

**BIG DATA**

## **10.大数据可视化**

# 什么是数据可视化

---

- 数据可视化是指将大型数据集中的数据以图形图像形式表示，并利用数据分析和开发工具发现其中未知信息的处理过程
- 数据可视化技术的基本思想是将数据库中每一个数据项作为单个图元素表示，大量的数据集构成数据图像，同时将数据的各个属性值以多维数据的形式表示，可以从不同的维度观察数据，从而对数据进行更深入的观察和分析

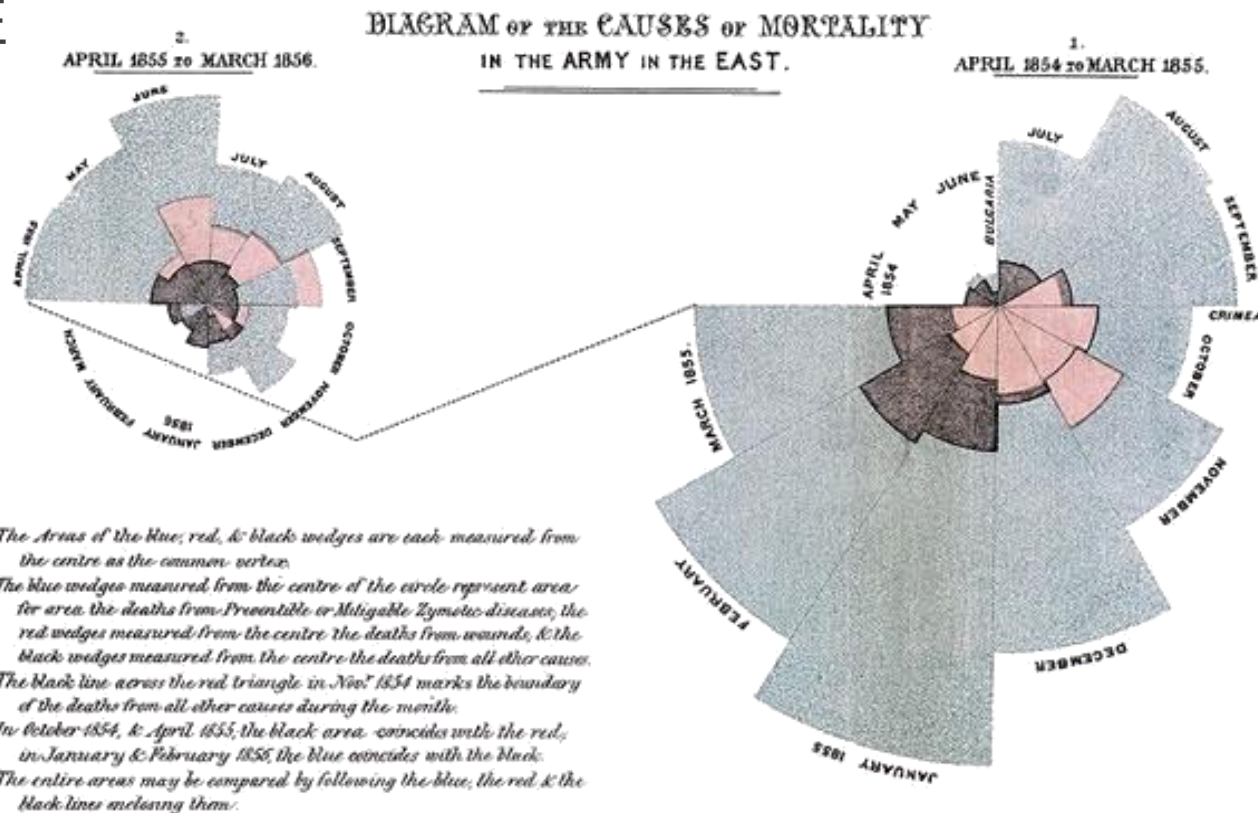
# 霍乱地图

- 霍乱地图分析了霍乱患者分布与水井分布之间的关系，发现在有一口井的供水范围内患者明显偏多，据此找到了霍乱爆发的根源是一个被污染的水泵



# 玫瑰图

- 数据可视化历史上的另一个经典之作是1857年“提灯女神”南丁格尔设计的“鸡冠花图”（又称玫瑰图），它以图形的方式直观地呈现了英国在克里米亚战争中牺牲的战士数量和死亡原因，有力地说明了改善军队医院的医疗条件对于减少战争伤亡的重要性



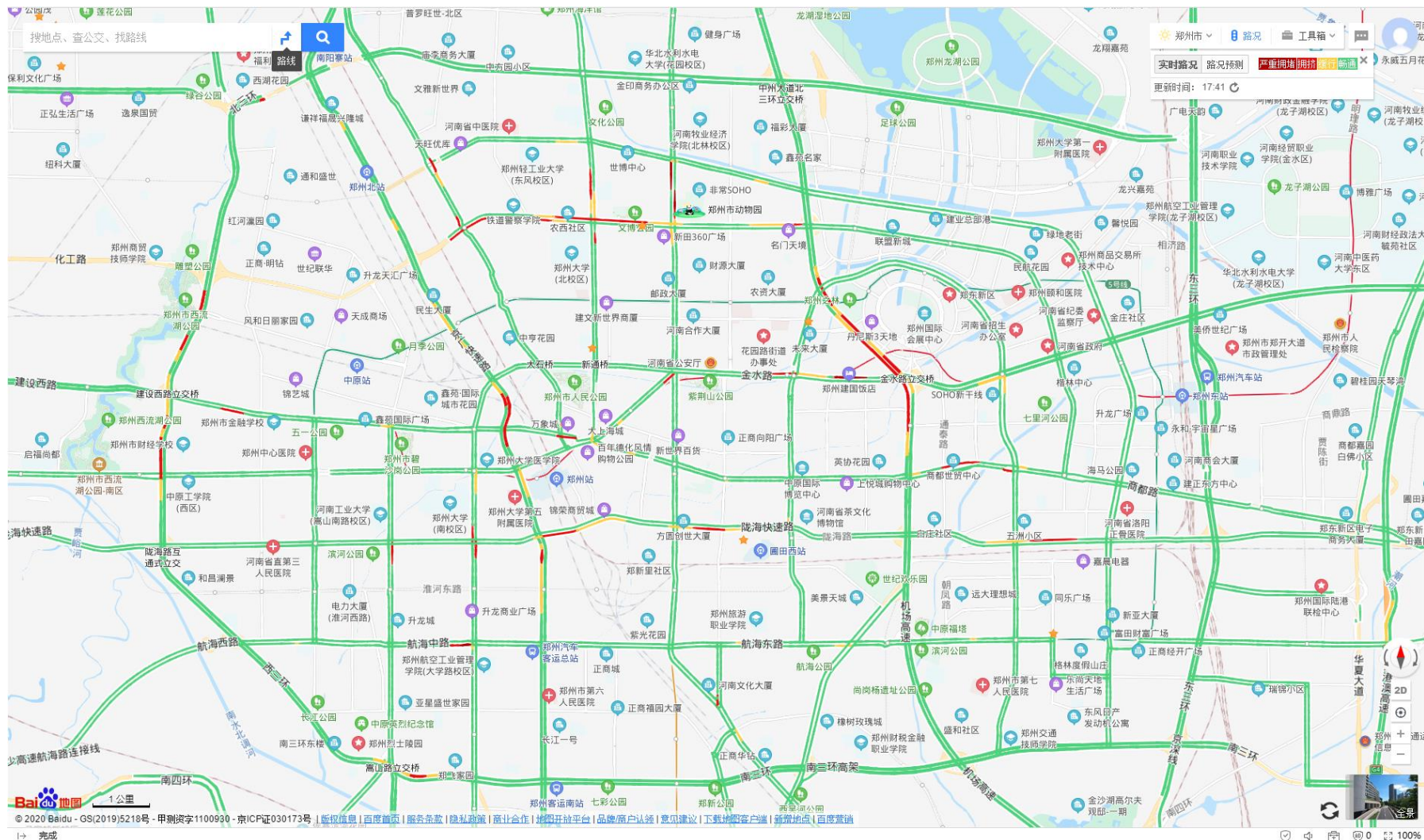
# 数据可视化的发展

- 20世纪50年代，随着计算机的出现和计算机图形学的发展，人们可以利用计算机技术在电脑屏幕上绘制出各种图形图表，可视化技术开启了全新的发展阶段。
- 最初，可视化技术被大量应用于**统计学**领域，用来绘制统计图表，比如圆环图、柱状图和饼图、直方图、时间序列图、等高线图、散点图等，后来，又逐步应用于**地理信息系统、数据挖掘分析、商务智能工具**等，有效促进了人类对不同类型数据的分析与理解。
- 随着大数据时代的到来，每时每刻都有海量数据在不断生成，需要我们对数据进行及时、全面、快速、准确的分析，呈现数据背后的价值，这就更需要可视化技术协助我们更好地理解和分析数据，可视化成为大数据分析最后的一环和对用户而言最重要的一环。



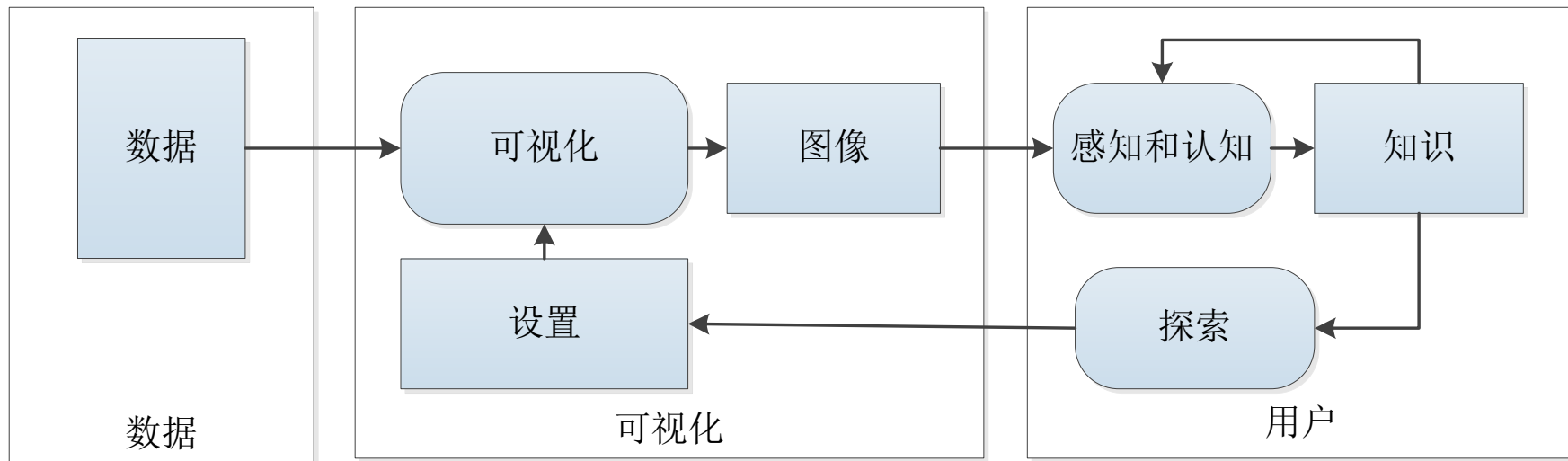
# 数据可视化的应用：观测、跟踪数据

## • 郑州市实时路况



# 数据可视化的应用：分析数据

- 用户参与的可视化分析过程



# 数据可视化的应用：辅助理解数据

- 爱奇艺人物关系图谱

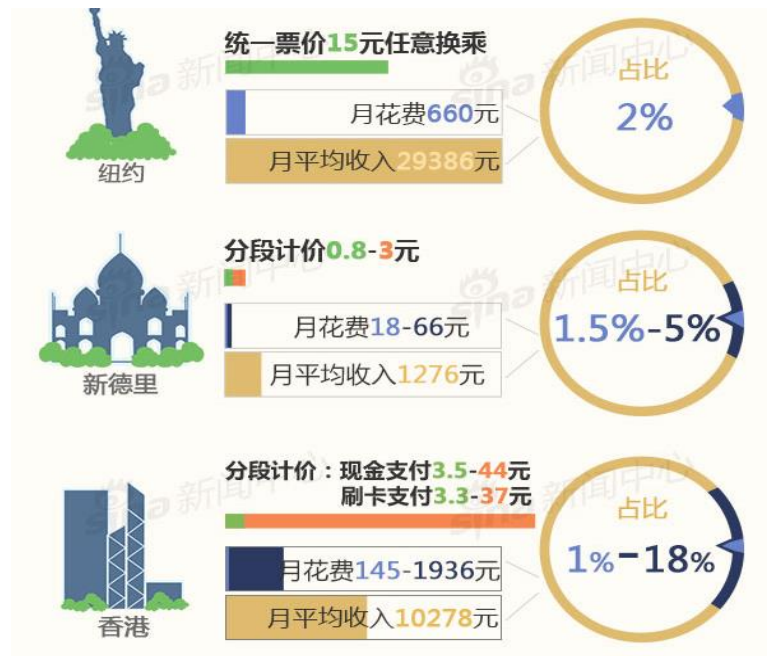
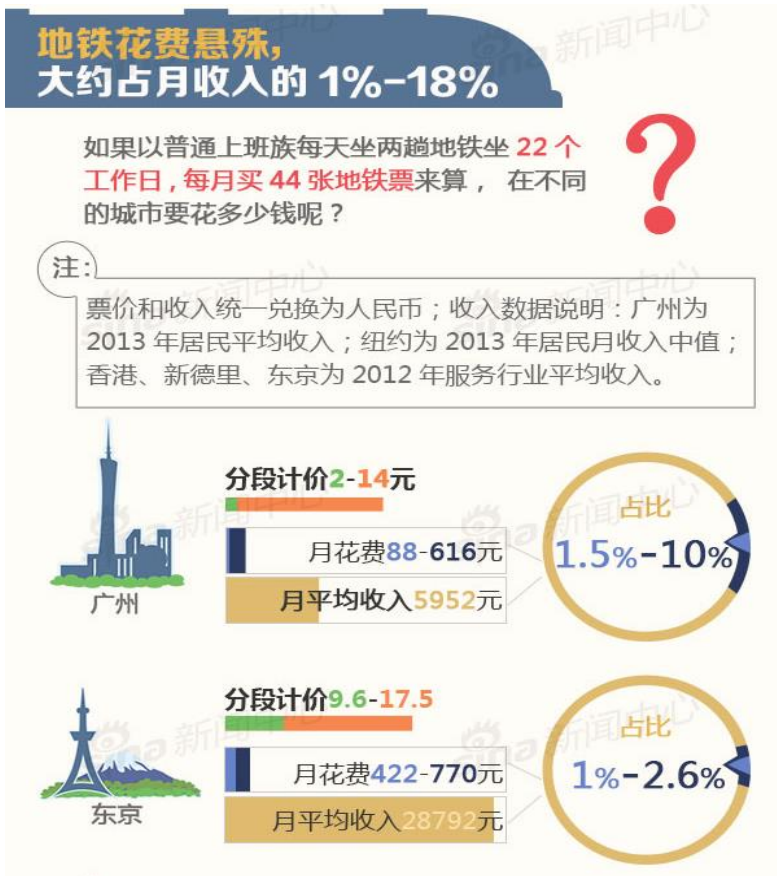
李荣浩的关系图谱





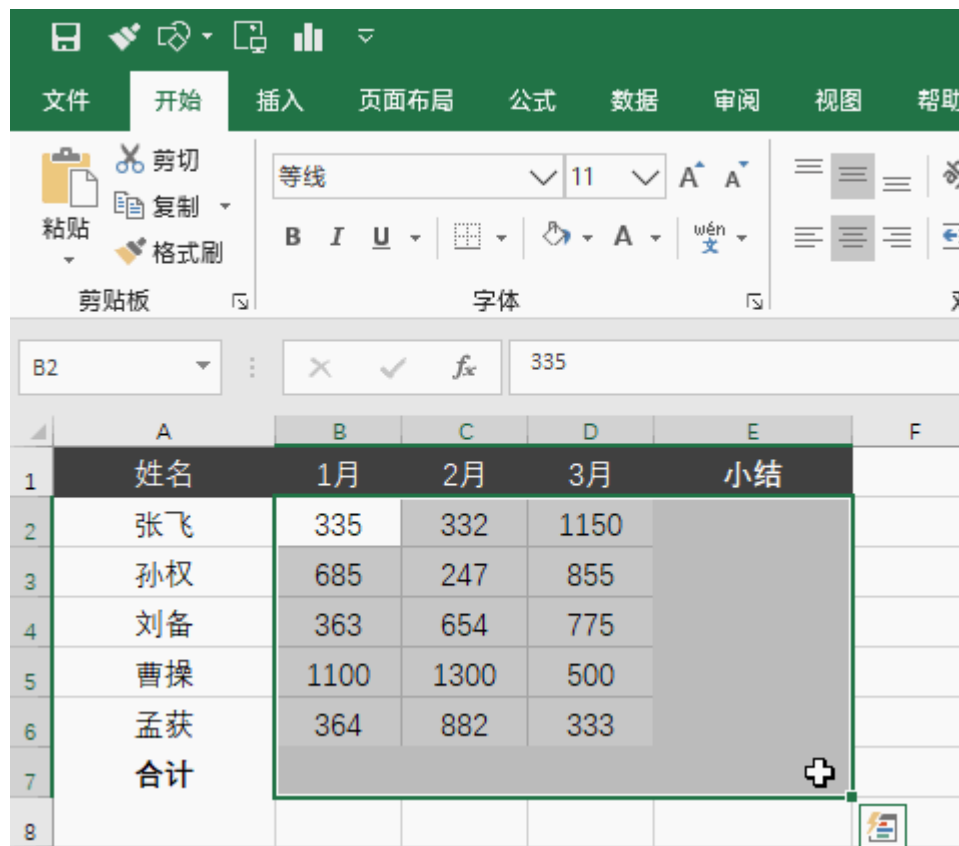
# 数据可视化的应用：增强数据吸引力

## • 一个可视化的图表新闻实例



# 可视化工具：入门级工具

- Excel是微软公司的办公软件Office家族的系列软件之一，可以进行各种数据的处理、统计分析和辅助决策操作，已经广泛地应用于管理、统计、金融等领域

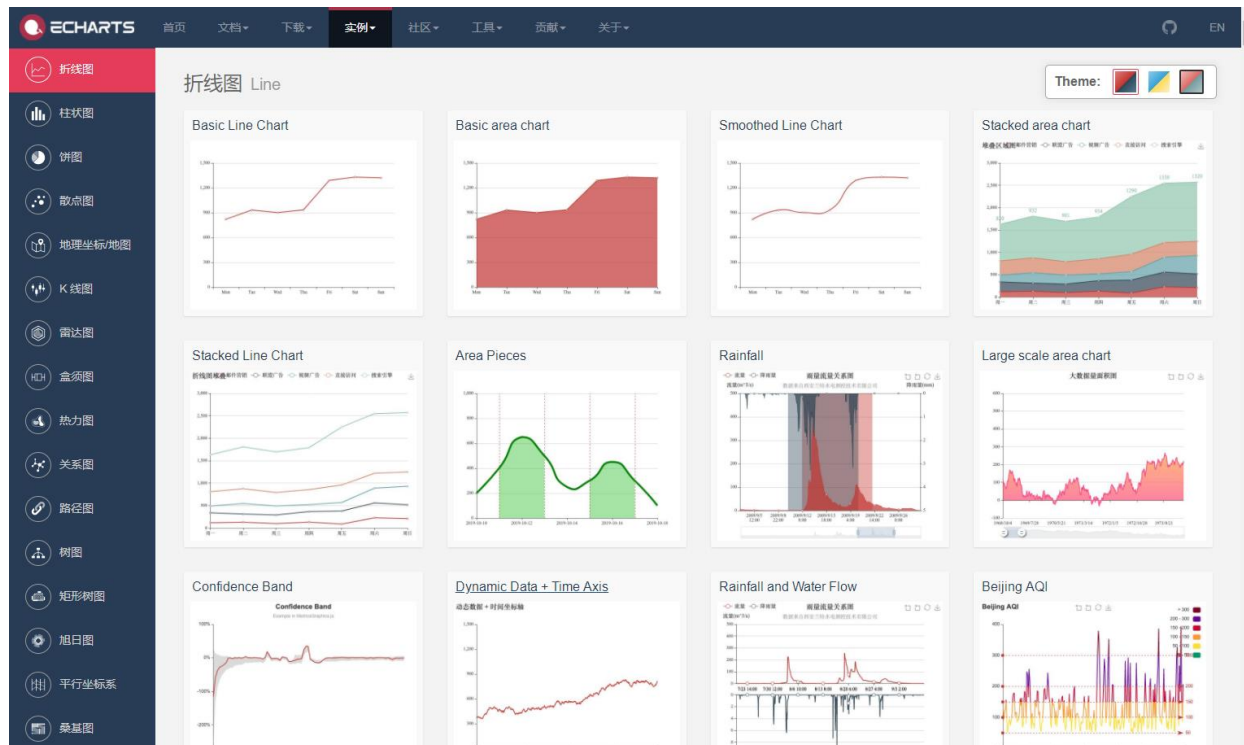


The screenshot displays the Microsoft Excel interface. The ribbon at the top includes '文件' (File), '开始' (Home), '插入' (Insert), '页面布局' (Layout), '公式' (Formulas), '数据' (Data), '审阅' (Review), '视图' (View), and '帮助' (Help). The '开始' ribbon is active, showing options for '剪贴板' (Clipboard) and '字体' (Font). The font settings are set to '等线' (Equity), size 11, bold, italic, and underlined. The formula bar shows 'B2' and the value '335'. The data table is as follows:

	A	B	C	D	E	F
1	姓名	1月	2月	3月	小结	
2	张飞	335	332	1150		
3	孙权	685	247	855		
4	刘备	363	654	775		
5	曹操	1100	1300	500		
6	孟获	364	882	333		
7	合计					
8						

# 可视化工具：Echarts

- 百度开源的图形库
- 支持折线图、柱状图、散点图、饼图、K线图，用于统计的盒形图，用于地理数据可视化的地图、热力图、线图，用于关系数据可视化的关系图、treemap、旭日图，多维数据可视化的平行坐标，还有用于 BI 的漏斗图，仪表盘，并且支持图与图之间的混搭。



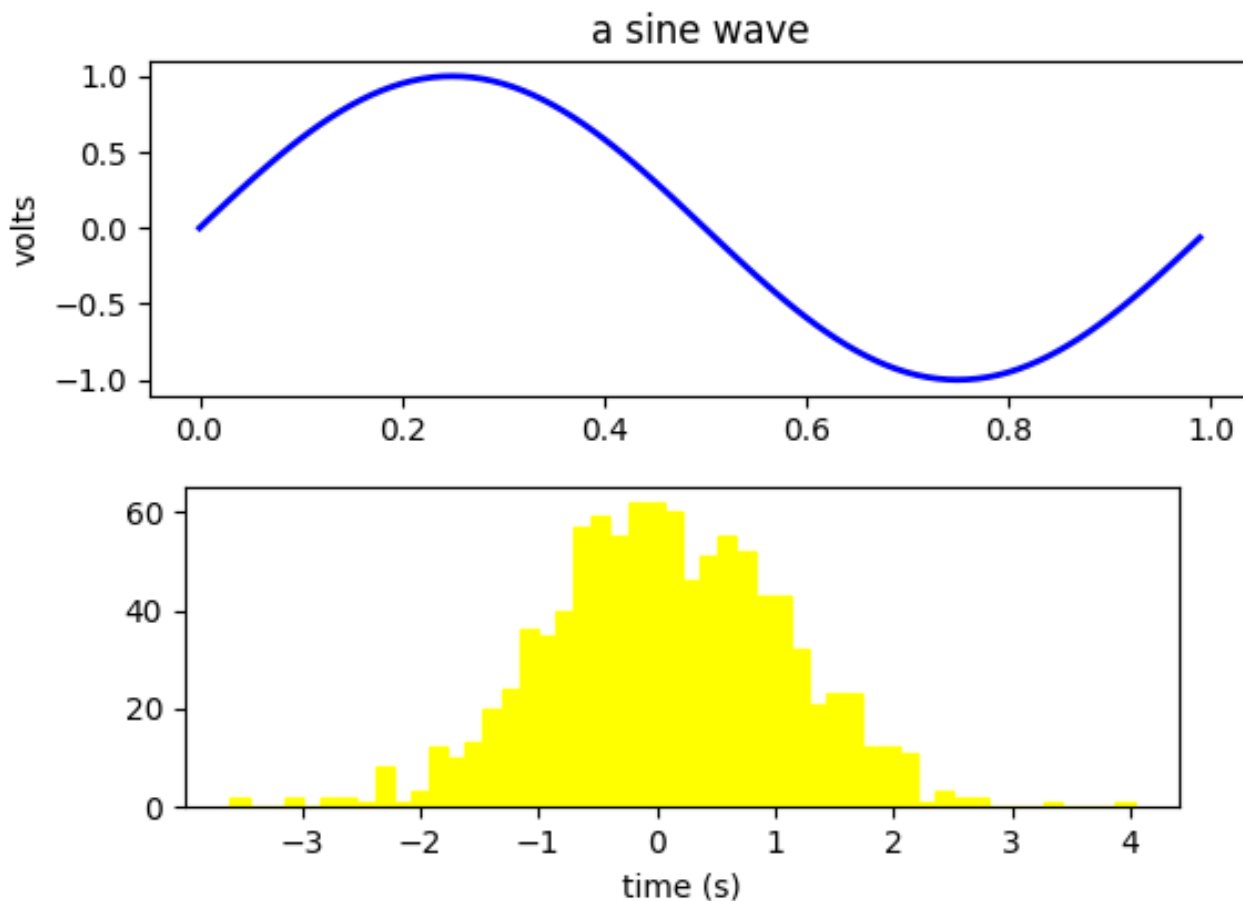
## 可视化工具：D3

- D3是最流行的可视化库之一，是一个用于网页作图、生成互动图形的JavaScript函数库，提供了一个D3对象，所有方法都通过这个对象调用。D3能够提供大量线性图和条形图之外的复杂图表样式，例如Voronoi图、树形图、圆形集群和单词云等。



# 可视化工具：matplotlib

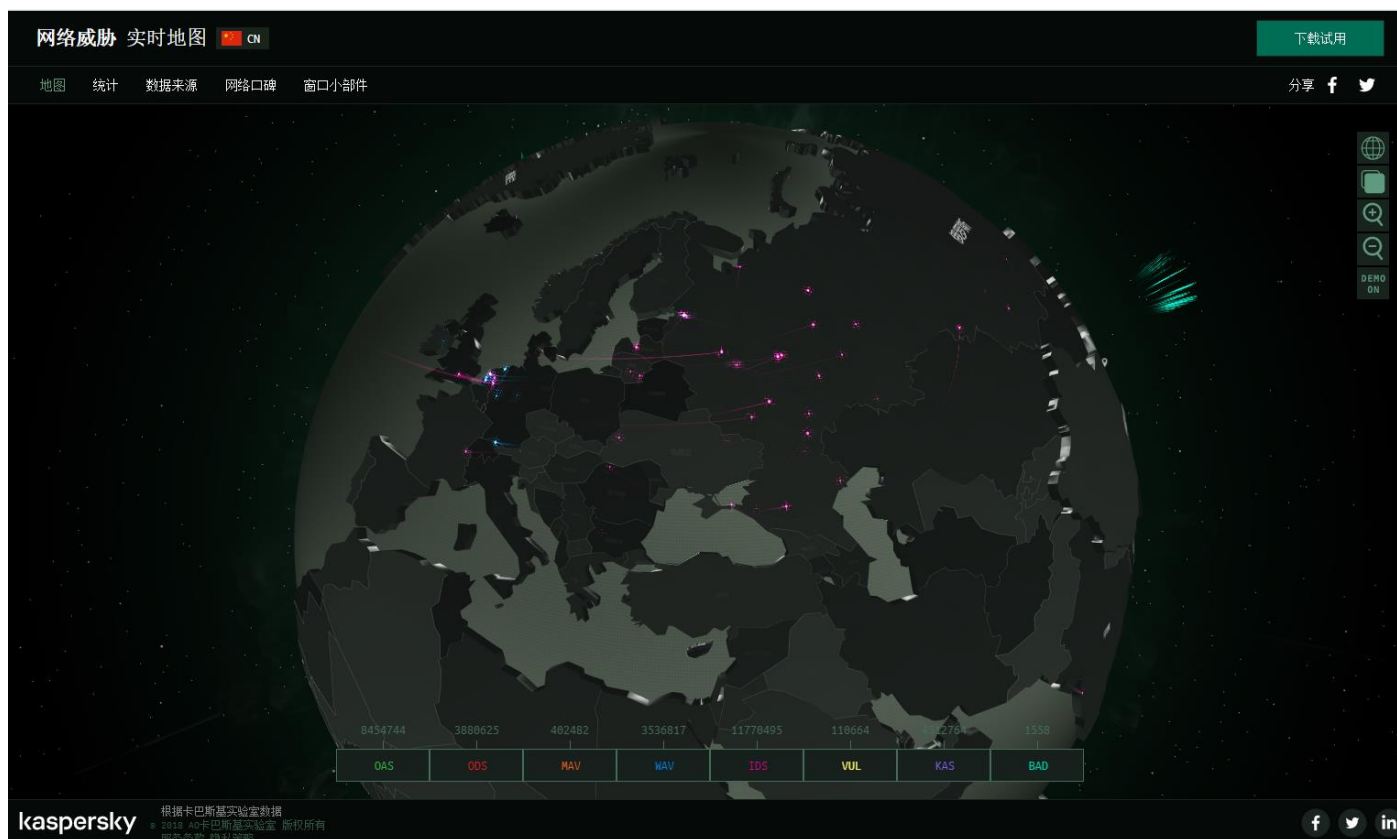
- python的绘图工具，用来对python计算后的数据进行绘制



# 可视化典型案例：全球黑客活动

- 卡巴斯基利用“蜜罐”攻击陷阱显示出所有实时渗透攻击活动。地图中的每一条线代表的都是一次攻击活动，借此可以了解每一天、每一分钟甚至每一秒世界上发生了多少次恶意渗透。

<https://cybermap.kaspersky.com/>

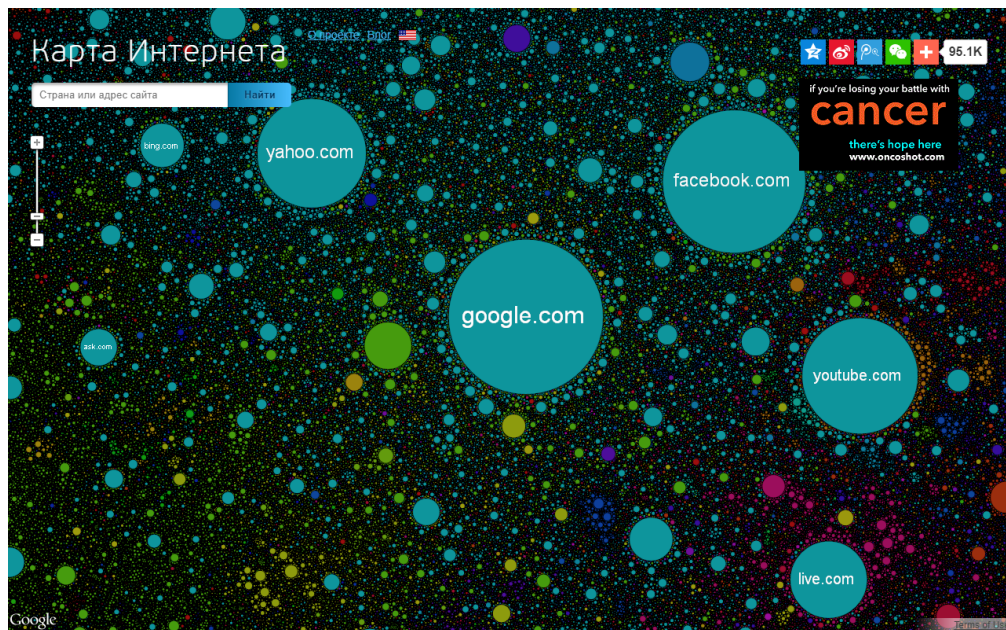




# 可视化典型案例：互联网地图

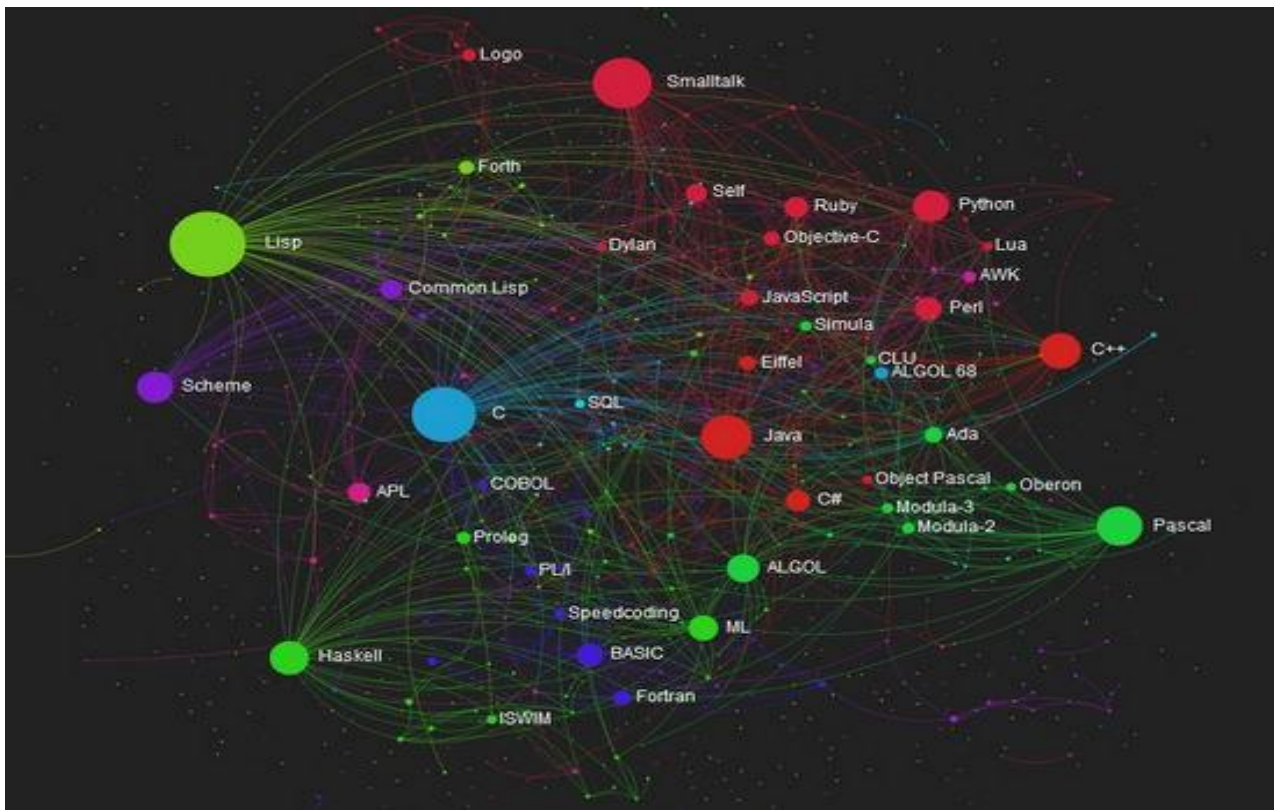
• 为了探究互联网这个庞大的宇宙，俄罗斯工程师 Ruslan Enikeev 将全球 196 个国家的 35 万个网站数据整合起来，并根据 200 多万个网站链接将这些“星球”通过关系链联系起来，每一个“星球”的大小根据其网站流量来决定，而“星球”之间的距离远近则根据链接出现的频率、强度和用户跳转时创建的链接来确定，由此绘制得到了“互联网地图”。

<http://internet-map.net>



# 可视化典型案例：编程语言之间的影响力

- Ramio Gómez利用来自Freebase上的编程语言维护表里的数据，绘制了编程语言之间的影响力关系图，图中的每个节点代表一种编程语言，之间的连线代表该编程语言对其他语言有影响，有影响力的语言会连线多个语言，相应的节点也会越大。



# 可视化典型案例：百度迁徙

- 2014年1月25日晚间，央视与百度合作，启用百度地图定位可视化大数据播报春节期间全国人口迁徙情况，引起广泛关注。



# 可视化工具：Echarts

---

- ECharts是由百度商业前端数据可视化团队研发的图表库，可以流畅的运行在 PC 和移动设备上，兼容当前绝大部分浏览器（IE8/9/10/11，Chrome，Firefox，Safari等），底层依赖轻量级的 Canvas 类库 ZRender，提供直观，生动，可交互，可高度个性化定制的数据可视化图表
- ECharts 提供了非常丰富的图表类型，常规的折线图，柱状图，散点图，饼图，K线图，用于统计的盒形图，用于地理数据可视化的地图，热力图，线图，用于关系数据可视化的关系图，treemap，多维数据可视化的平行坐标，还有用于 BI 的漏斗图，仪表盘，并且支持图与图之间的混搭，满足用户绝大部分用户分析数据时的图表制作需求

# 可视化工具：Echarts

---

- Echarts开发环境
- ECharts是一款可视化开发库，底层用的是javascript封装，所以可以在网页HTML中嵌入ECharts代码显示数据图表。
- 因为ECharts底层是javascript，所以可以像javascript一样，直接嵌入到HTML中，如下：

```
<!DOCTYPE html>
<html>
<header>
  <meta charset="utf-8">
  <!-- 引入 ECharts 文件 -->
  <script src="echarts.js"> </script>
</header>
</html>
```



# 可视化工具：Echarts

---

- 绘制一个简单的图表
- (1)在绘图前我们需要为 ECharts 准备一个具备高宽的 dom 容器。

```
<body>
```

```
  <!-- 为 ECharts 准备一个具备大小（宽高）的Dom -->
```

```
  <div id="main" style="width: 600px;height:400px;"> </div>
```

```
</body>
```

- (2)然后就可以通过 echarts.init 方法初始化一个 echarts 实例并通过 setOption 方法生成各类图

```
var myChart = echarts.init(document.getElementById('main'));
```

```
var option = { ... }
```

```
myChart.setOption(option);
```



# 可视化工具：Echarts

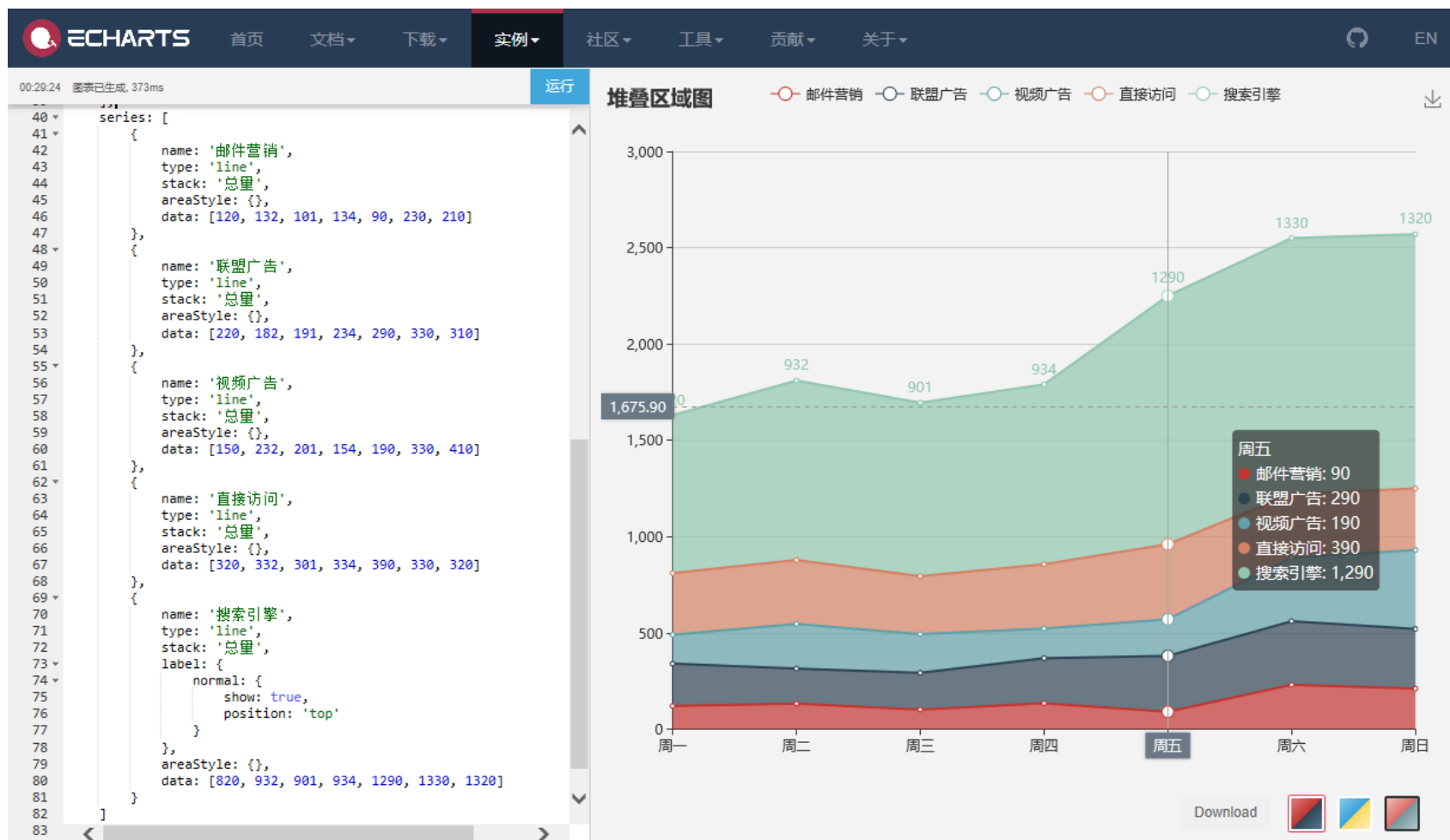
---

- (3)图片导出
- ECharts可以很方便的导出制作的图表只要在代码中加入如下代码，即可显示上图右上角的工具栏，其中最右边蓝框中的图标即为导出图表。

```
toolbox: {  
    show : true,  
    feature : {  
        dataZoom: {},  
        dataView: {readOnly: false},  
        magicType: {type: ['line', 'bar']},  
        restore: {},  
        saveAsImage: {}  
    }  
}
```

# 可视化工具：Echarts

- series（系列列表），即在同一个图表上可以制作一系列的多个图表，然后叠在一起，形成一个图表。



# 可视化工具：D3

---

- D3 的全称是 (Data-Driven Documents) , 顾名思义, 它是一个被数据驱动文档。听名字有点抽象, 说简单一点, 其实就是一个 JavaScript 的函数库, 使用它主要是用来做数据可视化的
- 学习 D3 最好的地方是: <http://d3js.org/>
- D3可视化库的安装
  - D3 是一个 JavaScript 函数库, 并不需要通常所说的 “安装” 。它只有一个文件, 在 HTML 中引用即可。有两种方法:
  - 方法一: 下载 D3.js 的文件, 解压后, 在 HTML 文件中包含相关的 js 文件即可。
  - 方法二: 可以直接包含网络的链接, 这种方法较简单:  
<script src="http://d3js.org/d3.v3.min.js" charset="utf-8"> </script>
  - 但使用的时候要保持网络连接有效, 不能在断网的情况下使用。

# 可视化工具：D3

---

- 预备知识：
- HTML：超文本标记语言，用于设定网页的内容，比如 `<body>` 和 `<p>` 标签
- CSS：层叠样式表，用于设定网页的样式
- JavaScript：一种直译式脚本语言，用于设定网页的行为
- DOM：文档对象模型，用于修改文档的内容和结构
- SVG：可缩放矢量图形，用于绘制可视化的图形

# 可视化工具：D3

---

- 1、添加元素
- 选择body标签，为之添加一个p标签，并设置其内容为New paragraph

```
1. <html>
2.   <head>
3.     <meta charset="utf-8">
4.     <title>D3测试</title>
5.     <script type="text/javascript" src="http://d3js.org/d3.v3.min.js"> </script>
6.   </head>
7.   <body>
8.     <script type="text/javascript">
9.       d3.select("body").append("p").text("New paragraph!");
10.    </script>
11.  </body>
12. </html>
```

# 可视化工具：D3

- 2、数据绑定
- 接受几乎任何数值数组，字符串，或对象（本身包含其他数组或键/值对）。

```
1. <!DOCTYPE html>
2. <html>
3.   <head>
4.     <title>testD3-1.html</title>
5.     <script type="text/javascript" src="http://d3js.org/d3.v3.min.js"></script>
6.   </head>
7.
8.   <body>
9.     This is my HTML page. <br>
10.  </body>
11.  <script type="text/javascript">
12.    var dataset = [ 5, 10, 15, 20, 25 ];
13.
14.    d3.select("body").selectAll("p")
15.      .data(dataset)
16.      .enter()
17.      .append("p")
18.      .text("New paragraph!");
19.  </script>
20. </html>
```



# 可视化工具：D3

---

- 语法说明：
- `d3.select( "body" )`：查找DOM中的body。
- `selectAll( "p" )`：选择DOM中的所有段落。由于没有存在，这将返回一个空的选择。这个选择为空，代表段落很快就会存在。
- `data(dataset)`：计数和分析我们的数据值。有五个值，之后我们的数据集执行了5次，每个值一次。
- `enter()`：绑定数据和DOM元素。这个方法将数据传递到DOM中。如果数据值比相应的DOM元素多，就用`enter()`创建一个新元素的占位符。
- `append( "p" )`：通过`enter()`创建的占位符 在DOM中插入一个p元素。
- `text( "New paragraph!" )`：为新创建的p标签插入一个文本值。

# 可视化工具：D3

- 3、用层画条形图
  - 为同类层添加样式

```
1. div.bar {  
2.     display: inline-block;  
3.     width: 20px;  
4.     height: 75px; /* We'll override this later */  
5.     background-color: teal;  
6. }
```

- 声明要为某类层设置属性

```
1. .attr("class", "bar")
```

- 为每个特定的层设置属性

```
1. .style("height", function(d) {  
2.     var barHeight = d * 5; //Scale up by factor of 5  
3.     return barHeight + "px";  
4. });
```

- 设置层之间的间隔

```
1. margin-right: 2px;
```

# 可视化工具：D3

---

- 4、SVG概要：简单形状
- SVG标签包含一些视觉元素，包括矩形，圆形，椭圆形，线条，文字和路径等。
- 基于像素的坐标系，其中浏览器的左上角是原点(0,0)。x,y的正方向分别是右和下。
- 矩形。使用x和y的指定左上角的坐标，width和height指定的尺寸。绘制SVG的矩形可以这样写：

```
<rect x="0" y="0" width="500" height="50"/>
```

- 圆。使用cx和cy，指定指定半径的中心的坐标，和r表示半径。例如：

```
<circle cx="250" cy="25" r="25"/>
```

## 可视化工具：D3

- 椭圆。使用cx和cy，指定指定半径的中心的坐标，rx和ry分别指定x方向和y方向上圆的半径，例如：

```
<ellipse cx="250" cy="25" rx="100" ry="25"/>
```

- 线。使用x1和Y1到指定线的一端的坐标，x2和y2指定的另一端的坐标。stroke指定描边使得线是可见的。例如：

```
<line x1="0" y1="0" x2="500" y2="50" stroke="black"/>
```

- 文本。使用 x和y指定文本的位置。例如：

```
<text x="250" y="25">Easy-peasy</text>
```

- 可以给文本设置样式。例如：

```
<text x="250" y="155" font-family="sans-serif" font-size="25" fill="gray">Easy-peasy</text>
```

# 可视化工具：D3

---

- 5、SVG概要：默认样式
- SVG的默认样式没有中风是黑色填充。如果你想别的，你就必须将样式应用到你的元素。常见的SVG性质：
  - 1. 填充 (fill) -颜色值。正如用CSS，颜色可以被指定为
    - 命名的颜色- green
    - 十六进制值- # 3388aa或38A
    - RGB值- RGB(10,150,20)
    - RGB与Alpha透明度- RGBA(10,150,20,0.5)
  - 2. 描边 (stroke) -颜色值。
  - 3. 描边宽度 (stroke-width) -数字（通常以像素为单位）。
  - 4. 不透明度 (opacity) – 0.0（完全透明）和1.0（完全不透明）之间的数值。
  - 5. 有了文字，也可以使用CSS样式，如：
    - 字体家庭 (font-family)
    - 字体大小 (font-size)

# 可视化工具：D3

```
<svg width=500 height=960>
  <rect x="0" y="0" width="500" height="50"/>
  <ellipse cx="250" cy="225" rx="100" ry="25"/>
  <line x1="0" y1="120" x2="500" y2="50" stroke="black"/>
  <text x="250" y="155" font-family="sans-serif"
    font-size="25" fill="gray">Easy-peasy</text>
  <circle cx="25" cy="80" r="20"
    fill="rgba(128, 0, 128, 0.75)"
    stroke="rgba(0, 255, 0, 0.25)"
    stroke-width="100"/>
  <circle cx="75" cy="80" r="20"
    fill="rgba(0, 255, 0, 0.75)"
    stroke="rgba(0, 0, 255, 0.25)" stroke-width="10"/>
  <circle cx="125" cy="80" r="20"
    fill="rgba(255, 255, 0, 0.75)"
    stroke="rgba(255, 0, 0, 0.25)" stroke-width="10"/>
  <rect x="0" y="300" width="30" height="30" fill="purple"/>
  <rect x="20" y="305" width="30" height="30" fill="blue"/>
  <rect x="40" y="310" width="30" height="30" fill="green"/>
  <rect x="60" y="315" width="30" height="30" fill="yellow"/>
  <rect x="80" y="320" width="30" height="30" fill="red"/>
  <circle cx="25" cy="425" r="22" class="pumpkin"/>
  <circle cx="25" cy="525" r="20" fill="rgba(128, 0, 128, 1.0)"/>
  <circle cx="50" cy="525" r="20" fill="rgba(0, 0, 255, 0.75)"/>
  <circle cx="75" cy="525" r="20" fill="rgba(0, 255, 0, 0.5)"/>
  <circle cx="100" cy="525" r="20" fill="rgba(255, 255, 0, 0.25)"/>
  <circle cx="125" cy="525" r="20" fill="rgba(255, 0, 0, 0.1)"/>
  <circle cx="25" cy="625" r="20" fill="purple"
    stroke="green" stroke-width="10"
    opacity="0.9"/>
  <circle cx="65" cy="625" r="20" fill="green"
    stroke="blue" stroke-width="10"
    opacity="0.5"/>
  <circle cx="105" cy="625" r="20" fill="yellow"
    stroke="red" stroke-width="10"
    opacity="0.1"/>
</svg>
```

