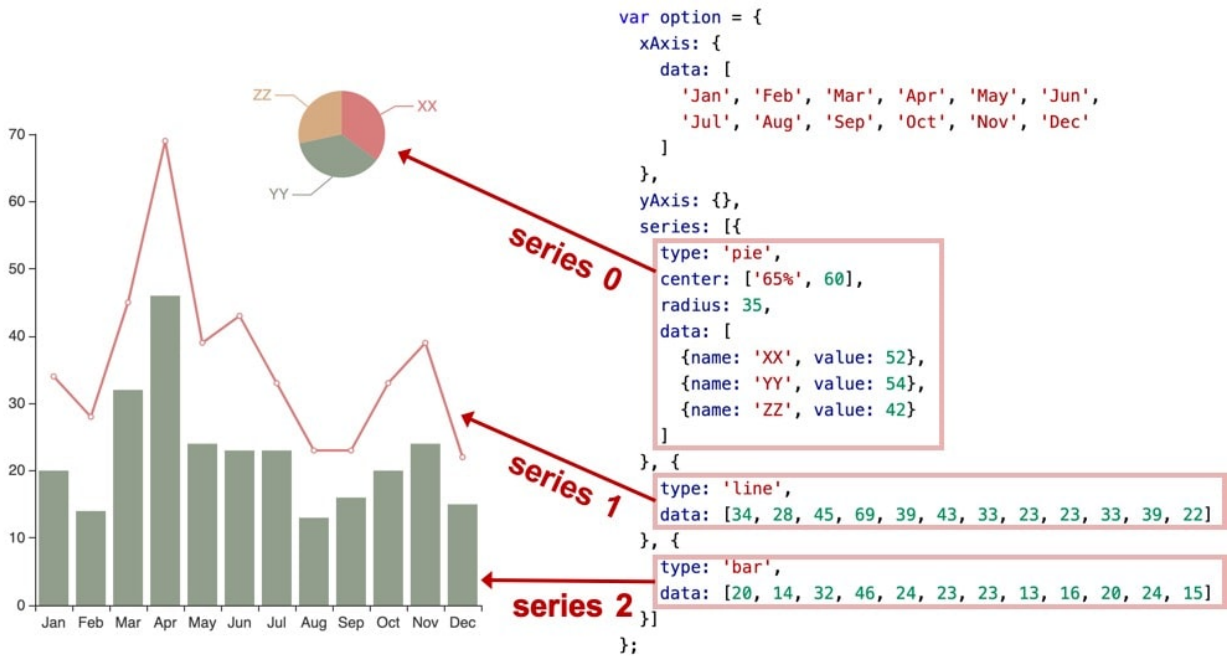
# 系列（series）

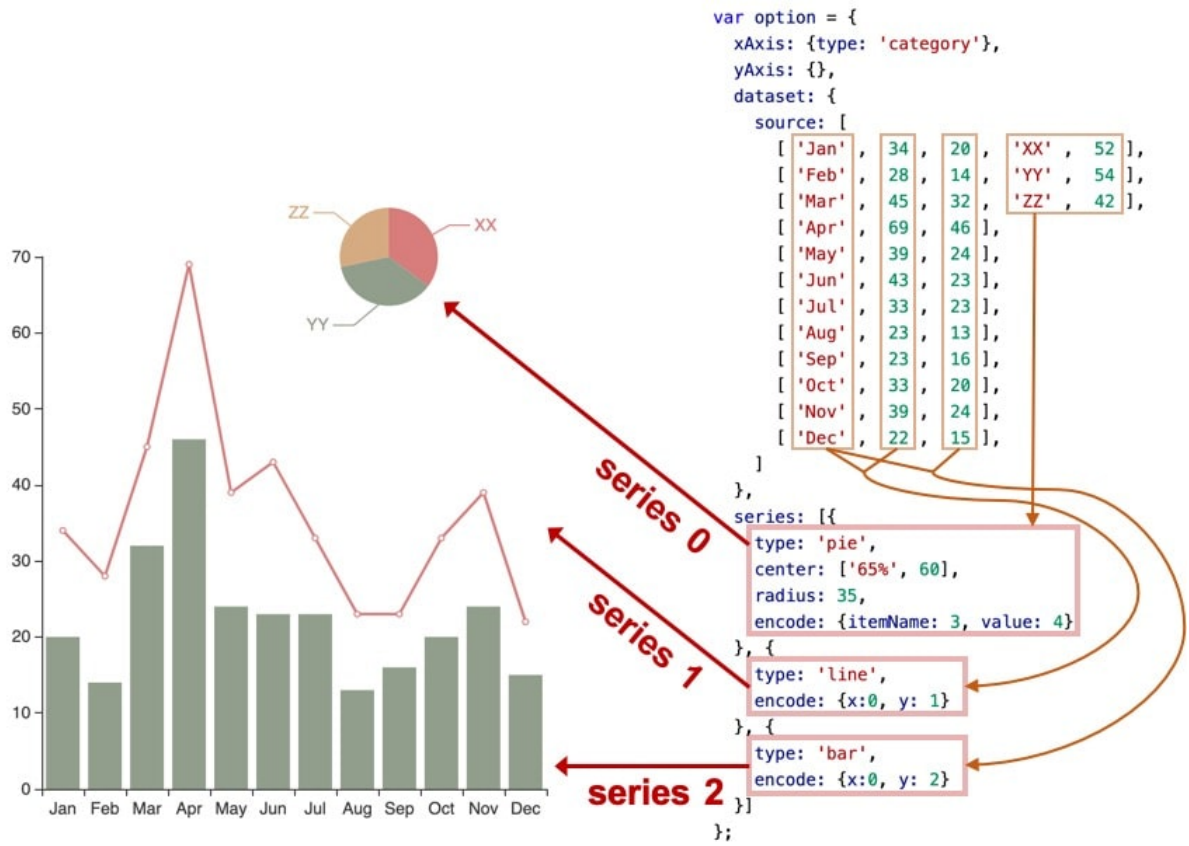
系列（series）是很常见的名词。在 echarts 里，系列（series）是指：一组数值以及他们映射成的图。“系列”这个词原本可能来源于“一系列的数据”，而在 echarts 中取其扩展的概念，不仅表示数据，也表示数据映射成为的图。所以，一个 系列 包含的要素至少有：一组数值、图表类型（series.type）、以及其他的关于这些数据如何映射成图的参数。

echarts 里系列类型（series.type）就是图表类型。系列类型（series.type）至少有：line（折线图）、bar（柱状图）、pie（饼图）、scatter（散点图）、graph（关系图）、tree（树图）、...

如下图，右侧的 option 中声明了三个 系列（series）：pie（饼图系列）、line（折线图系列）、bar（柱状图系列），每个系列中有他所需要的数据（series.data）。




类同地，下图中是另一种配置方式，系列的数据从 dataset 中取：

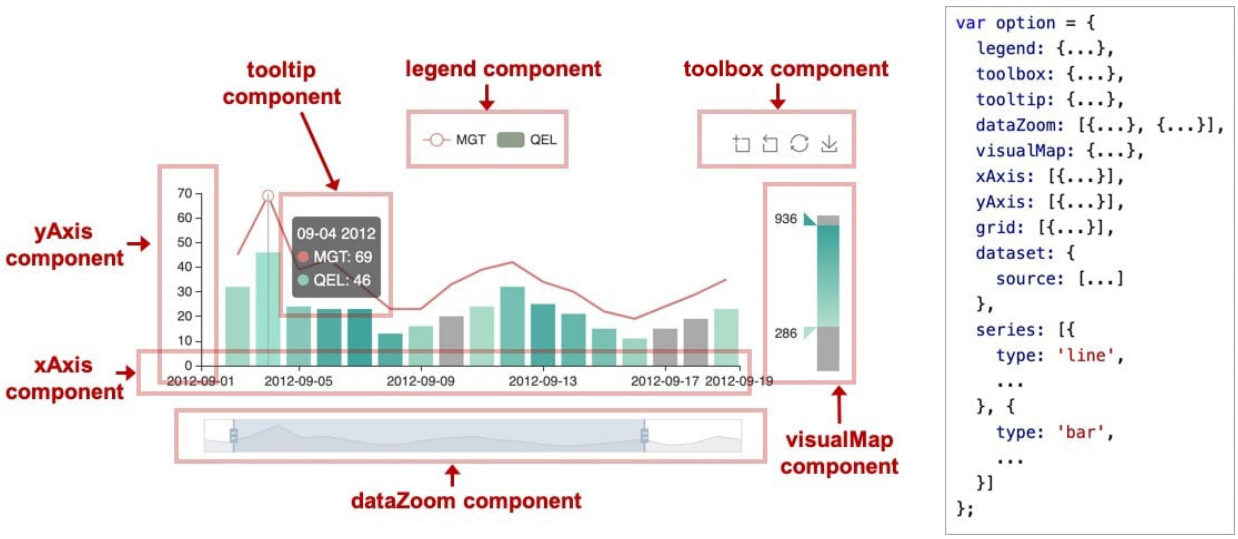



## 组件（component）

在系列之上，echarts 中各种内容，被抽象为“组件”。例如，echarts 中至少有这些组件：`xAxis`（直角坐标系 X 轴）、`yAxis`（直角坐标系 Y 轴）、`grid`（直角坐标系底板）、`angleAxis`（极坐标系角度轴）、`radiusAxis`（极坐标系半径轴）、`polar`（极坐标系底板）、`geo`（地理坐标系）、`dataZoom`（数据区缩放组件）、`visualMap`（视觉映射组件）、`tooltip`（提示框组件）、`toolbox`（工具栏组件）、`series`（系列）、...

我们注意到，其实系列（`series`）也是一种组件，可以理解为：系列是专门绘制“图”的组件。

如下图，右侧的 `option` 中声明了各个组件（包括系列），各个组件就出现在图中。



注：因为系列是一种特殊的组件，所以有时候也会出现 “组件和系列” 这样的描述，这种语境下的 “组件” 是指：除了 “系列” 以外的其他组件。

## 用 option 描述图表

上面已经出现了 `option` 这个概念。echarts 的使用者，使用 `option` 来描述其对图表的各种需求，包括：有什么数据、要画什么图表、图表长什么样子、含有什么组件、组件能操作什么事情等等。简而言之，`option` 表述了：数据、数据如何映射成图形、交互行为。

```
// 创建 echarts 实例。 var dom = document.getElementById('dom-id'); var chart = echarts.init(dom); // 用 option 描述 `数据`、`数据如何映射成图形`、`交互行为` 等。 // option 是个大的 JavaScript 对象。 var option = { // option 每个属性是一类组件。 legend: {...}, grid: {...}, tooltip: {...}, toolbox: {...}, dataZoom: {...}, visualMap: {...}, // 如果有多个同类组件，那么就是个数组。例如这里有三个 X 轴。 xAxis: [ // 数组每项表示一个组件实例，用 type 描述“子类型”。 {type: 'category', ...}, {type: 'category', ...}, {type: 'value', ...} ], yAxis: [{...}, {...}], // 这里有多个系列，也是构成一个数组。 series: [ // 每个系列，也有 type 描述“子类型”，即“图表类型”。 {type: 'line', data: [['AA', 332], ['CC', 124], ['FF', 412], ... ]}, {type: 'line', data: [2231, 1234, 552, ... ]}, {type: 'line', data: [[4, 51], [8, 12], ... ]} ] }; // 调用 setOption 将 option 输入 echarts，然后 echarts 渲染图表。 chart.setOption(option);
```

系列里的 `series.data` 是本系列的数据。而另一种描述方式，系列数据从 `dataset` 中取：

```
var option = { dataset: { source: [ [121, 'XX', 442, 43.11], [663, 'ZZ', 311, 91.14], [913, 'ZZ', 312, 92.12], ... ] }, xAxis: {}, yAxis: {}, series: [ // 数据从 dataset 中取，encode 中的数值是 dataset.source 的维度 index （即第几列） {type: 'bar', encode: {x: 1, y: 0}}, {type: 'bar', encode: {x: 1, y: 2}}, {type: 'scatter', encode: {x: 1, y: 3}}, ... ] };
```

## 组件的定位

不同的组件、系列，常有不同的定位方式。

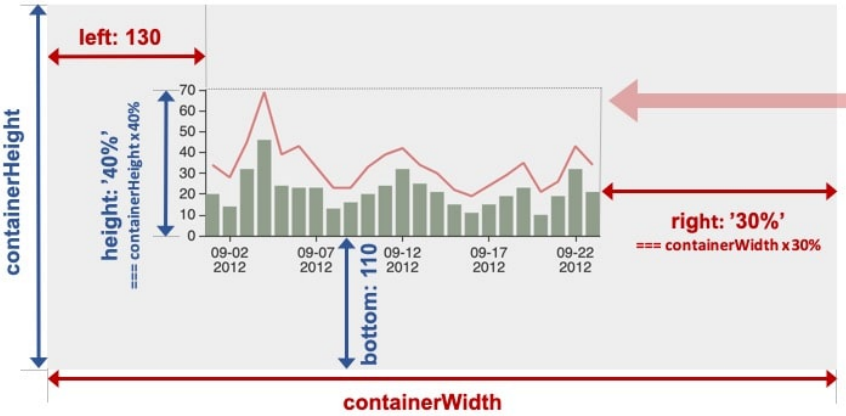
**[类 CSS 的绝对定位]**

多数组件和系列，都能够基于 `top` / `right` / `down` / `left` / `width` / `height` 绝对定位。 这种绝对定位的方式，类似于 `CSS` 的绝对定位（`position: absolute`）。绝对定位基于的是 echarts 容器 DOM 节点。

其中，他们每个值都可以是：

- 绝对数值（例如 `bottom: 54` 表示：距离 echarts 容器底边界 `54` 像素）。
- 或者基于 echarts 容器高宽的百分比（例如 `right: '20%'` 表示：距离 echarts 容器右边界的距离是 echarts 容器宽度的 `20%`）。

如下图的例子，对 `grid` 组件（也就是直角坐标系的底板）设置 `left`、`right`、`height`、`bottom` 达到的效果。




```
var option = {
  grid: {
    left: 130,
    right: '30%',
    height: '40%',
    bottom: 110
  },
  xAxis: {...},
  yAxis: {...},
  series: [{...}, {...}]
};
```

我们可以注意到，`left` `right` `width` 是一组（横向）、`top` `bottom` `height` 是另一组（纵向）。这两组没有什么关联。每组中，至多设置两项就可以了，第三项会被自动算出。例如，设置了 `left` 和 `right` 就可以了，`width` 会被自动算出。

**[中心半径定位]**

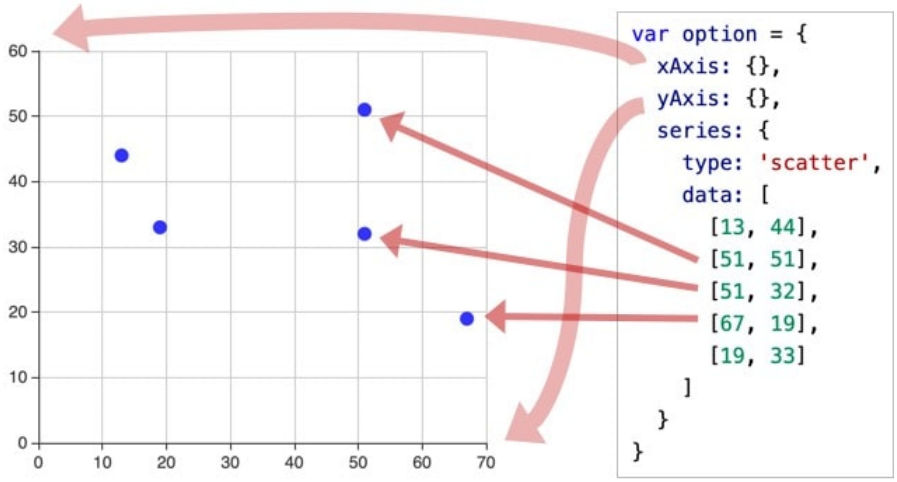
少数圆形的组件或系列，可以使用“中心半径定位”，例如，`pie`（饼图）、`sunburst`（旭日图）、`polar`（极坐标系）。
中心半径定位，往往依据 `center`（中心）、`radius`（半径）来决定位置。

**[其他定位]**

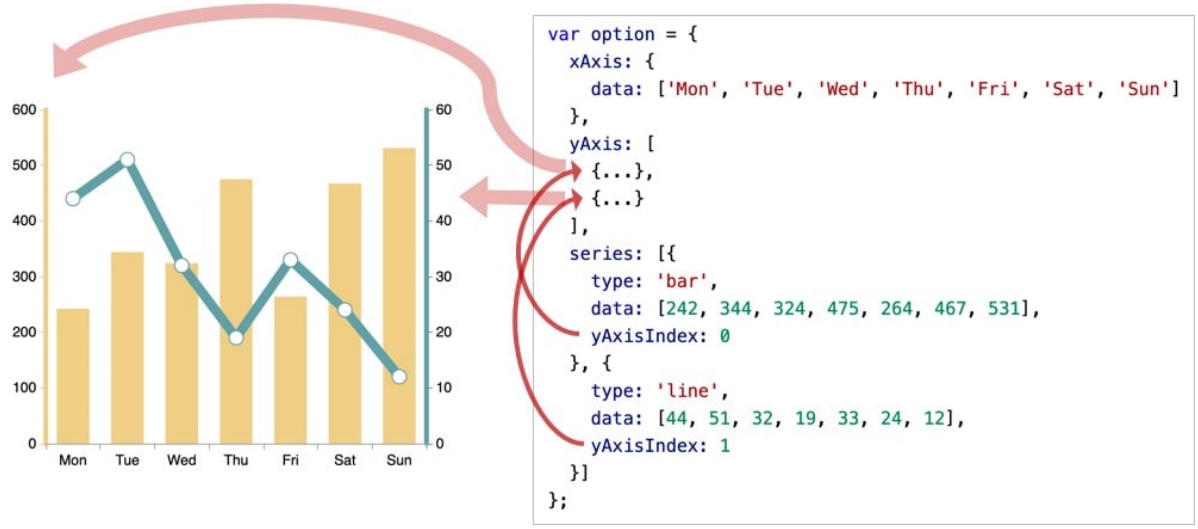
少数组件和系列可能有自己的特殊的定位方式。在他们的文档中会有说明。

## 坐标系

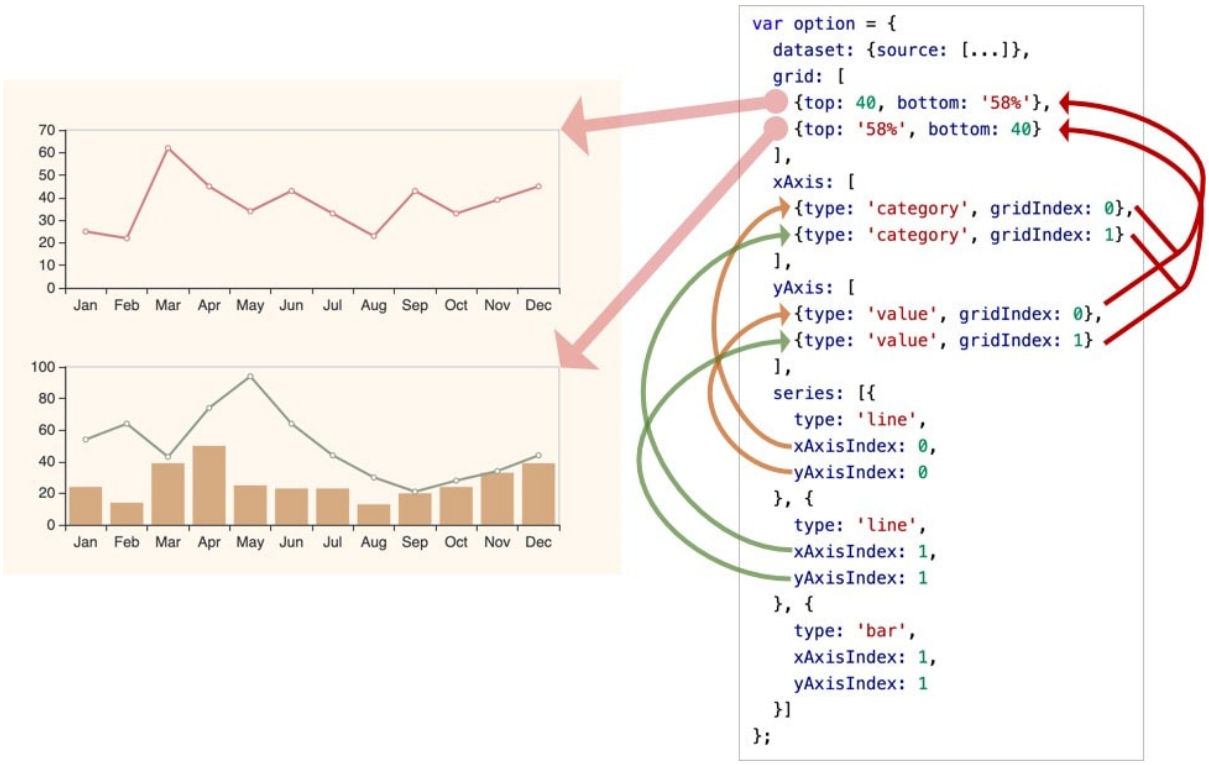
很多系列，例如 `line`（折线图）、`bar`（柱状图）、`scatter`（散点图）、`heatmap`（热力图）等等，需要运行在 “坐标系” 上。坐标系用于布局这些图，以及显示数据的刻度等等。例如 echarts 中至少支持这些坐标系：直角坐标系、极坐标系、地理坐标系（GEO）、单轴坐标系、日历坐标系 等。其他一些系列，例如 `pie`（饼图）、`tree`（树图）等等，并不依赖坐标系，能独立存在。还有一些图，例如 `graph`（关系图）等，既能独立存在，也能布局在坐标系中，依据用户的设定而来。
一个坐标系，可能由多个组件协作而成。我们以最常见的直角坐标系来举例。直角坐标系中，包括有 `xAxis`（直角坐标系 X 轴）、`yAxis`（直角坐标系 Y 轴）、`grid`（直角坐标系底板）三种组件。`xAxis`、`yAxis` 被 `grid` 自动引用并组织起来，共同工作。
我们来看下图，这是最简单的使用直角坐标系的方式：只声明了 `xAxis`、`yAxis` 和一个 `scatter`（散点图系列），echarts 暗自为他们创建了 `grid` 并关联起他们：




再来看下图，两个 `yAxis`，共享了一个 `xAxis`。两个 `series`，也共享了这个 `xAxis`，但是分别使用不同的 `yAxis`，使用 `yAxisIndex` 来指定它自己使用的是哪个 `yAxis`：




再来看下图，一个 echarts 实例中，有多个 `grid`，每个 `grid` 分别有 `xAxis`、`yAxis`，他们使用 `xAxisIndex`、`yAxisIndex`、`gridIndex` 来指定引用关系：




另外，一个系列，往往能运行在不同的坐标系中。例如，一个 scatter（散点图）能运行在 直角坐标系、极坐标系 、地理坐标系（GEO） 等各种坐标系中。同样，一个坐标系，也能承载不同的系列，如上面出现的各种例子，直角坐标系 里承载了 line（折线图）、bar（柱状图）等等。