In [4]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "last"
```

1 pyecharts 库的基本使用用法

- ECharts是由百度团队开发的，可高度个性化定制的数据可视化图表库。
- pyecharts 是一个用于生成 Echarts 图表的类库。实际上就是 Echarts 与 Python 的对接。
- 使用 pyecharts 可以生成独立的网页，也可以在 flask , Django 中集成使用。
- pyecharts包含的图表
  - Bar（柱状图/条形图）
  - Bar3D（3D 柱状图）
  - Boxplot（箱形图）
  - EffectScatter（带有涟漪特效动画的散点图）
  - Funnel（漏斗图）
  - Gauge（仪表盘）
  - Geo（地理坐标系）
  - Graph（关系图）
  - HeatMap（热力图）

由于pytechart并没有内置在Anaconda中，因此需要额外通过pipi来安装pyecharts：
pip install pyecharts==0.5.11
pip install pyecharts_snapshot

In [3]:
```
# 导入pyecharts
import pyecharts
print(dir(pyecharts))
```

```
['Bar', 'Bar3D', 'Boxplot', 'Candlestick', 'EffectScatter', 'Funnel', 'Gauge', 'Geo', 'GeoLines', 'Graph', 'Grid', 'HeatMap', 'Kline', 'Line', 'Line3D', 'Liquid', 'Map', 'NULL', 'Overlap', 'Page', 'Parallel', 'Pie', 'Polar', 'Radar', 'Sankey', 'Scatter', 'Scatter3D', 'Style', 'Surface3D', 'ThemeRiver', 'Timeline', 'Tree', 'TreeMap', 'WordCloud', '__author__', '__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name__', '__package__', '__path__', '__spec__', '__version__', '_version', 'base', 'chart', 'charts', 'conf', 'configure', 'constants', 'custom', 'datasets', 'echarts', 'enable_nteract', 'engine', 'exceptions', 'js_extensions', 'jupyter_image', 'online', 'utils']
```

- add()

主要方法，用于添加图表的数据和设置各种配置项

- print_echarts_options()

打印输出图表的所有配置项

- render()

默认将会在根目录下生成一个 render.html 的文件，支持 path 参数，设置文件保存位置，如 render(r"e:\my_first_chart.html")，文件用浏览器打开。

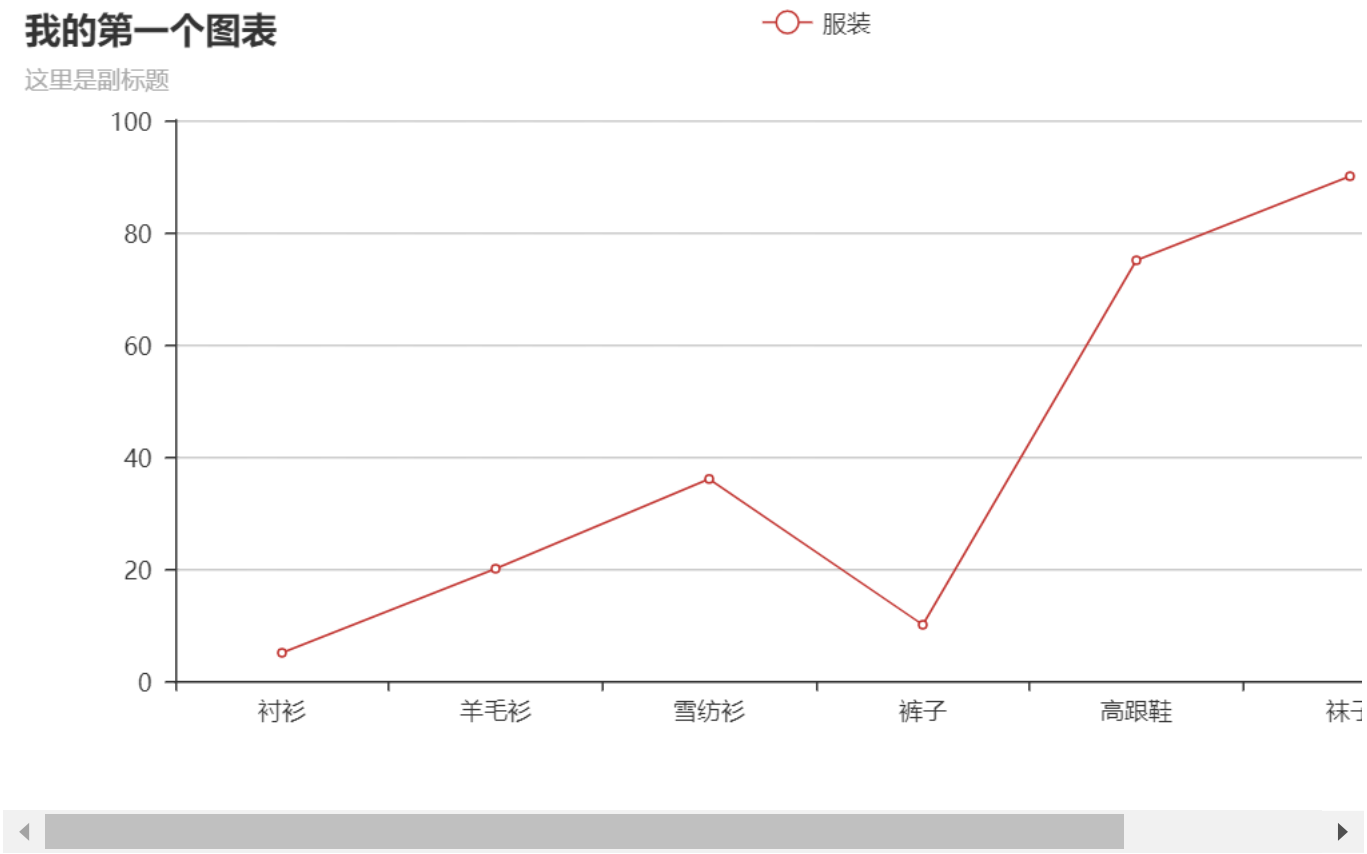
Note： 可以按右边的下载按钮将图片下载到本地，如果想要提供更多实用工具按钮，请在 add() 中设置 is_more_utils 为 True

2 常用图表

2.1 折线图

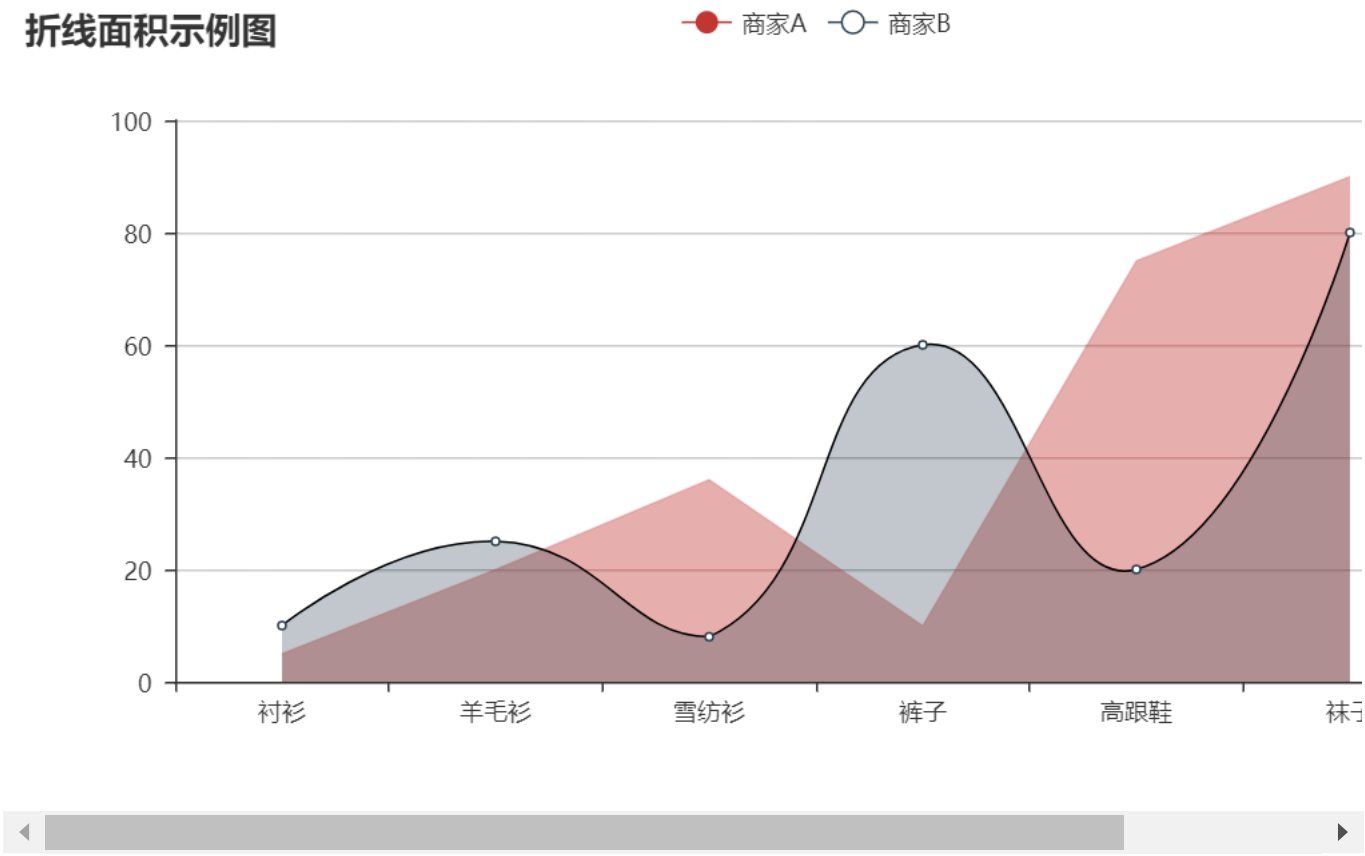
In [5]:
```
from pyecharts import Line

line = Line("我的第一个图表", "这里是副标题")
line.add("服装", ["衬衫", "羊毛衫", "雪纺衫", "裤子", "高跟鞋", "袜子"], [5, 20, 36, 10, 75, 90])
```

Out[5]:



In [6]:
```
from pyecharts import Line

attr = ['衬衫','羊毛衫','雪纺衫','裤子','高跟鞋','袜子']
v1 = [5,20,36,10,75,90]
v2 = [10,25,8,60,20,80]
line = Line('折线面积示例图')
line.add('商家A',attr,v1,
    line_opacity = 0.2,        #线条不透明度
    area_opacity = 0.4,
    symbol = None)

line.add('商家B',attr,v2,
    line_color = '#000',       #黑色
    area_opacity = 0.3,        #填充不透明度
    is_smooth = True)

line
```

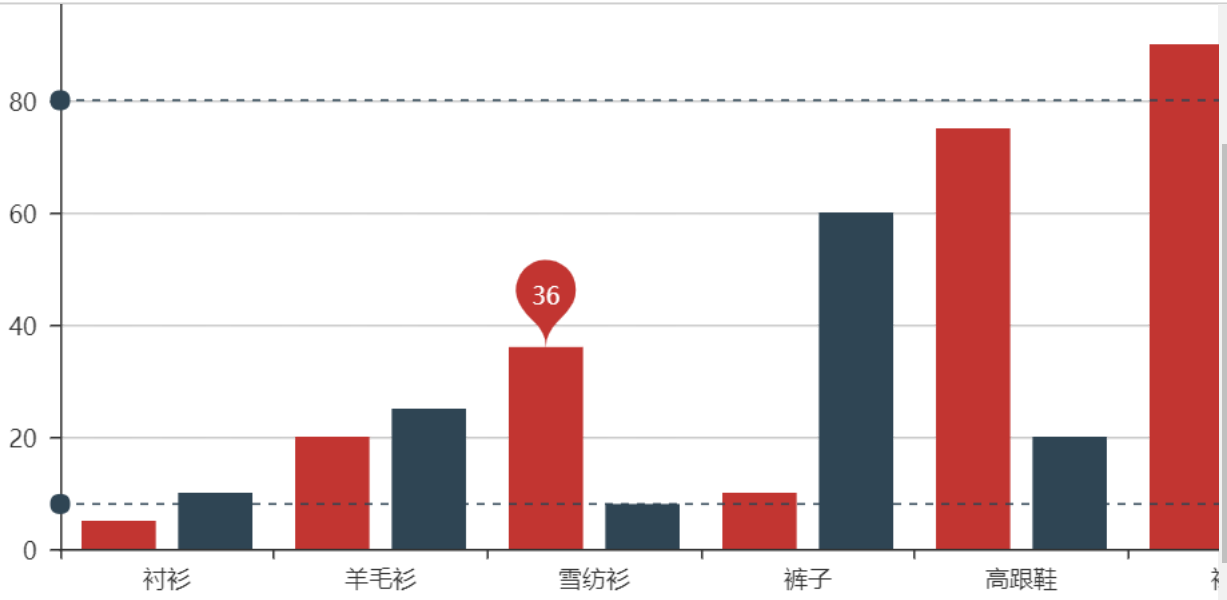
Out[6]:



2.2 柱状图

可以按右边的下载按钮将图片下载到本地，如果想要提供更多实用工具按钮，请在 add() 中设置 is_more_utils 为 True.

In [7]:
```
#导入类库
from pyecharts import Bar

attr = ['衬衫','羊毛衫','雪纺衫','裤子','高跟鞋','袜子']
v1 = [5,20,36,10,75,90]
v2 = [10,25,8,60,20,80]

#创建对象
bar = Bar("柱状图数据堆叠示例","副标题")
#average表示显示平均值
bar.add("商家A",attr,v1,mark_point=["average"],is_stack=False,is_more_utils=True)
#min,max表示标注最大值与最小值点
bar.add("商家B",attr,v2,mark_line=["min","max"])
# #生成html文件
# bar.render("bar.html")

bar
```



2.3 仪表盘

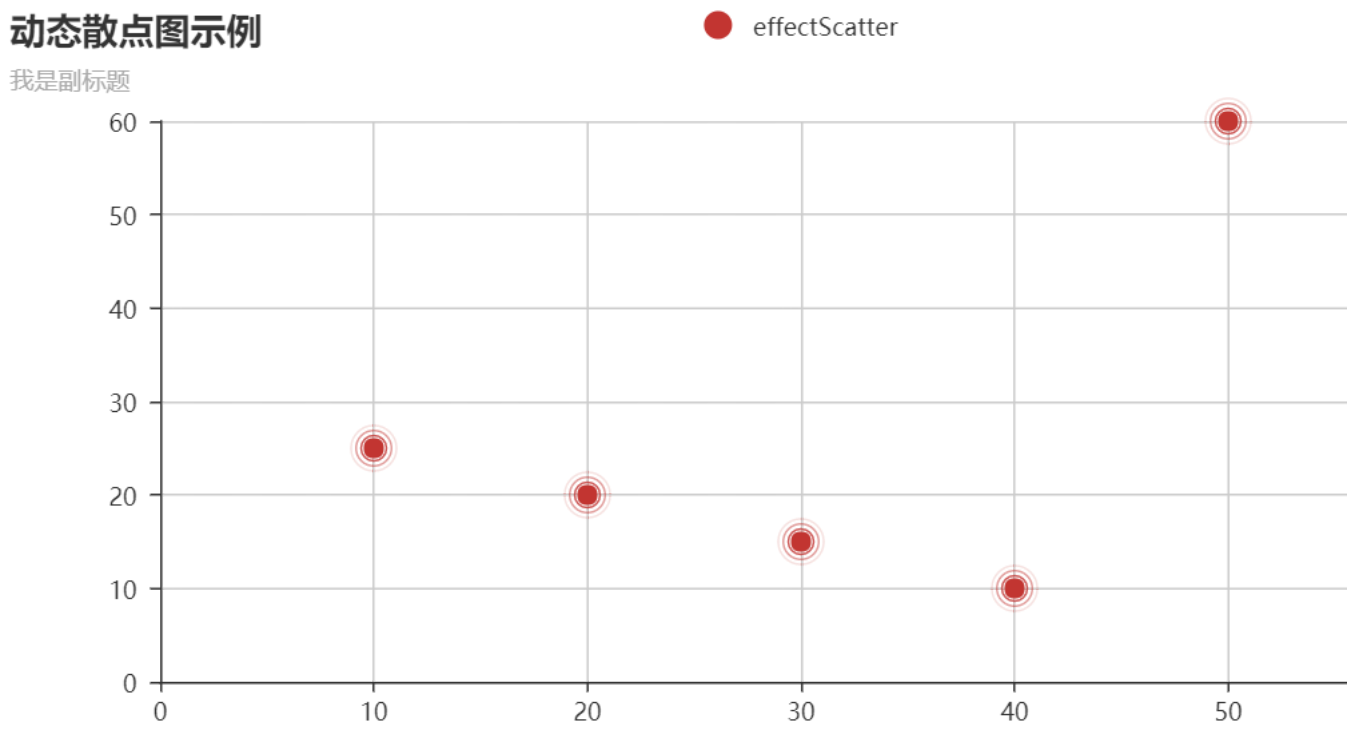
In [8]:
```
#导入类库
from pyecharts import Gauge

#创建对象
gauge = Gauge('仪表盘示例',"我是副标题")
#第三个参数表示比例
gauge.add("业务指标","完成率",66.66,is_more_utils=True)
# #生成html文件
# gauge.render("gauge.html")
gauge
```

Out[8]:

2.4 动态散点图

In [9]:
```
# 导入类库
from pyecharts import EffectScatter

#横坐标
v1 = [10,20,30,40,50,60]
#纵坐标
v2 = [25,20,15,10,60,33]
#创建对象
es = EffectScatter("动态散点图示例","我是副标题")
es.add("effectScatter",v1,v2,is_more_utils=True)
# #生成html文件
# es.render("effectScatter.html")
es
```

Out[9]:



2.5 词云

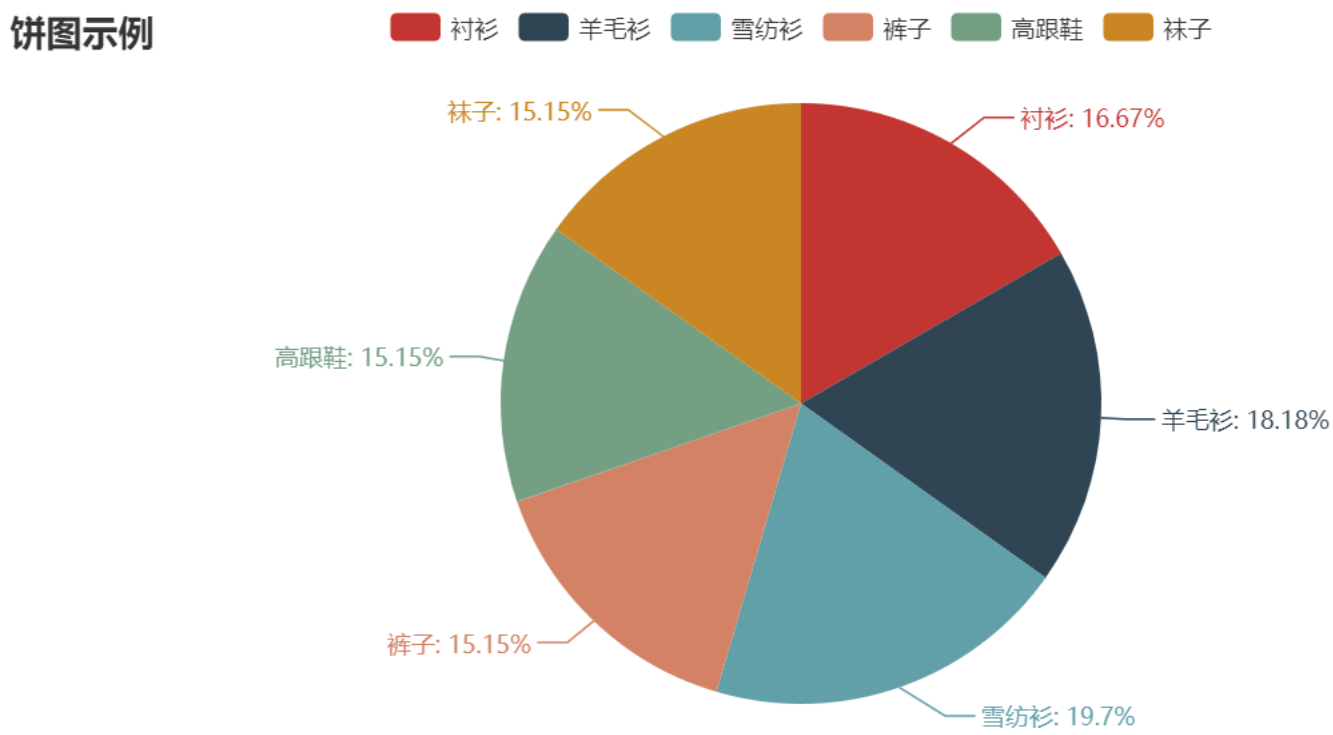
In [10]:
```
#导入类库
from pyecharts import WordCloud

#要显示的每个词
name = ['网络','数据分析',"python",'Hadoop','flask']
#每个词出现的次数
value = [10000,6000,20000,4000,3000]
#创建对象
wd = WordCloud(width=1300,height=620)
#size_range:字体大小范围
wd.add("",name,value,word_size_range=(30,100),is_more_utils=True)
# #生成html文件
# wd.render('wordcloud.html')
wd
```

Out[10]:

2.6 饼图

In [11]:
```
# 导入类库
from pyecharts import Pie

#要显示的内容
attr = ['衬衫','羊毛衫','雪纺衫','裤子','高跟鞋','袜子']
#出现比例或次数
v1 = [11,12,13,10,10,10]
#创建对象
pie = Pie("饼图示例")
pie.add("",attr,v1,is_label_show=True,is_more_utils=True)
# #生成html文件
# pie.render('pie.html')
pie
```

Out[11]:

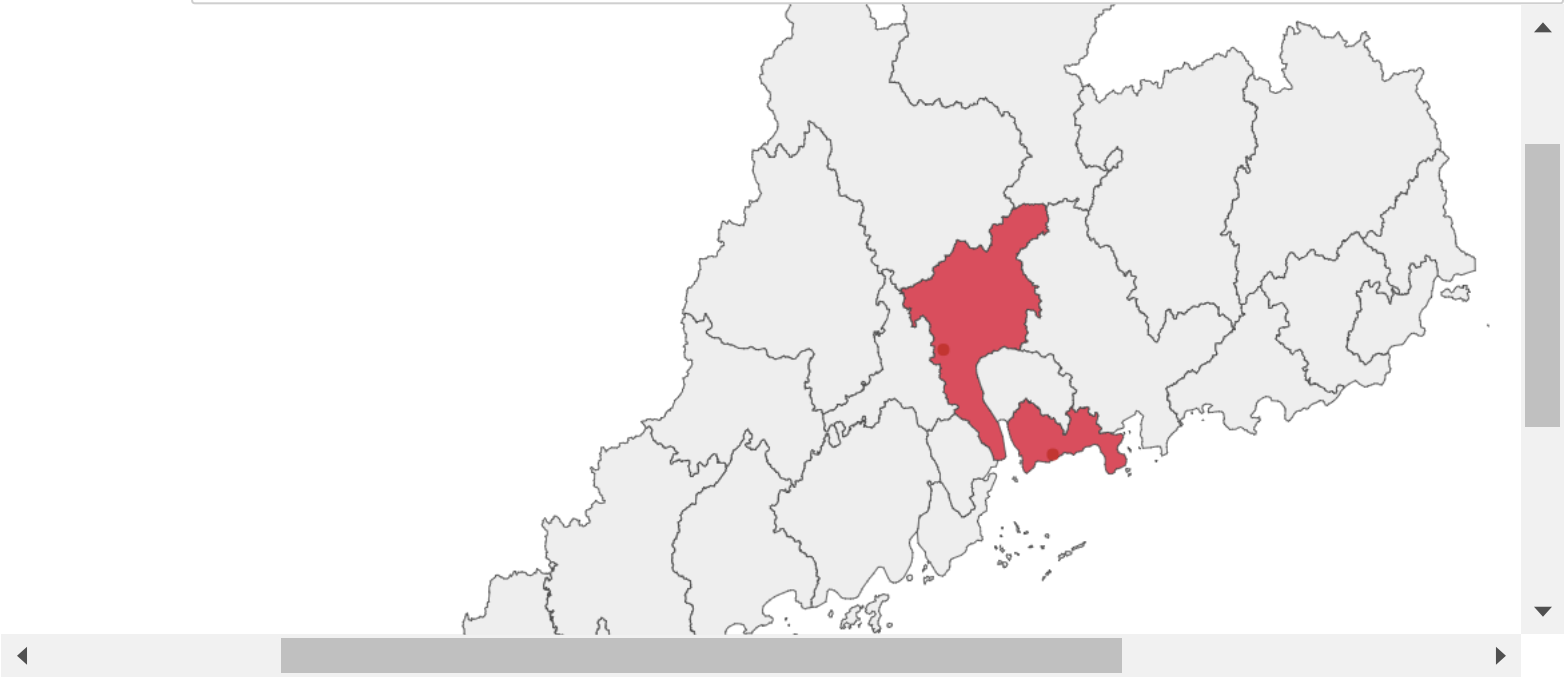


2.7 地图

安装下列地图数据包

- pip install echarts-countries-pypkg
- pip install echarts-china-provinces-pypkg
- pip install echarts-china-cities-pypkg
- pip install echarts-china-counties-pypkg
- pip install echarts-china-misc-pypkg
- pip install echarts-united-kingdom-pypkg

In [12]:
```
from pyecharts import Map

value = [22859,244221]
#城市必须要加上'市', 否则无效
attr = ['广州市','深圳市']
#创建对象
maps = Map("Map结合VisualMap示例",width=1200,height=600)
maps.add("客户",attr,value,maptype='广东',is_visualmap=True,visual_text_color='#007',is_more_util
# #生成html文件
# maps.render('map.html')
maps
```



3 安装和切换主题

echarts-themes-pypkg 提供了 vintage, macarons, infographic, shine 和 roma 主题。

- 安装主题插件:pip install echarts-themes-pypkg
- 切换主题：bar.use_theme("主题名称")

通过具体图类自定义主题只是暂时的设置，如果想更换运行环境内所有图表主题，可做如下操作实现

In [18]:
```
from pyecharts import configure
configure(global_theme='dark')
```

4 图形绘制过程

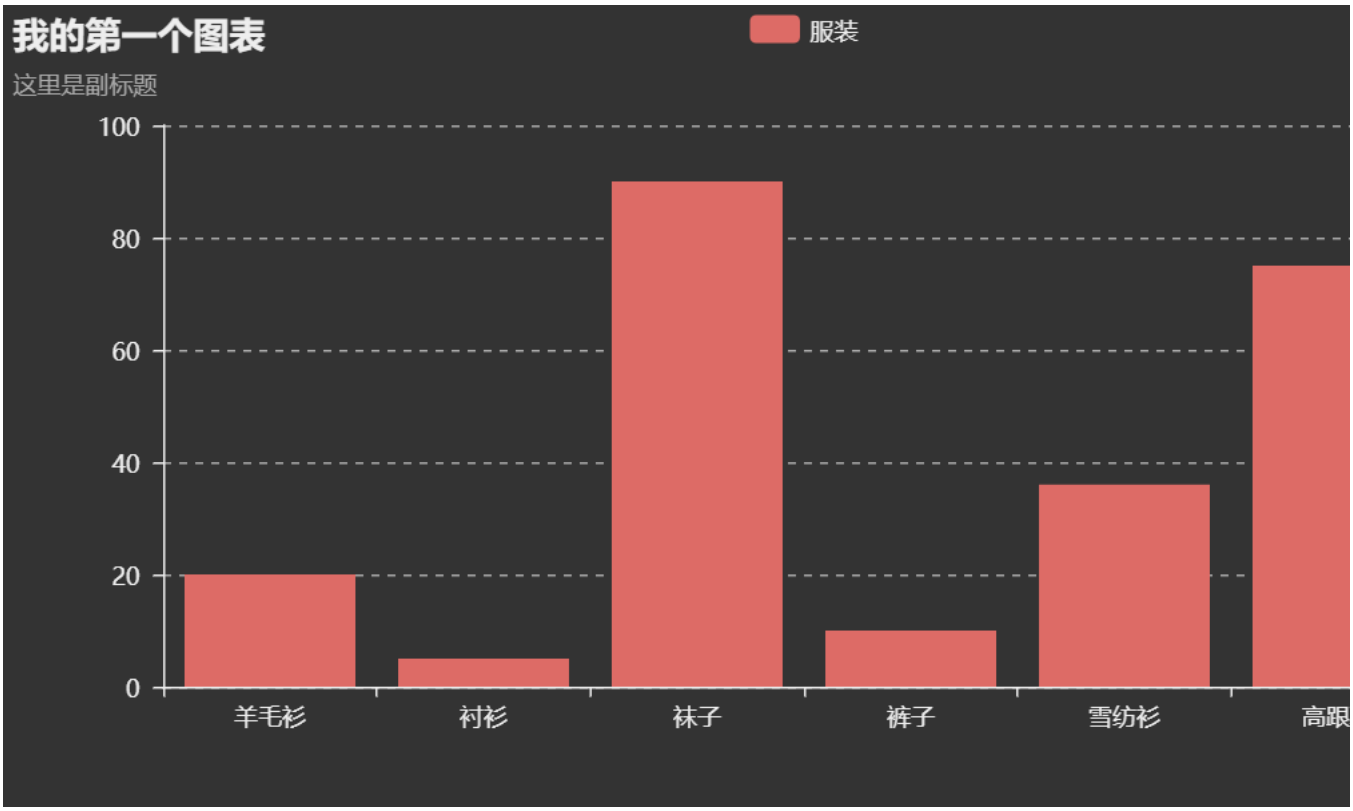
基本上所有的图表类型都是这样绘制的:

- chart_name = Type() 初始化具体类型图表
  - bar = Bar("我的第一个图表", "这里是副标题")
- add() 添加数据及配置项
  - bar.add("服装", ["衬衫", "羊毛衫", "雪纺衫", "裤子", "高跟鞋", "袜子"], [5, 20, 36, 10, 75, 90])
  - add() 数据一般为两个列表（长度一致），如果你的数据是字典或者是带元组的字典。可利用 cast() 方法转换
  - cast(seq)：转换数据序列，将带字典和元组类型的序列转换为 k_lst,v_lst 两个列表
    - 元组列表

        - [(A1, B1), (A2, B2), (A3, B3), (A4, B4)] --> k_lst[A[i1, i2...]], v_lst[B[i1, i2...]]
      - 字典列表
        - [{A1: B1}, {A2: B2}, {A3: B3}, {A4: B4}] --> k_lst[A[i1, i2...]], v_lst[B[i1, i2...]]
      - 字典
        - {A1: B1, A2: B2, A3: B3, A4: B4} -- > k_lst[A[i1, i2...]], v_lst[B[i1, i2...]]
- render() 生成本地文件（html/svg/jpeg/png/pdf/gif)
  - bar.render(path="我的第一个图表.html")

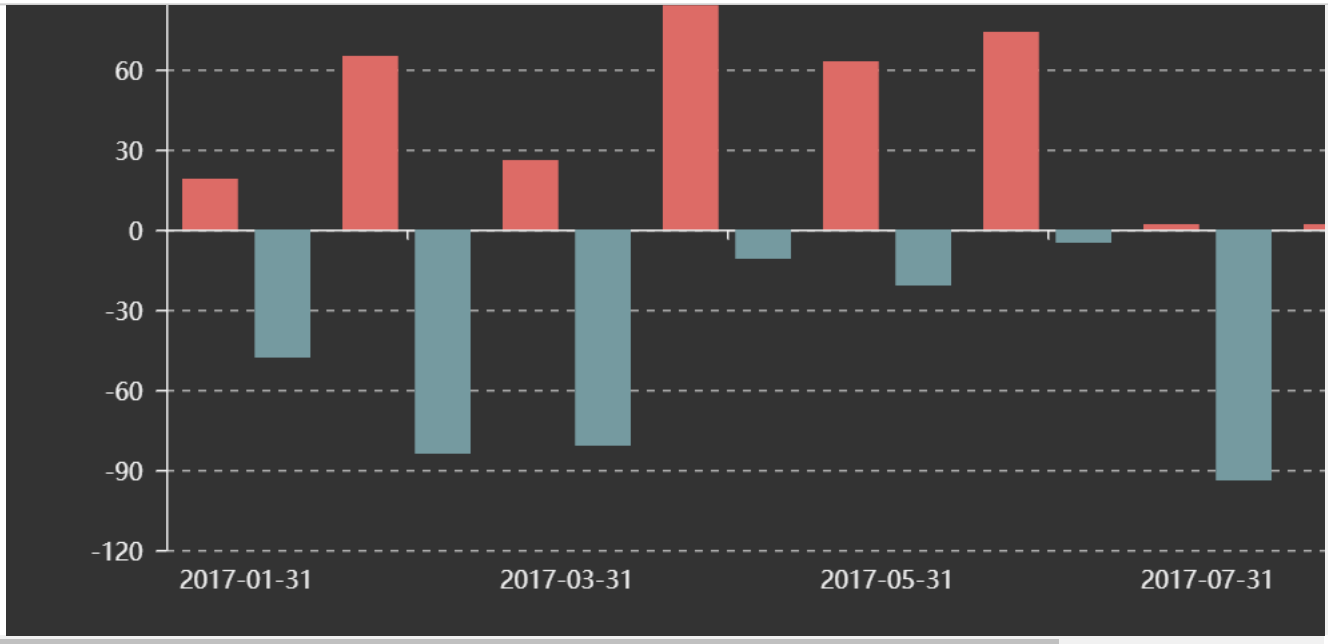
In [14]:
```
from pyecharts import Bar
bar.use_theme("dark")

bar = Bar("我的第一个图表", "这里是副标题")
# 元组列表
# data = [('衬衫', 5), ('羊毛衫', 20), ('雪纺衫', 36), ('裤子', 10), ('高跟鞋', 75), ('袜子', 90)]

# 字典
data = {'羊毛衫': 20, '衬衫': 5, '袜子': 90, '裤子': 10, '雪纺衫': 36, '高跟鞋': 75}

# 字典列表
# data = [{'羊毛衫': 20}, {'衬衫': 5}, {'袜子': 90}, {'裤子': 10}, {'雪纺衫': 36}, {'高跟鞋': 75}]

attr, value = bar.cast(data)
bar.add("服装", attr, value)
bar
```

Out[14]:



4.1 使用Pandas&Numpy处理数据

In [15]:
```
import pandas as pd
import numpy as np
np.random.seed(1026)
import pyecharts

index = pd.date_range('1/1/2017', periods=8, freq='M')
profit = np.random.randint(0, 100, 8)
loss = np.random.randint(-100, 0, 8)
_index = [i for i in index.format()]

bar = Bar('损益情况图')
bar.add('收益', _index, profit)
bar.add('损失', _index, loss, is_more_utils=True)
bar
```



5 图表配置

pyecharts 的主要配置文档，介绍关于 pyecharts 的详细配置项,可以参考：https://pyecharts.org/#/zh-cn/global_options (https://pyecharts.org/#/zh-cn/global_options)

图形初始化：图表类初始化所接受的参数（所有类型的图表都一样）。

** Bar(title, subtitle, width, height, title_pos, title_top, title_color, subtitle_color, title_text_size, subtitle_text_size, background_color, page_title, renderer) **

- title -> str
  - 主标题文本，支持 \n 换行，默认为 ""
- subtitle -> str
  - 副标题文本，支持 \n 换行，默认为 ""
- width -> int
  - 画布宽度，默认为 800（px）
- height -> int
  - 画布高度，默认为 400（px）
- title_pos -> str/int
  - 标题距离左侧距离，默认为'left'，有'auto', 'left', 'right', 'center'可选，也可为百分比或整数
- title_top -> str/int
  - 标题距离顶部距离，默认为'top'，有'top', 'middle', 'bottom'可选，也可为百分比或整数
- title_color -> str

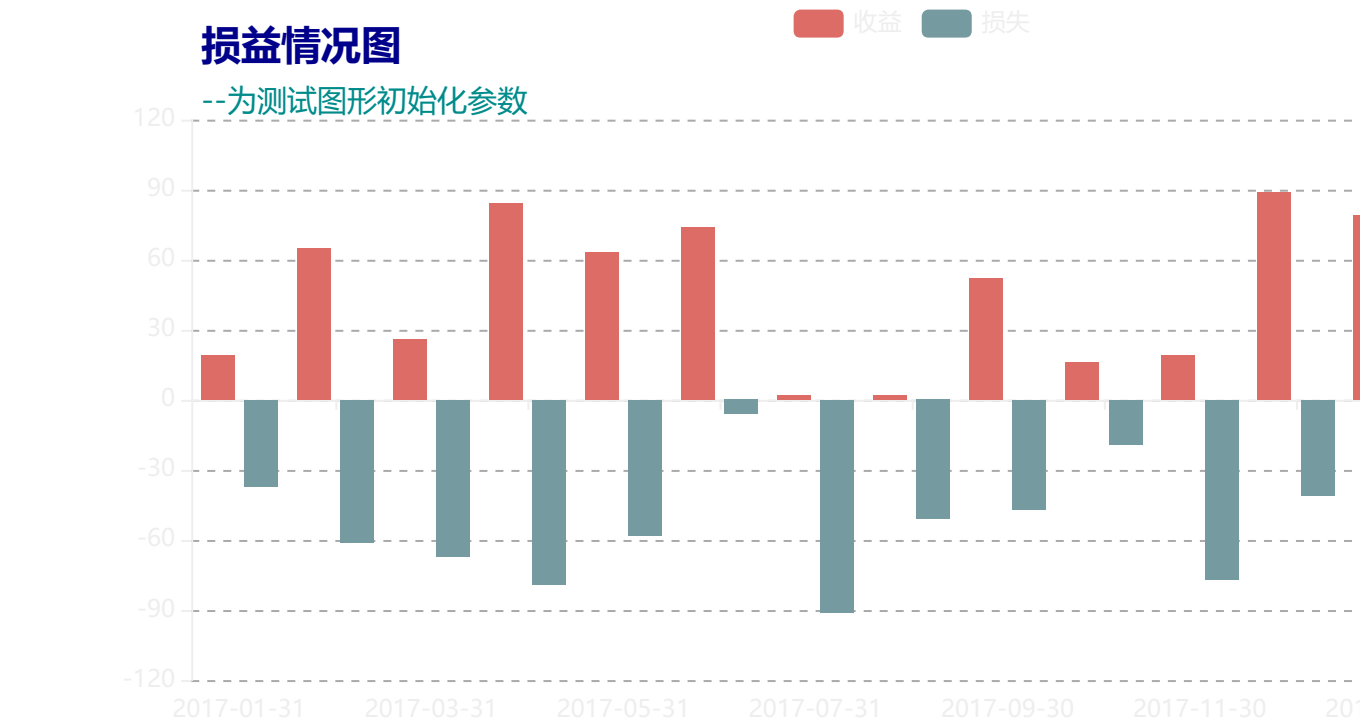
    - 主标题文本颜色，默认为 '#000'
- subtitle_color -> str
    - 副标题文本颜色，默认为 '#aaa'
- title_text_size -> int
    - 主标题文本字体大小，默认为 18
- subtitle_text_size -> int
    - 副标题文本字体大小，默认为 12
- background_color -> str
    - 画布背景颜色，默认为 '#fff'
- page_title -> str
    - 指定生成的 html 文件中 title标签的值。默认为'Echarts'
- renderer -> str
    - 指定使用渲染方式，有 'svg' 和 'canvas' 可选，默认为 'canvas'。3D 图仅能使用 'canvas'。

RGB颜色参考：http://tool.oschina.net/commons?type=3 (http://tool.oschina.net/commons?type=3)

In [16]:
```
# 更换运行环境内所有图表主题为默认主题
from pyecharts import configure
configure(global_theme='shine')
```

In [19]:
```
import pandas as pd
import numpy as np
np.random.seed(1026)
from pyecharts import Bar

index = pd.date_range('1/1/2017', periods=15, freq='M')
profit = np.random.randint(0, 100, 15)
loss = np.random.randint(-100, 0, 15)
_index = [i for i in index.format()]

bar = Bar(title='损益情况图', subtitle='--为测试图形初始化参数',
          width=900, height=400,
          title_pos='10%', title_top='2%',
          title_color='#00008B', subtitle_color='#008B8B',
          title_text_size=20, subtitle_text_size=15,
          background_color='#CDC9C9',
          page_title='PyECharts',
          renderer='svg')

bar.add('收益', _index, profit)
bar.add('损失', _index, loss)
bar
```

Out[19]:



通用配置项：通用配置项均在 add() 中设置

- xyAxis：直角坐标系中的 x、y 轴(Line、Bar、Scatter、EffectScatter、Kline)

- dataZoom：dataZoom 组件用于区域缩放，从而能自由关注细节的数据信息，或者概览数据整体，或者去除离群点的影响。(Line、Bar、Scatter、EffectScatter、Kline、Boxplot)
  - is_datazoom_show -> bool
    - 是否使用区域缩放组件，默认为 False
  - datazoom_type -> str
    - 区域缩放组件类型，默认为'slider'，有'slider', 'inside', 'both'可选
  - datazoom_range -> list
    - 区域缩放的范围，默认为[50, 100]

    - datazoom_orient -> str
        - datazoom 组件在直角坐标系中的方向，默认为 'horizontal'，效果显示在 x 轴。如若设置为 'vertical' 的话效果显示在 y 轴。

- legend：图例组件。图例组件展现了不同系列的标记(symbol)，颜色和名字。可以通过点击图例控制哪些系列不显示。
    - is_legend_show -> bool
        - 是否显示顶端图例，默认为 True
    - legend_orient -> str
        - 图例列表的布局朝向，默认为'horizontal'，有'horizontal', 'vertical'可选
    - legend_pos -> str
        - 图例组件离容器左侧的距离，默认为'center'，有'left', 'center', 'right'可选，也可以为百分数，如"%60"
    - legend_top -> str
        - 图例组件离容器上侧的距离，默认为'top'，有'top', 'center', 'bottom'可选，也可以为百分数，如"%60"
    - legend_selectedmode -> str/bool
        - 图例选择的模式，控制是否可以通过点击图例改变系列的显示状态。默认为'multiple'，可以设成 'single' 或者 'multiple' 使用单选或者多选模式。也可以设置为 False 关闭显示状态。
    - legend_text_size -> int
        - 图例名称字体大小
    - legend_text_color -> str
        - 图例名称字体颜色

- label：图形上的文本标签，可用于说明图形的一些数据信息，比如值，名称等。
    - is_label_show -> bool
        - 是否正常显示标签，默认不显示。标签即各点的数据项信息

- lineStyle：带线图形的线的风格选项(Line、Polar、Radar、Graph、Parallel)
    - line_width -> int
        - 线的宽度，默认为 1
    - line_opacity -> float
        - 线的透明度，0 为完全透明，1 为完全不透明。默认为 1
    - line_curve -> float
        - 线的弯曲程度，0 为完全不弯曲，1 为最弯曲。默认为 0
    - line_type -> str
        - 线的类型，有'solid', 'dashed', 'dotted'可选。默认为'solid'
    - line_color -> str
        - 线的颜色

- grid3D：3D笛卡尔坐标系组配置项，适用于 3D 图形。（Bar3D, Line3D, Scatter3D)

- axis3D：3D 笛卡尔坐标系 X，Y，Z 轴配置项，适用于 3D 图形。（Bar3D, Line3D, Scatter3D)

- visualMap：是视觉映射组件，用于进行『视觉编码』，也就是将数据映射到视觉元素（视觉通道）

- markLine & markPoint：图形标记组件，用于标记指定的特殊数据，有标记线和标记点两种。（Bar、Line、Kline）

- tooltip：提示框组件，用于移动或点击鼠标时弹出数据内容

- toolbox：右侧实用工具箱
  - is_toolbox_show -> bool
    - 指定是否显示右侧实用工具箱，默认为 True。
  - is_more_utils -> bool
    - 指定是否提供更多的实用工具按钮。默认只提供『数据视图』和『下载』按钮

In [20]:

```
import pandas as pd
import numpy as np
np.random.seed(1026)
from pyecharts import Line

profit = np.random.randint(100, 1000, 52).cumsum()
index = ['{}周'.format(i) for i in range(1,53)]

line = Line(title='累计收益图', subtitle='--为测试通用配置项参数', width=1200)
line.add('累计收益', index, profit, is_more_utils=True,
#          is_convert=True,
#          is_xaxislabel_align=True,
#          is_xaxis_inverse=True,
#          is_yaxis_show=False,
         is_splitline_show=False,
         xaxis_interval=2,
         xaxis_margin=10,
         xaxis_name='周数', xaxis_name_size=16, xaxis_name_gap=35, xaxis_name_pos='middle',
#          xaxis_pos='top',  # x 坐标轴位置，有'top','bottom'可选
         xaxis_rotate=30,  # x 轴刻度标签旋转的角度，默认为 0，即不旋转。旋转的角度从 -90 度到 90
         yaxis_name='累计收益', yaxis_name_size=16, yaxis_name_gap=50, yaxis_name_pos='middle',

#          datazoom
         is_datazoom_show=True,
         datazoom_type='both',  # 区域缩放组件类型，默认为'slider'，有'slider', 'inside', 'both'
         datazoom_range=[20,50],  # 区域缩放的范围，默认为[50, 100]

#          label：图形上的文本标签，可用于说明图形的一些数据信息，比如值，名称等。
         is_label_show=True,
#          label_formatter='{b} - {c}',

#          lineStyle：带线图形的线的风格选项(Line、Polar、Radar、Graph、Parallel)
         line_width=5,
         line_opacity=.3,
         line_curve=.7,
         line_type='dotted',  # 线的类型，有'solid', 'dashed', 'dotted'可选。默认为'solid'
         line_color='green',
        )
line
```

Out[20]: