Diario di lavoro

Luogo	SAM Trevano
Data	12.12.2019

Lavori svolti

Per prima cosa oggi ho modificato il costruttore di Qdc, aggiungendo I campi che avevo aggiunto qualche lezione fa (perito 2, percorso files). Quindi ho aggiunto questi valori anche nella insert eseguita in QdcViewModel. Subito dopo ho aggiunto queste due righe nel costruttore di QdcView (xaml.cs):

```
DataInizio.SelectedDate = DateTime.Today;
DataConsegna.SelectedDate = DateTime.Today;
```

questo perché altrimenti all'avvio della maschera veniva visualizzata 01.01.0001 come data di default dei campi DataInizio e DataFine invece che la data odierna. Ho inoltre aggiunto un formato specifico ai due campi per l'inserimento degli orari di lavoro:

```
StringFormat=HH:mm
```

A questo punto ho cominciato a inserire un sistema di controllo dei campi del form prima di eseguire l'inserimento. Nel metodo generato dal bottone ho inserito un if che controlla che i campi non siano ne vuoti ("") ne null:

```
if (!string.IsNullOrEmpty(Titolo) && !string.IsNullOrEmpty(NomeFormatore) && . . .
```

Ho eseguito questo controllo per tutti i campi. Per i campi di tipo diverso da string ho semplicemente messo un ToString():

```
!string.IsNullOrEmpty(OraInizio.ToString())
```

Se qualcuno di questi campi è vuoto viene generato un messaggio di errore:

```
MessageBox.Show("Inserisci tutte le informazioni");
```

Ho poi eseguito lo stesso tipo di controllo per la descrizione nel metodo OnSaveDescrizione, generato quando viene premuto il pulsante per salvare appunto la descrizione:

```
if (!string.IsNullOrEmpty(Descrizione))
```

I controlli dei requisiti li avevo già implementati la scorsa lezione. Il metodo che ho usato è diverso ed è descritto nel diario del 10.12.2019.

Dopo aver finito con i controlli sono ritornato sul salvataggio dei dati nel file Word. Quello che avevo fatto fin'ora lo avevo commentato e lasciato in sospeso. Avevo scritto il codice per salvare le informazioni principali di un Qdc nel metodo OnClick del bottone che salva le informazioni (in QdcView). Ora ho spostato e modificato questo codice nel ViewModel.
Per prima cosa ho modificato il metodo OnSaveInfo, ovvero il metodo generato dal pulsante in QdcView. Ho aggiunto i controlli necessari per i percorsi dei file:
se il percorso del file di template è uguale a quello del file di salvataggio viene generato il seguente messaggio:

```
if(PathTemplate == PathSave)
{
    MessageBox.Show("Inserisci un percorso differente per il salvataggio");
}
```

Dopodiché controllo che il percorso del file in quale salvare i dati non esista già, per evitare una sovrascrittura:

```
if (File.Exists((string)PathSave))
{
    MessageBox.Show("Il percorso del file di salvataggio esiste già")
}
```

Infine eseguo un try-catch sulla copia del file, se viene intercettato un errore stampo il seguente messaggio:

```
MessageBox.Show("Errore, il percorso inserito non è valido");
```

Poi ho aggiunto nel ViewModel il metodo che si occupa di scrivere le informazioni sui file. Ho creato il metodo SaveInFile che riceve un Qdc come parametro:

```
private void SaveInFile(Qdc q)
{
    var pathTemplate = q.PathTemplate;
    var pathSave = q.PathSave;
    var titoloKey = "<Titolo>";
    var nomeFormatoreKey = "<NomeFormatore>";
    . . .
    var descrizioneKey = "<Descrizione>";

    DateTime OraInizioAdj = DateTime.Parse(q.OraInizio.ToString());
    OraInizioAdj.ToString("HH:mm");
    DateTime OraFineAdj = DateTime.Parse(q.OraFine.ToString());
    OraFineAdj.ToString("HH:mm");

    try
    {
        object filename = pathSave;
        // Se il file esiste
        if (File.Exists((string)filename))
        {
            object readOnly = false;
            object isVisible = false;
            wordApp.Visible = false;
            //  apertura del file copiato
            aDoc = wordApp.Documents.Open(ref filename, ref missing,
                ref readOnly, ref missing, ref missing, ref missing,
                ref missing, ref missing, ref missing, ref missing,
                ref missing, ref isVisible, ref missing, ref missing,
                ref missing, ref missing);
            aDoc.Activate();
            // richiama la funzione FindAndReplace passandogli due parametri,
            // il testo da sostituire e con cosa sostituirlo
            this.FindAndReplace(wordApp, titoloKey, q.Titolo);
            this.FindAndReplace(wordApp, nomeFormatoreKey, q.NomeFormatore);
            . . .
            this.FindAndReplace(wordApp, descrizioneKey, q.Descrizione);

            //  salva il file
            aDoc.Save();
            wordApp.Quit();
        }
        else
        {
            MessageBox.Show("File does not exist.", "No File", MessageBoxButton.OK);
        }
    }
}
```

Questo metodo è stato già descritto in uno dei precedenti diari. Le piccole differenze sono che per scrivere all'interno del file word invece di prendere il valore dai TextBox li prendo direttamente dal database. Inoltre ho dovuto inserire un metodo per ricavare solamente l'orario dai campi OraInizio e OraFine. (ovviamente per la scrittura sul file è presente anche il metodo **FindAndReplace** che ho già descritto in passato). Questo metodo viene richiamato dopo aver eseguito l'Update della descrizione:

```
SaveInFile(q);
```

Alla fine ho fatto la stessa cosa anche per i requisiti. Ho inserito gli stessi due metodi con qualche piccola differenza dovuta alla differenza dei dati che vanno inseriti.

Problemi riscontrati e soluzioni adottate

Dopo aver eseguito qualche test ho notato che la scrittura sul database delle date e degli orari non funzionava correttamente. Le date venivano salvate sempre con il valore 01.01.0001, qualsiasi valore inserivo nel campo (che era di tipo DatePicker). Per risolvere l'errore ho dovuto cambiare i campi in TextBox e inserire questa riga nel costruttore della View:

```
DataInizio.Text = DateTime.Today.ToString().Remove(10);
```

In questo modo nel campo viene inserita la data di oggi (rimuovendo l'orario che è un campo a sé stante). Nel caso venisse scritto un valore con un formato non valido viene salvata la data corrente.
Il metodo che ho usato per gli orari è simile. Gli orari erano già inseriti in dei TextBox. L'errore in questo caso era che se non veniva modificato l'orario di default (00:00) sul database veniva salvato come 01.01.0001 00:00. Per risolvere questo errore prima di eseguire l'inserimento dal ViewModel ho inserito questo if che inserisce la data odierna:

```
if (OraInizio == DateTime.MinValue)
{
    OraInizio = DateTime.Today;
}
```

Ho ancora diversi problemi non risolti:

- Le date devono essere scritte nel formato americano per essere salvate correttamente sul database.
- Gli orari non vengono scritti correttamente sul file Word: viene salvata sia la data che l'ora.
- Questa lezione non ho avuto ancora il tempo di risolvere l'errore della scrittura di un TextBox direttamente dal ViewModel.

Punto della situazione rispetto alla pianificazione

Sono abbastanza in ritardo. Dato le diverse ore di assenza ho cominciato a lavorare un po' a casa per cercare di recuperare il più possibile.

Programma di massima per la prossima giornata di lavoro

La prossima lezione sarà interamente dedicata alla documentazione. Gli errori da risolvere cercherò di risolverli a casa.