Methods

`rollback`(self, dt)	Roll provided date backward to next offset only if not on offset.
`rollforward`(self, dt)	Roll provided date forward to next offset only if not on offset.

pandas.tseries.offsets.BusinessMonthEnd.rollback

`BusinessMonthEnd.`**`rollback`**(*self*, *dt*)

Roll provided date backward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

pandas.tseries.offsets.BusinessMonthEnd.rollforward

`BusinessMonthEnd.`**`rollforward`**(*self*, *dt*)

Roll provided date forward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

__call__	
apply	
apply_index	
copy	
isAnchored	
is_anchored	
is_on_offset	
onOffset	

Properties

`BusinessMonthEnd.freqstr`
`BusinessMonthEnd.kwds`
`BusinessMonthEnd.name`
`BusinessMonthEnd.nanos`
`BusinessMonthEnd.normalize`
`BusinessMonthEnd.rule_code`

pandas.tseries.offsets.BusinessMonthEnd.freqstr

`BusinessMonthEnd.freqstr`

pandas.tseries.offsets.BusinessMonthEnd.kwds

`property BusinessMonthEnd.kwds`

pandas.tseries.offsets.BusinessMonthEnd.name

`property BusinessMonthEnd.name`

pandas.tseries.offsets.BusinessMonthEnd.nanos

`property BusinessMonthEnd.nanos`

pandas.tseries.offsets.BusinessMonthEnd.normalize

`BusinessMonthEnd.normalize = False`

pandas.tseries.offsets.BusinessMonthEnd.rule_code

`property BusinessMonthEnd.rule_code`

Methods

BusinessMonthEnd.apply(self, other)	
BusinessMonthEnd.apply_index(self, other)	Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.
BusinessMonthEnd.copy(self)	
BusinessMonthEnd.isAnchored(self)	
BusinessMonthEnd.onOffset(self, dt)	
BusinessMonthEnd.is_anchored(self)	
BusinessMonthEnd.is_on_offset(self, dt)	
BusinessMonthEnd.__call__(self, other)	Call self as a function.

pandas.tseries.offsets.BusinessMonthEnd.apply

`BusinessMonthEnd.`**`apply`**`(`*self*, *other*`)`

pandas.tseries.offsets.BusinessMonthEnd.apply_index

`BusinessMonthEnd.`**`apply_index`**`(`*self*, *other*`)`

Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.

Parameters

i [DatetimeIndex]

Returns

y [DatetimeIndex]

pandas.tseries.offsets.BusinessMonthEnd.copy

`BusinessMonthEnd.`**`copy`**`(`*self*`)`

pandas.tseries.offsets.BusinessMonthEnd.isAnchored

`BusinessMonthEnd.`**`isAnchored`**`(`*self*`)`

pandas.tseries.offsets.BusinessMonthEnd.onOffset

`BusinessMonthEnd.`**`onOffset`**`(`*self*, *dt*`)`

pandas.tseries.offsets.BusinessMonthEnd.is_anchored

`BusinessMonthEnd.`**`is_anchored`**`(`*self*`)`

pandas.tseries.offsets.BusinessMonthEnd.is_on_offset

`BusinessMonthEnd.`**`is_on_offset`**`(`*self*, *dt*`)`

pandas.tseries.offsets.BusinessMonthEnd.__call__

`BusinessMonthEnd.`**`__call__`**`(`*self*, *other*`)`

Call self as a function.

3.8.10 BusinessMonthBegin

BusinessMonthBegin([n, normalize])	DateOffset of one business month at beginning.

pandas.tseries.offsets.BusinessMonthBegin

class pandas.tseries.offsets.**BusinessMonthBegin**(*n=1*, *normalize=False*)

DateOffset of one business month at beginning.

Attributes

base	Returns a copy of the calling offset object with n=1 and all other attributes equal.

pandas.tseries.offsets.BusinessMonthBegin.base

property BusinessMonthBegin.**base**

Returns a copy of the calling offset object with n=1 and all other attributes equal.

freqstr	
kwds	
name	
nanos	
rule_code	

Methods

rollback(self, dt)	Roll provided date backward to next offset only if not on offset.
rollforward(self, dt)	Roll provided date forward to next offset only if not on offset.

pandas.tseries.offsets.BusinessMonthBegin.rollback

BusinessMonthBegin.**rollback**(*self*, *dt*)

Roll provided date backward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

pandas.tseries.offsets.BusinessMonthBegin.rollforward

`BusinessMonthBegin.`**`rollforward`**(*self*, *dt*)

Roll provided date forward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

__call__	
apply	
apply_index	
copy	
isAnchored	
is_anchored	
is_on_offset	
onOffset	

Properties

BusinessMonthBegin.freqstr
BusinessMonthBegin.kwds
BusinessMonthBegin.name
BusinessMonthBegin.nanos
BusinessMonthBegin.normalize
BusinessMonthBegin.rule_code

pandas.tseries.offsets.BusinessMonthBegin.freqstr

`BusinessMonthBegin.`**`freqstr`**

pandas.tseries.offsets.BusinessMonthBegin.kwds

`property` `BusinessMonthBegin.`**`kwds`**

pandas.tseries.offsets.BusinessMonthBegin.name

property BusinessMonthBegin.**name**

pandas.tseries.offsets.BusinessMonthBegin.nanos

property BusinessMonthBegin.**nanos**

pandas.tseries.offsets.BusinessMonthBegin.normalize

BusinessMonthBegin.**normalize = False**

pandas.tseries.offsets.BusinessMonthBegin.rule_code

property BusinessMonthBegin.**rule_code**

Methods

BusinessMonthBegin.apply(self, other)	
BusinessMonthBegin.apply_index(self, other)	Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.
BusinessMonthBegin.copy(self)	
BusinessMonthBegin.isAnchored(self)	
BusinessMonthBegin.onOffset(self, dt)	
BusinessMonthBegin.is_anchored(self)	
BusinessMonthBegin.is_on_offset(self, dt)	
BusinessMonthBegin.__call__(self, other)	Call self as a function.

pandas.tseries.offsets.BusinessMonthBegin.apply

BusinessMonthBegin.**apply**(*self*, *other*)

pandas.tseries.offsets.BusinessMonthBegin.apply_index

BusinessMonthBegin.**apply_index**(*self*, *other*)

Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.

Parameters

i [DatetimeIndex]

Returns

y [DatetimeIndex]

pandas.tseries.offsets.BusinessMonthBegin.copy

`BusinessMonthBegin.copy(self)`

pandas.tseries.offsets.BusinessMonthBegin.isAnchored

`BusinessMonthBegin.isAnchored(self)`

pandas.tseries.offsets.BusinessMonthBegin.onOffset

`BusinessMonthBegin.onOffset(self, dt)`

pandas.tseries.offsets.BusinessMonthBegin.is_anchored

`BusinessMonthBegin.is_anchored(self)`

pandas.tseries.offsets.BusinessMonthBegin.is_on_offset

`BusinessMonthBegin.is_on_offset(self, dt)`

pandas.tseries.offsets.BusinessMonthBegin.__call__

`BusinessMonthBegin.__call__(self, other)`
Call self as a function.

3.8.11 CustomBusinessMonthEnd

`CustomBusinessMonthEnd`([n, normalize, …])	DateOffset subclass representing custom business month(s).

pandas.tseries.offsets.CustomBusinessMonthEnd

class `pandas.tseries.offsets.`**`CustomBusinessMonthEnd`**(*n=1, normalize=False, weekmask='Mon Tue Wed Thu Fri', holidays=None, calendar=None, offset=datetime.timedelta(0)*)

DateOffset subclass representing custom business month(s).

Increments between end of month dates.

Parameters

n [int, default 1] The number of months represented.

normalize [bool, default False] Normalize start/end dates to midnight before generating date range.

weekmask [str, Default 'Mon Tue Wed Thu Fri'] Weekmask of valid business days, passed to `numpy.busdaycalendar`.

holidays [list] List/array of dates to exclude from the set of valid business days, passed to `numpy.busdaycalendar`.

calendar [pd.HolidayCalendar or np.busdaycalendar] Calendar to integrate.

offset [timedelta, default timedelta(0)] Time offset to apply.

Attributes

base	Returns a copy of the calling offset object with n=1 and all other attributes equal.
cbday_roll	Define default roll function to be called in apply method.
month_roll	Define default roll function to be called in apply method.
offset	Alias for self._offset.

pandas.tseries.offsets.CustomBusinessMonthEnd.base

property `CustomBusinessMonthEnd.`**base**

Returns a copy of the calling offset object with n=1 and all other attributes equal.

pandas.tseries.offsets.CustomBusinessMonthEnd.cbday_roll

`CustomBusinessMonthEnd.`**cbday_roll**

Define default roll function to be called in apply method.

pandas.tseries.offsets.CustomBusinessMonthEnd.month_roll

`CustomBusinessMonthEnd.`**month_roll**

Define default roll function to be called in apply method.

pandas.tseries.offsets.CustomBusinessMonthEnd.offset

property `CustomBusinessMonthEnd.`**offset**

Alias for self._offset.

freqstr	
kwds	
m_offset	
name	
nanos	
rule_code	

Methods

apply_index(self, other)	Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.
rollback(self, dt)	Roll provided date backward to next offset only if not on offset.
rollforward(self, dt)	Roll provided date forward to next offset only if not on offset.

pandas.tseries.offsets.CustomBusinessMonthEnd.apply_index

`CustomBusinessMonthEnd.`**`apply_index`**`(`*self*, *other*`)`

Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.

Parameters

i [DatetimeIndex]

Returns

y [DatetimeIndex]

pandas.tseries.offsets.CustomBusinessMonthEnd.rollback

`CustomBusinessMonthEnd.`**`rollback`**`(`*self*, *dt*`)`

Roll provided date backward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

pandas.tseries.offsets.CustomBusinessMonthEnd.rollforward

`CustomBusinessMonthEnd.`**`rollforward`**`(`*self*, *dt*`)`

Roll provided date forward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

__call__	
apply	
copy	
isAnchored	
is_anchored	
is_on_offset	
onOffset	

Properties

CustomBusinessMonthEnd.freqstr	
CustomBusinessMonthEnd.kwds	
CustomBusinessMonthEnd.m_offset	
CustomBusinessMonthEnd.name	
CustomBusinessMonthEnd.nanos	
CustomBusinessMonthEnd.normalize	
CustomBusinessMonthEnd.rule_code	

pandas.tseries.offsets.CustomBusinessMonthEnd.freqstr

CustomBusinessMonthEnd.**freqstr**

pandas.tseries.offsets.CustomBusinessMonthEnd.kwds

property CustomBusinessMonthEnd.**kwds**

pandas.tseries.offsets.CustomBusinessMonthEnd.m_offset

CustomBusinessMonthEnd.**m_offset**

pandas.tseries.offsets.CustomBusinessMonthEnd.name

property CustomBusinessMonthEnd.**name**

pandas.tseries.offsets.CustomBusinessMonthEnd.nanos

property CustomBusinessMonthEnd.**nanos**

pandas.tseries.offsets.CustomBusinessMonthEnd.normalize

CustomBusinessMonthEnd.**normalize = False**

pandas.tseries.offsets.CustomBusinessMonthEnd.rule_code

property CustomBusinessMonthEnd.**rule_code**

Methods

CustomBusinessMonthEnd.apply(self, other)	
CustomBusinessMonthEnd.copy(self)	
CustomBusinessMonthEnd.isAnchored(self)	
CustomBusinessMonthEnd.onOffset(self, dt)	
CustomBusinessMonthEnd.is_anchored(self)	
CustomBusinessMonthEnd.is_on_offset(self, dt)	
CustomBusinessMonthEnd.__call__(self, other)	Call self as a function.

pandas.tseries.offsets.CustomBusinessMonthEnd.apply

`CustomBusinessMonthEnd.`**`apply`**(*self*, *other*)

pandas.tseries.offsets.CustomBusinessMonthEnd.copy

`CustomBusinessMonthEnd.`**`copy`**(*self*)

pandas.tseries.offsets.CustomBusinessMonthEnd.isAnchored

`CustomBusinessMonthEnd.`**`isAnchored`**(*self*)

pandas.tseries.offsets.CustomBusinessMonthEnd.onOffset

`CustomBusinessMonthEnd.`**`onOffset`**(*self*, *dt*)

pandas.tseries.offsets.CustomBusinessMonthEnd.is_anchored

`CustomBusinessMonthEnd.`**`is_anchored`**(*self*)

pandas.tseries.offsets.CustomBusinessMonthEnd.is_on_offset

`CustomBusinessMonthEnd.`**`is_on_offset`**(*self*, *dt*)

pandas.tseries.offsets.CustomBusinessMonthEnd.__call__

`CustomBusinessMonthEnd.__call__`(*self*, *other*)
Call self as a function.

3.8.12 CustomBusinessMonthBegin

`CustomBusinessMonthBegin`([n, normalize, ...])	DateOffset subclass representing custom business month(s).

pandas.tseries.offsets.CustomBusinessMonthBegin

class `pandas.tseries.offsets.`**`CustomBusinessMonthBegin`**(*n=1*, *normalize=False*, *weekmask='Mon Tue Wed Thu Fri'*, *holidays=None*, *calendar=None*, *offset=datetime.timedelta(0)*)

DateOffset subclass representing custom business month(s).

Increments between beginning of month dates.

Parameters

n [int, default 1] The number of months represented.

normalize [bool, default False] Normalize start/end dates to midnight before generating date range.

weekmask [str, Default 'Mon Tue Wed Thu Fri'] Weekmask of valid business days, passed to `numpy.busdaycalendar`.

holidays [list] List/array of dates to exclude from the set of valid business days, passed to `numpy.busdaycalendar`.

calendar [pd.HolidayCalendar or np.busdaycalendar] Calendar to integrate.

offset [timedelta, default timedelta(0)] Time offset to apply.

Attributes

`base`	Returns a copy of the calling offset object with n=1 and all other attributes equal.
`cbday_roll`	Define default roll function to be called in apply method.
`month_roll`	Define default roll function to be called in apply method.
`offset`	Alias for self._offset.

pandas.tseries.offsets.CustomBusinessMonthBegin.base

property `CustomBusinessMonthBegin.`**`base`**

Returns a copy of the calling offset object with n=1 and all other attributes equal.

pandas.tseries.offsets.CustomBusinessMonthBegin.cbday_roll

`CustomBusinessMonthBegin.`**`cbday_roll`**

Define default roll function to be called in apply method.

pandas.tseries.offsets.CustomBusinessMonthBegin.month_roll

`CustomBusinessMonthBegin.`**`month_roll`**

Define default roll function to be called in apply method.

pandas.tseries.offsets.CustomBusinessMonthBegin.offset

property `CustomBusinessMonthBegin.`**`offset`**

Alias for self._offset.

freqstr	
kwds	
m_offset	
name	
nanos	
rule_code	

Methods

apply_index(self, other)	Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.
rollback(self, dt)	Roll provided date backward to next offset only if not on offset.
rollforward(self, dt)	Roll provided date forward to next offset only if not on offset.

pandas.tseries.offsets.CustomBusinessMonthBegin.apply_index

`CustomBusinessMonthBegin.`**`apply_index`**(*self*, *other*)

Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.

Parameters

i [DatetimeIndex]

Returns

y [DatetimeIndex]

pandas.tseries.offsets.CustomBusinessMonthBegin.rollback

`CustomBusinessMonthBegin.`**`rollback`**`(`*self, dt*`)`

Roll provided date backward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

pandas.tseries.offsets.CustomBusinessMonthBegin.rollforward

`CustomBusinessMonthBegin.`**`rollforward`**`(`*self, dt*`)`

Roll provided date forward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

__call__	
apply	
copy	
isAnchored	
is_anchored	
is_on_offset	
onOffset	

Properties

CustomBusinessMonthBegin.freqstr
CustomBusinessMonthBegin.kwds
CustomBusinessMonthBegin.m_offset
CustomBusinessMonthBegin.name
CustomBusinessMonthBegin.nanos
CustomBusinessMonthBegin.normalize
CustomBusinessMonthBegin.rule_code

pandas.tseries.offsets.CustomBusinessMonthBegin.freqstr

`CustomBusinessMonthBegin.`**`freqstr`**

pandas.tseries.offsets.CustomBusinessMonthBegin.kwds

property CustomBusinessMonthBegin.**kwds**

pandas.tseries.offsets.CustomBusinessMonthBegin.m_offset

CustomBusinessMonthBegin.**m_offset**

pandas.tseries.offsets.CustomBusinessMonthBegin.name

property CustomBusinessMonthBegin.**name**

pandas.tseries.offsets.CustomBusinessMonthBegin.nanos

property CustomBusinessMonthBegin.**nanos**

pandas.tseries.offsets.CustomBusinessMonthBegin.normalize

CustomBusinessMonthBegin.**normalize = False**

pandas.tseries.offsets.CustomBusinessMonthBegin.rule_code

property CustomBusinessMonthBegin.**rule_code**

Methods

CustomBusinessMonthBegin.apply(self, other)	
CustomBusinessMonthBegin.copy(self)	
CustomBusinessMonthBegin.isAnchored(self)	
CustomBusinessMonthBegin.onOffset(self, dt)	
CustomBusinessMonthBegin.is_anchored(self)	
CustomBusinessMonthBegin.is_on_offset(self, dt)	
CustomBusinessMonthBegin.__call__(self, other)	Call self as a function.

pandas.tseries.offsets.CustomBusinessMonthBegin.apply

`CustomBusinessMonthBegin.`**`apply`**(*self*, *other*)

pandas.tseries.offsets.CustomBusinessMonthBegin.copy

`CustomBusinessMonthBegin.`**`copy`**(*self*)

pandas.tseries.offsets.CustomBusinessMonthBegin.isAnchored

`CustomBusinessMonthBegin.`**`isAnchored`**(*self*)

pandas.tseries.offsets.CustomBusinessMonthBegin.onOffset

`CustomBusinessMonthBegin.`**`onOffset`**(*self*, *dt*)

pandas.tseries.offsets.CustomBusinessMonthBegin.is_anchored

`CustomBusinessMonthBegin.`**`is_anchored`**(*self*)

pandas.tseries.offsets.CustomBusinessMonthBegin.is_on_offset

`CustomBusinessMonthBegin.`**`is_on_offset`**(*self*, *dt*)

pandas.tseries.offsets.CustomBusinessMonthBegin.__call__

`CustomBusinessMonthBegin.`**`__call__`**(*self*, *other*)

Call self as a function.

3.8.13 SemiMonthOffset

SemiMonthOffset([n, normalize, day_of_month])	

Attributes

pandas.tseries.offsets.SemiMonthOffset

class pandas.tseries.offsets.**SemiMonthOffset**(*n=1*, *normalize=False*, *day_of_month=None*)

Attributes

base	Returns a copy of the calling offset object with n=1 and all other attributes equal.

pandas.tseries.offsets.SemiMonthOffset.base

property SemiMonthOffset.**base**

Returns a copy of the calling offset object with n=1 and all other attributes equal.

freqstr	
kwds	
name	
nanos	
rule_code	

Methods

rollback(self, dt)	Roll provided date backward to next offset only if not on offset.
rollforward(self, dt)	Roll provided date forward to next offset only if not on offset.

pandas.tseries.offsets.SemiMonthOffset.rollback

SemiMonthOffset.**rollback**(*self*, *dt*)

Roll provided date backward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

pandas.tseries.offsets.SemiMonthOffset.rollforward

SemiMonthOffset.**rollforward**(*self*, *dt*)

Roll provided date forward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

__call__	
apply	
apply_index	
copy	
isAnchored	
is_anchored	
is_on_offset	
onOffset	

Properties

SemiMonthOffset.freqstr
SemiMonthOffset.kwds
SemiMonthOffset.name
SemiMonthOffset.nanos
SemiMonthOffset.normalize
SemiMonthOffset.rule_code

pandas.tseries.offsets.SemiMonthOffset.freqstr

SemiMonthOffset.**freqstr**

pandas.tseries.offsets.SemiMonthOffset.kwds

property SemiMonthOffset.**kwds**

pandas.tseries.offsets.SemiMonthOffset.name

property SemiMonthOffset.**name**

pandas.tseries.offsets.SemiMonthOffset.nanos

property SemiMonthOffset.**nanos**

pandas.tseries.offsets.SemiMonthOffset.normalize

SemiMonthOffset.**normalize = False**

pandas.tseries.offsets.SemiMonthOffset.rule_code

property SemiMonthOffset.**rule_code**

Methods

SemiMonthOffset.apply(self, other)	
SemiMonthOffset.apply_index(self, other)	Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.
SemiMonthOffset.copy(self)	
SemiMonthOffset.isAnchored(self)	
SemiMonthOffset.onOffset(self, dt)	
SemiMonthOffset.is_anchored(self)	
SemiMonthOffset.is_on_offset(self, dt)	
SemiMonthOffset.__call__(self, other)	Call self as a function.

pandas.tseries.offsets.SemiMonthOffset.apply

SemiMonthOffset.**apply**(*self*, *other*)

pandas.tseries.offsets.SemiMonthOffset.apply_index

SemiMonthOffset.**apply_index**(*self*, *other*)

Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.

Parameters

i [DatetimeIndex]

Returns

y [DatetimeIndex]

pandas.tseries.offsets.SemiMonthOffset.copy

SemiMonthOffset.**copy**(*self*)

pandas.tseries.offsets.SemiMonthOffset.isAnchored

SemiMonthOffset.**isAnchored**(*self*)

pandas.tseries.offsets.SemiMonthOffset.onOffset

SemiMonthOffset.**onOffset**(*self*, *dt*)

pandas.tseries.offsets.SemiMonthOffset.is_anchored

SemiMonthOffset.**is_anchored**(*self*)

pandas.tseries.offsets.SemiMonthOffset.is_on_offset

SemiMonthOffset.**is_on_offset**(*self*, *dt*)

pandas.tseries.offsets.SemiMonthOffset.__call__

SemiMonthOffset.**__call__**(*self*, *other*)

Call self as a function.

3.8.14 SemiMonthEnd

SemiMonthEnd([n, normalize, day_of_month])	Two DateOffset's per month repeating on the last day of the month and day_of_month.

pandas.tseries.offsets.SemiMonthEnd

class pandas.tseries.offsets.**SemiMonthEnd**(*n=1*, *normalize=False*, *day_of_month=None*)

Two DateOffset's per month repeating on the last day of the month and day_of_month.

Parameters

n [int]

normalize [bool, default False]

day_of_month [int, {1, 3,…,27}, default 15]

Attributes

base	Returns a copy of the calling offset object with n=1 and all other attributes equal.

pandas.tseries.offsets.SemiMonthEnd.base

property SemiMonthEnd.**base**

Returns a copy of the calling offset object with n=1 and all other attributes equal.

freqstr	
kwds	
name	
nanos	
rule_code	

Methods

rollback(self, dt)	Roll provided date backward to next offset only if not on offset.
rollforward(self, dt)	Roll provided date forward to next offset only if not on offset.

pandas.tseries.offsets.SemiMonthEnd.rollback

SemiMonthEnd.**rollback**(*self*, *dt*)

Roll provided date backward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

pandas.tseries.offsets.SemiMonthEnd.rollforward

SemiMonthEnd.**rollforward**(*self*, *dt*)

Roll provided date forward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

__call__	
apply	
apply_index	
copy	
isAnchored	
is_anchored	
is_on_offset	
onOffset	

Properties

SemiMonthEnd.freqstr	
SemiMonthEnd.kwds	
SemiMonthEnd.name	
SemiMonthEnd.nanos	
SemiMonthEnd.normalize	
SemiMonthEnd.rule_code	

pandas.tseries.offsets.SemiMonthEnd.freqstr

SemiMonthEnd.**freqstr**

pandas.tseries.offsets.SemiMonthEnd.kwds

property SemiMonthEnd.**kwds**

pandas.tseries.offsets.SemiMonthEnd.name

property SemiMonthEnd.**name**

pandas.tseries.offsets.SemiMonthEnd.nanos

property SemiMonthEnd.**nanos**

pandas.tseries.offsets.SemiMonthEnd.normalize

SemiMonthEnd.**normalize = False**

pandas.tseries.offsets.SemiMonthEnd.rule_code

property SemiMonthEnd.**rule_code**

Methods

SemiMonthEnd.apply(self, other)	
SemiMonthEnd.apply_index(self, other)	Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.
SemiMonthEnd.copy(self)	
SemiMonthEnd.isAnchored(self)	
SemiMonthEnd.onOffset(self, dt)	
SemiMonthEnd.is_anchored(self)	
SemiMonthEnd.is_on_offset(self, dt)	

continues on next page

Table 249 – continued from previous page

SemiMonthEnd.__call__(self, other)	Call self as a function.

pandas.tseries.offsets.SemiMonthEnd.apply

SemiMonthEnd.**apply**(*self*, *other*)

pandas.tseries.offsets.SemiMonthEnd.apply_index

SemiMonthEnd.**apply_index**(*self*, *other*)

Vectorized apply of DateOffset to DatetimeIndex, raises NotImplentedError for offsets without a vectorized implementation.

Parameters

i [DatetimeIndex]

Returns

y [DatetimeIndex]

pandas.tseries.offsets.SemiMonthEnd.copy

SemiMonthEnd.**copy**(*self*)

pandas.tseries.offsets.SemiMonthEnd.isAnchored

SemiMonthEnd.**isAnchored**(*self*)

pandas.tseries.offsets.SemiMonthEnd.onOffset

SemiMonthEnd.**onOffset**(*self*, *dt*)

pandas.tseries.offsets.SemiMonthEnd.is_anchored

SemiMonthEnd.**is_anchored**(*self*)

pandas.tseries.offsets.SemiMonthEnd.is_on_offset

SemiMonthEnd.**is_on_offset**(*self*, *dt*)

pandas.tseries.offsets.SemiMonthEnd.__call__

SemiMonthEnd.**__call__**(*self*, *other*)

Call self as a function.

3.8.15 SemiMonthBegin

SemiMonthBegin([n, normalize, day_of_month])	Two DateOffset's per month repeating on the first day of the month and day_of_month.

pandas.tseries.offsets.SemiMonthBegin

class pandas.tseries.offsets.**SemiMonthBegin**(*n=1*, *normalize=False*, *day_of_month=None*)

Two DateOffset's per month repeating on the first day of the month and day_of_month.

Parameters

n [int]

normalize [bool, default False]

day_of_month [int, {2, 3,. . . ,27}, default 15]

Attributes

base	Returns a copy of the calling offset object with n=1 and all other attributes equal.

pandas.tseries.offsets.SemiMonthBegin.base

property SemiMonthBegin.**base**

Returns a copy of the calling offset object with n=1 and all other attributes equal.

freqstr	
kwds	
name	
nanos	
rule_code	

Methods

rollback(self, dt)	Roll provided date backward to next offset only if not on offset.
rollforward(self, dt)	Roll provided date forward to next offset only if not on offset.

pandas.tseries.offsets.SemiMonthBegin.rollback

`SemiMonthBegin.`**`rollback`**(*self*, *dt*)

Roll provided date backward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

pandas.tseries.offsets.SemiMonthBegin.rollforward

`SemiMonthBegin.`**`rollforward`**(*self*, *dt*)

Roll provided date forward to next offset only if not on offset.

Returns

TimeStamp Rolled timestamp if not on offset, otherwise unchanged timestamp.

__call__	
apply	
apply_index	
copy	
isAnchored	
is_anchored	
is_on_offset	
onOffset	

Properties

SemiMonthBegin.freqstr
SemiMonthBegin.kwds
SemiMonthBegin.name
SemiMonthBegin.nanos
SemiMonthBegin.normalize
SemiMonthBegin.rule_code