




What's new in Pandas

Tom Augspurger / Software Engineer / Anaconda

Project highlights from the last year(ish)

- Community health
- Logo, website, and documentation
- Pandas + Numba = 
- Extension Array Interface Update
- Missing Data

Roughly covers pandas 0.25.x, 1.0.x, and master

Community and Project Health

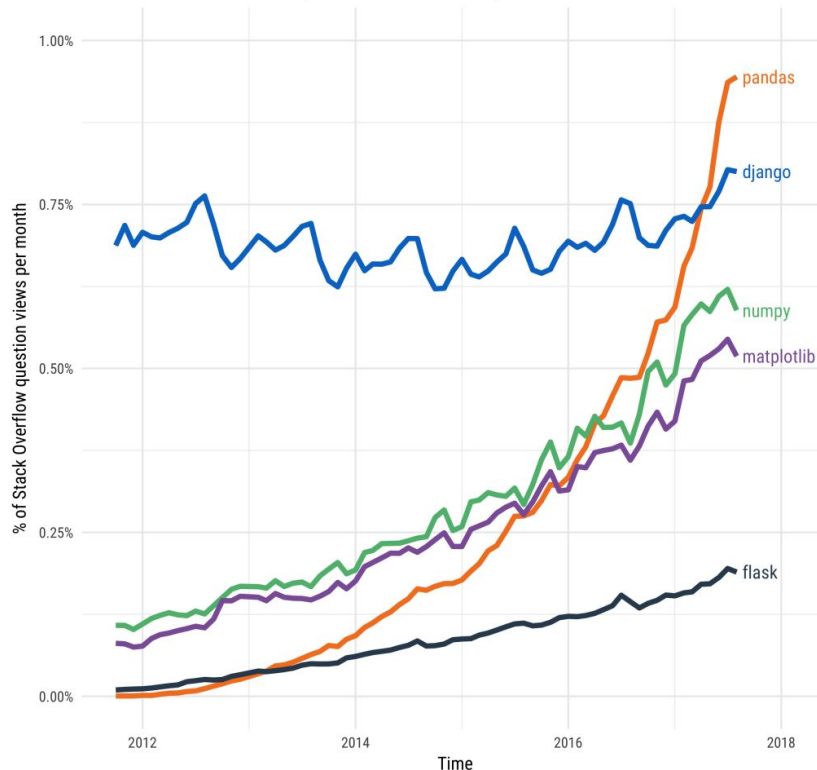


Project & Community Health

- Pandas is an important project in the ecosystem

Stack Overflow Traffic to Questions About Selected Python Packages

Based on visits to Stack Overflow questions from World Bank high-income countries



[Why is Python Growing so Quickly? Stackoverflow, 2017](#)

Project & Community Health

- \$200,000 grant from CZI Open Source Initiative
 - Maintenance
 - Native StringArray implementation
 - Improvements to the extension array interface
- NumFOCUS Small Development Grant
 - New [Getting Started Guide](#)
 - Benchmarking

Logo, Website & Documentation Theme

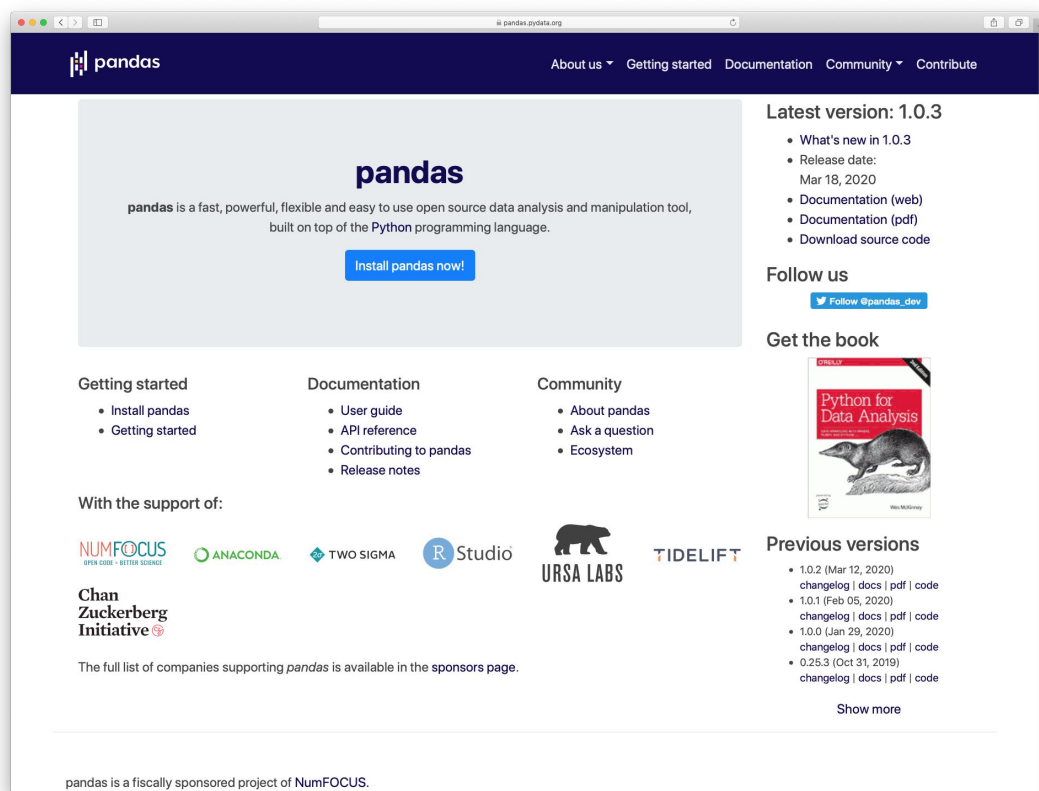


pandas

New Logo, sponsored by Indeed

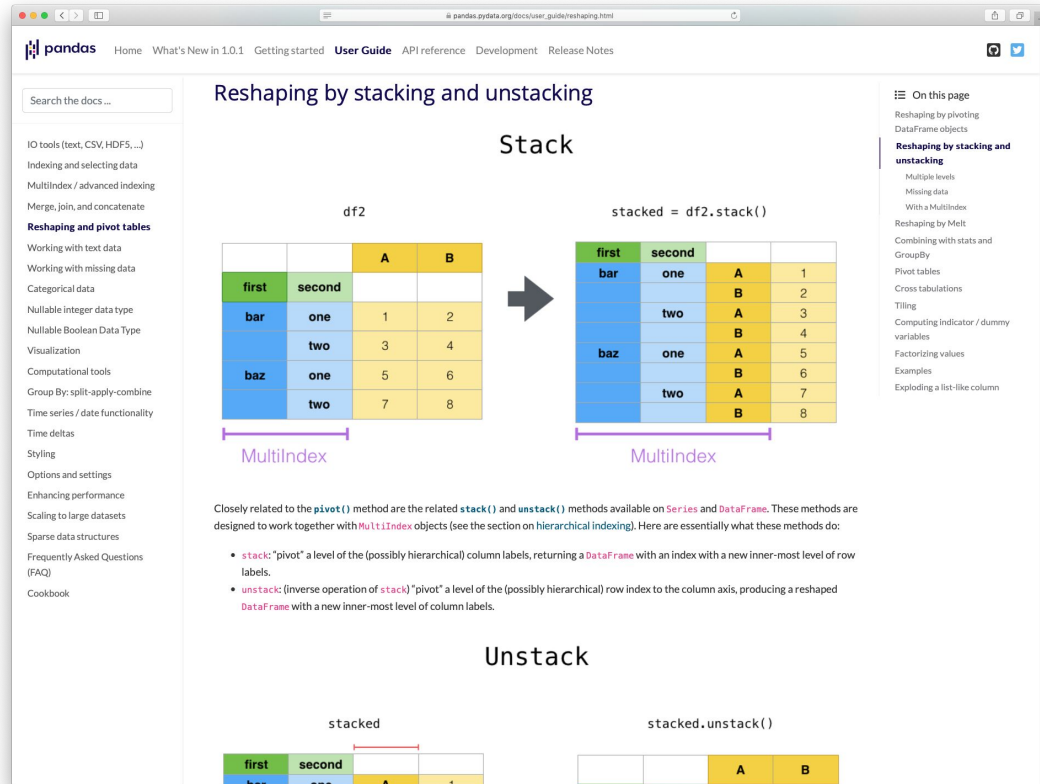
Logo, Website & Documentation Theme

- New website, by Marc Garcia & others.



Logo, Website & Documentation Theme

- New Sphinx theme, by Joris van den Bossche & others



The screenshot shows the pandas documentation page for 'Reshaping by stacking and unstacking'. The page features a sidebar with navigation links, a main content area with diagrams and text, and a right-hand 'On this page' section.

Stack

The 'Stack' section illustrates the transformation of a DataFrame with a MultiIndex into a single-level DataFrame. The initial DataFrame, `df2`, has a MultiIndex on the columns with levels 'first' and 'second'. The `stacked = df2.stack()` operation stacks these levels into a single column index, resulting in a DataFrame where the columns are 'first' and 'second', and the values are stacked vertically.

first	second	A	B
bar	one	1	2
baz	one	5	6
bar	two	3	4
baz	two	7	8

MultiIndex

`stacked = df2.stack()`

first	second	A	B
bar	one	A	1
bar	two	B	2
baz	one	A	3
baz	two	B	4
baz	one	A	5
baz	two	B	6
baz	one	A	7
baz	two	B	8

MultiIndex

Closely related to the `pivot()` method are the related `stack()` and `unstack()` methods available on `Series` and `DataFrame`. These methods are designed to work together with `MultiIndex` objects (see the section on [hierarchical indexing](#)). Here are essentially what these methods do:

- `stack`: "pivot" a level of the (possibly hierarchical) column labels, returning a `DataFrame` with an index with a new inner-most level of row labels.
- `unstack`: (inverse operation of `stack`) "pivot" a level of the (possibly hierarchical) row index to the column axis, producing a reshaped `DataFrame` with a new inner-most level of column labels.

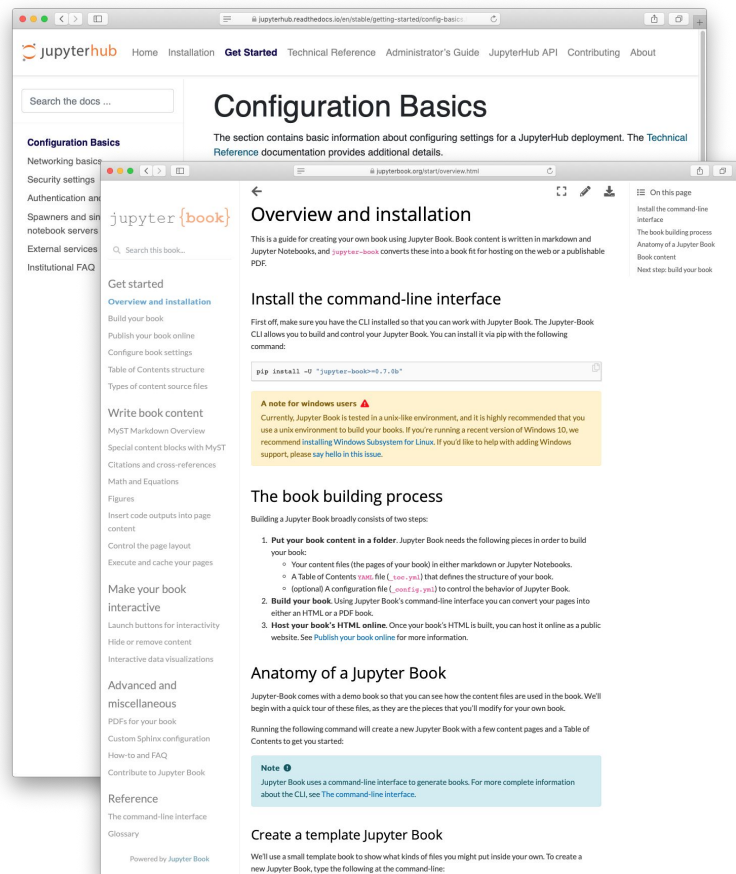
Unstack

The 'Unstack' section shows the reverse operation. The `stacked` DataFrame is transformed back into its original shape using `stacked.unstack()`, restoring the MultiIndex on the columns.

first	second	A	B
bar	one	A	1

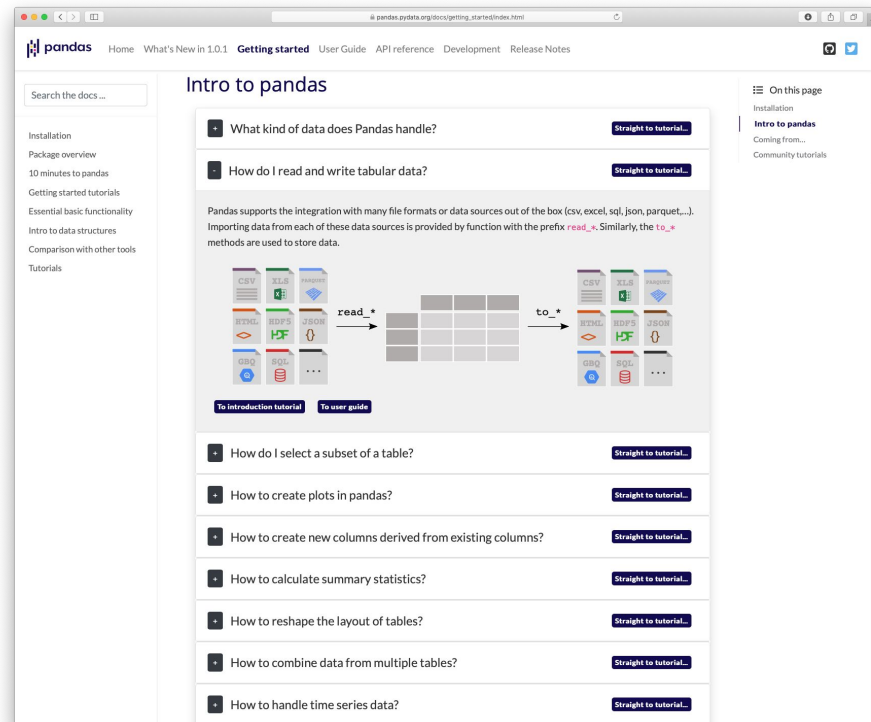
Logo, Website & Documentation Theme

- pydata-sphinx-theme usable by other projects.



New Getting Started Guide

- Funding by NumFOCUS Small Development Grant



New Features



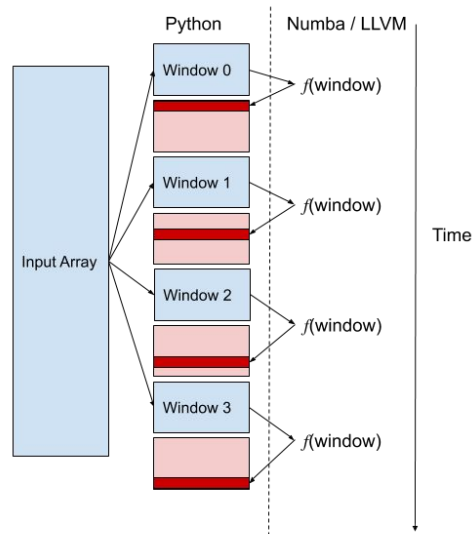
Pandas + Numba

- pandas 1.0 Added support for numba-jitted `.rolling().apply`
- Expanding to GroupBy `.transform()`, `.aggregate()`, and `.apply()`

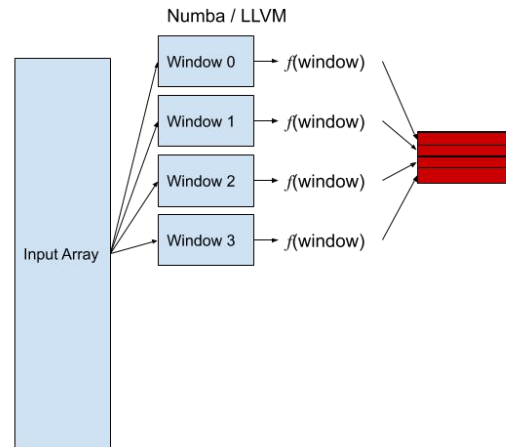
Pandas + Numba Demo



Not just a
`numba.jit(udf)`



Not just a
`numba.jit(udf)`



Extension Arrays

- Added nullable integer, boolean, and string types
- Improves missing data handling with these dtypes

Extension Arrays Demo



Missing Data

- New `pandas.NA` scalar to indicate missing data
- Used by `IntegerArray`, `BooleanArray`, `StringArray`
- Denotes *missing* data, distinct from *bad* data.

```

In [2]: pd.NA == 0
Out[2]: <NA>

In [3]: pd.NA + 1
Out[3]: <NA>

In [4]: pd.NA | False
Out[4]: <NA>

In [5]: pd.NA | True
Out[5]: True

In [6]:

```

Missing Data: What is NA?

- Typically, “missing”, “unknown”
 - $NA \neq NaN$
 - “Unknown” \neq “known to be bad”
 - $0 / 0$ is known to be bad
 - $NA / 0$ is unknown
- Key Behaviors
 - Skipped in reductions
 - Kleene Logic in boolean
 - Propagate NA in arithmetics, comparisons
 - False in indexing (*à la* SQL)

Missing Data: The Future

- New “nullable” types use `pd.NA`
- Adding `FloatArray`, possibly `TimestampArray`, to use it.
- Use `DataFrame.convert_dtypes` to opt into new types
- Adding `use_nullable_dtypes=True` flag to `read_csv`, etc.
- Adding a config option `pandas.options.use_nullable_dtypes` to opt in globally



Thanks!

Tom Augspurger / Software Engineer / Anaconda