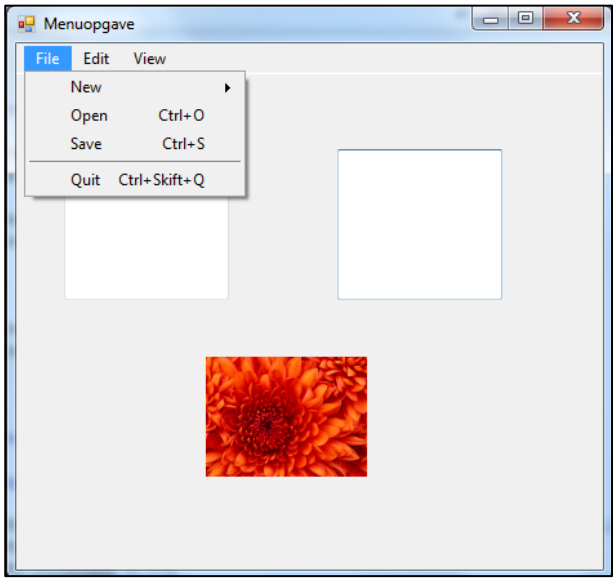
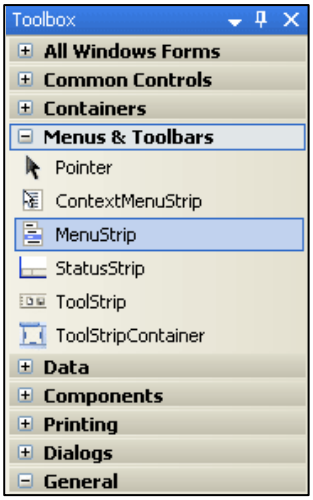
Kapitel 5 Tilføj menuer til Windows formular

I dette kapitel skal vi se hvordan man tilføjer menuer til din formular. Du skal tilføje en File, Edit og View menu. Alle menupunkterne skal indeholde punkter og endvidere undermenuer. Her er hvad vi skal lave.

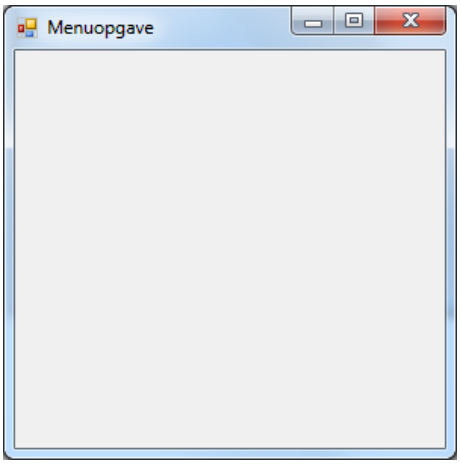


Start et nyt projekt ved at vælge File og New Project. Opret en ny **Windows Application**. Giv projektet et passende navn. Når din nye formular dukker op kan du tilføje en menulinje relativt nemt.

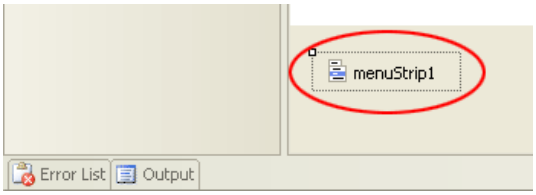
Se på Toolbox'en i venstre side af programvinduet. Ligesom der er en sektion der hedder Common Controls, er der en sektion til **Menus and Toolbars**. Klik på plus tegnet ved siden af for at se følgende:



Den du skal bruge er **MenuStrip**, som er fremhævet i billede ovenfor. Dobbeltklik på MenuStrip og du vil se en menu bar der vises I toppen af formularen:

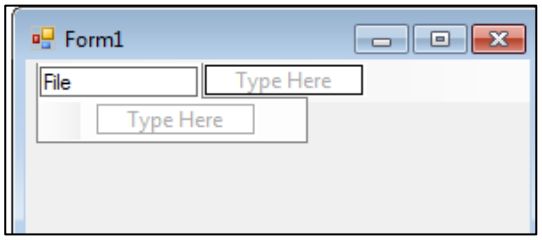


Bemærk også hvad der dukker op i bunden af programvinduet:

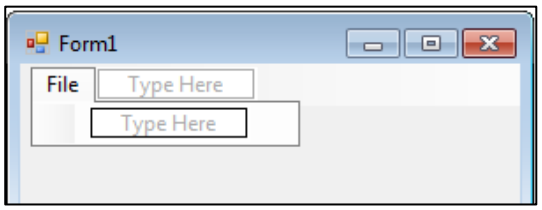


Det er MenuStrip objektet. Standardnavnet for MenuStrip er **menuStrip1**. Hvis din MenuStrip ikke er markeret, kan du klikke på ikonet i bunden. Når du gør dette vil betingelserne for MenuStrip vises over i Properties vinduet i højre side af programvinduet.

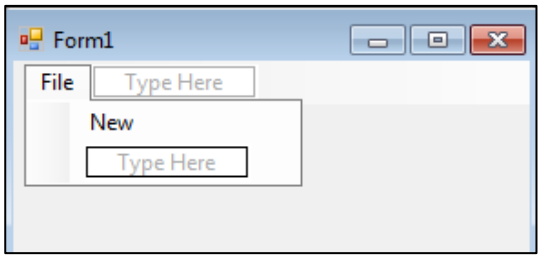
Det er ganske nemt at tilføje menupunkter. Klik inden i området i toppen hvor der står ”Type Here”. Skriv navnet **File**.



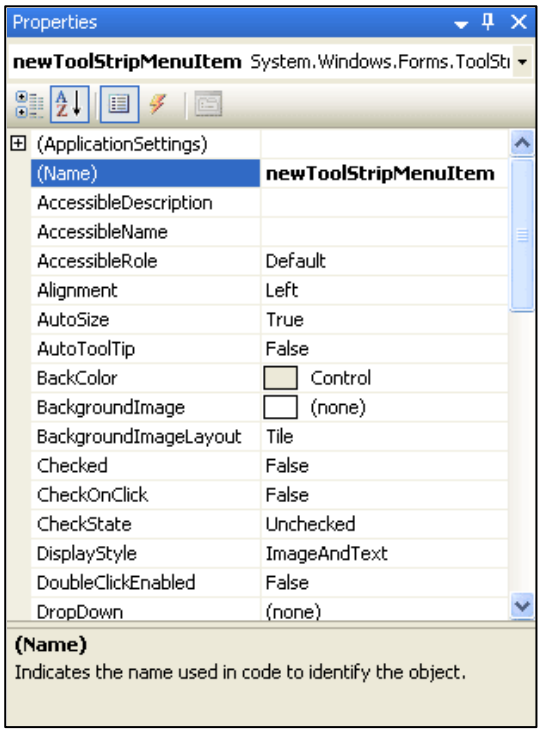
Tryk på Enter tasten og din menuen vil se sådan ud:



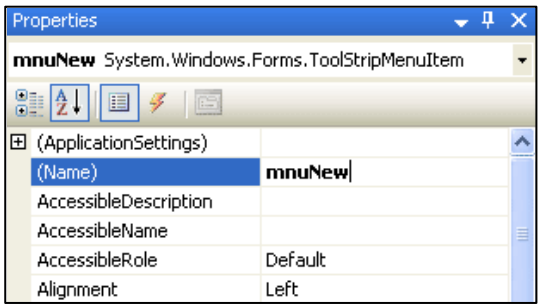
Du har nu oprettet et hovedmenupunkt. For at tilføje punkter til din File menu, kan du klike inden i den anden ”Type Here” som vist ovenfor. Tast nu ordet **New**. Tryk på Enter tasten for at tilføje et menupunkt:



Klik tilbage på ordet New efter du har trykket på Enter tasten. Dette vil markere netop dette menupunkt og ingen andre. Først når du har oprette et menupunkt har det dets egen Properties som du kan ændre på. Med punktet New markeret. Skal du se nærmer på vinduet Properties i højre side af programvinduet.

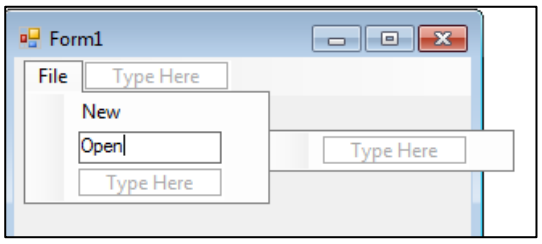


Det er **Name** betingelsen vi er interesseret i. Det er for langt. Skift det til **mnuNew** som vist i billedet nedenfor:



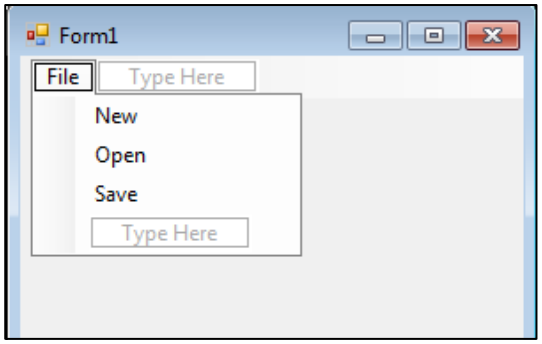
Hvis du ruller ned vil du også kunne se Text betingelsen. Der står New. Det er det du tastede ind menulinjen da du oprettet punktet. Du kan godt skifte det her hvis du hellere vil det. Men lad bare Text betingelsen blive ved New.

Klik tilbage i menuen toppen af din formular og stil dig i ”Type Here” lige under New. Skriv ordet Open:

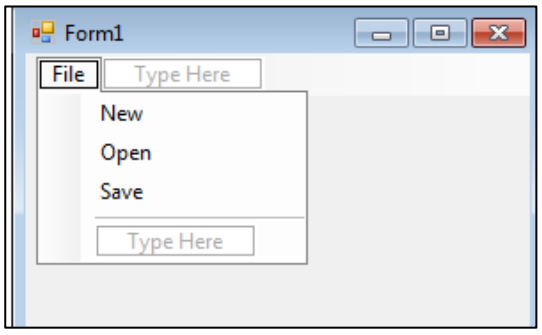


Tryk på Enter tasten for oprette menupunktet Open. Skift betingelsen **Name**, som du gjorde før med New, til **mnuOpen**.

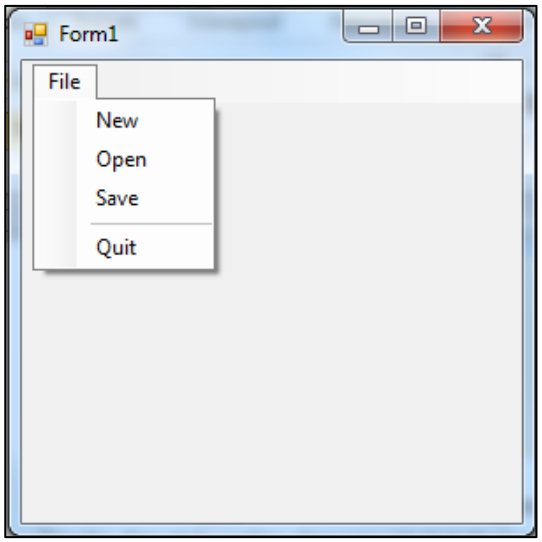
Opret et menupunkt med navnet Save under Open. Skift **Name** betingelsen til **mnuSave**. Din File menu vil nu se sådan ud:



Vi vil oprette endnu to menupunkter, en adskiller streg, og et menupunkt der hedder Quit. Når du skal oprette en adskiller linje, klikker du i ”Type Here” nedenfor Save. Tast nu en bindestreg. Når du taster Enter vil C# automatisk lave bindestregen om til en adskiller streg. Det burde se sådan ud:

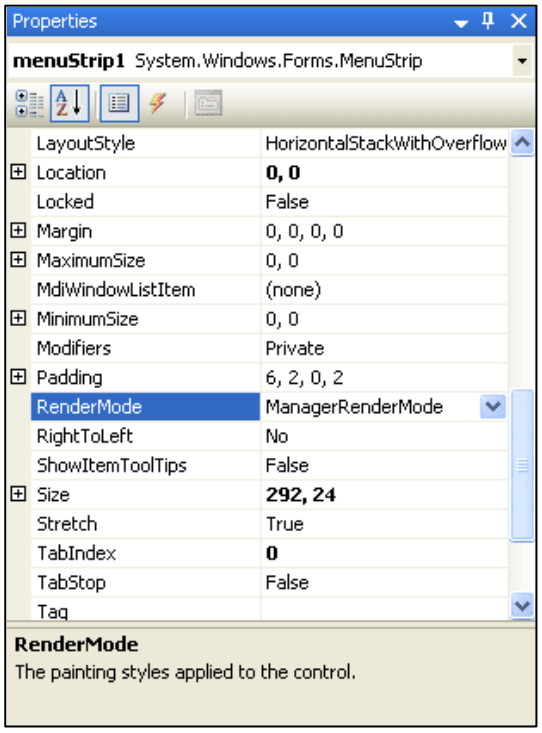


Tilføj punktet **Quit** nedenfor adskiller stregen. Skift betingelsen Name til **mnuQuit**. Din File menu er nu færdig. For at se hvordan den ser ud, skal du køre programmet. Du burde have en blå menulinje i toppen af dit programvindue med en File menupunkt. Klik på File i menuen:

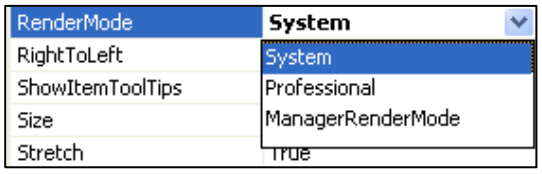


Selvfølgelig er der ikke noget af det der virker endnu, da vi ikke har skrevet noget kode endnu. Det vil vi dog snart gøre. Hvis du ikke kan se en blå menu linje i toppen af dit vindue, kan vi hurtig ændre på det. Klik på det røde X for at stoppe dit program og vend tilbage til C#. Klik et vilkårligt sted i den blå menu. Eller klik på **menuStrip1** i bunden af skærmen.

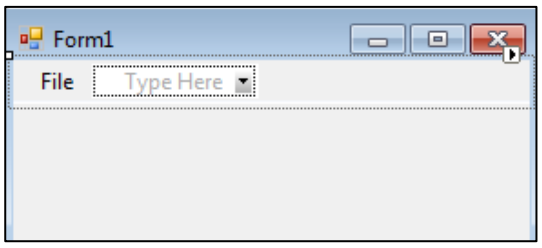
Når MenuStrip er markeret, skal du se på dets betingelser i Properties vinduet. Find den betingelse der hedder **RenderMode**:



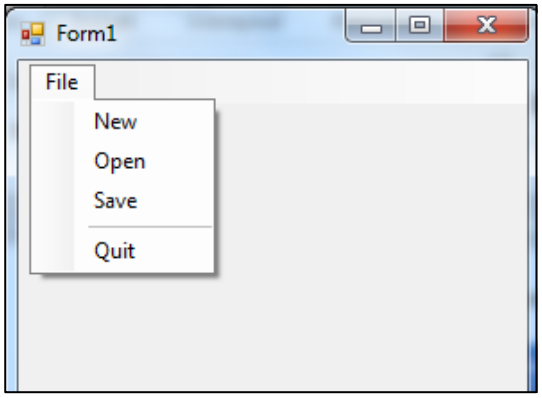
Klik på listepilen for at se de to andre muligheder. Professional gør ikke særligt meget. Vælg derfor System.



Tag nu et kig på din MenuStrip. Den skulle gerne have ændret sig til dette:



Når du kører dit program ser det ud på følgende måde:

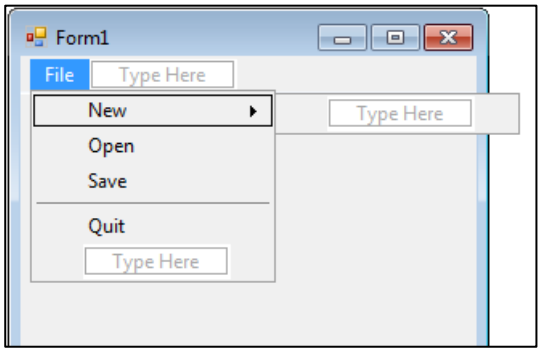


I den næste del skal du lære hvordan du opretter undermenuer.

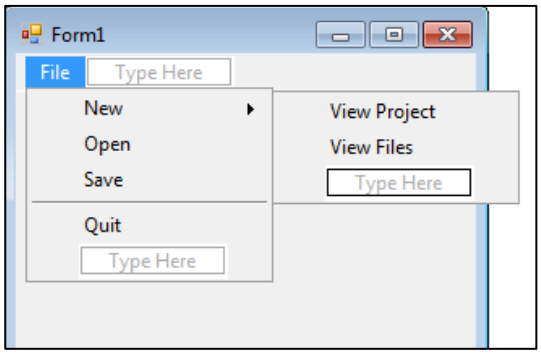
Undermenuer i C#

Det er lige så let at oprette undermenuer. En undermenu er det der dukker op fra et hovedmenupunkt.

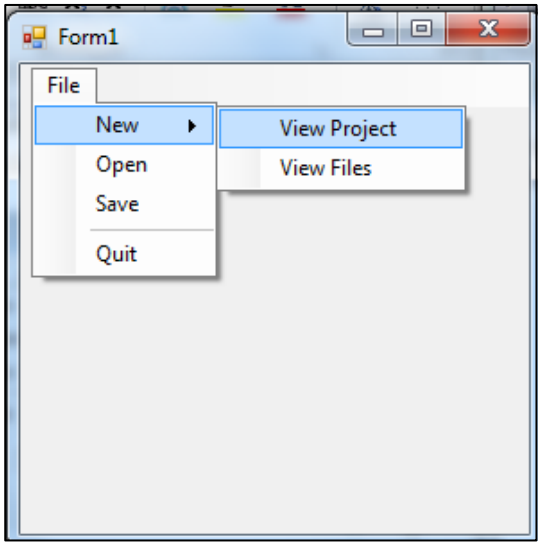
Stop dit program og vend tilbage til din formular. Klik på punktet New for at vælge det. Du brude se en boks ”Type Here” på højre side af New:



Klik indenfor og skriv View Project. Tryk på Enter tasten og skriv View Files i boksen nedenunder. Din menu vil nu se sådan ud:



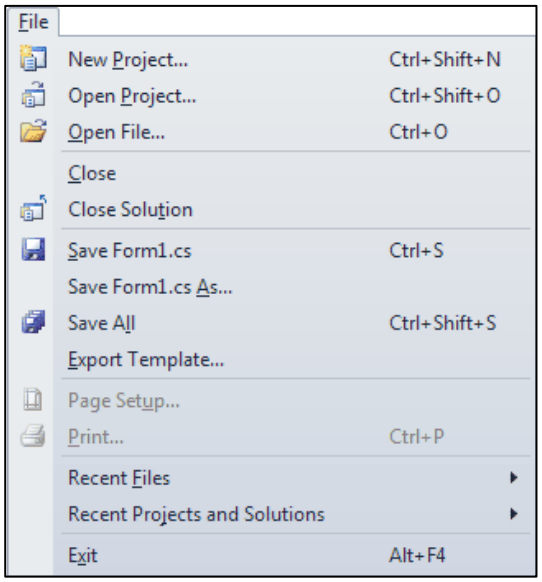
Når du kører dit program vil undermenuen så sådan ud:



Undermenuer er ganske nemme at oprette! I den næste del skal vi lære at oprette genvej til dine menupunkter.

Genveje til menupunkter i C#

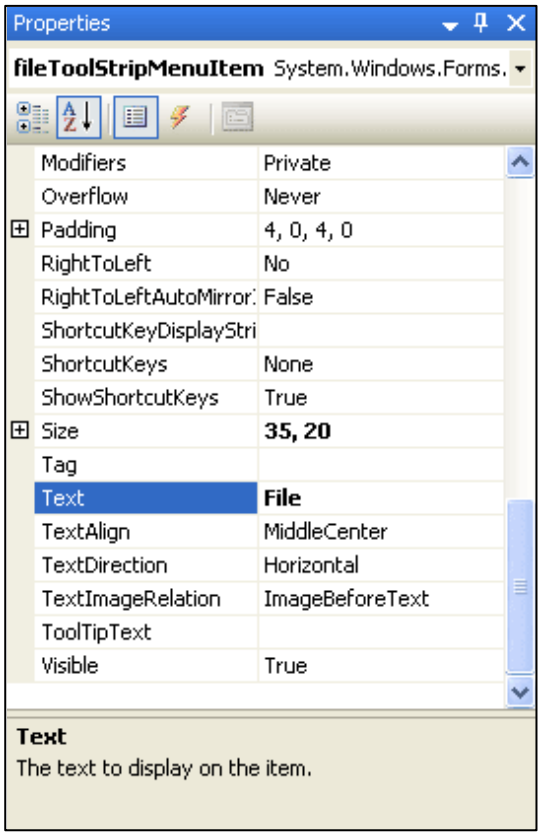
Menuer har som regel genveje. Disse er de understregede bogstaver du ser, når du klikke i en menu. De har indimellem genvejstast på højre side af menupunktet. Som eksempel kan vi se på menupunktet File i C# og alle de understegninger der er og deres genvejstaster:



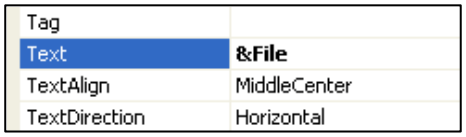
For at se dem skal du trykke på ALT tasten. Når du ser et understreget bogstav taster du denne tast kombinere med bogstavet der er understreget. Hvis du taster ”F” vil du f.eks. se at menuen ruller ud. Hvis du taster et af de understreget ord i File menuen vil du starte det pågældende menupunkt.

Du kan også bruge genvejstasten i højre side af menupunktet. Hold CTRL + SHIFT + N neden for at åbne dialogboksen New Project.

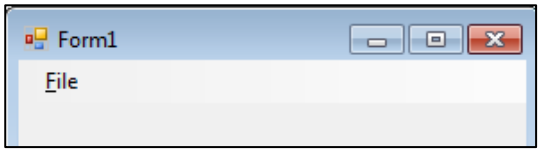
For at tilføje genveje til dine egen menuer, skal du klikke på punktet File for at vælge det. Tag nu et kig på dets betingelser i Properties vinduet. Rul ned til du ser punktet text:



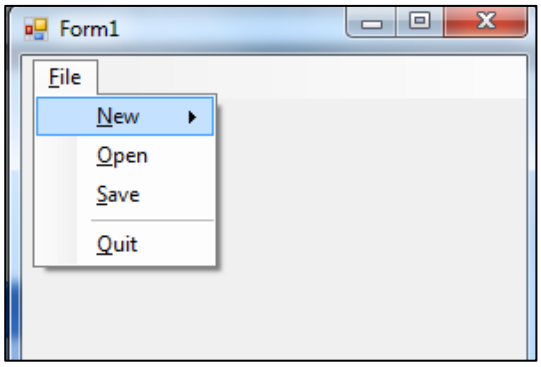
For at lave understregning til et bogstav, skal du skrive et ampersand symbol (&) foran bogstavet du vil benytte som genvej. I billedet nedenfor har vi tilføjet et & lige før ”F” i File:



Og sådan kommer menuen til at se ud med & tilføjet:

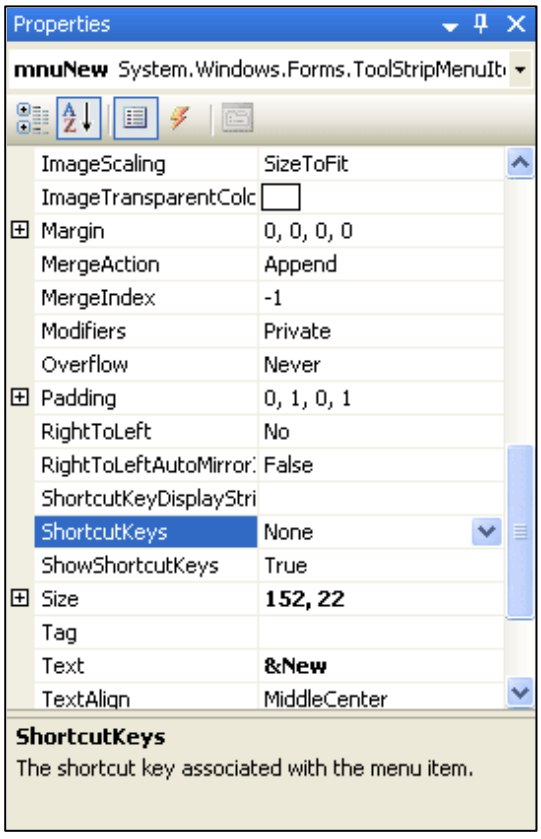


Som du kan se er der nu en understregning under bogstavet ”F”. I det næste billede har vi tilføjet flere understregninger i menuen File:

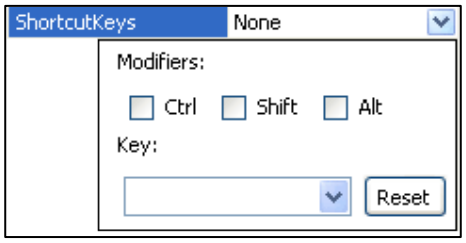


Tilføj de samme understregninger til dine egen File menu. Husk at klikke på menupunktet for at vælge det, find derefter text betingelsen, og tilføj et & symbol foran bogstavet du vil bruge som genvej. Når du kører programmet skal du huske at taste ALT, ellers vil du ikke se de understreget bogstaver.

Genvejstasten er ligeså nem at tilføje. Klik på menupunktet **New** for at vælge det. Find betingelsen **ShortcutKeys** i Properties vinduet:

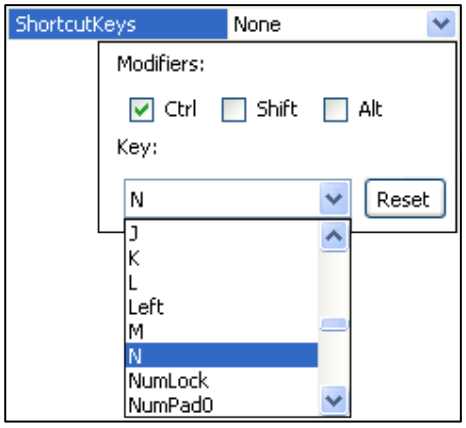


Lige nu er den sat til None. Klik på listepilen for at se følgende muligheder:

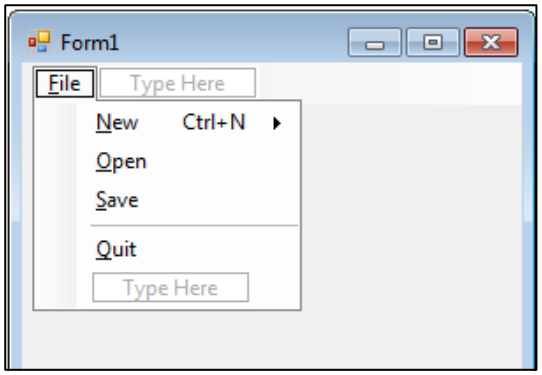


Mulighederne er CTRL, SHIFT og ALT tasten. Du kan vælge en eller flere af dem. For at aktivere en genvej, skal du holde disse taster nede først. Hvis du derfor vil have brugeren til at holde CTRL og SHIFT tasten nede plus et bogstav eller symbol, skal du afmærke disse muligheder i boksen ovenfor.

Bogstaver og symboler kan findes Key listefeltet. Klik på listepilen for at se følgende:

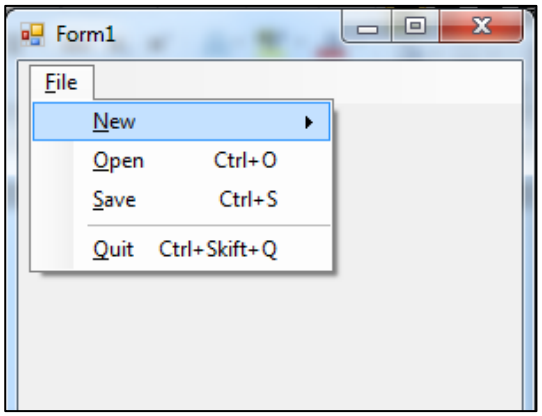


I billedet ovenfor har vi valgt CTRL og bogstavet ”N”. Når du klikker tilbage i menuen ser du følgende:



Som du kan se er genvejene til menupunktet New en understregning og CTRL + N.

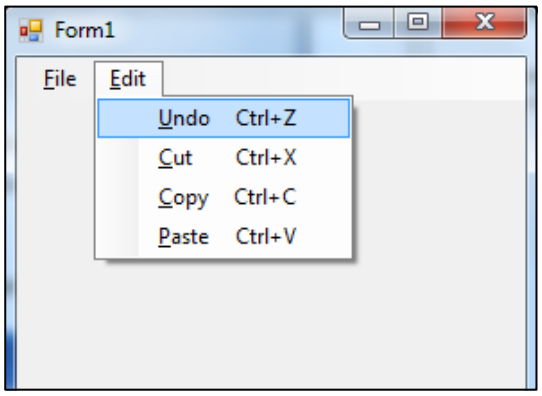
Så på det næste billede og tilføj de samme genvejstaster til din File menu:



Dem du skal tilføje er de sidste tre: Open, Save og Quit. Vi skal snart se på kodningen af menupunkterne, men her en lille øvelse du først skal lave.

Øvelse

Tilføj menupunktet **Edit** med følgende punkter:

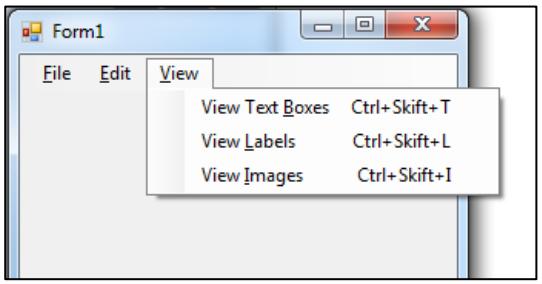


Opret også de viste genvejstaster. Betingelsen Name skal for de enkelte punkter være:

Undo: mnuUndo
Cut: mnuCut
Copy: mnuCopy
Paste: mnuPaste

Øvelse

Opret menupunktet **View** til din menulinje med følgende punkter:



Du skal igen tilføje genvejstaster. Betingelsen Name for de enkelte punkter skal være:

View Text Boxes: mnuViewTextBoxes
View Labels: mnuViewLabels
View Images: mnuViewImages

Nu er det på tide at vi får skrevet lidt kode til de forskellige menupunkter.

C# kode til din Quit menu

Det er klart at det ikke hjælper meget med en menu, hvis der ikke sker noget, når du klikke på de forskellige punkter. Vi skal derfor tilføje kode til de forskellige menupunkter. Vi starter med punktet Quit, som gerne skulle være i din File menu. Der skal kun bruges en linje med kode til dette punkt.

Vend tilbage til din formular, og klik på menulinjen. Klik på punktet **File** for at se det menu. Dobbeltklik på punktet Quit og du ser nu kodevinduet. Din markør står og blinker i mellem de to bløde parenteser i koden til Quit:

```
private void mnuQuit_Click(object sender, EventArgs e)
{
    |
}
```

Bemærk at betingelsen Name du gav menupunktet bliver brugt i koden: **mnuQuit**. Når en bruger klikker på menupunktet Quit, vil du gerne have at programmet afsluttes. For at lukke en Windows applikation, kan du bruge følgende:

Application.Exit( );

Tilføj denne linje mellem de to bløde parenteser i koden til Quit. Kør programmet og se om det virker. Benyt tasterne CTRL og SHIT og bogstavet Q. Programmet skulle gerne afsluttes med det samme. Det gør det på grund af genvejstasten du angav.

For at se dine understregninger i brug, starter du dit program igen. Tast ALT og du skulle gerne se alle understregningerne dukke op for File, Edit og View. Tast ”F” og menu folder sig ud. Tast nu ”Q” og programmet skulle gerne lukke ned.

Du kan tilføje mere kode til menupunkterne – faktisk alt hvad du ønsker. Du ser under tiden en dialogboks der dukker op når Quit punktet bruges:

"Are you sure you want to Quit?"

For at tilføje en dialogboks til din kode kan du prøve følgende:

```
{
if (MessageBox.Show("Really Quit?", "Exit", MessageBoxButtons.OKCancel) == DialogResult.OK)
Application.Exit();
}
```

Dette vil vise dig en dialogboks med OK og Annuller knapper. C# venter på at brugeren klikker på en af knapperne. Vores kode bruger en if sætning for at undersøge hvilken knap der er blevet klikket på. For at se hvilken knap der er klikket på tilføjer vi følgende i slutningen af linjen:

== DialogResult.OK

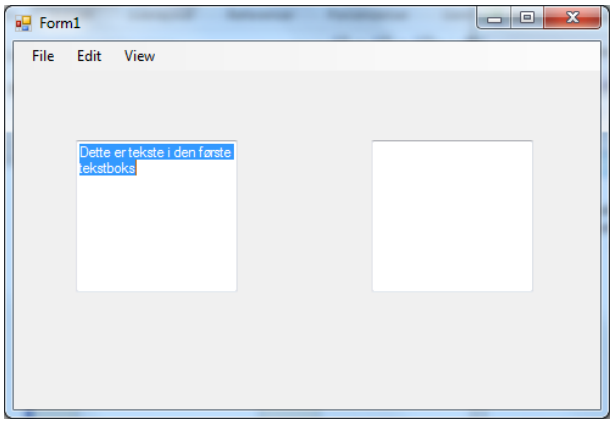
Vi læser linjen som "IF (Hvis) resultatet af dialogboksen er OK da afsluttes Applikationen."

I den næste del skal vi se på koden til menupunktet Edit.

Edit menuen

Vi gemmer resten af File menuen til sidst, da det er lidt mere kompliceret. Vores Edit menu er ikke så svær, da der kun er en linjes kode til hvert punkt.

For at se Cut, Copy og Paste i aktion, skal vi tilføje to tekstbokse til vores formular. Sæt betingelsen **MultiLine** til true for begge bokse. Det ville tillade at vi kan tilføje mere end en linje tekst. I den første tekstboks bestemmer du selv betingelsen for Text. Din formular skal gerne se sådan ud:



Vi skal nu aktivere menupunkterne Cut, Copy og Paste, samt Undo. Vi starter med at benytte Cut (Klip) på den markerede tekst i den første boks og derefter vil vi Undo (Fortryde) handlingen.

Vend tilbage til din formular. Klik på den blå menulinje i toppen og klik på menupunktet Edit. Dobbeltklik på punktet **Cut**. C# vil oprette den tilhørende kode. Hvis du gav punktet Cut Name betingelsen **mnuCut** vil din kode se sådan ud:

```
private void mnuCut_Click(object sender, EventArgs e)
{
    |
}
```

Koden der skal til for at klippe en hvilken som helst markeret tekst er ganske simpel. Tilføj denne linje mellem dine krøllede parenteser:

textBox1.Cut( );

Cut() er en method, der er indbygget i C#. Den virker på tekstbokse, blandt mange andre ting, og gør det den siger – Cuts (eller klipper!)

Før du afprøver den skal du vende tilbage til din formular og dobbeltklikke på menupunktet **Undo**. Tilføj følgende linje med kode:

textBox1.Undo( );

Prøv det nu. Kør dit program og marker teksten i din tekstboks. Benyt menuen Cut for at klippe teksten. Prøv derefter Undo menuen for at genskabe teksten.

Du kan også undersøge om der er markeret noget tekst. Ændre din kode til følgende:

```
{
if (textBox1.SelectedText != "")
textBox1.Cut();
}
```

Vi burger en if sætning til at undersøge om en betingelse af en tekstboks der hedder **SelectedText**. Det kan fortælle dig om der er markeret noget tekst. Vi bruger operatoren ”Does Not Equal” (!=) efterfulgt af to anførselstegn. Et par anførselstegn uden mellemrum betyder at det er en blank tekststreng. Hvis der ikke er markeret noget tekst er if sætningen true (sand). I det tilfælde kan Cut opgaven udføres.

Du kan manipulere med markerede tekster med betingelsen SelectedText. I koden vist nedenfor overfører vi den markerede tekst til en streng variable og viser resultatet i en meddelelsesboks.:

```
String someText;

{
if (textBox1.SelectedText != "")
someText = textBox1.SelectedText;
MessageBox.Show(someText);
}
```

I menuen Undo kan du måske være interesseret i operationen overhovedet kan udføres (fortrydes). Hvis du vil undersøge dette, er der en anden betingelse til tekstbokse man kan bruge og det er **CanUndo**. Du bruger den på følgende måde:

```
{
if (textBox1.CanUndo == true)
textBox1.Undo( );
}
```

Kun hvis operationen kan fortrydes (Undone) vil kode i if sætningen blive udført.

Men hvis du kører dit program og klipper (cut) noget tekst, vil der ske det at når du klikker på Undo to gange vil teksten først blive sat ind og slettet igen! For at undgå dette kan du slette undo operationen. Ændre din kode til dette. Den nye linje med kode er skrevet med fed nedenfor:

```
{
if (textBox1.CanUndo == true)
textBox1.Undo();
textBox1.ClearUndo();
}
```

Du tilføjer derfor ClearUndo() efter punktummet i textBox1. Prøv igen og du vil se at når du klikker på Undo to gange vil du ikke få teksten tilbage.

I næste del skal vi se hvordan du bruger Copy (kopiere) og Paste (Sæt ind) i C#.

Copy og Paste i C#

For at kopiere noget over i udklipsholderen, skal du markere teksten og klikke på menupunktet Copy i Edit. Når teksten er kopieret til udklipsholderen, kan den indsættes (Paste) et andet sted. Det vil vi lægge ind i vores menu system.

Dobbeltklik på **Copy** under menupunktet **Edit**. Du vil nu se koden der hører til Copy. Tilføj følgende kode imellem de krøllede parenteser:

```
textBox1.Copy( );
```

Når du bruger denne metode til at kopiere tekst i tekstboks vil den automatisk overføre teksten til Windows udklipsholder. Du kan dog starte med at undersøge om der er noget tekst markeret først. Ændre din kode til:

```
{
if (textBox1.SelectionLength > 0)
textBox1.Copy();
}
```

Vi gør igen brug af en if sætning. Denne gang undersøger vi længeden på det markeret. Det gør vi med tekstboks betingelsen **SelectionLength**. Den returnerer antallet af tegn der er I teksten der er i markeringen. Vi vil sikre os at den er større end nul.

Vi vil gøre brug af den anden tekstboks til at indsætte et kopieret. Vi skal derfor have fat i menupunktet Paste, hvor vi gør brug af den samme teknik som før. Tilføj følgende koden i mellem de krøllede parenteser til din Paste kode:

```
textBox2.Paste( );
```

Bemærk at vi nu gør brug af textBox2 og ikke textBox1. Efter punktummet skal du kun tilføje metoden Paste.

Prøv din Edit menu igen. Marker taksten i den første tekstboks. Klik på **Edit** og **Copy**. Klik nu i den anden tekstboks og vælg **Edit** og **Paste**.

Her kan du også undersøge om der er noget date i udklipsholderen, og om det er tekst og ikke et billede for eksempel. Tilføj denne (lidt lange) linje med kode:

```
{
if (Clipboard.GetDataObject().GetDataPresent(DataFormats.Text) == true)
textBox2.Paste();
Clipboard.Clear();
}
```

At få fat på de data der ligger i udklipsholderen, kan være lidt besværligt, men her undersøger vi formatet på den date der ligger i udklipsholderen med DataFormat. Hvis det er tekst vil vores if sætning være true og koden kan udføres. Bemærk den sidste linje:

Clipboard.Clear();

Som du måske havde forventet vil dette slette hvad der måtte være i udklipsholderen. Du behøver dog ikke den linje, så du kan slette den hvis du foretrækker dette. Undersøg hvad den gør både med og uden denne linje.

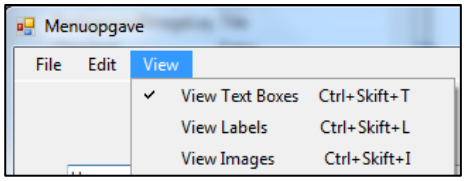
View Menuen

Vi har tre punkter i vores View menu. Vi vil dog kun implementere to af dem. I den første, View TextBoxes, skal vi se en smart programmeringsteknik med boolske variabler – hvordan man slår dem til og fra.

Vend tilbage til din formular, og dobbeltklik på menupunktet **View Text Boxes**. C# vil nu generere følgende kode til dig:

```
private void mnuViewTextboxes_Click(object sender, EventArgs e)
{

}
```

Det vi skal nu er at vi skal få tekstboksen til at skjule sig når der klikkes, og vises igen når der klikkes igen. Et ikon vises eller skjules ved siden af menupunktet. Her er en billede af det vi skal lave:



Når du skal have vist et ikon ved siden af et menupunkt, bruger du betingelsen **Checked**. Tilføj følgende til din kode for View Textboxes imellem de krøllede parenteser:

mnuViewTextboxes.Checked = true;

Du taster bare et punktum efter navnet på dit menupunkt. Derefter vælger du **Checked** betingelsen fra IntelliSense listen. Checked er en Boolesk værdi, der enten kan være sand eller falsk (det har enten en markering eller så har det det ikke).

Kør dit program og klik på dit menupunkt **View Textboxes**. Der dukker nu et ikon op. Det sker fordi default værdien for betingelsen Checked er false. Det kan kun være true hvis du klikker på menupunktet, og dermed køre din kode du har tilføjet.

Spørgsmålet er nu hvordan vi får symbolet væk igen når der klikkes igen. Det er klart at vi er nød til at sætte det til false, hvilket betyder at der ikke er et symbol. Men hvordan skrives koden?

En smart programmeringskode er at skifte den booleske værdi til og fra. Du kan gøre det ved hjælp af en NOT operator (!). Skift din kode til dette:

mnuViewTextBoxes.Checked = !mnuViewTextBoxes.Checked;

I stedet for at skifte værdien Checked til true, gør vi følgende:

!mnuViewTextboxes.Checked;

Dette siger ”NOT Checked”. Men betyder ikke “Uncheked”. Det du gør, er at du sætter den booleske variable til hvad den ikke er. Husk: Checked kan enten være true ELLER false. Så hvis Checked er true, sættes den til false, og omvendt. Resultatet bliver gemt i betingelsen på venstre side af lighedstegnet.

Kør dit program igen og se om det virker. Klik på menupunktet og se efter ikonet. Klik igen og det forsvinder. Det at slå de booleske variabler til og fra er meget almindelig i programmering, og kan spare dig for en masse besværlig kodning!

For rent praktisk at gøre med tekstboksene kan du tilføje en if sætning for at undersøge om variablen er true. Det vi skal gøre er at vi gør tekstboksen synlig hvis der er tjekket af, og skjult hvis der ikke er tjekket af. Tilføj følgende kode efter de linjer du allerede har:

```
if (mnuViewTextBoxes.Checked)
{
textBox1.Visible = true;
textBox2.Visible = true;
}
else
{
textBox1.Visible = false;
textBox2.Visible = false;
}
```

Betingelsen vi skifter på er **Visible** – synligheden af tekstboksene. Som navnet siger, skjuler eller viser den et objekt. Igen er det en boolesk variable. Vi kunne derfor også have skrevet følgende:

textBox1.Visible = !textBox1.Visible;

Brugen af en NOT operator vil skifter mellem synlig eller skjult. Vi tilføjer en if sætning fordi det er praktisk at kunne undersøge hvad variablen indeholder, i stedet for bare at antage det.

En linje du kan fundere lidt over:

if (mnuViewTextboxes.Checked)

Delen indenfor parenteserne kunne også være skrevet på denne måde:

if (mnuViewTextboxes.Checked == true)

I if sætningen vil C# prøve at finde ud koden om i parentesen er true. Du kan derfor udelade ”==” delen, da den ikke behøves. Hvis du vil undersøge for værdien false, kan du bruge NOT operatoren igen. F.eks. på denne måde:

if (!mnuViewTextboxes.Checked)

Det er det samme som at sige:

if (mnuViewTextboxes.Checked == false)

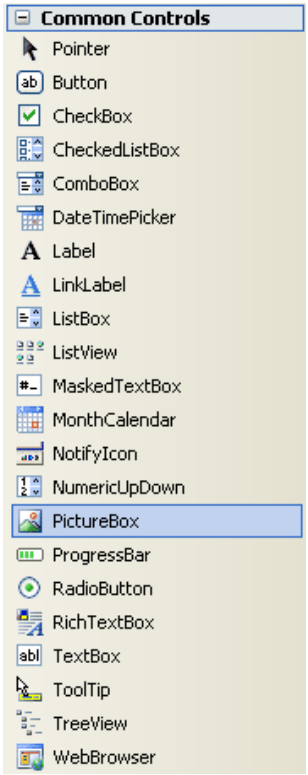
Man betragter det som værende mere professionelt at bruge NOT operatoren. Det betyder det samme, så du bruger selvfølgelig den der passer dig bedst.

Tilføj et billede

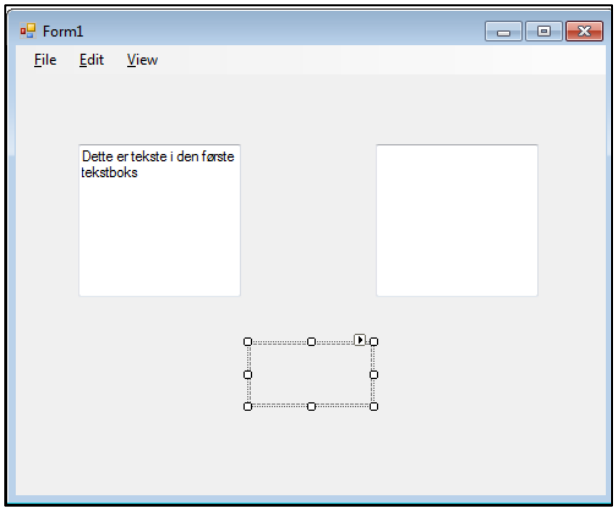
Du har tidligere set Open File dialogboksen. Det er en der dukker op når du vælger Filer og Åbn (**File** og **Open**). Du kan derefter navigere rundt i mapperne og lede efter den fil du vil åbne. Til vores menupunkt View Images vil vi lave noget der er anelse mere kompleks – vi vil have vores egen Open File dialogboks, der giver dig mulighed for at vælge billeder på din computer. Når du vælger et billede vil det blive et nyt kontrolelement i din formular.

Vi har derfor brug for et sted i vores formular hvor vi kan vise et billede. Vi har brug for en Picture Box.

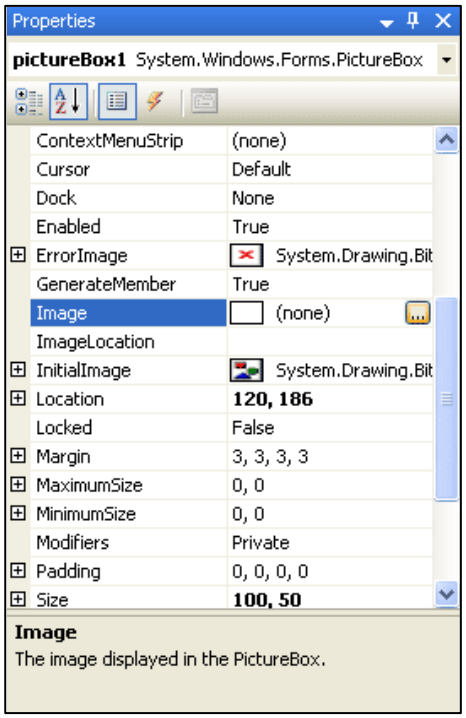
Tag et kik i din Toolbox i venstre side af C#. Under **Common Controls** finder du **PictureBox**:



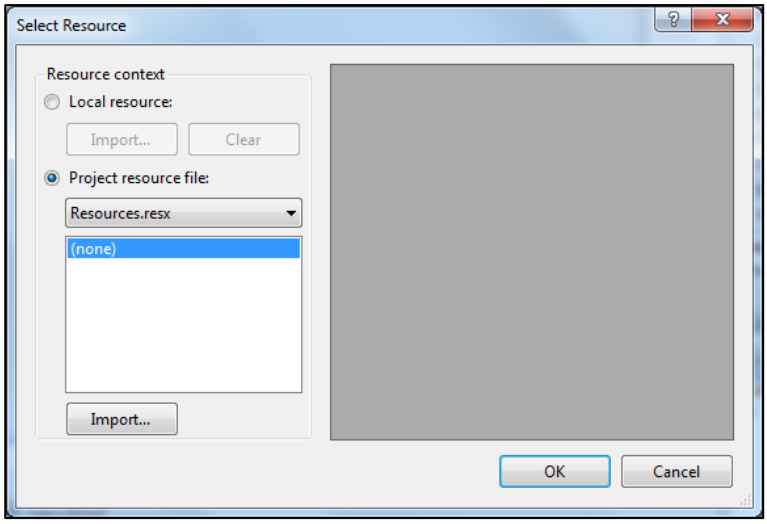
Når du har valgt værkstøjet PictureBox klikker du på din formular for at tilføje et nyt PictureBox kontrolelement. Formular ser nu sådan ud:



Dit nye kontrolelement PictureBox er tomt når du tilføjer det. For at tilføje et billede skal du se nærmere på **Image** i Properties Window:

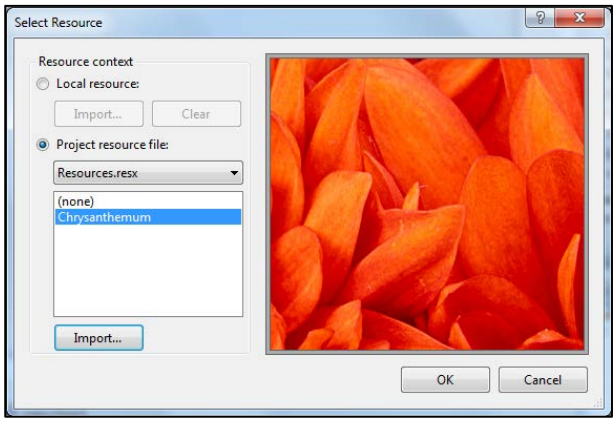


Klik på knappen med de tre prikker for at se følgende dialogboks:

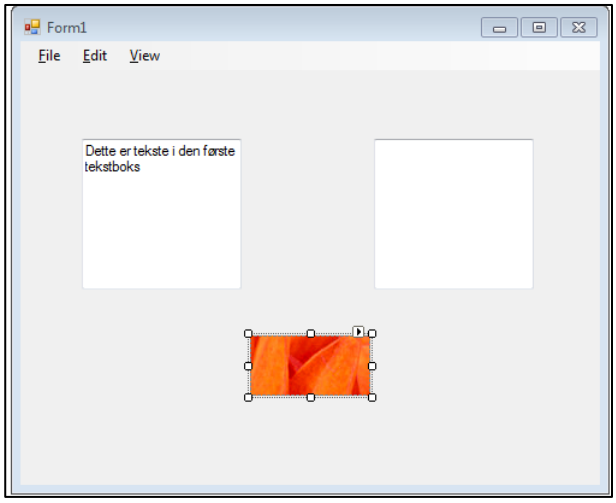


Klik på knappen Import. Nu ser du dialogboksen Open. Find et passende billede frem. Når du har valgt "Project resource file" vil C# kopiere billedet ind I en mappe I dit projekt. (Det er praktisk hvis du skal distribuere dit program.)

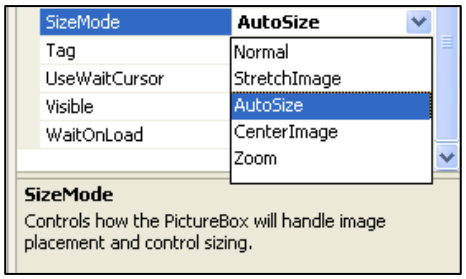
I billedet nedenfor er der valgt et billede af en blomst:



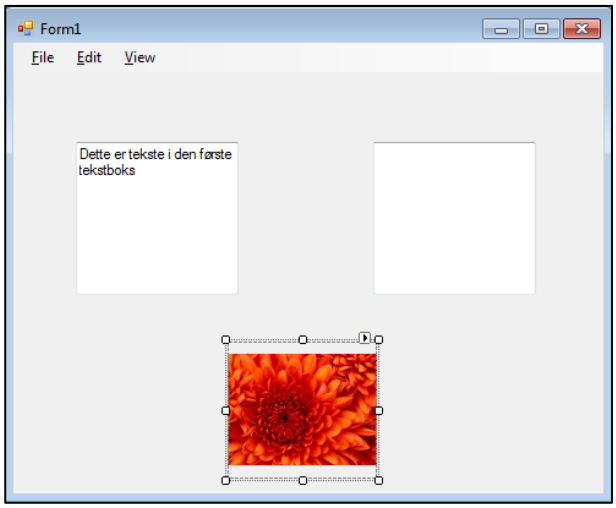
Klik OK for at vende tilbage til din formular:



Bemærk at billedet er for stort til PictureBox'en. Find betingelsen **SizeMode** i Properties Window:



Som du kan se er der lidt at vælge imellem. Vælg **Zoom** og du vil nu kunne se billedet.

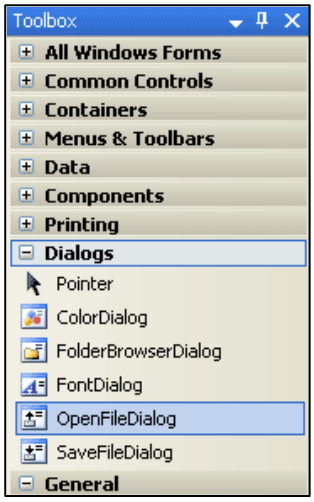


Hvis du kører dit program vil du kunne se dit billede i formularen. Der er dog ingen kant på. Hvis du vil have en kant, kan du undersøge betingelsen **BorderStyle** for kontrolelementet PictureBox.

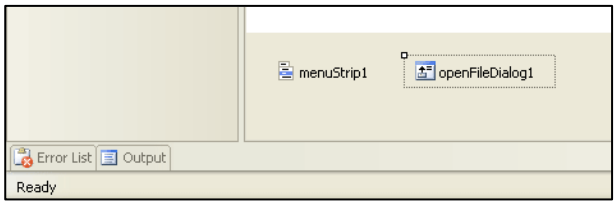
Dialogboksen Open File i C#

Vi vil nu give brugeren mulighed for selv at tilføje et billede til billedboksen, i stedet for at bruge det billede vi har valgt. For at gøre dette mulig skal vi have vist dialogboksen Open File, når brugeren klikker på menuen **View** og **View Images**.

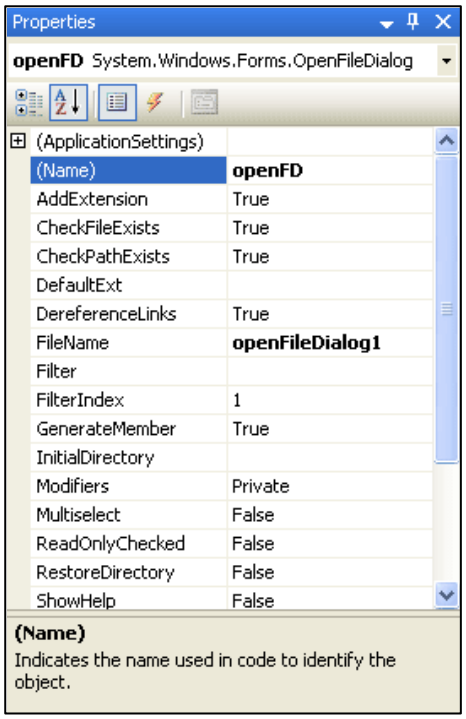
I C# kan du tilføje dialogboksen med hjælp fra et af de indbyggede objekter. Tag et kik på Toolbox'en til venstre. Der er en kategori der hedder Dialogs:



Alle de dialogbokse du kender fra Windows er vist her. Den der er fremhævet er den vi skal bruge – **OpenFileDialog**. Dobbeltklik på dette kontrolelement. Det vil kunne ses i bunden af C#, ved siden af dit menuStrip1 objekt:



Du vil ikke kunne se noget i din formular. Det skyldes at Dialog kontrolelementer er skjult. Den du ser i billedet ovenfor har standard navnet **openFileDialog1**. Det er en anelse langt, så i Properties vinduet ændre du **Name** betingelsen til **openFD**:



Kontrolelementet i bunden af C# ændre sig også:

Mens det nye kontrolelement er markeret kan du prøve at se i Properties vinduet. Du kan se at der er betingelser til **Filter**, **FileName**, **InitialDirectory** og **Title**. Vi vil ændre dem ved hjælp af kode. Der er dog en vigtig ting du skal huske på omkring dialogboksen Open File: De åbner rent faktisk ikke filer! Det dialogboksen Open File gør er, og det samme gælder for de andre Dialog kontrolelementer, at den lader dig vælge hvilen fil der skal åbnes. Du skal skrive yderligere kode for at åbne filen. Det eneste vi rent faktisk gør er at vi finder et filnavn.

Vi er interesseret i at få vist dialogboksen når der klikkes på menuen **View** og **View Images**. Dobbeltklik på dit element i menuen **View**. Du ser nu følgende kode:

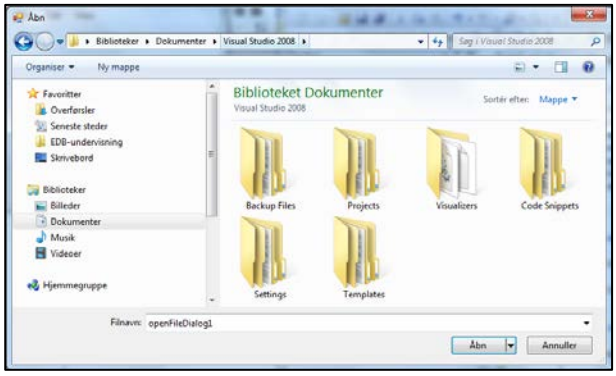
```
private void mnuViewImages_Click(object sender, EventArgs e)
{
    |
}
```

For at få vist dialogboksen Open tilføjer du denne linje kode i mellem de krøllede parenteser:

openFD.ShowDialog( );

Du skriver altså navnet på dit kontrolelement og derefter et punktum. Efter punktummet vælger du **showDialog** fra IntelliSense liste. Som navnet også foreslå viser den dialogboksen.

Kør dit program og prøv det. Du burde se noget i stil med den vist nedenfor når du klikker på **View** og **View Images**:

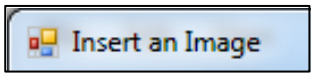


Da vi ikke har angivet betingelser endnu, vil en default placering blive vist. Det er i dette tilfælde Dokument mappen. Default navnet er openFileDialog1. du har dog mulighed for at ændre disse indstillinger.

Vi starter med at en Title. Default Title er Open. Tilføj denne linje kode før den første linje:

openFD.Title = "Insert an Image";

Denne gang bruger du betingelsen **Title** og sætter den til teksten ”Insert an Image”. Du kan selvfølgelig skrive hvad du finder passende. Når du kører programmet vil du kunne se den nye titel:



En anden ting du kan ændre er den placering den skal se i. Default placeringen er Debug mappen tilhørende dit projekt. Du kan nulstille den med betingelsen InitialDirectory. Tilføj følgende linje til din kode, før de to andre linjer:

openFD.InitialDirectory = "C:";

Her angiver vi at default mappen skal være C. Det forudsætter at brugeren har et c-drev. Hvis du vil angive default placeringen til Dokumentmappen for en hvilken som helst computer, kan du prøve dette efter lighedstegnet, i stedet for ”C:”:

= System.Environment.GetFolderPath(Environment.SpecialFolder.Personal);

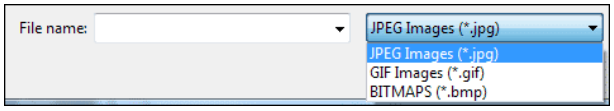
Denne linje finder mappen frem til Dokumentmappen:

Til feltet filnavn kan du bruge betingelsen **FileName**. Tilføj denne linje til din kode (tilføj den før den sidste linje):

openFD.FileName = "";

Her angiver vi filnavnet til at være en tomt. Kør dit program og du ser da at feltet Filnavn i din dialogboks er tomt, og cursoren står og blinker. Vælg en vilkårlig fil og filnavnet vil optræde i feltet.

Det næste du skal gøre er at du skal vælge nogle filtyper. Det er til den liste du ser nedenfor lige under Filnavn:



Vi vil have mulighed for at vælge JPG, GIF og BMP billeder. Når du angiver filtyperne, sætter du et filter op for hvilke filer brugeren kan benytte. Det gøres med betingelsen Filter. Vi er jo ikke interesseret i at brugeren indsætter en tekstfil i din billedboks!

Tilføj følgende kode før den sidste linje:

openFD.Filter = "JPEG|*.jpg";

Bemærk det der kommer efter lighedstegnet:

"JPEG|*.jpg";

Hvis vi kun bruger et filter betyder det at vi ikke vil kunne se andre filtyper. For at tilføje andre filtyper skal du tilføje flere filtyper som vist nedenfor:

openFD.Filter = "JPEG Images|*.jpg|GIF Images|*.gif";

Som du kan se er linjen noget rodet. Bemærk at du adskiller de forskellige filtyper med en lodret streg. Men du skal ogsåbrueg en lodret streg til at adskille tekste i listeboksen fra den brugte filtype. Tilføj BMP filtpen og din kode ser nu sådan ud:

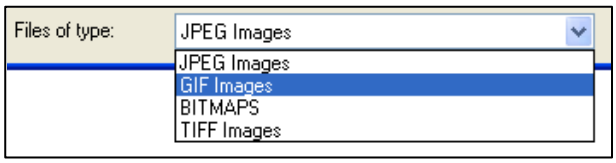
openFD.Filter = "JPEG Images|*.jpg|GIF Images|*.gif|BITMAPS|*.bmp";

I linjen ovenfor har vi nu tre filtyper.

Her er yderligere nogle billedfiltyper og deres filefternavn:

TIFF Images: *.tif or *.tiff
PNG Images: *.png
PICT Images: *pct or *.pict

Der er selvfølgelig mange andre. I billedet nedenfor, har vi tilføjet filtypen TIFF til listen:



Hvis du vil vise alle filtyper kan du bruge en stjerne i stedet for filefternavnet. For eksempel:

openFD.Filter = "JPEG Images|*.jpg|All Files|*.*";

Men vi har stadig ikke indsat noget billede. Når vi skal indsætte et billede i billedboksen, skal vi bruge filnavnet som brugeren har valgt. Du kan tilføje en string variabel til din kode:

string Chosen_File = "";

Du kan derefter tilgå betingelsen FileName for openFD. På denne måde:

Chosen_File = openFD.FileName;

Filnavnet vil da blive lagt i variablen vi kalder **Chosen_File**.

Når vi skal placere et nyt billede i billedboksen i din formular, skal du bruge Image betingelsen:

pictureBox1.Image

For at placere den valgte fil i Image betingelsen, bruger du følgende:

pictureBox1.Image = Image.FromFile(Chosen_File**);**

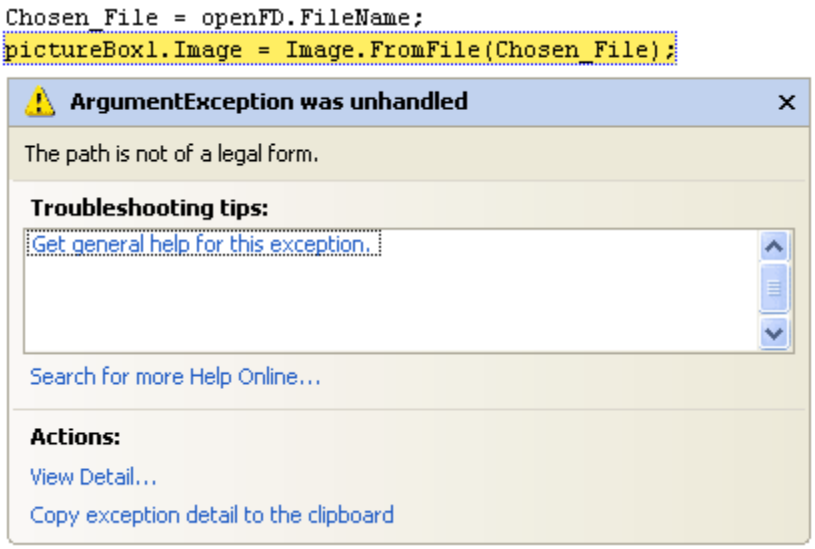
Efter lighedstegnet kan du se objektet **Image**. Denne har en method der hedder **FromFile()**. I parenteserne for denne method kan du skrive navnet på billedfilen. I vores tilfælde er det billedfilen der er gemt i variablen Chosen_file.

Tilføj den nye linje til din kode og du ser nu følgende (de sidste filtre er ikke taget med):

```
private void mnuViewImages_Click(object sender, EventArgs e)
{
    string Chosen_File = "";

    openFD.InitialDirectory = "C:";
    openFD.Title = "Insert an Image";
    openFD.FileName = "";
    openFD.Filter = "JPEG Images|*.jpg|GIF Images|*.gif";

    openFD.ShowDialog();

    Chosen_File = openFD.FileName;
    pictureBox1.Image = Image.FromFile(Chosen_File);
}
```

Kør dit program og test det. Vælg et billede der skal åbnes. Du vil se at det nye billede erstatter det gamle billede i din billedboks.

Der er dog et problem. I stedet for at klikke på Open, klik da på **Cancel**. Du får nu en fejlmeddelelse:



Da Annuller knappen blev klikket er der ikke noget billednavn i variablen Chosen_File. Så programmet "fejler" for dig. Du er nød til at tage hånd om det i din kode:

For at undersøge om der er klikket på Annuller knappen, kan du bruge følgende:

```
if (openFD.ShowDialog() == DialogResult.Cancel)
{
MessageBox.Show("cancel button clicked");
}
```

Der er et indbygget objekt der hedder **DialogResult**. Du tjekker om den har værdien **Cancel**. Når du tilføjer en else sætning får vi følgende kode:

```
private void mnuViewImages_Click(object sender, EventArgs e)
{
    string Chosen_File = "";

    openFD.InitialDirectory = "C:";
    openFD.Title = "Insert an Image";
    openFD.FileName = "";
    openFD.Filter = "JPEG Images|*.jpg|GIF Images|*.gif";

    if (openFD.ShowDialog() == DialogResult.Cancel)
    {
        MessageBox.Show("Operation Cancelled");
    }
    else
    {
        Chosen_File = openFD.FileName;
        pictureBox1.Image = Image.FromFile(Chosen_File);
    }

}
```

Ændre din kode så den ser ud som vist ovenfor. Kør dit program igen. Det crasher ikke længere når du klikker på Cancel knappen.

Du kan også bruge følgende IF sætning, i stedet for den viste ovenfor:

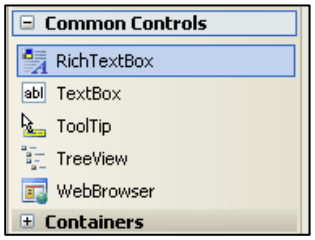
```
if (openFD.ShowDialog() != DialogResult.Cancel)
{
Chosen_File = openFD.FileName;
pictureBox1.Image = Image.FromFile(Chosen_File);
}
```

Vi har brugt NOT symbolet (!). Vi undersøger om DialogResult IKKE er lig med Cancel.

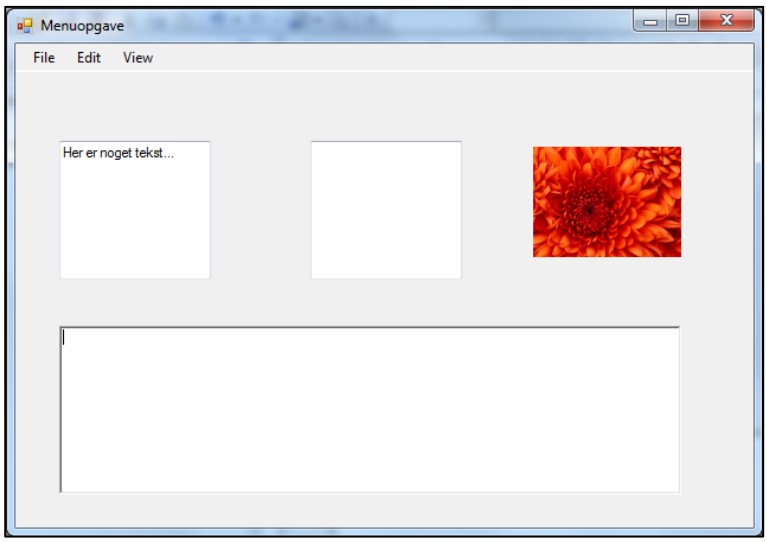
Åben en tekst fil med dialogboksen Open File

Vi kan gøre genbrug af dialogboksen Open File som vi har tilføjet. I stedet for at filtrere efter billeder, vil vi filtrere efter tekstfiler. Vi skal også tilføje en anden slags tekstboks – en Rich Text Box. Det gør det nemt for os at indlæse teksten fra en fil direkte ind i vores program.

Vend tilbage til Designer View, så du kan se din formular. Udvid din Toolbox og find RichTextBox – den ligger under Common Controls:



Dobbeltklik på elementet for at tilføje en **RichTextBox** til din formular. Du kan være nød til at justere højden og bredden på din formular, og eventuelt flytte lidt på dine kontrolelementer. Din formular burde se nogenlunde sådan ud når du har tilføjet din RichTextBox:



Din RichTextBox er den du ser i bunden. Den har det samme udseende som en normal tekstboks – forskellen er at du kan foretage dig mere med denne. En Method du finder i det nye kontrolelement er **LoadFile**(). Vi skal gøre brug af denne for at hente en tekstfil.

Nu da vi har tilføjet en RichTextBox, kan vi tilføje noget kode. Find kodedelen frem for dit menupunkt **Open** under **File**. Det ser sådan ud:

```
private void mnuOpen_Click(object sender, EventArgs e)
{
    |
}
```

Vi kan tilføje samme linjer som før. Så tilføj følgende kode:

string Chosen_File = "";

openFD.InitialDirectory = "C:";
openFD.Title = "Open a Text File";
openFD.FileName = "";

Det eneste vi har ændret her er titlen. I den næste linje kan vi tilføje et filter:

openFD.Filter = "Text Files|*.txt|Word Documents|*.doc";

Din RichTextBox kan åbne simple tekstfiler, såvel som Word dokumenter. Så vi har tilføjet begge til filter betingelsen. Bemærk at den dog ikke er helt perfekt hvad angår Word dokumenter!

Nu skal vi have vist vores dialogboks Open File, så vi kan vælge en fil. Tilføj følgende kode:

```
if (openFD.ShowDialog() != DialogResult.Cancel)
{
Chosen_File = openFD.FileName;
richTextBox1.LoadFile(Chosen_File, RichTextBoxStreamType.PlainText);
}
```

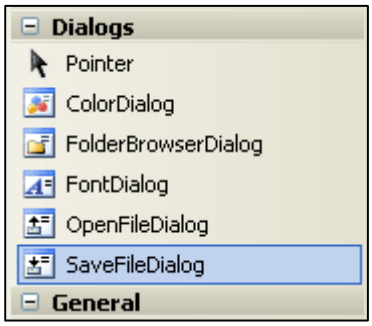
Det er mere eller mindre det samme som tidligere. Men bemærk dog linjen, der tilføjer tekst filen til din RichTextBox.

Senere skal du se en bedre måde at åbne en tekstfil. Men nu skal du bare køre dit program, og se om det virker. Du burde have mulighed for at tilføje en simpel tekst fil til din RichTextBox.

Tilføj en Gem som dialogboks

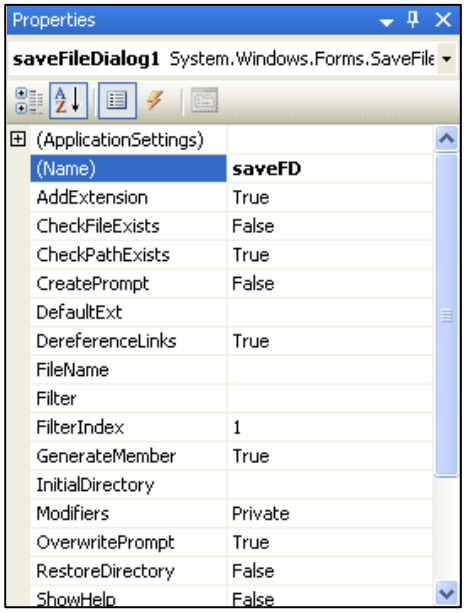
En anden Method du kan bruge i forbindelse med din **RichTextBox** er **SaveFile**( ). Som navnet foreslår giver den dig mulighed for at gemme den fil du arbejder med i din tekstboks. Vi skal bruge den i forbindelse med et andet objekt. Denne gang skal vi bruge kontrolelementet **SaveFileDialog** i stedet for OpenFileDialog.

Vend tilbage til din formular og find kontrolelementet SaveFileDialog i værktøjskassen:



Dobbeltklik for at tilføje den til dit projekt. Den vises nu i bunden af skærmen:

Marker **saveFileDialog1**. Se på betingelserne i højre side. Skift betingelsen **Name** til **SaveFD**:



Vend tilbage til din **File** menu i din Menu Strip. Klik på **File** og dobbeltklik derefter på menupunktet **Save**. Du vil nu se kode til dette element:

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

Koden til at gemme er næsten den sammen som til menupunktet Open. I stedet for at skrive openFD er det saveFD. Her er koden:

```
private void mnuSave_Click(object sender, EventArgs e)
{
  string Saved_File = "";

  saveFD.InitialDirectory = "C:";
  saveFD.Title = "Save a Text File";
  saveFD.FileName = "";

  saveFD.Filter = "Text Files|*.txt|All Files|*.*";

  if (saveFD.ShowDialog() != DialogResult.Cancel)
  {
      Saved_File = saveFD.FileName;
      richTextBox1.SaveFile(Saved_File, RichTextBoxStreamType.PlainText);
  }
}
```

Du burde kunne gennemskue hvad der sker i koden ovenfor. Linjen der gemmer denne:

richTextBox1.SaveFile(Saved_File, RichTextBoxStreamType.PlainText);

Der er dog en bedre måde at manipulere med filer. Du lærer hvordan du skal behandle tekstfiler i et senere kapitel. Men tilføj koden ovenfor og kør dit program. Klik på **File** og **Open** for at tilføje en tekst til din tekstboks. Foretag lidt ændringer. Klik derefter på **File** og **Save**. Dine ændringer skulle gerne være permanente.