Search the docs ...

Input/output

General functions

pandas.melt

pandas.pivot

pandas.pivot_table

pandas.crosstab

pandas.cut

pandas.qcut

pandas.merge

pandas.merge_ordered

pandas.merge_asof

pandas.concat

pandas.get_dummies

pandas.factorize

pandas.unique

pandas.wide_to_long

pandas.isna

pandas.isnull

pandas.notna

pandas.notnull

pandas.to_numeric

pandas.to_datetime

pandas.to_timedelta

pandas.date_range

pandas.bdate_range

pandas.period_range

pandas.notnull

`pandas.notnull(obj)` [source]

Detect non-missing values for an array-like object.

This function takes a scalar or array-like object and indicates whether values are valid (not missing, which is `NaN` in numeric arrays, `None` or `NaN` in object arrays, `NaT` in datetimelike).

Parameters: **obj** : ***array-like or object value***

Object to check for *not* null or *non*-missing values.

Returns: **bool or array-like of bool**

For scalar input, returns a scalar boolean. For array input, returns an array of boolean indicating whether each corresponding element is valid.

See also

`isna`

Boolean inverse of pandas.notna.

`Series.notna`

Detect valid values in a Series.

`DataFrame.notna`

Detect valid values in a DataFrame.

`Index.notna`

Detect valid values in an Index.

Examples

Scalar arguments (including strings) result in a scalar boolean.

```
>>> pd.notna('dog')
True
```

```
>>> pd.notna(pd.NA)
False
```

```
>>> pd.notna(np.nan)
False
```

ndarrays result in an ndarray of booleans.

```
>>> array = np.array([[1, np.nan, 3], [4, 5, np.nan]])
>>> array
array([[ 1., nan,  3.],
       [ 4.,  5., nan]])
>>> pd.notna(array)
array([[ True, False,  True],
       [ True,  True, False]])
```

For indexes, an ndarray of booleans is returned.

```
>>> index = pd.DatetimeIndex(["2017-07-05", "2017-07-06", None,
...                           "2017-07-08"])
>>> index
DatetimeIndex(['2017-07-05', '2017-07-06', 'NaT', '2017-07-08'],
              dtype='datetime64[ns]', freq=None)
>>> pd.notna(index)
array([ True,  True, False,  True])
```

For Series and DataFrame, the same type is returned, containing booleans.

```
>>> df = pd.DataFrame([['ant', 'bee', 'cat'], ['dog', None, 'fly']])
>>> df
     0     1    2
0  ant   bee  cat
1  dog  None  fly
>>> pd.notna(df)
      0      1     2
0  True   True  True
1  True  False  True
```

```
>>> pd.notna(df[1])
0     True
1    False
Name: 1, dtype: bool
```

Previous
pandas.notna

Next
pandas.to_numeric

Created using Sphinx 4.4.0.