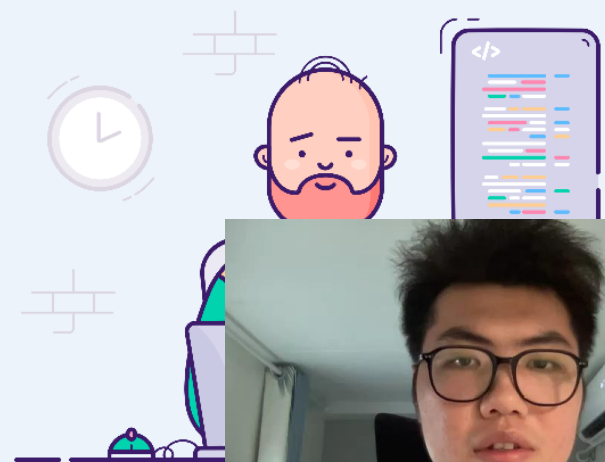


# 第 3 章 Blockly 数据输入、显示、可视化



# 3.1 数据的输入和读取

1. 读取或输入数据
2. 数据表格讲解
3. 数据表格运算

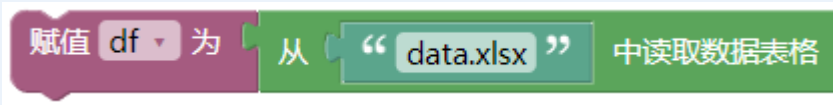


# 1、读取或输入数据 (1/2)

- 首先导入依赖库



- 读取Excel文件:



- 从词典中直接输入数据:



- 最终结构:



```
df = None

import pandas as pd
import matplotlib.pyplot as p
import statistics
import math
plt.rcParams["axes.unicode_m
plt.rcParams["font.sans-serif

df = pd.DataFrame({'A': [1, 2,
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

The screenshot shows a drag-and-drop programming environment with a sidebar on the left and a workspace on the right. The sidebar lists various categories: 逻辑 (Logic), 循环 (Loops), 运算 (Operations), 文本 (Text), 列表 (Lists), 颜色 (Colors), 字典和元组 (Dictionaries and Tuples), 数据表格 (Data Tables), 数据表格运算 (Data Table Operations), 数据预处理 (Data Preprocessing), 可视化 (Visualization), 输出 (Output), 公式 (Formulas), 变量 (Variables), and 函数 (Functions). The '数据表格' (Data Tables) category is selected. The workspace contains several blocks: '导入依赖库, 系统: Windows' (Import dependencies, system: Windows), '从 “ ” 中读取数据表格' (Read data table from " "), '创建数据表格, 数据是' (Create data table, data is) followed by '创建一个词典, 内容:' (Create a dictionary, content:), '设置 item 的列名为' (Set column name of item to), '设置 item 的索引为' (Set index of item to), '获取 item 的第 1 列数据' (Get column 1 data of item), '获取 item 的 “ ” 列数据' (Get column " " data of item), '获取 item 的第 1 行数据' (Get row 1 data of item), and '在 item 设置或添加 “ ”' (Set or add " " to item).



## 2、数据表格讲解

### 重新设置列名

The screenshot shows a Jupyter Notebook cell with the following steps:

- Import pandas as pd.
- Create a DataFrame with two columns: 'A' (values 1, 2, 3) and 'B' (values 4, 5, 6).
- Output the DataFrame 'df'.
- Set the column names of 'df' to 'A1' and 'B1'.
- Output the DataFrame 'df' again.

- 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

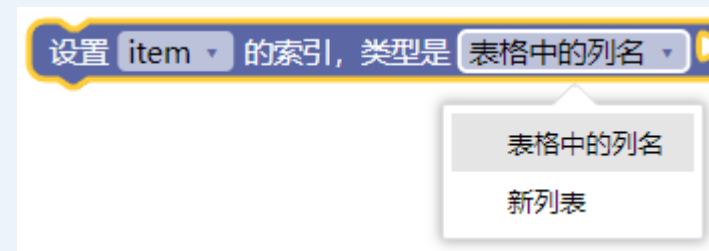
   A1 B1
0  1  4
1  2  5
2  3  6
```



## 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行作为索引，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



## 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

```
在 df 设置或添加 “C” 列, 数据为 列表: 7 , 8 , 9  
输出 df  
在 df 设置或添加 “D” 列, 数据为 列表: “D1” , “D2” , “D3”  
输出 df
```

	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）

```
导入依赖库, 系统: Windows
赋值 df 为 从 "data/ch3-nan.xlsx" 中读取数据表格
输出 df
```

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 `fillna` 方法可以实现缺失值处理

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
```

方案有:

	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



# 3、数据表格运算

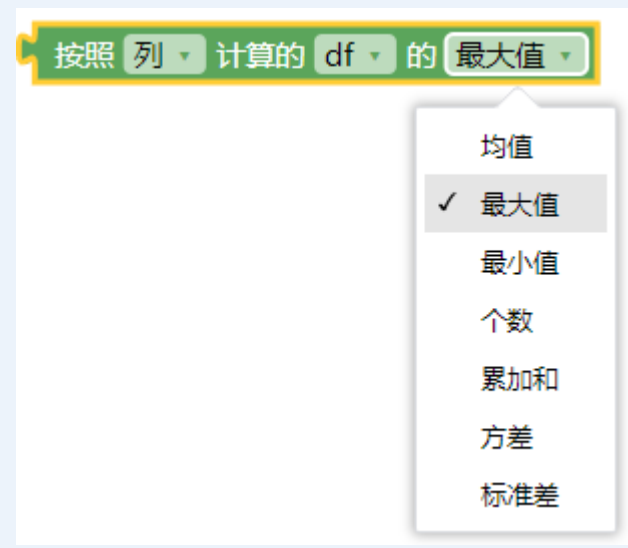
- 基础运算



	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

