Gestione file in C#

Prima parte

Una delle attività più importanti nella programmazione, è quella di poter gestire i file. La gestione di file significa fondamentalmente poter

1. leggere un file e visualizzarne il contenuto
2. scrivere un file per memorizzare informazioni
3. aggiungere informazioni ad un file già esistente

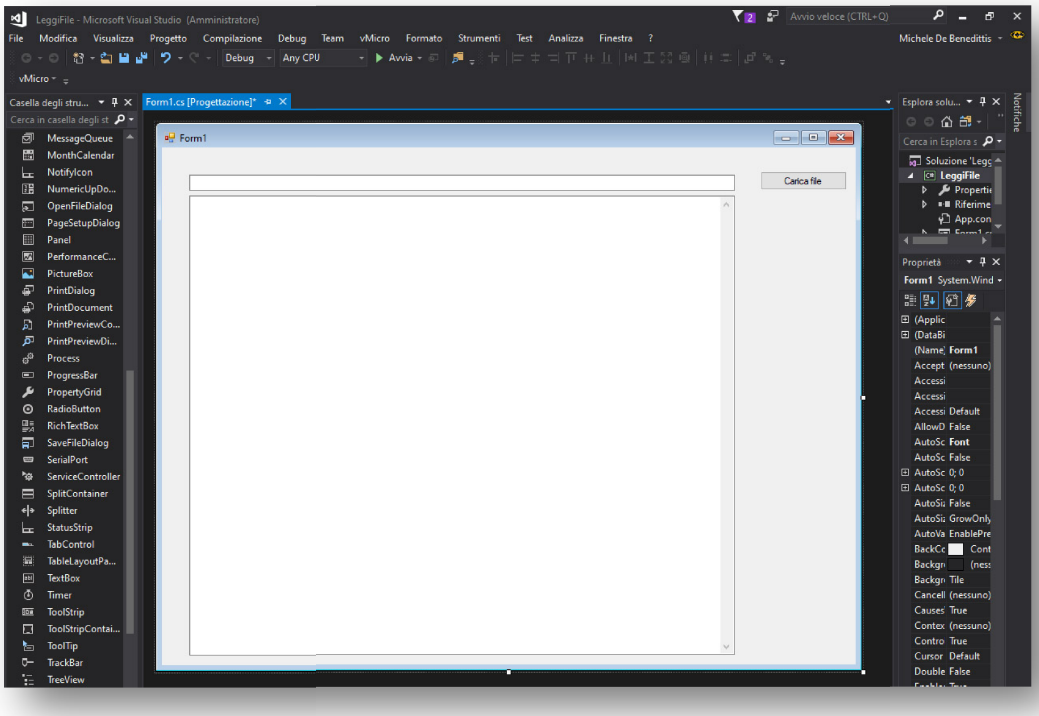
Ora il primo passo è quello di creare una applicazione semplice che ci consente la lettura di un file di testo e visualizzare tutto il suo contenuto.

Con Visual Studio, creiamo un nuovo progetto e lo chiamiamo 'LeggiFile'.

Aggiungiamo sul nostro form tre controlli:

1. un TextBox per l'inserimento del nome del file da leggere (*txtFile*)
2. un Button per avviare la lettura del file (*btnLoad*)
3. un altro TextBox (multi linea) per la visualizzazione del contenuto del file (*txtContenutoFile*)

Il form potrebbe essere come questo:



Per una corretta visualizzazione del contenuto del file, bisogna impostare due proprietà del secondo TextBox (*txtContenutoFile*):

1. **Multiline** impostata su '**True**', consente la visualizzazione su più righe

2. **ScrollBars** impostata su '**Vertical**', consente la visualizzazione di una barra di scorrimento verticale nel caso in cui il contenuto ecceda la grandezza del controllo

Portiamoci ora nella finestra del codice; per lavorare con i file, è necessario inserire il namespace '**System.IO;**', ai namespaces esistenti, senza il quale non possiamo accedere ai nostri files.

Ora programmiamo il nostro Button.

Come al solito, clicchiamo due volte sul pulsante in modo che Visual Studio possa inserire il codice necessario per la gestione dell'**evento Click** del bottone *btnLoad*:

```
private void btnLoad_Click(object sender, EventArgs e)
 {
      ……
 }
```

Parleremo un giorno di come vengono creati gli eventi senza l'aiuto di Visual Studio. Si tratta di un argomento di programmazione avanzata.

Per la lettura di un file, utilizzeremo una classe 'statica', cioè di una classe che non verrà istanziata quindi si potrà utilizzare direttamente. Questa classe è '**File**'.

Uno dei metodi (statici) è

ReadAllText(filepath)

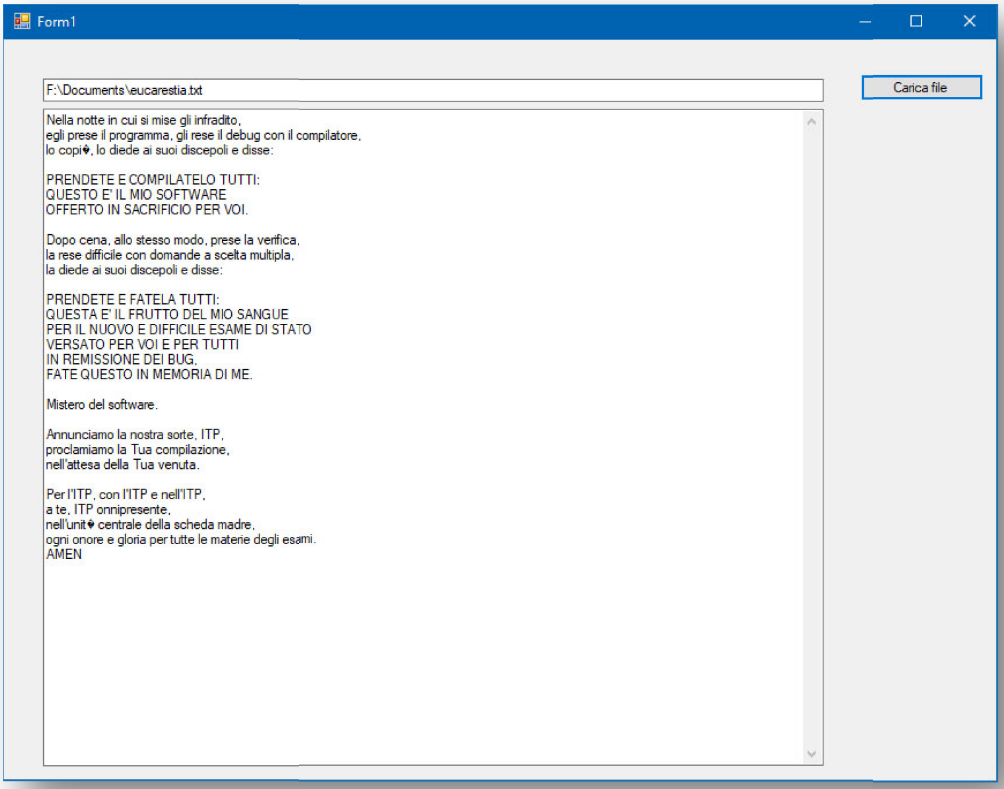
dove l'argomento '*filepath*', di tipo string, rappresenta il percorso del file da leggere. Tale percorso lo inseriremo nel primo TextBox e, nell'evento Click, lo passeremo come argomento alla classe **File**. Così:

```
private void btnLoad_Click(object sender, EventArgs e)
 {
     txtContenutoFile.Text = File.ReadAllText(txtFile.Text);
 }
```

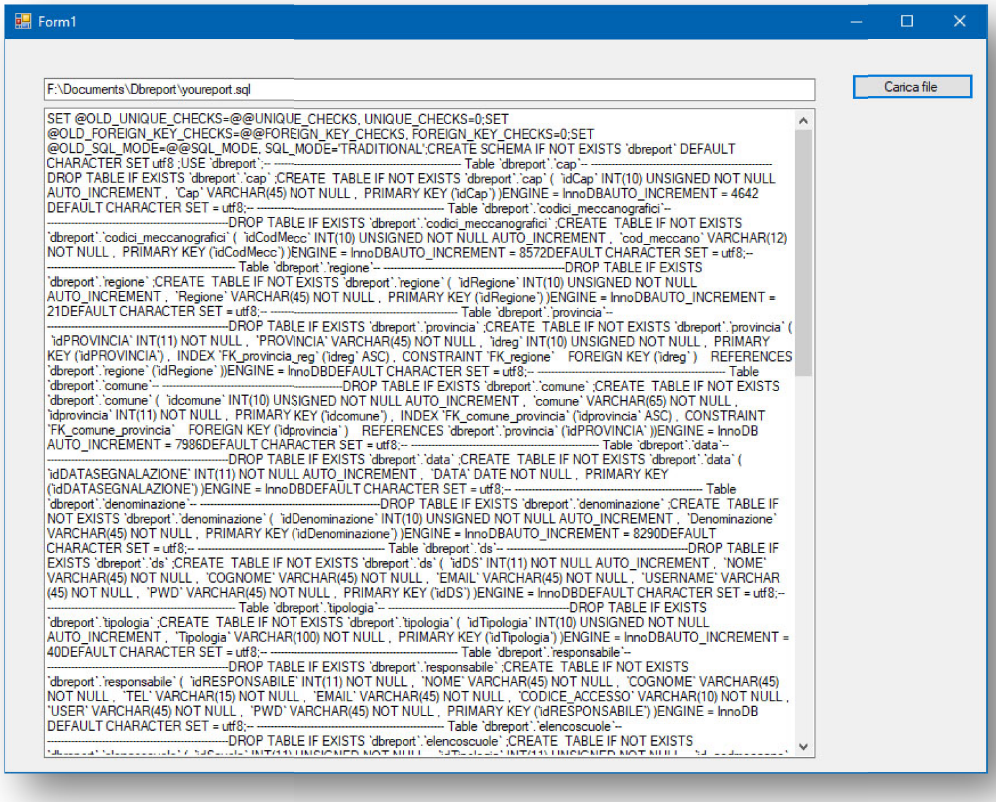
Lanciamo ora l'applicazione.

Cercate un file di testo all'interno del vostro PC, copiate il percorso con il nome del file, inseritelo nel primo TextBox, cliccate sul Button.

In figura un mio esempio:



Ora provo con un file di testo abbastanza lungo:

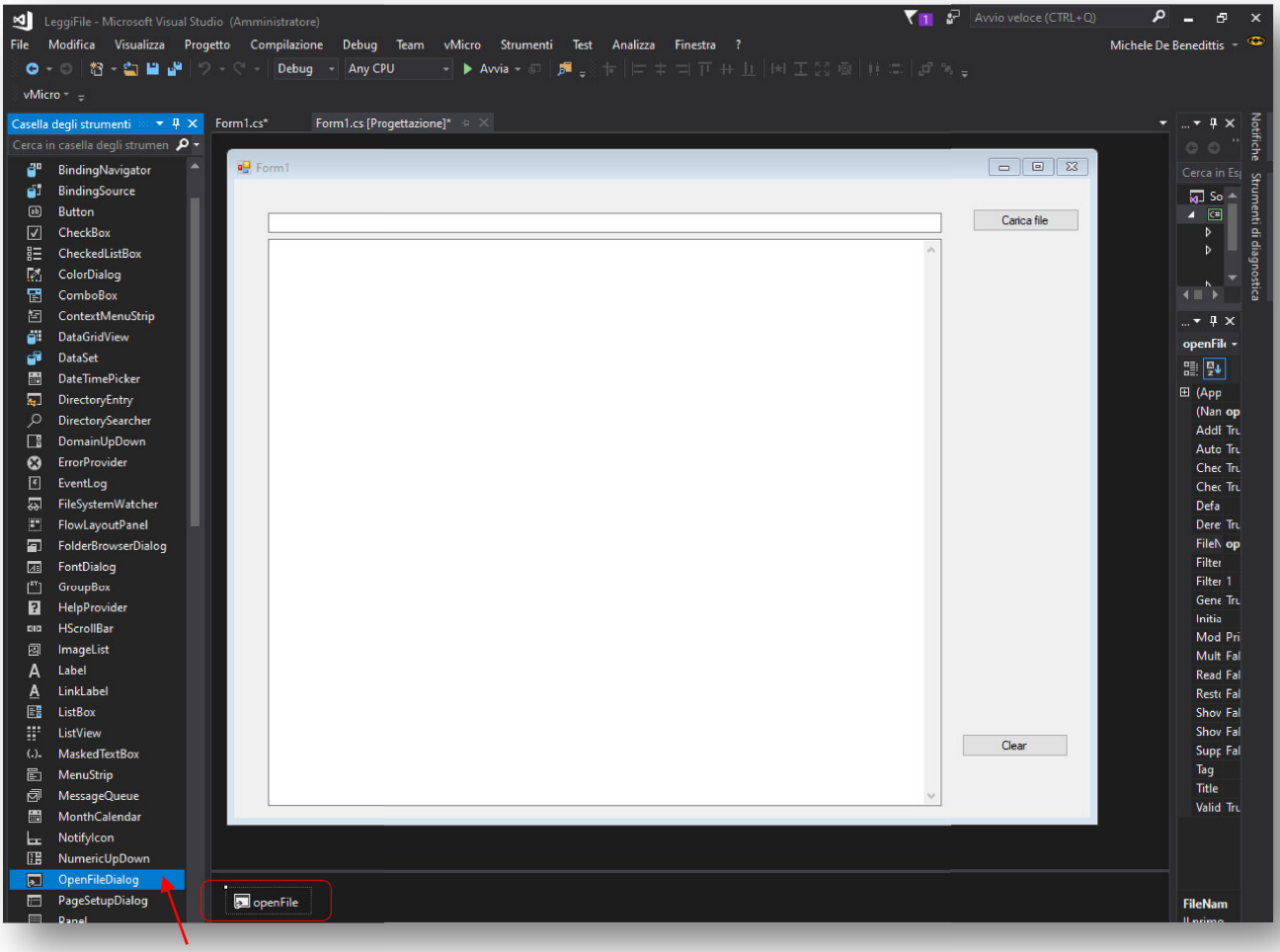


come vedete, è comparsa la barra di scorrimento.

Ma ora è il caso di complicarci un po' la vita con l'aggiunta di un altro componente. Il controllo è:

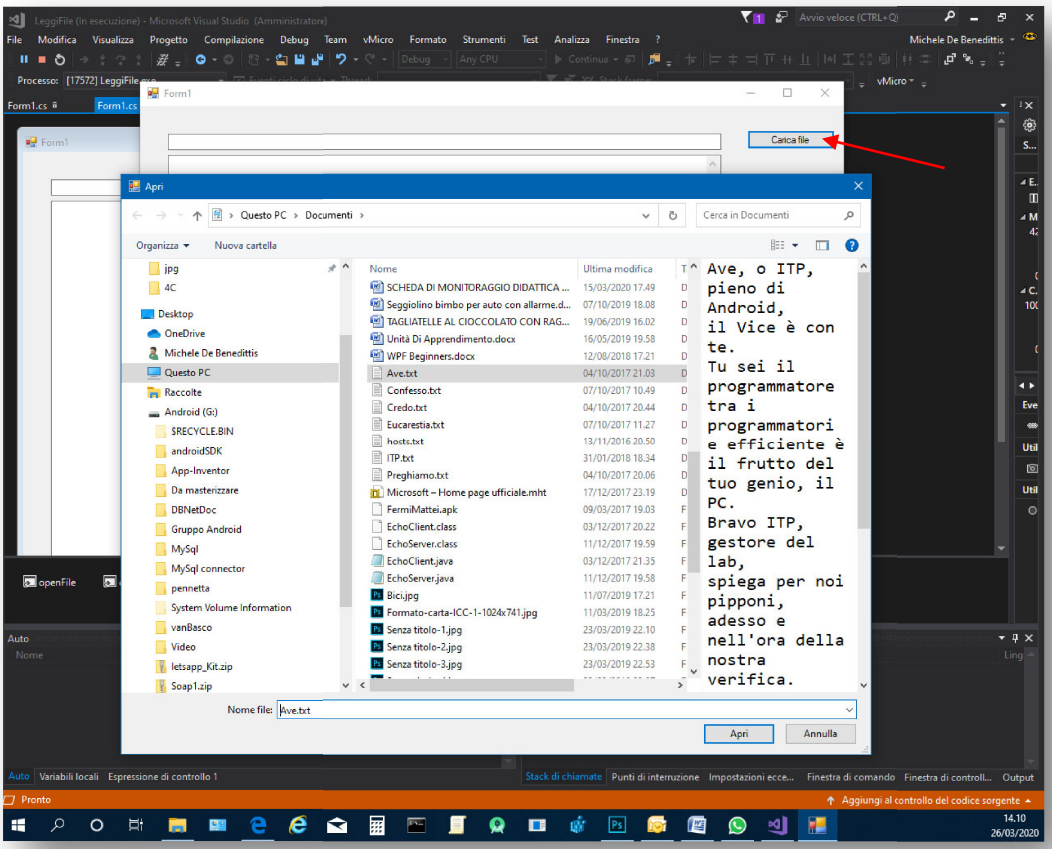
OpenFileDialog

che non è un controllo visibile perché non ha una interfaccia grafica, ma consente di esplorare il nostro HD per la selezione di un file da aprire:

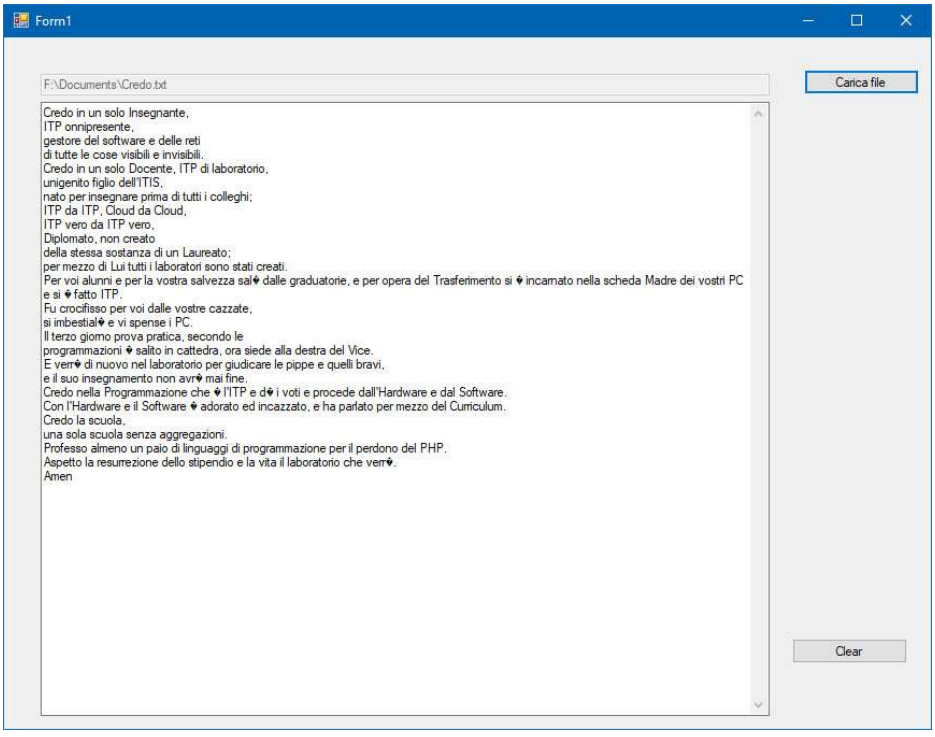


Selezionate il controllo e trascinatelo sul form. Come al solito gli ho cambiato la proprietà '**(Name)**' e l'ho chiamato '**openFile**' come da riquadro rosso. Poi ho aggiunto un secondo Button, l'ho chiamato '**btnClear**', che mi consente di 'pulire' il primo ed il secondo TextBox da eventuali azioni precedenti.

Nelle seguenti due immagini, l'esecuzione:



Dopo aver cliccato sul Button ‘Carica file’, si apre la finestra di dialogo per la selezione del file dal mio HD. Scelgo il file e lo visualizzo:



Questo è il codice dell'evento Click:

```
      private void btnLoad_Click(object sender, EventArgs e)
        {
            if(((Button)sender).Text== "Carica file")
            {
                openFile.ShowDialog();
                txtFile.Text = openFile.FileName;
                txtContenutoFile.Text = File.ReadAllText(txtFile.Text);
                txtFile.Enabled = false;
            }else
            {
                txtFile.Clear();
                txtContenutoFile.Clear();
                txtFile.Enabled = true;
            }
        }
```

Come si vede, l'evento Click, gestisce entrambi i Button, non ho scritto due gestori separati. Lascio a voi capire come ho fatto (è semplicissimo).