Java GC Roots

Garbage-Collection Roots—The Source of All Object Trees

Every object tree must have one or more root objects. As long as the application can reach those roots, the whole tree is reachable. But when are those root objects considered reachable? Special objects called garbage-collection roots (GC roots; see Figure 2.2) are always reachable and so is any object that has a garbage-collection root at its own root.

There are four kinds of GC roots in Java:

* **Local variables** are kept alive by the stack of a thread. This is not a real object virtual reference and thus is not visible. For all intents and purposes, local variables are GC roots.
* **Active Java threads** are always considered live objects and are therefore GC roots. This is especially important for thread local variables.
* **Static variables** are referenced by their classes. This fact makes them de facto GC roots. Classes themselves can be garbage-collected, which would remove all referenced static variables. This is of special importance when we use application servers, [OSGi containers](http://www.wikipedia.com/osgi) or class loaders in general. We will discuss the related problems in the Problem Patterns section.
* **JNI References** are Java objects that the native code has created as part of a JNI call. Objects thus created are treated specially because the JVM does not know if it is being referenced by the native code or not. Such objects represent a very special form of GC root, which we will examine in more detail in the Problem Patterns section below.

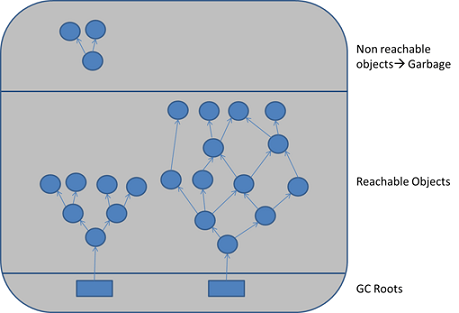


Figure 2.2: GC roots are objects that are themselves referenced by the JVM and thus keep every other object from being garbage-collected.

Therefore, a simple Java application has the following GC roots:

* Local variables in the main method
* The main thread
* Static variables of the main class

# GC roots

The so-called GC (Garbage Collector) roots are objects special for garbage collector. Garbage collector collects those objects that are not GC roots and are not accessible by references from GC roots.

There are several kinds of GC roots. One object can belong to more than one kind of root. The root kinds are:

* **Class** - class loaded by system class loader. Such classes can never be unloaded. They can hold objects via static fields. Please note that classes loaded by custom class loaders are not roots, unless corresponding instances of java.lang.Class happen to be roots of other kind(s).
* **Thread** - live thread
* **Stack Local** - local variable or parameter of Java method
* **JNI Local** - local variable or parameter of JNI method
* **JNI Global** - global JNI reference
* **Monitor Used** - objects used as a monitor for synchronization
* **Held by JVM** - objects held from garbage collection by JVM for its purposes. Actually the list of such objects depends on JVM implementation. Possible known cases are: the system class loader, a few important exception classes which the JVM knows about, a few pre-allocated objects for exception handling, and custom class loaders when they are in the process of loading classes. **Unfortunately, JVM provides absolutely no additional detail for such objects. Thus it is up to the analyst to decide to which case a certain "Held by JVM" belongs.**

If an object is a root, it is specially marked in all views showing individual objects. For example, the following picture shows a fragment of [paths](https://www.yourkit.com/docs/java/help/paths.jsp) view:

