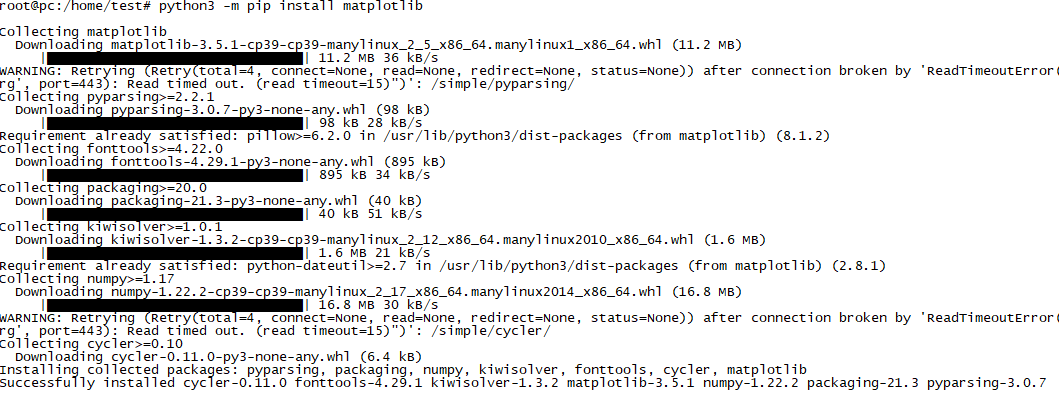
数据可视化

# 安装

Matplotlib和Pygal是两个极具代表性的功能包。

python3 -m pip install matplotlib



python3 -m pydoc --help

root@pc:/home/test# python3 -m pydoc -p 8899

Server ready at http://localhost:8899/

Server commands: [b]rowser, [q]uit

server>



# 折线图

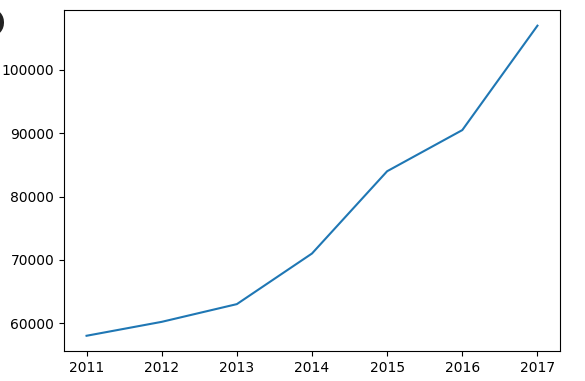
import matplotlib.pyplot as plt

x\_data=['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data=[58000, 60200, 63000, 71000, 84000, 90500, 107000]

plt.plot(x\_data, y\_data)

plt.show()



import matplotlib.pyplot as plt

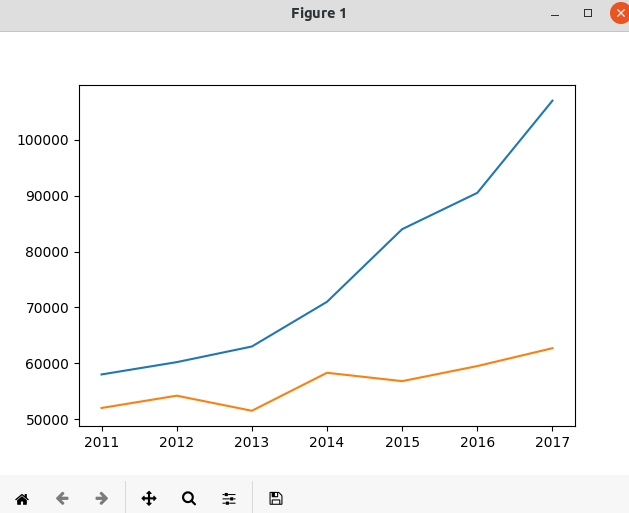
x\_data=['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data=[58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2=[52000, 54200, 51500, 58300, 56800, 59500, 62700]

plt.plot(x\_data, y\_data, x\_data, y\_data2)

plt.show()



import matplotlib.pyplot as plt

x\_data=['2011', '2012', '2013', '2014', '2015', '2016', '2017']

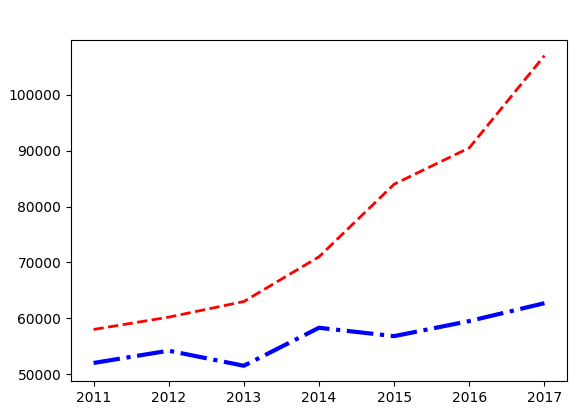
y\_data=[58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2=[52000, 54200, 51500, 58300, 56800, 59500, 62700]

plt.plot(x\_data, y\_data, color='red', linewidth=2.0, linestyle='--')

plt.plot(x\_data, y\_data2, color='blue', linewidth=3.0, linestyle='-.')

plt.show()



## 管理图例

对于复式折线图，应该为每条折线都添加图例，可通过legend()函数来实现。

legend可以传两个list参数：

第一个list用于引用折线图上的每条折线(hanles参数)

第二个list代表为每条这天添加的图例(label参数)

matplotlib也允许在调用plot函数时为每条折线分别传入label参数，这样在调用legend时无须传入labels, handles参数了。

import matplotlib.pyplot as plt

x\_data=['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data=[58000, 60200, 63000, 71000, 84000, 90500, 107000]

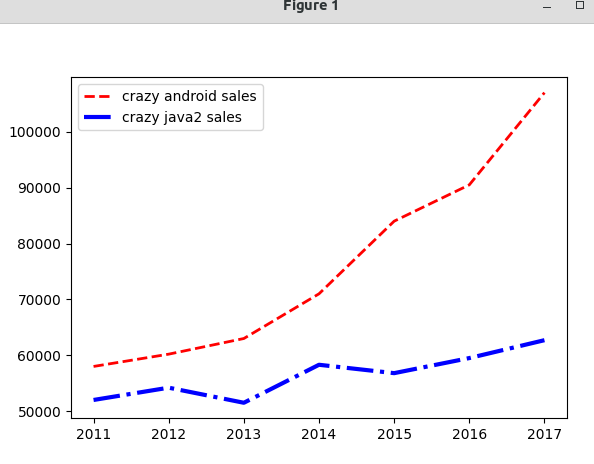
y\_data2=[52000, 54200, 51500, 58300, 56800, 59500, 62700]

ln1, = plt.plot(x\_data, y\_data, color='red', linewidth=2.0, linestyle='--')

ln2, = plt.plot(x\_data, y\_data2, color='blue', linewidth=3.0, linestyle='-.')

plt.legend(handles=[ln1, ln2], labels=['crazy android sales', 'crazy java2 sales'])

plt.show()



import matplotlib.pyplot as plt

x\_data=['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data=[58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2=[52000, 54200, 51500, 58300, 56800, 59500, 62700]

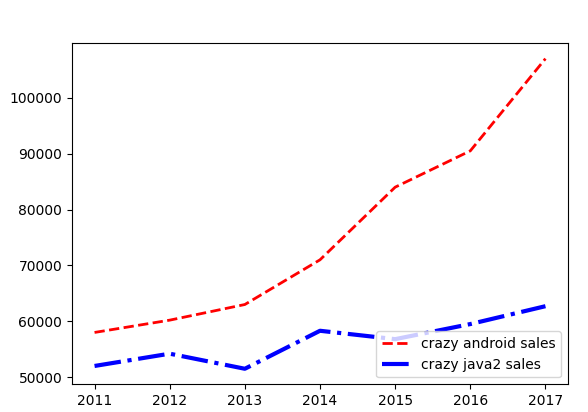
ln1, = plt.plot(x\_data, y\_data, color='red', linewidth=2.0, linestyle='--')

ln2, = plt.plot(x\_data, y\_data2, color='blue', linewidth=3.0, linestyle='-.')

#plt.legend(handles=[ln1, ln2], labels=['crazy android sales', 'crazy java2 sales'])

plt.legend(handles=[ln1, ln2], labels=['crazy android sales', 'crazy java2 sales'], loc='lower right')

plt.show()



## 管理坐标轴

xlabel,ylabel设置坐标轴的名称

xticks, yticks分别改变x轴，y轴的刻度值（允许使用文本作为刻度值）

gca获取坐标轴的信息对象

import matplotlib.pyplot as plt

x\_data=['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data=[58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

plt.plot(x\_data, y\_data, color = 'red', linewidth = 2.0, linestyle = '--', label='crazy java2')

plt.plot(x\_data, y\_data2, color = 'blue', linewidth = 3.0, linestyle = '-.', label='crazy android')

plt.legend(loc='best')

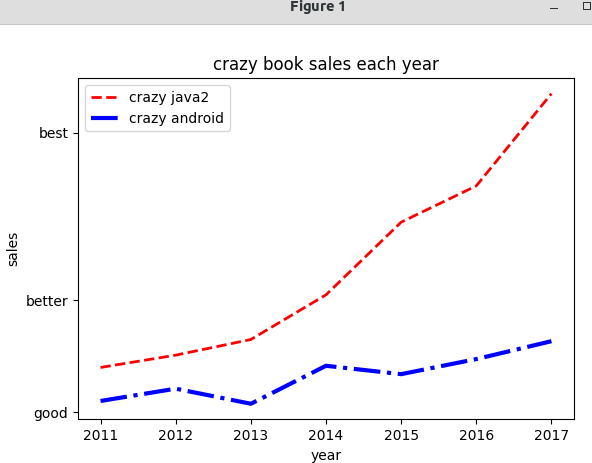
plt.xlabel('year')

plt.ylabel('sales')

plt.title('crazy book sales each year')

plt.yticks([50000, 70000, 100000], [r'good', r'better', r'best'])

plt.show()



import matplotlib.pyplot as plt

x\_data=['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data=[58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2=[52000, 54200, 51500,58300, 56800, 59500, 62700]

plt.plot(x\_data, y\_data, color='red', linewidth=2.0, linestyle='--', label='crazy java2')

plt.plot(x\_data, y\_data2, color='blue', linewidth=3.0, linestyle='-.', label='crazy android')

import matplotlib.font\_manager as fm

my\_font = fm.FontProperties(fname='cmb10.ttf')

plt.legend(loc='best')

plt.xlabel('year')

plt.ylabel('sales')

plt.title('crazy book sales')

plt.yticks([50000, 70000, 100000], [r'good', r'very good', r'very very good'])

ax = plt.gca()

ax.xaxis.set\_ticks\_position('bottom')

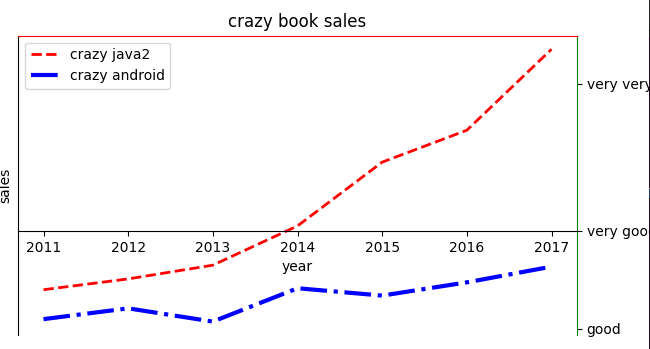
ax.yaxis.set\_ticks\_position('right')

ax.spines['right'].set\_color('green')

ax.spines['top'].set\_color('red')

ax.spines['bottom'].set\_position(('data', 70000))

plt.show()



## 管理多个子图

import matplotlib.pyplot as plt

import numpy as np

plt.figure()

# 定义从-pi到pi之间的数据，平均取64个数据点

x\_data = np.linspace(-np.pi, np.pi, 64, endpoint=True)

# 将整个figure分成两行两列，第三个参数表示该图形放在第1个网格

plt.subplot(2,2,1)

plt.plot(x\_data, np.sin(x\_data))

plt.gca().spines['right'].set\_color('red')

plt.gca().spines['top'].set\_color('blue')

plt.gca().spines['bottom'].set\_position(('data', 0))

plt.gca().spines['left'].set\_position(('data', 0))

plt.title('正弦曲线')

#也可以直接传一个三位数

plt.subplot(222)

plt.plot(x\_data, np.cos(x\_data))

plt.gca().spines['right'].set\_color('none')

plt.gca().spines['top'].set\_color('none')

plt.gca().spines['bottom'].set\_position(('data', 0))

plt.gca().spines['left'].set\_position(('data', 0))

plt.title('余弦曲线')

plt.subplot(223)

plt.plot(x\_data, np.tan(x\_data))

plt.gca().spines['right'].set\_color('none')

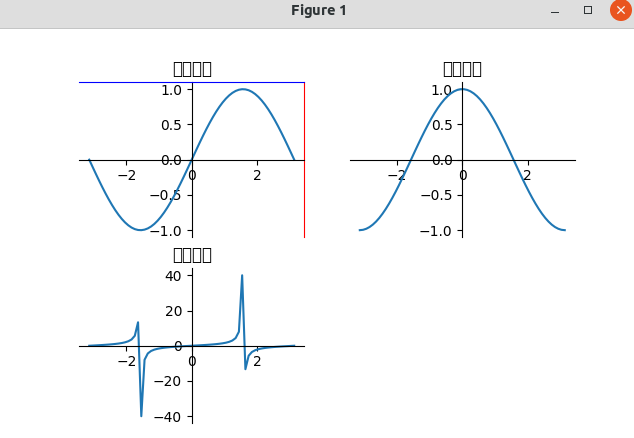
plt.gca().spines['top'].set\_color('none')

plt.gca().spines['bottom'].set\_position(('data', 0))

plt.gca().spines['left'].set\_position(('data', 0))

plt.title('正切曲线')

plt.show()



plt.subplot(2,2,1)

plt.subplot(222)

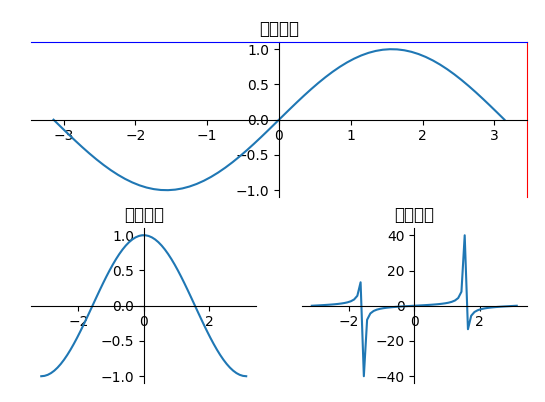
plt.subplot(223)

=>

plt.subplot(2,1,1)

plt.subplot(223)

plt.subplot(224)



import matplotlib.pyplot as plt

import numpy as np

import matplotlib.gridspec as gridspec

plt.figure()

x\_data = np.linspace(-np.pi, np.pi, 64, endpoint=True)

gs = gridspec.GridSpec(2,3) #两行三列

ax1 = plt.subplot(gs[0,:]) #0行

ax2 = plt.subplot(gs[1,0]) #1行0列

ax3 = plt.subplot(gs[1,1:3]) #1行1列和2列

ax1.plot(x\_data, np.sin(x\_data))

ax1.spines['right'].set\_color('red')

ax1.spines['top'].set\_color('blue')

ax1.spines['bottom'].set\_position(('data', 0))

ax1.spines['left'].set\_position(('data', 0))

ax1.set\_title('正弦曲线')

ax2.plot(x\_data, np.cos(x\_data))

ax2.spines['right'].set\_color('none')

ax2.spines['top'].set\_color('none')

ax2.spines['bottom'].set\_position(('data', 0))

ax2.spines['left'].set\_position(('data', 0))

ax2.set\_title('余弦曲线')

ax3.plot(x\_data, np.tan(x\_data))

ax3.spines['right'].set\_color('none')

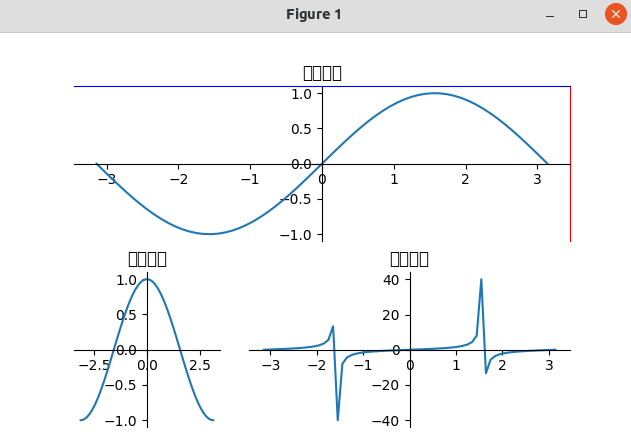
ax3.spines['top'].set\_color('none')

ax3.spines['bottom'].set\_position(('data', 0))

ax3.spines['left'].set\_position(('data', 0))

ax3.set\_title('正切曲线')

plt.show()



# 功能丰富的数据图

## 饼图

plt.pie(x = data, # 绘图数据

labels=labels, # 添加编程语言标签

explode=explode, # 突出显示Python

colors=colors, # 设置饼图的自定义填充色

autopct='%.3f%%', # 设置百分比的格式，此处保留3位小数

pctdistance=0.8, # 设置百分比标签与圆心的距离

labeldistance = 1.15, # 设置标签与圆心的距离

startangle = 180, # 设置饼图的初始角度

center = (4, 4), # 设置饼图的圆心（相当于X轴和Y轴的范围）

radius = 3.8, # 设置饼图的半径（相当于X轴和Y轴的范围）

counterclock = False, # 是否逆时针，这里设置为顺时针方向

wedgeprops = {'linewidth': 1, 'edgecolor':'green'},# 设置饼图内外边界的属性值

textprops = {'fontsize':12, 'color':'black'}, # 设置文本标签的属性值

frame = 1) # 是否显示饼图的圆圈，此处设为显示

import matplotlib.pyplot as plt

data = [0.16881, 0.14966, 0.07471, 0.06992, 0.04762, 0.03541, 0.02925, 0.02411, 0.02316, 0.01409, 0.36326]

labels = ['Java', 'C', 'C++', 'Python', 'Visual Basic .NET', 'C#', 'PHP', 'JavaScript', 'SQL', 'Assembly langugage', 'Other']

#将排在第4位的语言python分离出来

explode = [0, 0, 0, 0.3, 0, 0, 0, 0, 0, 0, 0]

colors = ['red', 'pink', 'magenta','purple','orange']

plt.axes(aspect='equal')

plt.xlim(0,8)

plt.ylim(0,8)

plt.pie(x=data, labels=labels, explode=explode, colors=colors, autopct='%.3f%%', labeldistance=1.15,

startangle=180, center=(4,4), radius=3.8, counterclock=False,

wedgeprops = {'linewidth': 1, 'edgecolor':'green'},

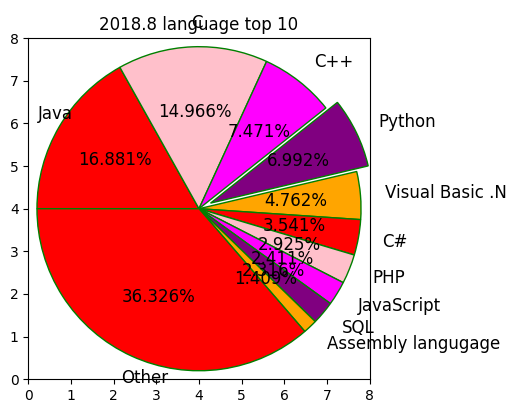
textprops = {'fontsize':12, 'color':'black'}, frame=1)

plt.xticks()

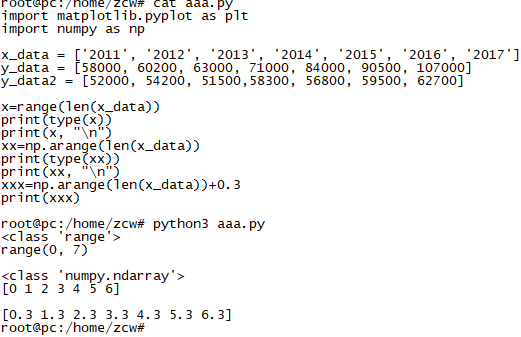
plt.yticks()

plt.title('2018.8 language top 10')

plt.show()



## 柱状图



import matplotlib.pyplot as plt

import numpy as np

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

plt.bar(x=x\_data, height=y\_data, label='crazy java2', color='steelblue', alpha=0.8)

plt.bar(x=x\_data, height=y\_data2, label='crazy android', color='indianred', alpha=0.8)

for x,y in enumerate(y\_data):

plt.text(x, y+100, '%s' % y, ha='center', va='bottom')

for x,y in enumerate(y\_data2):

plt.text(x, y+100, '%s' % y, ha='center', va='top')

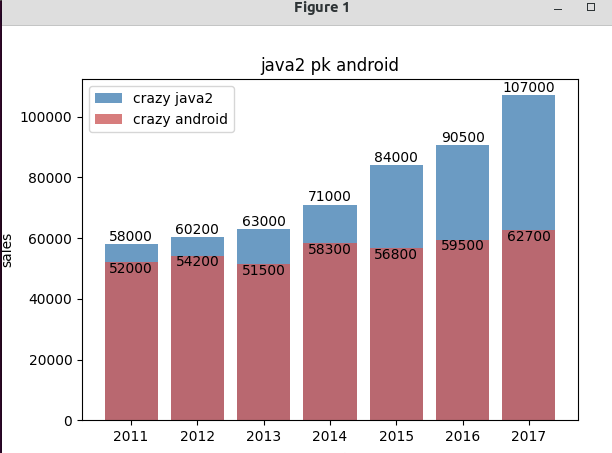
plt.title('java2 pk android')

plt.xlabel('year')

plt.ylabel('sales')

plt.legend()

plt.show()



import matplotlib.pyplot as plt

import numpy as np

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

#plt.bar(x=x\_data, height=y\_data, label='crazy java2', color='steelblue', alpha=0.8)

#plt.bar(x=x\_data, height=y\_data2, label='crazy android', color='indianred', alpha=0.8)

bar\_width=0.3

plt.bar(x=range(len(x\_data)), height=y\_data, label='crazy java2', color='steelblue', alpha=0.8, width=bar\_width)

#0.05使得两根柱子之间有一丝缝隙

plt.bar(x=np.arange(len(x\_data))+bar\_width+0.05, height=y\_data2, label='crazy android', color='indianred', alpha=0.8, width=bar\_width)

for x,y in enumerate(y\_data):

plt.text(x, y+100, '%s' % y, ha='center', va='bottom')

for x,y in enumerate(y\_data2):

plt.text(x, y+100, '%s' % y, ha='center', va='top')

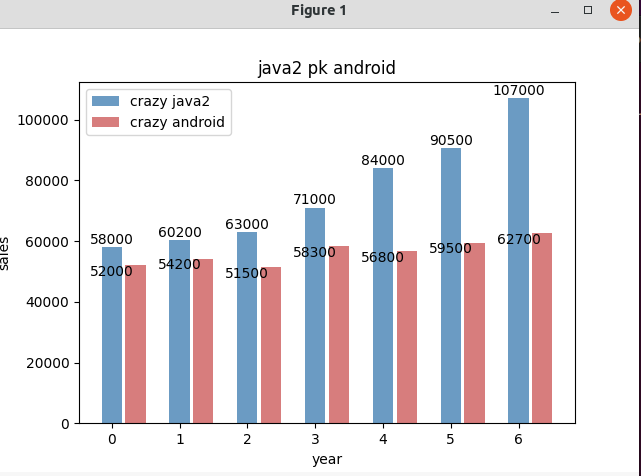
plt.title('java2 pk android')

plt.xlabel('year')

plt.ylabel('sales')

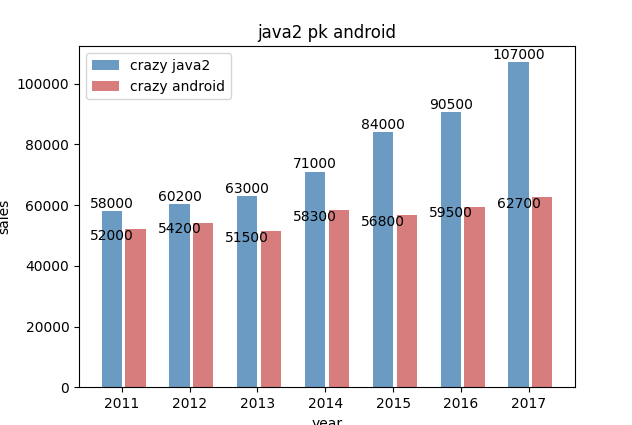
plt.legend()

plt.show()



#为x轴设置刻度

plt.xticks(np.arange(len(x\_data))+bar\_width/2, x\_data)



## 水平柱状图

import matplotlib.pyplot as plt

import numpy as np

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

bar\_width=0.3

plt.barh(y=range(len(x\_data)), width=y\_data, label='crazy java2', color='steelblue', alpha=0.8, height=bar\_width)

plt.barh(y=np.arange(len(x\_data))+bar\_width, width=y\_data2, label='crazy android', color='indianred', alpha=0.8, height=bar\_width)

for y,x in enumerate(y\_data):

plt.text(x+5000, y-bar\_width/2, '%s' % x, ha='center', va='bottom')

for y,x in enumerate(y\_data2):

plt.text(x+5000, y+bar\_width/2, '%s' % x, ha='center', va='top')

plt.title('java2 pk android')

plt.xlabel('year')

plt.ylabel('sales')

plt.legend()

plt.show()



## 散点图

绘制了三幅散点图

plt.scatter(x\_data, np.sin(x\_data), c='purple', # 设置点的颜色

s=50, # 设置点半径

alpha = 0.5, # 设置透明度

marker='p', # 设置使用五边形标记

linewidths=1, # 设置边框的线宽

edgecolors=['green', 'yellow']) # 设置边框的颜色

import matplotlib.pyplot as plt

import numpy as np

plt.figure()

x\_data = np.linspace(-np.pi, np.pi, 64, endpoint=True)

plt.scatter(x\_data, np.sin(x\_data), c='purple', s=50, alpha=0.5, marker='p', linewidths=1, edgecolors=['green', 'yellow'])

plt.scatter(x\_data[0], np.sin(x\_data)[0], c='red', s=150, alpha=1)

plt.scatter(x\_data[63], np.sin(x\_data)[63], c='black', s=150, alpha=1)

plt.gca().spines['right'].set\_color('none')

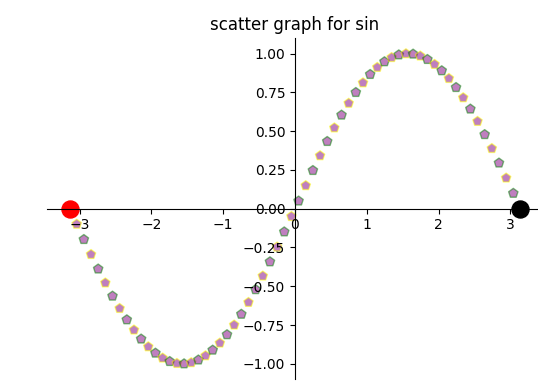
plt.gca().spines['top'].set\_color('none')

plt.gca().spines['bottom'].set\_position(('data', 0))

plt.gca().spines['left'].set\_position(('data', 0))

plt.title('scatter graph for sin')

plt.show()



## 等高线图

contourf: 为等高线填充颜色

contour绘制等高线

import matplotlib.pyplot as plt

import numpy as np

delta=0.025

x = np.arange(-3.0, 3.0, delta)

y = np.arange(-2.0, 2.0, delta)

X,Y = np.meshgrid(x,y)

Z1=np.exp(-X\*\*2 - Y\*\*2)

Z2=np.exp(-(X-1)\*\*2 - (Y-1)\*\*2)

z=(Z1-Z2)\*2

# 为等高线图填充颜色, 16指定将等高线分为几部分

plt.contourf( x,y,z,16, alpha=0.75, cmap='rainbow')

# 绘制等高线

c = plt.contour(x,y,z,16, colors='green', linewidth=0.5)

plt.clabel(c, inline=True, fontsize=10)

plt.xticks(())

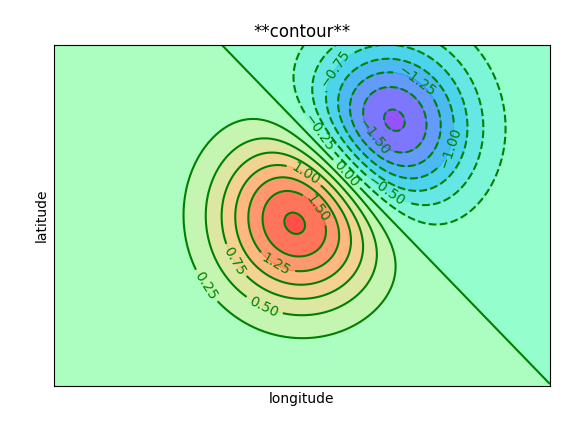
plt.yticks(())

plt.title('\*\*contour\*\*')

plt.xlabel('longitude')

plt.ylabel('latitude')

plt.show()



## 3D图

# 使用Pygal生成数据图(条状图)

import pygal

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

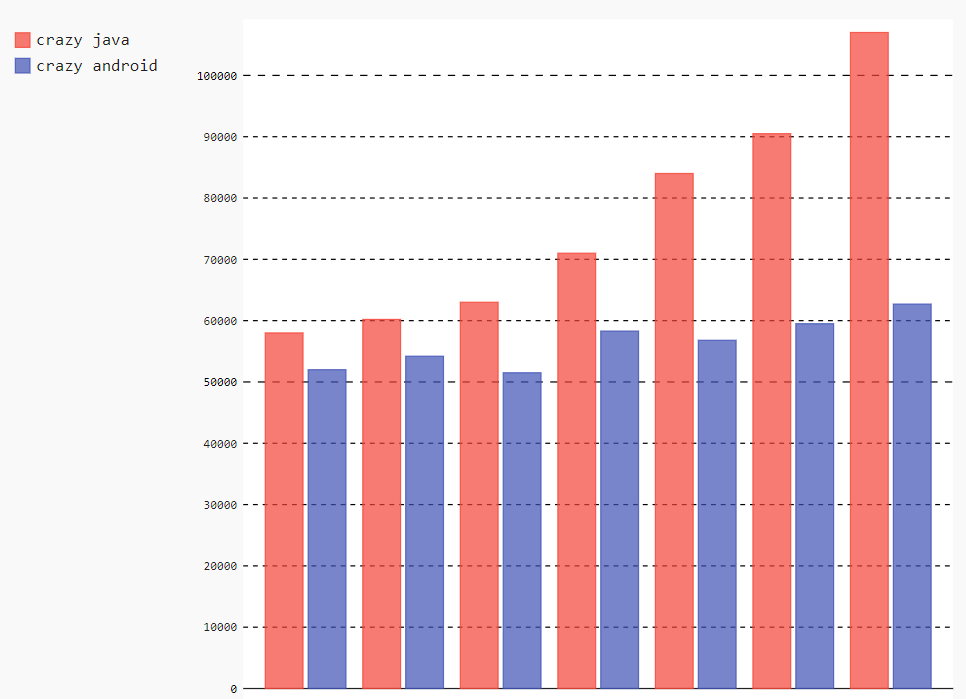
y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

bar = pygal.Bar()

bar.add('crazy java', y\_data)

bar.add('crazy android', y\_data2)

bar.render\_to\_file('fk\_books.svg')



import pygal

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

bar = pygal.Bar()

bar.add('crazy java', y\_data)

bar.add('crazy android', y\_data2)

bar.x\_labels=x\_data

bar.title = 'crazy book sales each year'

bar.x\_lable\_rotation=45 #设置x轴的刻度值旋转45度

bar.legend\_at\_bottom = True

bar.margin=35

bar.show\_y\_guides=True

bar.show\_x\_guides=True

bar.render\_to\_file('222.svg')



# Pygal支持的常见数据图

## 折线图

import pygal

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

line = pygal.Line()

line.add('crazy java', y\_data)

line.add('crazy android', y\_data2)

line.x\_labels=x\_data

line.y\_lables = [20000, 40000, 60000, 80000, 100000]

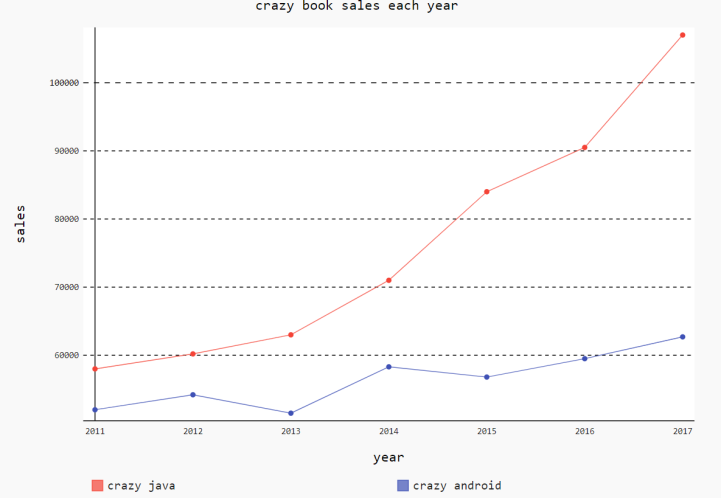
line.title = 'crazy book sales each year'

line.x\_title='year'

line.y\_title='sales'

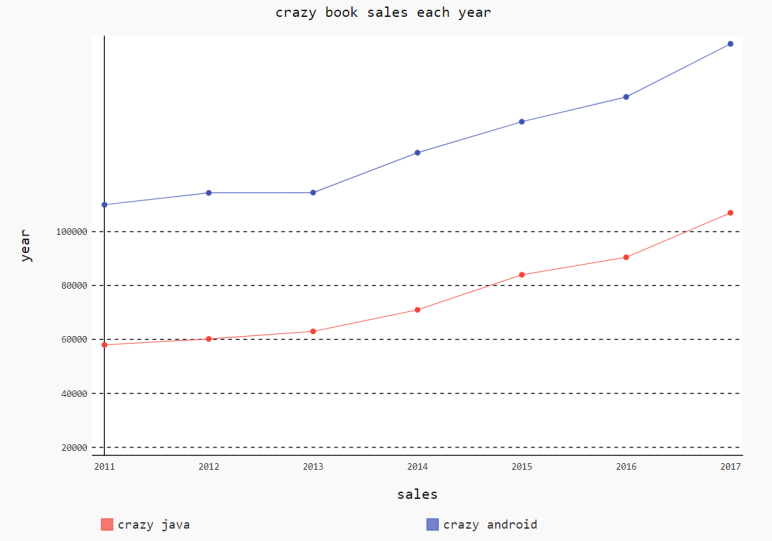
line.legend\_at\_bottom = True

line.render\_to\_file('111.svg')



## 叠加折线图

stacked\_line = pygal.StackedLine()



## 水平折线图

import pygal

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

h\_line = pygal.HorizontalLine()

h\_line.add('crazy java', y\_data)

h\_line.add('crazy android', y\_data2)

h\_line.x\_labels=x\_data

h\_line.y\_labels = [20000, 40000, 60000, 80000, 100000]

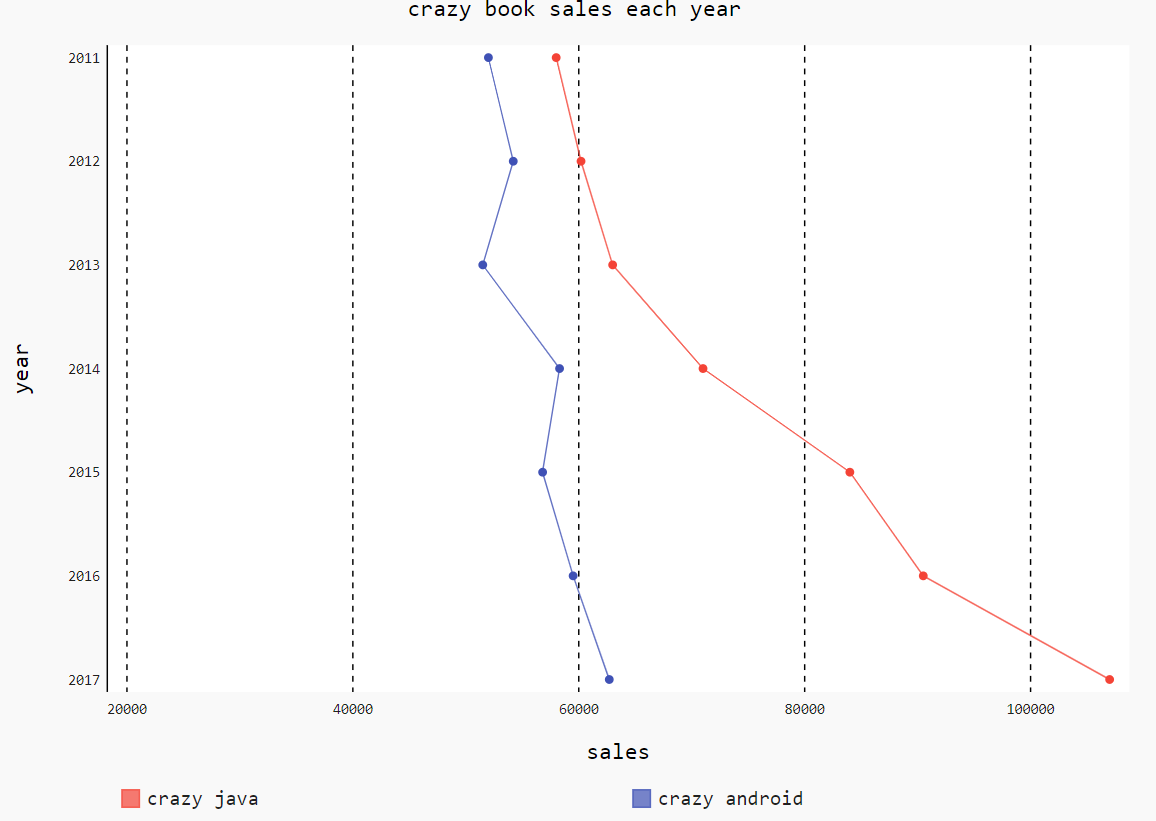
h\_line.title = 'crazy book sales each year'

h\_line.x\_title='sales'

h\_line.y\_title='year'

h\_line.legend\_at\_bottom = True

h\_line.render\_to\_file('222.svg')



## 水平柱状图

%s/h\_line/h\_bar/g

import pygal

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

h\_bar = pygal.HorizontalBar()

h\_bar.add('crazy java', y\_data)

h\_bar.add('crazy android', y\_data2)

h\_bar.x\_labels=x\_data

h\_bar.y\_labels = [20000, 40000, 60000, 80000, 100000]

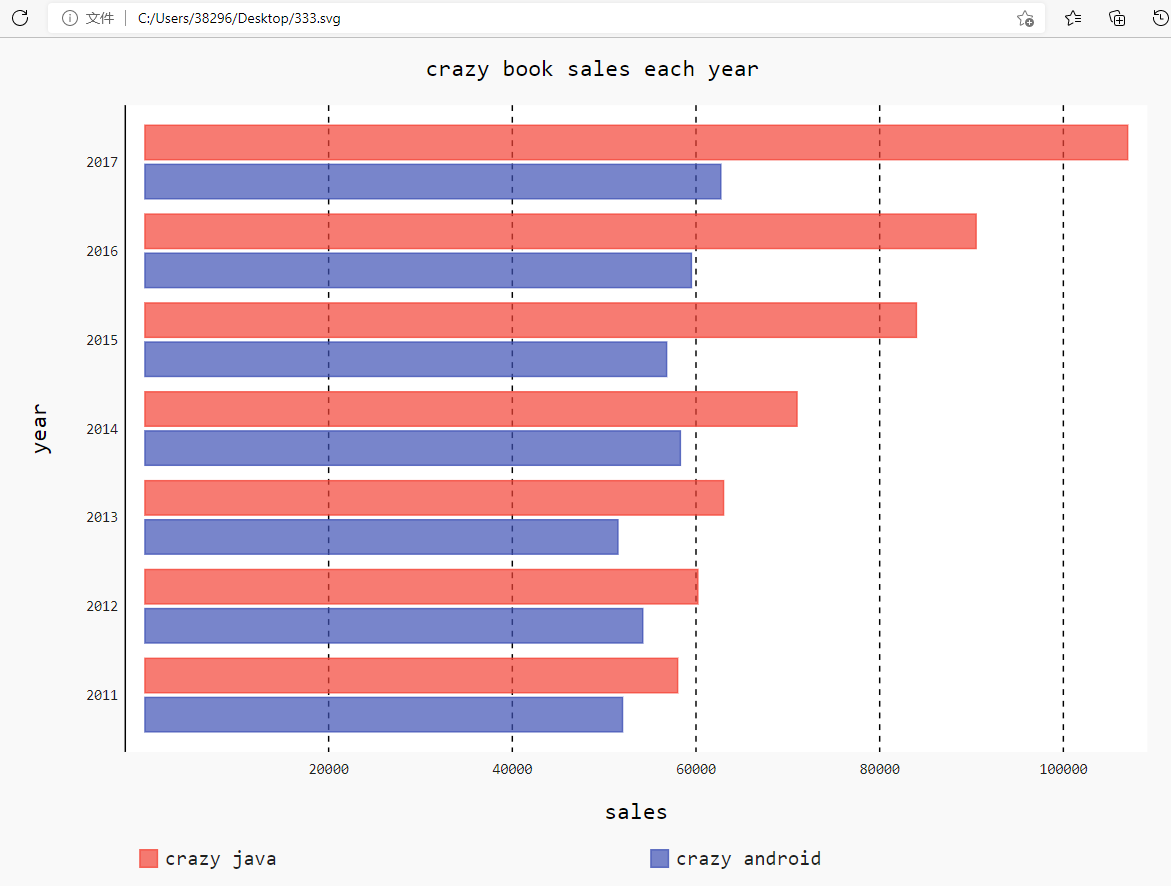
h\_bar.title = 'crazy book sales each year'

h\_bar.x\_title='sales'

h\_bar.y\_title='year'

h\_bar.legend\_at\_bottom = True

h\_bar.render\_to\_file('333.svg')



## 叠加柱状图

import pygal

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

stacked\_bar = pygal.StackedBar()

stacked\_bar.add('crazy java', y\_data)

stacked\_bar.add('crazy android', y\_data2)

stacked\_bar.x\_labels=x\_data

stacked\_bar.y\_labels = [20000, 40000, 60000, 80000, 100000]

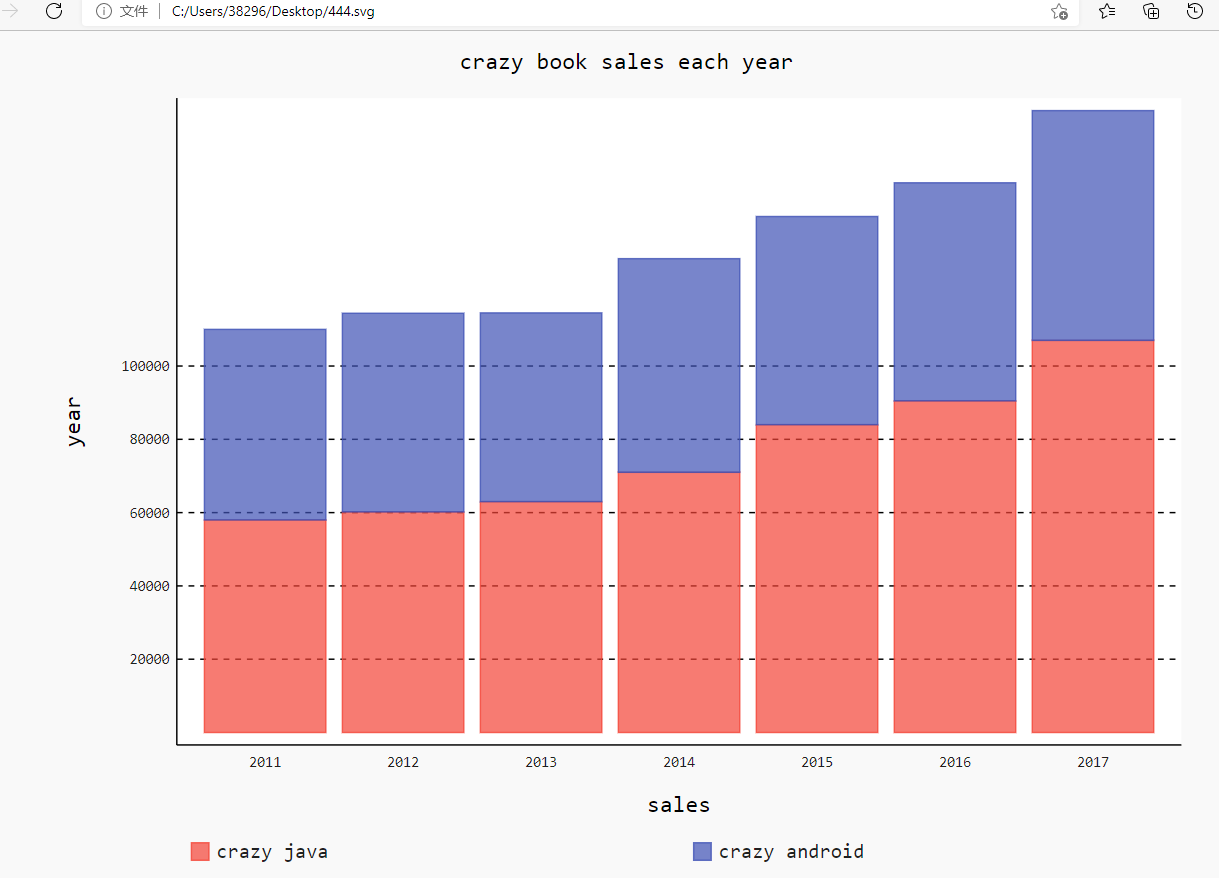
stacked\_bar.title = 'crazy book sales each year'

stacked\_bar.x\_title='sales'

stacked\_bar.y\_title='year'

stacked\_bar.legend\_at\_bottom = True

stacked\_bar.render\_to\_file('444.svg')



## 点图Dot

import pygal

x\_data = ['2011', '2012', '2013', '2014', '2015', '2016', '2017']

y\_data = [58000, 60200, 63000, 71000, 84000, 90500, 107000]

y\_data2 = [52000, 54200, 51500,58300, 56800, 59500, 62700]

dot = pygal.Dot()

dot.dots\_size=5

dot.add('crazy java', y\_data)

dot.add('crazy android', y\_data2)

dot.x\_labels=x\_data

#dot.y\_labels = [20000, 40000, 60000, 80000, 100000]

dot.y\_labels = ['crazy java2', 'crazy android4.4']

dot.y\_label\_rotation=45

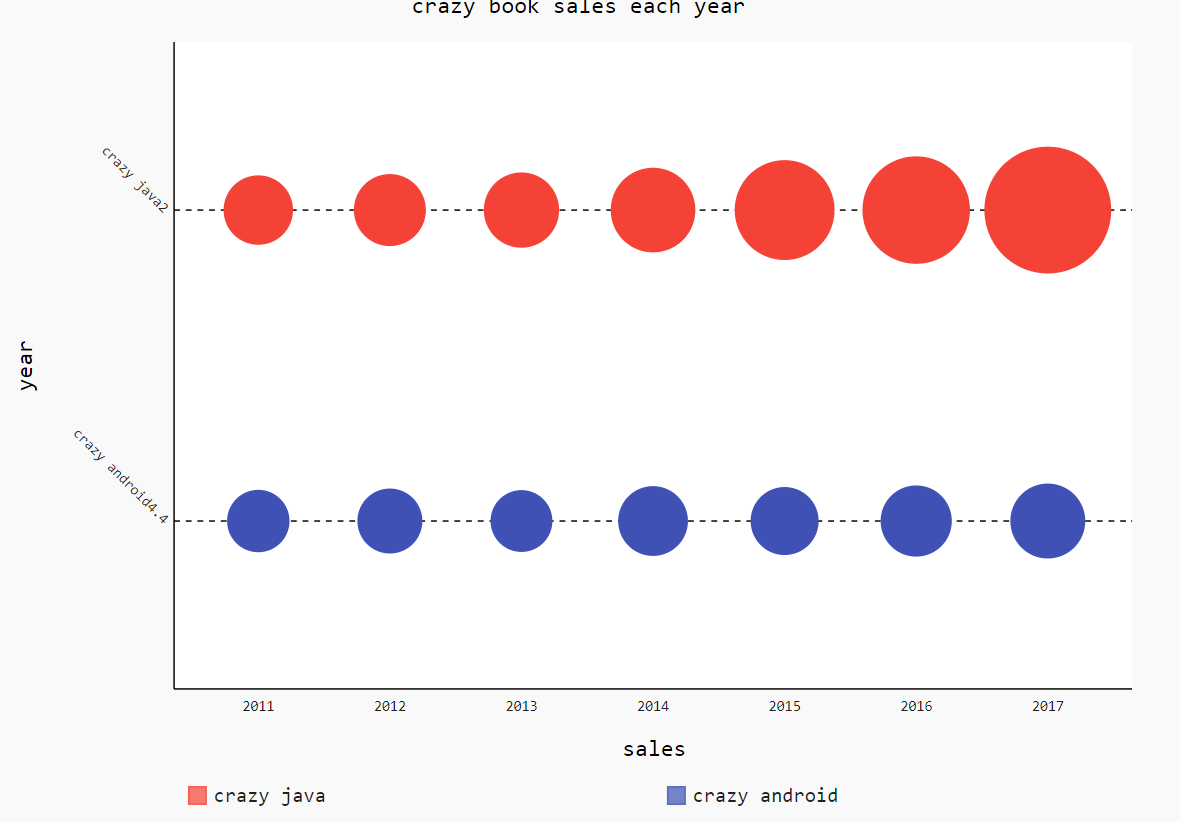
dot.title = 'crazy book sales each year'

dot.x\_title='sales'

dot.y\_title='year'

dot.legend\_at\_bottom = True

dot.render\_to\_file('Dot.svg')



## 饼图Pie

import pygal

data=[0.16881, 0.14966, 0.07471, 0.06992, 0.04762, 0.03541, 0.02925, 0.02411, 0.02316, 0.01409, 0.36326]

labels=['Java', 'C', 'C++', 'Python', 'vb .NET', 'C#', 'php', 'jsp', 'sql', 'assembly language', 'other']

pie = pygal.Pie()

for i,per in enumerate(data):

pie.add(labels[i], per)

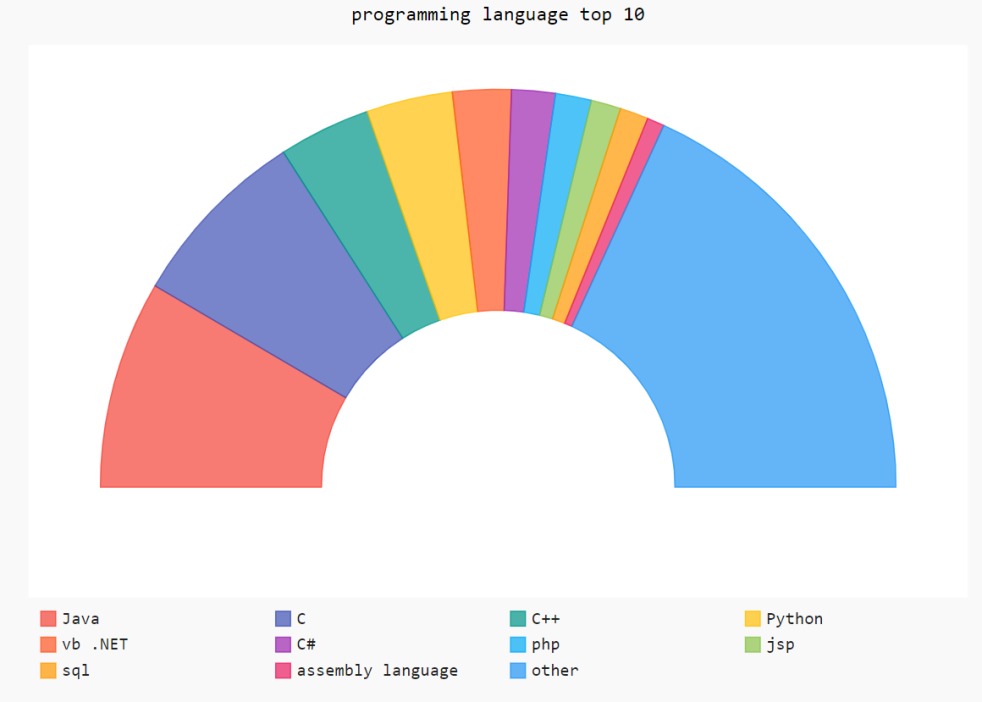
pie.title = 'programming language top 10'

pie.inner\_radius=0.4

pie.legend\_at\_bottom = True

pie.half\_pie=True

pie.render\_to\_file('Pie.svg')



## 仪表图Gauge

import pygal

data=[0.16881, 0.14966, 0.07471, 0.06992, 0.04762, 0.03541, 0.02925, 0.02411, 0.02316, 0.01409, 0.36326]

labels=['Java', 'C', 'C++', 'Python', 'vb .NET', 'C#', 'php', 'jsp', 'sql', 'assembly language', 'other']

pie = pygal.Gauge()

for i,per in enumerate(data):

pie.add(labels[i], per)

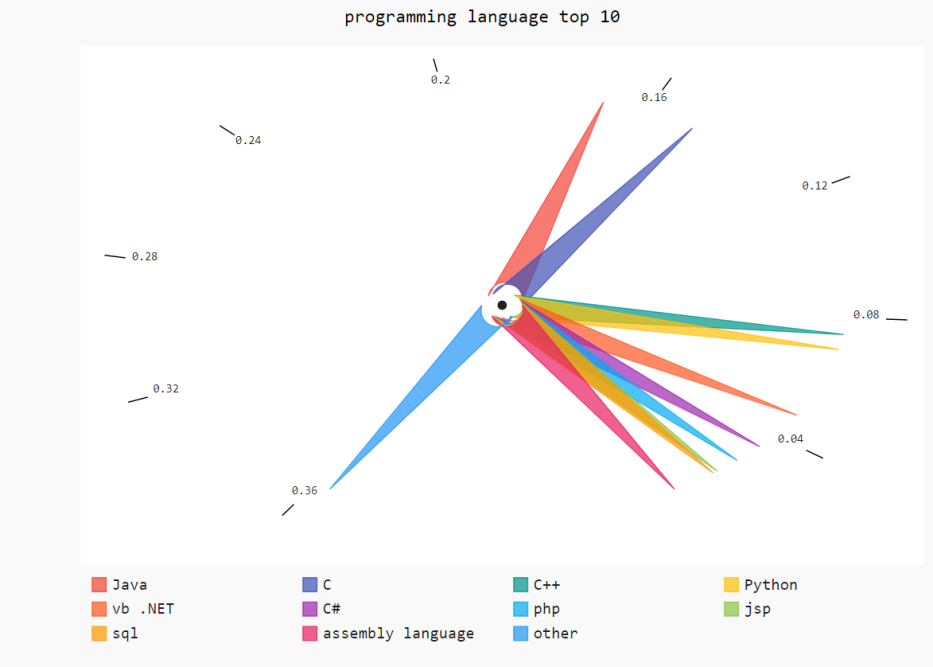
pie.title = 'programming language top 10'

#pie.inner\_radius=0.4

pie.legend\_at\_bottom = True

#pie.half\_pie=True

pie.render\_to\_file('Gauge.svg')



## 雷达图Radar

import pygal

data = [[5, 4.0, 5, 5, 5],

[4.8, 2.8, 4.8, 4.8, 4.9],

[4.5, 2.9, 4.6, 4.0, 4.9],

[4.0, 4.8, 4.9, 4.0, 5],

[3.0, 4.2, 2.3, 3.5, 2],

[4.8, 4.3, 3.9, 3.0, 4.5]]

labels=['Java', 'C', 'C++', 'Python','C#', 'PHP']

rader = pygal.Radar()

for i, per in enumerate(labels):

rader.add(labels[i], data[i])

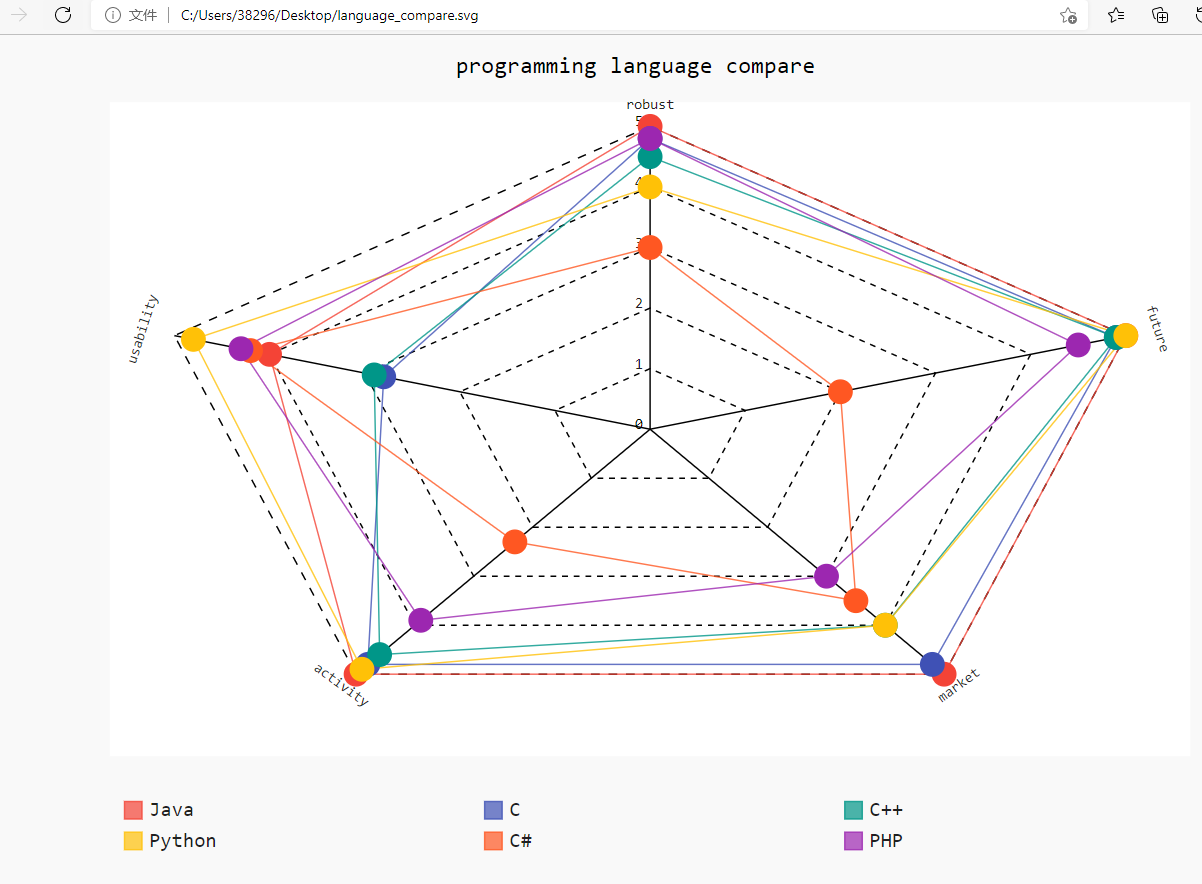
rader.x\_labels=['robust', 'usability', 'activity', 'market', 'future']

rader.title = 'programming language compare'

rader.dots\_size = 8

rader.legend\_at\_bottom = True

rader.render\_to\_file('language\_compare.svg')



# 处理数据

## csv读取器

reffer to: https://www.cnblogs.com/lnn123/p/10482124.html

#open默认utf-8打开文件，但文件是gbk编码，所以报错

#解决方案有两个：

#open时指定encoding为gbk

#文件另存为utf-8编码

import csv

filename='guangzhou-2017.csv'

#with open(filename) as f #默认utf-8打开文件，但文件是gbk编码，所以报错

with open(filename, encoding='gbk') as f:

reader = csv.reader(f)

row1 = next(reader)

row2 = next(reader)

print("row1: ", row1)

print("row2: ", row2)

root@pc:/home/zcw#

root@pc:/home/zcw# python3 test.py

row1: ['Date', 'Max TemperatureC', 'Min TemperatureC', 'Description', 'WindDir', 'WindForce']

row2: ['2017-1-1', '24', '13', '晴', '西南风', '1级']

root@pc:/home/zcw#

## csv和matplotlib

import csv

from datetime import datetime

from matplotlib import pyplot as plt

filename='guangzhou-2017.csv'

with open(filename, encoding='gbk') as f:

reader = csv.reader(f)

header\_row = next(reader)

print(header\_row)

start\_date = datetime(2017, 6,30)

end\_date = datetime(2017, 8,1)

dates,highs,lows = [],[],[]

for row in reader:

d = datetime.strptime(row[0], '%Y-%m-%d')

if start\_date < d < end\_date:

dates.append(d)

highs.append(int(row[1]))

lows.append(int(row[2]))

fig = plt.figure(dpi=128, figsize=(12,9))

plt.plot(dates, highs, c='red', label='最高气温', alpha=0.5, linewidth=2.0, linestyle='-', marker='v')

plt.plot(dates, lows, c='blue', label='最低气温', alpha=0.5, linewidth=3.0, linestyle='-.', marker='o')

plt.title('2017.7 temperature in guangzhou')

plt.xlabel('date')

plt.ylabel('temp')

fig.autofmt\_xdate() #该方法绘制斜着的日期标签

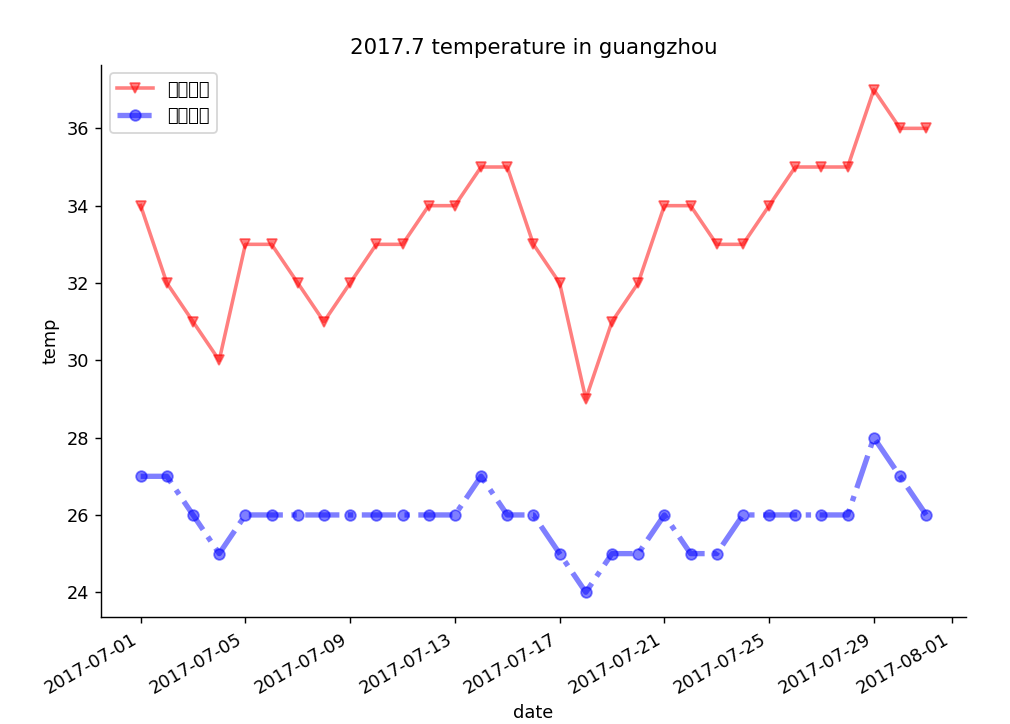
plt.legend() #显示图例

ax = plt.gca()

ax.spines['right'].set\_color('none')

ax.spines['top'].set\_color('none')

plt.show()



## csv和pygal

import csv

import pygal

filename='guangzhou-2017.csv'

with open(filename, encoding='gbk') as f:

reader = csv.reader(f)

header\_row = next(reader)

print(header\_row)

shades,sunnys, cloudys, rainys = 0,0,0,0

for row in reader:

if '阴' in row[3]:

shades += 1

elif '晴' in row[3]:

sunnys += 1

elif '多云' in row[3]:

cloudys += 1

elif '雨' in row[3]:

rainys += 1

else:

print(row[3])

pie = pygal.Pie()

pie.add('SHADE', shades)

pie.add('SUN', sunnys)

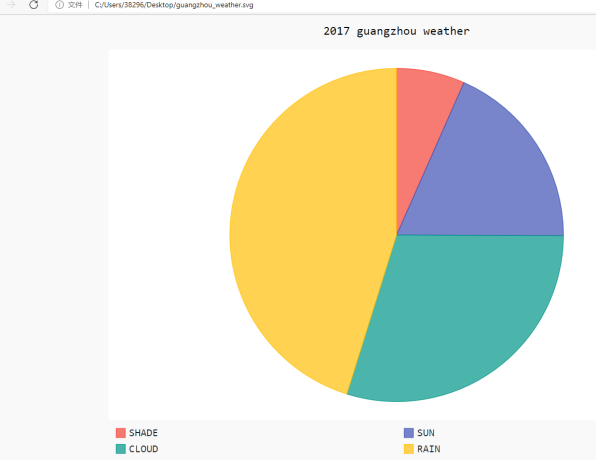
pie.add('CLOUD', cloudys)

pie.add('RAIN', rainys)

pie.title = '2017 guangzhou weather'

pie.legend\_at\_bottom = True

pie.render\_to\_file('guangzhou\_weather.svg')



## json

filename='gdp\_json.json'

with open(filename, encoding='gbk') as f:

li = json.load(f)

for e in li:

if e['Year'] == 2016 and e['Country Code'] == 'CHN':

print(e['Country Name'], e['Value'])

root@pc:/home/zcw# python3 test.py

China 11199145157649.2

## json和matplotlib

import json

from matplotlib import pyplot as plt

import numpy as np

filename='gdp\_json.json'

with open(filename, encoding='gbk') as f:

li = json.load(f)

country\_gdps = [{},{},{},{}, {}]

country\_codes = ['CHN', 'USA', 'JPN', 'RUS', 'CAN']

for e in li:

for i, country\_code in enumerate(country\_codes):

if e['Country Code'] == country\_code:

year = e['Year']

if 2000 < year < 2017:

country\_gdps[i][year] = e['Value']

gdp\_li = [[],[],[],[],[]]

x\_data = range(2001, 2017)

for i in range(len(gdp\_li)):

for year in x\_data:

#除以1e8，让数值变成以亿为单位

gdp\_li[i].append(country\_gdps[i][year]/1e8)

bar\_width=0.5

fig = plt.figure(dpi=128, figsize=(15,8))

colors=['indianred', 'steelblue', 'gold', 'lightpink', 'seagreen']

countries=['中国', '美国', '日本', '俄罗斯', '加拿大']

for i in range(len(colors)):

plt.bar(x=np.arange(len(x\_data))+bar\_width\*i, height=gdp\_li[i], label=countries[i], color=colors[i], alpha=0.8, width=bar\_width)

if i < 2: #仅为中国、美国的条柱上绘制GDP数值

for x,y in enumerate(gdp\_li[i]):

plt.text(x, y+100, '%.0f' % y, ha='center', va='bottom')

plt.xticks(np.arange(len(x\_data))+bar\_width\*2, x\_data)

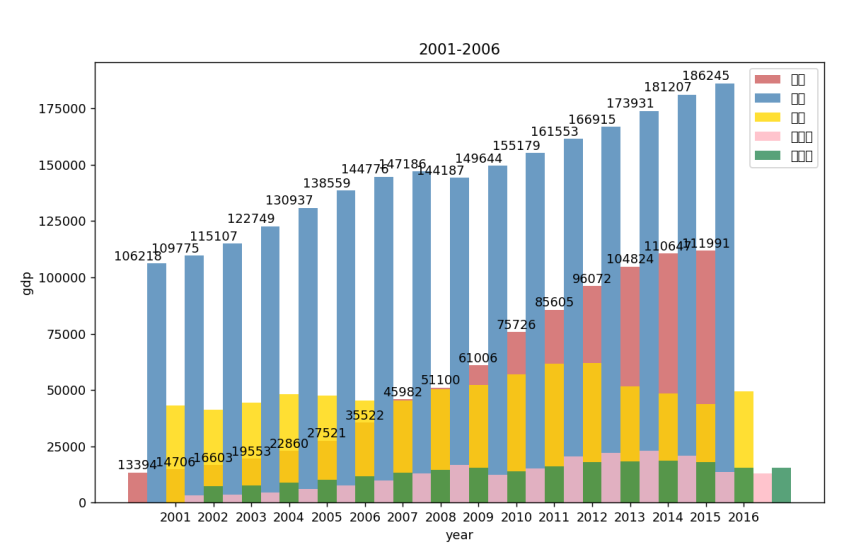
plt.title('2001-2006')

plt.xlabel('year')

plt.ylabel('gdp')

plt.legend()

plt.show()



## json和pygal

import json

import pygal

filename='gdp\_json.json'

filename1='population-figures-by-country.json'

with open(filename, encoding='gbk') as f:

li = json.load(f)

with open(filename1, encoding='gbk') as f:

population\_li = json.load(f)

avg\_gdps = [{},{},{},{}]

country\_codes = ['USA', 'JPN', 'RUS', 'CAN']

for e in li:

for i, country\_code in enumerate(country\_codes):

if e['Country Code'] == country\_code:

year = e['Year']

if 2000 < year < 2011:

for item in population\_li:

if item['Country\_Code'] == country\_code:

avg\_gdps[i][year] = round(e['Value'] / item['Population\_in\_%d' % year])

gdp\_li = [[],[],[],[]]

x\_data = range(2001, 2011)

for i in range(len(gdp\_li)):

for year in x\_data:

gdp\_li[i].append(avg\_gdps[i][year])

countries=['USA', 'JPN', 'RUS', 'CAN']

bar = pygal.Bar()

for i in range(len(countries)):

bar.add(countries[i], gdp\_li[i])

bar.width=1100

bar.title='2001-2010'

bar.x\_title='year'

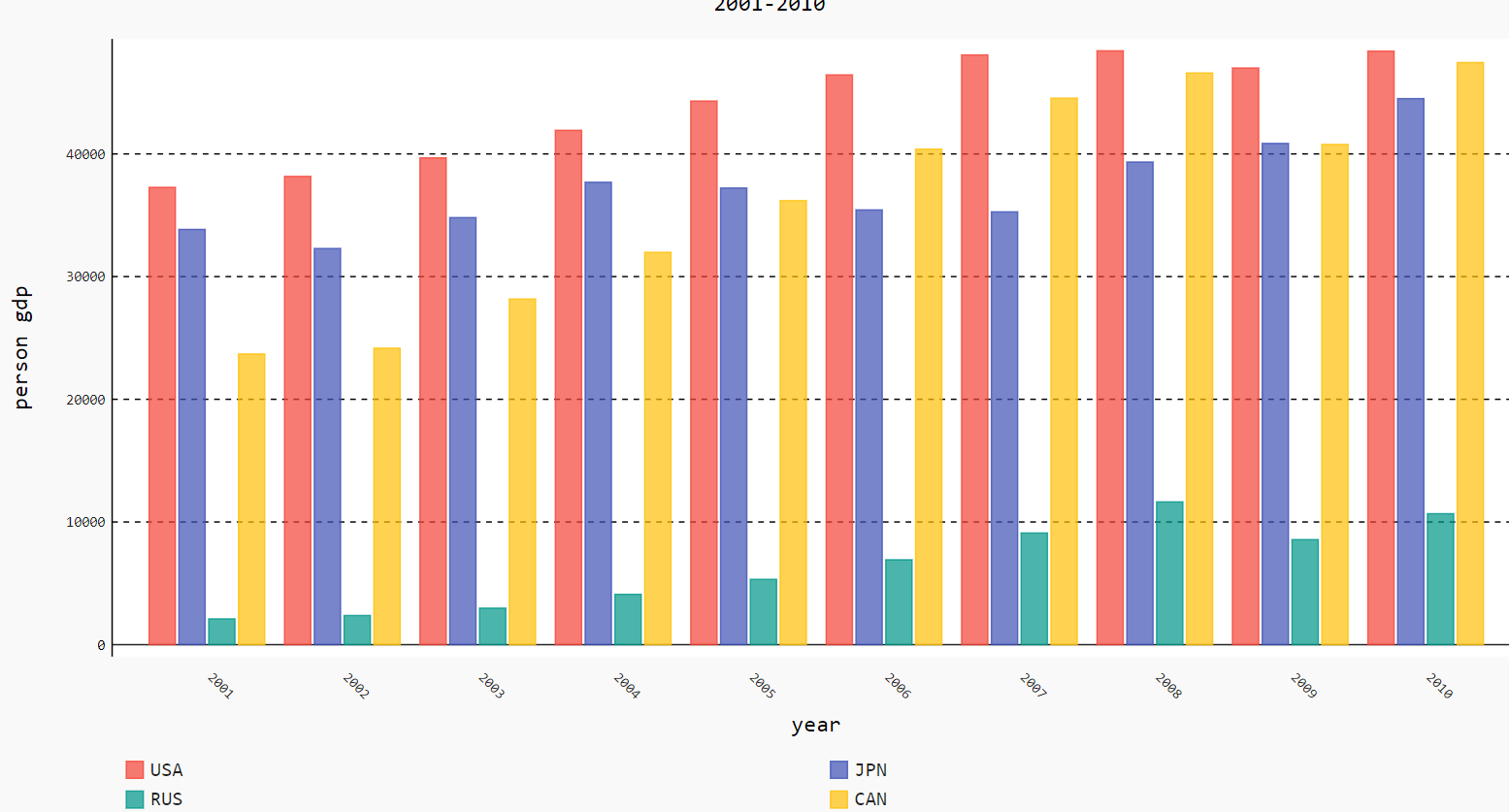
bar.y\_title='person gdp'

bar.x\_labels = x\_data

bar.x\_label\_rotation=45

bar.legend\_at\_bottom=True

bar.render\_to\_file('avg\_gdp.svg')



# 数据清洗

数据清洗的目的：识别出异常的数据，保证数据处理的鲁棒性

异常包括：

数据格式错误

数据丢失

import csv

import pygal

from datetime import datetime

from datetime import timedelta

filename='guangzhou-2017.csv'

with open(filename, encoding='gbk') as f:

reader = csv.reader(f)

header\_row = next(reader)

print(header\_row)

shades,sunnys,cloudys,rainys=0,0,0,0

prev\_day = datetime(2016, 12,31)

for row in reader:

try:

cur\_day = datetime.strptime(row[0], '%Y-%m-%d') #将第一列值格式化为日期

desc = row[3]

except ValueError:

print(cur\_day, 'data occurs error')

else:

diff = cur\_day - prev\_day

if diff != timedelta(days=1):

print('%s之前少了%d天的数据' % (cur\_day, diff.days-1))

prev\_day = cur\_day

if '阴' in desc:

shades += 1

elif '晴' in desc:

sunnys += 1

elif '云' in desc:

cloudys += 1

elif '雨' in desc:

rainys += 1

else:

print(desc)

pie = pygal.Pie()

pie.add('SHADE', shades)

pie.add('SUN', sunnys)

pie.add('CLOUD', cloudys)

pie.add('RAIN', rainys)

pie.title='2017 guangzhou weather'

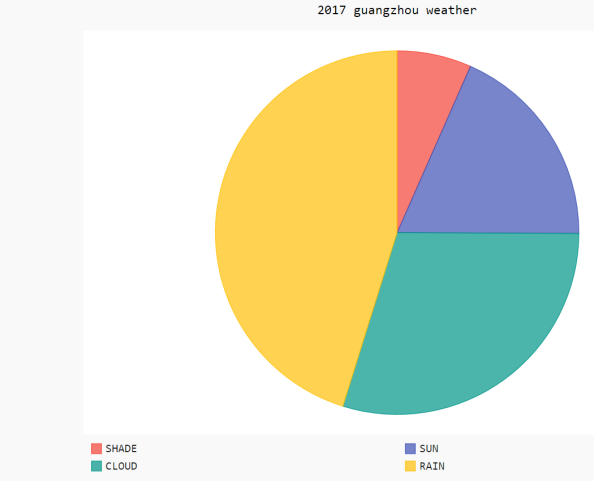
pie.legend\_at\_bottom = True

pie.render\_to\_file('111.svg')

root@pc:/home/zcw# python3 test.py

['Date', 'Max TemperatureC', 'Min TemperatureC', 'Description', 'WindDir', 'WindForce']

2017-03-06 00:00:00之前少了2天的数据



# 读取网络数据

wget http://lishi.tianqi.com/guangzhou/201701.html

FAQ1: 网页使用了utf8编码，不再是gbk编码

FAQ2:因为网页html元素发生了变化，所以教材默认的例子程序，正则表达式匹配后，都是空值，导致：

root@pc:/home/zcw# python3 test.py

http://lishi.tianqi.com/guangzhou/201701.html

============================

[]

=============================

Traceback (most recent call last):

File "/home/zcw/test.py", line 30, in <module>

uls = re.findall(pattern1, table[0]) #既然是空值，必然是越界

IndexError: list index out of range

root@pc:/home/zcw#

reffer to: https://jingyan.baidu.com/article/d2b1d1023b8b071c7f37d414.html

## 未修改的test.py

import re

from datetime import datetime

from datetime import timedelta

from matplotlib import pyplot as plt

from urllib.request import \*

def get\_html(city, year, month):

url = 'http://lishi.tianqi.com/' + city + '/' + str(year) + str(month) + '.html'

print(url)

request = Request(url)

request.add\_header('User-Agent', 'Mozilla/5.0 (Windows NT 10.0; WOW64)'+'AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.99 Safari/537.36')

response = urlopen(request)

return response.read().decode('utf8')

dates,highs,lows=[],[],[]

city='guangzhou'

year='2017'

months=['01', '02', '03', '04', '05', '06', '07', '08', '09', '10', '11', '12']

prev\_day = datetime(2016,12,31)

for month in months:

html = get\_html(city, year, month)

text = "".join(html.split())

pattern = re.compile('<divclass="tqtongji2">(.\*?)</div><divstyle="clear:both">')

table = re.findall(pattern, text)

print("============================")

print(table)

print("=============================")

pattern1 = re.compile('<ul>(.\*?)</ul>')

uls = re.findall(pattern1, table[0])

for ul in uls:

patten2 = re.compile('<li>(.\*?)</li>')

lis = re.findall(pattern2, ul)

d\_str = re.findall('>(.\*?)</a>', lis[0])[0]

try:

cur\_day = datetime.strptime(d\_str, '%Y-%m-%d')

high = int(lis[1])

low = int(lis[2])

except ValueError:

print(cur\_day, '数据出现错误')

else:

diff = cur\_day - prev\_day

if diff != timedelta(days=1):

print('%s之前少了%d天的数据' % (cur\_day, diff.days - 1))

dates.append(cur\_day)

highs.append(high)

lows.append(low)

prev\_day = cur\_day

#配置图形

fig = plt.figure(dpi=128, figsize=(12,9))

plt.plot(dates, highs, c='red', label='最高气温', alpha=0.5, linewidth = 2.0)

plt.plot(dates, highs, c='blue', label='最低气温', alpha=0.5, linewidth = 2.0)

#为两个数据的绘图区域填充颜色

plt.fill\_between(dates, highs, lows, facecolor='green', alpha=0.1)

plt.title('guangzhou weather')

plt.ylabel('temperature')

plt.xlabel('date')

#该方法绘制斜着的日期标签

plt.autofmt\_xdate()

plt.legend()

ax=plt.gca()

ax.spines['right'].set\_color('none')

ax.spines['top'].set\_color('none')

plt.show()

## 修改后的test.py

dates,highs,lows=[],[],[]

city='guangzhou'

year='2017'

months=['01', '02', '03', '04', '05', '06', '07', '08', '09', '10', '11', '12']

prev\_day = datetime(2016,12,31)

for month in months:

html = get\_html(city, year, month)

text = "".join(html.split()) #去掉所有的空格和换行

pattern = re.compile('<divclass="tian\_three">(.\*?)</div><script>')

table = re.findall(pattern, text)

pattern1 = re.compile('<ulclass="thrui">(.\*?)</ul>')

table1 = re.findall(pattern1, table[0])

pattern2=re.compile('>(.\*?)</li>')

lis=re.findall(pattern2, table1[0])

for li in lis:

divs=re.findall('>(.\*?)</div>', li)

d\_str=re.findall('(.\*)星期', divs[0])[0]

low\_str=re.findall('(.\*)℃', divs[1])[0]

high\_str=re.findall('(.\*)℃', divs[2])[0]

try:

cur\_day = datetime.strptime(d\_str, '%Y-%m-%d')

high = int(high\_str)

low = int(low\_str)

except ValueError:

print(cur\_day, '数据出现错误')

else:

diff = cur\_day - prev\_day

if diff != timedelta(days=1):

print('%s之前少了%d天的数据' % (cur\_day, diff.days - 1))

dates.append(cur\_day)

highs.append(high)

lows.append(low)

prev\_day = cur\_day

#配置图形

fig = plt.figure(dpi=128, figsize=(12,9))

plt.plot(dates, highs, c='red', label='最高气温', alpha=0.5, linewidth = 2.0)

plt.plot(dates, highs, c='blue', label='最低气温', alpha=0.5, linewidth = 2.0)

#为两个数据的绘图区域填充颜色

plt.fill\_between(dates, highs, lows, facecolor='green', alpha=0.1)

plt.title('guangzhou weather')

plt.ylabel('temperature')

plt.xlabel('date')

#该方法绘制斜着的日期标签

plt.autofmt\_xdate()

plt.legend()

ax=plt.gca()

ax.spines['right'].set\_color('none')

ax.spines['top'].set\_color('none')

plt.show()