```
/*
 *            DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
 *                    Version 2, December 2004
 *
 * Everyone is permitted to copy and distribute verbatim or modified
 *  copies of this license document, and changing it is allowed as long
 * as the name is changed.
 *
 *            DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
 *   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
 *
 *  0. You just DO WHAT THE FUCK YOU WANT TO.
 */

import java.util.Arrays;
import org.virtualbox_4_3.GuestSessionStatus;
import org.virtualbox_4_3.IEvent;
import org.virtualbox_4_3.IEventListener;
import org.virtualbox_4_3.IEventSource;
import org.virtualbox_4_3.IGuestProcess;
import org.virtualbox_4_3.IGuestSession;
import org.virtualbox_4_3.IMachine;
import org.virtualbox_4_3.ISession;
import org.virtualbox_4_3.LockType;
import org.virtualbox_4_3.ProcessCreateFlag;
import org.virtualbox_4_3.ProcessWaitForFlag;
import org.virtualbox_4_3.ProcessWaitResult;
import org.virtualbox_4_3.VBoxEventType;
import org.virtualbox_4_3.VirtualBoxManager;

public class GuestSessionProcess {

   private static String errorIfNull(String systemProperty) {
      String value = System.getProperty(systemProperty);
      if (value != null) {
         return value;
      } else {
         throw new RuntimeException(systemProperty + " is not set");
      }
```

```
40.    }
41.
42.    public static void main(String[] args) {
43.       String vmId = errorIfNull("vbox.vm");
44.       String guestUser = errorIfNull("vbox.guest.user");
45.       String guestPass = errorIfNull("vbox.guest.pass");
46.       String exec = errorIfNull("vbox.guest.exec");
47.
48.       System.out.println("Starting");
49.       VirtualBoxManager vboxManager = VirtualBoxManager.createInstance(null);
50.       try {
51.          if (System.getProperty("vbox.ws") != null) {
52.             String host = System.getProperty("vbox.ws.host", "http://localhost:18083");
53.             String user = System.getProperty("vbox.ws.user", "");
54.             String pass = System.getProperty("vbox.ws.pass", "");
55.             vboxManager.connect(host, user, pass);
56.          }
57.          System.out.println("Connected");
58.          try {
59.             ISession session = vboxManager.getSessionObject();
60.             IMachine vm = vboxManager.getVBox().findMachine(vmId);
61.             vm.lockMachine(session, LockType.Shared);
62.             try {
63.                System.out.println("Machine locked");
64.                IGuestSession guestSess = session.getConsole().getGuest().createSession(guestUser, guestPass, null, null);
65.                try {
66.                   System.out.println("Session created");
67.
68.                   guestSess.waitFor(1L, 30 * 1000L);
69.                   if (!guestSess.getStatus().equals(GuestSessionStatus.Started)) {
70.                      throw new RuntimeException("Guest session did not start after 30 sec");
71.                   }
72.                   IGuestProcess process = guestSess.processCreate(exec, null, null, Arrays.asList(ProcessCreateFlag.WaitForStdOut), 0L);
73.
74.                   IEventSource es = process.getEventSource();
75.                   IEventListener el = es.createListener();
76.                   es.registerListener(el, Arrays.asList(VBoxEventType.Any), false);
77.
78.                   try {
```

```
                    System.out.println("Guest process created");
                    ProcessWaitResult pwr = process.waitFor((long) ProcessWaitForFlag.Start.value(), 30 * 1000L);
                    System.out.println("Process wait result: " + pwr);
                    boolean keepLooping = true;
                    do {
                       IEvent ev = es.getEvent(el, 200);
                       if (ev != null) {
                          es.eventProcessed(el, ev);
                       }
                       /* This is how you should normally do it, but waiting for Stdout is not currently implemented - see http://hyperbox.altherian.org/kb/guessProcessHandling.txt
                       ProcessWaitResult wr = process.waitForArray(Arrays.asList(ProcessWaitForFlag.StdOut, ProcessWaitForFlag.Terminate), 200L);
                       System.out.println("Process wait result: " + wr);
                        */
                       byte[] stdOut = process.read(1L, 64L, 0L);
                       System.out.print(new String(stdOut));
                       keepLooping = !process.getStatus().toString().contains("Terminated");
                    } while (keepLooping);
                    System.out.println("Process exit code: " + process.getExitCode());
                 } finally {
                    es.unregisterListener(el);
                    if (!process.getStatus().toString().contains("Terminated")) {
                       process.terminate();
                    }
                 }
              } finally {
                 System.out.println("Session close");
                 guestSess.close();
              }
           } catch (Throwable t) {
              t.printStackTrace();
           } finally {
              System.out.println("Machine unlock");
              session.unlockMachine();
           }
        } finally {
           if (System.getProperty("vbox.ws") != null) {
              vboxManager.disconnect();
```

```
            }
            System.out.println("Disconnected");
         }
      } finally {
         vboxManager.cleanup();
         System.out.println("Closing");
      }
   }

}
```