WPs provide the solution to specific problems when working with shared pointers. The weak\_ptr, henceforth referred to as WP, is a type of smart pointer that provides a solution to problems associated with accessing an object that may or may not be "alive". This is the basic problem that WPs solve. Here we explain what a WP is, what a WP does and how one works.

What is a weak\_ptr? A WP is itself an object that stores a reference to an object pointed to by a shared pointer. The WP is a tool for programmers to safely access objects that may not exist. Additionally, the WP is a tool to break a reference cycle that causes memory leaks.

What does a WP do? A WP has six methods that provide information about the object it points to and about the shared pointers that point to an object.

**Functions of shared\_ptr**

|  |  |
| --- | --- |
| reset | Releases ownership of the object being pointed to (managed object). |
| swap | Swaps a managed object for a different managed object. |
| use\_count | Returns the number of shared\_ptr objects that point to an object. |
| expired | Checks to see if the object being pointed to exists. |
| lock | Creates a shared\_ptr that points to the object. |
| owner\_before | Provides owner based ordering of WPs. |

Some of the characteristics of WPs are 1. WPs only work in conjunction with shared pointers 2.WPs do not increase the number of strong references to the object it points to. 3. WPs have no ability to access the object it points to. A WP's main job is to give information about the shared pointers and the object shared pointers point to.

How does a WP work? We know multiple shared pointers point to the same object. A danger exists if a program tries to access an object pointed to by a shared\_ptr if the object no longer exists in memory. It is sometimes difficult to keep track of numerous shared pointers. A WP works by calling its member functions on shared pointers such as, “expired”. This member function checks to see if the object being pointed to still exists. This can inform the program if it is safe to access the managed object. Additionally, a WP forms a “weak” reference to an object. This can prevent memory leaks by allowing the programmer to prevent strong references cycles by using a WP(creates a weak reference) instead of shared pointer (creates a strong reference).