pyecharts生成各种图表

一、pyecharts简介

Echarts是一个由百度开源的数据可视化，能够进行良好的交互性，精巧的图表设计，得到了众多开发者的认可。而Python是一门富有表达力的语言，很适合用于数据处理。分析遇上数据可视化时，pyecharts诞生了。

pyecharts 是一个用于生成 Echarts 图表的类库，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。**特点**：

- 简洁的API设计，使用如丝滑般流畅，支持链式调用
- 囊括了30+种常见图表，应有尽有
- 支持主流Notebook环境，Jupyter Notebook和JupyterLab
- 可轻松集成至Flask，Sanic，Django等主流Web框架
- 高度灵活的配置项，可轻松搭配出精美的图表
- 详细的文档和示例，帮助开发者重启的上手项目
- 超过400个以上的地图文件，并支持原生百度地图，为地理数据可视化提供灵活的支持

pyecharts的版本，v1.0版本支持python3.6及以上版本。

1.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）
Kline（K线图）
```

```
Line（折线/面积图）
Line3D（3D 折线图）
Liquid（水球图）
Map（地图）
Parallel（平行坐标系）
Pie（饼图）
Polar（极坐标系）
Radar（雷达图）
Sankey（桑基图）
Scatter（散点图）
Scatter3D（3D 散点图）
ThemeRiver（主题河流图）
WordCloud（词云图）
```

2.通用配置

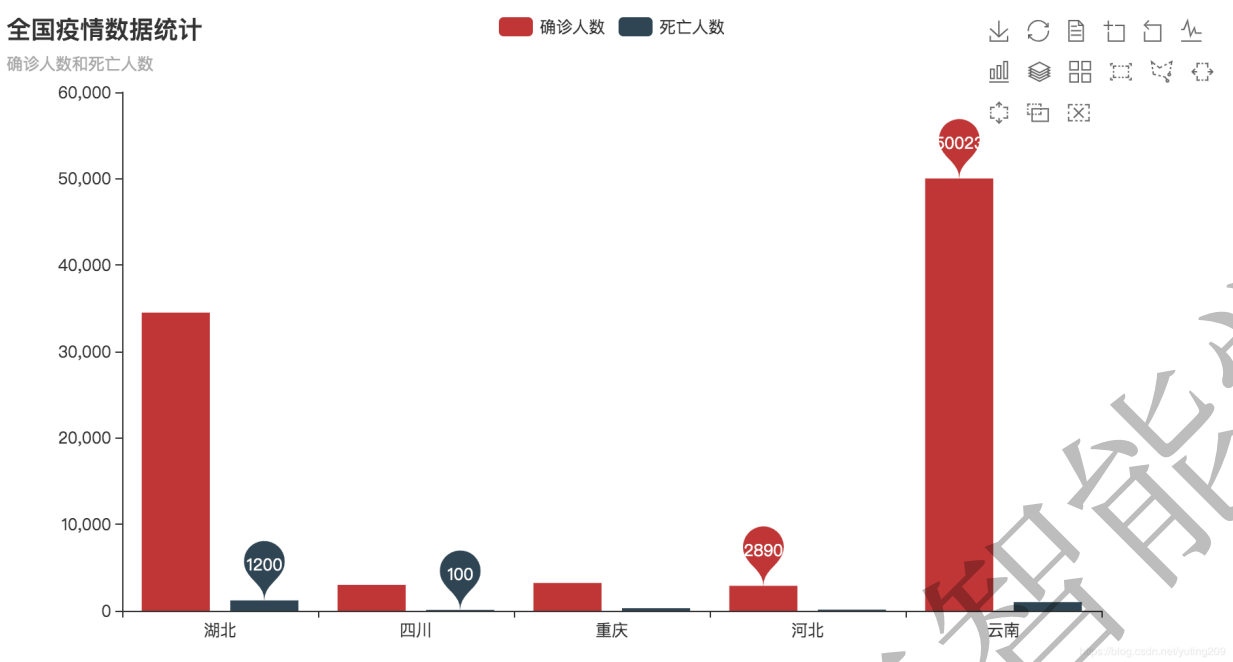
```
xAxis，yAxis：平面直角坐标系中的 x、y 轴。(Line、Bar、Scatter、EffectScatter、Kline)
legend：图例组件。图例组件展现了不同系列的标记(symbol)，颜色和名字。可以通过点击图例控制哪些系列不显示。
label：图形上的文本标签，可用于说明图形的一些数据信息，比如值，名称等。
lineStyle：带线图形的线的风格选项(Line、Polar、Radar、Graph、Parallel)
markLine&markPoint：图形标记组件，用于标记指定的特殊数据，有标记线和标记点两种。（Bar、Line、Kline）
tooltip：提示框组件，用于移动或点击鼠标时弹出数据内容
toolbox：右侧实用工具箱
```

二、pyecharts的使用

1. 柱状图

```
from pyecharts.charts import Bar
from pyecharts import options

# 1.准备数据
cate = ['湖北', '四川', '重庆', '河北', '云南']
data1 = [34500, 3000, 3218, 2890, 50023]
data2 = [1200, 100, 300, 130, 1004]
```

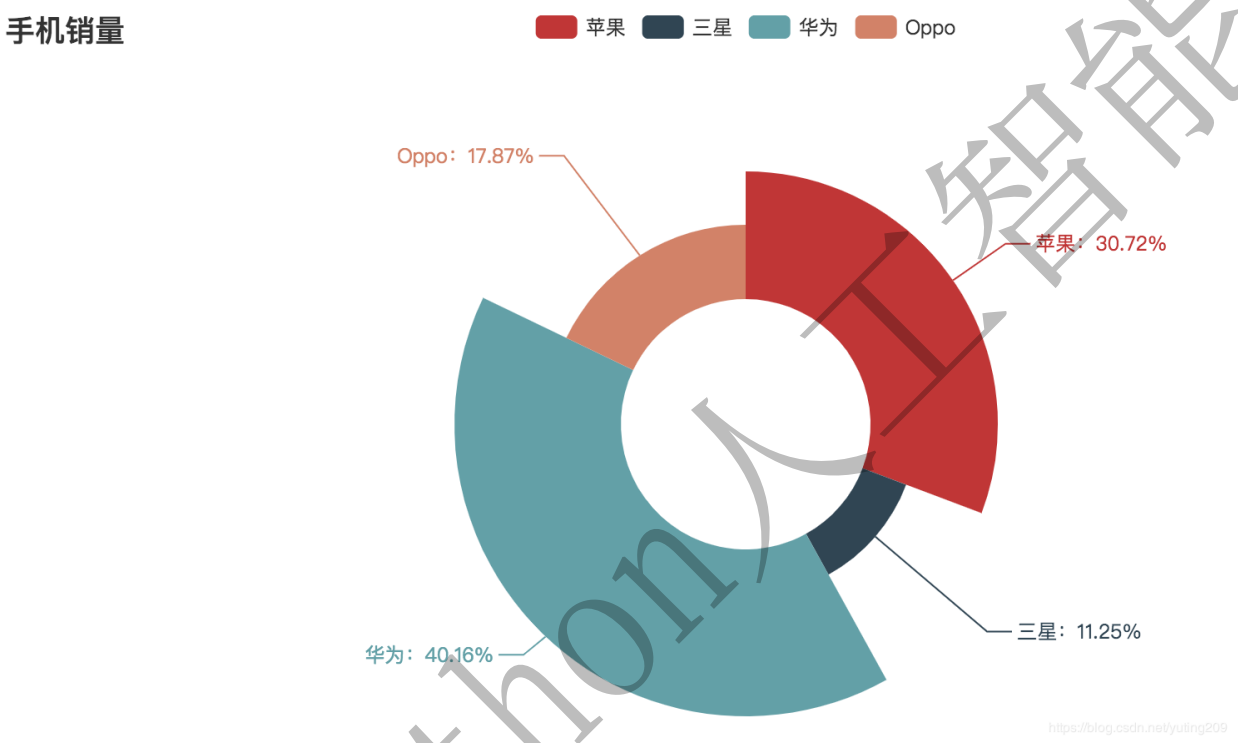
```
# 2. 创建图表对象
bar = Bar()

# 3. 关联数据
bar.add_xaxis(cate)    # 确定x轴上要显示的内容
bar.add_yaxis('确诊人数', data1)
bar.add_yaxis('死亡人数', data2)

# 4. 设置图表
# 全局设置
bar.set_global_opts(
        # 设置标题信息
        title_opts=options.TitleOpts(title='全国疫情数据统计',
subtitle='确诊人数和死亡人数'),
        # 显示工具箱
        toolbox_opts=options.ToolboxOpts())

# 系列设置
bar.set_series_opts(
        # 设置是否显示数值
        label_opts=options.LabelOpts(is_show=False),
        # 添加标记点
        markpoint_opts=options.MarkPointOpts(data=[
            options.MarkPointItem(type_='min', name='最小值'),
            options.MarkPointItem(type_='max', name='最大值')
        ]))


# 5. 数据渲染 - 生成图表
bar.render('files/柱状图.html')
```



2. 饼状图

```
from pyecharts.charts import Pie
from pyecharts import options

# 1. 准备数据
data = [('苹果', 153), ('三星', 56), ('华为', 200), ('Oppo', 89)]

# 2. 创建图标对象
pie = Pie()

# 3. 关联数据
pie.add(
        # 设置系列名称
        series_name='手机销量',
        # 设置需要展示的数据
        data_pair=data,
        # 设置圆环空心部分和数据显示部分的比例
        radius=['30%', '70%'],
        # 设置饼是不规则的
        rosetype='radius'
)

# 设置数据显示的格式
pie.set_series_opts(label_opts=options.LabelOpts(formatter='{b}:
{d}%'))
```

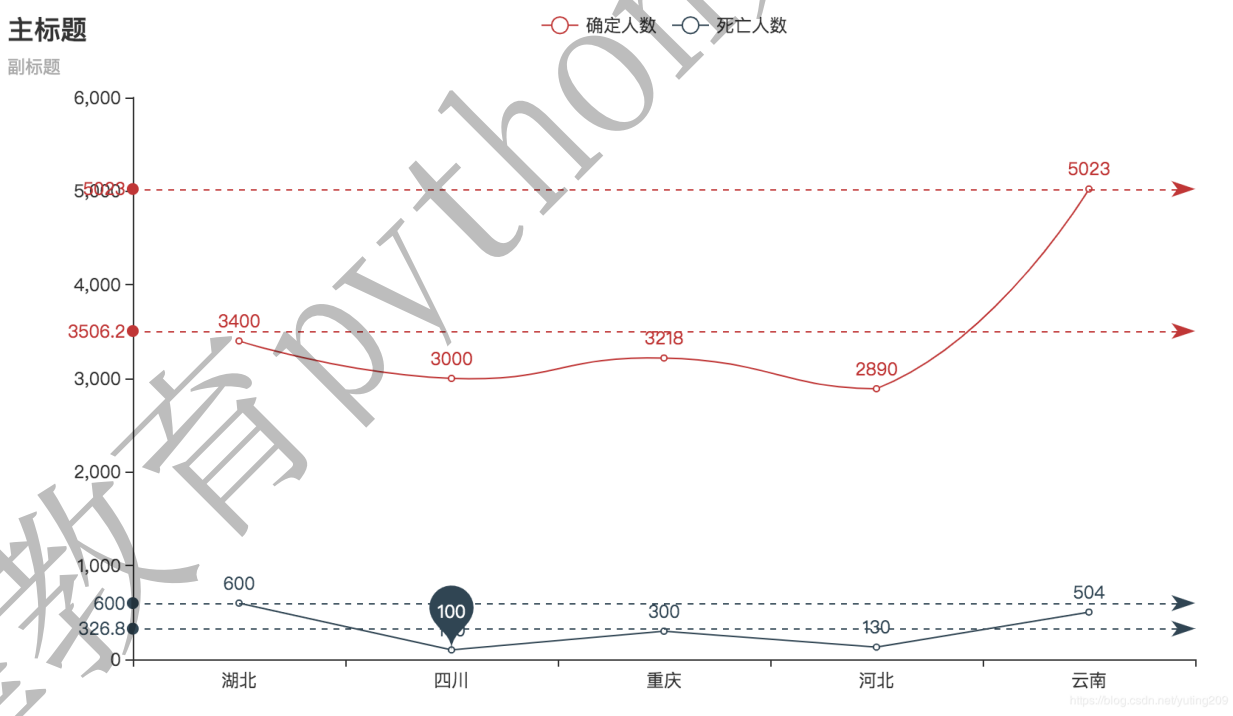
```
# pie.set_series_opts(label_opts=options.LabelOpts(formatter='{b}的占比:百分之{d}'))

# 设置图表的标题
pie.set_global_opts(title_opts=options.TitleOpts(title='手机销量'))

# 5. 渲染数据
pie.render('files/饼状图.html')
```



3. 折线图

```
from pyecharts.charts import Line
from pyecharts import options

# 1.准备数据
cate = ['湖北', '四川', '重庆', '河北', '云南']
data1 = [3400, 3000, 3218, 2890, 5023]
data2 = [600, 100, 300, 130, 504]

# 2.创建图表
line = Line()

# 3. 关联数据
line.add_xaxis(cate)
line.add_yaxis('确诊人数', data1,
```

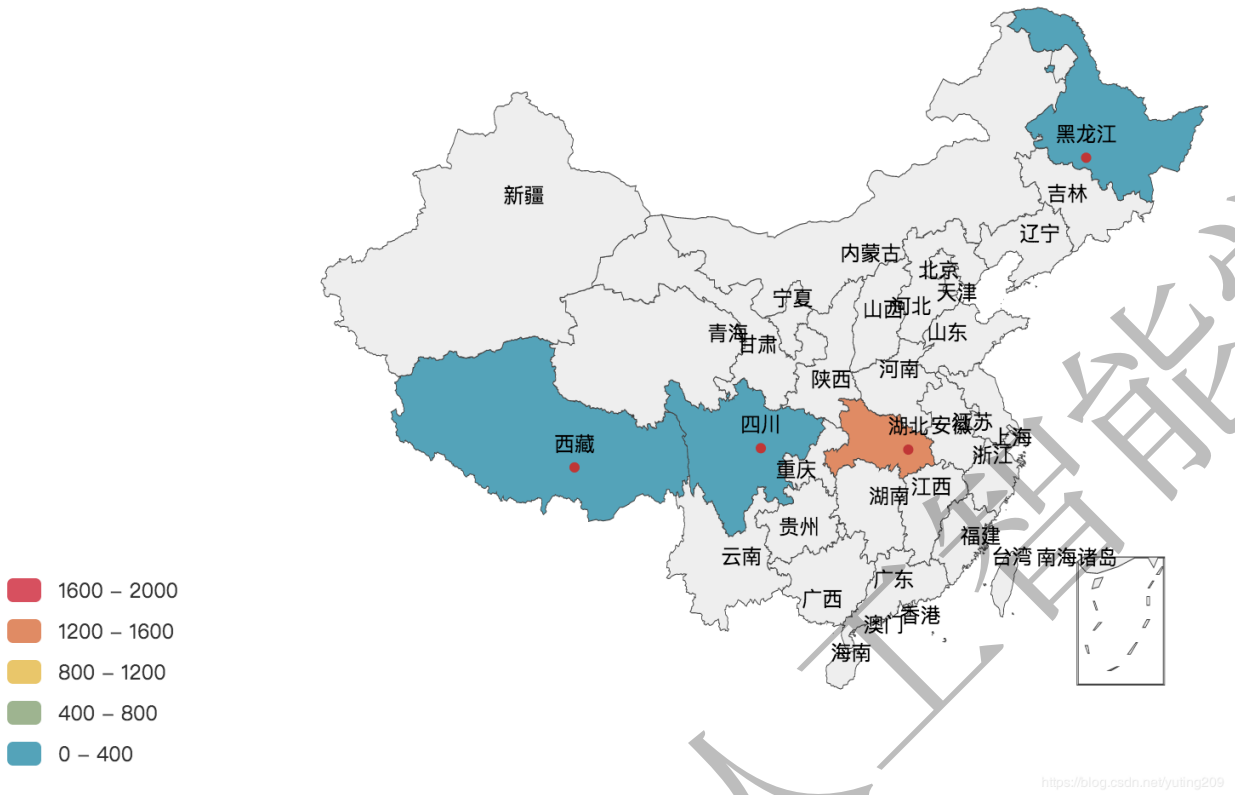
```
                # 设置折线是否平滑
                is_smooth=True)

line.add_yaxis('死亡人数', data2,
markpoint_opts=options.MarkPointOpts(
        data=[options.MarkPointItem(type_='min', name='最小值')]
))

line.set_series_opts(markline_opts=options.MarkLineOpts(
        # 设置平均值的标记线
        data=[options.MarkPointItem(type_='average', name='平均值'),
              # 设置最大值的标记线
              options.MarkPointItem(type_='max', name='最大值')]
))

line.set_global_opts(title_opts=options.TitleOpts(title='主标题',
subtitle='副标题'))

# 5. 渲染数据
line.render('templates/折线图.html')
```



4. 地图

```
from pyecharts.charts import Map
from pyecharts import options
```

```
# 1. 准备数据
data = [('湖北', 1500), ('四川', 340), ('西藏', 34), ('黑龙江', 123)]


# 2.创建地图对象
map1 = Map()

# 3. 关联数据
map1.add('疫情数据', data, 'china')


# 4. 设置
map1.set_global_opts(
        # 设置颜色块标记范围
        visualmap_opts=options.VisualMapOpts(max_=2000,
is_piecewise=True),
        # 隐藏顶部的数据导航显示
        legend_opts=options.LegendOpts(is_show=False),
        title_opts=options.TitleOpts(title='大标题', subtitle='副标
题')
)

# 5. 渲染数据
map1.render('templates/地图.html')
```



5. 组合图

```
from pyecharts.charts import Line, Bar, Grid
from pyecharts import options

# 1. 先创建需要组合在一起的单独的图表
cate = ['重庆', '黑龙江', '香港', '台湾', '上海']
data1 = [579, 925, 1035, 427, 640]
data2 = [570, 487, 699, 253, 533]

bar = Bar()
line = Line()

bar.add_xaxis(cate)
bar.add_yaxis('确诊人数', data1)
bar.add_yaxis('治愈人数', data2)
bar.set_global_opts(title_opts=options.TitleOpts(title='疫情信息'))

line.add_xaxis(cate)
line.add_yaxis('确诊人数', data1)
```

```
line.add_yaxis('治愈人数', data2)

# 2. 创建Grid对象（组合图表对象）
grid = Grid(init_opts=options.InitOpts(width='1220px',
height='800px'))

# 3.添加需要组合的所有的图表对象
grid.add(bar, grid_opts=options.GridOpts(
        pos_top='50',
        pos_left='60',
        width='550',
        height='300'
))

grid.add(line, grid_opts=options.GridOpts(
        width='550',
        height='300',
        pos_top='50',
        pos_right='0'
))


# 4.渲染
grid.render('templates/组合.html')
```