Сборка мусора (Garbage Collection)

Основы подсчета ссылок

Переменная PHP хранится в контейнере, называемом "zval". Контейнер zval, помимо типа и значения переменной, также содержит два дополнительных элемента. Первый называется "is_ref" и представляет булево значение, указывающее, является переменная частью "набора ссылок" или нет. Благодаря этому элементу PHP знает как отличать обычные переменные от ссылок. Так как PHP содержит пользовательские ссылки, которые можно создать оператором &, контейнер zval также содержит внутренний механизм подсчета ссылок для оптимизации использования памяти. Эта вторая часть дополнительной информации, называемая "refcount" (счетчик ссылок), содержит количество имен переменных (также называемых символами), которые указывают на данный контейнер zval. Все имена переменных хранятся в таблице имен, отдельной для каждой области видимости переменных. Такая область видимости существует для главного скрипта, а также для каждой функции и метода.

Контейнер zval создается при создании новой переменной, которой присваивается константа, например:

$a = "new string";

В данном примере создается новый символ a в текущей области видимости и новый контейнер переменной с типом string и значением new string. Бит "is_ref" по умолчанию задается равным FALSE, т.к. не создано ни одной пользовательской ссылки. Значение же "refcount" задается равным 1, т.к. только одно имя переменной указывает на данный контейнер. Отметим, что если "refcount" равен 1, то "is_ref" будет всегда равен FALSE.

Присвоение этой переменной другой увеличивает счетчик ссылок.

$b = $a;

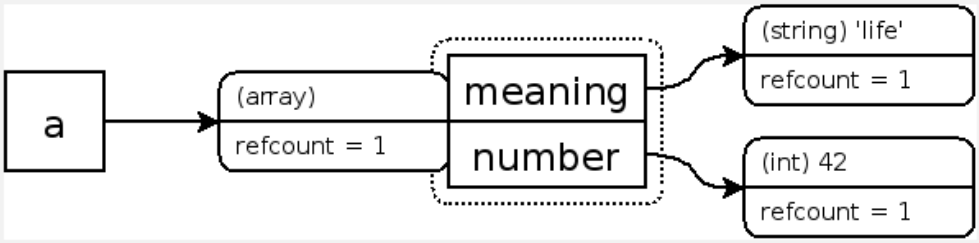
Счетчик ссылок здесь равен 2, т.к. a и b ссылаются на один и тот же контейнер переменной. PHP достаточно умен, чтобы не копировать контейнер, пока в этом нет необходимости. Как только "refcount" станет равным нулю, контейнер уничтожается. "refcount" уменьшается на единицу при уходе переменной из области видимости (например, в конце функции) или при удалении этой переменной (например при вызове unset()).

Составные типы данных

Все несколько усложняется с составными типами данных, такими как массивы (array) и объекты (object). В отличии от скалярных (scalar) значений, массивы и объекты хранят свои свойства в собственных таблицах имен. Это значит, что следующий пример создаст сразу три zval контейнера:

$a = array('meaning' => 'life', 'number' => 42);

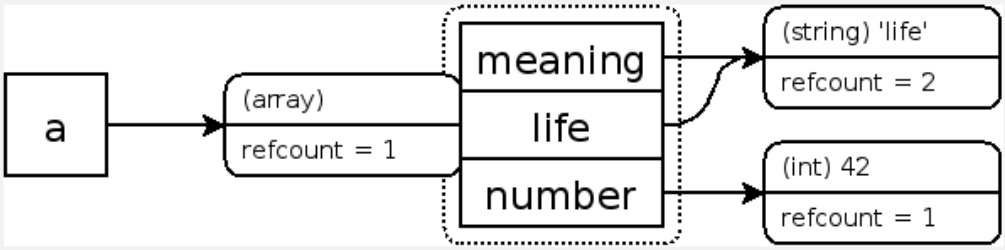
Графически:



Результат - три контейнера: a, meaning и number. Похожие правила применяются и для увеличения и уменьшения "refcounts". Ниже мы добавляем еще один элемент массива и устанавливаем ему значение уже существующего элемента:

$a = array('meaning' => 'life', 'number' => 42); $a['life'] = $a['meaning'];

Графически:



Удаление элемента из массива происходит точно так же, как и удаление имени переменной из области видимости: уменьшается "refcount" контейнера, на который ссылается элемент массива. Опять же, при достижении "refcount" нуля, контейнер удаляется из памяти.