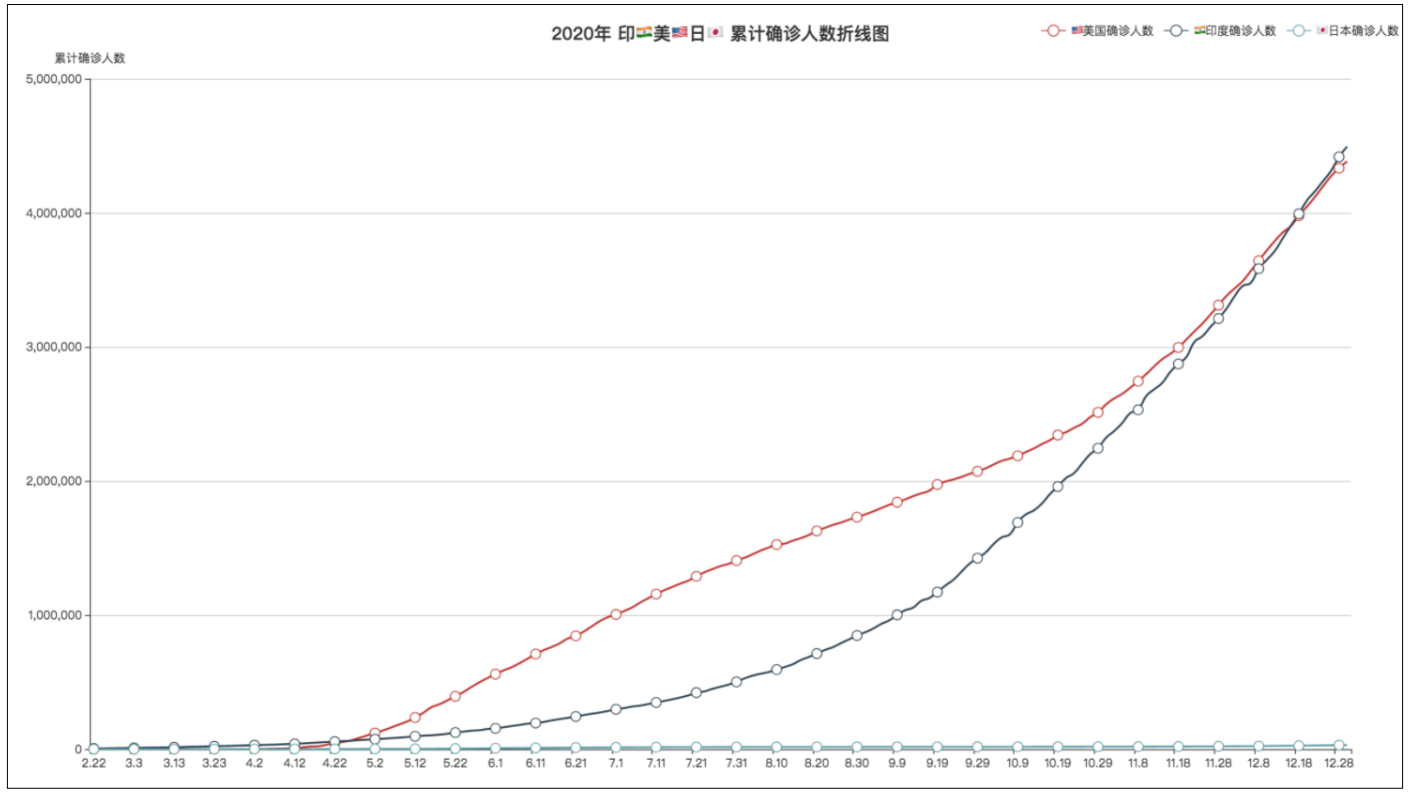
Python 基础综合案例

数据可视化 - 折线图可视化

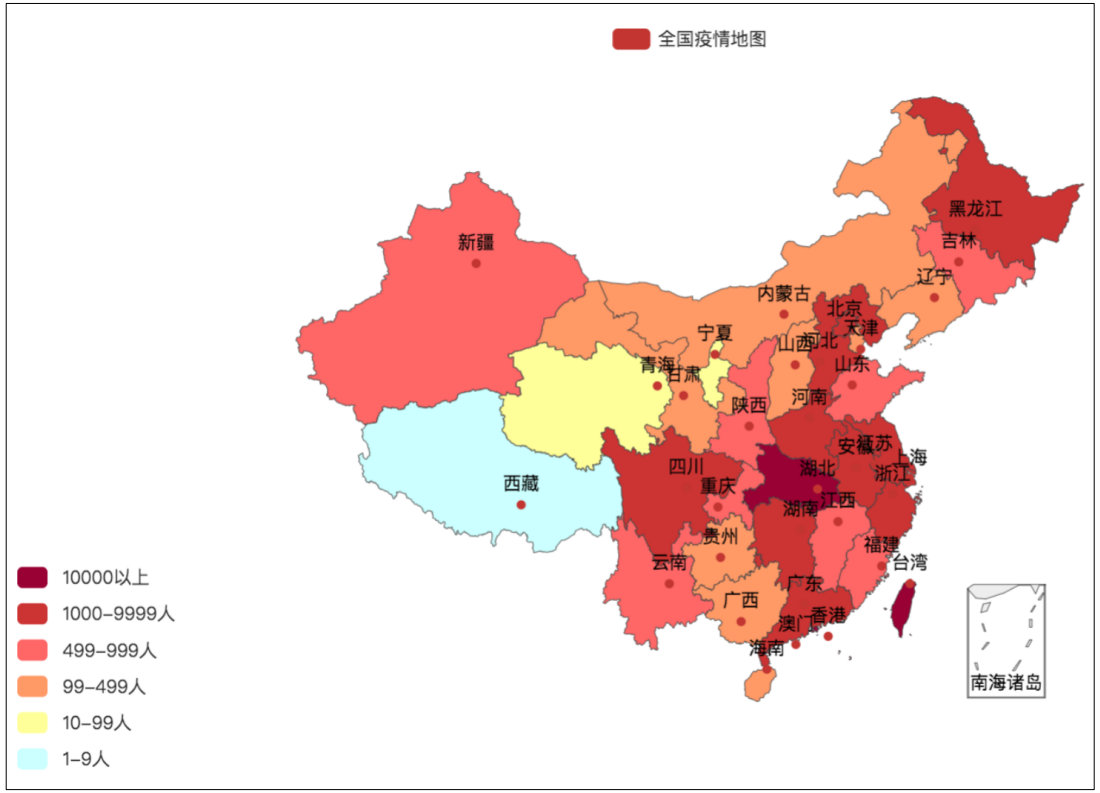
黑马程序员 www.itheima.com
传智教育旗下 高端IT教育品牌

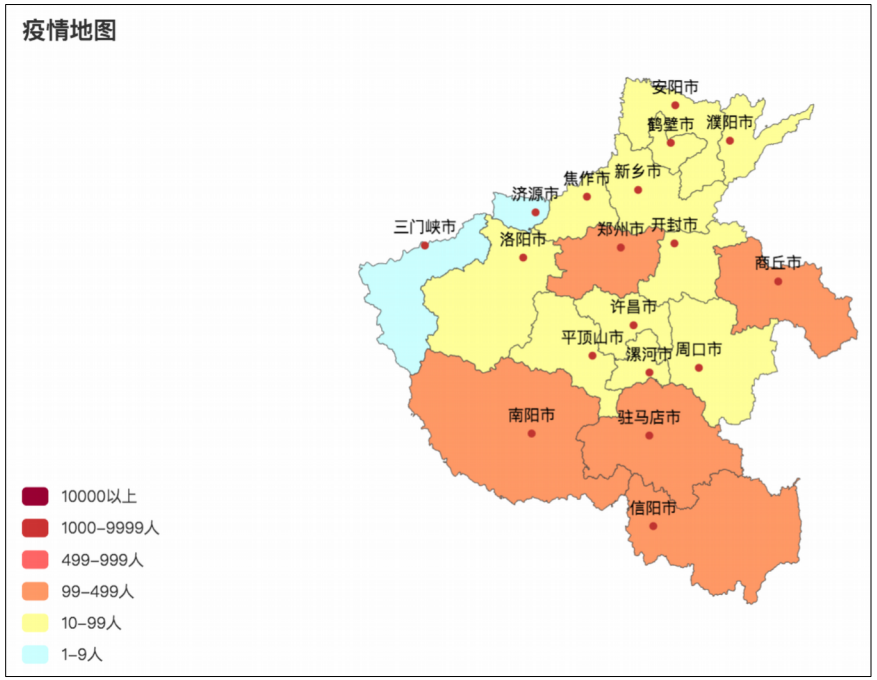
效果一： 2020 年印美日新冠累计确诊人数

2020 年是新冠疫情爆发的一年， 随着疫情的爆发， 国内外确诊人数成了大家关心的热点，
相信大家都有看过类似的疫情报告． 本案例对印度美国日本三个国家确诊人数的进行了
可视化处理， 形成了可视化的疫情确诊人数报告．

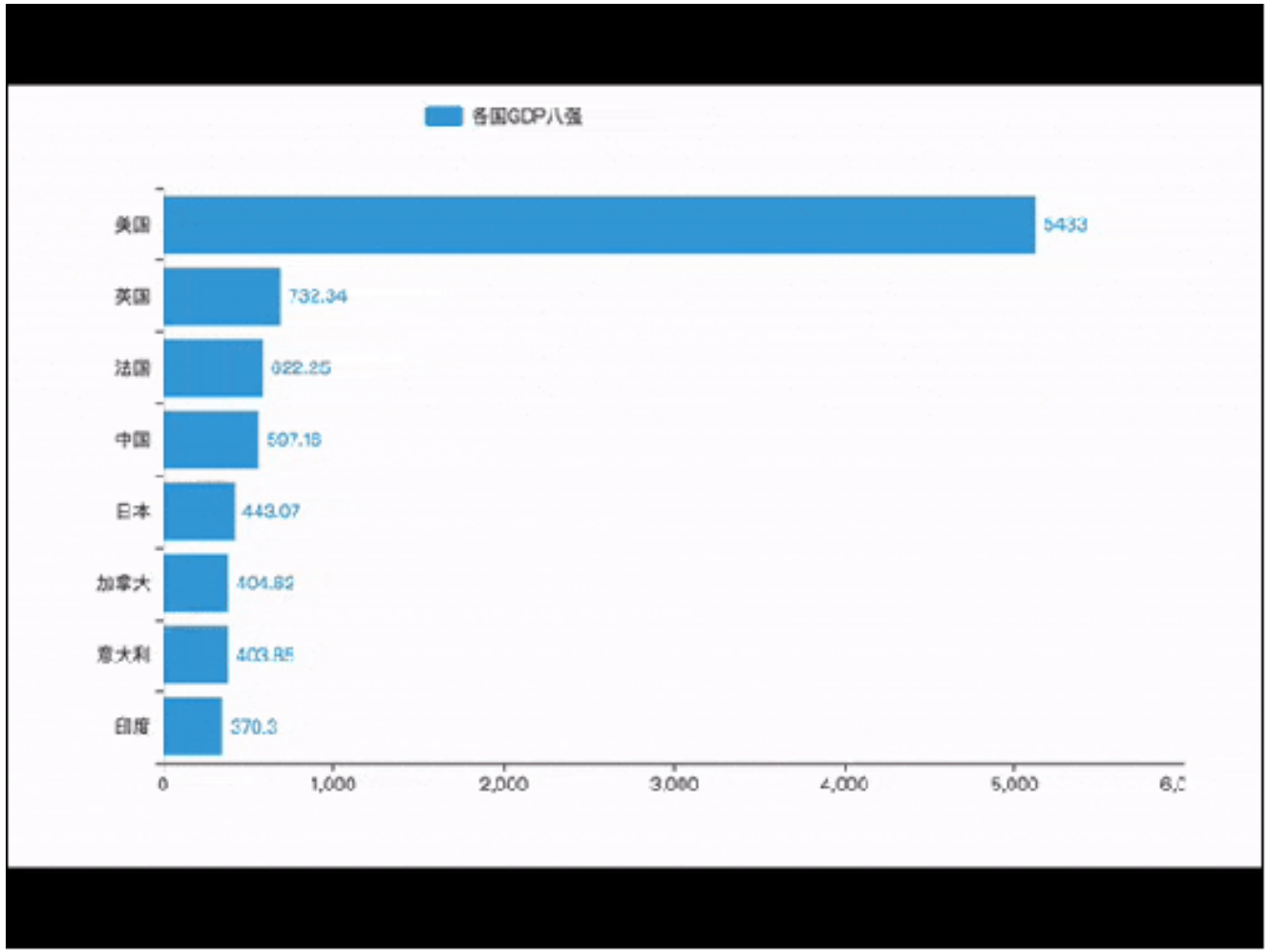


效果二：全国疫情地图可视化

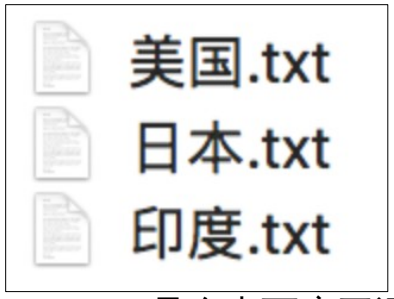




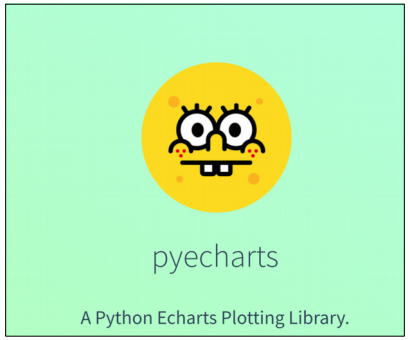
效果三：动态 GDP 增长图



数据来源

- **本案例数据全部来自 << 百度疫情实时大数据报告 >> ，及公开的全球各国 GDP 数据**
- 

Echarts 是个由百度开源的数据可视化，凭借着良好的交互性，精巧的图表设计，得到了众多开发者的认可． 而 Python 是门富有表达力的语言，很适合用于数据处理． 当数据分析遇上数据可视化时 pyecharts 诞生了．



学习目标

Learning Objectives

可视化案例的学习目标：

- 通过案例，回忆巩固 Python 基础的语法
- 锻炼编程能力，熟练语法的使用

目录

Contents

学习目标

Learning Objectives

1. 知道什么是 json
2. 掌握如何使用 json 进行数据转化

什么是 json

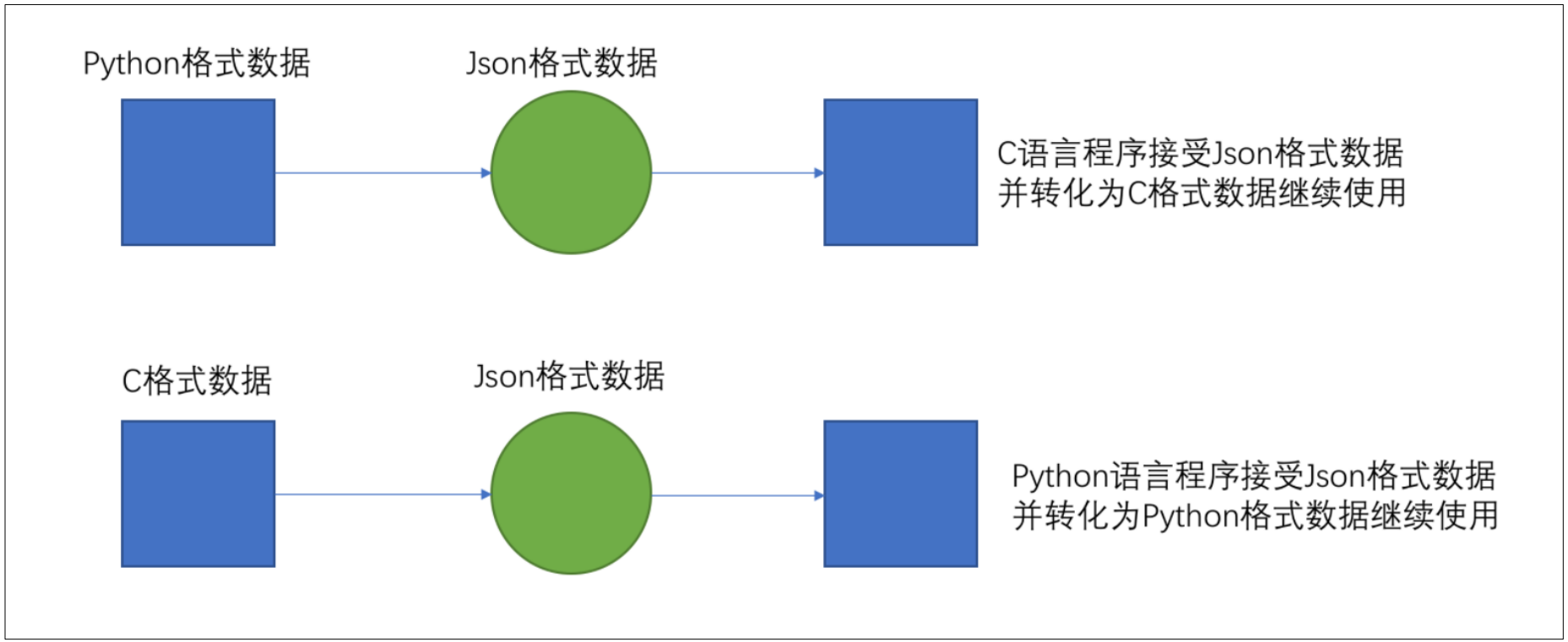
- JSON 是一种轻量级的数据交互格式。可以按照 JSON 指定的格式去组织和封装数据
- JSON 本质上是一个带有特定格式的字符串

- **主要功能**： json 就是一种在各个编程语言中流通的数据格式，负责不同编程语言中的数据传递和交互．类似于：
  - 国际通用语言 - **英语**
  - 中国 56 个民族不同地区的通用语言 - **普通话**

json 有什么用

- 各种编程语言存储数据的容器不尽相同，在 Python 中有字典 dict 这样的数据类型， 而其它语言可能没有对应的字典。

为了让不同的语言都能够相互通用的互相传递数据， JSON 就是一种非常良好的中转数据格式。如下图，以 Python 和 C 语言互传数据为例：



json 格式数据转化

- json 格式的数据要求很严格， 下面我们看一下他的要求

```
# json 数据的格式可以是：
{"name":"admin","age":18}


#  也可以是：
[{"name":"admin","age":18},{"name":"root","age":16},{"name":" 张三 ","age":20}]
```

Python 数据和 Json 数据的相互转化

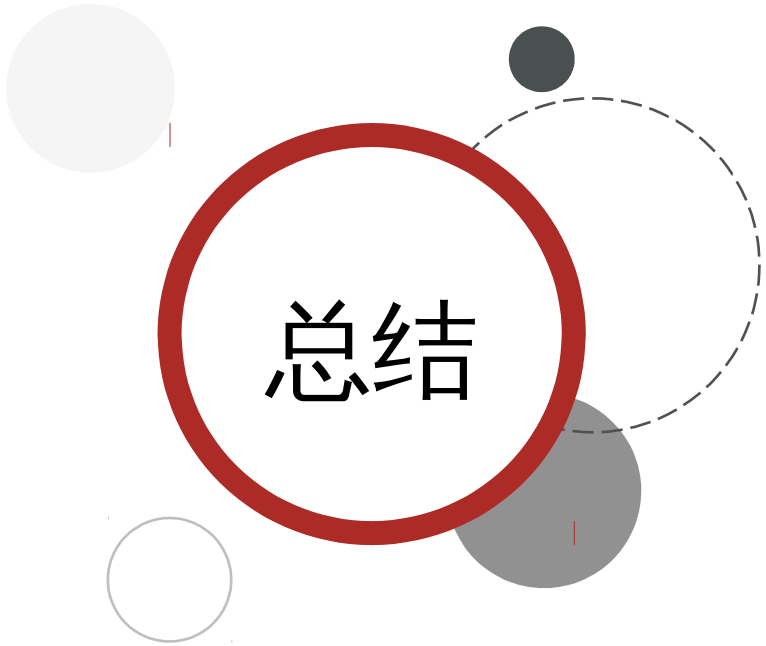
- Python 数据和 Json 数据的相互转化

```
# 导入 json 模块
import json

# 准备符合格式 json 格式要求的 python 数据
data = [{"name": " 老王 ", "age": 16}, {"name": " 张三 ", "age": 20}]

# 通过 json.dumps(data) 方法把 python 数据转化为了 json 数据
data = json.dumps(data)

# 通过 json.loads(data) 方法把 json 数据转化为了 python 数据
data = json.loads(data)
```



1. json ：是一种轻量级的数据交互格式，采用完全独立于编程语言的文本格式来存储和表示数据（就是字符串）

Python 语言使用 JSON 有很大优势，因为： JSON 无非就是一个单独的字典或一个内部元素都是字典的列表

所以 JSON 可以直接和 Python 的字典或列表进行无缝转换。

2. json 格式数据转化

通过 json.dumps(data) 方法把 python 数据转化为了 json 数据

```
data = json.dumps(data)
```

如果有中文可以带上： ensure_ascii=False 参数来确保中文正常转换

通过 json.loads(data) 方法把 josn 数据转化为了 python 列表或字典

```
data = json.loads(data)
```

目录

Contents

pyecharts 模块

- 如果想要做出数据可视化效果图， 可以借助 pyecharts 模块来完成

- **概况 ：**

  Echarts 是个由百度开源的数据可视化，凭借着良好的交互性，精巧的图表设计，得到了众多开发者的认可． 而 Python 是门富有表达力的语言，很适合用于数据处理． 当数据分析遇上数据可视化时 pyecharts 诞生了．

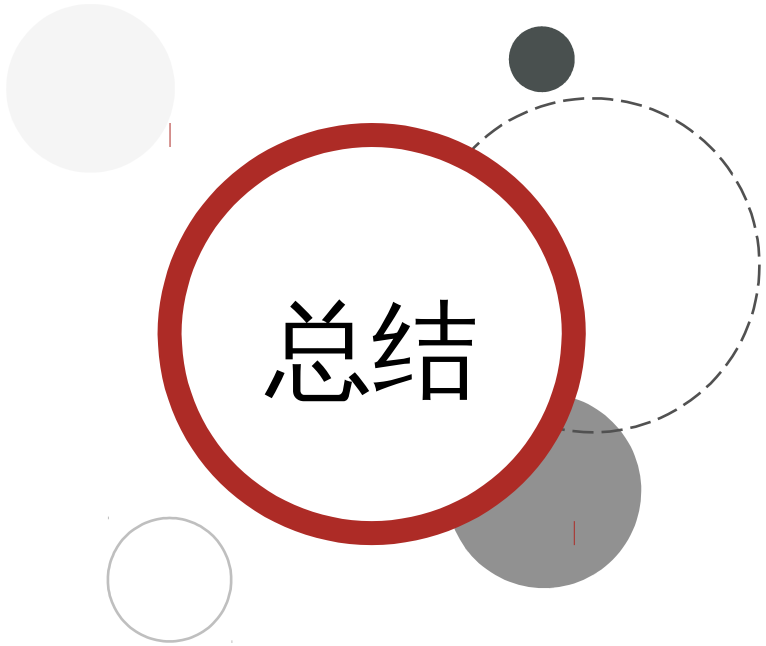
pyecharts 模块安装

- 使用在前面学过的 pip 命令即可快速安装 PyEcharts 模块

- pip install pyecharts

```
C:\Users\javac>pip install pyecharts -i https://pypi.tuna.tsinghua.edu.cn/simple
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting pyecharts
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/f1/f4/66f4340de85545340f54c230352419d21dfa55f01fa00aec137b283ce95c/pyecharts-1.9.1-py3-none-any.whl (135 kB)
     ---------------------------------------- 135.6/135.6 KB 1.3 MB/s eta 0:00:00
Collecting jinja2
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/bc/c3/f068337a370801f372f2f8f6bad74a5c140f6fda3d9de154052708dd3c65/Jinja2-3.1.2-py3-none-any.whl (133 kB)
     ---------------------------------------- 133.1/133.1 KB 4.0 MB/s eta 0:00:00
Collecting prettytable
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/5f/ab/64371af206988d7b15c8112c9c277b8eb4618397c01471e52b902a17f59c/prettytable-3.3.0-py3-none-any.whl (26 kB)
Collecting simplejson
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/59/94/2fba254afb6808db7fb95ba4727bb1f329021660ba9a2f2f51f59207e263/simplejson-3.17.6-cp310-cp310-win_amd64.whl (75 kB)
     ---------------------------------------- 75.9/75.9 KB 4.1 MB/s eta 0:00:00
Collecting MarkupSafe>=2.0
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/3d/4b/15e5b9d40c4b58e97ebcb8ed5845a215fa5b7cf49a7f1cc7908f8db9cf46/MarkupSafe-2.1.1-cp310-cp310-win_amd64.whl (17 kB)
Collecting wcwidth
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/59/7c/e39aca596badaf1b78e8f547c807b04dae603a433d3e7a7e04d67f2ef3e5/wcwidth-0.2.5-py2.py3-none-any.whl (30 kB)
Installing collected packages: wcwidth, simplejson, prettytable, MarkupSafe, jinja2, pyecharts
Successfully installed MarkupSafe-2.1.1 jinja2-3.1.2 prettytable-3.3.0 pyecharts-1.9.1 simplejson-3.17.6 wcwidth-0.2.5
WARNING: You are using pip version 22.0.4; however, version 22.2 is available.
You should consider upgrading via the 'D:\dev\python\python3.10.4\python.exe -m pip install --upgrade pip' command.

C:\Users\javac>
```



1. 开发可视化图表使用的技术栈是：

Echarts 框架的 Python 版本： PyEcharts 包

2. 如何安装 PyEcharts 包：

pip install pyecharts

3. 如何查看官方示例

打开官方画廊：

https://gallery.pyecharts.org/#/README

目录

Contents

学习目标

Learning Objectives

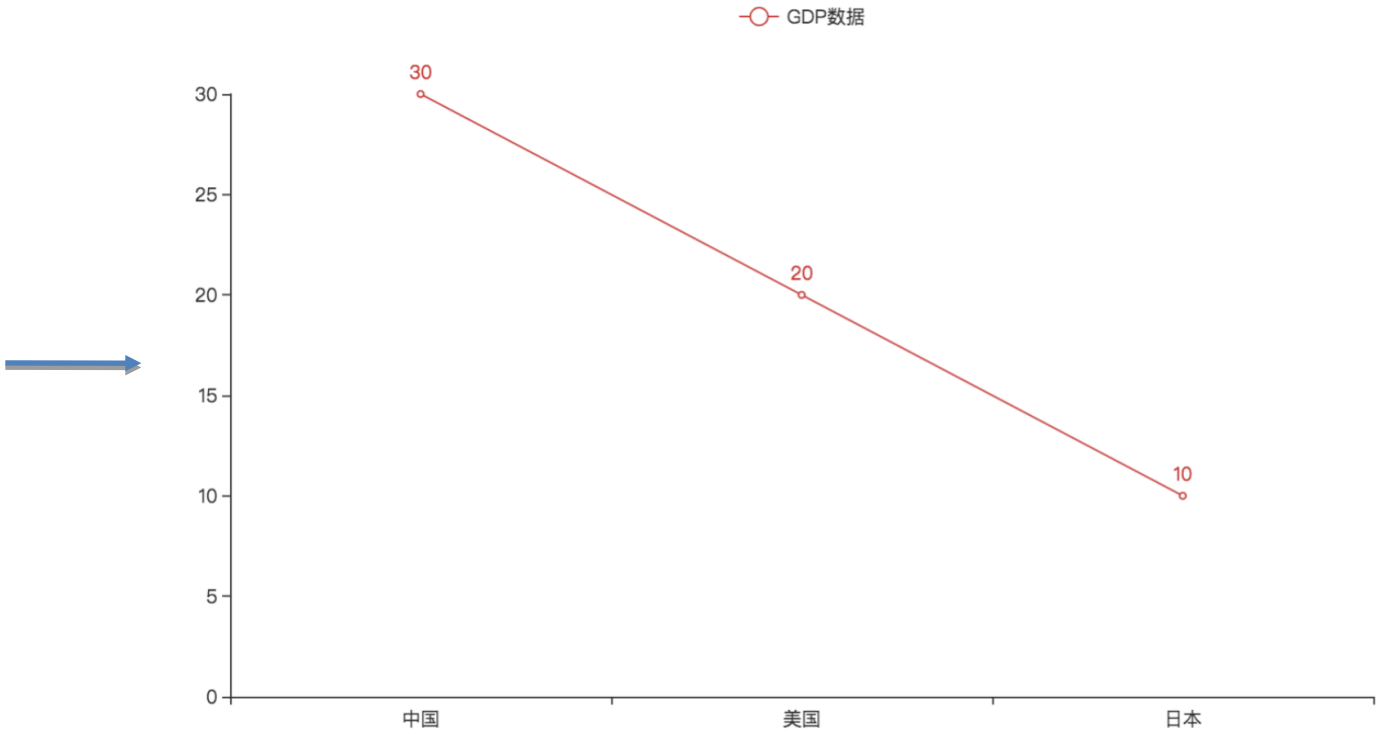
1. 构建一个基础的折线图
2. 使用全局配置项设置属性

pyechars 入门

- **基础折线图**

```python
# 导包，导入Line功能构建折线图对象
from pyecharts.charts import Line

# 得到折线图对象
line = Line()
# 添加x轴数据
line.add_xaxis(["中国", "美国", "英国"])
# 添加y轴数据
line.add_yaxis("GDP", [30, 20 ,10])
# 生成图表
line.render()
```

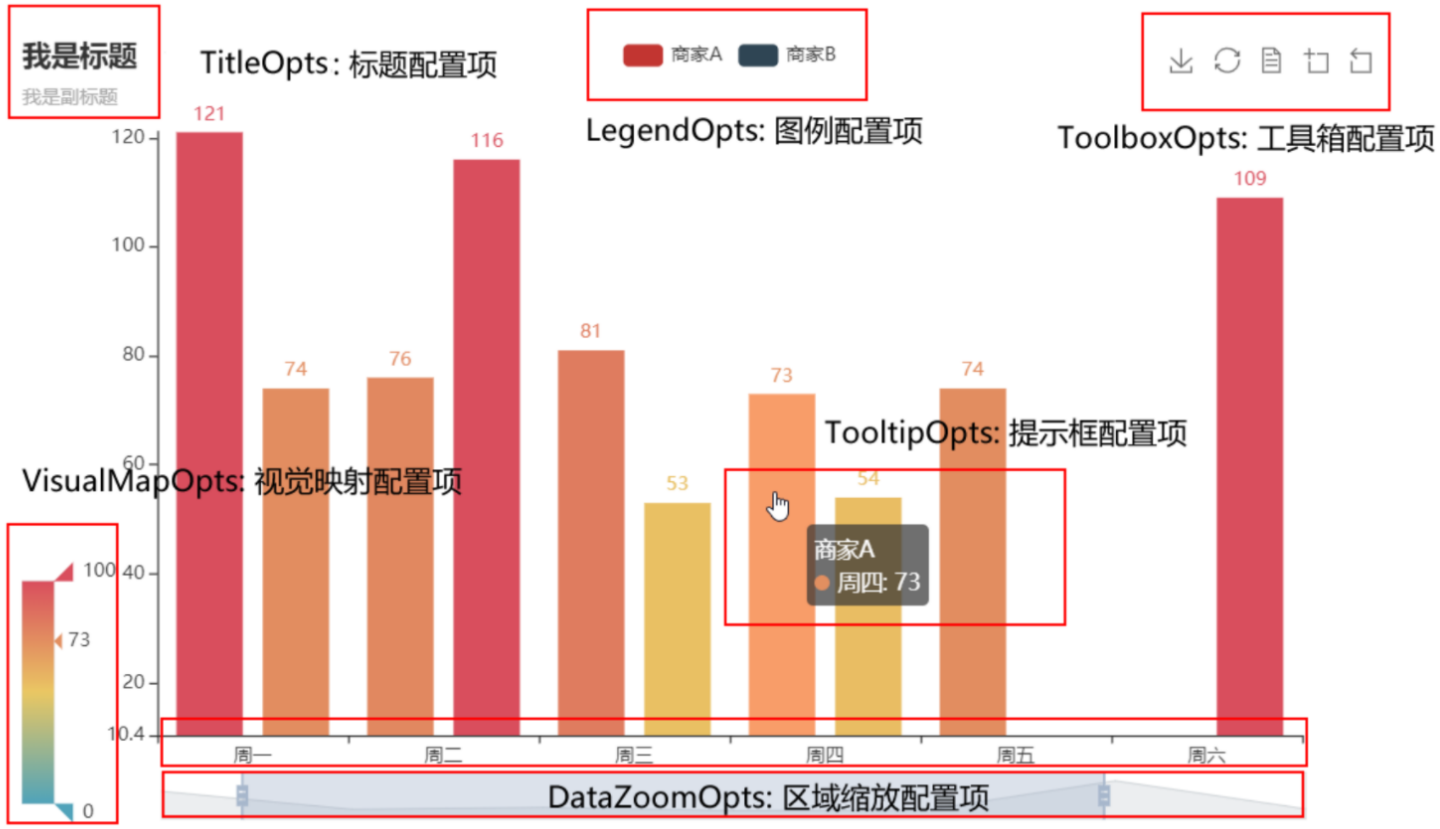


pyecharts 有哪些配置选项

- pyecharts 模块中有很多的配置选项， 常用到 2 个类别的选项：
  - 全局配置选项
  - 系列配置选项

set_global_opts 方法

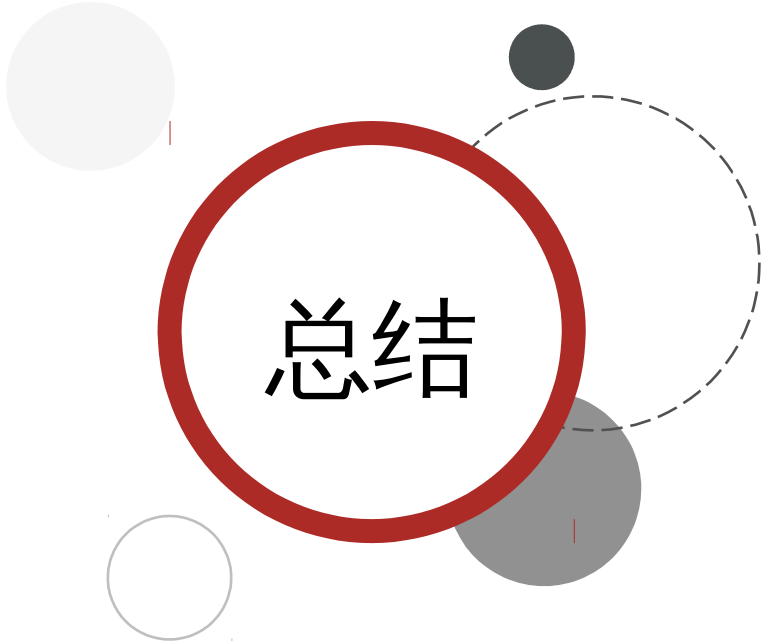
- 这里**全局配置选项**可以通过 set_global_opts 方法来进行配置， 相应的选项和选项的功能如下：



- 系列配置项，我们在后面构建案例的讲解

set_global_opts 方法

```
line.set_global_opts(
    title_opts=TitleOpts("测试", pos_left="center", pos_bottom="1%"),
    legend_opts=LegendOpts(is_show=True),
    toolbox_opts=ToolboxOpts(is_show=True),
    visualmap_opts=VisualMapOpts(is_show=True),
    tooltip_opts=TooltipOpts(is_show=True),
)
```



1. pyecharts 模块中有很多的配置选项， 常用到三个类别的选项：
   全局配置选项
   系列配置选项
2. 全局配置项能做什么？

- 配置图表的标题
- 配置图例
- 配置鼠标移动效果
- 配置工具栏
- 等整体配置项

目录

Contents

学习目标

Learning Objectives

1. 能够通过 json 模块对数据进行处理

数据处理

- **原始数据格式：**

```
jsonp_1629350871167_29498(
    {"status":0,
     "msg":"success",
     "data":
         [
             {"name":"日本",
              "trend":{"updateDate":["2.21","2.22","2.23","2.24","2.25","2.26","2.27","2.28"
                       "list":[
                                {"name":"确诊",
                                 "data":[93,105,132,144,156,164,186,210,230,239,254,268,284,
                                }
                              ]
                       }
              }
         ]
     }
);
```

数据处理

- **导入模块：**

```
# 导入json模块
import json
```

- **对数据进行整理，让数据符合 json 格式：**

```
# 把不符合 json 数据格式的 "jsonp_1629350871167_29498(" 去掉
data = data.replace("jsonp_1629350871167_29498(", "")
# 把不符合 json 数据格式的 ");" 去掉
data = data[:-2]
# 数据格式符合 json 格式后，对数据进行转化
data = json.loads(data)
# 获取日本的疫情数据
data = data["data"][0]['trend’]
# x1_data 存放日期数据
x1_data = data['updateDate’]
# y1_data 存放人数数据
y1_data = data['list'][0]["data"]
# 获取 2020 年的数据
x1_data = data['updateDate'][:314]
# 获取 2020 年的数据
y1_data = data['list'][0]["data"][:314]
```

目录

Contents

学习目标

Learning Objectives

1. 通过 pyecharts 完成疫情折线图

导入模块

- **导入模块：**

```
# 导入折线图模块
from pyecharts.charts import Line
# 导入配置选项模块
import pyecharts.options as opts
```

折线图相关配置项

- 折线图相关配置项

配置项	作用	代码实例
init_opts	对折线图初始化设置宽高	init_opts=opts.InitOpts(width="1600px", height="800px")
.add_xaxis	添加 x 轴数据	.add_xaxis(列表)
.add_yaxis	添加 y 轴数据	

折线图相关配置项

- 创建折线图

```
# 创建折线图
l = Line(init_opts=opts.InitOpts(width="1600px", height="800px"))
```

  - 这里的 Line() 是构建类对象，我们先不必理解是什么意思，后续在 Python 高阶中进行详细讲解。
  - 目前我们简单的会用即可

- 添加数据

```
l.add_xaxis()  # 添加x轴数据

l.add_yaxis()  # 添加y轴数据
```

折线图相关配置项

- .add_yaxis 相关配置选项：

配置项	作用	代码实例
series_name	设置图例名称	series_name=" 美国确诊人数 "
y_axis	输入 y 轴数据	y_axis=[" 列表 "]
symbol_size	设置点的大小	symbol_size=10
label_opts	标签设置项：不显示标签	label_opts=opts.LabelOpts(is_show=False)
linestyle_opts	线条宽度和样式	linestyle_opts=opts.LineStyleOpts(width=2)

折线图相关配置项

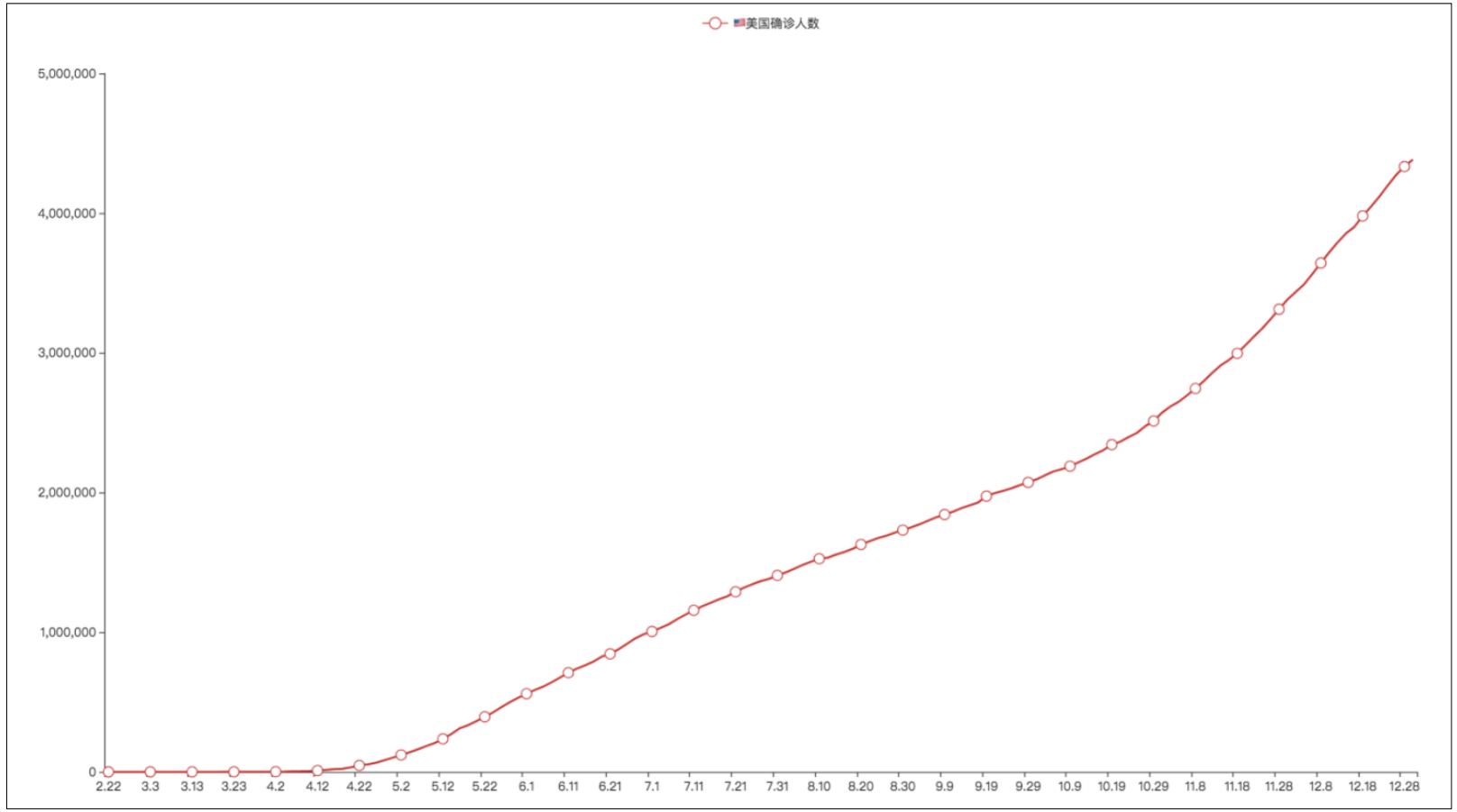
- .add_yaxis 相关配置选项：

```
l1 = (
            Line(init_opts=opts.InitOpts(width="1600px", height="800px"))

              .add_xaxis(xaxis_data=x1_data[0:-1:2])  # 添加x轴数据

              .add_yaxis(  # 配置y轴
            series_name="🇺🇸美国确诊人数",  # 设置图例名称
            y_axis=y1_data[0:-1],  # 输入y轴数据
            symbol_size=10,  # 设置点的大小
            label_opts=opts.LabelOpts(is_show=False),  # 标签设置项：显示标签
            linestyle_opts=opts.LineStyleOpts(width=2)  # 线条宽度和样式
          )
        )
```

折线图相关配置项


美国确诊人数
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
0
2.22
3.3
3.13
3.23
4.2
4.12
4.22
5.2
5.12
5.22
6.1
6.11
6.21
7.1
7.11
7.21
7.31
8.10
8.20
8.30
9.9
9.19
9.29
10.9
10.19
10.29
11.8
11.18
11.28
12.8
12.18
12.28

全局配置选项

- .set_global_opts 全局配置选项：

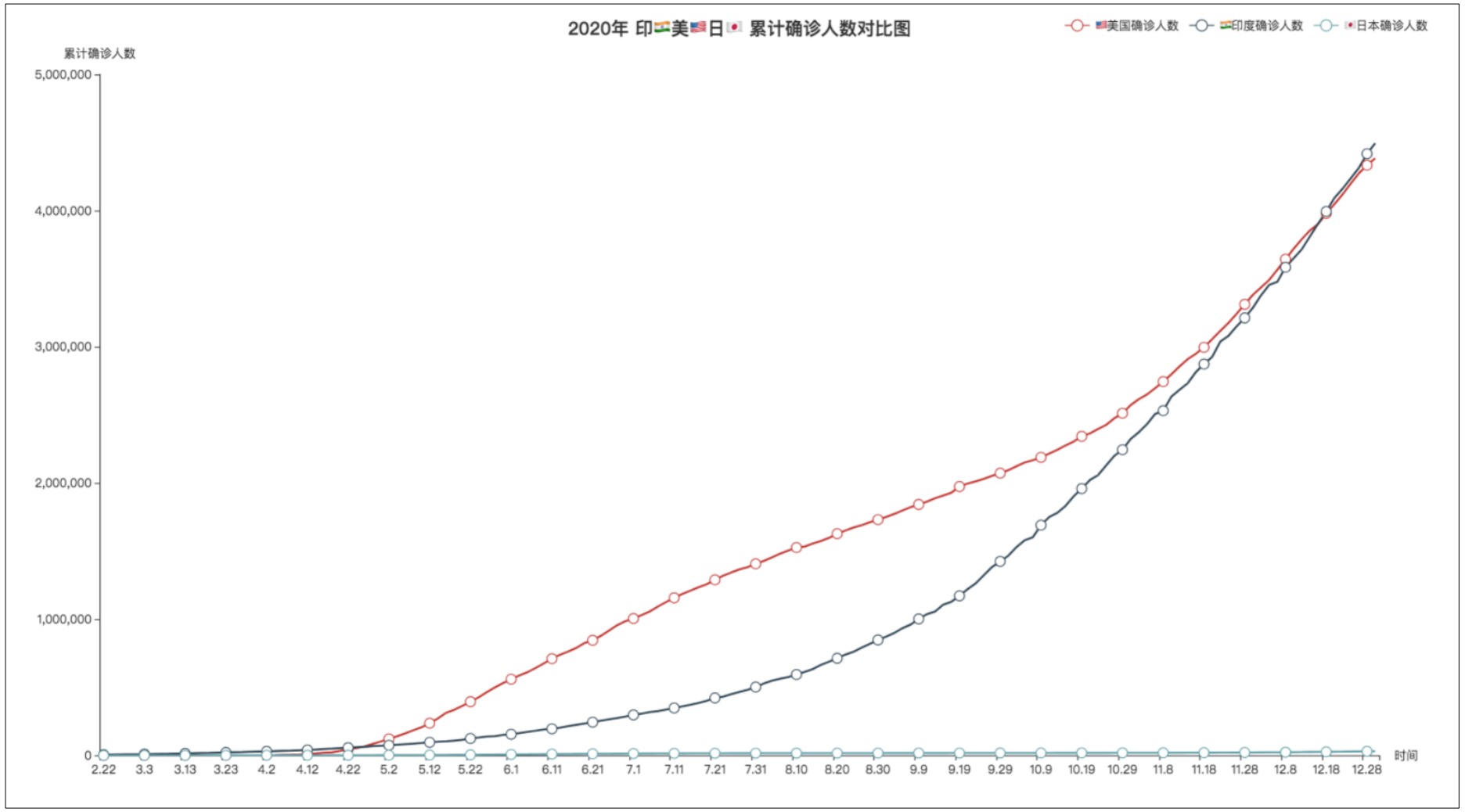
配置项	作用	代码实例
title_opts	设置图标题和位置	title_opts=opts.TitleOpts(title=" 标题 ", pos_left="center")
yaxis_opts	y 轴配置项	yaxis_opts=opts.AxisOpts(name=" 累计确诊人数 ")
xaxis_opts	x 轴配置项	xaxis_opts=opts.AxisOpts(name=" 时间 ")
legend_opts	图例配置项	legend_opts=opts.LegendOpts(pos_left='70%')

全局配置选项

- .set_global_opts 全局配置选项：

```
.set_global_opts(
        #  设置图标题和位置
        title_opts=opts.TitleOpts(title="2020 年 印□□美□□日□□ 累计确诊人数对比图 ",pos_left="center"),
        # x 轴配置项
        xaxis_opts=opts.AxisOpts(name=“ 时间” ),    #  轴标题
        # y 轴配置项
        yaxis_opts=opts.AxisOpts(name=“ 累计确诊人数” ),    #  轴标题
        #  图例配置项
        legend_opts=opts.LegendOpts(pos_left=‘70%‘),    #  图例的位置
)
```

全局配置选项


2020年 印美日 累计确诊人数对比图
美国确诊人数
印度确诊人数
日本确诊人数
累计确诊人数
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
0
2.22
3.3
3.13
3.23
4.2
4.12
4.22
5.2
5.12
5.22
6.1
6.11
6.21
7.1
7.11
7.21
7.31
8.10
8.20
8.30
9.9
9.19
9.29
10.9
10.19
10.29
11.8
11.18
11.28
12.8
12.18
12.28
时间

传智教育旗下高端IT教育品牌