[EChart教程 2](#_Toc17818)

[5分钟上手ECharts 2](#_Toc10774)

[获取ECharts 2](#_Toc9639)

[引入ECharts 2](#_Toc3030)

[绘制表格 2](#_Toc4815)

[自定义构建ECharts 2](#_Toc17602)

[ECharts样式简介 2](#_Toc16384)

[颜色主题(Theme) 2](#_Toc10737)

[调色盘 3](#_Toc27725)

[直接样式设置 3](#_Toc11038)

[高亮样式：emphasis 3](#_Toc2480)

[异步数据加载和更新 3](#_Toc22232)

[使用dataset管理数据 3](#_Toc18163)

[入门例子 3](#_Toc31064)

[数据到图形映射 4](#_Toc32187)

[其他 5](#_Toc29391)

[在图表中加入交互组件 5](#_Toc19976)

[移动端自适应 5](#_Toc2417)

[left/right/top/bottom/width/height定位 5](#_Toc15020)

[center/radius定位 5](#_Toc23687)

[组件横向/纵向布局 5](#_Toc30343)

[Media Query 5](#_Toc2372)

[数据视觉映射 6](#_Toc20614)

[数据和维度 6](#_Toc22337)

[visualMap组件 7](#_Toc23737)

[ECharts中的事件和行为 7](#_Toc12740)

[ECharts API 7](#_Toc11005)

[类方法 7](#_Toc7222)

[实例方法 7](#_Toc8201)

[action 8](#_Toc901)

[events 8](#_Toc27622)

# EChart教程

## 5分钟上手ECharts

### 获取ECharts

npm安装，npm install echarts --save

cdn引入；查找网址<https://cdnjs.com/libraries/echarts>

官网、GitHub下载后引入。

### 引入ECharts

<script>引入；

import \* as echarts from ‘echart’;

var echarts = require(‘echarts’);

### 绘制表格

var myChart = echarts.init(domNode);

myChart.setOption(option);

## 自定义构建ECharts

构建ECharts，去除不需要的功能，压缩代码文件大小。

定制下载：<http://echarts.baidu.com/download.html>

在线构建：<http://echarts.baidu.com/builder.html>

还可通过node、webpack、rollup等插件构建。

按需引入ECharts模块列表，分chart、component两块：<https://github.com/apache/incubator-echarts/blob/master/index.js>

## ECharts样式简介

### 颜色主题(Theme)

echarts.init(dom,themeName);

ECharts4内置默认主题、light、dark主题。

主题在线编辑：<http://echarts.baidu.com/theme-builder/>

### 调色盘

设置图表颜色集，有全局调色盘、系列调色盘。

option = {

color: [‘#c23531’...], //全局调色盘

series: [ {

type: ‘pie’,

color: ['#37A2DA'...] //系列调色盘

} ]

}

### 直接样式设置

设置itemStyle、lineStyle、areaStyle、label

### 高亮样式：emphasis

配置鼠标悬浮时的样式，一般使用默认值。

## 异步数据加载和更新

获取数据前，使用默认值显示(‘’,[]等)；获取数据后，调用setOption刷新显示。

showLoading()/hideLoading()；显示/隐藏加载动画，在setOption前调用。

### 使用dataset管理数据

ECharts4支持使用option.dataset用于单独的数据集声明；从而数据可以单独管理，被多个组件复用。

### 入门例子

dataset支持二维数组、对象数组等格式。

dataset: { //二维数组格式

// 提供一份数据。

source: [

['product', '2015', '2016', '2017'],

['Matcha Latte', 43.3, 85.8, 93.7],

['Milk Tea', 83.1, 73.4, 55.1],

['Cheese Cocoa', 86.4, 65.2, 82.5],

['Walnut Brownie', 72.4, 53.9, 39.1]

]

}

dataset: { //对象数组格式

// 这里指定了维度名的顺序，从而可以利用默认的维度到坐标轴的映射。

// 如果不指定 dimensions，也可以通过指定 series.encode 完成映射

dimensions: ['product', '2015', '2016', '2017'],

source: [

{product: 'Matcha Latte', '2015': 43.3, '2016': 85.8, '2017': 93.7},

{product: 'Milk Tea', '2015': 83.1, '2016': 73.4, '2017': 55.1},

{product: 'Cheese Cocoa', '2015': 86.4, '2016': 65.2, '2017': 82.5},

{product: 'Walnut Brownie', '2015': 72.4, '2016': 53.9, '2017': 39.1}

]

}

使用dataset，series需要变动。

series: [

{type: 'bar'},

{type: 'bar'},

{type: 'bar'}

]

### 数据到图形映射

seriesLayoutBy，指定dataset按column、row映射，默认column。对二维数组格式有效。

series: [ {type: ‘bar’,seriesLayoutBy: ‘row’}...]

encode，指定series.dimensions映射到坐标轴、提示框等的规则

**维度(dimensions)**

维度定义：包含在dataset.source中、dataset.dimensions、series.dimensions。

dataset.sourceHeader，true-数组第一行/列是维度，false-数组全为数值数据。

dataset: {

dimensions: [

{name: 'score'},// 可以简写为 string，表示维度名。

'amount', // 可以在 type 中指定维度类型。

{name: 'product', type: 'ordinal'},

null //不设置维度

],

source: [...]

}

维度类型：number--数值(默认)，time--时间，ordinal--文本，float--浮点(有益性能)，int--整数(有益性能)。

**encode映射**

encode对象结构，key为坐标系、标签等名称，value为维度名、维度序号(number格式，从0开始)；可通过数组指定多个维度。

series: [

{ type: ‘bar’,

encode: { x: [‘2015’], y: ‘product’, tooltip: [‘product’,’2015’]}

}... ]

**视觉通道(颜色、尺寸等)的映射**

使用visualMa票进行映射。

### 其他

可声明多个dataset，通过datasetIndex区分。

dataset = [ {...},{...}];

series = [ {datasetIndex: 0,...}, {datasetIndex:1,...} ]

声明series.data后，会使用series.data而非dataset的数据。

## 在图表中加入交互组件

ECharts提供多种组件：图例组件legend、标题组件title、视觉映射组件visualMap、缩放组件dataZoom、时间组件timeline、提示组件tooltip、工具栏toolbox。

## 移动端自适应

ECharts内部图表、组件定位方式类似绝对定位。

### left/right/top/bottom/width/height定位

绝对值，纯数字，不跟px；{width: 300}

百分比，字符串；{height: ‘70%’}

位置描述，字符串，左右-center、上下-middle；{top: ‘middle’}

### center/radius定位

center，格式：[x,y]，x、y为绝对值或百分比，相对左上角；{center: [30,30]}

radius，格式：[x,y]，x、y为绝对值或百分比，为内外半径；{radius: [‘70%’,’80%’]}

### 组件横向/纵向布局

组件设置横向、纵向，在组件配置的orient、layout属性上，值：vertical、horizontal。

### Media Query

通过query匹配条件，合并不同的option项，实现自适应。多个query同时满足，后者优先。

query支持width、height、aspectRatio属性，可加前缀min、max。

option = {

baseOption: { // 这里是基本的『原子option』。

title: {...},

legend: {...},

series: [{...}, {...}, ...],

...

},

media: [ // 这里定义了 media query 的逐条规则。

{

query: {

minWidth: 200,//最小宽度

maxHeight: 300,//最大高度

minAspectRatio: 1.3 //长宽比小于1.3

}, // 这里写规则。

option: { // 这里写此规则满足下的option。

legend: {...},

...

}

},

{ // 这条里没有写规则，表示『默认』，

option: { // 即所有规则都不满足时，采纳这个option。

legend: {...},

...

}

}

]

};

## 数据视觉映射

visualMap组件元素：图形类别-symbol、图像大小-symbolSize、颜色-color、透明度-opacity、颜色透明度-colorAlpha、颜色明暗度-colorLightness、颜色饱和度-colorSaturation、色调-colorHue。

### 数据和维度

ECharts的数据存放于series.data中，为数据项的集合；每个数据项分为数据值、其他信息；数据值为数值(一维)、数组(多维)。

### visualMap组件

定义把哪个数据维度映射到什么视觉元素。

## ECharts中的事件和行为

ECharts事件分两类：鼠标事件、组件交互事件。

通过on(eventName,callback)绑定事件回调函数，eventName为dom事件名称，全小写。

echart.on(‘click’,params=>{})；每个事件都包含params。

echart.dispatchAction({...})；主动触发事件。

# ECharts API

[http://echarts.baidu.com/api.html#echarts](http://echarts.baidu.com/api.html" \l "echarts)

## 类方法

init(dom [,theme,opt])；创建echart实例。

connect(group:string|array)；多个图表实例联动。group为图表实例的group属性。

disconnect(group:string)；解除联动

dispose(target:EChart|dom)；销毁图表

getInstanceByDom(target:dom)；获取dom上的图表实例。

registerMap(name:string,geoJson:object [,specialArea:objec])；注册地图

getMap(name:string)；获取地图

registerTheme(name:string,theme:object)；注册主题

## 实例方法

group；图表的分组，用于联动。

setOption(option:object [opt:object])；设置图表配置。opt属性：notMerge、lazyUpdate、silent。

getWidth()/getHeight()/getDom()/getOption()；获取宽、高、dom、option。

resize([opt:object])；改变图表尺寸，默认为容器宽高。opt属性：width、height、silent。

dispatchAction(payload:object)；触发图表行为。

on(eventName:string,handler:Function [,context:onject])；绑定事件处理函数。

off(eventName:string [,handler:Function])；解绑事件，不传handler则解绑事件所有处理函数。

showLoading()/hideLoading()；展示/隐藏加载效果。

getDataURL(opt:object)；导出图表图片，返回一个base64的URL，可设置为图片的src

var img = new Image();

img.src = myChart.getDataURL({

pixelRatio: 2,

backgroundColor: '#fff',

excludeComponents: [‘toolbox’]

});

getConnectedDataURL(opt:object)；导出联动图表图片，返回URL。

clear()；清空图表，会清空组件和图表

dispose()；销毁实例

isDisposed()；实例是否销毁

## action

## events