사용자가 원하는 형태로 툴팁을 표시하는 예제입니다. 이 예제에서는 툴팁의 포맷을 HTML 태그를 이용하였고 툴팁에 이미지가 삽입되었습니다.

```
<Column3DChart showDataTips="true" dataTipJsFunction="dataTipFunc">

function dataTipFunc(seriesId, seriesName, index, xName, yName, data, values) {
  return "<table cellpadding='0' cellspacing='1'>"
    +"<tr>"
      +"<td align='center' colspan='2' style='border-bottom:solid 1px
       #8b8b8b;'><img src='../rMateChartH5/Assets/Images/monitor.png'></td>"
    ...
    +"</tr></table>";
}
```

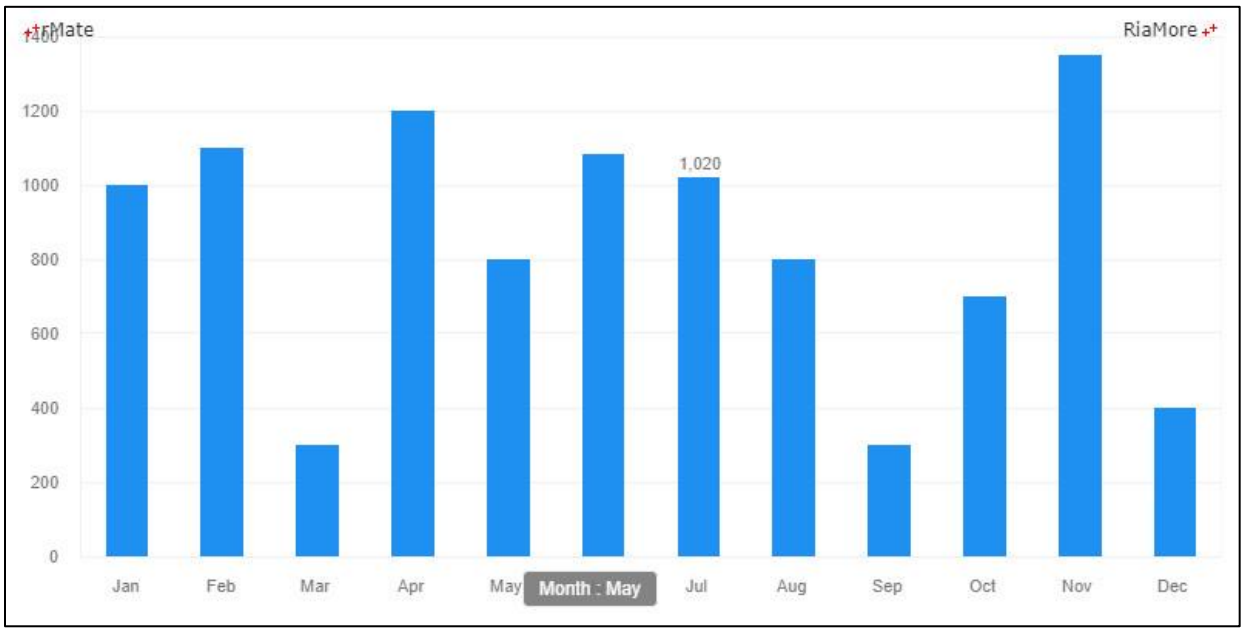


See the CodePen 알메이트 차트 - 사용자 정의 툴팁 표시

사용자 정의 축 툴팁 표시

사용자가 축 레이블에 마우스 오버했을 때 툴팁을 표시할 수 있고, 자바스크립트 함수를 이용하여 사용자가 원하는 형태로 툴팁을 표시하는 것도 가능합니다. 다음은 X 축에 사용자 정의 툴팁을 표시하는 예제입니다. showHorizontalAxisDataTip 속성이 “true” 로 설정되었고 axisDataTipJsFunction 속성에 실행될 자바스크립트 함수명이 지정되었습니다.

```
<Column2DChart showDataTips="true" showHorizontalAxisDataTip="true"
        axisDataTipJsFunction="axisDataTipFunc">

function axisDataTipFunc(id, text) {
  return "Month : " + text;
}
```



See the CodePen 알메이트 차트 - 사용자 정의 축 툴팁 표시

3.6 십자선과 확대

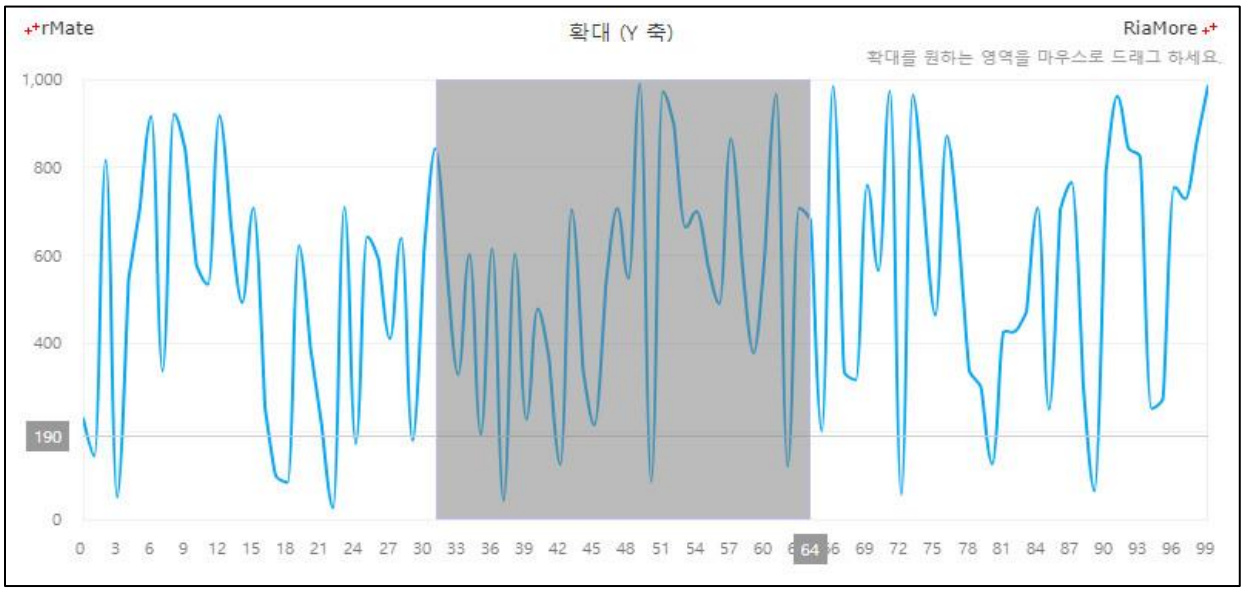
차트의 영역에 마우스 포인터를 올려놓았을 때 십자선을 표시하고, 마우스를 드래그하여 줌 기능을 실행할 수 있습니다. 십자선의 표시는 차트 노 (예, <Area2DChart>)의 <annotationElements> 속성에 <CrossRangeZoomer> 노드를 정의함으로써 가능합니다.
다음은 <CrossRangeZoomer> 노드의 주요 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명
enableCrossHair	true(*), false	마우스 포인터에 십자선을 표시할지 여부를 설정합니다.
enableZooming	true(*), false	줌 기능을 사용할지 여부를 설정합니다.
horizontalLabelFormatter	CurrencyFormatter, DateFormatter, NumberFormatter 기본값: null	십자선의 수평선 레이블의 포맷터를 설정합니다.
horizontalLabelPlacement	left(*), right	십자선의 수평선 레이블을 차트의 어느 위치에 표시할지를 지정합니다.
horizontalStroke	<Stroke>	십자선의 수평선의 색의 스타일을 지정합니다.
rangeUpdateJsFunction	자바스크립트 함수명	한 화면에 여러 개의 차트가 표시될 때, 차트들의 십자선을 동기화하기 위해서 마우스 이벤트가 발생하면 실행될 자바스크립트 함수명을 지정합니다.
showValueLabels	true(*), false	현재 마우스 포인터에 대한 좌표값을 표시할지 여부를 설정합니다.
syncCrossRangeZoomer	식별자 (id)	<DualChart> 에서 <mainChart> 와 <subChart> 의 십자선을 동기화해서 표시하기 위해서 동기화할 인스턴스의 식별자를 지정합니다.

verticalLabelFormatter	CurrencyFormatter, DateFormatter, NumberFormatter 기본값: null	십자선의 수직선 레이블의 포맷터를 설정합니다.
verticalLabelPlacement	bottom(*), top	십자선의 수직선 레이블을 차트의 어느 위치에 표시할지를 지정합니다.
verticalStroke	<Stroke>	십자선의 수직선의 색의 스타일을 지정합니다.
zoomRangeFill	<SolidColor>	마우스를 드래그한 영역의 색의 스타일을 지정합니다.
zoomRangeStroke	<Stroke>	마우스를 드래그한 영역의 테두리 선의 색의 스타일을 지정합니다.
zoomType	zoomType	줌 기능에서 어떤 축에 따라서 줌을 실행할지를 지정합니다.

다음은 영역 차트에 수평축으로 줌 기능이 적용되는 십자선을 표시하는 예제입니다. 마우스 포인터를 차트위에 올리고 드래그하면 드래그된 영역으로 줌됩니다.

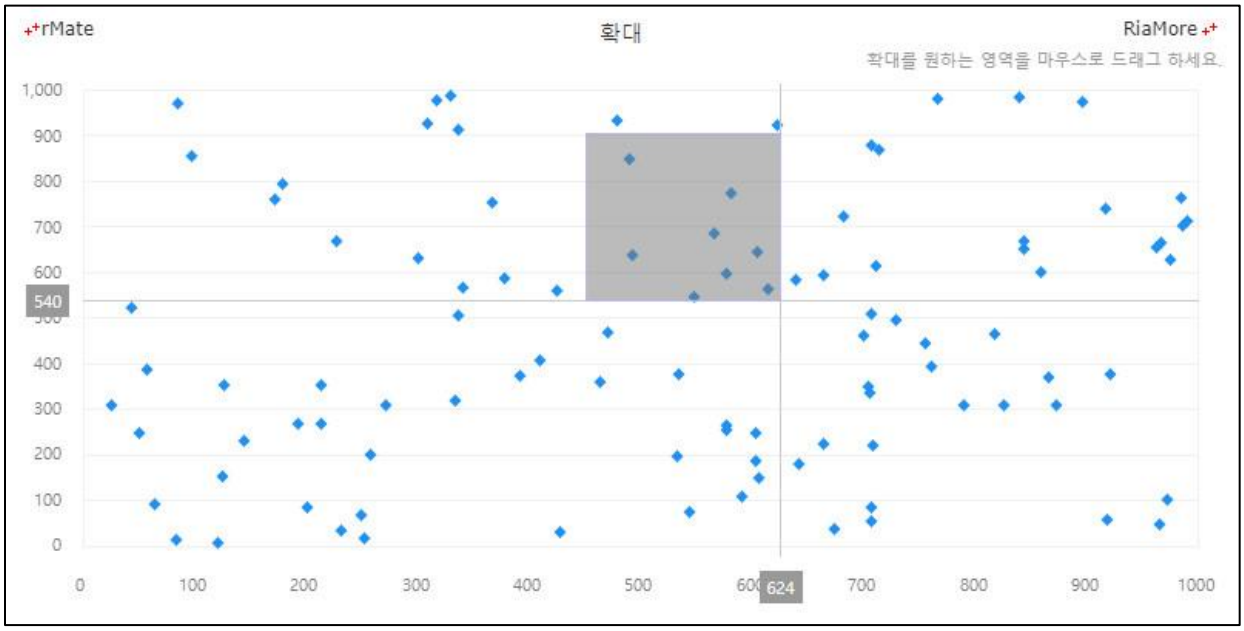
```
<annotationElements>
  <CrossRangeZoomer enableZooming="true" enableCrossHair="true"
        zoomType="horizontal"/>
</annotationElements>
```



See the CodePen 알메이트 차트 - 십자선과 수평 확대 기능

다음은 수직축과 수평축으로 줌 기능이 구현된 플롯 차트의 예제입니다.

```
<annotationElements>
  <CrossRangeZoomer enableZooming="true" enableCrossHair="true" zoomType="both"/>
</annotationElements>
```



See the CodePen 알메이트 차트 - 십자선과 수평 수직 확대 기능

다음은 캔들스틱 차트에서 메인 차트와 서브 차트의 십자선을 하나의 십자선처럼 표시하는 예제입니다. 메인 차트와 서브 차트의 십자선을 하나처럼 표시하기 위해서 syncCrossRangeZoomer 속성에 상대 차트의 <CrossRangeZoomer> 노드의 식별자를 지정하였습니다.

```
<rMateChart backgroundColor="#FFFFFF" borderStyle="none">
  ...
  <mainChart>
    ...
    <annotationElements>
      <CrossRangeZoomer id="candleCRZ" enableZooming="false"
       syncCrossRangeZoomer="{columnCRZ}" zoomType="both"
       horizontalLabelFormatter="{nft}"/>
    </annotationElements>
    ...
  <subChart>
    <annotationElements>
      <CrossRangeZoomer id="columnCRZ" enableZooming="false"
       syncCrossRangeZoomer="{candleCRZ}" zoomType="both"
       verticalLabelPlacement="top" horizontalLabelFormatter="{nft}"/>
    </annotationElements>
  ...
</rMateChart>
```

See the CodePen 알메이트 차트 - 메인 차트와 서브 차트에 십자선 표시

다중 차트에서 십자선 동기화

한 화면에 여러 개의 차트가 표현되어 있을 때 차트들의 십자선을 서로 동기화하여 표시할 수 있습니다. 이 기능의 실행은 <CrossRangeZoomer> 노드의 rangeUpdateJsFunction 속성에 알메이트 차트가 제공하는 API 함수, updateCrossRange() 를 실행하는 자바스크립트 함수명을 지정함으로써 가능합니다. 다음은 라인 차트와 컬럼 차트의 십자선을 서로 동기화하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<annotationElements>
  <CrossRangeZoomer rangeUpdateJsFunction="rangeFunc"/>
</annotationElements>

function rangeFunc(type, x, y) {
  for(var o in rMateChartH5.instances)
    rMateChartH5.instances[o].updateCrossRange(type, x, y);
}
```

See the CodePen 알메이트 차트 - 다중 차트에서 십자선 동기화

3.7 배경과 격자선

차트의 배경에 격자선을 표시하거나 이미지를 이용하여 보기좋게 표현할 수 있습니다. 이러한 작업은 <backgroundElements> 속성에 <GridLines> 과 <Image> 노드를 설정함으로써 가능합니다.

격자선 표시

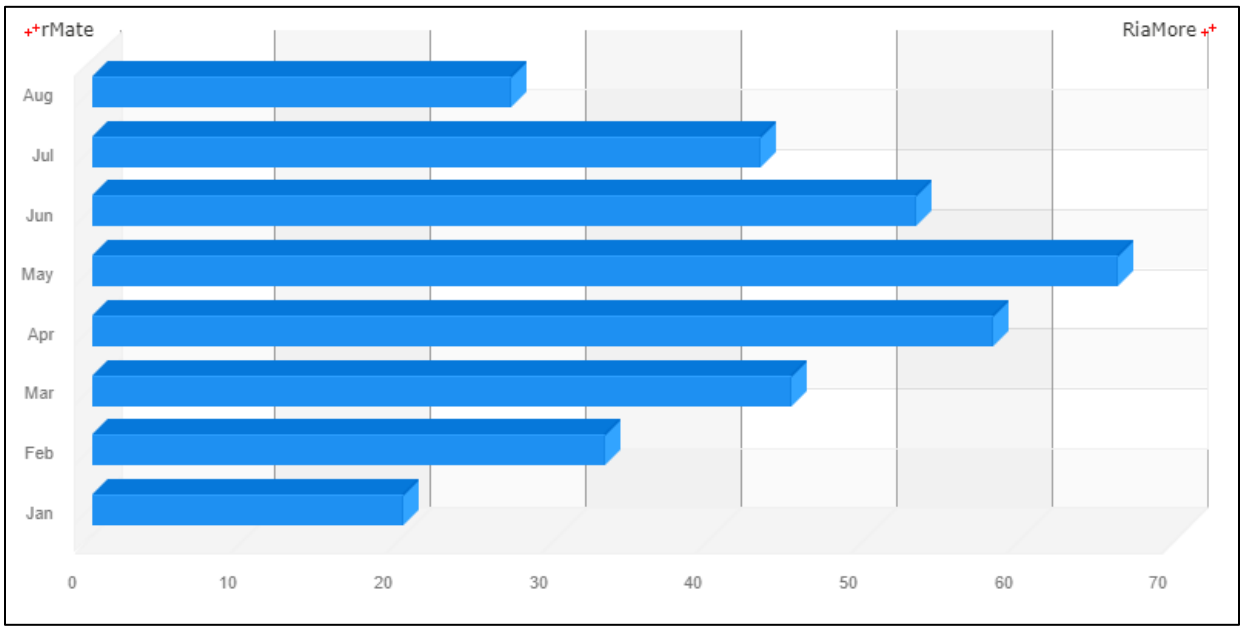
차트의 배경에 격자선을 표시하기 위해서는 차트 노드의 <backgroundElements> 속성에 <GridLines> 노드를 정의해야 합니다. 다음은 <GridLines> 노드의 주요 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명
direction	both, horizontal(*), vertical	표시되는 배경 격자선의 유형을 지정합니다.
horizontalAlternateFill	<SolidColor>	이웃한 두 수평 격자선 사이의 공간에 색을 번갈아서 칠할 때, 두번째 사용될 색의 스타일을 지정합니다.
horizontalCenterStroke	<Stroke>	차트의 중앙에 표시되는 수평선의 색의 스타일을 지정합니다.
horizontalChangeCount	숫자 기본값: 1	이웃한 두 수평 격자선 사이의 공간에 색을 번갈아서 칠할 때, 몇 번마다 색을 변경할지를 지정합니다.
horizontalFill	<SolidColor>	이웃한 두 수평 격자선 사이의 공간에 칠할 색의 스타일을 지정합니다.
horizontalShowCenterLine	true, false(*)	차트의 중앙에 수평선을 표시할지 여부를 설정합니다.
horizontalStroke	<Stroke>	수평 격자선의 색의 스타일을 지정합니다.
verticalAlternateFill	<SolidColor>	이웃한 두 수직 격자선 사이의 공간에 색을 번갈아서 칠할 때, 두번째 사용될 색의 스타일을 지정합니다.

verticalCenterStroke	<Stroke>	차트의 중앙에 표시되는 수직선의 색의 스타일을 지정합니다.
verticalChangeCount	숫자 기본값: 1	이웃한 두 수직 격자선 사이의 공간에 색을 번갈아서 칠할 때, 몇 번마다 색을 변경할지를 지정합니다.
verticalFill	<SolidColor>	이웃한 두 수직 격자선 사이의 공간에 칠할 색의 스타일을 지정합니다.
verticalShowCenterLine	true, false(*)	차트의 중앙에 수직선을 표시할지 여부를 설정합니다.
verticalStroke	<Stroke>	수직 격자선의 색의 스타일을 지정합니다.

₩다음은 3D 바 차트의 배경에 수직, 수평 격자선을 표시한 예제입니다. 이 예제에서는 이웃한 격자선 사이의 공간에 칠하는 색이 설정되었습니다.

```
<backgroundElements>
  <GridLines direction="both" verticalChangeCount="1" horizontalChangeCount="1"
       horizontalTickAligned="false">
    <horizontalStroke>
      <Stroke color="#999999" alpha="0.5" weight="1"/>
    </horizontalStroke>
    <horizontalFill>
      <SolidColor color="#EEEEEE" alpha="0.5"/>
    </horizontalFill>
    <horizontalAlternateFill>
      <SolidColor color="#FFFFFF" alpha="0.5"/>
    </horizontalAlternateFill>
    <verticalStroke>
      <Stroke color="#999999" alpha="0.5" weight="1"/>
    </verticalStroke>
    <verticalFill>
      <SolidColor color="#FFFFFF" alpha="0.5"/>
    </verticalFill>
    <verticalAlternateFill>
      <SolidColor color="#EEEEEE" alpha="0.5"/>
    </verticalAlternateFill>
    <horizontalOriginStroke>
      <Stroke color="#999999" alpha="0" weight="1"/>
    </horizontalOriginStroke>
  </GridLines>
</backgroundElements>
```



See the CodePen 알메이트 차트 - 차트에 격자선 표시

주의

<GridLines>> 노드를 레이아웃에 정의하지 않더라도 차트의 유형에 따라서 자동으로 기본 격자선이 표시됩니다. (예, 바 차트: 수직 격자선, 컬럼 차트: 수평 격자선). 차트에 격자선이 표시되는 것을 원하지 않을 경우에는 다음과 같이 <GridLines> 노드의 direction 속성을 “none” 으로 설정해야 합니다.

```
<backgroundElements>
  <GridLines direction="none">
</backgroundElements>
```

배경에 이미지 표시

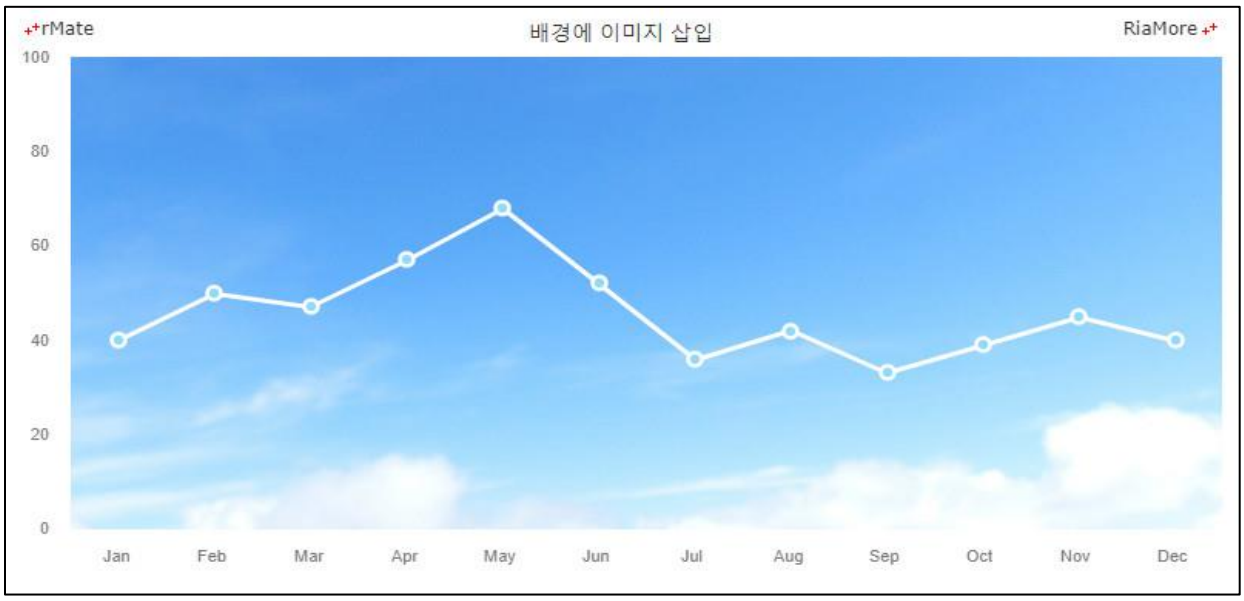
차트의 배경에 이미지를 표시하기 위해서는 차트 노드의 <backgroundElements> 속성에 <Image> 노드를 정의해야 합니다. 다음은 <Image> 노드의 주요 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명
alpha	0 과 1 사이의 숫자 기본값: 1	이미지의 투명도를 설정합니다.

bottom	숫자 기본값: NaN	차트의 하단 가장자리와 이미지가 표시될 위치 사이의 여백의 크기를 지정합니다.
height	숫자	이미지의 세로 크기를 지정합니다.
left	숫자 기본값: NaN	차트의 좌측 가장자리와 이미지가 표시될 위치 사이의 여백의 크기를 지정합니다.
maintainAspectRatio	true, false(*)	원본 이미지의 비율을 유지할지 여부를 설정합니다.
right	숫자 기본값: NaN	차트의 우측 가장자리와 이미지가 표시될 위치 사이의 여백의 크기를 지정합니다.
source	텍스트 (url)	이미지 파일이 존재하는 URL 을 지정합니다.
top	숫자 기본값: NaN	차트의 상단 가장자리와 이미지가 표시될 위치 사이의 여백의 크기를 지정합니다.
width	숫자	이미지의 가로 크기를 지정합니다.

다음은 라인 차트의 배경에 이미지를 표시하는 예제입니다.

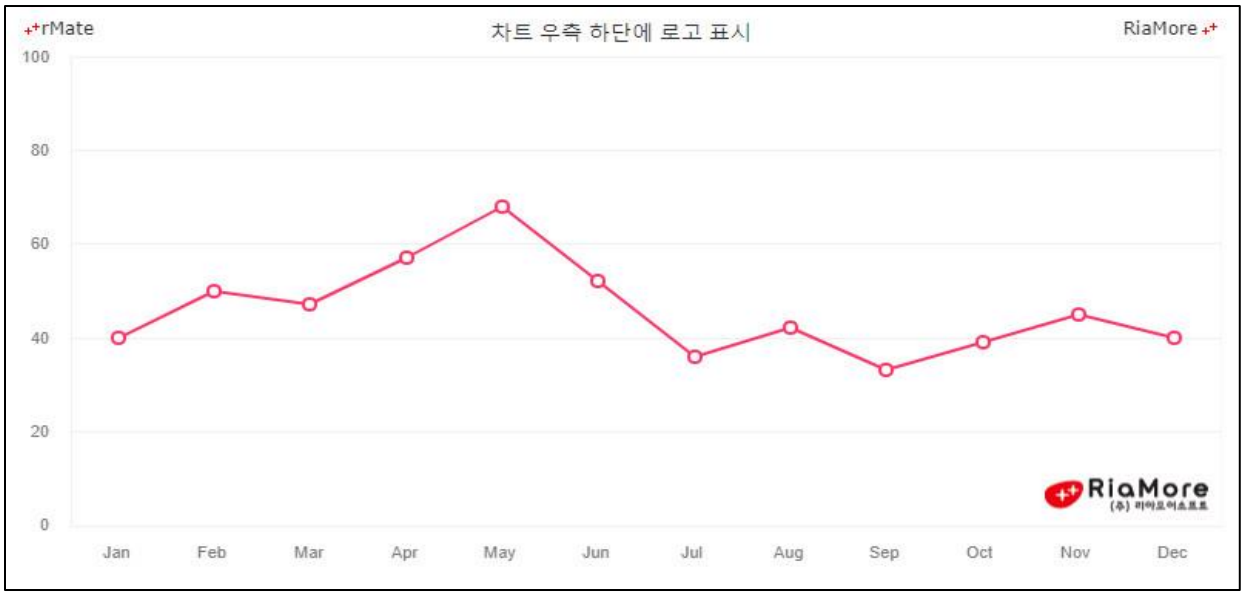
```
<backgroundElements>
  <Image source="../rMateChartH5/Assets/Images/chart_background.jpg"
       maintainAspectRatio="false" alpha="1"/>
</backgroundElements>
```



See the CodePen 알메이트 차트 - 배경에 이미지 표시

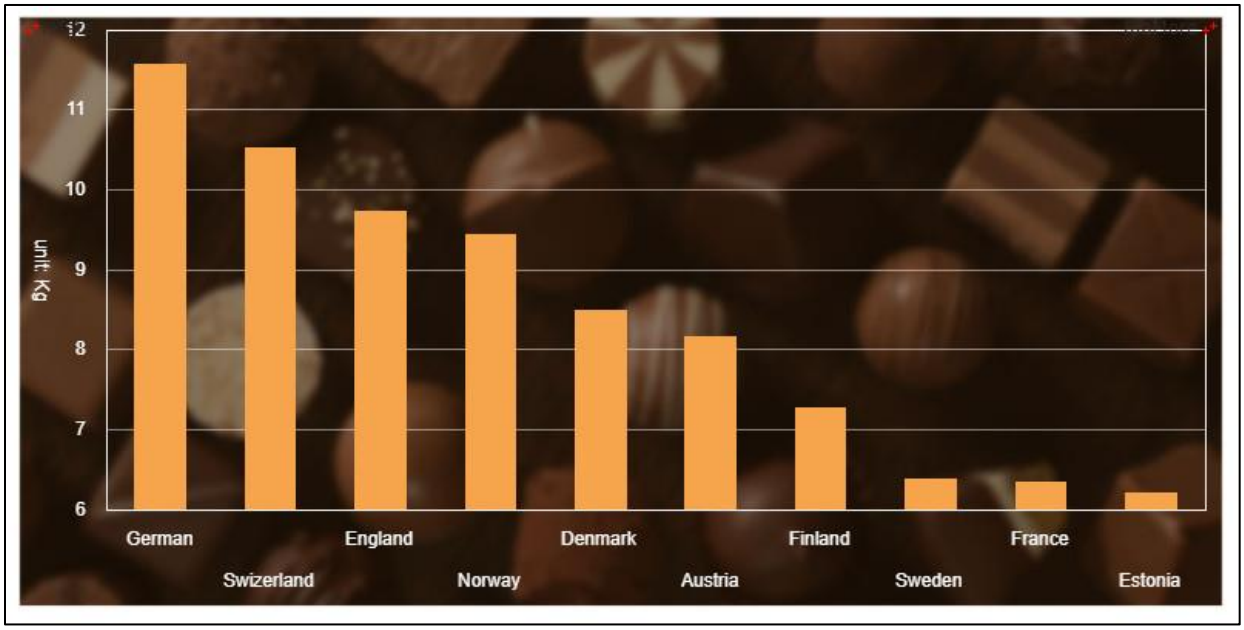
다음은 라인 차트의 배경 우측 하단에 로고 이미지를 표시하는 예제입니다.

```
<backgroundElements>
  <GridLines/>
  <CanvasElement>
    <Image right="10" bottom="10" source="http://www.riamore.net/image/logo2.png"/>
  </CanvasElement>
</backgroundElements>
```



See the CodePen 알메이트 차트 - 배경 우측 하단에 로고 이미지 표시

차트의 축을 포함한 전체 영역에 배경 이미지를 표시할 수도 있습니다. 이는 차트가 생성되는 <DIV> 요소에 다음과 같이 HTML background 스타일 속성을 설정함으로써 가능합니다.



See the CodePen 알메이트 차트 - DIV 배경 이미지

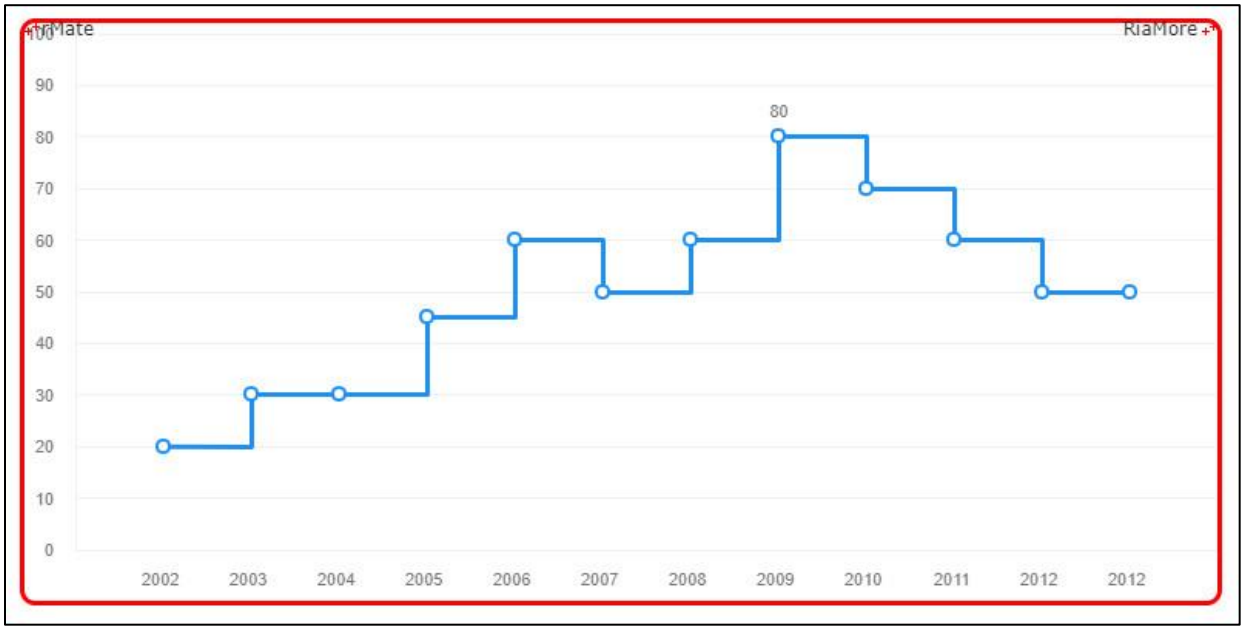
차트 경계선

<rMateChart> 노드의 경계선에 대한 속성값들을 설정해서 차트에 경계선을 표시하고 이에 대한 스타일링을 할 수 있습니다. 차트 경계선 설정에 관련된 속성은 다음과 같습니다.

속성명	속성명 (*: 기본값)	설명
borderStyle	none(*), solid, inset, outset	none: 경계선을표시하지 않습니다. solid: 직선경계선을 표시합니다. inset: 3D 눌려진(inset) 경계선을 표시합니다. outset: 3D 돌출된(outset) 경계선을 표시합니다.
borderColor	#16 진수 컬러 코드 표기	경계선의 색을 지정합니다.
borderRadius	0(*) 이상의 숫자	경계선 모서리의 둥근 정도를 지정합니다.
borderThickness	1(*) 이상의 숫자	경계선의 굵기를 지정합니다.

다음은 경계선 속성값들을 설정한 코드와 이를 적용해서 출력한 차트의 예입니다.

```
<rMateChart borderStyle="solid" borderColor="#FF0000" borderRadius="10"
        borderThickness="5">
...
</rMateChart>
```



See the CodePen 알메이트 차트 - 차트의 테두리 선

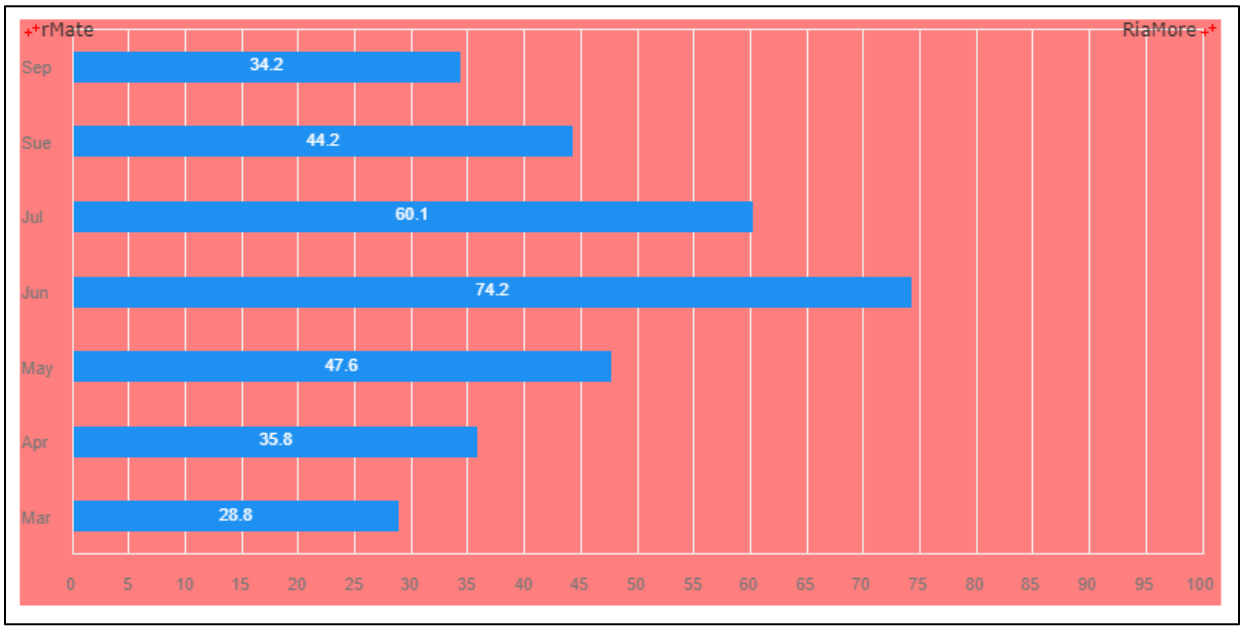
차트의 배경색 설정하기

차트의 배경에 원하는 색을 설정할 수 있습니다. 배경색 설정에 관련된 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
backgroundColor	#16 진수 컬러 코드 표기 기본값: #FFFFFF	배경색을 지정합니다.
backgroundAlpha	0 과 1(*) 사이의 숫자	배경색의 투명도를 지정합니다.

다음은 배경색 속성값들을 설정한 코드와 이를 적용해서 출력한 차트의 예입니다.

```
<rMateChart backgroundColor="#FF0000" backgroundAlpha="0.5">
  ...
</rMateChart>
```



See the CodePen 알메이트 차트 - 차트의 배경색 설정

3.8 선긋기

차트의 배경에 직선을 표시할 수 있습니다. 이러한 작업은 <backgroundElements> 속성에 <AxisMarker> 노드를 정의하고 <AxisMarker> 노드의 <lines> 속성에 <AxisLine> 노드를 설정함으로써 가능합니다. 다음은 <AxisLine> 노드의 주요 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명
dashLinePattern	숫자 기본값: 5	lineStyle 속성값이 “dashLine” 일 경우, 점선의 길이를 지정합니다.
endValue	숫자	선이 끝나는 지점의 값을 지정합니다. value 속성값이 지정되어 있으면 이 값은 무시됩니다.
horizontal	true(*), false	표시되는 선이 수평선인지 여부를 설정합니다.
label	텍스트	선에 표시될 레이블을 지정합니다.
lineStyle	dashLine, normal(*)	선의 유형을 지정합니다.
startValue	숫자	선이 시작되는 지점의 값을 지정합니다. value 속성값이 지정되어 있으면 이 값은 무시됩니다.
stroke	<Stroke>	선의 색의 스타일을 지정합니다.
value	숫자	선이 표시되는 지점의 값을 지정합니다. value 속성값이 지정되면 startValue, endValue 속성은 무시됩니다.

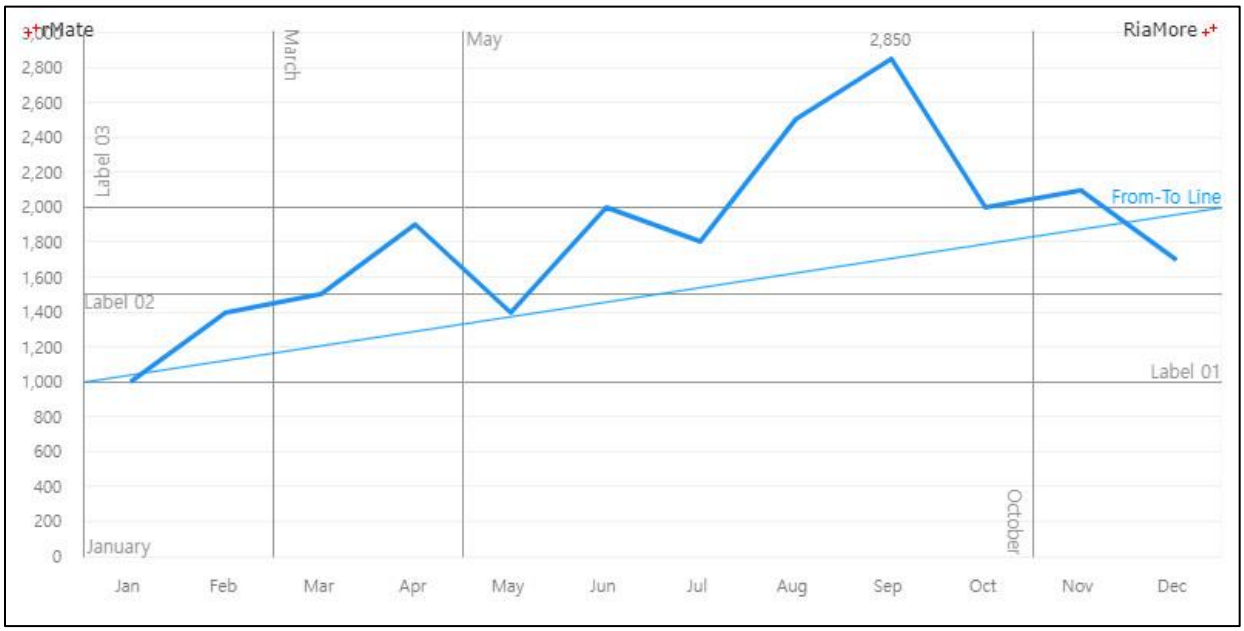
다음은 차트에 하한선과 상한선을 점선으로 표시하는 예제입니다.

```
<backgroundElements>
  <GridLines/>
  <AxisMarker>
    <lines>
      <AxisLine value="2500" lineStyle="dashLine" label="High">
        <stroke>
          <Stroke color="#FF7171" weight="2"/>
        </stroke>
      </AxisLine>
      <AxisLine value="500" lineStyle="dashLine" label="Low" labelUpDown="down">
        <stroke>
          <Stroke color="#6799FF" weight="2"/>
        </stroke>
      </AxisLine>
    </lines>
    <ranges>
      <AxisRange startValue="500" endValue="2500">
        <fill>
          <SolidColor color="#eeeeee" alpha="0.4"/>
        </fill>
      </AxisRange>
    </ranges>
  </AxisMarker>
</backgroundElements>
```

See the CodePen 알메이트 차트 - 차트에 선긋기

다음은 차트에 수직선, 수평선 그리고 대각선을 함께 표시한 예제입니다.

```
<backgroundElements>
  <GridLines/>
  <AxisMarker>
    <lines>
      <AxisLine value="1000" label="Label 01" stroke="{stroke1}" labelUpDown="up"
       color="#969596"/>
      <AxisLine value="1500" label="Label 02" stroke="{stroke1}" labelAlign="left"
       labelUpDown="down" color="#969596"/>
      <AxisLine value="2000" label="Label 03" stroke="{stroke1}" labelUpDown="up"
       labelAlign="left" labelRotation="90" color="#969596"/>
      <AxisLine value="Jan" label="January" stroke="{stroke1}" color="#969596"
       labelUpDown="down" labelAlign="right" linePosition="left"
       horizontal="false"/>
      <AxisLine value="Mar" label="March" stroke="{stroke1}" color="#969596"
       labelUpDown="down" labelRotation="90" labelAlign="left" linePosition="left"
       horizontal="false"/>
      <AxisLine value="May" label="May" stroke="{stroke1}" color="#969596"
       labelUpDown="down" labelAlign="left" linePosition="left"
       horizontal="false"/>
      <AxisLine value="Oct" label="October" stroke="{stroke1}" color="#969596"
       labelUpDown="up" labelRotation="90" linePosition="right"
       horizontal="false"/>
      <AxisLine startValue="1000" endValue="2000" label="From-To Line"
       color="#0099FF" labelAlign="right" labelUpDown="up">
        <stroke>
          <Stroke color="#0099FF" weight="1"/>
        </stroke>
      </AxisLine>
    </lines>
  </AxisMarker>
</backgroundElements>
```



See the CodePen 알메이트 차트 - 차트에 수직선, 수평선, 대각선 표시

화살표 선 긋기

<AxisLine> 노드의 enableArrowHead 값을 “true” 로 설정하면 화살표 선을 표시할 수 있습니다. 다음 표는 <AxisLine> 노드에서 화살표를 표시하는데 필요한 속성들에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
arrowHeadLocation	start, end(*)	선의 시작과 끝 중에서 화살표를 표시할 위치를 지정합니다.
arrowHeadType	open(*), close	화살표 머리의 유형을 설정합니다.
arrowLength	숫자 기본값: 10	화살표 머리의 길이를 지정합니다
enableArrowHead	true, false(*)	선에 화살표를 표시할지 여부를 지정합니다.

다음은 차트에 화살표 선을 표시한 예제입니다.

```
<backgroundElements>
  <GridLines/>
  <AxisMarker>
    <lines>
      <AxisLine horizontal="false" startValue="Sep" endValue="Nov"
       verticalStartValue="2000" verticalEndValue="2300" label="연중 최고 매출, 기록
       경신" stroke="{stroke1}" labelUpDown="up" labelAlign="right"
       labelYOffset="10" color="#969596" enableArrowHead="true"
       arrowHeadType="close"/>
      <AxisLine horizontal="false" startValue="Aug" endValue="Oct"
       verticalStartValue="600" verticalEndValue="1000" label="연중 최저 매출"
       stroke="{stroke1}" labelUpDown="up" labelAlign="right" labelYOffset="10"
       color="#969596" enableArrowHead="true" arrowHeadType="close"/>
    </lines>
  </AxisMarker>
</backgroundElements>
```

See the CodePen 알메이트 차트 - 차트에 화살표 선 표시

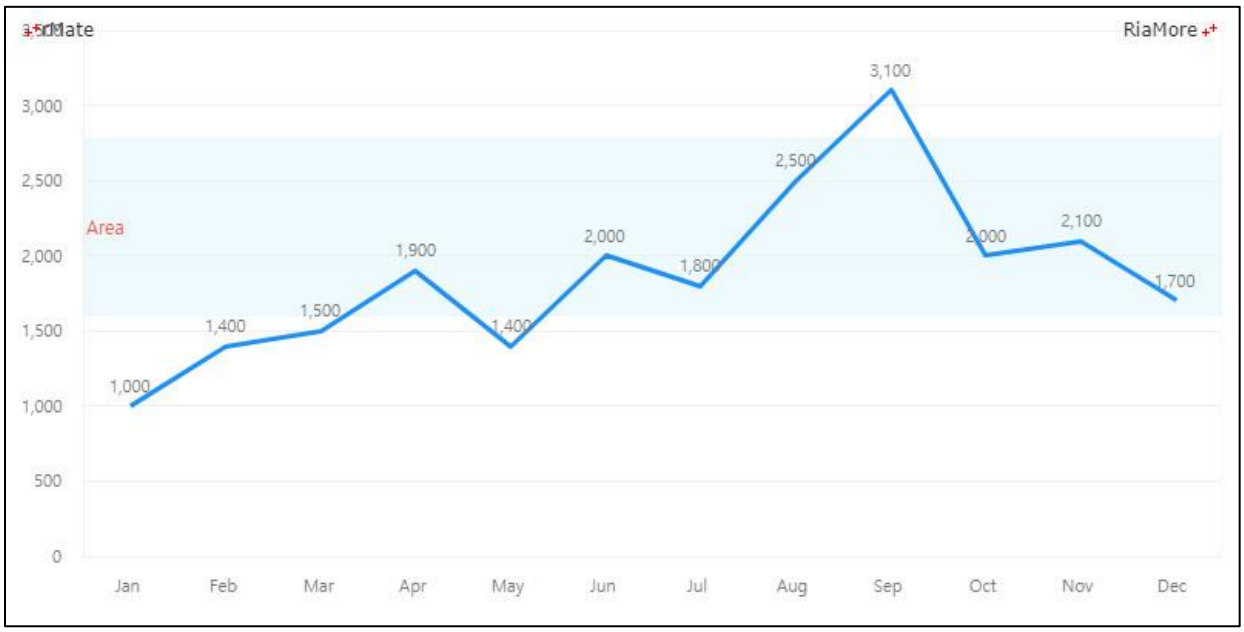
3.9 범위 지정하기

차트의 배경에 범위(영역)를 표시할 수 있습니다. 이러한 작업은 <backgroundElements> 속성에 <AxisMarker> 노드를 정의하고, <AxisMarker> 노드의 <ranges> 속성에 <AxisRange> 노드를 설정함으로써 가능합니다. 다음은 <AxisRange> 노드의 주요 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명
endValue	숫자	범위(영역)가 끝나는 지점의 값을 지정합니다.
fill	<LinearGradient>, <RadialGradient>, <SolidColor>	범위(영역)의 공간에 칠할 색의 스타일을 지정합니다.
fill	true(*), false	표시되는범위(영역)가 수평 범위(영역)인지 여부를 설정합니다.
label	텍스트	범위(영역)에 표시될 레이블을 지정합니다.
labelHorizontalAlign	center(*), left, right	레이블의 수평 정렬을 설정합니다.
labelVerticalAlign	bottom, middle(*), top	레이블의 수직 정렬을 설정합니다.
startValue	숫자	범위(영역)가 시작되는 지점의 값을 지정합니다.

다음은 차트에 범위를 지정하고 지정된 범위의 좌측에 레이블을 표시하는 예제입니다.

```
<backgroundElements>
  <GridLines/>
  <AxisMarker>
    <ranges>
      <AxisRange startValue="1600" endValue="2800" label="Area"
       labelHorizontalAlign="left" color="#EA594E">
        <fill>
          <SolidColor color="#e5f6fe" alpha="0.6"/>
        </fill>
      </AxisRange>
    </ranges>
  </AxisMarker>
</backgroundElements>
```



See the CodePen 알메이트 차트 - 차트에 범위 표시

다음은 수직 범위와 수평 범위를 함께 표시하는 예제입니다.

```
<backgroundElements>
  <GridLines/>
  <AxisMarker>
    <ranges>
      <AxisRange startValue="400" endValue="800" label="Range 1" fontSize="11"
       labelHorizontalAlign="left" fontFamily="Malgun Gothic">
        <fill>
          <SolidColor color="#00FF99" alpha="0.2"/>
        </fill>
      </AxisRange>
      <AxisRange startValue="Mar" endValue="May" label="Range 2" labelYOffset="-10"
       fontSize="11" labelVerticalAlign="bottom" horizontal="false"
       fontFamily="Malgun Gothic">
        <fill>
          <SolidColor color="#0099FF" alpha="0.2"/>
        </fill>
      </AxisRange>
      <AxisRange startValue="Jul" endValue="Nov" label="Range 3" labelYOffset="-10"
       fontSize="11" labelVerticalAlign="bottom" horizontal="false"
       fontFamily="Malgun Gothic">
        <fill>
          <SolidColor color="#dd99FF" alpha="0.2"/>
        </fill>
      </AxisRange>
    </ranges>
  </AxisMarker>
</backgroundElements>
```

See the CodePen 알메이트 차트 - 차트에 수직 범위와 수평 범위를 함께 표시

3.10 메모

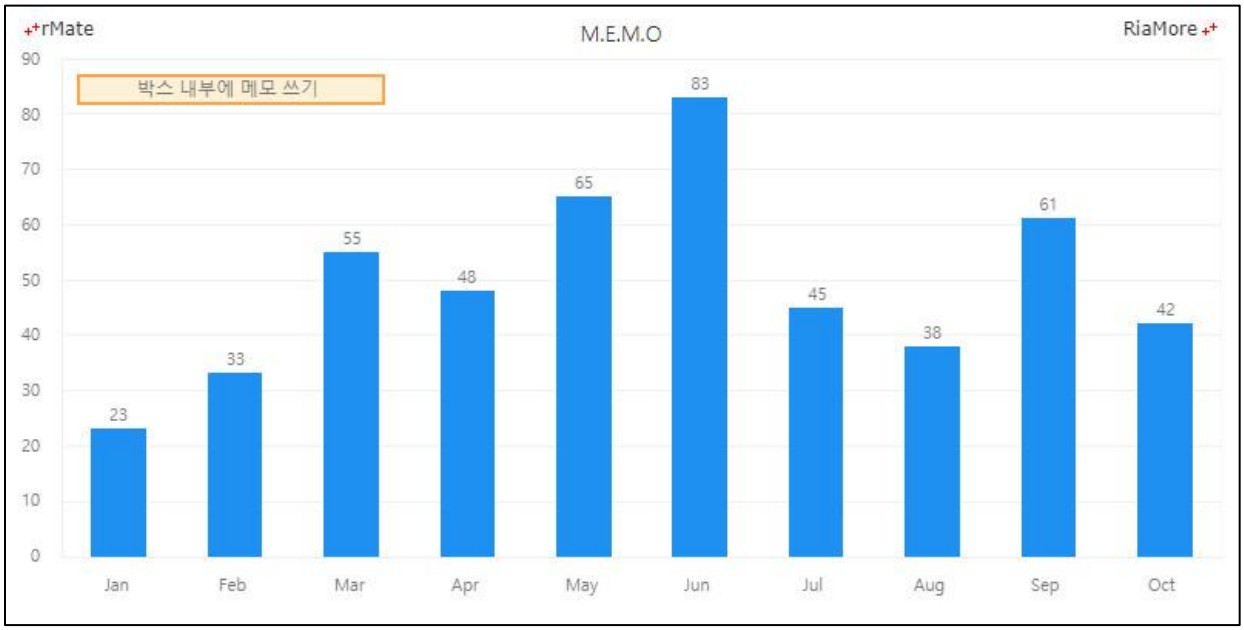
차트의 내용을 설명하는 메모(주석)를 차트에 표시할 수 있습니다. 이러한 작업은 <annotationElements> 속성에 <CanvasElement> 노드를 정의하고, <CanvasElement> 노드의 자식 노드에 <CanvasLabel> 노드를 설정함으로써 가능합니다. 다음은 <CanvasLabel> 노드의 주요 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명
backgroundColor	#16 진수 컬러 코드 표기 기본값: #FFFFFF	메모가 표시되는 사각형 박스의 배경색의 스타일을 지정합니다.
borderColor	#16 진수 컬러 코드 표기 기본값: #000000	메모가 표시되는 사각형 박스의 경계선의 색의 스타일을 지정합니다.
bottom	숫자	차트의 하단 가장자리와 메모가 표시될 위치 사이의 여백의 크기를 지정합니다.
height	숫자	메모가 표시되는 사각형 박스의 높이를 지정합니다.
itemClickJsFunction	자바스크립트 함수명	메모를 클릭하면 실행될 자바스크립트 함수명을 지정합니다.
left	숫자	차트의 좌측 가장자리와 메모가 표시될 위치 사이의 여백의 크기를 지정합니다.
right	숫자	차트의 우측 가장자리와 메모가 표시될 위치 사이의 여백의 크기를 지정합니다.
text	텍스트	메모로 표시될 텍스트를 지정합니다.
textAlign	center(*), left, right	메모로 표시될 텍스트의 수평 정렬을 설정합니다.
top	숫자	차트의 상단 가장자리와 메모가 표시될 위치 사이의 여백의 크기를 지정합니다.
verticalAlign	bottom, middle(*), top	메모로 표시될 텍스트의 수직 정렬을 설정합니다.

width	숫자	메모가 표시되는 사각형 박스의 넓이를 지정합니다.

다음은 차트의 좌측 상단(left = “10”, top = “10”)에 메모를 표시하는 예제입니다.

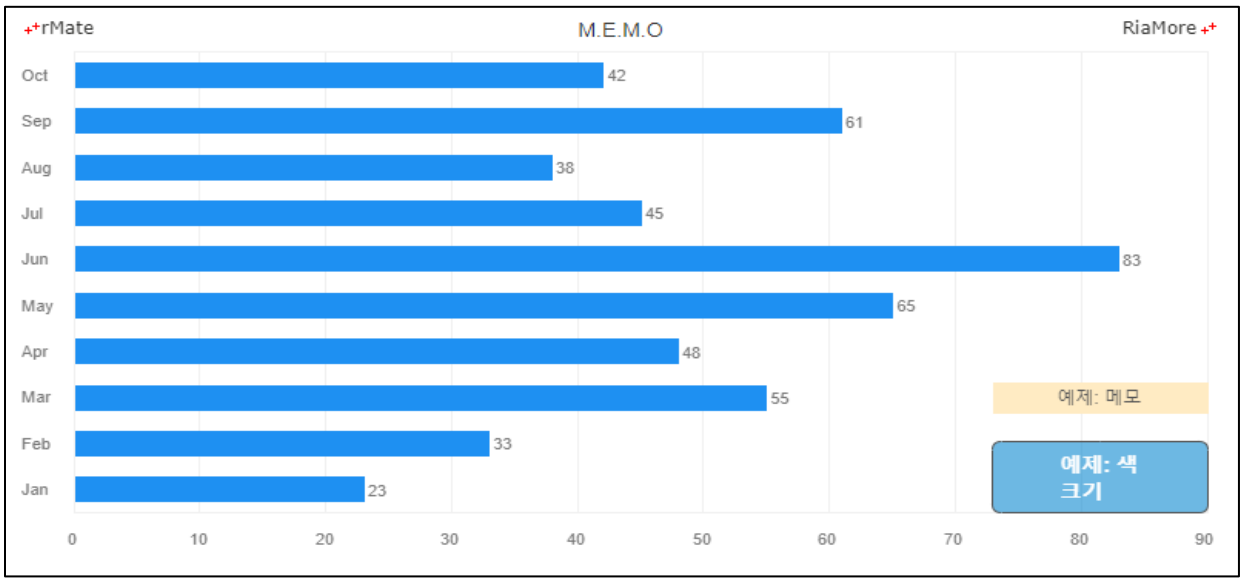
```
<annotationElements>
  <CanvasElement>
    <CanvasLabel fontFamily="arial" left="10" top="10" height="20" width="200"
       textAlign="center" fontSize="12" text="This is the memo box."
       borderStyle="solid" borderColor="#f6a54c" borderThickness="2"
       backgroundAlpha="0.7" backgroundColor="#ffebc3"/>
  </CanvasElement>
</annotationElements>
```



See the CodePen 알메이트 차트 - 차트의 좌측 상단에 메모 표시

다음은 차트의 우측 하단에 두 개의 메모를 표시하는 예제입니다.

```
<annotationElements>
  <CanvasElement>
    <CanvasLabel width="140" height="22" bottom="70" right="0" textAlign="center"
       text="Example: Memo" fontSize="12" color="#555555" borderColor="#7D7F7E"
       borderThickness="2" borderStyle="none" backgroundColor="#ffebc3"
       backgroundAlpha="1" fontFamily="Arial"/>
    <CanvasLabel width="140" height="50" bottom="0" right="0" textAlign="center"
       text="Example: Color Size" fontSize="14" color="#ffffff" fontWeight="bold"
       borderColor="#555555" borderThickness="1" borderStyle="solid"
       backgroundColor="#0c88d0" backgroundAlpha="0.6" borderRadius="6"/>
  </CanvasElement>
</annotationElements>
```



See the CodePen 알메이트 차트 - 차트의 우측 하단에 메모 표시

상대 좌표에 메모 표시

메모를 표시하는 위치를 절대 좌표(left, right, top, bottom)를 기준으로 설정하면 처음 생성했던 차트의 가로, 세로 크기가 달라질 경우 처음 의도했던 위치가 아닌 다른 위치에 표시될 수 있습니다. 차트의 크기가 달라졌을 때 메모의 위치가 변경되는 것을 방지하려면 상대 좌표를 기준으로 메모의 위치를 설정해야 합니다. 다음 표는 <CanvasLabel> 노드에서 메모의 위치를 상대 좌표로 설정하기 위한 속성들에 관한 것입니다.

속성명	유효값 (*: 기본값)	설명
percentLeft	1 과 100 사이의 숫자 기본값: 0	문자열이 표시되는 수평위치를 X 축 최대값의 위치를 100 으로 했을 때의 퍼센트 값으로 설정합니다.
percentTop	1 과 100 사이의 숫자 기본값: 0	문자열이 표시되는 수직위치를 Y 축 최소값의 위치를 100 으로 했을 때의 퍼센트 값으로 설정합니다.

주의

> 상대 좌표(percentLeft, percentTop)와 절대 좌표(left, right, top, bottom)를 동시에 설정할 경우 절대 좌표가 적용됩니다.

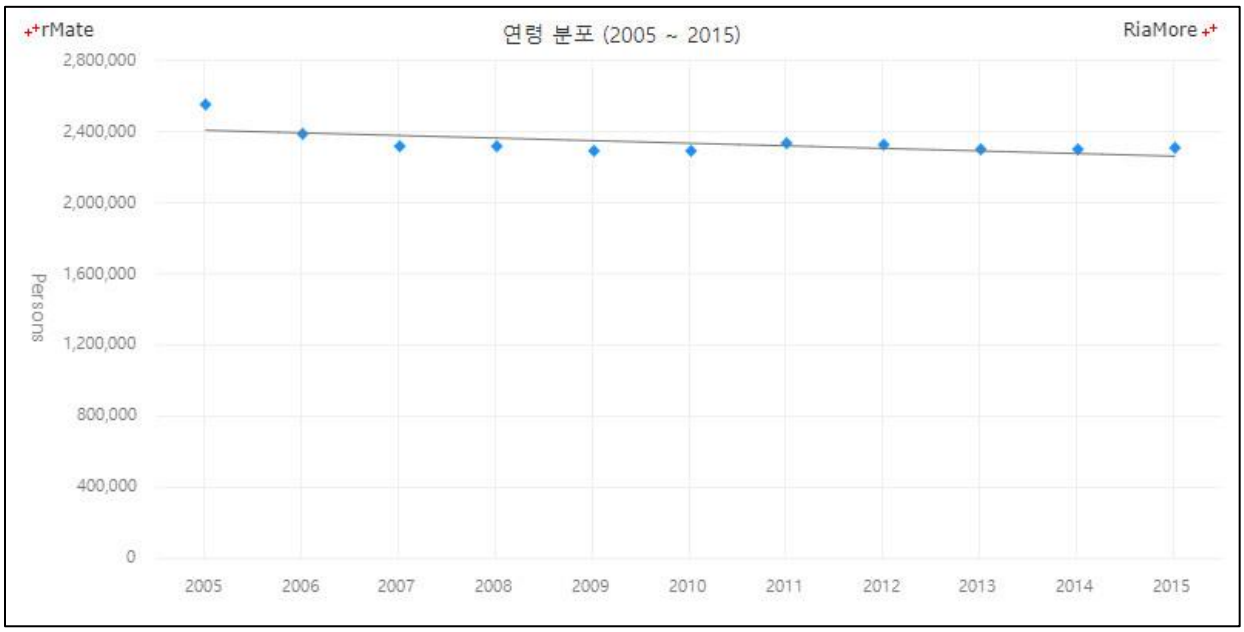
3.11 추세선

카테시안 계열의 차트 (예, 바 차트, 컬럼 차트, 라인 차트, etc) 유형에서는 추세선을 표시할 수 있습니다. 이러한 작업은 데이터 시리즈 (예, <Column2DSeries> 노드)의 showTrendLine 속성을 “true”로 설정함으로써 가능합니다. 다음은 추세선의 표시와 관련된 속성들을 설명한 표입니다.

속성명	유효값 (*: 기본값)	설명
movingAveragePeriod	숫자	trendLineType 속성값이 “movingAverage” 인 경우, 이동 평균선의 범위값을 지정합니다.
polynomialOrder	숫자	trendLineType 속성값이 “polynomial” 인 경우, 다항식에 대한 차수를 설정합니다.
showTrendLine	true, false(*)	차트에 추세선을 표시할지 여부를 설정합니다.
showTrendLineEquation	true, false(*)	차트에 표현되는 추세선의 공식을 표시할지 여부를 설정합니다.
showTrendLineRSquared	true, false(*)	차트에 표현되는 추세선의 R-squared 값을 표시할지 여부를 설정합니다.
trendLineForm	curve(*), segment, step, reverseStep	추세선의 유형을 지정합니다.
trendLineType	linear(*), logarithmic, exponential, polynomial, power, movingAverage	추세선의 계산 유형을 지정합니다.

다음은 플롯 차트에서 선형(linear) 추세선을 표시하는 예제입니다.

```
<series>
  <Plot2DSeries yField="population" displayName="Population" showTrendLine="true"/>
</series>
```



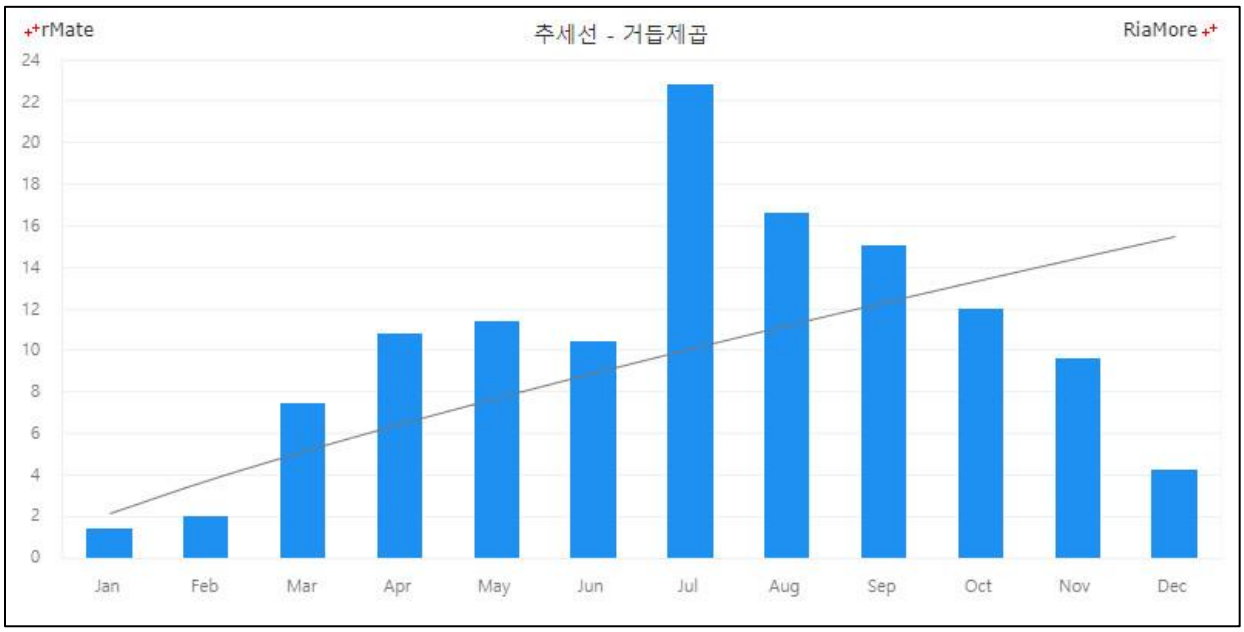
See the CodePen 알메이트 차트 - 플롯 차트에서 선형 추세선 표시

주의

> trendLineType 속성의 기본값이 “linear” 이므로 위 예제에서는 trendLineType 속성이 설정되지 않았지만 자동으로 선형 추세선이 표시됩니다.

다음은 컬럼 차트에서 거듭제곱(power) 추세선을 표시하는 예제입니다.

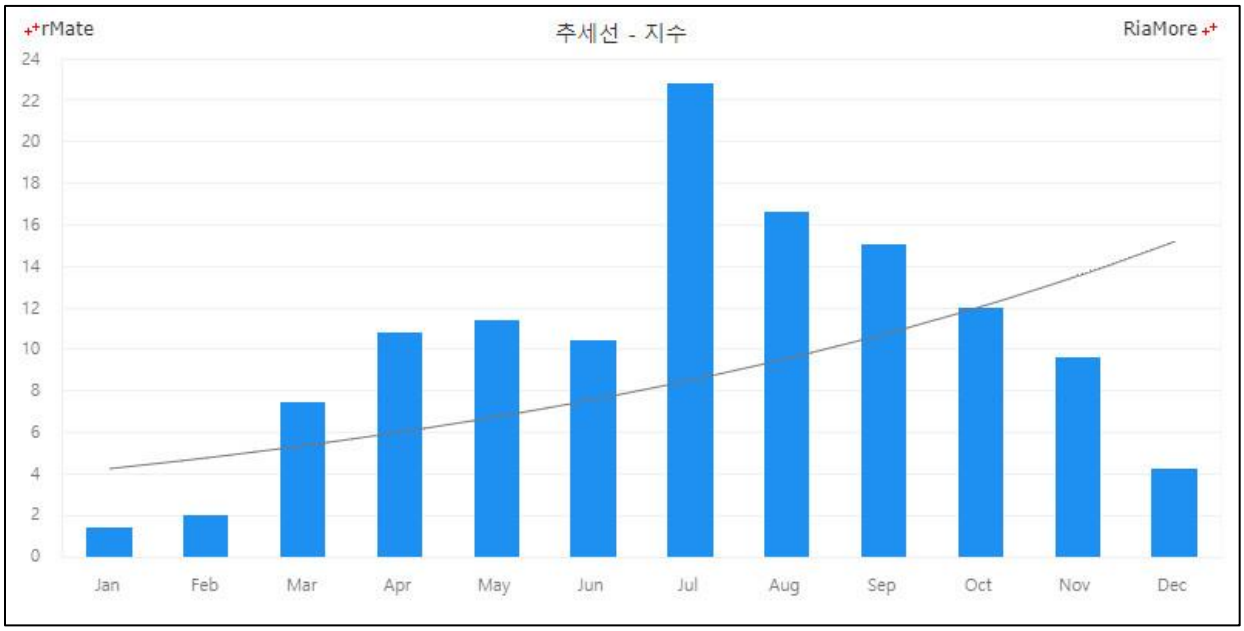
```
<series>
  <Column2DSeries yField="Vancouver" displayName="Vancouver" showTrendLine="true"
       trendLineType="power"/>
</series>
```



See the CodePen 알메이트 차트 - 컬럼 차트에서 거듭제곱 추세선 표시

```
<series>
  <Column2DSeries yField="Vancouver" displayName="Vancouver" showTrendLine="true"
       trendLineType="exponential"/>
</series>
```

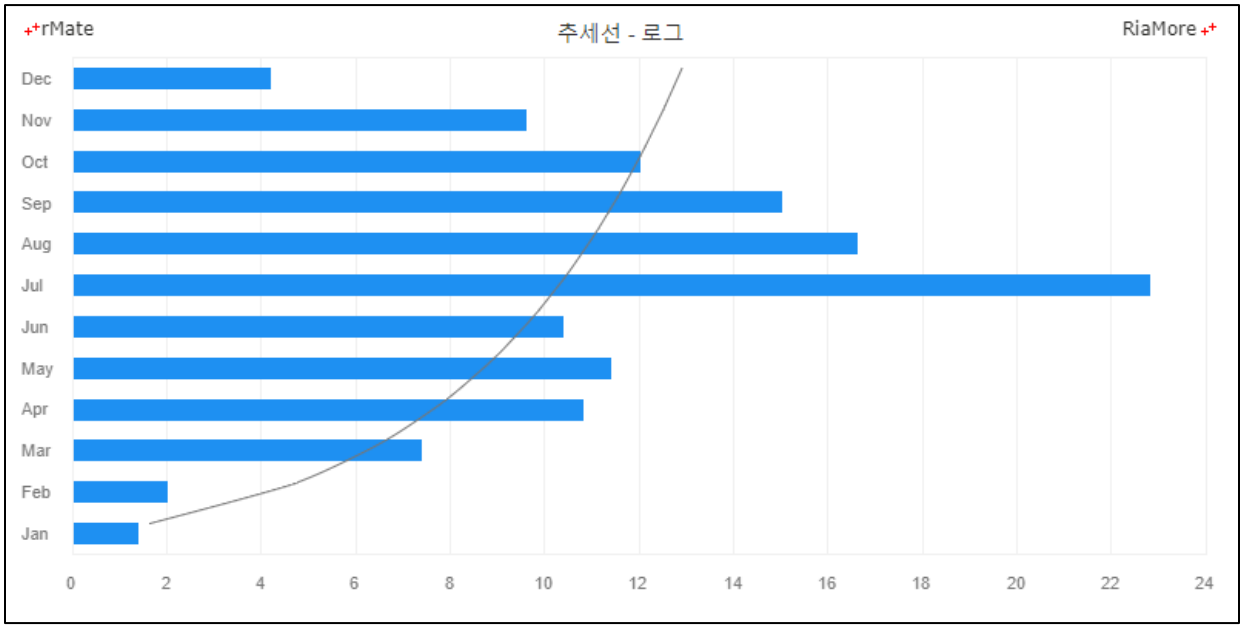
다음은 컬럼 차트에서 지수(exponential) 추세선을 표시하는 예제입니다.



See the CodePen 알메이트 차트 - 컬럼 차트에서 지수 추세선 표시

다음은 바 차트에서 로그(logarithmic) 추세선을 표시하는 예제입니다.

```
<series>
  <Bar2DSeries xField="Vancouver" displayName="Vancouver" showTrendLine="true"
       trendLineType="logarithmic"/>
</series>
```



See the CodePen 알메이트 차트 - 바 차트에서 로그 추세선 표시

4. 차트 종류

4.1 라인 차트

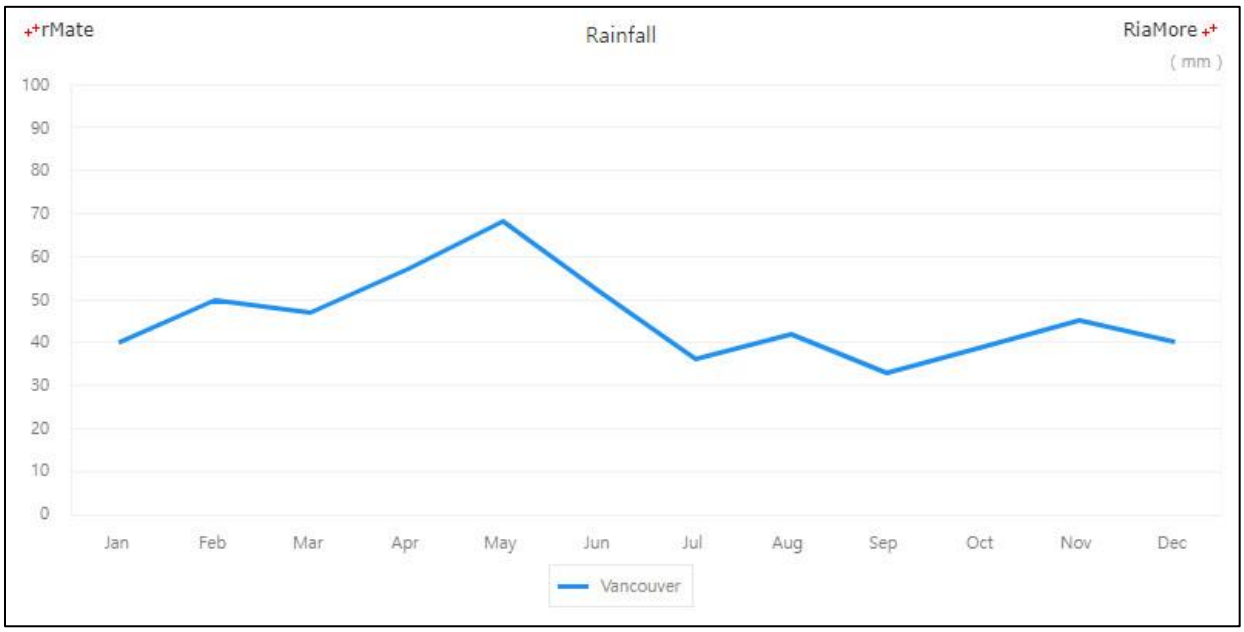
라인 차트는 여러 분야에서 일반적으로 많이 활용되는 차트 종류 중의 하나로써 일련의 데이터 포인트들을 선으로 연결하여 표현하는 차트입니다. 라인 차트는 <Line2DChart> 노드의 series 속성값에 <Line2DSeries> 노드를 설정하여 생성할 수 있습니다. 라인 차트에서 그려지는 선의 형태는 <Line2DSeries> 노드의 form 속성을 통해서 설정할 수 있습니다. 다음에는 form 속성에 설정 가능한 값과 이에 따른 표현 방식이 설명되어 있습니다.

- segment : 각 데이터 포인트를 직선으로 연결합니다. (기본값)
- curve : 각 데이터 포인트를 곡선으로 연결합니다.
- step : 수평 직선으로 시작해서 각 데이터 포인트를 계단선으로 연결합니다.
- reverseStep : 수직 직선으로 시작해서 각 데이터 포인트를 계단선으로 연결합니다.

직선 라인 차트

다음은 선의 모양을 직선으로 표현하는 직선 라인 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DChart showDataTips="true" dataTipDisplayMode="axis">
  ...
  <series>
    <Line2DSeries form="segment" yField="Vancouver" displayName="Vancouver">
      <showDataEffect>
        <SeriesClip duration="1000"/>
      </showDataEffect>
    </Line2DSeries>
  </series>
  <annotationElements>
    <CrossRangeZoomer enableZooming="false" horizontalLabelFormatter="{numFmt}"
       horizontalStrokeEnable="false"/>
  </annotationElements>
</Line2DChart>
```



See the CodePen 알메이트 차트 - 직선 라인 차트

주의

> form 속성의 기본값이 “segment” 이므로 form 속성을 설정하지 않으면 자동으로 segment(직선) 형의 라인 차트가 생성됩니다.

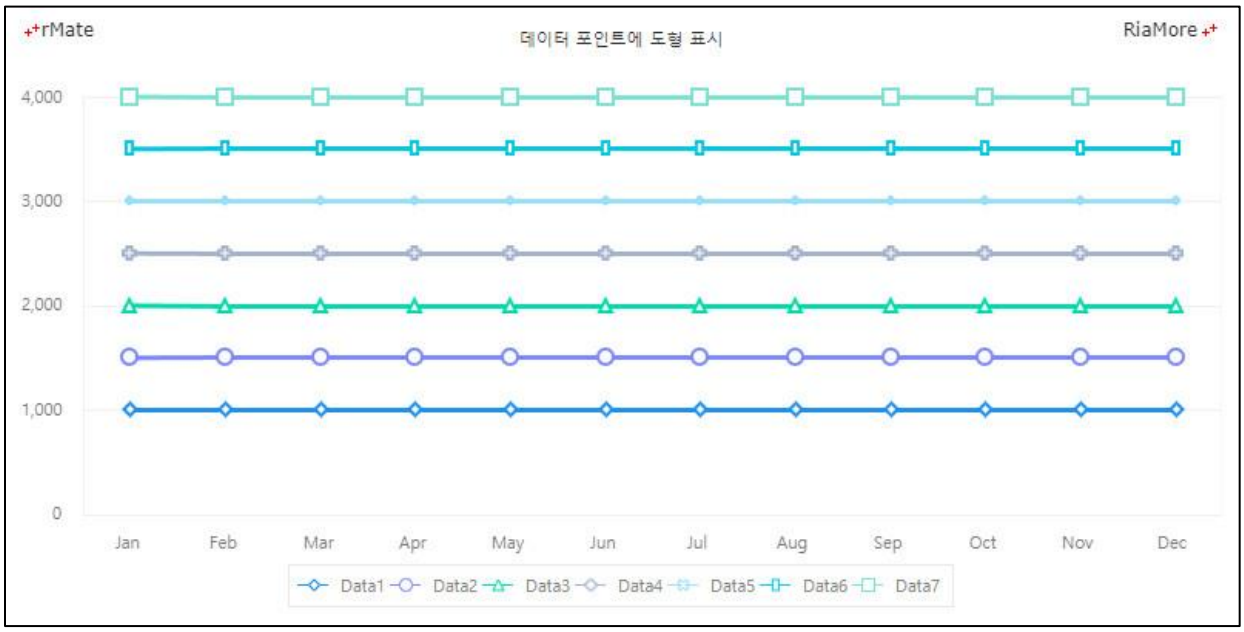
데이터 포인트에 도형 표시

라인 차트에서 <Line2DSeries> 노드의 itemRenderer 속성(기본값: “none”)을 설정하여 데이터 포인트에 특정한 모양의 도형을 표시할 수 있습니다. itemRenderer 속성에 설정 가능한 값과 표시되는 도형의 모양은 다음과 같습니다.

- CircleItemRenderer : 데이터 포인트에 원을 표시합니다.
- TriangleItemRenderer : 데이터 포인트에 삼각형을 표시합니다.
- RectangleItemRenderer : 데이터 포인트에 사각형을 표시합니다.
- DiamondItemRenderer : 데이터 포인트에 다이아몬드를 표시합니다.
- CrossItemRenderer : 데이터 포인트에 십자형을 표시합니다.
- XShapeItemRenderer : 데이터 포인트에 X 자 모양을 표시합니다.
- IShapeItemRenderer : 데이터 포인트에 I 자 모양을 표시합니다.

다음은 7 가지 데이터 렌더러를 통해서 라인 차트의 데이터 포인트에 도형을 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 도형의 크기는 radius 속성을 통해서 조절이 가능하고 도형의 내부에 칠해지는 색은 fill 속성(이 예제에서는 #ffffff)을 통해서 설정이 가능합니다.

```
<Line2DChart showDataTips="true">
  ...
  <series>
    <Line2DSeries yField="Data1" radius="6" displayName="Data1"
       itemRenderer="DiamondItemRenderer" fill="#ffffff" />
    <Line2DSeries yField="Data2" radius="6" displayName="Data2"
       itemRenderer="CircleItemRenderer" fill="#ffffff" />
    <Line2DSeries yField="Data3" radius="6" displayName="Data3"
       itemRenderer="TriangleItemRenderer" fill="#ffffff" />
    <Line2DSeries yField="Data4" radius="6" displayName="Data4"
       itemRenderer="CrossItemRenderer" fill="#ffffff" />
    <Line2DSeries yField="Data5" radius="6" displayName="Data5"
       itemRenderer="XShapeItemRenderer" fill="#ffffff" />
    <Line2DSeries yField="Data6" radius="6" displayName="Data6"
       itemRenderer="IShapeItemRenderer" fill="#ffffff" />
    <Line2DSeries yField="Data7" radius="6" displayName="Data7"
       itemRenderer="RectangleItemRenderer" fill="#ffffff" />
  </series>
</Line2DChart>
```

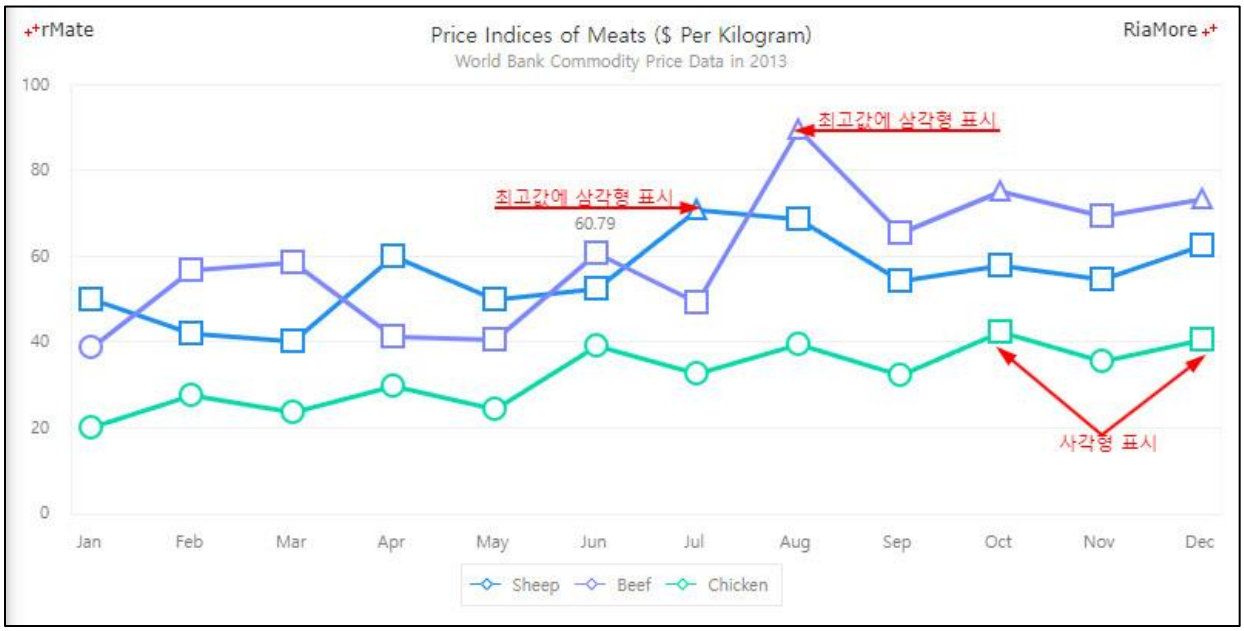


See the CodePen 알메이트 차트 - 데이터 포인트에 도형 표시

동일 라인에 다른 도형 표시

라인에 표시되는 도형은 시리즈 노드의 itemRenderer 속성에 설정되기 때문에 한 라인(데이터 시리즈)의 데이터 포인트들에는 모두 동일한 도형이 표시됩니다. 만약 데이터의 값에 따라서 도형의 종류를 다르게 표시하고자 할 경우에는 <Line2DSeries> 노드의 itemRendererJsFunction 속성에 자바스크립트 함수명을 지정하여 구현할 수 있습니다. 다음은 세 개의 라인에 각 데이터 포인트들의 값에 따라서 다른 도형을 표시하는 예제입니다.

```
<series>
  <Line2DSeries yField="Sheep" fill="#ffffff" radius="8" displayName="Sheep"
        itemRenderer="DiamondItemRenderer" itemRendererJsFunction="itemRendererFunc"
        />
  <Line2DSeries labelPosition="up" yField="Beef" fill="#ffffff" radius="8"
        displayName="Beef" showValueLabels="[5]" itemRenderer="DiamondItemRenderer"
        itemRendererJsFunction="itemRendererFunc" />
  <Line2DSeries yField="Chicken" fill="#ffffff" radius="8" displayName="Chicken"
        itemRenderer="DiamondItemRenderer" itemRendererJsFunction="itemRendererFunc"
        />
</series>

function itemRendererFunc(id, index, data, values) {
  if(values[1] < 40)
    return "CircleItemRenderer";
  else if(values[1] < 70)
    return "RectangleItemRenderer";
  else
    return "TriangleItemRenderer";
}
```

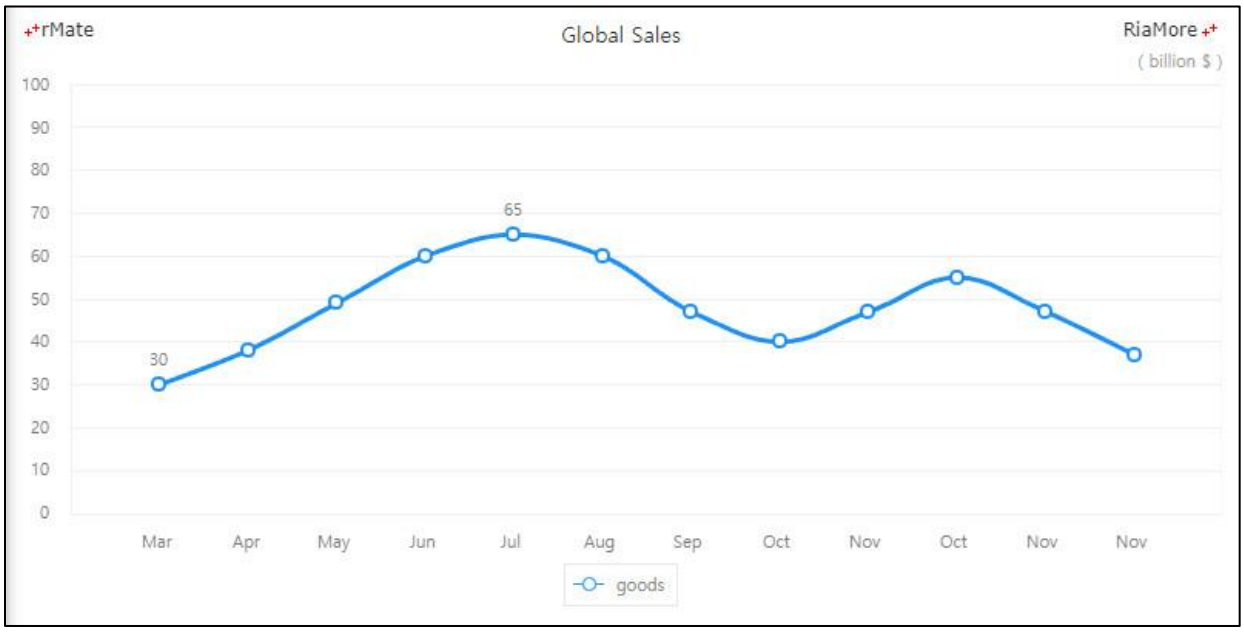


See the CodePen 알메이트 차트 - 동일 라인에 다른 도형 표시

곡선 라인 차트

다음은 선의 모양을 곡선으로 표현하는 곡선 라인 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 데이터 포인트에 원(CircleItemRenderer)이 표시되며, 처음과 다섯 번째 데이터 포인트(showValueLabels = “[0, 4]”)의 위(labelPosition = “up”)에 데이터 값(레이블)이 표시되도록 설정되었습니다.

```
<Line2DChart showDataTips="true" >
  ...
  <series>
    <Line2DSeries form="curve" itemRenderer="CircleItemRenderer" labelPosition="up"
        yField="goods" displayName="goods" showValueLabels="[0,4]">
      <showDataEffect>
        <SeriesInterpolate duration="1000"/>
      </showDataEffect>
    </Line2DSeries>
  </series>
</Line2DChart>
```



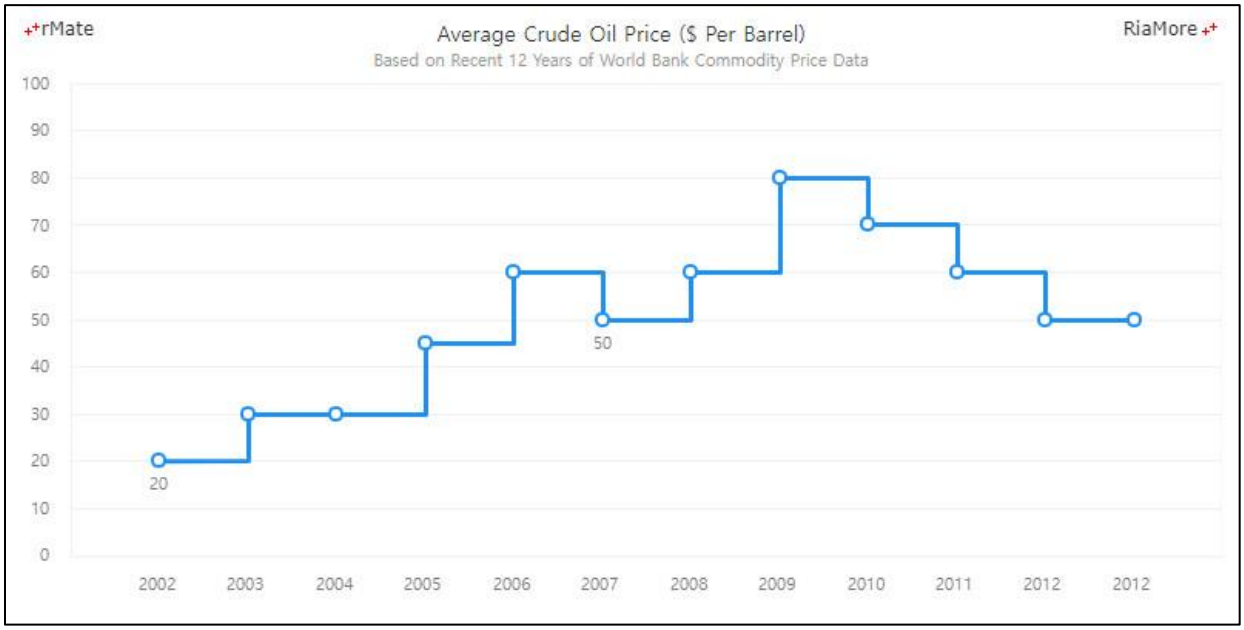
See the CodePen 알메이트 차트 - 곡선 라인 차트

계단 라인 차트

다음은 계단 선 모양을 표현하는 계단 라인 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 데이터 포인트에 원(CircleItemRenderer)이 표시되며, 처음과 여섯 번째

데이터 포인트(showValueLabels = “[0, 5]”)의 아래(labelPosition = “down”)에 데이터 값(레이블)이 표시되도록 설정되었습니다.

```
<Line2DChart showDataTips="true" >
  ...
  <series>
    <Line2DSeries labelPosition="down" showValueLabels="[0,5]" yField="Price"
       form="step" radius="5" itemRenderer="CircleItemRenderer" displayName="$ Per
       Barrel">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </Line2DSeries>
  </series>
</Line2DChart>
```



See the CodePen 알메이트 차트 - 계단 라인 차트

점선 라인 차트

라인 차트에 표시되는 선을 직선이 아닌 점선으로 표현이 가능합니다. <Line2DSeries> 노드의 lineStyle 속성을 “dashLine” 으로 설정하면 점선이 표현됩니다. 점선의 길이에 관한 설정은 다음에서 설명하는 속성을 참조하십시요.

속성명	유효값 (*: 기본값)	설명
dashLinePattern	숫자 기본값: 10	점선의 길이를 픽셀 단위로 지정합니다. 예를 들어 값이 “5” 이면 5px 길이의 선과 5px 길이의 공백이 반복해서 표현됩니다.

다음은 dashLinePattern 속성값을 “3” 으로 설정하여 점선을 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DSeries yField="goods" lineStyle="dashLine" dashLinePattern="3"
       itemRenderer="CircleItemRenderer" radius="5" displayName="goods" />
<Line2DSeries yField="income" lineStyle="dashLine" dashLinePattern="3"
       itemRenderer="CircleItemRenderer" radius="5" displayName="income" />
<Line2DSeries yField="service" lineStyle="dashLine" dashLinePattern="3"
       itemRenderer="CircleItemRenderer" radius="5" displayName="service" />
```

See the CodePen 알메이트 차트 - 점선 라인 차트

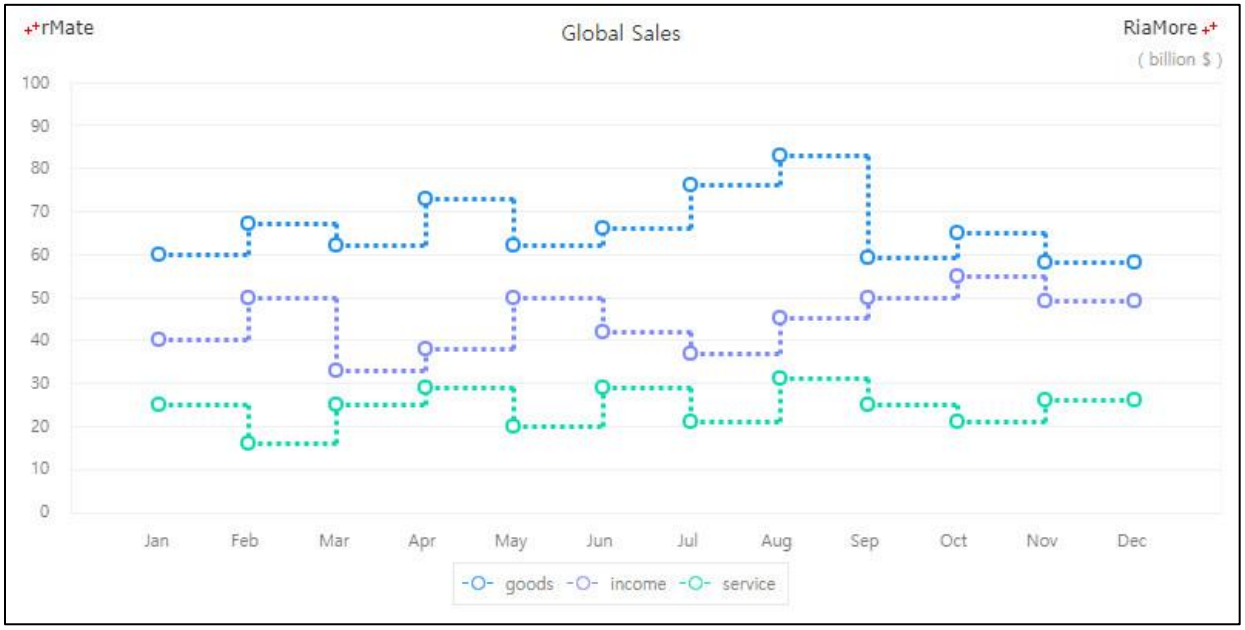
곡선 라인 차트나 계단 라인 차트에도 점선을 표현할 수 있습니다. 다음은 form 속성값을 “curve”, lineStyle 속성값을 “dashLine” 으로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DChart showDataTips="true">
  ...
  <series>
    <Line2DSeries yField="goods" fill="#ffffff" radius="5" form="curve"
        lineStyle="dashLine" dashLinePattern="3" itemRenderer="CircleItemRenderer"
        displayName="goods" />
    <Line2DSeries yField="income" fill="#ffffff" radius="5" form="curve"
        lineStyle="dashLine" dashLinePattern="3" itemRenderer="CircleItemRenderer"
        displayName="income" />
  </series>
</Line2DChart>
```

See the CodePen 알메이트 차트 - 곡선 점선 라인 차트

다음은 form 속성값을 “step”, lineStyle 속성값을 “dashLine” 으로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DChart showDataTips="true">
  ...
  <series>
    <Line2DSeries fill="#ffffff" yField="goods" radius="5" form="step"
        lineStyle="dashLine" dashLinePattern="3" itemRenderer="CircleItemRenderer"
        displayName="goods" />
    <Line2DSeries fill="#ffffff" yField="income" radius="5" form="step"
        lineStyle="dashLine" dashLinePattern="3" itemRenderer="CircleItemRenderer"
        displayName="income" />
    <Line2DSeries fill="#ffffff" yField="service" radius="5" form="step"
        lineStyle="dashLine" dashLinePattern="3" itemRenderer="CircleItemRenderer"
        displayName="service" />
  </series>
</Line2DChart>
```



See the CodePen 알메이트 차트 - 계단 점선 라인 차트

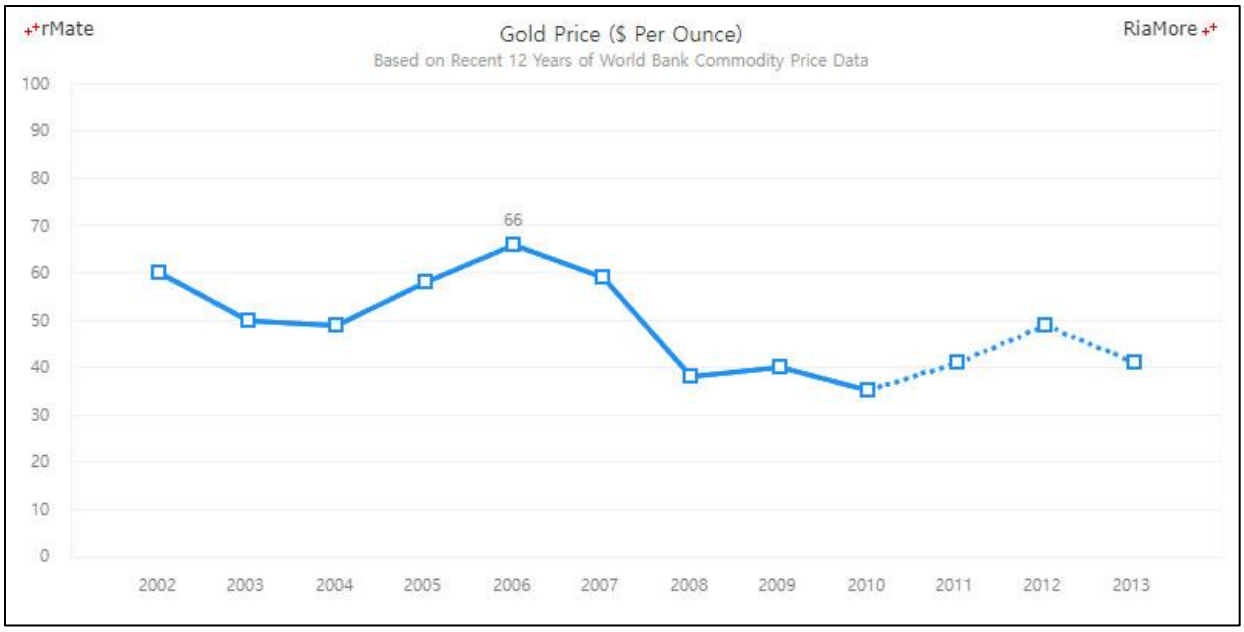
점선과 직선을 한 라인에 표시

라인 차트에서 한 라인에 점선과 직선을 함께 표현할 수 있습니다. 이 때 점선이 시작되는 위치는 <Line2DSeries> 노드에 설정되는 다음 속성들에 의해서 결정됩니다.

속성명	유효값 (*: 기본값)	설명
dashLinePlacement	before(*), after	dashLineSeperatePos 속성에 설정된 데이터 아이템 이전 혹은 이후에 점선을 표시할지 여부를 지정합니다.
dashLineSeperatePos	숫자 기본값: 0	점선이 표시되기 시작(혹은 종료)하는 기준이 되는 데이터 아이템을 지정합니다.

다음은 9 번째 데이터 포인트(dashLineSeperatePos = “8”) 이후(dashLinePlacement = “after”)에 점선을 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DChart showDataTips="true" >
  ...
  <series>
    <Line2DSeries lineStyle="dashLine" dashLinePattern="3" dashLineSeperatePos="8"
        dashLinePlacement="after" labelPosition="up" showMaxValueLabel="true"
        yField="Price" radius="5" itemRenderer="RectangleItemRenderer"
        displayName="($ Per Ounce)" />
  </series>
</Line2DChart>
```

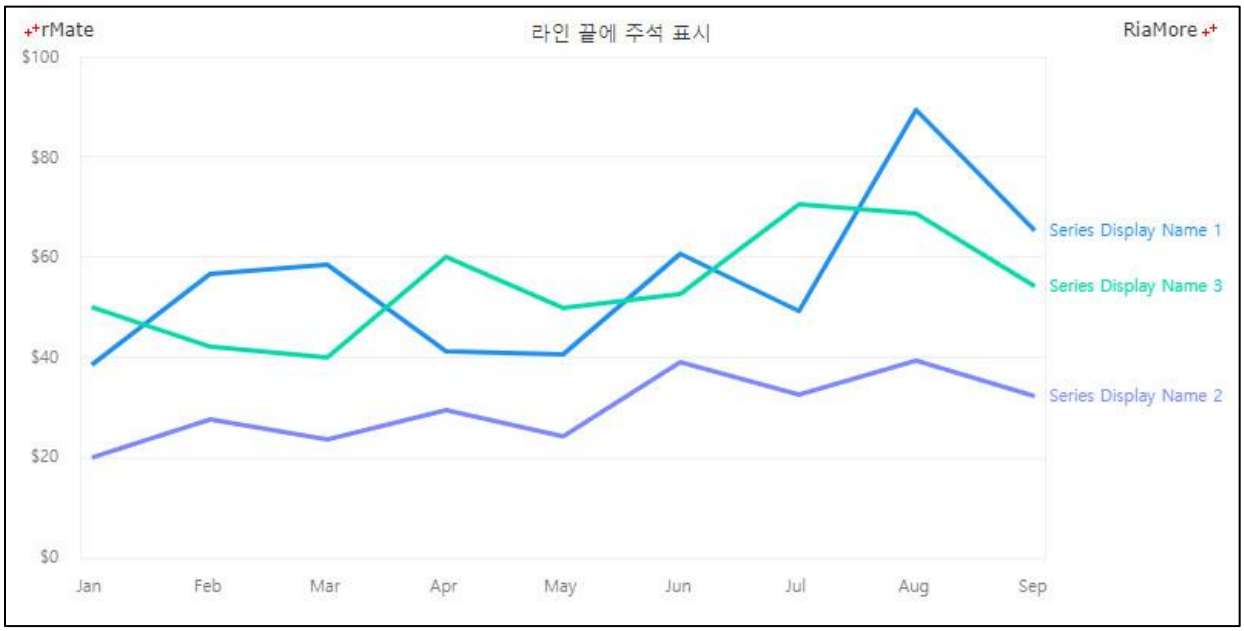


See the CodePen 알메이트 차트 - 점선과 직선을 한 라인에 표현

라인 끝에 주석 표시

여러 개의 데이터 시리즈 값들을 표현하는 라인들이 함께 표시되어 있는 경우, 차트에 범례가 있더라도 범례 항목과 해당 데이터 시리즈를 표현하는 라인을 직관적으로 알아보기 어려울 수 있습니다. 이 경우 라인 차트의 라인 끝에 해당 데이터 시리즈의 설명을 표시하면 차트를 직관적으로 이해하기에 도움이 됩니다. 라인의 끝에 설명을 표시하기 위해서는 <Line2DChart> 노드의 endPointDisplayName 속성을 “true” 로 지정합니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DChart showDataTips="true" gutterTop="6" endPointDisplayName="true"
       dataTipDisplayMode="axis">
  ...
  <series>
    <Line2DSeries yField="Data1" displayName="Series Display Name 1" />
    <Line2DSeries yField="Data2" displayName="Series Display Name 2" />
    <Line2DSeries yField="Data3" displayName="Series Display Name 3" />
  </series>
</Line2DChart>
```



See the CodePen 알메이트 차트 - 라인 끝에 주석 표시

4.2 컬럼 차트

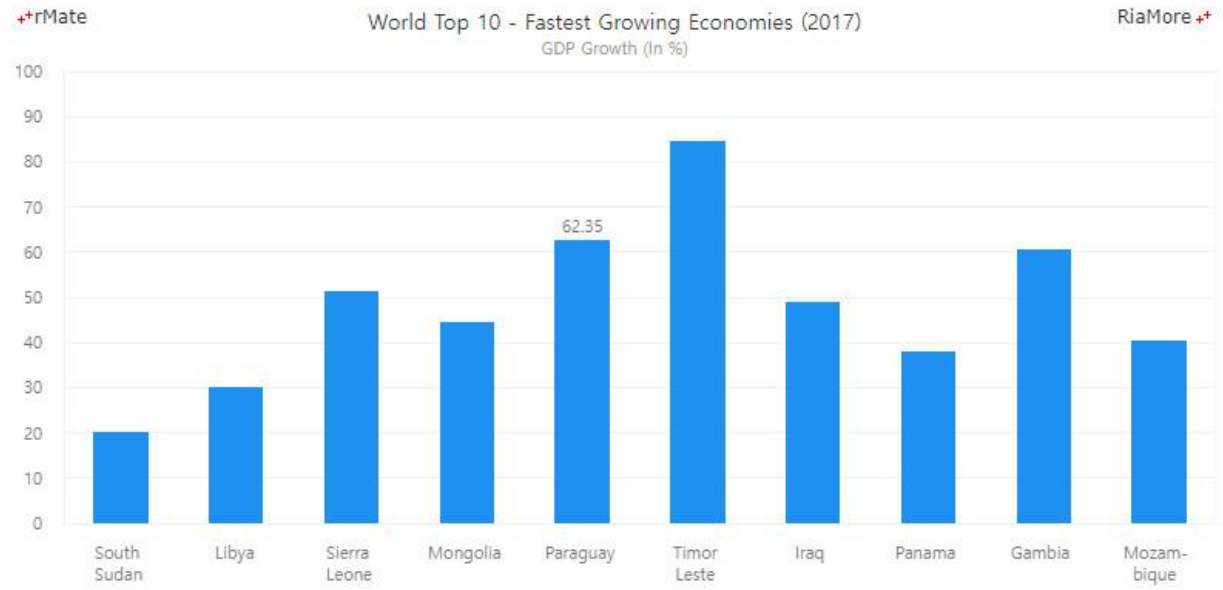
컬럼 차트는 데이터의 크기를 세로 막대의 길이로 표현하는 차트입니다. 일반적으로 숫자 축은 차트의 왼쪽에 세로 축(Y 축)으로 표시되고 카테고리 축은 차트의 하단에 가로 축(X 축)으로 표시됩니다. 컬럼 차트는 <Column2DChart> 노드의 series 속성값에 <Column2DSeries> 노드를 설정하여 생성할 수 있습니다.

컬럼 차트의 데이터 시리즈가 여러 개일 경우 (<Column2DSeries> 노드가 여러 개 설정 됨) 데이터 시리즈들이 표현되는 방식은 <Column2DChart> 노드의 type 속성의 설정에 따릅니다.

- clustered : 데이터 시리즈들이 카테고리 별로 클러스터링되어 표현됩니다. (기본값)
- overlaid : 데이터 시리즈들이 카테고리 별로 이전에 표현된 데이터 시리즈 위에 덮어씌워진 형태로 표현됩니다. 따라서 제일 마지막에 표현되는 데이터 시리즈가 제일 전면에 표현됩니다.
- stacked : 데이터 시리즈들이 카테고리 별로 이전에 표현된 데이터 시리즈 위에 스택 형태로 표현됩니다. 따라서 제일 마지막에 표현되는 데이터 시리즈가 최상위 스택에 표현됩니다.
- 100% : 데이터 시리즈들이 카테고리 별로 100% 스택 형태로 표현됩니다. 전체 데이터 시리즈의 합에 대해서 표현되는 데이터 시리즈가 차지하는 상대적 비율만큼 컬럼에서 차지하는 크기가 할당됩니다.

다음은 컬럼 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DChart showDataTips="true">
  ...
  <series>
    <Column2DSeries yField="GDP" displayName="GDP Growth (In %)" />
  </series>
</Column2DChart>
```



See the CodePen 알메이트 차트 - 컬럼 차트

클러스터 컬럼 차트

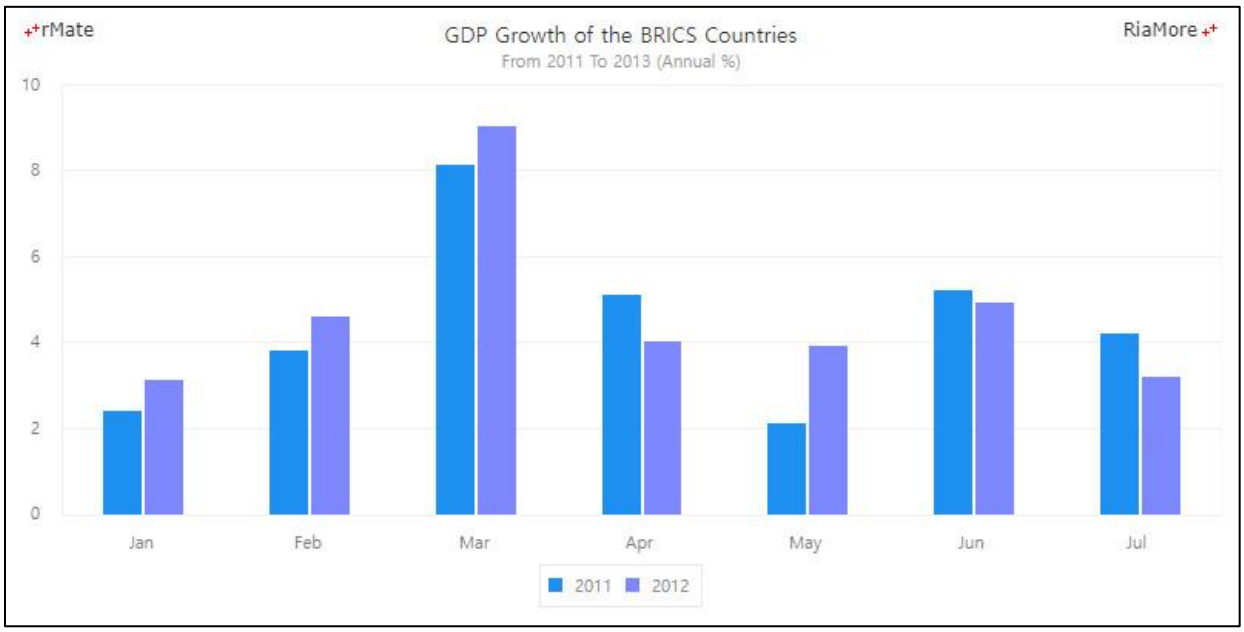
클러스터 타입의 컬럼 차트는 컬럼들이 카테고리 별로 그루핑되어 있어 여러 개의 데이터 시리즈를 직관적으로 비교할 수 있는 장점을 제공하지만 데이터 시리즈가 많아지면 시각적 복잡성을 증가시키는 단점이 있습니다. 클러스터 타입의 컬럼 차트는 <Column2DChart> 노드의 type 속성을 “clustered” 로 설정하여 생성합니다.

주의

> type 속성의 기본값이 “clustered” 이므로 type 속성을 설정하지 않으면 자동으로 clustered 타입의 컬럼 차트가 생성됩니다.

다음은 <Column2DChart> 노드의 type 속성을 “clustered” 로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DChart showDataTips="true" type="clustered">
...
<series>
<Column2DSeries yField="2011" displayName="2011" />
<Column2DSeries yField="2012" displayName="2012" />
</series>
</Column2DChart>
```



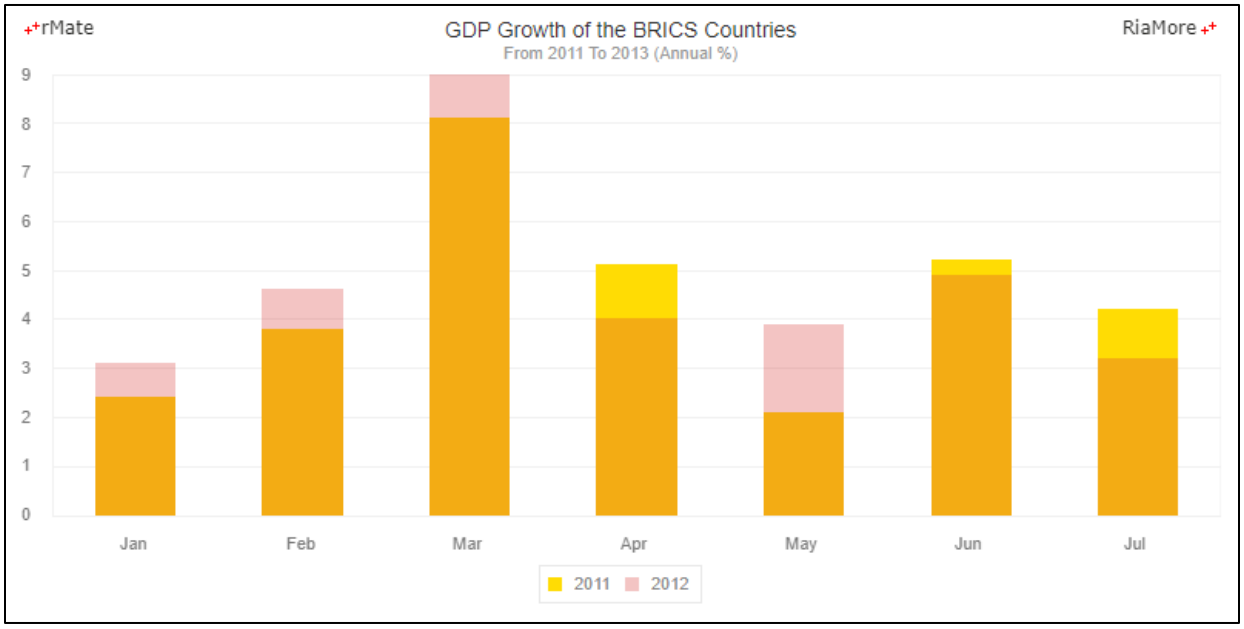
See the CodePen 알메이트 차트 - 클러스터 컬럼 차트

오버레이 컬럼 차트

오버레이 타입의 컬럼 차트는 카테고리 별로 데이터 시리즈들이 하나의 컬럼으로 겹쳐서 표현됩니다. 따라서 이전에 표현된 데이터 시리즈의 값이 이후에 표현되는 데이터 시리즈의 값보다 작을 경우 이후에 표현되는 데이터 시리즈에 완전히 가려질 수 있습니다. 이 경우에 모든 데이터 시리즈들을 표시하려면 컬럼의 투명도(<Column2DSeries> 노드에 설정되는 색상의 alpha 속성)를 설정해야 합니다. 오버레이 타입의 컬럼 차트는 <Column2DChart> 노드의 type 속성을 “overlaid”로 설정하여 생성합니다. 다음은 <Column2DChart> 노드의 type 속성을 “overlaid” 로 설정하는 코드와 이를

```
<Column2DChart showDataTips="true" type="overlaid">
  ...
  <series>
    <Column2DSeries yField="2011" displayName="2011" >
      <fill>
        <SolidColor color="#FFDC04" />
      </fill>
    </Column2DSeries>
    <Column2DSeries yField="2012" displayName="2012" >
      <fill>
        <SolidColor color="#D63A39" alpha="0.3"/>
      </fill>
    </Column2DSeries>
  </series>
</Column2DChart>
```

적용해서 출력한 차트의 예제입니다. 이 예제에서는 두 번째 데이터 시리즈의 색상을 설정할 때(yField = “2012”, <SolidColor> node 노드의 alpha 속성) alpha 속성값이 적용되었습니다.

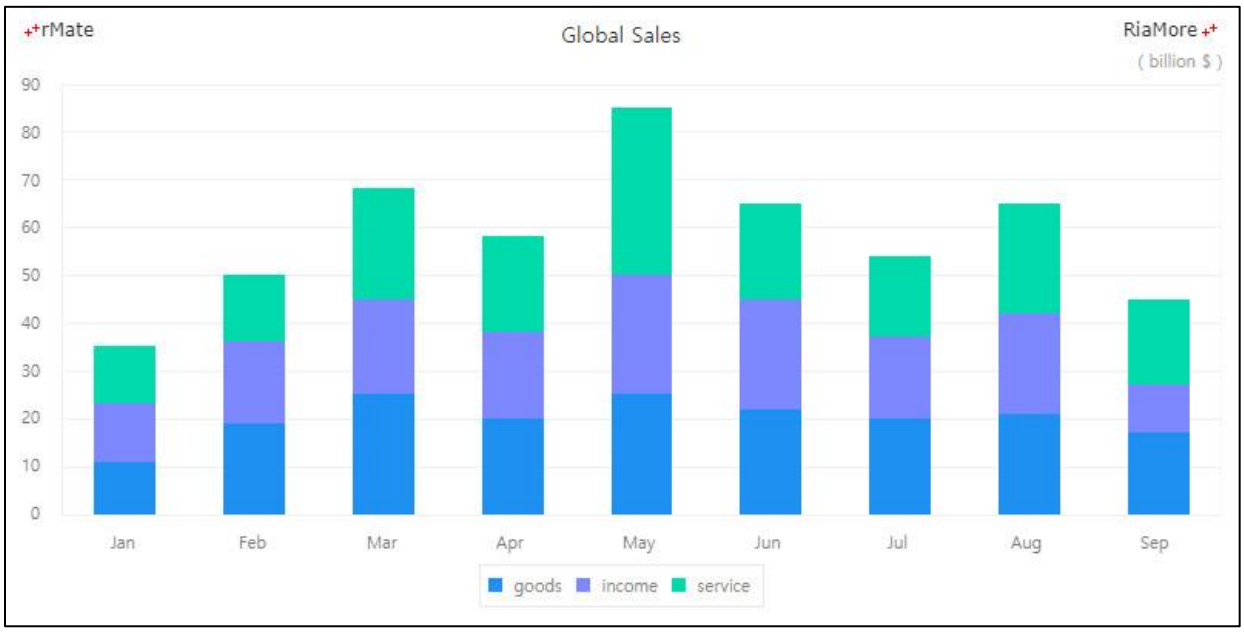


See the CodePen 알메이트 차트 - 오버레이 컬럼 차트

스택 컬럼 차트

스택 타입의 컬럼 차트는 카테고리 별로 데이터 시리즈들이 하나의 컬럼에 스택으로 쌓인 형태로 표현됩니다. 카테고리 별로 전체 데이터 시리즈의 값들을 직관적으로 비교하기에 좋은 유형의 차트입니다. 스택 타입의 컬럼 차트는 <Column2DChart> 노드의 type 속성을 “stacked” 으로 설정하여 생성합니다. 다음은 <Column2DChart> 노드의 type 속성을 “stacked” 으로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DChart showDataTips="true" type="stacked">
  ...
  <series>
    <Column2DSeries yField="goods" displayName="goods" />
    <Column2DSeries yField="income" displayName="income" />
    <Column2DSeries yField="service" displayName="service" />
  </series>
</Column2DChart>
```

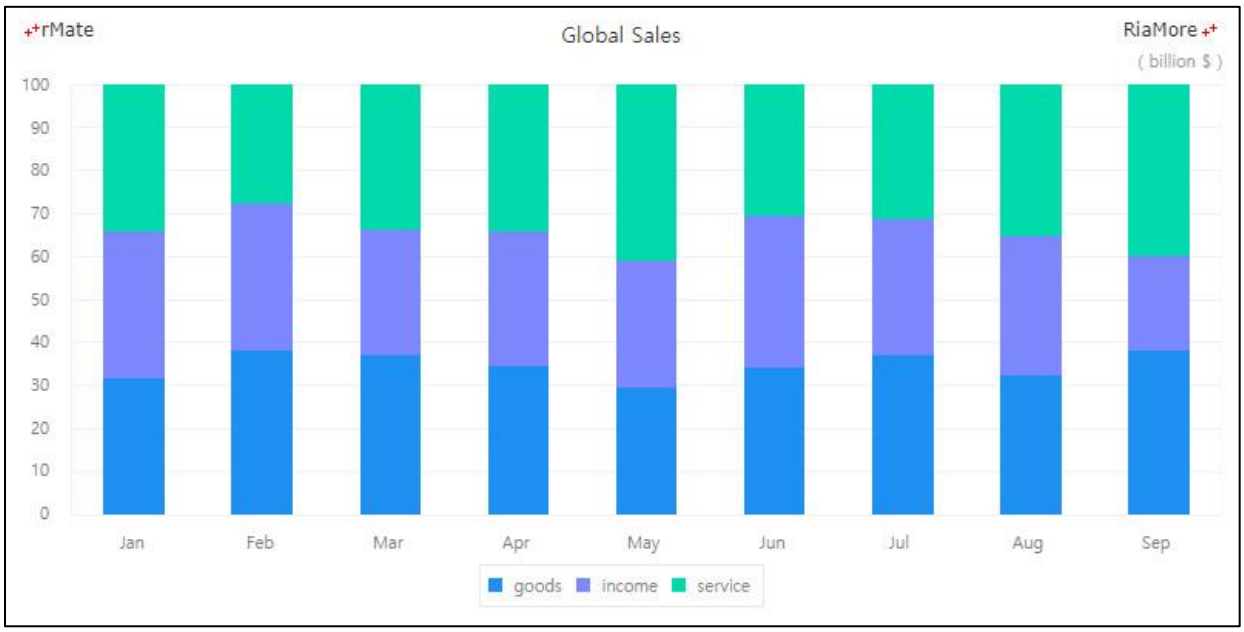


See the CodePen 알메이트 차트 - 스택 컬럼 차트

100% 컬럼 차트

100% 타입의 컬럼 차트는 카테고리 별로 데이터 시리즈들이 하나의 컬럼에 100% 스택으로 쌓인 형태로 표현됩니다. 이때 컬럼을 구성하는 각 데이터 시리즈들의 크기는 전체 데이터 시리즈 값의 합(100%)에 대한 상대적인 비율에 따라서 결정됩니다. 다음은 <Column2DChart> 노드의 type 속성을 “100%” 로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DChart showDataTips="true" type="100%">
  ...
  <series>
    <Column2DSeries yField="goods" displayName="goods" />
    <Column2DSeries yField="income" displayName="income" />
    <Column2DSeries yField="service" displayName="service" />
  </series>
</Column2DChart>
```

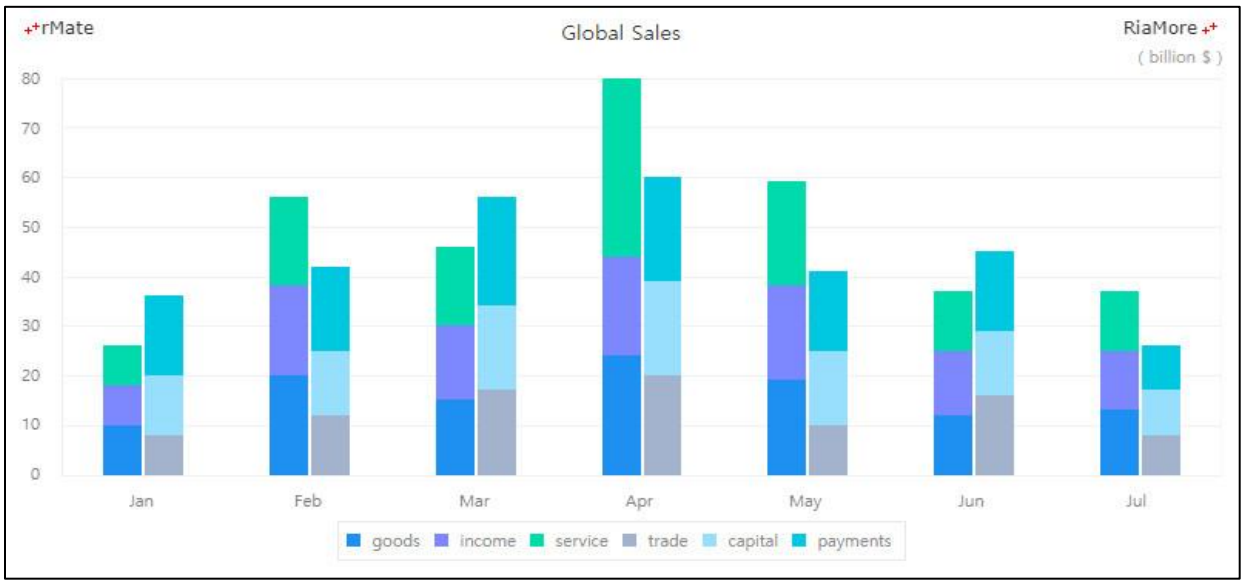


See the CodePen 알메이트 차트 - 100% 컬럼 차트

스택 다중 시리즈 컬럼 차트

스택 타입의 컬럼 여러 개를 클러스터로 표현할 수 있습니다. 이를 위해서 하나의 스택 컬럼으로 표현하고자 하는 데이터 시리즈(<Column2DSeries> 노드)들을 <Column2DSet> 노드에 설정해야 합니다. 정의된 <Column2DSet> 노드 수에 해당하는 컬럼들이 한 카테고리에 클러스터링됩니다. 다음은 세 개의 데이터 시리즈가 스택으로 표현된 두 개의 컬럼을 클러스터링하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DChart showDataTips="true">
  ...
  <series>
    <Column2DSet type="stacked">
      <series>
        <Column2DSeries yField="goods" displayName="goods" />
        <Column2DSeries yField="income" displayName="income" />
        <Column2DSeries yField="service" displayName="service" />
      </series>
    </Column2DSet>
    <Column2DSet type="stacked">
      <series>
        <Column2DSeries yField="trade" displayName="trade" />
        <Column2DSeries yField="capital" displayName="capital" />
        <Column2DSeries yField="payments" displayName="payments" />
      </series>
    </Column2DSet>
  </series>
</Column2DChart>
```

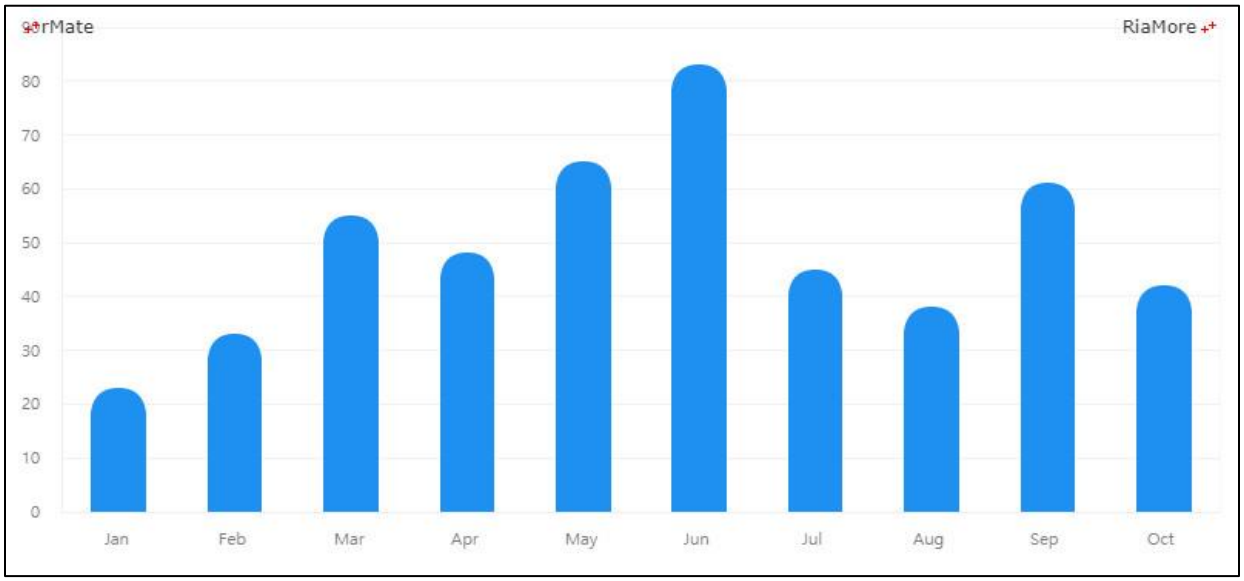


See the CodePen 알메이트 차트 - 스택 다중 시리즈 컬럼 차트

컬럼 끝을 반원 모양으로 설정

<Column2DSeries> 노드의 itemRenderer 속성을 “SemiCircleColumnItemRenderer” 로 설정하면 컬럼 끝의 모양을 반원으로 표현할 수 있습니다.

```
<Column2DSeries yField="Vancouver" displayName="Vancouver"
itemRenderer="SemiCircleColumnItemRenderer" />
```



See the CodePen 알메이트 차트 - 컬럼 끝을 반원 모양으로 설정

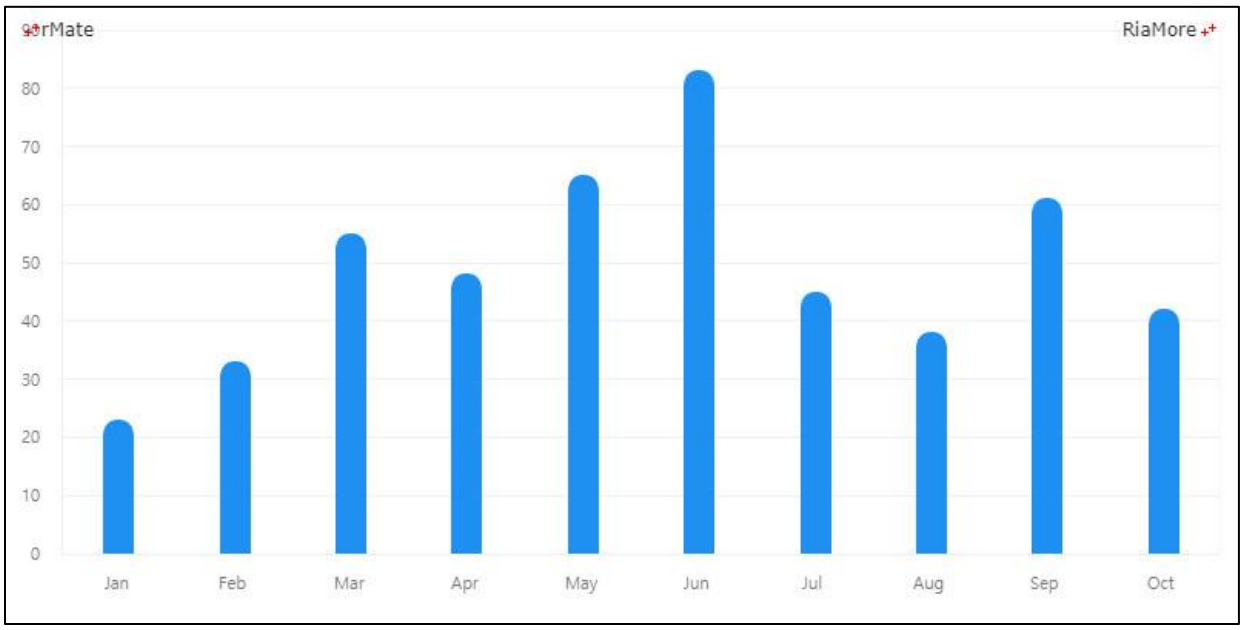
컬럼의 넓이 조절

<Column2DChart> 노드의 columnWidthRatio 속성과 maxColumnWidth 속성을 이용하여 컬럼의 넓이를 조절할 수 있습니다. 두 속성에 대한 설명은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
columnWidthRatio	0 과 1 사이의 숫자 기본값: 0.5	카테고리에 표현되는 컬럼 넓이의 비율을 지정합니다. 예를들어 값이 1 이면 사용 가능한 전체 넓이가 컬럼의 넓이로 설정되고, 값이 0.6 이면 사용 가능한 넓이의 60%가 컬럼의 넓이로 설정됩니다. maxColumnWidth 속성에 값이 설정되면 둘 중 작은 값이 적용됩니다.
maxColumnWidth	숫자 기본값: NaN	컬럼의 넓이를 픽셀값으로 지정합니다. columnWidthRatio 속성에 값이 설정되면 둘 중 작은 값이 적용됩니다.

다음은 columnWidthRatio 속성값이 “1” 이고, maxColumnWidth 속성값이 “10” 일 때 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DChart showDataTips="true" columnWidthRatio="1" maxColumnWidth="10">
  ...
</Column2DChart>
```



See the CodePen 알메이트 차트 - 컬럼의 넓이 조절

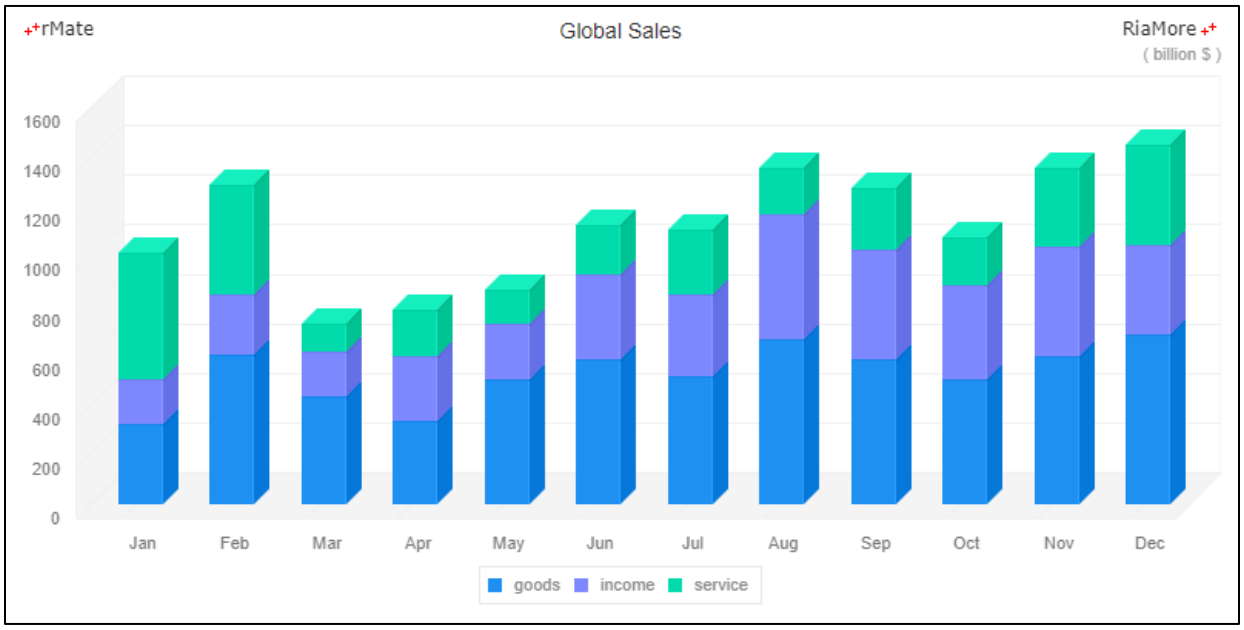
3D 컬럼 차트

<Column3DChart> 노드의 series 속성값에 <Column3Deries> 노드를 설정하면 3D 모양의 컬럼 차트를 생성할 수 있습니다. 이 때 3D 큐브의 모양은 cubeAngleRatio 속성과 cubeDepth 속성을 이용하여 조절할 수 있습니다. 두 속성에 대한 설명은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
cubeAngleRatio	숫자 기본값: 1	큐브의 각도 비율을 지정합니다. 속성값에 대한 세로 비율을 나타내고, 값이 1 이면 45 도를 의미합니다.
cubeDepth	숫자 기본값: 30	큐브의 깊이를 지정합니다.

다음은 스택 타입의 3D 컬럼 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column3DChart showDataTips="true" type="stacked">
  ...
  <series>
    <Column3DSeries yField="goods" displayName="goods" />
    <Column3DSeries yField="income" displayName="income" />
    <Column3DSeries yField="service" displayName="service" />
  </series>
</Column3DChart>
```

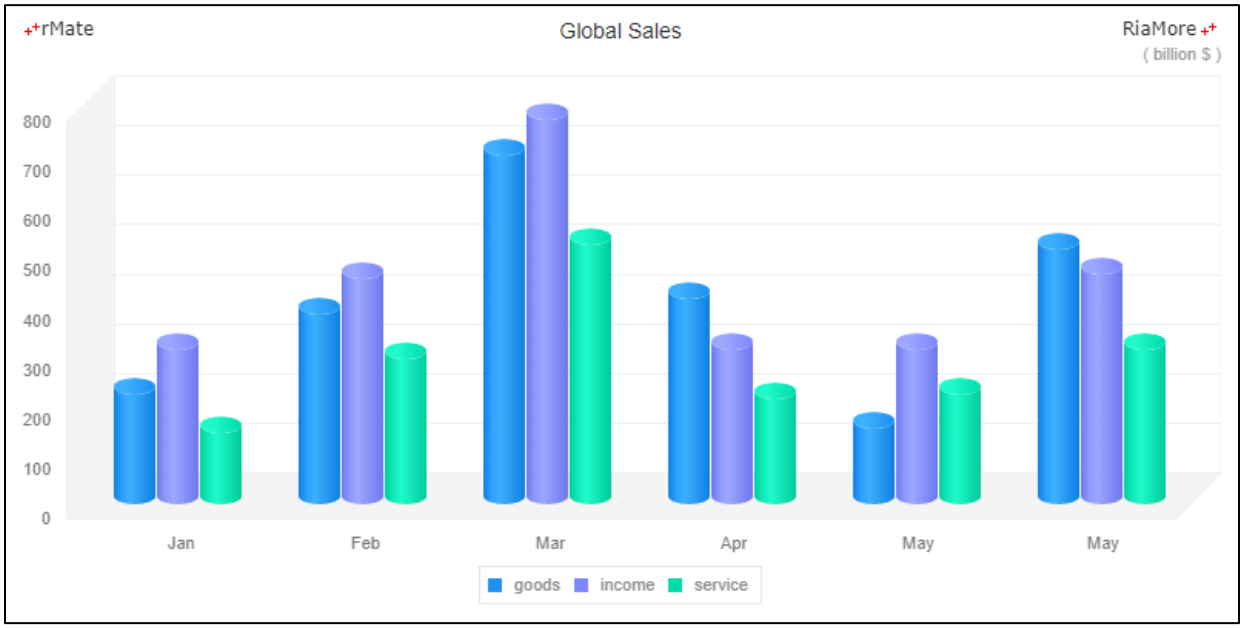


See the CodePen 알메이트 차트 - 3D 컬럼 차트

3D 실린더 (원통) 컬럼 차트

<Column3DSeries> 노드의 itemRenderer 속성을 “CylinderItemRenderer” 로 설정하면 3D 실린더 모양의 컬럼 차트를 생성할 수 있습니다. 다음은 itemRenderer 속성을 “CylinderItemRenderer” 로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column3DChart showDataTips="true" cubeAngleRatio="1" columnWidthRatio="0.7">
  ...
  <series>
    <Column3DSeries yField="goods" displayName="goods"
        itemRenderer="CylinderItemRenderer" />
    <Column3DSeries yField="income" displayName="income"
        itemRenderer="CylinderItemRenderer" />
    <Column3DSeries yField="service" displayName="service"
        itemRenderer="CylinderItemRenderer" />
  </series>
</Column3DChart>
```



See the CodePen 알메이트 차트 - 3D 실린더 (원통) 컬럼 차트

4.3 바 차트

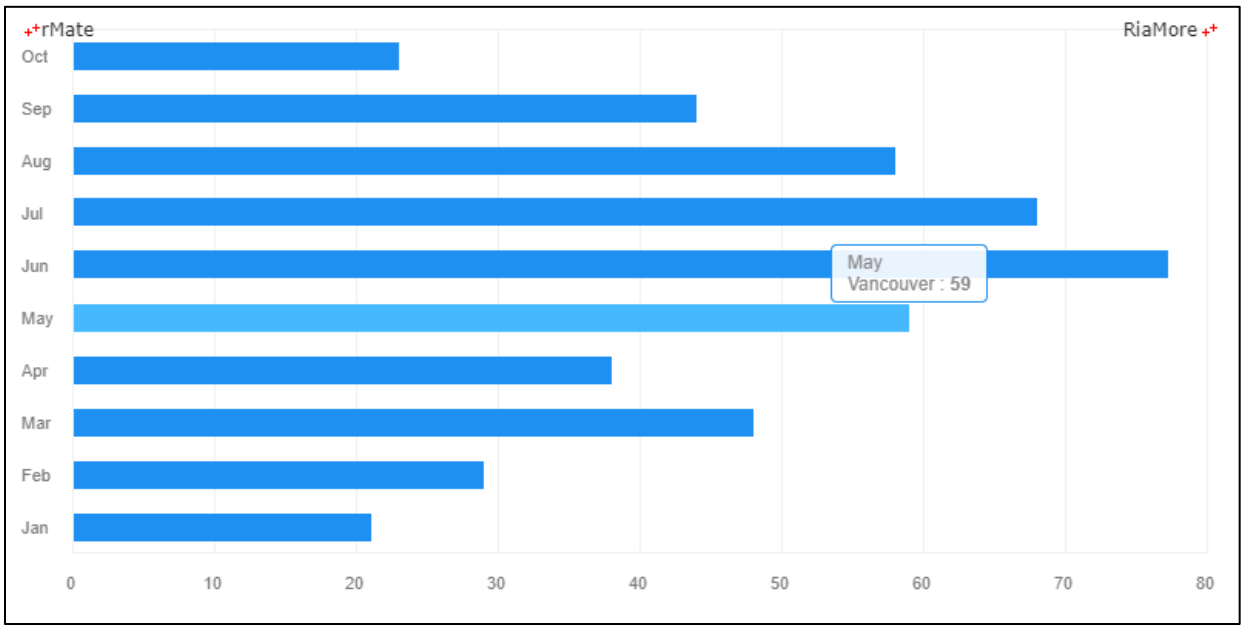
바 차트는 데이터의 크기를 가로 막대의 길이로 표현하는 차트입니다. 일반적으로 숫자 축은 차트의 하단에 가로 축(X 축)으로 표시되고 카테고리 축은 차트의 왼쪽에 세로 축(Y 축)으로 표시됩니다. 바 차트는 <Bar2DChart> 노드의 series 속성값에 <Bar2DSeries> 노드를 설정하여 생성할 수 있습니다.

바 차트의 데이터 시리즈가 여러 개일 경우 (<Bar2DSeries> 노드가 여러 개 설정 됨), 데이터 시리즈들이 표현되는 방식은 <Bar2DChart> 노드의 type 속성의 설정에 따릅니다. 다음에는 type 속성에 설정 가능한 값과 이에 따른 표현 방식이 설명되어 있습니다.

- clustered : 데이터 시리즈들이 카테고리 별로 클러스터링되어 표현됩니다. (기본값)
- overlaid : 데이터 시리즈들이 카테고리 별로 이전에 표현된 데이터 시리즈 위에 덮어씌워진 형태로 표현됩니다. 따라서 제일 마지막에 표현되는 데이터 시리즈가 제일 전면에 표현됩니다.
- stacked : 데이터 시리즈들이 카테고리 별로 이전에 표현된 데이터 시리즈 위에 스택 형태로 표현됩니다. 따라서 제일 마지막에 표현되는 데이터 시리즈가 최상위 스택에 표현됩니다.
- 100% : 데이터 시리즈들이 카테고리 별로 100% 스택 형태로 표현됩니다. 전체 데이터 시리즈의 합에 대해서 표현되는 데이터 시리즈가 차지하는 상대적 비율만큼 바에서 차지하는 크기가 할당됩니다.

다음은 바 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DChart showDataTips="true">
  <series>
    <Bar2DSeries xField="Vancouver" displayName="Vancouver" />
  </series>
</Bar2DChart>
```



See the CodePen 알메이트 차트 - 바 차트

클러스터 바 차트

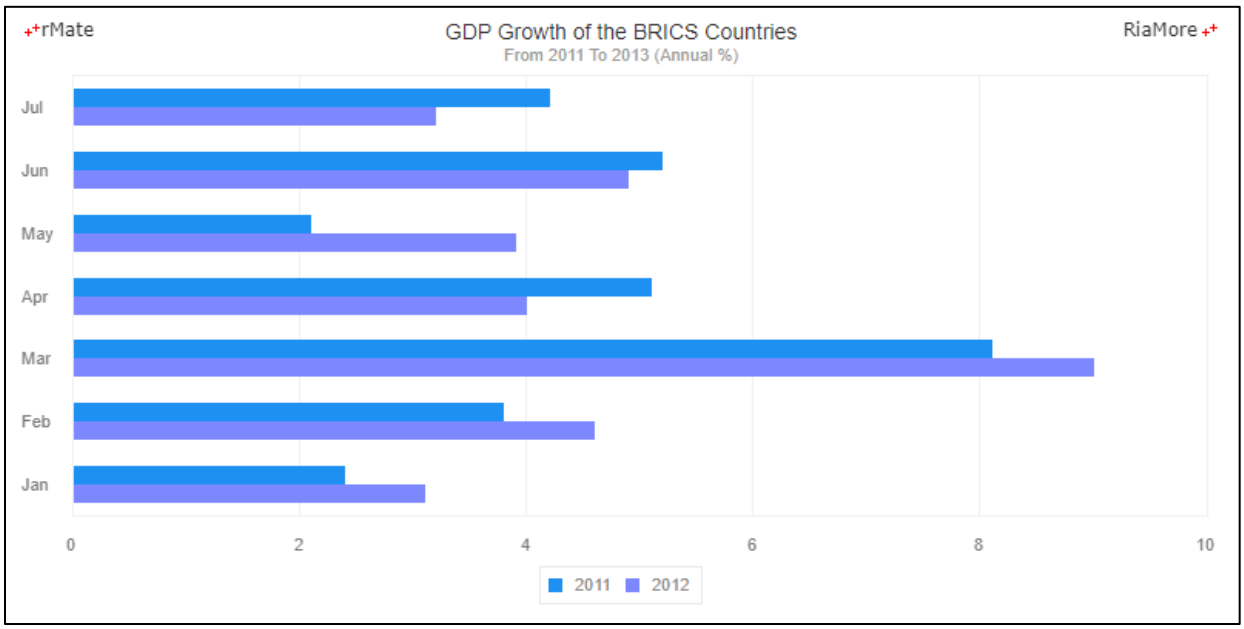
클러스터 타입의 바 차트는 바들이 카테고리 별로 그루핑되어 있어 여러 개의 데이터 시리즈를 직관적으로 비교할 수 있는 장점을 제공하지만 데이터 시리즈가 많아지면 시각적 복잡성을 증가시키는 단점이 있습니다 클러스터 타입의 바 차트는 <Bar2DChart> 노드의 type 속성을 “clustered” 로 설정하여 생성합니다.

주의

> type 속성의 기본값이 “clustered” 이므로 type 속성을 설정하지 않으면 자동으로 clustered 타입의 바 차트가 생성됩니다.

다음은 <Bar2DChart> 노드의 type 속성을 “clustered” 으로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DChart showDataTips="true" type="clustered">
  ...
  <series>
    <Bar2DSeries xField="2011" displayName="2011" />
    <Bar2DSeries xField="2012" displayName="2012" />
  </series>
</Bar2DChart>
```



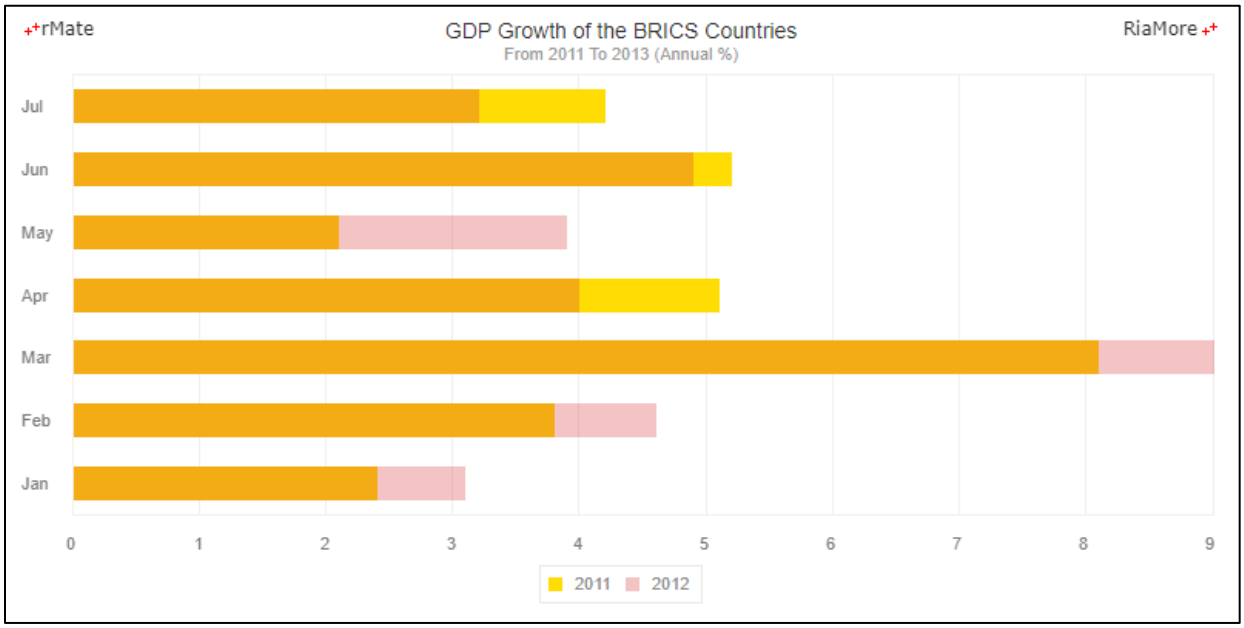
See the CodePen 알메이트 차트 - 클러스터 바 차트

오버레이 바 차트

오버레이 타입의 바 차트는 카테고리 별로 데이터 시리즈들이 하나의 바로 겹쳐서 표현됩니다. 따라서 이전에 표현된 데이터 시리즈의 값이 이후에 표현되는 데이터 시리즈의 값보다 작을 경우 이후에 표현되는 데이터 시리즈에 완전히 가려질 수 있습니다. 이 경우에 모든 데이터 시리즈들을 표시하려면 바의 투명도 (<Bar2DSeries> 노드에 설정되는 색상의 alpha 속성)를 설정해야 합니다. 오버레이 타입의 바 차트는 <Bar2DChart> 노드의 type 속성을 “overlaid” 로 설정하여 생성합니다.

다음은 <Bar2DChart> 노드의 type 속성을 “overlaid” 로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 두 번째 데이터 시리즈의 색상을 설정할 때 (yField=“2012”, <SolidColor> 노드의 alpha 속성), alpha 속성값이 적용되었습니다.

```
<Bar2DChart showDataTips="true" type="overlaid">
  ...
  <series>
    <Bar2DSeries xField="2011" displayName="2011" >
      <fill>
        <SolidColor color="#FFDC04" />
      </fill>
    </Bar2DSeries>
    <Bar2DSeries xField="2012" displayName="2012" >
      <fill>
        <SolidColor color="#D63A39" alpha="0.3"/>
      </fill>
    </Bar2DSeries>
  </series>
</Bar2DChart>
```

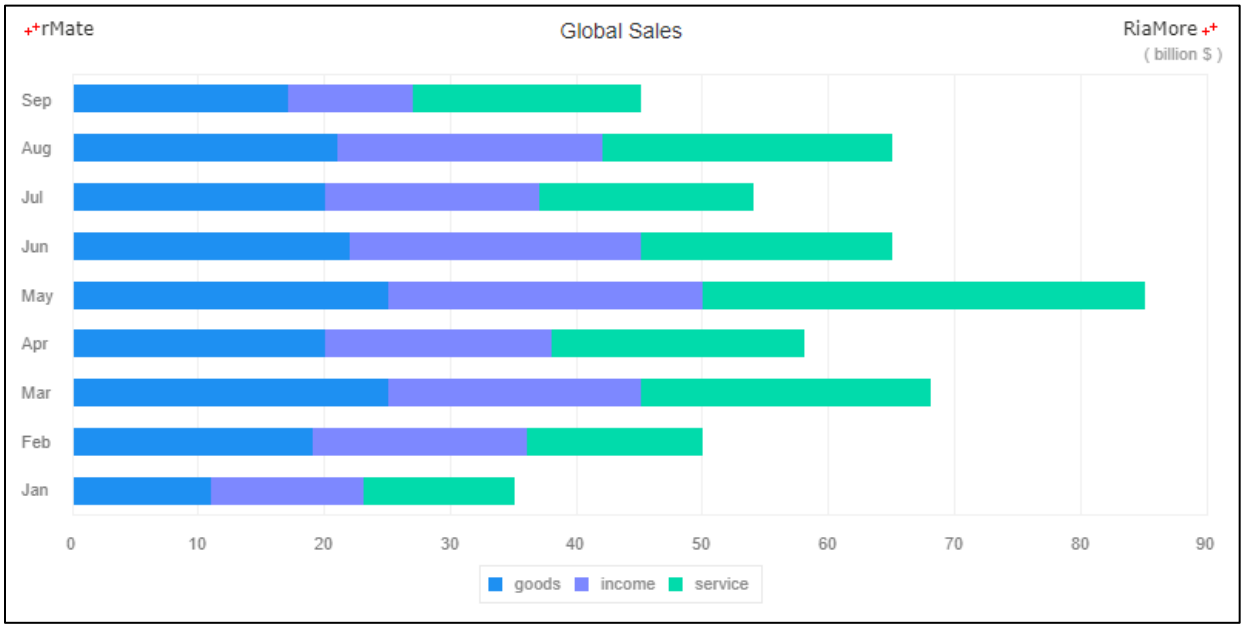


See the CodePen 알메이트 차트 - 오버레이 바 차트

스택 바 차트

스택 타입의 바 차트는 카테고리 별로 데이터 시리즈들이 하나의 바에 스택으로 쌓인 형태로 표현됩니다. 카테고리 별로 전체 데이터 시리즈의 값들을 직관적으로 비교하기에 좋은 유형의 차트입니다. 스택 타입의 바 차트는 <Bar2DChart> 노드의 type 속성을 “stacked” 으로 설정하여 생성합니다. 다음은 <Bar2DChart> 노드의 type 속성을 “stacked” 으로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DChart showDataTips="true" type="stacked">
  ...
  <series>
    <Bar2DSeries xField="goods" displayName="goods" />
    <Bar2DSeries xField="income" displayName="income" />
    <Bar2DSeries xField="service" displayName="service" />
  </series>
</Bar2DChart>
```

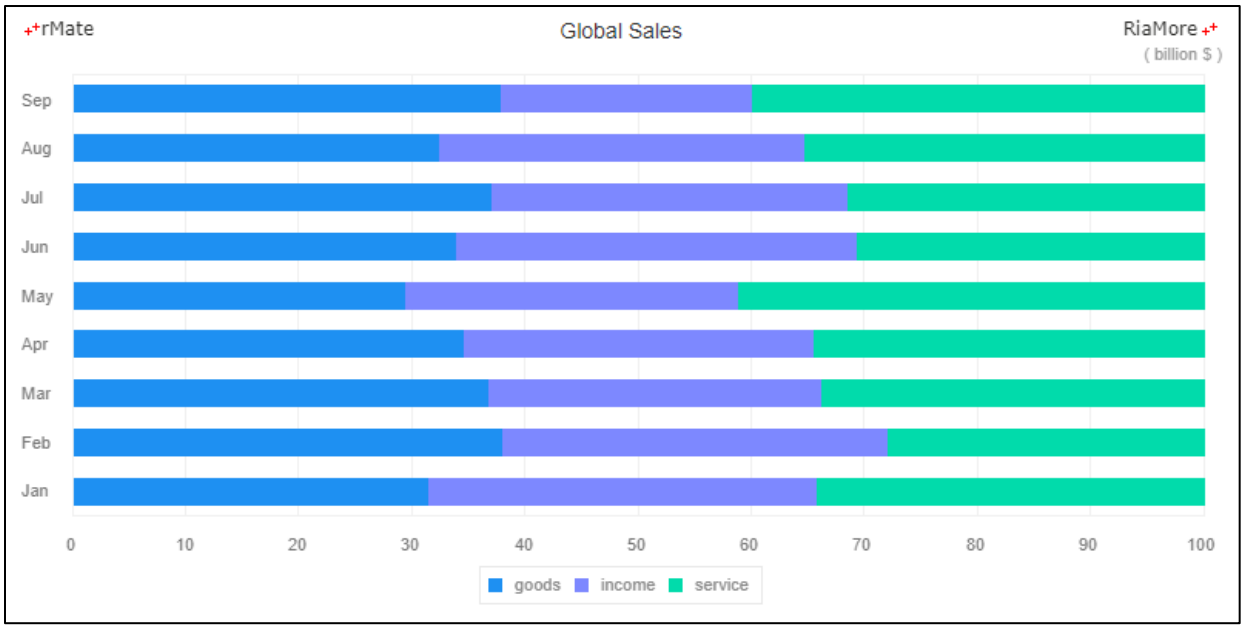


See the CodePen 알메이트 차트 - 스택 바 차트

100% 바 차트

100% 타입의 바 차트는 카테고리 별로 데이터 시리즈들이 하나의 바에 100% 스택으로 쌓인 형태로 표현됩니다. 이 때 바를 구성하는 각 데이터 시리즈들의 크기는 전체 데이터 시리즈 값의 합(100%)에 대한 상대적인 비율에 따라서 결정됩니다. 다음은 <Bar2DChart> 노드의 type 속성을 “100%” 로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DChart showDataTips="true" type="100%">
  ...
  <series>
    <Bar2DSeries xField="goods" displayName="goods" />
    <Bar2DSeries xField="income" displayName="income" />
    <Bar2DSeries xField="service" displayName="service" />
  </series>
</Bar2DChart>
```



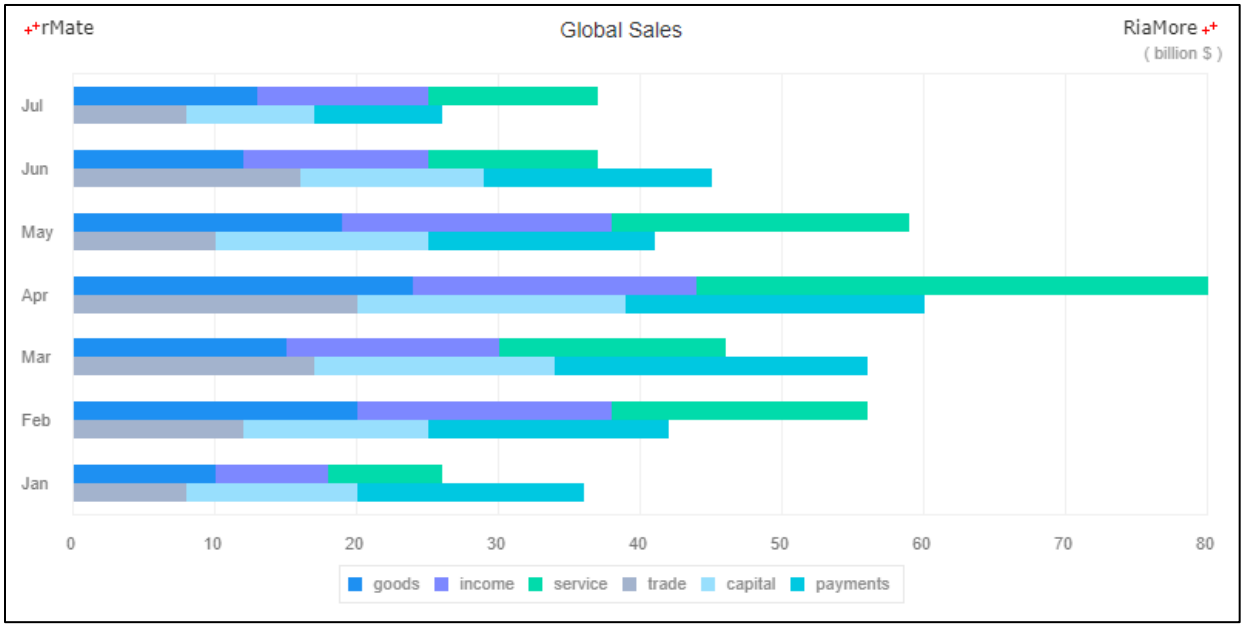
See the CodePen 알메이트 차트 - 100% 바 차트

스택 다중 시리즈 바 차트

스택 타입의 바 여러 개를 클러스터로 표현할 수 있습니다. 이를 위해서 하나의 스택 바로 표현하고자 하는 데이터 시리즈 (<Bar2DSeries> 노드)들을 <Bar2DSet> 노드에 설정해야 합니다.
정의된 <Bar2DSet> 노드 수에 해당하는 바들이 한 카테고리에 클러스터링됩니다. 다음은 세 개의

```
<Bar2DChart showDataTips="true">
  ...
  <series>
    <Bar2DSet type="stacked">
        ...
    </Bar2DSet>
    <Bar2DSet type="stacked">
        ...
    </Bar2DSet>
  </series>
</Bar2DChart>
```

데이터 시리즈가 스택으로 표현된 두 개의 바를 클러스터링하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.



See the CodePen 알메이트 차트 - 스택 다중 시리즈 바 차트

바 넓이 조절

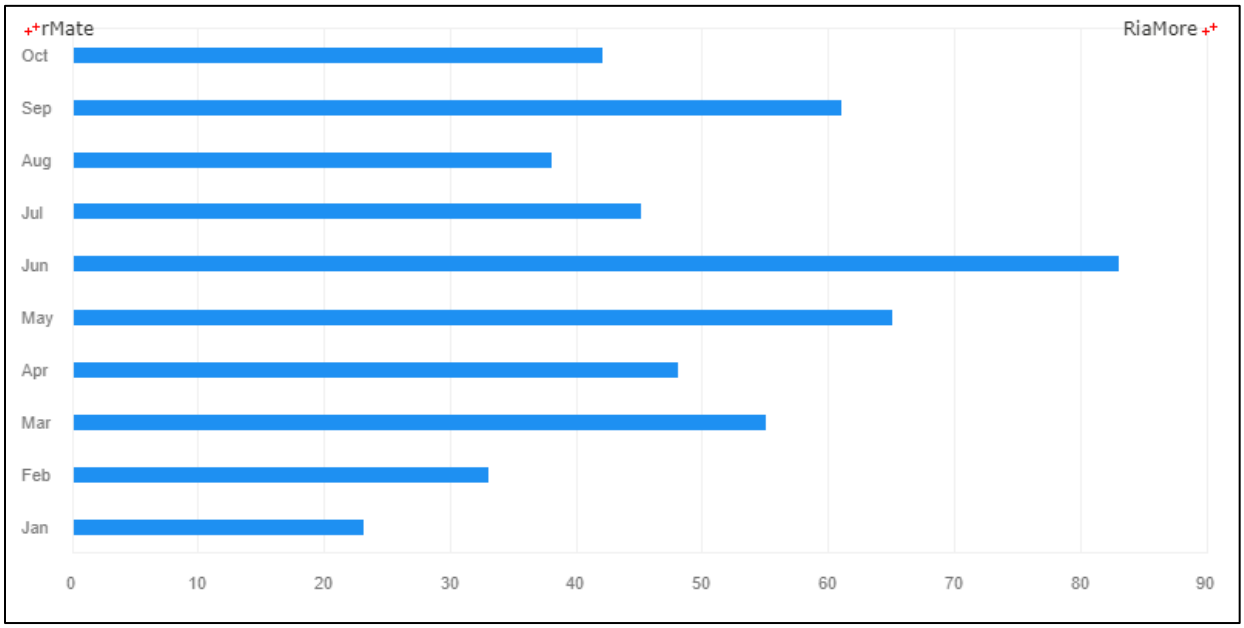
<Bar2DChart> 노드의 barWidthRatio 속성과 maxBarWidth 속성을 이용하여 바의 넓이를 조절할 수 있습니다. 두 속성에 대한 설명은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
barWidthRatio	0 과 1 사이의 숫자 기본값: 0.65	카테고리에 표현되는 바 넓이의 비율을 지정합니다. 예를 들어 값이 1 이면 사용 가능한 전체 넓이가 바의 넓이로 설정되고, 값이 0.6 이면 사용 가능한 넓이의 60% 가 바의 넓이로 설정됩니다. maxBarWidth 속성에 값이 설정되면 둘 중 작은 값이 적용됩니다.
maxBarWidth	숫자	바의 넓이를 픽셀값으로 지정합니다.

	기본값: NaN	barWidthRatio 속성에 값이 설정되면 둘 중 작은 값이 적용됩니다.

다음은 barWidthRatio 속성이 “1” 이고, maxBarWidth 속성이 “5” 일 때 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DChart showDataTips="true" barWidthRatio="1" maxBarWidth="5">
  ...
</Bar2DChart>
```



See the CodePen 알메이트 차트 - 바 넓이 조절

3D 바 차트

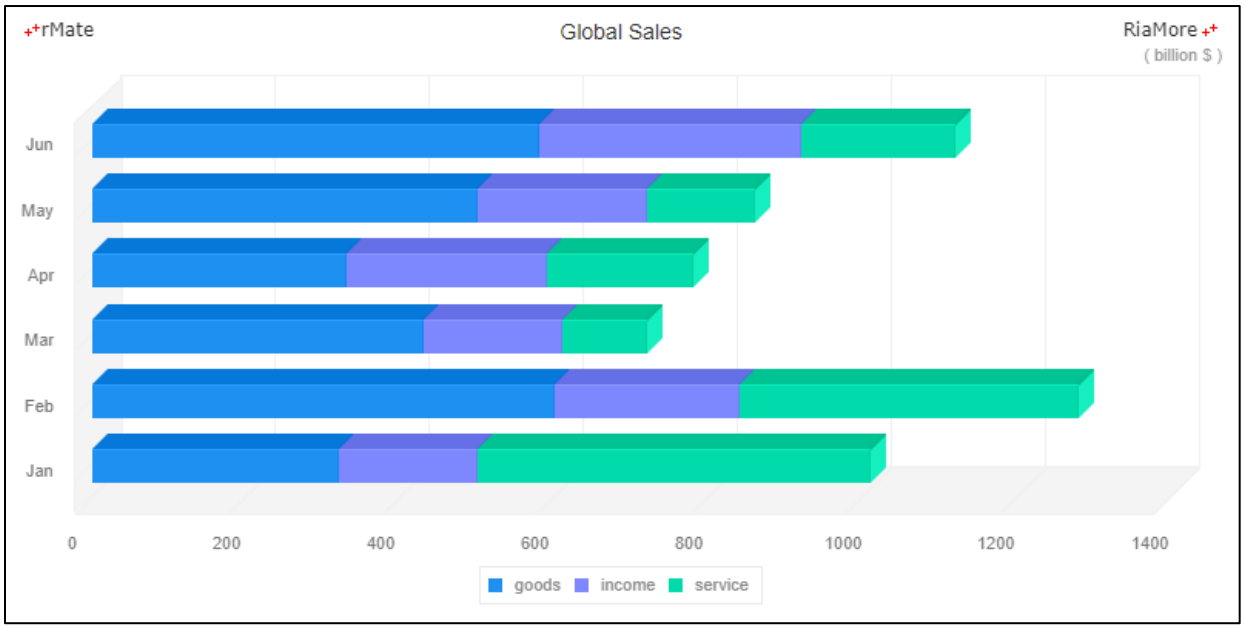
<Bar3DChart> 노드의 series 속성값에 <Bar3Deries> 노드를 설정하면 3D 모양의 바 차트를 생성할 수 있습니다. 이 때 3D 큐브의 모양은 cubeAngleRatio 속성과 cubeDepth 속성을 이용하여 조절할 수 있습니다. 두 속성에 대한 설명은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
cubeAngleRatio	숫자	큐브의 각도 비율을 지정합니다.

	기본값: 1	cubeDepth 속성값에 대한 세로 비율을 나타내고, 값이 1 이면 45 도를 의미합니다.
cubeDepth	숫자 기본값: 30	큐브의 깊이를 지정합니다.

다음은 스택 타입의 3D 바 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar3DChart showDataTips="true" type="stacked">
  ...
  <series>
    <Bar3DSeries xField="goods" displayName="goods" />
    <Bar3DSeries xField="income" displayName="income" />
    <Bar3DSeries xField="service" displayName="service" />
  </series>
</Bar3DChart>
```

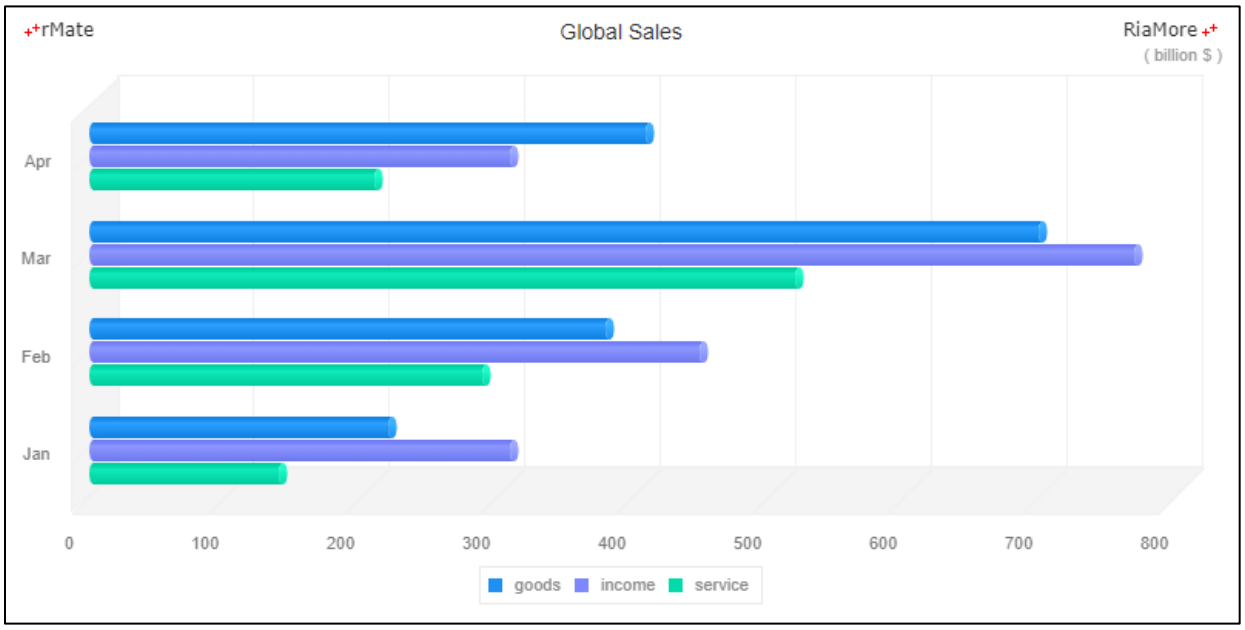


See the CodePen 알메이트 차트 - 3D 바 차트

실린더 (원통) 바 차트

<Bar3DSeries> 노드의 itemRenderer 속성을 “BarCylinderItemRenderer” 로 설정하면 3D 실린더 모양의 바 차트를 생성할 수 있습니다. 다음은 itemRenderer 속성을 “BarCylinderItemRenderer” 로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar3DChart showDataTips="true" cubeAngleRatio="1" barWidthRatio="0.7">
  ...
  <series>
    <Bar3DSeries xField="goods" displayName="goods"
       itemRenderer="BarCylinderItemRenderer" />
    <Bar3DSeries xField="income" displayName="income"
       itemRenderer="BarCylinderItemRenderer" />
    <Bar3DSeries xField="service" displayName="service"
       itemRenderer="BarCylinderItemRenderer" />
  </series>
</Bar3DChart>
```

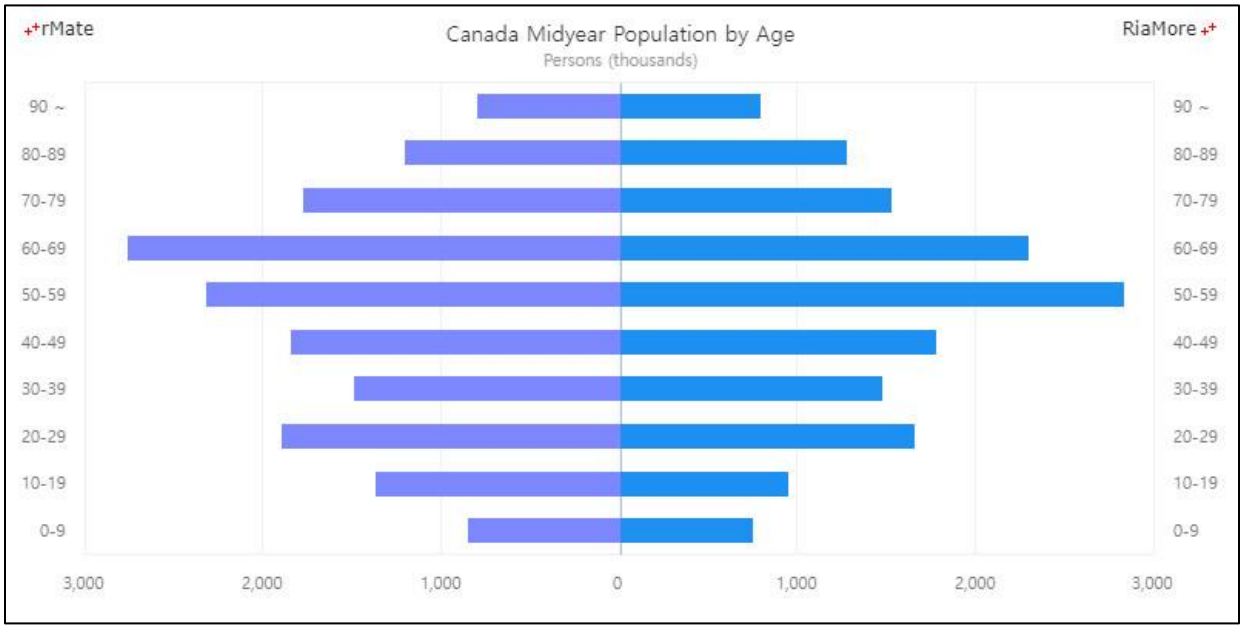


See the CodePen 알메이트 차트 - 3D 실린더 (원통) 바 차트

바 차트에서 음수값 적용

일반적으로 바 차트는 왼쪽에 위치한 카테고리 축에서 시작하여 값이 커질 수록 오른쪽으로 바의 길이가 길어지는 형태로 표현됩니다. 만약 시리즈의 값이 모두 음수이면 카테고리 축은 오른쪽에 생기고 음수의 절대값이 커질 수록 왼쪽으로 바가 길어지는 형태로 표현됩니다. 다음은 남자(Man) 데이터 필드의 값은 모두 양수인 데이터 시리즈 (xField = “Man”) 와 여자(Woman) 데이터 필드의 값은 모두 음수인 데이터 시리즈 (xField = “Woman”) 를 바 차트로 표현하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DChart showDataTips="true" type="overlaid" dataTipJsFunction="dataTipFunc">
  ...
  <series>
    <Bar2DSeries xField="Man" displayName="Man" />
    <Bar2DSeries xField="Woman" displayName="Woman" />
  </series>
</Bar2DChart>

var chartData = [
  {"age" : "0-9", "Man" : 749.8, "Woman" : -855.1},
  {"age" : "10-19", "Man" : 949.2, "Woman" : -1367},
  {"age" : "20-29", "Man" : 1655.9, "Woman" : -1900},
  ...
];
```

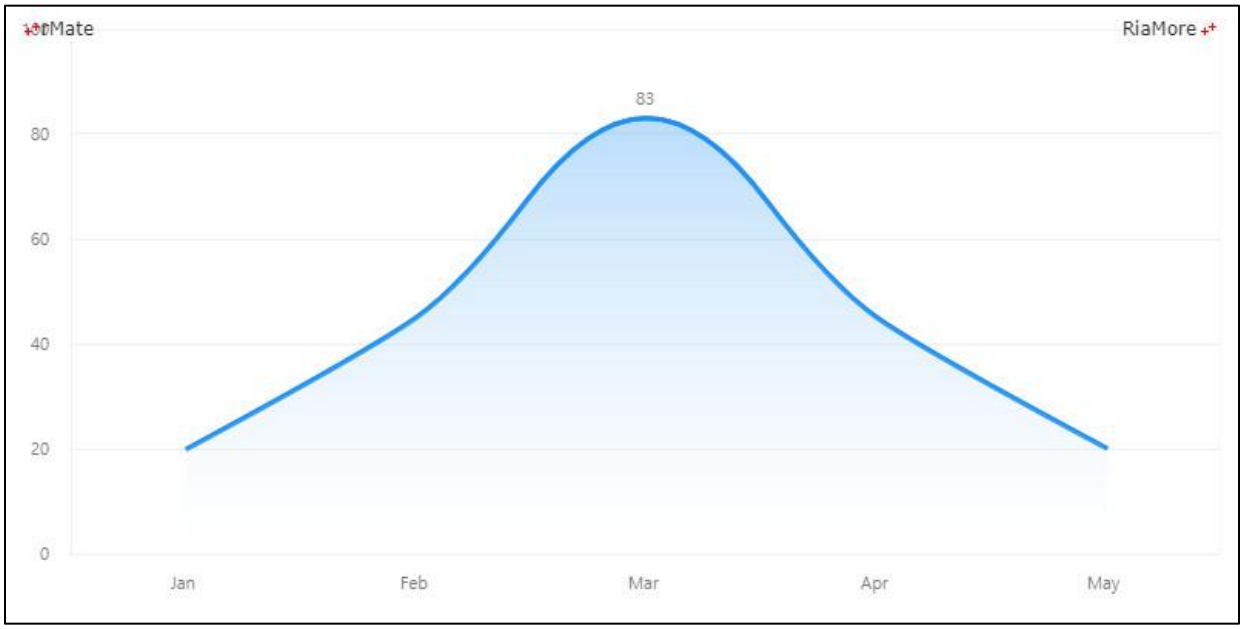


See the CodePen 알메이트 차트 - 바 차트에서 음수값 적용

4.4 영역 차트

영역 차트는 라인 차트에 기반한 차트입니다. 영역 차트가 라인 차트와 다른 점은 일반적으로 선과 축 사이의 영역이 색 혹은 패턴으로 채워져서 강조하는 효과를 사용자에게 제공한다는 것입니다. 영역 차트는 <Area2DChart> 노드의 series 속성값에 <Area2DSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 영역 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Area2DChart showDataTips="true" dataTipDisplayMode="axis">
...
  <Area2DSeries labelPosition="up" yField="Vancouver" showValueLabels="[2]"
       form="curve">
...
</Area2DChart>
```



See the CodePen 알메이트 차트 - 영역 차트

차트에 그려지는 영역이 여러 개일 경우 (<Area2DSeries> 노드가 여러 개 설정 됨), 영역들이 차트에 표현되는 방식은 <Area2DChart> 노드의 type 속성의 설정에 따릅니다. 다음에는 type 속성에 설정 가능한 값과 이에 따른 표현 방식이 설명되어 있습니다.

- overlaid : 영역은 이전에 표현된 영역들 위에 덮어씌워진 형태로 표현됩니다. 따라서 제일 마지막에 표현되는 영역이 제일 전면에 표현됩니다. (기본값)

- stacked : 영역은 이전에 표현된 영역들 위에 스택 형태로 표현됩니다. 따라서 제일 마지막에 표현되는 영역이 최상위 스택에 표현됩니다.
- 100% : 영역은 100% 스택 형태로 표현됩니다. 전체 영역의 합에 대해서 표현되는 영역이 차지하는 상대적 비율만큼 해당 영역에 할당됩니다.

차트에 그려지는 선의 형태는 <Area2DSeries> 노드의 form 속성을 통해서 설정할 수 있습니다. 다음에는 form 속성에 설정 가능한 값과 이에 따른 표현 방식이 설명되어 있습니다.

- segment : 각 데이터 포인트를 직선으로 연결합니다. (기본값)
- curve : 각 데이터 포인트를 곡선으로 연결합니다.
- step : 수평 직선으로 시작해서 각 데이터 포인트를 계단선으로 연결합니다.
- reverseStep : 수직 직선으로 시작해서 각 데이터 포인트를 계단선으로 연결합니다.

오버레이 영역 차트

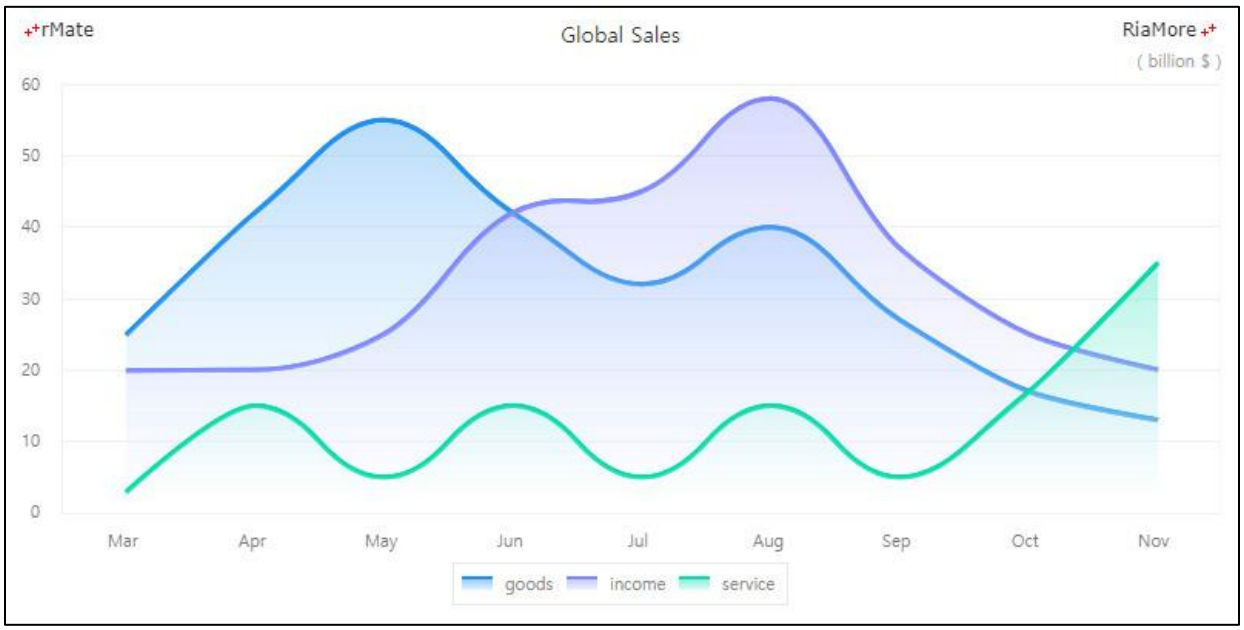
오버레이 타입의 영역 차트는 <Area2DChart> 노드의 type 속성을 “overlaid” 로 설정하여 생성합니다.

주의

> type 속성의 기본값이 “overlaid” 이므로 type 속성을 설정하지 않으면 자동으로 overlaid 타입의 영역 차트가 생성됩니다.

다음은 <Area2DChart> 노드의 type 속성을 “overlaid” 로, <Area2DSeries> 노드의 form 속성을 “curve” 로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Area2DChart type="overlaid" showDataTips="true" dataTipDisplayMode="axis">
  ...
  <series>
    <Area2DSeries yField="goods" form="curve" displayName="goods" />
    <Area2DSeries yField="income" form="curve" displayName="income" />
    <Area2DSeries yField="service" form="curve" displayName="service" />
  </series>
</Area2DChart>
```



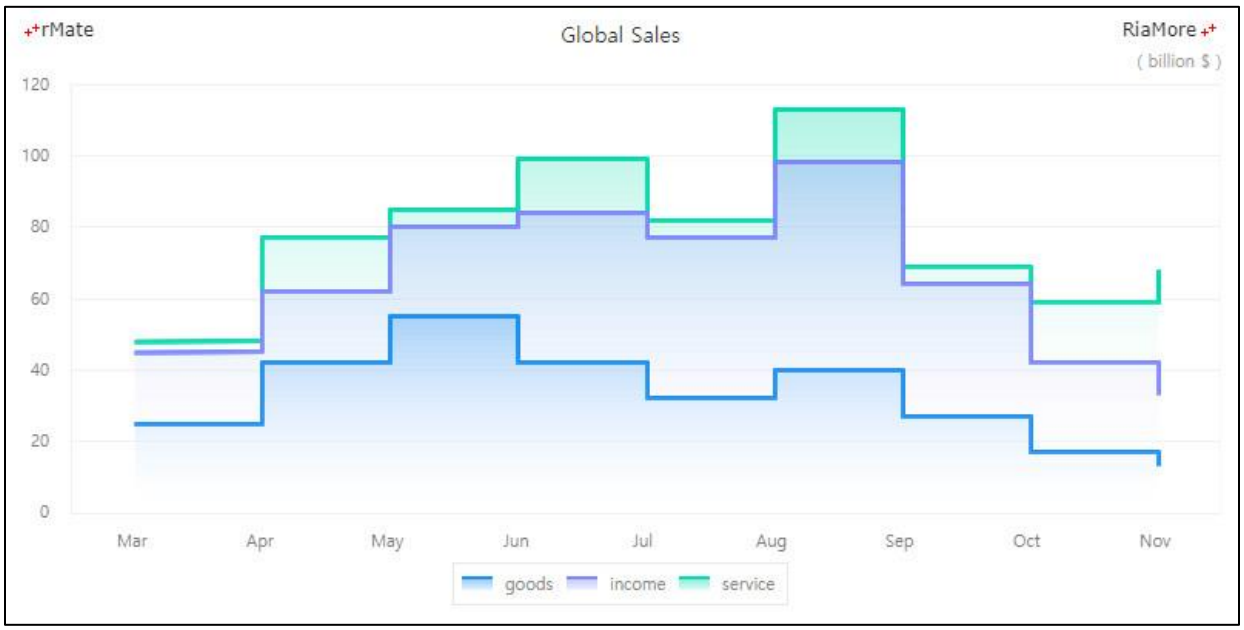
See the CodePen 알메이트 차트 - 오버레이 영역 차트

스택 영역 차트

스택 타입의 영역 차트는 <Area2DChart> 노드의 type 속성을 “stacked” 으로 설정하여 생성합니다.

다음은 <Area2DChart> 노드의 type 속성을 “stacked” 으로, <Area2DSeries> 노드의 form 속성을 “step” 으로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Area2DChart type="stacked" showDataTips="true" dataTipDisplayMode="axis">
  ...
  <series>
    <Area2DSeries yField="goods" form="step" displayName="goods" />
    <Area2DSeries yField="income" form="step" displayName="income" />
    <Area2DSeries yField="service" form="step" displayName="service" />
  </series>
</Area2DChart>
```



See the CodePen 알메이트 차트 - 스택 영역 차트

100% 영역 차트

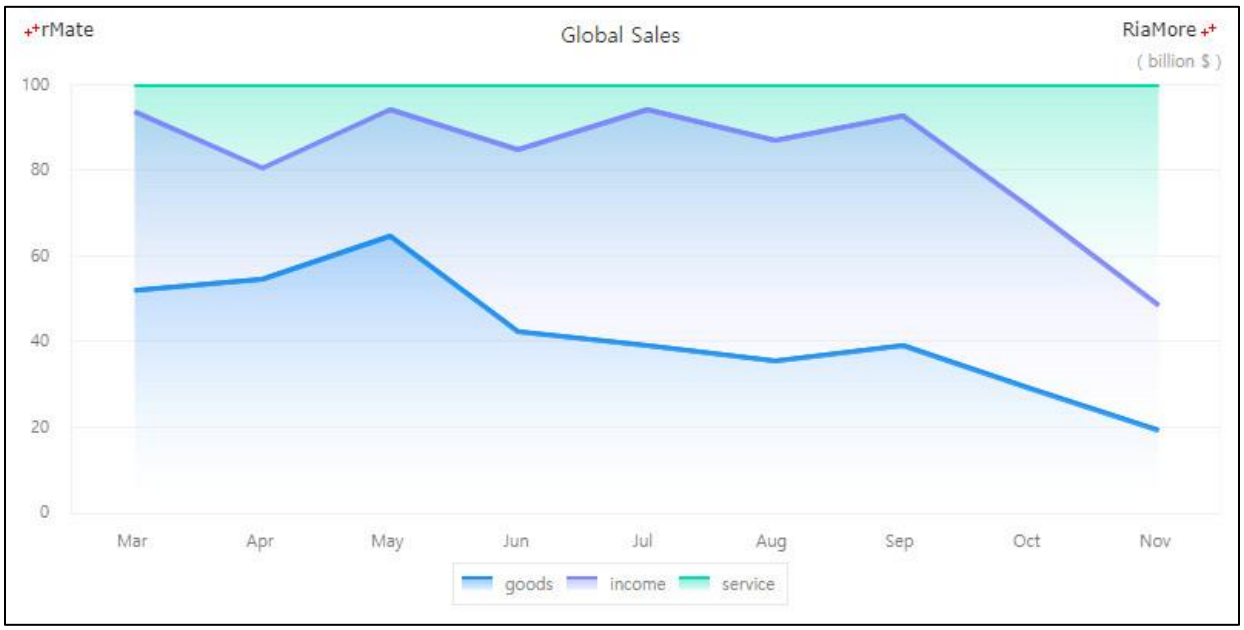
100% 타입의 영역 차트는 <Area2DChart> 노드의 type 속성을 “100%” 로 설정하여 생성합니다.

다음은 <Area2DChart> 노드의 type 속성을 “100%” 로, <Area2DSeries> 노드의 form 속성을 “segment” 로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

주의

form 속성의 기본값이 “segment” 이므로, form 속성에 값을 지정하지 않으면 자동으로 segment 유형의 선이 표현됩니다.

```
<Area2DChart type="100%" showDataTips="true" dataTipDisplayMode="axis">
  ...
  <series>
    <Area2DSeries yField="goods" form="segment" displayName="goods" />
    <Area2DSeries yField="income" form="segment" displayName="income" />
    <Area2DSeries yField="service" form="segment" displayName="service" />
  </series>
</Area2DChart>
```



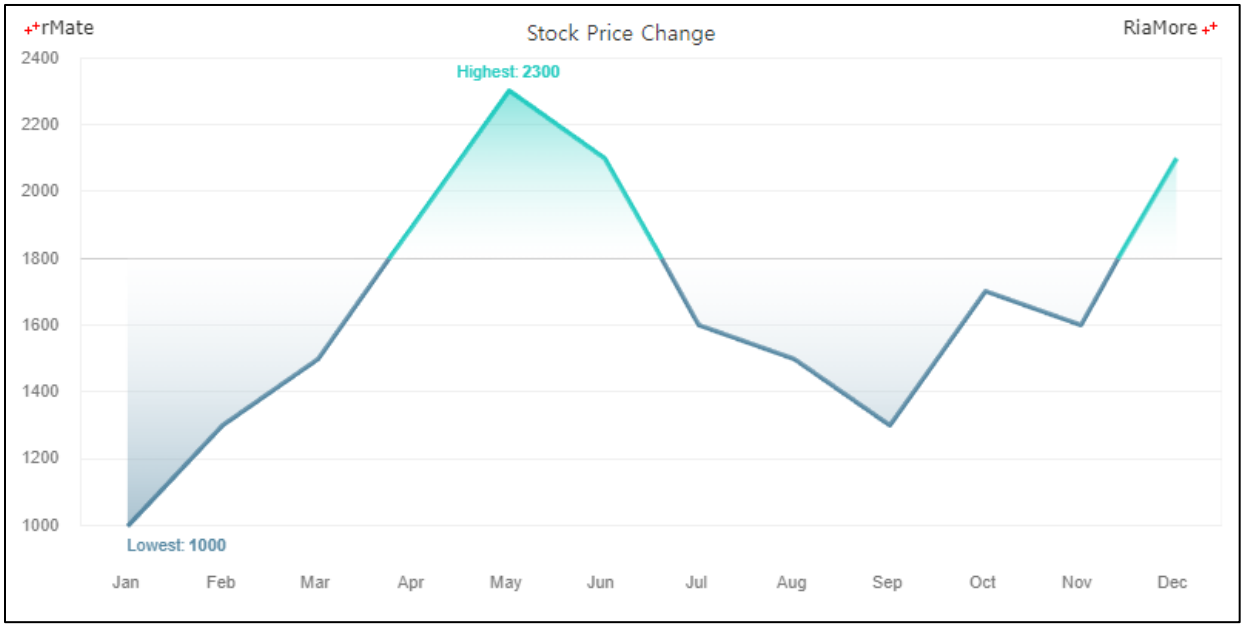
See the CodePen 알메이트 차트 - 100% 1 영역 차트

베이스라인 설정

영역 차트에 기준값을 지정하고 이를 차트 내에 직선(베이스 라인, baseline)으로 표시한 후 베이스 라인 위 영역과 아래 영역에 대한 스타일링을 다르게 지정할 수 있습니다.
기준값은 <Area2DSeries> 노드의 baseValue 속성에 설정하고 베이스 라인 위 영역에 대한 선과 배경색은 <areaStroke> 과 <areaFill> 속성에, 베이스 라인 아래 영역에 대한 선과 배경색은 <areaDeclineStroke> 과 <areaDeclineFill> 속성에 설정합니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 최대값, 최소값을 차트에 표현하기 위해

```
<Area2DChart showDataTips="true">
  ...
  <Area2DSeries yField="Profit" displayName="Profit" baseValue="1800"
        showMaxValueLabel="true" showMinValueLabel="true" upLabelJsFunction="upFunc"
        downLabelJsFunction="downFunc">
  ...
</Area2DChart>

function upFunc(seriesId, index, data, values){
  return "<font color='#21cbc0'>Highest: <b>" + values[1] + "</b></font>";
}

function downFunc(seriesId, index, data, values){
  return "<font color='#5587a2'>Lowest: <b>" + values[1] + "</b></font>";
}
```

showMaxValueLabel 과 showMinValueLabel 속성을 “true” 로 설정하였고, upLabelJsFunction 과 downLabelJsFunction 속성에 사용자 정의 함수명이 설정되었습니다.



See the CodePen 알메이트 차트 - 영역 차트에 베이스라인 설정

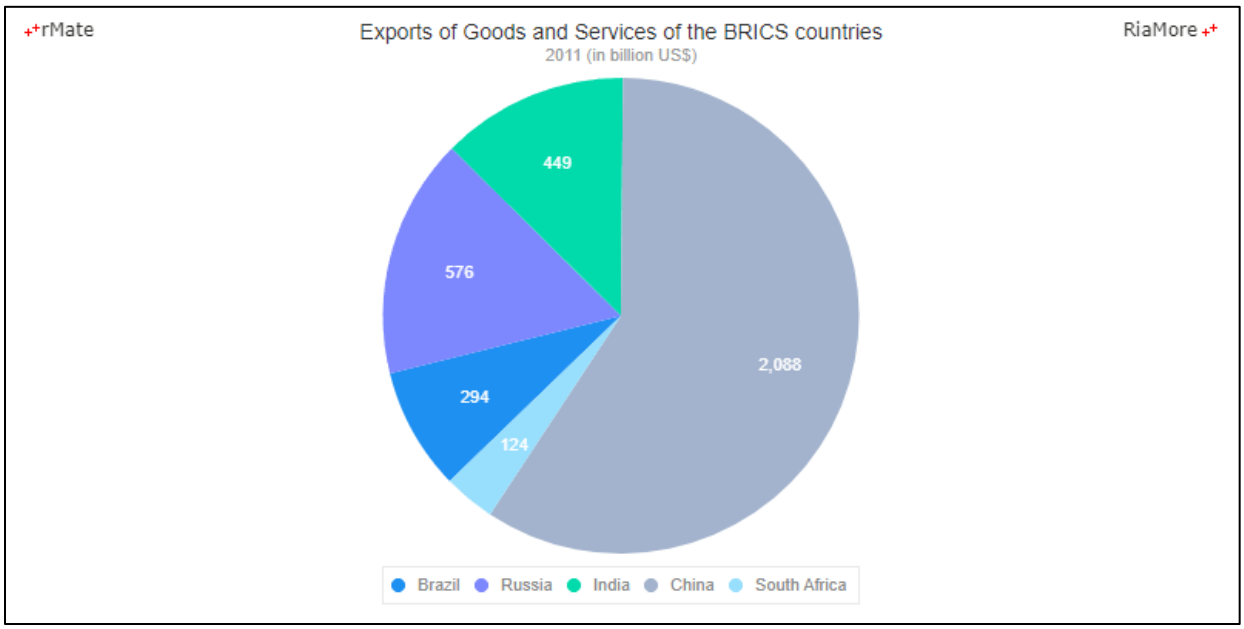
4.5 파이 차트

파이 차트는 데이터 시리즈의 전체 합을 100% 로 했을 때 각 데이터들이 차지하는 상대적인 비율에 따라서 슬라이스의 크기를 할당하여 원형 차트를 표현합니다. 따라서 각 슬라이스 곡선의 길이는 데이터의 크기와 비례합니다. 도넛 차트는 일종의 파이 차트이며 파이 차트의 중앙 부분이 잘려나간 형태입니다. 중앙의 잘려나간 영역에 텍스트 정보를 표현할 수 있습니다. 파이(도넛) 차트의 사용은 일반적으로 데이터 시리즈를 구성하는 카테고리의 수가 6 개 이하일 때가 적절하며 이보다 많아지면 가독성이 떨어져서 차트를 이해하기 어려울 수 있습니다. 따라서 이 경우는 컬럼 차트와 같이 다른 적절한 차트 유형을 사용하는 것을 고려할 필요가 있습니다.

파이 차트

파이 차트는 <Pie2DChart> 노드의 series 속성값에 <PieD2Series> 노드를 설정하여 생성할 수 있습니다. 파이 차트에서 처음 슬라이스가 시작되는 위치는 <Pie2DSeries> 노드의 startAngle 속성(기본값: “0”)에 의해서 결정됩니다. 이 값은 원의 중심에서 오른쪽으로 수평 방향 즉 시계 3 시 방향(기본값: “0”)에서 시작하여 시계 방향으로 진행하는 각도 값에 해당합니다. 다음은 처음 슬라이스(Brazil)가 시계 3 시 방향에서 시작하여 시계 방향으로 “136” 도 위치에서 시작하는 파이 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Pie2DChart showDataTips="true">
  <series>
    <Pie2DSeries nameField="Country" field="Export" labelPosition="inside"
       color="#ffffff" startAngle="136">
      <showDataEffect>
        <SeriesSlide duration="1000"/>
      </showDataEffect>
    </Pie2DSeries>
  </series>
</Pie2DChart>

var chartData = [
  {"Country" : "Brazil", "Export" : 294},
  {"Country" : "Russia", "Export" : 576},
  {"Country" : "India", "Export" : 449},
  {"Country" : "China", "Export" : 2088},
  {"Country" : "South Africa", "Export" : 124}
];
```



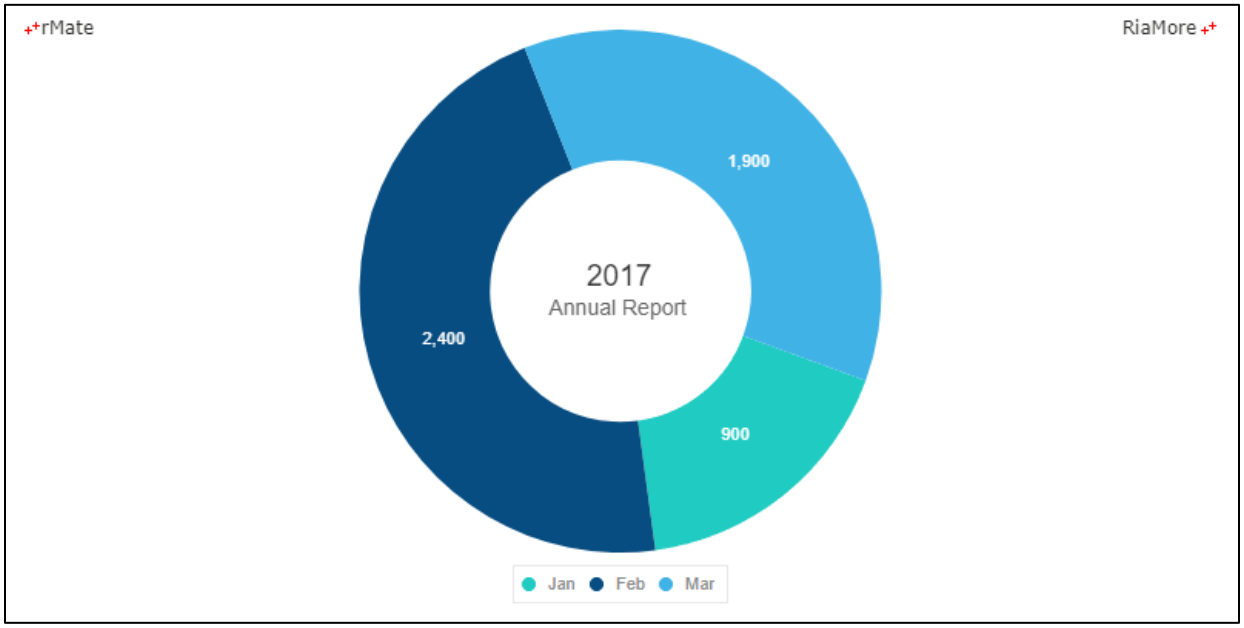
See the CodePen 알메이트 차트 - 파이 차트

도넛 차트

도넛 차트는 <Pie2DChart> 노드의 innerRadius 속성을 설정하여 생성할 수 있습니다. innerRadius 속성의 기본값은 “0” 이며 파이 차트 원의 반지름에 대한 중앙에 잘려나간 원의 반지름의 비율을 의미합니다. 다음은 중앙에 잘려나간 원의 반지름을 파이 차트 원의 반지름의 절반(“0.5”)으로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Pie2DChart innerRadius="0.5" showDataTips="true">
  <series>
    <Pie2DSeries nameField="Month" field="Profit" startAngle="20"
       renderDirection="clockwise" labelPosition="inside" color="#ffffff">
      <showDataEffect>
        <SeriesZoom duration="1000"/>
      </showDataEffect>
    </Pie2DSeries>
  </series>
  <backgroundElements>
    <CanvasElement>
      <Label text="2017" height="24" horizontalCenter="0" verticalCenter="-10"
       fontSize="19" color="#333333" backgroundAlpha="0"/>
      <Label text="Annual Report" height="24" horizontalCenter="0"
       verticalCenter="10" fontSize="14" color="#666666" backgroundAlpha="0"/>
    </CanvasElement>
  </backgroundElements>
</Pie2DChart>
```

위 예에서는 중앙에 잘려나간 원에 텍스트 정보(2017, Annual Report)가 표시되어 있습니다. 이는 차트의 배경(<backgroundElements> 노드)에 <CanvasElement> 노드와 <Label> 노드를 설정하여 표시할 수 있습니다. <Label> 노드에는 표현하고자 하는 텍스트의 내용(text 속성)과 텍스트가 표시되는 좌표 정보(horizontalCenter, verticalCenter 속성)가 설정되어 있습니다.



See the CodePen 알메이트 차트 - 도넛 차트

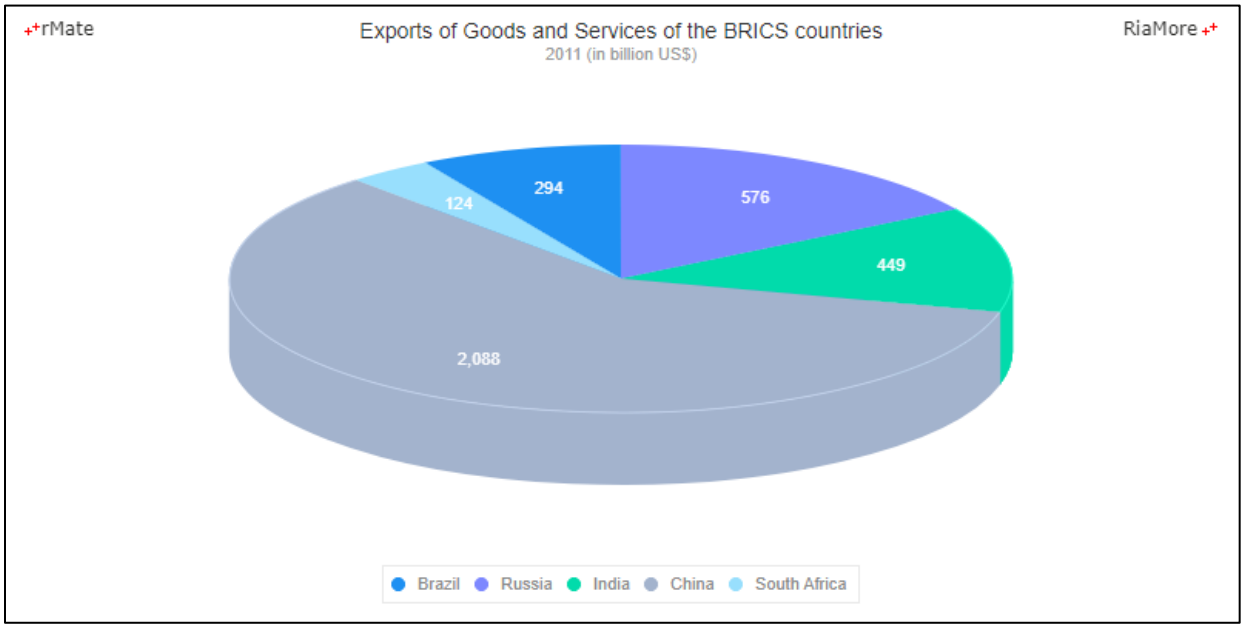
3D 파이(도넛) 차트

3D 파이 차트는 <Pie3DChart> 노드의 series 속성값에 <Pie3DSeries> 노드를 설정하여 생성할 수 있습니다. 3D 파이 차트에서 실린더의 모양은 <Pie3DChart> 노드의 depth 속성과 elevation 속성을 이용하여 조절할 수 있습니다. 두 속성에 대한 설명은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
depth	숫자 기본값: 50	실린더의 깊이를 지정합니다.
elevation	-90 과 90 사이의 숫자 기본값: 70	실린더의 기울기 각도를 지정합니다.

다음은 depth 속성과 elevation 속성이 기본값(depth: “50”, elevation:“70”, 지정되지 않음)인 3D 파이 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

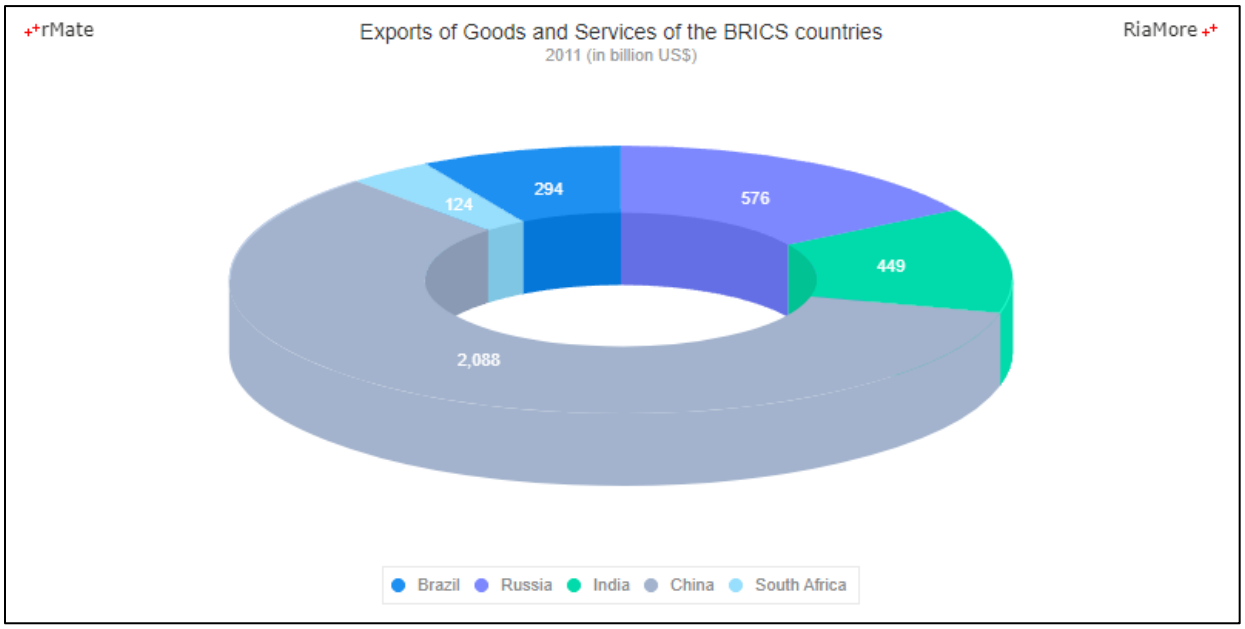
```
<Pie3DChart showDataTips="true" paddingLeft="100" paddingTop="50"
       paddingRight="100" paddingBottom="50">
  <series>
    <Pie3DSeries nameField="Country" field="Export" labelPosition="inside"
       color="#ffffff" startAngle="240">
      <showDataEffect>
        <SeriesInterpolate duration="1000"/>
      </showDataEffect>
    </Pie3DSeries>
  </series>
</Pie3DChart>
```



See the CodePen 알메이트 차트 - 3D 파이 차트

다음은 innerRadius 속성을 “0.5” 로 설정하여 3D 도넛 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Pie3DChart showDataTips="true" paddingLeft="100" paddingTop="50"
        paddingRight="100" paddingBottom="50">
  <series>
    <Pie3DSeries innerRadius="0.5" nameField="Country" field="Export"
        labelPosition="inside" color="#ffffff" startAngle="240">
      <showDataEffect>
        <SeriesInterpolate duration="1000"/>
      </showDataEffect>
    </Pie3DSeries>
  </series>
</Pie3DChart>
```

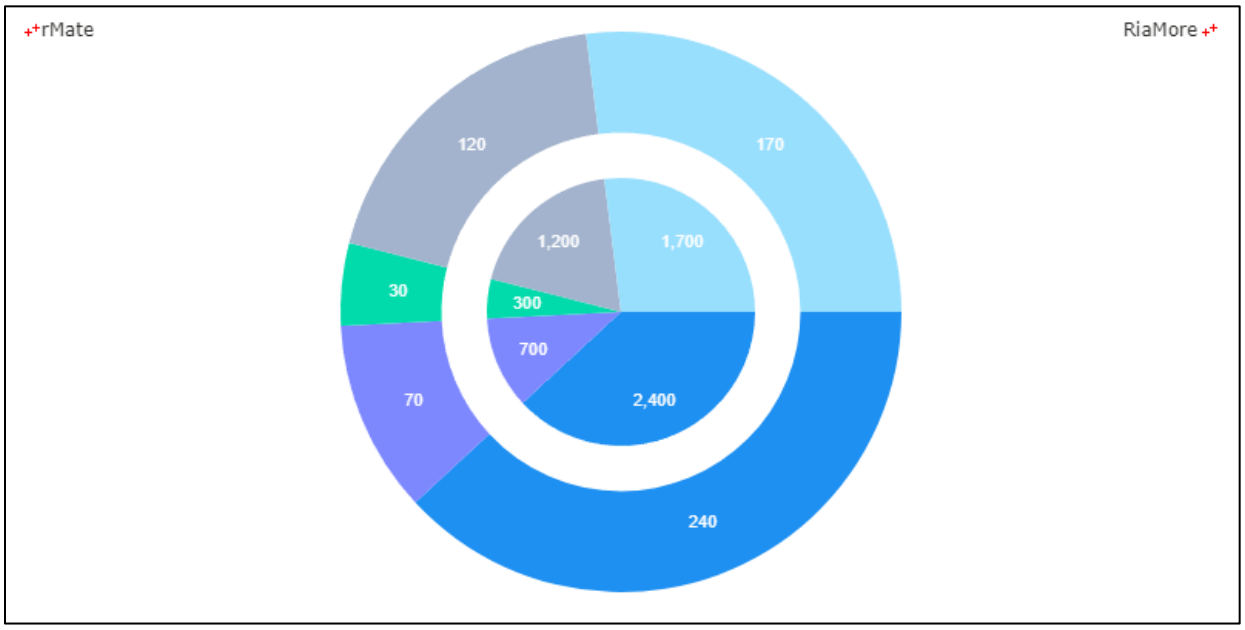


See the CodePen 알메이트 차트 - 3D 도넛 차트

스택 파이 차트

두 개 이상의 데이터 시리즈를 파이 차트에 설정하여 스택 파이 차트를 생성할 수 있습니다. 이 때 제일 바깥쪽 원에 표시될 데이터 시리즈를 제일 마지막에 선언하며 innerStackRatio 속성을 이용하여 안쪽 원에 표시될 데이터 시리즈와의 위치를 적절히 조절합니다. 다음은 두 개의 데이터 시리즈를 이용하여 스택 파이 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Pie2DChart showDataTips="true">
  <series>
    <Pie2DSeries nameField="Month" field="Profit" displayName="Profit"
        labelPosition="inside" formatter="{numFmt}" color="#ffffff" >
      <showDataEffect>
        <SeriesSlide direction="right" duration="1000"/>
      </showDataEffect>
    </Pie2DSeries>
    <Pie2DSeries nameField="Month" field="Cost" displayName="Cost"
        labelPosition="inside" formatter="{numFmt}" color="#ffffff"
        insideLabelRatio="0.8" innerStackRatio="0.06" >
      <showDataEffect>
        <SeriesSlide direction="right" duration="1000"/>
      </showDataEffect>
    </Pie2DSeries>
  </series>
</Pie2DChart>
```

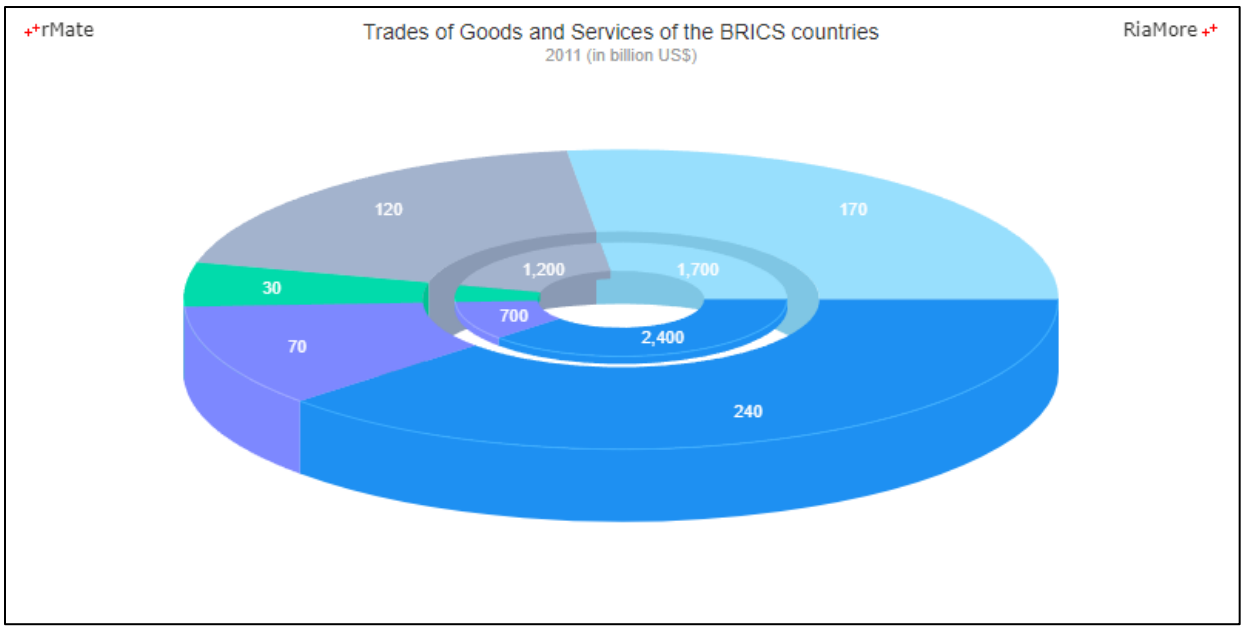


See the CodePen 알메이트 차트 - 스택 파이 차트

스택 도넛 차트

스택 파이 차트에서 가장 먼저 정의된 데이터 시리즈(<Pie3DSeries> 노드)의 innerRadius 속성에 값을 지정하여 스택 도넛 차트를 생성할 수 있습니다. 다음은 스택 파이 차트 예제에서 가장 먼저 정의된 <Pie3DSeries> 노드의 innerRadius 속성을 “0.5” 로 설정하여 생성된 차트입니다.

```
<Pie3DChart showDataTips="true" depth="50" paddingLeft="100" paddingTop="50"
       paddingRight="100" paddingBottom="50">
  <series>
    <Pie3DSeries nameField="Month" field="Profit" innerRadius="0.5"
       displayName="Profit" labelPosition="inside" formatter="{numFmt}"
       color="#ffffff" >
      <showDataEffect>
        <SeriesSlide direction="right" duration="1000"/>
      </showDataEffect>
    </Pie3DSeries>
    <Pie3DSeries nameField="Month" field="Cost" displayName="Cost"
       labelPosition="inside" formatter="{numFmt}" color="#ffffff"
       insideLabelRatio="0.8" innerStackRatio="0.06" >
      <showDataEffect>
        <SeriesSlide direction="right" duration="1000"/>
      </showDataEffect>
    </Pie3DSeries>
  </series>
</Pie3DChart>
```

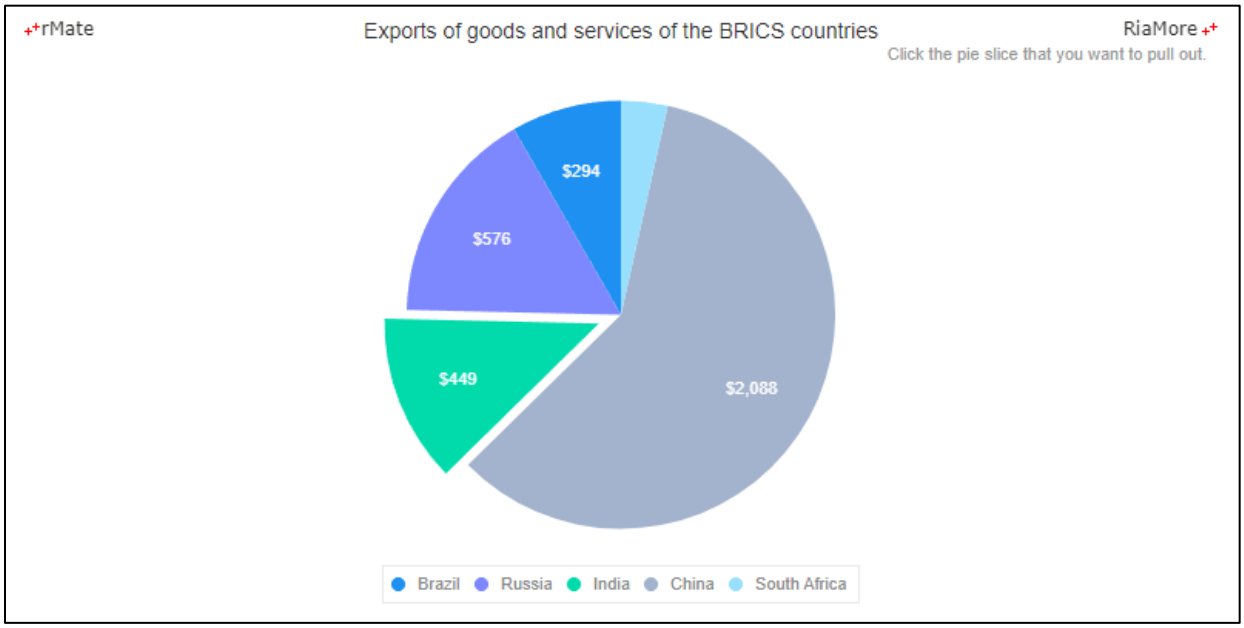


See the CodePen 알메이트 차트 - 스택 도넛 차트

파이 차트 생성시 특정 슬라이스 밖으로 나오기

<Pie2DSeries> (<Pie3DSeries>) 노드의 explodable(기본값: “true”) 속성은 사용자가 특정 슬라이스를 클릭하면 클릭된 슬라이스를 밖으로 나오게할지 여부를 설정합니다. 특정 슬라이스를 밖으로 나오게 하는 기능을 차트가 생성되는 시점에도 적용할 수 있습니다. <Pie2DSeries> (<Pie3DSeries>) 노드의 perWedgeExplodeRadius 속성에 전체 슬라이스 중에서 어떤 슬라이스를 차트가 생성되는 순간에 밖으로 나오게할지를 설정할 수 있습니다. (예, 총 5 개의 슬라이스 중에서 3 번째 슬라이스를 밖으로 나오게 하는 설정: perWedgeExplodeRadius = “[0,0,0.1,0,0]”) 다음은 파이 차트의 총 5 개의 슬라이스 중에서 3 번째 슬라이스를 밖으로 나오게 하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Pie2DChart showDataTips="true">
  <series>
    <Pie2DSeries nameField="Country" field="Export"
       perWedgeExplodeRadius="[0,0,0.1,0,0]" labelPosition="inside"
       formatter="{numFmt}" color="#ffffff" renderDirection="counterClockwise"
       startAngle="90">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </Pie2DSeries>
  </series>
</Pie2DChart>
```

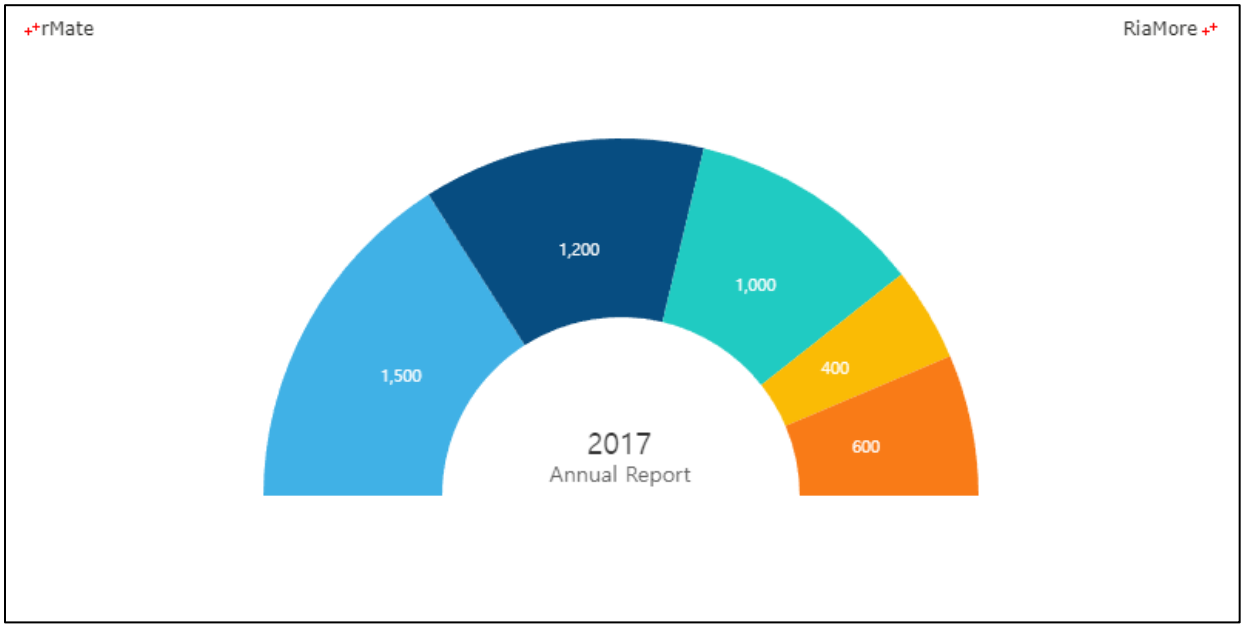


See the CodePen 알메이트 차트 - 파이 차트 생성시 특정 슬라이스 밖으로 나오기

반원 파이 차트

반원 파이 차트는 여러 개의 차트를 한 페이지에 표현하거나 대시보드 생성과 같은 용도로 유용하게 활용될 수 있습니다. 반원 파이 차트는 <HalfPie2DChart> 노드의 series 속성값에 <HalfPie2DSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 반원 파이 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<HalfPie2DChart showDataTips="true" paddingTop="70" paddingBottom="70">
  <series>
    <HalfPie2DSeries nameField="Month" field="Profit" labelPosition="inside"
       color="#fff">
      <showDataEffect>
        <SeriesInterpolate duration="1000"/>
      </showDataEffect>
    </HalfPie2DSeries>
  </series>
  <annotationElements>
    <CanvasElement>
      <Label horizontalCenter="0" height="24" fontSize="19" verticalCenter="86"
        text="2017" color="#333333"/>
      <Label horizontalCenter="0" height="19" fontSize="14" verticalCenter="106"
        text="Annual Report" color="#666666"/>
    </CanvasElement>
  </annotationElements>
</HalfPie2DChart>
```



See the CodePen 알메이트 차트 - 반원 파이 차트

레이블(데이터 값) 지시선

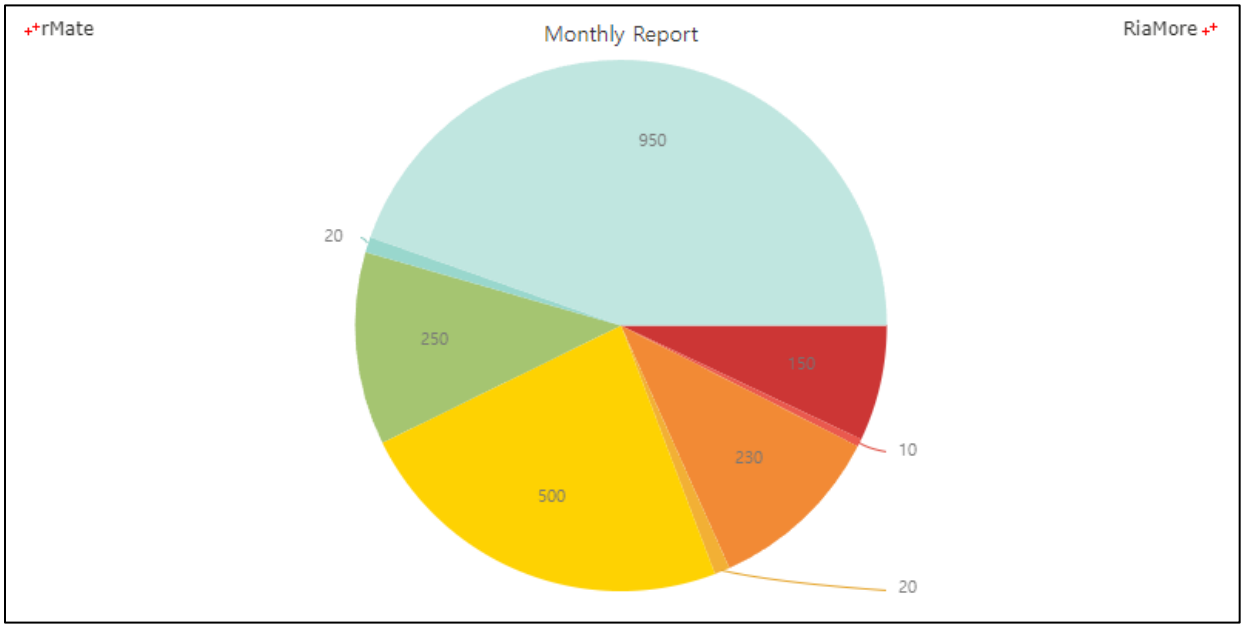
파이 차트에서 슬라이스의 데이터 값(레이블)을 표시하는 방법은 labelPosition (기본값: “none”) 속성을 이용해서 조절할 수 있습니다. labelPosition 속성에 대한 설명은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
labelPosition	inside, outside, callout, insideWithCallout, none(*)	inside: 슬라이스 내부에 레이블을 표시합니다. outside: 슬라이스 외부에 레이블을 표시합니다. callout: 슬라이스 외부에 레이블을 표시하고 슬라이스와 레이블을 지시선으로 연결합니다. insideWithCallout: 기본적으로 슬라이스 내부에 레이블을 표시하고 레이블을 표시할 충분한 공간이 없는 경우에는 callout 과 같은 방법으로 표시합니다. none: 레이블을 표시하지 않습니다.

차트의 슬라이스에 레이블을 표시할 충분한 공간이 없을 경우에는 레이블을 슬라이스의 외부에 표시하고 슬라이스와 레이블을 지시선으로 연결할 수 있습니다. labelPosition 속성값이 “callout” 혹은 “insideWithCallout” 일 경우, 레이블과 해당 슬라이스를 연결하는 지시선이 표시됩니다. labelPosition 속성값이 “callout” 이면 모든 슬라이스의 레이블이 슬라이스 외부에

```
<Pie2DChart showDataTips="true">
  <series>
    <Pie2DSeries labelPosition="insideWithCallout" labelYOffset="-2" field="Profit"
       nameField="Month" displayName="Profit" formatter="{numFmt}">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
      <fills>
        <SolidColor color="#CC3635" />
        <SolidColor color="#EA594E" />
        <SolidColor color="#F28A35" />
        <SolidColor color="#F2B035" />
        <SolidColor color="#FED202" />
        <SolidColor color="#A5C571" />
        <SolidColor color="#99D7CD" />
        <SolidColor color="#C0E6E0" />
      </fills>
    </Pie2DSeries>
  </series>
</Pie2DChart>
```

표시되고, “insideWithCallout” 이면 기본적으로 슬라이스의 내부에 레이블이 표시되고 레이블을 표시할 충분한 공간이 없는 경우에만 레이블이 슬라이스 외부에 표시됩니다. 다음은 labelPosition 속성을 “insideWithCallout” 으로 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.



See the CodePen 알메이트 차트 - 파이 차트에 레이블(데이터 값) 지시선 표시

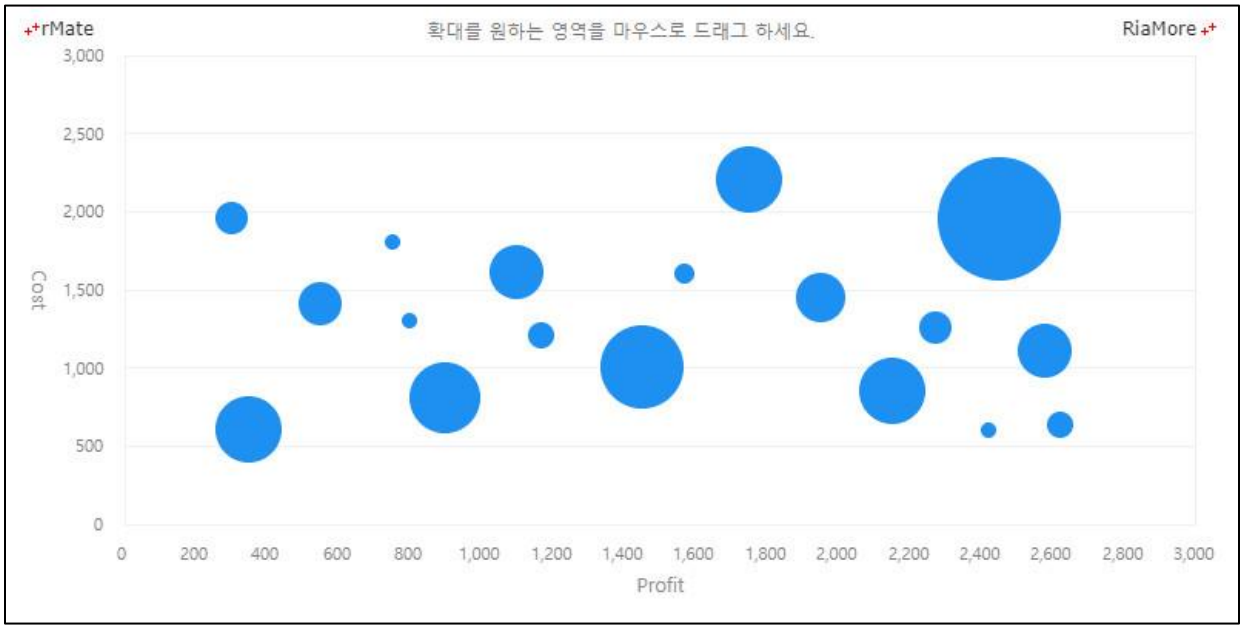
4.6 버블 차트

버블 차트는 데이터 포인트가 버블로 표시되고 추가적인 데이터 차원을 버블 크기로 나타내는 플롯(스캐터) 차트의 변형입니다. 플롯 차트와 마찬가지로 일반적으로 버블 차트는 카테고리 축을 사용하지 않고, 가로 축과 세로 축 모두 값으로 표현되는 축을 사용합니다.

버블 차트는 <Bubble2DChart> 노드의 series 속성값에 <Bubble2DSeries> 노드를 설정하여 생성할 수 있습니다. 버블의 크기는 버블의 최소 크기와 최대 크기에 기반하여 데이터 값의 상대적 비율에 따라서 자동으로 계산됩니다. 버블의 최소 크기와 최대 크기는 <Bubble2DChart> 노드의 minRadius 속성과 maxRadius 속성에 설정됩니다.

다음은 버블 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bubble2DChart showDataTips="true" minRadius="5" maxRadius="40">
  ...
  <series>
    <Bubble2DSeries displayName="Profit/Cost/Revenue" xField="Profit" yField="Cost"
       radiusField="Revenue">
      ...
    </Bubble2DSeries>
   </series>
   ...
</Bubble2DChart>
```

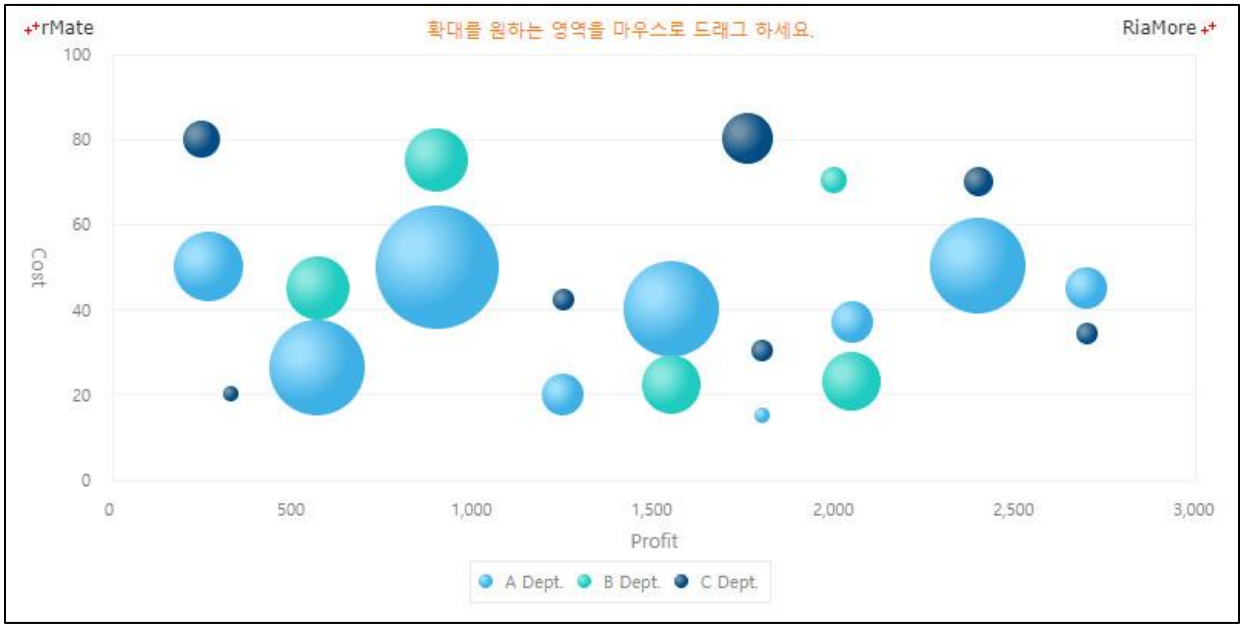


See the CodePen 알메이트 차트 - 버블 차트

다중 시리즈 버블 차트

여러 개의 데이터 시리즈를 버블 차트에 적용할 수 있습니다. 다음은 A, B, C 부서 각각의 데이터 시리즈에 대한 Profit/Cost/Revenue 관계를 버블 차트로 표현하는 코드와이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 <Bubble2DSeries> 노드의 <fill> 속성에 <RadialGradient> 노드를 적용하여 버블에 방사형(Radial) 그라디언트 색을 칠하였습니다. 방사형(Radial) 그라디언트 색 설정에 관해서는 영역에 대한 방사형(Radial) 그라디언트 색 설정하기를 참조하십시요.

```
<Bubble2DChart maxRadius="40" minRadius="5" showDataTips="true">
  ...
  <series>
    <Bubble2DSeries displayName="A Dept." xField="A_Profit" yField="A_Cost"
       radiusField="A_Revenue">
      ...
    </Bubble2DSeries>
    <Bubble2DSeries displayName="B Dept." xField="B_Profit" yField="B_Cost"
       radiusField="B_Revenue">
      ...
    </Bubble2DSeries>
    <Bubble2DSeries displayName="C Dept." xField="C_Profit" yField="C_Cost"
       radiusField="C_Revenue">
      ...
    </Bubble2DSeries>
  </series>
  ...
</Bubble2DChart>
```

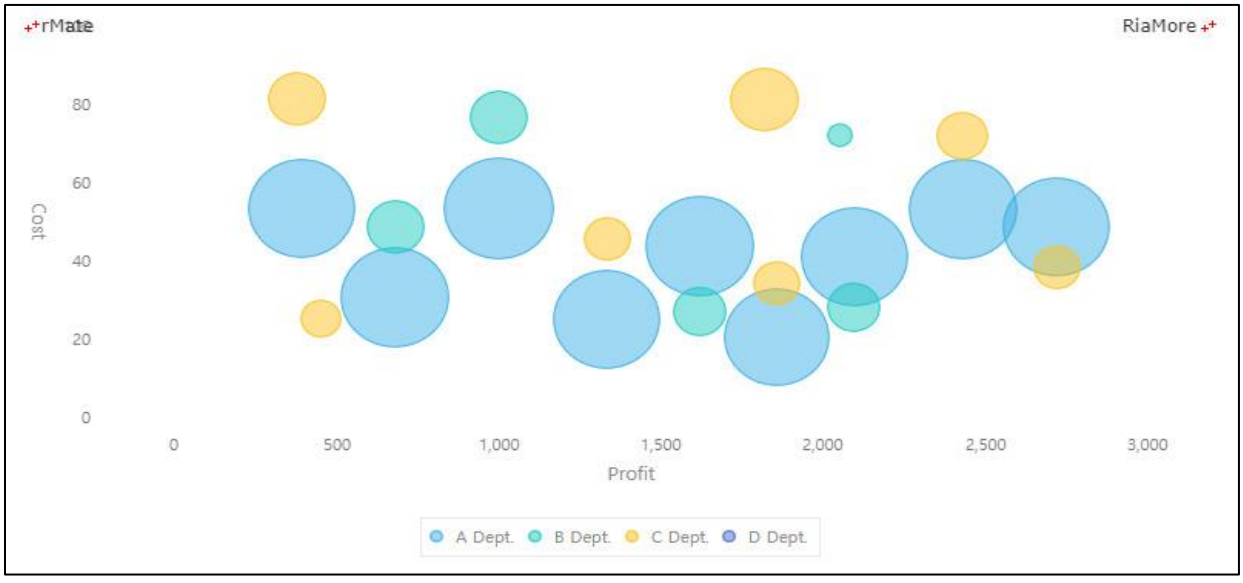


See the CodePen 알메이트 차트 - 다중 시리즈 버블 차트

3D 버블 차트

3D 형태의 X, Y 축을 적용한 3D 버블 차트를 생성할 수 있습니다. 3D 버블 차트는 <Bubble3DChart> 노드의 series 속성값에 <Bubble3DSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 3D 버블 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 <Bubble3DSeries> 노드의 <fill> 속성에 <SolidColor> 노드를 적용하여 버블에 반투명(alpha = “0.5”)한 색을 칠하였습니다. 색 설정에 관해서는 영역에 대한 색 설정하기를 참조하십시요.

```
<Bubble3DChart showDataTips="true" minRadius="5" maxRadius="40">
  ...
  <series>
    <Bubble3DSeries displayName="A Dept." xField="A_Profit" yField="A_Cost"
       radiusField="A_Revenue">
      ...
    </Bubble3DSeries>
    <Bubble3DSeries displayName="B Dept." xField="B_Profit" yField="B_Cost"
       radiusField="B_Revenue">
      ...
    </Bubble3DSeries>
    <Bubble3DSeries displayName="C Dept." xField="C_Profit" yField="C_Cost"
       radiusField="C_Revenue">
      ...
    </Bubble3DSeries>
    <Bubble3DSeries displayName="D Dept." xField="D_Profit" yField="D_Cost"
       radiusField="D_Revenue">
      ...
    </Bubble3DSeries>
  </series>
  ...
</Bubble3DChart>
```



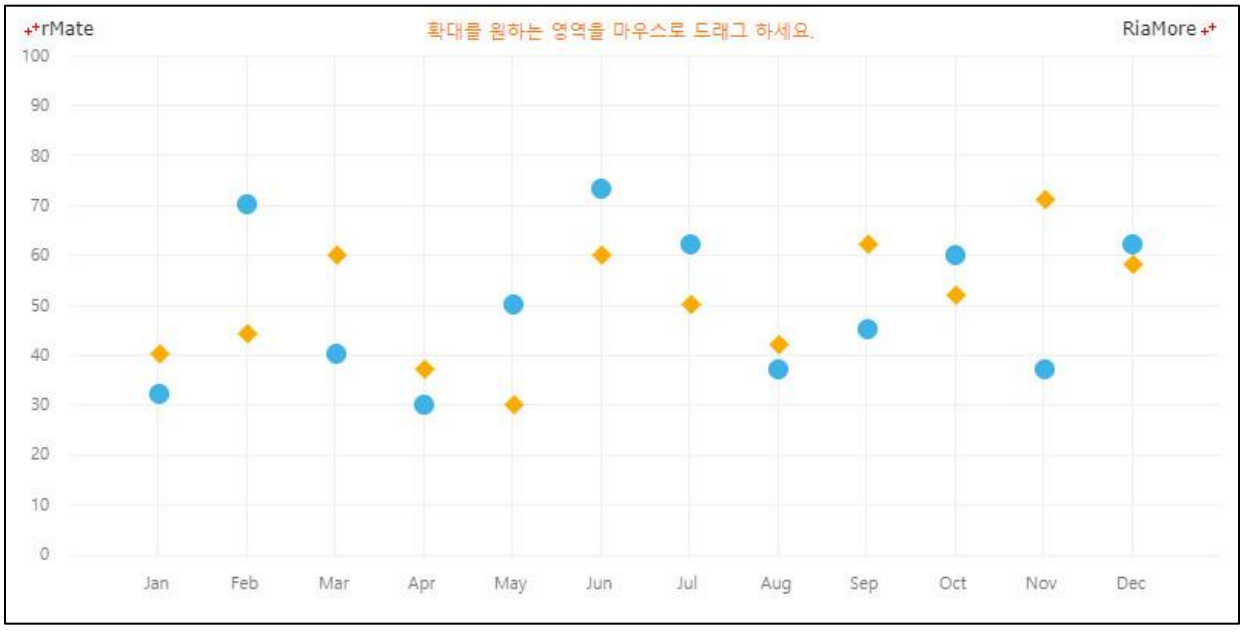
See the CodePen 알메이트 차트 - 3D 버블 차트

4.7 플롯 차트

플롯 차트는 스캐터 차트라고도 하며 데이터 포인트가 카테시안 축 상에서 X, Y 좌표(2 개의 변수 값)에 일반적으로 작은 도형으로 표현됩니다. 도형에 색을 칠할 경우 3 개의 변수 값을 데이터 포인트에 적용할 수 있습니다. 플롯 차트에 표시가 가능한 도형은 라인 차트의 데이터 포인트에 표시 가능한 도형과 동일합니다. 데이터 포인트에 도형 표시를 참조하십시요. 플롯 차트는 <Plot2DChart> 노드의 series 속성값에 <Plot2DSeries> 노드를 설정하여 생성할 수 있습니다.

다음은 플롯 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Plot2DChart showDataTips="true" >
  ...
  <series>
    <Plot2DSeries yField="Profit" radius="6.5" displayName="Profit">
      ...
    </Plot2DSeries>
    <Plot2DSeries yField="Cost" radius="6.5" displayName="Cost">
      ...
    </Plot2DSeries>
  </series>
</Plot2DChart>
```



See the CodePen 알메이트 차트 - 플롯 차트

플롯 차트 중앙에 십자선 표시

플롯 차트의 원하는 위치에 십자선을 표시할 수 있습니다. 다음은 차트의 중앙에 가로선과 세로선을 출력한 차트의 예제입니다. 이 예제에서는 가로선과 세로선을 표시하기 위해서 <GridLines> 노드의 <horizontalCenterStroke> 속성과 <verticalCenterStroke> 속성을 이용하였습니다.

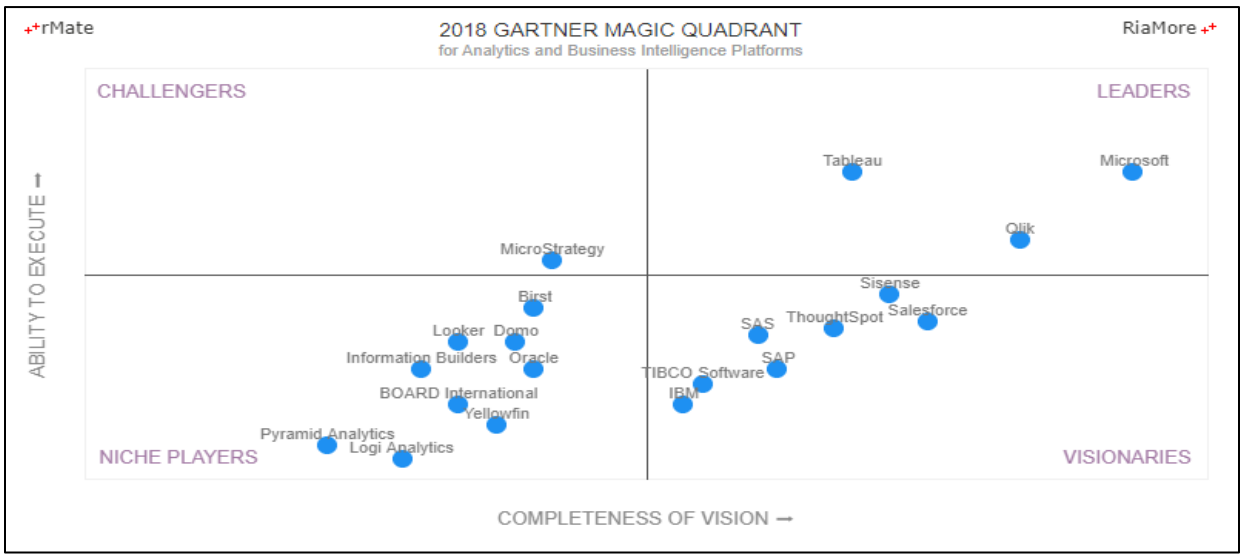
```
<Plot2DChart showDataTips="true">
  ...
  <series>
    <Plot2DSeries xField="Cost" yField="Profit" radius="7" ...>
      ...
    </Plot2DSeries>
    <Plot2DSeries xField="Revenue" yField="Profit" radius="6.5" ...>
      ...
    </Plot2DSeries>
    <Plot2DSeries xField="Cost" yField="Revenue" radius="6" ...>
      ...
    </Plot2DSeries>
  </series>
  <backgroundElements>
    <GridLines horizontalShowCenterLine="true" verticalShowCenterLine="true">
      <horizontalCenterStroke>
        <Stroke color="#c5c5c5"/>
      </horizontalCenterStroke>
      <verticalCenterStroke>
        <Stroke color="#c5c5c5"/>
      </verticalCenterStroke>
    </GridLines>
  </backgroundElements>
...
</Plot2DChart>
```

See the CodePen 알메이트 차트 - 플롯 차트에 십자선 표현

사분면 차트(쿼드런트 차트)

플롯 차트에서 축의 minimum 속성값을 음수로 지정하고 <GridLines> 노드의 horizontalShowOrigin 속성값과 verticalShowOrigin 속성값을 “true” 로 설정하면 사분면 차트를 생성할 수 있습니다. 다음은 가트너 매직 쿼드런트를 표현한 예제입니다. 레이블(회사명)을 표시하기 위해서 labelPosition 속성과 upLabelField 속성이 설정되었습니다.

```
<Plot2DChart showDataTips="true" dataTipJsFunction="dataTipFunc">
  <verticalAxis>
    <LinearAxis id="vAxis" minimum="-30" maximum="30" title="ABILITY TO EXECUTE
        ➞"/>
  </verticalAxis>
  <horizontalAxis>
    <LinearAxis id="hAxis" minimum="-30" maximum="30" title="COMPLETENESS OF VISION
        ➞"/>
  </horizontalAxis>
  <series>
    <Plot2DSeries labelPosition="up" upLabelField="company" upLabelYOffset="3"
       xField="x" yField="y" radius="6.5" itemRenderer="CircleItemRenderer">
      ...
    </Plot2DSeries>
  </series>
  <backgroundElements>
    <GridLines direction="none" verticalShowOrigin="true"
       horizontalShowOrigin="true" />
  </backgroundElements>
  <verticalAxisRenderers>
    <Axis2DRenderer axis="{vAxis}" verticalAxisTitleAlignment="vertical"
       showLabels="false" />
  </verticalAxisRenderers>
  <horizontalAxisRenderers>
    <Axis2DRenderer axis="{hAxis}" showLabels="false" />
  </horizontalAxisRenderers>
</Plot2DChart>
```



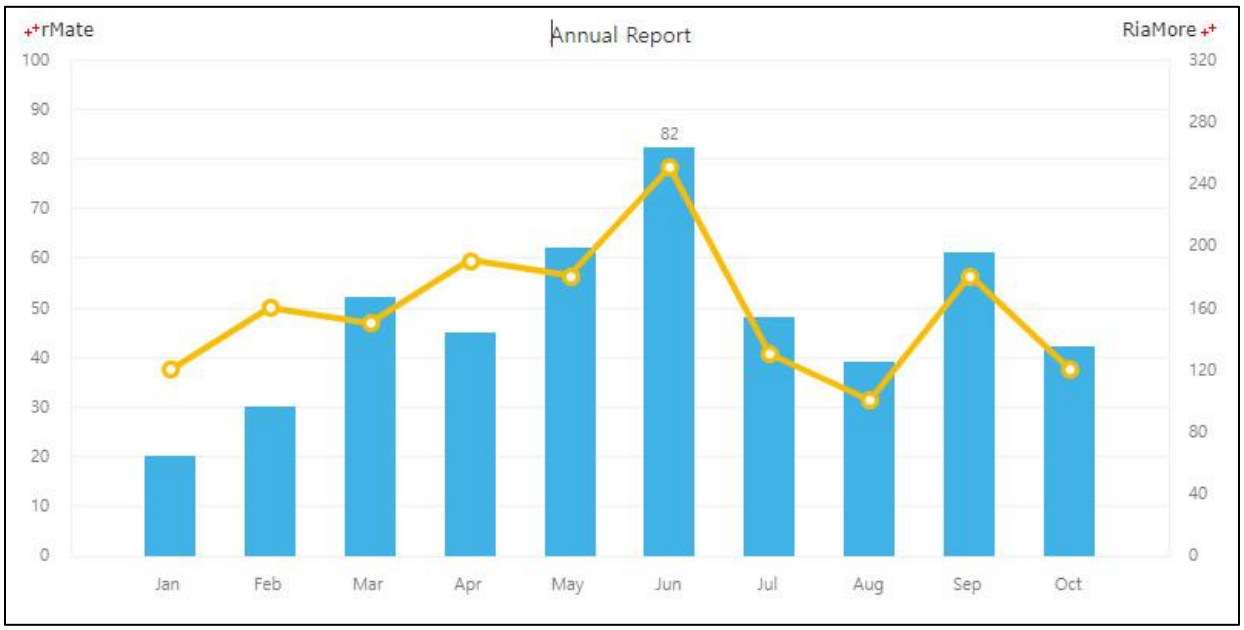
See the CodePen 알메이트 차트 - 사분면 차트 (쿼드런트 차트)

4.8 콤비네이션 차트

콤비네이션 차트는 컬럼 차트와 라인 차트(혹은 영역 차트)를 결합한 형태의 차트입니다. 이 때 차트에 표시되는 컬럼과 라인은 X 축의 동일한 카테고리(예, 1 월, 2 월, 3 월, ..)에 대해서 다른 데이터 카테고리(예, 컬럼:수입, 라인:비용)를 표현합니다. 따라서 어떤 데이터 카테고리의 값이 높고 낮은지 한눈에 비교하기 좋은 차트 유형입니다. 콤비네이션 차트는 <Combination2DChart> 노드의 series 속성값에 <Column2DSeries> 노드와 <Line2DSeries>(혹은 <Area2DSeries> 노드) 노드를 함께 설정하여 생성할 수 있습니다.

다음은 <Column2DSeries> 노드와 <Line2DSeries> 노드를 <Combination2DChart> 노드에 함께 정의하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 컬럼 시리즈와 라인 시리즈에 서로 다른 Y 축을 적용하기 위하여 두 개의 <verticalAxis> 노드를 정의하였습니다. 축 생성에 관한 더 자세한 내용은 축과 스케일을 참조하십시요.

```
<Combination2DChart showDataTips="true">
  <horizontalAxis>
    <CategoryAxis categoryField="Month" padding="1"/>
  </horizontalAxis>
  <verticalAxis>
    <LinearAxis id="vAxis1" formatter="{numfmt}" maximum="100" interval="10"/>
  </verticalAxis>
  <series>
    <Column2DSeries labelPosition="outside" yField="Profit" displayName="Profit"
       showValueLabels="[5]" columnWidthRatio="0.54">
      ...
    </Column2DSeries>
    <Line2DSeries radius="6" yField="Cost" displayName="Cost"
       itemRenderer="CircleItemRenderer">
      <verticalAxis>
        <LinearAxis id="vAxis2" interval="40" maximum="320"/>
      </verticalAxis>
      ...
    </Line2DSeries>
  </series>
  <verticalAxisRenderers>
    <Axis2DRenderer axis="{vAxis1}" showLine="false"/>
    <Axis2DRenderer axis="{vAxis2}" showLine="false"/>
  </verticalAxisRenderers>
</Combination2DChart>
```



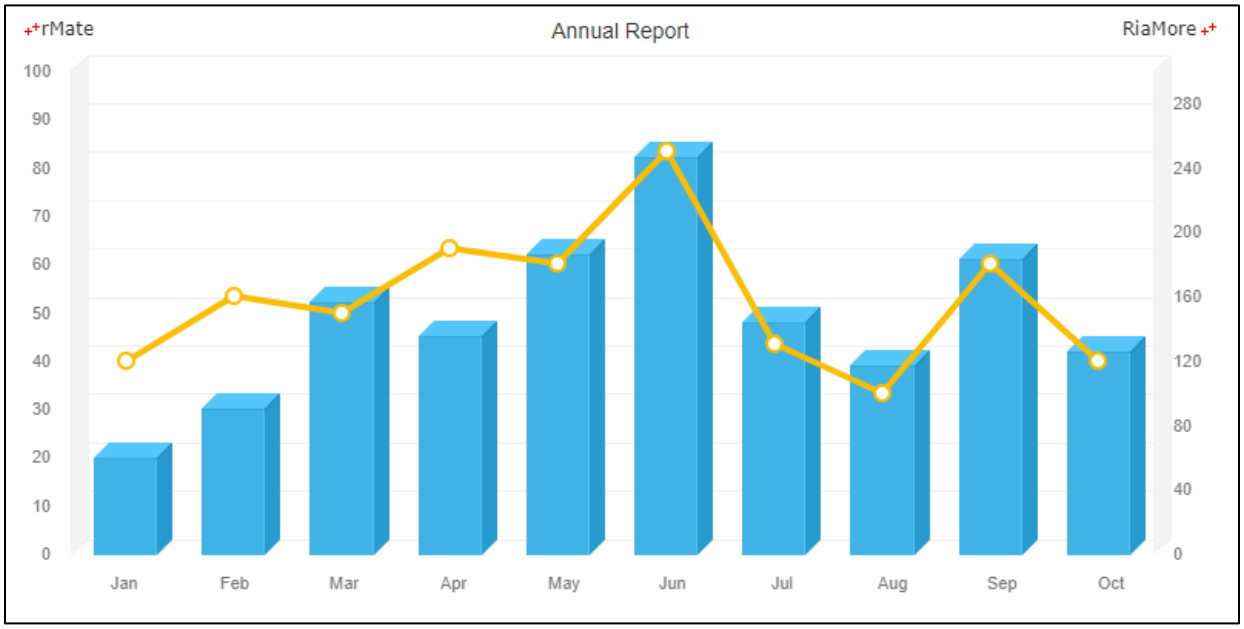
See the CodePen 알메이트 차트 - 콤비네이션 차트

3D 콤비네이션 차트

3D 콤비네이션 차트는 <Combination3DChart> 노드를 설정하여 생성할 수 있습니다.

다음은 <Column3DSeries> 노드와 <Line2DSeries> 노드를 <Combination3DChart> 노드에 정의하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 컬럼 시리즈와 라인 시리즈에 서로 다른 Y 축을 적용하기 위하여 두 개의 <verticalAxis> 노드를 정의하였고, 3D 축 생성을 위해서 <verticalAxisRenderers> 속성에 <Axis3DRenderer> 노드를 정의하였습니다. 축 생성에 관한 더 자세한 내용은 축과 스케일을 참조하십시요.

```
<Combination3DChart showDataTips="true">
  ...
  <verticalAxis>
    <LinearAxis id="vAxis1" formatter="{numfmt}" maximum="100" interval="10"/>
  </verticalAxis>
  <series>
    <Column3DSeries yField="Profit" displayName="Profit" ...>
      ...
    </Column3DSeries>
    <Line2DSeries yField="Cost" itemRenderer="CircleItemRenderer" ...>
      <verticalAxis>
        <LinearAxis id="vAxis2" maximum="300"/>
      </verticalAxis>
      ...
    </Line2DSeries>
  </series>
  <verticalAxisRenderers>
    <Axis2DRenderer axis="{vAxis1}" showLine="true"/>
    <Axis2DRenderer axis="{vAxis2}" showLine="true"/>
  </verticalAxisRenderers>
</Combination3DChart>
```



See the CodePen 알메이트 차트 - 3D 콤비네이션 차트

다중 컬럼 시리즈를 적용한 콤비네이션 차트

콤비네이션 차트에 표현되는 특정 유형의 차트에 다중 데이터 시리즈가 적용될 수 있습니다. 이 때는 다중 데이터 시리즈가 적용되는 차트 시리즈에 Set 노드 (<Area2DSet>, <Column2DSet>, <Column3DSet>)를 정의해야 합니다.

다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 두 개의 컬럼 시리즈를 <Column2DSet> 노드에 정의하였고, 라인 시리즈(<Line2DSeries>)에는 우측에 표시된 Y 축이 적용되었습니다. 축 생성에 관한 더 자세한 내용은 축과 스케일을 참조하십시요.

```
<Combination2DChart showDataTips="true">
  ...
  <series>
    <Area2DSeries yField="Profit" form="curve" displayName="Profit">
      ...
    <Column2DSet type="clustered">
      <series>
        <Column2DSeries yField="Cost" displayName="Cost">
          ...
        <Column2DSeries yField="Revenue" displayName="Revenue">
          ...
      </series>
    </Column2DSet>
    <Line2DSeries yField="Revenue" ...>
      ...
  </series>
  ...
</Combination2DChart>
```

See the CodePen 알메이트 차트 - 다중 컬럼 시리즈를 적용한 콤비네이션 차트

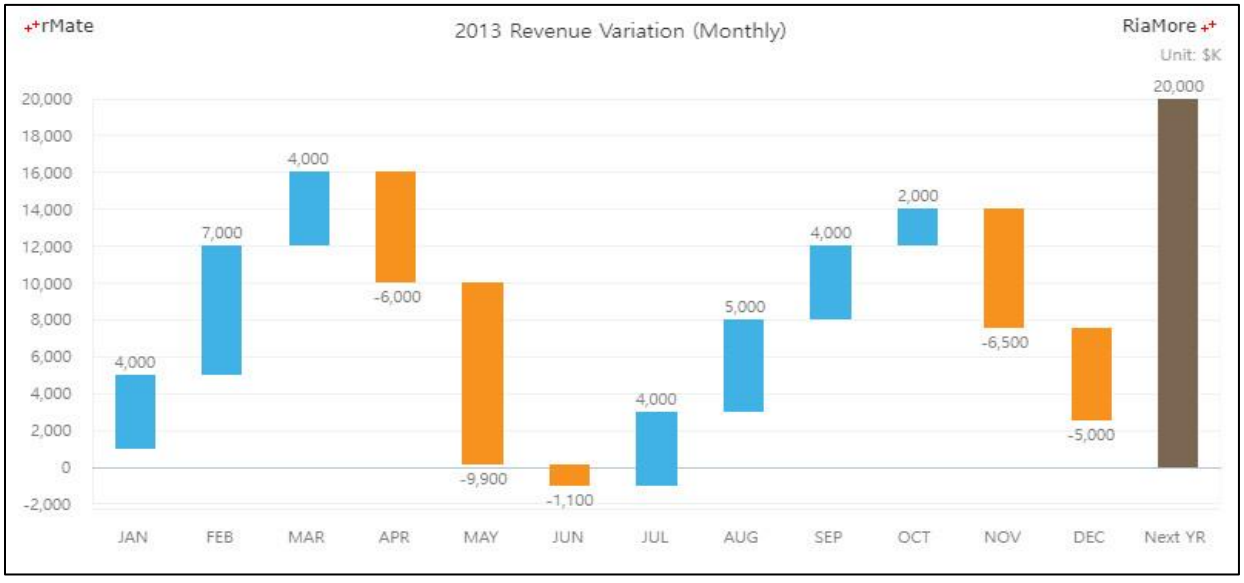
4.9 From-To 차트

From-To 차트는 시작 데이터(From)와 끝 데이터(To) 두 가지 변수값을 표현하는 차트로써, 두 가지 변수값에 대한 범위, 차이, 분포 등을 표현하는데 유용하게 활용되는 차트의 유형입니다. From-To 차트에 적용가능한 데이터 시리즈는 컬럼 시리즈(Column2DSeries, Column3DSeries), 바 시리즈(Bar2DSeries, Bar3DSeries) 그리고 영역 시리즈 (Area2DSeries)입니다. From-To 차트에서는 해당 데이터 시리즈에서 지원하는 속성들을 모두 사용할 수 있으며, 추가적으로 minField 속성을 이용하여 시작 데이터(From)를 설정합니다.

워터폴 차트

컬럼 시리즈를 From-To 차트에 적용하여 워터폴(Waterfall) 차트를 생성할 수 있습니다. 다음은 컬럼 시리즈를 이용하여 워터폴(Waterfall) 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 데이터 객체의 from 필드를 minField 속성(minField = “from”)에 to 필드를 yField 속성(yField = “to”)에 지정하였습니다.

```
<Column2DChart showDataTips="true" dataTipJsFunction="dataTipFunc">
  ...
  <series>
    <Column2DSeries id="series1" minField="from" yField="to" ...>
      ...
    </Column2DSeries>
  </series>
  ...
</Column2DChart>
```

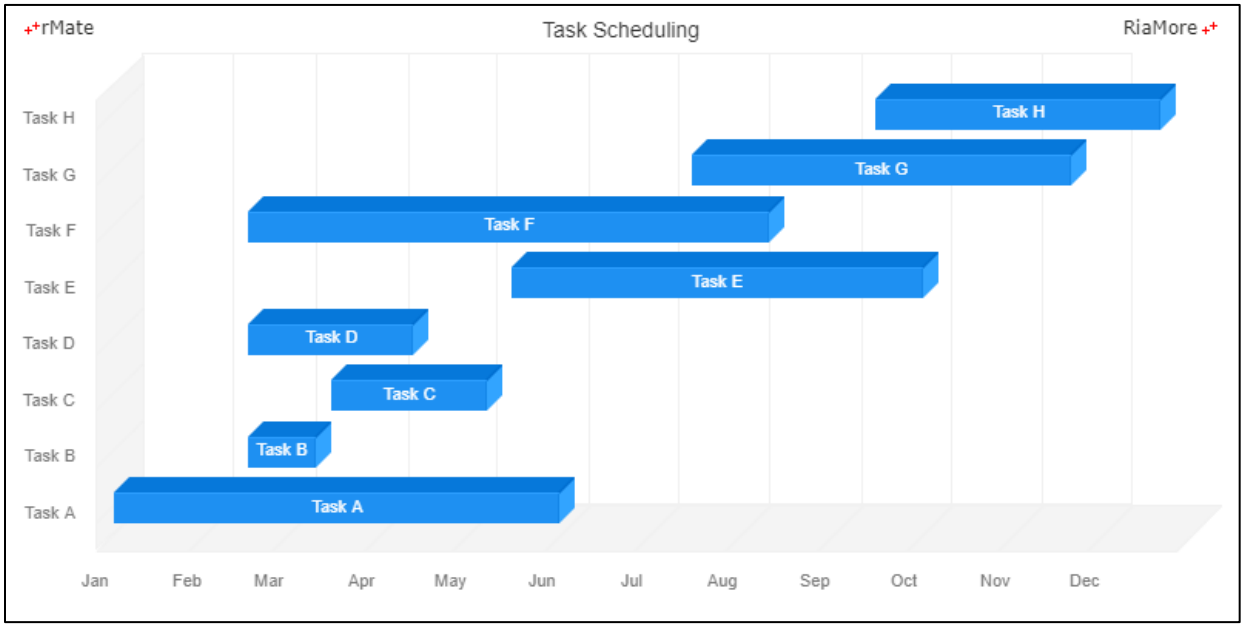


See the CodePen 알메이트 차트 - 워터폴 차트

From-To 바 차트

다음은 3D 바 시리즈(<Bar3DSeries>)를 이용하여 From-To 바 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제는 업무 태스크를 계획하는 용도로 From-To 차트를 활용할 수 있음을 보여줍니다.

```
<Bar3DChart showDataTips="true">
  ...
  <series>
    <Bar3DSeries id="series1" minField="date1" xField="date2"
        labelPosition="inside" insideLabelField="cat" color="#ffffff"
        insideLabelYOffset="-2">
      ...
    </Bar3DSeries>
  </series>
  ...
</Bar3DChart>
```

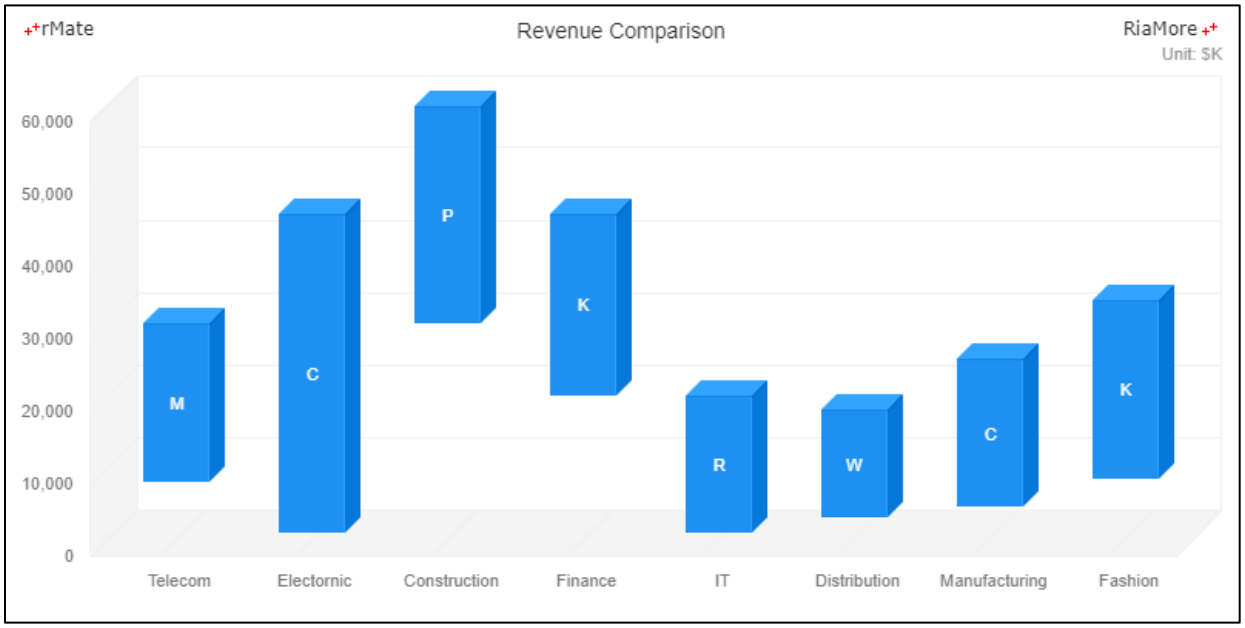


See the CodePen 알메이트 차트 - From-To 바 차트

From-To 컬럼 차트

다음은 3D 컬럼 시리즈(<Column3DSeries>)를 이용하여 From-To 컬럼 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제는 각 비지니스 영역별로 매출의 변화를 보여주기 위한 용도로 From-To 차트를 활용할 수 있음을 보여줍니다.

```
<Column3DChart showDataTips="true" dataTipJsFunction="dataTipFunc"
       columnWidthRatio="0.5">
  ...
  <series>
    <Column3DSeries id="series1" minField="min" yField="max" labelPosition="inside"
       insideLabelJsFunction="labelFunc" color="#ffffff">
      ...
    </Column3DSeries>
  </series>
</Column3DChart>
```

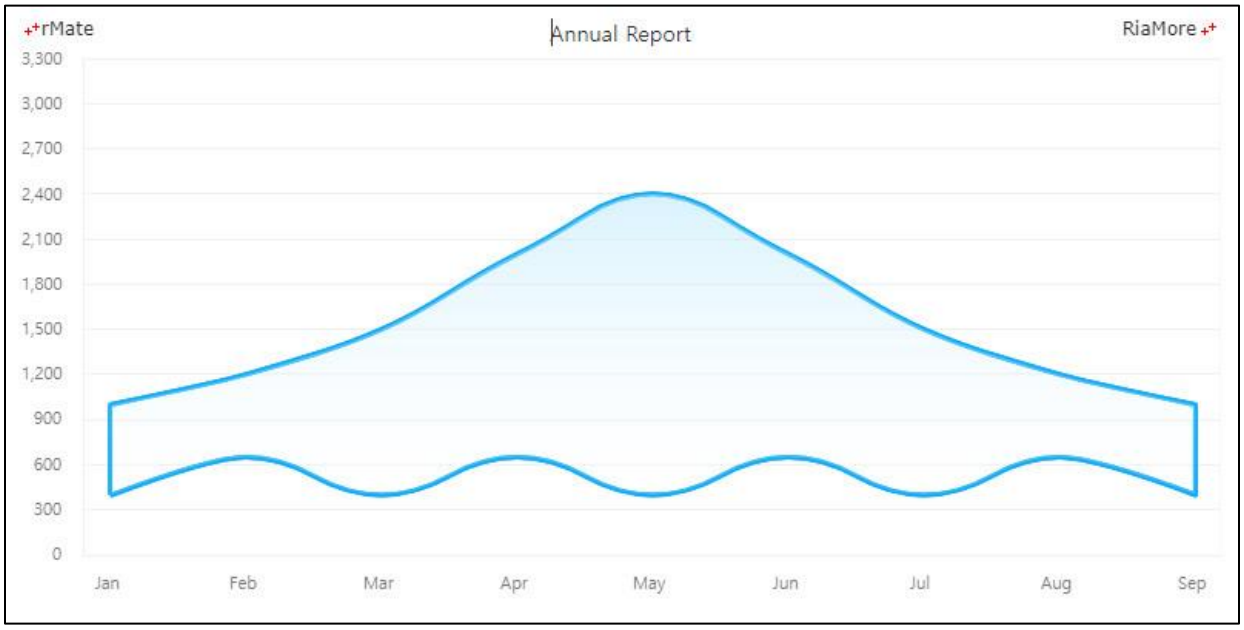


See the CodePen 알메이트 차트 - From-To 컬럼 차트

From-To 영역 차트

다음은 영역 시리즈(<Area2DSeries>)를 이용하여 From-To 영역 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제는 특정 기간에 대한 값들의 범위를 보여주기 위한 용도로 From-To 차트를 활용할 수 있음을 보여줍니다.

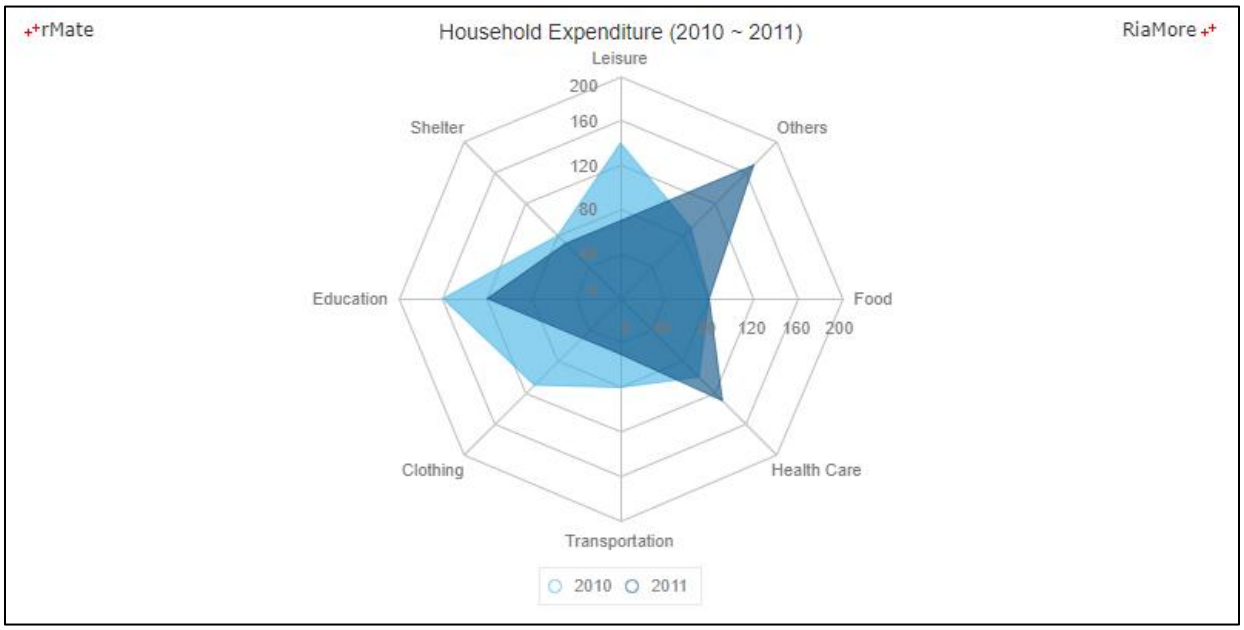
```
<Area2DChart showDataTips="true">
  ...
  <Area2DSeries minField="Cost" yField="Profit" form="curve" displayName="Profit">
    <areaFill>
      <SolidColor color="#b7e7fc" alpha="0.5"/>
    </areaFill>
    <areaStroke>
      <Stroke color="#0daaf7" weight="3"/>
    </areaStroke>
    ...
  </Area2DSeries>
  ...
</Area2DChart>
```



See the CodePen 알메이트 차트 - From-To 영역 차트

4.10 방사형 차트

방사형 차트는 파이 차트와 같은 폴라(Polar) 차트 계열과 닮았지만 다르게 작동합니다. 방사형 차트는 다중 변수 값을 2 차원 차트 상에 표시하는데, 표시되는 데이터 포인트들은 서로 연결되어 도형과 같은 모양을 갖게 됩니다. 다음은 8 가지 변수 값을 가지는 방사형 차트의 예제입니다.



See the CodePen 알메이트 차트 - 방사형 차트

방사형 차트는 <RadarChart> 노드의 series 속성값에 <RadarSeries> 노드를 설정하여 생성할 수 있습니다. 방사형 차트의 축은 두 가지가 있는데 하나는 데이터 값의 크기가 표시되는(데이터 포인트) 축으로써 <radialAxis> 속성에 정의하고, 다른 하나는 카테고리 명이 표시되는 축으로써 <angularAxis> 속성에 정의합니다. <radialAxis> 속성에는 수치값을 표현하는 <LinearAxis> 노드를 정의합니다. 위 방사형 차트 샘플에서는 다음과 같이 두 개의 <Axis2DRenderer> 노드를 정의하여

```
<radialAxis>
  <LinearAxis id="rAxis"/>
</radialAxis>
<radialAxisRenderers>
  <Axis2DRenderer axis="{rAxis}" horizontal="true" visible="true"
       tickPlacement="outside"/>
  <Axis2DRenderer axis="{rAxis}" horizontal="false" visible="true"
       tickPlacement="outside"/>
</radialAxisRenderers>
```

차트의 중앙에서 테두리까지 가로(horizontal = “true”)와 세로(horizontal = “false”), 두 개의 <LinearAxis>를 표시하였습니다.

<angularAxis> 속성에는 카테고리를 표현하는 <CategoryAxis> 노드를 정의합니다. 카테고리 명들은 방사형 차트의 바깥쪽 원호에 표시됩니다. 위 방사형 차트 샘플에서는 <AngularAxisRenderer> 노드를 정의하여 <CategoryAxis>를 표시하였습니다.

```
<angularAxis>
  <CategoryAxis id="aAxis" categoryField="catName" displayName="Category"/>
</angularAxis>
<angularAxisRenderers>
  <AngularAxisRenderer axis="{aAxis}"/>
</angularAxisRenderers>
```

축에 관한 자세한 설명은 축과 스케일을 참조하십시요.

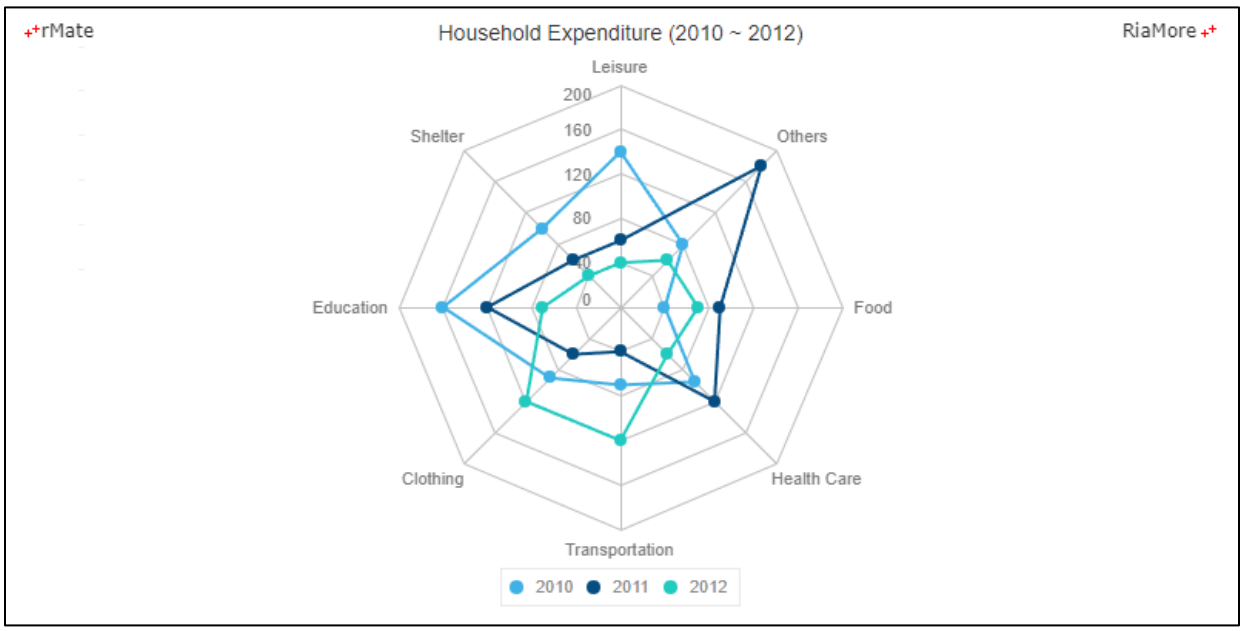
방사형 차트에서 데이터는 적용되는 카테고리 수 만큼의 객체의 배열로 정의합니다. 위 방사형 차트 샘플에서는 다음과 같이 8 개의 카테고리에 해당하는 객체가 정의된 데이터가 적용되었습니다. 각 객체에는 2010 년 자료(year2010)와 2011 년 자료(year2011)가 정의되어 있습니다.

```
var chartData = [
{"catName" : "Food", "year2010" : 80, "year2011" : 80},
{"catName" : "Health Care", "year2010" : 100, "year2011" : 130},
{"catName" : "Transportation", "year2010" : 80, "year2011" : 50},
{"catName" : "Clothing", "year2010" : 110, "year2011" : 50},
{"catName" : "Education", "year2010" : 160, "year2011" : 120},
{"catName" : "Shelter", "year2010" : 80, "year2011" : 70},
{"catName" : "Leisure", "year2010" : 140, "year2011" : 70},
{"catName" : "Others", "year2010" : 90, "year2011" : 170}
];
```

폴리곤형 방사형 차트

폴리곤형 방사형 차트는 <RadarChart> 노드의 type 속성을 “polygon” 으로 지정하여 생성합니다. 방사형 차트의 데이터 포인트에 표시되는 마커의 크기는 <RadarSeries> 노드의 radius 속성에 마커의 반지름 크기로 설정할 수 있으며 마커의 테두리 선은 <lineStroke> 속성에, 마커에 채워지는 색은 <fill> 속성에 정의합니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<RadarChart type="polygon" paddingTop="25" paddingBottom="25" showDataTips="true">
  ...
  <series>
    <RadarSeries field="year2010" radius="4" displayName="2010"
       fillLineArea="false">
      <fill>
        <SolidColor color="#40b1e6"/>
      </fill>
      <lineStroke>
        <Stroke color="#40b1e6" weight="2"/>
      </lineStroke>
      ...
    </RadarSeries>
    <RadarSeries field="year2011" radius="4" displayName="2011"
       fillLineArea="false">
      ...
    </RadarSeries>
    <RadarSeries field="year2012" radius="4" displayName="2012"
       fillLineArea="false">
      ...
    </RadarSeries>
  </series>
</RadarChart>
```



See the CodePen 알메이트 차트 - 폴리곤형 방사형 차트

원형 방사형 차트

원형 방사형 차트는 <RadarChart> 노드의 type 속성을 circle 으로 지정하여 생성합니다. 방사형 차트의 각 데이터 시리즈가 표현하는 도형(데이터 포인트들을 연결하면 생기는 도형)의 내부에 색을 칠할 수 있습니다. 도형 내부의 색은 <areaFill> 속성을 이용하여 설정할 수 있습니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<RadarChart type="circle" paddingTop="25" paddingBottom="25" showDataTips="true">
  ...
  <series>
    <RadarSeries field="year2010" displayName="2010">
      ...
      <areaFill>
        <SolidColor color="#4352a5" alpha="0.3"/>
      </areaFill>
    </RadarSeries>
    <RadarSeries field="year2012" displayName="2012">
      ...
    </RadarSeries>
    <RadarSeries field="year2013" displayName="2013">
      ...
    </RadarSeries>
  </series>
</RadarChart>
```

See the CodePen 알메이트 차트 - 원형 방사형 차트

4.11 목표 대비 실적 차트

목표 대비 실적 차트는 목표값과 실적값 두 데이터 시리즈를 표현하는 차트입니다. 목표 대비 실적 차트에는 2D 리니어 유형과 3D 실린더 유형이 있습니다. 2D 리니어 유형의 목표 대비 실적 차트는 <Combination2DChart> 노드의 series
속성값에 <VTarget2DResultSeries> 노드와 <VTarget2DGoalSeries> 노드를 설정하여 생성하고, 3D 실린더 유형의 목표 대비 실적 차트는 <Combination3DChart> 노드의 series
속성값에 <VTarget3DResultSeries> 노드(혹은 <HTarget3DResultSeries> 노드)와 <VTarget3DGoalSeries> 노드(혹은 <HTarget3DGoalSeries> 노드)를 설정하여 생성합니다. 그리고 목표와 실적을 짧은 선(대시)과 점(닷)으로 표시하는 대시 앤 닷 목표 대비 실적 차트는 <Column2DChart> 노드의 series 속성값에 <DashDotSeries>를 설정하여 생성합니다. 아래의 표는 위 설명을 정리한 것입니다.

노드	2D 리니어	3D 실린더 컬럼	3D 실린더 바	대시 앤 닷
Chart	<Combination2DChart>	<Combination3DChart>	<Combination3DChart>	<Column2DChart>
Target Series	<VTarget2DGoalSeries> <HTarget2DGoalSeries>	<VTarget3DGoalSeries>	<HTarget3DGoalSeries>	<DashDotSeries>
Actual Series	<VTarget2DResultSeries> <HTarget2DResultSeries>	<VTarget3DResultSeries>	<HTarget3DResultSeries>	<DashDotSeries>

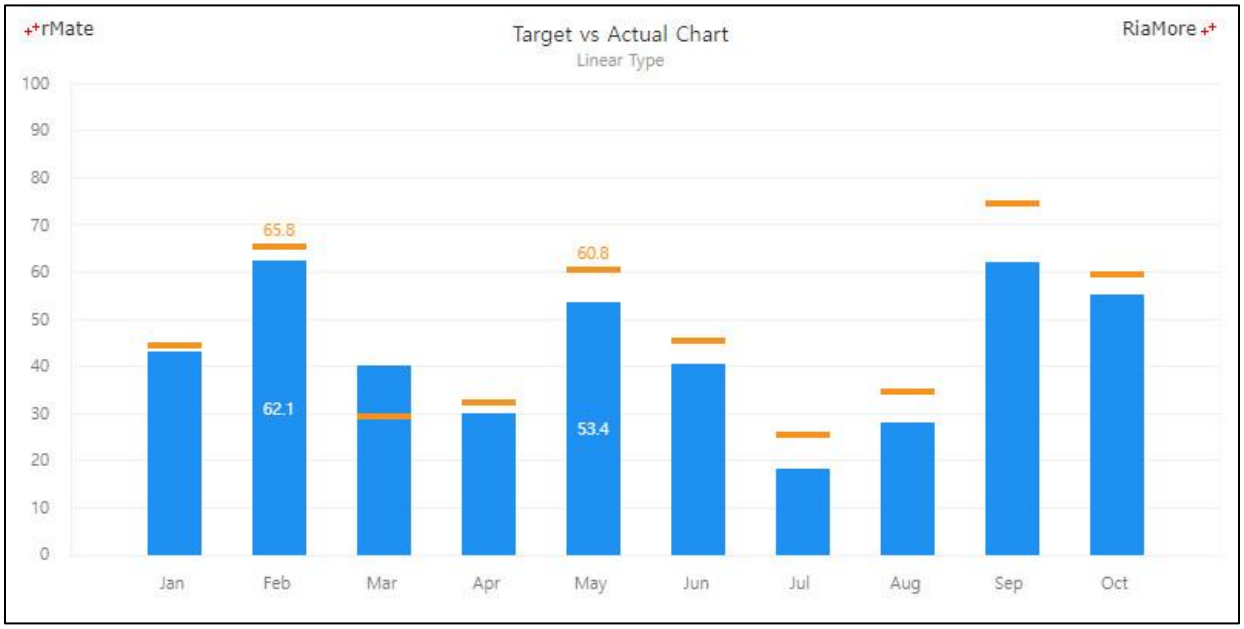
주의

> 목표 대비 실적 차트에서는 데이터 시리즈를 정의할 때 반드시 실적에 해당되는 데이터 시리즈를 먼저 정의한 후 목표에 해당되는 데이터 시리즈를 정의해야 합니다.

2D 리니어 목표 대비 실적 차트

다음은 2D 리니어 유형의 목표 대비 실적 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Combination2DChart showDataTips="true">
  ...
  <series>
    <VTarget2DResultSeries labelPosition="inside" yField="Result"
       displayName="Result" color="#ffffff" showValueLabels="[1,4]"
       columnWidthRatio="0.54">
      ...
    </VTarget2DResultSeries>
    <VTarget2DGoalSeries labelPosition="outside" yField="Goal" displayName="Goal"
       color="#f7921e" showValueLabels="[1,4]" columnWidthRatio="0.54">
      ...
    </VTarget2DGoalSeries>
  </series>
</Combination2DChart>
```

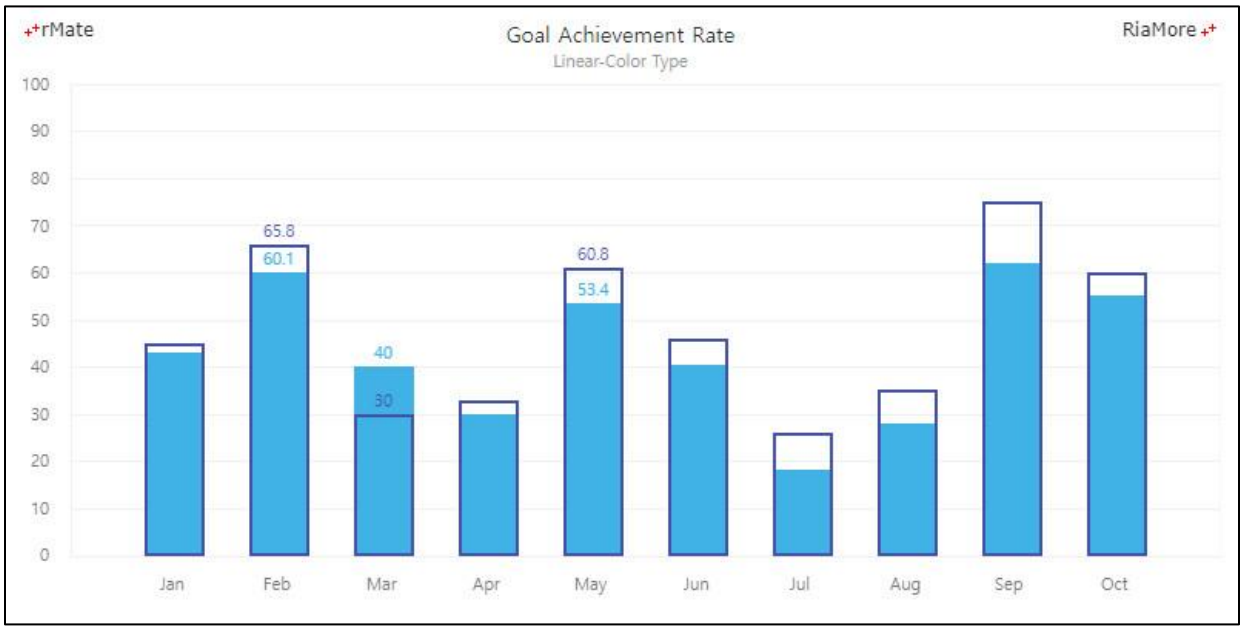


See the CodePen 알메이트 차트 - 목표 대비 실적 차트 - 2D 리니어

2D 리니어 목표 대비 실적 차트 - 컬러

목표와 실적을 표시하는 컬럼의 바탕과 테두리 선을 다르게 설정하고 목표 컬럼의 바탕색을 투명처리하여, 색상으로 목표와 실적이 구분되는 2D 리니어 유형의 목표 대비 실적 차트를 생성할 수 있습니다. 다음은 이를 적용해서 출력한 차트의 예제입니다.

```
<VTarget2DResultSeries labelPosition="outside" yField="Result" displayName="Result"
        color="#ff0000" showValueLabels="[1,2,4]">
  <fill>
    <SolidColor color="#ff0000"/>
  </fill>
  <stroke>
    <Stroke color="#ff0000" weight="2"/>
  </stroke>
</VTarget2DResultSeries>
<VTarget2DGoalSeries labelPosition="outside" yField="Goal" displayName="Goal"
        color="#000000" showValueLabels="[1,2,4]" itemRenderer="BoxItemRenderer">
  <fill>
    <SolidColor color="#000000" alpha="0"/>
  </fill>
  <stroke>
    <Stroke color="#000000" weight="2"/>
  </stroke>
</VTarget2DGoalSeries>
```



See the CodePen 알메이트 차트 - 목표 대비 실적 차트 - 2D 리니어 (컬러)

2D 리니어 목표 대비 실적 차트 – 오버레이

실적을 표시하는 컬럼의 넓이를 목표를 표시하는 컬럼의 넓이 보다 작게 설정하면 실적 컬럼이 목표 컬럼 안에 보이는 2D 리니어 유형의 목표 대비 실적 차트를 생성할 수 있습니다. 다음은 이를 적용해서 출력한 차트의 예제입니다.

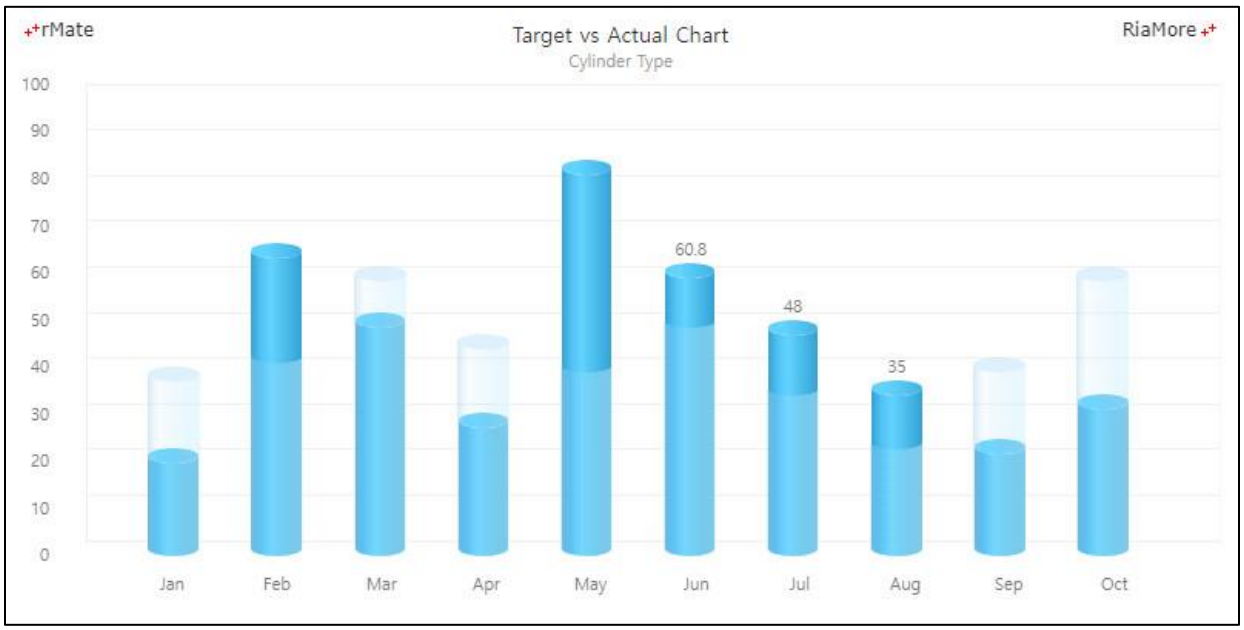
```
<VTarget2DResultSeries labelPosition="outside" yField="Result" displayName="Result"
        color="#f7921e" showValueLabels="[1,2,4]" columnWidthRatio="0.4">
  <fill>
    <SolidColor color="#f7921e"/>
  </fill>
</VTarget2DResultSeries>
<VTarget2DGoalSeries yField="Goal" displayName="Goal"
        itemRenderer="BoxItemRenderer">
  <fill>
    <SolidColor color="#29a3f6" alpha="0"/>
</fill>
</VTarget2DGoalSeries>
```

See the CodePen 알메이트 차트 - 목표 대비 실적 차트 - 2D 리니어 (오버레이)

3D 실린더 컬럼 목표 대비 실적 차트

다음은 3D 실린더 컬럼 유형의 목표 대비 실적 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Combination3DChart showDataTips="true">
  ...
  <series>
    <VTarget3DResultSeries yField="Result" displayName="Result"
       labelPosition="outside" showValueLabels="[5,6,7]" columnWidthRatio="0.5">
      ...
    </VTarget3DResultSeries>
    <VTarget3DGoalSeries yField="Goal" displayName="Goal" columnWidthRatio="0.52">
      ...
    </VTarget3DGoalSeries>
  </series>
</Combination3DChart>
```

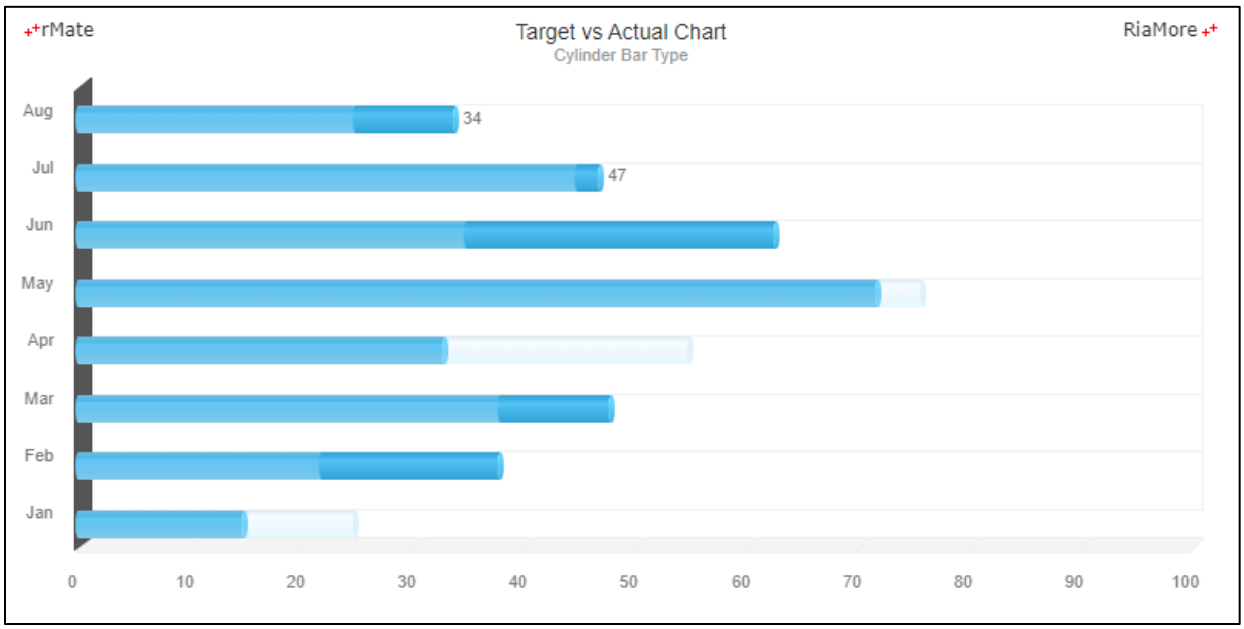


See the CodePen [알메이트 차트 - 목표 대비 실적 차트 - 3D 실린더 컬럼]

3D 실린더 바 목표 대비 실적 차트

다음은 3D 실린더 바 유형의 목표 대비 실적 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Combination3DChart showDataTips="true">
  ...
  <series>
    <HTarget3DResultSeries xField="Result" displayName="Result" barWidthRatio="0.5"
       labelPosition="outside" showValueLabels="[6,7]" outsideLabelXOffset="6"
       outsideLabelYOffset="-2">
      ...
    </HTarget3DResultSeries>
    <HTarget3DGoalSeries xField="Goal" displayName="Goal" barWidthRatio="0.52">
      ...
    </HTarget3DGoalSeries>
  </series>
  ...
</Combination3DChart>
```

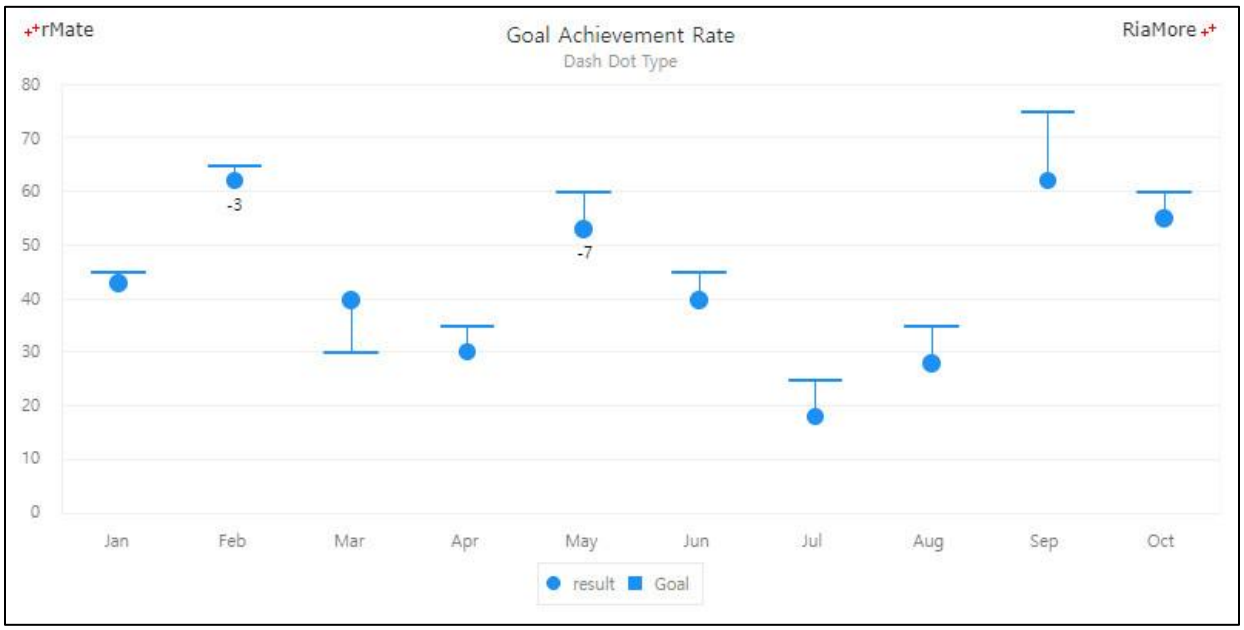


See the CodePen 알메이트 차트 - 목표 대비 실적 차트 - 3D 실린더 바

대시 앤 닷 목표 대비 실적 차트

대시 앤 닷 목표 대비 실적 차트는 <Column2DChart> 노드의 series 속성값에 <DashDotSeries>를 설정하여 생성합니다. 목표값은 <DashDotSeries> 노드의 goalField 속성에 설정하고 실적값은 <DashDotSeries> 노드의 resultField 속성에 설정합니다. 다음은 대시 앤 닷 목표 대비 실적 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

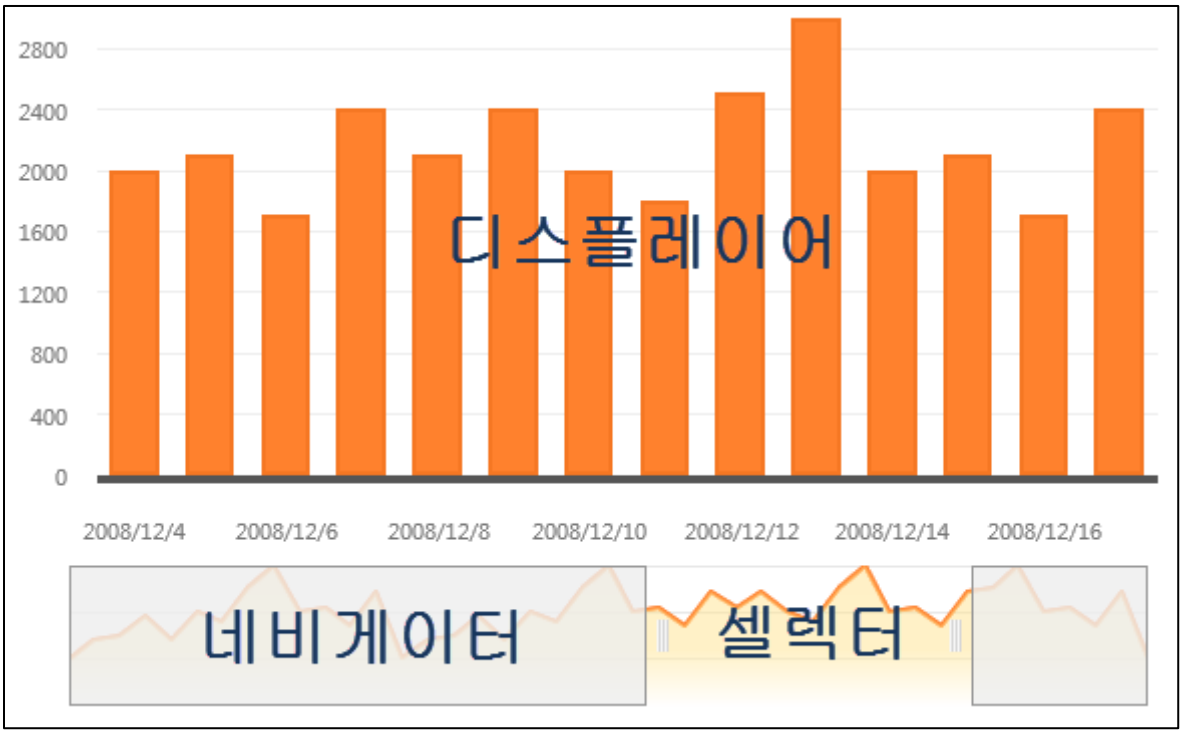
```
<Column2DChart showDataTips="true" columnWidthRatio="0.5"
        itemClickJsFunction="itemClick" >
  <series>
    <DashDotSeries labelPosition="outside" resultField="Result" goalField="Goal"
        goalDisplayName="Goal" resultDisplayName="result" color="#000000"
        showValueLabels="[1,4]" >
    </DashDotSeries>
  </series>
</Column2DChart>
```



See the CodePen 알메이트 차트 - 목표 대비 실적 차트 - 대시 앤 닷

4.12 히스토리 차트

히스토리 차트는 기본적으로 스크롤 차트와 같은 개념의 차트입니다. 두 차트의 차이점은 히스토리 차트에서는 조회를 원하는 데이터의 위치를 스크롤바를 통해서 지정하는 것이 아니라, 네비게이터와 셀렉터를 통해서 지정한다는 것입니다. 이 때 지정된 데이터들은 디스플레이어에 표시됩니다. 다음 그림은 히스토리 차트를 표현한 것입니다.



히스토리 차트의 레이아웃은 다음과 같은 구조로 작성됩니다.

```
<HistoryChart>
  <displayerChart>
    <Displayer id="chart1" showDataTips="true" fontFamily="Malgun Gothic">
      ...
    </Displayer>
  </displayerChart>
  <navigator>
    <Navigator id="navi" width="100%" height="80" gutterLeft="0" gutterRight="0"
       paddingLeft="30" paddingRight="10">
      ...
    </Navigator>
  </navigator>
  <selector>
    <HistoryRangeSelector width="100%" startingRange="center"
       visibleItemSize="30"/>
  </selector>
</HistoryChart>
```

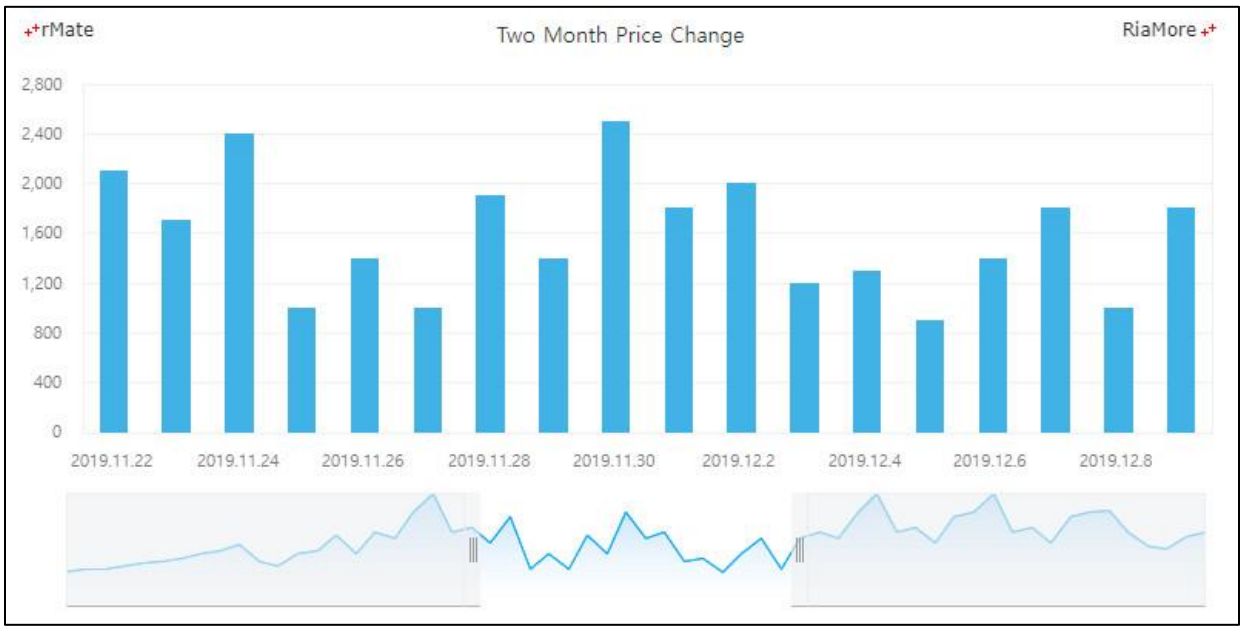
<Displayer> 노드와 <Navigator> 노드에 설정이 가능한 데이터 시리즈는 카테시안 차트의 데이터 시리즈입니다. 다음은 <HistoryChart> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
displayerChart	<Displayer>	히스토리 차트의 디스플레이어를 설정합니다.
navigator	<Navigator>	히스토리 차트의 네비게이터를 설정합니다.
selector	<HistoryRangeSelector>	히스토리 차트의 설렉터를 설정합니다.

<HistoryChart> 노드의 displayerChart 속성과 navigator 속성에 지정 가능한 유효값은 각각 <Displayer> 노드와 <Navigator> 노드입니다. <Displayer> 노드는 카테시안 차트 노드의 공통 속성을 모두 가지며 <Navigator> 노드는 <Area2DChart> 노드의 모든 속성을 가집니다.

다음은 디스플레이어에 컬럼 시리즈를 네비게이터에 영역 시리즈를 표현하는 히스토리 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<HistoryChart>
  <displayerChart>
    <Displayer id="chart1" showDataTips="true">
      ...
      <Column2DSeries id="columnSeries" yField="Data1" displayName="Data1">
      ...
    </Displayer>
  </displayerChart>
  <navigator>
    <Navigator id="navi" width="100%" height="80" gutterLeft="0" gutterRight="0"
       paddingLeft="30" paddingRight="10">
      ...
      <Area2DSeries name="A" yField="Data1">
      ...
    </Navigator>
  </navigator>
  <selector>
    <HistoryRangeSelector width="100%" startingRange="center"
       visibleItemSize="30"/>
  </selector>
</HistoryChart>
```



See the CodePen 알메이트 차트 - 히스토리 컬럼 차트

위 예제에서 <selector> 속성에 정의된 <HistoryRangeSelector> 노드의 주요 속성에 대한 설명은 아래와 같습니다.

속성명	유효값 (*: 기본값)	설명
liveDragging	true(*), false	셀렉터가 변경되면 변경된 데이터 영역을 디스플레이어 즉시 표시할지 여부를 설정합니다.
startingRange	left, center(*), right	차트가 처음 생성되었을 때 셀렉터에 선택될(디스플레이어에 표시될) 데이터 영역을 지정합니다.
visibleItemSize	숫자 기본값: 10	디스플레이어 차트에 표시할 데이터의 개수를 지정합니다.

셀렉터의 스타일은 rMateChartH5.css 파일에 설정되어 있고, 다음 디렉토리에서 찾으실 수 있습니다.

Installation directory/rMateChartH5/Assets/Css/

CSS 스타일의 내용은 다음과 같습니다.

```
.rMateH5__ChartRangeSelector_LeftThumb, .rMateH5__ChartRangeSelector_RightThumb{
  width:10px;
  height:100%;
  cursor:col-resize;
  border-bottom:solid 1px #b5b5b7;
  background:url("./selector.png") no-repeat 50% 50%;
  background-color:rgb(236, 240, 241);
  filter:progid:DXImageTransform.Microsoft.Alpha(opacity=60);
  background-color:rgba(236, 240, 241, 0.6);
}
.rMateH5__ChartRangeSelector_Left, .rMateH5__ChartRangeSelector_Right{
  border:solid 1px #eee;
  border-bottom:solid 1px #b5b5b7;
  background-color:rgb(236, 240, 241);
  filter:progid:DXImageTransform.Microsoft.Alpha(opacity=60);
  background-color:rgba(236, 240, 241, 0.6);
}
.rMateH5__ChartRangeSelector_Center{
  cursor:pointer;
  background-color:rgb(255, 255, 255);
  filter:progid:DXImageTransform.Microsoft.Alpha(opacity=0);
  background-color:rgba(255, 255, 255, 0);
}
```

4.13 실시간 차트

실시간 차트는 실시간으로 변동되는 데이터를 모니터링하기에 유용한 차트입니다. 실시간 차트는 <RealTimeChart> 노드를 정의하여 생성할 수 있으며, 적용 가능한 데이터 시리즈는<Column2DSeries>, <Column3DSeries>, <Line2DSeries>, <Area2DSeries> 입니다. 데이터는 원격 호출(RPC, Remote Procedure Call) 방식으로 서버에서 읽어와서 차트에 로드됩니다. 실시간 차트는 현재 화면에 어느 시점의 데이터가 표시되는지에 따라서 다음 두 유형으로 구분할 수 있습니다.

- 데이터의 개수: 서버나 기타 경로를 통하여 데이터를 받으면 차트는 화면 우측 끝에서부터 즉시 데이터를 표시하기 시작해서 displayDataSize 속성값에 설정된 수만큼 한 화면에 표시합니다. 이후에는 데이터가 하나씩 로드될 때마다 좌측 끝의 오래된 데이터는 쉬프트되어 사라집니다.
- 시간: 서버나 기타 경로를 통하여 데이터를 받으면 차트는 화면 우측 끝에서부터 즉시 데이터를 표시하기 시작해서 timePeriod 속성값에 설정된 시간 범위내에 있는 데이터들을 한 화면에 표시합니다. 이후에는 데이터가 하나씩 로드될 때마다 좌측 끝의 오래된 데이터는 쉬프트되어 사라집니다.

실시간 차트의 두 가지 유형에 대한 설정과 관련된 <RealTimeChart> 노드의 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
dataDisplayType	dataSize(*), time	차트의 화면에 표시되는 데이터 개수의 기준을 설정합니다. dataSize: displayDataSize 속성에 지정된 개수의 데이터를 화면에 표시합니다. time: timePeriod 속성에 지정된 시간 범위 내의 데이터를 표시합니다.
displayDataSize	숫자 기본값: 20	dataDisplayType 속성이 “dataSize” 일 경우, 화면에 표시할 데이터의 개수를 지정합니다.
interval	숫자 (second)	입력되는 데이터의 주기를 지정합니다.

		dataDisplayType 속성이 “time” 일 경우에만 유효합니다.
timePeriod	숫자 (second)	dataDisplayType 속성이 “time” 일 경우, 화면에 표시할 데이터의 시간 범위를 지정합니다.

원격 호출(RPC, Remote Procedure Call)에 의한 데이터 로드는 <HttpServiceRepeater> 노드에 의해서 이루어 집니다. 다음은 <HttpServiceRepeater> 노드 설정시 유의해야 할 사항입니다.

- <HttpServiceRepeater> 노드는 반드시 <RealTimeChart> 노드와 같은 레벨(형제 노드)에 정의해야 합니다.
- <HttpServiceRepeater> 노드의 target 속성값에 <RealTimeChart> 노드의 id 속성값(실시간 차트의 id)을 지정해야 합니다.
- <HttpServiceRepeater> 노드의 url 속성에는 차트의 데이터를 로드하는 URL 이 지정됩니다. 지정된 URL 로부터 읽어오는 데이터는 XML 데이터 형식이어야 하고 반드시 다음과 같은 형태로 작성되어야 합니다.

```
<?xml version="1.0" encoding="utf-8" ?>
<items> // XML 의 루트 노드는 <items> 이어야 합니다.
  <item> // 한 데이터 셋은 <items> 노드로 정의해야 합니다.
    <Time>13:8:27</Time> // 사용자가 원하는 형태로 작성합니다.
    <Volume>5527</Volume>
    <Price>309</Price>
  </item>
</items>
```

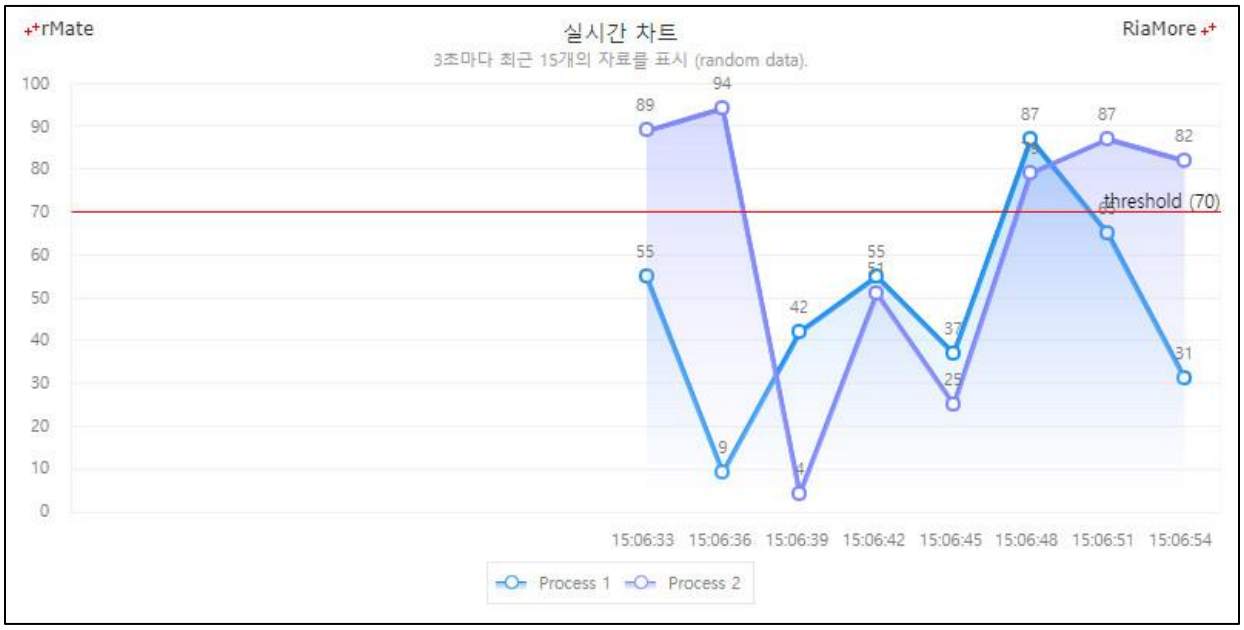
<HttpServiceRepeater> 노드에서 사용 가능한 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
interval	숫자 (second) default: 30	RPC 요청 주기(초 단위)를 지정합니다.
target	텍스트	<RealTimeChart> 노드의 id 속성값을 지정합니다.
url	텍스트	RPC 요청을 보낼 URL 을 지정합니다.
useFaultAlert	true(*), false	원격 호출 (RPC)이 실패할 경우 경고 메시지를 표시할 지 여부를 설정합니다.

데이터 개수 기준 실시간 차트

다음은 데이터 개수를 기준(dataDisplayType = “dataSize”)으로 설정한 실시간 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 화면에 표시되는 데이터의 개수를 15(displayDataSize = “15”)로 설정하였습니다. 그리고 실시간 차트에서는 setData() 함수를 호출하지 않고 예제에서와 같이 setLayout() 함수만 호출합니다.

```
function chartReadyHandler(id) {
  document.getElementById(id).setLayout(layoutStr);
}

<RealTimeChart id="chart" dataDisplayType="dataSize" displayDataSize="15"
       showDataTips="true">
  ...
  <Area2DSeries labelPosition="up" yField="P1" displayName="Process 1"
       itemRenderer="CircleItemRenderer"/>
  <Area2DSeries labelPosition="up" yField="P2" displayName="Process 2"
       itemRenderer="CircleItemRenderer"/>
  ...
</RealTimeChart>
<HttpServiceRepeater url="http://demo.riamore.net/chartTest/process3Data.jsp"
       target="{chart}" interval="3" method="get"/>
```

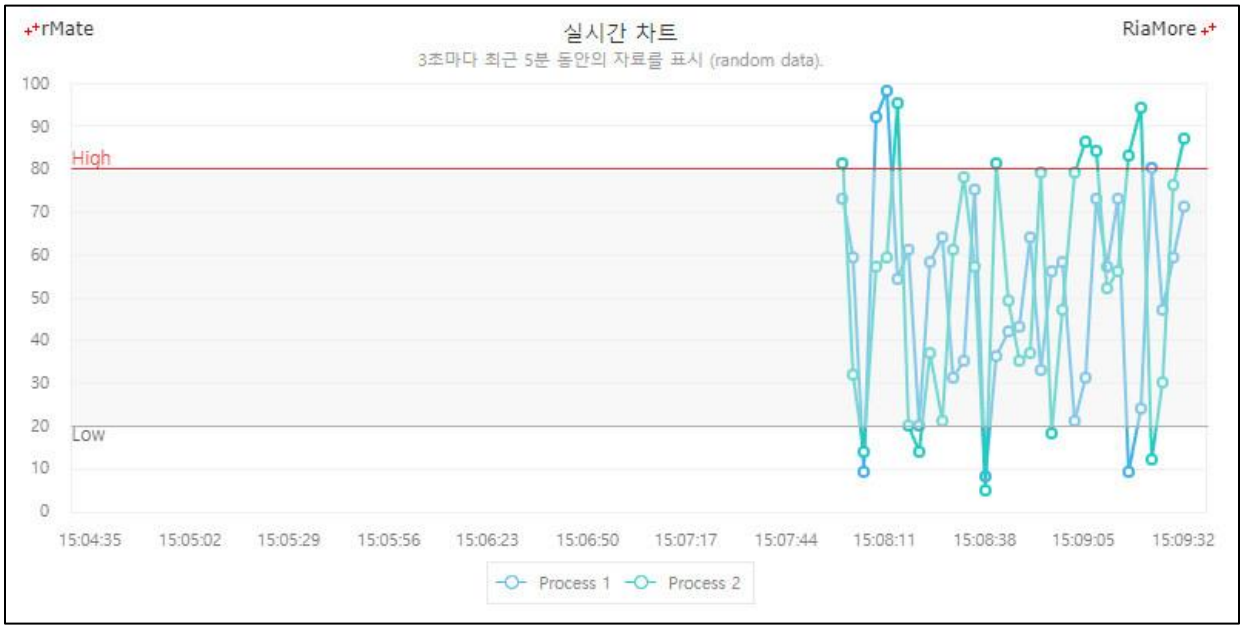


See the CodePen 알메이트 차트 - 데이터 개수 기준 실시간 차트

시간 기준 실시간 차트

다음은 시간을 기준(dataDisplayType = ”time”)으로 설정한 실시간 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 화면에 표시되는 데이터의 시간 범위를 300 초(timePeriod = “300”)로 설정하였습니다. 그리고 실시간 차트에서는 setData() 함수를 호출하지 않고 예제에서와 같이 setLayout() 함수만 호출합니다.

```
function chartReadyHandler(id) {
  document.getElementById(id).setLayout(layoutStr);
}

<RealTimeChart id="chart" dataDisplayType="time" timePeriod="300" interval="3"
       showDataTips="true">
  ...
  <Line2DSeries xField="Time" yField="P1" displayName="Process 1" radius="4"
       itemRenderer="CircleItemRenderer">
  ...
</RealTimeChart>
<HttpServiceRepeater url="http://demo.riamore.net/chartTest/process3Data.jsp"
       target="{chart}" interval="3" method="get"/
```



See the CodePen 알메이트 차트 - 시간 기준 실시간 차트

<HttpServiceRepeater>를 이용한 차트 자동 갱신

<HttpServiceRepeater> 노드를 항상 <RealTimeChart> 노드와 함께 적용할 필요는 없습니다. 일반적인 차트라도 일정한 시간 간격을 가지고 데이터를 주기적으로 갱신(Refresh)하고자 할 경우에는 <HttpServiceRepeater> 노드를 사용할 수 있습니다. 이 때 chart 노드의 id

속성값은 <HttpServiceRepeater> 노드의 target 속성값과 반드시 일치시켜야 합니다.
(<Column3DChart id="chart" ...>, <HttpServiceRepeater target="{chart}" ...>)
다음은 <Column3DChart> 노드와<HttpServiceRepeater> 노드를 함께 적용한 코드의 예제입니다.

```
<rMateChart cornerRadius="12" borderStyle="solid">
  <Column3DChart id="chart" showDataTips="true">
    <horizontalAxis>
      <CategoryAxis categoryField="Month" />
    </horizontalAxis>
    <series>
      <Column3DSeries yField="Profit" displayName="Profit">
        <showDataEffect>
          <SeriesInterpolate />
        </showDataEffect>
      </Column3DSeries>
    </series>
  </Column3DChart>
  <HttpServiceRepeater url="http://demo.riamore.net/chartTest/singleData.jsp"
       target="{chart}" interval="60"/>
</rMateChart>
```

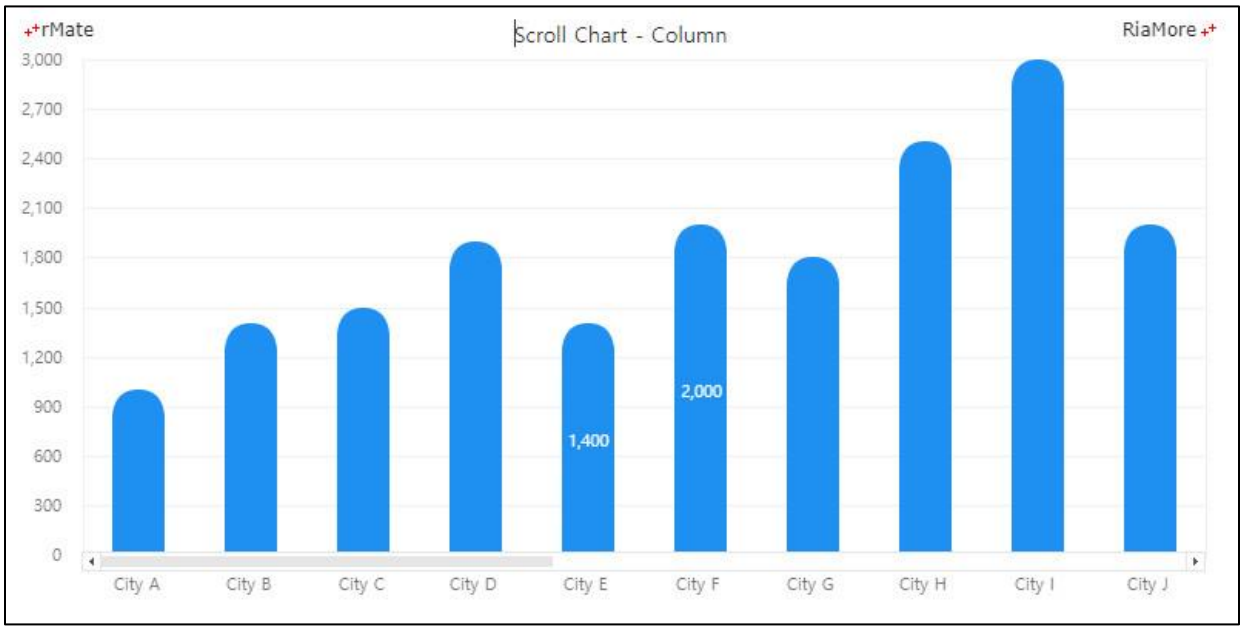
4.14 스크롤 차트

표현할 데이터의 수가 많은 경우, 지정된 개수(<ScrollableAxisRenderer> 노드의 visibleItemSize 속성값)의 데이터만 한 화면에 표시하고 나머지 데이터는 스크롤바 기능을 이용하여 조회가 가능하도록 하는 차트를 스크롤 차트라고 합니다. 스크롤 차트를 생성하기 위해서는 일반 차트를 생성하는 것과 동일한 방법으로 레이아웃을 작성하고 스크롤바가 위치할 축의 속성(<horizontalAxis>, <verticalAxis>)에 <CategoryLinearAxis> 노드를 설정하고, 축 렌더러 속성(<horizontalAxisRenderers>, <verticallAxisRenderers>)에 <ScrollableAxisRenderer> 노드를 설정합니다.<CategoryLinearAxis> 노드는 <LinearAxis> 노드 클래스를 상속받은 클래스이기 때문에 <LinearAxis> 노드에서 사용하는 모든 속성을 사용할 수 있습니다.
<ScrollableAxisRenderer> 노드에서 사용 가능한 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
arrowScrollSize	숫자 기본값: 1	사용자가 스크롤바의 화살표를 클릭했을 때 이동할 스크롤 크기를 지정합니다.
scrollBarPlacement	left(*), right, top, bottom	차트에서 스크롤바의 위치틑 지정합니다.
scrollPosition	숫자 기본값: 0	스크롤바의 시작 위치를 지정합니다.
trackScrollSize	숫자 기본값: visibleItemSize 속성값	사용자가 스크롤바의 트랙을 클릭했을 때 이동할 스크롤 크기를 지정합니다.
visibleItemSize	숫자 기본값: 5	차트에 표시할 데이터의 개수를 지정합니다.

다음은 컬럼 차트에 스크롤 차트 기능을 적용하는 코드와 출력된 결과입니다.

```
<Column2DChart showDataTips="true" gutterRight="10" columnWidthRatio="0.5">
  <verticalAxis>
    <LinearAxis interval="300" formatter="{numfmt}"/>
  </verticalAxis>
  <horizontalAxis>
    <CategoryLinearAxis id="hAxis" categoryField="City"/>
  </horizontalAxis>
  <horizontalAxisRenderers>
    <ScrollableAxisRenderer axis="{hAxis}" visibleItemSize="10" />
  </horizontalAxisRenderers>
  <series>
    <Column2DSeries labelPosition="inside" color="#ffffff" yField="Data1"
       displayName="Data1" itemRenderer="SemiCircleColumnItemRenderer"
       showValueLabels="[4,5]"/>
  </series>
</Column2DChart>
```



See the CodePen 알메이트 차트 - 스크롤 가능한 컬럼 차트

위 예제에서는 한 화면에 표현되는 데이터의 개수를 10 (visibleItemSize = “10”)으로 설정했습니다.

스크롤바의 스타일은 rMateChartH5.css 파일에 설정되어 있고, 다음 디렉토리에서 찾으실 수 있습니다.

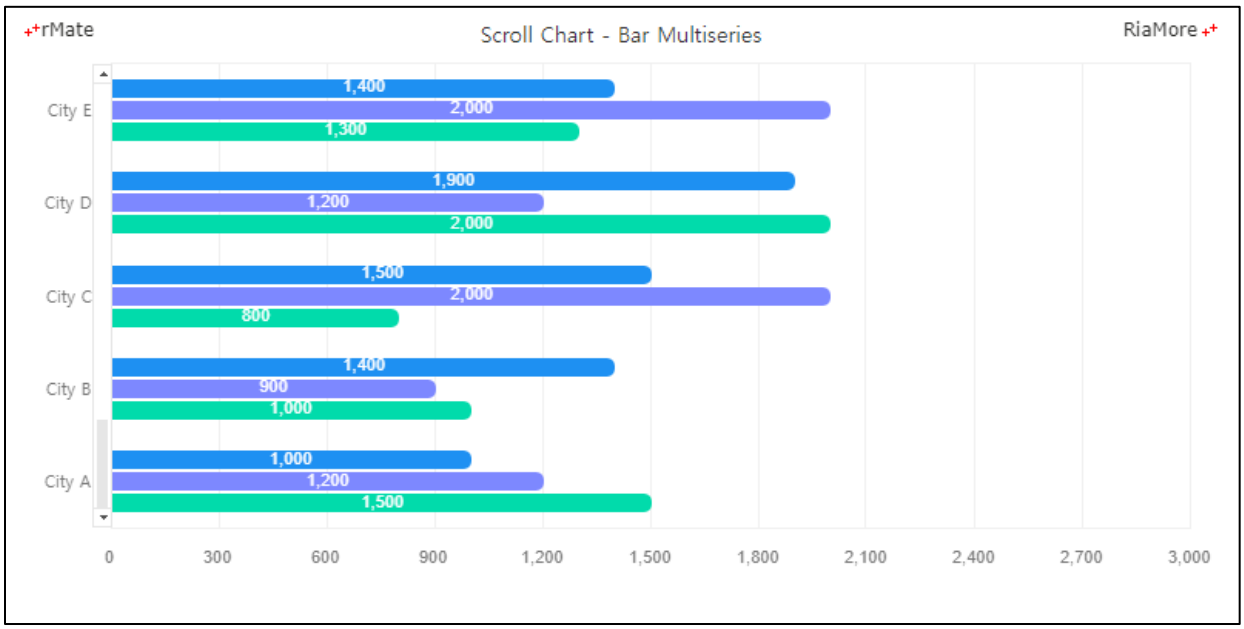
설치 디렉토리/rMateChartH5/Assets/Css/

CSS 스타일의 내용은 다음과 같습니다.

```
.rMate__ScrollBar {
  background-color:#fff;
}
.rMate__HScrollTrack, .rMate__VScrollTrack {
  border:1px solid #e6e6e6;
}
.rMate__HScrollThumb, .rMate__VScrollThumb {
  cursor:pointer;
  background-color:#e6e6e6;
  transition:background-color 0.2s;
}
.rMate__HScrollThumb:hover, .rMate__VScrollThumb:hover{
background-color:#888888;
}
  .rMate__HScrollThumbHeader{
}
.rMate__HScrollUpArrow {
  border:1px solid #e6e6e6;
  cursor:pointer;
  background:url("./scroll_left_arrow.png") no-repeat 50% 50%;
}
.rMate__HScrollDownArrow {
  border:1px solid #e6e6e6;
  cursor:pointer;
  background:url("./scroll_right_arrow.png") no-repeat 50% 50%;
}
.rMate__VScrollThumbHeader{
}
.rMate__VScrollUpArrow {
  border:1px solid #e6e6e6;
  border-right:none;
  cursor:pointer;
  background:url("./scroll_up_arrow.png") no-repeat 50% 50%;
}
.rMate__VScrollDownArrow {
  border:1px solid #e6e6e6;
  border-right:none;
  cursor:pointer;
  background:url("./scroll_down_arrow.png") no-repeat 50% 50%;
}
```

다음은 바 차트에 세로 스크롤바를 표시하는 코드와 출력된 결과입니다.

```
<Bar2DChart showDataTips="true" gutterRight="20" gutterTop="10" gutterBottom="50"
       gutterLeft="60" barWidthRatio="0.7">
  <horizontalAxis>
    <LinearAxis interval="300" formatter="{numfmt}"/>
  </horizontalAxis>
  <verticalAxis>
    <CategoryLinearAxis id="hAxis" categoryField="City" ticksBetweenLabels="true"/>
  </verticalAxis>
  <verticalAxisRenderers>
    <ScrollableAxisRenderer axis="{hAxis}" visibleItemSize="5"/>
  </verticalAxisRenderers>
  <series>
    <Bar2DSeries ... />
    <Bar2DSeries ... />
    <Bar2DSeries ... />
  </series>
</Bar2DChart>
```



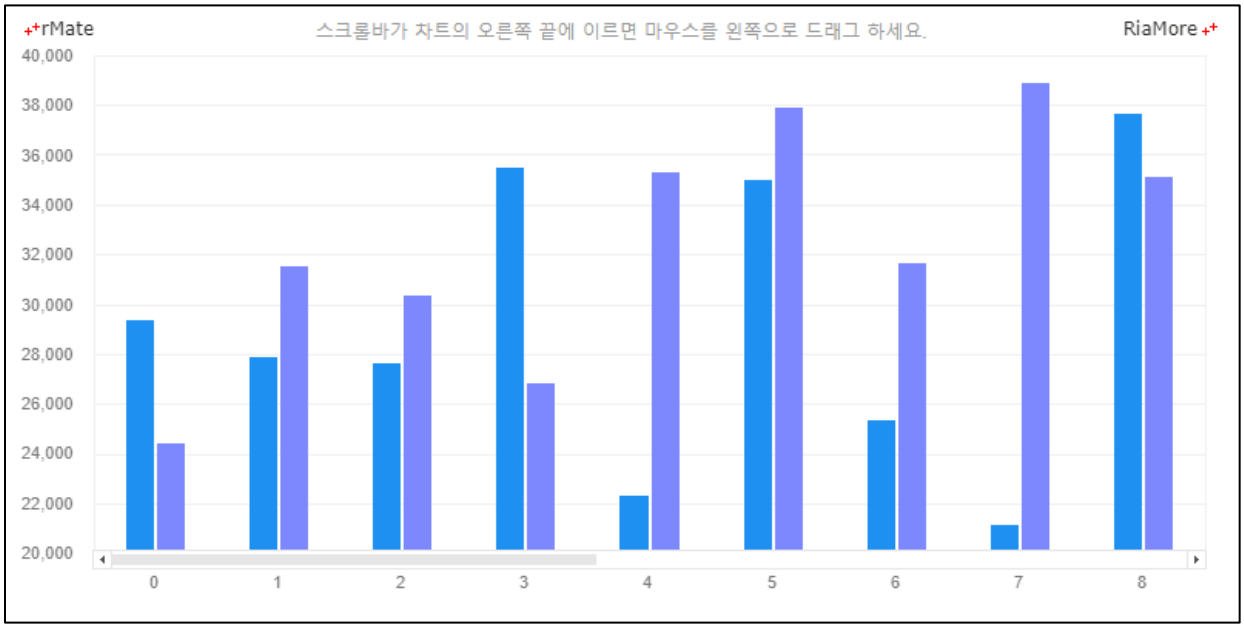
See the CodePen 알메이트 차트 - 스크롤 가능한 바 차트

레이지 로드

스크롤 차트에서 전체 데이터를 차트가 생성되는 시점에 한번에 로드한다면, 데이터를 처리하는 시간이 많이 소요될 것이고 결과적으로 차트가 화면에 표현되기까지 사용자가 오랜 시간을 기다려야하는 문제가 생길 수 있습니다. 어짜피 한 화면에서 사용자가 식별할 수 있는 데이터의 수는 한정되어 있기 때문에, 한 화면에 표현 하기 적당한 데이터의 수 단위로 처리를 하는 것이 효율적일 것입니다. 그러기 위해서는 스크롤바의 위치가 끝 부분에 도달하면 자동으로 새로운 자료를 로드하는 것이 필요한데, 이렇게 데이터를 처리하는 방식을 레이지 로딩(Lazy Loading)이라고 합니다.
알메이트 차트에서는 스크롤바의 이벤트를 받아서 레이지 로딩을 처리할 수 있도록하는 lazyJsFunction 속성을 제공합니다.

다음은 컬럼 차트의 한 화면에 19 개의 데이터를 표현한다고 가정할 때 레이지 로딩을 처리하기 위한 레이아웃과 코드입니다.

```
<Column2DChart showDataTips="true" gutterRight="10" lazyJsFunction="lazyDataFunc">
  ...
</Column2DChart>

var xhr, // Ajax object
index = 19; // 한 화면에 19 개의 데이터를 표현
function lazyDataFunc(id){
  var param = {};
  param.url = dataURL + (index + 1);
  param.success = function() {
    var data;
    if (xhr.readyState == 4 && xhr.status >= 200 && xhr.status < 300) {
      if (xhr.responseXML.xml)
        data = xhr.responseXML.xml;
      else
        data = xhr.responseXML;
      document.getElementById(id).addData(data);
      index += 20;
    }
  }
  ajax(param);
}
```



See the CodePen 알메이트 차트 - 스크롤 가능한 컬럼 차트에서 레이지 로드

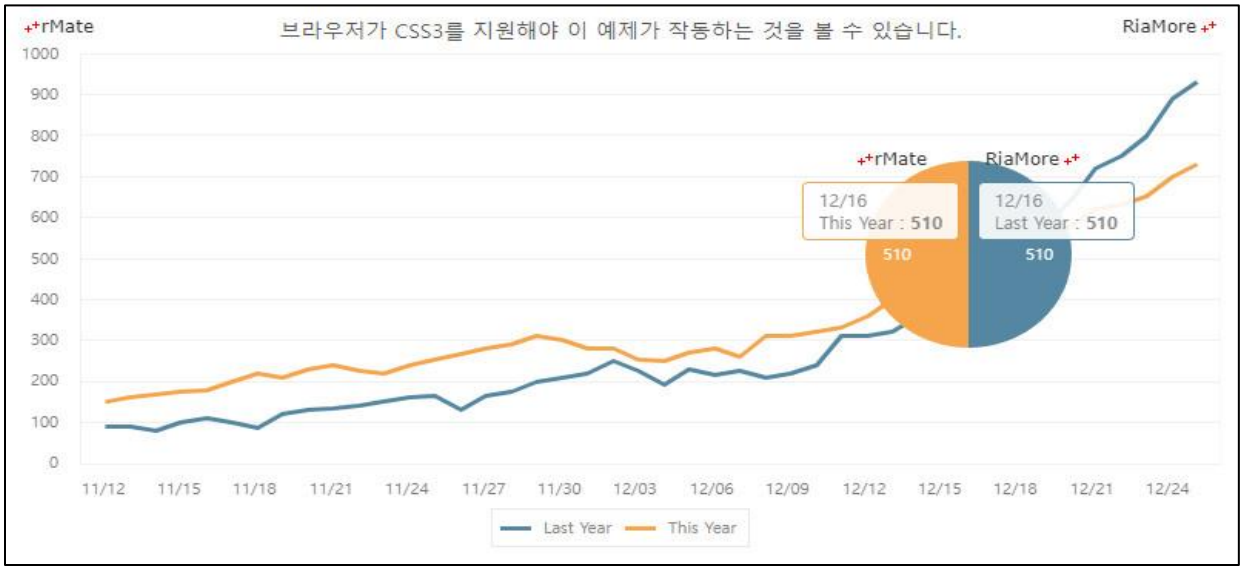
4.15 이벤트 차트

이벤트 차트는 독립적인 차트 유형이 아니라 차트의 데이터가 미리 정의된 특정 조건을 만족하는 경우, 해당 데이터 포인트에 이미지 표현 혹은 애니메이션 효과를 통해서 사용자에게 높은 시각적 정보를 제공하는 기능이라고 할 수 있습니다. 이벤트 차트 기능을 구현하려면 데이터 시리즈 노드(<Line2DSeries>, <Equalizer2DSeries>, 등)의 htmlJsFunction 속성에 이벤트 처리를 실행하는 자바스트립트 함수명을 지정해야 합니다.

데이터 포인트에 차트 표시

다음은 라인 차트에서 금년도 값이 전년도 값보다 같거나 커지는 지점(데이터 포인트)에 애니메이션 효과를 표현하고, 사용자가 해당 지점(데이터 포인트)에 마우스 오버를 하면 파이 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DSeries xField="Date" yField="Data2" htmlJsFunction="userFunction"
        displayName="Last Year">

function userFunction(id, index, data, values) {
  ...
  if(past <= current) {
    if (document.getElementsByClassName("chart_animate_element").length > 0)
      return;
    return {
      content : "",
      className : "chart_animate_element",
      events : {
        "mouseover" : function(event){
          if(currentPointChartId.length > 0)
            return;
          currentPointElem = event.target;
          var target = event.currentTarget;
          var div = document.createElement("div");
          var chartId;
          div.id = "tooltip_chartHolder_" + Math.floor(Math.random() * 1000);
          div.className = "tooltip_chart";
          target.appendChild(div);
          chartId = div.id.replace("Holder", "");
          currentPointChartId = chartId;
          rMateChartH5.create(chartId, div.id, "", "100%", "100%", function(id){
            document.getElementById(id).setLayout(layoutStr2);
            document.getElementById(id).setData(chartData2);
          });
        },
        "mouseout" : function(event){
          ...
        }
      }
    };
  }
  return;
}
```



See the CodePen [알메이트 차트 - 데이터 포인트에 차트 표시]

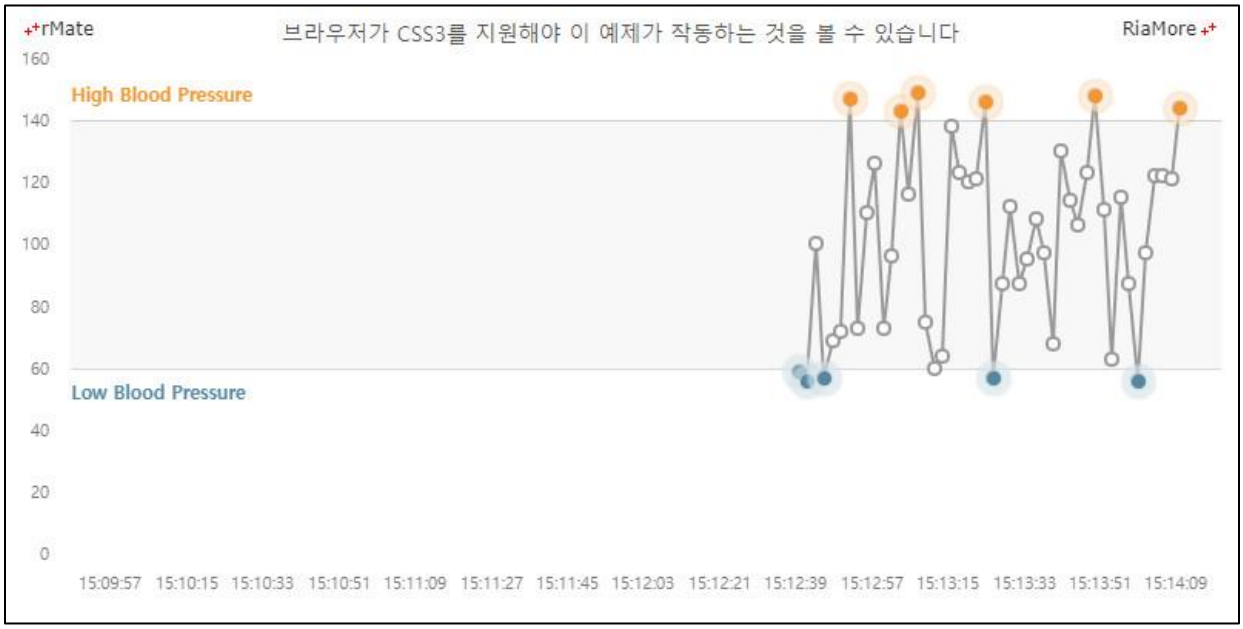
위 예제에서는 애니메이션 효과를 표현하기 위해서 해당 데이터 포인트에 CSS 스타일 (chart_animate_element)을 적용하였습니다. 다음은 적용된 CSS 스타일의 소스 코드입니다.

```
.chart_animate_element{
  width:12px;
  height:12px;
  border-radius:100%;
  box-sizing:border-box;
  border-color:#20ccc1;
  background-color:#20ccc1;
  animation:animate 2s 0s ease-out infinite
}

@keyframes animate{
  0%{box-shadow:0 0 4px 3px rgba(32, 204, 193, 0), 0 0 0px 0px rgba(255, 255, 255,
       0.5), 0 0 0px 0px rgba(32, 204, 193, 0);}
  10%{box-shadow:0 0 4px 3px rgba(32, 204, 193, 0.5), 0 0 6px 5px rgba(255, 255,
       255, 0.5), 0 0 6px 7px rgba(32, 204, 193, 0.5);}
  100%{box-shadow:0 0 4px 3px rgba(32, 204, 193, 0), 0 0 0px 20px rgba(255, 255,
       255, 0.5), 0 0 0px 20px rgba(32, 204, 193, 0);}
}

@-webkit-keyframes animate{
  0%{box-shadow:0 0 4px 3px rgba(32, 204, 193, 0), 0 0 0px 0px rgba(255, 255, 255,
       0.5), 0 0 0px 0px rgba(32, 204, 193, 0);}
  10%{box-shadow:0 0 4px 3px rgba(32, 204, 193, 0.5), 0 0 6px 5px rgba(255, 255,
       255, 0.5), 0 0 6px 7px rgba(32, 204, 193, 0.5);}
  100%{box-shadow:0 0 4px 3px rgba(32, 204, 193, 0), 0 0 0px 20px rgba(255, 255,
       255, 0.5), 0 0 0px 20px rgba(32, 204, 193, 0);}
}
```

데이터 포인트에 경고 표시

다음은 혈압을 체크하는 실시간 차트에서 미리 정해진 범위의 혈압 수치를 벗어날 경우(고혈압, 저혈압), 해당 데이터 포인트에 경고 표시를 하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DSeries xField="Time" yField="Data" displayName="Process 1"
        htmlJsFunction="userFunction" itemRenderer="CircleItemRenderer">

function userFunction(id, index, data, values){
  var className = "high",
  value = values[1];
  if(value < minValue || value > maxValue){
    if(value < minValue)
      className = "low";
    return {
      content : "",
      className : "odd_pressure " + className + "_blood_pressure",
      events : {
        "click" : (function(a){
          return function(event){
            event.target.parentNode.removeChild(event.target);
            alert("Time: " + a.Time + "\nBlood Pressure: " + a.Data);
          };
        })(data)
      }
    };
  }
  return;
}
```

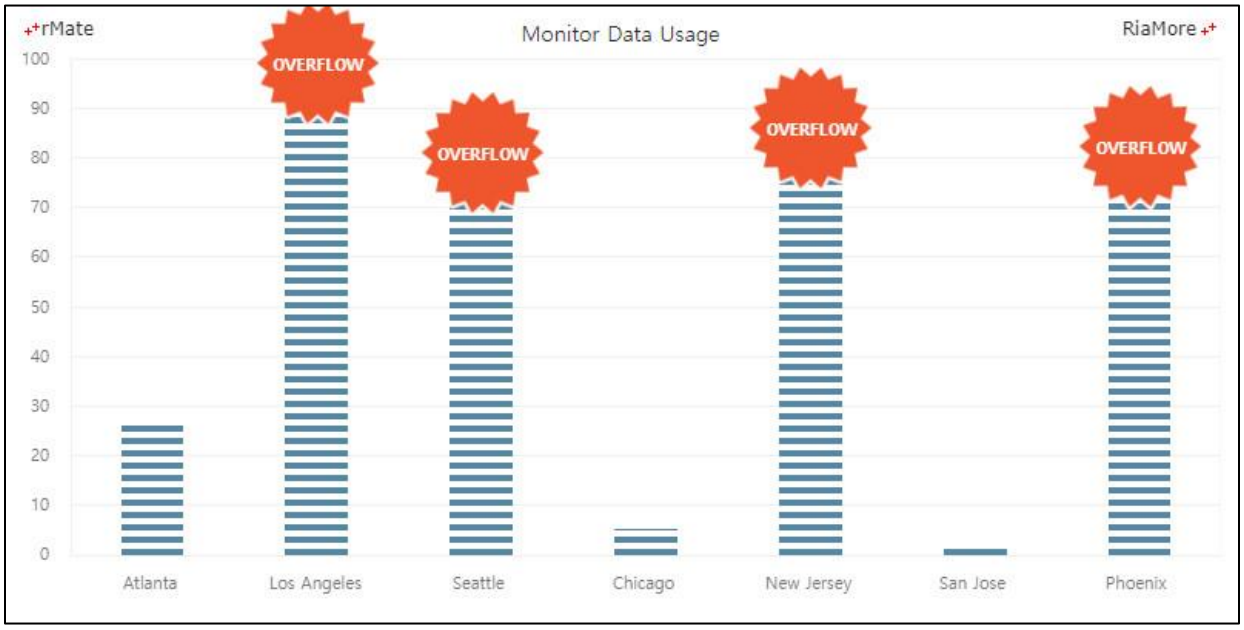


See the CodePen 알메이트 차트 - 데이터 포인트에 경고 표시

위 예제에서는 경고 표시를 하기 위해서 해당 데이터 포인트에 CSS 스타일 ("odd_pressure " + className + "_blood_pressure")을 적용하였고, 경고가 표시된 데이터 포인트에 마우스를 클릭할 경우 혈압 수치를 메시지로 표시합니다.

다음은 데이터 사용량을 표시하는 이퀄라이저 차트에서 사용량이 미리 정의한 값 (“70”) 보다 클 경우 해당 데이터 포인트에 오버플로우 경고 표시를 하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Equalizer2DSeries yField="value" showDataEffect="{ss}"
        htmlJsFunction="userFunction">

function userFunction(id, index, data, values){
  var value = values[1];
  if (value > 70) {
    return {
      content : "OVERFLOW",
      className : "data_overflow"
    };
  }
  return;
}
```

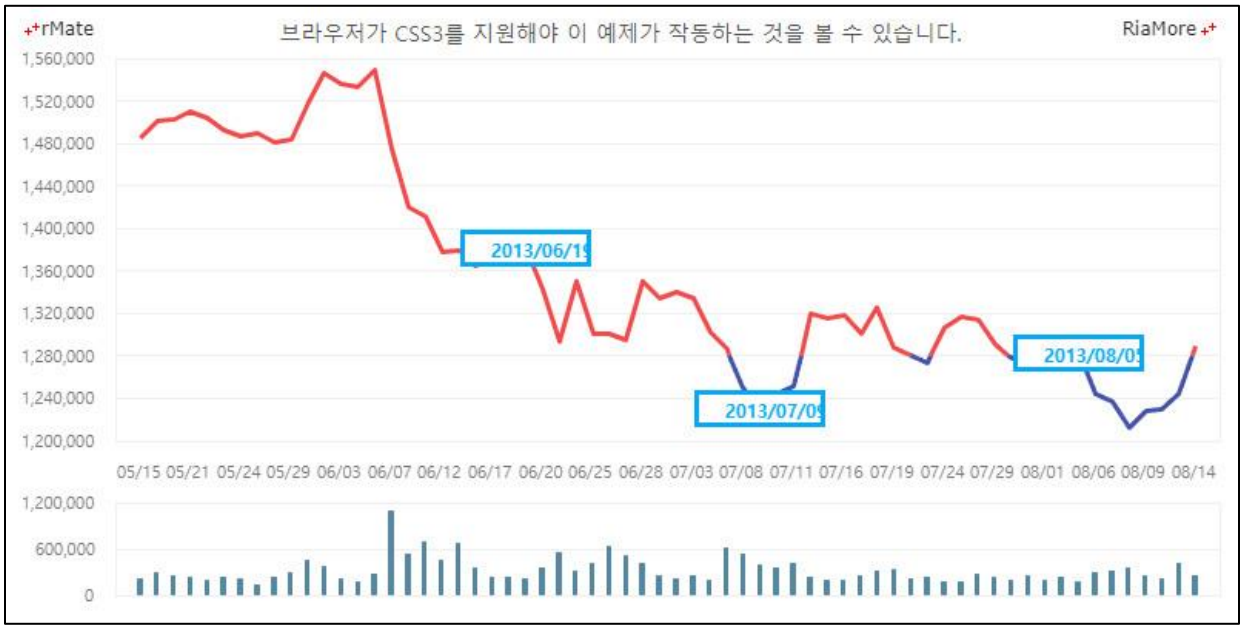


See the CodePen 알메이트 차트 - 이퀄라이저 차트에서 경고 표시

데이터 포인트에 롤링 텍스트 표시

다음은 캔들라인 차트에서 특정 일(X 축 값)의 데이터 포인트에 롤링 텍스트를 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<CandleLine2DSeries yField="open" htmlJsFunction="userFunction">

function userFunction(id, index, data, values){
  ...
  if (xValue == "2013/06/19")
    rollDiv = getRollDiv(xValue, "Interest rate hikes", "Stock collapse");
  else if(xValue == "2013/07/09")
    rollDiv = getRollDiv(xValue, "STOCK STIMULUS", "10% Rise");
  else if(xValue == "2013/08/05")
  rollDiv = getRollDiv(xValue, "Continuous interest rate hikes", "Stock collapse");
  if (rollDiv)
    return {
      content : rollDiv,
      className : "roll_wrapper"
    };
}

function getRollDiv(date, str1, str2){
  ...
  wrapDiv.appendChild(dateDiv);
  wrapDiv.appendChild(contentDiv1);
  wrapDiv.appendChild(contentDiv2);
  return wrapDiv;
}
```



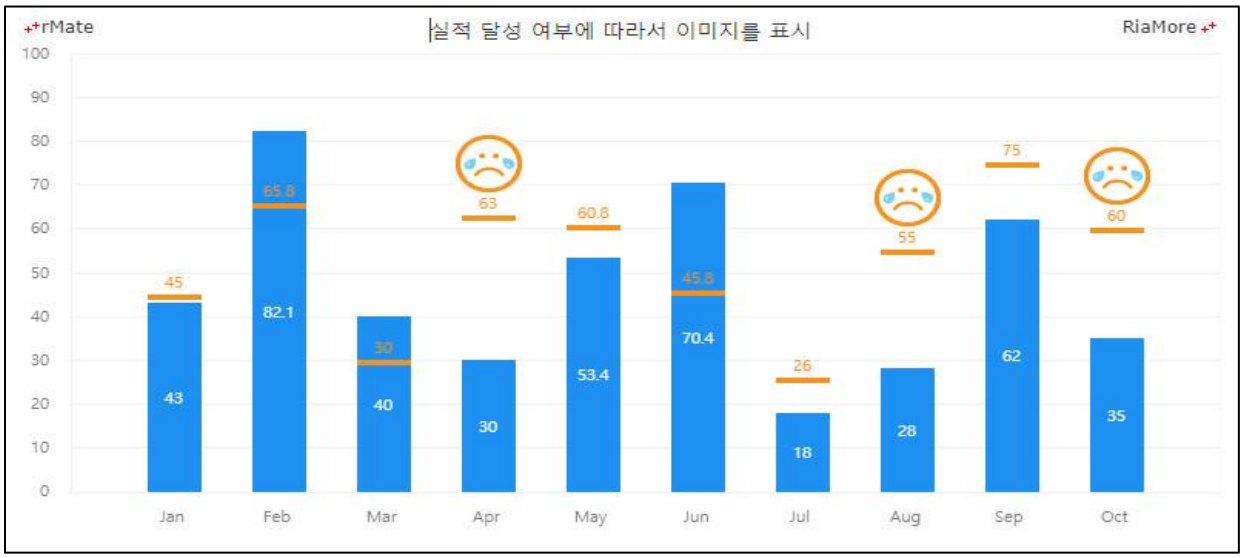
See the CodePen 알메이트 차트 - 데이터 포인트에 롤링 텍스트 표시

위 예제에서는 특정 일의 데이터 포인트에 div 를 생성하고 해당 div 에 CSS 스타일(wrapDiv, dateDiv, contentDiv1, contentDiv2)을 적용하여 롤링 텍스트를 표시합니다.

데이터 포인트에 애니메이션 GIF 표시

특정 조건을 만족하는 데이터 포인트에 애니메이션 GIF 를 표시할 수도 있습니다. 다음은 애니메이션 GIF 가 표시될 div 를 생성하고 CSS 스타일을 적용하여 특정 데이터 포인트에 애니메이션 GIF 를 표시하는 두 가지 예제입니다.

See the CodePen 알메이트 차트 - 컬럼 차트에서 애니메이션 GIF 표시

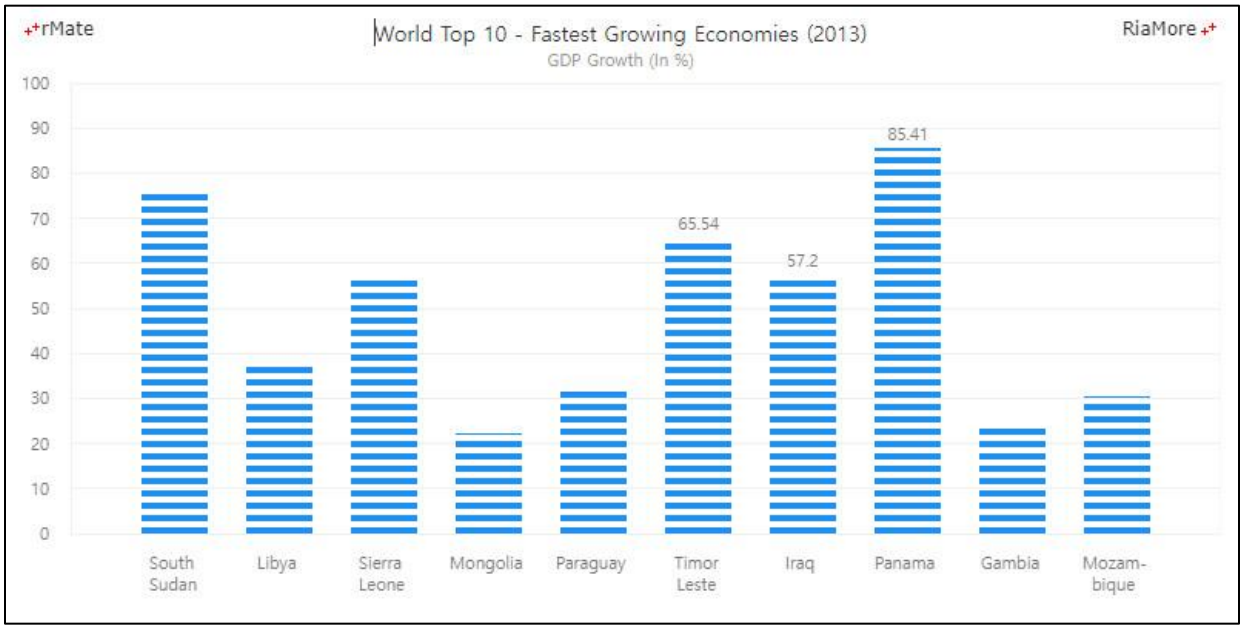


See the CodePen 알메이트 차트 - 데이터 포인트에 애니메이션 GIF 표시

4.16 이퀄라이저 차트

이퀄라이저 차트의 숫자 축은 차트의 왼쪽에 세로 축(Y 축)으로 표시되고, 카테고리 축은 차트의 하단에 가로 축(X 축)으로 표시되며, 데이터의 크기는 세로 막대의 길이로 표현된다는 면에서 컬럼 차트와 동일하지만, 표현되는 세로 막대는 이퀄라이저 모양을 한 여러 개의 작은 막대로 이루어집니다. 데이터 값의 크기에 따라서 이퀄라이저 모양을 한 작은 막대의 개수가 정해지고, 막대의 넓이는 컬럼 차트에서 적용되는 속성을 그대로 이용하여 조절할 수 있습니다. 이퀄라이저 차트는 <Equalizer2DChart> 노드의 series 속성값에 <Equalizer2DSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 이퀄라이저 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Equalizer2DChart showDataTips="true" selectionMode="single"
       columnWidthRatio="0.65">
...
  <series>
    <Equalizer2DSeries labelPosition="outside" yField="GDP" displayName="GDP Growth
       (In %)" showDataEffect="{ss}" showValueLabels="[5,6,7]" />
  </series>
</Equalizer2DChart>
```

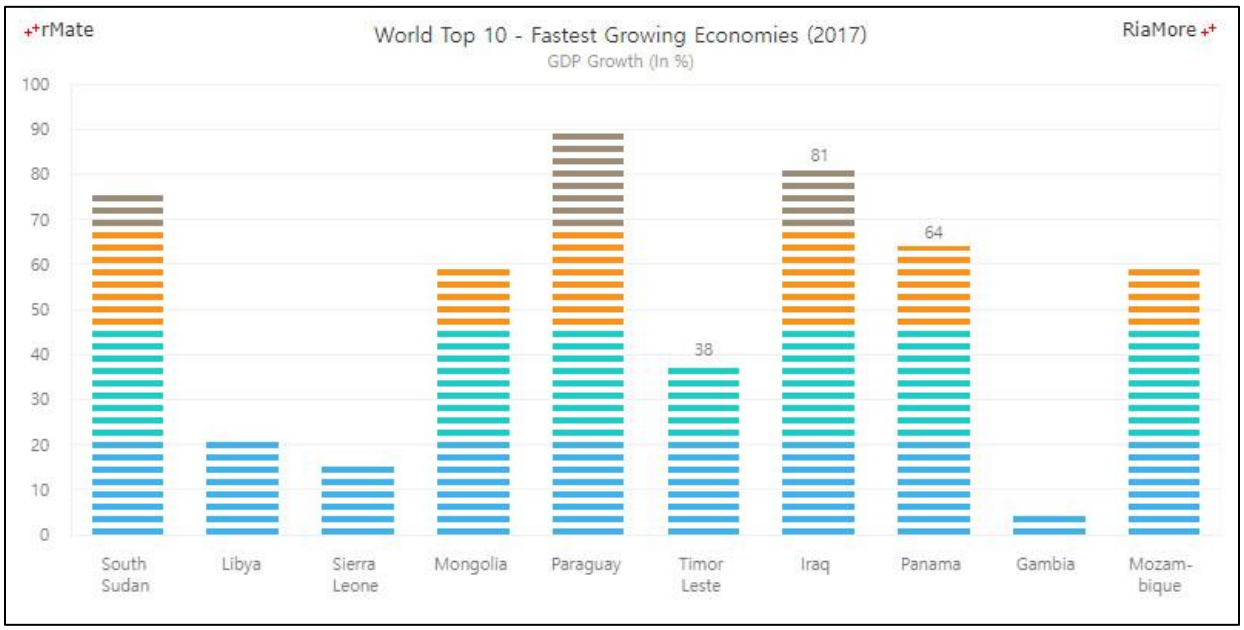


See the CodePen 알메이트 차트 - 이퀄라이저 차트

이퀄라이저 막대 색 설정

하나의 데이터 포인트를 표현하는 이퀄라이저 모양을 한 작은 막대들에 대해서 서로 다른 색을 설정할 수 있습니다. 다음은 사용자 정의 함수를 이용하여 데이터 값의 범위에 따라서 작은 막대에 적용되는 색을 서로 다르게 설정한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 2000 밀리세컨드마다 데이터를 동적으로 변경시키고, 변경된 데이터 값에 따라서 차트를 다시 표현하기 위해서 changeData(id) 함수가 호출되었습니다.

```
<Equalizer2DChart ...>
  ...
  <Equalizer2DSeries labelPosition="outside" yField="GDP" displayName="GDP Growth
       (In %)" fillJsFunction="fillJsFunc" showDataEffect="{ss}"
       showValueLabels="[5,6,7]"/>
</Equalizer2DChart>

function fillJsFunc (id, index, data, values){
  var value = values[1];
  if (value < 23)
    return "#40b2e6";
  else if (value < 46)
    return "#20cbc2";
  else if (value < 69)
    return "#f7921c";
  else
  return "#9b8c77";
}

function changeData(id){
  setTimeout(function(){
    var i = 0, obj,
    tempChartData = [],
    n = chartData.length;
    for ( ; i < n ; i += 1) {
      obj = {};
      obj.Country = chartData[i].Country;
      obj.GDP = Math.floor(Math.random() * 100);
      tempChartData.push(obj);
    }
    chartData = tempChartData;
    document.getElementById("chart1").setData(chartData);
    changeData();
  }, 2000);
}
```



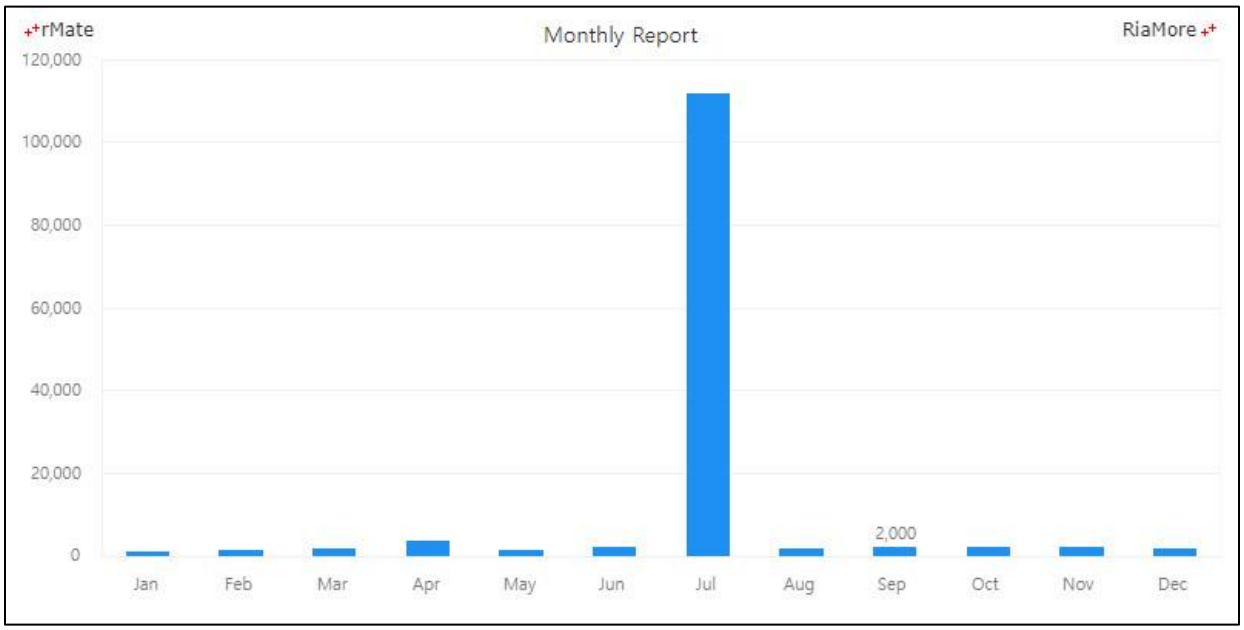
See the CodePen 알메이트 차트 - 이퀄라이저 막대 색 설정

4.17 브로큰 축 차트

브로큰 축 차트는 데이터 시리즈를 구성하는 일부 데이터 값이 대부분의 데이터 값들과 편차가 심할 경우 활용됩니다. 데이터 값의 편차가 심한 경우에 일반적인 차트(컬럼 차트, 라인 차트 등)를 적용하면 편차가 심한 특정 데이터 때문에 나머지 대부분의 데이터들을 비교하기 어렵게 됩니다.

다음 예를 보십시요. 6 월(Jun)을 제외한 나머지 데이터는 3,000 부근의 값이지만 6 월(Jun)의 데이터 값은 110,000 을 넘습니다. 결과적으로 생성된 컬럼 차트에서는 6 월을 제외한 나머지 달들의 데이터 값을 한눈에 비교할 수 없습니다.

```
<Column2DChart showDataTips="true" columnWidthRatio="0.5">
  <horizontalAxis>
    <CategoryAxis categoryField="Month"/>
  </horizontalAxis>
  <verticalAxis>
    <LinearAxis id="vAxis" formatter="{numfmt}" />
  </verticalAxis>
  <series>
    <Column2DSeries yField="Profit" labelPosition="outside" displayName="Profit"
       showValueLabels="[8]">
      ...
    </Column2DSeries>
  </series>
</Column2DChart>
```

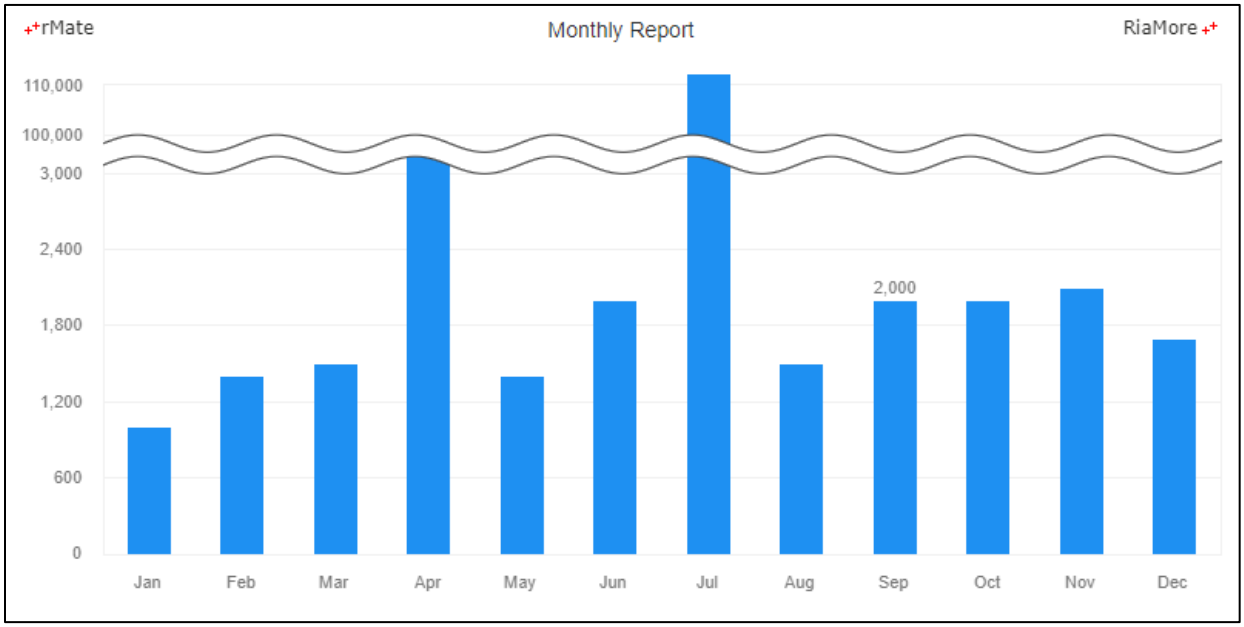


See the CodePen 알메이트 차트 - 컬럼 차트 - 데이터 값의 편차가 심한 경우

브로큰 축 컬럼 차트

브로큰 축 컬럼 차트를 생성하기 위해서는 컬럼 차트 생성을 위한 레이아웃의 <verticalAxis> 속성에 <BrokenAxis> 노드를 정의합니다. 위 예에서 사용된 데이터 시리즈를 그대로 적용한 브로큰 축 컬럼 차트는 아래와 같습니다.

```
<Column2DChart showDataTips="true" columnWidthRatio="0.5">
  <horizontalAxis>
    <CategoryAxis categoryField="Month"/>
  </horizontalAxis>
  <verticalAxis>
    <BrokenAxis id="vAxis" brokenMinimum="3000" brokenMaximum="100000"
       maximum="116000" brokenRatio="0.8" formatter="{numfmt}" />
  </verticalAxis>
  <verticalAxisRenderers>
    <BrokenAxis2DRenderer axis="{vAxis}"/>
  </verticalAxisRenderers>
  <series>
    <Column2DSeries yField="Profit" labelPosition="outside" displayName="Profit"
       showValueLabels="[8]">
      ...
    </Column2DSeries>
  </series>
</Column2DChart>
```



See the CodePen 알메이트 차트 - 브로큰 축 컬럼 차트

주의

브로큰 축 차트는 컬럼 차트, 바 차트, 라인 차트, 영역 차트에서만 사용할 수 있습니다.

주의

브로큰 축 차트에서는 <HistoryChart> 노드(히스토리 차트)가 지원되지 않습니다.

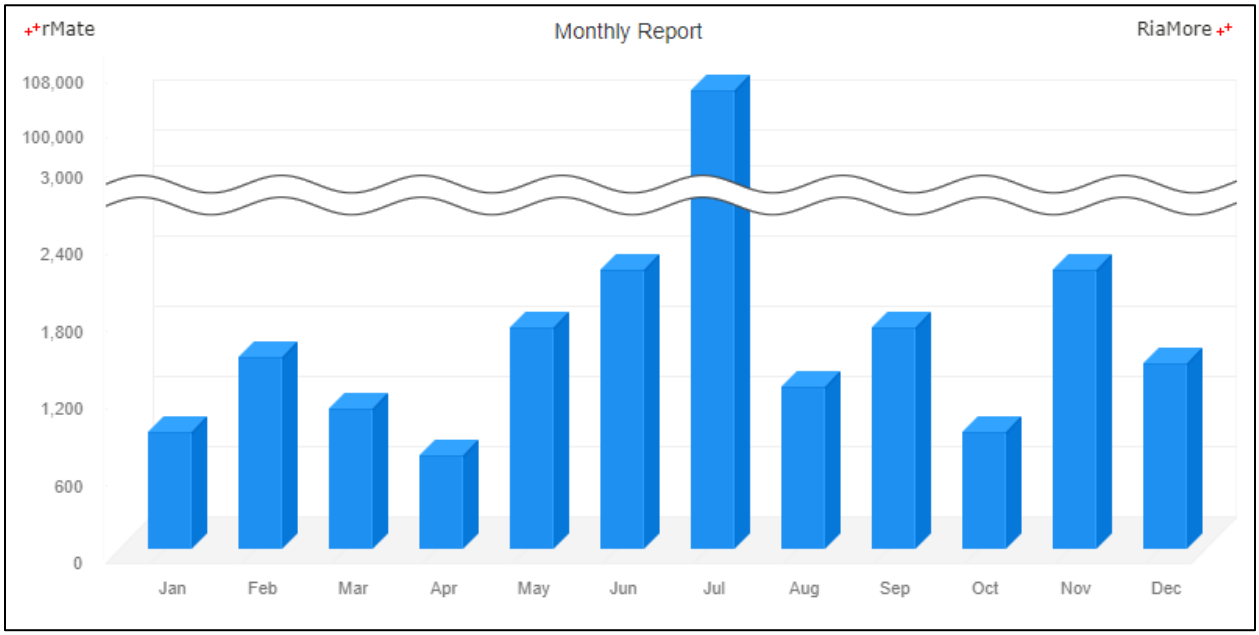
<BrokenAxis> 사용되는 주요 속성에 대한 설명은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
brokenMaximum	숫자 기본값: NaN	브로큰 축의 최대값(두 번째 물결무늬가 표시되는 축 상의 지점)을 지정합니다.
brokenMaximumInterval	숫자 기본값: NaN	브로큰 축의 최대값(brokenMaximum)보다 큰 값들이 축에 표시되는 간격을 지정합니다.
brokenMinimum	숫자 기본값: NaN	브로큰 축의 최소값(첫 번째 물결무늬가 표시되는 축 상의 지점)을 지정합니다.
brokenMinimumInterval	숫자 기본값: NaN	브로큰 축의 최소값(brokenMinimum)보다 작은 값들이 축에 표시되는 간격을 지정합니다.
brokenOffset	숫자 기본값: 14	두 물결무늬 사이의 간격을 지정합니다.
brokenRatio	0 과 1 사이의 숫자 기본값: 0.5	축의 시작 지점에서 브로큰 축의 최소값까지의 크기와 브로큰 축의 최대값에서 축의 끝 지점까지의 크기의 비율을 지정합니다.

브로큰 축 3D 컬럼 차트

3D 컬럼 차트에 브로큰 축 차트를 적용할 수 있습니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column3DChart showDataTips="true" columnWidthRatio="0.5">
  <horizontalAxis>
    <CategoryAxis categoryField="Month"/>
  </horizontalAxis>
  <verticalAxis>
    <BrokenAxis id="vAxis" brokenMinimum="3000" brokenMaximum="100000"
       brokenRatio="0.8" formatter="{numfmt}"/>
  </verticalAxis>
  <verticalAxisRenderers>
    <BrokenAxis3DRenderer axis="{vAxis}"/>
  </verticalAxisRenderers>
  <series>
    <Column3DSeries yField="Cost" displayName="Cost" />
  </series>
</Column3DChart>
```

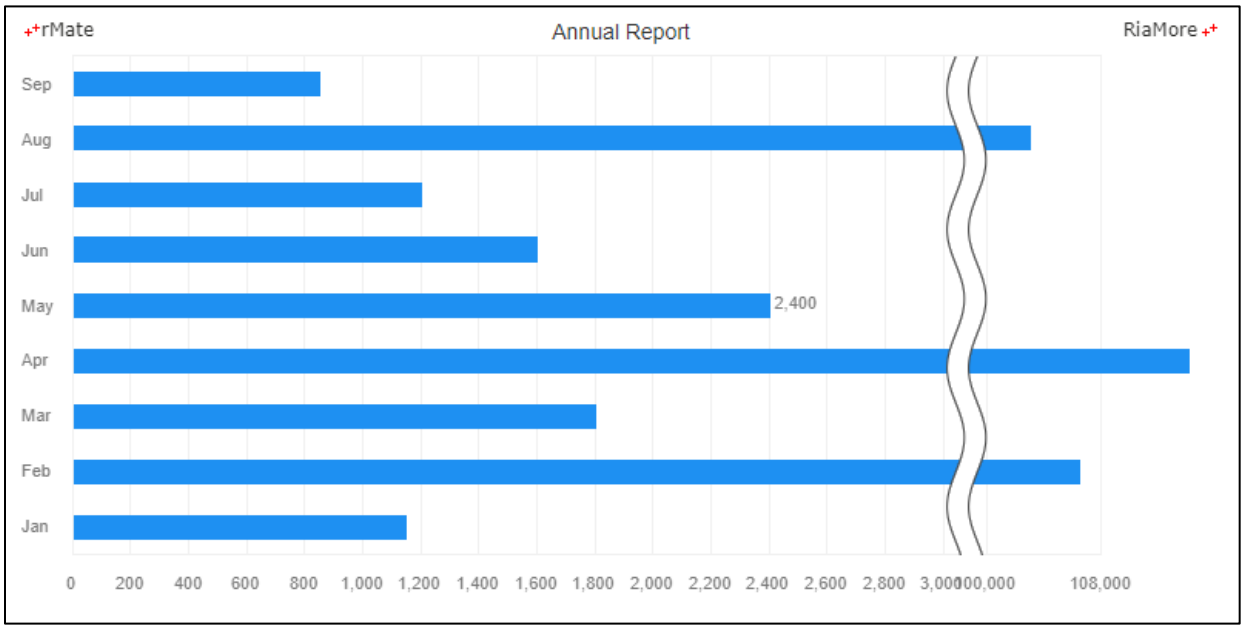


See the CodePen 알메이트 차트 - 브로큰 축 3D 컬럼 차트

브로큰 축 바 차트

브로큰 축 바 차트를 생성하기 위해서는 <BrokenAxis> 노드를 <horizontalAxis> 속성에 정의합니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DChart showDataTips="true" barWidthRatio="0.45">
  <verticalAxis>
    <CategoryAxis categoryField="Month"/>
  </verticalAxis>
  <horizontalAxis>
    <BrokenAxis id="hAxis" brokenMinimum="3000" brokenMaximum="100000"
       brokenRatio="0.8" formatter="{numfmt}"/>
  </horizontalAxis>
  <horizontalAxisRenderers>
    <BrokenAxis2DRenderer axis="{hAxis}"/>
  </horizontalAxisRenderers>
  <series>
    <Bar2DSeries xField="Revenue" labelPosition="outside" displayName="Revenue"
       showValueLabels="[4]" outsideLabelYOffset="-2">
      ...
    </Bar2DSeries>
  </series>
</Bar2DChart>
```

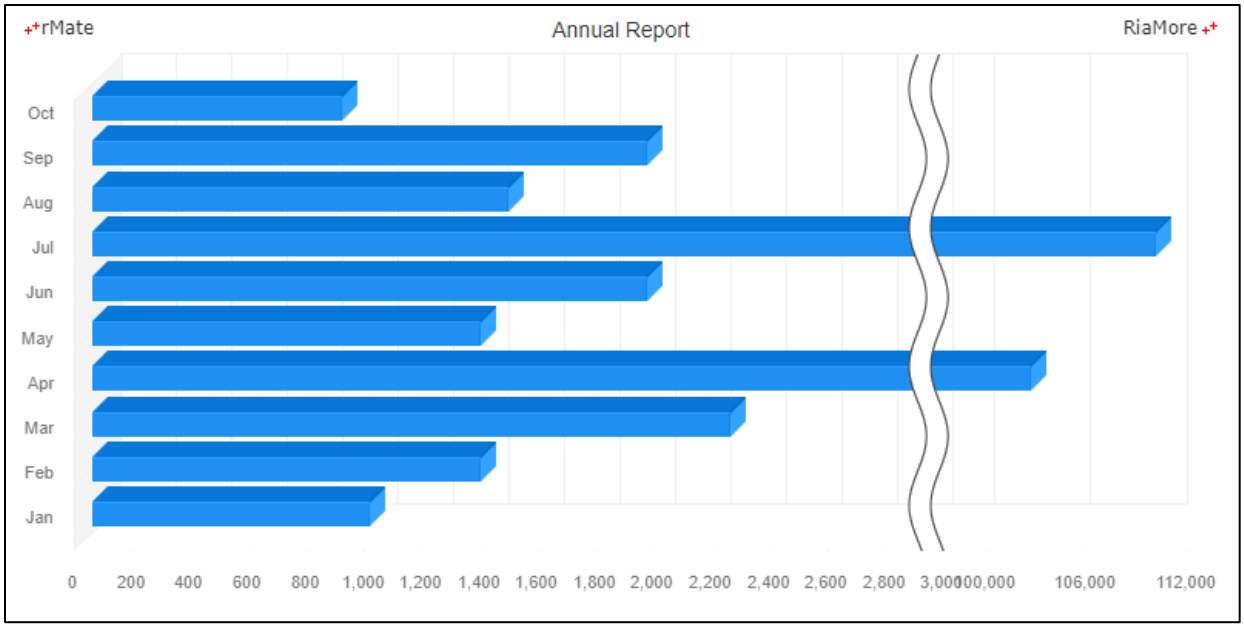


See the CodePen 알메이트 차트 - 브로큰 축 바 차트

브로큰 축 3D 바 차트

3D 바 차트에 브로큰 축 차트를 적용할 수 있습니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar3DChart showDataTips="true">
  <verticalAxis>
    <CategoryAxis categoryField="Month"/>
  </verticalAxis>
  <horizontalAxis>
    <BrokenAxis id="hAxis" brokenMinimum="3000" brokenMaximum="100000"
       brokenRatio="0.8" formatter="{numfmt}"/>
  </horizontalAxis>
  <horizontalAxisRenderers>
    <BrokenAxis3DRenderer axis="{hAxis}"/>
  </horizontalAxisRenderers>
  <series>
    <Bar3DSeries xField="Profit" displayName="Profit">
      ...
    </Bar3DSeries>
  </series>
</Bar3DChart>
```

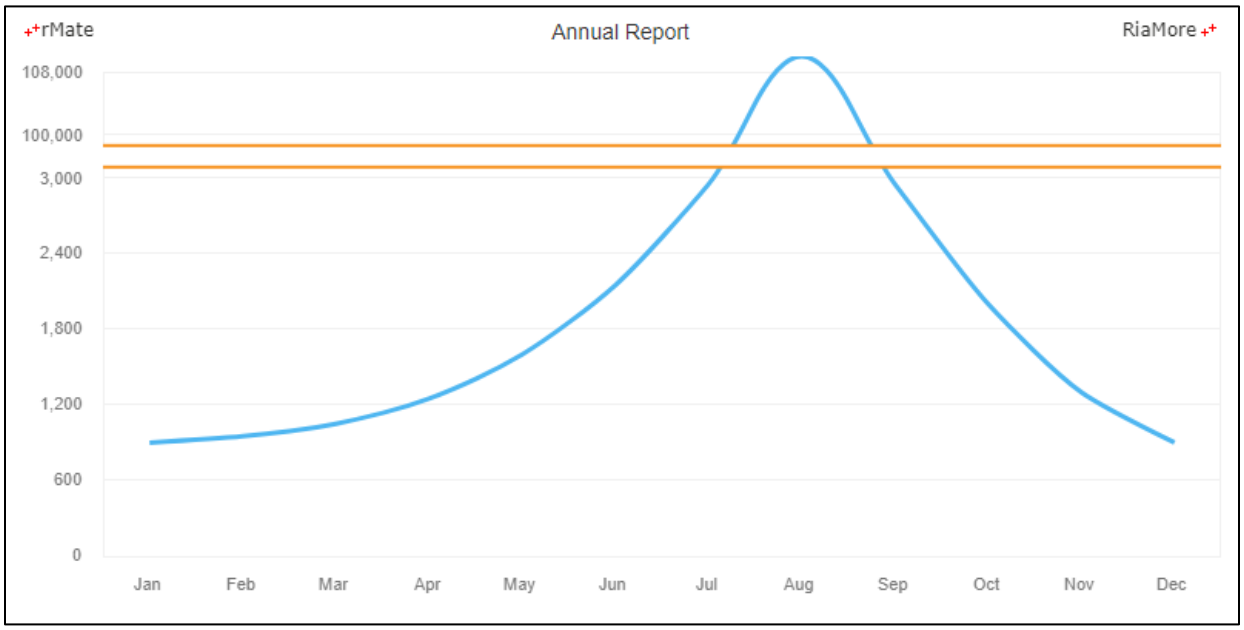


See the CodePen 알메이트 차트 - 브로큰 축 3D 바 차트

브로큰 축 라인 차트

라인 차트에 브로큰 축 차트를 적용할 수 있습니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DChart showDataTips="true">
  ...
  <verticalAxis>
    <BrokenAxis id="vAxis" brokenMinimum="3000" brokenMaximum="100000"
       brokenRatio="0.8" formatter="{numfmt}"/>
  </verticalAxis>
  ...
  <Line2DSeries yField="Revenue" displayName="Revenue" form="curve">
  ...
  <annotationElements>
    <BrokenAxisLine flatRatio="1" brokenAxisAreaColor="#ffffff">
      <brokenAxisLineStroke>
        <Stroke color="#f7921e" weight="2"/>
      </brokenAxisLineStroke>
    </BrokenAxisLine>
  </annotationElements>
  ...
</Line2DChart>
```



See the CodePen 알메이트 차트 - 브로큰 축 라인 차트

물결무늬 설정

차트에 표시되는 물결무늬에서 물결 곡선의 정도와 선, 물결 사이의 공간에 칠해지는 색을 조절할 수 있습니다. 위에서 설명된 브로큰 축 라인 차트에서는 flatRatio 속성을 “1” 로 지정하여 물결무늬 부분이 직선으로 표현되었습니다. 이 작업은 <annotationElements> 속성에 <BrokenAxisLine> 노드를 정의함으로써 가능합니다.

<BrokenAxisLine> 노드에서 사용되는 주요 속성에 대한 설명은 다음과 같습니다.

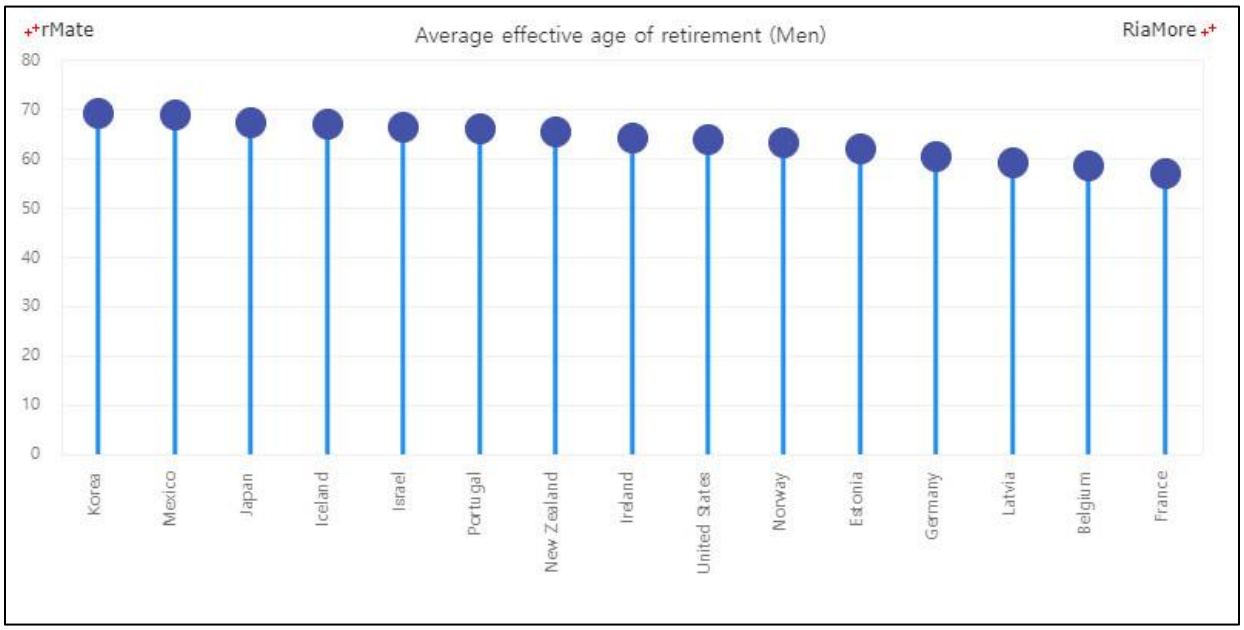
속성명	유효값 (*: 기본값)	설명
brokenAxisAreaColor	#16 진수 컬러 코드 표기 기본값: #ffffff>	두 물결무늬 사이의 공간에 칠할 색을 지정합니다.
brokenAxisLineStroke	<Stroke>	물결무늬 선의 스타일을 지정합니다.
direction	horizontal(*), vertical	물결무늬 선이 그려질 방향을 지정합니다. 기본값은 “horizontal” 이고, 바 차트에서의 기본값은 “vertical” 입니다.
flatRatio	0 과 1 사이의 숫자 기본값: 0.2	물결무늬 곡선의 정도를 비율로 지정합니다. 값이 0 에 가까울 수록 곡선의 정도가 강해집니다.

4.18 롤리팝 차트

롤리팝 차트는 막대의 끝에 원(circle)이 표시된다는 점을 제외하면 컬럼 차트와 동일한 유형의 차트입니다. 롤리팝 차트가 컬럼 차트에 비해서 유용하게 활용되는 경우는 차트에 표시되는 대부분의 값들이 80-90% 정도로 높은 값들일 때입니다. 이런 경우에 긴 막대들이 차트에서 나열되면 시각적으로 보기에 어렵고 값의 차이를 알기 쉽지 않기 때문입니다. 롤리팝 차트는 <LollipopChart> 노드의 series 속성값에 <LollipopSeries> 노드를 설정하여 생성할 수 있습니다.

다음은 롤리팝 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<LollipopChart showDataTips="true" dataTipFormatter="{numFmt}">
  ...
  <LollipopSeries yField="Men" displayName="Effective Age (Men)"
       showDataEffect="{ss}" itemRenderer="CircleLollipopItemRenderer"
       maxPopSize="8">
  ...
</LollipopChart>
```



See the CodePen 알메이트 차트 - 롤리팝 차트

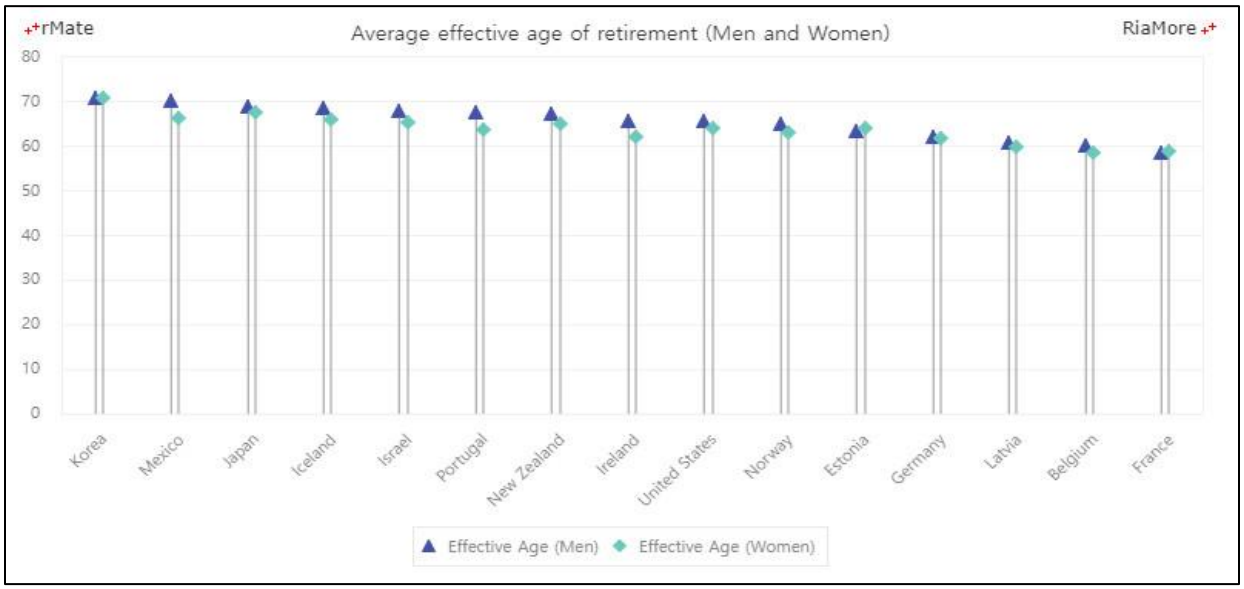
<LollipopSeries> 노드의 주요 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
itemRenderer	CircleLollipopItemRenderer(*) TriangleLollipopItemRenderer RectangleLollipopItemRenderer InvertedTriangleLollipopItemRenderer DiamondLollipopItemRenderer	차트상의 데이터 포인트(데이터 아이템)에 표현될 도형을 렌더링하는 클래스를 설정합니다.
maxPopSize	숫자 기본값: NaN	데이터 포인트에 표시되는 도형(사탕)의 최대 크기를 설정합니다.

다중 시리즈 롤리팝 차트

series 속성값에 여러 개의 <LollipopSeries> 노드를 설정하여 다중 시리즈 롤리팝 차트를 생성할 수 있습니다. 다음은 여러 개의 <LollipopSeries> 노드를 설정하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<LollipopChart showDataTips="true" dataTipFormatter="{numFmt}">
  ...
    <LollipopSeries yField="Men" displayName="Effective Age (Men)"
        showDataEffect="{ss}" itemRenderer="CircleLollipopItemRenderer"
        maxPopSize="5">
    ...
    <LollipopSeries yField="Women" displayName="Effective Age (Women)"
        showDataEffect="{ss}" itemRenderer="CircleLollipopItemRenderer"
        maxPopSize="5">
    ...
</LollipopChart>
```

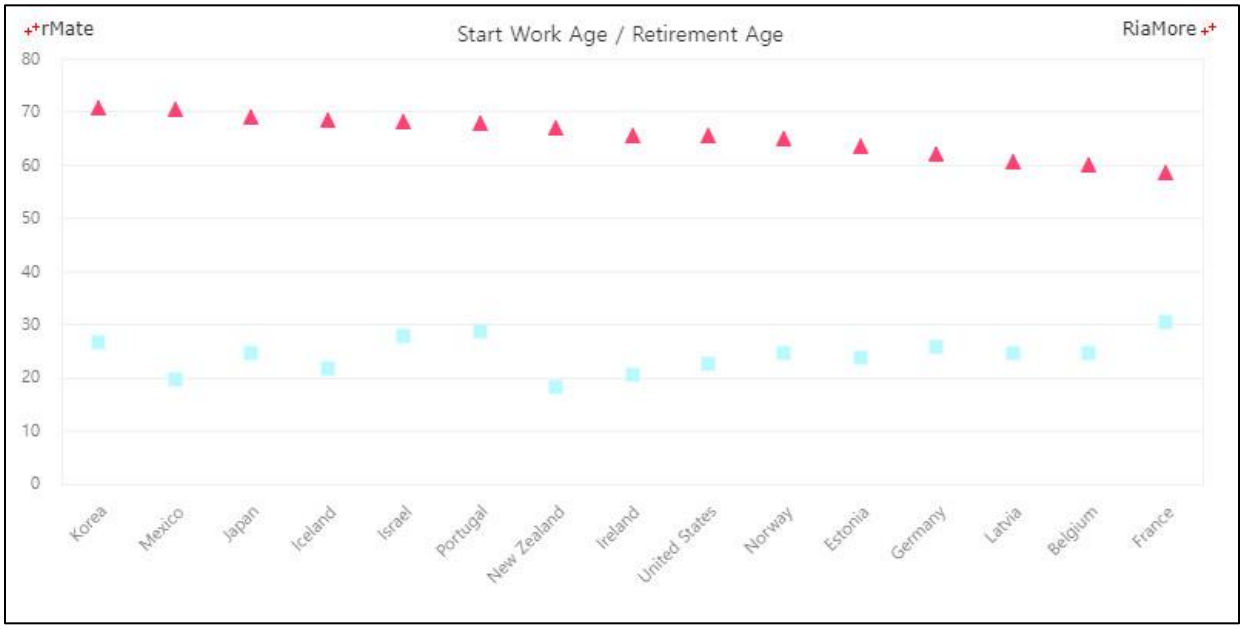


See the CodePen 알메이트 차트 - 다중 시리즈 롤리팝 차트

오버레이 롤리팝 차트

다중 시리즈 롤리팝 차트에서 <LollipopChart> 노드의 type 속성을 “overlaid” 으로 설정하면 원(막대 사탕)이 오버레이 형태로 표시됩니다. 이 때 막대 선(lineStroke 속성)을 설정하는 <Stroke> 노드의 weight 와 alpha 속성값을 조절하여 막대 선을 표시하지 않고 원(막대 사탕)만 오버레이 형태로 표현할 수 있습니다. 다음은 오버레이 롤리팝 차트를 표현하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<LollipopChart showDataTips="true" dataTipFormatter="{numFmt}" type="overlaid">
  ...
    <LollipopSeries yField="Normal" displayName="Normal Age (Men)"
        showDataEffect="{ss}" itemRenderer="CircleLollipopItemRenderer"
        maxPopSize="5">
    ...
    <LollipopSeries yField="Effective" displayName="Effective Age (Men)"
        showDataEffect="{ss}" itemRenderer="CircleLollipopItemRenderer"
        maxPopSize="5">
    ...
</LollipopChart>
```



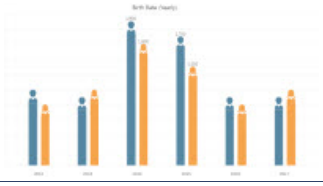
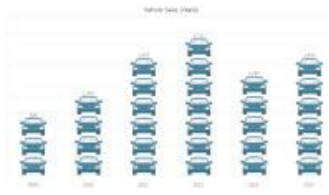
See the CodePen 알메이트 차트 - 오버레이 롤리팝 차트

4.19 이미지 차트

이미지 차트는 컬럼 차트와 동일하지만(컬럼 차트에서 사용되는 속성을 그대로 사용합니다.), 데이터 값을 세로 막대의 크기가 아니라 이미지의 크기로 표현합니다. 이미지 차트는 <ImageChart>노드의 series 속성값에 <ImageSeries> 노드를 설정하여 생성할 수 있습니다. 표시되는 이미지가 존재하는 URL 은 <ImageSeries> 노드의 <imgSource> 속성에 <ImageSourceItem> 노드로 정의합니다. 이미지 차트의 유형은 <ImageSeries> 노드의 imageDisplayType 속성과 <ImageSourceItem> 노드의 maintainAspectRatio 속성에 의해서 결정되는데 이에 대한 설명은 아래와 같습니다.

- imageDisplayType : 단일 이미지, 단일 이미지 반복, 다중 이미지를 사용할지 여부를 지정합니다.
- maintainAspectRatio : 원(Original) 이미지의 비율과 동일한 이미지를 차트에 표현할지 여부를 지정합니다.

두 속성값(imageDisplayType, maintainAspectRatio)에 의해서 생성되는 이미지 차트의 유형은 아래 표와 같이 요약할 수 있습니다.

imageDisplayType	maintainAspectRatio	설명	예제 모습
single	True (same ratio)	데이터를 단일 이미지로 표현하되 이미지 고유 사이즈는 유지하고 남는 여백은 막대를 세웁니다.	
	False (different ratio)	데이터를 단일 이미지로 표현하되 이미지 고유 사이즈가 아닌 차트가 결정한 사이즈대로 표현합니다.	
singleRepeat	True (same ratio)	데이터를 단일 이미지로 표현하되 이미지 고유 사이즈는 유지하고 남는 여백은 이미지의 반복으로 처리합니다.	
	False (different ratio)	singleRepeat 유형에서의 차등비율은 존재하지 않습니다.	

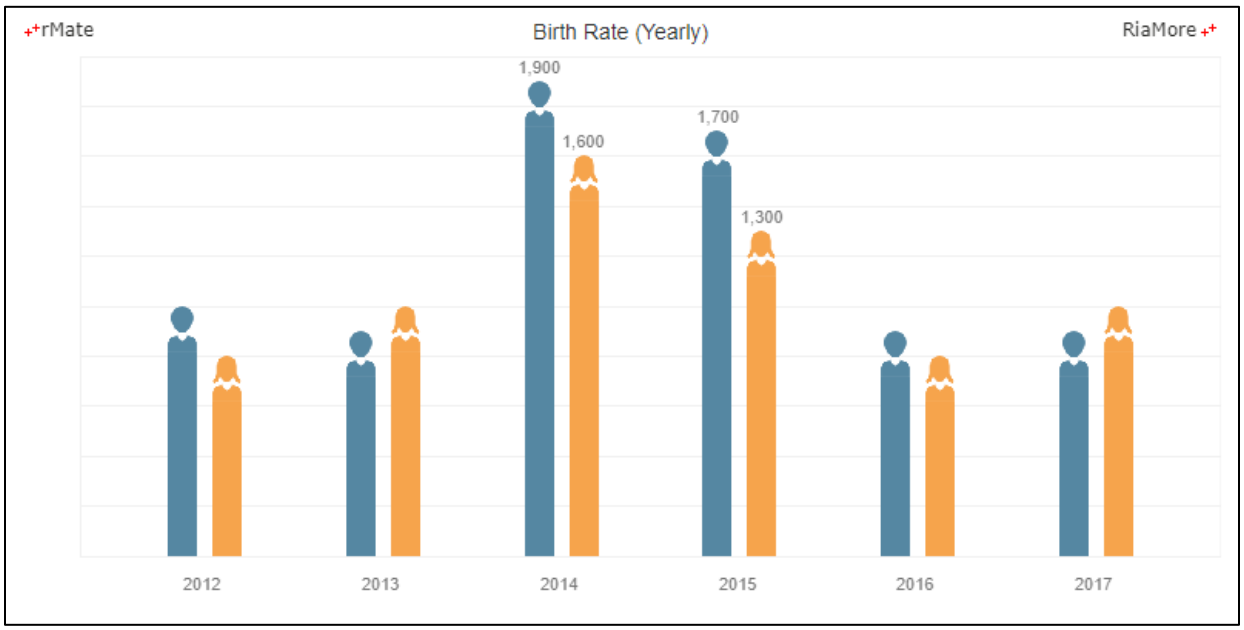
multiple	True (same ratio)	Multiple 유형에서의 차등비율은 존재하지 않습니다.	
	False (different ratio)	데이터를 다중 이미지로 표현합니다. 각각의 이미지는 고유 값을 갖습니다. 이 값은 차트의 사이즈에 의해 계산되어 이미지로 전달됩니다.	

정배율 단일 이미지

다음은 단일 이미지(<ImageSeries imageDisplayType="single" ...>), 정배율(<ImageSourceItem maintainAspectRatio="true" ...>) 유형의 이미지 차트를 생성하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 X 축과 데이터 포인트 사이의 공간에 표시되는 막대의 색에 이미지와 동일한 색을 적용하였습니다.

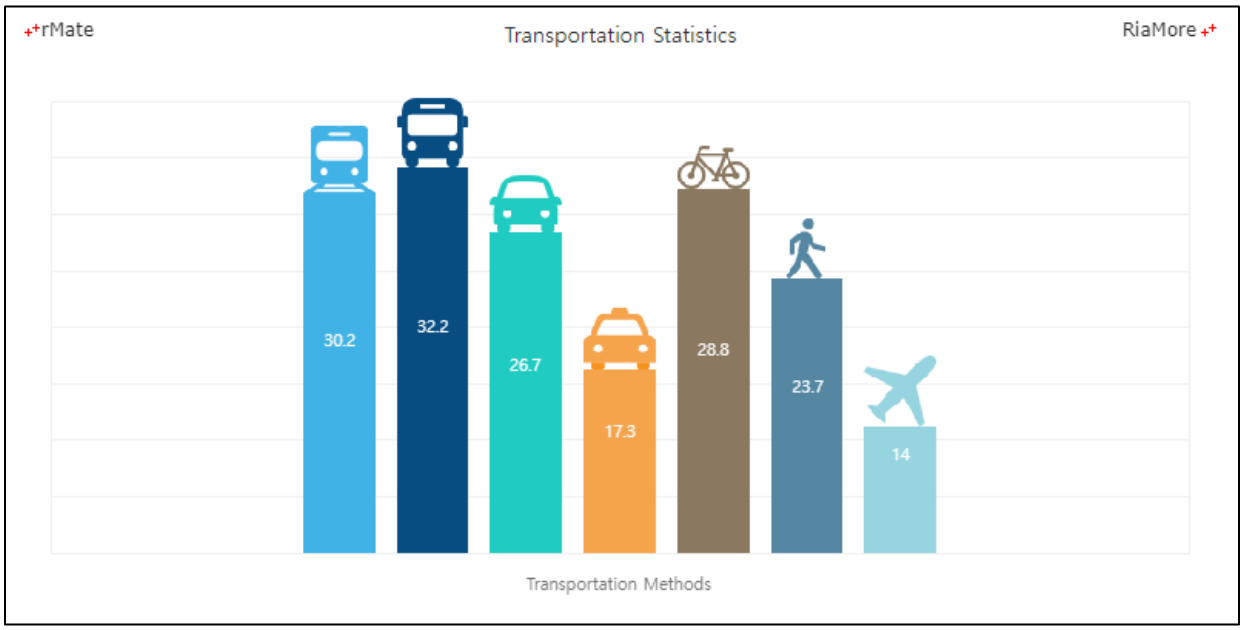
이미지	URL	막대의 색
	<ImageSourceItem url="../rMateChartH5/Assets/Images/woman.png"/>	<ImageSeries ... fill="#5587a2">
	<ImageSourceItem url="../rMateChartH5/Assets/Images/man.png"/>	<ImageSeries ... fill="#f6a44c">

```
<ImageChart id="chart" showDataTips="true" showLabelVertically="true">
  ...
  <ImageSeries labelPosition="outside" yField="Data1" imageDisplayType="single"...>
    <imgSource>
      <ImageSourceItem maintainAspectRatio="true"
       url="../rMateChartH5/Assets/Images/man5.png"/>
    </imgSource>
  </ImageSeries>
  <ImageSeries labelPosition="outside" yField="Data2" imageDisplayType="single"...>
    <imgSource>
      <ImageSourceItem maintainAspectRatio="true"
       url="../rMateChartH5/Assets/Images/woman4.png"/>
      </imgSource>
  </ImageSeries>
  ...
</ImageChart>
```



See the CodePen 알메이트 차트 - 정배율 단일 이미지 차트

다음은 정배율 단일 이미지 차트의 다른 예제입니다. 마찬가지로 X 축과 데이터 포인트 사이의 공간에 표시되는 막대의 색에 이미지와 동일한 색을 적용하였습니다.

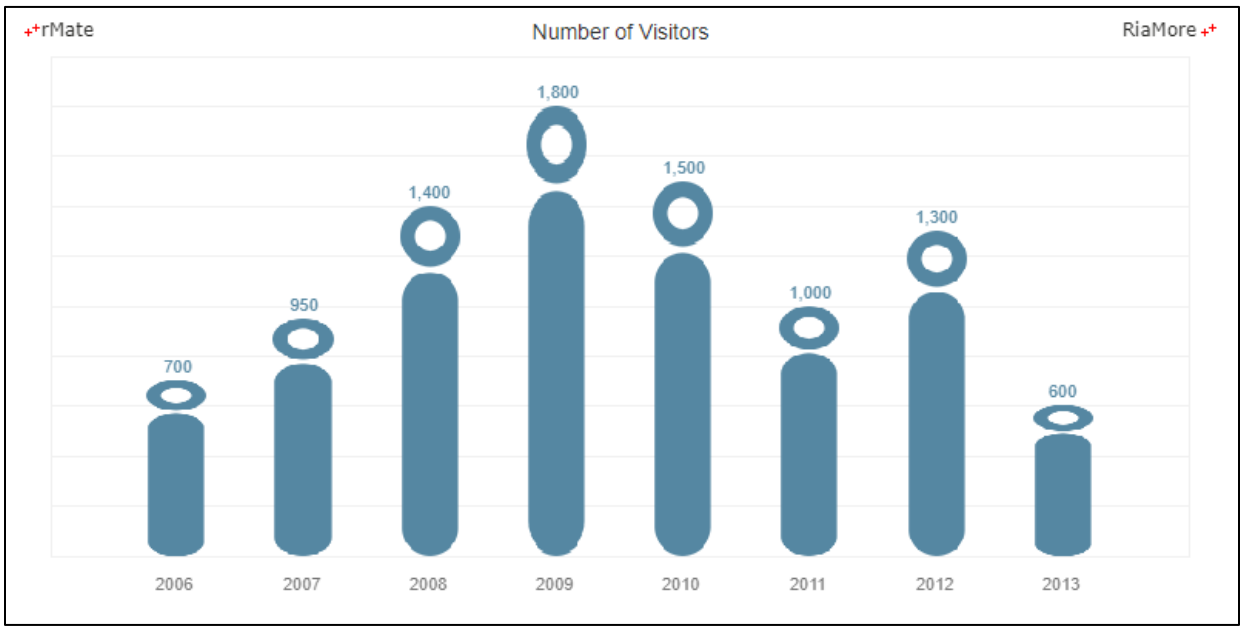


See the CodePen 알메이트 차트 - 정배율 단일 이미지 차트 (이동수단)

차등 배율 단일 이미지 차트

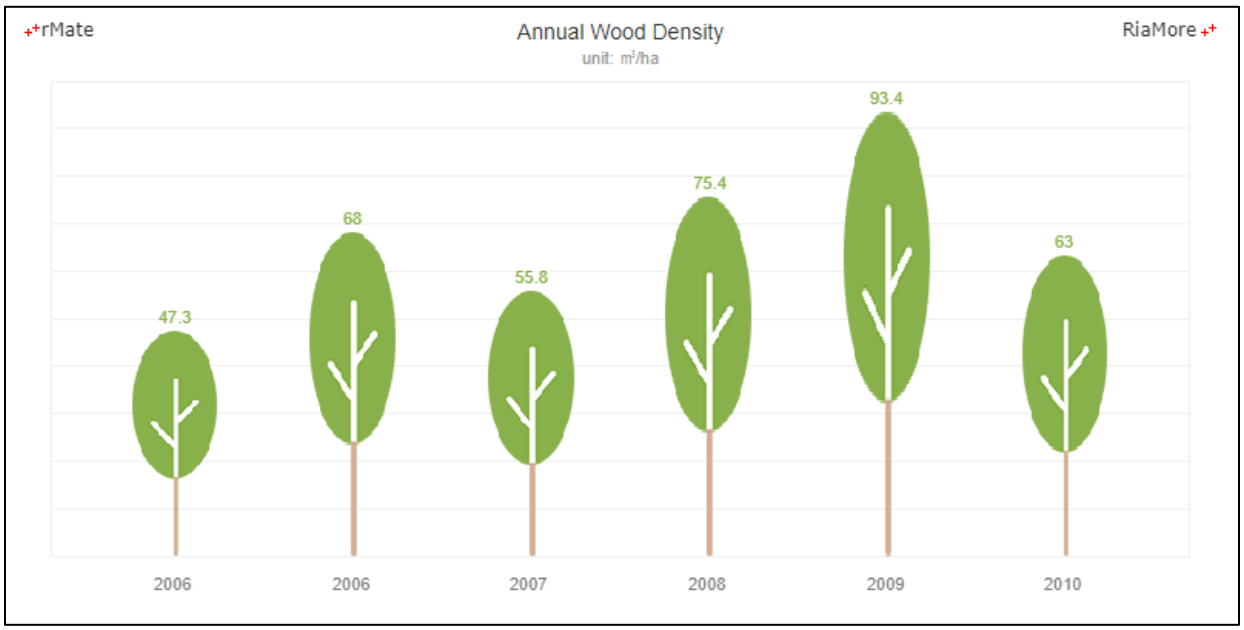
다음은 단일 이미지(<ImageSeries imageDisplayType="single" ...>), 차등 배율(<ImageSourceItem maintainAspectRatio="false" ...>) 유형의 이미지 차트를 생성하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 X 축과 데이터 포인트 사이의 공간에 원(Original) 이미지를 데이터 크기에 따라서 이미지의 크기를 조절하여 표현합니다.

```
<ImageChart id="chart" showDataTips="true" gutterLeft="20" gutterRight="20"
        showLabelVertically="true" columnWidthRatio="0.5">
  ...
  <ImageSeries yField="Data1" imageDisplayType="single" labelPosition="outside"
        formatter="{numFmt}" color="#5587a2">
    <imgSource>
      <ImageSourceItem maintainAspectRatio="false"
        url="../rMateChartH5/Assets/Images/visit_human2.png"/>
    </imgSource>
  </ImageSeries>
  ...
</ImageChart>
```



See the CodePen 알메이트 차트 - 차등 배율 단일 이미지 차트

다음은 차등 배율 단일 이미지 차트의 다른 예제입니다. 마찬가지로 X 축과 데이터 포인트 사이의 공간에 원(Original) 이미지를 데이터 크기에 따라서 이미지의 크기를 조절하여 표현하였습니다.

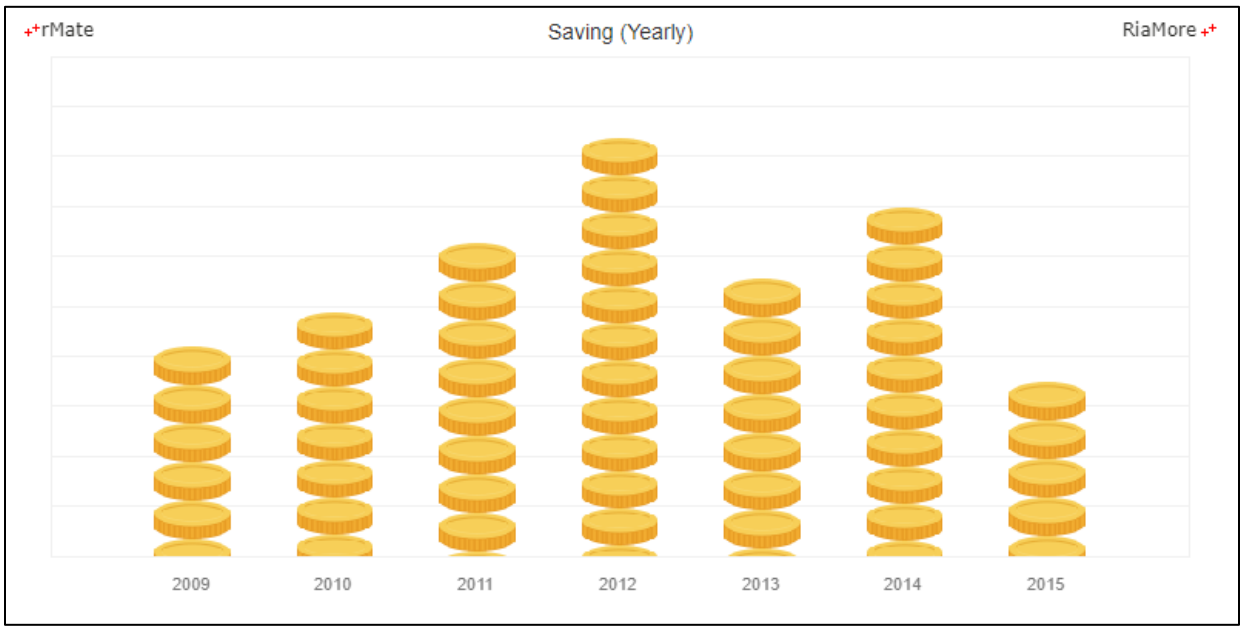


See the CodePen 알메이트 차트 - 차등 배율 단일 이미지 차트 (나무)

정배율 반복 이미지 차트

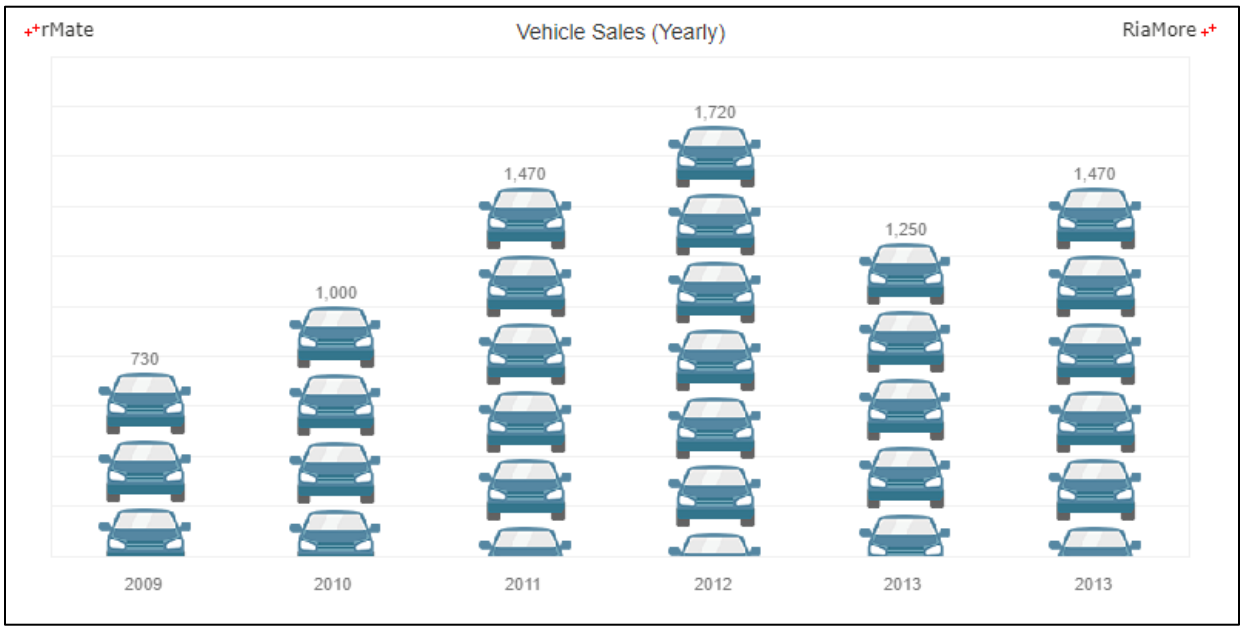
다음은 단일 이미지를 반복(<ImageSeries imageDisplayType="singleRepeat" ...>)해서 표시하는 유형의 이미지 차트를 생성하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다. 반복 이미지 차트에서는 maintainAspectRatio 속성값에는 오직 "true" 만이 적용 가능합니다. 따라서이 예제에서는 maintainAspectRatio 속성값의 설정이 생략되었습니다.

```
<ImageChart id="chart" showDataTips="true" gutterLeft="20" gutterRight="20"
        showLabelVertically="true" columnWidthRatio="0.56">
  ...
  <ImageSeries yField="Data1" imageDisplayType="singleRepeat" formatter="{numFmt}">
    <imgSource>
      <ImageSourceItem url="../rMateChartH5/Assets/Images/coin.png"/>
    </imgSource>
  </ImageSeries>
  ...
</ImageChart>
```



See the CodePen 알메이트 차트 - 정배율 반복 이미지 차트

다음은 정배율 반복 이미지 차트의 다른 예제입니다.



See the CodePen 알메이트 차트 - 정배율 반복 이미지 차트 (자동차)

차등 배율 다중 이미지 차트

다음은 다중 이미지를 (<ImageSeries imageDisplayType="multiple" ...>) 차등 배율로 표시하는 유형의 이미지 차트를 생성하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 3 개의 이미지가 사용되었고, 각 이미지의 단위 값이 <ImageSourceItem> 노드의 value 속성에 설정되었습니다.

```
<ImageChart id="chart" showDataTips="true" gutterLeft="20" gutterRight="20"
        showLabelVertically="true">
  ...
  <ImageSeries yField="Data1" imageDisplayType="multiple" labelPosition="outside"
        formatter="{numFmt}">
    <imgSource>
      <ImageSourceItem maintainAspectRatio="false"
       url="../rMateChartH5/Assets/Images/person1.png" value="100"/>
      <ImageSourceItem maintainAspectRatio="false"
       url="../rMateChartH5/Assets/Images/person2.png" value="200"/>
      <ImageSourceItem maintainAspectRatio="false"
       url="../rMateChartH5/Assets/Images/person3.png" value="300"/>
    </imgSource>
  </ImageSeries>
  ...
</ImageChart>
```

See the CodePen 알메이트 차트 - 차등 배율 다중 이미지 차트

다음은 차등 배율 다중 이미지 차트의 다른 예제입니다.

See the CodePen 알메이트 차트 - 차등 배율 다중 이미지 차트 (연도별 수출)

4.20 이미지 매트릭스 차트

이미지 매트릭스 차트는 데이터 값에 해당하는 개수만큼의 이미지를 차트에 표시합니다. 이 때 표시해야 하는 이미지 개수와 정해진 차트 공간의 크기에 따라서 이미지 크기를 자동으로 계산합니다. 차트에는 여러 이미지 파일을 표시할 수도 있고, 하나의 이미지 파일만 표시할 수도 있습니다. 이미지 매트릭스 차트는 <ImageMatrixChart> 노드의 series 속성값에 <ImageMatrixSeries>노드를 설정하여 생성할 수 있습니다. 다음은 하나의 이미지 파일을 이용하여 데이터 값(22)에 해당하는 개수의 이미지를 차트에 표현하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<ImageMatrixChart showDataTips="true" paddingLeft="20" paddingRight="20">
  ...
  <ImageMatrixSeries field="coffeValue"
       imageURL="../rMateChartH5/Assets/Images/coffee.png" displayName="Coffee"
       horizontalGap="10" verticalGap="10">
  </ImageMatrixSeries>
  ...
</ImageMatrixChart>
```

See the CodePen 알메이트 차트 - 이미지 매트릭스 차트

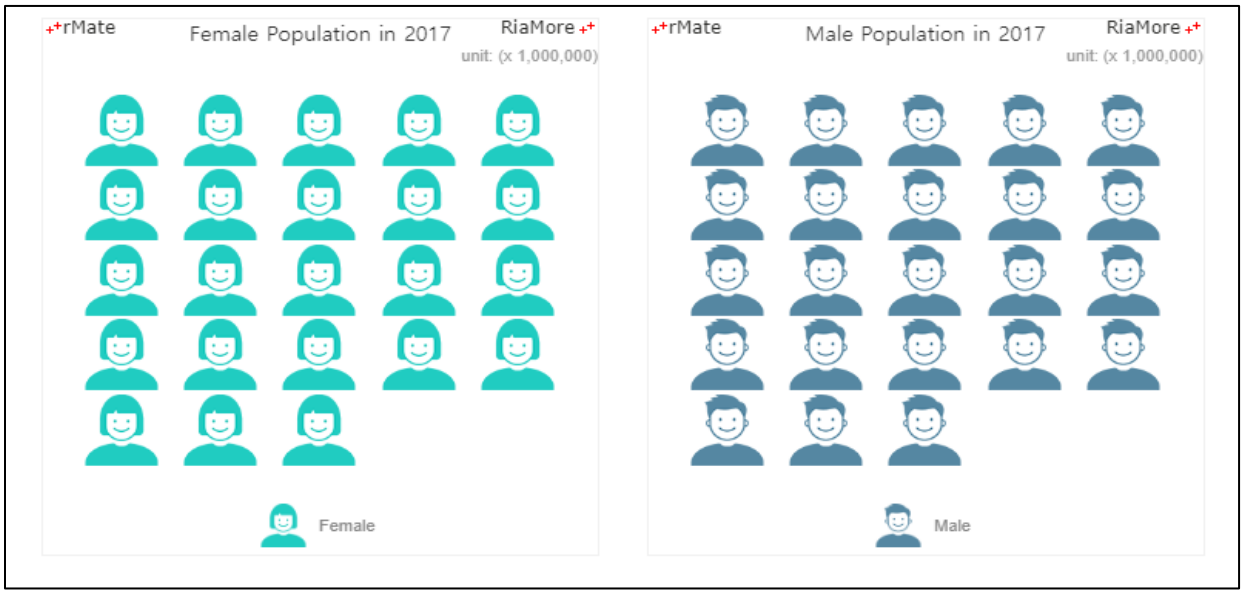
이미지와 이미지 사이의 공간의 크기는 <ImageMatrixSeries> 노드의 verticalGap 과 horizontalGap 속성값에 의해서 결정됩니다.

속성명	유효값 (*: 기본값)	설명
horizontalGap	숫자 기본값: 2	이미지와 이미지 사이의 수평 공간의 크기를 지정합니다.
verticalGap	숫자 기본값: 2	이미지와 이미지 사이의 수직 공간의 크기를 지정합니다.

다중 이미지 매트릭스 차트

다음은 두 개의 이미지를 이용하여 실적값(11)과 목표값(3)의 개수만큼 이미지를 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<ImageMatrixChart showDataTips="true" padding="20" borderStyle="solid"
        borderWidth="1" borderColor="#f1f1f1">
  ...
  <ImageMatrixSeries field="sales"
        imageURL="../rMateChartH5/Assets/Images/icon4.png" displayName="Actual">
  </ImageMatrixSeries>
  <ImageMatrixSeries field="goal"
        imageURL="../rMateChartH5/Assets/Images/icon5.png" displayName="Target">
  </ImageMatrixSeries>
  ...
</ImageMatrixChart>
```

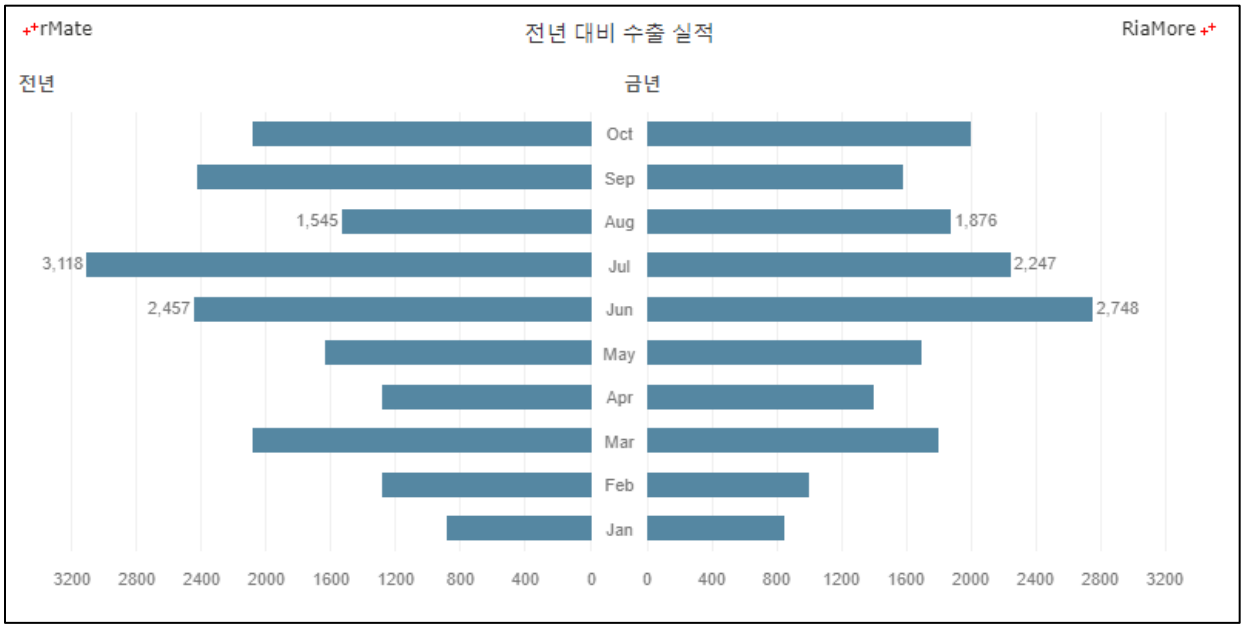


See the CodePen 알메이트 차트 - 다중 이미지 매트릭스 차트

4.21 윙 차트

바 윙 차트는 <Bar2DWingChart> 노드의 series 속성값에 <Bar2DWingSeries> 노드를 설정하여 생성할 수 있습니다. 우측 방향으로 표시되는 가로 막대 값이 있는 데이터 필드는 <Bar2DWingSeries> 노드의 xField 속성에, 좌측 방향으로 표시되는 가로 막대 값이 있는 데이터 필드는 xFieldOpp 속성에 지정합니다. 다음은 바 윙 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DWingChart showDataTips="true" paddingTop="10" barWidthRatio="0.56">
  ...
  <Bar2DWingSeries xField="Profit" xFieldOpp="Cost" labelPosition="outside"
       showValueLabels="[5,6,7]" showValueLabelsOpp="[5,6,7]" displayName="This
       Year" displayNameOpp="Last Year">
  </Bar2DWingSeries>
  ...
</Bar2DWingChart>
```

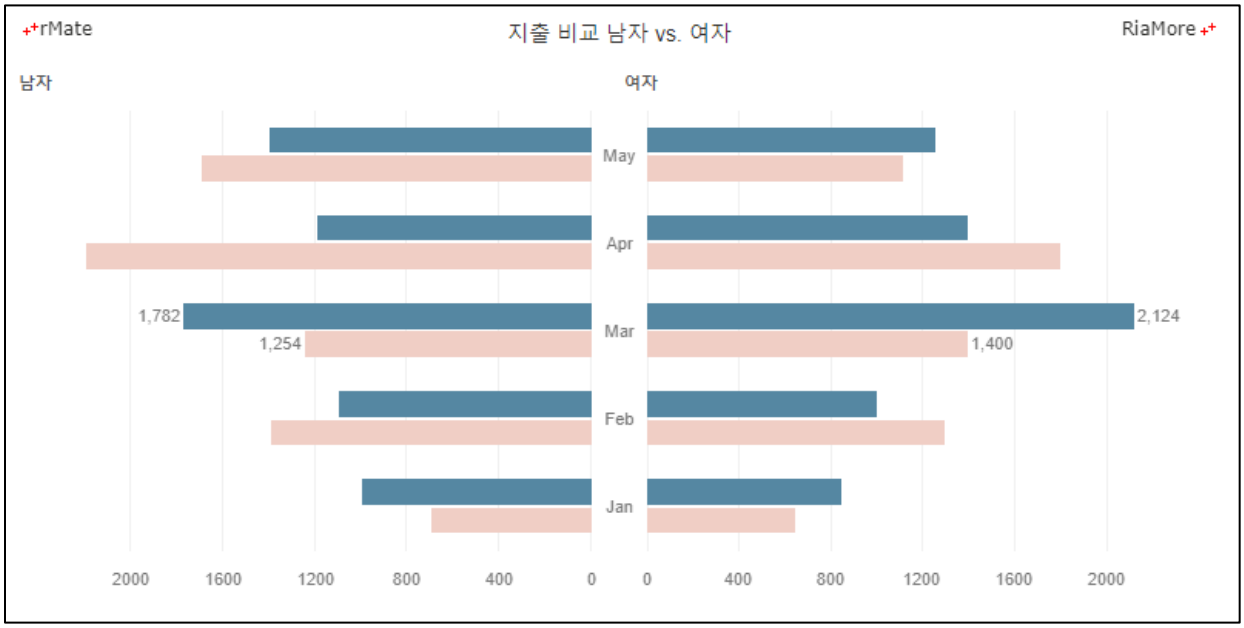


See the CodePen 알메이트 차트 - 바 윙 차트

다중 시리즈 바 윙 차트

일반적인 바 차트와 마찬가지로 다중 데이터 시리즈를 바 윙 차트에 표현할 수 있습니다. 표현하려는 개수만큼 <Bar2DWingSeries> 노드를 정의하여 다중 시리즈 바 윙 차트를 생성합니다. 다음은 두 개의 데이터 시리즈를 가진 바 윙 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DWingChart showDataTips="true" paddingTop="10">
  ...
  <Bar2DWingSeries xField="Profit" xFieldOpp="Profit2" labelPosition="outside"
       displayName="Female" displayNameOpp="Male" showValueLabels="[2]"
       showValueLabelsOpp="[2]" halfWidthOffset="1">
  </Bar2DWingSeries>
  <Bar2DWingSeries xField="Cost" xFieldOpp="Cost2" labelPosition="outside"
       displayName="Female" displayNameOpp="Male" showValueLabels="[2]"
       showValueLabelsOpp="[2]" halfWidthOffset="1">
  </Bar2DWingSeries>
  ...
</Bar2DWingChart>
```

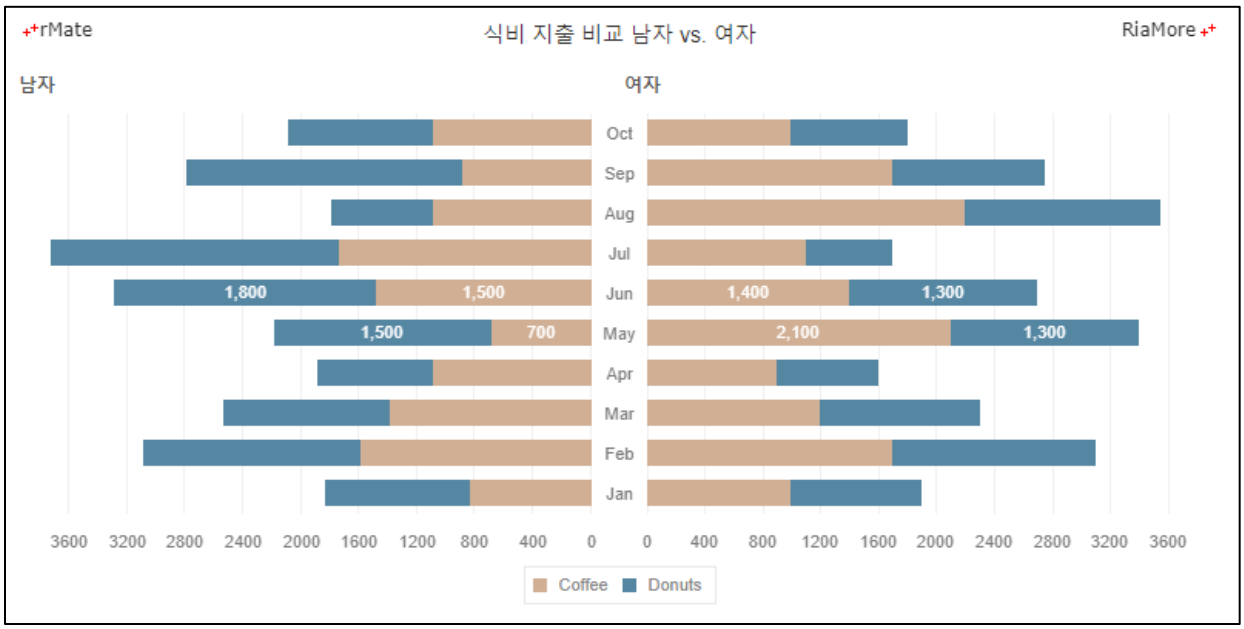


See the CodePen 알메이트 차트 - 다중 시리즈 바 윙 차트

스택 바 윙 차트

일반적인 바 차트와 마찬가지로 스택 타입의 바 윙 차트를 표현할 수 있습니다. 바의 스택으로 표현하려는 개수만큼 <Bar2DWingSeries> 노드를 정의하고, <Bar2DWingChart> 노드의 type 속성을 “stacked” 으로 지정하여 스택 바 윙 차트를 생성합니다. 다음은 두 개의 데이터 시리즈를 가진 스택 바 윙 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DWingChart type="stacked" showDataTips="true" paddingTop="10">
  ...
  <Bar2DWingSeries xField="Profit" xFieldOpp="Profit2" ...>
  </Bar2DWingSeries>
  <Bar2DWingSeries xField="Cost" xFieldOpp="Cost2" ...>
  </Bar2DWingSeries>
  ...
</Bar2DWingChart>
```

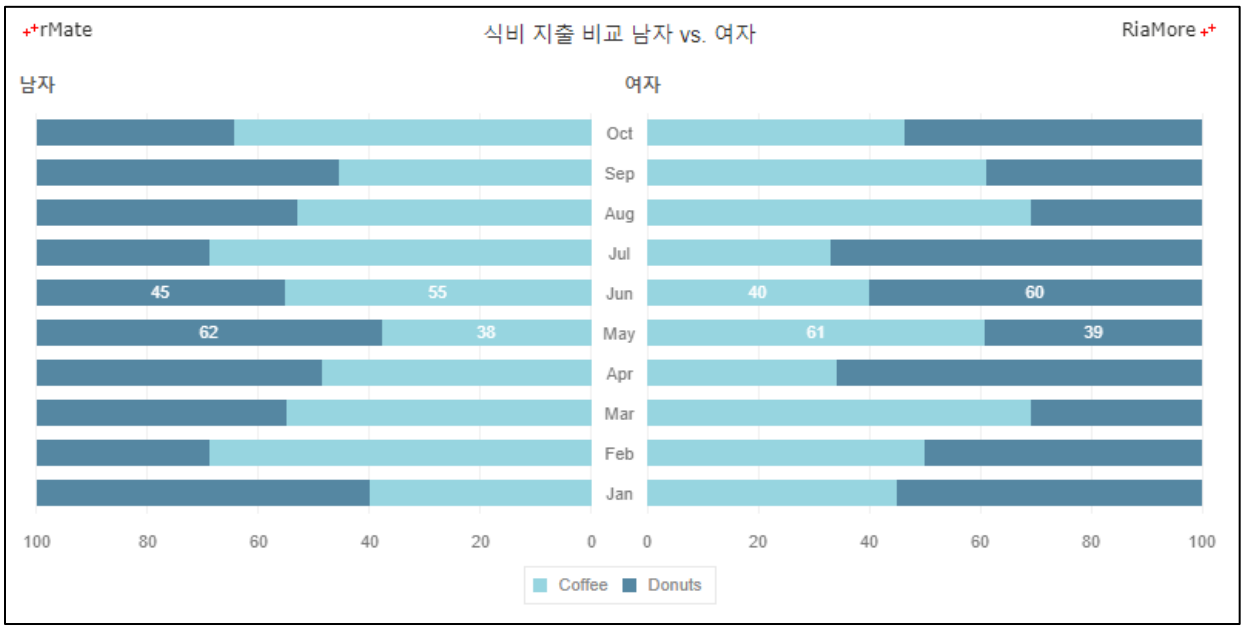


See the CodePen 알메이트 차트 - 스택 바 윙 차트

100% 바 윙 차트

일반적인 바 차트와 마찬가지로 100% 타입의 바 윙 차트를 표현할 수 있습니다. 바의 100% 스택으로 표현하려는 개수만큼 <Bar2DWingSeries> 노드를 정의하고, <Bar2DWingChart> 노드의 type 속성을 “100%” 으로 지정하여 100% 타입의 바 윙 차트를 생성합니다. 다음은 두 개의 데이터 시리즈를 가진 100% 타입의 바 윙 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DWingChart type="100%" showDataTips="true" paddingTop="10">
  ...
  <Bar2DWingSeries xField="Profit" xFieldOpp="Profit2" labelPosition="inside"
       displayName="Coffee" showValueLabels="[4,5]" showValueLabelsOpp="[4,5]"
       formatter="{nft}" color="#ffffff">
  </Bar2DWingSeries>
  <Bar2DWingSeries xField="Cost" xFieldOpp="Cost2" labelPosition="inside"
       displayName="Donuts" showValueLabels="[4,5]" showValueLabelsOpp="[4,5]"
       formatter="{nft}" color="#ffffff">
  </Bar2DWingSeries>
  ...
</Bar2DWingChart>
```

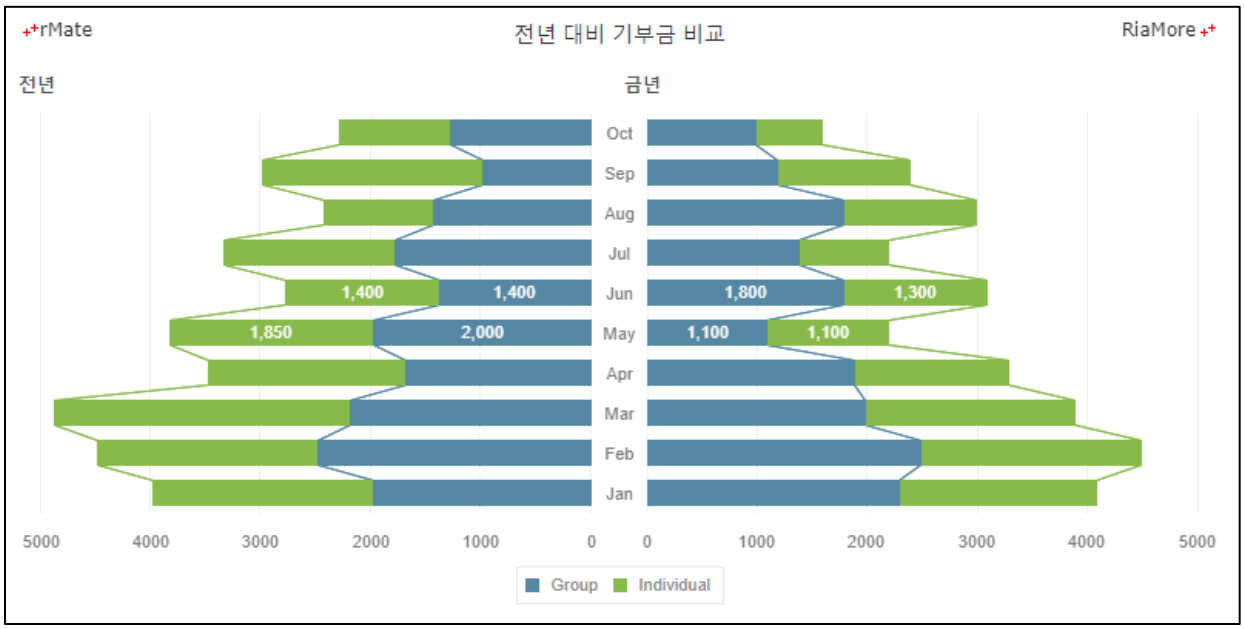


See the CodePen 알메이트 차트 - 100% 바 윙 차트

바 윙 차트에 연결선 표시

바 윙 차트에 표시되는 가로 막대의 값들을 연결하는 선을 표시할 수 있습니다. 바 윙 차트에 연결선을 표시하기 위해서는 <Bar2DWingSeries> 노드의 lineToEachItems 속성을 “true” 로 지정합니다.
다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DWingChart type="stacked" showDataTips="true" paddingTop="10">
  ...
  <Bar2DWingSeries xField="Profit" xFieldOpp="Profit2" lineToEachItems="true"
       labelPosition="inside" displayName="Group" showValueLabels="[4,5]"
       showValueLabelsOpp="[4,5]" color="#ffffff">
  </Bar2DWingSeries>
  <Bar2DWingSeries xField="Cost" xFieldOpp="Cost2" lineToEachItems="true"
       labelPosition="inside" displayName="Individual" showValueLabels="[4,5]"
       showValueLabelsOpp="[4,5]" color="#ffffff">
  </Bar2DWingSeries>
  ...
</Bar2DWingChart>
```

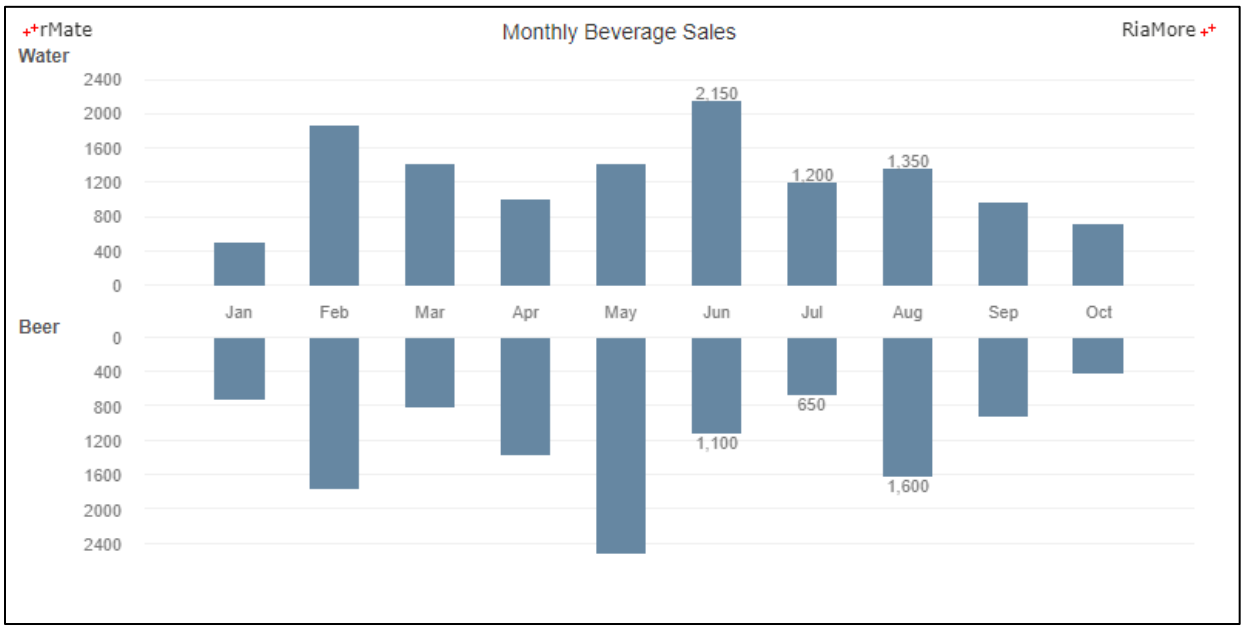


See the CodePen 알메이트 차트 - 바 윙 차트에 연결선 표시

컬럼 윙 차트

컬럼 윙 차트는 <Column2DWingChart> 노드의 series 속성값에 <Column2DWingSeries> 노드를 설정하여 생성할 수 있습니다. 윗 방향으로 표시되는 세로 막대 값이 있는 데이터 필드는 <Column2DWingSeries> 노드의 yField 속성에, 아래 방향으로 표시되는 세로 막대 값이 있는 데이터 필드는 yFieldOpp 속성에 지정합니다. 다음은 컬럼 윙 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DWingChart showDataTips="true" columnWidthRatio="0.52">
  ...
  <Column2DWingSeries yField="Profit" yFieldOpp="Cost" labelPosition="outside"
       showValueLabels="[5,6,7]" showValueLabelsOpp="[5,6,7]">
  </Column2DWingSeries>
  ...
</Column2DWingChart>
```

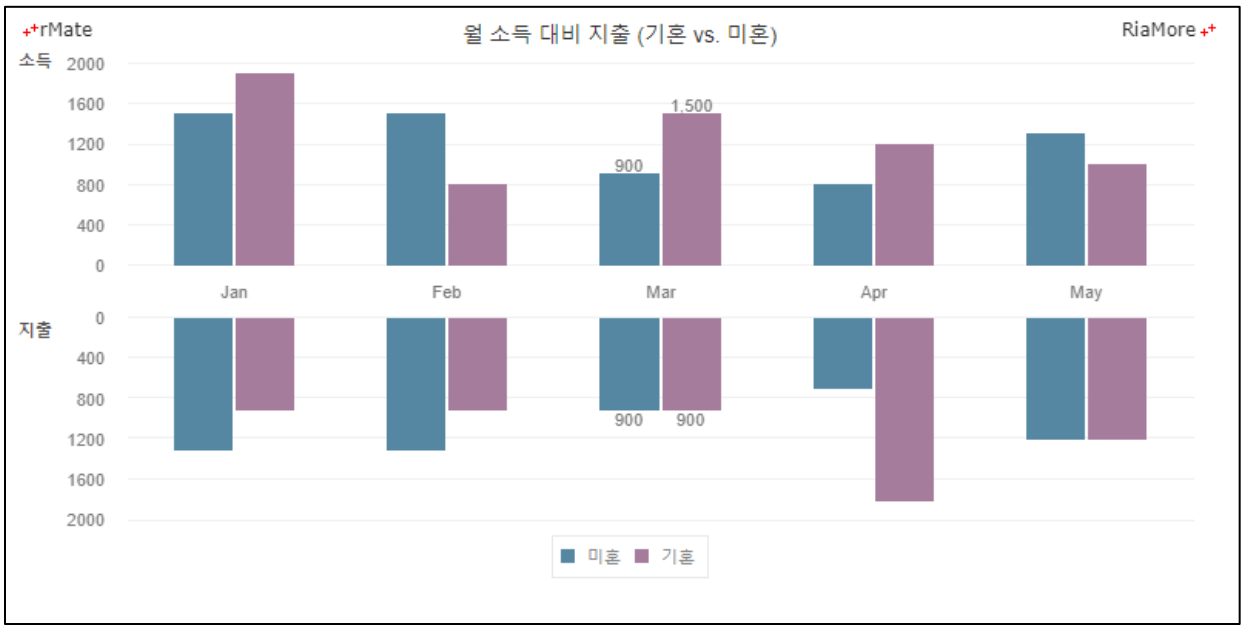


See the CodePen 알메이트 차트 - 컬럼 윙 차트

다중 시리즈 컬럼 윙 차트

일반적인 컬럼 차트와 마찬가지로 다중 데이터 시리즈를 컬럼 윙 차트에 표현할 수 있습니다. 표현하려는 개수만큼 <Column2DWingSeries> 노드를 정의하여 다중 시리즈 컬럼 윙 차트를 생성합니다. 다음은 두 개의 데이터 시리즈를 가진 컬럼 윙 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DWingChart showDataTips="true" columnWidthRatio="0.58">
  ...
  <Column2DWingSeries yField="Profit" yFieldOpp="Profit2"
       showLabelVertically="true" labelPosition="outside" displayName="Single"
       displayNameOpp="Single" styleName="seriesLabel" showValueLabels="[2]"
       showValueLabelsOpp="[2]" halfWidthOffset="1">
  </Column2DWingSeries>
  <Column2DWingSeries yField="Cost" yFieldOpp="Cost2" showLabelVertically="true"
       labelPosition="outside" displayName="Married" displayNameOpp="Married"
       styleName="seriesLabel" showValueLabels="[2]" showValueLabelsOpp="[2]"
       halfWidthOffset="1">
  </Column2DWingSeries>
  ...
</Column2DWingChart>
```

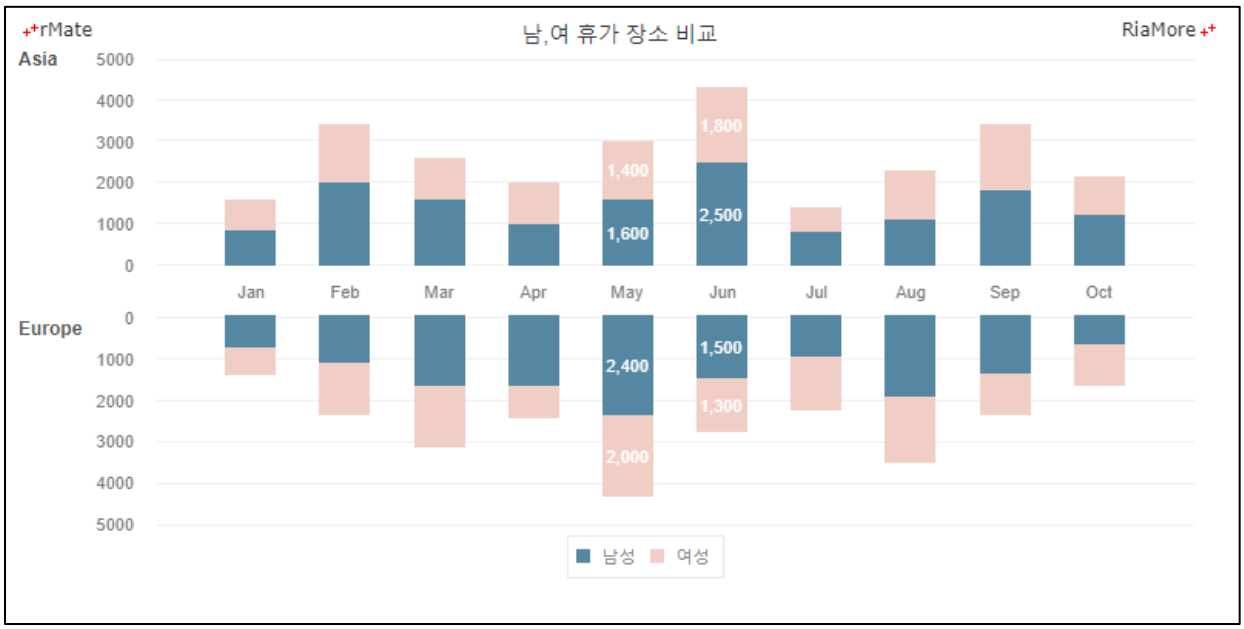


See the CodePen 알메이트 차트 - 다중 시리즈 컬럼 윙 차트

스택 컬럼 윙 차트

일반적인 컬럼 차트와 마찬가지로 스택 타입의 컬럼 윙 차트를 표현할 수 있습니다. 컬럼의 스택으로 표현하려는 개수만큼 <Column2DWingSeries> 노드를 정의하고, <Column2DWingChart> 노드의 type 속성을 “stacked” 으로 지정하여 스택 컬럼 윙 차트를 생성합니다. 다음은 두 개의 데이터 시리즈를 가진 스택 컬럼 윙 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DWingChart type="stacked" showDataTips="true" columnWidthRatio="0.58">
  ...
  <Column2DWingSeries yField="Profit" yFieldOpp="Profit2"
        showLabelVertically="true" labelPosition="outside" displayName="Single"
        displayNameOpp="Single" styleName="seriesLabel" showValueLabels="[2]"
        showValueLabelsOpp="[2]" halfWidthOffset="1">
  </Column2DWingSeries>
  <Column2DWingSeries yField="Cost" yFieldOpp="Cost2" showLabelVertically="true"
        labelPosition="outside" displayName="Married" displayNameOpp="Married"
        styleName="seriesLabel" showValueLabels="[2]" showValueLabelsOpp="[2]"
        halfWidthOffset="1">
  </Column2DWingSeries>
  ...
</Column2DWingChart>
```

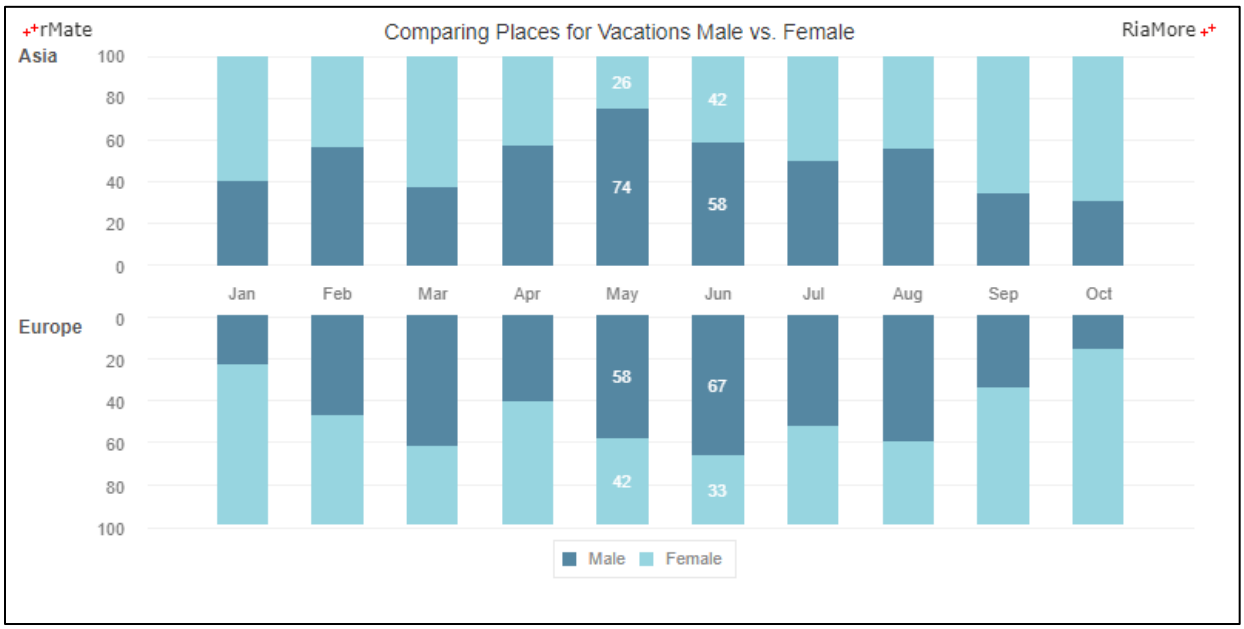


See the CodePen 알메이트 차트 - 스택 컬럼 윙 차트

100% 컬럼 윙 차트

일반적인 컬럼 차트와 마찬가지로 100% 타입의 컬럼 윙 차트를 표현할 수 있습니다. 컬럼의 100% 스택으로 표현하려는 개수만큼 <Column2DWingSeries> 노드를 정의하고, <Column2DWingChart> 노드의 type 속성을 “100%” 으로 지정하여 100% 타입의 컬럼 윙 차트를 생성합니다. 다음은 두 개의 데이터 시리즈를 가진 100% 타입의 컬럼 윙 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DWingChart type="100%" showDataTips="true" columnWidthRatio="0.54">
  ...
  <Column2DWingSeries yField="Profit" yFieldOpp="Profit2" labelPosition="inside"
       displayName="Male" showValueLabels="[4,5]" showValueLabelsOpp="[4,5]"
       formatter="{nft}" color="#ffffff">
  </Column2DWingSeries>
  <Column2DWingSeries yField="Cost" yFieldOpp="Cost2" labelPosition="inside"
       displayName="Female" showValueLabels="[4,5]" showValueLabelsOpp="[4,5]"
       formatter="{nft}" color="#ffffff">
  </Column2DWingSeries>
  ...
</Column2DWingChart>
```

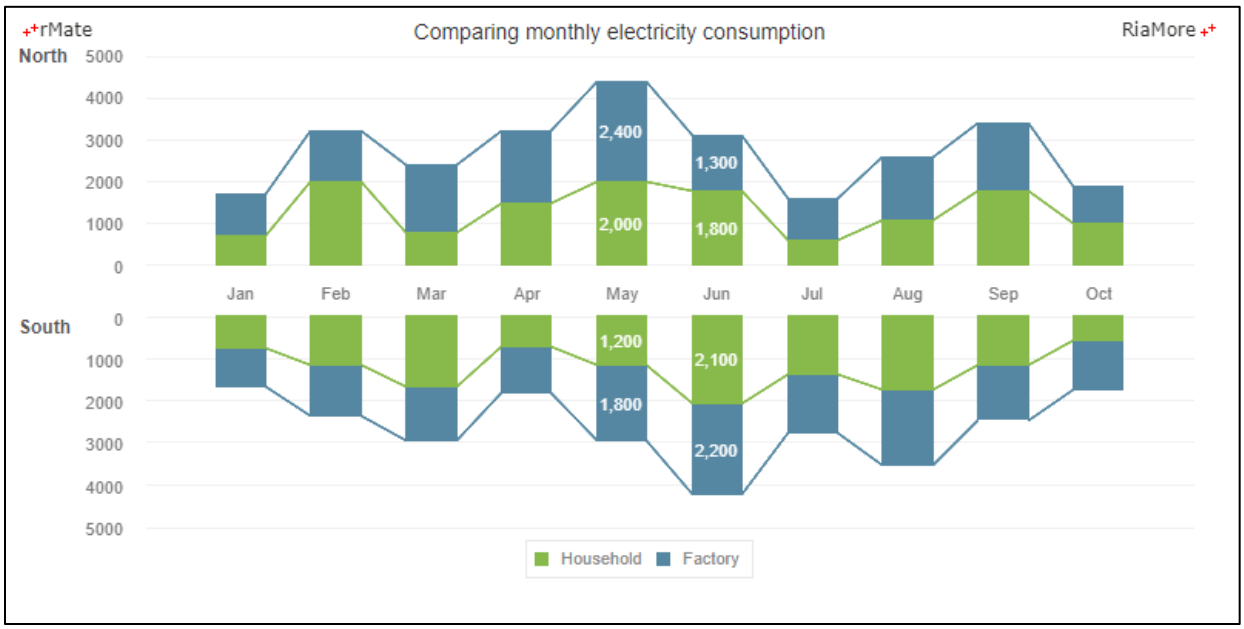


See the CodePen 알메이트 차트 - 100% 컬럼 윙 차트

컬럼 윙 차트 연결선 표시

컬럼 윙 차트에 표시되는 세로 막대의 값들을 연결하는 선을 표시할 수 있습니다. 컬럼 윙 차트에 연결선을 표시하기 위해서는 <Column2DWingSeries> 노드의 lineToEachItems 속성을 “true” 로 지정합니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DWingChart type="stacked" showDataTips="true" columnWidthRatio="0.54">
  ...
  <Column2DWingSeries yField="Profit" yFieldOpp="Profit2" lineToEachItems="true"
       labelPosition="inside" displayName="Household" showValueLabels="[4,5]"
       showValueLabelsOpp="[4,5]" color="#ffffff">
  </Column2DWingSeries>
  <Column2DWingSeries yField="Cost" yFieldOpp="Cost2" lineToEachItems="true"
       labelPosition="inside" displayName="Factory" showValueLabels="[4,5]"
       showValueLabelsOpp="[4,5]" color="#ffffff">
  </Column2DWingSeries>
  ...
</Column2DWingChart>
```

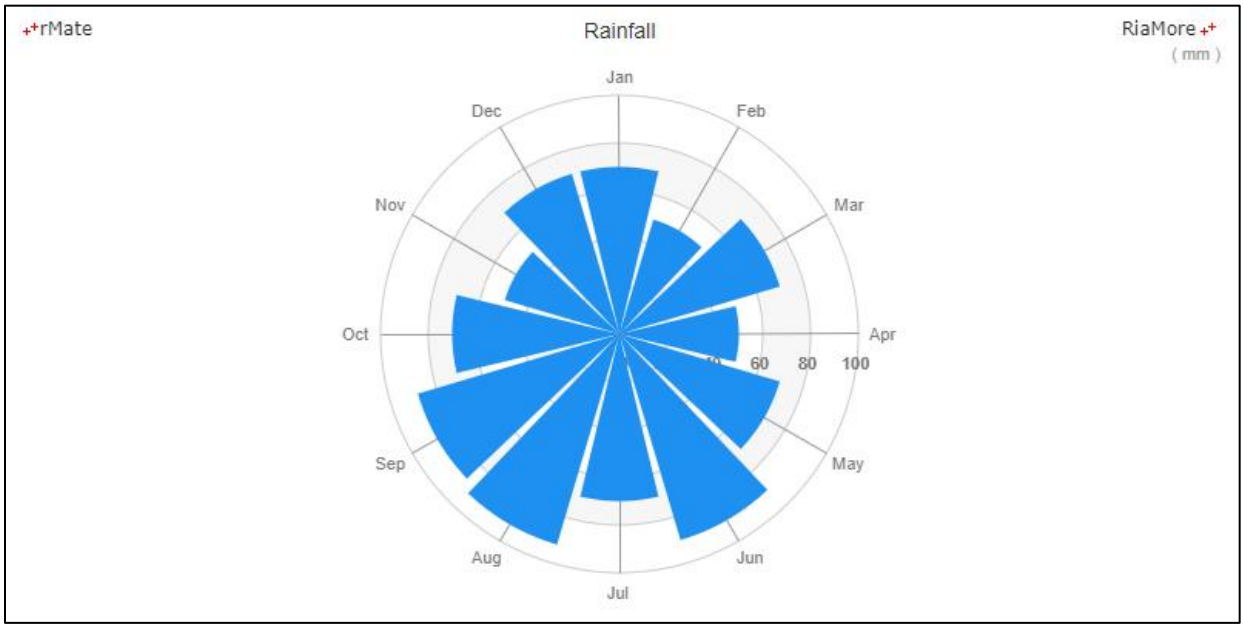


See the CodePen 알메이트 차트 - 컬럼 윙 차트에 연결선 표시

4.22 윈드로즈 차트

윈드로즈 차트는 특정 위치에서 바람의 방향과 속도의 크기를 시간별로 간결하게 표현하는 용도로 개발된 차트입니다. 최근에는 다양한 용도로 활용되는 폴라 차트(Polar Chart)의 한 유형이며, 일반적으로 8 ~ 16 개의 방사형 바퀴살을 한 차트에 표현하는 것이 적절합니다. 여러 개의 데이터 시리즈가 레이아웃에 정의되면 스택 형태의 방사형 바퀴살로 표현됩니다. 다음은 월별 강수량을 윈드로즈 차트로 표현하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<WindRoseChart showDataTips="true" dataTipDisplayMode="mouse" roseRatio="0.9">
  ...
  <WindRoseSeries field="Vancouver" displayName="Vancouver">
  </WindRoseSeries>
  ...
</WindRoseChart>
```



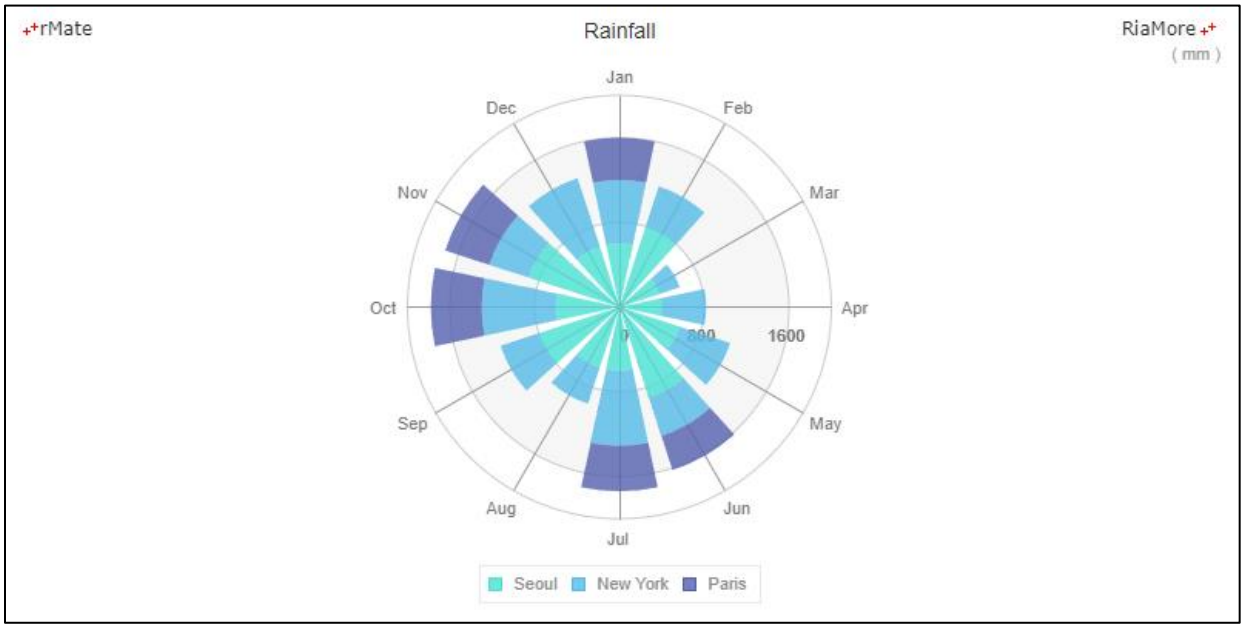
See the CodePen 알메이트 차트 - 윈드로즈 차트

스택 윈드로즈 차트

윈드로즈 차트 생성을 위한 레이아웃에 하나 이상의 데이터 시리즈를 정의할 경우, 스택 형태로 표현됩니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 3

개의 데이터 시리즈가 정의되었고, 각 데이터 시리즈에 대한 색(color)과 투명도(alpha = “0.8”)가 <SolidColor> 노드에 설정되었습니다.

```
<WindRoseChart showDataTips="true" dataTipDisplayMode="mouse" paddingBottom="30">
  ...
  <WindRoseSeries field="Seoul" displayName="Seoul">
  </WindRoseSeries>
  <WindRoseSeries field="NewYork" displayName="New York">
  </WindRoseSeries>
  <WindRoseSeries field="Paris" displayName="Paris">
  </WindRoseSeries>
  ...
</WindRoseChart>
```



See the CodePen 알메이트 차트 - 스택 윈드로즈 차트

4.23 캔들스틱 차트

캔들스틱 차트는 주식, 선물과 같은 상품의 거래를 표현하고 분석하는데 유용한 차트입니다. 하나의 캔들스틱은 특정시점(일)의 가격 변동에 대한 4 가지 정보를 표현하는데 이는 시가, 종가, 저가, 고가로 이루어져 있습니다. 그러나 전문적인 주식 차트와 같이 마우스를 이용하여 특정한 지점에 선을 표시하거나 기호를 삽입하는 것과 같은 기능은 지원하지 않습니다. 캔들스틱 차트에서 지원하는 기능은 다음과 같습니다.

- 차트에 최소값, 최대값을 표시할 수 있습니다.
- 최소값, 최대값을 사용자가 원하는 형태로 표현할 수 있습니다.
- 공시에 대한 데이터가 존재한다면 공시기호로 이를 표현할 수 있습니다.
- 공시기호를 아이템 렌더러(itemRenderer)를 통해 표현가능하며 div 객체를 이용하여 특정 문자나 이미지를 표시할 수 있습니다.
- div 로 표현된 공시기호는 마우스 클릭 이벤트를 받을 수 있습니다.
- 시가와 종가에 따라 선 색상, 막대 테두리, 막대 배경 색상 등을 변경할 수 있습니다.
- 차트에 보여지는 데이터 아이템의 개수를 변경 할 수 있습니다.

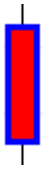
캔들스틱 차트는 <Candlestick2DChart> 노드의 series 속성값에 <Candlestick2DSeries> 노드를 설정하여 생성할 수 있습니다. 캔들스틱에 표현되는 시가, 종가, 저가, 고가 데이터는 <Candlestick2DSeries> 노드의 다음 속성에 설정합니다.

속성명	유효값 (*: 기본값)	설명
openField	텍스트	시가 정보가 저장된 필드명을 지정합니다.
closeField	0 과 1(*) 사이의 숫자	종가 정보가 저장된 필드명을 지정합니다.
lowField	0(*) 과 1 사이의 숫자	저가 정보가 저장된 필드명을 지정합니다.
highField	0(*) 과 1 사이의 숫자	고가 정보가 저장된 필드명을 지정합니다.

캔들스틱의 선과 배경의 스타일은 <Candlestick2DSeries> 노드의 다음 속성에 설정합니다.

속성명	유효값 (*: 기본값)	설명
fill	<SolidColor>	종가가 시가보다 큰 캔들스틱의 배경색의 스타일을 지정합니다.
stroke	<Stroke>	종가가 시가보다 큰 캔들스틱의 선의 스타일을 지정합니다.
boxStroke	<Stroke>	종가가 시가보다 큰 캔들스틱의 박스선의 색의 스타일을 지정합니다..
declineFill	<SolidColor>	종가가 시가보다 작은 캔들스틱의 배경색의 스타일을 지정합니다.
declineStroke	<Stroke>	종가가 시가보다 작은 캔들스틱의 선의 스타일을 지정합니다.
declineBoxStroke	<Stroke>	종가가 시가보다 작은 캔들스틱의 박스선의 색의 스타일을 지정합니다.

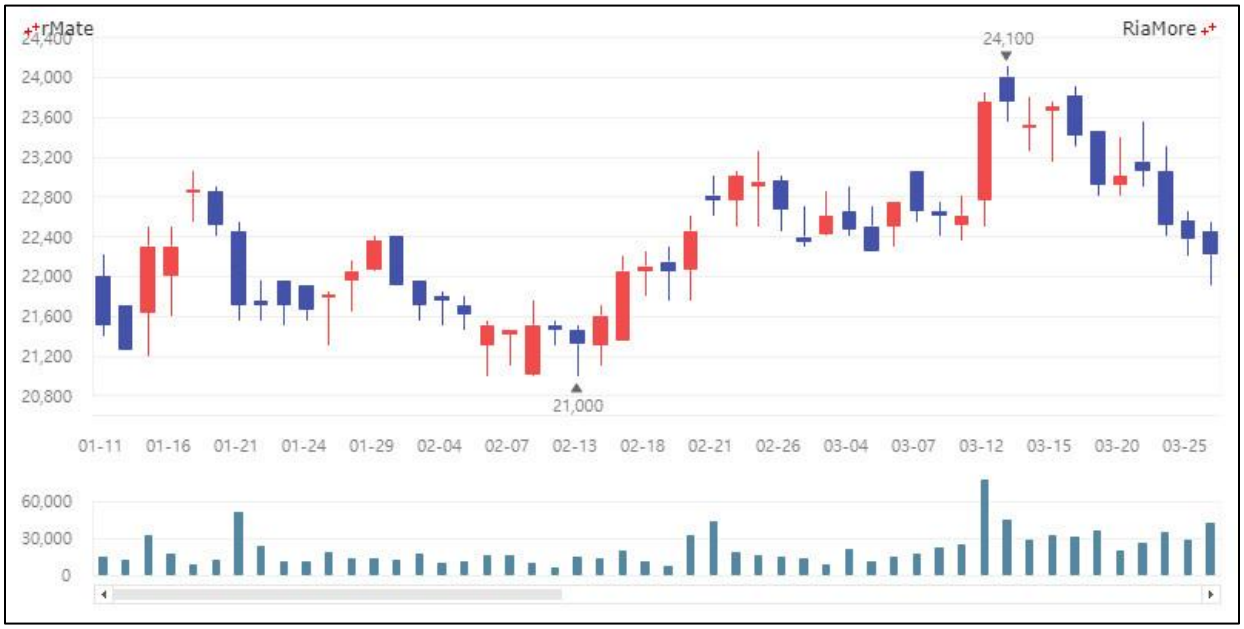
다음은 위에서 설명한 속성에 값을 적용해서 표시된 캔들스틱의 예제입니다.

캔들스틱	속성값 설정
	<stroke> <Stroke color="#000000"/> </stroke>
	<fill> <SolidColor color="#ff0000"/> </fill>
	<boxStroke> <Stroke color="#0000ff" weight="3"/> </boxStroke>

캔들스틱 차트는 일반적으로 하단에 보조 차트를 함께 보여줍니다. 보조 차트는 컬럼 차트로 표현하며 일반적으로 메인 차트의 캔들스틱에 대한 거래량이나 거래금액 정보를 보여줍니다. 메인 차트와 보조 차트는 <DualChart> 노드에 <mainChart> 속성과 <subChart> 속성으로 정의하는데 구조는 다음과 같습니다.

```
<DualChart leftGutterSyncEnable="true" rightGutterSyncEnable="true">
  <mainChart>
    <Candlestick2DChart showDataTips="true" paddingBottom="0"
       dataTipDisplayMode="axis">
      ...
    </Candlestick2DChart>
  </mainChart>
  <subChart>
    <Column2DChart showDataTips="true" height="20%" gutterTop="6" gutterBottom="6">
      ...
    </Column2DChart>
  </subChart>
</DualChart>
```

다음은 캔들스틱 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 메인 차트와 보조 차트에 표시되는 십자선(CrossRangeZoomer)을 동기화하기 위해서 <CrossRangeZoomer> 노드의 syncCrossRangeZoomer 속성에 서로의 <CrossRangeZoomer> 속성의 고유값(id = “candleCRZ”, id = “columnCRZ”)을 지정하였습니다. 그리고 한 화면에 표시되는 데이터 아이템(캔들스틱)의 개수를 보기에 적당한 수(visibleItemSize = “50”)로 설정하고, 화면에 표시되지 못하는 데이터 아이템을 스크롤바를 이용하여 조회할 수 있도록 <DualScrollBar> 노드를 <dataSelector> 속성에 지정하였습니다.

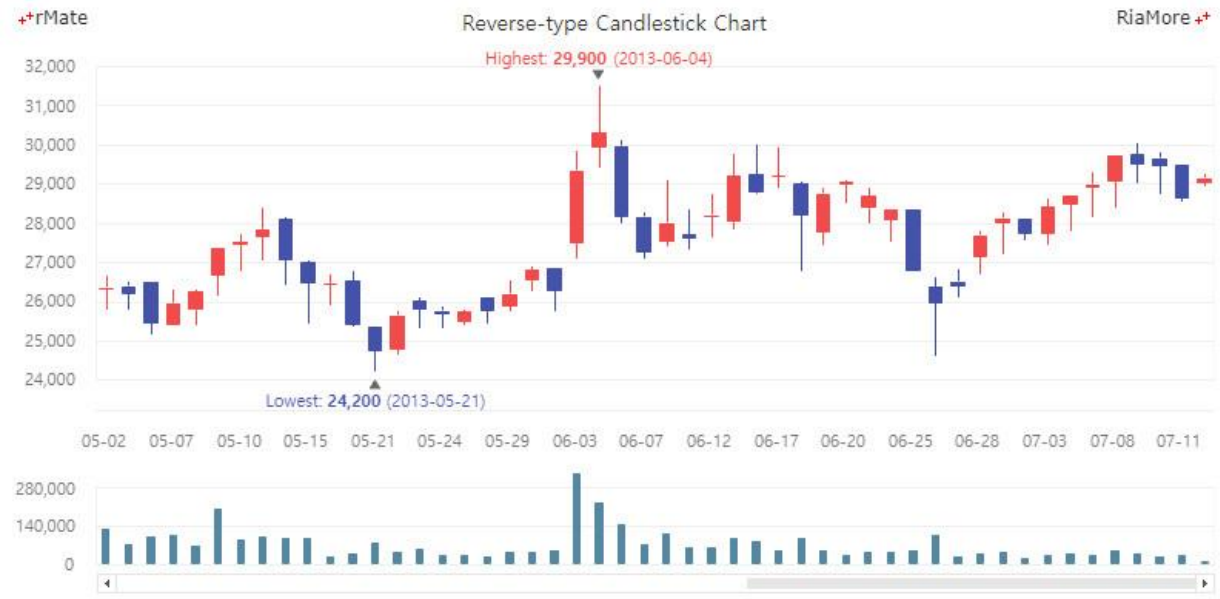


See the CodePen 알메이트 차트 - 캔들스틱 차트

역방향 캔들스틱 차트

<DualScrollBar> 노드의 inverted 속성을 “true” 로 설정하면 최근 데이터를 먼저 화면에 표시할 수 있습니다. 따라서 최초에 차트가 생성되면 스크롤바 커서는 우측 끝에 위치하게 됩니다.

```
<dataSelector>
  <DualScrollBar inverted="true" visibleItemSize="50"/>
</dataSelector>
```



See the CodePen 알메이트 차트 - 역방향 캔들스틱 차트

공시 기호 표시

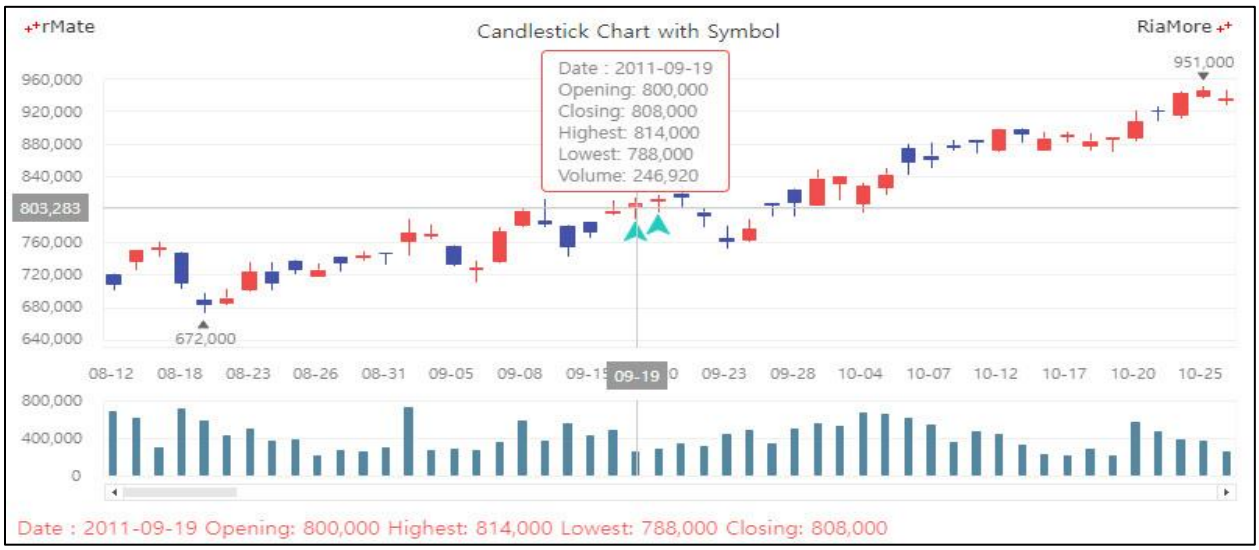
캔들스틱 차트에서 특정 데이터에 대한 공시 정보를 기호로 표시할 수 있습니다. 공시 정보를 기호로 표시하기 위해서 <Candlestick2DSeries> 노드의 symbolType 속성을 “normal” 로 지정하고 symbolRenderer 속성에 표시하고자 하는 기호를 지정합니다.

속성명	유효값 (*: 기본값)	설명
symbolField	텍스트	공시 정보를 표시할지 여부가 표시된 데이터 필드명을 지정합니다.
symbolType	normal(*), div	공시 정보의 표시 유형을 지정합니다.

		normal: symbolRenderer 속성에 지정된 기호를 표시합니다. div: symbolLabelField 속성에 지정된 데이터 필드의 값을 표시합니다.
symbolRenderer	UpArrowItemRenderer(*), DiamondItemRenderer, CircleItemRenderer, TriangleItemRenderer, CrossItemRenderer, XShapeItemRenderer, IShapeItemRenderer, RectangleItemRenderer	symbolType 속성값이 “normal” 일 경우, 공시 정보로 표시할 기호를 지정합니다.
symbolLabelField	0(*) 과 1 사이의 숫자	symbolType 속성값이 “div” 일 경우, 공시 정보로 표시할 텍스트가 저장된 데이터 필드명을 지정합니다.

다음은 공시 정보를 삼각형 기호(symbolRenderer = “TriangleItemRenderere”)로 표시하는 캔들스틱 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Candlestick2DSeries openField="open" closeField="close" highField="high"
        lowField="low" showMinValueLabel="true" showMaxValueLabel="true"
        symbolField="gongsi" symbolRenderer="TriangleItemRenderere">
...
</Candlestick2DSeries>

  "date" : "20110916",
  "open" : 794000,
  "high" : 811000,
  "low" : 792000,
  "close" : 798000,
  "admnt" : 476632
},{
  "date" : "20110919",
  "open" : 800000,
  "high" : 814000,
  "low" : 788000,
  "close" : 808000,
  "admnt" : 246920,
  "gongsi" : true
},{
```

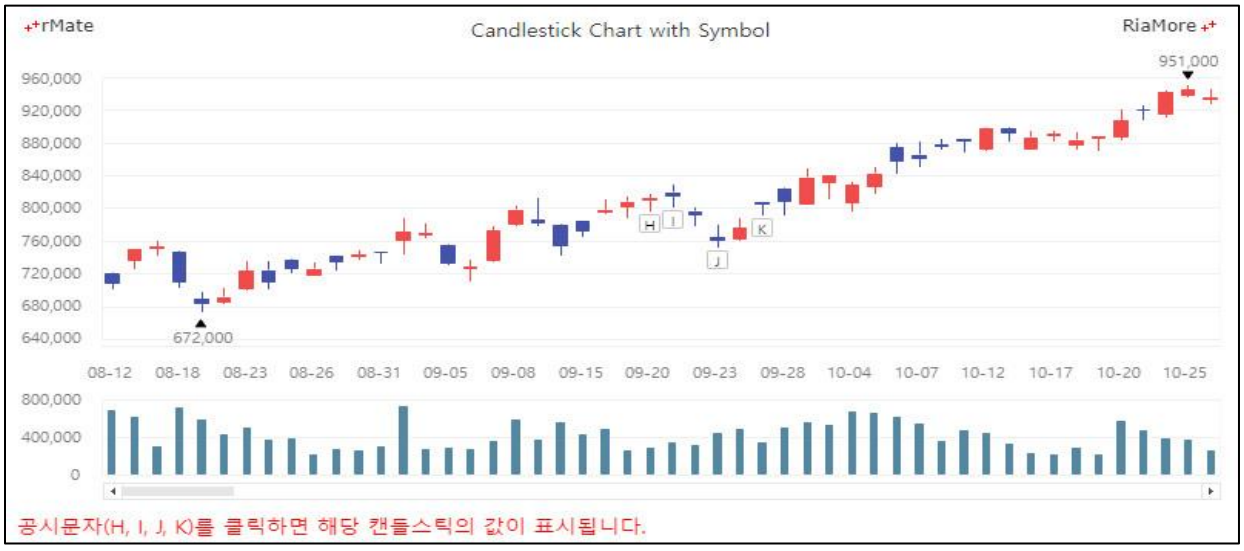


See the CodePen 알메이트 차트 - 캔들스틱 차트 - 공시기호 표시

공시 문자 표시

다음은 공시 정보를 문자로 표시하는 캔들스틱 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 symbolType 속성이 “div” 로 지정되었고, 공지 정보로 표시할 문자가 저장된 데이터 필드명이 symbolLabelField 속성에 지정되었습니다. 공시문자를 클릭하면 실행될 자바스크립트 함수명은 symbolClickJsFunction 속성에 지정되었습니다.

```
<Candlestick2DSeries openField="open" closeField="close" highField="high"
        lowField="low" showMinValueLabel="true" showMaxValueLabel="true"
        symbolField="gongsi" symbolType="div" symbolLabelField="gongsiLabel"
        symbolClickJsFunction="gongsiDataFunc" symbolColor="#000">
...
</Candlestick2DSeries>
```

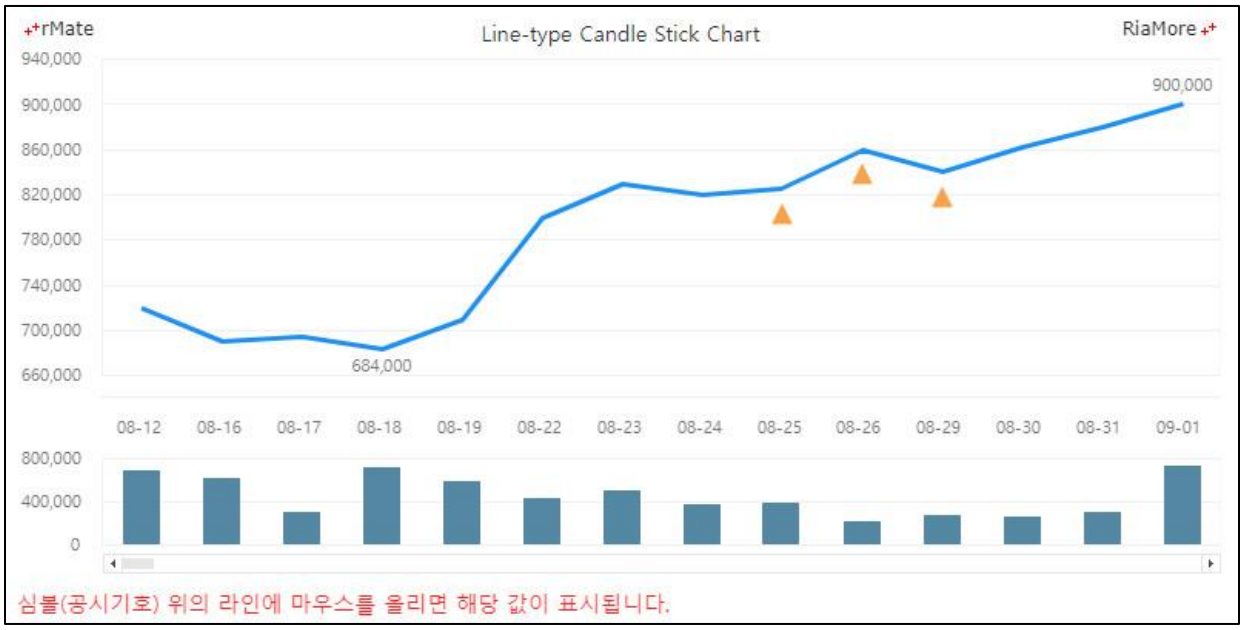


See the CodePen 알메이트 차트 - 캔들스틱 차트 - 공시문자 표시

캔들라인 차트

위에서 설명한 캔들스틱 차트와 동일하지만 데이터 포인트에 캔들스틱을 표시하지 않고 데이터 포인트를 하나의 선으로 연결하여 표현할 수 있습니다. 특정 기간 동안의 데이터의 변동 현황을 파악하는 목적으로 유용하게 활용될 수 있습니다. 캔들라인 차트는 <CandleLine2DSeries> 노드의 series 속성값에 <CandleLine2DSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 캔들라인 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<DualChart leftGutterSyncEnable="true" rightGutterSyncEnable="true">
  <mainChart>
    <CandleLine2DChart showDataTips="true" paddingBottom="0"
       dataTipJsFunction="dataTipFunc">
      ...
      <CandleLine2DSeries yField="open" showMinValueLabel="true"
       showMaxValueLabel="true" symbolField="gongsi"
       symbolRenderer="TriangleItemRenderer">
      </CandleLine2DSeries>
    </CandleLine2DChart>
  </mainChart>
  <subChart>
    <Column2DChart showDataTips="true" height="20%" paddingTop="0"
       paddingBottom="0" gutterTop="6" gutterBottom="6">
      ...
      <Column2DSeries yField="admnt" itemRenderer="BoxItemRenderer">
      </Column2DSeries>
    </Column2DChart>
  </subChart>
</DualChart>
```



See the CodePen 알메이트 차트 - 캔들라인 차트

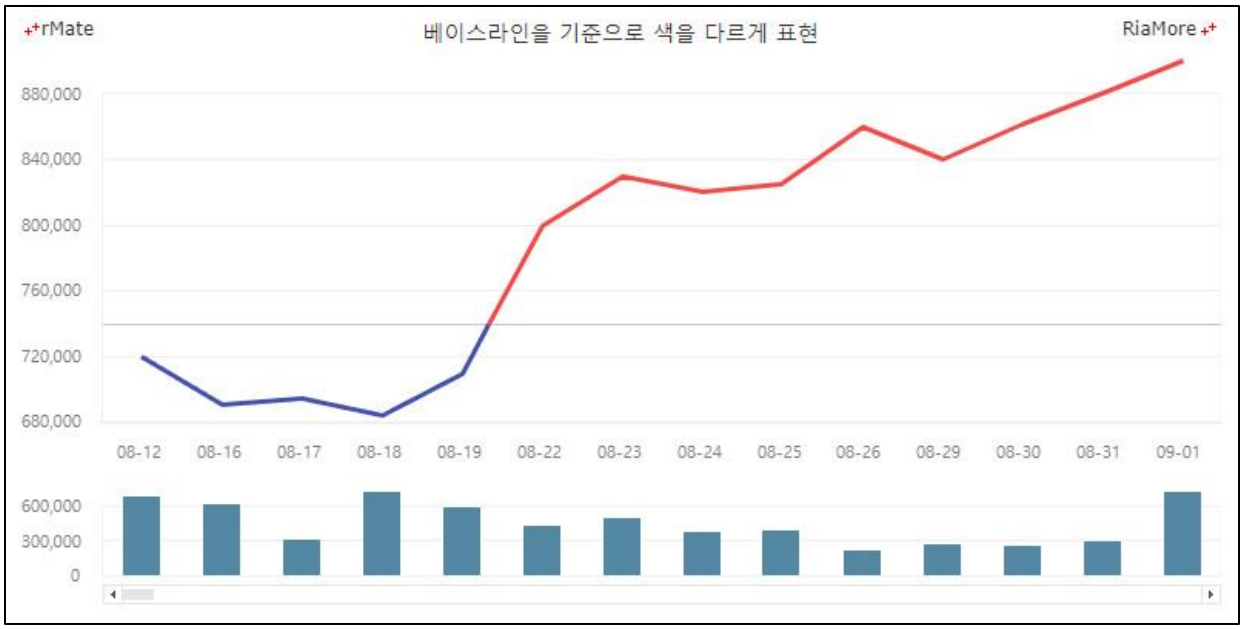
캔들라인 차트에 직선(베이스라인, baseline)을 표시하고, 이 선의 값보다 큰 선과 작은 선의 스타일을 다르게 적용하여 표현할 수 있습니다. 다음은 이에 대한 코드와 이를 적용하여 출력한 차트의 예제입니다.

베이스 라인선의 값보다 큰 선과 작은 선의 스타일을 다르게 적용하는 코드.

```
<CandleLine2DSeries yField="open" baseValue="740000" >
  <lineStroke>
    <Stroke color="#f14b4b" weight="3"/>
  </lineStroke>
  <lineDeclineStroke>
    <Stroke color="#4452aa" weight="3"/>
  </lineDeclineStroke>
</CandleLine2DSeries>
```

베이스 라인을 표시하는 코드

```
<backgroundElements>
  <AxisMarker>
    <lines>
      <AxisLine value="740000">
        <stroke>
          <Stroke color="#cccccc"/>
        </stroke>
      </AxisLine>
    </lines>
  </AxisMarker>
</backgroundElements>
```

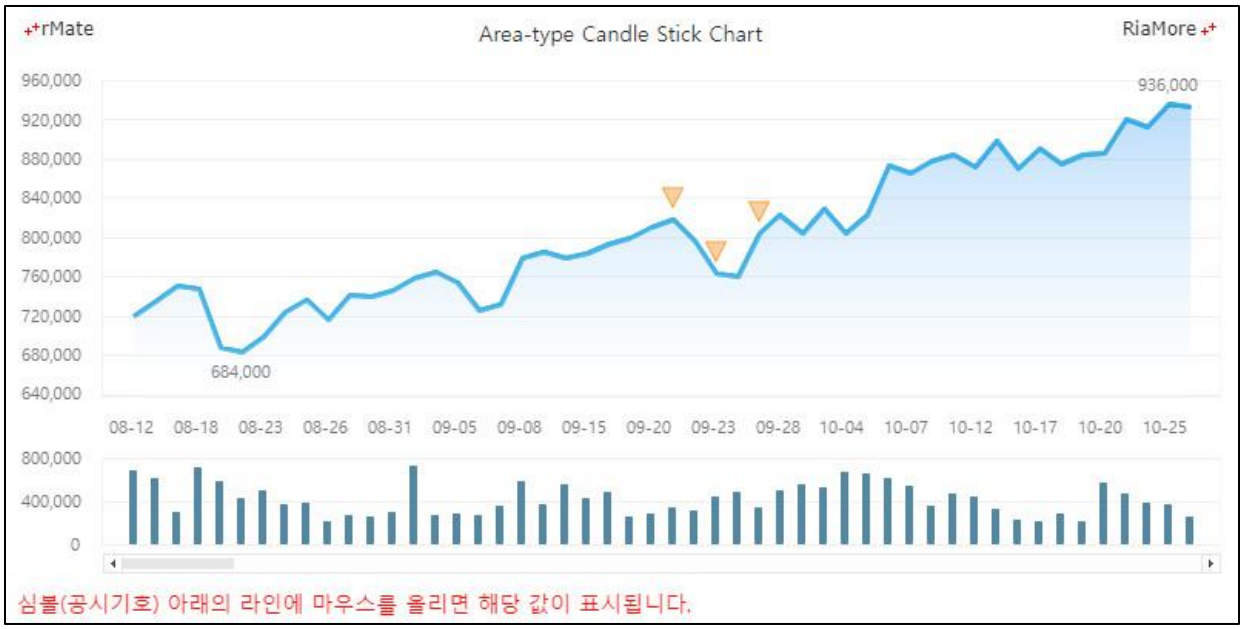


See the CodePen 알메이트 차트 - 캔들라인 차트 (베이스라인 표시)

캔들 영역 차트

캔들영역 차트는 캔들라인 차트와 동일하지만 데이터 포인트를 연결하는 라인과 가로 축 사이의 영역을 특정 색상으로 표현합니다. 캔들영역 차트는 <CandleArea2DChart>노드의 series 속성값에 <CandleArea2DSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 캔들영역 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<DualChart leftGutterSyncEnable="true" rightGutterSyncEnable="true">
  <mainChart>
    <CandleArea2DChart showDataTips="true" paddingBottom="0"
        dataTipJsFunction="dataTipFunc">
      ...
      <CandleArea2DSeries yField="open" showMinValueLabel="true"
       showMaxValueLabel="true" symbolField="gongsi"
       symbolRenderer="DownTriangleItemRenderer" symbolPosition="top">
      </CandleArea2DSeries>
      ...
    </CandleArea2DChart>
  </mainChart>
  <subChart>
    <Column2DChart showDataTips="true" height="20%" paddingTop="0"
        paddingBottom="0" gutterTop="6" gutterBottom="6">
      ...
      <Column2DSeries yField="admnt" itemRenderer="BoxItemRenderer">
      </Column2DSeries>
      ...
    </Column2DChart>
  </subChart>
</DualChart>
```



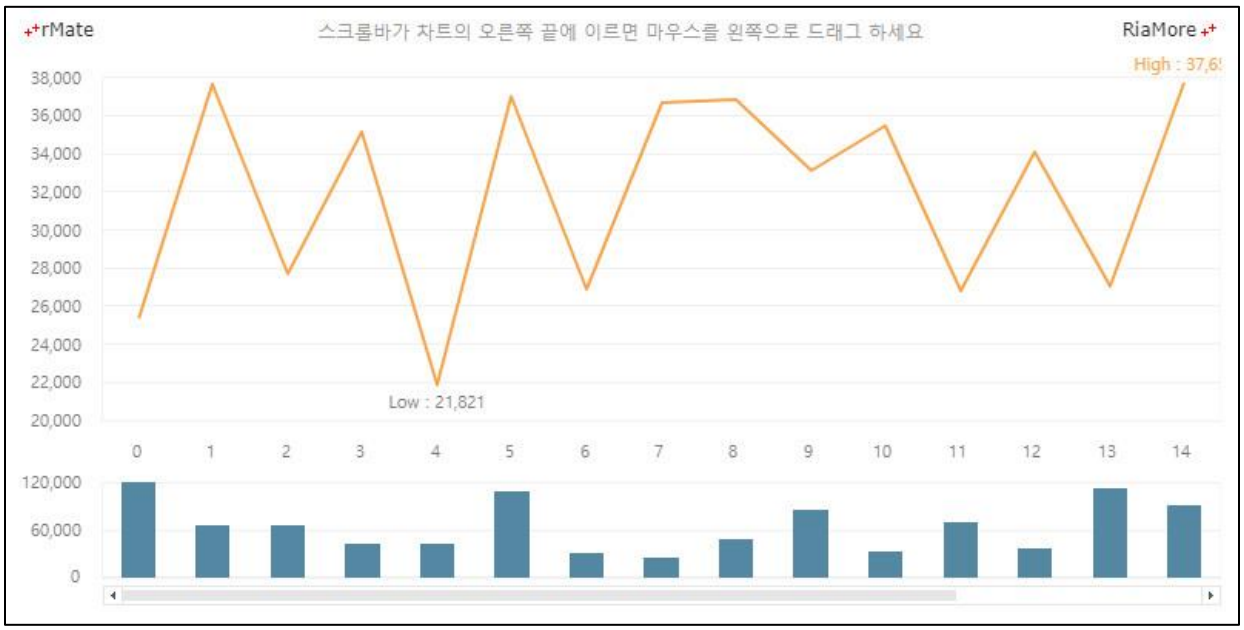
See the CodePen 알메이트 차트 - 캔들영역 차트

레이지 로드

차트가 생성되는 시점에 모든 데이터를 로드하지 않고 레이지 로드 방식을 캔들스틱 차트에 적용할 수 있습니다. 차트 데이터에 레이지 로드를 적용하려면 <CandleLine2DChart> (<Candlestick2DChart>, <CandleArea2DChart>) 노드의 lazyJsFunction 속성에 필요한 양의 데이터를 Ajax 방식으로 로드하는 자바스크립트 함수명을 지정해야 합니다. 이 때 차트의 데이터 함수는 데이터가 존재하는 URL 을 호출하는 setDataURL() 함수를 사용해야 합니다.

다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서 스크롤바의 커서가 우측 끝에 위치했을 때 메인 차트 영역에 마우스를 놓고 왼쪽으로 드래그하면 레이지 로드가 실행됩니다.

```
var dataURL = "https://www.riamore.net/demo/chart/makeLazyData.jsp?index=";
function chartReadyHandler(id) {
  document.getElementById(id).setLayout(layoutStr);
  document.getElementById(id).setDataURL(dataURL + 0);
}

<CandleLine2DChart showDataTips="true" paddingBottom="0"
       lazyJsFunction="lazyDataFunc">
  ...
</CandleLine2DChart>

var xhr, // ajax object
index = 19; // The total number of data processed.
function lazyDataFunc(id){
  var param = {};
  param.url = dataURL + (index + 1);
  param.success = function() {
    var data;
    if (xhr.readyState == 4 && xhr.status >= 200 && xhr.status < 300) {
      if (xhr.responseXML.xml)
        data = xhr.responseXML.xml;
      else
        data = xhr.responseXML;
      document.getElementById(id).addData(data);
      index += 20; // Data Size: 20
    }
  }

  ajax(param);
}
```



See the CodePen 알메이트 차트 - 캔들라인 차트 - 레이지 로드

4.24 트리맵 차트

트리맵 차트는 계층 구조의 데이터를 표현하는데 적합한 차트로써 데이터의 상하 관계를 중첩된 사각형(nested rectangle) 형태로 표현합니다. 계층형 구조의 데이터를 표현하기 때문에 데이터도 이에 맞도록 구성되어 있어야 합니다. 다음은 강북구의 수유와 미아 지역 식당에서 중국 음식에 대한 점수를 트리맵 차트로 표현하기 위한 데이터의 예제입니다. 한 계층의 데이터는 이름(name)과 항목(items)의 쌍으로 이루어져 있고, 하위 계층의 데이터는 항목(items) 필드에 다시 이름(name)과 항목(items)의 쌍으로 정의됩니다.

```
var chartData = [{
  "name":"Gangbuk-gu",
  "items": [{
    "name": "Suyu",
    "items" : [
      {"Food":"Radish", "Value":30},
      {"Food":"Soup", "Value":152},
      {"Food":"Seafood", "Value":243},
      {"Food":"Stew", "Value":298},
      {"Food":"Noodles", "Value":383},
      {"Food":"Fry", "Value":203},
      {"Food":"Rice", "Value":98},
      {"Food":"Salad", "Value":140},
      {"Food":"Grilled meat", "Value":244},
      {"Food":"Say source", "Value":50}
    ]
  },{
    "name" : "Mia",
    "items" : [
      {"Food":"Noodles", "Value":321},
      {"Food":"stew", "Value":295},
      {"Food":"Grilled meat", "Value":272},
      {"Food":"Seafood", "Value":221},
      {"Food":"Fry", "Value":213},
      {"Food":"Soup", "Value":190},
      {"Food":"Salad", "Value":135},
      {"Food":"Rice", "Value":129}
    ]
  }]
}];
```

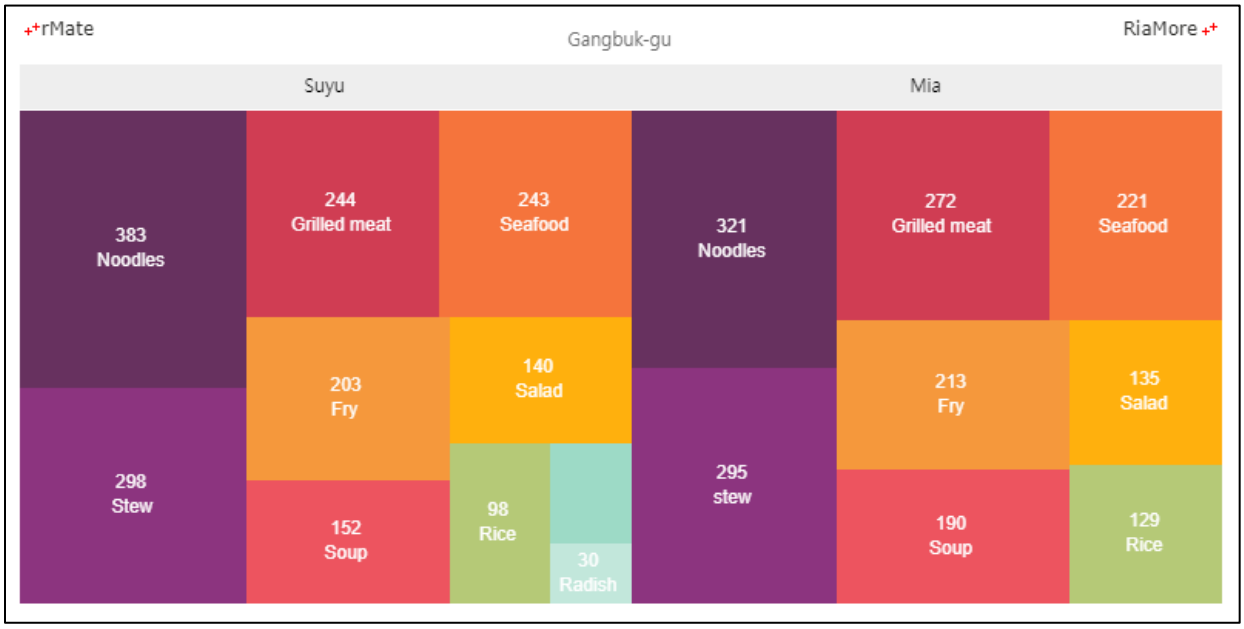
트리맵 차트는 <TreeMapChart> 노드의 series 속성값에 <TreeMapSeries> 노드를 설정하여 생성할 수 있습니다. <TreeMapSeries> 노드에 설정되는 주요 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
groupField	텍스트	계층형 데이터의 그룹 필드명을 지정합니다.

groupNameField	텍스트	계층형 데이터의 그룹명을 지정합니다.
showGroupHeader	true(*), false	계층형 데이터의 그룹 명을 차트에 표시할지 여부를 지정합니다.
nameField	텍스트	데이터의 레이블이 저장된 필드명을 지정합니다.
weightField	텍스트	사각형의 크기에 해당하는 데이터 값이 저장된 필드명을 지정합니다.

다음은 위에서 설명된 데이터를 이용하여 트리맵 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<TreeMapChart showDataTips="true" buttonMode="true" selectionMode="single">
  <series>
    <TreeMapSeries weightField="Value" nameField="Food" groupField="items"
       groupNameField="name" displayName="tree" showGroupHeader="true"
       color="#ffffff" labelPosition="inside" labelJsFunction="labelFunc"
       groupHeaderFontColor="#000000" fontSize="12">
      <groupHeaderColor>
        <SolidColor color="#eeeeee" />
      </groupHeaderColor>
      <fills>
        <SolidColor color="#67315f"/>
        ...
      </fills>
    </TreeMapSeries>
  </series>
</TreeMapChart>
```

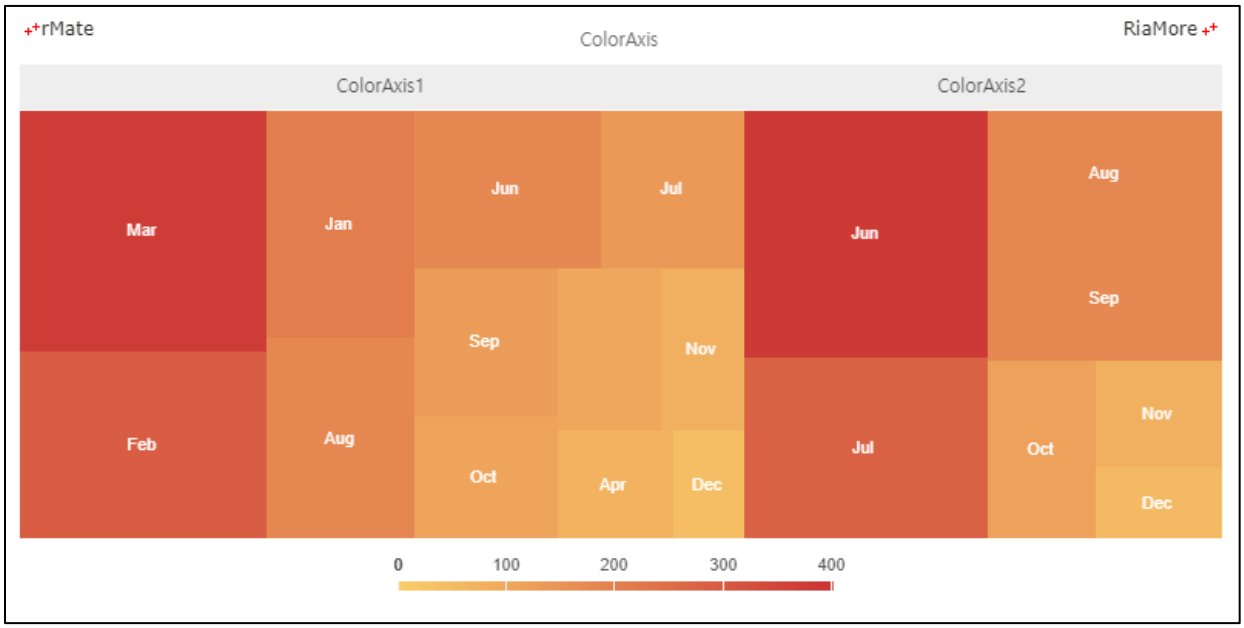


See the CodePen 알메이트 차트 - 트리맵 차트

트리맵 차트에서 컬러축 표시

트리맵 차트에서 색과 데이터 값을 연관시켜 표현할 수 있습니다. 이는 컬러축을 이용함으로써 가능한데, 컬러축은 <TreeMapChart> 노드의 <colorAxis> 속성에 <ColorAxis> 노드를 정의하여 생성할 수 있습니다. 다음은 컬러축이 표현된 트리맵 차트를 생성하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 색의 범위를 colors 속성에 "[#fcd26b, #cc3635]" 으로 지정하였습니다. 최소값(minimum = "0")의 색은 #fcd26b 이고, 최대값(maximum = "400")의 색은 #cc3635 이며, 컬러축에는 100 (interval = "100") 단위로 레이블이 표시됩니다. 사각형에 표시되는 색은 컬러축과 데이터 값이 연동되어 자동으로 표시됩니다.

```
<TreeMapChart showDataTips="true">
  <colorAxis>
    <ColorAxis id="colorAxis" maximum="400" minimum="0" interval="100"
       colors="[#fcd26b,#cc3635]"/>
  </colorAxis>
  ...
</TreeMapChart>
```



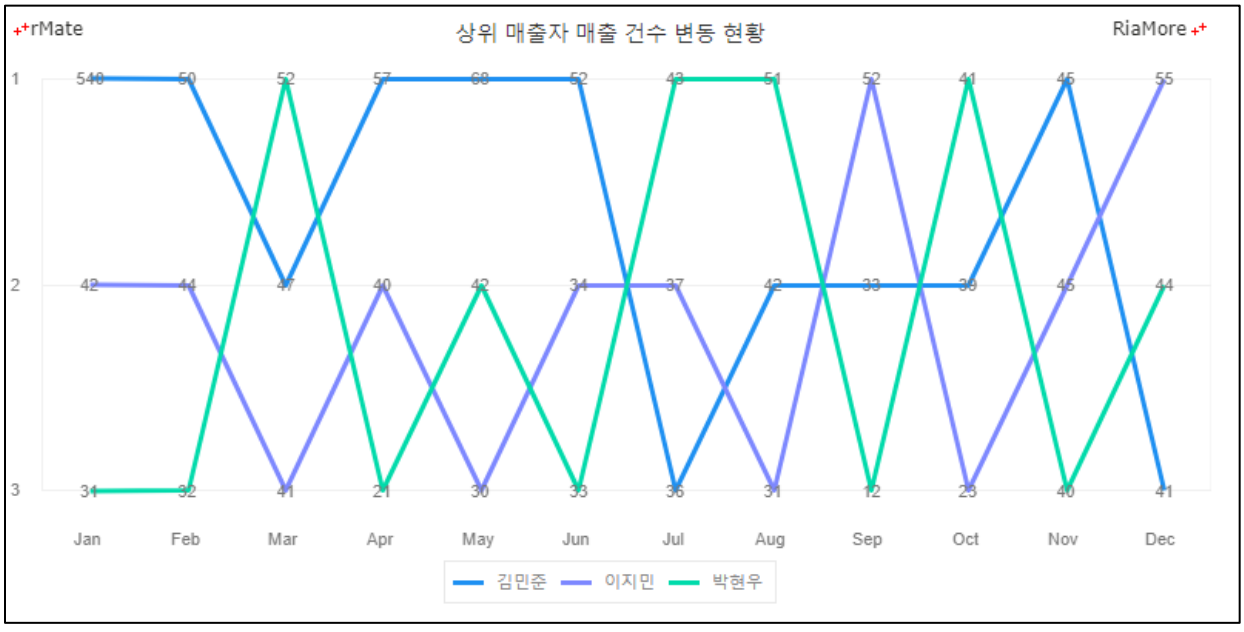
See the CodePen 알메이트 차트 - 트리맵 차트에서 컬러축 표시

4.25 범프 차트

범프 차트는 차트에 표현되는 요소(카테고리)들의 값의 크기 보다는 시간에 따른 성과순위의 변화를 파악하는데 도움을 주는 차트입니다. 순위를 나타내는 축은 차트의 좌측에 세로 축(Y 축)으로 표시되고, 시간은 차트의 하단에 가로 축(X 축)으로 표시됩니다. 범프 차트는 <Bump2DChart> 노드의 series 속성값에 <Bump2DSeries> 노드를 설정하여 생성할 수 있습니다.

다음은 범프 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 월별로 변동되는 매출 건수를 직선으로 연결하여 표현합니다.

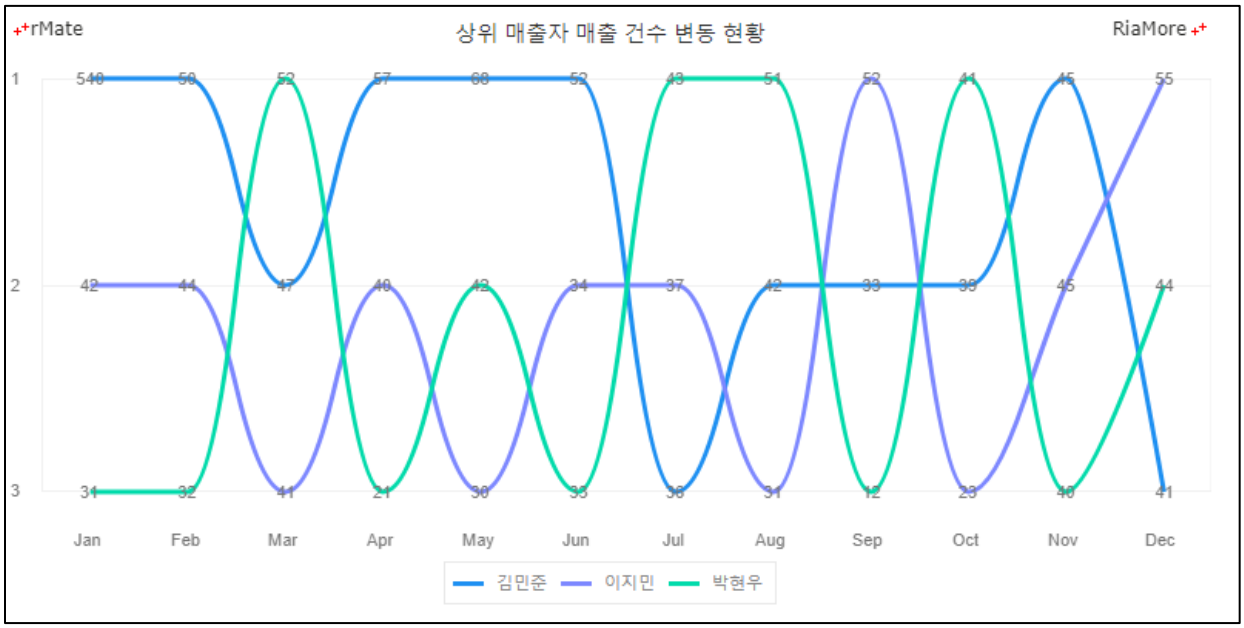
```
<Bump2DChart showDataTips="true" dataTipDisplayMode="axis">
  ...
    <Bump2DSeries id="bump1" yField="김민준" displayName="김민준"
       labelPosition="inside" form="segment">
    ...
    <Bump2DSeries id="bump2" yField="이지민" displayName="이지민"
       labelPosition="inside" form="segment">
    ...
    <Bump2DSeries id="bump3" yField="박현우" displayName="박현우"
       labelPosition="inside" form="segment">
  ...
</Bump2DChart>
```



See the CodePen 알메이트 차트 - 범프 차트

혹(범프, Bump)을 연결하는 선을 곡선으로 표현할 수 있습니다. 다음 예제에서는 월별로 변동되는 매출 건수를 곡선으로 연결하여 표현합니다.

```
<Bump2DChart showDataTips="true" dataTipDisplayMode="axis">
  ...
    <Bump2DSeries id="bump1" yField="김민준" displayName="김민준"
       labelPosition="inside" itemRenderer="CircleItemRenderer" form="curve">
    ...
    <Bump2DSeries id="bump2" yField="이지민" displayName="이지민"
       labelPosition="inside" itemRenderer="CircleItemRenderer" form="curve">
    ...
    <Bump2DSeries id="bump3" yField="박현우" displayName="박현우"
       labelPosition="inside" itemRenderer="CircleItemRenderer" form="curve">
  ...
</Bump2DChart>
```



See the CodePen 알메이트 차트 - 범프 차트 (곡선)

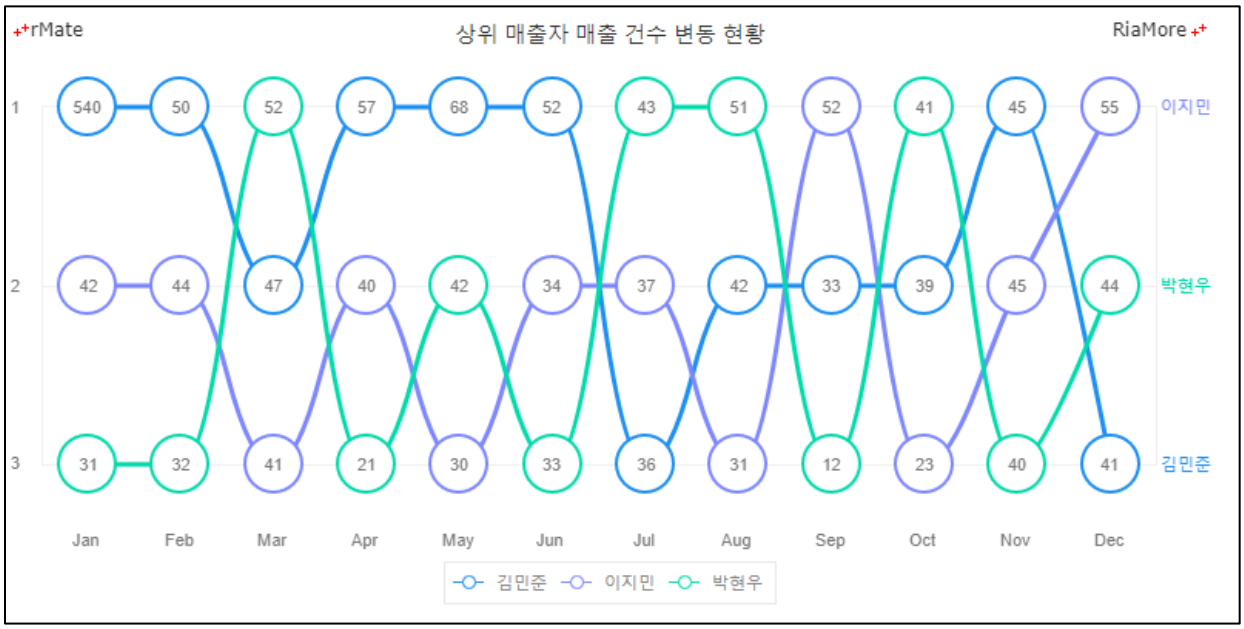
혹(범프, Bump)에 도형 표시

범프 차트에서 <Bump2DSeries> 노드의 itemRenderer 속성(기본값: “none”)을 설정하여 데이터 포인트에 특정한 모양의 도형을 표시할 수 있습니다. itemRenderer 속성에 설정 가능한 값과 표시되는 도형의 모양은 다음과 같습니다.

- CircleItemRenderer : 데이터 포인트에 원을 표시합니다.
- TriangleItemRenderer : 데이터 포인트에 삼각형을 표시합니다.
- RectangleItemRenderer : 데이터 포인트에 사각형을 표시합니다.
- DiamondItemRenderer : 데이터 포인트에 다이아몬드를 표시합니다.
- CrossItemRenderer : 데이터 포인트에 십자형을 표시합니다.
- XShapeItemRenderer : 데이터 포인트에 X 자 모양을 표시합니다.
- IShapeItemRenderer : 데이터 포인트에 I 자 모양을 표시합니다.

다음 예제 차트에서는 혹(범프, Bump)에 원을 표시하는 데이터 렌더러가 적용되었습니다.

```
<Bump2DChart showDataTips="true" dataTipDisplayMode="axis">
  ...
    <Bump2DSeries id="bump1" yField="김민준" displayName="김민준"
       labelPosition="inside" itemRenderer="CircleItemRenderer" form="curve">
    ...
    <Bump2DSeries id="bump2" yField="이지민" displayName="이지민"
       labelPosition="inside" itemRenderer="CircleItemRenderer" form="curve">
    ...
    <Bump2DSeries id="bump3" yField="박현우" displayName="박현우"
       labelPosition="inside" itemRenderer="CircleItemRenderer" form="curve">
  ...
</Bump2DChart>
```



See the CodePen 알메이트 차트 - 범프 차트 (범프에 원 표시)

4.26 워드클라우드 차트

워드클라우드 차트는 각 단어의 중요도(인기도)를 한눈에 알아볼 수 있도록 높은 시각적 효과를 제공하는 차트입니다. 단어의 중요도는 데이터에 나타난 단어의 빈도수가 얼마나 높은지 혹은 단어에 주어진 수치값(가중치)이 얼마나 큰지에 따라서 결정되며 차트 상에는 단어의 크기와 색상을 달리하여 표현됩니다. 워드클라우드 차트는 <WordCloudChart> 노드의 series 속성값에 <WordCloudSeries> 노드를 설정하여 생성할 수 있습니다. 차트에 표현되는 단어의 크기는 <WordCloudSeries> 노드에 정의되는 다음 두 속성에 의해서 결정됩니다.

속성명	유효값 (*: 기본값)	설명
minFontSize	숫자 기본값: 10	단어의 최소 폰트 크기를 지정합니다.
maxFontSize	숫자 기본값: 100	단어의 최대 폰트 크기를 지정합니다.

단어의 빈도수로 워드클라우드 차트 표현

다음은 데이터에 나타나는 단어의 빈도수에 의해서 차트에 표현되는 단어의 크기와 색상이 결정되는 워드클라우드 차트를 생성하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다. 단어의 빈도수에 따라서 적용될 색은 <WordCloudSeries> 노드의 <fills> 속성에 정의됩니다, 이 예제에서는 빈도수가 가장 높은 단어에 사용될 색으로 “#5473b3” 이 지정되었습니다.

```
data =
        ["PRAY","NEW","DO","DESIGN","CONCENTRATE","GREAT","LEAD","MOMENT","LOVE","DO
        ","NEW","PRAY","GREAT","DESIGN","CREATIVE","LOVE","STUNNING","PRAY","BABY","
        LOVELY","DESIGN","GET","GOOD","NEW","COMMUNICATE","LOVE","NEW","SUCCESS","LO
        VELY","LOVE","LEARN","PRAY","LOVE","COOL","DO","LOVE","GORGEOUS","HAPPY","GO
        AL","STORY","LOVE","STUNNING","COMMUNICATE","PRAY","GREAT","HAVE","LOVE","PE
        OPLE","MIRACLE","COMMUNICATE","WALK","LOVE","DO","COOL","WALK","PRAY","LIKE"
        ,"LOVELY","NEW","DESIGN","TEACH","LOVE","DO"];

<WordCloudChart showDataTips="true" selectionMode="single">
  ...
  <WordCloudSeries>
   ...
</WordCloudChart>
```

See the CodePen 알메이트 차트 - 단어의 빈도수로 워드클라우드 차트 표현

단어에 지정된 가중치로 워드클라우드 차트 표현

다음은 데이터에 지정된 각 단어의 수치값(가중치)에 의해서 차트에 표현되는 단어의 크기와 색상이 결정되는 워드클라우드 차트를 생성하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.
사용되는 데이터에서 단어의 텍스트와 단어의 가중치에 해당하는
필드명은 <WordCloudSeries> 노드의 textField 속성과 weightField 속성에 각각 지정합니다.

속성명	유효값 (*: 기본값)	설명
textField	텍스트	단어의 텍스트가 저장된 필드명을 지정합니다.
weightField	텍스트	단어의 가중치가 저장된 필드명을 지정합니다.

```
var chartData = [
  {"text":"السلام", "value":13},
  {"text":"Ahoj", "value":12},
  {"text":"こんにち", "value":3},
  ...
];

<WordCloudChart showDataTips="true">
  ...
  <WordCloudSeries textField="text" weightField="value">
    ...
  </WordCloudSeries>
  ...
</WordCloudChart>
```

See the CodePen 알메이트 차트 - 단어에 지정된 가중치로 워드클라우드 차트 표현

주의

> 브라우저와 클라이언트 PC 의 폰트 환경에 의해서 설치가 안되어 있거나 지원하지 않는 폰트라면 올바르게 출력되지 않을 수 있습니다.

동적 워드 클리우드 차트 표현

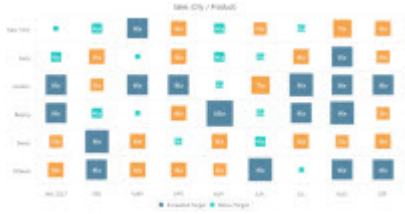
다음은 setTimeout() 함수를 이용해서 3,000ms 마다 한 번씩 차트를 다시 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서 적용되는 데이터는 makeData() 함수를 이용해서 동적으로 생성됩니다.

```
function chartReadyHandler(id) {
  document.getElementById(id).setLayout(layoutStr);
  document.getElementById(id).setData(makeData());
  setTimeout(changeData, 3000);
}

function makeData(){
  var i, n,
  chartData = [],
  data =
        ["PRAY","NEW","DO","DESIGN","CONCENTRATE","GREAT","LEAD","MOMENT","LOVE","DO
        ","NEW","PRAY","GREAT","DESIGN","CREATIVE","LOVE","STUNNING","PRAY","BABY","
        LOVELY","DESIGN","GET","GOOD","NEW","COMMUNICATE","LOVE","NEW","SUCCESS","LO
        VELY","LOVE","LEARN","PRAY","LOVE","COOL","DO","LOVE","GORGEOUS","HAPPY","GO
        AL","STORY","LOVE","STUNNING","COMMUNICATE","PRAY","GREAT","HAVE","LOVE","PE
        OPLE","MIRACLE","COMMUNICATE","WALK","LOVE","DO","COOL","WALK","PRAY","LIKE"
        ,"LOVELY","NEW","DESIGN","TEACH","LOVE","DO"];
  for(i = 0, n = data.length ; i < n ; i += 1){
    chartData.push({
      text : data[i],
      weight : Math.floor(Math.random(10) * 100)
    });
  }
  return chartData;
}

<WordCloudChart showDataTips="true">
  ...
  <WordCloudSeries textField="text" weightField="weight">
    ...
  </WordCloudSeries>
  ...
</WordCloudChart>
```

See the CodePen [알메이트 차트 - 동적 워드클라우드 차트 표현]

4.27 매트릭스 차트

매트릭스 차트는 X, Y 좌표 상에 Z 값을 표현하는 차트입니다. X, Y 축의 레이블에는 일반적으로 카테고리 명이 표시되며 X, Y 좌표상에 표시되는 Z 값은 매트릭스 차트의 유형에 따라서 도형, 이미지, 사각형 박스, 플롯(plot)이 될 수 있습니다. 매트릭스 차트는 <Matrix2DChart> 노드의 series 속성값에 <Matrix2DSeries> 노드를 설정하여 생성할 수 있습니다. 매트릭스 차트의 유형은 <Matrix2DChart> 노드의 type 속성에 지정된 값에 따라서 결정되는데 이는 아래 표와 같습니다.

type 속성값	설명	예제 모습
renderer (default)	X, Y 좌표상에 도형을 표시합니다. 도형의 크기는 Z 값에 의해서 정해집니다.	
image	X, Y 좌표상에 이미지를 표시합니다. 이미지의 크기는 Z 값에 의해서 정해집니다.	
fill	X, Y 좌표상에 사각형 박스를 표시합니다. Z 값은 무시됩니다.	
plot	X, Y 좌표상에 플롯(plot)을 표시합니다. Z 값은 무시됩니다.	

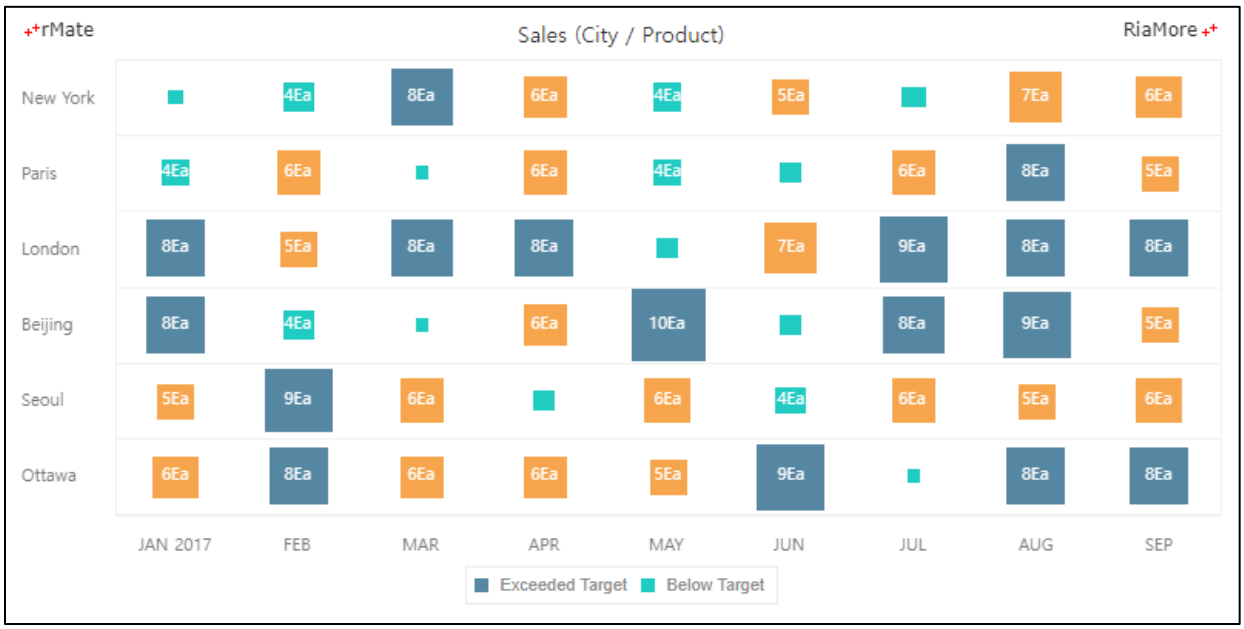
<Matrix2DChart> 노드의 type 속성값이 “renderer” 혹은 “image” 일 경우, Z 값에 따라서 정해지는 도형 혹은 이미지의 크기는 <Matrix2DChart> 노드의 drawType 속성에 지정된 값에 따라서 결정되는데 이는 아래 표와 같습니다.

속성명	유효값 (*: 기본값)	설명
type	renderer(*), image, fill, plot	매트릭스 차트의 유형을 설정합니다.
drawType	radius(*), area	매트릭스 차트의 유형이 “renderer” 혹은 “image” 일 경우, 도형 혹은 이미지의 크기를 설정하는 기준을 정합니다. radius: 반지름 기준 area: 면적 기준

도형 표시 매트릭스 차트

다음은 X, Y 좌표 상에 도형을 표시하는 매트릭스 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. type 속성값이 “renderer” 로 지정되었고 도형의 크기는 도형의 반지름(drawType = “radius”) 기준으로 설정되었습니다.

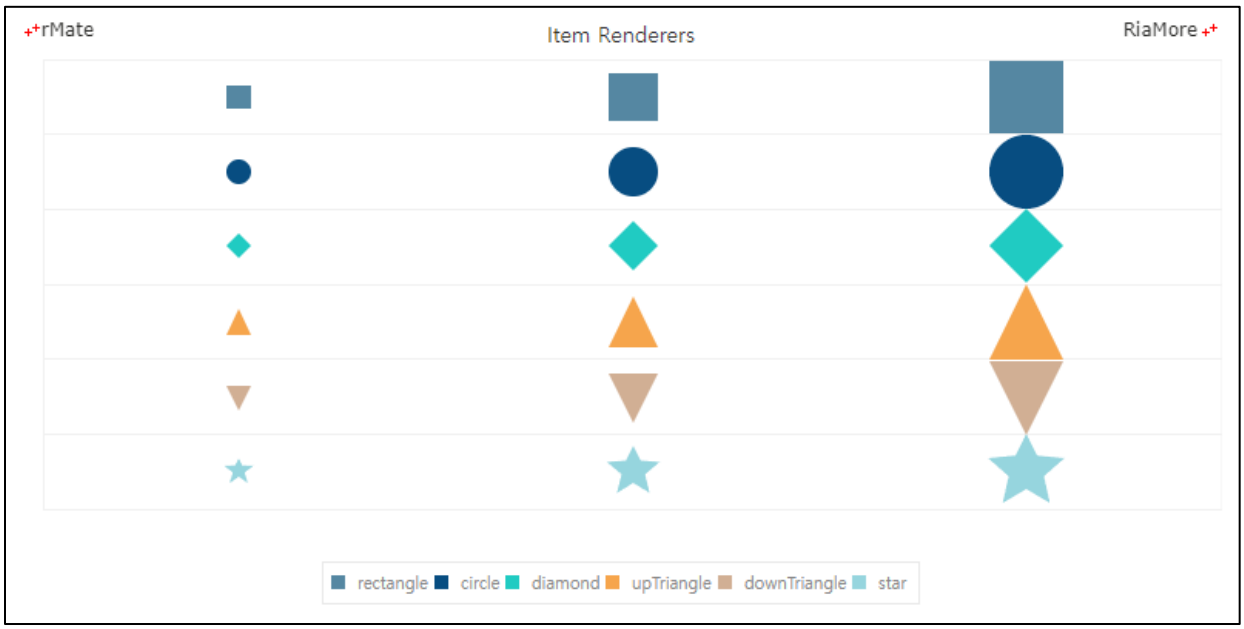
```
<Matrix2DChart showDataTips="true" type="renderer" drawType="radius"
       dataTipJsFunction="tipFunc">
  ...
  <Matrix2DSeries xField="Data1_x" yField="Data1_y" zField="Data1"
       renderer="rectangle" ... />
  <Matrix2DSeries xField="Data2_x" yField="Data2_y" zField="Data2"
       renderer="rectangle" ... />
  <Matrix2DSeries xField="Data3_x" yField="Data3_y" zField="Data3"
       renderer="rectangle" ... />
  ...
</Matrix2DChart>
```



See the CodePen 알메이트 차트 - 도형 표시 매트릭스 차트

매트릭스 차트에 표시할 도형의 종류는 <Matrix2DSeries> 노드의 renderer 속성에 설정합니다.
다음은 renderer 속성에 지정가능한 값과 이 도형들을 모두 표현한 매트릭스 차트의 예제입니다.

renderer 속성값	표시되는 도형
rectangle	
circle	
diamond	
upTriangle	
downTriangle	
star	

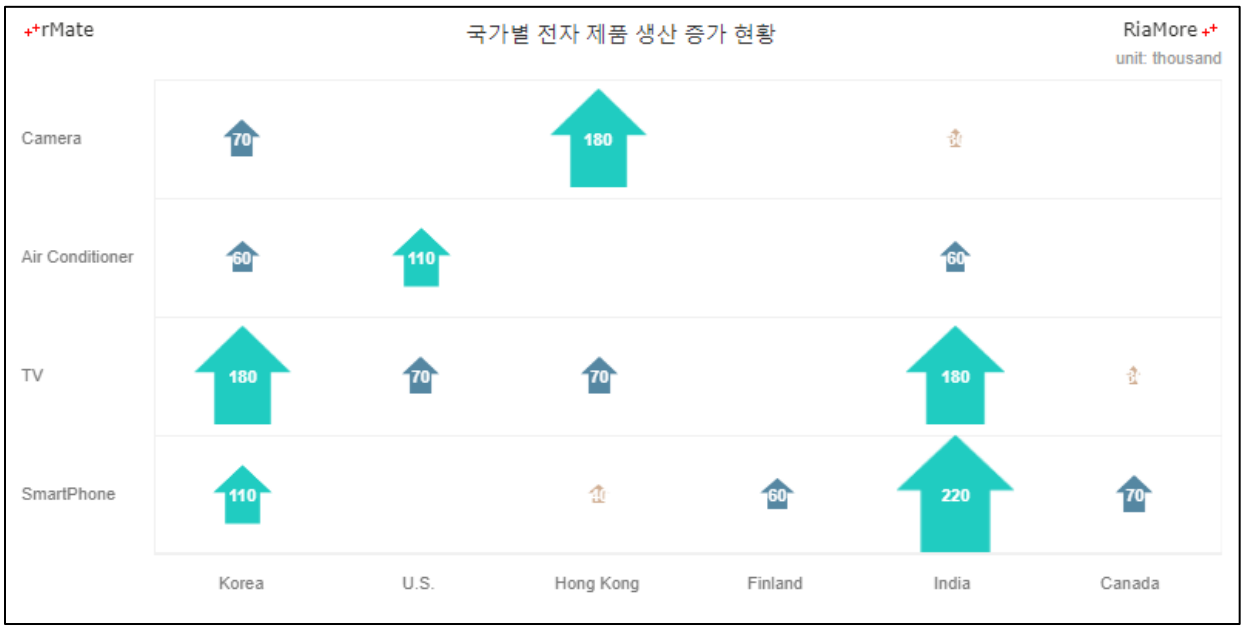


See the CodePen 알메이트 차트 - 매트릭스 차트에 표시 가능한 도형

이미지 표시 매트릭스 차트

다음은 X, Y 좌표 상에 이미지를 표시하는 매트릭스 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. type 속성값이 “image” 로 지정되었고 이미지의 크기는 반지름(drawType = “radius”) 기준으로 설정되었습니다.

```
<Matrix2DChart showDataTips="true" type="image" drawType="radius">
  ...
  <Matrix2DSeries xField="Data1_x" fontSize="11" yField="Data1_y" zField="Data1"
       imageSource="../rMateChartH5/Assets/Images/arrow_01.png"... />
  <Matrix2DSeries xField="Data2_x" fontSize="11" yField="Data2_y" zField="Data2"
       imageSource="../rMateChartH5/Assets/Images/arrow_02.png" ... />
  <Matrix2DSeries xField="Data3_x" fontSize="11" yField="Data3_y" zField="Data3"
       imageSource="../rMateChartH5/Assets/Images/arrow_03.png" ... />
  ...
</Matrix2DChart>
```

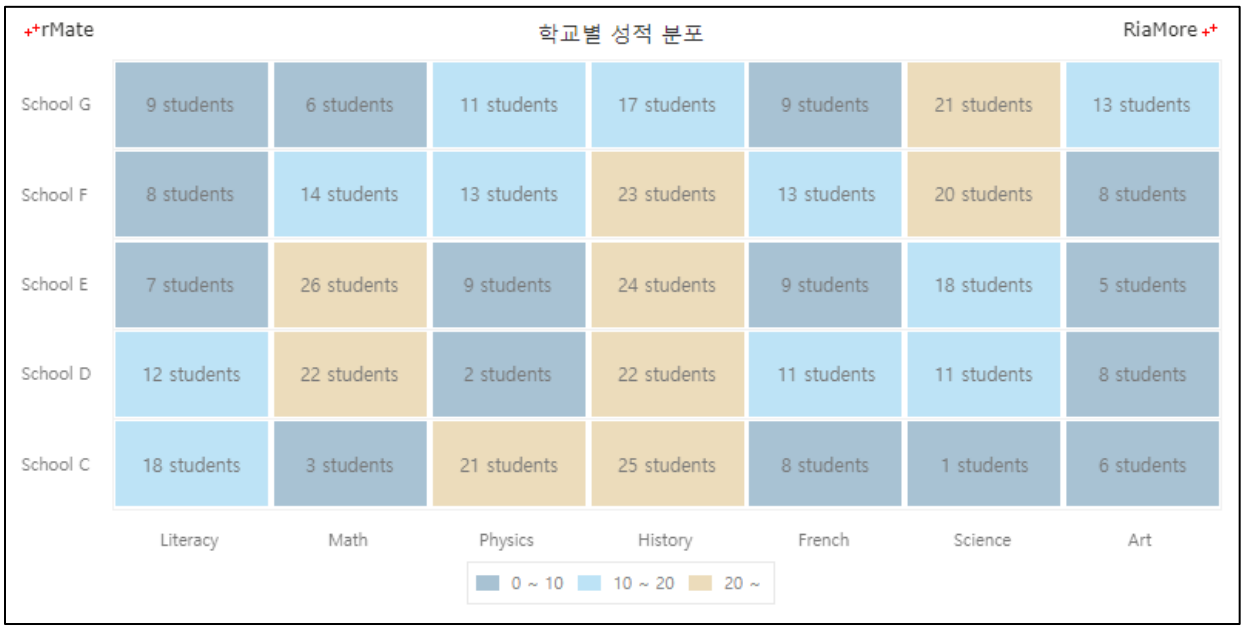


See the CodePen 알메이트 차트 - 이미지 표시 매트릭스 차트

사각형 박스 표시 매트릭스 차트

다음은 X, Y 좌표 상에 사각형 박스를 표시하는 매트릭스 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. type 속성값이 “fill” 로 지정되었고 사각형 박스의 색상은 사용자 정의 함수(fillJsFunc())에 의해서 적용되는데, 함수명은 <Matrix2DSeries> 노드의 fillJsFunction 속성값에 지정되었습니다.

```
<Matrix2DChart showDataTips="true" type="fill" dataTipJsFunction="tipFunc" >
  ...
  <Matrix2DSeries xField="Data_x" yField="Data_y" zField="Data"
        fillJsFunction="fillJsFunc" ... />
  ...
</Matrix2DChart>

<Style>.font{fontFamily:"Malgun
        Gothic";fontSize:11;}.seriesFont{color:#777;fontFamily:"Malgun
        Gothic";fontSize:12;}</Style>
...
function fillJsFunc (seriesId, index, param, values) {
  if (values[2] < 10)
    return "#a8c2d3";
  else if (values[2] >= 10 && values[2] <20)
    return "#bde3f6";
  else (values[2] >= 20)
    return "#ecdcbb";
}
```



See the CodePen 알메이트 차트 - 사각형 박스 표시 매트릭스 차트

플롯(Plot) 표시 매트릭스 차트

다음은 X, Y 좌표 상에 플롯(plot)을 표시하는 매트릭스 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. type 속성값이 “plot” 로 지정되었고 플롯의 색상은 사용자 정의 함수(fillJsFunc())에 의해서 적용되는데, 함수명은 <Matrix2DSeries> 노드의 fillJsFunction 속성값에 지정되었습니다.

```
<Matrix2DChart showDataTips="true" type="plot" dataTipJsFunction="tipFunc">
  ...
  <Matrix2DSeries xField="Data_x" yField="Data_y" zField="Data"
       fillJsFunction="fillJsFunc" gridRadius="5" ... >
    <stroke>
      <Stroke color="#999999"/>
    </stroke>
  </Matrix2DSeries>
  ...
</Matrix2DChart>

function fillJsFunc (seriesId, index, param, values) {
  if (values[2] < 7000)
    return "#5587a2";
  else if (values[2] >= 7000 && values[2] < 10000)
    return "#20ccc0";
  else (values[2] >= 10000)
    return "#f6a54c";
}
```

See the CodePen 알메이트 차트 - 플롯(plot) 표시 매트릭스 차트

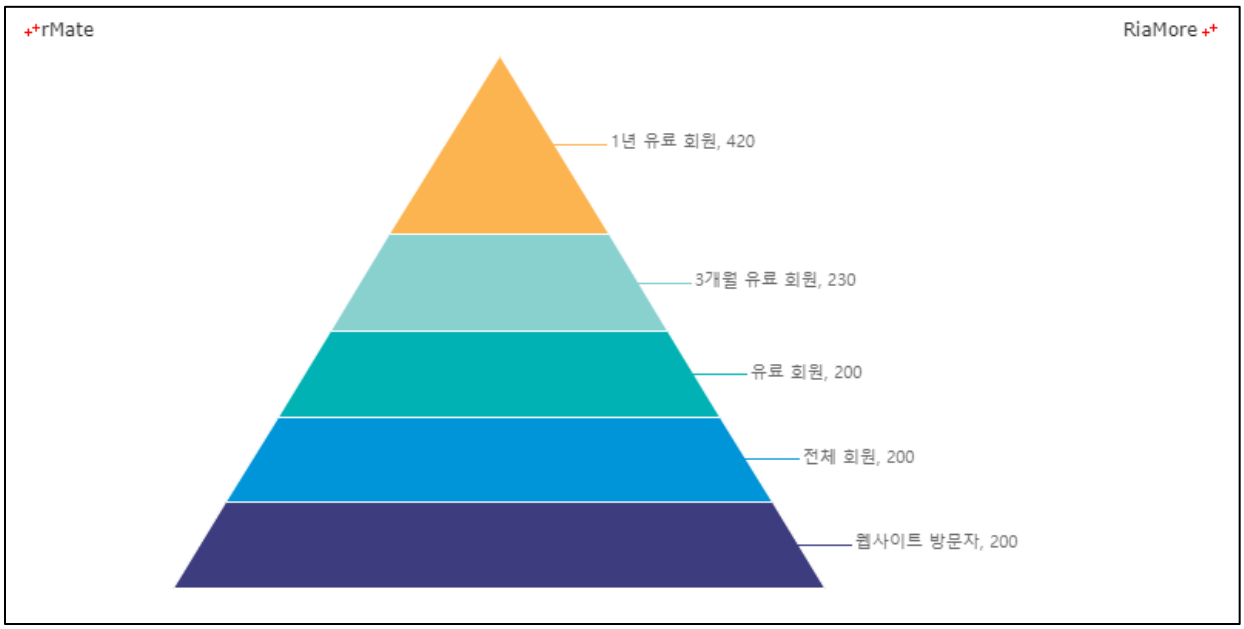
4.28 피라미드 차트

피라미드 차트는 삼각형을 선으로 나눈 각 영역에 관련된 주제의 데이터를 표현하는 차트입니다. 전체적으로 삼각형이기 때문에 각 영역의 넓이는 서로 다릅니다. 각 영역의 넓이가 다르기 때문에 어떤 주제와 관련된 각 영역의 데이터의 계층관계를 표현하는데 유용합니다. 예를 들면, 넓은 영역일수록 일반적인 내용을 좁은 영역일 수록 구체적인 내용을 표현할 수 있습니다. 하지만 넓이의 크기가 데이터 값의 크기를 의미하지는 않습니다. 데이터 값의 크기는 영역의 길이의 크기와 비례합니다. 각 영역이 삼각형에 표시되는 순서는 데이터에 정의되는 순서와 같습니다. 제일 먼저 정의되는 데이터가 삼각형의 제일 상단 영역에 표시됩니다. 파라미드 차트는 <Pyramid2DChart> 노드의 series 속성값에 <Pyramid2DSeries> 노드를 설정하여 생성할 수 있습니다. 각 영역에 표시되는 레이블과 영역의 길이는 <Pyramid2DSeries> 노드의 다음 속성에 설정합니다.

속성명	유효값 (*: 기본값)	설명
nameField	텍스트	영역의 레이블이 저장된 필드명을 지정합니다.
weightField	텍스트	영역의 값이 저장된 필드명을 지정합니다.

다음은 일반적인 피라미드 차트를 생성하는 코드와 데이터, 그리고 이를 적용해서 출력한 차트의 예제입니다.

```
var chartData = [{
  "label" : "1 year paid membership",
  "data" : 420
},{
  "label" : "3 months paid membership",
  "data" : 230
...
}];

<Pyramid2DChart showDataTips="true">
  ...
  <Pyramid2DSeries weightField="data" nameField="label" labelPosition="callout"
       leftBottomRatio="0.7" rightBottomRatio="0.7" color="#666" fontSize="11"
       labelYOffset="-2">
    <fills>
      <SolidColor color="#FBB44F"/>
      <SolidColor color="#89D1CF"/>
      <SolidColor color="#00B2B3"/>
      <SolidColor color="#0095D8"/>
      <SolidColor color="#3C3C7F"/>
    </fills>
  </Pyramid2DSeries>
  ...
</Pyramid2DChart>
```



See the CodePen 알메이트 차트 - 피라미드 차트

피라미드 차트에서 표현되는 삼각형의 모양은 <Pyramid2DSeries> 노드의 다음 속성들을 이용하여 조정할 수 있습니다.

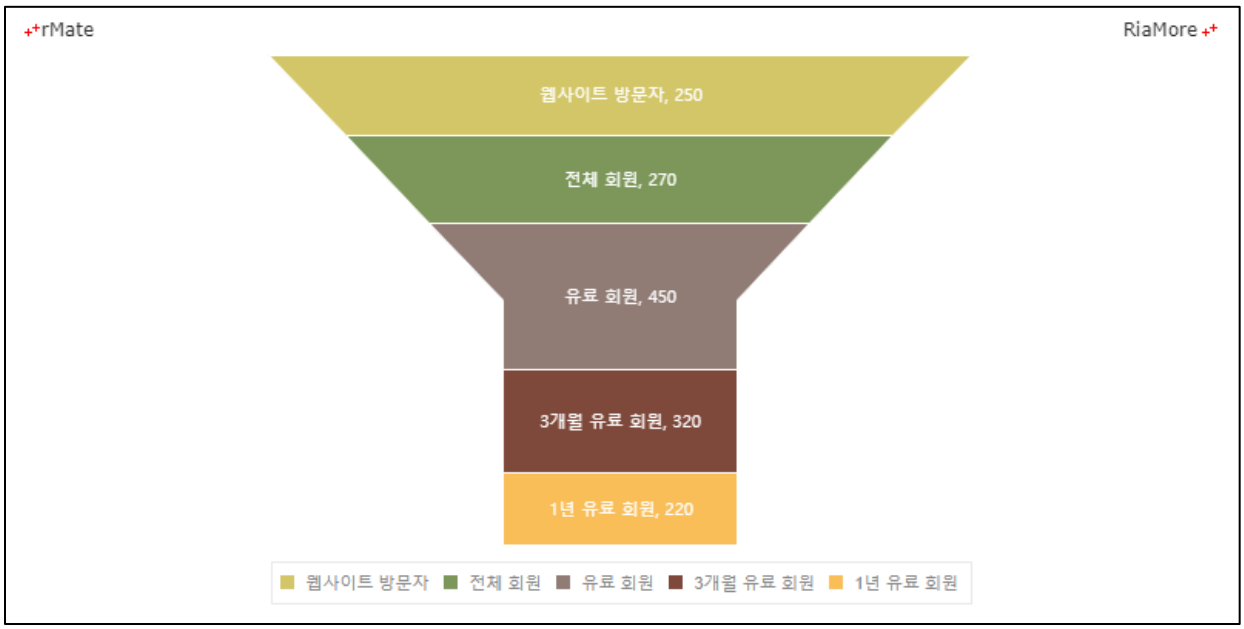
속성명	유효값 (*: 기본값)	설명
leftBottomRatio	0 과 1(*) 사이의 숫자	삼각형의 좌측 하단 모서리의 각도 비율을 지정합니다.
rightBottomRatio	0 과 1(*) 사이의 숫자	삼각형의 우측 하단 모서리의 각도 비율을 지정합니다.
leftTopRatio	0(*) 과 1 사이의 숫자	삼각형의 좌측 상단 모서리의 각도 비율을 지정합니다.
rightTopRatio	0(*) 과 1 사이의 숫자	삼각형의 우측 상단 모서리의 각도 비율을 지정합니다.
bottomRatio	-1(*): 이 속성을 사용하지 않음 0 과 1 사이의 숫자	삼각형의 좌우측 하단 모서리의 각도 비율을 지정합니다.

topRatio	-1(*): 이 속성을 사용하지 않음 0 과 1 사이의 숫자	삼각형의 좌우측 상단 모서리의 각도 비율을 지정합니다.
horizontalRatio	-1(*): 이 속성을 사용하지 않음 0 과 1 사이의 숫자	좌우측 수평 비율을 지정합니다.
verticalRatio	-1(*): 이 속성을 사용하지 않음 0 과 1 사이의 숫자	좌우측 수직 비율을 지정합니다.

퓨널(Funnel) 피라미드 차트

위에서 설명한 피라미드 차트의 삼각형 모양을 조정하는 속성들 중 topRatio, bottomRatio, horizontalRatio 속성을 이용해서 깔대기(퓨널, Funnel) 모양의 피라미드 차트를 생성할 수 있습니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Pyramid2DChart showDataTips="true">
  ...
  <Pyramid2DSeries weightField="data" nameField="label" topRatio="0.6"
       horizontalRatio="0.2" bottomRatio="0.2" color="#fff" fontSize="11">
      ...
</Pyramid2DChart>
```

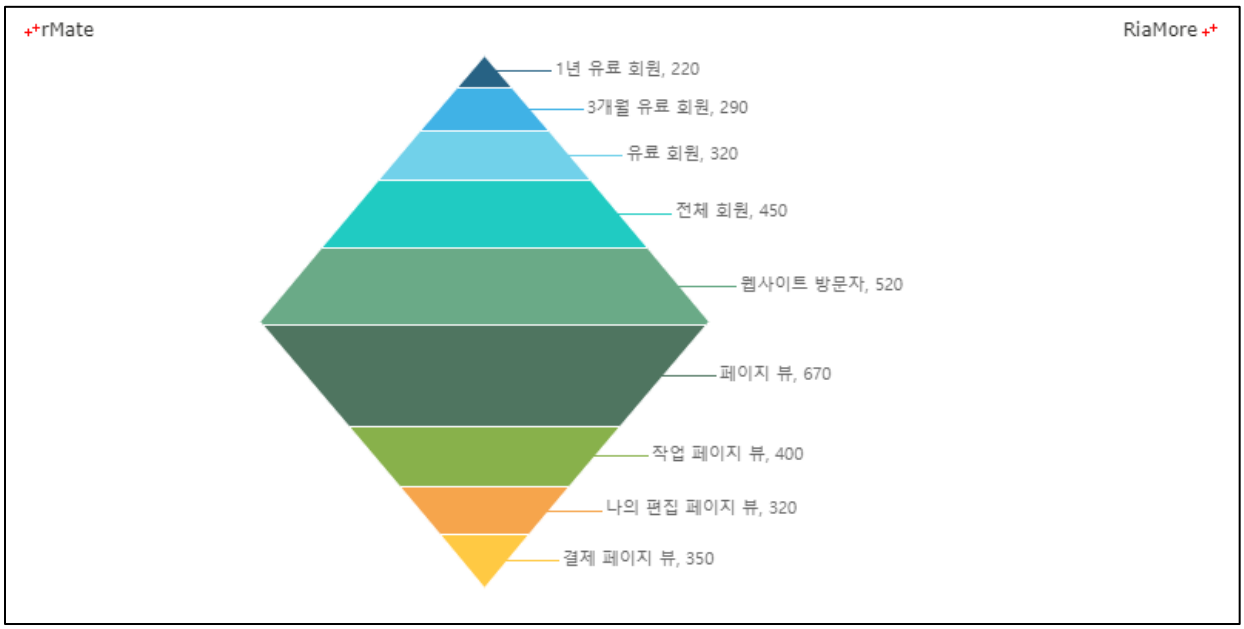


See the CodePen 알메이트 차트 - 퓨널(Funnel) 피라미드 차트

다이아몬드 피라미드 차트

위에서 설명한 피라미드 차트의 삼각형 모양을 조정하는 속성들 중 topRatio, bottomRatio, horizontalRatio 속성을 이용해서 다이아몬드 모양의 피라미드 차트를 생성할 수 있습니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Pyramid2DChart showDataTips="true" selectionMode="single">
  ...
  <Pyramid2DSeries labelPosition="callout" weightField="data" nameField="label"
       topRatio="0" bottomRatio="0" horizontalRatio="0.5" fontSize="11"
       color="#666666" labelYOffset="-2">
    ...
</Pyramid2DChart>
```



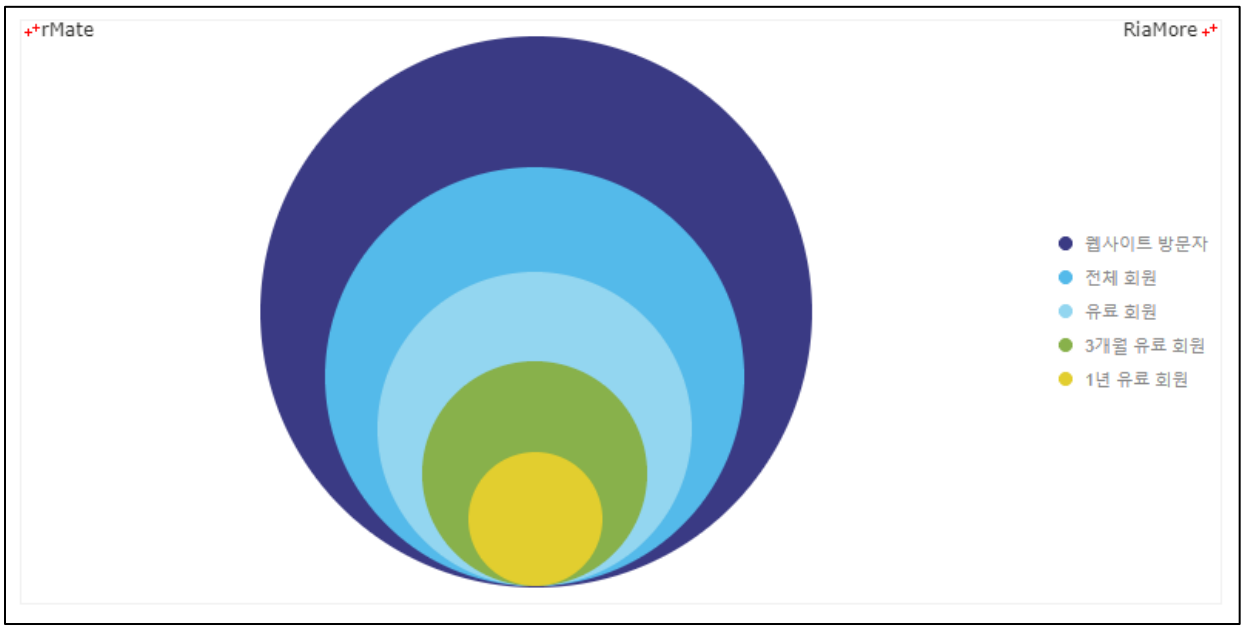
See the CodePen 알메이트 차트 - 다이아몬드 피라미드 차트

오버레이 버블 차트

오버레이 버블 차트는 원으로 데이터의 계층 관계를 표현하는 차트입니다. 큰 원부터 시작해서 원의 크기에 따라 순차적으로 상위 원에 포함되는 모양으로 표현됩니다. 따라서 피라미드 차트와 마찬가지로 어떤 주제와 관련된 각 영역의 데이터의 계층관계를 표현하는데 유용한 차트 유형입니다.
오버레이 버블 차트는 <OverlayBubbleChart> 노드의 series
속성값에 <OverlayBubbleSeries> 노드를 설정하여 생성할 수 있습니다.

다음은 오버레이 버블 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<OverlayBubbleChart showDataTips="true" paddingTop="10" paddingBottom="10">
  ...
  <OverlayBubbleSeries field="data" nameField="label" labelPosition="inside">
    <fills>
      <SolidColor color="#3A3A84"/>
      ...
    </fills>
    ...
</OverlayBubbleChart>
```

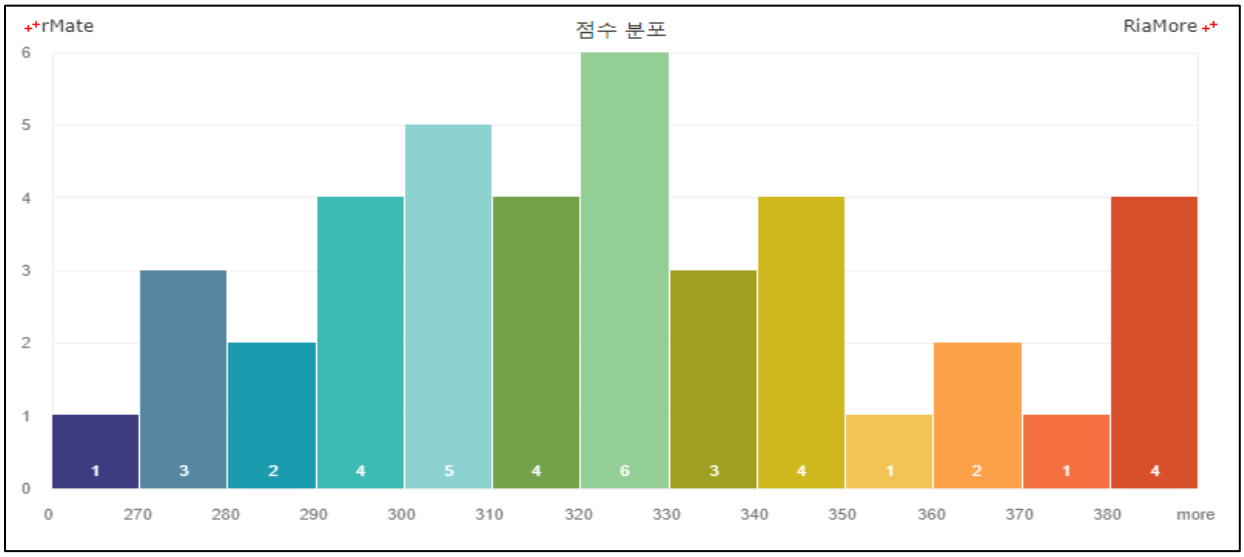


See the CodePen 알메이트 차트 - 오버레이 버블 차트

4.29 히스토그램 차트

히스토그램 차트는 컬럼 차트와 아주 유사하지만 컬럼 차트와 다른 점은 수평축 상의 연속적인 구간(bin)들에 대한 측정값이 수직 막대로 표현된다는 것입니다. 이와는 다르게 컬럼 차트에서는 비연속적인 카테고리 데이터에 대한 측정값이 세로 막대로 표현됩니다. 그리고 일반적으로 컬럼 차트에서는 이웃한 수직 막대 사이에 간격을 두지만 히스토그램 차트는 수직 막대들 사이의 간격을 두지않고 연속적으로 표현합니다. 히스토그램 차트는 <Histogram2DChart> 노드의 series 속성값에 <Histogram2DSeries> 노드를 설정하여 생성할 수 있습니다. 그리고 수평축의 생성을 위해서 <horizontalAxis> 속성에 <HistogramCategoryAxis> 노드를 설정해야 하며, 축의 스타일링은 <HistogramAxis2DRenderer> 노드를 <horizontalAxisRenderers> 속성에 정의하여 할 수 있습니다. 다음은 특정 학급의 성적 분포를 히스토그램 차트로 표현하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Histogram2DChart binRange="[270,280,290,300,310,320,330,340,350,360,370,380]">
  <horizontalAxis>
    <HistogramCategoryAxis id="hAxis" categoryField="histogramXField"/>
  </horizontalAxis>
  <horizontalAxisRenderers>
    <HistogramAxis2DRenderer axis="{hAxis}"/>
  </horizontalAxisRenderers>
  <series>
    <Histogram2DSeries labelPosition="inside" yField="grade" displayName="Number of
        Persons" itemRenderer="BoxItemRenderer" labelAlign="bottom"
        insideLabelYOffset="-6" color="#ffffff">
      <fills>
        <SolidColor color="#3d3c80"/>
        ...
      </fills>
    </Histogram2DSeries>
  </series>
</Histogram2DChart>
```



See the CodePen 알메이트 차트 - 히스토그램 차트

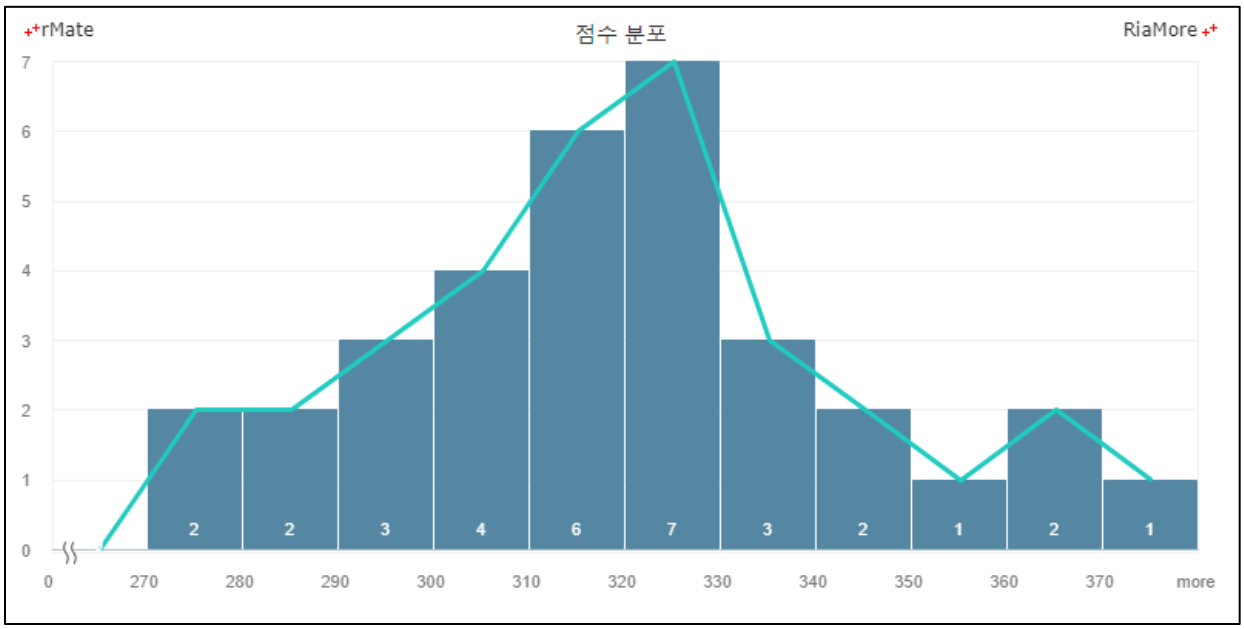
히스토그램 차트의 수평축에 표시되는 구간(bin)은 <Histogram2DChart> 노드의 binRange 속성에 지정합니다. 차트에 적용되는 데이터들은 각 구간에 자동으로 합산되며, 구간을 초과하는 데이터는 additionalBinRange 속성(기본값: “more”)에 정의된 명칭의 구간에 합산됩니다.

속성명	유효값 (*: 기본값)	설명
binRange	숫자 배열 기본값: [0,20,40,60,80,100]	구간을 지정합니다.
additionalBinRange	텍스트 기본값: more	binRange 속성에 설정된 값을 초과하는 구간에 대한 명칭을 지정합니다.

히스토그램 차트에 분포선 표시

히스토그램 차트에 분포선은 <Histogram2DChart> 노드에 <Line2DSeries> 노드를 정의하여 표현합니다. 이 때 <Line2DSeries> 노드의 yField 속성에는 <Histogram2DSeries> 노드에 정의된 yField 속성값과 동일한 값을 지정합니다. 다음은 분포선을 표시하기 위해서 yField 속성을 점수 필드명(grade)으로 지정한 코드와 이를 적용적용해서 출력한 차트의 예제입니다.

```
<Histogram2DChart binRange="[270,280,290,300,310,320,330,340,350,360,370]">
  ...
  <Histogram2DSeries labelPosition="inside" yField="grade" displayName="Number of
       Persons" itemRenderer="BoxItemRenderer" labelAlign="bottom"
       insideLabelYOffset="-6" color="#ffffff">
    ...
  </Histogram2DSeries>
  <Line2DSeries yField="grade">
    ...
  </Line2DSeries>
  ...
</Histogram2DChart>
```

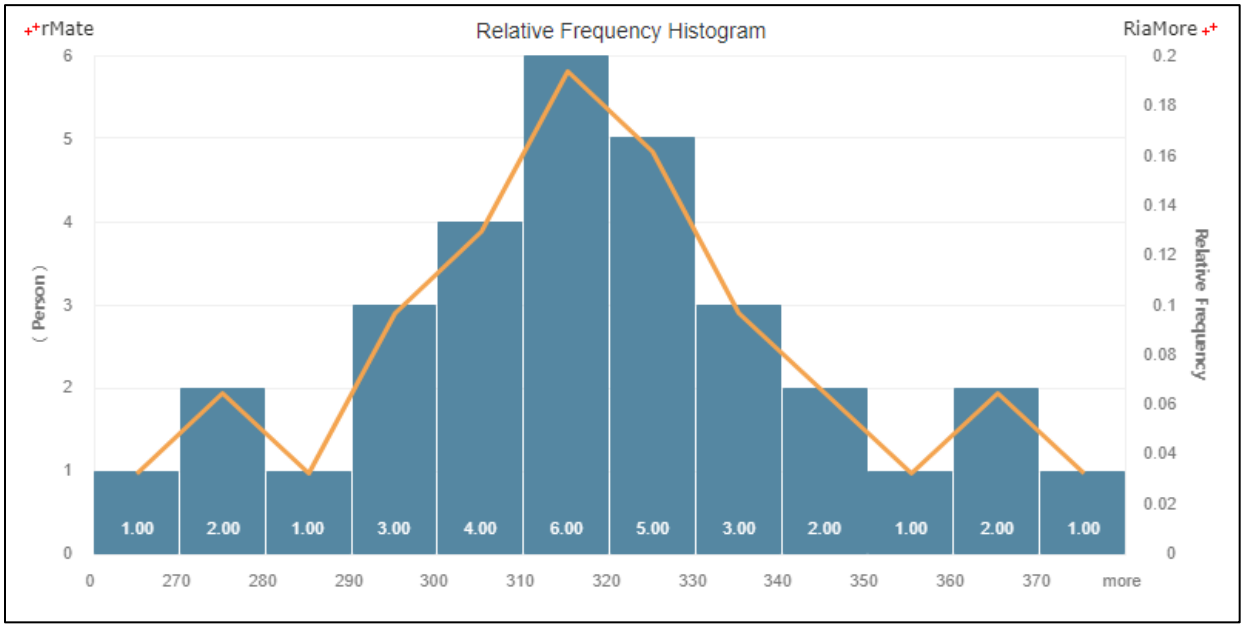


See the CodePen 알메이트 차트 - 히스토그램 차트에 분포선 표시

히스토그램 차트에 상대분포 표시

히스토그램 차트의 상대분포는 특정 구간(bin)에 합산된 데이터 개수를 전체 데이터 개수로 나눈 값입니다. 상대분포 선은 <Line2DSeries> 노드의 yField 속성에 “relativeFrequency” 를 지정하고 상대분포 값을 위한 세로 축은 차트의 오른쪽에 표시합니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 <Line2DSeries> 노드의 <verticalAxis> 속성에 <LinearAxis> 노드를 정의하여 차트의 오른쪽에 상대분포 값을 위한 축을 표시하였습니다.

```
<Histogram2DChart binRange="[270,280,290,300,310,320,330,340,350,360,370]">
  ...
    <LinearAxis id="vAxis1" title="( Person )"/>
  ...
    <Histogram2DSeries formatter="{nft}" labelPosition="inside" yField="grade"
        displayName="Number of Persons" itemRenderer="BoxItemRenderer"
        labelAlign="bottom" insideLabelYOffset="-10" color="#ffffff">
      ...
    <Line2DSeries yField="relativeFrequency">
      ...
       <LinearAxis id="vAxis2" title="Relative Frequency"/>
      ...
  <verticalAxisRenderers>
    <Axis2DRenderer axis="{vAxis1}" placement="left" ...>
      ...
    </Axis2DRenderer>
    <Axis2DRenderer axis="{vAxis2}" placement="right" ...>
      ...
    </Axis2DRenderer>
  </verticalAxisRenderers>
</Histogram2DChart>
```

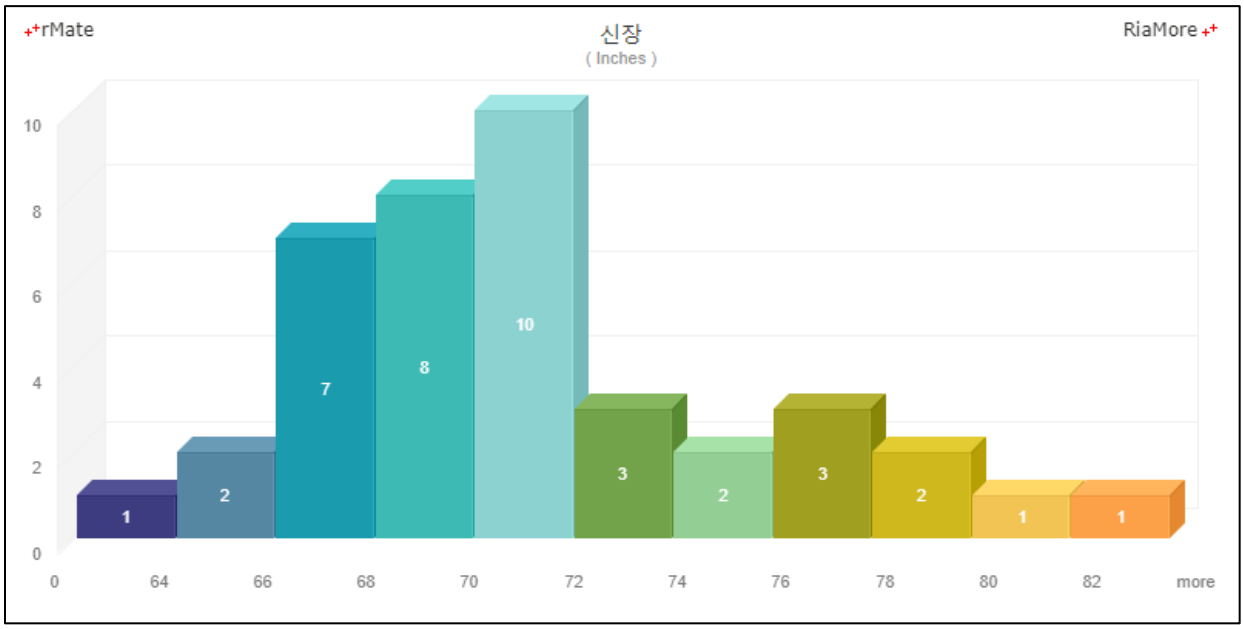


See the CodePen 알메이트 차트 - 히스토그램 차트에 상대분포 표시

3D 히스토그램 차트

3D 히스토그램 차트는 <Histogram3DChart> 노드의 series 속성값에 <Histogram3DSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Histogram3DChart binRange="[64,66,68,70,72,74,76,78,80,82]">
  ...
  <Histogram3DSeries labelPosition="inside" yField="height" displayName="Inches"
       color="#ffffff">
    <fills>
      <SolidColor color="#3d3c80"/>
      ...
    </fills>
  </Histogram3DSeries>
  ...
</Histogram3DChart>
```

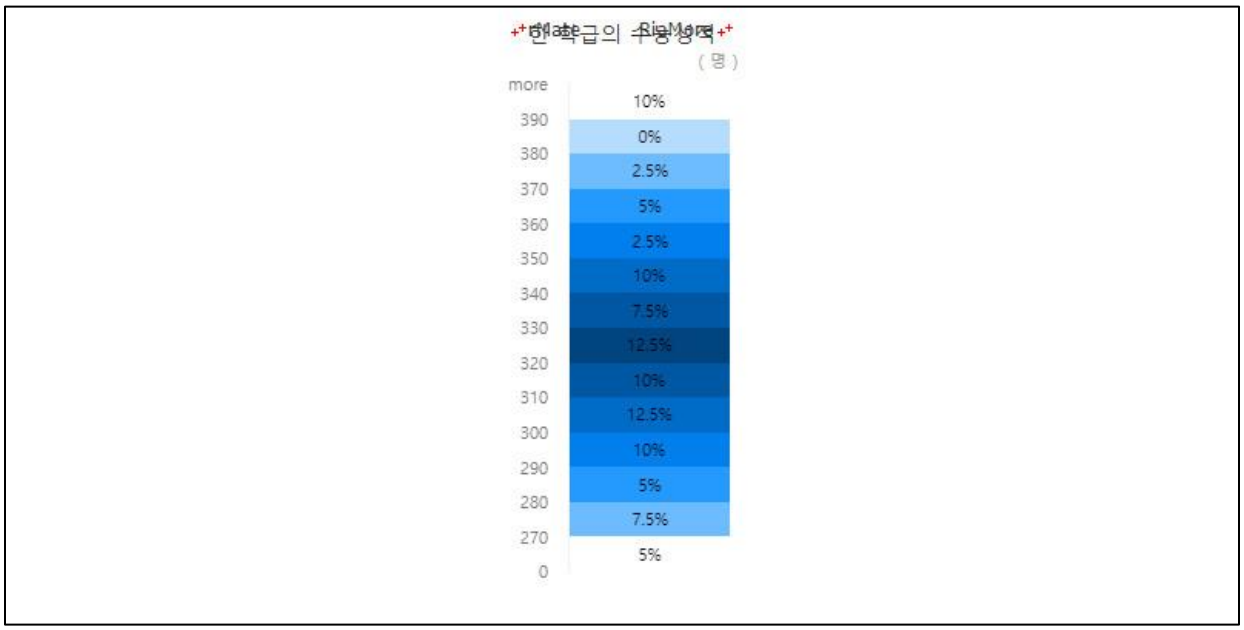


See the CodePen 알메이트 차트 - 3D 히스토그램 차트

세로 히스토그램 차트

세로 히스토그램 차트는 2D 히스토그램 차트의 축 XY 를 바꿔서 제작한 차트입니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<VHistogram2DChart
       binRange="[270,280,290,300,310,320,330,340,350,360,370,380,390]">
  ...
  <VHistogram2DSeries labelPosition="inside" xField="grade" sortOnColor="true">
    <showDataEffect>
      <SeriesInterpolate/>
    </showDataEffect>
  </VHistogram2DSeries>
  ...
```



See the CodePen 알메이트 차트 - 세로 히스토그램 차트

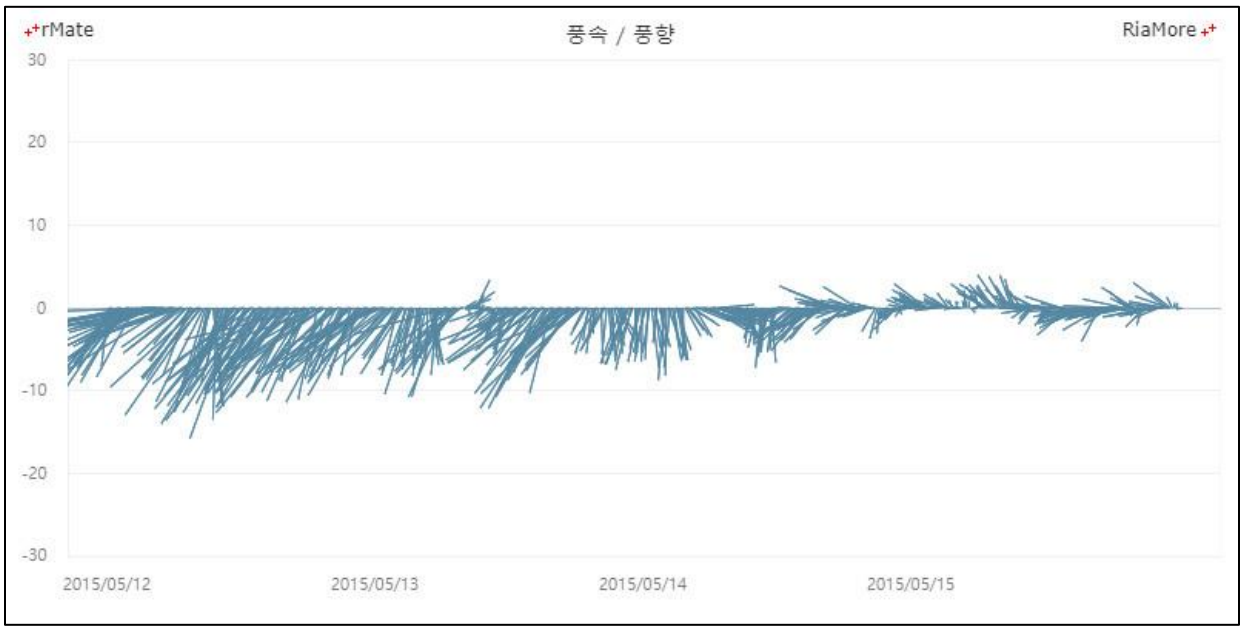
4.30 벡터 차트

벡터 차트는 데이터의 속도 값과 각도 값을 선으로 표현하는 차트입니다. 데이터의 속도 값은 선의 길이로 표현하고 데이터의 각도 값은 선이 표시되는 각도 방향으로 표현됩니다. 각도 값, “0” 는 12 시 방향이고 시계 방향으로 각도 값이 증가하게 됩니다. 벡터 차트는 <Vector2DChart> 노드의 series 속성값에 <Vector2DSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 <Vector2DSeries> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
degreeField	텍스트	각도 값이 저장된 데이터 필드명을 지정합니다.
velocityField	텍스트	속도 값이 저장된 데이터 필드명을 지정합니다.
meterField	텍스트	레이어(층) 값이 저장된 데이터 필드명을 지정합니다.
referenceAngle	숫자 기본값: 0	기본 각도 값을 지정합니다. degreeField 속성 값과 더해진 값이 차트에 표현됩니다.
showArrow	true, false(*)	선에 화살을 표시할지 여부를 설정합니다.

다음은 날짜별 바람의 방향과 속도를 표현하는 벡터 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Vector2DChart showDataTips="true">
  <horizontalAxis>
    <DateTimeAxis id="hAxis" dataUnits="minutes" labelUnits="days"
       formatter="{dateFmt}" displayName="Date" alignLabelsToUnits="true"
       displayLocalTime="true" padding="220"/>
  </horizontalAxis>
  <verticalAxis>
    <LinearAxis maximum="30" minimum="-30"/>
  </verticalAxis>
  <series>
    <Vector2DSeries xField="date" velocityField="speed" degreeField="degree">
      ...
    </Vector2DSeries>
  </series>
</Vector2DChart>
```

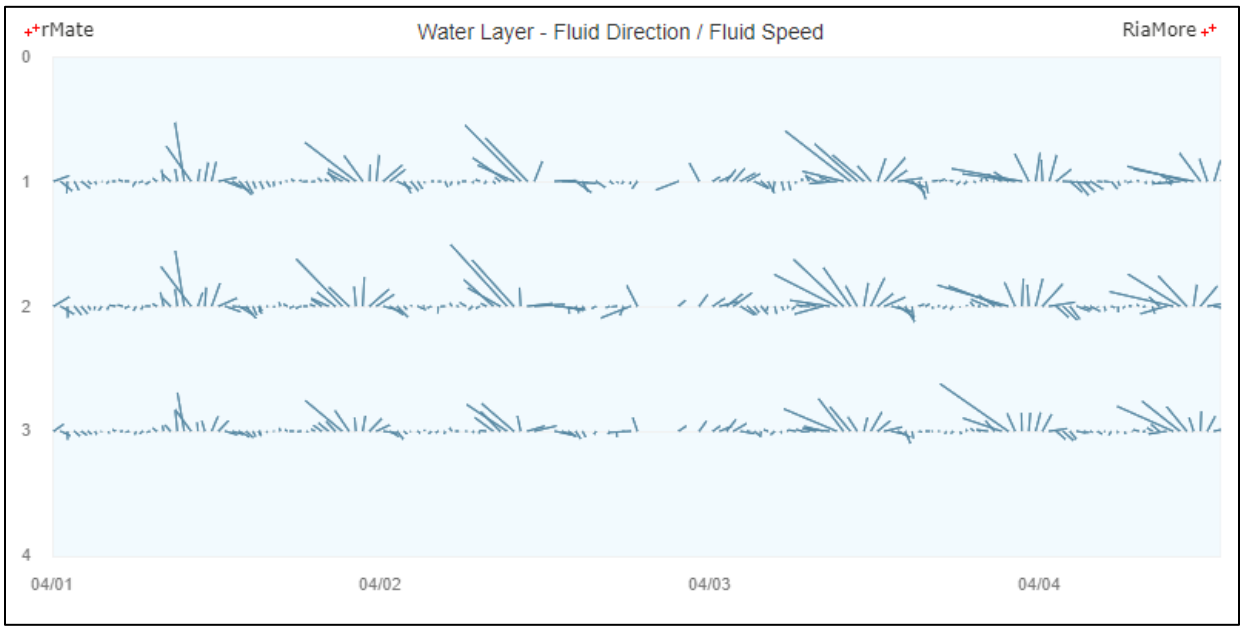


See the CodePen 알메이트 차트 - 벡터 차트

다중 레이어(시리즈) 벡터 차트

다음은 meterField 속성을 이용하여 다중 레이어(시리즈) 벡터 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Vector2DChart id="chart1" showDataTips="true" mouseSensitivity="5">
  ...
    <Vector2DSeries xField="date" velocityField="speed1" degreeField="degree1"
       meterField="meter1">
      ...
    </Vector2DSeries>
    <Vector2DSeries xField="date" velocityField="speed2" degreeField="degree2"
       meterField="meter2">
      ...
    </Vector2DSeries>
    <Vector2DSeries xField="date" velocityField="speed3" degreeField="degree3"
       meterField="meter3">
      ...
    </Vector2DSeries>
  ...
</Vector2DChart>
```



See the CodePen 알메이트 차트 - 다중 레이어(시리즈) 벡터 차트

위 예제에서는 3 개의 <Vector2DSeries> 노드가 설정되었고, 각각 해당 레이어 정보를 가진 데이터 필드명을 meterField 속성에 지정하였습니다.

4.31 에러바 차트

에러바 차트는 데이터 값들의 표준 오차 값을 차트에 함께 표시합니다. 에러바를 차트에 표시하기 위해서는 데이터 시리즈 (<Column2DSeries>, <Bar2DSeries> 등) 노드의 showErrorBar 속성을 “true” 로 설정해야 합니다. 그리고 데이터 값에 해당하는 필드의 형식은 아래와 같이 값들의 배열이어야 합니다.

```
var chartData = [{
  "Month" : "Jan",
  "temperature" :
        [19,20.74,24.39,17.28,17.39,7.78,11.11,16.11,11,15,12.78,12,10,17.22,17,21.3
        6,22.74,15,24.14,16,21.53,28.11,30.67,29.39,35.35,24,20,24,21,20]
},{
  "Month" : "Fen",
  "temperature" :
        [26.72,28.28,33.28,31.71,32.22,30.61,32,35,35,31.72,31.11,30,30,31.11,32.89,
        33.78,32,28,33,28,26,24.39,25.61,26,28,31,31,26,25,26]
},{
  ...
}];
```

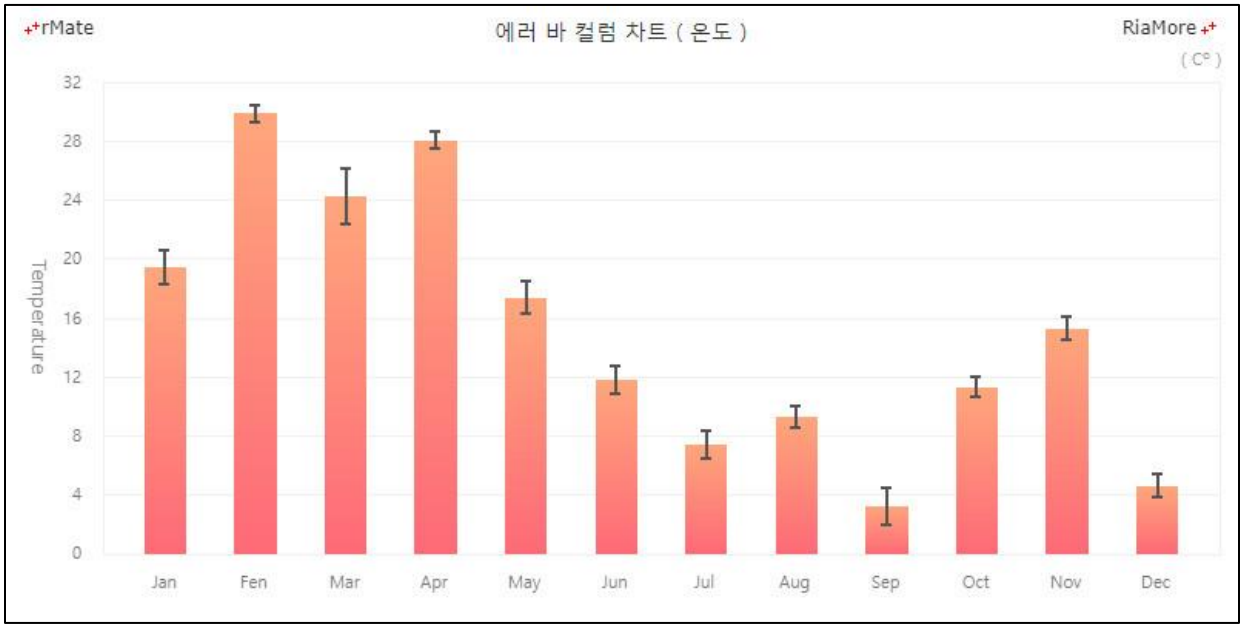
다음은 차트에 에러바를 표현하는데 필요한 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
showErrorBar	true, false(*)	차트에 에러바를 표시할 지 여부를 설정합니다.
errorBarDirection	both(*), plus, minus	에러바 표시 방법을 지정합니다. plus: 최대 표준 오차만 표시합니다. minus: 최소 표준 오차만 표시합니다. both: 최대, 최소 표준 오차 모두를 표시합니다.
errorBarFixedValue	숫자	오차 범위에 고정값을 지정합니다.
errorBarLength	숫자	차트에 표시되는 오차 범위의 가로 크기를 지정합니다.
errorBarPercentValue	숫자	오차 범위의 백분율 값을 지정합니다.
errorBarStroke	<Stroke>	에러바의 선의 스타일을 지정합니다.

컬럼 차트에 에러바 표시

다음은 컬럼 차트에 에러바를 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DChart showDataTips="true" dataTipFormatter="{cft}">
  ...
    <Column2DSeries yField="temperature" displayName="Temperature"
       showErrorBar="true" errorBarDirection="both">
      <errorBarStroke>
        <Stroke color="#555555" weight="2"/>
      </errorBarStroke>
      ...
    </Column2DSeries>
  ...
</Column2DChart>
```

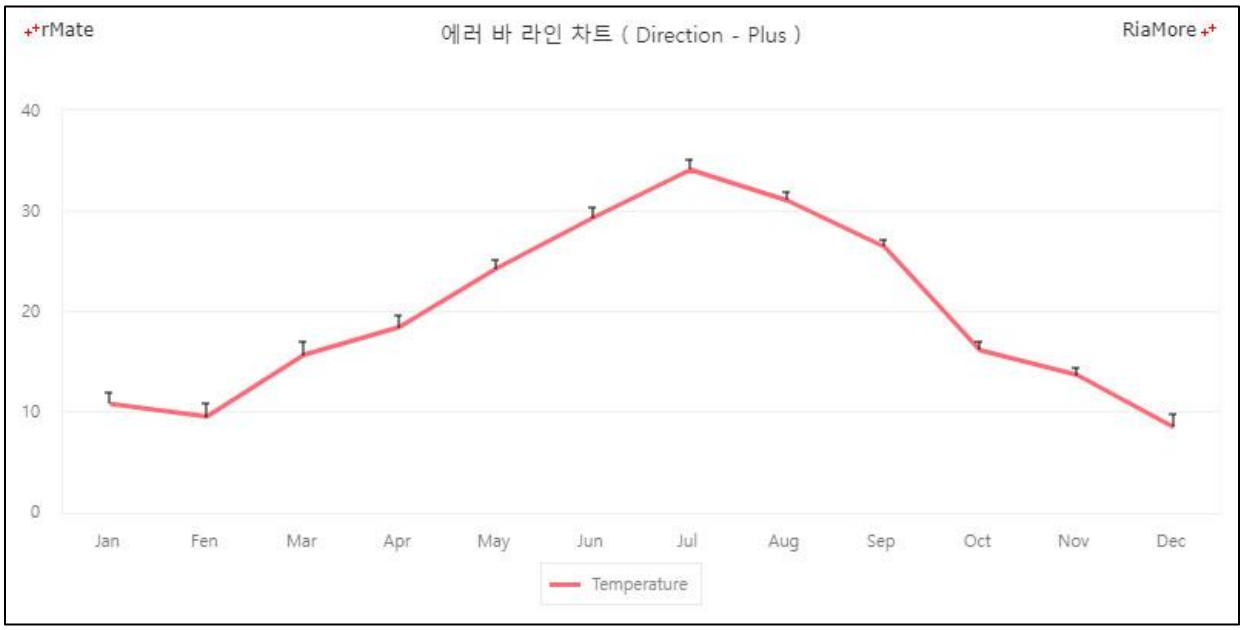


See the CodePen 알메이트 차트 - 컬럼 차트에 에러바 표시

라인 차트에 에러바 표시

다음은 라인 차트에 에러바를 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DChart showDataTips="true">
  ...
    <Line2DSeries yField="temperature" displayName="Temperature"
       showErrorBar="true" errorBarDirection="plus" errorBarLength="5">
  ...
</Line2DChart>
```



See the CodePen 알메이트 차트 - 라인 차트에 에러바 표시

위 예제에서는 에러바의 가로 크기를 5 (errorBarLength = “5”)로 지정하였고, 최대 표준 오차만 표시 (errorBarDirection = “plus”) 하도록 설정되었습니다.

영역 차트에 에러바 표시

다음은 영역 차트에 에러바를 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Area2DChart showDataTips="true" dataTipDisplayMode="axis">
  ...
    <Area2DSeries yField="Vancouver" displayName="Vancouver" showErrorBar="true"
       errorBarDirection="minus">
  ...
</Area2DChart>
```

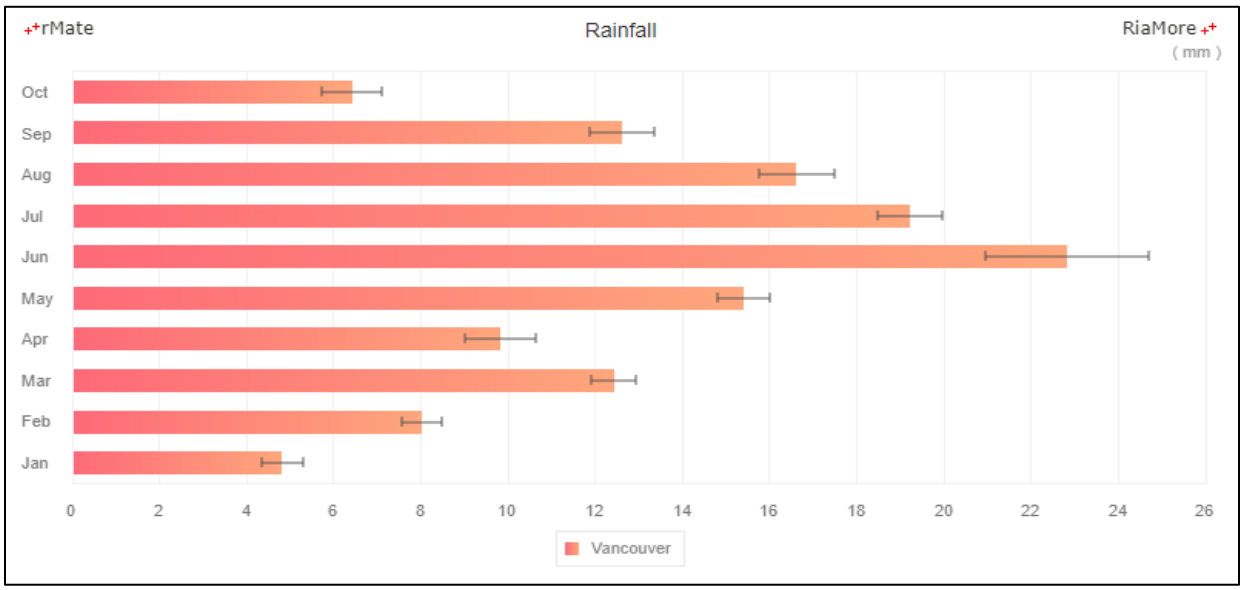
See the CodePen 알메이트 차트 - 영역 차트에 에러바 표시

위 예제에서는 최소 표준 오차만 표시 (errorBarDirection = “minus”) 하도록 설정되었습니다.

바 차트에 에러바 표시

다음은 바 차트에 에러바를 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Bar2DChart showDataTips="true">
  ...
    <Bar2DSeries xField="Vancouver" displayName="Vancouver" showErrorBar="true">
  ...
</Bar2DChart>
```

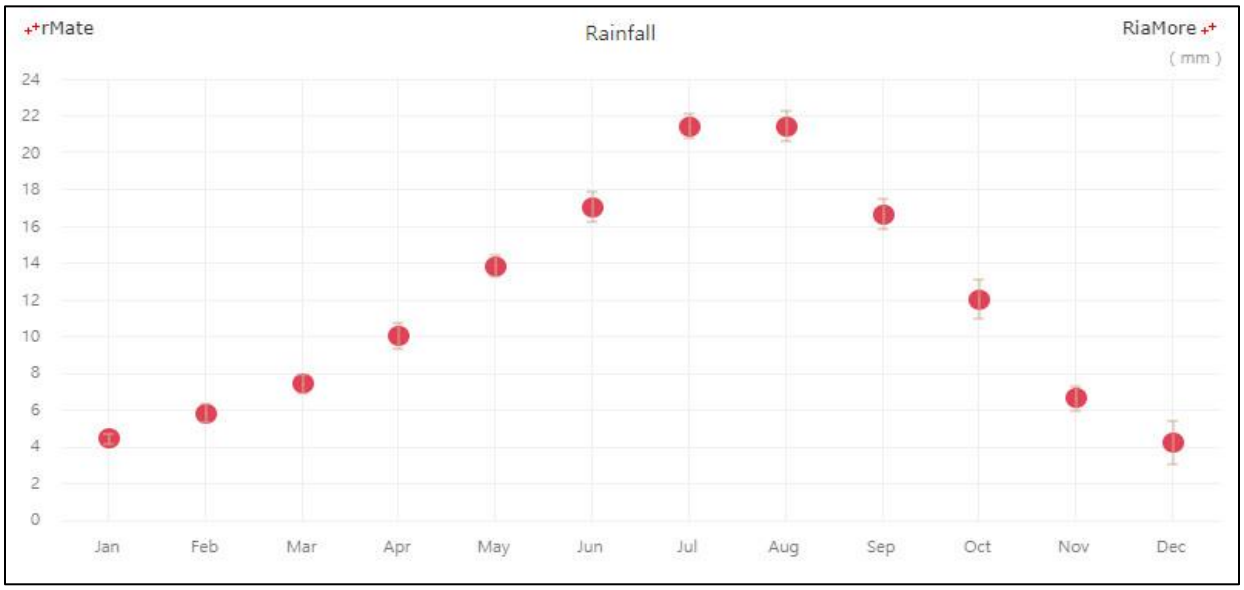


See the CodePen 알메이트 차트 - 바 차트에 에러바 표시

플롯 차트에 에러바 표시

다음은 플롯 차트에 에러바를 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

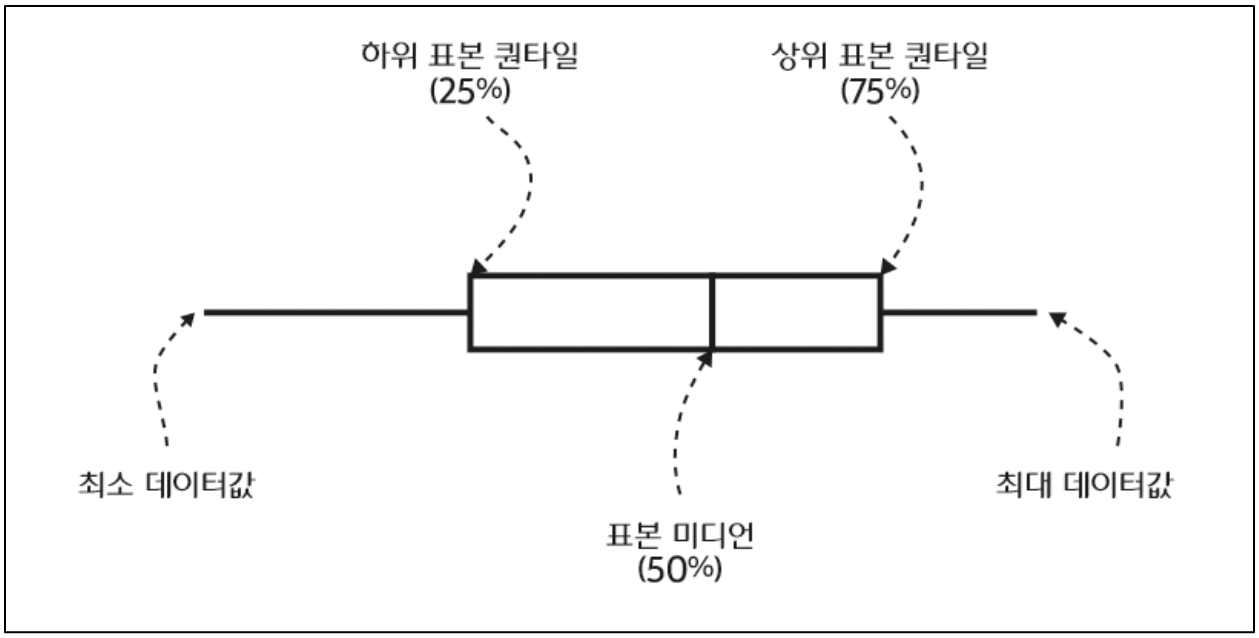
```
<Plot2DChart showDataTips="true">
  ...
    <Plot2DSeries yField="Vancouver" displayName="Vancouver" showErrorBar="true"
       radius="7" itemRenderer="CircleItemRenderer">
  ...
</Plot2DChart>
```



See the CodePen 알메이트 차트 - 플롯 차트에 에러바 표시

4.32 박스 플롯 차트

박스 플롯 차트는 데이터의 분포를 표현하는 데 유용한 차트 유형이며, 백분위 수 (percentile)의 1 사분위(하위 25% 이하, first quartile), 2 사분위(하위 25% ~ 50%, second quartile), 3 사분위(50% ~ 상위 25%, third quartile), 4 사분위(상위 25% 이내, fourth quartile)의 값을 박스와 선으로 표현합니다. 아래 그림은 박스 플롯을 가로 모양으로 표현한 것입니다.



에러바 차트와 마찬가지로 박스 플롯 차트의 값에 해당하는 필드의 형식은 배열이어야 합니다.

```
var chartData = [{
  "tier" : "Bronze",
  "population" : [2500,4500,3000,3500,4000]
},{
  "tier" : "Silver",
  "population" : [2600,4200,1300,3200,4200]
},{
  ...
}];
```

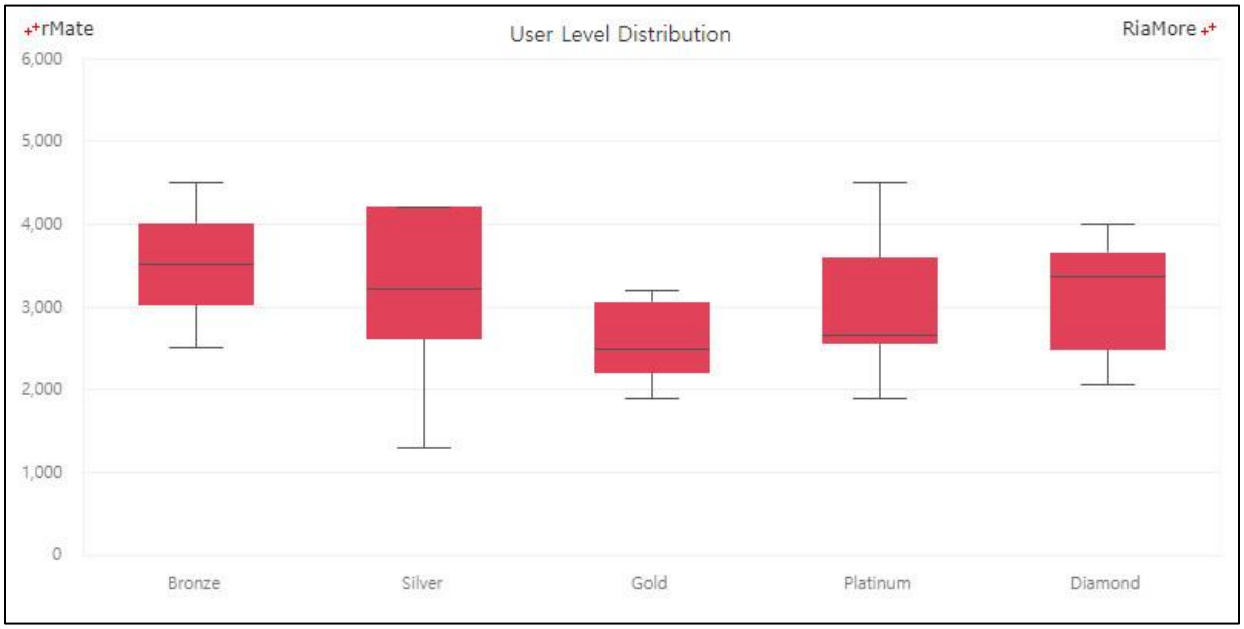
박스 플롯 차트는 <BoxPlotChart> 노드의 series 속성값에 <BoxPlotSeries> 노드를 설정하여 생성할 수 있습니다. 다음은 <BoxPlotSeries> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
boxStroke	<Stroke>	박스 선의 스타일을 지정합니다.

halfWidthOffset	숫자	다중 데이터 시리즈일 경우, 이웃하는 박스 사이의 공간을 픽셀값으로 지정합니다.
maxColumnWidth	숫자	박스의 최대 넓이를 픽셀값으로 지정합니다.
medianStroke	<Stroke>	미디언 값(50%)을 표현하는선의 스타일을 지정합니다.
medianStroke	<Stroke>	선의 스타일을 지정합니다.
whiskerWidthRatio	숫자 기본값: 0.5	수염(whisker)의 가로 크기를 박스의 가로 크기에 대한 비율로 지정합니다.

다음은 박스 플롯 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<BoxPlotChart showDataTips="true" dataTipFormatter="{nft}" columnWidthRatio="0.52">
  ...
    <BoxPlotSeries yField="population" displayName="Number of Users">
  ...
</BoxPlotChart>
```

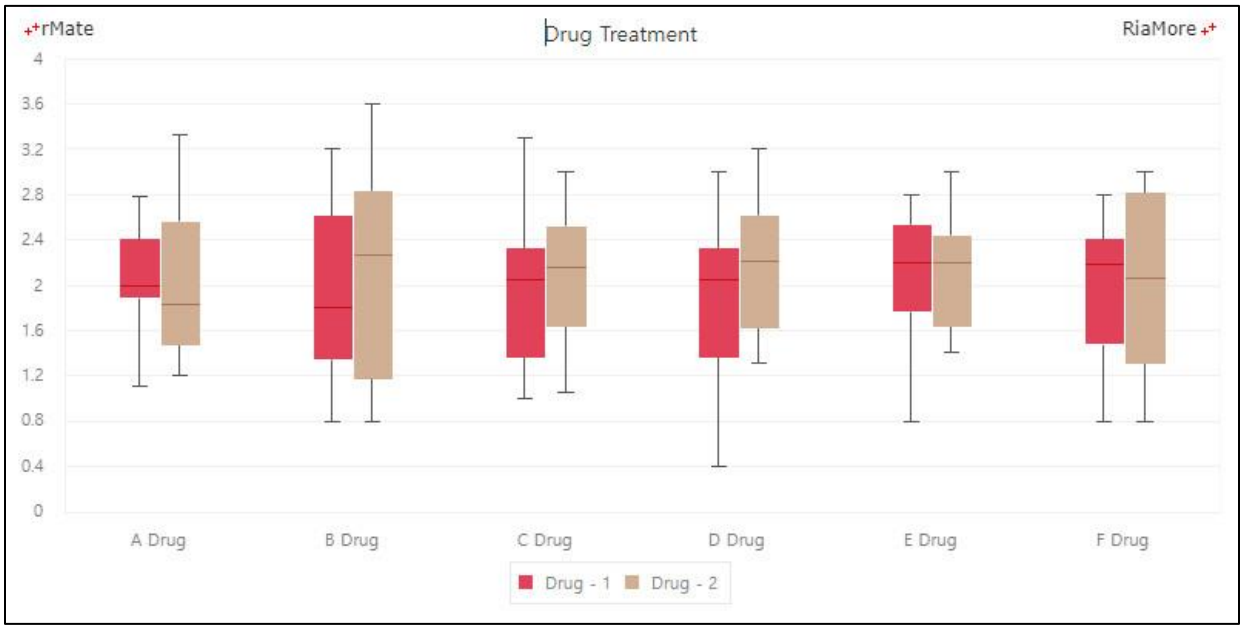


See the CodePen 알메이트 차트 - 박스 플롯 차트

다중 시리즈 박스 플롯 차트

다음은 두 개의 박스 플롯 데이터 시리즈를 표현하는 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<BoxPlotChart showDataTips="true" dataTipFormatter="{nft}" columnWidthRatio="0.42">
  ...
    <BoxPlotSeries yField="Drug1" displayName="Drug - 1">
      <medianStroke>
        <Stroke color="#c7111e" weight="1"/>
      </medianStroke>
      <stroke>
        <Stroke color="#555555"/>
      </stroke>
      <boxStroke>
        <Stroke color="#e14159" weight="1"/>
      </boxStroke>
    </BoxPlotSeries>
    <BoxPlotSeries yField="Drug2" displayName="Drug - 2">
      ...
    </BoxPlotSeries>
  ...
</BoxPlotChart>
```

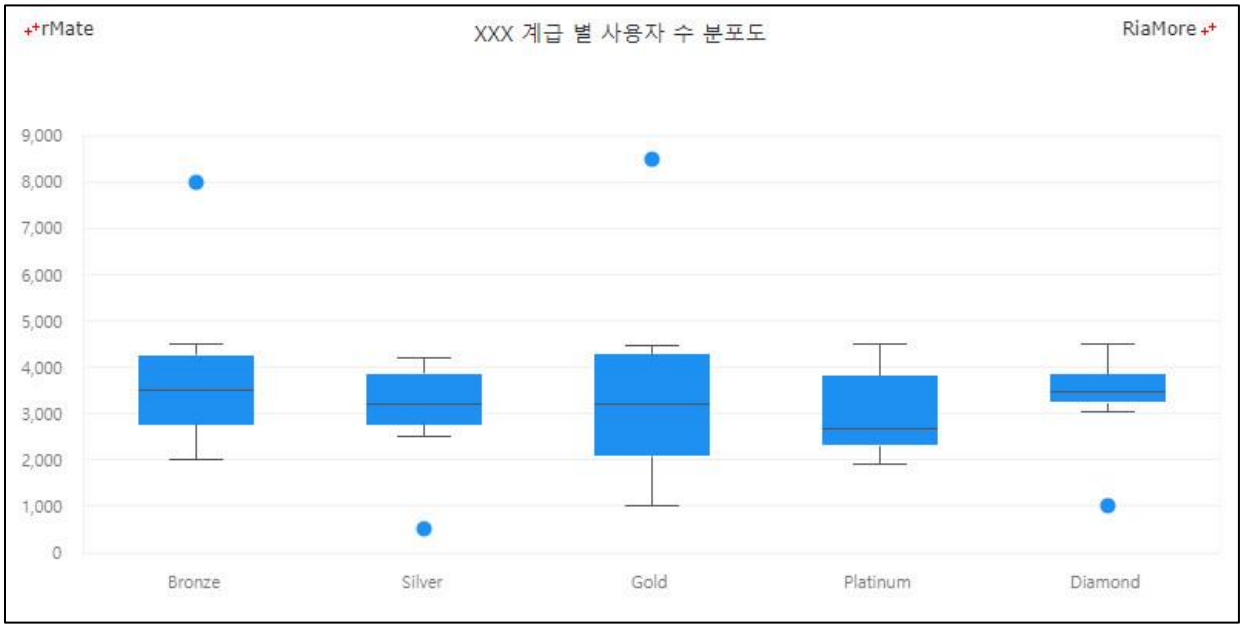


See the CodePen 알메이트 차트 - 다중 시리즈 박스 플롯 차트

박스 플롯 차트 이상치 표시

다음은 박스플롯에 이상치를 표시를 적용해서 출력한 차트의 예제입니다. 이상치를 표시할 경우 이상치를 추가로 고려하여 차트의 모양이 크게 변경됩니다.

```
<BoxPlotChart>
  ...
    <BoxPlotSeries showOutliers="true">
    </BoxPlotSeries>
  ...
</BoxPlotChart>
```



See the CodePen 알메이트 차트 - 박스 플롯 차트 이상치

4.33 슬라이드 차트

슬라이드 차트는 이벤트 차트와 마찬가지로 하나의 독립적인 차트 유형은 아닙니다. 여러 개의 차트를 하나의 차트로 표현하며, 슬라이드 효과을 위해서 앞으로 가기와 뒤로 가기 버튼이 표시됩니다. 슬라이드 차트의 생성은 다음과 같은 과정을 통해서 이루어집니다.

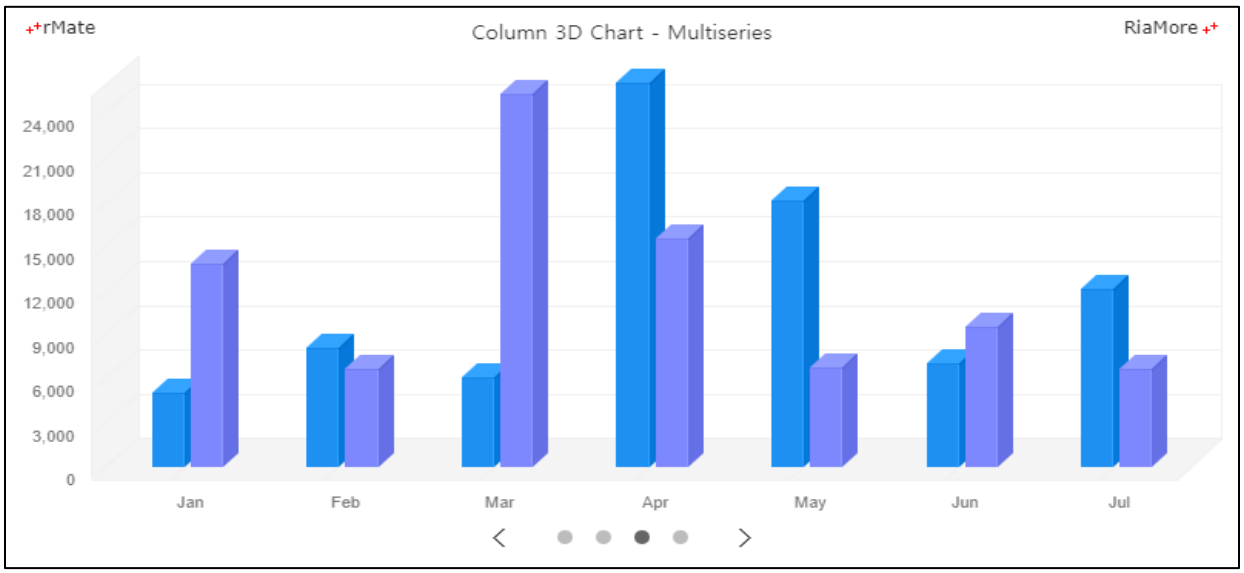
1. 슬라이드로 표현할 차트들을 정하고 각각의 레이아웃과 데이터 셋을 설정합니다. 슬라이드 차트에는 모든 차트 유형이 지원됩니다.
2. 설정된 레이아웃들과 데이터 셋들을 각각 배열로 정의합니다.
3. 레이아웃 배열과 데이터 셋 배열을 setSlideLayoutSet(), setSlideDataSet() 함수로 호출합니다.

주의

> 슬라이드 차트를 생성할 때는 일반적으로 하나의 차트를 생성할 때 호출하는 setLayout(), setData() 함수를 호출하지 않고, setSlideLayoutSet(), setSlideDataSet() 함수를 호출합니다.

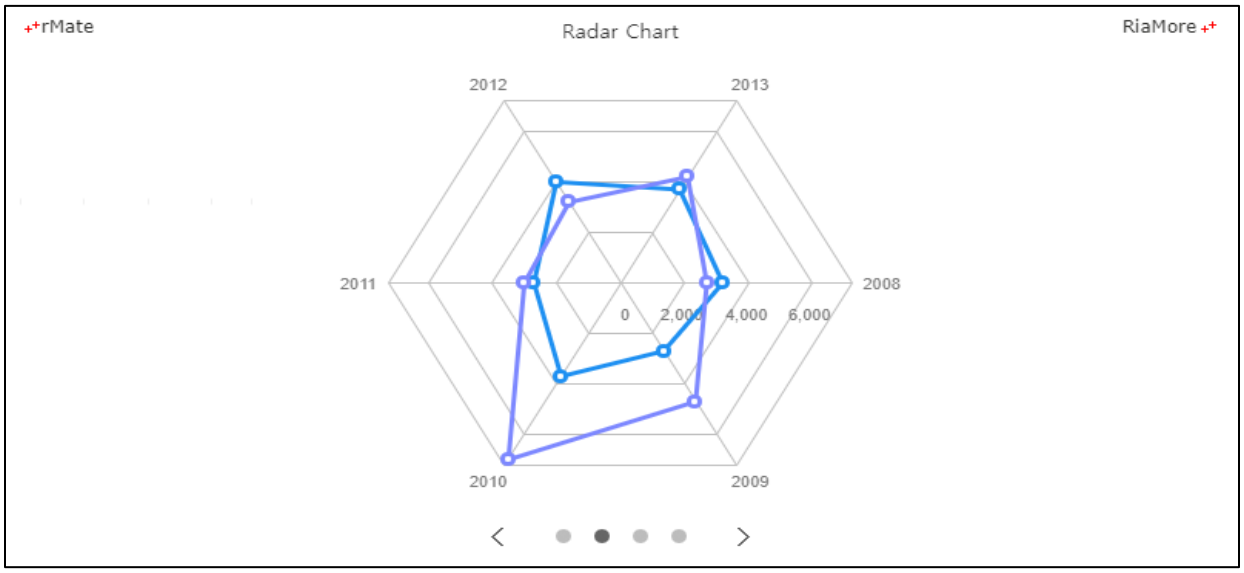
다음은 컬럼 차트, 라인 차트, 3D 컬럼 차트, 방사형 차트를 하나의 슬라이드 차트로 생성하는 코드와 이를 이를 적용해서 출력한 차트의 예제입니다.

```
function chartReadyHandler(id) {
  var layout1 = getCartesianLayout("Column2D","Column Chart",["Profit"]);
  var layout2 = getCartesianLayout("Line2D","Line Chart",["Profit"]);
  var layout3 = getCartesianLayout("Column3D","Column 3D Chart -
       Multiseries",["Profit","Cost"]);
  layoutSet = [layout1, layout2, layout3, radarLayout];
  dataSet = [chartData, chartData2, chartData2, chartData3];
  document.getElementById(id).setSlideLayoutSet(layoutSet);
  document.getElementById(id).setSlideDataSet(dataSet);
}

var chartData = [
...
var chartData2 = [
...
var chartData3 =
...;

function getCartesianLayout(type, title, dataField) {
  var layout="<rMateChart borderStyle='none'>"
  +...;
  layout +="</series>"
  +...;
  return layout;
}
var radarLayout =
  "<rMateChart borderStyle='none'>"
  +...;
```



See the CodePen 알메이트 차트 - 슬라이드 차트

위 예제에서는 getCartesianLayout() 함수를 이용하여 컬럼 차트, 라인 차트, 3D 컬럼 차트 생성을 위한 레이아웃을 작성하고, 방사형 차트 생성을 위한 레이아웃은 radarLayout 변수에 정의합니다. 이렇게 작성된 레이아웃들은 chartReadyHandler() 함수에서 layoutSet 변수에 배열로 정의됩니다. 3 개의 데이터셋이 각각 chartData, chartData2, chartData3 변수에 정의되었고, chartData2 변수는 2 개의 차트(라인 차트, 3D 컬럼 차트)에서 사용됩니다. 데이터셋 변수들은 chartReadyHandler() 함수에서 dataset 변수에 배열로 정의됩니다. 다음은 각 차트가 표시될 때 애니메이션 효과를 보여주는 슬라이드 차트의 예제입니다.



See the CodePen 알메이트 차트 - 애니메이션 효과가 적용된 슬라이드 차트

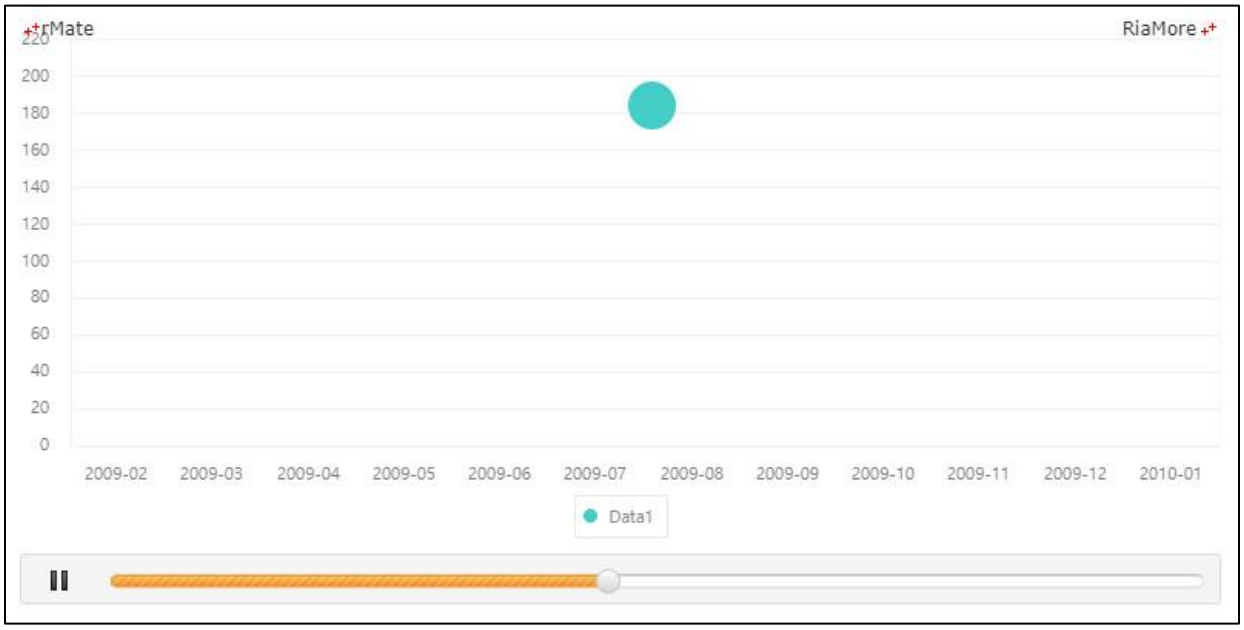
4.34 모션 차트

모션 차트는 한 차트에서 데이터가 변동되는 과정을 슬라이드 애니메이션 효과로 표현하는 차트입니다. 차트의 하단에 재생, 멈춤을 지시하는 버튼과 진행상황에 대한 상태 바가 표시됩니다. 모션 차트는 현재 버블 시리즈, 컬럼 시리즈, 라인 시리즈가 지원되며, <MotionChart> 노드의 series 속성값에 <MotionBubbleSeries>, <MotionColumnSeries>, <MotionLineSeries> 노드를 설정하여 생성할 수 있습니다.

모션 버블 차트

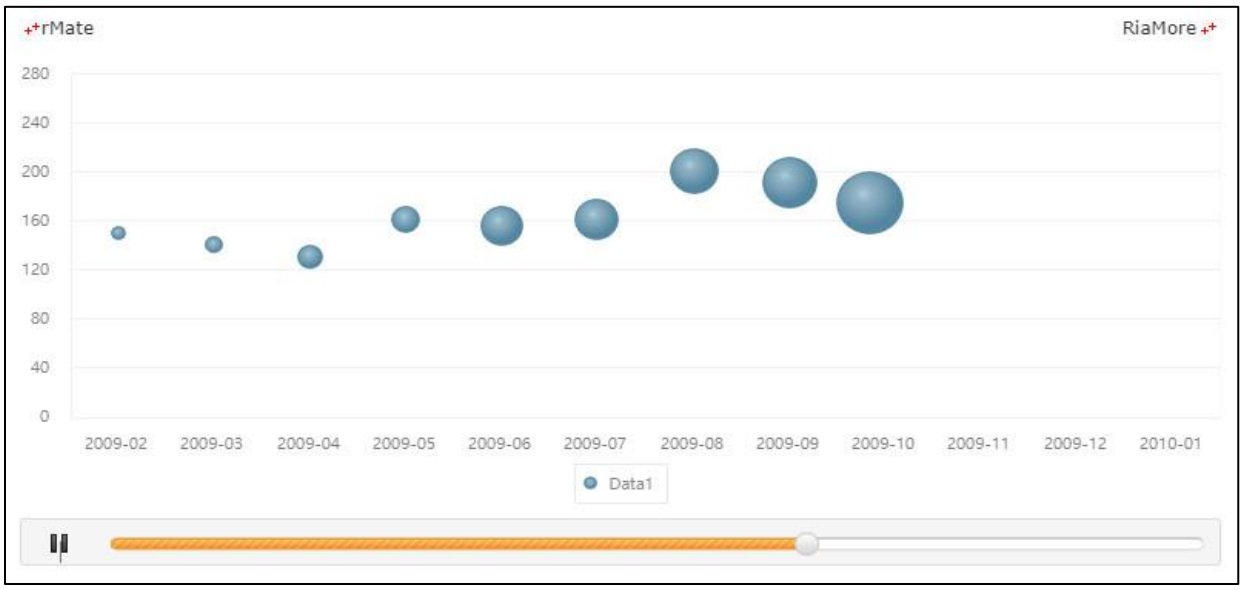
다음은 모션 버블 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<MotionChart showDataTips="true">
  ...
    <MotionBubbleSeries labelPosition="inside" yField="Data1" radiusField="Data2"
       displayName="Data1" formatter="{numFmt}" alwayShowLabels="false">
  ...
</MotionChart>
```



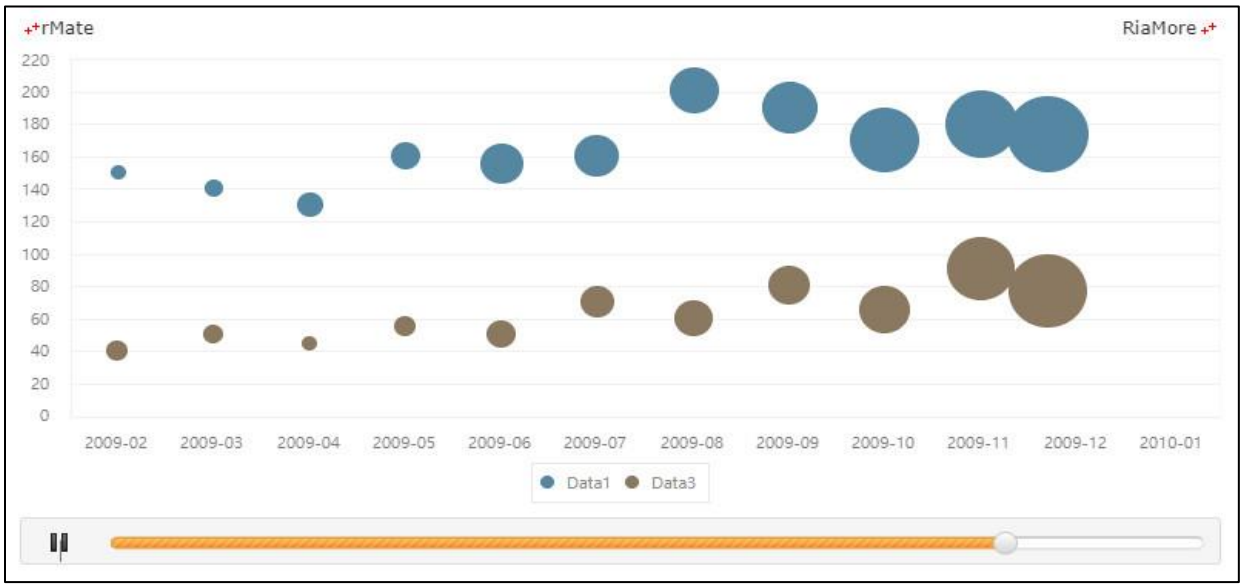
See the CodePen 알메이트 차트 - 모션 버블 차트

다음은 모션 버블 차트 생성시 <MotionBubbleSeries> 노드의 showTrailItems 속성값을 “true” 로 지정하여, 버블이 이동된 자취(trail)를 보여주는 예제입니다.



See the CodePen 알메이트 차트 - 모션 버블 차트 - 버블이 이동된 자취

다음은 모션 버블 차트 생성시 2 개의 <MotionBubbleSeries> 노드를 정의하고, showTrailItems 속성값을 “true” 로 지정하여, 2 개의 버블이 이동된 자취(trail)를 보여주는 예제입니다.

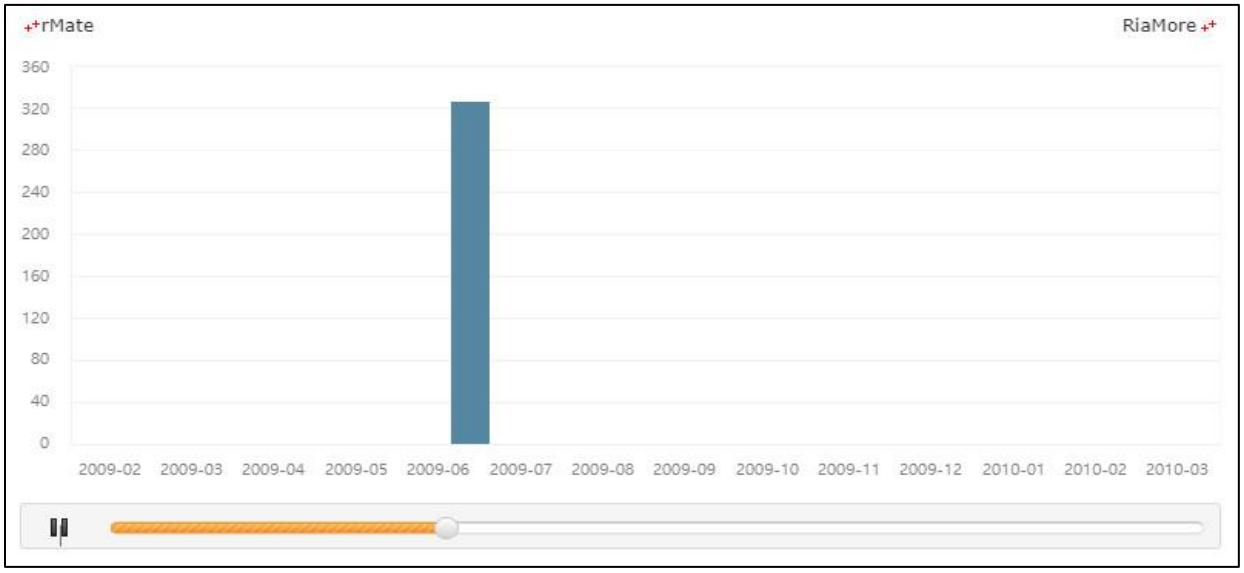


See the CodePen 알메이트 차트 - 다중 시리즈 모션 버블 차트

모션 컬럼 차트

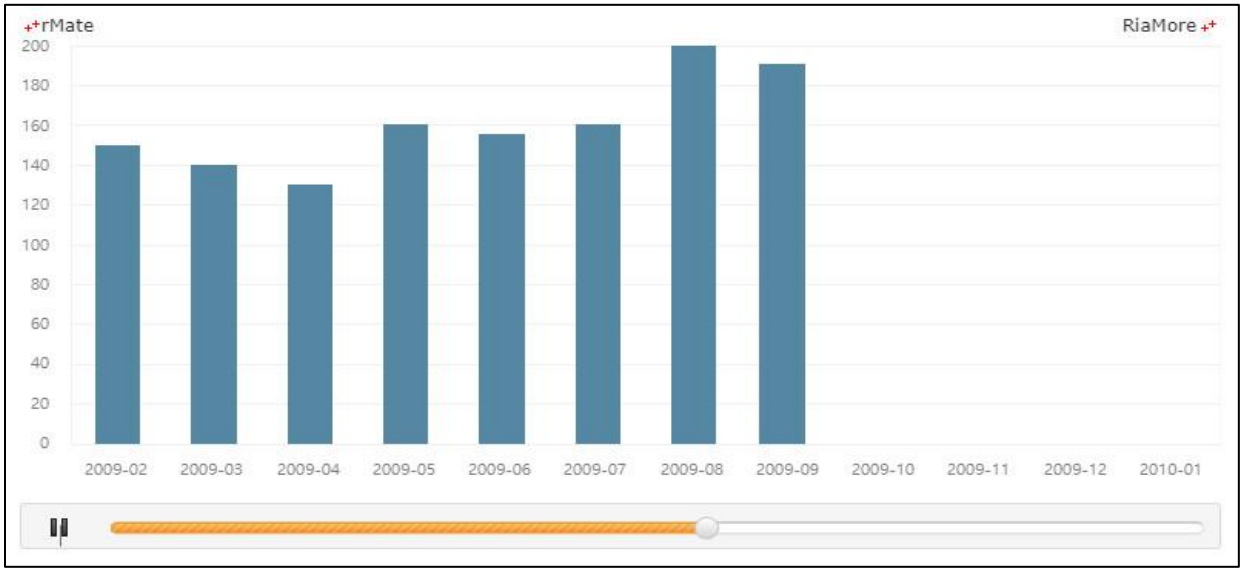
다음은 모션 컬럼 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<MotionChart showDataTips="true">
  ...
    <MotionColumnSeries labelPosition="inside" displayName="Data1" yField="Data1"
       formatter="{numFmt}" color="#ffffff">
  ...
</MotionChart>
```



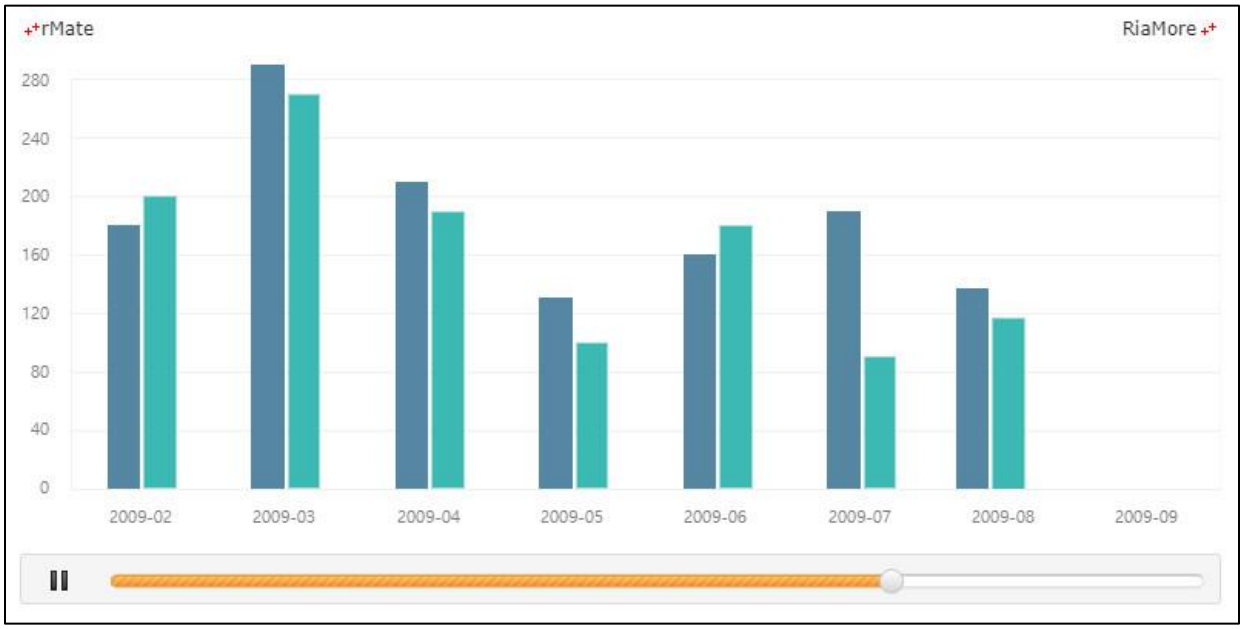
See the CodePen 알메이트 차트 - 모션 컬럼 차트

다음은 모션 컬럼 차트 생성시 <MotionColumnSeries> 노드의 showTrailItems 속성값을 “true” 로 지정하여, 컬럼이 이동된 자취(trail)를 보여주는 예제입니다.



See the CodePen 알메이트 차트 - 모션 컬럼 차트 - 컬럼이 이동된 자취

다음은 모션 컬럼 차트 생성시 2 개의 <MotionColumnSeries> 노드를 정의하고, showTrailItems 속성값을 “true” 로 지정하여, 2 개의 컬럼이 이동된 자취(trail)를 보여주는 예제입니다.

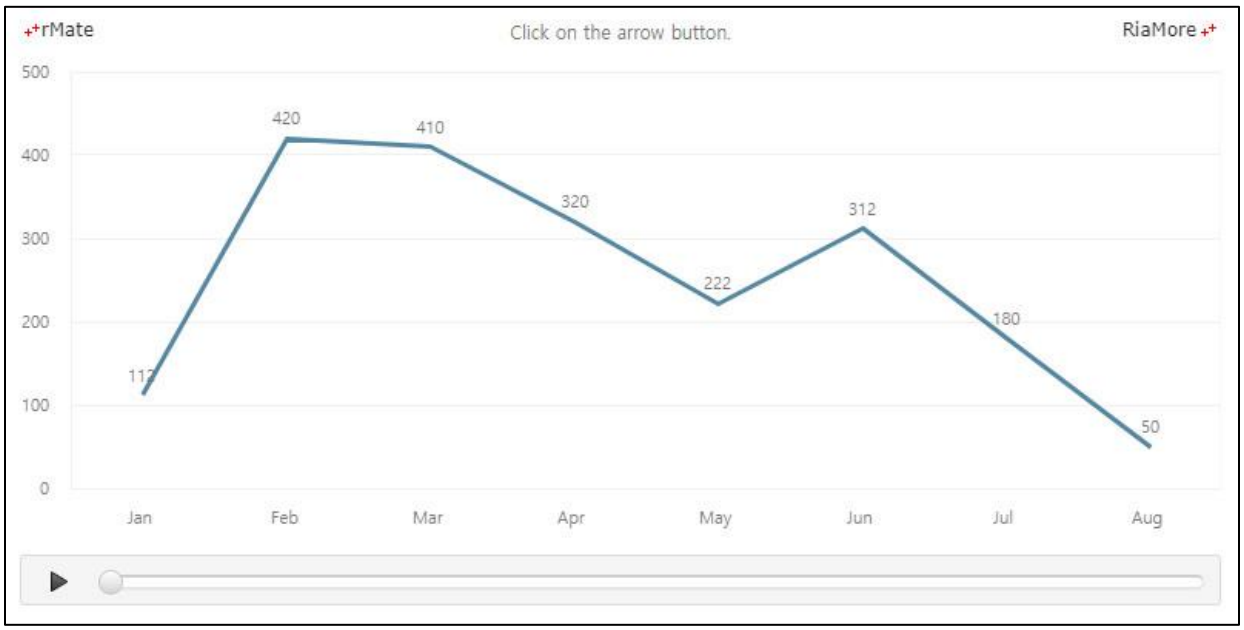


See the CodePen 알메이트 차트 - 다중 시리즈 모션 컬럼 차트

모션 라인 차트

다음은 모션 라인 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<MotionChart showDataTips="true">
  ...
    <MotionLineSeries labelPosition="up" displayName="Profit/Cost/Revenue"
       yField="Data1">
  ...
</MotionChart>
```



See the CodePen 알메이트 차트 - 모션 라인 차트

모션 라인 차트에서는 화면에 애니메이션으로 변화되는 개수만큼의 데이터셋이 배열로 정의되어야 합니다. 다음은 위 예제에 적용된 데이터셋입니다.

```
var chartData = [
[
  {"Month":"Jan", "Data1":112},
  {"Month":"Feb", "Data1":160},
  {"Month":"Mar", "Data1":148},
  ...
],[
  {"Month":"Jan", "Data1":112},
  {"Month":"Feb", "Data1":420},
  {"Month":"Mar", "Data1":410},
  ...
],[
  ...
];
```

4.35 실시간 프리미엄 차트

실시간 프리미엄 차트는 실시간 차트의 기능보다 향상된 기능을 제공합니다. 데이터 로드를 위해서 실시간 차트처럼 원격 호출(RPC, Remote Procedure Call) 방식을 이용하지만 다음과 같은 다른 특징들이 있습니다.

- 차트에 표현되는 데이터 시리즈별로 원격 호출을 설정할 수 있습니다. 예를 들어 차트에 표현되는 데이터 시리즈가 3 개라고 할 때, 각 데이터 시리즈를 위한 3 개의 다른 URL 과 주기를 지정할 수 있습니다.
- 실시간 차트에서는 새로운 데이터가 읽혀지면 차트의 오른쪽 끝에 표현되어 시간이 갈 수록 왼쪽으로 이동하지만, 실시간 프리미엄 차트에서는 왼쪽 끝에 첫 데이터가 표현되고 새로운 데이터들은 점점 오른쪽에 표현됩니다. 그리고 설정된 시간 범위 동안 화면에 존재하게 됩니다. 각 데이터 시리즈별로 주기와 표시되는 시간 범위를 다르게 설정할 수 있습니다.
- 초기 데이터를 차트에 표현해야 할 경우, 차트의 로딩이 완료된 시점에 초기 데이터를 읽어서 표시할 수 있습니다.
- 실시간 프리미엄 차트의 X 축에는 <DateTimeAxis> 노드만 정의할 수 있습니다. 실시간 차트에서 사용 가능한 <CategoryAxis> 노드는 실시간 프리미엄 차트에서 지원되지 않습니다.

실시간 프리미엄 차트는 <HttpMultiServiceRepeater> 노드를 chart 노드와 동일한 레벨(형제 노드)에 정의하여 생성할 수 있습니다. 이 때 실시간 차트에서와 같이 <RealTimeChart> 노드를 정의할 필요는 없습니다. 실시간 프리미엄 차트 생성을 원하는 chart 노드(예, <Column2DChart>, <Combination2DChart>, 등)와 동일한 레벨(형제 노드)에 <HttpMultiServiceRepeater> 노드를 정의하기만 하면 됩니다.

속성명	유효값 (*: 기본값)	설명
baseURL	텍스트	RPC 요청을 보낼 기본 URL 을 지정합니다. <RPCItem> 노드의 url 속성값과 결합된 값(“baseURL + RPCItem 의 url”)이 RPC 요청을 보낼 최종 URL 이 됩니다.
requestTimeout	숫자 (second)	RPC 요청에 대한 최대 응답 대기 시간을 지정합니다.
rpcList	<RPCItem> 배열	RPC 요청 정보를 <RPCItem> 노드에 설정합니다.

showErrorMessage	true(*), false	원격 호출(RPC)이 실패할 경우 경고 메시지를 표시할지 여부를 설정합니다.
targetController	텍스트	chart 노드의 id 속성값을 지정합니다.

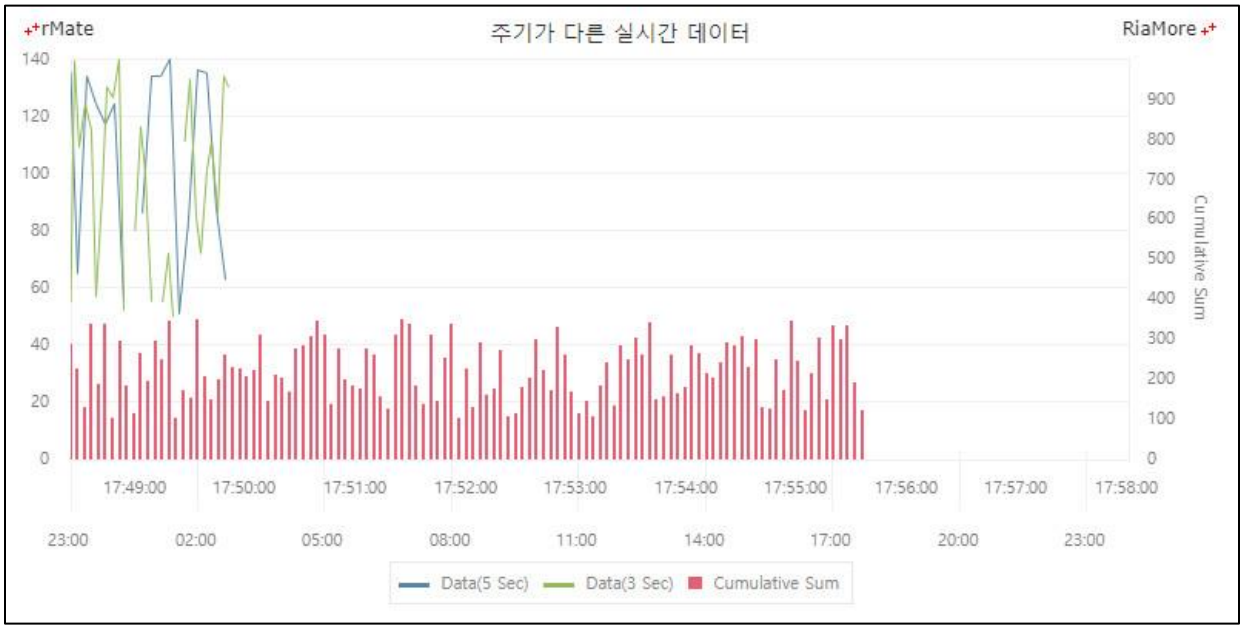
<RPCItem> 노드에서 사용 가능한 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
concurrency	multiple(*), single, last	RPC 요청이 동시에 여러개 발생할 경우 처리 방법을 지정합니다. multiple: 모든 요청을 보냅니다. single: 경고 메시지와 함께 한번에 하나의 요청만 보냅니다. last: 마지막 요청만 보내고 나머지는 취소합니다
interval	숫자 (second)	RPC 요청 주기(초 단위)를 지정합니다.
method	get(*), post	RPC 요청 방법을 지정합니다.
name	텍스트	<RPCItem> 노드의 이름을 지정합니다. 반드시 임의의 이름을 지정해야 합니다.
retryCount	숫자 기본값: 30	RPC 요청 실패 시에 재시도할 횟수를 지정합니다.
target	텍스트	<RPCItem>이 적용될 해당 데이터 시리즈의 id 속성값을 지정합니다.
url	텍스트	RPC 요청을 보낼 URL 을 지정합니다. <HttpMultiServiceRepeater> 노드의 baseURL 속성값과 결합된 값(<HttpMultiServiceRepeater> 노드의 “baseURL + url”)이 RPC 요청을 보낼 최종 URL 이 됩니다.

다음은 두 개의 라인 시리즈와 하나의 컬럼 시리즈를 표현하는 콤비네이션 차트(<Combination2DChart>)에 실시간 프리미엄 차트를 적용하는 코드와 출력된

결과입니다. 실시간 차트에서와 마찬가지로 setData() 함수는 호출하지 않고 setLayout() 함수만 호출합니다.

```
function chartReadyHandler(id) {
  document.getElementById(id).setLayout(layoutStr);
}

<Combination2DChart id="chart" showDataTips="true" dataTipMode="multiple">
  ...
    <Column2DSeries id="columnSeries" xField="date" yField="data60"
        displayName="Cumulative Sum">
      ...
    </Column2DSeries>
    <Line2DSeries id="lineSeries" xField="date" yField="data5" displayName="Data(5
        Sec)">
      ...
    </Line2DSeries>
    <Line2DSeries id="lineSeries2" xField="date" yField="data3" displayName="Data(3
        Sec)" verticalAxis="{vAxis2}" horizontalAxis="{hAxis2}">
      ...
    </Line2DSeries>
  ...
</Combination2DChart>
<HttpMultiServiceRepeater baseURL="http://demo.riamore.net/chartTest/"
        targetController="{chart}" requestTimeout="30">
  <RPCList>
    <RPCItem name="rpc1" url="data3Interval.php" target="{lineSeries2}"
        interval="3" concurrency="last" retryCount="30"/>
    <RPCItem name="rpc2" url="data5Interval.php" target="{lineSeries}" interval="5"
        concurrency="last" retryCount="30"/>
    <RPCItem name="rpc3" url="data23ToCurrent2.php" target="{columnSeries}"
        interval="600" concurrency="last" retryCount="30"/>
  </RPCList>
</HttpMultiServiceRepeater>
```



See the CodePen 알메이트 차트 - 실시간 프리미엄 차트

위 예제에서는 3 개의 <RPCItem> 노드가 설정되었고 각각 2 개의 라인 시리즈와 1 개의 컬럼 시리즈의 데이터를 원격 요청합니다. 2 개의 라인 시리즈(<Line2DSeries>)에 대한 원격 요청 주기는 각각 3 초(interval = “3”), 5 초(interval = “5”)이고, 컬럼 시리즈<Column2DSeries>에 대한 원격 요청 주기는 10 분(interval = “600”)입니다. 2 개의 라인 시리즈의 초기 데이터는 없으며 차트가 최초에 생성된 다음 10 분 동안(데이터 시간 범위) 설정된 주기에 원격 요청하여 하나씩 데이터를 표시하고, 10 분이 지나면 화면에 표시된 데이터를 모두 삭제한 후 다시 10 분 동안의 데이터를 표시하는 것을 반복합니다. 컬럼 시리즈는 10 분 동안에 누적된 데이터를 표시하는 예제인데, 10 분 주기에 원격 요청하여 컬럼을 하나씩 표시하게 됩니다. 이 예제에서 사용된 서버 스크립트의 소스 파일은 아래 디렉토리에서 찾으실 수 있습니다.

설치 디렉토리/Samples/RealtimeServerSamples/

4.36 게이지 차트

게이지 차트는 대시보드를 작성하는데 유용하게 사용되는 차트 유형입니다. 원형, 반원형, 직선형 모양이 지원되며 데이터 값은 눈금을 가르키는 바늘 혹은 지시자를 통해서 표현합니다. (반)원형 게이지 차트에는 (반)원형 게이지 내부 공간에 텍스트를 표현할 수 있는 스코어카드형(<Gauge> 노드로 설정)과 스피드미터형(원형: <CircularGauge> 노드로 설정, 반원형: <HalfCircularGauge> 노드로 설정)이 지원됩니다. 직선형 게이지 차트에는 사각형(수평: <HLinearGauge>, 수직: <VLinearGauge>) 게이지와 실린더형(수평: <HCylinderGauge>, 수직: <VCylinderGauge>) 게이지가 지원됩니다. 원형, 반원형 게이지에서 공통적으로 적용되는 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
minimum	숫자 기본값: 0	게이지의 최소값을 지정합니다.
minimumAngle	0(*) 과 360 사이의 숫자	게이지의 최소값에 해당하는 각도를 지정합니다.
maximum	숫자 기본값: 100	게이지의 최대값을 지정합니다.
maximumAngle	0 과 360(*) 사이의 숫자	게이지의 최대값에 해당하는 각도를 지정합니다.

주의

> 스코어카드형 게이지와 스피트미터형 게이지(원형, 반원형)의 각도값 “0” 이 적용되는 위치가 다른 점에 유의하십시요. 스코어카드형 게이지에서 각도값 “0” 의 위치는 12 시 방향이고, 스피트미터형 게이지(원형, 반원형)에서 각도값 “0” 의 위치는 3 시 방향입니다.

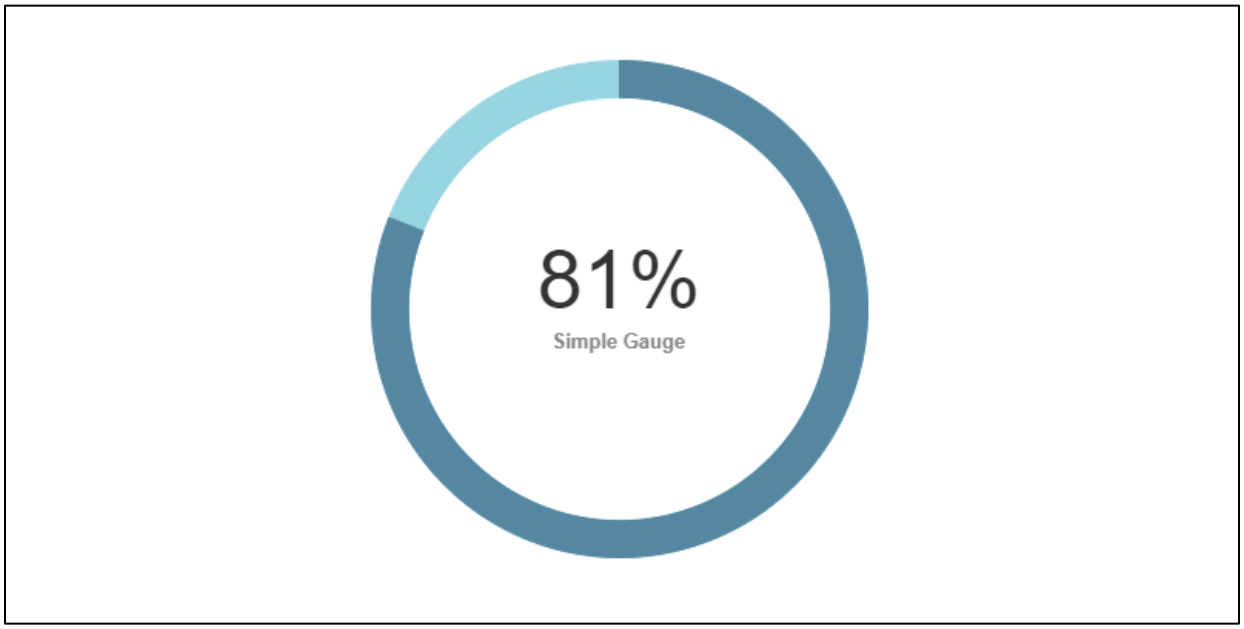
원형 스코어 카드

스코어카드형 게이지에서는 차트의 내부 공간에 텍스트를 표시하여 차트에 관한 유용한 정보를 사용자에게 제공할 수 있습니다. 게이지의 내부 공간의 크기는 innerRatio 속성을 설정하여 조절할 수 있습니다.

속성명	유효값 (*: 기본값)	설명
innerRatio	0(*) 과 1 사이의 숫자	전체 게이지에서 차지하는 내부 공간의 비율을 지정합니다. 게이지의 중심에서 원의 둘레까지(반지름)가 “1” 에 해당합니다.

다음은 스코어카드형 게이지 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 게이지의 내부 공간에 텍스트를 표시하기 위해서 <backgroundElements> 속성을 설정하였습니다.

```
<Gauge minimum="0" maximum="100" minimumAngle="0" maximumAngle="360"
       innerRatio="0.85" fontSize="53" labelJsFunction="valueFunc"
       dataTipJsFunction="dataTipFunc" labelYOffset="-20"
       foregroundColors="[#5587a2]" backgroundColors="[#97d5e0]" color="#333333">
  <backgroundElements>
    <CanvasElement>
      <Label fontWeight="bold" height="23" fontSize="13" horizontalCenter="0"
       verticalCenter="20" text="Simple Gauge" color="#888888"/>
    </CanvasElement>
  </backgroundElements>
</Gauge>
```



See the CodePen 알메이트 차트 - 원형 스코어카드

반원형 스코어카드

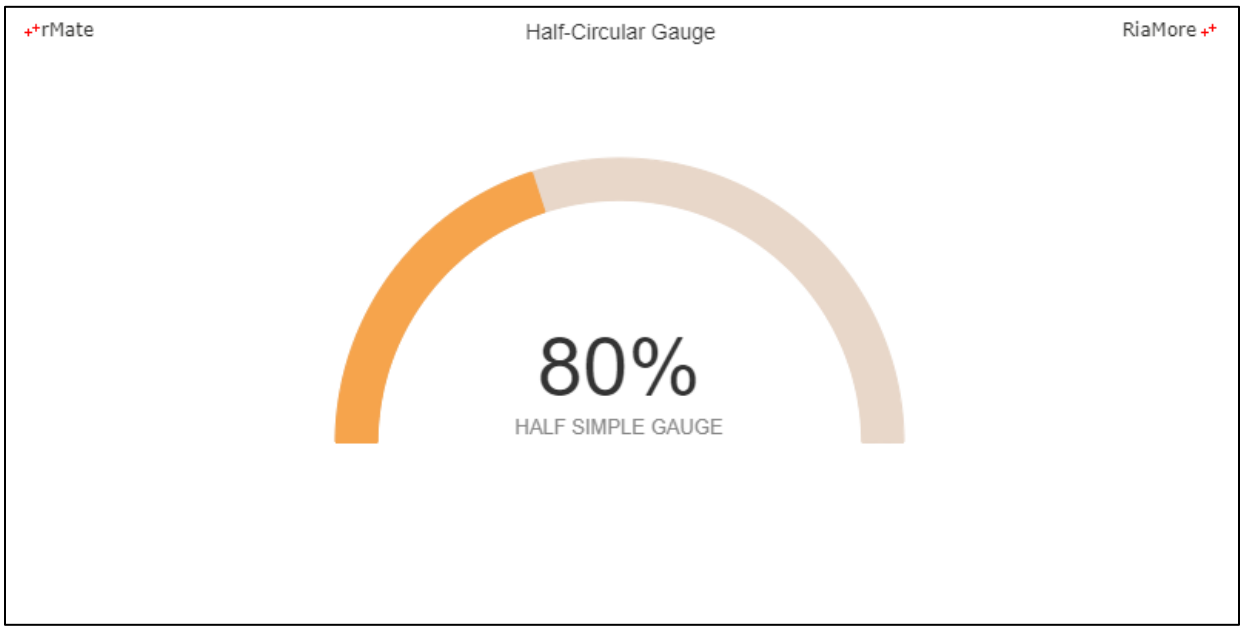
<Gauge> 노드의 maximumAngle 속성을 조절하여 스코어카드형 게이지의 크기를 조절할 수 있습니다. 다음은 maximumAngle 속성값을 “180” 으로 설정하여 반원 모양의 스코어카드형 게이지를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 데이터 값을 표시하는 게이지의 영역(노란색)이 시작되는 위치를 9 시 방향(startAngle = “-90” 혹은 startAngle = “270”)으로 설정하였습니다.

속성명	유효값 (*: 기본값)	설명
startAngle	0(*) 과 360 사이의 값	게이지에서 데이터 값의 영역이 시작되는 각도를 지정합니다.

주의

스코어카드형 게이지와 스피트미터형 게이지(원형, 반원형)의 각도값 “0” 이 적용되는 위치가 다른 점에 유의하십시요. 스코어카드형 게이지에서 각도값 “0” 의 위치는 12 시 방향이고, 스피트미터형 게이지(원형, 반원형)에서 각도값 “0” 의 위치는 3 시 방향입니다.

```
<Gauge minimum="0" maximum="200" minimumAngle="0" maximumAngle="180" startAngle="-
       90" height="380" formatter="{cft}" innerRatio="0.85"
       labelJsFunction="valueLabelFunc" foregroundColors="[#f6a44c]"
       backgroundColors="[#e8d7c9]" color="#333333" fontSize="53"
       verticalOriginRatio="0.7" labelYOffset="-50">
  <backgroundElements>
    <CanvasElement>
      <Label fontSize="13" height="17" color="#888888" horizontalCenter="0"
       verticalCenter="65" text="HALF SIMPLE GAUGE"/>
    </CanvasElement>
  </backgroundElements>
</Gauge>
```



See the CodePen 알메이트 차트 - 반원형 스코어카드

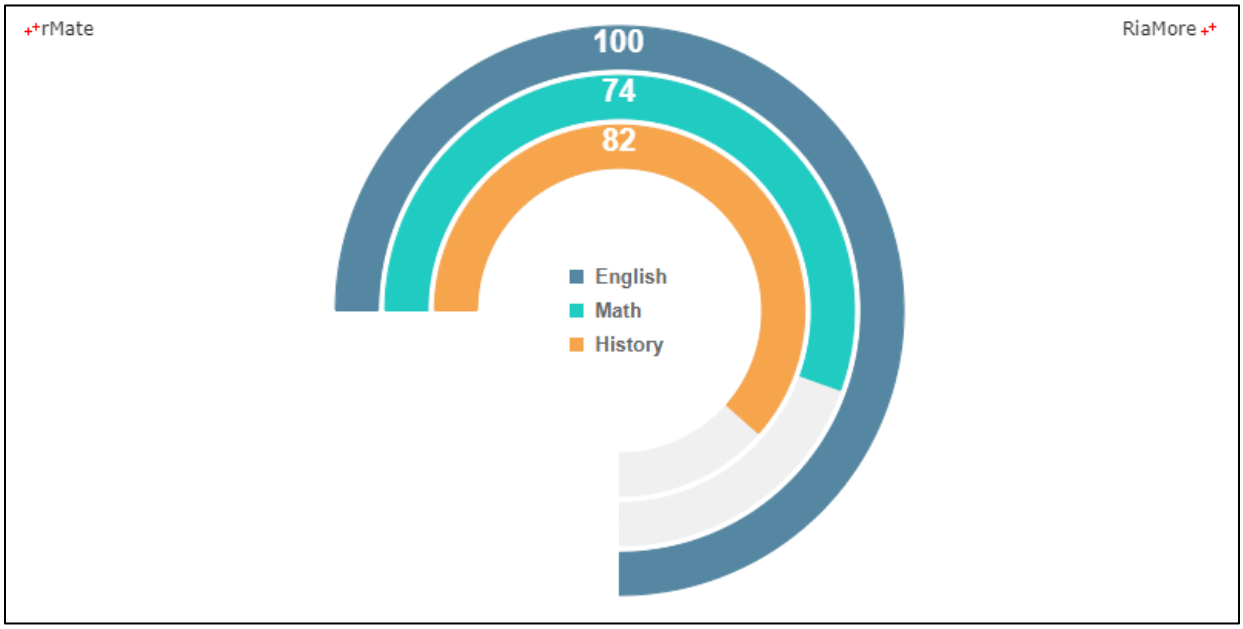
다중값 스코어카드

스코어카드형 게이지에서는 다중값을 표현할 수 있습니다. 이 때 적용되는 데이터의 형식은 레이블과 값이 저장된 객체의 배열이어야 합니다. 다음은 <Gauge> 노드에서 데이터의 레이블과 값이 저장된 필드명을 지정하는 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
nameField	텍스트	데이터의 레이블이 저장된 필드명을 지정합니다.
valueField	텍스트	데이터 값이 저장된 필드명을 지정합니다.

다음은 다중값을 표현하는 스코어카드형 게이지를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Gauge minimum="0" maximum="100" minimumAngle="0" maximumAngle="270" startAngle="-
        90" nameField="name" valueField="value" innerRatio="0.5"
        backgroundColors="[#f0f0f0]" foregroundColors="[#5587a2,#20cbc2,#f6a54c]"
        color="#ffffff" fontSize="20" fontWeight="bold" labelYOffset="-4">
  <backgroundElements>
    <Box width="100%" height="100%" horizontalAlign="center"
        verticalAlign="middle">
      <SubLegend direction="vertical" fontSize="13" color="#666666"
        borderStyle="none">
        <LegendItem label="English">
          ...
        </LegendItem>
        <LegendItem label="History">
          ...
        </LegendItem>
      </SubLegend>
    </Box>
  </backgroundElements>
</Gauge>

var chartData =
[
  {"name" : "English", "value" : 100},
  {"name" : "Math", "value" : 74},
  {"name" : "History", "value" : 82}
];
```



See the CodePen 알메이트 차트 - 다중값 스코어카드

원형 게이지

스피드미터형 게이지는 데이터 값을 바늘이 가르키는 방향으로 표현합니다. 바늘의 모양은 다음 속성들을 이용하여 다양한 형태로 표시가 가능합니다.

속성명	유효값 (*: 기본값)	설명
needleBackLengthRatio	0(*) 과 1 사이의 숫자	바늘 커버의 중심을 기준으로 바늘 뒷부분의 길이를 지정합니다. 게이지의 중심에서 원의 둘레까지(반지름)가 “1” 에 해당합니다.
needleCoverFill	<SolidColor>	바늘 커버의 색의 스타일을 지정합니다.
needleCoverRadius	숫자 기본값: 15	바늘 커버의 반지름 크기를 지정합니다.
needleCoverStroke	<Stroke>	바늘 커버 선의 스타일을 지정합니다.
needleFill	<SolidColor>	바늘 색의 스타일을 지정합니다.
needleLengthRatio	0 과 1 사이의 숫자 기본값: 0.85	바늘의 길이를 지정합니다. 게이지의 중심에서 원의 둘레까지(반지름)가 “1” 에 해당합니다.
needlePointStyle	steeple(*), rounding	바늘 포인트의 모양을 지정합니다.
needleStroke	<Stroke>	바늘 선의 스타일을 지정합니다.
needleThickness	숫자 기본값: 10	바늘 선의 두께를 지정합니다.

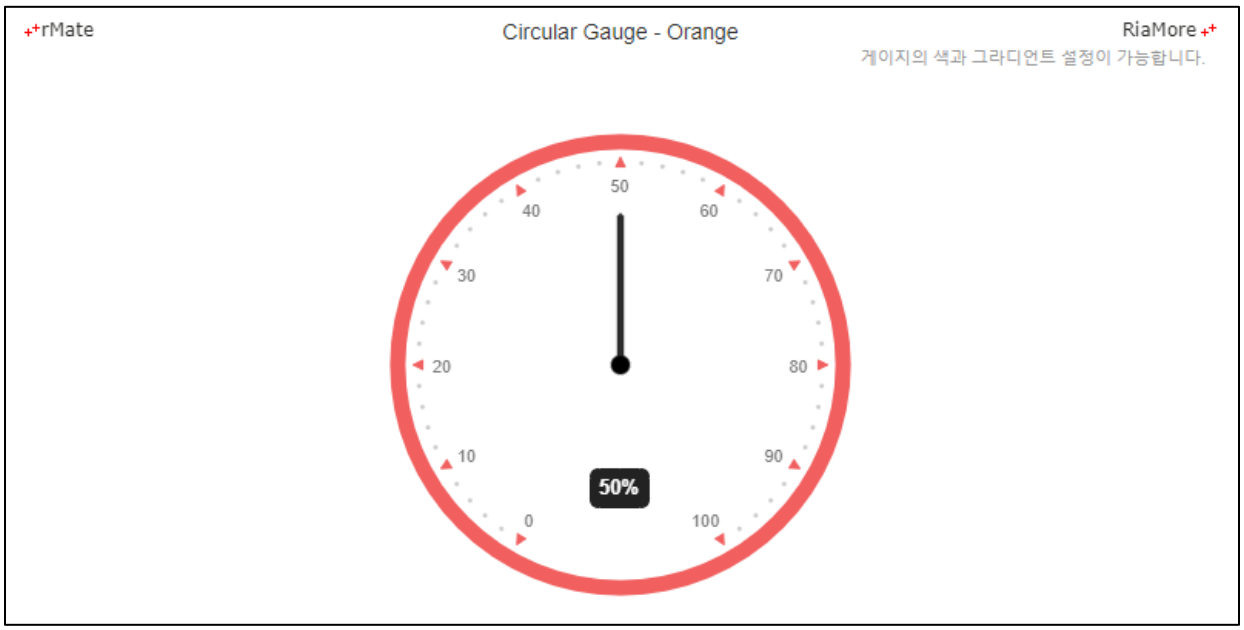
원형 게이지의 모양은 다음 속성들을 이용하여 다양한 형태로 표현이 가능합니다.

속성명	유효값 (*: 기본값)	설명

frameFill	<SolidColor>	게이지 프레임 색의 스타일을 지정합니다.
frameStroke	<Stroke>	게이지 프레임 선의 색의 스타일을 지정합니다.
tickFill	<SolidColor>	게이지 눈금(tick) 색의 스타일을 지정합니다.
tickStroke	<Stroke>	게이지 눈금(tick) 선의 색의 스타일을 지정합니다.
minorTickFill	<SolidColor>	게이지 보조 눈금(minor tick) 색의 스타일을 지정합니다.
minorTickStroke	<Stroke>	게이지 보조 눈금(minor tick) 선의 색의 스타일을 지정합니다.
outFrameFill	<SolidColor>	showOutFrame 속성값이 “true” 일 때 게이지 프레임 색의 스타일을 지정합니다.
outFrameStroke	<Stroke>	showOutFrame 속성값이 “true” 일 때 게이지 프레임 선의 색의 스타일을 지정합니다.

다음은 스피드미터형 게이지 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

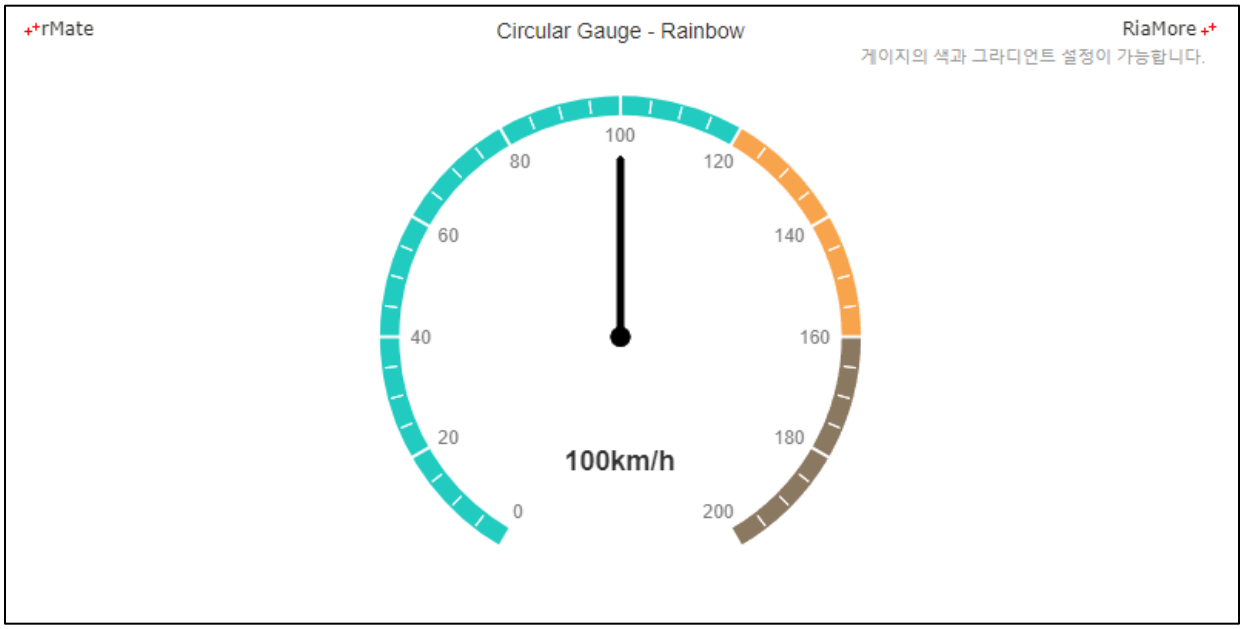
```
<CircularGauge minimumAngle="30" maximumAngle="330" startAngle="90"
       labelJsFunction="labelFunc" value="50" interval="10" minorInterval="2"
       needleLengthRatio="0.7" needleBackLengthRatio="0" valueXOffset="0"
       valueYOffset="80" tickLabelStyleName="tickText"
       valueLabelStyleName="valueText" editMode="true" liveDragging="true"
       bounceAnimating="true" labelGap="10" tickGap="-4" showDataTip="true"
       needleCoverRadius="6" pointThickness="2" needleThickness="4"
       majorTickType="triangle">
  <frameStroke>
    <Stroke color="#f15f5f" weight="10"/>
  </frameStroke>
  <tickStroke>
    <Stroke color="#f15f5f"/>
  </tickStroke>
  <tickFill>
    <SolidColor color="#f15f5f"/>
  </tickFill>
</CircularGauge>
```



See the CodePen 알메이트 차트 - 원형 게이지

다음은 전체 트랙을 눈금에 따라서 3 가지 범위로 나누고 다른 스타일을 적용한 스피드미터형 게이지 차트를 생성하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<CircularGauge startAngle="90" minimumAngle="30" maximumAngle="330" minimum="0"
        maximum="200" labelJsFunction="labelFunc" needleCoverRadius="6"
        interval="20" minorInterval="5" padding="0" labelGap="14" tickGap="-1"
        formatter="{numFmt}" tickLabelStyleName="tickText"
        valueLabelStyleName="valueText" needlePointStyle="steeple"
        pointThickness="2" needleThickness="4" needleLengthRatio="0.75"
        editMode="true" showDataTip="true" valueYOffset="80"
        animationDuration="1000" majorTickType="line" minorTickType="line"
        minorTickRadius="4" tickRadius="7" showTrackColor="true" trackValues="[0,
        120, 160, 200]" trackColors="[#21cbc0,#f8a44c,#8a7860]"
        trackAlphas="[1,1,1]" trackInnerRadius="0.92" trackOuterRadius="1">
  <frameStroke>
    ...
  <tickStroke>
    ...
  <minorTickStroke>
    ...
  <needleFill>
    ...
  <needleStroke>
    ...
  <needleCoverFill>
    ...
  <needleCoverStroke>
    ...
</CircularGauge>
```



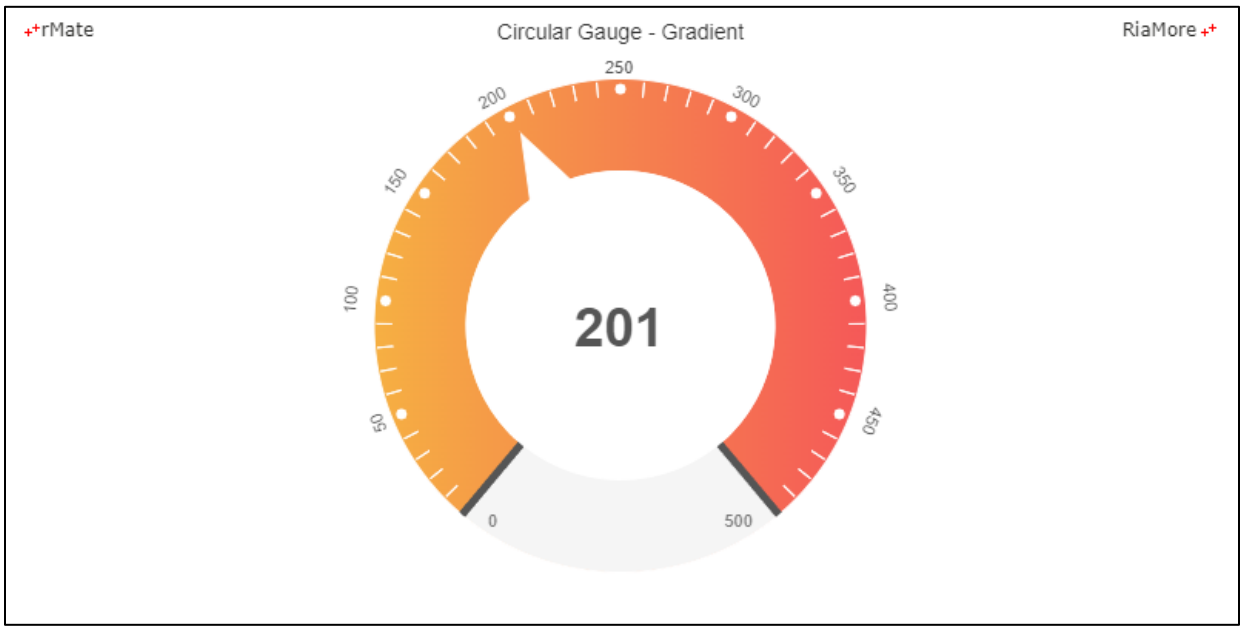
See the CodePen 알메이트 차트 - 스피드미터형 원형 게이지

위에서 설명한 예제에서는 트랙을 3 가지로 나누고 다른 스타일을 설정하기 위해서 다음 속성들을 이용합니다.

속성명	유효값 (*: 기본값)	설명
trackValues	숫자 배열	minimum 속성값과 maximum 속성값 사이의 값으로 트랙의 범위들을 지정합니다. 예를 들어 trackValues = “[0,50,80,100]”, minimum = “0”, maximum = “100” 이면, 0 ~ 50, 50 ~ 80, 80 ~ 100 세가지 범위가 설정됩니다.
trackColors	<Stroke>	trackValues 속성에 설정된 각 범위에 순서대로 색을 지정합니다.
trackAlphas	<SolidColor>	trackValues 속성에 설정된 각 범위에 순서대로 색의 투명도를 지정합니다.

게이지 프레임의 스타일에 그라디언트 효과를 설정할 수 있습니다. 다음은 이를 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

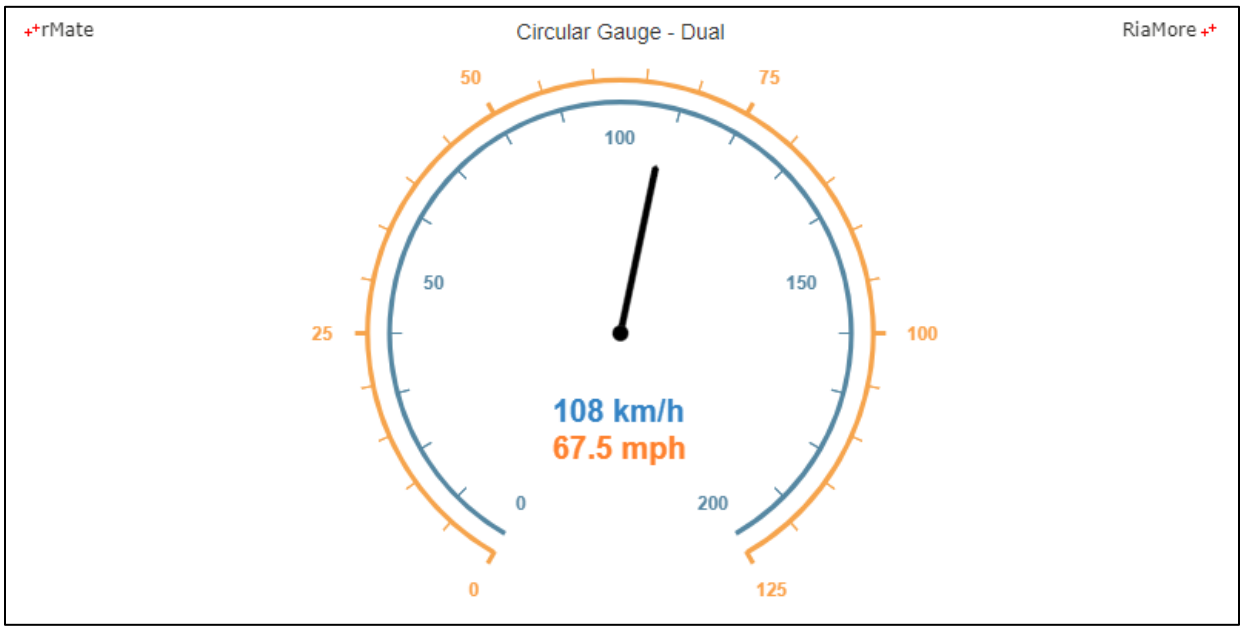
```
<CircularGauge padding="8" startAngle="90" minimumAngle="40" maximumAngle="320"
       minimum="0" maximum="500" value="200" interval="50" minorInterval="10"
       formatter="{numFmt}" tickLabelStyleName="tickText"
       valueLabelStyleName="valueText" editMode="true" majorTickType="circle"
       liveDragging="false" tickGap="-3" labelGap="-5" showDataTip="false"
       tickLabelPlacement="outside" tickColor="#1B699A" needleCoverRadius="100"
       needleThickness="100" pointThickness="0" tickRadius="2"
       needleLengthRatio="0.9" minorTickType="line" minorTickRadius="5"
       needlePointStyle="rounding" isValueTop="true" animationDuration="1000"
       bounceAnimating="true" rotateTickLabel="true" showOutFrame="true"
       outFrameLabelXOffset="0" outFrameLabelYOffset="0.2">
  <frameFill>
    <LinearGradient angle="0">
      <entries>
        <GradientEntry color="#f6af43" ratio="0"/>
        <GradientEntry color="#f55a58" ratio="1"/>
      </entries>
    </LinearGradient>
  </frameFill>
  ...
</CircularGauge>
```



See the CodePen 알메이트 차트 - 스피드미터형 원형 게이지 - 그라디언트 효과

속성명 앞에 second 를 붙여서 보조 트랙을 표시할 수 있습니다. 다음은 이를 이용하여 킬로미터(km)와 마일(mi)을 동시에 게이지에 표시하는 코드와 이를 적용해서출력한 차트의 예제입니다.

```
<CircularGauge startAngle="90" minimumAngle="30" maximumAngle="330" editMode="true"
        liveDragging="true" coverRadiusRatio="0.1" needleThickness="4"
        needleCoverRadius="5" needleLengthRatio="0.7" needleBackLengthRatio="0"
        needlePointStyle="steeple" pointThickness="2" labelGap="14"
        valueYOffset="60" tickLabelStyleName="tickText"
        valueLabelStyleName="valueText" labelJsFunction="labelFunc" minimum="0"
        maximum="125" interval="25" minorInterval="5" tickGap="-14" tickRadius="4"
        minorTickRadius="4" minorTickType="line" majorTickType="line"
        tickLabelPlacement="outside" showTrackColor="true" trackValues="[0,125]"
        trackColors="[#f6a44c]" trackAlphas="[1]" trackInnerRadius="1.03"
        trackOuterRadius="1.05" secondTickLabelStyleName="tickText2"
        secondLabelJsFunction="labelFunc2" secondMinimum="0" secondMaximum="200"
        showSecondTick="true" secondInterval="50" secondTickGap="8"
        secondTickRadius="4" secondMinorInterval="10" showSecondMinorTick="true"
        secondMinorTickGap="8" secondMinorTickRadius="4" secondMinorTickType="line"
        showSecondValueLabel="true" showSecondTickLabels="true"
        showSecondTrackColor="true" secondTrackValues="[0,125]"
        secondTrackColors="[#5587a2]" secondTrackAlphas="[1]"
        secondTrackInnerRadius="0.94" secondTrackOuterRadius="0.96">
  ...
  <secondMinorTickStroke>
    <Stroke color="#5587a2"/>
  </secondMinorTickStroke>
  <secondTickStroke>
    <Stroke color="#5587a2" weight="3"/>
  </secondTickStroke>
</CircularGauge>
```

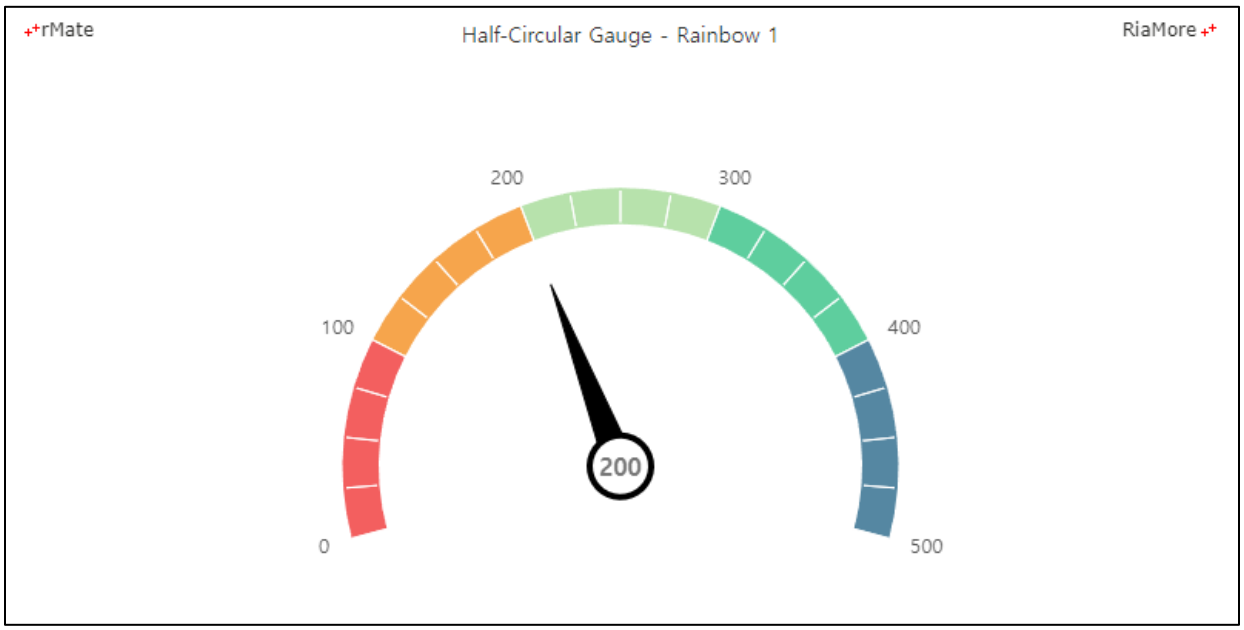


See the CodePen 알메이트 차트 - 스피드미터형 원형 게이지 - 보조 트랙

반원형 게이지

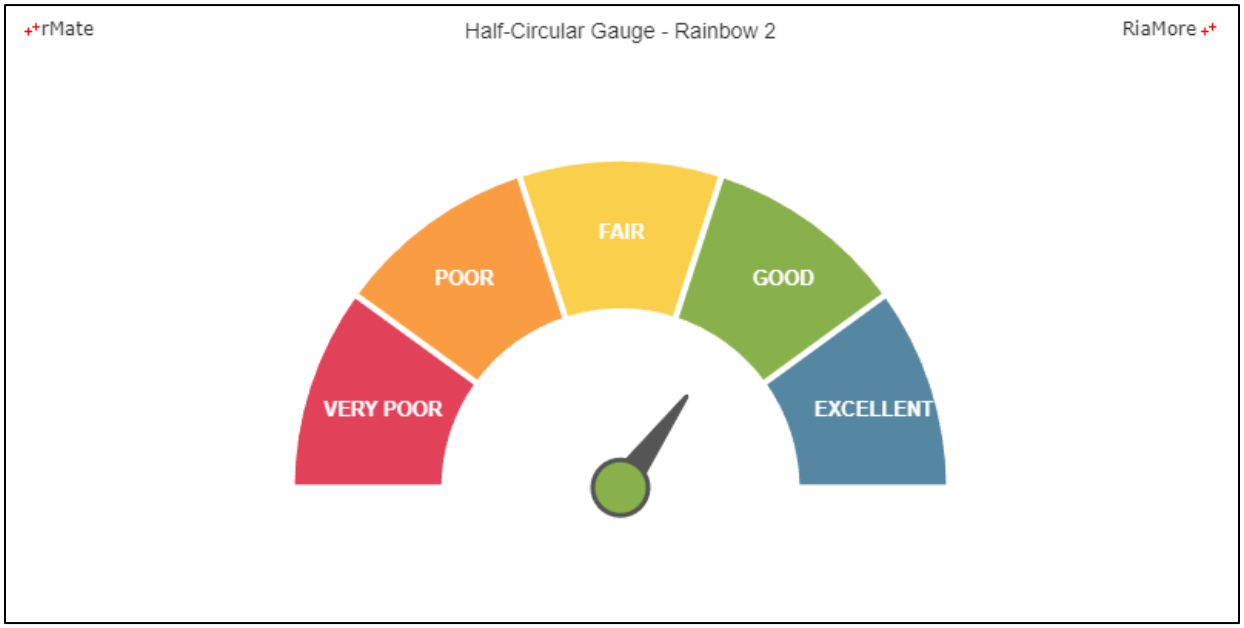
스피드미터형 반원 게이지는 <HalfCircularGauge> 노드를 설정하여 생성할 수 있습니다. 반원형 게이지에서 사용되는 속성들의 적용 방식은 원형 게이지에서 속성들을 적용하는 방식과 동일합니다. 다음은 반원형 게이지를 생성하는 코드와 이를 적용해서출력한 차트의 예제입니다.

```
<HalfCircularGauge startAngle="165" minimumAngle="0" maximumAngle="210" minimum="0"
        maximum="500" interval="100" minorInterval="25" formatter="{numFmt}"
        editMode="false" liveDragging="true" showDataTip="true" valueXOffset="0"
        valueYOffset="0" isValueTop="true" tickLabelStyleName="tickText"
        valueLabelStyleName="valueLabelStyle" needleCoverRadius="20"
        needleThickness="20" pointThickness="0" needleLengthRatio="0.7"
        needlePointStyle="steeple" needleBackLengthRatio="0" tickGap="0"
        labelGap="15" tickLabelPlacement="outside" majorTickType="line"
        tickRadius="12" minorTickType="line" minorTickRadius="10"
        showTrackColor="true" trackValues="[0,100,200,300,400,500]"
        trackColors="[#f35f5f,#f6a54c,#b7e2ac,#5ece9e,#5587a2]"
        trackAlphas="[1,1,1,1,1]" trackInnerRadius="0.87" trackOuterRadius="1">
...
</HalfCircularGauge>
```



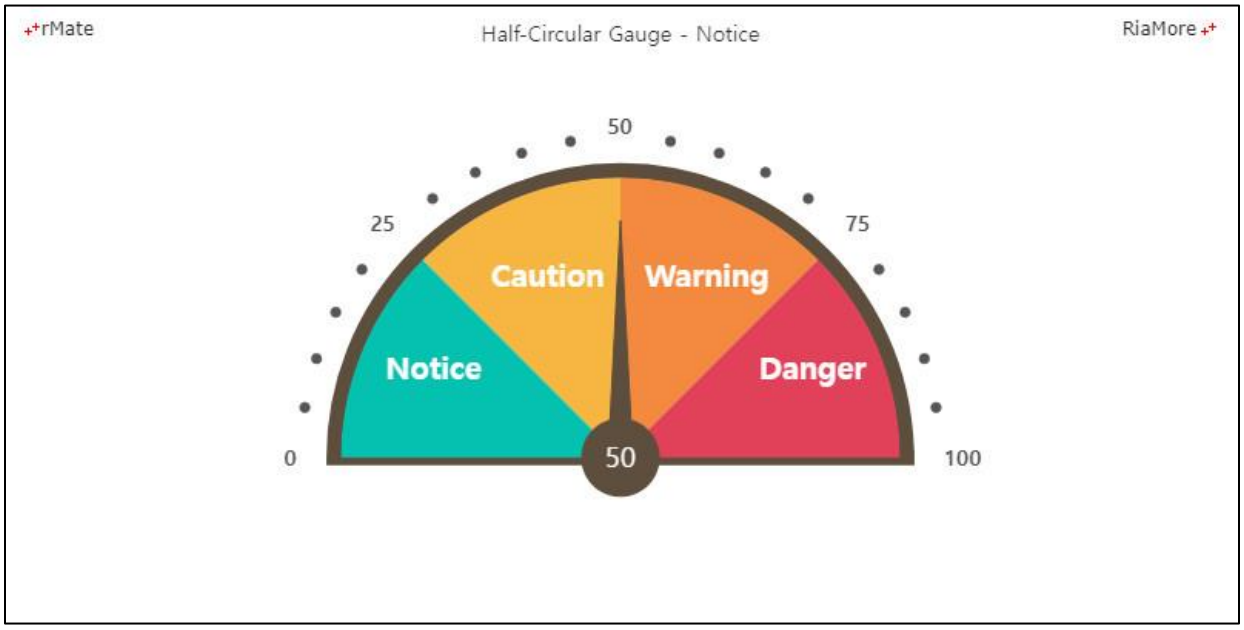
See the CodePen 알메이트 차트 - 반원형 게이지

다음은 트랙의 레이블에 텍스트를 표시하는 반원형 게이지의 예제입니다.



See the CodePen 알메이트 차트 - 반원형 게이지 - 트랙의 레이블에 텍스트 표시

다음은 트랙에 눈금과 텍스트를 표시하는 반원형 게이지의 예제입니다.

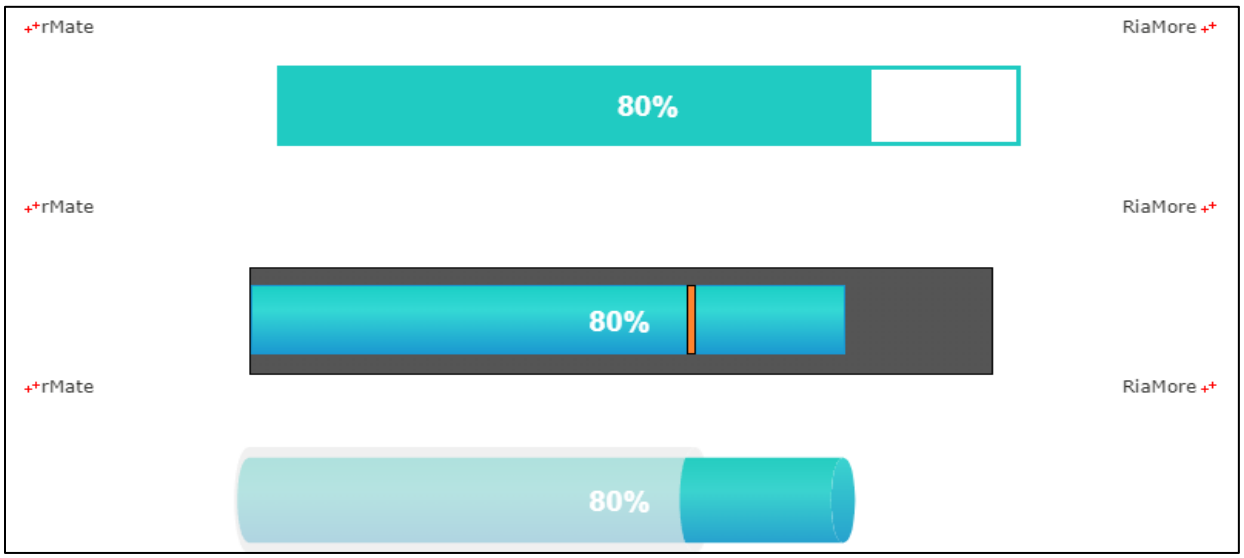


See the CodePen 알메이트 차트 - 반원형 게이지 - 트랙에 눈금과 텍스트 표시

수평 게이지

수평 게이지는 가로 직사각형 형태로 표현되며, 실린더형과 직선형이 지원됩니다. 실린더형 수평 게이지는 <HCylinderGauge> 노드를 설정하여 생성하고, 직선형 수평
게이지는 <HLinearGauge> 노드를 설정하여 생성합니다.

다음은 수평 게이지를 표현한 예제입니다.

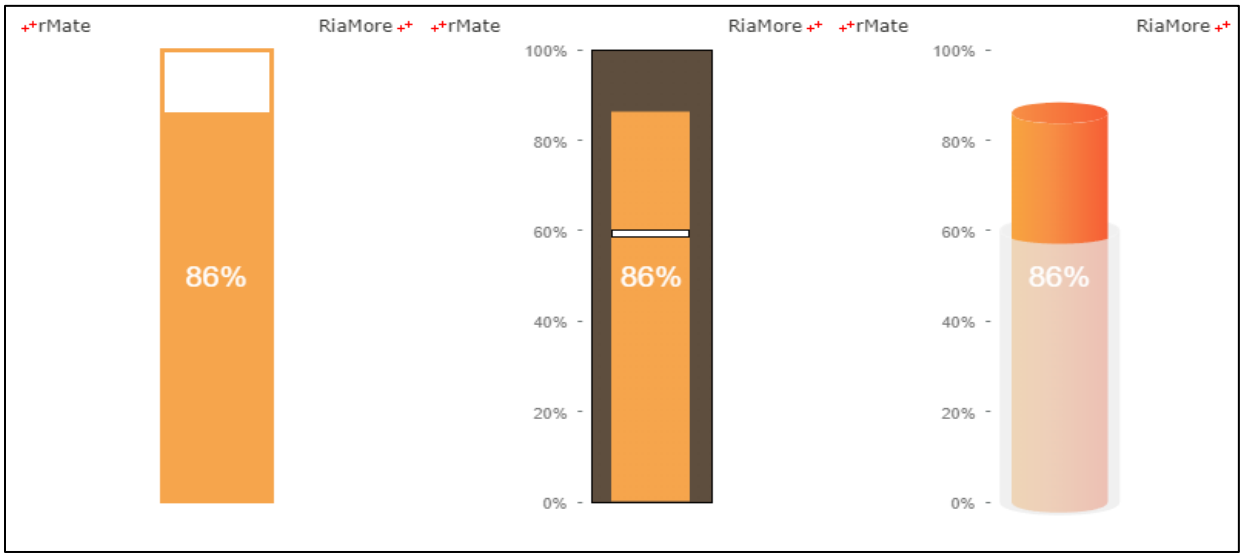


See the CodePen 알메이트 차트 - 수평 게이지

수직 게이지

수직 게이지는 세로 직사각형 형태로 표현되며, 실린더형과 직선형이 지원됩니다. 실린더형 수직 게이지는 <VCylinderGauge> 노드를 설정하여 생성하고, 직선형 수직
게이지는 <VLinearGauge> 노드를 설정하여 생성합니다. 수평/수직 게이지는 보통 목표값에 대비한 실적값의 달성 정도를 표현합니다. 실린더형에서는 실적값을 안쪽 실린더의 길이로 표현하고, 직선형에서는 안쪽 막대의 길이로 표현합니다.

다음은 수직 게이지를 표현한 예제입니다.



See the CodePen 알메이트 차트 - 수직 게이지

4.37 픽토리얼 차트

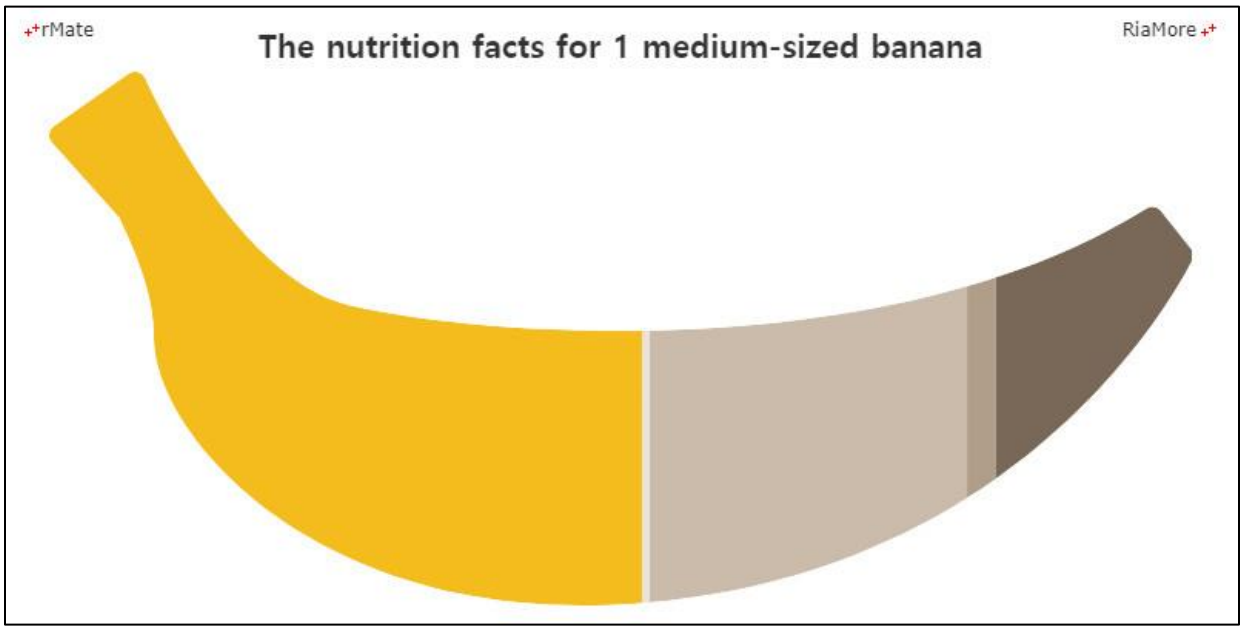
픽토리얼 차트는 SVG 를 활용하는 차트로 일련의 데이터의 수치를 SVG 의 크기, 개수 또는 내부에 색상을 채움으로 표현하는 차트입니다. 픽토리얼 차트는 <PictorialChart> 노드의 series 속성값에 <PictorialSeries> 노드를 설정하여 생성할 수 있습니다. 픽토리얼 차트에서 수치를 표현하는 방법은 <PictorialSeries> 노드의 itemRenderer 에 따라 다르게 표현할 수 있습니다. 다음에는 itemRenderer 속성에 설정 가능한 값과 이에 따른 표현 방식이 설명되어 있습니다.

- PictorialH100PerItemRenderer : 값을 하나의 SVG 에 가로로 나누어 색상으로 표현합니다.(기본값)
- PictorialV100PerItemRenderer : 값을 하나의 SVG 에 세로로 나누어 색상으로 표현합니다.
- PictorialHFillItemRenderer : 값의 개수에 맞춰 SVG 를 가로로 나열하고, 각 값들은 색상을 채워 표현합니다.
- PictorialHSizeItemRenderer : 값의 개수에 맞춰 SVG 를 가로로 나열하고, 각 값들은 크기로 표현합니다.
- PictorialVFillItemRenderer : 값의 개수에 맞춰 SVG 를 세로로 나열하고, 각 값들은 색상을 채워 표현합니다.
- PictorialBarItemRenderer : Bar 차트와 비슷한 모양으로 SVG 들을 나열하고, 각 값들은 개수와 색상으로 표현합니다.
- PictorialColumnItemRenderer : Column 차트와 비슷한 모양으로 SVG 들을 나열하고, 각 값들은 개수와 색상으로 표현합니다.
- PictorialCountItemRenderer : 왼쪽 상단부터 오른쪽으로 SVG 들을 나열하고, 필요한 경우 줄바꿈이 허용됩니다. 각 값들은 개수와 색상으로 표현합니다.
- PictorialCountStackItemRenderer : PictorialCountItemRenderer 와 동일하나, 차트의 itemRenderer 가 모두 PictorialCountStackItemRenderer 일 경우 각 Series 를 하나로 이어서 값들을 표현합니다.

PictorialH100PerItemRenderer

다음은 itemRenderer 로 PictorialH100PerItemRenderer 를 적용한 차트의 예제입니다.

```
<PictorialChart>
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_banana.svg"
       itemRenderer="PictorialH100PerItemRenderer">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </PictorialSeries>
  </series>
</PictorialChart>
```



See the CodePen 알메이트 차트 – 픽토리얼 H100Per 차트

PictorialV100PerItemRenderer

다음은 itemRenderer 로 PictorialV100PerItemRenderer 를 적용한 차트의 예제입니다.

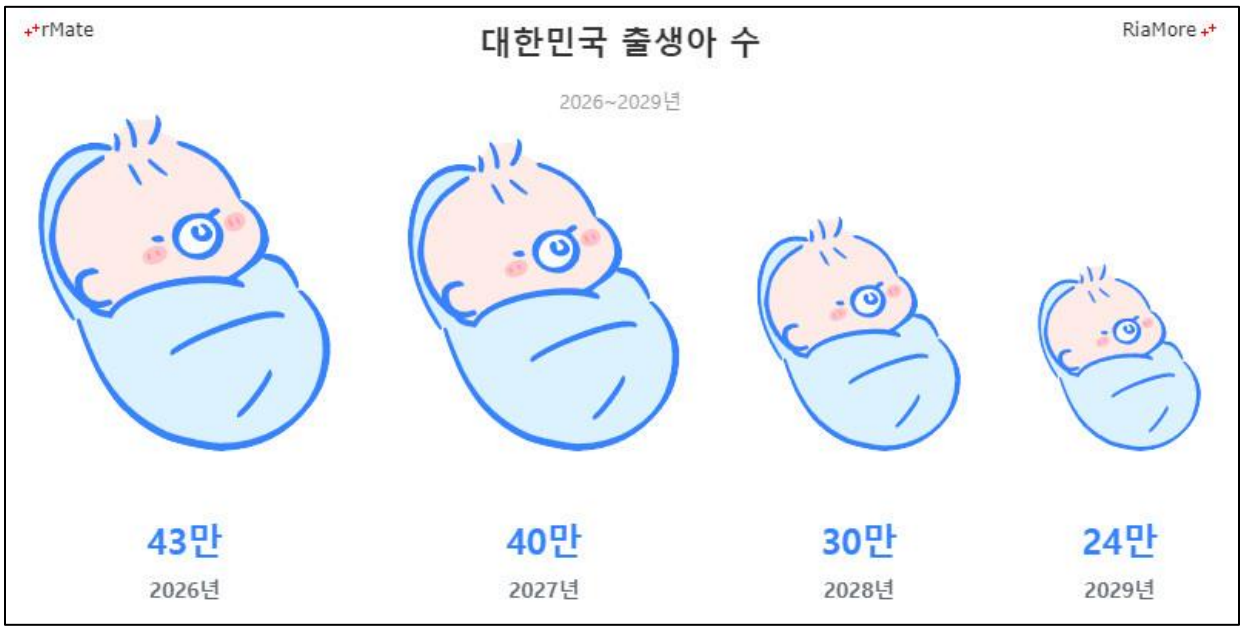
```
<PictorialChart>
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_coffee.svg"
       itemRenderer="PictorialV100PerItemRenderer">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </PictorialSeries>
  </series>
</PictorialChart>
```

See the CodePen 알메이트 차트 – 픽토리얼 V100Per 차트

PictorialHSizeItemRenderer

다음은 itemRenderer 로 PictorialHSizeItemRenderer 를 적용한 차트의 예제입니다. 데이터의 개수만큼 Item 을 가로로 추가하며, 값에 따라서 Item 의 크기가 커집니다

```
<PictorialChart>
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_baby.svg"
       itemRenderer="PictorialHSizeItemRenderer">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </PictorialSeries>
  </series>
</PictorialChart>
```



See the CodePen 알메이트 차트 – 픽토리얼 HSize 차트

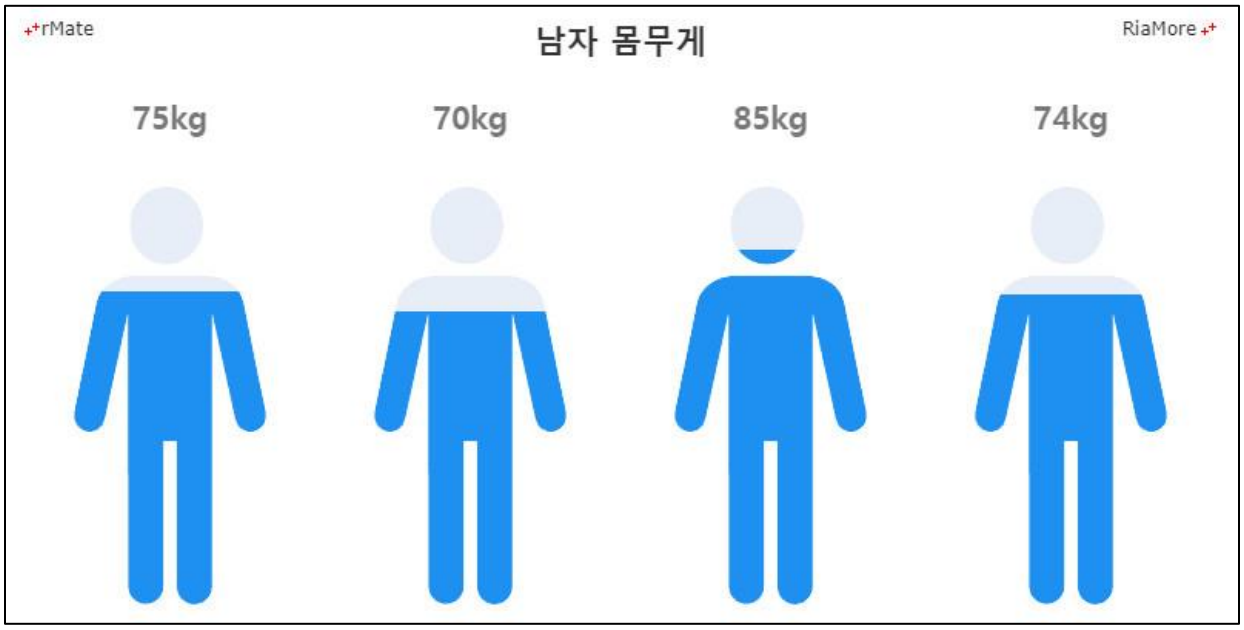
Icon 위치선정 로직

아래의 Pictorial 차트에서는 PictorialChart 하위에 PictorialSeries 하위에 ChartItem 하위에 Icon 이라는 단위를 정의합니다. Icon 의 경우 입력된 SVG 에 반드시 정비례하게 크기가 조절되며, 차트 크기에 너비와 높이 중 큰 쪽을 맞췄을 때, 작은 부분은 최소 여백으로 처리하여 중앙정렬합니다. <PictorialChart> 노드의 iconHorizontalAlign, iconVerticalAlign 로 정렬을 변경할 수 있고, iconVerticalMinGap, iconHorizontalMinGap 으로 최소 여백보다 크게 여백을 설정할 수 있습니다.

PictorialHFillItemRenderer

다음은 itemRenderer 로 PictorialHFillItemRenderer 를 적용한 차트의 예제입니다. 데이터의 개수만큼 Item 을 가로로 추가하며, 각 Item 은 1 개의 Icon 을 가지고 있습니다. Icon 의 최대값은 <PictorialChart> 노드의 perIconValue 로 설정할 수 있습니다.

```
<PictorialChart perIconValue="100">
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_man.svg"
       itemRenderer="PictorialHFillItemRenderer">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </PictorialSeries>
  </series>
</PictorialChart>
```

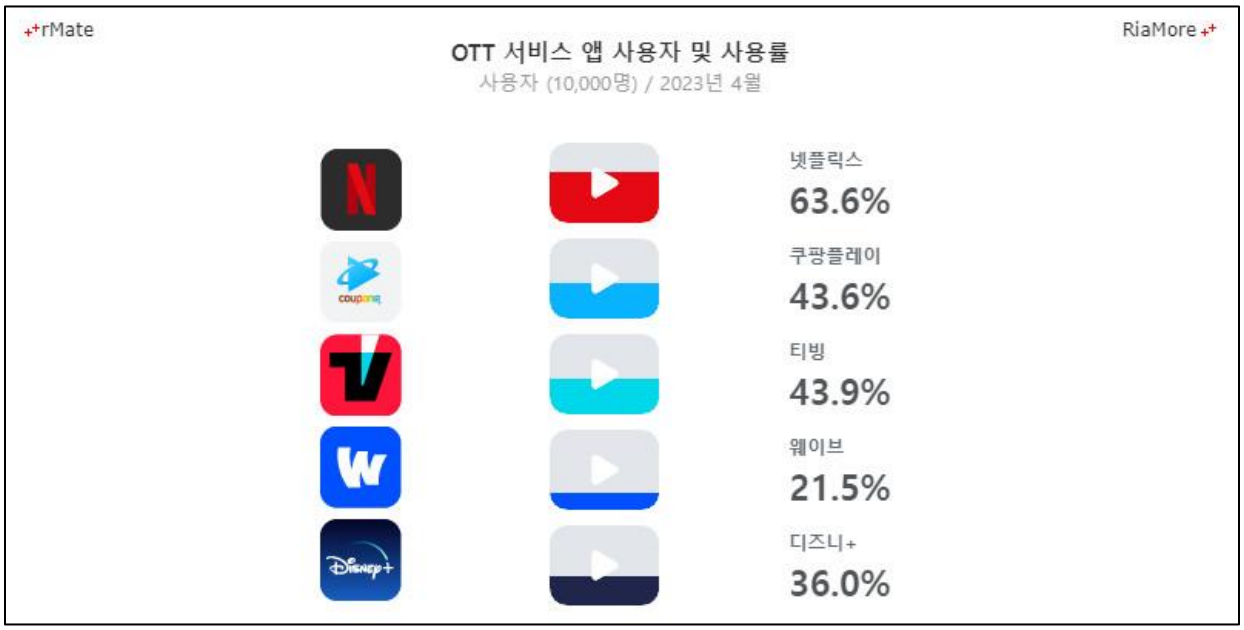


See the CodePen 알메이트 차트 – 픽토리얼 HFill 차트

PictorialVFillItemRenderer

다음은 itemRenderer 로 PictorialVFillItemRenderer 를 적용한 차트의 예제입니다. 데이터의 개수만큼 Item 을 세로로 추가하며, 각 Item 은 1 개의 Icon 을 가지고 있습니다. Icon 의 최대값은 <PictorialChart> 노드의 perIconValue 로 설정할 수 있습니다.

```
<PictorialChart perIconValue="100">
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_play.svg"
       itemRenderer="PictorialVFillItemRenderer">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </PictorialSeries>
  </series>
</PictorialChart>
```

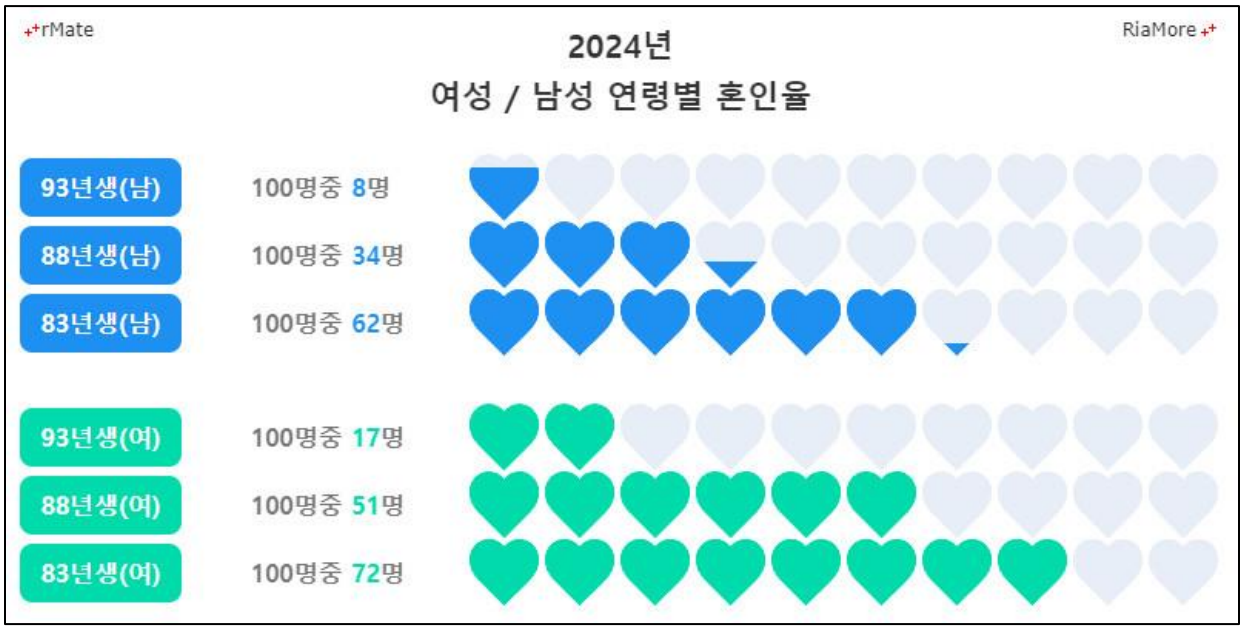


See the CodePen 알메이트 차트 – 픽토리얼 VFill 차트

PictorialBarItemRenderer

다음은 itemRenderer 로 PictorialBarItemRenderer 를 적용한 차트의 예제입니다. Bar 차트 모양의 ItemRenderer 로 값을 SVG 의 개수와 색상을 채워 표현합니다. Icon 의 최대값은 <PictorialChart> 노드의 perIconValue 로 설정할 수 있습니다. 각 Item 은 값에서 perIconValue 를 나눈 후 올림한 것 만큼의 개수을 가지고 있습니다. (예: data 105, perIconValue 10 일 경우 Icon 은 11 개이고 마지막 아이콘은 5/10 만큼의 색상을 채웁니다.)

```
<PictorialChart perIconValue="10">
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_heart.svg"
       itemRenderer="PictorialBarItemRenderer">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </PictorialSeries>
  </series>
</PictorialChart>
```

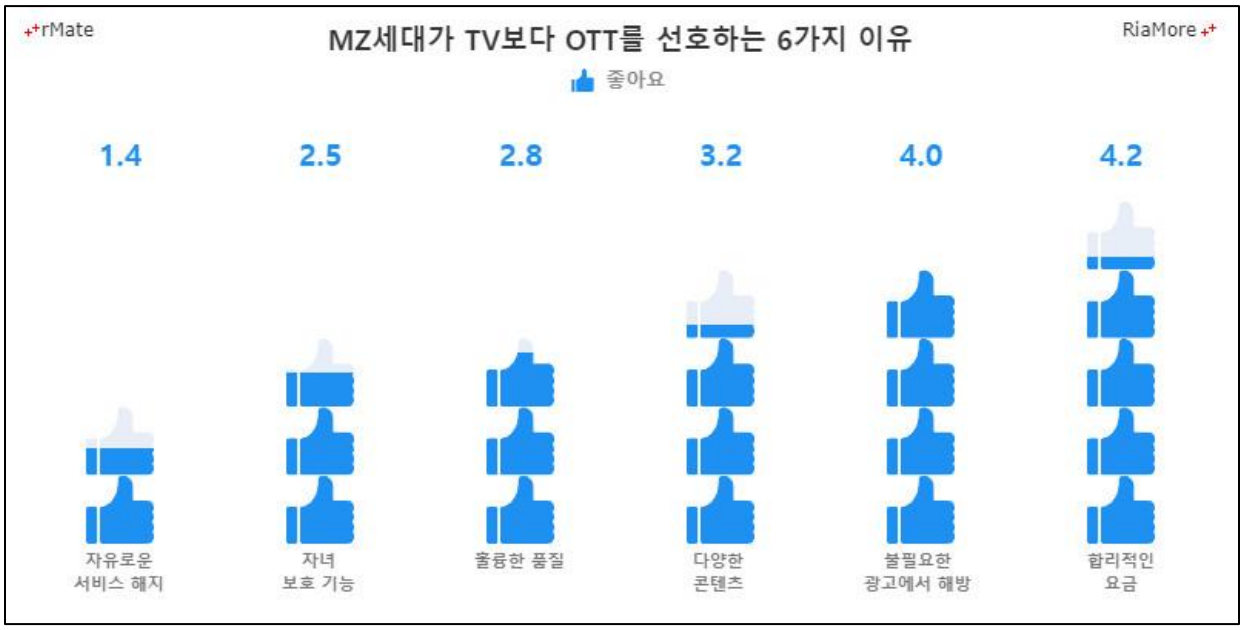


See the CodePen 알메이트 차트 – 픽토리얼 Bar 차트

PictorialColumnItemRenderer

다음은 itemRenderer 로 PictorialColumnItemRenderer 를 적용한 차트의 예제입니다. Column 차트 모양의 ItemRenderer 로 값을 SVG 의 개수와 색상을 채워 표현합니다. Icon 의 최대값은 <PictorialChart> 노드의 perIconValue 로 설정할 수 있습니다. 각 Item 은 값에서 perIconValue 를 나눈 후 올림한 것 만큼의 개수을 가지고 있습니다. (예: data 105, perIconValue 10 일 경우 Icon 은 11 개이고 마지막 아이콘은 5/10 만큼의 색상을 채웁니다.)

```
<PictorialChart perIconValue="1000">
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_like.svg"
       itemRenderer="PictorialBarItemRenderer">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </PictorialSeries>
  </series>
</PictorialChart>
```



See the CodePen 알메이트 차트 – 픽토리얼 Column 차트

PictorialCountItemRenderer

다음은 itemRenderer 로 PictorialCountItemRenderer 를 적용한 차트의 예제입니다. 좌측 상단부터 값을 SVG 의 개수와 색상을 채워 표현합니다. itemRows, itemColumns 로 행과 열의 개수를 지정하고, itemColumns 를 기준으로 줄바꿈하여 아랫줄에서 다시 SVG 를 표현합니다. Icon 의 최대값은 <PictorialChart> 노드의 perIconValue 로 설정할 수 있습니다. 각 Item 은 값에서 perIconValue 를 나눈 후 올림한 것 만큼의 개수을 가지고 있습니다. (예: data 105, perIconValue 10 일 경우 Icon 은 11 개이고 마지막 아이콘은 5/10 만큼의 색상을 채웁니다.)

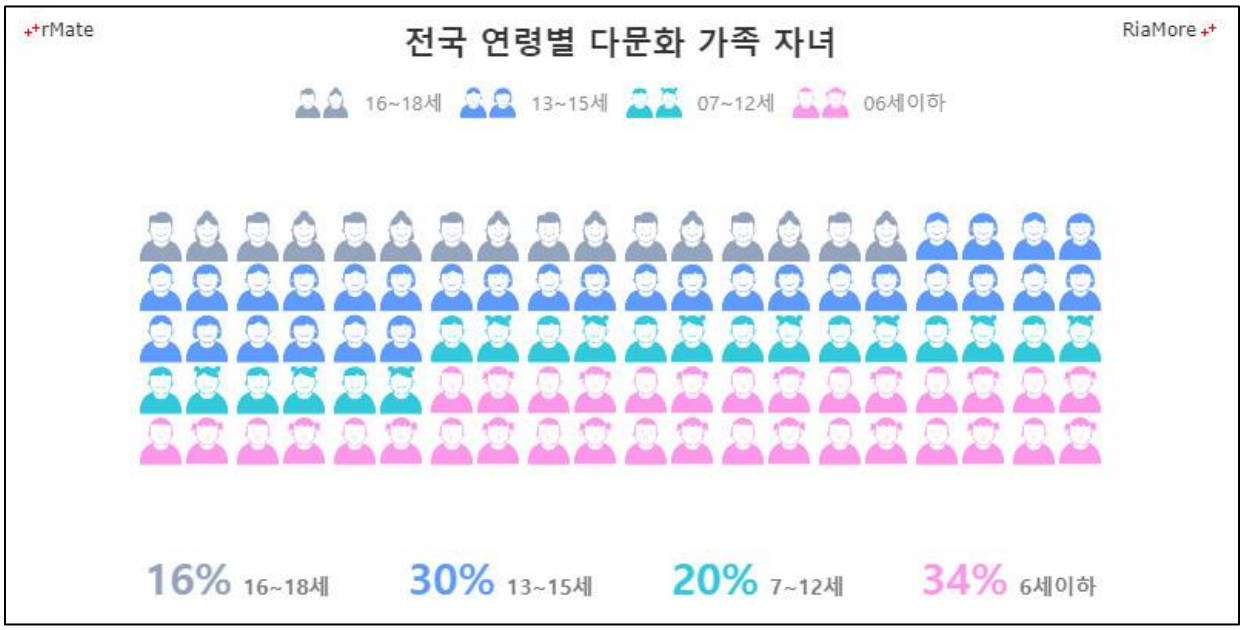
```
<PictorialChart perIconValue="88000" itemColumns="11">
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_kombucha.svg" itemRenderer="
        PictorialCountItemRenderer">
      <showDataEffect>
        <SeriesInterpolate/>
      </showDataEffect>
    </PictorialSeries>
  </series>
</PictorialChart>
```

See the CodePen 알메이트 차트 – 픽토리얼 Count 차트

PictorialCountStackItemRenderer

다음은 itemRenderer 로 PictorialCountStackItemRenderer 를 적용한 차트의 예제입니다. PictorialCountItemRenderer 와 동일하게 동작되나, 만약 모든 ItemRenderer 가 PictorialCountStackItemRenderer 라면, 모든 데이터가 이전 데이터에 이어서 그려집니다.

```
<PictorialChart perIconValue="2">
  ...
  <series>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_1618.svg" itemRenderer="
        PictorialCountStackItemRenderer"/>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_1315.svg" itemRenderer="
  PictorialCountStackItemRenderer"/>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_0712.svg" itemRenderer="
  PictorialCountStackItemRenderer"/>
    <PictorialSeries url="…/ rMateChartH5/Assets/Images/picto_0006.svg" itemRenderer="
  PictorialCountStackItemRenderer"/>
   </series>
</PictorialChart>
```



See the CodePen 알메이트 차트 – 픽토리얼 CountStack 차트

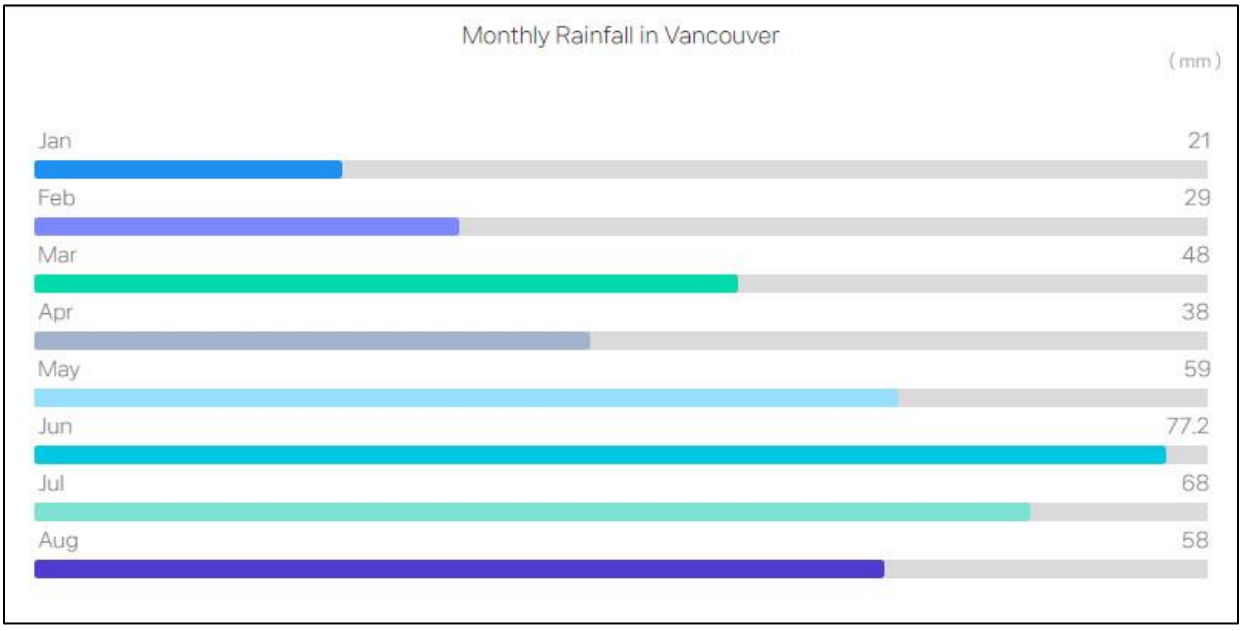
4.38 오버레이바 차트

기본적으로 바 차트와 동일하나, 바 두개를 겹쳐서 표현된 차트입니다. 뒤쪽 바는 항상 100% 비율로 표시되고, 색상과 굵기를 변경할 수 있습니다. 그리고 라벨이 양쪽 끝에 출력되는게 특징입니다.

클러스터 오버레이바 차트

다음은 오버레이바 차트를 표시하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<OverlayBar2DChart>
  ...
  <series>
    <OverlayBar2DSeries xField="Rainfall">
      <backgroundFill>
        <SolidColor color="#dbdbdb"/>
      </backgroundFill>
    </OverlayBar2DSeries>
  </series>
</OverlayBar2DChart>
```



See the CodePen 알메이트 차트 – 오버레이바 차트

5. 차트 디자인과 스타일링

5.1 테마 적용하기

한 프로젝트에서 많고 다양한 종류의 차트를 개발할 경우 각 차트의 일관된 디자인 스타일을 유지하면서 개발자들이 디자인적인 요소에 시간을 빼앗기지 않도록 해주는 것이 테마(Theme)입니다. 알메이트 차트에서는 기본적으로 6 가지 테마(simple, cyber, modern, lovely, pastel, old)를 제품과 함께 제공하고 사용자가 원하는 테마를 직접 만들어서 사용할 수도 있습니다. 제공되는 테마 중에서 old 테마는 알메이트 차트 버전 2.5 이하에서 기본적으로 적용되는 디자인 테마입니다.

테마 적용 방법

알메이트 차트에서 기본으로 제공하는 6 가지 테마를 사용하려면 다음과 같이 theme.js 파일을 포함(include)시킵니다. 이 때 반드시 theme.js 파일은 차트 라이브러리(rMateChartH5.js) 보다 나중에 선언되어야 합니다.

```
<head>
  <link rel="stylesheet" type="text/css"
       href="../rMateChartH5/Assets/Css/rMateChartH5.css"/>
  <script type="text/javascript"
       src="../LicenseKey/rMateChartH5License.js"></script>
  <script type="text/javascript" src="../rMateChartH5/JS/rMateChartH5.js"></script>
  <script type="text/javascript"
       src="../rMateChartH5/Assets/Theme/theme.js"></script>
...
</head>
```

다음은 theme.js 파일을 포함(include)한 이 후 simple 테마를 적용하는 자바스크립트 문장을 실행한 것입니다.

```
<script>
  ...
  document.getElementById("chart1").setTheme("simple");
  ...
</script>
```

테마가 한번 적용된 후, 기본(default) 테마를 다시 적용하기를 원하면 다음 자바스크립트 문장을 실행합니다.

```
<script>
  ...
  document.getElementById("chart1").setTheme("default");
  ...
</script>
```

기본으로 제공되는 6 가지 테마 중에서 특정 테마를 삭제할 수 있습니다. 예를 들어 simple 테마를 삭제할 경우, simple 테마가 적용된 모든 차트는 기본(default) 테마로 돌아옵니다. 이 때 만약 simple 테마가 적용된 차트가 없다면 기본으로 적용되는 6 가지 테마 중에서 simple 테마는 삭제되게 됩니다. 다음은 simple 테마를 삭제하는 자바스크립트 문장을 실행한 것입니다.

```
<script>
  ...
  rMateChartH5.removeTheme("simple");
  ...
</script>
```

사용자 정의 테마

사용자가 원하는 테마를 직접 등록하고 이를 적용할 수 있습니다. 다음은 MyTheme.js 파일에 myTheme 이라는 테마를 생성하고 이를 적용하는 예제입니다.

```
(function() {
  var myTheme
  function themeExtend(target, obj) {
    for (var o in obj) {
      if(!target[o])
      target[o] = obj[o];
    }
  }
  function intoProp(prop, theme, colors, alpha, weight, gradient) {
    var i, n, o, color;
    if (alpha == null || alpha == undefined)
      alpha = 1;
    if (weight == null || weight == undefined)
      weight = 1;
    if (gradient == null || gradient == undefined)
      gradient = false;
    for (i = 0, n = colors.length ; i < n ; i += 1) {
      color = colors[i % colors.length];
      if (!gradient) {
        o = { color : color, weight : weight, alpha : alpha};
      } else {
        o = {};
        o.angle = 90;
        o.entries = [];
        o.entries.push({ color : color, ratio : 1, alpha : alpha});
```

```
      }
      if (!theme.series[i])
        theme.series[i] = {};
      theme.series[i][prop] = o;
    }
  }

  rMateChartH5.themes = {};
  rMateChartH5.setupTheme = {};
  rMateChartH5.setupTheme.myTheme = {
    defaultColors : [
"#3BA0CC","#775F47","#A8C545","#DDCFAC","#777777","#9FB5CD","#2AA2A0","#37C07C","#3
       F596E","#7E92A5","#A48D70","#77B6D0","#CEE4ED","#555555",
"#B195A6","#D4C4CE","#67425A","#B9DCDA","#3798B9","#89D5AF","#869726","#C1CC89","#8
       36E2C","#C5BA8E","#816D5B","#DEE3E7","#84929F","#3B4B5B"
    ],
    chartCommon : {
      dataTipBackgroundColorOnSeries : false,
      dataTipBorderColor : "#323232"
    },
    barChart : {
      backgroundElements : [{
        direction : "vertical",
        verticalStroke : {"color":"#ececec"}
      }],
    }
  };

  myTheme = rMateChartH5.setupTheme.myTheme;
  themeExtend(myTheme.barChart, myTheme.chartCommon);

  rMateChartH5.setupTheme.myTheme = {
    defaultColors : myTheme.defaultColors,
    Column2DChart : myTheme.chartCommon,
    Pie2DChart : myTheme.chartCommon,
    Plot2DChart : myTheme.chartCommon,
    Bar2DChart : myTheme.barChart
  };
})();
```

위와 같이 MyTheme.js 파일을 작성한 다음, 아래와 같이 이를 포함(include)하여 myTheme 테마를 적용할 수 있습니다.

```
<head>
  <link rel="stylesheet" type="text/css"
       href="../rMateChartH5/Assets/Css/rMateChartH5.css"/>
  <script type="text/javascript"
       src="../LicenseKey/rMateChartH5License.js"></script>
  <script type="text/javascript" src="../rMateChartH5/JS/rMateChartH5.js"></script>
  <script type="text/javascript" src="./MyTheme.js"></script>
  ...
  <script>
    ...
    document.getElementById("chart1").setTheme("myTheme");
    ...
  </script>
</head>
```

5.2 색

차트의 다양한 요소들에 대한 색을 설정하는 방법은 해당 요소의 색 속성에 16 진수 컬러 코드와 필요한 속성값을 지정하는 것입니다. 알메이트 차트에서의 색 설정은 다음과 같이 크게 4 가지로 구분할 수 있습니다.

- 선에 대한 색 설정
- 영역에 대한 색 설정
- 영역에 대한 선형(Linear) 그라디언트 색 설정
- 영역에 대한 방사형(Radial)그라디언트 색 설정

기본색 코드

사용자에 의해서 차트 요소의 색이 직접 지정되지 않으면 알메이트 차트는 시스템에 설정된 기본색을 이용합니다. 기본색은 버전에 따라서 다를 수 있으며, 알메이트 차트 버전 7.0 에서 사용되는 기본색은 아래의 표와 같습니다. 사용자가 색을 지정하지 않을 경우, 아래 표에 나와있는 색 코드가 차트를 생성할 때 순서대로 적용됩니다.

기본색	라인 차트 기본색	컬럼 차트 기본색	파이 차트 기본색	매트릭스 차트 기본색
ff812d	03a9f5	41b2e6	40b1e6	5587a2
ffd100	fabc05	4452a8	f97b17	074d81
81d733	8e5d9b	fabc05	fabb05	20cbc2
666666	00b6aa	5587a2	20cbc2	f6a54c
43cbff	88b14b	69cabc	074d81	d1af94
3284c3	8a7860	6c4a85		96d4df
d4155b	d23579	8a7860		
f71075	96d4df	96d4df		
a48d70	57646d	f7a33b		

15b671	d09d5c	d8c6aa		
00998e	9896a4	49588b		
ce4acc	f79d2e	68c0c9		
9a0197	55b48b	476f80		
aa94de	3991a7	cc3a5f		
8d65d2	d1af94	ffc853		
e44747	dd4760	4d4a69		
f99b2b	615c60	b9d7d7		
fde145	5587a2	efc471		
00c574	54bae9	eb8952		
4c9cff	ffc853	439896		
005d6e	e17034	8ab840		
ada59d	888888	8a8a8a		
88827b	53bb9d	634265		
584737	765e8b	f38072		
36ccda				
2aa8b4				
8647a7				
ba7dda				

주의

기본 색 코드는 테마를 적용하여 사용자가 원하는 색으로 변경할 수 있습니다. 테마의 사용법에 관해서는 테마 적용하기를 참조하십시요.

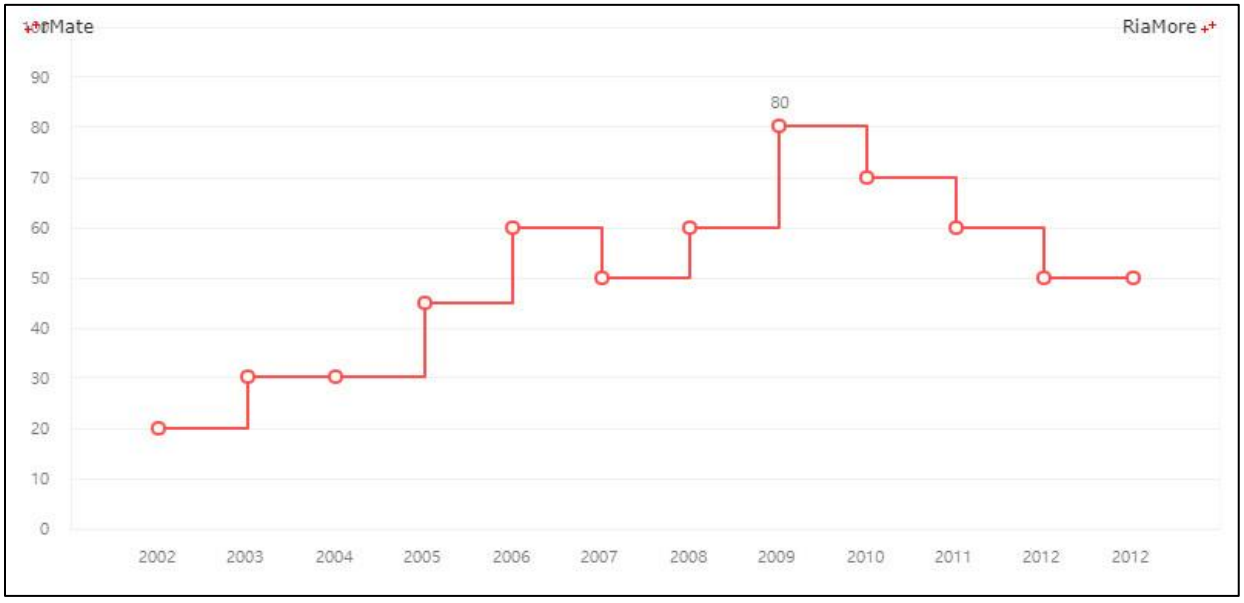
선에 대한 색 설정하기

선에 대한 색을 설정하는 방법은 해당 차트 요소의 선 속성에 <Stroke> 노드를 정의하는 것입니다. <Stroke> 노드의 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
color	#16 진수 컬러 코드 표기 기본값: #000000	선의 색상을 지정합니다.
alpha	0 과 1(*) 사이의 숫자	선 색상의 투명도를 지정합니다.
id	텍스트	<Stroke> 객체의 식별자를 지정합니다.
weight	1(*) 이상의 숫자	선의 굵기를 지정합니다.

다음은 <Stroke> 노드를 라인 차트의 <lineStroke> 속성에 정의하여 라인 차트의 선을 설정한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<lineStroke>
  <Stroke color="#FF5050" weight="2"/>
</lineStroke>
```



See the CodePen 알메이트 차트 - 선에 대한 색 설정하기

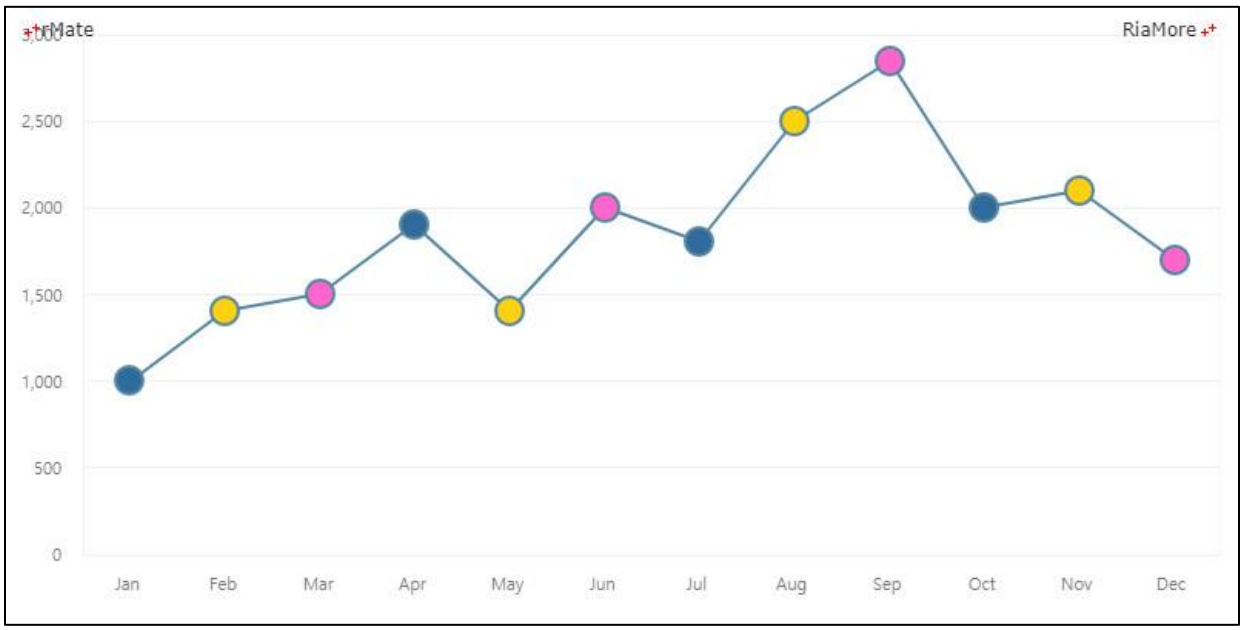
영역에 대한 색 설정하기

영역에 대한 색을 설정하는 방법은 해당 차트 요소의 영역 속성에 <SolidColor> 노드를 정의하는 것입니다. <SolidColor> 노드의 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
color	#16 진수 컬러 코드 표기 기본값: #000000	영역의 색상을 지정합니다.
alpha	0 과 1(*) 사이의 숫자	영역 색상의 투명도를 지정합니다.
id	텍스트	<SolidColor> 객체의 식별자를 지정합니다.

다음은 <SolidColor> 노드를 라인 차트의 <fills> 속성에 정의하여 라인 차트의 데이터 아이템 렌더러(CircleItemRenderer)의 영역에 색을 채우는 코드와 이를 적용해서 출력한 차트의 예제입니다. 이 예제에서는 전체 12 개의 데이터 아이템에 대한 색 채우기를 위해 3 가지 종류의 색이 번갈아 가면서 사용됩니다.

```
<Line2DSeries yField="Profit" itemRenderer="CircleItemRenderer">
  <fills>
    <SolidColor color="#316a9d"/>
    <SolidColor color="#ffd110"/>
    <SolidColor color="#ff66cc"/>
  </fills>
...
</Line2DSeries>
```



See the CodePen 알메이트 차트 - 영역에 대한 색 설정하기

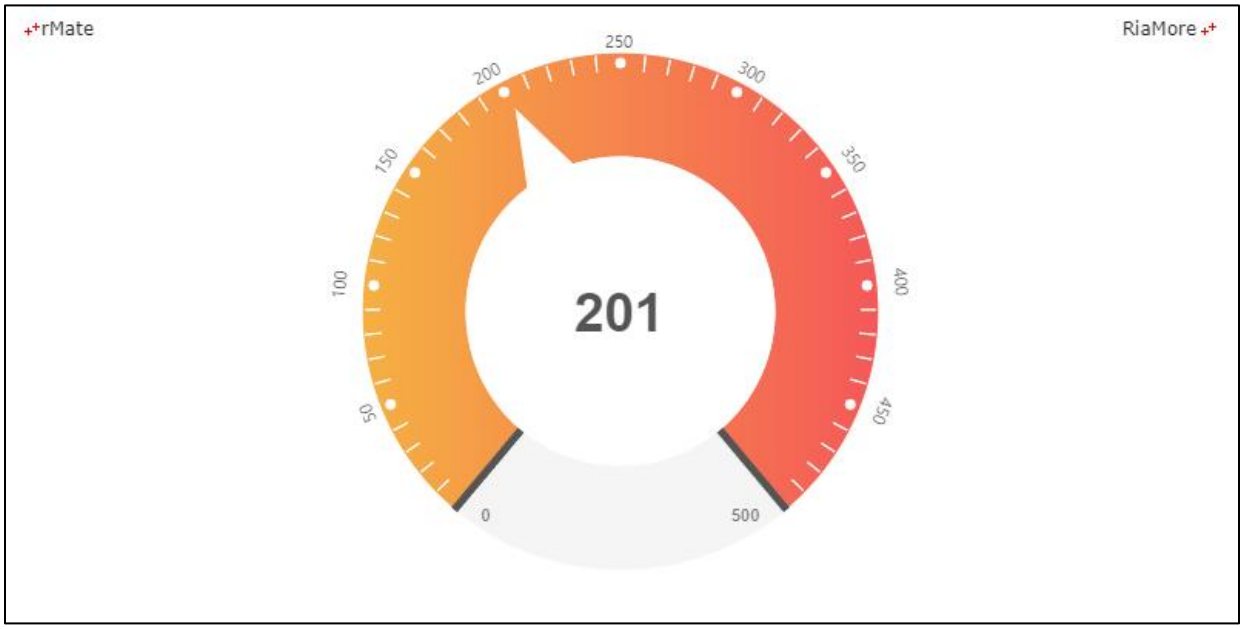
영역에 대한 선형(Linear) 그라디언트 색 설정하기

영역에 색을 채우기 위해 선형(Linear) 그라디언트를 적용할 수 있습니다. 선형 그라디언트가 적용 가능한 차트의 요소는 <AxisRange> 와 <CircularGauge> 입니다. <LinearGradient> 노드의 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
entries	<GradientEntry>	그라디언트 요소를 지정합니다.
angle	0(*) 과 360 사이의 숫자	그라디언트 효과의 방향을 지정합니다.
id	텍스트	<LinearGradient> 객체의 식별자를 지정합니다.

다음은 원형 게이지의 프레임에 선형 그라디언트 색을 설정하는 코드와 이를 적용해서 출력한 차트의 예입니다.

```
<frameFill>
  <LinearGradient angle="0">
    <entries>
      <GradientEntry color="#f6af43" ratio="0"/>
      <GradientEntry color="#f55a58" ratio="1"/>
    </entries>
  </LinearGradient>
</frameFill>
```



See the CodePen 알메이트 차트 - 영역에 대한 선형(Linear) 그라디언트 색 설정하기

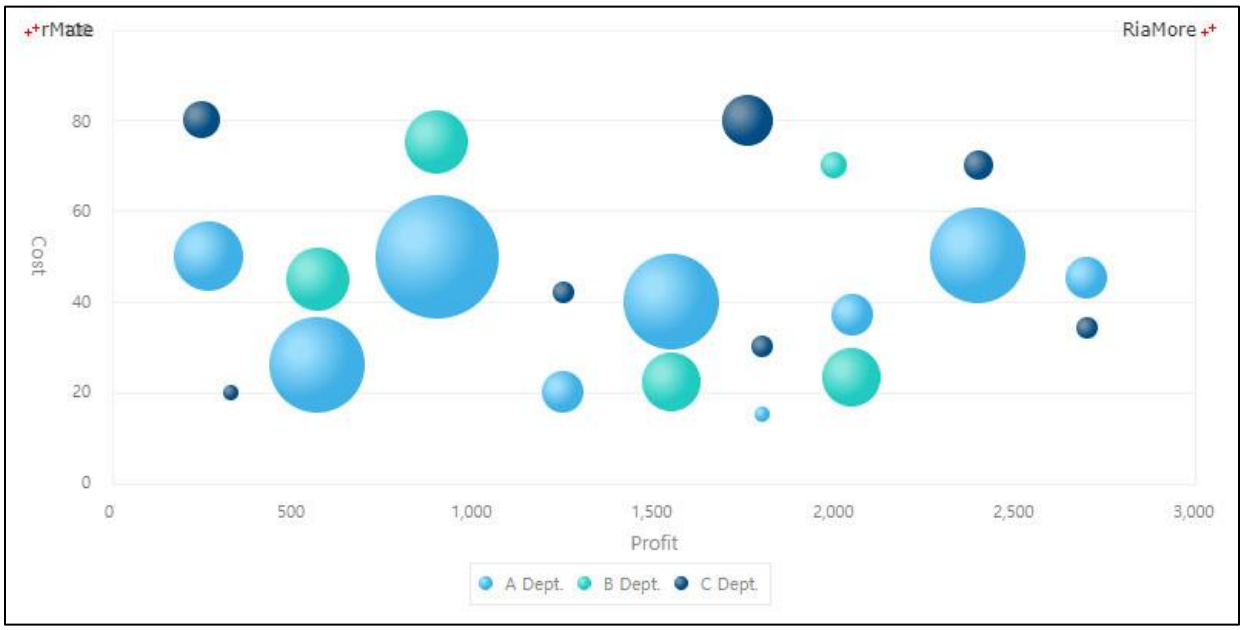
영역에 대한 방사형(Radial) 그라디언트 색 설정하기

영역에 색을 채우기 위해 방사형(Radial) 그라디언트를 적용할 수 있습니다. 방사형 그라디언트가 적용 가능한 차트의 요소는 <AxisRange> 와 <CircularGauge> 입니다. <RadialGradient> 노드의 속성은 다음과 같습니다.

속성명	유효값 (*: 기본값)	설명
entries	<GradientEntry>	그라디언트 요소를 지정합니다.
angle	0(*) 과 360 사이의 숫자	중심에서 그라디언트 초점까지의 상대 위치를 지정합니다.

focalPointRatio	-1 과 1 사이의 숫자 기본값: 0	중심에서 그라디언트 초점까지의 거리를 지정합니다. 값이 양수이면 중심에서 오른쪽, 값이 음수이면 중심에서 왼쪽에 그라디언트 초점이 존재하게됩니다. 예를 들어, 값이 “0.25” 이면 중심에서 오른쪽에 그라디언트의 초점이 생성되고, 만약 값이 “-0.25” 이면 중심에서 왼쪽에 그라디언트의 초점이 생성됩니다.
id	텍스트	<RadialGradient> 객체의 식별자를 지정합니다.

다음은 버블 차트의 버블에 방사형 그라디언트 색을 설정하는 코드와 이를 적용해서 출력한 차트의 예입니다.



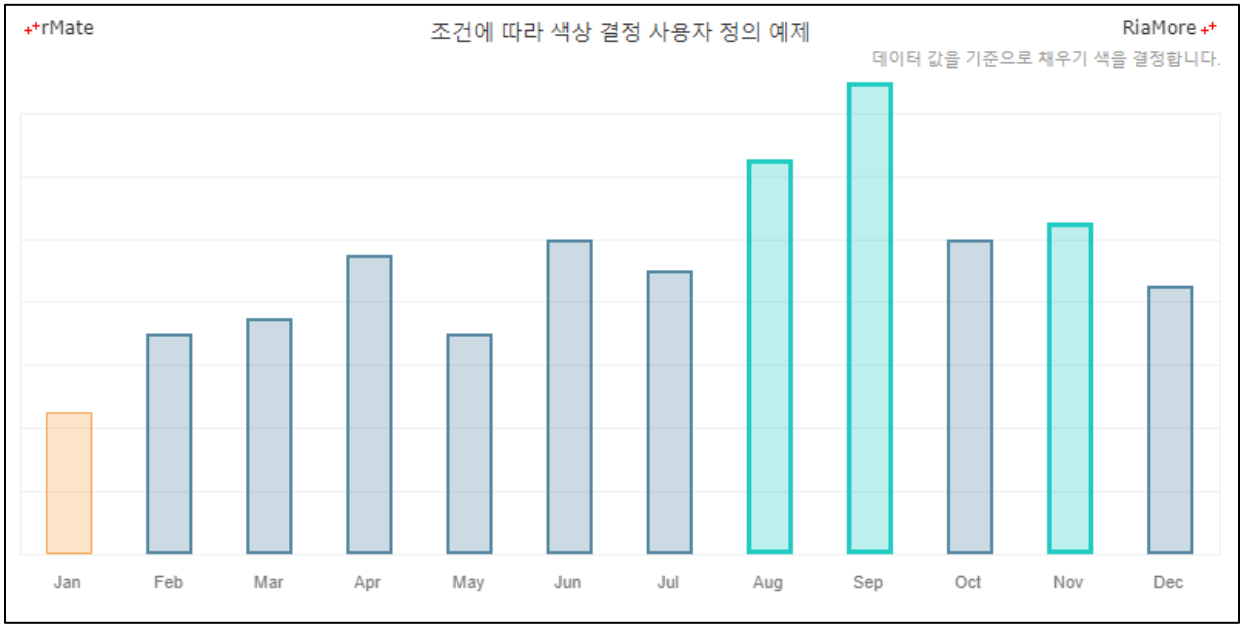
See the CodePen 알메이트 차트 - 영역에 대한 방사형(Radial) 그라디언트 색 설정하기

사용자 정의 색 지정

데이터 아이템에 대한 채우기 및 테두리 선을 데이터 값에 따라서 사용자가 원하는 색으로 지정할 수 있습니다. 데이터 아이템의 채우기 색에 대한 사용자 정의 함수명은 fillJsFunction 속성에, 데이터

아이템의 테두리 선의 색에 대한 사용자 정의 함수명은 strokeJsFunction 속성에 설정합니다. 다음은 컬럼 차트에서 막대의 채우기 색과 테두리 색을 데이터 값에 따라서 다르게 표시하는 예제입니다.

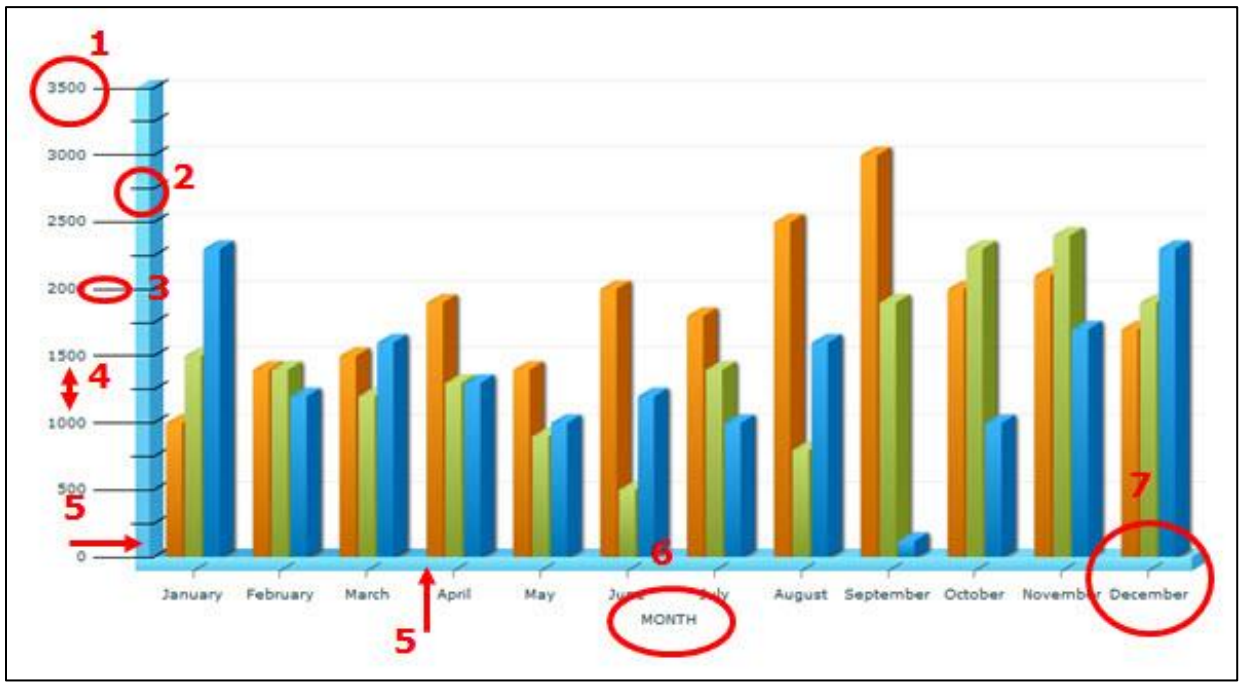
```
<Column2DSeries yField="Profit" fillJsFunction="fillJsFunc"
        strokeJsFunction="strokeJsFunc">

function fillJsFunc(seriesId, index, data, values) {
  if(values[1] > 2000)
    return {"color":"#20cbc2", "alpha":0.3};
  else if(values[1] > 1000)
    return {"color":"#5587a2", "alpha":0.3};
  else
    return {"color":"#f6a54c", "alpha":0.3};
}

function strokeJsFunc(seriesId, index, data, values) {
  if(values[1] > 2000)
    return {"color":"#20cbc2", "weight":3};
  else if(values[1] > 1000)
    return {"color":"#5587a2", "weight":2};
  else
    return {"color":"#f6a54c", "weight":1};
}
```



See the CodePen 알메이트 차트 - 사용자 정의 색 지정

5.3 축 스타일링과 축 레이블

다음 그림은 축을 구성하는 요소들을 표시한 것입니다. 각 구성 요소에는 번호가 표시되어 있는데 이에 대한 명칭과 설명은 아래 표와 같습니다.



번호	구성 요소에 대한 명칭	설명
1	축 레이블	축의 틱(눈금)에 표시되는 숫자 (<LinearAxis>, <ColorAxis>), 날짜(<DateTimeAxis>) 혹은 문자(<CategoryAxis>)입니다.
2	보조 틱	축의 보조 눈금입니다.
3	틱	축의 눈금입니다.
4	축 숫자 간격	숫자 레이블이 표시될 때 이웃하는 숫자 사이의 간격(그림상에서 숫자 사이의 간격은 500)입니다.
5	축 선 스타일	축 선의 스타일입니다. 그림상에서는 3D 축이 표현되었습니다.
6	축 제목	축의 제목(타이틀) 입니다. 그림상에서는 수평축(X 축)에만 표시되어 있고, 수직축(Y 축)에는 생략되었습니다.

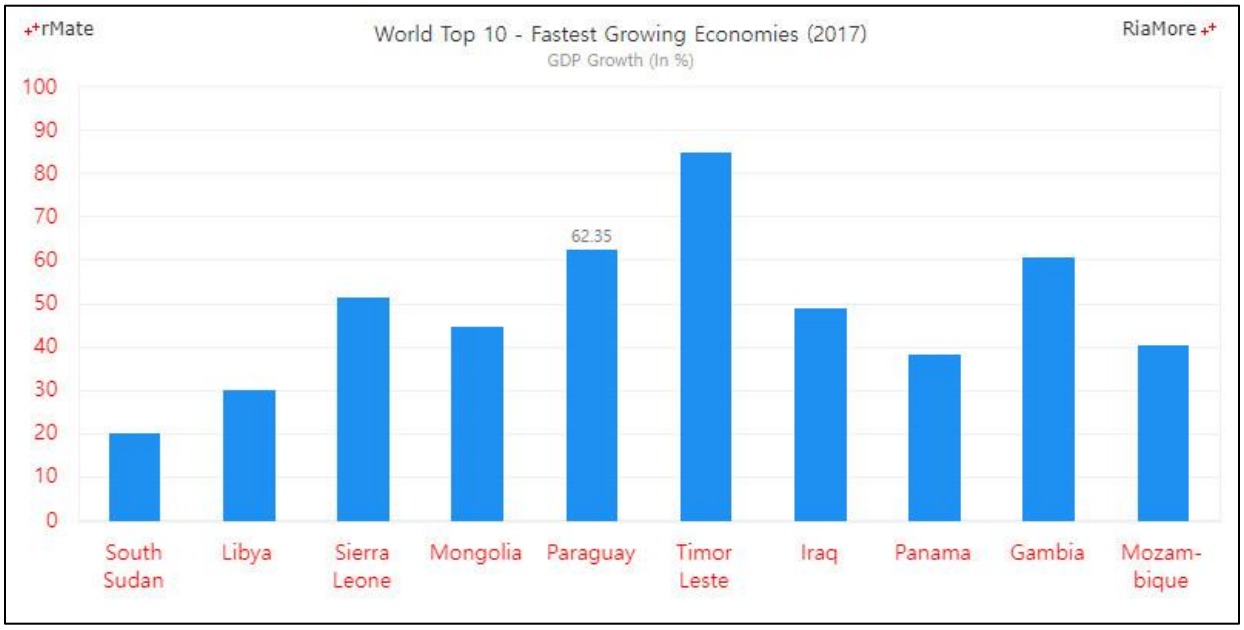
7	틱에 대한 레이블의 위치	그림상에는 레이블이 틱 아래에 위치합니다. 레이블을 틱과 틱 사이에 표시할 수도 있습니다.

축 레이블에 폰트와 스타일 설정하기

축 레이블에 대한 폰트, 스타일 등에 대한 설정은 2D 축일 경우 <Axis2DRenderer> 노드, 3D 축일 경우 <Axis3DRenderer> 노드를 정의함으로써 가능합니다. 작업할 노드의 styleName 속성값에 스타일명을 지정하고, 지정된 스타일명으로 CSS 스타일을 작성하여 <Style> 노드에 정의합니다.

다음은 X 축 레이블의 폰트 크기를 14px, 색상을 붉은색(#ff0000)으로 설정하는 코드와 이를 적용해서 출력한 차트의 예입니다. 이 때 주의할 점은 <Style> 노드에 정의되는 스타일명은 반드시 마침표(.)로 시작해야 한다는 것입니다.

```
<horizontalAxis>
  <CategoryAxis id="hAxis" categoryField="Country"/>
</horizontalAxis>
...
<horizontalAxisRenderers>
  <Axis2DRenderer axis="{hAxis}" styleName="axisLabel" showLine="true"/>
</horizontalAxisRenderers>
...
<Style>.axisLabel{fontSize:14px;color:#ff0000;}
</Style>
```

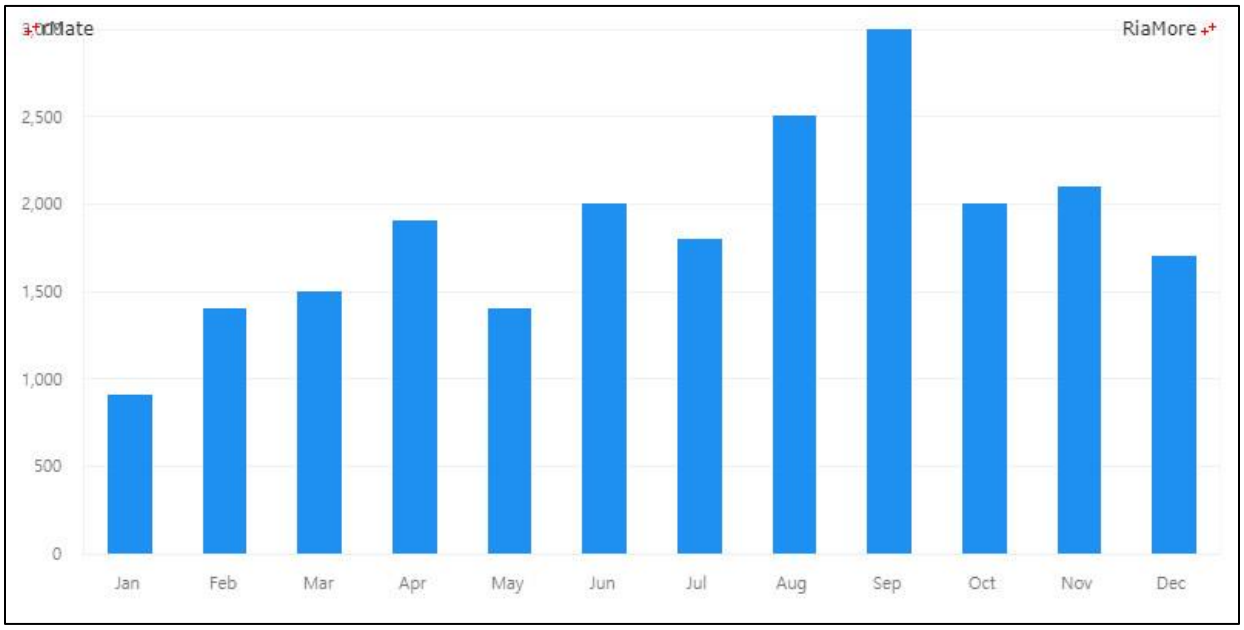


See the CodePen 알메이트 차트 - 축 레이블에 폰트와 스타일 설정하기

축 레이블에 포맷터 적용하기

포맷터를 적용해서 축 레이블을 표현할 수 있습니다. 포맷터에 대한 자세한 사용법은 포맷터 사용하기를 참조하십시요. 축 레이블에는 알메이트 차트가 지원하는 3 가지 포맷터 (숫자 포맷터, 통화 포맷터, 날짜 포맷터)를 모두를 적용할 수 있습니다. 다음은 천단위 콤마를 Y 축 레이블에 표시하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<NumberFormatter id="numfmt" useThousandsSeparator="true"/>
...
<verticalAxis>
  <LinearAxis interval="500" formatter="{numfmt}"/>
</verticalAxis>
```

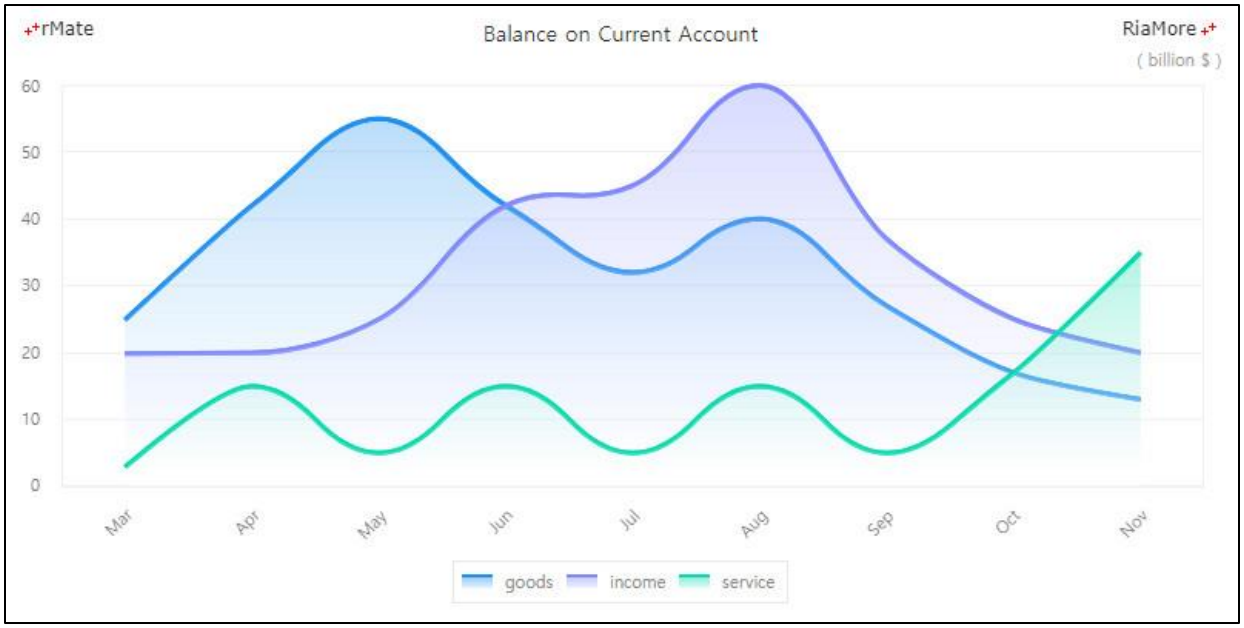


See the CodePen 알메이트 차트 - 축 레이블에 포맷터 적용하기

축 레이블 회전하기

<Style> 노드에 축 레이블의 CSS 스타일을 설정하여 축 레이블을 회전하여 표시할 수 있습니다. 다음은 X 축 레이블을 45 도 회전하여 표시하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<horizontalAxisRenderers>
  <Axis2DRenderer axis="{hAxis}" styleName="xAxisStyle" showLine="true"/>
</horizontalAxisRenderers>
...
<Style>.xAxisStyle {fontFamily:"Malgun Gothic";fontSize:11;labelRotation:45;}
</Style>
```



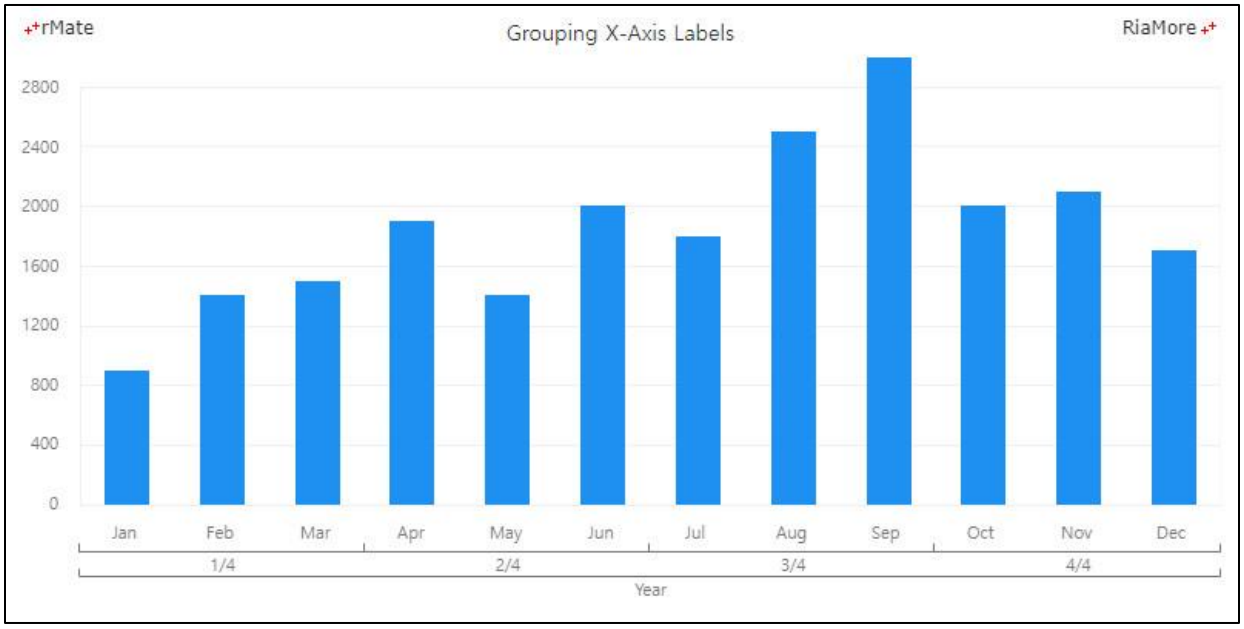
See the CodePen 알메이트 차트 - 축 레이블 회전하기

그룹 레이블 표현하기

카테고리 축에 표시되는 레이블을 그루핑하여 그룹명을 함께 표시할 수 있습니다. 예를 들어, 월에 대해서는 이를 분기로 분기에 대해서는 이를 년으로 그루핑할 수 있습니다. 그룹 레이블을 차트에 표현하기 위해서는 차트 데이터 객체 정의 시에 그루핑 필드(필드명은 <CategoryAxis> 노드의 groupCategoryField 속성에 설정되는 값)와 아이템 필드(필드명은 items)를 설정해야 합니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다. groupCategoryField 속성의

```
<horizontalAxis>
  <CategoryAxis categoryField="Month" groupCategoryField="groupCategory"/>
</horizontalAxis>

var chartData = [{
  "groupCategory" : "Year",
  "items": [{
    "groupCategory" : "1/4",
      "items" : [{"Month":"Jan","Profit":900},
        ...,
```

기본값은 “groupCategory” 이며, groupCategoryField 속성 설정이 생략되면
자동으로 “groupCategory” 으로 설정됩니다.

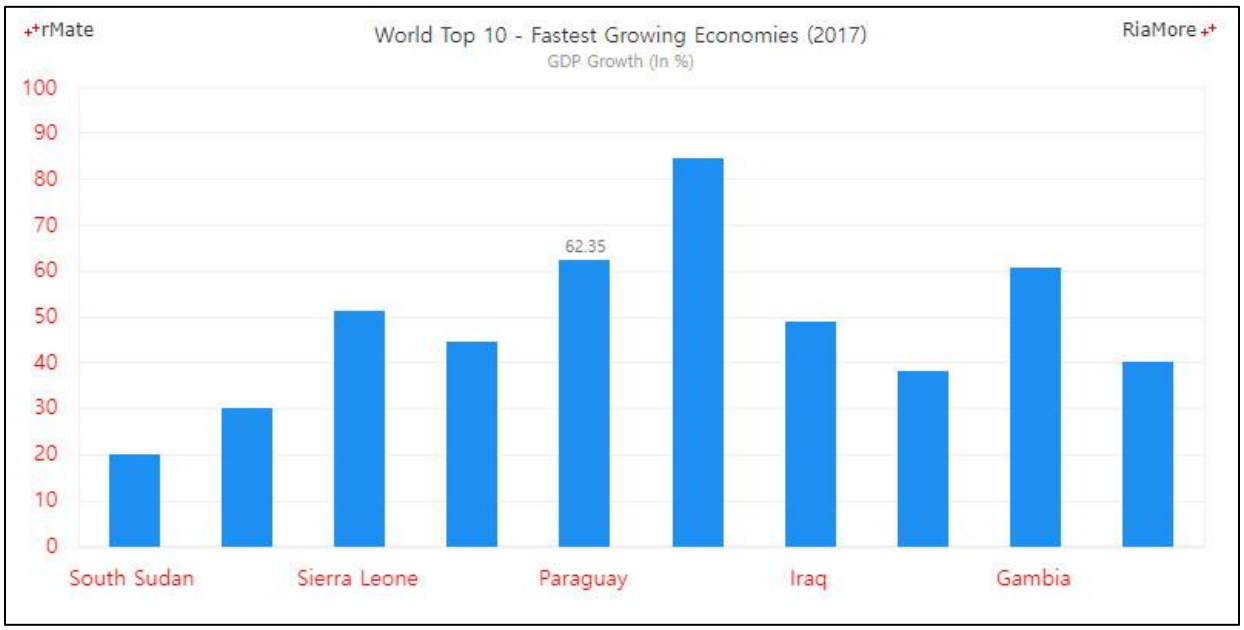


See the CodePen 알메이트 차트 - 그룹 레이블 표현하기

축 레이블 겹침

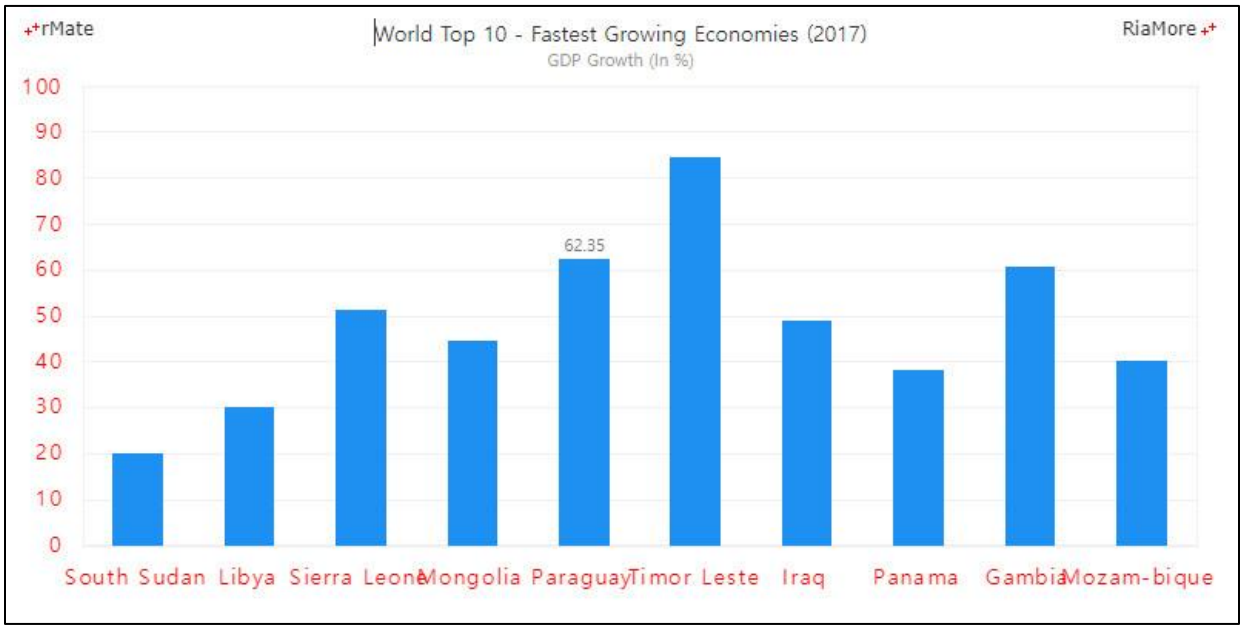
알메이트 차트는 모든 축 레이블을 표시하기에 충분한 공간이 없을 경우, 일부 레이블을 생략하여 겹치는 문제를 자동으로 해결합니다. 다음은 X 축 레이블이 겹치는 문제를 없애기 위해서 최초 레이블(South Sudan)이 표시된 후 두번째 레이블부터는 번갈아 가면서 생략되는 예를 보여줍니다.

```
var chartData = [
  {"Country":"South Sudan", "GDP":20},
  {"Country":"Libya", "GDP":30},
  {"Country":"Sierra Leone", "GDP":51.2},
  {"Country":"Mongolia", "GDP":44.5},
  {"Country":"Paraguay", "GDP":62.35},
  {"Country":"Timor Leste", "GDP":84.46},
  {"Country":"Iraq", "GDP":48.9},
  {"Country":"Panama", "GDP":38},
  {"Country":"Gambia", "GDP":60.5},
  {"Country":"Mozambique", "GDP":40.1}
];
```



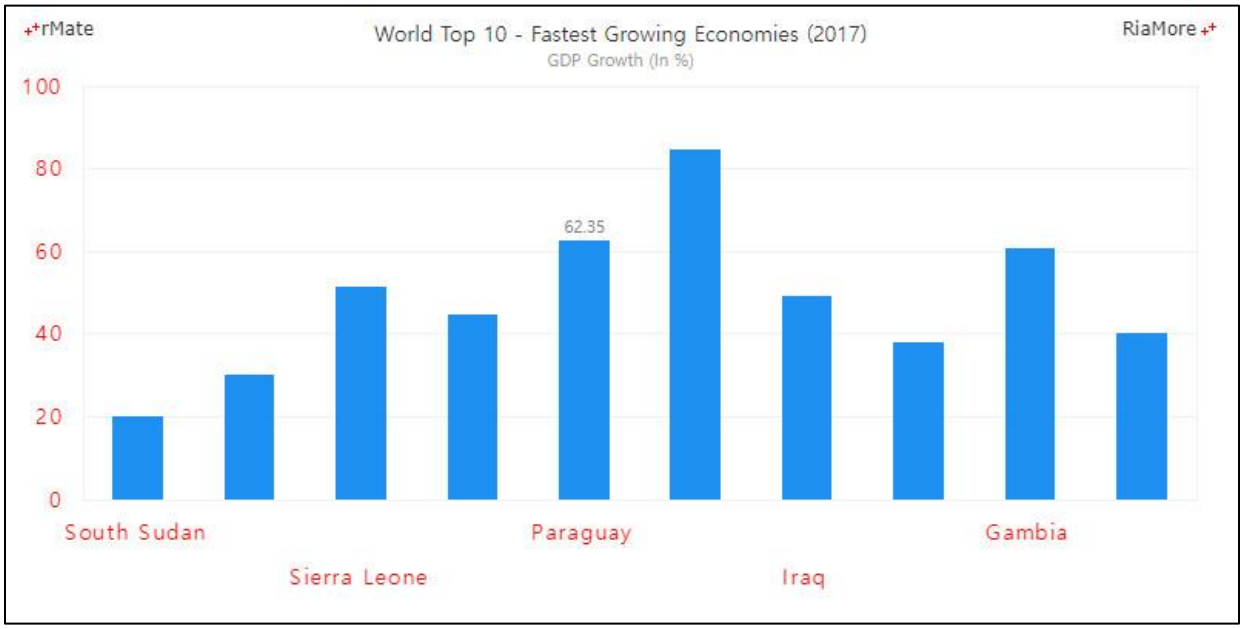
만약 레이블이 겹치는 것을 무시하고 모든 레이블을 차트에 표시하고 싶으면 다음과 같이 <Axis2DRenderer> (혹은 <Axis3DRenderer>) 노드의 canDropLabels 속성값을 “false” (기본값: “true”)로 설정하면 됩니다.

```
<horizontalAxisRenderers>
  <Axis2DRenderer axis="{hAxis}" styleName="axisLabel" showLine="true"
       canDropLabels="false" />
</horizontalAxisRenderers>
```



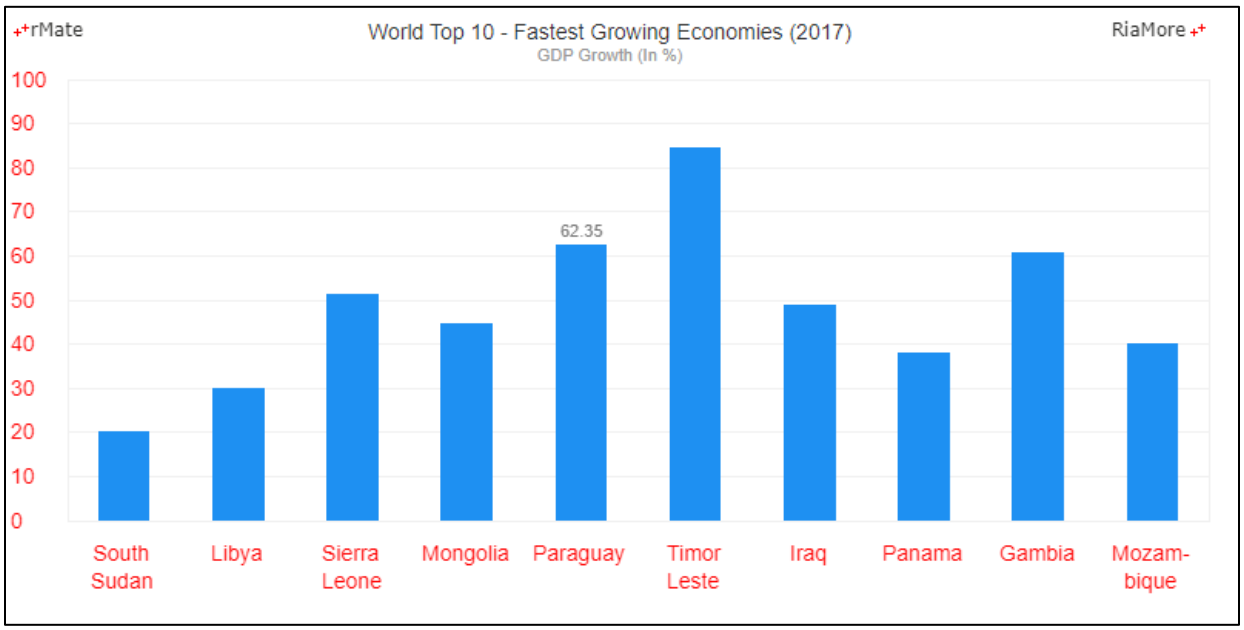
레이블이 겹치지 않고 모두 표시되기를 원할 경우, canStagger 속성값을 “true” (기본값: “false”)로 설정할 수 있습니다. canStagger 속성값이 “true” 로 설정되면 축 레이블이 두 단계로 표현되어 거의 대부분의 레이블이 안겹친 상태로 표시될 수 있습니다. 이에 대한 결과는 아래와 같습니다.

```
<horizontalAxisRenderers>
  <Axis2DRenderer axis="{hAxis}" styleName="axisLabel" showLine="true"
       canDropLabels="false" canStagger="true" />
</horizontalAxisRenderers>
```



canStagger 속성을 이용하지 않고 전체 레이블을 안 겹치게 표현하는 또 다른 방법은 데이터 설정 시에
 태그를 텍스트에 삽입하는 것입니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

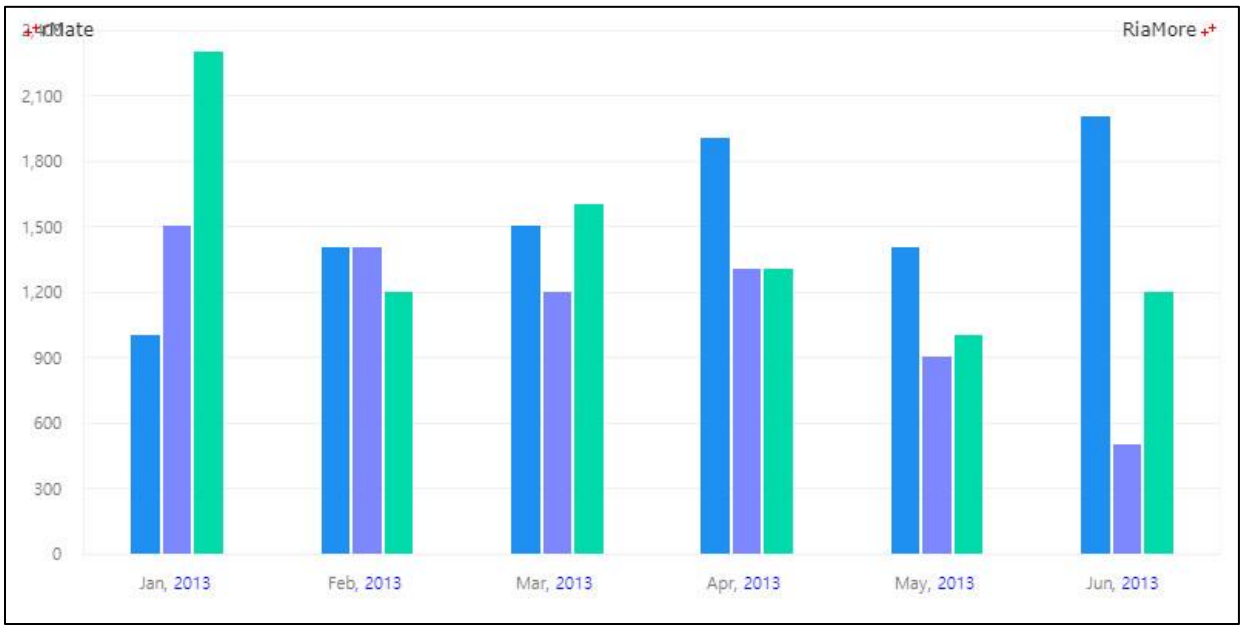
```
var chartData = [
  {"Country":"South<br>Sudan", "GDP":20},
  {"Country":"Libya", "GDP":30},
  {"Country":"Sierra<br>Leone", "GDP":51.2},
  {"Country":"Mongolia", "GDP":44.5},
  {"Country":"Paraguay", "GDP":62.35},
  {"Country":"Timor<br>Leste", "GDP":84.46},
  {"Country":"Iraq", "GDP":48.9},
  {"Country":"Panama", "GDP":38},
  {"Country":"Gambia", "GDP":60.5},
  {"Country":"Mozam-<br>bique", "GDP":40.1}
];
```



사용자 정의 축 레이블 표현하기

자바스크립트 함수(사용자 정의 함수)를 이용해서 텍스트를 포맷팅한 후 이를 리턴한 값을 축 레이블로 표시할 수 있습니다. 사용자 정의 함수에 대한 자세한 사용법은 사용자 정의 함수 사용하기를 참조하십시요. 사용자 정의 함수를 이용해서 축 레이블을 표현하기 위해서는 축 타입 노드(예, <CategoryAxis>)의 labelJsFunction 속성값에 실행할 자바스크립트 함수명을 지정하고, 해당 함수의 코드를 작성해야 합니다. 다음은 X 축 레이블을 자바스크립트 함수를 이용해서 표현하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<horizontalAxis>
  <CategoryAxis categoryField="Month" displayName="Date"
        labelJsFunction="axisLabelFunc"/>
</horizontalAxis>
...

function axisLabelFunc (id, value) {
  return value + "<font color='#0000ff'>, 2013</font>";
}
```

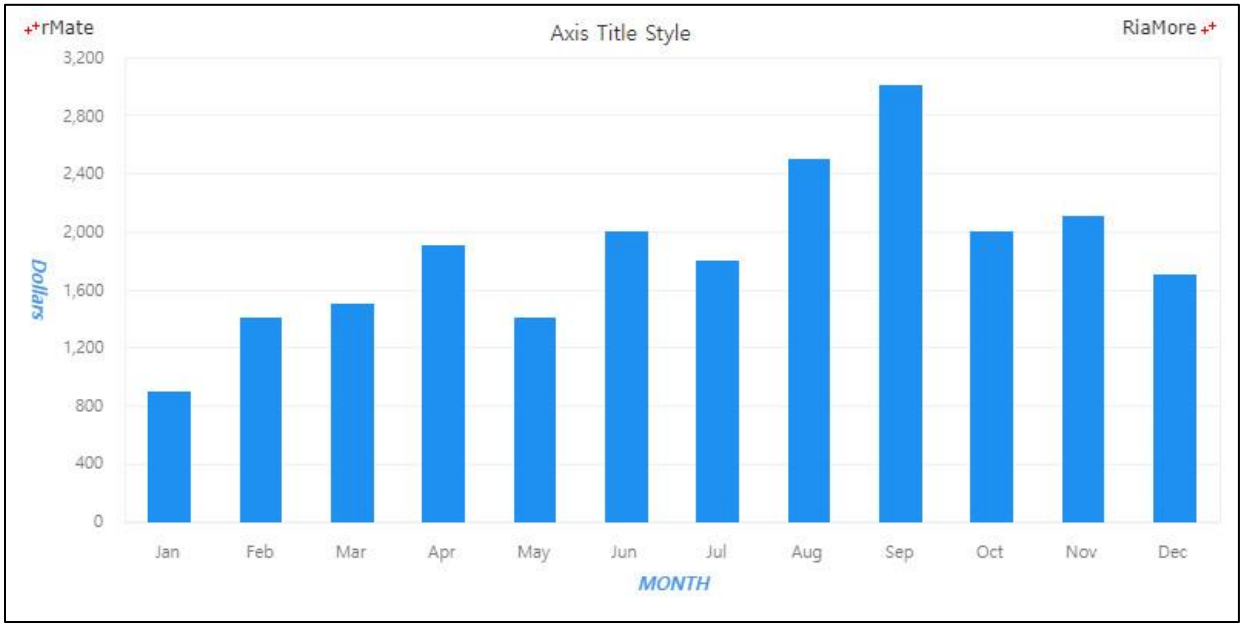


See the CodePen 알메이트 차트 - 사용자 정의 축 레이블 표현하기

축 타이틀 표현하기

축의 타이틀을 표현하기 원할 경우, 축 타입 노드(예, <CategoryAxis>)의 title 속성을 설정하면 됩니다. title 속성값이 축의 타이틀로 표시되며, 축 타이틀의 스타일링은 축 레이블과 같은 방법으로 <Axis2DRenderer>(혹은 <Axis3DRenderer>) 노드의 axisTitleStyleName 속성값에 스타일명을 지정하고, 지정된 스타일명으로 CSS 스타일을 작성하여 <Style> 노드에 정의합니다.

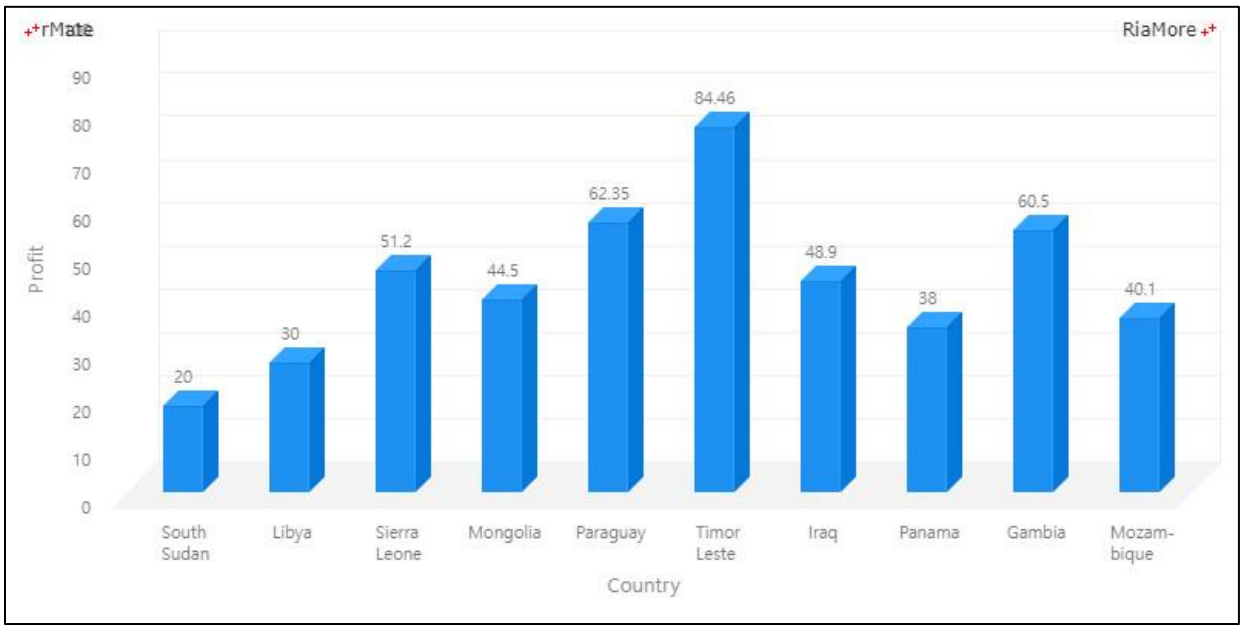
```
<horizontalAxis>
  <CategoryAxis id="hAxis" categoryField="Month" ticksBetweenLabels="false"
       title="MONTH" displayName="Month"/>
</horizontalAxis>
...
<horizontalAxisRenderers>
<Axis2DRenderer axis="{hAxis}" showLine="true" axisTitleStyleName="title"/>
  </horizontalAxisRenderers>
...
<Style>.title{color:0x4691E1; fontSize:12; fontWeight:bold; fontStyle:italic;}
</Style>
```



See the CodePen 알메이트 차트 - 축 타이틀 표현하기

세로축(Y 축) 타이틀의 텍스트가 길 경우, 타이틀을 표시하기 위해 차트의 공간을 많이 사용하는 문제가 있을 수 있습니다. 이 경우 verticalAxisTitleAlignment
속성값을 “vertical” (기본값: “horizontal”)로 지정하여 세로축(Y 축) 타이틀을 세로로 표기할 수 있습니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<verticalAxis>
  <LinearAxis id="vAxis" maximum="100" interval="10" title="Profit"/>
</verticalAxis>
<verticalAxisRenderers>
  <Axis2DRenderer axis="{vAxis}" showLine="false"
      verticalAxisTitleAlignment="vertical" />
</verticalAxisRenderers>
```

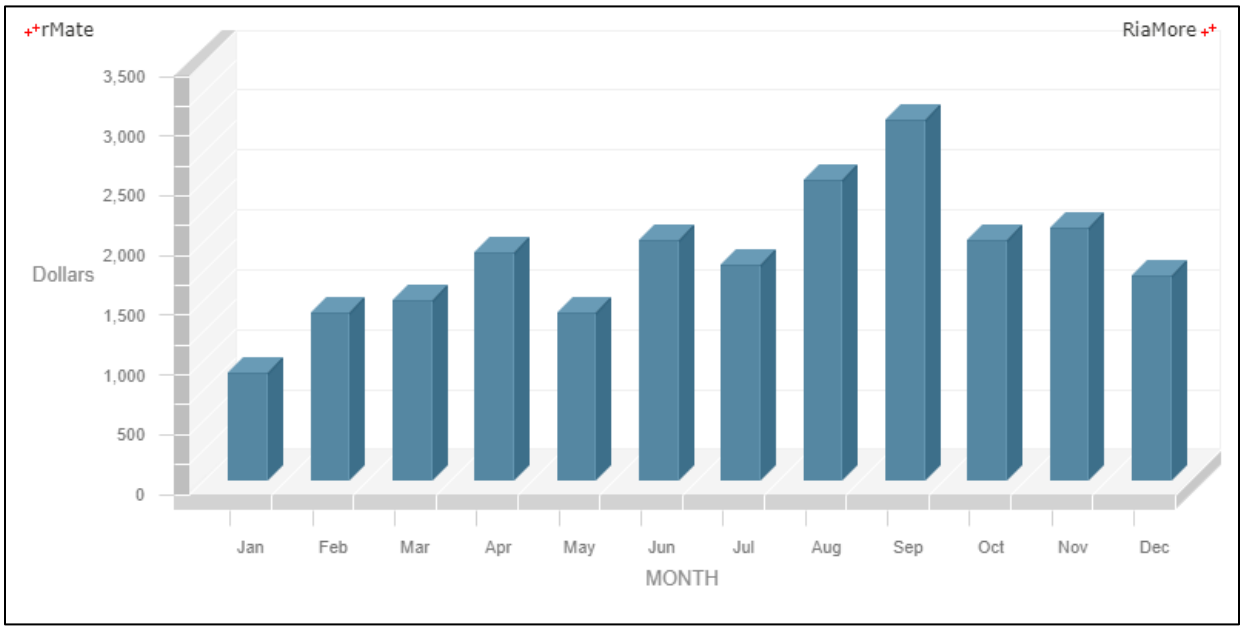


See the CodePen 알메이트 차트 - Y 축 타이틀 세로 쓰기

축의 선, 틱, 보조 틱 설정하기

축의 선, 틱, 보조 틱에 대한 스타일링은 각각 <axisStroke>, <tickStroke>, <minorTickStroke> 속성들을 설정하여 할 수 있습니다. 각 속성들에는 <Stroke> 객체 노드를 정의해야 합니다. 다음은 이에 대한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<horizontalAxis>
  <CategoryAxis id="hAxis" .../>
...
<verticalAxis>
  <LinearAxis id="vAxis" .../>
...
<horizontalAxisRenderers>
<Axis3DRenderer axis="{hAxis}" tickPlacement="inside" minorTickPlacement="cross"
       placement="bottom" showLine="true" ...>
  <axisStroke>
  ...
  <tickStroke>
  ...
  <minorTickStroke>
  ...
<verticalAxisRenderers>
  <Axis3DRenderer axis="{vAxis}" minorTickLength="0" tickPlacement="outside"
       minorTickPlacement="cross" placement="left" ...>
    <axisStroke>
    ...
    <tickStroke>
    ...
    <minorTickStroke>
    ...
```



See the CodePen 알메이트 차트 - 축의 선, 틱, 보조 틱 설정하기

틱과 보조 틱의 길이와 위치는 다음 표에 정의된 속성들을 설정하여 조정할 수 있습니다. (보조)틱을 표시하고 싶지않을 경우에는 tickPlacement(minorTickPlacement) 속성값을 “none” 으로 설정합니다. 그리고 축의 선과 레이블에 대한 표시 여부는 showLine 속성과 showLabels

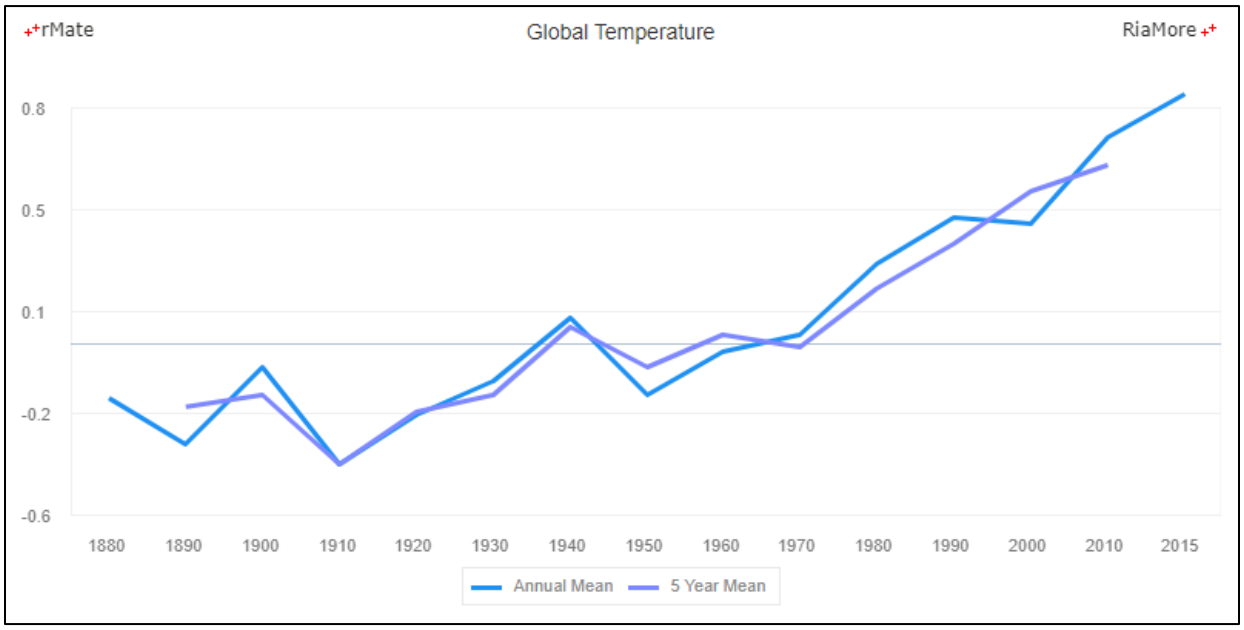
속성을 “true” 혹은 “false” 로 설정하여 조정할 수 있습니다. 틱과 레이블 사이의 간격은 labelGap 속성을 이용하여 조정합니다.

속성명	유효값 (*: 기본값)	설명
tickLength	숫자	눈금의 길이를 픽셀 단위로 지정합니다.
tickPlacement	inside, outside(*), cross, none	눈금의 위치를 지정합니다. inside: 눈금을 차트 영역 안에 표시합니다. outside: 눈금을 축 레이블 영역에 표시합니다. cross: 눈금을 축 상에 표시합니다. none: 눈금을 표시하지 않습니다.
minorTickLength	숫자	보조 눈금의 길이를 픽셀 단위로 지정합니다.
minorTickPlacement	inside, outside(*), cross, none	보조 눈금의 위치를 지정합니다. inside: 보조 눈금을 차트 영역안에 표시합니다. outside: 보조 눈금을 축 레이블 영역에 표시합니다. cross: 보조 눈금을 축 상에 표시합니다. none: 보조 눈금을 표시하지 않습니다.
showLine	true(*), false	축의 선을 표시할지 여부를 지정합니다.
showLabels	true(*), false	축의 레이블을 표시할지 여부를 지정합니다.
labelGap	숫자	눈금의 끝 부분과 레이블 사이의 간격을 지정합니다.

축의 눈금(틱)수 설정하기

축에 표시되는 눈금(틱)의 수는 기본적으로 차트 시스템이 최적의 개수를 자동으로 설정합니다. 사용자가 원하는 눈금의 수를 직접 설정하고 싶은 경우에는 축(예, <LinearAxis>) 노드의 tickCount 속성을 이용할 수 있습니다. 다음은 <LinearAxis> 노드에서 tickCount="5" 와 같이 설정하여 표현한 차트의 예제입니다.

```
<verticalAxis>
  <LinearAxis id="vAxis" maximum="1" tickCount="5"/>
</verticalAxis>
```

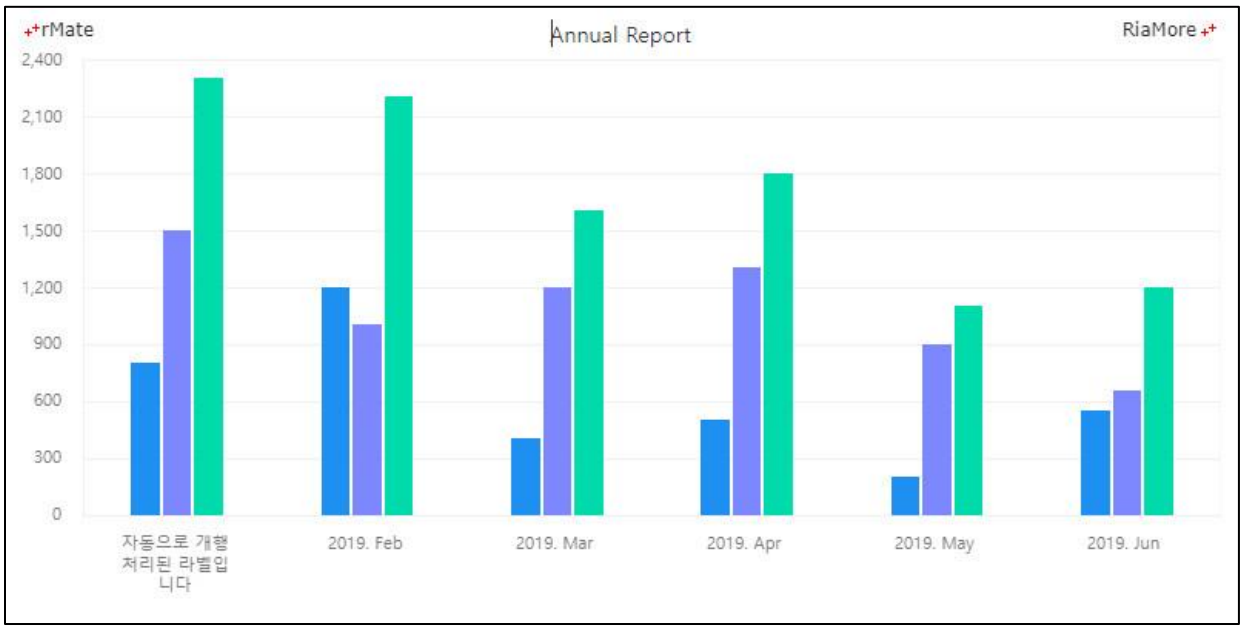


See the CodePen 알메이트 차트 - 축의 눈금수 설정하기

축 레이블 자동 개행처리

레이블이 긴 경우, 자동 개행처리 옵션을 지정할 수 있습니다. 축 렌더러(예, <Axis2DRenderer>, <Axis3DRenderer>)의 autoLineBreak 속성을 "true"로 설정하면 레이블이 표현되는 공간의 길이(lineBreakWidth 속성, 기본값 : "10") 보다 긴 레이블은 자동으로 개행처리되어 표시됩니다. 다음은 수평축(X 축) 레이블을 자동으로 개행처리하여 출력한 차트의 예제입니다.

```
<horizontalAxisRenderers>
  <Axis2DRenderer axis="{hAxis}" autoLineBreak="true"/>
</horizontalAxisRenderers>
```



See the CodePen 알메이트 차트 - 축 레이블 자동 개행처리

5.4 레이블

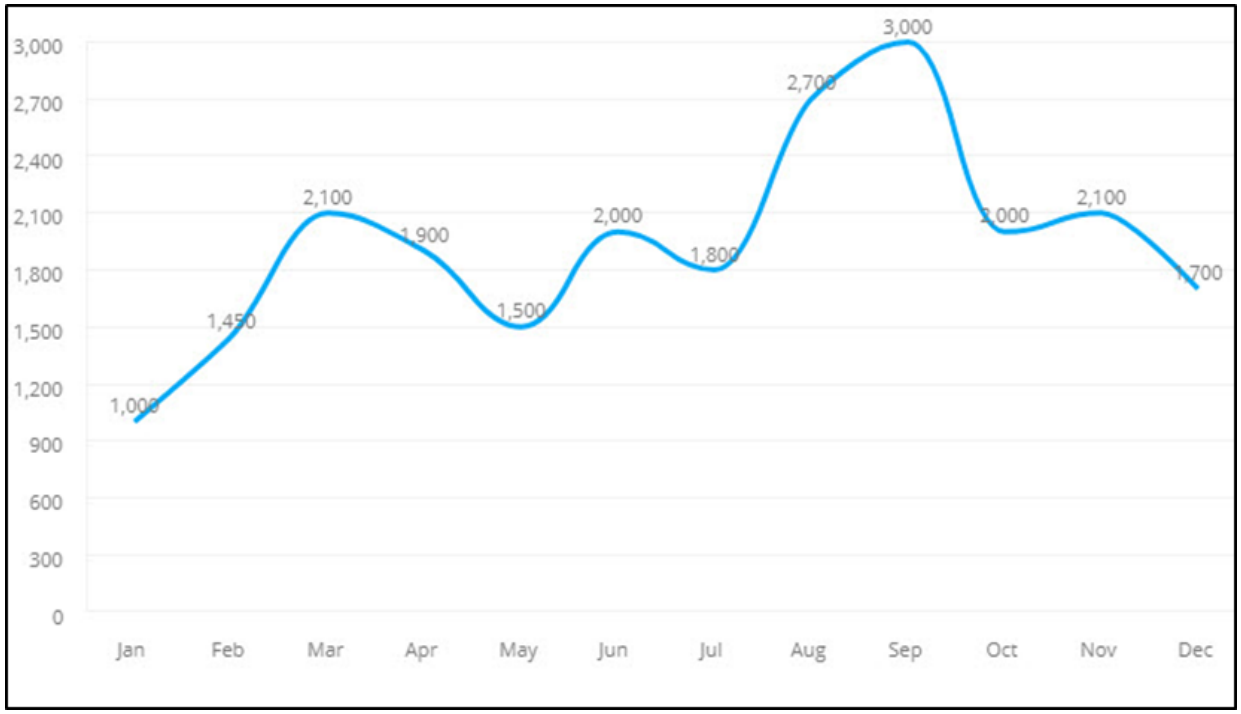
차트에 레이블(데이터 값)을 표시하는 방법은 차트의 종류마다 조금씩 다릅니다. 하지만 모든 차트에 공통적으로 적용되는 속성이 있는데, 이는 레이블이 표시되는 위치를 지정하는 labelPosition 속성입니다. 다음은 차트의 유형별로 labelPosition 속성에 설정할 수 있는 값을 나타내는 표입니다.

차트 종류	유효값 (*: 기본값)	설명
영역 차트, 라인 차트	up, down, both, none(*)	up: 데이터 포인트 위에 레이블을 표시합니다. down: 데이터 포인트 아래에 레이블을 표시합니다. both: 데이터 포인트 위, 아래에 레이블을 표시합니다. none: 레이블을 표시하지 않습니다.
바 차트, 컬럼 차트, 이미지 차트, 윙 차트	inside, outside, both, none(*)	inside: 바(칼럼) 안에 레이블을 표시합니다. outside: 바(칼럼) 밖에 레이블을 표시합니다. both: 바(칼럼) 안과 밖에 레이블을 표시합니다. none: 레이블을 표시하지 않습니다.
파이 차트	inside, outside, callout, insideWithCallout, none(*)	inside: 슬라이스 안에 레이블을 표시합니다. outside: 슬라이스 밖에 레이블을 표시합니다. callout: 슬라이스 밖에 레이블을 표시하고 지시선으로 해당 슬라이스와 연결합니다. insideWithCallout: 기본적으로 슬라이스 안에 레이블을 표시하고, 레이블이 겹칠 경우 callout 방식으로 레이블을 표시합니다. none: 레이블을 표시하지 않습니다.

버블 차트, 매트릭스 차트	inside, none(*)	inside: 버블(매트릭스) 안에 레이블을 표시합니다. none: 레이블을 표시하지 않습니다.
피라미드 차트	inside(*), callout, none	inside: 영역 안에 레이블을 표시합니다. callout: 영역 밖에 레이블을 표시하고 지시선으로 해당 영역과 연결합니다. none: 레이블을 표시하지 않습니다.
방사형 차트	outside, none(*)	outside: 영역 밖에 레이블을 표시합니다 none: 레이블을 표시하지 않습니다.

다음 이미지는 labelPosition 속성값을 “up” 으로 적용해서 출력한 라인 차트의 예제입니다.

```
<Line2DSeries labelPosition="up" yField="Profit" form="curve"/>
```

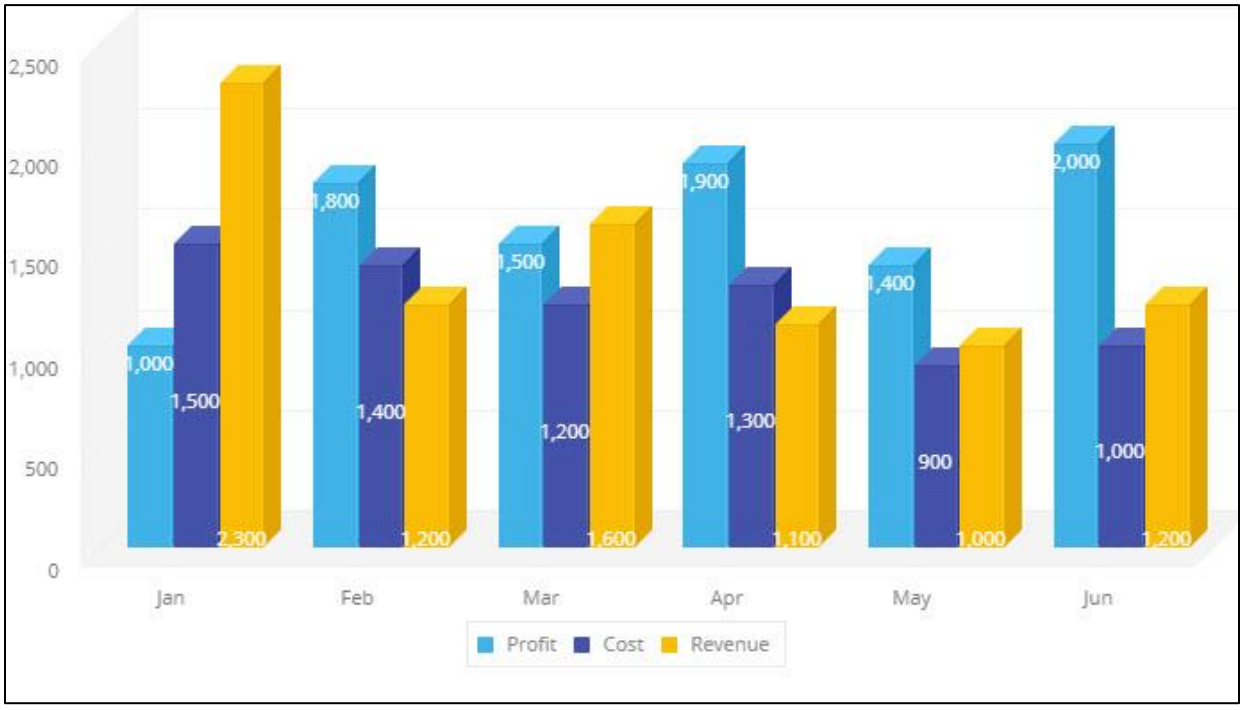


바 차트(혹은 컬럼 차트)는 labelPosition 속성값이 “inside” 일 때 바(칼럼)의 어느 위치에 레이블을 표시할지를 지정할 수 있습니다. 이는 labelAlign 속성값을 이용하여 가능합니다. 다음은 바 차트(혹은 컬럼 차트)에서 labelAlign 속성에 설정할 수 있는 값을 나타내는 표입니다.

차트 종류	유효값 (*: 기본값)	설명
바 차트	left, center(*), right	left: 바의 왼쪽끝에 레이블을 표시합니다. center: 바의 중앙에 레이블을 표시합니다. right: 바의 오른쪽 끝에 레이블을 표시합니다.
컬럼 차트	top, middle(*), bottom	top: 칼럼의 상단에 레이블을 표시합니다. middle: 칼럼의 중앙에 레이블을 표시합니다. bottom: 칼럼의 하단에 레이블을 표시합니다.

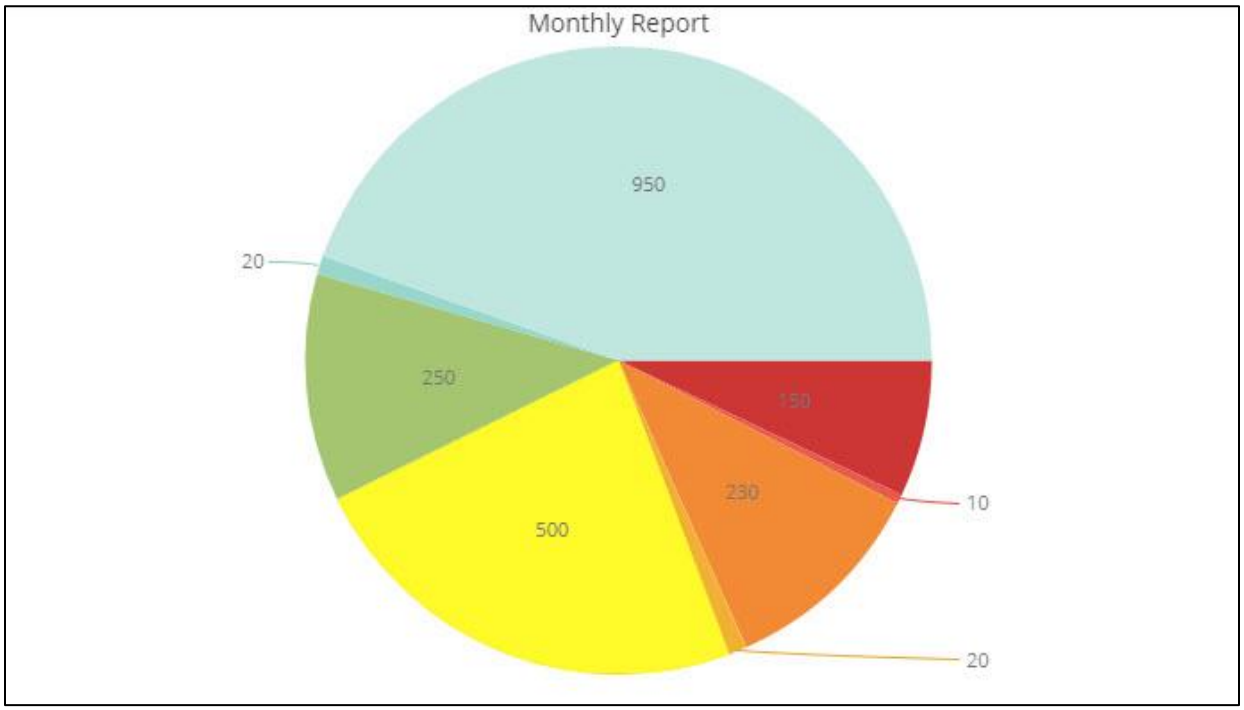
다음 이미지는 위 속성값을 적용하여 출력한 3D 컬럼 차트의 예입니다.

```
<Column3DSeries labelPosition="inside" labelAlign="top" yField="Profit" />
<Column3DSeries labelPosition="inside" labelAlign="middle" yField="Cost" />
<Column3DSeries labelPosition="inside" labelAlign="bottom" yField="Revenue" />
```



다음 이미지는 labelPosition 속성값을 “insideWithCallout” 으로 적용해서 출력한 파이 차트의 예입니다.

```
<Pie2DSeries field="Profit" nameField="Month" labelPosition="insideWithCallout">
```



labelXOffset, labelYOffset 속성 설정

레이블이 표시되는 위치는 위에서 설명한 labelPosition 속성과 labelAlign 속성에 의해서 자동으로 정해지지만 경우에 따라서 레이블의 위치를 사용자가 원하는 위치로 조정해야 할 필요가 있을 수 있습니다. 이 경우에 사용할 수 있는 속성이 labelXOffset 과 labelYOffset 입니다. 레이블의 가로 위치를 조정하고자 할 경우에는 labelXOffset 속성을 레이블의 세로 위치를 조정하고자 할 경우에는 labelYOffset 속성을 설정합니다.

주의

> 바(칼럼) 차트에서는 labelPosition 속성값이 “inside” 이면 해당 속성명이 insideLabelXOffset 과 insideLabelYOffset 이며, labelPosition 속성값이 “outside” 이면 해당 속성명이 outsideLabelXOffset 과 outsideLabelYOffset 입니다.

> 라인(영역) 차트에서는 labelPosition 속성값이 “up” 이면 해당 속성명이 upLabelXOffset 과 upLabelYOffset 이며, labelPosition 속성값이 “down” 이면 해당 속성명이 downLabelXOffset 과 downLabelYOffset 입니다.

레이블 겹침 방지

데이터 포인트의 수가 많은 경우 이웃 레이블들이 서로 겹쳐서 표현되는 경우가 있습니다. 이웃 레이블들이 서로 겹치지 않도록 하기 위해서는 시리즈 노드의 canDropSeriesLabels 속성값을 “true” 로 설정하면 됩니다. 다음은 플롯 차트에서 canDropSeriesLabels 속성값을 “true” 로 설정하여 이웃 레이블들의 겹침을 방지하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

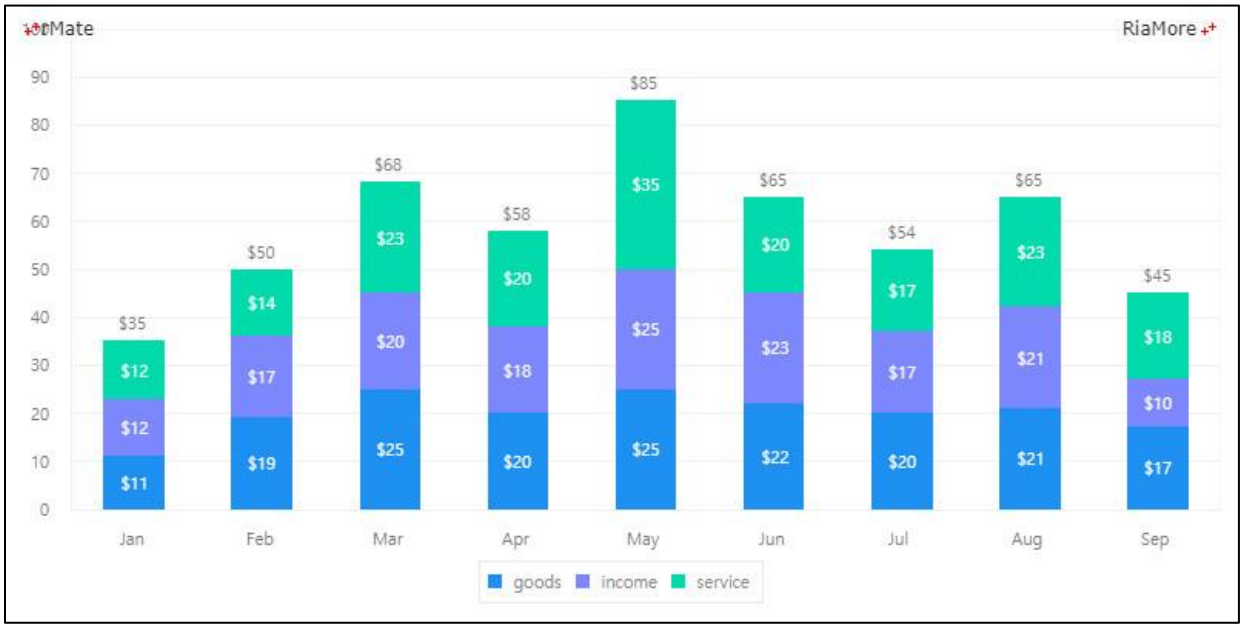
```
<Plot2DChart showDataTips="true" dataTipMode="single" canDropSeriesLabels="true">
  ...
    <Plot2DSeries labelPosition="up" yField="Profit" xField="Cost">
  ...
</Plot2DChart>
```

See the CodePen 알메이트 차트 - 레이블 겹침 방지

사용자 정의 레이블 표현하기

자바스크립트 함수(사용자 정의 함수)를 이용해서 텍스트를 포맷팅한 후 이를 리턴한 값을 레이블로 표시할 수 있습니다. 사용자 정의 함수에 대한 자세한 사용법은 사용자 정의 함수 사용하기를 참조하십시요. 사용자 정의 함수를 이용해서 레이블을 표현하기 위해서는 시리즈 노드(예, <Column2DSeries>)의 insideLabelJsFunction 속성값(labelPosition 속성값이 “inside” 인 경우)에 실행할 자바스크립트 함수명을 지정하고, 해당 함수의 코드를 작성해야 합니다. 스택 다중 시리즈 컬럼 차트에서는 각 시리즈의 데이터 합계를 함께 표시하는 경우가 많은데, 합계값을 사용자 정의 함수로 표현하기 위해서는 totalLabelJsFunction 속성값에 실행할 자바스크립트 함수명을 지정하고, 해당 함수의 코드를 작성해야 합니다. 다음은 화폐 기호($)를 레이블과 합계 레이블에 표현하기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DSet type="stacked" showTotalLabel="true"
        totalLabelJsFunction="totalLabelFunc">
  <series>
    <Column2DSeries labelPosition="inside" yField="goods"
        insideLabelJsFunction="seriesLabelFunc" color="#ffffff" />
    <Column2DSeries labelPosition="inside" yField="income"
        insideLabelJsFunction="seriesLabelFunc" color="#ffffff" />
    <Column2DSeries labelPosition="inside" yField="service"
        insideLabelJsFunction="seriesLabelFunc" color="#ffffff" />
  </series>
</Column2DSet>

function seriesLabelFunc(seriesID, index, data, values) {
  return "$" + values[1];
}
function totalLabelFunc(index, data, total){
  return "$" + total;
}
```



See the CodePen 알메이트 차트 - 사용자 정의 레이블 표현하기

5.5 효과 적용하기

차트가 화면에 표현될 때 애니메이션 효과를 줄 수 있습니다. 차트 생성시 효과를 적용하기 위해서는 해당 차트의 시리즈 노드에 showDataEffect 속성을 설정해야 합니다. showDataEffect 속성에 설정 가능한 객체는 다음과 같습니다.

- SeriesInterpolate : 축의 최소값에서 데이터 포인트까지의 데이터 변화를 애니메이션 효과로 나타냅니다.
- SeriesSlide : 한 방향에서 반대 방향으로 미끄러지는 애니메이션 효과를 표현합니다.
- SeriesZoom : 차트의 한 기준점으로부터 확대되어지는 애니메이션 효과를 표현합니다.
- SeriesClip : 차트의 기준점(상, 하, 좌, 우)에서 반대 방향으로 연결되어 나가는 애니메이션 효과를 표현합니다.

주의

<SeriesClip> 은 라인 차트와 영역 차트에서만 지원됩니다.

SeriesInterpolate 적용하기

다음은 컬럼 차트 시리즈의 showDataEffect 속성에 SeriesInterpolate 노드를 설정하여 효과를 주기위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Column2DSeries yField="GDP" displayName="GDP Growth (In %)" >
  <showDataEffect>
    <SeriesInterpolate duration="1200" elementOffset="60"/>
  </showDataEffect>
</Column2DSeries>
```

See the See See the CodePen 알메이트 차트 - SeriesInterpolate 효과 적용하기

다음은 <SeriesInterpolate> 노드에서 설정 가능한 속성을 정리한 표입니다.

속성명	유효값 (*: 기본값)	설명
duration	밀리세컨드 기본값: 500	전체 효과가 완료되는데 필요한 시간을 지정합니다.

elementOffset	밀리세컨드 기본값: 20	한 데이터 요소에 대한 효과가 시작된 이후 다음 데이터 요소에 대한 효과가 시작되기까지의 지연 시간을 지정합니다.
minimumElementDuration	밀리세컨드 기본값: 0	한 데이터 요소의 효과가 완료되는데 필요한 시간을 지정합니다. duration 속성값이 minimumElementDuration 속성값보다 작을 경우, minimumElementDuration 속성은 무시됩니다.
offset>	밀리세컨드 기본값: 0	시리즈 사이의 지연 시간을 지정합니다. 다중 시리즈에서 시리즈와 시리즈 간에 지연 시간을 두고 싶으면 이 속성을 사용합니다.

SeriesInterpolate 효과를 적용할 때 elementOffset 속성값에 음수를 설정하면 효과가 역순으로 표현됩니다. 다음은 이에 대한 코드입니다.

```
<SeriesInterpolate duration="1200" elementOffset="-60"/>
```

See the CodePen 알메이트 차트 - SeriesInterpolate 효과 (elementOffset 속성값에 음수 설정)

SeriesInterpolate 효과를 적용할 때 easingFunction 속성을 설정하여 바운스 효과를 줄 수 있습니다. 다음은 이에 대한 코드입니다.

```
<SeriesInterpolate duration="2000" elementOffset="60"
easingFunction="EaseOutElastic"/>
```

See the CodePen 알메이트 차트 - SeriesInterpolate 효과 (바운스 효과)

SeriesSlide 효과 적용하기

다음은 컬럼 차트 시리즈의 showDataEffect 속성에 SeriesSlide 노드를 설정하여 효과를 주기위한 코드입니다.

```
<Column2DSeries yField="GDP" displayName="GDP Growth (In %)" >
  <showDataEffect>
    <SeriesSlide direction="left" duration="1200" elementOffset="60"/>
  </showDataEffect>
</Column2DSeries>
```

See the CodePen 알메이트 차트 - SeriesSlide 효과 적용하기

<SeriesSlide> 노드에서 설정 가능한 속성은 <SeriesInterpolate> 노드에서 사용 가능한 모든 속성과 다음 direction 속성입니다.

속성명	유효값 (*: 기본값)	설명
direction	left(*), right, up, down	슬라이딩되는 방향을 지정합니다.

hideDataEffect 속성에 <SeriesSlide> 노드를 정의하면 차트의 데이터가 재 설정(setData 함수 호출)될 때 숨기기 효과를 줄 수 있습니다. 다음은 컬럼 차트 시리즈의 hideDataEffect 속성과 showDataEffect 속성에 <SeriesSlide> 노드를 설정하여 효과를 주기위한 코드입니다.

```
<Column2DSeries yField="GDP" displayName="GDP Growth (In %)" >
  <showDataEffect>
    <SeriesSlde direction="left" duration="1200" elementOffset="60"/>
  </showDataEffect>'
  <hideDataEffect>
    <SeriesSlide direction="down" duration="1200" elementOffset="60"/>
  </hideDataEffect>
</Column2DSeries>
```

See the CodePen 알메이트 차트 - SeriesSlide 효과 (숨기기 효과 적용하기)

SeriesZoom 효과 적용하기

다음은 컬럼 차트 시리즈의 showDataEffect 속성에 SeriesZoom 노드를 설정하여 효과를 주기위한 코드입니다.

```
<Column2DSeries yField="GDP" displayName="GDP Growth (In %)" >
  <showDataEffect>
    <SeriesZoom relativeTo="chart" horizontalFocus="center" verticalFocus="center"
       duration="1200" elementOffset="60"/>
  </showDataEffect>
</Column2DSeries>
```

See the CodePen 알메이트 차트 - SeriesZoom 효과 적용하기

<SeriesZoom> 노드에서 설정 가능한 속성은 <SeriesInterpolate> 노드에서 사용 가능한 모든 속성과 다음 표에 정리된 속성입니다.

속성명	유효값 (*: 기본값)	설명
horizontalFocus	center(*), left, right	줌이 시작되는 수평 초점을 지정합니다.
verticalFocus	center(*), top, bottom	줌이 시작되는 수직 초점을 지정합니다.
relativeTo	chart(*), series	줌 효과가 시작되는 경계(chart 혹은 series)를 지정합니다.

다음은 relativeTo 속성값에 “series” 를 지정하는 코드입니다.

```
<Column2DSeries yField="GDP" displayName="GDP Growth (In %)" >
  <showDataEffect>
    <SeriesZoom relativeTo="series" horizontalFocus="center" verticalFocus="center"
       duration="1200" elementOffset="60"/>
  </showDataEffect>
</Column2DSeries>
```

See the CodePen 알메이트 차트 - SeriesZoom 효과 적용하기 (relativeTo 속성값에 series 를 지정)

SeriesClip 효과 적용하기

다음은 라인 차트 시리즈의 showDataEffect 속성에 SeriesClip 노드를 설정하여 효과를 주기위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
<Line2DSeries yField="Vancouver" displayName="Vancouver">
  <showDataEffect>
    <SeriesClip duration="1000"/>
  </showDataEffect>
</Line2DSeries>
```

See the CodePen 알메이트 차트 - SeriesClip 효과 적용하기

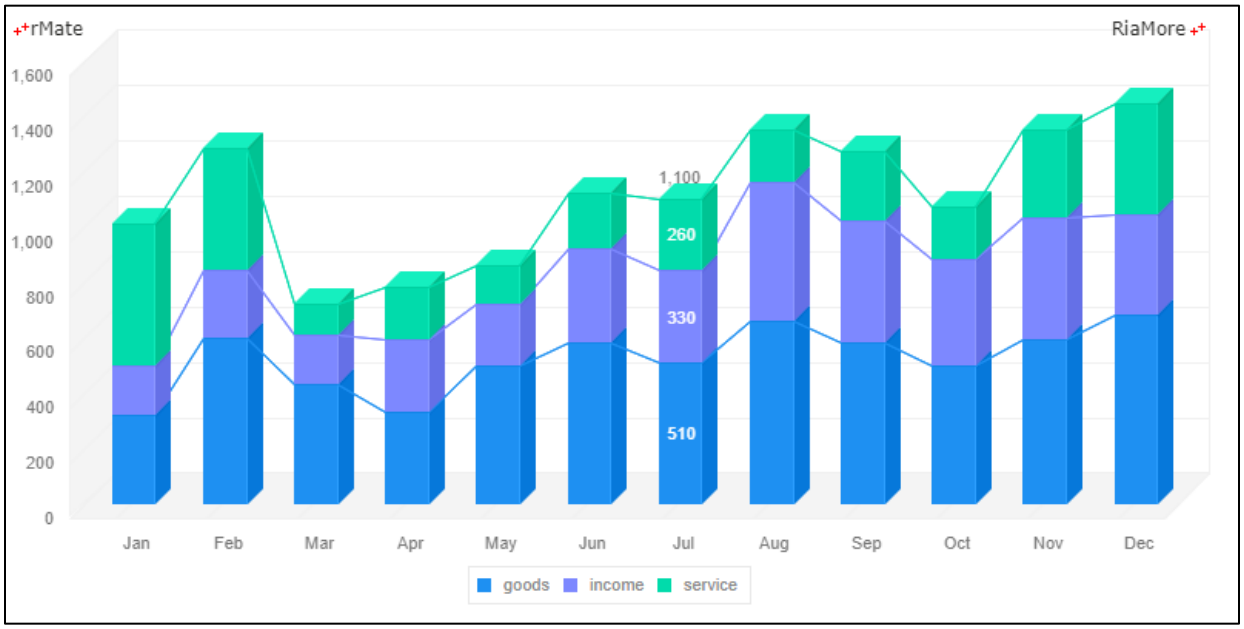
SeriesClip 노드에서 설정 가능한 속성은 <SeriesInterpolate> 노드에서 사용 가능한 모든 속성과 다음 direction 속성입니다.

속성명	유효값 (*: 기본값)	설명
direction	left, right(*), up, down	효과가 진행되는 방향을 지정합니다.

5.6 데이터 연결선

박스 형태로 그래프가 표현되는 차트 종류(바 차트, 컬럼 차트, 윙 차트)에서 인접한 데이터 포인트를 연결하는 연결선을 그을 수 있습니다. 연결선을 긋기 위해서는 해당 차트의 시리즈 노드에 lineToEachItems 속성값을 “true” 로 설정합니다. 다음은 3D 컬럼 차트에서 연결선을 긋기 위한 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
...
  <Column3DSeries labelPosition="inside" yField="goods" displayName="goods"
       lineToEachItems="true" showValueLabels="[6]" color="#ffffff">
...
```



See the CodePen 알메이트 차트 - 데이터 연결선

5.7 시각 장애인을 위한 기능과 패턴

시각 장애인을 위한 대체 텍스트 사용하기

알메이트 차트는 차트의 내용을 인식하기 어려운 시각 장애인을 위해서 대체 텍스트 기능을 제공합니다. 차트에 데이터(차트에 대한 자세한 설명)를 삽입하였을 경우 알메이트 차트는 차트 내에 삽입된 데이터를 대체 텍스트로 저장합니다. 사용자는 대체 텍스트를 읽어 줄 수 있는 프로그램이나 기기를 이용하여 대체 텍스트 내용을 이해할 수 있습니다. 대체 텍스트를 읽어 줄 수 있는 프로그램이나 기기와 관련하여 알메이트 차트에서 취급하거나 제공하는 기능은 없습니다. 또한 이 기능의 이용을 위해 사용자가 특별히 조작하거나 수정해야할 부분도 없습니다. 대체 텍스트 기능의 사용을 위해서는 다음과 같이 chartVars 변수를 설정해야 합니다. chartVars 변수 설정에 대한 자세한 내용은 차트 생성 함수를 참조하십시요.

```
chartVars += "&accessibility=true";
```

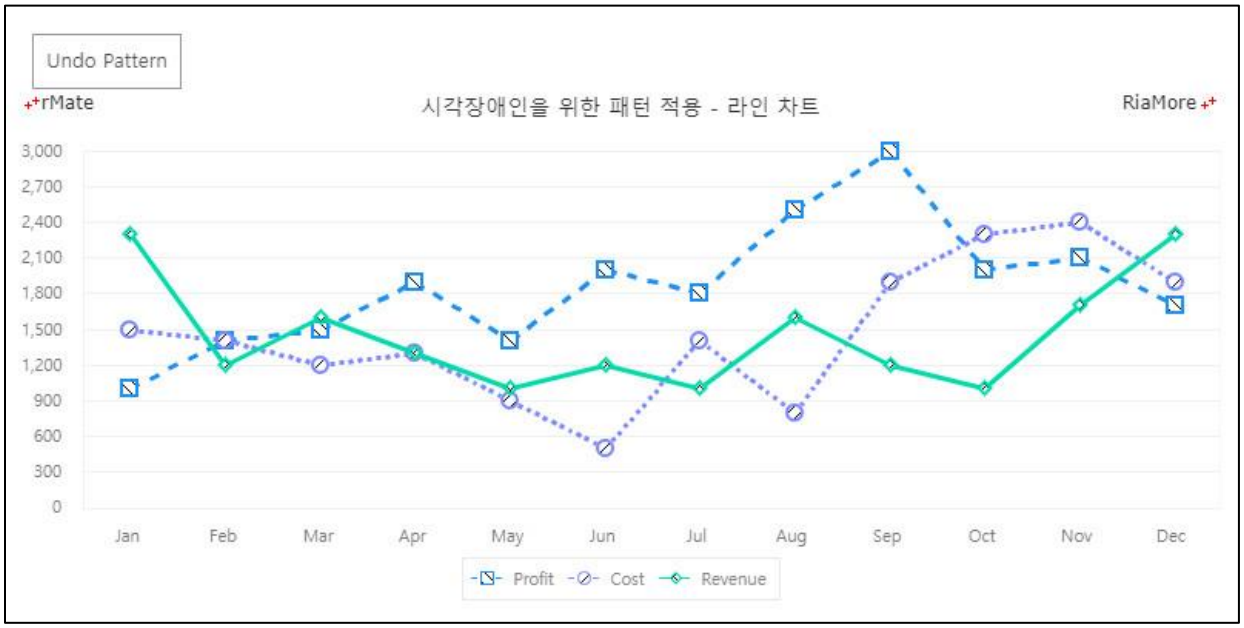
패턴 사용하기

색맹, 색약 사용자들은 차트에 적용된 색을 구분하기 어려울 수 있습니다. 이런 사용자들에게는 차트에 서로 다른 색을 적용해서 차트의 가독성을 높이기 보다는 서로 다른 패턴을 적용해서 차트의 가독성을 높이는 것이 더 바람직할 것입니다. 알메이트 차트에서는 기본적으로 20 가지의 패턴을 제공하며 사용자는 이를 수정하거나 자신이 원하는 패턴을 등록할 수 있습니다. 패턴을 사용하기 위해서는 다음과 같이 chartVars 변수를 설정하고, 패턴 이미지가 존재하는 디렉토리 명(url)과 패턴 이미지 파일명을 등록해야 합니다. 아래 코드 예는 알메이트 차트에서 기본으로 제공하는 20 가지의 패턴 이미지 파일을 등록하는 것을 보여줍니다.

```
chartVars += "&usePattern=true";

rMateChartH5.patternImageBaseUrl = "../rMateChartH5/Assets/Patterns/";

rMateChartH5.patternImagesUrl = [
  "diagonal_ltr.png",
  "diagonal_rtl.png",
  "diagonal.png",
  "horizontal.png",
  ...
];
```

위와 같이 패턴 이미지가 존재하는 디렉토리와 파일명들을 등록한 다음, 패턴을 이용하려고 하는 차트의 enablePattern 속성값을 “true” 로 설정하면 차트에 패턴이 적용됩니다. 다음은 라인 차트에 패턴을 사용하는 코드와 이를 적용해서 출력한 차트의 예제입니다.

```
...
<Line2DChart showDataTips="true" enablePattern="true">
...
```

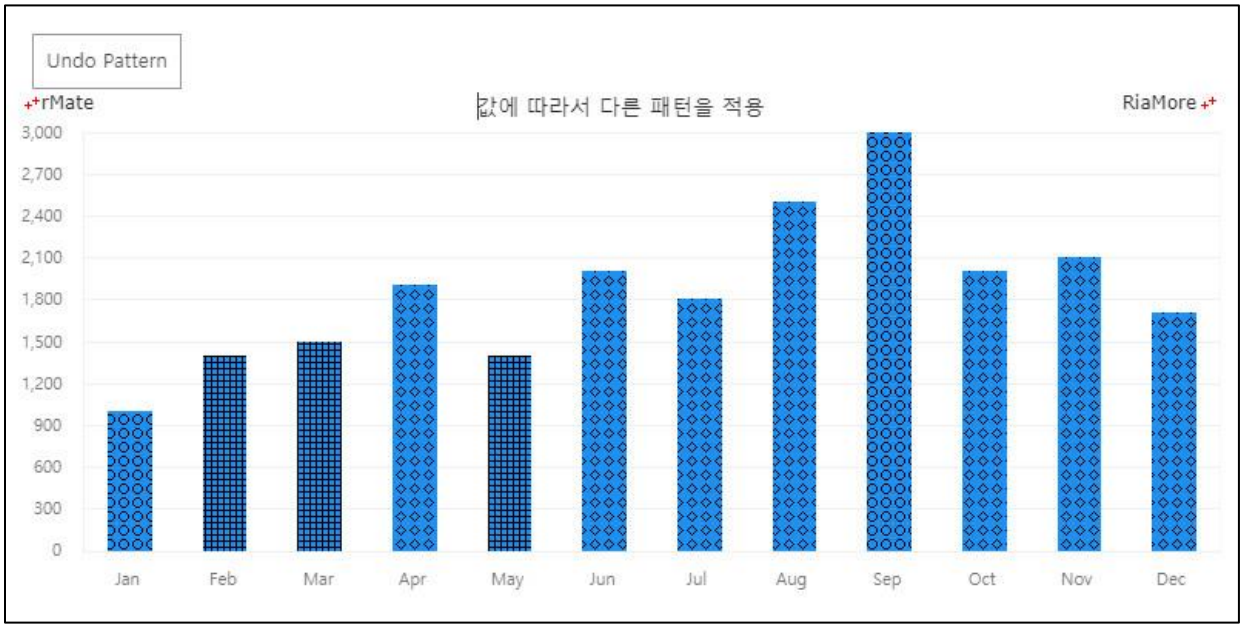


See the CodePen 알메이트 차트 - 차트에 패턴 적용

사용자 정의 패턴 사용하기

패턴 사용시 각 시리즈에 적용되는 패턴의 종류는 등록된 패턴 이미지 파일 중에서 자동으로 하나를 선택하여 적용합니다. 하지만 경우에 따라서 사용자가 원하는 패턴을 직접 적용할 필요가 있습니다. 이를 위해서 사용자 정의 패턴을 사용합니다. 사용자 정의 패턴을 적용하기 위해서는 시리즈의 patternJsFunction 속성값에 실행할 자바스크립트 함수명을 지정하고, 해당 함수의 코드를 작성해야 합니다. 다음은 이를 컬럼 차트에 사용하는 코드와 이를 적용해서 출력한 차트의 예입니다. 이 예에서는 값이 1500 이하이면 등록된 패턴 이미지 배열 중에서 5 번째 인덱스에 해당하는 이미지(both.png)를 적용하고, 값이 1500 이상 2500 이하이면 등록된 패턴 이미지 배열 중에서 14 번째 인덱스에 해당하는 이미지(diamond.png)를 적용합니다.

```
...
<Column2DChart showDataTips="true" patternMode="true">
...
<Column2DSeries yField="Profit" displayName="Profit"
        patternJsFunction="patternFunc">
...

function patternFunc(id, values, patternIndex) {
  if (values[1] <= 1500)
    return 5;
  else if (values[1] >= 1500 && values[1] <= 2500)
    return 14;
  else
    return 19;
}
```



See the CodePen 알메이트 차트 - 차트에 사용자 정의 패턴 사용하기

6. 고급 사용자를 위한 기능

6.1 Setter/Getter 활용

알메이트 차트는 Setter/Getter 를 사용하여 클래스의 속성을 직접 변경할 수 있습니다. 이 기능은 이미 생성된 차트의 모양 중 일부를 변경할때 유용하게 사용할 수 있습니다.

차트에 입력된 레이아웃을 기준으로 root, chart, axis, series 등의 클래스들을 가져올 수 있고, 해당 클래스들의 속성을 변경할 수 있습니다.

예를들어 Column2DChart 의 verticalAxis(LinearAxis)의 maximum 값을 변경하고 싶다면, 아래와 같이 root -> chart -> verticalAxis 로 접근하여 maximum 값을 설정하면 됩니다.

```
// 차트 생성시 사용한 레이아웃
<Column2DChart>
    <verticalAxis>
        <LinearAxis maximum="10"/>
    </verticalAxis>
</Column2DChart>

// 버튼 등의 이벤트 발생 시 verticalAxis(LinearAxis)의 maximum 값을 변경하는 로직
var rootElem = document.getElementById(chartId),
     root = rootElem.getRoot(),
     chart = root.chart, // 위 레이아웃 기준으로 Column2DChart
     linearAxis = chart.verticalAxis; // 위 레이아웃 기준으로 LinearAxis

linearAxis.maximum = 20
```

예를들어 Column2DSeries 의 fill 값을 변경하고 싶다면, 아래와 같이 root -> chart -> series 로 접근하여 fill 값을 설정하면 됩니다.

(series 속성은 클래스 배열값을 갖으며, 레이아웃에 Column2DSeries 를 여러개 사용했을때는 순서대로 0, 1, 2...번째 series 값을 가져오면 됩니다.)

```
// 차트 생성시 사용한 레이아웃
<Column2DChart>
    <series>
        <Column2DSeries> // 0 번째 시리즈
        </Column2DSeries>
        <Column2DSeries> // 1 번째 시리즈
        </Column2DSeries>
    </series>
</Column2DChart>

// 버튼 등의 이벤트 발생 시 Column2DSeries 에 접근하여 fill 을 변경하는 로직
var rootElem = document.getElementById(chartId),
    root = rootElem.getRoot(),
    chart = root.chart, // 위 레이아웃 기준으로 Column2DChart
    series0 = chart.series[0]; // 위 레이아웃 기준으로 Column2DSeries
```

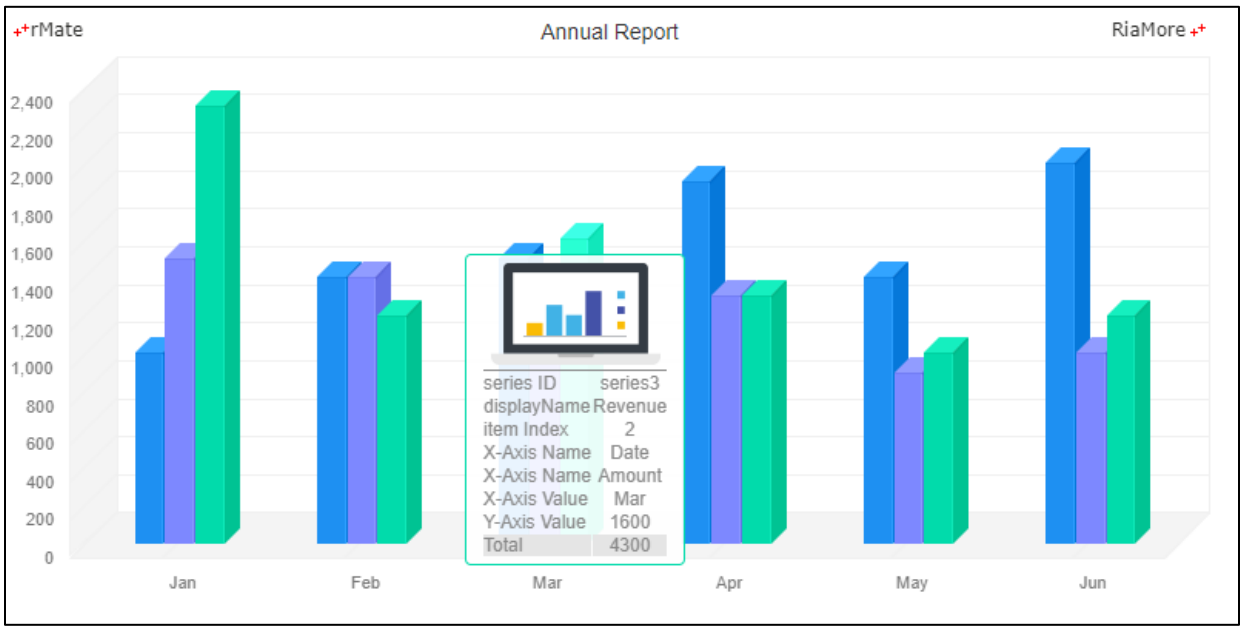
클래스와 속성에 대한 자세한 내용은 API 를 확인해주세요. 속성 중 id 나 styleName 등은 사용할 수 없습니다.

6.2 사용자 정의 함수 사용하기

알메이트 차트는 차트에 표시되는 텍스트의 내용이나 형식을 사용자가 원하는 형태로 표현할 수 있도록 사용자가 작성한 자바스크립트 함수를 실행하는 기능을 제공합니다. 이러한 사용자 정의 함수 기능은 차트의 특정 요소에 이벤트가 발생하면(예, 클릭되었을 때) 실행될 수도 있습니다. (차트의 이벤트 처리에 대한 자세한 내용은 이벤트 처리를 참조하십시요.)

예를 들어서, 차트의 데이터 포인트에 마우스를 올렸을 때 표시되는 툴팁의 텍스트는 데이터셋에 설정된 텍스트가 기본적으로 표시됩니다. 경우에 따라서 사용자가 툴팁에 특정 이미지를 함께 표시하기를 원하거나 정해진 값보다 큰 데이터 포인트의 툴팁에 표시되는 텍스트에 추가적인 텍스트의 삽입을 원할 수도 있습니다. 이러한 작업은 사용자 정의 함수 기능을 활용하면 쉽게 구현이 가능합니다. 다음은 사용자 정의 함수를 이용하여 툴팁에 이미지와 추가적인 텍스트를 삽입한 에제입니다. 예제 차트의 컬럼에 마우스 포인터를 올리면 사용자 정의 함수로 구현된 툴팁을 보실 수 있습니다.

```
<Column3DChart showDataTips="true" dataTipJsFunction="dataTipFunc">

function dataTipFunc(seriesId, seriesName, index, xName, yName, data, values) {
  return "<table cellpadding='0' cellspacing='1'>"
    +"<tr>"
      +"<td align='center' colspan='2' style='border-bottom:solid 1px #8b8b8b;'><img
       src='../rMateChartH5/Assets/Images/monitor.png'></td>"
    +"</tr><tr>"
      +"<td >series ID</td><td align='center'>" + seriesId + "</td>"
    +"</tr><tr>"
      +"<td>displayName</td><td align='center'>" + seriesName + "</td>"
    +"</tr><tr>"
      +"<td>item Index</td><td align='center'>" + index + "</td>"
    +"</tr><tr>"
      +"<td>X-Axis Name</td><td align='center'>" + xName + "</td>"
    ...
    +"</tr></table>";
}

function getSum(values) {
  var sum = 0;
  for(var v in values) {
    sum += Number(values[v]) || 0;
  }
  return sum;
}
```



See the CodePen 알메이트 차트 - 사용자 정의 함수를 사용하여 툴팁 표시

사용자 정의 함수를 이용하여 차트의 특정 구성 요소의 표현을 변경하고자 할 경우에는 해당 구성 요소가 표현되는 노드의 (예, <Column2DChart> 노드) 속성에 실행될 자바스크립트 함수를 설정해야 합니다. 다음 표는 사용자 정의 자바스크립트 함수의 설정이 가능한 노드와 해당 노드의 속성을 설명한 내용입니다.

속성명	유효값 (*: 기본값)	설명
dataTipJsFunction	모든 차트 레벨 노드	사용자가 원하는 툴팁을 표현하기 위한 자바스크립트 함수를 설정합니다.
fillJsFunction	모든 시리즈 노드 <Gauge> <LinearGauge>	사용자가 원하는 색을 데이터 포인트에 표현된 아이템에(예, 컬럼 차트에서 컬럼) 적용하기 위한 자바스크립트 함수를 설정합니다.
labelJsFunction	<BrokenAxis> <CategoryAxis> <CategoryLinearAxis> <DateTimeAxis> <HistogramCategoryAxis>	사용자가 원하는 축의 레이블을 표현하기 위한 자바스크립트 함수를 설정합니다.

	<LinearAxis> <HistogramCategoryAxis> <LinearAxis> <LogAxis> <Axis2DRenderer> <Axis2DWingRenderer> <Axis3DRenderer> <BrokenAxis2DRenderer> <BrokenAxis3DRenderer> <HistogramAxis2DRenderer> <ScrollableAxisRenderer>	
groupLabelJsFunction	<CategoryAxis>	사용자가 원하는 축의 그룹 레이블을 표현하기 위한 자바스크립트 함수를 설정합니다.
downLabelJsFunction	<Area2DSeries> <CandleArea2DSeries> <CandleLine2DSeries> <Line2DSeries> <MotionLineSeries>	사용자가 원하는 레이블(선 아래에 표시)을 표현하기 위한 자바스크립트 함수를 설정합니다.
upLabelJsFunction	<Area2DSeries> <CandleArea2DSeries> <CandleLine2DSeries> <Line2DSeries> <MotionLineSeries>	사용자가 원하는 레이블(선 위에 표시)을 표현하기 위한 자바스크립트 함수를 설정합니다.
insideLabelJsFunction	<Bar2DSeries> <Bar2DWingSeries> <Bar3DSeries> <Column2DSeries> <Column2DWingSeries> <Column3DSeries> <Equalizer2DSeries> <Histogram2DSeries> <Histogram3DSeries> <HTarget2DGoalSeries> <HTarget2DResultSeries> <HTarget3DGoalSeries> <HTarget3DResultSeries> <ImageSeries> <Matrix2DSeries> <MotionColumnSeries> <VTarget2DGoalSeries>	사용자가 원하는 레이블(박스 안에 표시)을 표현하기 위한 자바스크립트 함수를 설정합니다.

	<VTarget2DResultSeries> <VTarget3DGoalSeries> <VTarget3DResultSeries> <WingSeries>	
outsideLabelJsFunction	<Bar2DSeries> <Bar2DWingSeries> <Bar3DSeries> <Column2DSeries> <Column2DWingSeries> <Column3DSeries> <Equalizer2DSeries> <Histogram2DSeries> <Histogram3DSeries> <HTarget2DGoalSeries> <HTarget2DResultSeries> <HTarget3DGoalSeries> <HTarget3DResultSeries> <ImageSeries> <MotionColumnSeries> <VTarget2DGoalSeries> <VTarget2DResultSeries> <VTarget3DGoalSeries> <VTarget3DResultSeries> <WingSeries>	사용자가 원하는 레이블(박스 밖에 표시)을 표현하기 위한 자바스크립트 함수를 설정합니다.
totalLabelJsFunction	<Area2DSet> <Bar2DSet> <Bar3DSet> <Column2DSet> <Column3DSet>	사용자가 원하는 합계 레이블을 표현하기 위한 자바스크립트 함수를 설정합니다.
tickLabelJsFunction	<CircularGauge>	사용자가 원하는 눈금 레이블을 표현하기 위한 자바스크립트 함수를 설정합니다.
valueLabelJsFunction	<HCylinderGauge> <HLinearGauge> <VCylinderGauge> <VLinearGauge>	사용자가 원하는 게이지 값을 표현하기 위한 자바스크립트 함수를설정합니다.
maxLabelJsFunction	<Candlestick2DSeries>	캔들스틱 차트의 화면에 보이는 값 중에서 가장 큰 값을 사용자가 원하는 형태로 표현하기 위한 자바스크립트 함수를 설정합니다.

minLabelJsFunction	<Candlestick2DSeries>	캔들스틱 차트의 화면에 보이는 값 중에서 가장 작은 값을 사용자가 원하는 형태로 표현하기 위한 자바스크립트 함수를 설정합니다.

6.3 이벤트 처리

알메이트 차트는 사용자가 차트와 인터렉션하기 위한 방법으로 이벤트 처리 기능을 제공합니다. 이벤트 처리를 통해서 차트의 특정 데이터 포인트의 값이 미리 정의한 어떤 조건과 부합할 경우 특별한 처리를 할 수 있습니다. 또한 사용자가 차트 내의 구성 요소 (배경, 제목, 레이블, 데이터 포인트 등)에 마우스를 클릭하거나 호버링(hovering)할 경우 혹은 차트의 값이 변경되거나 확대(Zooming)가 일어날 경우 필요한 정보를 보여주기 위한 사용자 정의 자바스크립트 함수를 작성할 수 있습니다.

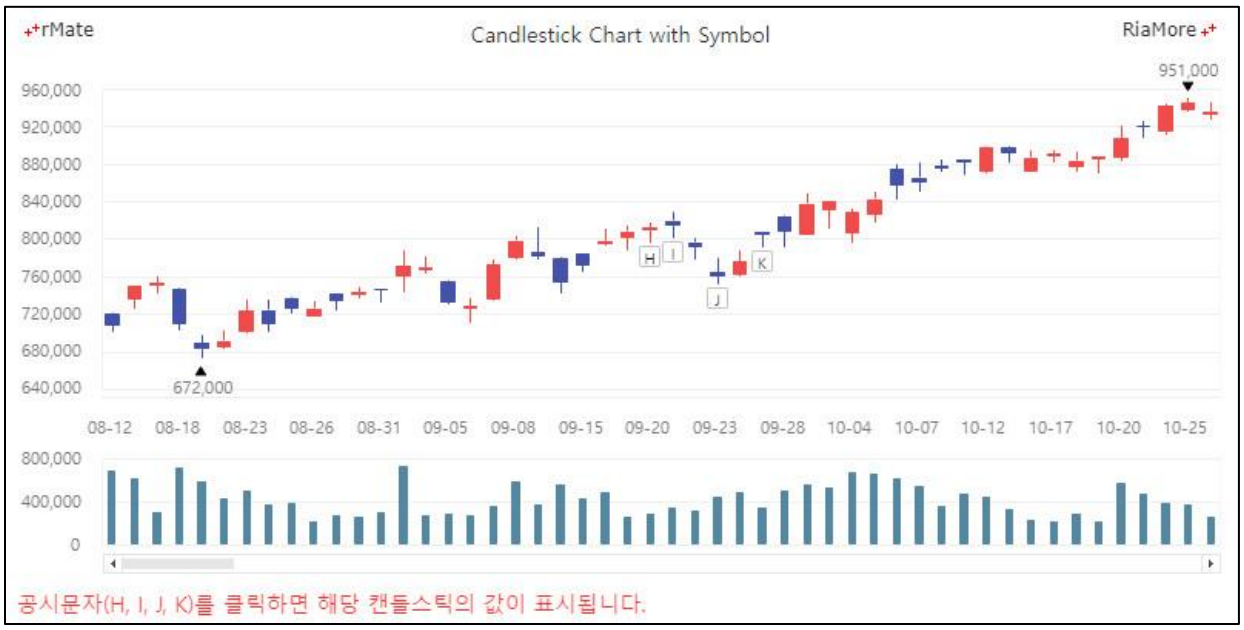
다른 차팅 제품들에 비해서 알메이트 차트의 이벤트 처리 방식의 장점은 개발자에게 쉬운 코딩 인터페이스를 제공한다는 것입니다. 알메이트 차트에서는 다른 모든 차트 제품들이 사용하는 방식인 이벤트 리스너를 등록하는 자바스크립트 코딩을 할 필요가 없습니다. 레이아웃을 작성할 때 이벤트 처리가 필요한 차트 요소를 정의하는 XML 노드 상에 특정 이벤트 속성값을 자바스크립트 함수명으로 설정하는 간단한 방식을 사용합니다.

클릭 이벤트 처리하기

다음은 캔들스틱 차트에서 사용자가 심볼을 클릭하면 공시자료를 보여주는 예제입니다.

```
<Candlestick2DSeries openField="open" closeField="close" highField="high"
        lowField="low" symbolClickJsFunction="gongsiDataFunc" symbolColor="#000">

function gongsiDataFunc(seriesId, index, data, values) {
  var gongsi = "Click the symbol";
  if(data.gongsi)
    gongsi = data;
  setGongsiData(gongsi);
}
```

위 예제 코드에서는 캔들스틱 차트의 시리즈를 정의하는 <Candlestick2DSeries> 노드의 심볼 클릭 이벤트 속성값(symbolClickJsFunction)에 자바스크립트 함수명(gongsiDataFunc)이 설정되었습니다. 자바스크립트 함수에서는 사용 가능한 차트 데이터들이 입력 파라메터 값으로 넘어오는데 이를 이용해서 원하는 정보를 차트상에 표시하는 것이 다음과 같이 가능합니다. 다음 차트에서 9 월 20 일과 9 월 23 일 사이의 데이터 포인트 상의 사각형 심볼(H, I, J, K)을 클릭하면 차트 하단의 박스에 자세한 가격 정보가 표시되는 것을 볼 수 있습니다.



See the CodePen 알메이트 차트 - 캔들스틱 차트 - 클릭 이벤트 처리

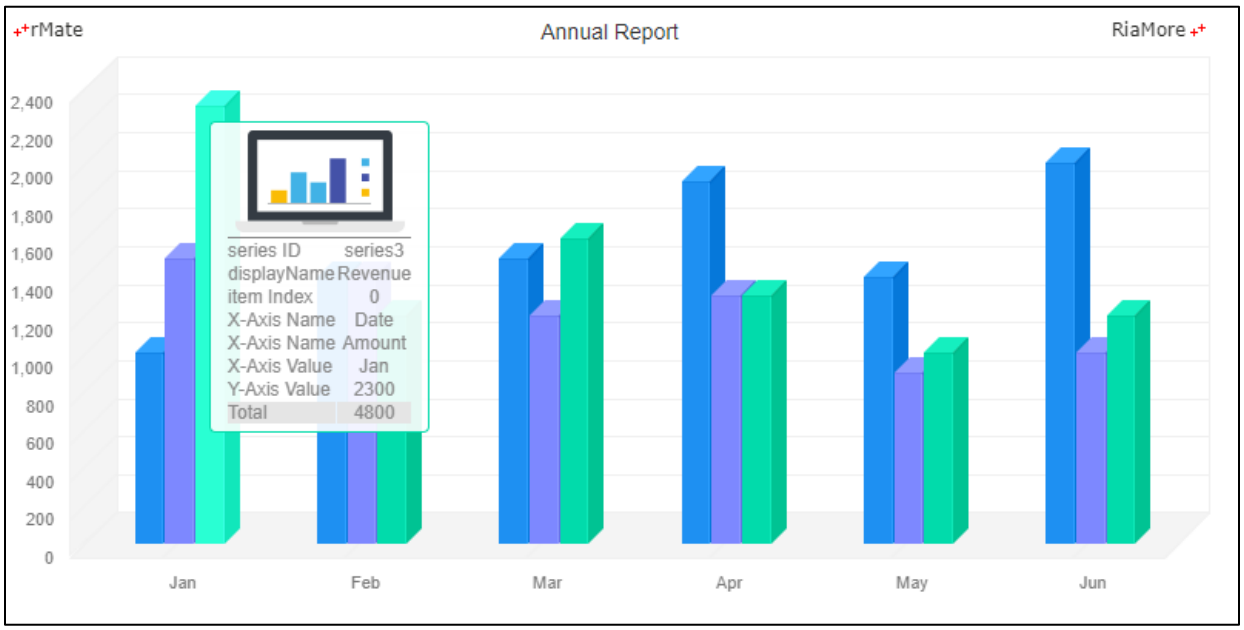
호버링(Hovering) 이벤트 처리하기

알메이트 차트에서는 사용자가 차트의 데이터 포인트에 마우스를 호버링(hovering)하면 기본적으로 툴팁(Tooltips)이 표시됩니다. 알메이트 차트에서 기본으로 제공하는 툴팁의 형태와 다른 형태의 툴팁을 차트 사용자에게 보여주기를 원할 경우에는 마우스 호버링 이벤트에 원하는 포맷의 툴팁을 만드는 자바스크립트 함수를 설정할 수 있습니다. 다음은 컬럼 차트의 컬럼에 사용자가 마우스를

```
<Column3DChart showDataTips="true" dataTipJsFunction="dataTipFunc">

function dataTipFunc(seriesId, seriesName, index, xName, yName, data, values) {
  return "<table cellpadding='0' cellspacing='1'>"
    +"<tr>"
      +"<td align='center' colspan='2' style='border-bottom:solid 1px #8b8b8b;'><img
       src='../rMateChartH5/Assets/Images/monitor.png'></td>"
    +"</tr><tr>"
      +"<td >series ID</td><td align='center'>" + seriesId + "</td>"
    ...
    +"</tr></table>";
}

function getSum(values) {
  var sum = 0;
  for(var v in values) {
    sum += Number(values[v]) || 0;
  }
  return sum;
}
```

호버링할 경우 HTML 포맷의 툴팁을 화면에 표시하는 자바스크립트 함수를 레이아웃에 설정한 코드 예제입니다.

다음 차트의 칼럼에 마우스를 옮기면 이미지와 값이 HTML 로 포맷된 형태의 툴팁이 표시됩니다.



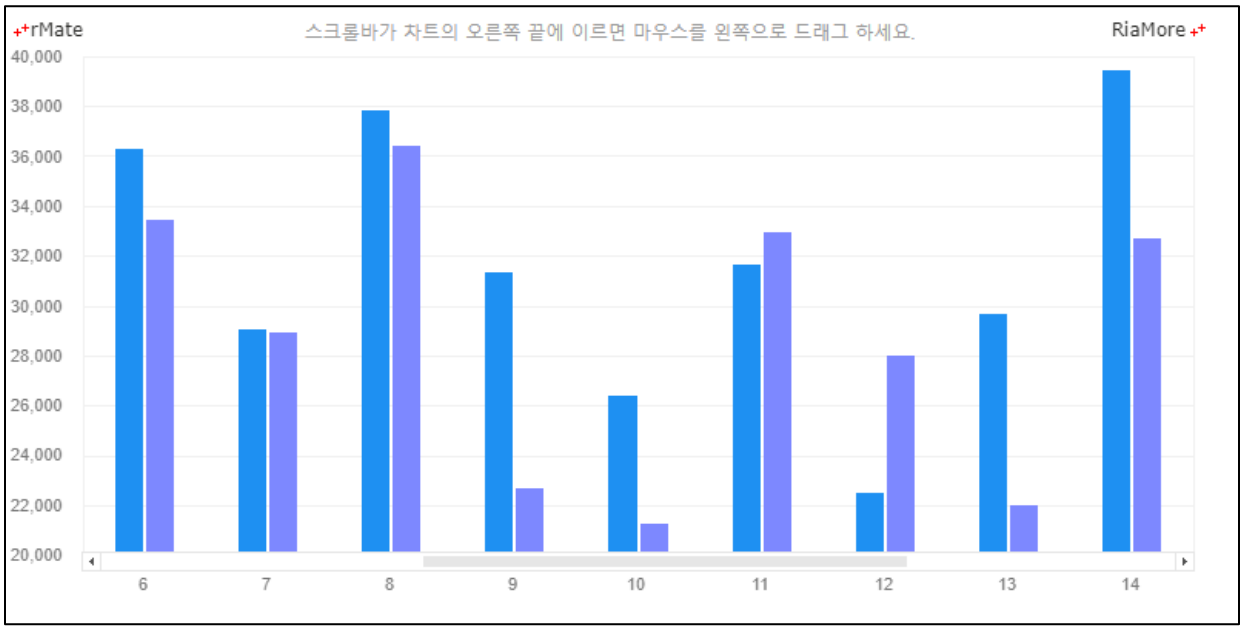
See the CodePen 알메이트 차트 - 사용자 정의 툴팁 표시 - 호버링 이벤트 처리

스크롤바 이벤트를 이용한 동적 자료 로드

컬럼 차트 혹은 라인 차트에서 표현해야 할 데이터의 수가 한 화면에 보여줄 수 있는 데이터 수보다 많을 경우, 스크롤바를 이용해서 차트를 표현하게 됩니다. 이 때 전체 데이터를 차트가 생성되는 시점에 한번에 로드한다면, 데이터를 처리하는 시간이 많이 소요될 것이고 결과적으로 차트가 화면에 표현되기까지 사용자가 오랜 시간을 기다려야하는 문제가 생길 수 있습니다. 어짜피 한 화면에서 사용자가 식별할 수 있는 데이터의 수는 한정되어 있기 때문에, 한 화면에 표현하기 적당한 데이터의 수 단위로 처리를 하는 것이 효율적일 것입니다. 그러기 위해서는 스크롤바의 위치가 끝 부분에 도달하면 자동으로 새로운 자료를 로드하는 것이 필요한데, 이렇게 데이터를 처리하는 방식을 레이지 로딩(Lazy Loading)이라고 합니다. 알메이트 차트에서는 스크롤바의 이벤트를 받아서 레이지 로딩을 처리할 수 있도록하는 lazyJsFunction 속성을 제공합니다. 다음은 컬럼 차트의 한 화면에 19 개의 데이터를 표현한다고 가정할 때 레이지 로딩을 처리하기 위한 레이아웃과 코드입니다.

```
<Column2DChart lazyJsFunction="lazyDataFunc">

var xhr, // Ajax object
index = 19; // Represents 19 data points on the screen
function lazyDataFunc(id) {
  var param = {};
  param.url = dataURL + (index + 1);
  param.success = function() {
    var data;
    if (xhr.readyState == 4 && xhr.status >= 200 && xhr.status < 300) {
      if (xhr.responseXML.xml)
        data = xhr.responseXML.xml;
      else
        data = xhr.responseXML;
      document.getElementById(id).addData(data);
      index += 20;
    }
  }
  ajax(param);
}
```

다음 컬럼 차트에서 최초에 차트가 생성된 후 스크롤바를 이동하면, 19 번째 데이터(스크롤바의 끝)가 표시되는 지점에서 자동으로 데이터가 로드되는 것을 확인할 수 있습니다.



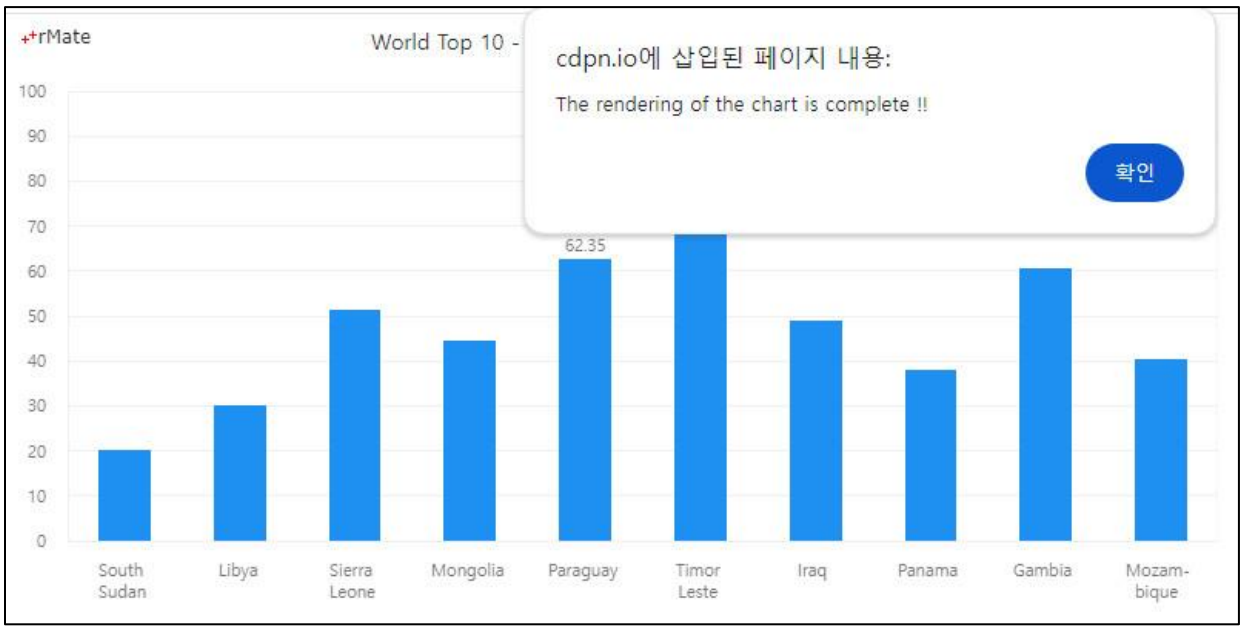
See the CodePen 알메이트 차트 - 레이지 로드

차트 로드 완료 이벤트 처리하기

경우에 따라서는 차트의 생성이 완료되자 마자 웹 페이지 내에서 원하는 처리를 해야할 경우가 있습니다. 알메이트 차트는 차트 렌더링이 완료되면 이벤트를 처리하는 기능을 제공합니다. 레이아웃 XML 의 차트 레벨 노드(예, <Column2DChart>, <Line2DChart> 등)에서 displayCompleteCallFunction 속성에 자바스크립트 함수를 지정하면 차트의 렌더링이 완료되자 마자 해당 자바스크립트 함수를

```
<Column2DChart displayCompleteCallFunction="displayCallFunction">

function displayCallFunction() {
  alert("The rendering of the chart is complete !!");
}
```

실행할 수 있습니다. 다음은 컬럼 차트의 렌더링이 완료되자 마자 완료 메시지를 자바스크립트 alert 함수를 이용해서 보여주는 코드 예제입니다.



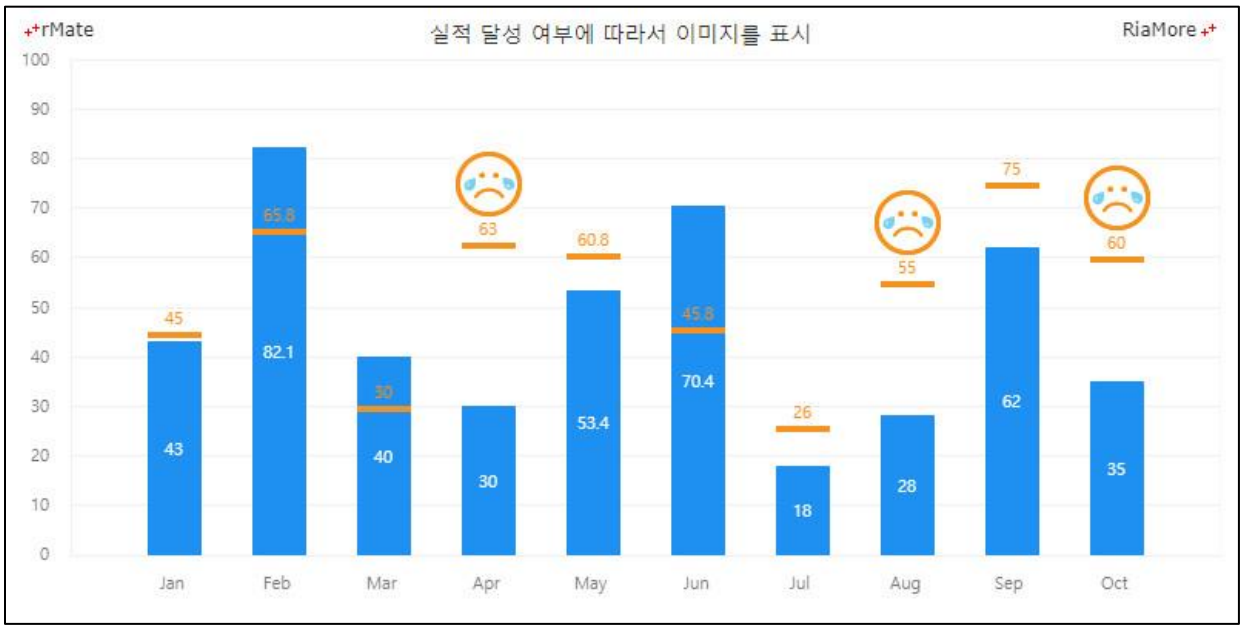
See the CodePen 알메이트 차트 - 차트 로드 완료 이벤트 처리하기

차트의 데이터 값(이벤트)에 따라서 필요한 정보 표현하기

차트에 표현되는 데이터 값에도 이벤트를 적용할 수 있는데 이는 알메이트 차트만의 강력한 이벤트 처리 기능입니다. 예를 들어 목표 대비 실적 차트에서 목표를 100% 이상 달성한 월에는 스마일 이미지를 표현하고, 실적값이 목표값의 80% 에 이르지 않으면 크라이 이미지를 표현할 수 있습니다. 그 이외에도 사용자의 요구사항과 개발자의 다양한 아이디어를 적용하여 인터렉티브한(Interactive)

차트를 생성할 수 있습니다. 다음은 목표 대비 실적 차트에서 데이터 이벤트를 적용하는 레이아웃과 코드입니다.

```
<VTarget2DResultSeries id="result" yField="Result" htmlJsFunction="userElementFunc">
...
<VTarget2DGoalSeries id="goal" yField="Goal" htmlJsFunction="userElementFunc">

function userElementFunc(id, index, data, values) {
  var src,
  goalValue = data.Goal,
  resultValue = data.Result;
  ...
}
```



See the CodePen 알메이트 차트 - 데이터 값(이벤트)에 따라서 필요한 정보 표현하기

알메이트 차트에서 사용 가능한 이벤트

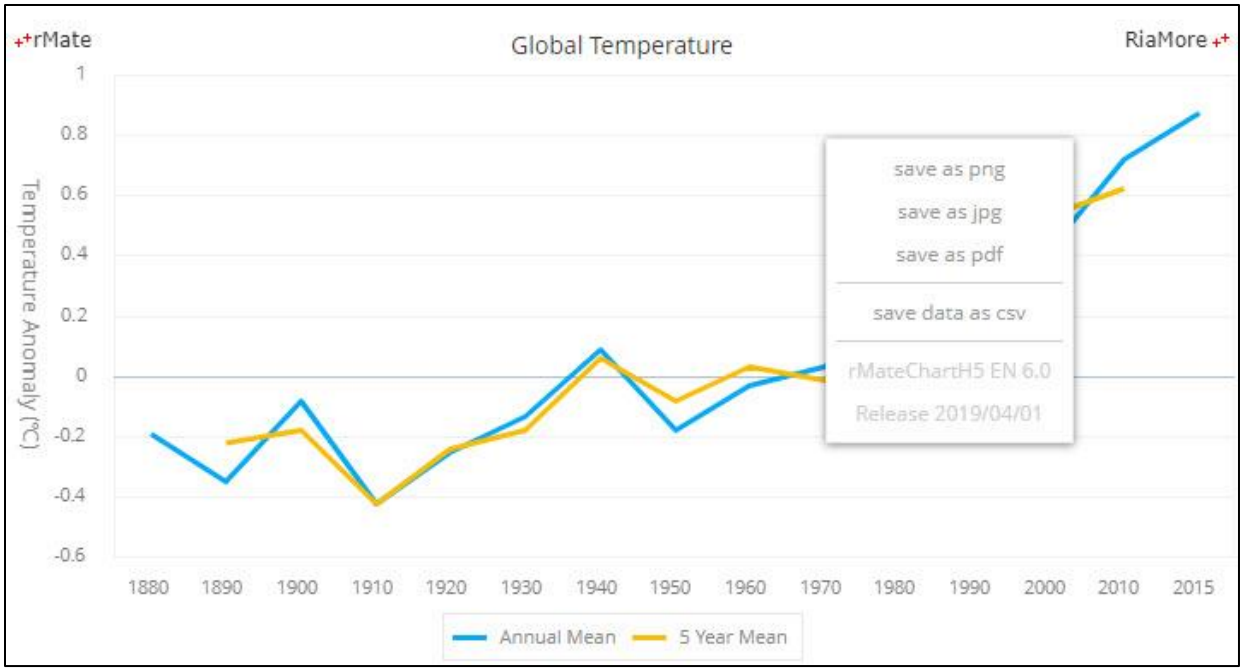
이벤트 타입	차트 요소	속성	XML 노드
클릭	축	axisClickJsFunction (카테시안 차트)	<Area2DChart>, <Bar2DChart>, <Bar2DWingChart>, <Bar3DChart>, <Bubble2DChart>, <Candlestick2DChart>, <Column2DChart>, <Column2DWingChart>, <Combination2DChart>,

			<Combination3DChart>, <Displayer>, <Displayer3D>, <ImageChart>, <ImageMatrixChart>, <Line2DChart>, <Matrix2DChart>, <MotionChart>, <Plot2DChart>, <Pyramid2DChart>, <RealTimeChart>, <TreeMapChart>, <Vector2DChart>, <WordCloudChart>
	차트	chartClickJsFunction	모든 차트 레벨 노드
	데이터 포인트	itemClickJsFunction	모든 차트 레벨 노드
	텍스트	itemClickJsFunction	<Caption>, <Label>, <Legend>, <SubCaption>, <SubLegend>
	심볼	symbolClickJsFunction	<CandleArea2DSeries>, <CandleLine2DSeries>, <Candlestick2DSeries>
더블 클릭	차트	chartDoubleClickJsFunction	모든 차트 레벨 노드
	데이터 포인트	itemDoubleClickJsFunction	모든 차트 레벨 노드
마우스 다운	데이터 포인트	itemDownJsFunction	모든 차트 레벨 노드
마우스 업	데이터 포인트	itemUpJsFunction	모든 차트 레벨 노드
마우스 무브	데이터 포인트	itemMoveJsFunction	모든 차트 레벨 노드
	축	axisDataTipJsFunction	카테시안 차트
	차트	chartOverJsFunction	모든 차트 레벨 노드

호버링 (마우스 오버)	데이터 포인트	dataTipJsFunction	카테시안 차트, <CircularGauge>, <HalfCircularGauge>, <HistoryRangeSelector>, <Navigator>
		itemOverJsFunction	모든 차트 레벨 노드
		innerTipJsFunction	<HCylinderGauge>, <HLinearGauge>, <VCylinderGauge>, <VLinearGauge>
		targetTipJsFunction	<HCylinderGauge>, <HLinearGauge>, <VCylinderGauge>, <VLinearGauge>
	범례 아이템	titleJsFunction	<Legend>, <SubLegend>
마우스 아웃	차트	chartOutJsFunction	모든 차트 레벨 노드
	데이터 포인트	itemOutJsFunction	모든 차트 레벨 노드
변경	값	valueChangeFunction	<CircularGauge>, <Gauge>, <HalfCircularGauge>, <LinearGauge>
	스크롤바	lazyJsFunction	카테시안 차트
	십자선	rangeUpdateJsFunction	<CrossRangeZoomer>
데이터 이벤트	값	htmlJsFunction	모든 시리즈 레벨 노드
렌더링	차트	displayCompleteCallFunction	모든 차트 레벨 노드

6.4 컨텍스트 메뉴

컨텍스트 메뉴 기능은 이전 버전에서 제공되던 사용자 정의 메뉴와 동일한 기능입니다. 이전 버전에서는 사용자 메뉴 항목을 자바스크립트로 작성해야 했지만, 버전 7.0 에서는 사용자가 마우스 오른쪽 버튼을 차트에 클릭하기만 하면 메뉴가 자동으로 표시됩니다. 다음은 마우스 오른쪽 버튼을 차트에 클릭하여 컨텍스트 메뉴를 표시한 화면입니다.



컨텍스트 메뉴 표시하지 않기

컨텍스트 메뉴 기능을 사용하지 않기 위해서는 rMateChartH5.create 함수의 세번째 인자(chartVars)의 useContextMenu 값을 false 로 설정합니다.

```
rMateChartH5.create("chart1", "chartHolder", "useContextMenu=false",
"100%", "100%");
```

컨텍스트 메뉴 내용 변경하기

컨텍스트 메뉴의 내용을 변경하기 위해서는 알메이트 차트가 제공하는 API 함수, setContextMenuItems() 를 호출하여야 합니다. 정의한 내용에 추가로 차트버전과 릴리즈 정보는 항상 표시됩니다.

```
rMateChartH5.calls("chart1",
	…,
	"setContextMenuItems" : [
		{name : "png 다운로드", clickFunc : function(){downloadPNG();}},
		{name : "jpg 다운로드", clickFunc : function(){downloadJPG();}}
	]
);

function downloadPNG(){
	…
};
function downloadJPG(){
	…
};
```

6.5 데이터 에디터

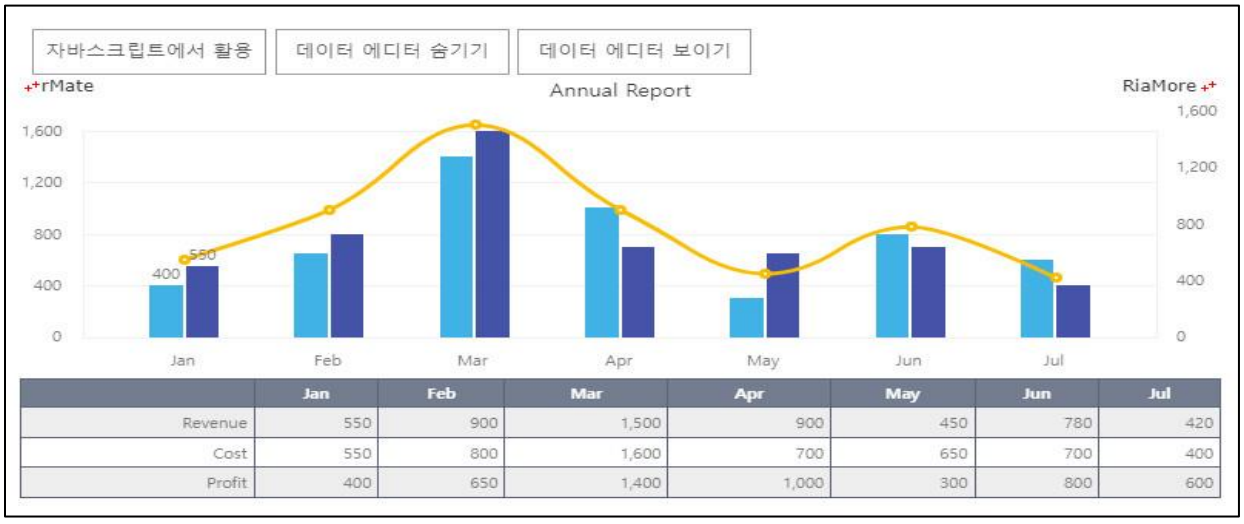
알메이트 차트는 차트의 하단에 차트에 표현된 데이터를 표시하는 테이블 형식의 데이터 에디터를 표시할 수 있습니다. 데이터 에디터에 표시된 값을 직접 수정할 수 있으며, 수정된 값은 차트에 즉시 반영되므로 시뮬레이션을 위한 작업에 유용하게 활용될 수 있습니다. 데이터 에디터 기능을 사용하기 위해서는 chartVars 변수에 useDataEditor 속성을 “true” 로 설정해야 하고, 알메이트 차트에서 사용하는 자바스크립트 에디터 라이브러리 파일에 대한 위치를 지정해 주어야 합니다. (예, rMateChartH5.dataEditorUrl = “../rMateChartH5/JS/editablegrid-2.0.1.js”) 자바스크립트 에디터 라이브러리 파일은 제품에 포함되어 있습니다. 데이터 에디터의 사용과 관련된 속성은 <DataEditor> 노드에 설정되며, <DataEditor> 노드는 <Options> 노드의 자식 노드로 정의되어야 합니다. 다음 표는 <DataEditor> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
alternatingItemColors	#16 진수 컬러 코드의 배열	테이블의 행의 배경색을 번갈아 가면서 칠할 경우 색을 설정합니다. 두 개 이상의 색을 지정해야 합니다.
borderColor	#16 진수 컬러 코드 표기 기본값: #000000	테이블의 테두리 선의 색을 지정합니다.
borderStyle	none, solid(*), inset, outset	테이블의 테두리 선의 스타일을 지정합니다.
editable	true(*), false	데이터 에디터의 값의 변경을 허용할지 여부를 설정합니다.
editorHeight	숫자 기본값: 80	데이터 에디터의 높이를 지정합니다.
headerColors	#16 진수 컬러 코드의 배열	테이블 헤더의 배경색을 설정합니다.
headerHeight	숫자 기본값: 22	데이터 에디터의 헤더의 높이를 지정합니다

reverseRow	true, false(*)	데이터셋에 나중에 정의된 데이터 값이 데이터 에디터 행에 먼저 표시될지 여부를 설정합니다.
showHeaders	true(*), false	데이터 에디터의 헤더를 표시할지 여부를 설정합니다.
showOnInit	true, false(*)	데이터 에디터를 차트가 생성되면 보이게할지 여부를 설정합니다.
styleName	텍스트	<Style> 노드에 설정되는 스타일명을 지정합니다.
textAlign	left, center(*), right	데이터 에디터의 셀에 표시되는 텍스트의 정렬 방식을 설정합니다.
useCloseButton	true(*), false	데이터 에디터 닫기 버튼을 표시할지 여부를 설정합니다.

다음은 데이터셋에 나중에 정의된 Revenue 값이 데이터 에디터의 제일 처음 행에 표시되도록 설정된 (reverseRow = “true”) 예제입니다.

```
var chartVars = "rMateOnLoadCallFunction=chartReadyHandler";
chartVars += "&useDataEditor=true";
rMateChartH5.dataEditorUrl = "../rMateChartH5/JS/editablegrid-2.0.1.js";

<DataEditor showOnInit="true" formatter="{numFmt}" useCloseButton="false"
       editorHeight="94" reverseRow="true" textAlign="right"/>

var chartData =
{"Month" : "Jan", "Profit" : 400, "Cost" : 550, "Revenue" : 550},
{"Month" : "Feb", "Profit" : 650, "Cost" : 800, "Revenue" : 900},
...
```

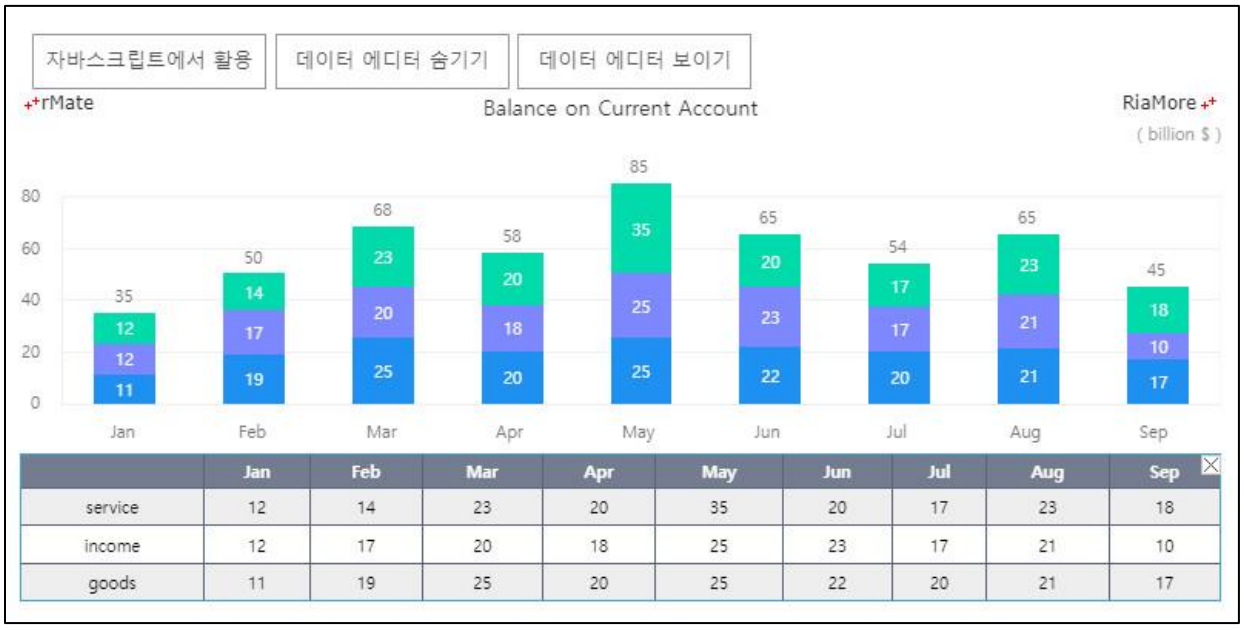


	Jan	Feb	Mar	Apr	May	Jun	Jul
Revenue	550	900	1,500	900	450	780	420
Cost	550	800	1,600	700	650	700	400
Profit	400	650	1,400	1,000	300	800	600

See the CodePen 알메이트 차트 - 데이터 에디터 (reverseRow = 'true' 설정)

다음은 데이터 에디터의 스타일을 지정하여 표현한 예제입니다. 아래 예제에서는 <DataEditor> 노드의 styleName 속성값이 “gridStyle” 로 지정되었고, “gridStyle” 이라는 이름의 스타일이 <Style> 노드에 정의되어 있습니다.

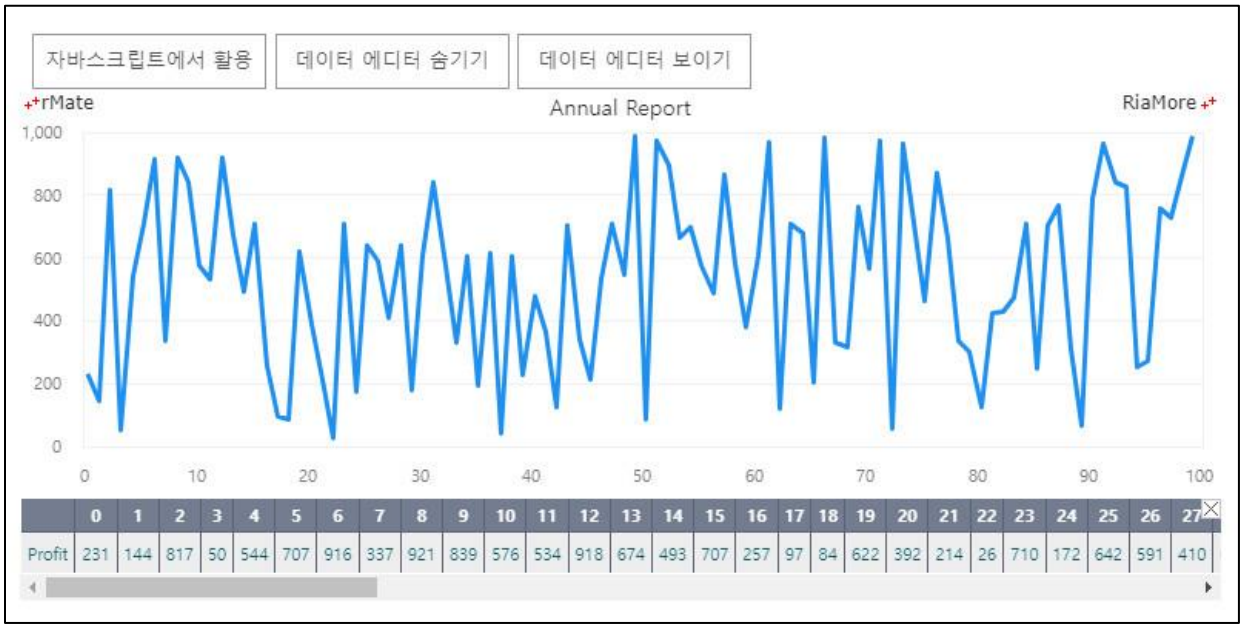
```
var chartVars = "rMateOnLoadCallFunction=chartReadyHandler";
chartVars += "&useDataEditor=true";
rMateChartH5.dataEditorUrl = "../rMateChartH5/JS/editablegrid-2.0.1.js";

<DataEditor showOnInit="true" formatter="{numFmt}" editorHeight="96"
        reverseRow="true" styleName="gridStyle"/>
...
<Style>
.gridStyle{color:#000000;alternatingItemColors:#f7f7f7,#ffffff;headerColors:#7dcad0,
        #7dcad0;headerStyleName:gridHeaderStyle;horizontalGridLines:true;horizontalG
        ridLineColor:#5C8484;headerLineColor:#44a4c8;selectionColor:#ADC1C1;rollOver
        Color:#CC9999;fontWeight:bold;verticalAlign:middle;verticalGridLineColor:#ab
        d6e6;fontWeight:normal;borderColor:#44a4c8;}.gridHeaderStyle{color:#ffffff;}
        }
</Style>
```



See the CodePen 알메이트 차트 - 데이터 에디터의 스타일 설정

다음은 데이터 에디터에 표시되는 데이터의 양이 많을 경우, 수평 스크롤바가 자동으로 표시되는 예제입니다.

```
var chartVars = "rMateOnLoadCallFunction=chartReadyHandler";
chartVars += "&useDataEditor=true";
rMateChartH5.dataEditorUrl = "../rMateChartH5/JS/editablegrid-2.0.1.js";

<DataEditor showOnInit="true" editorHeight="67" reverseRow="true"
        styleName="gridStyle"/>
<Style>
.gridStyle{color:#095c69;alternatingItemColors:#efeeee,#efeeee;headerColors:#7dcad0,
        #7dcad0;headerStyleName:gridHeaderStyle;horizontalGridLines:true;horizontalG
        ridLineColor:#ffffff;headerLineColor:#ffffff;fontWeight:bold;fontSize:12;ver
        ticalAlign:middle;verticalGridLineColor:#ffffff;fontWeight:normal;borderColo
        r:#ffffff;}.gridHeaderStyle{color:#fefefe;}}
</Style>
```



See the CodePen 알메이트 차트 - 데이터 에디터에 스크롤바 표시

6.6 포멧터 사용하기

알메이트 차트는 통화(<CurrencyFormatter> 노드), 날짜(<DateFormatter> 노드) 그리고 숫자(<NumberFormatter> 노드)에 대한 포맷을 사용자가 원하는 형태로 표시할 수 있도록 설정하는 기능을 제공합니다. 포맷터 노드를 <rMateChart> 노드의 자식 노드로 정의하고 포맷터 노드의 id 속성값을 지정한 후, 포맷터의 적용이 필요한 노드(예, 숫자 축의 레이블에 포맷터를 적용한다면 <LinearAxis> 노드)의 formatter 속성에 포맷터 노드의 id 속성값을 지정하여 포맷터를 적용할 수 있습니다.

통화 포맷터

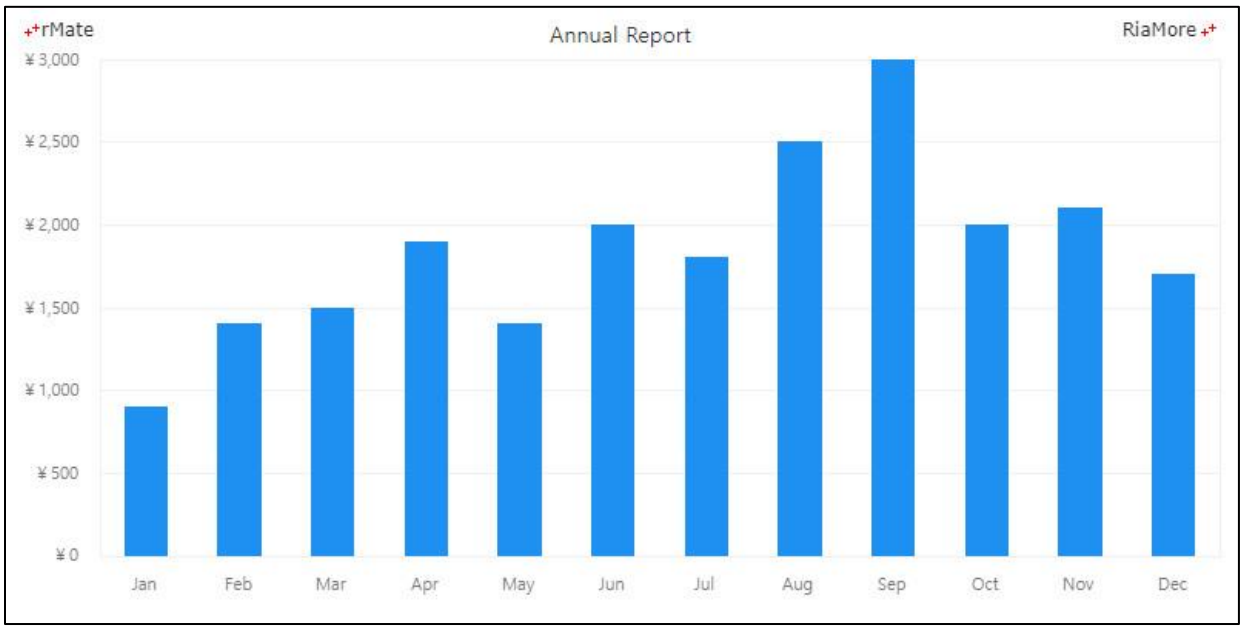
통화 포맷터(<CurrencyFormatter> 노드)를 통해서 사용자가 원하는 형태의 통화 표시를 할 수 있습니다. 다음 표는 <CurrencyFormatter> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
alignSymbol	left(*), right	currencySymbol 속성에 지정된 통화 기호를 숫자를 기준으로 어디에 표시할지를 설정합니다.
currencySymbol	텍스트	통화 기호를 지정합니다.
decimalSeparatorTo	텍스트 기본값: dot (.)	소수점 구분 기호를 지정합니다.
id	텍스트	<CurrencyFormatter> 노드의 식별자를 지정합니다.
precision	숫자 기본값: -1	소수점 구분 기호의 오른쪽에 표시될 숫자의 최대 자리수를 설정합니다. -1: 표시 가능한 모든 소수점이하 숫자를 표시합니다.
rounding	down, up, nearest, none(*)	반올림 방법을 설정합니다.

thousandsSeparatorTo	텍스트 기본값: comma (,)	천단위 구분자를 지정합니다.
useNegativeSign	true(*), false	음수 표시를 위해서 마이너스 기호(-)를 표시할지 여부를 설정합니다.
useThousandsSeparator	true(*), false	천단위 구분자를 표시할지 여부를 설정합니다.

다음은 세로 축(Y 축)의 레이블에 통화 포맷터를 적용한 예제입니다.

```
<CurrencyFormatter id="fmt" currencySymbol="￥" alignSymbol="left"/>
...
<verticalAxis>
  <LinearAxis id="vAxis" interval="500" formatter="{fmt}"/>
</verticalAxis>
```



See the CodePen 알메이트 차트 - 세로 축(Y 축)의 레이블에 통화 포맷터 적용

날짜 포맷터

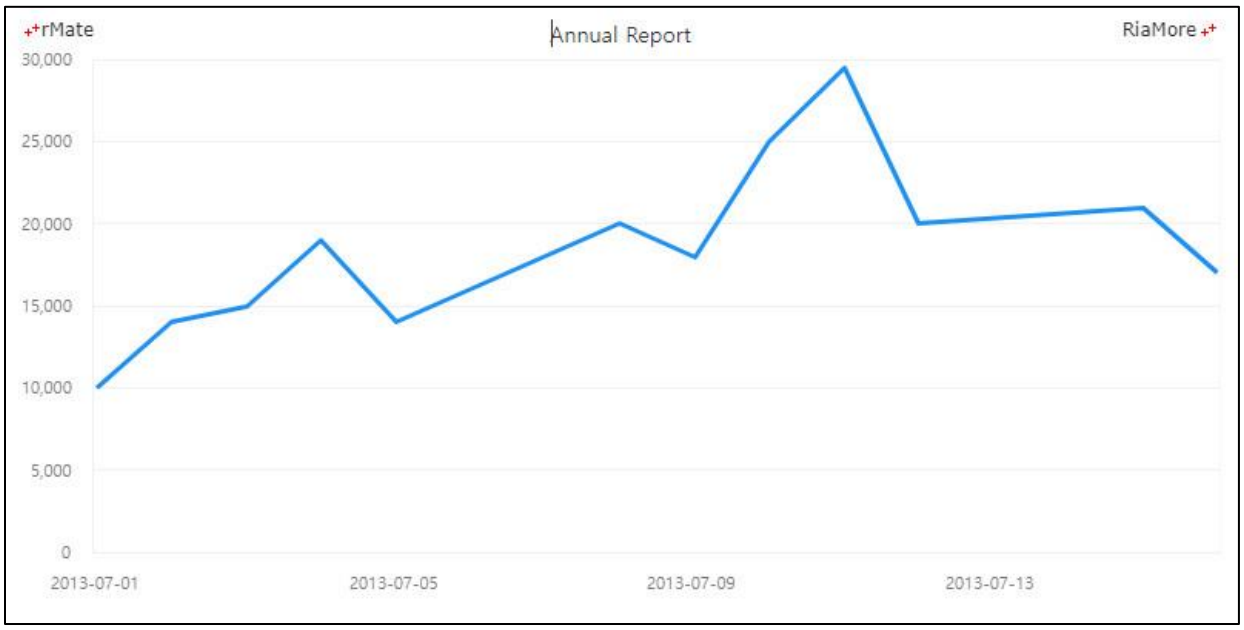
날짜 포맷터(<DateFormatter> 노드)를 통해서 사용자가 원하는 형태의 날짜 표시를 할 수 있습니다. 다음 표는 <DateFormatter> 노드의 주요 속성에 대한 설명입니다.

<table>
<tr><th>속성명</th><th>유효값 (*: 기본값)</th><th>설명</th></tr>
<tr><td>formatString</td><td>텍스트</td><td>표시할 날짜 형식을 설정합니다.

<table>
<tr><td>Y</td><td>연(Year)</td><td>YY=05, YYYY=2005</td></tr>
<tr><td>M</td><td>월(Month)</td><td>M=7, MM=07, MMM=Jul</td></tr>
<tr><td>D</td><td>일(Day)</td><td>D=4, DD=05</td></tr>
<tr><td>H</td><td>시(Hour)</td><td>H=1, HH=01</td></tr>
<tr><td>N</td><td>분(Minute)</td><td>N=1, NN=01</td></tr>
<tr><td>S</td><td>초(Second)</td><td>SS=30</td></tr>
</table>
</td></tr>
<tr><td>id</td><td>텍스트</td><td><DateFormatter> 노드의 식별자를 지정합니다.</td></tr>
</table>

다음은 가로 축(X 축)의 레이블에 날짜 포맷터를 적용한 예제입니다.

```
<DateFormatter id="dateFmt" formatString="YYYY-MM-DD"/>
...
<horizontalAxis>
  <DateTimeAxis id="hAxis" formatter="{dateFmt}" dataUnits="days" labelUnits="days"
       interval="4" alignLabelsToUnits="false" displayLocalTime="true"/>
</horizontalAxis>
```



See the CodePen 알메이트 차트 - 가로 축(X 축)의 레이블에 날짜 포맷터 적용

숫자 포맷터

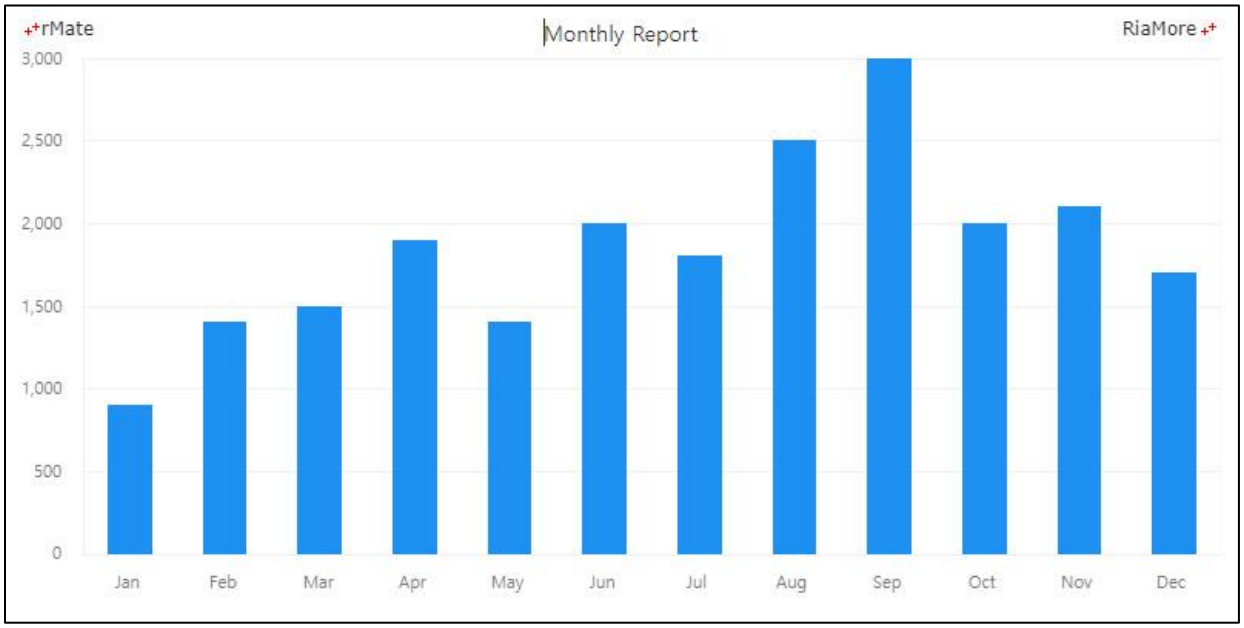
숫자 포맷터(<NumberFormatter> 노드)를 통해서 사용자가 원하는 형태의 숫자 표시를 할 수 있습니다. 다음 표는 <NumberFormatter> 노드의 주요 속성에 대한 설명입니다.

속성명	유효값 (*: 기본값)	설명
decimalSeparatorTo	텍스트 기본값: dot (.)	소수점 구분 기호를 지정합니다.
id	텍스트	<NumberFormatter> 노드의 식별자를 지정합니다.
precision	숫자 기본값: -1	소수점 구분 기호의 오른쪽에 표시될 숫자의 최대 자리수를 설정합니다. -1: 표시 가능한 모든 소수점이하 숫자를 표시합니다.
rounding	down, up, nearest, none(*)	반올림 방법을 설정합니다.

thousandsSeparatorTo	텍스트 기본값: comma (,)	천단위 구분자를 지정합니다.
useNegativeSign	true(*), false	음수 표시를 위해서 마이너스 기호(-)를 표시할지 여부를 설정합니다.
useThousandsSeparator	true(*), false	천단위 구분자를 표시할지 여부를 설정합니다.

다음은 세로 축(Y 축)의 레이블에 천단위 구분자를 표시하기 위해서 숫자 포맷터를 적용한 예제입니다.

```
<NumberFormatter id="numfmt" useThousandsSeparator="true"/>
...
<verticalAxis>
  <LinearAxis interval="500" formatter="{numfmt}"/>
</verticalAxis>
```



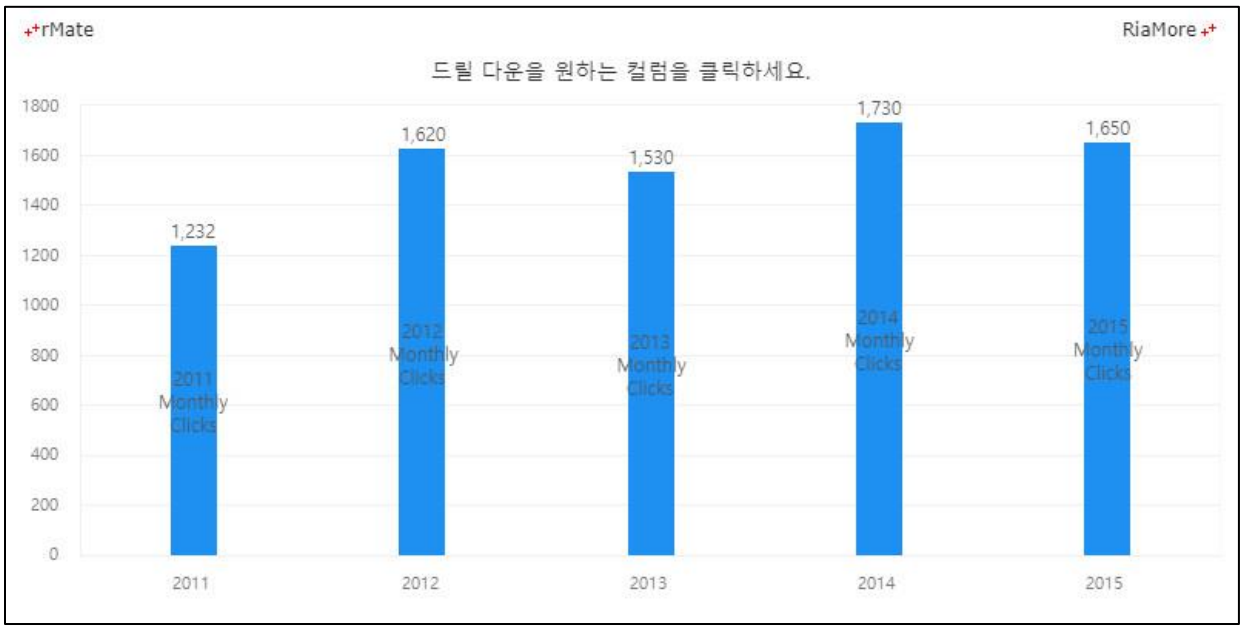
See the CodePen 알메이트 차트 - 세로 축(Y 축)의 레이블에 천단위 구분자 표시

6.7 드릴 다운

알메이트 차트는 차트에 표현된 데이터를 더 자세히 분석할 수 있는 드릴 다운 기능을 제공합니다. 예를 들어, 연도별 이익을 표현하는 차트에서 특정한 연도를 클릭하면 클릭된 연도의 월별 이익 현황을 볼 수 있고, 또한 특정 월을 클릭하면 해당 월의 요일별 이익 현황을 볼 수 있는 기능과 같은 것입니다. 드릴 다운 기능은 차트 레벨 노드(예, <Column2DChart> 노드)의 itemClickJsFunction 속성에 드릴 다운 기능을 수행하는 자바스크립트 함수를 지정함으로써 가능합니다. 자바스크립트 함수에서는 드릴 다운시 보여줄 데이터셋을 설정하고 알메이트 차트가 제공하는 API 함수, setDataDrilldown() 를 호출하여야 합니다.

다음은 연도별 이익을 표현하는 컬럼 차트에 드릴 다운 기능을 구현한 예제입니다. 컬럼을 클릭하면 클릭된 컬럼의 월별, 요일별 자료로 드릴 다운이 실행됩니다. 드릴 다운이 실행된 상태에서 이전 데이터로 되돌아가기 위해서는 차트의 좌측 상단에 표시되는 Reset 버튼(초기 상태로 돌아 감) 혹은 Back 버튼(바로 전 상태로 돌아 감)을 클릭합니다.

```
<Column2DChart showDataTips="true" itemClickJsFunction="itemClick"
        maxColumnWidth="15" buttonMode="true">
...
function itemClick(seriesId, displayText, index, data, values) {
  var data, depth;
  depth = document.getElementById("chart1").getDrillDownDepth();
  if(depth == 2){
    alert("No data has been found.");
    return;
  }
  document.getElementById("chart1").setDataDrilldown(makeData(depth));
}

function makeData (depth) {
  var i = 0, n, arr = [], obj,
  categoryDatas = [
    ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov",
       "Dec"],
    ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
  ];
  n = categoryDatas[depth].length;
  for(i ; i < n ; i += 1){
    obj = {};
    obj.Month = categoryDatas[depth][i];
    obj.Profit = Math.round(Math.random() * 1500);
    arr.push(obj);
  }
  return arr;
}
```

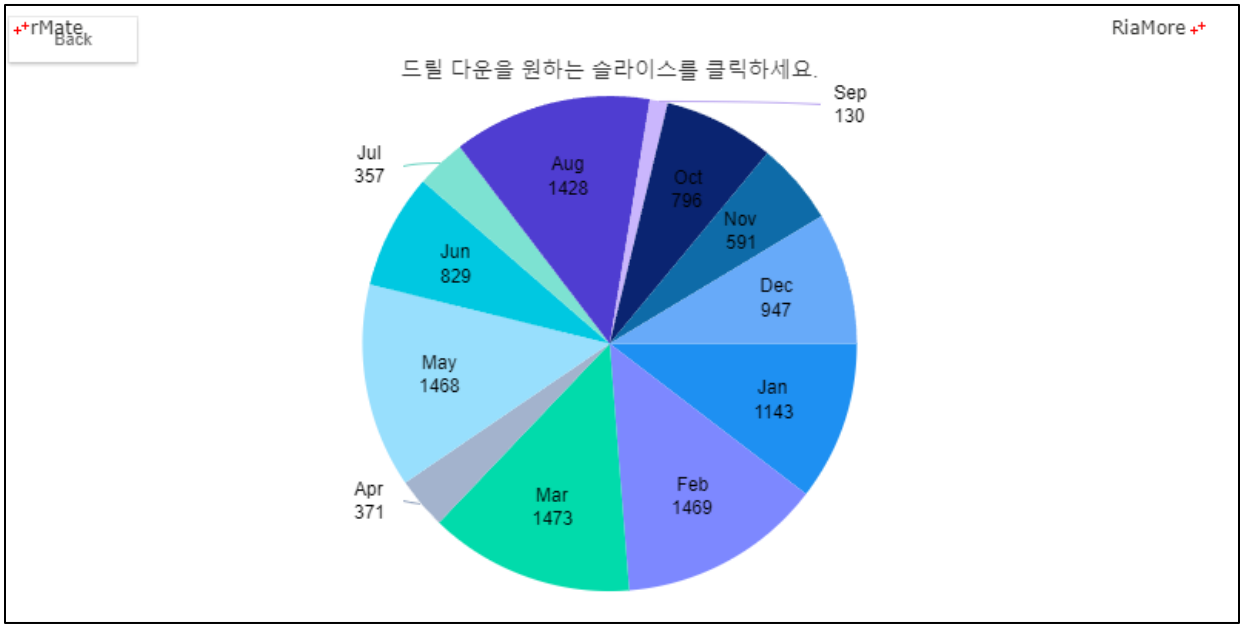


See the CodePen 알메이트 차트 - 컬럼 차트에서 드릴 다운 기능

다음은 연도별 이익을 표현하는 파이 차트에 드릴 다운 기능을 구현한 예제입니다. 파이 조각을 클릭하면 클릭된 조각의 월별, 요일별 자료로 드릴 다운이 실행됩니다. 드릴 다운이 실행된 상태에서

```
<Pie2DChart itemClickJsFunction="itemClick" buttonMode="true" explodable="false">
...
function itemClick(seriesId, displayText, index, data, values) {
  var data, depth;
  depth = document.getElementById("chart1").getDrillDownDepth();
  if(depth == 2){
    alert("No data has been found.");
    return;
  }
  document.getElementById("chart1").setDataDrilldown(makeData(depth));
}

function makeData (depth) {
  var i = 0, n, arr = [], obj,
  categoryDatas = [
    ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov",
       "Dec"],
    ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
  ];
  n = categoryDatas[depth].length;
  for(i ; i < n ; i += 1){
    obj = {};
    obj.Month = categoryDatas[depth][i];
    obj.Profit = Math.round(Math.random() * 1500);
    arr.push(obj);
  }

  return arr;
}
```

이전 데이터로 되돌아가기 위해서는 차트의 좌측 상단에 표시되는 Reset 버튼(초기 상태로 돌아 감) 혹은 Back 버튼(바로 전 상태로 돌아 감)을 클릭합니다.



See the CodePen 알메이트 차트 - 파이 차트에서 드릴 다운 기능

6.8 차트 내보내기

알메이트 차트는 화면에 표현된 차트의 모양 그대로 이미지 혹은 pdf 파일로 내보내기 하거나 차트의 데이터만 csv 포맷으로 내보내기 할 수 있습니다. 내보내기 결과에 의해 파일이 저장되는 장소는 사용자의 로컬 PC 혹은 서버 모두 가능합니다.

PC 에 내보내기

화면에 표현된 차트를 사용자의 PC 에 내보내기 하는 것은 다음 두 가지 방법으로 가능합니다.

- 컨텍스트 메뉴 : 마우스 오른쪽 버튼을 차트에 클릭하고 내보내기를 원하는 포맷의 파일을 선택합니다. 알메이트 차트에서는 png, jpg, pdf 포맷으로 차트의 모양을 그대로 내보내기할 수 있으며, csv 포맷으로 차트의 데이터를 내보내기할 수 있습니다.
- 자바스크립트 : 자바스크립트 함수를 이용해서 내보내기하는 API 함수(rMateChartH5.downloadToLocal)를 호출합니다. 아래 코드는 png 와 pdf 포맷으로 차트를 내보내기하는 예제입니다. rMateChartH5.downloadToLocal() 함수의 네 번째 인자는 이미지 파일 내보내기를 원할 경우 saveAsImage() 함수를 호출하고, pdf 파일 내보내기를 원할 경우에는 getSnapshot() 함수를 호출해야 합니다. 예제에서 사용된 서버 스크립트(downloadLocal.jsp) 파일은 제품의 Samples\SnapshotServerSamples 디렉토리에서 찾으실 수 있습니다.

```
// png 포맷으로 내보내기
rMateChartH5.downloadToLocal("rMateChartH5", "png",
  "http://demo.riamore.net/chartTest/snapshotSample/downloadLocal.jsp",
  function() {
    return document.getElementById("chart1").saveAsImage();
});

// pdf 포맷으로 내보내기
rMateChartH5.downloadToLocal("rMateChartH5", "pdf",
  "http://demo.riamore.net/chartTest/snapshotSample/downloadLocal.jsp",
  function() {
    return document.getElementById("chart1").getSnapshot("jpeg");
  },
  (function(){
    var holder = document.getElementById("chartHolder"),
    width = holder.offsetWidth, height = holder.offsetHeight;
    return {portrait : width < height, size : {width : width, height : height}
  };
})());
```

서버에 내보내기

화면에 표현된 차트를 서버에 내보내기 하는 방식은 차트의 getSnapshot() 함수를 호출하여 차트 이미지를 base64 형태로 인코딩한 데이터를 서버로 전송(submit)한 후, 서버 스크립트를 통해서 처리하는 방식입니다. 이러한 작업은 서버를 통한 배치(batch) 작업이 필요한 경우 유용하게 활용될 수 있습니다. 다음은 차트를 서버에 내보내기 하는 기능에 대한 예제입니다. 예제에서 사용된 서버 스크립트(getImageSnapshot.jsp) 파일은 제품의 Samples\SnapshotServerSamples 디렉토리에서 찾으실 수 있습니다.

```
function GetSnapshot() {
  var base64src = document.getElementById("chart1").getSnapshot();
  var data = document.getElementById("data");
  var extension = document.getElementById("extension");
  extension.value = "png";
  data.value = base64src;
  document.getElementById("sumForm").submit();
}

<form id="sumForm" name="sumForm"
       action="http://demo.riamore.net/chartTest/snapshotSample/getImageSnapshot.ph
       p" method="post">
  <input type="hidden" id="data" name="data" type="text"/>
  <input type="hidden" id="extension" name="extension" value="png"/>
</form>
```

6.9 API 함수 활용

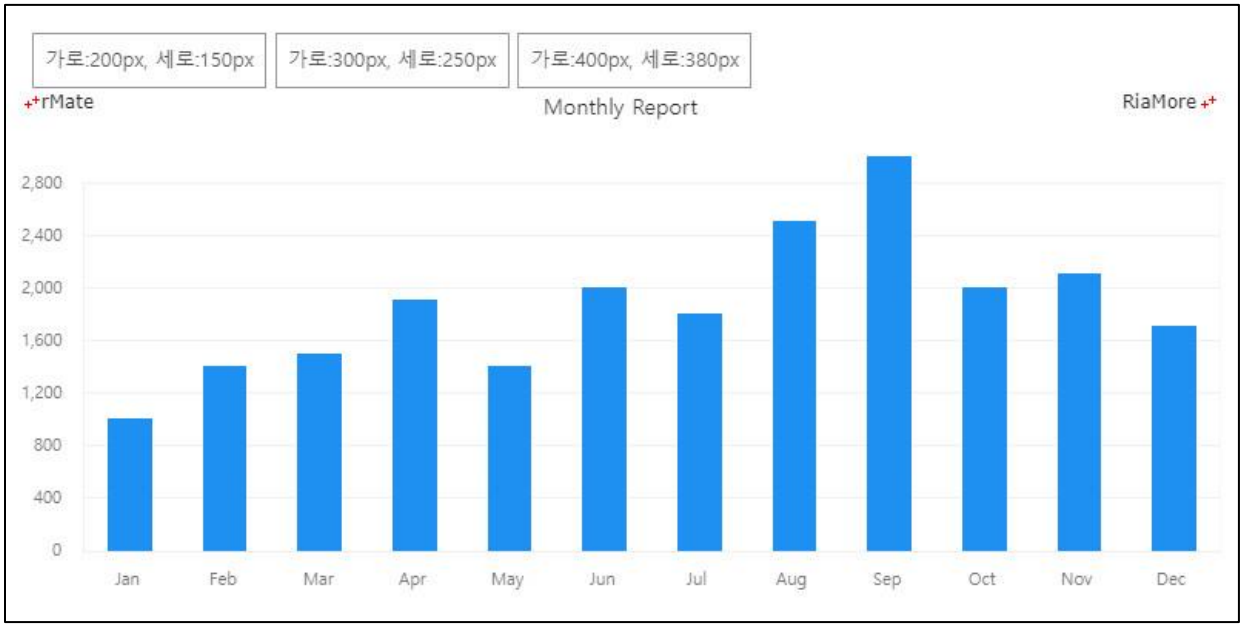
알메이트 차트에서 제공하는 API 함수는 차트 개발시 여러 가지 목적으로 유용하게 활용될 수 있습니다. API 함수를 호출하는 방식은 자바스크립트의 차트 엘리먼트를 통해서 해당 API 함수를 직접 호출하는 방식입니다.

```
document.getElementById("Chart Identifier").API Function Name();
```

차트의 크기 재설정

차트의 크기는 resize() 함수를 호출하여 조정할 수 있습니다. 다음은 차트가 위치한 의 크기를 변경한 후, resize() 함수를 호출하는 예제 코드입니다. 예제 코드에서 “chartHolder” 는 차트가 생성되는 부모 의 식별자이고, “chart1” 은 차트의 식별자입니다.

```
function changeChart(w, h) {
  var ch = document.getElementById("chartHolder")
  ch.style.width = w+"px";
  ch.style.height = h+"px";
  document.getElementById("chart1").resize();
}
```



See the CodePen 알메이트 차트 - 차트의 크기 재설정

알메이트 차트에서 제공하는 API 함수 리스트는 API 함수를 참조하십시요.

동적 레이아웃 생성과 데이터셋 적용

차트가 처음 생성된 이후에 데이터셋의 변경이 발생하여 변경된 데이터셋을 적용하여 차트를 다시 생성하고자 한다면 setData() 혹은 setDataURL() 함수를 호출합니다. 그리고 레이아웃의 변경이 발생하여 변경된 레이아웃을 적용하여 차트를 다시 생성하고자 한다면 setLayout() 혹은 setLayoutURL() 함수를 호출합니다. 다음은 좌측 상단의 버튼을 클릭하면 해당 유형의 차트를 생성하는 레이아웃을 적용하여 동적으로 차트를 생성하는 예제입니다.

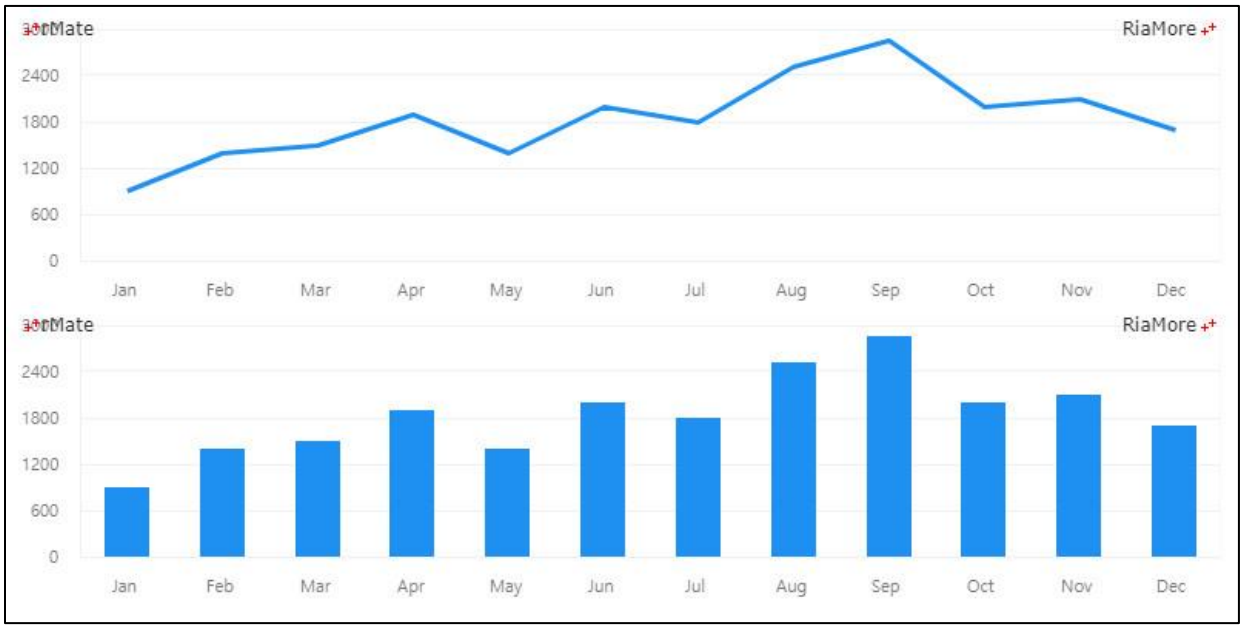
```
function changeColumn() {
  document.getElementById("chart1").setLayoutURL(layoutURL);
}
function changeLine() {
  document.getElementById("chart1").setLayout(layoutStr);
}
<div class="button button_top" onclick="changeColumn()">Column Chart</div>
<div class="button button_top" onclick="changeLine()">Line Chart</div>
```

See the CodePen 알메이트 차트 - 동적 레이아웃과 데이터셋 적용

한 페이지에 여러 개의 차트 생성

여러 개의 차트를 하나의 페이지에 생성하기 위해서는 rMateChartH5.create() 함수를 원하는 횟수 만큼 호출하면 됩니다. 이 때 함수의 파라미터 값으로 지정되는 차트의 식별자와 차트가 생성될 의 식별자는 반드시 구분이 되어야 하고, 차트가 생성될 는 HTML 문서에 구성이 되어 있어야 합니다. chartVars 변수에 설정되는 차트 생성 작업을 위한 준비가 완료되면 호출되는 함수는 차트 개발자의 편의에 따라서 각 차트별로 별도로 만들 수도 있고, 하나로 구현할 수도 있습니다. 다음은 하나의 chartVars 변수를 이용(차트 생성 작업을 위한 준비가 완료되면 호출되는 함수도 하나를 이용)하여 두 개의 차트를 한 화면에 구현하는 예제입니다.

```
var chartVars = "rMateOnLoadCallFunction=chartReadyHandler";
rMateChartH5.create("chart1", "chartHolder1", chartVars, "100%", "100%");
rMateChartH5.create("chart2", "chartHolder2", chartVars, "100%", "100%");

function chartReadyHandler(id) {
  document.getElementById(id).setLayout(id == "chart1" ? layoutStr : layoutStr2);
  document.getElementById(id).setData(chartData);
}
<div id="chartHolder1" style="height:200px;"/>
<div id="chartHolder2" style="width:100%;height:200px;float:left;"/>
```

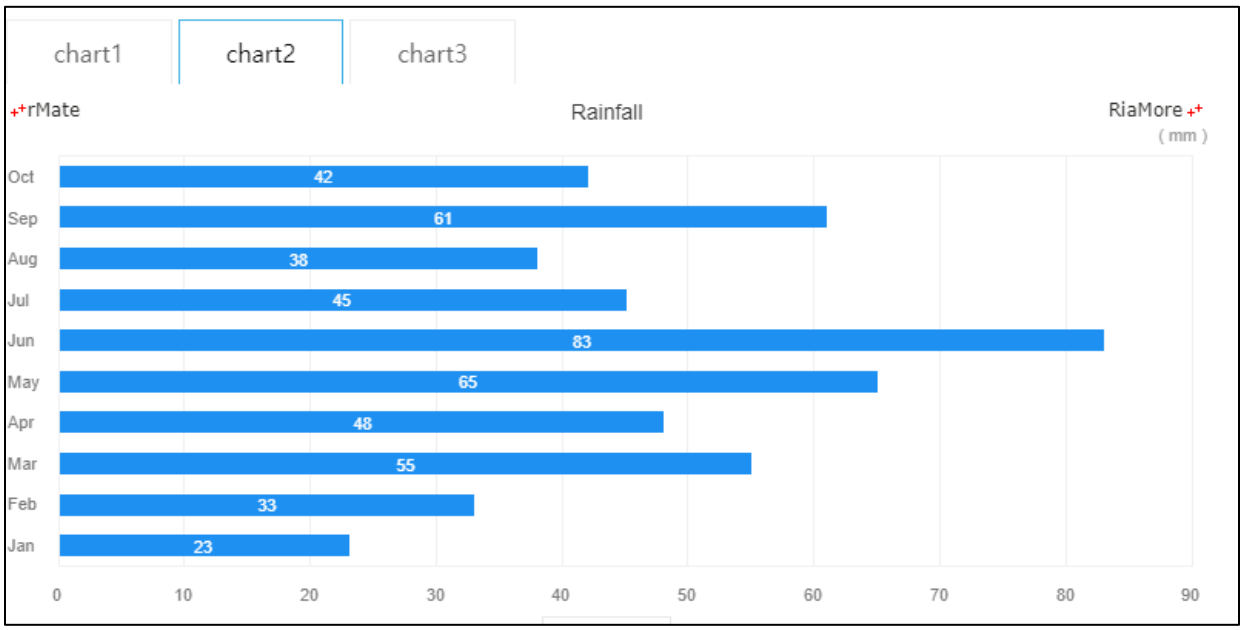


See the CodePen 알메이트 차트 - 한 페이지에 여러 개의 차트 생성

jQuery 탭 연동

한 페이지에 여러 개의 차트를 생성한 후 jQuery 탭과 연동하여 사용자가 탭을 클릭하면 해당 차트를 화면에 표시하는 기능을 구현할 수 있습니다. 다음은 이에 대한 예제 코드입니다.

```
var chartVars = "rMateOnLoadCallFunction=chartReadyHandler";
rMateChartH5.create("chart1", "chartHolder1", chartVars, "100%", "100%");

function chartReadyHandler(id) {
  document.getElementById(id).setLayout(window["layoutStr" + id.replace(/\D/g,
       "")]);
  document.getElementById(id).setData(chartData);
  document.getElementById(id).resize();
}

$(document).ready(function() {
  $("#jqueryTabs").tabs();
  $("#jqueryTabs > ul").click(function(event) {
    var i, a, as, content,
    tagName = event.target.tagName.toLowerCase();
    if (tagName == "a" || tagName == "li") {
      as = $("#jqueryTabs a");
      for (i = 0 ; i < as.length ; i += 1) {
        a = as[i];
        $(a.parentNode).removeClass("li_select");
        content = a.innerHTML;
        if(content == event.target.innerHTML) {
          $(a.parentNode).addClass("li_select");
          if(!document.getElementById(content))
            rMateChartH5.create(content, "chartHolder" + content.replace(/\D/g, ""),
       chartVars, "100%", "100%");
        }
      }
    }
  });
});

<div id="jqueryTabs">
  <ul>
    <li class="li_select"><a href="#chartHolder1">chart1</a></li>
    <li><a href="#chartHolder2">chart2</a></li>
    <li><a href="#chartHolder3">chart3</a></li>
  </ul>
  <div id="chartHolder1" style="width:850px;height:400px;"></div>
  <div id="chartHolder2" style="width:850px;height:400px;"></div>
  <div id="chartHolder3" style="width:850px;height:400px;"></div>
</div>
```



See the CodePen 알메이트 차트 - jQuery 탭 연동

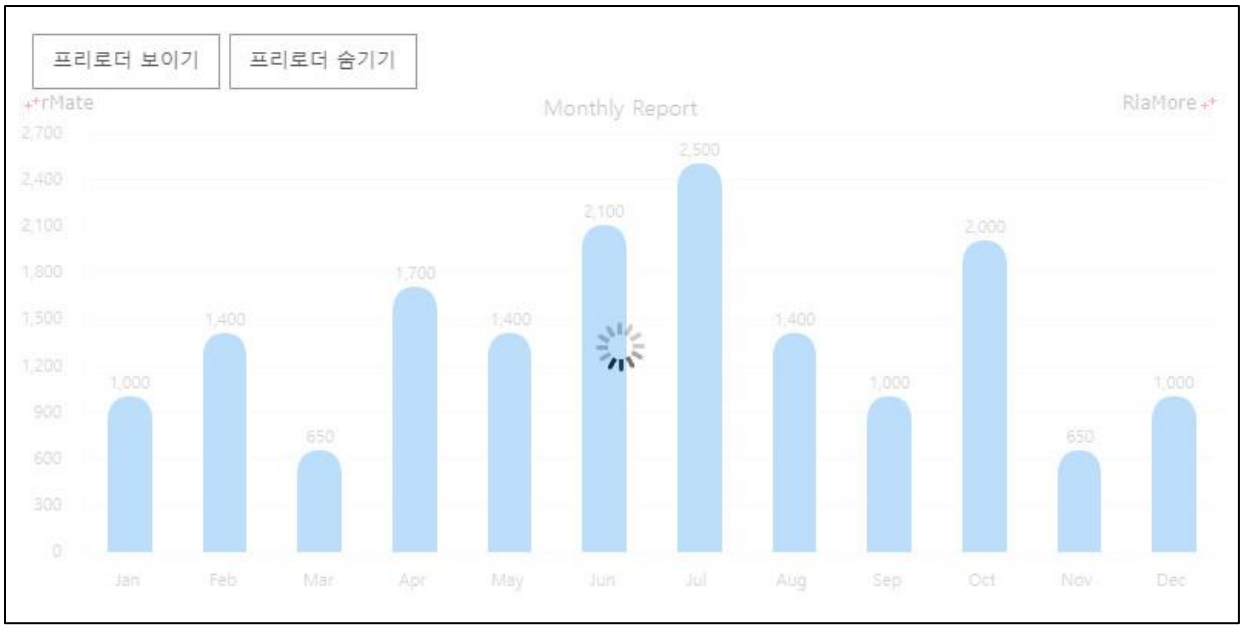
위 예제 코드를 살펴보면, rMateChartH5.create("chart1", "chartHolder1", chartVars, "100%", "100%"); 함수의 실행에 의해서 “chart1” 이 생성되고, “chart2” 와 “chart3” 는 $(document).ready(function() 함수 내에서 rMateChartH5.create(content, "chartHolder" + content.replace(/\D/g, ""), chartVars, "100%", "100%"); 함수의 호출에 의해서 생성되도록 구현되어 있습니다. 탭 클릭시에 활성화될 차트는 $("#jqueryTabs > ul").click(function(event) 함수에서 설정됩니다.

프리로더

showAdditionalPreloader() 와 removeAdditionalPreloader() 함수를 이용하여 프리로더 기능을 차트에 구현할 수 있습니다.

다음은 이에 대한 예제입니다. 아래 예제에서는 차트가 생성되는 순간에 프리로더 이미지가 보여지고, 좌측 상단의 버튼을 클릭하여 프리로더 이미지의 보이기와 감추기를 시뮬레이션할 수 있습니다.

```
function chartReadyHandler(id) {
  document.getElementById("chart1").showAdditionalPreloader();
  document.getElementById(id).setLayout(layoutStr);
  document.getElementById(id).setData(chartData);
  setTimeout(function(){
    document.getElementById("chart1").removeAdditionalPreloader();
  }, 1000);
}

function showPreloader() {
  document.getElementById("chart1").showAdditionalPreloader();
}

function hidePreloader() {
  document.getElementById("chart1").removeAdditionalPreloader();
}

<div class="button button_top" onclick="showPreloader()" >Show Preloader</div>
<div class="button button_top" onclick="hidePreloader()" >Hide Preloader</div>
```



See the CodePen 알메이트 차트 - 프리로더 설정

Null 데이터

차트에 표현할 데이터가 하나도 존재하지 않는 경우(null)에 이를 사용자에게 알려주는 함수는 hasNoData() 입니다. hasNoData() 함수의 파라미터는 “true” 혹은 “false” 입니다.

- true : 데이터가 없는 차트를 표시하고 데이터가 없다는 팝업 메시지를 보여줍니다.
- false : 화면에 차트와 메시지를 표시하지 않습니다.

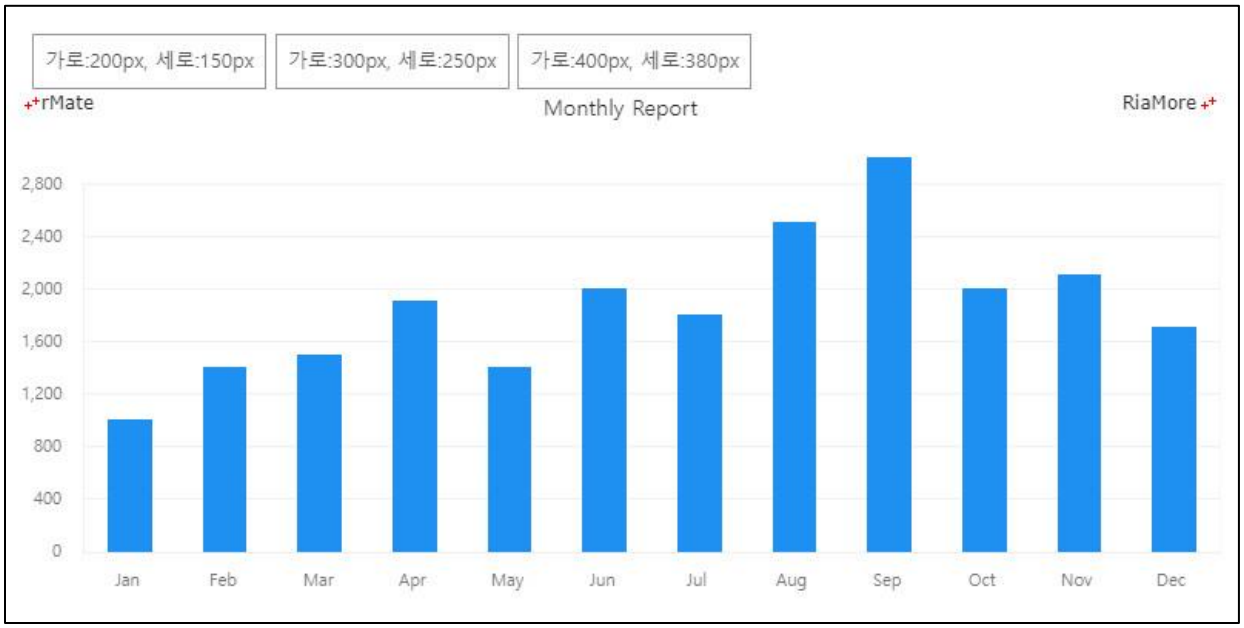
다음은 hasNoData() 함수를 이용하여 팝업 메시지를 보여주는 예제입니다.

```
function chartReadyHandler(id) {
  document.getElementById(id).setLayout(layoutStr);
  document.getElementById(id).setData(chartData);
  checkData(chartData);
}

var chartData = [];

function checkData(data){
  if(data.length <= 0)
  document.getElementById("chart1").hasNoData(true);
}
```

See the CodePen 알메이트 차트 - 차트에 표시할 데이터가 없을 경우

6.10 모바일 차트

알메이트 차트는 모바일 환경을 위한 반응형 웹 사이트에서도 효과적으로 활용될 수 있는 차팅 라이브러리입니다. 알메이트 차트를 이용해서 모바일 환경을 위한 차트를 작성할 때 모바일 환경을 위한 레이아웃을 별도로 작성하실 필요가 없습니다. 알메이트 차트에서는 차트를 위한 공간의 크기가 작아지면 작아진 공간에 맞추어 모든 차트의 구성 요소들의 크기가 자동으로 변경됩니다. 아래 예제 차트를 보면, 차트가 생성되는 의 크기는 3 가지로 다르지만 적용된 레이아웃은 모두 같은 레이아웃입니다.



See the CodePen 알메이트 차트 - 차트의 크기 재설정

모바일 차트 작성시 유의사항

모바일 환경을 위한 반응형 웹 사이트에 적용될 차트를 만들 때 주의할 사항은 의 크기를 픽셀 값으로 지정하지 않고 퍼센트 값으로 지정하는 것입니다. 퍼센트 값으로 크기를 지정하면 의 컨테이너(아래 예제에서는 <body>)에 설정된 전체 크기를 차트를 위한 공간으로 활용할 수 있기 때문에 모바일 환경과 같이 제한된 크기의 화면에서도 최대한 큰 공간을 사용할 수 있고, 다양한 크기의 화면 해상도를 효율적으로 지원할 수 있기 때문입니다.

```
<body>
  <div id="chartHolder" style="width:100%; height:100%;"></div>
</body>
```

그리고 또 유의할 점은 차트 내에 표시되는 타이틀, 축 레이블, 데이터 레이블 등과 같은 텍스트입니다. 위 예제 차트에서 보는 바와 같이 차트가 생성되는 공간의 크기가 변경되면 축 레이블의 개수는 자동으로 변경되지만 폰트 크기는 변경되지 않습니다. 따라서 반응형 웹 사이트에 적용될 차트를 생성할 때는 차트에 표시되는 텍스트의 수와 폰트 크기를 고려해서 디자인해야 합니다. 차트의 디자인은 차트 디자인과 스타일링을 참조하십시요.

touchMoveEnable 사용중지

모바일 장비의 크기가 작아서 차트가 화면의 거의 전체 크기를 차지하는 경우가 있습니다. 이 때는 사용자가 손가락으로 화면을 스크롤하려고 해도 차트에서 먼저 이벤트를 사용해버리기 때문에 스크롤을 하기 어렵게 됩니다. 이 경우에는 차트 노드(예, <Column2DChart>)의 touchMoveEnable 속성을 “false” 로 설정하여 알메이트 차트에서 터치 이벤트를 사용하지 않도록 설정할 수 있습니다.

```
<Column2DChart touchMoveEnable="false">
```