

# 01- Python编程概述

大纲：

- 课程介绍
- Python简介
- 课程项目的介绍
- Python的应用

## 课程介绍

- QQ号码： 8259558
- 邮箱： 8259558@qq.com
- 群号： 938217339
- 课程网站: [https://github.com/zhoujing204/python\\_course](https://github.com/zhoujing204/python_course)



群名称: 22物联Python课程群

群 号: 938217339

## 第一讲课件

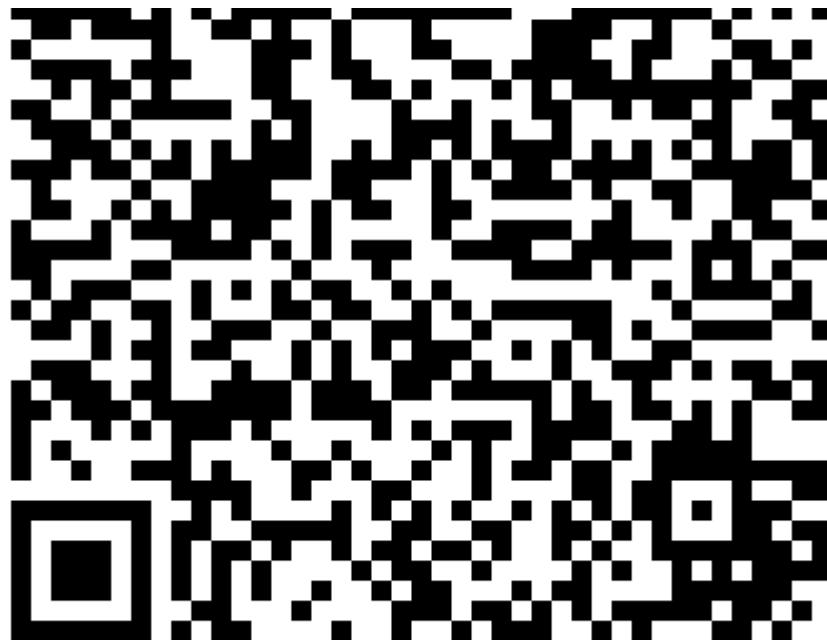
Python 仕 夷 中 走 蜥 鳄 , 蜥 鳄 是 一 种 非 常 活 泼 的 动 物 , 同 时 又 非 常 强 大 。 Py thon 语 言 也 是 如 此 , Py thon 是 一 门 非 常 优 雅 、 简 洁 的 语 言 , 学 习 起 来 非 常 容 易 , 但 是 又 非 常 的 强 大

In [4]:

```
import qrcode
from IPython.display import display

display(qrcode.make("https://github.com/zhoujing204/python_course/blob/main/src/01-introduction.ipynb"))
```





## 课程考核

- 平时成绩 (10%)
  - 考勤
  - 平时作业
- 课程实验 (60%)
  - 项目一：外星人入侵游戏
  - 项目二：数据可视化
- 期末考查 (30%)
  - 项目三：Django Web应用

## Python简介

### Python语言的诞生



- Python 的创始人为荷兰人吉多·范罗苏姆 (Guido van Rossum)
- 第一版 Python 发行于 1991 年，甚至比 Java 的历史都早。

- 在大部分时间内，Python 一直作为一个小众的编程语言，并没有大规模流行起来。
- 从2010年左右开始，随着大数据技术、人工智能技术的发展和普及，“简洁、具有良好扩展性”的 Python 非常契合大数据与人工智能技术对编程语言的要求，超越其他编程语言迅速崛起。

## 编程语言排名

信息来源: [TIOBE INDEX](#)

Aug 2024	Aug 2023	Change	Programming Language	Ratings	Change
1	1		Python	18.04%	+4.71%
2	3	▲	C++	10.04%	-0.59%
3	2	▼	C	9.17%	-2.24%
4	4		Java	9.16%	-1.16%
5	5		C#	6.39%	-0.65%
6	6		JavaScript	3.91%	+0.62%
7	8	▲	SQL	2.21%	+0.68%
8	7	▼	Visual Basic	2.18%	-0.45%
9	12	▲	Go	2.03%	+0.87%
10	14	▲	Fortran	1.79%	+0.75%
11	13	▲	MATLAB	1.72%	+0.67%

## Python语言可以做什么

- 数据科学
- 人工智能
- 科学运算
- Web应用开发
- 服务器管理
- 命令行程序
- 网络爬虫
- 嵌入式开发 (MicroPython)
- 游戏开发 (Pygame)

## Python语言的优缺点

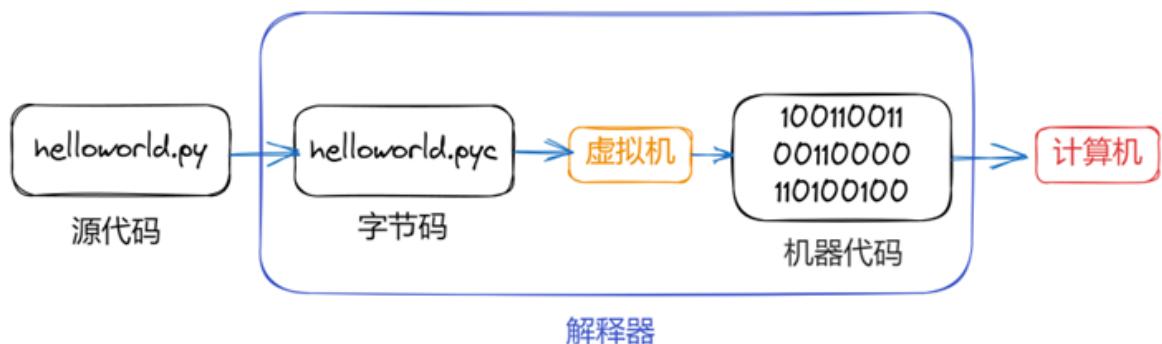
优点:

- 全能的语言
- 易于理解和使用
- 活跃的开源社区
- 丰富的软件库
- 非常适用于开发原型
- 生产力高

缺点：

- python是解释型语言，性能相对较差
- 多线程的问题
- 不能用于开发原生移动应用
- 内存消耗相对较大

## Python是解释型语言



- Python是解释型语言，和Java语言不同的是，解释器解释一句，执行一句。
- 相比编译型语言，解释型语言的性能相对较差。

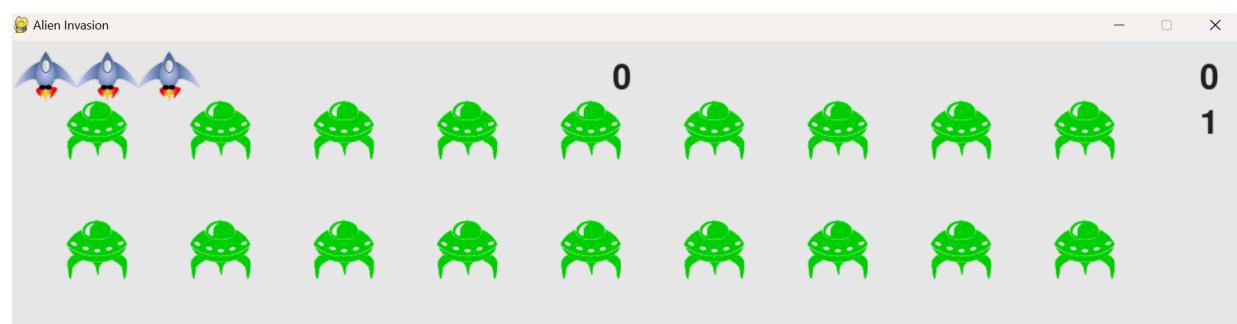
## 提升Python语言的性能

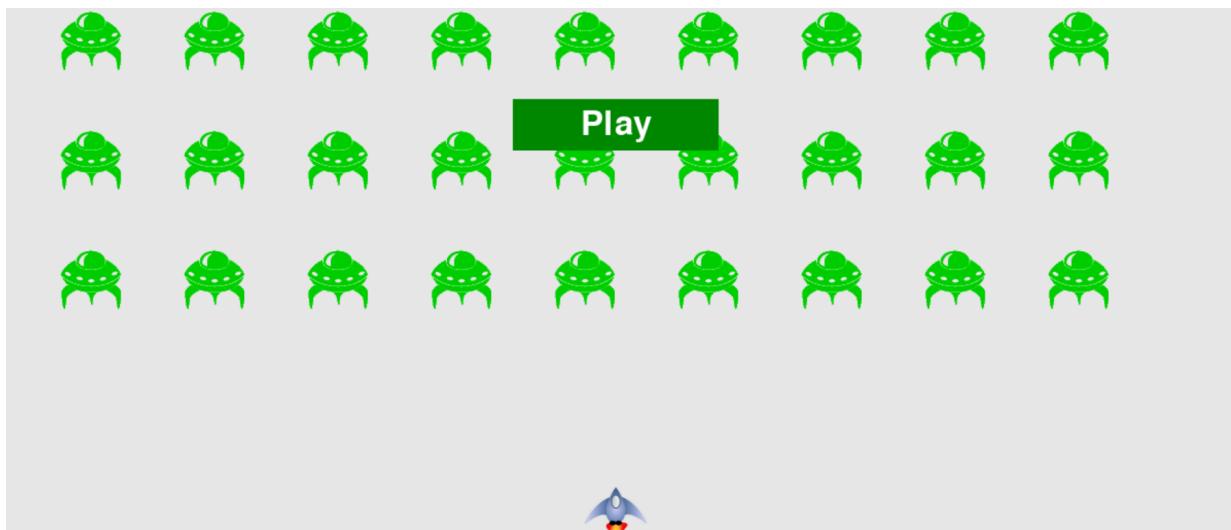
- Python语言的基础库和扩展库大量地使用了C语言来编写（CPython）。
- Python 3.11相比Python 3.10性能提升了10%-60%。
- 即时编译特性（JIT）即将加入Python 3.13。
- Python 3.13将GIL变得可选，将大大提升Python多线程性能。

## 课程项目的介绍

### 项目一：外星人入侵游戏

- 使用pygame媒体库来开发
- pygame不是游戏引擎，使用非常简单，非常适合来开发原型
- 通过该项目学习Python的基本语法和面向对象编程





## 项目二: 数据可视化

数据来源:

- 生成随机数据
- 从文件加载数据
- 从API获取数据

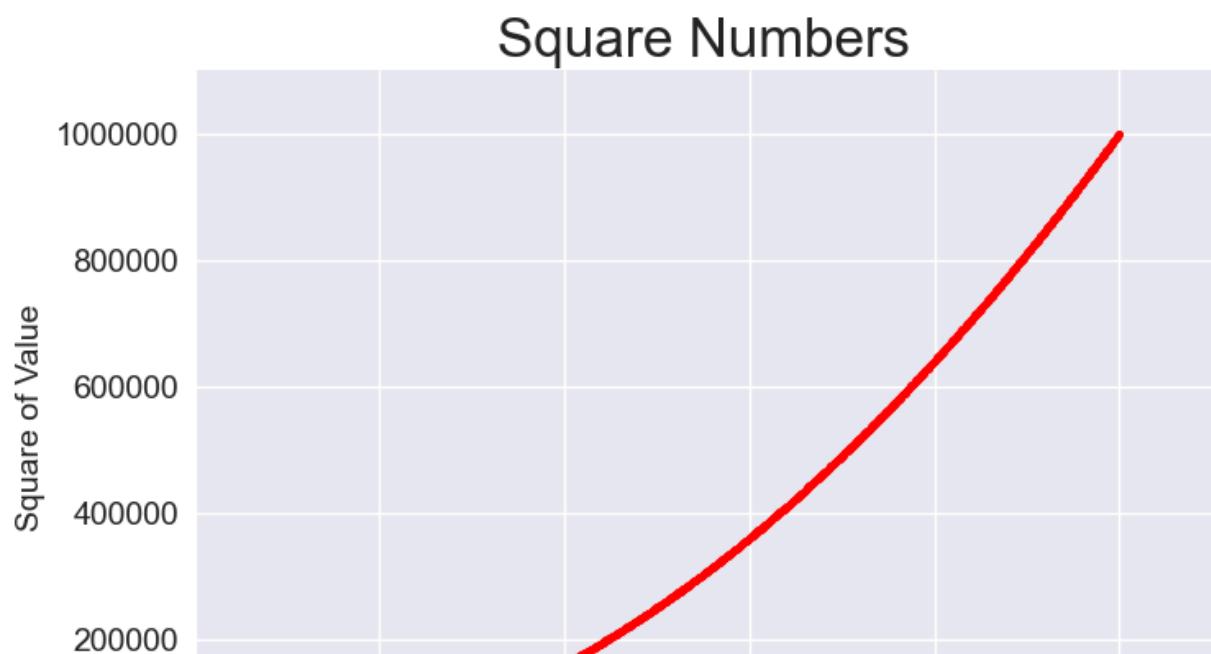
可视化数据:

- 折线图
- 柱状图
- 散点图

使用的Python包:

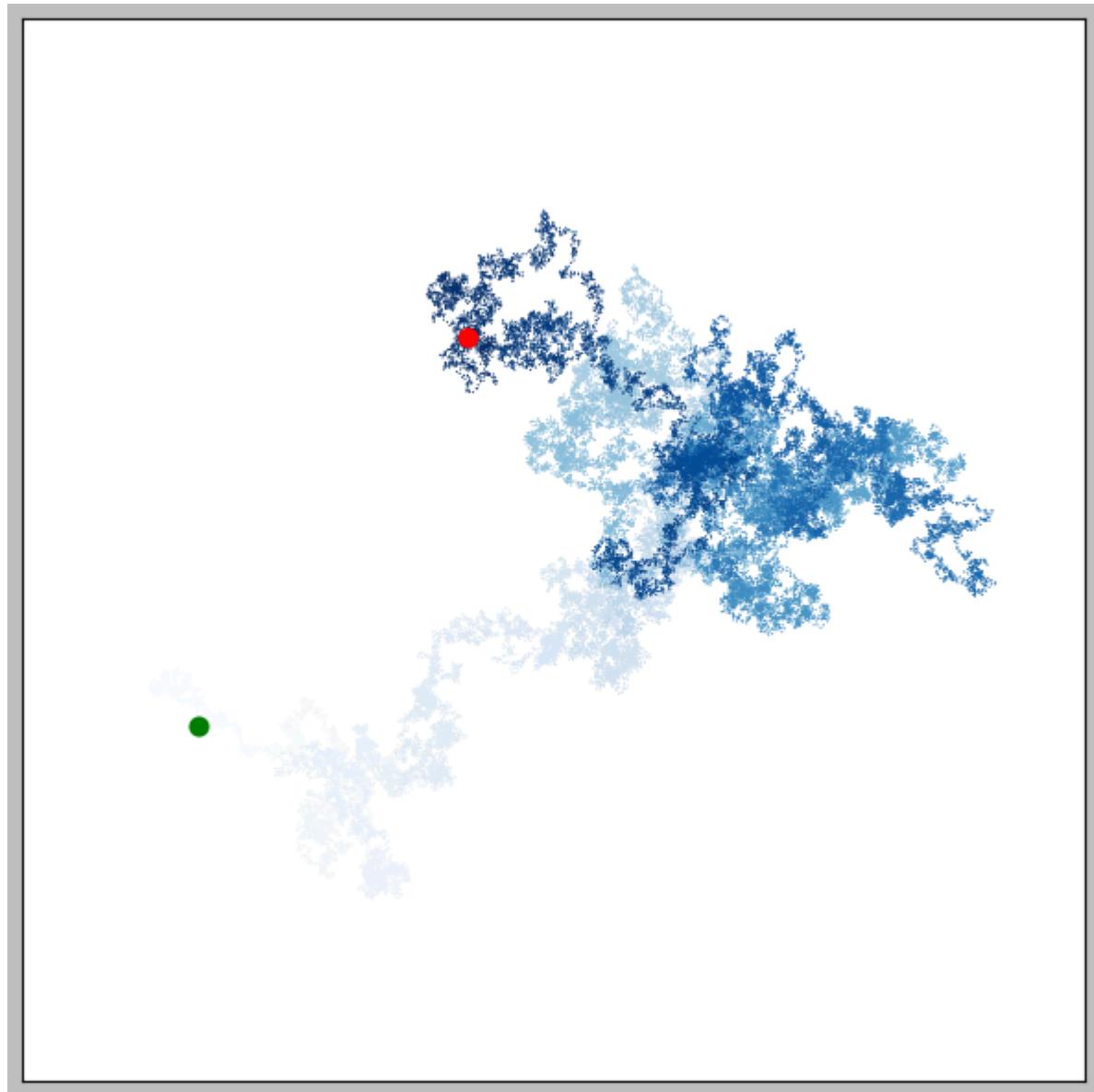
- matplotlib
- plotly

平方数据

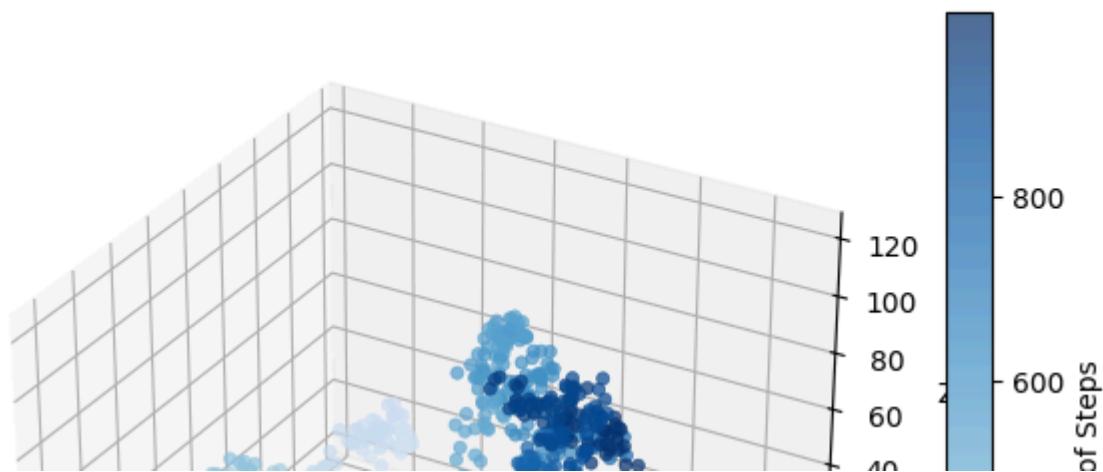


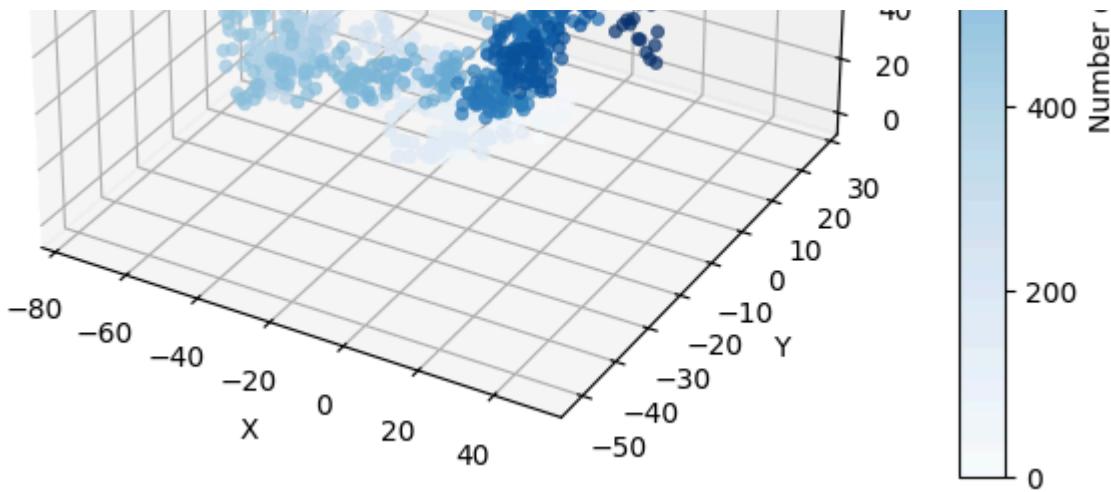


随机漫步



Three-Dimensional Random Walk





## 投掷骰子

In [5]:

```
import numpy as np
import plotly.express as px

num_sides = 6

# Make some rolls, and store results in a NumPy array.
results = np.random.randint(1, num_sides + 1, size=1000)

# Analyze the results using NumPy functions.
poss_results = np.arange(1, num_sides + 1)
frequencies = np.bincount(results, minlength=num_sides+1)[1:]

# Visualize the results.
title = "Results of Rolling One D6 1,000 Times"
labels = {'x': 'Result', 'y': 'Frequency of Result'}
fig = px.bar(x=poss_results, y=frequencies, title=title, labels=labels)
fig.show()
```

In [6]:

```
import numpy as np
import plotly.express as px

num_sides1 = 6
num_sides2 = 6
num_sides = num_sides1 + num_sides2

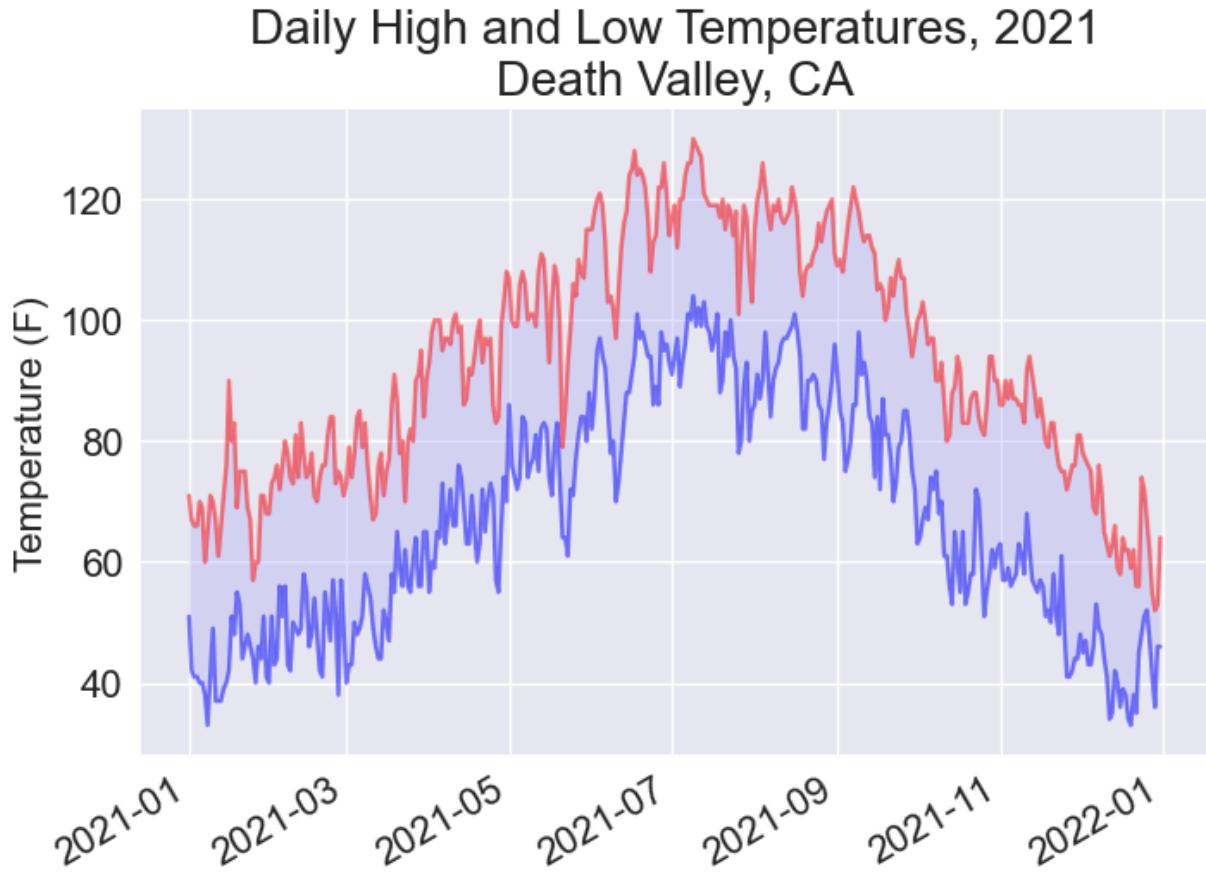
# Make some rolls, and store results in a NumPy array.
results1 = np.random.randint(1, num_sides1 + 1, size=1000)
results2 = np.random.randint(1, num_sides2 + 1, size=1000)
results = results1 + results2

# Analyze the results using NumPy functions.
poss_results = np.arange(2, num_sides + 1)
frequencies = np.bincount(results, minlength=num_sides+1)[2:]

# Visualize the results.
title = "Results of Rolling One D6 1,000 Times"
labels = {'x': 'Result', 'y': 'Frequency of Result'}
fig = px.bar(x=poss_results, y=frequencies, title=title, labels=labels)

fig.update_layout(xaxis_dtick=1)
fig.show()
```

## 天气数据



## 地震数据可视化

In [5]:

```

import pandas as pd
from pathlib import Path
import json

# Read data as a string and convert to a Python object.
path = Path('eq_data/eq_data_30_day_m1.geojson')
contents = path.read_text(encoding='utf-8')
all_eq_data = json.loads(contents)

# Examine all earthquakes in the dataset.
all_eq_dicts = all_eq_data['features'] # 一个列表，每个元素是一个字典

# 将嵌套在列表中的嵌套的字典展开，得到一个二维的dataframe
df_flat = pd.json_normalize(all_eq_dicts)
df_flat[:5]

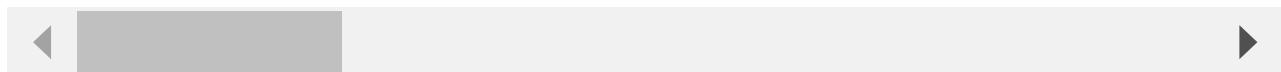
```

Out[5]:

	type	id	properties.mag	properties.place	properties.time	properties.updated
0	Feature	ak0224bju1jx	1.60	27 km NNW of Susitna, Alaska	1649051836769	1649052020
1	Feature	ak0224bjowco	1.60	63 km SE of Pedro Bay, Alaska	1649050396662	1649050785
2	Feature	ak0224bjnd7y	2.20	27 km SSE of Central Alaska	1649049962786	1649050629

3	Feature	us7000gzhx	3.70	south of Alaska	1649049432877	1649051314
4	Feature	hv72972837	2.92	49 km SE of Naalehu, Hawaii	1649048457870	1649051014

5 rows × 30 columns



In [7]:

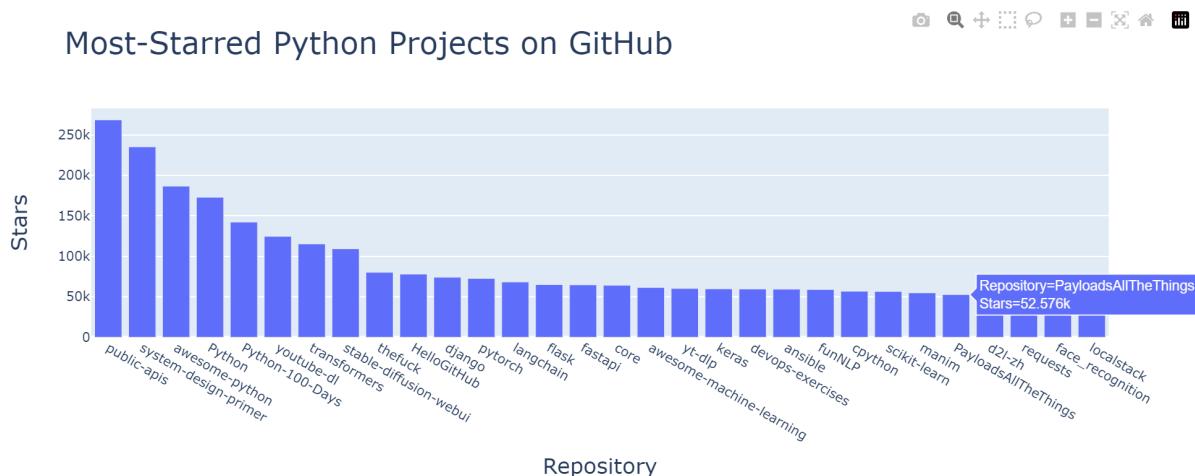
```
df = df_flat.rename(columns={'properties.mag': 'mag',
                             'properties.title': 'title'})
df['lon'] = df_flat['geometry.coordinates'].apply(lambda x: x[0])
df['lat'] = df_flat['geometry.coordinates'].apply(lambda x: x[1])
```

In [8]:

```
import plotly.express as px

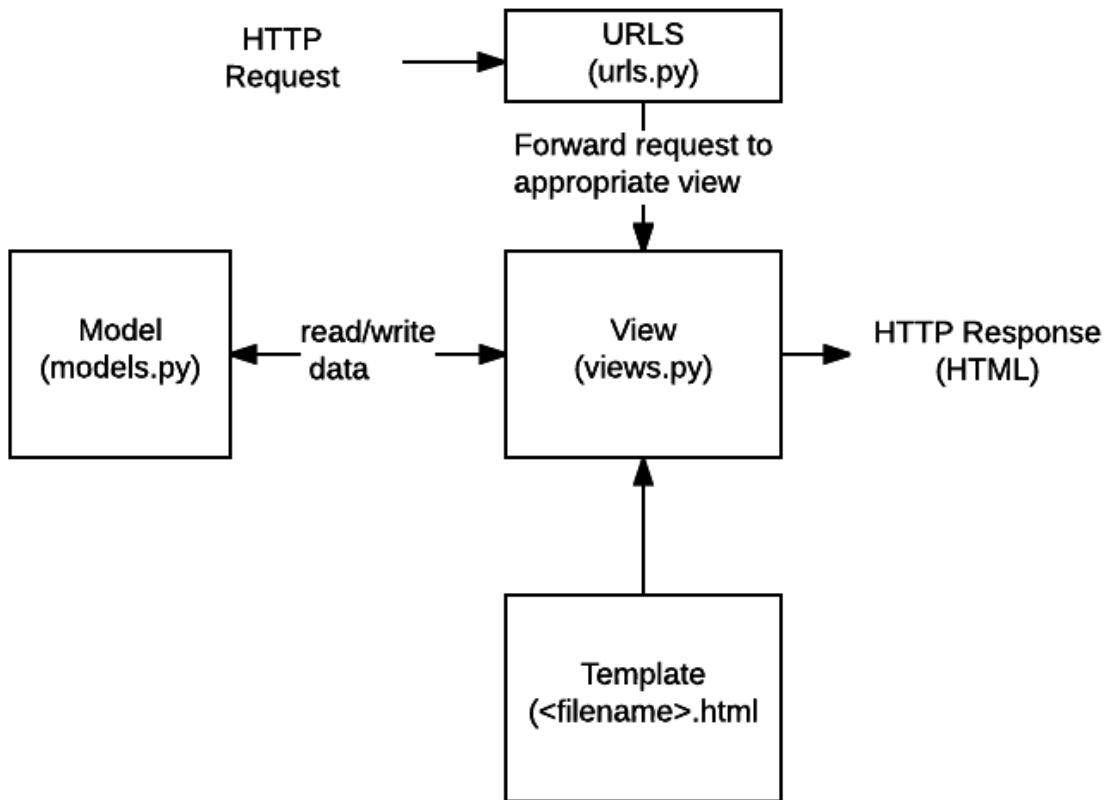
fig = px.scatter_geo(
    df,
    lon='lon',
    lat='lat',
    width=1200,
    height=800,
    title='全球地震散点图',
    size='mag',
    size_max=10,
    color='mag',
    color_continuous_scale='Viridis',
    labels={'color':'Magnitude'},
    projection='natural earth',
    hover_name='title',
)
fig.show()
```

## Github API数据可视化



## 项目三: Django Web应用

Django是Python最流行，功能最强大的Web应用框架，使用MVC架构。



Learning Log   Topics   Hello, lis.   Log out

# Track your learning.

Make your own Learning Log, and keep a list of the topics you're learning about. Whenever you learn something new about a topic, make an entry summarizing what you've learned.

[Register »](#)

## Git与Markdown

- Git: 版本控制管理工具 (Version Control Tool) , 普遍地应用于团队协作来开发软件。
  - 学习git: [learngitbranching](#)
- Markdown: 格式化的纯文本标记语言, 非常容易转换成HTML,pdf等其他格式, 因此在互联网上被广泛地使用。

## Python的应用展示

AI应用:

• 介绍与实践 - PyTorch, TensorFlow

- TensorFlow Playground: 一个用于可视化神经网络的在线工具
- 手写数字识别: MNIST
- Transformers: 深度学习模型的架构
- SAM2(Segment Anything2): 图像和视频的语义分割
- ComfyUI, StableDiffusion: 利用AI模型, 文字生成图像
- Gradio: 一个用于构建快速原型的Python库, 用于构建机器学习和深度学习模型的交互式界面

数据科学:

- 网络爬虫: BeautifulSoup, Scrapy
- 数据处理、数据分析: Numpy和Pandas
- 科学计算: SciPy, SymPy, Numpy, Pandas
- 数据可视化: Plotly, Dash, Streamlit

嵌入式开发:

- MicroPython

Web以及服务端的开发:

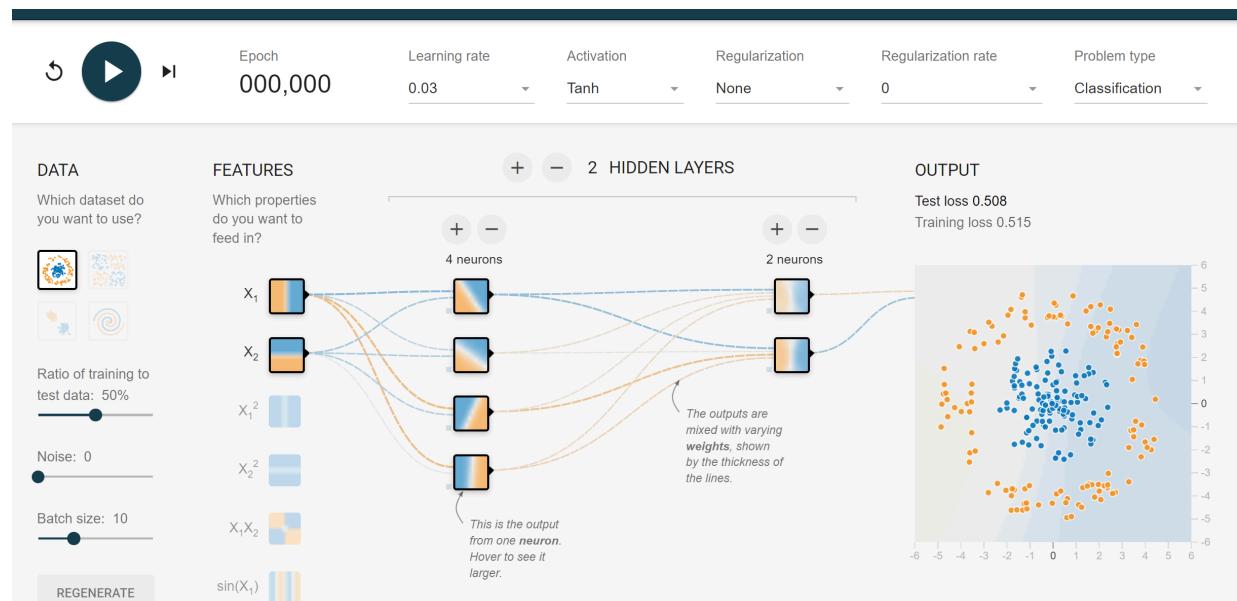
- Web框架: Django, Flask
- 服务器管理工具: Fabric, Ansible, SaltStack, psutil

命令行工具:

- 创建简易的Web服务器: http.server
- 下载流媒体: youtube-dl
- 网络调试工具: mitmproxy

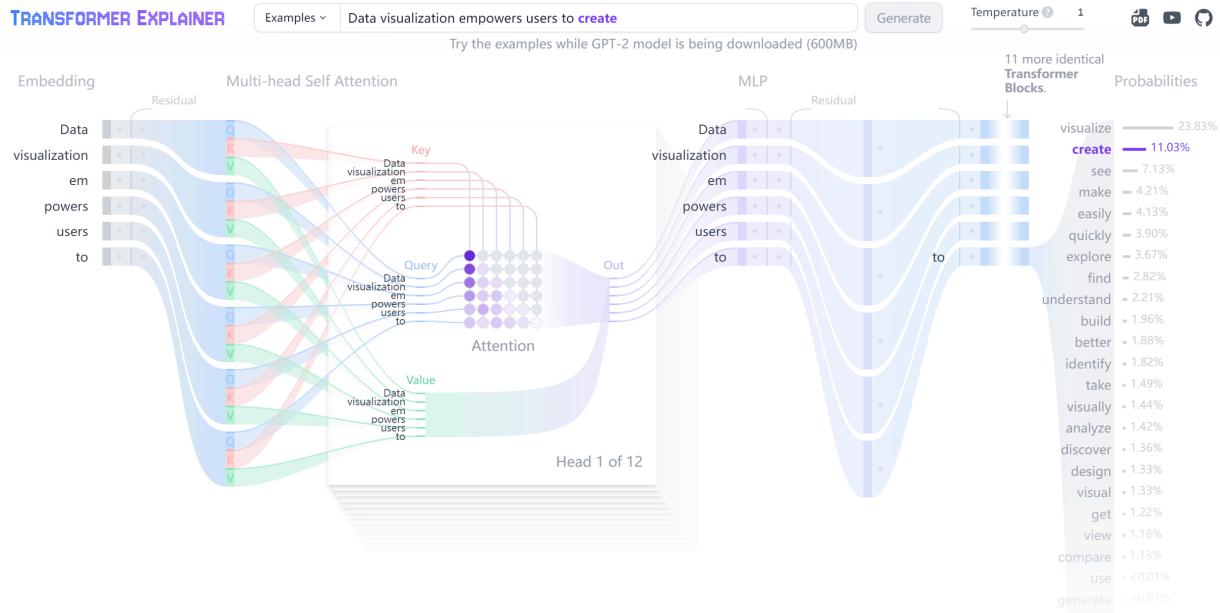
## AI应用

- 最流行的深度神经网络的框架: pytorch, tensorflow
- 了解深度神经网络的原理: [tensorflow playground](#)



## 大语言模型与Transformer:

- LLM (Large Language Model) 也就是大语言模型, 以gpt为代表的llm使用了transformer的架构
- 以gpt2为例简单介绍transformer架构: transformer explainer



## 扩散模型 (Diffusion Model) 的应用:

- 文生图 FLUX model

## FLUX.1 [merged]

Merge by [Sayak Paul](#) of 2 of the 12B param rectified flow transformers [FLUX.1 \[dev\]](#) and [FLUX.1 \[schnell\]](#) by [Black Forest Labs](#)

a tiny astronaut hatching from an egg on the moon

Run

Number of inference steps

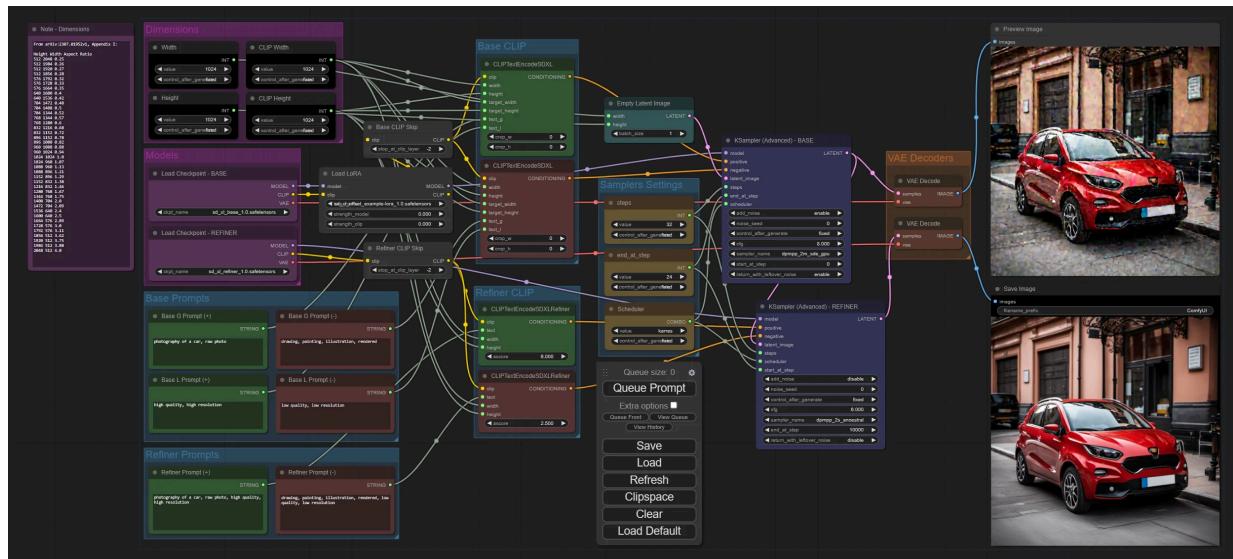
8





## Advanced Settings

- 使用ComfyUI来定制生成图片的工作流: [项目地址](#)

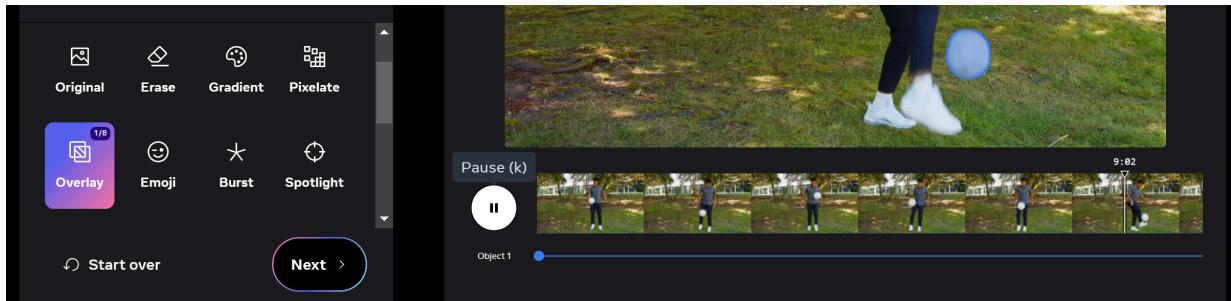


## 一些有趣的AI应用

- SAM2 (视频与图像的语义化分割) : [demo地址](#), [项目地址](#)
- 虚拟换衣应用: [Kolors-Virtual-Try-On](#)

The screenshot shows the Segment Anything 2 Demo website, which allows users to apply visual effects to selected objects in a video. Key features shown include:

- Segment Anything 2 Demo**: The main title.
- A Meta FAIR Demo**: Subtitle.
- Add effects**: A step indicator (3/3).
- Description**: Text explaining how to apply visual effects to selected objects and the background.
- Surprise Me**: A button to generate random effects.
- Preview Window**: Shows a person in a dynamic pose outdoors, with a green bounding box highlighting a selected area.
- Navigation**: Links for "About", "Dataset", and "AI Demos".



## Kolors Virtual Try-On in the Wild

Tech Report [Kolors](#) Official Website [Page](#)

Step 1. Upload a person image

Step 2. Upload a garment image

Step 3. Press "Run" to get try-on results

Person image

Examples

Garment image

Examples

Result

Seed: 0  Random seed

Seed used: 986196 Response: Success

[Run](#)

## 更多AI的应用

- AI被应用于游戏：AlphaGo
- AI被应用于生物医药：AlphaFold 3
- AI被应用于数学：AlphaGeometry
- AI在物理学、考古等等很多领域将会有越来越多地应用

## 数据科学

### 网络爬虫

In [5]:

```
from bs4 import BeautifulSoup
import requests

# 获取网页内容
url = 'https://example.com'
response = requests.get(url)
html_content = response.text
```

```
# 解析 HTML
soup = BeautifulSoup(html_content, 'html.parser')

# 提取标题
title = soup.title.string
print(f"网页标题: {title}")

# 提取所有链接
links = soup.find_all('a')
for link in links:
    print(f"链接文本: {link.text}, 链接地址: {link.get('href')})")
```

网页标题: Example Domain

链接文本: More information..., 链接地址: https://www.iana.org/domains/example

In [1]:

```
import requests
from bs4 import BeautifulSoup

# 发送请求并获取页面
url = "https://en.wikipedia.org/wiki/2024_Summer_Olympics_medal_table"
response = requests.get(url)

# 解析HTML内容
soup = BeautifulSoup(response.text, 'html.parser')

# 找到奖牌榜表格
table = soup.find('table', {'class': 'wikitable'})

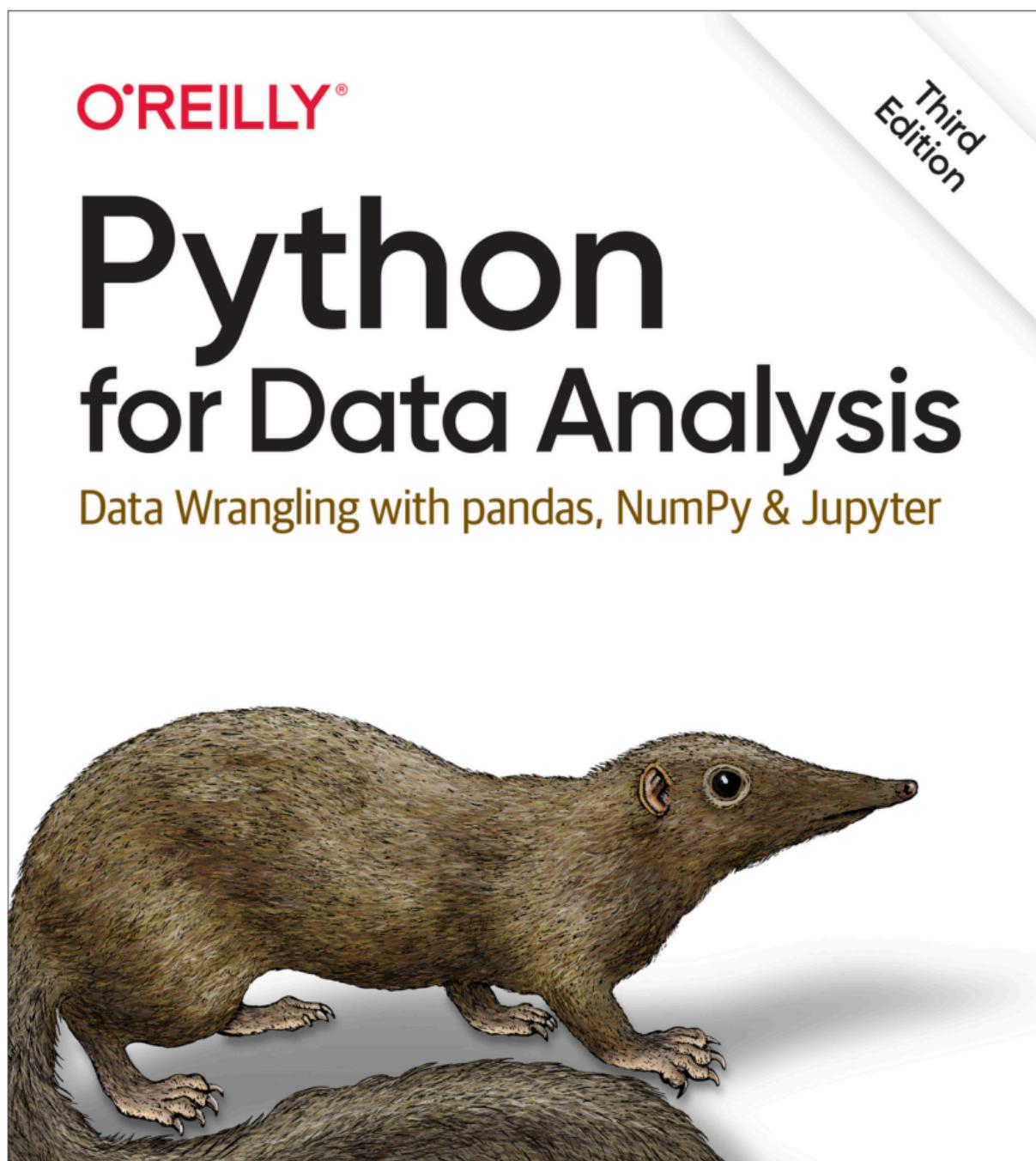
# 打印表头
headers = [header.text.strip() for header in table.find_all('th')]
print(headers)

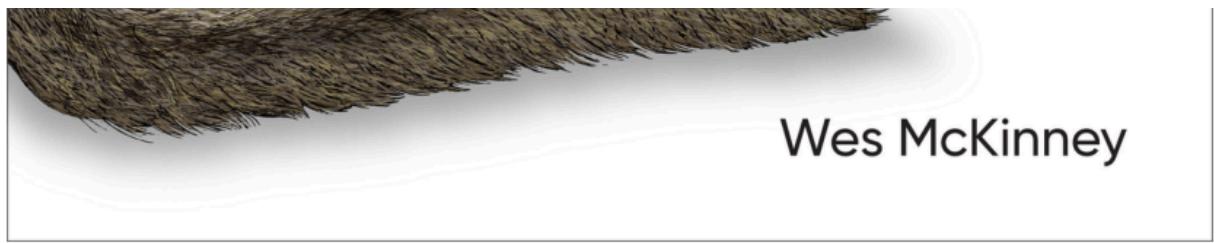
# 打印每一行的数据
for row in table.find_all('tr')[1:]: # 跳过表头
    cols = row.find_all('td')
    data = [col.text.strip() for col in cols]
    print(data)
```

```
['Rank', 'NOC', 'Gold', 'Silver', 'Bronze', 'Total', 'United States†', 'China', 'Japan', 'Australia', 'France*', 'Netherlands', 'Great Britain', 'South Korea', 'Italy', 'Germany', 'New Zealand', 'Canada', 'Uzbekistan', 'Hungary', 'Spain', 'Sweden', 'Kenya', 'Norway', 'Ireland', 'Brazil', 'Iran', 'Ukraine', 'Romania‡', 'Georgia', 'Belgium', 'Bulgaria', 'Serbia', 'Czech Republic', 'Denmark', 'Azerbaijan', 'Croatia', 'Cuba', 'Bahrain', 'Slovenia', 'Chinese Taipei', 'Austria', 'Hong Kong', 'Philippines', 'Algeria', 'Indonesia', 'Israel', 'Poland', 'Kazakhstan', 'Jamaica', 'South Africa', 'Thailand', 'Individual Neutral Athletes[A][B]', 'Ethiopia', 'Switzerland', 'Ecuador', 'Portugal', 'Greece', 'Argentina', 'Egypt', 'Tunisia', 'Botswana', 'Chile', 'Saint Lucia', 'Uganda', 'Dominican Republic', 'Guatemala', 'Morocco', 'Dominica', 'Pakistan', 'Turkey', 'Mexico', 'Armenia', 'Colombia', 'Kyrgyzstan', 'North Korea', 'Lithuania', 'India', 'Moldova', 'Kosovo', 'Cyprus', 'Fiji', 'Jordan', 'Mongolia', 'Panama', 'Tajikistan', 'Albania', 'Grenada', 'Malaysia', 'Puerto Rico', 'Cape Verde', 'Ivory Coast', 'Peru', 'Qatar', 'Refugee Olympic Team', 'Singapore', 'Slovakia', 'Zambia', 'Totals (91 entries)', '329', '330', '385', '1,044']
[1, '40', '44', '42', '126']
[2, '40', '27', '24', '91']
[3, '20', '12', '13', '45']
[4, '18', '19', '16', '53']
[5, '16', '26', '22', '64']
[6, '15', '7', '12', '34']
[7, '14', '22', '29', '65']
[8, '13', '9', '10', '32"]
[9, '12', '13', '15', '40"]
[10, '12', '13', '8', '33']
```

```
[1/4, '0', '1', '0', '1']
['0', '1', '0', '1']
['0', '1', '0', '1']
['0', '1', '0', '1']
['0', '1', '0', '1']
['0', '1', '0', '1']
['79', '0', '0', '3', '3']
['80', '0', '0', '2', '2']
['0', '0', '2', '2']
['0', '0', '2', '2']
['0', '0', '2', '2']
['84', '0', '0', '1', '1']
['0', '0', '1', '1']
['0', '0', '1', '1']
['0', '0', '1', '1']
['0', '0', '1', '1']
['0', '0', '1', '1']
['0', '0', '1', '1']
['0', '0', '1', '1']
[]
```

数据处理和数据分析： numpy与pandas





Wes McKinney

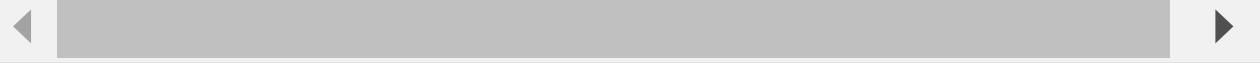
## 数据可视化

用于数据可视化的Python包:

- Matplotlib
- Plotly: [Plotly官网](#)
- Seaborn
- Bokeh

In [2]:

```
import plotly.express as px
df = px.data.gapminder().query("year == 2007").query("continent == 'Europe'")
df.loc[df['pop'] < 2.e6, 'country'] = 'Other countries' # Represent only Large countries
fig = px.pie(df, values='pop', names='country', title='Population of European continent')
fig.show()
```



## 数据可视化应用

- plotly: [dash官网范例](#)
- streamlit: [官网范例](#)

In [4]:

```
from dash import Dash, dcc, html, Input, Output
import plotly.express as px

app = Dash(__name__)

app.layout = html.Div([
    html.H4('Analysis of Iris data using scatter matrix'),
    dcc.Dropdown(
        id="dropdown",
        options=['sepal_length', 'sepal_width', 'petal_length', 'petal_width'],
        value=['sepal_length', 'sepal_width'],
        multi=True
    ),
    dcc.Graph(id="graph"),
])

@app.callback(
    Output("graph", "figure"),
    Input("dropdown", "value"))
def update_bar_chart(dims):
    df = px.data.iris() # replace with your own data source
    fig = px.scatter_matrix(
        df, dimensions=dims, color="species")
    return fig
```

```
app.run_server(debug=True)
```

## 科学计算

几乎所有的学科（包括社会科学）都需要使用科学计算：

- 微积分
- 线性代数
- 概率论与统计学

解微分方程  $\frac{dy}{dx} + 2 * y = 0$

In [13]:

```
# 符号运算

import sympy as sp

# 定义符号
x = sp.symbols('x')
y = sp.Function('y')(x)

# 定义微分方程
differential_eq = sp.Eq(sp.diff(y, x) + 2*y, 0)

# 求解微分方程
solution = sp.dsolve(differential_eq, y)

solution
```

Out[13]:  $y(x) = C_1 e^{-2x}$ 

In [14]:

```
# 数值运算

import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint

# 定义微分方程
def model(y, t):
    dydt = -2 * y
    return dydt

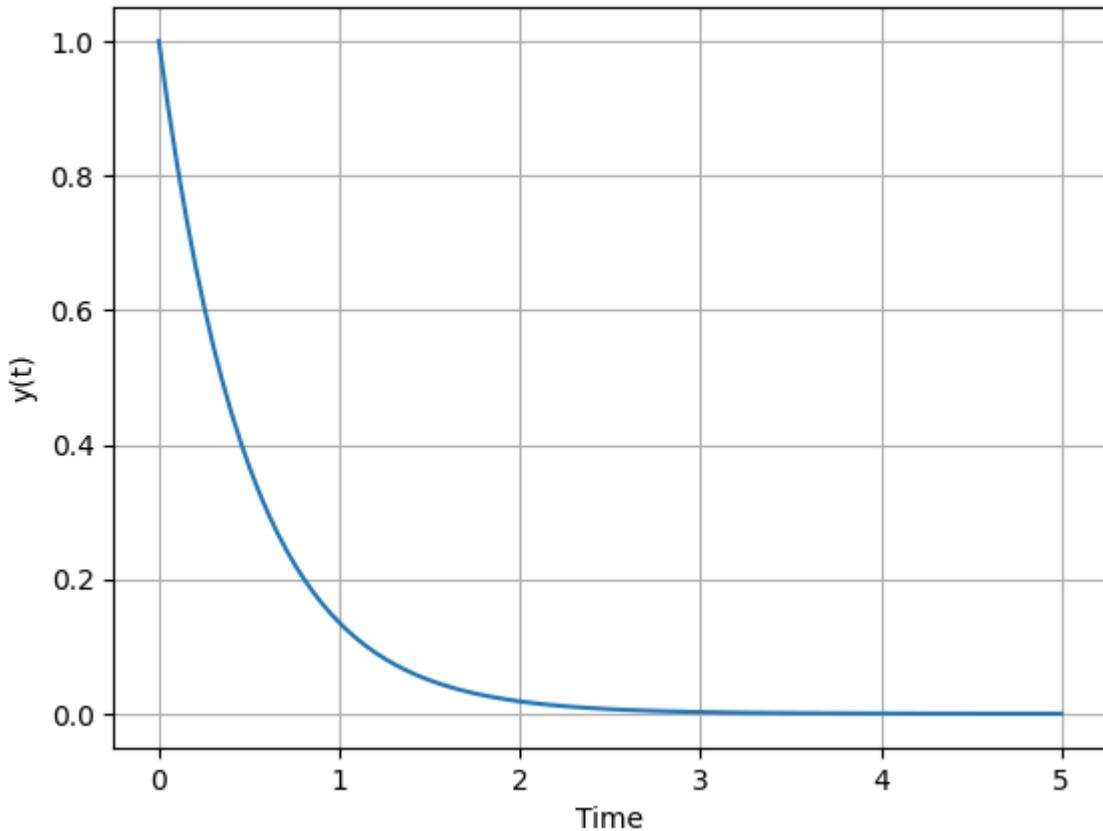
# 初始条件
y0 = 1

# 时间范围
t = np.linspace(0, 5, 100)

# 求解微分方程
solution = odeint(model, y0, t)

# 绘制结果
plt.plot(t, solution)
plt.title('Solution of dy/dt = -2y')
plt.xlabel('Time')
plt.ylabel('y(t)')
plt.grid()
plt.show()
```

### Solution of $dy/dt = -2y$



In [15]: # 线性代数的例子  $Ax=b$

```

import numpy as np

# 定系数矩阵 A 和常数向量 b
A = np.array([[2, 3],
              [4, 1]])

b = np.array([5, 11])

# 解线性方程组
solution = np.linalg.solve(A, b)

# 输出结果
print("解为: ")
print(f"x1 = {solution[0]}, x2 = {solution[1]}")

```

解为：

$x_1 = 2.8, x_2 = -0.2$

In [16]: import numpy as np

```

# 定义矩阵 A
A = np.array([[4, 7],
              [2, 6]])

# 计算逆矩阵
A_inv = np.linalg.inv(A)

# 输出结果
print("矩阵 A 的逆矩阵为: ")
print(A_inv)

```

矩阵 A 的逆矩阵为：

```
[[ 0.6 -0.7]
 [-0.2  0.4]]
```

## 嵌入式开发--MicroPython

MicroPython 是一种轻量级的 Python 实现，专为微控制器和嵌入式系统设计。 MicroPython 可以用于：

1. **硬件控制**：可以控制传感器、马达、LED 等硬件设备。
2. **物联网 (IoT)**：用于开发连接到互联网的设备，例如温度监测器和智能家居设备。
3. **机器人**：可以编写控制机器人的代码，实现自主移动和任务执行。
4. **数据采集**：从传感器收集数据，并进行实时处理或存储。
5. **教育**：适合编程和电子基础的教学，简单易学。
6. **快速原型开发**：快速开发和测试新想法，验证设计。

MicroPython支持的平台：

1. **ESP8266**：流行的Wi-Fi模块，适用于物联网项目。
2. **ESP32**：功能强大的微控制器，支持Wi-Fi和蓝牙，适合复杂应用。
3. **Raspberry Pi Pico**：基于RP2040芯片的小型开发板，适合教育和原型开发。
4. **STM32**：多种STM32系列微控制器，适用于工业和嵌入式应用。
5. **Pyboard**：专为MicroPython设计的开发板，具有多种接口。
6. **BBC micro:bit**：适合教育目的的小型开发板，易于使用。
7. **Nordic nRF系列**：用于蓝牙低能耗应用的微控制器。

## 总结





"The limits of my language are the limits of my world"—Ludwig Wittgenstein  
(1889-1951)

"我的语言的局限就是我的世界的局限."-- Ludwig Wittgenstein (1889-1951)

