17

Java Objects can be created in several different ways.

1. **From Java code**, when a Java method, either interpreted or compiled, executes one of the following bytecode instructions: `new`, `newarray`, `anewarray`, `multianewarray`.
2. **From native code**, when native methods, including those in standard class library, call one of JNI functions: `NewObject`, `NewObjectArray`, `NewStringUTF`, `NewDirectByteBuffer`, etc.
3. **Directly from VM runtime**, when a new object is created internally by JVM, for example, in response to `Object.clone()`, `Throwable.getStackTrace()`, `Class.getInterfaces()`, etc.

Unfortunately, there is no single point where you can collect objects from all these sources. However, there are means for intercepting all of them.

1. Objects instantiated from Java can be caught by an Instrumentation agent. The agent needs to define a ClassFileTransformer that will scan the bytecode of all loaded classes for object-creating instructions and modify it.

   Note: there is no need to intercept all `new` instructions, you can instrument `Object()` constructor instead. But you still need to intercept array allocation instructions.
2. JNI functions can be intercepted by JVMTI agent. You need to define your own native hooks for `NewObjectArray`, `NewStringUTF` etc. and then replace JNI function table. See JVMTI Referencefor the details.
3. Objects created by the VM can be caught by JVMTI Event Callback mechanism. The desired event is VMObjectAlloc.

   Note: JVM will not post `VMObjectAlloc` event for objects allocated from Java or by JNI functions.

All other ways of object instantiation (cloning, reflection, deserialization) fall into one of the above categories.

Get JDK 8 Demos and Samples from Oracle Java SE Downloads website.
There is a sample JVMTI agent for exactly this question.

Look under

- `jvmti/heapTracker`
- `jvmti/hprof`

edited Mar 10 '16 at 21:44 answered Mar 10 '16 at 21:25