Weak Reference Objects

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]weakref.rst`, line 1)

Unknown directive type "highlight".

```
.. highlight:: c
```

Python supports *weak references* as first-class objects. There are two specific object types which directly implement weak references. The first is a simple reference object, and the second acts as a proxy for the original object as much as it can.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]weakref.rst`, line 14)

Unknown directive type "c:function".

```
.. c:function:: int PyWeakref_Check(ob)

   Return true if *ob* is either a reference or proxy object.  This function
   always succeeds.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]weakref.rst`, line 20)

Unknown directive type "c:function".

```
.. c:function:: int PyWeakref_CheckRef(ob)

   Return true if *ob* is a reference object.  This function always succeeds.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]weakref.rst`, line 25)

Unknown directive type "c:function".

```
.. c:function:: int PyWeakref_CheckProxy(ob)

   Return true if *ob* is a proxy object.  This function always succeeds.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]weakref.rst`, line 30)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyWeakref_NewRef(PyObject *ob, PyObject *callback)

   Return a weak reference object for the object *ob*.  This will always return
   a new reference, but is not guaranteed to create a new object; an existing
   reference object may be returned.  The second parameter, *callback*, can be a
   callable object that receives notification when *ob* is garbage collected; it
   should accept a single parameter, which will be the weak reference object
   itself. *callback* may also be ``None`` or ``NULL``.  If *ob* is not a
   weakly-referencable object, or if *callback* is not callable, ``None``, or
   ``NULL``, this will return ``NULL`` and raise :exc:`TypeError`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]weakref.rst`, line 42)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyWeakref_NewProxy(PyObject *ob, PyObject *callback)
```

```
   Return a weak reference proxy object for the object *ob*.  This will always
   return a new reference, but is not guaranteed to create a new object; an
   existing proxy object may be returned.  The second parameter, *callback*, can
   be a callable object that receives notification when *ob* is garbage
   collected; it should accept a single parameter, which will be the weak
   reference object itself. *callback* may also be ``None`` or ``NULL``.  If *ob*
   is not a weakly-referencable object, or if *callback* is not callable,
   ``None``, or ``NULL``, this will return ``NULL`` and raise :exc:`TypeError`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]weakref.rst`, line 54)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyWeakref_GetObject(PyObject *ref)

   Return the referenced object from a weak reference, *ref*.  If the referent is
   no longer live, returns :const:`Py_None`.

   .. note::

      This function returns a :term:`borrowed reference` to the referenced object.
      This means that you should always call :c:func:`Py_INCREF` on the object
      except when it cannot be destroyed before the last usage of the borrowed
      reference.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]weakref.rst`, line 67)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyWeakref_GET_OBJECT(PyObject *ref)

   Similar to :c:func:`PyWeakref_GetObject`, but implemented as a macro that does no
   error checking.
```