```
// Application thread
JVM_Halt()


  before_exit(thread);
  vm_exit(code);
     VM_Exit op(code);
     VMThread::execute(&op);  <- never returns
       // VMThread
       VM_Exit::do_it():
```

```
        CompileBroker::set_should_block();
        wait_for_threads_in_native_to_block();


        set_vm_exited();


         exit_globals();
         exit_hook_t exit_hook = Arguments::exit_hook();
         // exit hook should exit.
         exit_hook(_exit_code);
         // ... but if it didn't, we must do it here
         vm_direct_exit(_exit_code);
           notify_vm_shutdown();  // later but trivial
```

```
// Application thread - waiting for last non-daemon to exit
jni_DestroyJavaVM:
  Threads::destroy_vm()
    thread->invoke_shutdown_hooks();
    before_exit(thread);
    thread->exit(true);
    // Grab heap lock so heap is parseable during verification
    MutexLocker ml(Heap_lock, Mutex::_no_safepoint_check_flag);
    // Stop VM thread.
    VMThread::wait_for_vm_thread_exit();
          // VMThread:
          this->loop(); // returns
          if (xtty != NULL) {
            ttyLocker ttyl;
            xtty->begin_elem("destroy_vm");
            xtty->stamp();
            xtty->end_elem();
          }
          _cur_vm_operation = &halt_op;  // empty VM_operation
          SafepointSynchronize::begin(); // take to safepoint
          if (VerifyBeforeExit) {
            HandleMark hm(VMThread::vm_thread());
            Universe::heap()->prepare_for_verify();
            Universe::verify();
          }

          CompileBroker::set_should_block();
          VM_Exit::wait_for_threads_in_native_to_block();
         // signal other threads that VM process is gone
         // VMThread terminates execution
    VMThread::destroy();
    VM_Exit::set_vm_exited();
    IdealGraphPrinter::clean_up();
    notify_vm_shutdown();
    exit_globals();
    delete thread;
    LogConfiguration::finalize();
    return;
return;
```

Items in red show common actions