Slice Objects



- Ellipsis Object

> The type object for slice objects. This is the same as `slice` in the Python layer.

> Return true if *ob* is a slice object; *ob* must not be `NULL`. This function always succeeds.

> Return a new slice object with the given values. The *start*, *stop*, and *step* parameters are used as the values of the slice object attributes of the same names. Any of the values may be `NULL`, in which case the `None` will be used for the corresponding attribute.
>
> Return `NULL` with an exception set if the new object could not be allocated.

> Retrieve the start, stop and step indices from the slice object *slice*, assuming a sequence of length *length*. Treats indices greater than *length* as errors.
>
> Returns `0` on success and `-1` on error with no exception set (unless one of the indices was not `None` and failed to be converted to an integer, in which case `-1` is returned with an exception set).
>
> You probably do not want to use this function.
>
> 3.2 The parameter type for the *slice* parameter was `PySliceObject*` before.

>

> Usable replacement for `PySlice_GetIndices`. Retrieve the start, stop, and step indices from the slice object *slice* assuming a sequence of length *length*, and store the length of the slice in *slicelength*. Out of bounds indices are clipped in a manner consistent with the handling of normal slices. *length* must not be negative.
>
> Return `0` on success and `-1` on error with an exception set.
>
> > [!NOTE] This function is considered not safe for resizable sequences. Its invocation should be replaced by a combination of `PySlice_Unpack` and `PySlice_AdjustIndices` where :
> >
> > if (PySlice_GetIndicesEx(slice, length, &start, &stop, &step, &slicelength) < 0) {
> > // return error
> >
> > }
> >
> > is replaced by :
> >
> > if (PySlice_Unpack(slice, &start, &stop, &step) < 0) {
> > // return error
> >
> > } slicelength = PySlice_AdjustIndices(length, &start, &stop, step);
>
> 3.2 The parameter type for the *slice* parameter was `PySliceObject*` before.
>
> 3.6.1 If `Py_LIMITED_API` is not set or set to the value between `0x03050400` and `0x03060000` (not including) or `0x03060100` or higher `!PySlice_GetIndicesEx` is implemented as a macro using `!PySlice_Unpack` and `!PySlice_AdjustIndices`. Arguments *start*, *stop* and *step* are evaluated more than once.
>
> 3.6.1 If `Py_LIMITED_API` is set to the value less than `0x03050400` or between `0x03060000` and `0x03060100` (not including) `!PySlice_GetIndicesEx` is a deprecated function.

> Extract the start, stop and step data members from a slice object as C integers. Silently reduce values larger than `PY_SSIZE_T_MAX` to `PY_SSIZE_T_MAX`, silently

> boost the start and stop values less than `PY_SSIZE_T_MIN` to `PY_SSIZE_T_MIN`, and silently boost the step values less than `-PY_SSIZE_T_MAX` to `-PY_SSIZE_T_MAX`.
>
> Return `-1` with an exception set on error, `0` on success.
>
> 3.6.1

> Adjust start/end slice indices assuming a sequence of the specified length. Out of bounds indices are clipped in a manner consistent with the handling of normal slices.
>
> *length* must not be negative. *step* must not be zero and must not be less than `-PY_SSIZE_T_MAX`, as guaranteed by `PySlice_Unpack`.
>
> Return the length of the slice. Always successful. Doesn't call Python code.
>
> 3.6.1

Ellipsis Object

> The type of Python `Ellipsis` object. Same as `types.EllipsisType` in the Python layer.

> The Python `Ellipsis` object. This object has no methods. Like `Py_None`, it is an `immortal` singleton object.
>
> 3.12 `Py_Ellipsis` is immortal.

□□□ #2
□ □□□ □□□ 2026-10-31 16:54:03 UTC
□ □□□ □□□ 2026-10-31 16:54:05 UTC