Strings

Python Reference

Indexing

A string 字符串 is text inside quotes. Each character 字符 has a position, called its index 索引. The first index is `0`, not `1`.

Read one character with square brackets:

```
word = "Python"
print(word[0])    # P  (the first character)
print(word[2])    # t
print(len(word))  # 6  (how many characters)
```

- Counting starts at `0`, so the last index is `len(word) - 1`.
- A negative index counts back from the end: `word[-1]` is the last character.

```
word = "Python"
print(word[-1])   # n
print(word[-2])   # o
```

- An index that is too large gives an error 错误 (an `IndexError`).

Slicing

A slice 切片 takes a part of a string. Write `word[start:end]`. The slice keeps `start` but stops **before** `end`.

```
word = "Python"
print(word[0:3])   # Pyt   (positions 0, 1, 2)
print(word[2:5])   # tho
```

- Leave out `start` to begin at `0`; leave out `end` to go to the end.

```
word = "Python"
print(word[:3])    # Pyt
print(word[3:])    # hon
```

- A third number is the step 步长. `word[::-1]` reverses 反转 the string.

```
print("Python"[::-1])   # nohtyP
```

String methods & length

A method 方法 is a function that belongs to a value. You call it with a dot:

```
name = "mei chen"
print(name.upper())     # MEI CHEN
print(name.title())     # Mei Chen
print(len(name))        # 8
```

Strings are immutable 不可变: a method returns a **new** string and never changes the original 原始 one.

```
name = "mei"
print(name.upper())     # MEI  (the returned value)
print(name)             # mei  (the original is unchanged)
```

Common methods (each returns a new value):

Method	Meaning	Example	Result
`.upper()` / `.lower()`	change case	`"Hi".lower()`	`hi`
`.strip()`	remove edge spaces	`" hi ".strip()`	`hi`
`.replace(a, b)`	swap text	`"cat".replace("c", "b")`	`bat`
`.split(sep)`	break into a list	`"a,b".split(",")`	`['a', 'b']`

Join strings with `+`. This is called concatenation 拼接:

```
first = "Mei"
last = "Chen"
print(first + " " + last)   # Mei Chen
```

f-strings

An f-string 格式化字符串 builds text from values. Put `f` before the quote, then write `{...}` around a value.

```
name = "Mei"
age = 17
print(f"{name} is {age} years old")   # Mei is 17 years old
```

- Any expression 表达式 can go inside the braces.
- `{value:.2f}` rounds to 2 decimal places 小数位.

```
price = 9.5
print(f"Two cost {price * 2}")        # Two cost 19.0
print(f"Pi is about {3.14159:.2f}")   # Pi is about 3.14
```