Informatik

SWE2 Project Report

Gregor Holli

Introduction

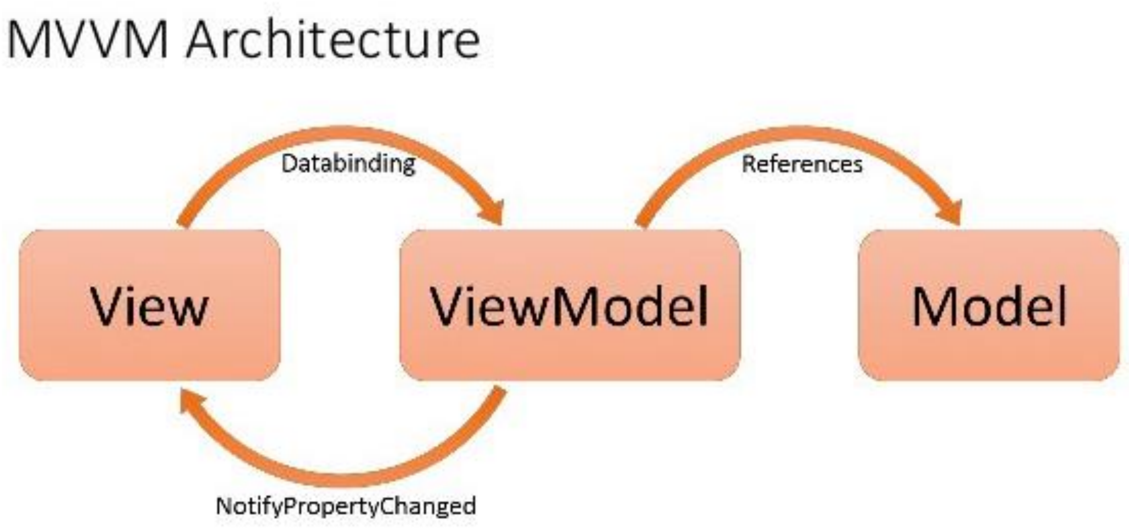
This report aims to compensate for previous assignments. It contains both a description of the project itself as well as my gained experiences and learnings.

WPF Framework

WPF stands for Windows Presentation Foundation, a versatile framework, which integrates powerful frontend tools to develop GUIs for Windows on .Net Core. While WPF has been added to .Net Framework as well, this project was developed using .Net Core.

MVVM Pattern

MVVM stands for Model – View – Viewmodel and offers a structure similar to the Model View Controller, while requiring Data Binding. It is a pattern which intends to separate the user interface or View from the data or Model. The connector between the two is the Viewmodel, the controller which Transforms the Model data into a visual format and offers extended methods and access to the necessary backend logic in the Data Access- and Business Layer.



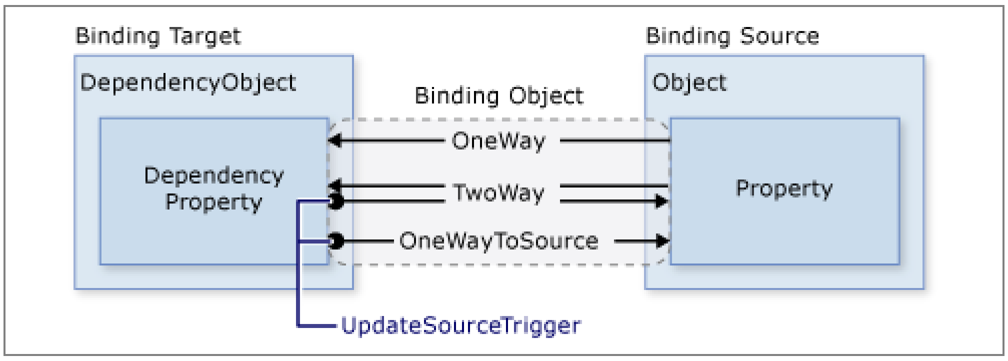
This pattern is specifically targeted for UI frameworks such as JavaFX, HTML5 and of course WPF.

Windows and UserControls

Any given WPF application starts with a Window, per default called the MainWindow. User Controls as a View very similar to Windows except they are designed to be reusable components. They can be placed in other controls or windows.

Data Binding

DataBinding allows for controls to be linked up with data or other controls. Elements can be bound to data from a variety of data sources. It is declared in the View XAML and can be notified via INotifyPropertyChanged or DependencyProperties



Data binding functions are supported by several properties and allow for a clean separation between UI and data.

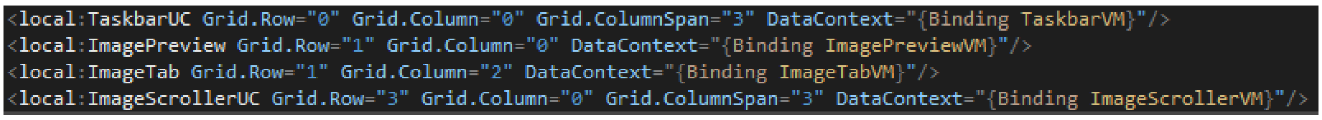
This clean separation however also means that errors with data binding can be easy to miss. Once several user controls and bindings are involved it can be easy to mess up, which Visual Studio, naturally would not be able to detect as an “error” or spelling mistake.

Data Access and Business Layer

Similar to their functions in the MVC pattern, the data access logic is separated from the presentation layer. The data access layer provides a centralized location for all calls to database and therefore makes the application easier to port to other databases.
Business logic is responsible for handling the way application data is processed and transformed withig the application itself.

Project Structure

The basic project structure of the PicDB project contains all previously mentioned elements. The MainWindow is comprised solely of separately implemented user controls, all with their own View Models and data bindings.



```
<local:TaskbarUC Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="3" DataContext="{Binding TaskbarVM}"/>
<local:ImagePreview Grid.Row="1" Grid.Column="0" DataContext="{Binding ImagePreviewVM}"/>
<local:ImageTab Grid.Row="1" Grid.Column="2" DataContext="{Binding ImageTabVM}"/>
<local:ImageScrollerUC Grid.Row="3" Grid.Column="0" Grid.ColumnSpan="3" DataContext="{Binding ImageScrollerVM}"/>
```

In the data access layer folder, the classes necessary for database interaction and the file system are present. The business logic for the photographers and pictures is placed in the Business Layer.

Reflection

While working on the project I found myself frequently distracted and starting over. Initially I had started using the Caliburn.Micro library, however found that I was having greater difficulties working with documentation and tutorials intended for stand alone WPF projects. Whether it be the way several ways different tutorials chose to implement Data Binding or the way some chose to fill their Viewmodels and Models with excessive logic, I found myself very confused and watching tutorials rather than practicing, which inevitably lead to a lot of wasted time.

By the end of the semester I had two to three versions of the same project which had each implemented some parts using different logical structures, however neither one was completed. While trying to fix these failed attempts at a fresh start I learned the value of planning ahead and consistency. Every setback should have been an opportunity to learn and was instead used as a personal excuse to give up. Several of these issues, however, simply came back in one way or another in the next attempt.

Without going into too much detail, it had been a difficult time for me, and my anxiety was getting the better of me far too often. The only way to learn was to mentally draw out the issue, study the documentation and with a clear picture in mind, find the easiest path to success. Doing so not only helped me progress, but it also helped me combat the general increase in stress I experienced whenever I had to tackle a task, which I deemed to be important.

I found out how much my personally perceived lack of discipline is dependent on how I fell about a task. The less significant I perceive a task to be, the easier it is to get started and therefore the easier it is finish, independent of how much work or effort is actually required.

False starts and low confidence in my work has become even more so an issue in the last year than ever before and I have plans to work on these with professional help.
I have found intentions to be a terrible indicator of the truth, even in myself. Not until actions are taken, has an intention or thought become a reality.
Until then they simply serve as distractions, making it more difficult to see the future ahead of oneself.
I have felt terrible for quite a while and I see now more so than ever before that what I thought are good reasons to be demotivated, are in fact just convincing excuses.

Now that I have finished raving over myself, I feel there is very little I can do in terms of further explaining my awful workflow, without sounding like a child.
Therefore, I would like to once again apologize, I understand that the current times are difficult for everyone.

Not that it would be necessary but here are some interesting sources I came up with:

http://wangxinliu.com/tech/program/WPF-DataBinding/

https://blog.rsuter.com/recommendations-best-practices-implementing-mvvm-xaml-net-applications/

https://www.c-sharpcorner.com/UploadFile/1e050f/tooltip-in-wpf/#:~:text=The%20ToolTip%20element%20in%20XAML,controls%20using%20C%23%20and%20XAML.

https://www.tutorialspoint.com/mvvm/index.htm