

Iterator Protocol

There are two functions specifically for working with iterators.

int **PyIter_Check**(PyObject *o)

Return true if the object *o* supports the iterator protocol.

PyObject* **PyIter_Next**(PyObject *o)

Return value: New reference.

Return the next value from the iteration *o*. The object must be an iterator (it is up to the caller to check this). If there are no remaining values, returns *NULL* with no exception set. If an error occurs while retrieving the item, returns *NULL* and passes along the exception.

To write a loop which iterates over an iterator, the C code should look something like this:

```
PyObject *iterator = PyObject_GetIter(obj);
PyObject *item;

if (iterator == NULL) {
    /* propagate error */
}

while (item = PyIter_Next(iterator)) {
    /* do something with item */
    ...
    /* release reference when done */
    Py_DECREF(item);
}

Py_DECREF(iterator);

if (PyErr_Occurred()) {
    /* propagate error */
}
else {
    /* continue doing useful work */
}
```