

第 3 章 Blockly 数据输入、 显示、可视化

目录

1. 读取或输入数据
2. 数据表格讲解
3. 数据表格运算
4. 显示数据图像

1、读取或输入数据 (1/2)

- 首先导入依赖库

导入依赖库，系统：Windows

- 读取Excel文件：

赋值 df 为 从 “data.xlsx” 中读取数据表格

- 从词典中直接输入数据：

赋值 df 为 创建数据表格，数据是 创建一个词典，内容：

- 最终结构：

导入依赖库，系统：Windows

赋值 df 为 创建数据表格，数据是 创建一个词典，内容：

“A” : 列表: 1, 2, 3

“B” : 列表: 4, 5, 6

输出 df

```
df = None

import pandas as pd
import matplotlib.pyplot as plt
import statistics
import math
plt.rcParams["axes.unicode_minus"]=False
plt.rcParams["font.sans-serif"]=["SimHei"]

df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
print(df)
```

1、读取或输入数据 (2/2)

- 让我们把df打印出来看看:

```
In [2]: df = None

import pandas as pd
import matplotlib.pyplot as plt
import statistics
import math
plt.rcParams["axes.unicode_minus"] = False
plt.rcParams["font.sans-serif"] = ["SimHei"]

df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
print(df)
```

	A	B
0	1	4
1	2	5
2	3	6

可以看到列名是A、B，索引是自动从0开始编号的

2、数据表格讲解

第一部分创建变量后，这个变量就是数据表格。我们可以对他进行很多数据处理操作。

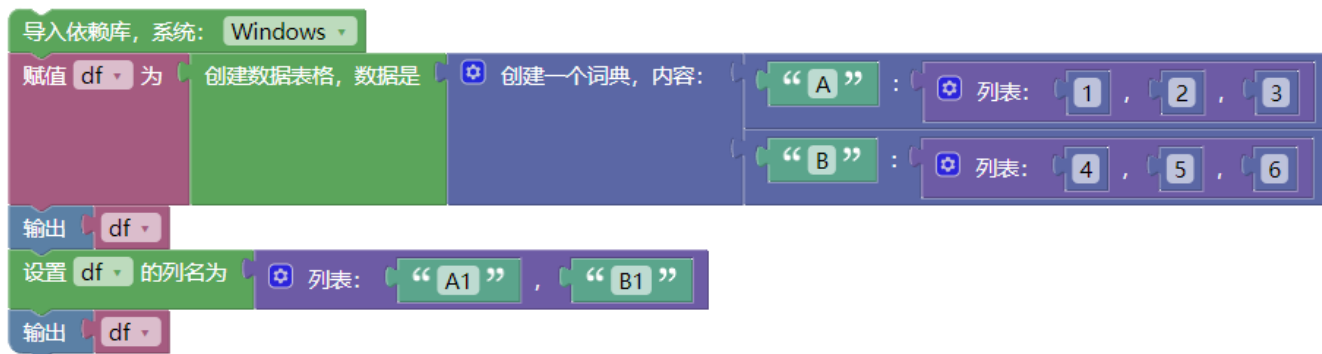
比如：

- 重新设置列名
- 重新设置索引
- 获取第n列数据
- 获取第n行数据
- 获取某一列数据
- 在df中添加新的一列



2、数据表格讲解

- 重新设置列名



- 可以看到df的列名已更改

```
In [3]: df = None

import pandas as pd
import matplotlib.pyplot as plt
import statistics
import math
plt.rcParams["axes.unicode_minus"] = False
plt.rcParams["font.sans-serif"] = ["SimHei"]

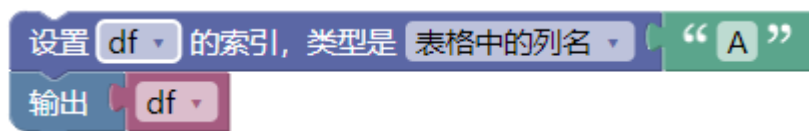
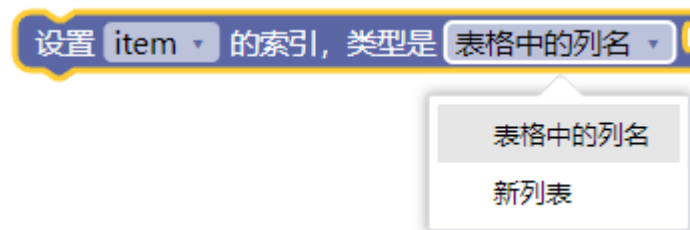
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
print(df)
df.columns = ['A1', 'B1']
print(df)
```

	A	B
0	1	4
1	2	5
2	3	6

	A1	B1
0	1	4
1	2	5
2	3	6

2、数据表格讲解

- 重新设置索引
- 这里有两种选择，使用原表格中的某一列作为索引，或者给出新列表作为索引
- 我们分别尝试，先用“A”列作为索引；然后再给出指定的列表作为索引，可以看到df的索引变化情况



	A	B
0	1	4
1	2	5
2	3	6

原df

	A	B
1	4	
2	5	
3	6	

使用A列作为索引

	A	B
S1	1	4
S2	2	5
S3	3	6

重新指定列表为索引

2、数据表格讲解

- 获取第n列数据
- 获取某一系列数据（根据列名）
- 获取第n行数据

输出 获取 df 的第 1 列数据

输出 获取 df 的 “A” 列数据

输出 获取 df 的第 1 行数据

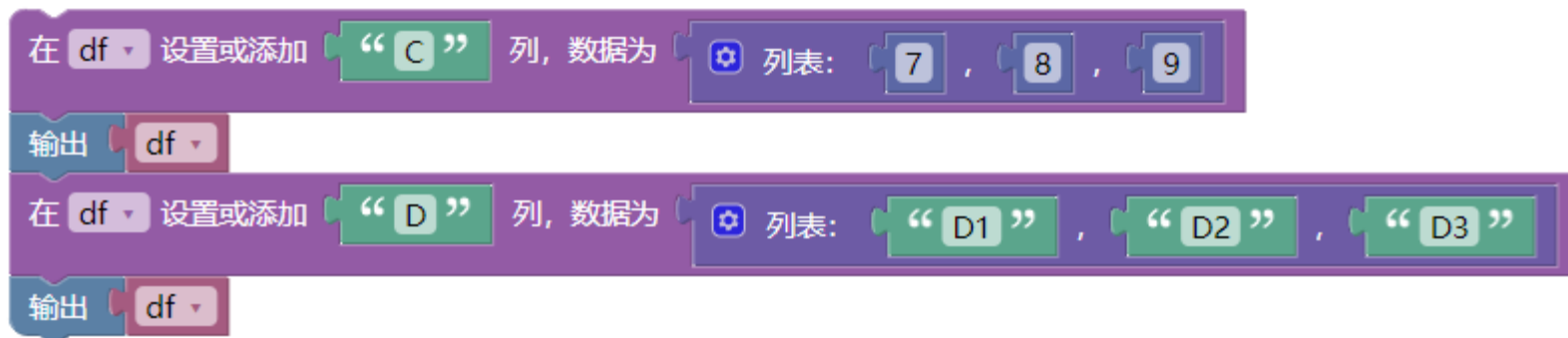
	A	B
0	1	4
1	2	5
2	3	6

原数据

```
0    1
1    2
2    3
Name: A, dtype: int64
0    1
1    2
2    3
Name: A, dtype: int64
A    1
B    4
Name: 0, dtype: int64
```


2、数据表格讲解

- 在df中添加新的一列，列表里可以是数字、字符串



```
A B
0 1 4
1 2 5
2 3 6
```

原数据

```
A B C
0 1 4 7
1 2 5 8
2 3 6 9
```

```
A B C D
0 1 4 7 D1
1 2 5 8 D2
2 3 6 9 D3
```

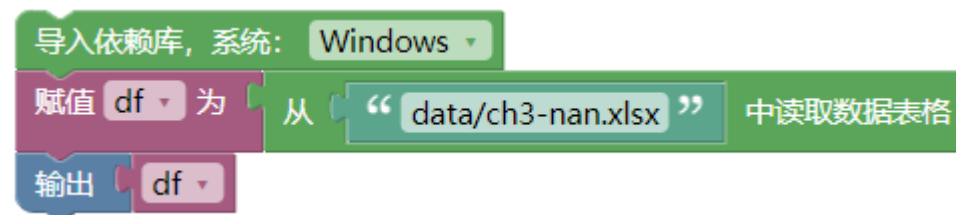
新数据

2、数据表格讲解

- 缺失值处理

读取ch3-nan.xlsx

数据本身有缺失值（空值）



A	B
1	4
2	5
	6
4	1
5	
	4

打印后可以观察到缺失值显示为NaN

	A	B
0	1.0	4.0
1	2.0	5.0
2	NaN	6.0
3	4.0	1.0
4	5.0	NaN
5	NaN	4.0

2、数据表格讲解

- 使用 数据预处理 的

可以实现缺失值处理
方案有：

填充 item 的缺失值，策略是 前值填充

- ✓ 前值填充
- 后值填充
- 指定值

填充 df 的缺失值，策略是 指定值 : 99

填充 df 的缺失值，策略是 指定值 : 计算 df 的 均值

A 3.0
B 4.0
dtype: float64

	A	B
0	1.0	4.0
1	2.0	5.0
2	NaN	6.0
3	4.0	1.0
4	5.0	NaN
5	NaN	4.0

	A	B
0	1.0	4.0
1	2.0	5.0
2	2.0	6.0
3	4.0	1.0
4	5.0	1.0
5	5.0	4.0

前值填充

	A	B
0	1.0	4.0
1	2.0	5.0
2	NaN	6.0
3	4.0	1.0
4	5.0	NaN
5	NaN	4.0

	A	B
0	1.0	4.0
1	2.0	5.0
2	4.0	6.0
3	4.0	1.0
4	5.0	4.0
5	NaN	4.0

后值填充

	A	B
0	1.0	4.0
1	2.0	5.0
2	NaN	6.0
3	4.0	1.0
4	5.0	NaN
5	NaN	4.0

	A	B
0	1.0	4.0
1	2.0	5.0
2	99.0	6.0
3	4.0	1.0
4	5.0	99.0
5	99.0	4.0

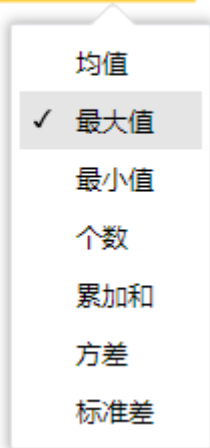
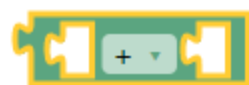
指定值 (99)

	A	B
0	1.0	4.0
1	2.0	5.0
2	3.0	6.0
3	4.0	1.0
4	5.0	4.0
5	3.0	4.0

(均值)

3、数据表格运算

- 基础运算



	A	B
0	1	4
1	2	5
2	3	6

原数据

- 其他函数

```
A    3
B    6
dtype: int64
```

获取每列的最大值

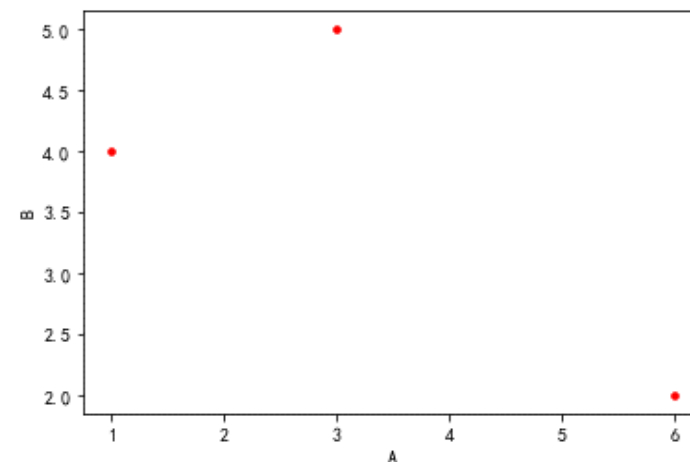
4、显示数据图像

- 可以为数据表格创建散点图、折线图、柱形图

- 散点图

赋值 `item` 为 创建 `df` 的散点图，标记是 `圆点`，标记颜色是 `红色`，标记大小是 `20`
第 `1` 列作为x轴，第 `2` 列作为y轴

	A	B
0	1	4
1	3	5
2	6	2

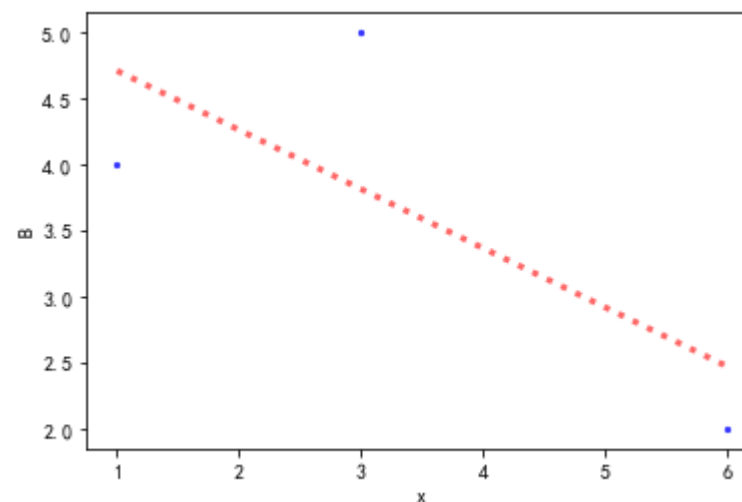


4、显示数据图像

- 散点图并拟合图像

创建 `df` 的散点图，标记是 `圆点`，标记颜色是 `蓝色`，标记大小是 `20`
第 `1` 列作为x轴，第 `2` 列作为y轴
并拟合图像，直线颜色 `红色`，线形 `虚线`，粗细 `3`

	A	B
0	1	4
1	3	5
2	6	2

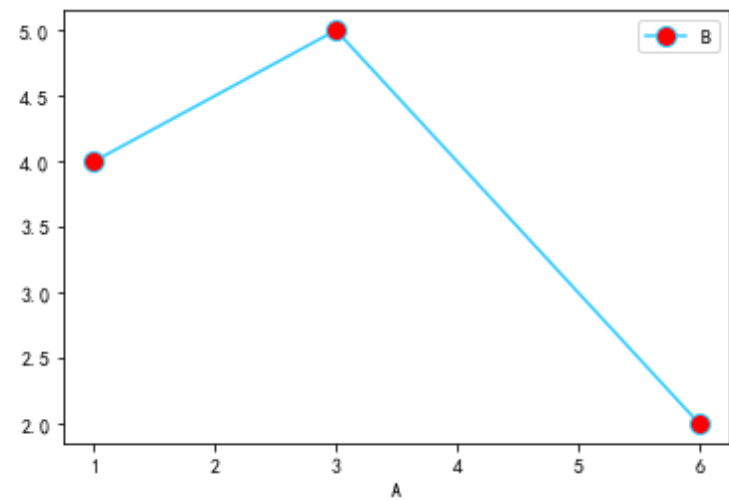


4、显示数据图像

- 折线图

赋值 `item` 为 创建 `df` 的折线图, 颜色是 , 线型是 `实线`
第 `1` 列作为x轴, 第 `2` 列作为y轴
标记是 `圆点`, 标记颜色是 , 标记大小是 `20`

	A	B
0	1	4
1	3	5
2	6	2

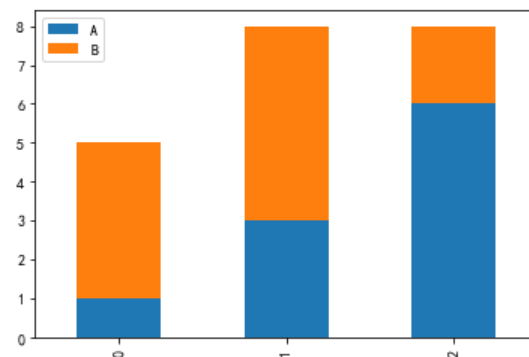


4、显示数据图像

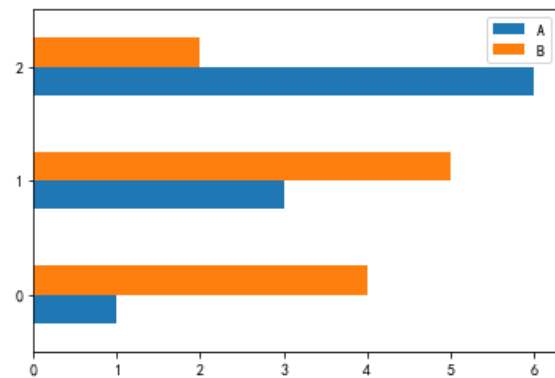
- 柱形图

	A	B
0	1	4
1	3	5
2	6	2

赋值 item 为 创建 df 垂直 的柱形图, 类型 堆积 , 透明度 1




赋值 item 为 创建 df 水平 的柱形图, 类型 不堆积 , 透明度 1



4、显示数据图像

- 保存图像

保存图表为 example .png

 example.png