

pandas.DataFrame.pipe

DataFrame.pipe(func, *args, **kwargs) [\[source\]](#)

Apply chainable functions that expect Series or DataFrames.

Parameters: **func** : *function*

Function to apply to the Series/DataFrame. `args`, and `kwargs` are passed into `func`. Alternatively a `(callable, data_keyword)` tuple where `data_keyword` is a string indicating the keyword of `callable` that expects the Series/DataFrame.

***args** : *iterable, optional*

Positional arguments passed into `func`.

****kwargs** : *mapping, optional*

A dictionary of keyword arguments passed into `func`.

Returns: the return type of `func`.

i

See also

`DataFrame.apply`

Apply a function along input axis of DataFrame.

`DataFrame.applymap`

Apply a function elementwise on a whole DataFrame.

`Series.map`

Apply a mapping correspondence on a `Series`.

Notes

Use `.pipe` when chaining together functions that expect Series, DataFrames or GroupBy objects.

Examples

Constructing a income DataFrame from a dictionary.

```
>>> data = [[8000, 1000], [9500, np.nan], [5000, 2000]]
>>> df = pd.DataFrame(data, columns=['Salary', 'Others'])
>>> df
   Salary  Others
0    8000   1000.0
1    9500     NaN
2    5000   2000.0
```

Functions that perform tax reductions on an income DataFrame.

```
>>> def subtract_federal_tax(df):
...     return df * 0.9
>>> def subtract_state_tax(df, rate):
...     return df * (1 - rate)
>>> def subtract_national_insurance(df, rate, rate_increase):
...     new_rate = rate + rate_increase
...     return df * (1 - new_rate)
```

Instead of writing

```
>>> (subtract_national_insurance(
...     subtract_state_tax(subtract_federal_tax(df), rate=0.12),
...     rate=0.05,
...     rate_increase=0.02))
```

You can write

```
>>> (df.pipe(subtract_federal_tax)
...     .pipe(subtract_state_tax, rate=0.12)
...     .pipe(subtract_national_insurance, rate=0.05, rate_increase=0.02))
   Salary  Others
0  5892.48   736.56
1  6997.32     NaN
2  3682.80  1473.12
```

If you have a function that takes the data as (say) the second argument, pass a tuple indicating which keyword expects the data. For example, suppose `national_insurance` takes its data as `df` in the second argument:

>>>

```
def subtract_national_insurance(rate, df, rate_increase):
    new_rate = rate + rate_increase
    return df * (1 - new_rate)
(df.pipe(subtract_federal_tax)
 .pipe(subtract_state_tax, rate=0.12)
 .pipe((subtract_national_insurance, 'df'),
       rate=0.05, rate_increase=0.02))
```

>>>

	Salary	Others
0	5892.48	736.56
1	6997.32	NaN
2	3682.80	1473.12

<

Previous

[pandas documentation](#)