

```

import java.io.BufferedReader;

/* Monitors given mailbox for new mail */
public class MonitorMailbox {
    public static void main(String argv[]) {

        String protocol = "imap";
        String host = "mail.bonnylundy.com";
        String user = "itp@bonnylundy.com";
        String password = "interact";
        String mbox = "INBOX";
        String freqarg = "5";
        System.out.println("\nTesting monitor\n");
        try {
            // Get a Session object
            Session session = Session.getDefaultInstance(
                System.getProperties(), null);
            session.setDebug(true);
            // Get a Store object
            Store store = session.getStore(protocol);
            // Connect
            store.connect(host, user, password);
            // Open a Folder
            Folder folder = store.getFolder(mbox);
            if (folder == null || !folder.exists()) {
                System.out.println("Invalid folder");
                System.exit(1);
            }
            folder.open(Folder.READ_WRITE);
            // Add messageCountListener to listen for new messages
            folder.addMessageCountListener(new MessageCountAdapter() {
                public void messagesAdded(MessageCountEvent ev) {
                    Message[] msgs = ev.getMessages();
                    System.out.println("Got " + msgs.length + " new messages");

                    // Just dump out the new messages
                    for (int i = 0; i < msgs.length; i++) {

                        try {
                            //msgs[i].writeTo(System.out);

                            String smasher = msgs[i].getContent().toString();
                            System.out.println("smash smash smash... "
                                + smasher);

                            //
                            if (smasher.toUpperCase().indexOf("SLEEP") != -1) {
                                String bangIt = "1;n/";
                                sendMe(bangIt);
                            }
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    }
                }
            });
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

```

-1) {
    } else if (smasher.toUpperCase().indexOf("WAKE") !=
        String bangIt = "2;n/";
        sendMe(bangIt);
-1) {
    } else if (smasher.toUpperCase().indexOf("SEX") !=
        String bangIt = "3;n/";
        sendMe(bangIt);
    } else if (smasher.toUpperCase()
        .indexOf("CONFUSED") != -1) {
        String bangIt = "4;n/";
        sendMe(bangIt);
    } else if (smasher.toUpperCase().indexOf("DIRTY") !=
-1) {
        String bangIt = "5;n/";
        sendMe(bangIt);
    } else if (smasher.toUpperCase().indexOf("Z") != -1)
{
        String bangIt = "6;n/";
        sendMe(bangIt);
    } else {
        System.out.println("got nothin, really");
    }
} catch (Exception ex) {
    ex.printStackTrace();
}
}
});
// Check mail once in "freq" MILLIseconds
int freq = Integer.parseInt(freqarg);
while (true) {
    Thread.sleep(freq); // sleep for freq milliseconds
    // This is to force the IMAP server to send us
    // EXISTS notifications.
    folder.getMessageCount();
}
} catch (Exception ex) {
    ex.printStackTrace();
}
}

```

```

public static void sendMe(String highNote) throws IOException {
    Socket echoSocket = null;
    PrintWriter out = null;
    BufferedReader in = null;
    try {

```

```
        echoSocket = new Socket("localhost", 3001);
        out = new PrintWriter(echoSocket.getOutputStream(), true);
        System.out.println("yea, whatever... " + echoSocket.getOutputStream
    ());
    } catch (UnknownHostException e) {
        System.err.println("Don't know about host: jitter patch.");
        System.exit(1);
    } catch (IOException e) {
        System.err.println("Couldn't get I/O for "
            + "the connection to: jitter patch.");
        System.exit(1);
    }
    System.out.println(highNote + ";");
    out.println(highNote);
}
}
```