The following scripts explore the Python ‘weakref’ (Weak references) module. The weakref module allows the Python programmer to create weak references to objects.

In the following, the term referent means the object which is referred to by a weak reference.

A weak reference to an object is not enough to keep the object alive: when the only remaining references to a referent are weak references, garbage collection is free to destroy the referent and reuse its memory for something else. However, until the object is actually destroyed the weak reference may return the object even if there are no strong references to it.

A primary use for weak references is to implement caches or mappings holding large objects, where it’s desired that a large object not be kept alive solely because it appears in a cache or mapping.

For example, if you have a number of large binary image objects, you may wish to associate a name with each. If you used a Python dictionary to map names to images, or images to names, the image objects would remain alive just because they appeared as values or keys in the dictionaries. The WeakKeyDictionary and WeakValueDictionary classes supplied by the weakref module are an alternative, using weak references to construct mappings that don’t keep objects alive solely because they appear in the mapping objects. If, for example, an image object is a value in a WeakValueDictionary, then when the last remaining references to that image object are the weak references held by weak mappings, garbage collection can reclaim the object, and its corresponding entries in weak mappings are simply deleted.

WeakKeyDictionary and WeakValueDictionary use weak references in their implementation, setting up callback functions on the weak references that notify the weak dictionaries when a key or value has been reclaimed by garbage collection. WeakSet implements the set interface, but keeps weak references to its elements, just like a WeakKeyDictionary does.

finalize provides a straight forward way to register a cleanup function to be called when an object is garbage collected. This is simpler to use than setting up a callback function on a raw weak reference, since the module automatically ensures that the finalizer remains alive until the object is collected.

Most programs should find that using one of these weak container types or finalize is all they need – it’s not usually necessary to create your own weak references directly. The low-level machinery is exposed by the weakref module for the benefit of advanced uses.

Not all objects can be weakly referenced; those objects which can include class instances, functions written in Python (but not in C), instance methods, sets, frozensets, some file objects, generators, type objects, sockets, arrays, deques, regular expression pattern objects, and code objects.