VB UCM Engineering Applicant Test

Version 2018-05-02

Introduction

This test is designed to take you through some tasks that would be typical for a 4Sight software developer. It should allow you to get a sense of the sort of work that you would be involved with at JSI, and will help us asses your suitability for our team.

You have 3 hours to complete this test. The test results must be emailed to JSI no later than 3 hours after receiving it. Please leave yourself sufficient time to document and package your results. The results package may contain code files and text documents. Please ensure it is clear which items relate to which question.

Many of the questions below require you to write some code. We prefer if the code is written in C#, Java or C++. However, we will accept other object oriented languages and frameworks if you feel uncomfortable with those options. If you are in need of a development environment, you can download express editions of Visual Studio and SQL Server or the community edition of intelliJ.

Question 1 – Reverse Engineering

For this question you will need to refer to the attached PCAP file: sent_email.pcap. If you have never heard of a PCAP file before, DON'T PANIC! Digging into unknown formats and protocols is part of what we do every day. Fortunately everything you need to answer this question can be found online...

You have been provided with a packet capture (PCAP) file which contains an intercept of an email being sent from a target's computer. Examine this file and answer the following questions:

1. What transport layer protocol is used to send the email? How do you know?
2. What application protocol is being used to send the email? How do you know?
3. What is the email address of the target?
4. What is the subject of the email?
5. What is the colour of the target's car?
6. What is the target's password?

Question 2 – Data Modelling

Create a set of types to store information about an intercepted email. Recall that each email has a set of **headers**, a **message body** and possibly one or more **attachments**. In addition, each email has a **sender** and one or more **recipients**. For more information about email transfer formats see RFCs 2822 and 2045.

Question 3 – Data Access

Design a set of database tables to persist the information in your email types. Write a short summary of the columns in each table. Time permitting, implement code to write data from the types to the database and read it back. You can use any data access technology you choose for this (e.g. ADO.NET, LINQ to SQL).

Question 4 – Parsing

Write code to extract the *name* and *email address* of the *sender* of an email message from the email headers. Assume the email headers are formatted according to the specification in RFC 2822. The input should be a single string containing all the email headers. The output should be two strings containing the sender's name and email address. Assume that an email has only one sender.

The following is an example of a set of email headers (from RFC 2822):

```
Received: from x.y.test by example.netvia TCP with ESMTP id ABC12345
   for <mary@example.net>;  21 Nov 1997 10:05:43 -0600
Received: from machine.example by x.y.test; 21 Nov 1997 10:01:22 -0600
From: John Doe <jdoe@machine.example>
To: Mary Smith <mary@example.net>
Subject: Saying Hello
Date: Fri, 21 Nov 1997 09:55:06 -0600
Message-ID: <1234@local.machine.example>
```

Given this input, your code should parse out "John Doe" and "jdoe@machine.example". Note that RFC 2822 allows for a variety of variations on the value of the "From" header. See section 3.4 of RFC 2822 for the full grammar of this header.

Question 5 – Unit Testing

Write a few (say, around 4) unit tests to exercise the code you wrote in your answers to questions 3 and 4. The tests should be executable using VSTest or an equivalent unit testing framework. Explain in comments what each test is meant to exercise.

If you are unfamiliar with unit testing frameworks and feel you don't have time to learn how to use VSTest, simply write a console application that runs a few test methods and prints out the results.

Question 6 – Technical Writing

You've been asked to design a web-based user interface that would allow a user to explore the intercepted emails stored in the database. The interface should be able to list all the emails in the database, and filter some criteria such as the sender's email address. Write a short (about 300 word) *functional specification* document describing what the interface would look like, and how a user would interact with it.

Do not actually write the code for the user interface!

Question 7 – Multithreading and Synchronization

For this question, refer to the Visual Studio 2013 solution contained in SyncQueue.zip. This solution relies on VSTest to execute a test. However, it could be easily refactored into a console application if you would prefer.

The attached solution (SyncQueue.sln) contains an implementation of a synchronized producer-consumer queue (SyncQueue<T>). This is a special queue that is meant to allow any number of threads to simultaneously enqueue and dequeue items. In the current implementation, the Dequeue() method blocks the calling thread until an item is available on the queue or until a specified timeout has elapsed.

However, there is a problem with the current implementation of SyncQueue that is illustrated in the accompanying unit test. Occasionally items get “stuck” in the queue – i.e. they do not get properly dequeued. The problem is magnified when more than one producer thread is active. The unit test uses two producer threads and fails just about every time.

Describe what you think is wrong with the SyncQueue<T> implementation. Write an enhanced version of the SyncQueue class that passes the unit test.