(continued from previous page)

```
/opt/conda/envs/pandas/lib/python3.7/site-packages/numpy/core/_asarray.py in␣
↪asarray(a, dtype, order)
     83
     84     """
---> 85     return array(a, dtype, copy=False, order=order)
     86
     87

/pandas-release/pandas/pandas/core/arrays/masked.py in __array__(self, dtype)
    143         We return an object array here to preserve our scalar values
    144         """
--> 145         return self.to_numpy(dtype=dtype)
    146
    147     def __arrow_array__(self, type=None):

/pandas-release/pandas/pandas/core/arrays/masked.py in to_numpy(self, dtype, copy, na_
↪value)
    125             ):
    126                 raise ValueError(
--> 127                     f"cannot convert to '{dtype}'-dtype NumPy array "
    128                     "with missing values. Specify an appropriate 'na_value' "
    129                     "for this dtype."

ValueError: cannot convert to 'float64'-dtype NumPy array with missing values.␣
↪Specify an appropriate 'na_value' for this dtype.
```

Use `arrays.IntegerArray.to_numpy()` with an explicit `na_value` instead.

```
In [42]: a.to_numpy(dtype="float", na_value=np.nan)
Out[42]: array([ 1.,  2., nan])
```

Reductions can return ``pd.NA``

When performing a reduction such as a sum with `skipna=False`, the result will now be `pd.NA` instead of `np.nan` in presence of missing values (GH30958).

pandas 0.25.x

```
>>> pd.Series(a).sum(skipna=False)
nan
```

pandas 1.0.0

```
In [43]: pd.Series(a).sum(skipna=False)
Out[43]: <NA>
```

value_counts returns a nullable integer dtype

`Series.value_counts()` with a nullable integer dtype now returns a nullable integer dtype for the values.

pandas 0.25.x

```
>>> pd.Series([2, 1, 1, None], dtype="Int64").value_counts().dtype
dtype('int64')
```

pandas 1.0.0

```
In [44]: pd.Series([2, 1, 1, None], dtype="Int64").value_counts().dtype
Out[44]: Int64Dtype()
```

See *Experimental NA scalar to denote missing values* for more on the differences between `pandas.NA` and `numpy.nan`.

`arrays.IntegerArray` comparisons return `arrays.BooleanArray`

Comparison operations on a *`arrays.IntegerArray`* now returns a *`arrays.BooleanArray`* rather than a NumPy array (GH29964).

pandas 0.25.x

```
>>> a = pd.array([1, 2, None], dtype="Int64")
>>> a
<IntegerArray>
[1, 2, NaN]
Length: 3, dtype: Int64

>>> a > 1
array([False,  True, False])
```

pandas 1.0.0

```
In [45]: a = pd.array([1, 2, None], dtype="Int64")

In [46]: a > 1
Out[46]:
<BooleanArray>
[False, True, <NA>]
Length: 3, dtype: boolean
```

Note that missing values now propagate, rather than always comparing unequal like `numpy.nan`. See *Experimental NA scalar to denote missing values* for more.

By default `Categorical.min()` now returns the minimum instead of np.nan

When *`Categorical`* contains `np.nan`, `Categorical.min()` no longer return `np.nan` by default (skipna=True) (GH25303)

pandas 0.25.x

```
In [1]: pd.Categorical([1, 2, np.nan], ordered=True).min()
Out[1]: nan
```

pandas 1.0.0

```
In [47]: pd.Categorical([1, 2, np.nan], ordered=True).min()
Out[47]: 1
```

Default dtype of empty `pandas.Series`

Initialising an empty *pandas.Series* without specifying a dtype will raise a *DeprecationWarning* now (GH17261). The default dtype will change from `float64` to `object` in future releases so that it is consistent with the behaviour of *DataFrame* and *Index*.

pandas 1.0.0

```
In [1]: pd.Series()
Out[2]:
DeprecationWarning: The default dtype for empty Series will be 'object' instead of
↪'float64' in a future version. Specify a dtype explicitly to silence this warning.
Series([], dtype: float64)
```

Result dtype inference changes for resample operations

The rules for the result dtype in *DataFrame.resample()* aggregations have changed for extension types (GH31359). Previously, pandas would attempt to convert the result back to the original dtype, falling back to the usual inference rules if that was not possible. Now, pandas will only return a result of the original dtype if the scalar values in the result are instances of the extension dtype's scalar type.

```
In [48]: df = pd.DataFrame({"A": ['a', 'b']}, dtype='category',
   ....:                   index=pd.date_range('2000', periods=2))
   ....:

In [49]: df
Out[49]:
             A
2000-01-01   a
2000-01-02   b

[2 rows x 1 columns]
```

pandas 0.25.x

```
>>> df.resample("2D").agg(lambda x: 'a').A.dtype
CategoricalDtype(categories=['a', 'b'], ordered=False)
```

pandas 1.0.0

```
In [50]: df.resample("2D").agg(lambda x: 'a').A.dtype
Out[50]: dtype('O')
```

This fixes an inconsistency between `resample` and `groupby`. This also fixes a potential bug, where the **values** of the result might change depending on how the results are cast back to the original dtype.

pandas 0.25.x

```
>>> df.resample("2D").agg(lambda x: 'c')

     A
0  NaN
```

pandas 1.0.0

```
In [51]: df.resample("2D").agg(lambda x: 'c')
Out[51]:
              A
2000-01-01   c

[1 rows x 1 columns]
```

Increased minimum version for Python

Pandas 1.0.0 supports Python 3.6.1 and higher (GH29212).

Increased minimum versions for dependencies

Some minimum supported versions of dependencies were updated (GH29766, GH29723). If installed, we now require:

Package	Minimum Version	Required	Changed
numpy	1.13.3	X	
pytz	2015.4	X	
python-dateutil	2.6.1	X	
bottleneck	1.2.1		
numexpr	2.6.2		
pytest (dev)	4.0.2		

For optional libraries the general recommendation is to use the latest version. The following table lists the lowest version per library that is currently being tested throughout the development of pandas. Optional libraries below the lowest tested version may still work, but are not considered supported.

Package	Minimum Version	Changed
beautifulsoup4	4.6.0	
fastparquet	0.3.2	X
gcsfs	0.2.2	
lxml	3.8.0	
matplotlib	2.2.2	
numba	0.46.0	X
openpyxl	2.5.7	X
pyarrow	0.13.0	X
pymysql	0.7.1	
pytables	3.4.2	
s3fs	0.3.0	X
scipy	0.19.0	
sqlalchemy	1.1.4	
xarray	0.8.2	
xlrd	1.1.0	
xlsxwriter	0.9.8	
xlwt	1.2.0	

See *Dependencies* and *Optional dependencies* for more.

Build Changes

Pandas has added a pyproject.toml file and will no longer include cythonized files in the source distribution uploaded to PyPI (GH28341, GH20775). If you're installing a built distribution (wheel) or via conda, this shouldn't have any effect on you. If you're building pandas from source, you should no longer need to install Cython into your build environment before calling `pip install pandas`.

Other API changes

- *core.groupby.GroupBy.transform* now raises on invalid operation names (GH27489)
- *pandas.api.types.infer_dtype()* will now return "integer-na" for integer and `np.nan` mix (GH27283)
- *MultiIndex.from_arrays()* will no longer infer names from arrays if `names=None` is explicitly provided (GH27292)
- In order to improve tab-completion, Pandas does not include most deprecated attributes when introspecting a pandas object using `dir` (e.g. `dir(df)`). To see which attributes are excluded, see an object's `_deprecations` attribute, for example `pd.DataFrame._deprecations` (GH28805).
- The returned dtype of *unique()* now matches the input dtype. (GH27874)
- Changed the default configuration value for `options.matplotlib.register_converters` from `True` to `"auto"` (GH18720). Now, pandas custom formatters will only be applied to plots created by pandas, through *plot()*. Previously, pandas' formatters would be applied to all plots created *after* a *plot()*. See *units registration* for more.
- *Series.dropna()* has dropped its `**kwargs` argument in favor of a single `how` parameter. Supplying anything else than `how` to `**kwargs` raised a `TypeError` previously (GH29388)
- When testing pandas, the new minimum required version of pytest is 5.0.1 (GH29664)
- `Series.str.__iter__()` was deprecated and will be removed in future releases (GH28277).
- Added `<NA>` to the list of default NA values for *read_csv()* (GH30821)

Documentation Improvements

- Added new section on *Scaling to large datasets* (GH28315).
- Added sub-section on *Query MultiIndex* for HDF5 datasets (GH28791).

Deprecations

- *Series.item()* and *Index.item()* have been _undeprecated_ (GH29250)
- `Index.set_value` has been deprecated. For a given index `idx`, array `arr`, value in `idx` of `idx_val` and a new value of `val`, `idx.set_value(arr, idx_val, val)` is equivalent to `arr[idx.get_loc(idx_val)] = val`, which should be used instead (GH28621).
- `is_extension_type()` is deprecated, `is_extension_array_dtype()` should be used instead (GH29457)
- *eval()* keyword argument "truediv" is deprecated and will be removed in a future version (GH29812)

- `DateOffset.isAnchored()` and `DatetOffset.onOffset()` are deprecated and will be removed in a future version, use `DateOffset.is_anchored()` and `DateOffset.is_on_offset()` instead (GH30340)
- `pandas.tseries.frequencies.get_offset` is deprecated and will be removed in a future version, use `pandas.tseries.frequencies.to_offset` instead (GH4205)
- `Categorical.take_nd()` and `CategoricalIndex.take_nd()` are deprecated, use `Categorical.take()` and `CategoricalIndex.take()` instead (GH27745)
- The parameter `numeric_only` of `Categorical.min()` and `Categorical.max()` is deprecated and replaced with `skipna` (GH25303)
- The parameter `label` in `lreshape()` has been deprecated and will be removed in a future version (GH29742)
- `pandas.core.index` has been deprecated and will be removed in a future version, the public classes are available in the top-level namespace (GH19711)
- *`pandas.json_normalize()`* is now exposed in the top-level namespace. Usage of `json_normalize` as `pandas.io.json.json_normalize` is now deprecated and it is recommended to use `json_normalize` as *`pandas.json_normalize()`* instead (GH27586).
- The `numpy` argument of *`pandas.read_json()`* is deprecated (GH28512).
- *`DataFrame.to_stata()`*, *`DataFrame.to_feather()`*, and *`DataFrame.to_parquet()`* argument "fname" is deprecated, use "path" instead (GH23574)
- The deprecated internal attributes `_start`, `_stop` and `_step` of *`RangeIndex`* now raise a `FutureWarning` instead of a `DeprecationWarning` (GH26581)
- The `pandas.util.testing` module has been deprecated. Use the public API in `pandas.testing` documented at *Testing functions* (GH16232).
- `pandas.SparseArray` has been deprecated. Use `pandas.arrays.SparseArray` (*`arrays.SparseArray`*) instead. (GH30642)
- The parameter `is_copy` of *`Series.take()`* and *`DataFrame.take()`* has been deprecated and will be removed in a future version. (GH27357)
- Support for multi-dimensional indexing (e.g. `index[:, None]`) on a *`Index`* is deprecated and will be removed in a future version, convert to a numpy array before indexing instead (GH30588)
- The `pandas.np` submodule is now deprecated. Import numpy directly instead (GH30296)
- The `pandas.datetime` class is now deprecated. Import from `datetime` instead (GH30610)
- *`diff`* will raise a `TypeError` rather than implicitly losing the dtype of extension types in the future. Convert to the correct dtype before calling `diff` instead (GH31025)

Selecting Columns from a Grouped DataFrame

When selecting columns from a `DataFrameGroupBy` object, passing individual keys (or a tuple of keys) inside single brackets is deprecated, a list of items should be used instead. (GH23566) For example:

```
df = pd.DataFrame({
    "A": ["foo", "bar", "foo", "bar", "foo", "bar", "foo", "foo"],
    "B": np.random.randn(8),
    "C": np.random.randn(8),
})
g = df.groupby('A')

# single key, returns SeriesGroupBy
```

(continues on next page)

(continued from previous page)

```
g['B']

# tuple of single key, returns SeriesGroupBy
g[('B',)]

# tuple of multiple keys, returns DataFrameGroupBy, raises FutureWarning
g[('B', 'C')]

# multiple keys passed directly, returns DataFrameGroupBy, raises FutureWarning
# (implicitly converts the passed strings into a single tuple)
g['B', 'C']

# proper way, returns DataFrameGroupBy
g[['B', 'C']]
```

Removal of prior version deprecations/changes

Removed SparseSeries and SparseDataFrame

`SparseSeries`, `SparseDataFrame` and the `DataFrame.to_sparse` method have been removed (GH28425). We recommend using a `Series` or `DataFrame` with sparse values instead. See *Migrating* for help with migrating existing code.

Matplotlib unit registration

Previously, pandas would register converters with matplotlib as a side effect of importing pandas (GH18720). This changed the output of plots made via matplotlib plots after pandas was imported, even if you were using matplotlib directly rather than *`plot()`*.

To use pandas formatters with a matplotlib plot, specify

```
>>> import pandas as pd
>>> pd.options.plotting.matplotlib.register_converters = True
```

Note that plots created by *`DataFrame.plot()`* and *`Series.plot()`* *do* register the converters automatically. The only behavior change is when plotting a date-like object via `matplotlib.pyplot.plot` or `matplotlib.Axes.plot`. See *Custom formatters for timeseries plots* for more.

Other removals

- Removed the previously deprecated keyword "index" from *`read_stata()`*, `StataReader`, and `StataReader.read()`, use "index_col" instead (GH17328)
- Removed `StataReader.data` method, use `StataReader.read()` instead (GH9493)
- Removed `pandas.plotting._matplotlib.tsplot`, use *`Series.plot()`* instead (GH19980)
- `pandas.tseries.converter.register` has been moved to *`pandas.plotting.register_matplotlib_converters()`* (GH18307)
- *`Series.plot()`* no longer accepts positional arguments, pass keyword arguments instead (GH30003)
- *`DataFrame.hist()`* and *`Series.hist()`* no longer allows `figsize="default"`, specify figure size by passinig a tuple instead (GH30003)
- Floordiv of integer-dtyped array by *`Timedelta`* now raises `TypeError` (GH21036)
- *`TimedeltaIndex`* and *`DatetimeIndex`* no longer accept non-nanosecond dtype strings like "timedelta64" or "datetime64", use "timedelta64[ns]" and "datetime64[ns]" instead (GH24806)

- Changed the default "skipna" argument in *pandas.api.types.infer_dtype()* from `False` to `True` (GH24050)
- Removed `Series.ix` and `DataFrame.ix` (GH26438)
- Removed `Index.summary` (GH18217)
- Removed the previously deprecated keyword "fastpath" from the *Index* constructor (GH23110)
- Removed `Series.get_value`, `Series.set_value`, `DataFrame.get_value`, `DataFrame.set_value` (GH17739)
- Removed `Series.compound` and `DataFrame.compound` (GH26405)
- Changed the default "inplace" argument in *DataFrame.set_index()* and *Series.set_axis()* from `None` to `False` (GH27600)
- Removed `Series.cat.categorical`, `Series.cat.index`, `Series.cat.name` (GH24751)
- Removed the previously deprecated keyword "box" from *to_datetime()* and *to_timedelta()*; in addition these now always returns *DatetimeIndex*, *TimedeltaIndex*, *Index*, *Series*, or *DataFrame* (GH24486)
- *to_timedelta()*, *Timedelta*, and *TimedeltaIndex* no longer allow "M", "y", or "Y" for the "unit" argument (GH23264)
- Removed the previously deprecated keyword "time_rule" from (non-public) `offsets.generate_range`, which has been moved to `core.arrays._ranges.generate_range()` (GH24157)
- *DataFrame.loc()* or *Series.loc()* with listlike indexers and missing labels will no longer reindex (GH17295)
- *DataFrame.to_excel()* and *Series.to_excel()* with non-existent columns will no longer reindex (GH17295)
- Removed the previously deprecated keyword "join_axes" from *concat()*; use `reindex_like` on the result instead (GH22318)
- Removed the previously deprecated keyword "by" from *DataFrame.sort_index()*, use *DataFrame.sort_values()* instead (GH10726)
- Removed support for nested renaming in *DataFrame.aggregate()*, *Series.aggregate()*, `core.groupby.DataFrameGroupBy.aggregate()`, `core.groupby.SeriesGroupBy.aggregate()`, *core.window.rolling.Rolling.aggregate()* (GH18529)
- Passing `datetime64` data to *TimedeltaIndex* or `timedelta64` data to `DatetimeIndex` now raises `TypeError` (GH23539, GH23937)
- Passing `int64` values to *DatetimeIndex* and a timezone now interprets the values as nanosecond timestamps in UTC, not wall times in the given timezone (GH24559)
- A tuple passed to *DataFrame.groupby()* is now exclusively treated as a single key (GH18314)
- Removed `Index.contains`, use `key in index` instead (GH30103)
- Addition and subtraction of `int` or integer-arrays is no longer allowed in *Timestamp*, *DatetimeIndex*, *TimedeltaIndex*, use `obj + n * obj.freq` instead of `obj + n` (GH22535)
- Removed `Series.ptp` (GH21614)
- Removed `Series.from_array` (GH18258)
- Removed `DataFrame.from_items` (GH18458)
- Removed `DataFrame.as_matrix`, `Series.as_matrix` (GH18458)

- Removed `Series.asobject` (GH18477)
- Removed `DataFrame.as_blocks`, `Series.as_blocks`, `DataFrame.blocks`, `Series.blocks` (GH17656)
- *pandas.Series.str.cat()* now defaults to aligning `others`, using `join='left'` (GH27611)
- *pandas.Series.str.cat()* does not accept list-likes *within* list-likes anymore (GH27611)
- *Series.where()* with `Categorical` dtype (or *DataFrame.where()* with `Categorical` column) no longer allows setting new categories (GH24114)
- Removed the previously deprecated keywords "start", "end", and "periods" from the *DatetimeIndex*, *TimedeltaIndex*, and *PeriodIndex* constructors; use *date_range()*, *timedelta_range()*, and *period_range()* instead (GH23919)
- Removed the previously deprecated keyword "verify_integrity" from the *DatetimeIndex* and *TimedeltaIndex* constructors (GH23919)
- Removed the previously deprecated keyword "fastpath" from `pandas.core.internals.blocks.make_block` (GH19265)
- Removed the previously deprecated keyword "dtype" from `Block.make_block_same_class()` (GH19434)
- Removed `ExtensionArray._formatting_values`. Use `ExtensionArray._formatter` instead. (GH23601)
- Removed `MultiIndex.to_hierarchical` (GH21613)
- Removed `MultiIndex.labels`, use *MultiIndex.codes* instead (GH23752)
- Removed the previously deprecated keyword "labels" from the *MultiIndex* constructor, use "codes" instead (GH23752)
- Removed `MultiIndex.set_labels`, use *MultiIndex.set_codes()* instead (GH23752)
- Removed the previously deprecated keyword "labels" from *MultiIndex.set_codes()*, `MultiIndex.copy()`, `MultiIndex.drop()`, use "codes" instead (GH23752)
- Removed support for legacy HDF5 formats (GH29787)
- Passing a dtype alias (e.g. 'datetime64[ns, UTC]') to *DatetimeTZDtype* is no longer allowed, use `DatetimeTZDtype.construct_from_string()` instead (GH23990)
- Removed the previously deprecated keyword "skip_footer" from *read_excel()*; use "skipfooter" instead (GH18836)
- *read_excel()* no longer allows an integer value for the parameter `usecols`, instead pass a list of integers from 0 to `usecols` inclusive (GH23635)
- Removed the previously deprecated keyword "convert_datetime64" from *DataFrame.to_records()* (GH18902)
- Removed `IntervalIndex.from_intervals` in favor of the *IntervalIndex* constructor (GH19263)
- Changed the default "keep_tz" argument in *DatetimeIndex.to_series()* from `None` to `True` (GH23739)
- Removed `api.types.is_period` and `api.types.is_datetimetz` (GH23917)
- Ability to read pickles containing *Categorical* instances created with pre-0.16 version of pandas has been removed (GH27538)
- Removed `pandas.tseries.plotting.tsplot` (GH18627)

- Removed the previously deprecated keywords "reduce" and "broadcast" from *DataFrame.apply()* (GH18577)
- Removed the previously deprecated `assert_raises_regex` function in `pandas._testing` (GH29174)
- Removed the previously deprecated `FrozenNDArray` class in `pandas.core.indexes.frozen` (GH29335)
- Removed the previously deprecated keyword "nthreads" from *read_feather()*, use "use_threads" instead (GH23053)
- Removed `Index.is_lexsorted_for_tuple` (GH29305)
- Removed support for nested renaming in *DataFrame.aggregate()*, *Series.aggregate()*, `core.groupby.DataFrameGroupBy.aggregate()`, `core.groupby.SeriesGroupBy.aggregate()`, *core.window.rolling.Rolling.aggregate()* (GH29608)
- Removed `Series.valid`; use *Series.dropna()* instead (GH18800)
- Removed `DataFrame.is_copy`, `Series.is_copy` (GH18812)
- Removed `DataFrame.get_ftype_counts`, `Series.get_ftype_counts` (GH18243)
- Removed `DataFrame.ftypes`, `Series.ftypes`, `Series.ftype` (GH26744)
- Removed `Index.get_duplicates`, use `idx[idx.duplicated()].unique()` instead (GH20239)
- Removed `Series.clip_upper`, `Series.clip_lower`, `DataFrame.clip_upper`, `DataFrame.clip_lower` (GH24203)
- Removed the ability to alter *DatetimeIndex.freq*, `TimedeltaIndex.freq`, or *PeriodIndex.freq* (GH20772)
- Removed `DatetimeIndex.offset` (GH20730)
- Removed `DatetimeIndex.asobject`, `TimedeltaIndex.asobject`, `PeriodIndex.asobject`, use `astype(object)` instead (GH29801)
- Removed the previously deprecated keyword "order" from *factorize()* (GH19751)
- Removed the previously deprecated keyword "encoding" from *read_stata()* and *DataFrame.to_stata()* (GH21400)
- Changed the default "sort" argument in *concat()* from `None` to `False` (GH20613)
- Removed the previously deprecated keyword "raise_conflict" from *DataFrame.update()*, use "errors" instead (GH23585)
- Removed the previously deprecated keyword "n" from `DatetimeIndex.shift()`, `TimedeltaIndex.shift()`, `PeriodIndex.shift()`, use "periods" instead (GH22458)
- Removed the previously deprecated keywords "how", "fill_method", and "limit" from *DataFrame.resample()* (GH30139)
- Passing an integer to *Series.fillna()* or *DataFrame.fillna()* with `timedelta64[ns]` dtype now raises `TypeError` (GH24694)
- Passing multiple axes to *DataFrame.dropna()* is no longer supported (GH20995)
- Removed `Series.nonzero`, use `to_numpy().nonzero()` instead (GH24048)
- Passing floating dtype `codes` to *Categorical.from_codes()* is no longer supported, pass `codes.astype(np.int64)` instead (GH21775)
- Removed the previously deprecated keyword "pat" from *Series.str.partition()* and *Series.str.rpartition()*, use "sep" instead (GH23767)

- Removed `Series.put` (GH27106)
- Removed `Series.real`, `Series.imag` (GH27106)
- Removed `Series.to_dense`, `DataFrame.to_dense` (GH26684)
- Removed `Index.dtype_str`, use `str(index.dtype)` instead (GH27106)
- `Categorical.ravel()` returns a *Categorical* instead of a `ndarray` (GH27199)
- The 'outer' method on Numpy ufuncs, e.g. `np.subtract.outer` operating on *Series* objects is no longer supported, and will raise `NotImplementedError` (GH27198)
- Removed `Series.get_dtype_counts` and `DataFrame.get_dtype_counts` (GH27145)
- Changed the default "fill_value" argument in `Categorical.take()` from `True` to `False` (GH20841)
- Changed the default value for the *raw* argument in `Series.rolling().apply()`, `DataFrame.rolling().apply()`, `Series.expanding().apply()`, and `DataFrame.expanding().apply()` from `None` to `False` (GH20584)
- Removed deprecated behavior of *Series.argmin()* and *Series.argmax()*, use *Series.idxmin()* and *Series.idxmax()* for the old behavior (GH16955)
- Passing a tz-aware `datetime.datetime` or *Timestamp* into the *Timestamp* constructor with the `tz` argument now raises a `ValueError` (GH23621)
- Removed `Series.base`, `Index.base`, `Categorical.base`, `Series.flags`, `Index.flags`, `PeriodArray.flags`, `Series.strides`, `Index.strides`, `Series.itemsize`, `Index.itemsize`, `Series.data`, `Index.data` (GH20721)
- Changed *Timedelta.resolution()* to match the behavior of the standard library `datetime.timedelta.resolution`, for the old behavior, use *Timedelta.resolution_string()* (GH26839)
- Removed `Timestamp.weekday_name`, `DatetimeIndex.weekday_name`, and `Series.dt.weekday_name` (GH18164)
- Removed the previously deprecated keyword "errors" in *Timestamp.tz_localize()*, *DatetimeIndex.tz_localize()*, and *Series.tz_localize()* (GH22644)
- Changed the default "ordered" argument in *CategoricalDtype* from `None` to `False` (GH26336)
- *Series.set_axis()* and *DataFrame.set_axis()* now require "labels" as the first argument and "axis" as an optional named parameter (GH30089)
- Removed `to_msgpack`, `read_msgpack`, `DataFrame.to_msgpack`, `Series.to_msgpack` (GH27103)
- Removed `Series.compress` (GH21930)
- Removed the previously deprecated keyword "fill_value" from `Categorical.fillna()`, use "value" instead (GH19269)
- Removed the previously deprecated keyword "data" from `andrews_curves()`, use "frame" instead (GH6956)
- Removed the previously deprecated keyword "data" from `parallel_coordinates()`, use "frame" instead (GH6956)
- Removed the previously deprecated keyword "colors" from `parallel_coordinates()`, use "color" instead (GH6956)
- Removed the previously deprecated keywords "verbose" and "private_key" from *read_gbq()* (GH30200)
- Calling `np.array` and `np.asarray` on tz-aware *Series* and *DatetimeIndex* will now return an object array of tz-aware *Timestamp* (GH24596)

-

Performance improvements

- Performance improvement in *DataFrame* arithmetic and comparison operations with scalars (GH24990, GH29853)
- Performance improvement in indexing with a non-unique *IntervalIndex* (GH27489)
- Performance improvement in `MultiIndex.is_monotonic` (GH27495)
- Performance improvement in *cut()* when `bins` is an *IntervalIndex* (GH27668)
- Performance improvement when initializing a *DataFrame* using a `range` (GH30171)
- Performance improvement in *DataFrame.corr()* when `method` is `"spearman"` (GH28139)
- Performance improvement in *DataFrame.replace()* when provided a list of values to replace (GH28099)
- Performance improvement in *DataFrame.select_dtypes()* by using vectorization instead of iterating over a loop (GH28317)
- Performance improvement in `Categorical.searchsorted()` and `CategoricalIndex.searchsorted()` (GH28795)
- Performance improvement when comparing a *Categorical* with a scalar and the scalar is not found in the categories (GH29750)
- Performance improvement when checking if values in a *Categorical* are equal, equal or larger or larger than a given scalar. The improvement is not present if checking if the *Categorical* is less than or less than or equal than the scalar (GH29820)
- Performance improvement in *Index.equals()* and `MultiIndex.equals()` (GH29134)
- Performance improvement in *infer_dtype()* when `skipna` is `True` (GH28814)

Bug fixes

Categorical

- Added test to assert the `fillna()` raises the correct `ValueError` message when the value isn't a value from categories (GH13628)
- Bug in `Categorical.astype()` where `NaN` values were handled incorrectly when casting to int (GH28406)
- *DataFrame.reindex()* with a *CategoricalIndex* would fail when the targets contained duplicates, and wouldn't fail if the source contained duplicates (GH28107)
- Bug in `Categorical.astype()` not allowing for casting to extension dtypes (GH28668)
- Bug where *merge()* was unable to join on categorical and extension dtype columns (GH28668)
- `Categorical.searchsorted()` and `CategoricalIndex.searchsorted()` now work on unordered categoricals also (GH21667)
- Added test to assert roundtripping to parquet with *DataFrame.to_parquet()* or *read_parquet()* will preserve Categorical dtypes for string types (GH27955)
- Changed the error message in `Categorical.remove_categories()` to always show the invalid removals as a set (GH28669)

- Using date accessors on a categorical dtyped *Series* of datetimes was not returning an object of the same type as if one used the `str.()` / `dt.()` on a *Series* of that type. E.g. when accessing *Series.dt.tz_localize()* on a *Categorical* with duplicate entries, the accessor was skipping duplicates (GH27952)
- Bug in *DataFrame.replace()* and *Series.replace()* that would give incorrect results on categorical data (GH26988)
- Bug where calling `Categorical.min()` or `Categorical.max()` on an empty Categorical would raise a numpy exception (GH30227)
- The following methods now also correctly output values for unobserved categories when called through `groupby(..., observed=False)` (GH17605) * `core.groupby.SeriesGroupBy.count()` * `core.groupby.SeriesGroupBy.size()` * *core.groupby.SeriesGroupBy.nunique()* * `core.groupby.SeriesGroupBy.nth()`

Datetimelike

- Bug in `Series.__setitem__()` incorrectly casting `np.timedelta64("NaT")` to `np.datetime64("NaT")` when inserting into a *Series* with datetime64 dtype (GH27311)
- Bug in *Series.dt()* property lookups when the underlying data is read-only (GH27529)
- Bug in `HDFStore.__getitem__` incorrectly reading tz attribute created in Python 2 (GH26443)
- Bug in *to_datetime()* where passing arrays of malformed `str` with errors="coerce" could incorrectly lead to raising `ValueError` (GH28299)
- Bug in *core.groupby.SeriesGroupBy.nunique()* where `NaT` values were interfering with the count of unique values (GH27951)
- Bug in *Timestamp* subtraction when subtracting a *Timestamp* from a `np.datetime64` object incorrectly raising `TypeError` (GH28286)
- Addition and subtraction of integer or integer-dtype arrays with *Timestamp* will now raise `NullFrequencyError` instead of `ValueError` (GH28268)
- Bug in *Series* and *DataFrame* with integer dtype failing to raise `TypeError` when adding or subtracting a `np.datetime64` object (GH28080)
- Bug in *Series.astype()*, *Index.astype()*, and *DataFrame.astype()* failing to handle `NaT` when casting to an integer dtype (GH28492)
- Bug in `Week` with `weekday` incorrectly raising `AttributeError` instead of `TypeError` when adding or subtracting an invalid type (GH28530)
- Bug in *DataFrame* arithmetic operations when operating with a *Series* with dtype *'timedelta64[ns]'* (GH28049)
- Bug in `core.groupby.generic.SeriesGroupBy.apply()` raising `ValueError` when a column in the original DataFrame is a datetime and the column labels are not standard integers (GH28247)
- Bug in `pandas._config.localization.get_locales()` where the `locales -a` encodes the locales list as windows-1252 (GH23638, GH24760, GH27368)
- Bug in *Series.var()* failing to raise `TypeError` when called with `timedelta64[ns]` dtype (GH28289)
- Bug in *DatetimeIndex.strftime()* and *Series.dt.strftime()* where `NaT` was converted to the string `'NaT'` instead of `np.nan` (GH29578)

- Bug in masking datetime-like arrays with a boolean mask of an incorrect length not raising an `IndexError` (GH30308)
- Bug in *`Timestamp.resolution`* being a property instead of a class attribute (GH29910)
- Bug in *`pandas.to_datetime()`* when called with `None` raising `TypeError` instead of returning `NaT` (GH30011)
- Bug in *`pandas.to_datetime()`* failing for *deques* when using `cache=True` (the default) (GH29403)
- Bug in *`Series.item()`* with `datetime64` or `timedelta64` dtype, `DatetimeIndex.item()`, and `TimedeltaIndex.item()` returning an integer instead of a *`Timestamp`* or *`Timedelta`* (GH30175)
- Bug in *`DatetimeIndex`* addition when adding a non-optimized `DateOffset` incorrectly dropping timezone information (GH30336)
- Bug in *`DataFrame.drop()`* where attempting to drop non-existent values from a DatetimeIndex would yield a confusing error message (GH30399)
- Bug in *`DataFrame.append()`* would remove the timezone-awareness of new data (GH30238)
- Bug in *`Series.cummin()`* and *`Series.cummax()`* with timezone-aware dtype incorrectly dropping its timezone (GH15553)
- Bug in `DatetimeArray`, `TimedeltaArray`, and `PeriodArray` where inplace addition and subtraction did not actually operate inplace (GH24115)
- Bug in *`pandas.to_datetime()`* when called with `Series` storing `IntegerArray` raising `TypeError` instead of returning `Series` (GH30050)
- Bug in *`date_range()`* with custom business hours as `freq` and given number of `periods` (GH30593)
- Bug in *`PeriodIndex`* comparisons with incorrectly casting integers to *`Period`* objects, inconsistent with the *`Period`* comparison behavior (GH30722)
- Bug in `DatetimeIndex.insert()` raising a `ValueError` instead of a `TypeError` when trying to insert a timezone-aware *`Timestamp`* into a timezone-naive *`DatetimeIndex`*, or vice-versa (GH30806)

Timedelta

- Bug in subtracting a *`TimedeltaIndex`* or `TimedeltaArray` from a `np.datetime64` object (GH29558)
-
-

Timezones

-
-

Numeric

- Bug in *DataFrame.quantile()* with zero-column *DataFrame* incorrectly raising (GH23925)
- *DataFrame* flex inequality comparisons methods (*DataFrame.lt()*, *DataFrame.le()*, *DataFrame.gt()*, *DataFrame.ge()*) with object-dtype and `complex` entries failing to raise `TypeError` like their *Series* counterparts (GH28079)
- Bug in *DataFrame* logical operations (&, |, ^) not matching *Series* behavior by filling NA values (GH28741)
- Bug in *DataFrame.interpolate()* where specifying axis by name references variable before it is assigned (GH29142)
- Bug in *Series.var()* not computing the right value with a nullable integer dtype series not passing through ddof argument (GH29128)
- Improved error message when using *frac* > 1 and *replace* = False (GH27451)
- Bug in numeric indexes resulted in it being possible to instantiate an *Int64Index*, *UInt64Index*, or *Float64Index* with an invalid dtype (e.g. datetime-like) (GH29539)
- Bug in *UInt64Index* precision loss while constructing from a list with values in the `np.uint64` range (GH29526)
- Bug in `NumericIndex` construction that caused indexing to fail when integers in the `np.uint64` range were used (GH28023)
- Bug in `NumericIndex` construction that caused *UInt64Index* to be casted to *Float64Index* when integers in the `np.uint64` range were used to index a *DataFrame* (GH28279)
- Bug in *Series.interpolate()* when using method=`index` with an unsorted index, would previously return incorrect results. (GH21037)
- Bug in *DataFrame.round()* where a *DataFrame* with a *CategoricalIndex* of *IntervalIndex* columns would incorrectly raise a `TypeError` (GH30063)
- Bug in *Series.pct_change()* and *DataFrame.pct_change()* when there are duplicated indices (GH30463)
- Bug in *DataFrame* cumulative operations (e.g. cumsum, cummax) incorrect casting to object-dtype (GH19296)
- Bug in *diff* losing the dtype for extension types (GH30889)
- Bug in *DataFrame.diff* raising an `IndexError` when one of the columns was a nullable integer dtype (GH30967)

Conversion

-
-

Strings

- Calling *Series.str.isalnum()* (and other "ismethods") on an empty Series would return an object dtype instead of bool (GH29624)
-

Interval

- Bug in *IntervalIndex.get_indexer()* where a *Categorical* or *CategoricalIndex* target would incorrectly raise a TypeError (GH30063)
- Bug in pandas.core.dtypes.cast.infer_dtype_from_scalar where passing pandas_dtype=True did not infer *IntervalDtype* (GH30337)
- Bug in *Series* constructor where constructing a Series from a list of *Interval* objects resulted in object dtype instead of *IntervalDtype* (GH23563)
- Bug in *IntervalDtype* where the kind attribute was incorrectly set as None instead of "O" (GH30568)
- Bug in *IntervalIndex*, *IntervalArray*, and *Series* with interval data where equality comparisons were incorrect (GH24112)

Indexing

- Bug in assignment using a reverse slicer (GH26939)
- Bug in *DataFrame.explode()* would duplicate frame in the presence of duplicates in the index (GH28010)
- Bug in reindexing a *PeriodIndex()* with another type of index that contained a *Period* (GH28323) (GH28337)
- Fix assignment of column via *.loc* with numpy non-ns datetime type (GH27395)
- Bug in Float64Index.astype() where np.inf was not handled properly when casting to an integer dtype (GH28475)
- *Index.union()* could fail when the left contained duplicates (GH28257)
- Bug when indexing with .loc where the index was a *CategoricalIndex* with non-string categories didn't work (GH17569, GH30225)
- *Index.get_indexer_non_unique()* could fail with TypeError in some cases, such as when searching for ints in a string index (GH28257)
- Bug in Float64Index.get_loc() incorrectly raising TypeError instead of KeyError (GH29189)
- Bug in *DataFrame.loc()* with incorrect dtype when setting Categorical value in 1-row DataFrame (GH25495)
- *MultiIndex.get_loc()* can't find missing values when input includes missing values (GH19132)
- Bug in Series.__setitem__() incorrectly assigning values with boolean indexer when the length of new data matches the number of True values and new data is not a Series or an np.array (GH30567)
- Bug in indexing with a *PeriodIndex* incorrectly accepting integers representing years, use e.g. ser.loc["2007"] instead of ser.loc[2007] (GH30763)

Missing

-
-

MultiIndex

- Constructor for *MultiIndex* verifies that the given `sortorder` is compatible with the actual `lexsort_depth` if `verify_integrity` parameter is `True` (the default) (GH28735)
- Series and MultiIndex *.drop* with *MultiIndex* raise exception if labels not in given in level (GH8594)
-

I/O

- *read_csv()* now accepts binary mode file buffers when using the Python csv engine (GH23779)
- Bug in *DataFrame.to_json()* where using a Tuple as a column or index value and using `orient="columns"` or `orient="index"` would produce invalid JSON (GH20500)
- Improve infinity parsing. *read_csv()* now interprets `Infinity`, `+Infinity`, `-Infinity` as floating point values (GH10065)
- Bug in *DataFrame.to_csv()* where values were truncated when the length of `na_rep` was shorter than the text input data. (GH25099)
- Bug in *DataFrame.to_string()* where values were truncated using display options instead of outputting the full content (GH9784)
- Bug in *DataFrame.to_json()* where a datetime column label would not be written out in ISO format with `orient="table"` (GH28130)
- Bug in *DataFrame.to_parquet()* where writing to GCS would fail with *engine='fastparquet'* if the file did not already exist (GH28326)
- Bug in *read_hdf()* closing stores that it didn't open when Exceptions are raised (GH28699)
- Bug in `DataFrame.read_json()` where using `orient="index"` would not maintain the order (GH28557)
- Bug in *DataFrame.to_html()* where the length of the `formatters` argument was not verified (GH28469)
- Bug in `DataFrame.read_excel()` with `engine='ods'` when `sheet_name` argument references a non-existent sheet (GH27676)
- Bug in *pandas.io.formats.style.Styler()* formatting for floating values not displaying decimals correctly (GH13257)
- Bug in *DataFrame.to_html()* when using `formatters=<list>` and `max_cols` together. (GH25955)
- Bug in `Styler.background_gradient()` not able to work with dtype `Int64` (GH28869)
- Bug in *DataFrame.to_clipboard()* which did not work reliably in ipython (GH22707)
- Bug in *read_json()* where default encoding was not set to `utf-8` (GH29565)
- Bug in `PythonParser` where str and bytes were being mixed when dealing with the decimal field (GH29650)

- *read_gbq()* now accepts progress_bar_type to display progress bar while the data downloads. (GH29857)
- Bug in pandas.io.json.json_normalize() where a missing value in the location specified by *record_path* would raise a TypeError (GH30148)
- *read_excel()* now accepts binary data (GH15914)
- Bug in *read_csv()* in which encoding handling was limited to just the string *utf-16* for the C engine (GH24130)

Plotting

- Bug in *Series.plot()* not able to plot boolean values (GH23719)
- Bug in *DataFrame.plot()* not able to plot when no rows (GH27758)
- Bug in *DataFrame.plot()* producing incorrect legend markers when plotting multiple series on the same axis (GH18222)
- Bug in *DataFrame.plot()* when kind='box' and data contains datetime or timedelta data. These types are now automatically dropped (GH22799)
- Bug in *DataFrame.plot.line()* and *DataFrame.plot.area()* produce wrong xlim in x-axis (GH27686, GH25160, GH24784)
- Bug where *DataFrame.boxplot()* would not accept a color parameter like *DataFrame.plot.box()* (GH26214)
- Bug in the xticks argument being ignored for *DataFrame.plot.bar()* (GH14119)
- *set_option()* now validates that the plot backend provided to 'plotting.backend' implements the backend when the option is set, rather than when a plot is created (GH28163)
- *DataFrame.plot()* now allow a backend keyword argument to allow changing between backends in one session (GH28619).
- Bug in color validation incorrectly raising for non-color styles (GH29122).
- Allow *DataFrame.plot.scatter()* to plot objects and datetime type data (GH18755, GH30391)
- Bug in *DataFrame.hist()*, xrot=0 does not work with by and subplots (GH30288).

Groupby/resample/rolling

- Bug in core.groupby.DataFrameGroupBy.apply() only showing output from a single group when function returns an *Index* (GH28652)
- Bug in *DataFrame.groupby()* with multiple groups where an IndexError would be raised if any group contained all NA values (GH20519)
- Bug in *pandas.core.resample.Resampler.size()* and *pandas.core.resample.Resampler.count()* returning wrong dtype when used with an empty *Series* or *DataFrame* (GH28427)
- Bug in *DataFrame.rolling()* not allowing for rolling over datetimes when axis=1 (GH28192)
- Bug in *DataFrame.rolling()* not allowing rolling over multi-index levels (GH15584).
- Bug in *DataFrame.rolling()* not allowing rolling on monotonic decreasing time indexes (GH19248).
- Bug in *DataFrame.groupby()* not offering selection by column name when axis=1 (GH27614)

- Bug in `core.groupby.DataFrameGroupby.agg()` not able to use lambda function with named aggregation (GH27519)
- Bug in *DataFrame.groupby()* losing column name information when grouping by a categorical column (GH28787)
- Remove error raised due to duplicated input functions in named aggregation in *DataFrame.groupby()* and *Series.groupby()*. Previously error will be raised if the same function is applied on the same column and now it is allowed if new assigned names are different. (GH28426)
- *core.groupby.SeriesGroupBy.value_counts()* will be able to handle the case even when the *Grouper* makes empty groups (GH28479)
- Bug in *core.window.rolling.Rolling.quantile()* ignoring `interpolation` keyword argument when used within a groupby (GH28779)
- Bug in *DataFrame.groupby()* where `any`, `all`, `nunique` and transform functions would incorrectly handle duplicate column labels (GH21668)
- Bug in `core.groupby.DataFrameGroupBy.agg()` with timezone-aware datetime64 column incorrectly casting results to the original dtype (GH29641)
- Bug in *DataFrame.groupby()* when using axis=1 and having a single level columns index (GH30208)
- Bug in *DataFrame.groupby()* when using nunique on axis=1 (GH30253)
- Bug in `GroupBy.quantile()` with multiple list-like q value and integer column names (GH30289)
- Bug in `GroupBy.pct_change()` and `core.groupby.SeriesGroupBy.pct_change()` causes `TypeError` when `fill_method` is `None` (GH30463)
- Bug in `Rolling.count()` and `Expanding.count()` argument where `min_periods` was ignored (GH26996)

Reshaping

- Bug in *DataFrame.apply()* that caused incorrect output with empty *DataFrame* (GH28202, GH21959)
- Bug in *DataFrame.stack()* not handling non-unique indexes correctly when creating MultiIndex (GH28301)
- Bug in *pivot_table()* not returning correct type `float` when `margins=True` and `aggfunc='mean'` (GH24893)
- Bug *merge_asof()* could not use `datetime.timedelta` for `tolerance` kwarg (GH28098)
- Bug in *merge()*, did not append suffixes correctly with MultiIndex (GH28518)
- *qcut()* and *cut()* now handle boolean input (GH20303)
- Fix to ensure all int dtypes can be used in *merge_asof()* when using a tolerance value. Previously every non-int64 type would raise an erroneous `MergeError` (GH28870).
- Better error message in *get_dummies()* when *columns* isn't a list-like value (GH28383)
- Bug in *Index.join()* that caused infinite recursion error for mismatched `MultiIndex` name orders. (GH25760, GH28956)
- Bug *Series.pct_change()* where supplying an anchored frequency would throw a `ValueError` (GH28664)
- Bug where *DataFrame.equals()* returned True incorrectly in some cases when two DataFrames had the same columns in different orders (GH28839)

- Bug in `DataFrame.replace()` that caused non-numeric replacer's dtype not respected (GH26632)
- Bug in `melt()` where supplying mixed strings and numeric values for `id_vars` or `value_vars` would incorrectly raise a `ValueError` (GH29718)
- Dtypes are now preserved when transposing a `DataFrame` where each column is the same extension dtype (GH30091)
- Bug in `merge_asof()` merging on a tz-aware `left_index` and `right_on` a tz-aware column (GH29864)
- Improved error message and docstring in `cut()` and `qcut()` when *labels=True* (GH13318)
- Bug in missing *fill_na* parameter to `DataFrame.unstack()` with list of levels (GH30740)

Sparse

- Bug in `SparseDataFrame` arithmetic operations incorrectly casting inputs to float (GH28107)
- Bug in `DataFrame.sparse` returning a `Series` when there was a column named `sparse` rather than the accessor (GH30758)
- Fixed `operator.xor()` with a boolean-dtype `SparseArray`. Now returns a sparse result, rather than object dtype (GH31025)

ExtensionArray

- Bug in `arrays.PandasArray` when setting a scalar string (GH28118, GH28150).
- Bug where nullable integers could not be compared to strings (GH28930)
- Bug where `DataFrame` constructor raised `ValueError` with list-like data and `dtype` specified (GH30280)

Other

- Trying to set the `display.precision`, `display.max_rows` or `display.max_columns` using `set_option()` to anything but a `None` or a positive int will raise a `ValueError` (GH23348)
- Using `DataFrame.replace()` with overlapping keys in a nested dictionary will no longer raise, now matching the behavior of a flat dictionary (GH27660)
- `DataFrame.to_csv()` and `Series.to_csv()` now support dicts as `compression` argument with key `'method'` being the compression method and others as additional compression options when the compression method is `'zip'`. (GH26023)
- Bug in `Series.diff()` where a boolean series would incorrectly raise a `TypeError` (GH17294)
- `Series.append()` will no longer raise a `TypeError` when passed a tuple of `Series` (GH28410)
- Fix corrupted error message when calling `pandas.libs._json.encode()` on a 0d array (GH18878)
- Backtick quoting in `DataFrame.query()` and `DataFrame.eval()` can now also be used to use invalid identifiers like names that start with a digit, are python keywords, or are using single character operators. (GH27017)
- Bug in `pd.core.util.hashing.hash_pandas_object` where arrays containing tuples were incorrectly treated as non-hashable (GH28969)
- Bug in `DataFrame.append()` that raised `IndexError` when appending with empty list (GH28769)

- Fix `AbstractHolidayCalendar` to return correct results for years after 2030 (now goes up to 2200) (GH27790)
- Fixed *`IntegerArray`* returning `inf` rather than `NaN` for operations dividing by `0` (GH27398)
- Fixed `pow` operations for *`IntegerArray`* when the other value is `0` or `1` (GH29997)
- Bug in *`Series.count()`* raises if use_inf_as_na is enabled (GH29478)
- Bug in *`Index`* where a non-hashable name could be set without raising `TypeError` (GH29069)
- Bug in *`DataFrame`* constructor when passing a 2D `ndarray` and an extension dtype (GH12513)
- Bug in *`DataFrame.to_csv()`* when supplied a series with a `dtype="string"` and a `na_rep`, the `na_rep` was being truncated to 2 characters. (GH29975)
- Bug where *`DataFrame.itertuples()`* would incorrectly determine whether or not namedtuples could be used for dataframes of 255 columns (GH28282)
- Handle nested NumPy `object` arrays in *`testing.assert_series_equal()`* for ExtensionArray implementations (GH30841)
- Bug in *`Index`* constructor incorrectly allowing 2-dimensional input arrays (GH13601, GH27125)

Contributors

A total of 308 people contributed patches to this release. People with a "+" by their names contributed a patch for the first time.

- Aaditya Panikath +
- Abdullah İhsan Seçer
- Abhijeet Krishnan +
- Adam J. Stewart
- Adam Klaum +
- Addison Lynch
- Aivengoe +
- Alastair James +
- Albert Villanova del Moral
- Alex Kirko +
- Alfredo Granja +
- Allen Downey
- Alp Arıbal +
- Andreas Buhr +
- Andrew Munch +
- Andy
- Angela Ambroz +
- Aniruddha Bhattacharjee +
- Ankit Dhankhar +
- Antonio Andraues Jr +

- Arda Kosar +
- Asish Mahapatra +
- Austin Hackett +
- Avi Kelman +
- AyowoleT +
- Bas Nijholt +
- Ben Thayer
- Bharat Raghunathan
- Bhavani Ravi
- Bhuvana KA +
- Big Head
- Blake Hawkins +
- Bobae Kim +
- Brett Naul
- Brian Wignall
- Bruno P. Kinoshita +
- Bryant Moscon +
- Cesar H +
- Chris Stadler
- Chris Zimmerman +
- Christopher Whelan
- Clemens Brunner
- Clemens Tolboom +
- Connor Charles +
- Daniel Hähnke +
- Daniel Saxton
- Darin Plutchok +
- Dave Hughes
- David Stansby
- DavidRosen +
- Dean +
- Deepan Das +
- Deepyaman Datta
- DorAmram +
- Dorothy Kabarozi +
- Drew Heenan +

- Eliza Mae Saret +
- Elle +
- Endre Mark Borza +
- Eric Brassell +
- Eric Wong +
- Eunseop Jeong +
- Eyden Villanueva +
- Felix Divo
- ForTimeBeing +
- Francesco Truzzi +
- Gabriel Corona +
- Gabriel Monteiro +
- Galuh Sahid +
- Georgi Baychev +
- Gina
- GiuPassarelli +
- Grigorios Giannakopoulos +
- Guilherme Leite +
- Guilherme Salomé +
- Gyeongjae Choi +
- Harshavardhan Bachina +
- Harutaka Kawamura +
- Hassan Kibirige
- Hielke Walinga
- Hubert
- Hugh Kelley +
- Ian Eaves +
- Ignacio Santolin +
- Igor Filippov +
- Irv Lustig
- Isaac Virshup +
- Ivan Bessarabov +
- JMBurley +
- Jack Bicknell +
- Jacob Buckheit +
- Jan Koch

- Jan Pipek +
- Jan Škoda +
- Jan-Philip Gehrcke
- Jasper J.F. van den Bosch +
- Javad +
- Jeff Reback
- Jeremy Schendel
- Jeroen Kant +
- Jesse Pardue +
- Jethro Cao +
- Jiang Yue
- Jiaxiang +
- Jihyung Moon +
- Jimmy Callin
- Jinyang Zhou +
- Joao Victor Martinelli +
- Joaq Almirante +
- John G Evans +
- John Ward +
- Jonathan Larkin +
- Joris Van den Bossche
- Josh Dimarsky +
- Joshua Smith +
- Josiah Baker +
- Julia Signell +
- Jung Dong Ho +
- Justin Cole +
- Justin Zheng
- Kaiqi Dong
- Karthigeyan +
- Katherine Younglove +
- Katrin Leinweber
- Kee Chong Tan +
- Keith Kraus +
- Kevin Nguyen +
- Kevin Sheppard

- Kisekka David +
- Koushik +
- Kyle Boone +
- Kyle McCahill +
- Laura Collard, PhD +
- LiuSeeker +
- Louis Huynh +
- Lucas Scarlato Astur +
- Luiz Gustavo +
- Luke +
- Luke Shepard +
- MKhalusova +
- Mabel Villalba
- Maciej J +
- Mak Sze Chun
- Manu NALEPA +
- Marc
- Marc Garcia
- Marco Gorelli +
- Marco Neumann +
- Martin Winkel +
- Martina G. Vilas +
- Mateusz +
- Matthew Roeschke
- Matthew Tan +
- Max Bolingbroke
- Max Chen +
- MeeseeksMachine
- Miguel +
- MinGyo Jung +
- Mohamed Amine ZGHAL +
- Mohit Anand +
- MomIsBestFriend +
- Naomi Bonnin +
- Nathan Abel +
- Nico Cernek +