Diario di lavoro

Luogo	SAM Trevano
Data	24.09.2019

Lavori svolti

Oggi ho ripreso direttamente da dove avevo finito l'ultima lezione, ovvero dalla view iniziale. Ho cercato di risolvere il problema legato al riferimento al ViewModel. Infatti il riferimento era corretto ma nonostante ciò veniva generato l'errore "il file ViewModel.cs non è presente nel namespace QDC.ViewModels". L'errore non si risolveva neanche dopo la compilazione del progetto. Ho provato a fare invece "ricompila soluzione" e l'errore non è persistito. L'altro errore era riferito alla riapertura del progetto dopo averlo chiuso (non caricava correttamente i file presenti nelle cartelle ViewModels e Views. Anche in questo caso mi è bastato fare "ricompila soluzione" e il caricamento dei file è avvenuto nel modo corretto. Per evitare problemi con il progetto che avevo creato ne ho rifatto velocemente uno uguale.
La prima cosa che ho fatto dopo aver creato le classi (uguali a quelle del progetto precedente, vedi diario 20.09.2019) è stata assicurarmi che i riferimenti funzionassero e che dopo il salvattaggio e la chiusura del progetto, alla successiva riapertura venisse caricato correttamente. Entrambe le operazioni sono andate a buon fine, quindi ho potuto contiunare.
Ho inserito il riferimento della mia MainView nel file MainWindow.xaml (il file che viene aperto quando si fa partire il programma):

```
xmlns:view="clr-namespace:QDCeValutazioni.Views"
<Grid>
        <view:MainView></view:MainView>
</Grid>
```

In questo modo ho reperito la view che avevo creato e quando faccio partire il progetto viene aperta direttamente la view desiderata (quella con la scelta tra crea QDC e seleziona QDC).
Dopodiché ho creato la classe Generale, che servirà per inserire le informazioni generiche di un quaderno dei compiti. Per crearla mi sono basato sul mockup che ho fatto in precedenza e ho usato `<Grid.RowDefinitions>` e `<Grid.ColumnDefinitions>` per renderlo responsive.
Poi ho creato la cartella Helper dove ho inserito la classe astratta BindableBase che implementa InotifyPropertyChanged per evitare di doverla implementare e inizializzare in ogni classe che creo. BindableBase si occupa quindi di gestire tutta la parte di notifica (nella view viene implementata tramite Binding Path: `Command="{Binding Path=GeneraleCommand}`).
creato anche la classe ViewBindableBase dove definirò i metodi per la gestione dei comandi/eventi e la comunicazione tra view e viewmodel. Questa classe implementa BindableBase.
Infine ho creato la classe DelegateCommand che implementa l'interfaccia IDelegateCommand. Queste classi servono per far si che alla view vengano passati i comandi in sola lettura.
Per ultima cosa ho aggiornato la documentazione inserendo i passaggi svolti oggi.

Problemi riscontrati e soluzioni adottate

Il passaggio da una view all'altra tramite un pulsante che implementa il comando `Command="{Binding Path=GeneraleCommand}` non funziona, entre il Binding del testo funziona correttamente. Questo significa che sto sbagliando qualcosa nella gestione dei metodi nel ViewModel che si occupano di cambiare la view da mostrare. Non ho ancoea scoperto quale sia l'errore, quindi sarà la prima cosa in programma per la prossima lezione.

Punto della situazione rispetto alla pianificazione
Rientro nei tempi della pianificazione.

Programma di massima per la prossima giornata di lavoro
Per prima cosa devo risolvere l'errore descritto sopra, poi potrò fare qualche test tra le due view. Il tempo della prossima lezione sarà occupato principalmente dalla creazione del database.