ELEKTA CODING CHALLENGE

SOFTWARE ENGINEER – REMOTE SERVICES

# PREREQUISITS

In order to complete this coding challenge, you will require the following:

* Visual Studio 2015 (A free community edition can be obtained from the following link [https://www.visualstudio.com/free-developer-offers/)](https://www.visualstudio.com/free-developer-offers/)
* The CodingExercise.zip
* The solution should build successfully on your system

If you have any difficulty with this then please contact your recruitment agency.

# THE CODING CHALLENGE

One of our junior developer has implemented a reporting application for a veterinary practice. He has sent his code to you for a walkthrough/code review.

We would like you to give him some guidance on how he can improve his code. Things that he/you may want to consider:

* The application should be extensible.
* The structure of projects in the solution.
* Good unit testing practices.
* Adherence to SOLID principles.
* Clean code principles.
* The solution should be object-oriented.

Refactor his code, making a note of the changes you have made and why.

# DELIVERABLES

E-mail your contact at Elekta with your solution. Your e-mail must include:

 A file attachment (ZIP file type) containing all of the files (i.e., C# source code, Visual Studio C# project) required to build your Microsoft .NET C# project. Include the provided source code files in your project.

# QUESTIONS

If you have any questions regarding this coding challenge, then please forward them to your Elekta contact and we will get back to you as soon as possible.

# Further Comments

Thanks for this, a very interesting a stimulating challenge. The WPF app should run ‘out of the box’, I’ve swapped out the Data access class for the IReportRepo interface to point to the ReportRepoTest class in the Bootstrapper class. I’ve commented the original piece, though I’ll just add a few thoughts and assumptions here.

I’ve used Caliburn Micro for the MVVM pattern and have been impressed with it.

Unit testing uses MOQ. Only 2 I’m afraid. However, I’m pretty excited about mocking and potentially activating a ViewModel in the test project.

My main aim has been to give a structure and working demo that would be extensible and we could progress as a team and demo to the client. My main design concern (other working code in a test project 😊) was the insinuation that we would continue to add a new animal class for each new animal that joined the practice. I believe this is taking inheritance too far, though I wouldn’t want to dissuade the programmer from using good OO principles. I feel we could pare this back the Animal : Owner relationship, allow the data structure to do the baulk of the work and keep the display code as clean as possible. That could be as I’ve done in the code with a data manager and Dapper, utilizing stored procs or a combination of these. As such the data structure would look like the schema below.

So, plenty left to do. Ironically enough the area where I ran out of time after thinking about data structure and a vehicle to deliver the demo to the client was the report writing itself. I have improved this with some basic error handling and making the string construction more efficient. I think the first task I would give to the Junior Dev would be to say that there is no need to reinvent the wheel when it comes to either DataGrid / BindableCollection / List<Object> TO csv file using C#. I’d ask him or her to evaluate the library options that have already been written to do this and implement the best.

Hopefully continue this discussion as part of the interview process.

Best regards Dave H

