Bestanden

Hoofdstuk 18

In dit hoofdstuk …

- Tekstbestanden
- Lezen en schrijven
- Mappen (Directories)
- `MessageBox`
- Standaard dialoogvensters
- Menu's

In dit hoofdstuk …

- Tekstbestanden
- Lezen en schrijven
- `MessageBox`
- Standaard dialoogvensters
- Menu's
- Mappen (Directories)

Inleiding

- Opslaan van gegevens
- Opslagcapaciteit = bytes
  - RAM
    - Korte toegangstijd
    - Duurder
    - Tijdelijk
  - Opslagmedia
    - Langere toegangstijd, maar grotere capaciteit
      - CD-roms 740MB, DVD 4.7 GB, Blu Ray 25GB, vast harde schijf 1TB
    - Goedkoper
    - Relatief permanent

De basisbegrippen van streaming

- Een bestand wordt beschouwd als een stroom gegevens die in één doorlopende (sequentiële) beweging wordt gemanipuleerd (lezen, schrijven)

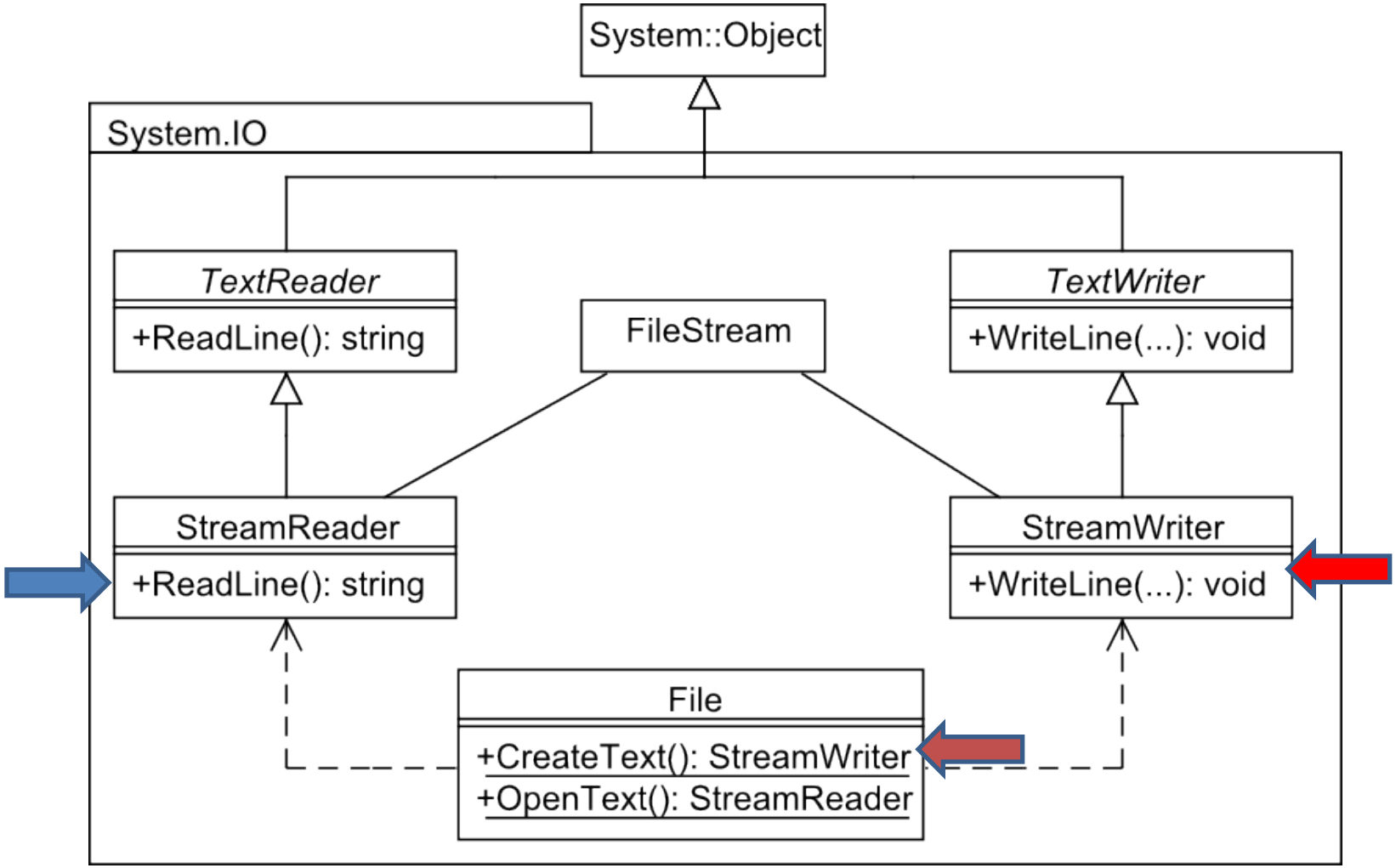
```
1. Open bestand
2. Inlezen gegevens van stream of
   Wegschrijven gegevens naar stream
3. Bestand sluiten
```

- Een bestand is een “system resource”. Niet vergeten te sluiten!
  Hoe kan je dit garanderen?

De basisbegrippen van streaming

- Vaak handig om tekstbestanden voor te stellen
  - als stel regels, met elk aantal tekens
  - afgesloten met (line feed) \n, (cariage return)\r of \r\n
    - C# zet zelf \r\n achter einde van elke regel
    - Je hoeft dus zelf niet expliciet ENTER te duwen op einde van de regel
- Ook binaire bestanden (ipv String)
  - Afbeeldingen, muziek,…
  - Opgeslagen in gecompliceerder formaat

Overzichtsdiagram



zelf

Bestandsuitvoer

```
using System.IO;
...
private void writeButton_Click(object sender, EventArgs e)
{
    StreamWriter outputStream =
                    File.CreateText("myfile.txt");
    outputStream.WriteLine("This file will");
    outputStream.WriteLine("contain 3");
    outputStream.WriteLine("lines of text.");
    outputStream.Close();
}
```

Bestandsuitvoer

Verwijder System.Shapes bib
2x Path definitie

- `using System.IO;`
- `File.CreateText(bestandsnaam)`
  - Maakt een `StreamWriter` object naar een tekstbestand
  - Als dit bestand nog niet bestaat, wordt het aangemaakt. Anders wordt het bestaande bestand overschreven
- `outputStream.WriteLine(string)`
  - Schrijft de `string` naar de stream (en dus in het bestand) en sluit af met een `NewLine` teken
- `outputStream.Close()`
  - Sluit de stream af
  - Dit statement zou beter in een `finally` block staan

Bestandslocaties

```
StreamWriter outputStream =
          File.CreateText("myfile.txt");
```

- Het bestand komt terecht in dezelfde locatie als het .exe-bestand
  - `Projectfolder\bin\Debug`
- Bij uitvoeren van programma geen schrijfrecht (`UnauthorizedAccessException`), bv:
  - `C:\Program Files\File Output`

zelf

Vraag

- Schrijf functie die getal inleest uit TextBox object en die vervolgens na het drukken op een knop een structuur tekent

```
x
xx
xxx
xxxx
```

waarbij de hoogte overeenkomt met het ingelezen getal. Dit moet komen in een apart bestand driehoek.txt.

Tip: Maak gebruik van outputStream.Write("…");

Bestandslocaties

```
StreamWriter outputStream =
     File.CreateText(@"C:\myfile.txt");
```

- Ofwel:
  - Lukt niet, Access denied
  - Lukt wel en in juiste map (administrator)
- Dus schrijven naar is geen goed idee!
  - C:\
  - C:\Windows (incl. submappen)
  - C:\Program Files (incl. submappen)

Zelf:
MSDN

Bestandslocaties

- Waar document wel bewaren?
  - Alle submappen C:\Users\An
    - ApplicationData
    - MyPictures
    - MyDocuments
    - Enz.
- Bv. Opslaan in C:\Users\An\MyDocuments\myFile.txt

```
string destination = Environment.GetFolderPath(
            Environment.SpecialFolder.MyDocuments);

string newFile = System.IO.Path.Combine(
                        destination, "myfile.txt");
```

zelf

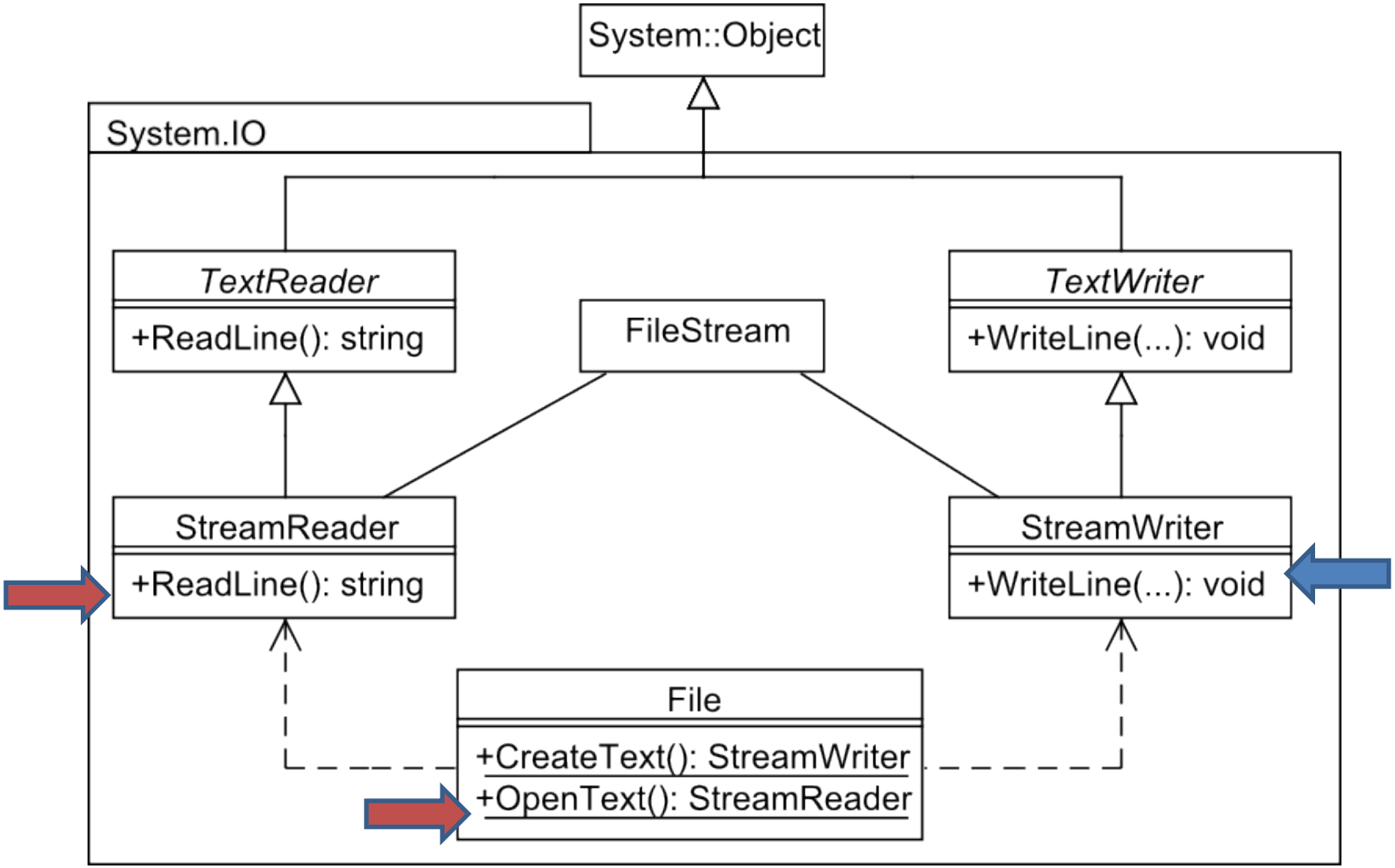
File Output herwerkt:

```
private void writeButton_Click(object sender, EventArgs e)
{
    string destination = Environment.GetFolderPath(
                            Environment.SpecialFolder.Desktop);

    string newFile = System.IO.Path.Combine(destination, "myfile.txt");
    StreamWriter outputStream = File.CreateText(newFile);
    outputStream.WriteLine("This file will");
    outputStream.WriteLine("contain 3");
    outputStream.WriteLine("lines of text.");
    outputStream.Close();
}
```

Zoek “Mijn Documenten”, bv: C:\Desktop

Vorm compleet pad, bv: C:\Desktop\myfile.txt

Overzichtsdiagram



Zelf

Bestandsinvoer

```
private void readButton_Click(object sender, EventArgs e)
{
    string sourcepath = Environment.GetFolderPath(
                            Environment.SpecialFolder.Desktop);
    string myfile = System.IO.Path.Combine(sourcepath, "myfile.txt");
    StreamReader inputStream = File.OpenText(myfile);
    string line = inputStream.ReadLine();
    while (line != null)
    {
        textBox.AppendText(line);
        textBox.AppendText(Environment.NewLine);
        line = inputStream.ReadLine();
    }
    inputStream.Close();
}
```

Bestandsinvoer

- `using System.IO;`
- `File.OpenText(bestandsnaam)`
  - Maakt een `StreamReader` object naar een (bestaand) tekstbestand
  - Als dit bestand nog niet bestaat: `FileNotFoundException`
- `inputStream.ReadLine()`
  - Leest een (volgende) regel uit het bestand tot `null` wordt teruggegeven
  - Ook mogelijk in 1 keer tot einde bestand lezen `ReadToEnd()`
- `inputStream.Close()`
  - Sluit de stream af
  - Dit statement zou beter in een `finally` block staan

Bestanden en exceptions

- Vrijwel elke IO operatie kan mislopen
  - `FileNotFoundException bij File.OpenText`
  - `IOException bij StreamReader.ReadLine en StreamWriter.WriteLine`
- Kijk dus goed naar de online documentatie en vang indien nodig de exception op
- Zeker: streams afsluiten (`Close`) in een `finally` block. Anders zou dit kunnen aanleiding geven tot memory leaks

Voorbeeld File Search met exceptions, p350, 351

```
try{
        …
}
catch (FileNotFoundException ex){
        MessageBox.Show("Error: File not found: " +textBox.Text
                        +". Re-enter name.");
}
catch (Exception ex) {
        MessageBox.Show("Error concerning file: " +textBox.Text
                        +". " + ex.Message);
}
finally{
        if (inputStream != null)  inputStream.Close();
}
```

Waarom finally blok en waarom if test?

Voorbeeld

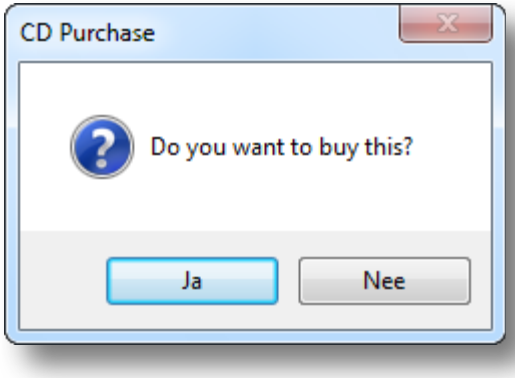
- ReadLine functie uit InputStreamReader heeft excepties
  - OutOfMemoryException
  - IOException
- Dmv Exception vang je beiden op
  - Geen info over detail fout

In dit hoofdstuk …

- Tekstbestanden
- Lezen en schrijven
- `MessageBox`
- Standaard dialoogvensters
- Menu's
- Mappen (Directories)

MessageBox

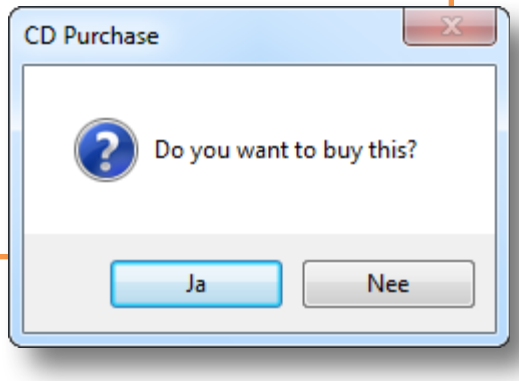
- Gebruiker belangrijke boodschap tonen
  - Enkel via label is niet straf genoeg
- Verschillende versies (overloading)
  - `MessageBox.Show(`*`boodschap`*`)`
  - `MessageBox.Show(`*`boodschap, titel`*`)`
  - `MessageBox.Show(`*`boodschap, titel, knoppen`*`)` mbv `MessageBoxButtons` enum
  - `MessageBox.Show(`*`boodschap, titel, knoppen, pictogram`*`)` mbv `MessageBoxIcon` enum
- Vaststellen welke knop geklikt werd (=Output van MessageBox)
  - `DialogResult.Abort`
  - `DialogResult.Cancel`
  - `DialogResult.Ignore`
  - `DialogResult.No`
  - `DialogResult.None`
  - `DialogResult.OK`
  - `...`
- MSDN Documentatie



MessageBox

```
MessageBox.Show("The age must be over 18!",
                "Age is out of range!",
                MessageBoxButton.OK,
                MessageBoxImage.Exclamation);
```


```
if (MessageBox.Show("Do you want to buy this?",
                    "CD Purchase",
                    MessageBoxButton.YesNo,
                    MessageBoxImage.Question) == DialogResult.Yes)
{
    MessageBox.Show("User clicked yes");
}
else
{
    MessageBox.Show("User clicked no");
}
```



Zelf

Oefening

- Schrijf code om MessageBox te tonen die vraagt:

  "Is Jouw leeftijd boven 18 jaar?"

- Toon toepasselijke reacties op antwoorden van gebruiker

```
if (MessageBox.Show("Do you want to buy this?",
                    "CD Purchase",
                    MessageBoxButton.YesNo,
                    MessageBoxImage.Question) == DialogResult.Yes)
{
    MessageBox.Show("User clicked yes");
}
else
{
    MessageBox.Show("User clicked no");
}
```

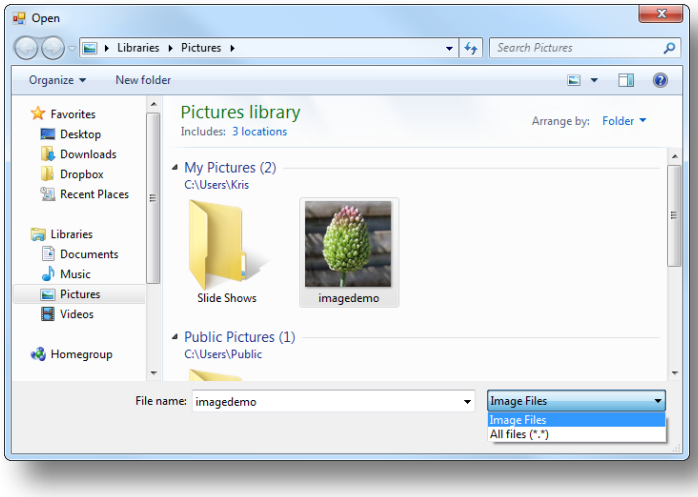
In dit hoofdstuk …

- Tekstbestanden
- Lezen en schrijven
- `MessageBox`
- Standaard dialoogvensters
- Menu's
- Mappen (Directories)

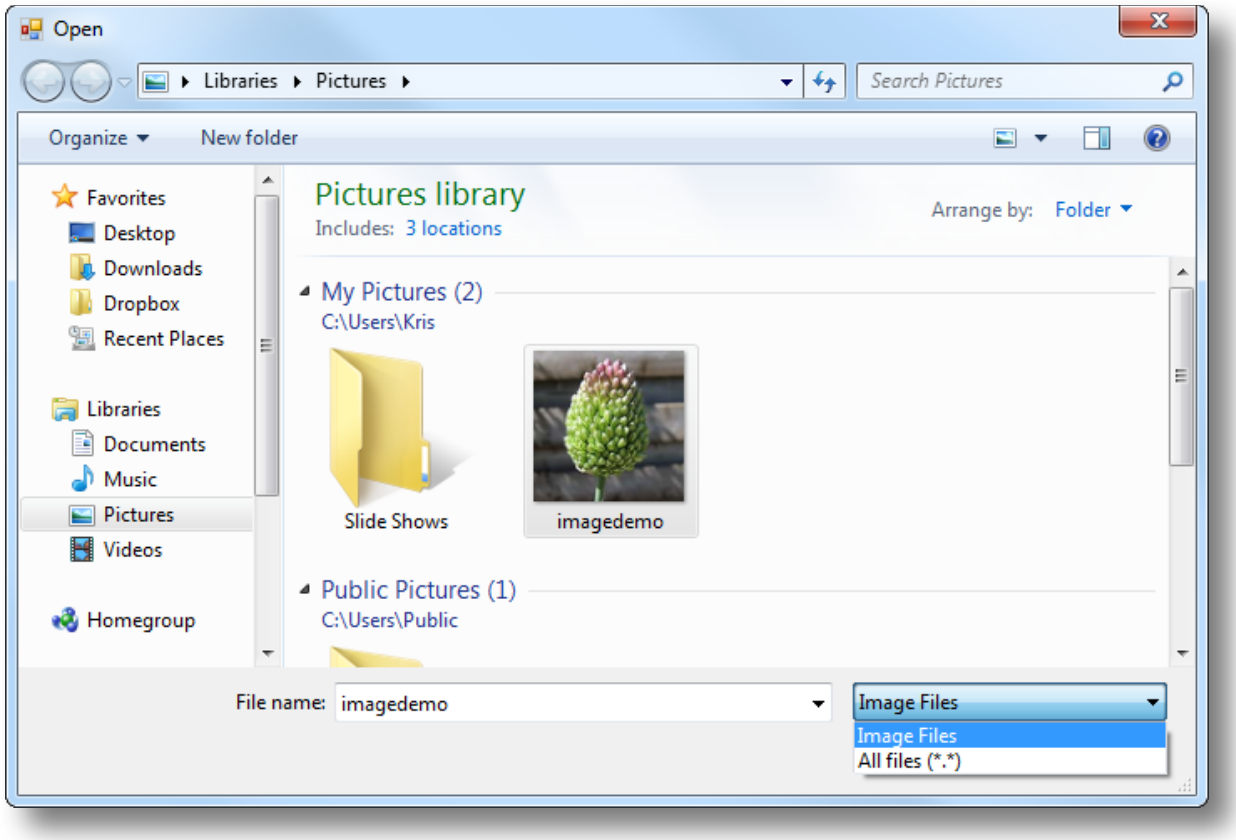
OpenFileDialog

Openen van bestanden

- Via bib using Microsoft.Win32
  - OpenFileDialog
  - SaveFileDialog
- Property: InitialDirectory
- Property: FileName
- Property: Filter
- Methode: ShowDialog
- Reageren op actie:
  - DialogResult.OK
  - DialogResult.CANCEL



OpenFileDialog



Zelf

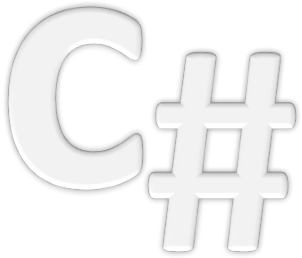
OpenFileDialog

```
OpenFileDialog openFileDialog1 = new OpenFileDialog();



string startdir = Environment.GetFolderPath(
                        Environment.SpecialFolder.Desktop);
openFileDialog1.InitialDirectory = startdir;
openFileDialog1.Filter = "Image Files (*.BMP,*.JPG, *.GIF)|
          *.BMP,*.JPG, *.GIF |All files (*.*)|*.*";

if (openFileDialog1.ShowDialog() == true)
{
    MessageBox.Show(openFileDialog1.FileName);
    string currentFile = openFileDialog1.FileName;
    MessageBox.Show(currentFile);
}
```

2 filters
- Image Files (*.BMP; *.JPG; *.GIF)
- All files (*.*)
- ZIE MSDN voor filter tekst files

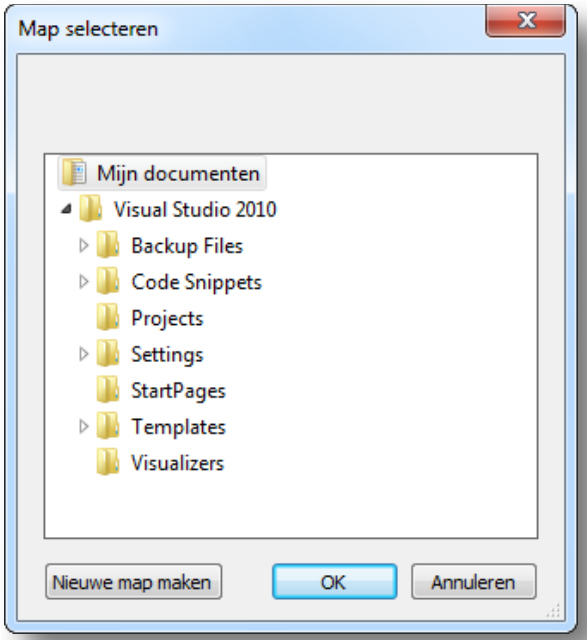
Zelf

SaveFileDialog

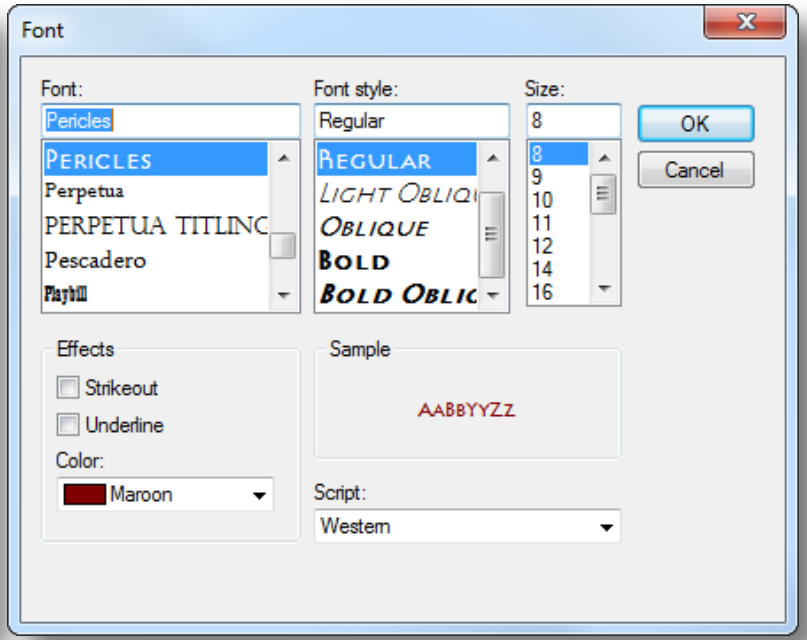



```
SaveFileDialog dialog= new SaveFileDialog();



dialog.InitialDirectory =  Environment.GetFolderPath(
                        Environment.SpecialFolder.MyPictures);
if (dialog.ShowDialog() == true)
{
    MessageBox.Show(dialog.FileName);
}
```

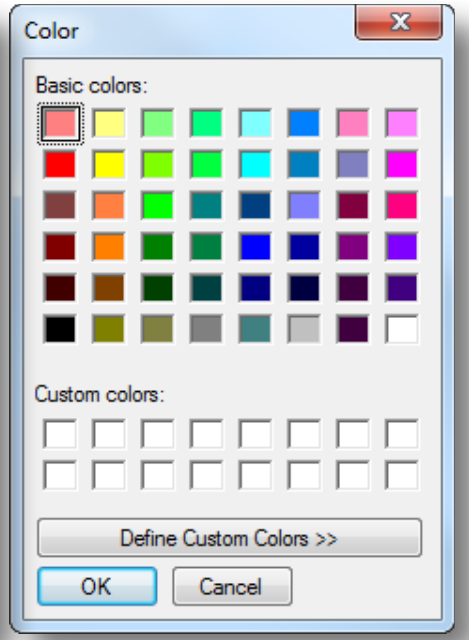
Andere dialoogvensters



`FolderBrowserDialog`



`FontDialog`



`ColorDialog`

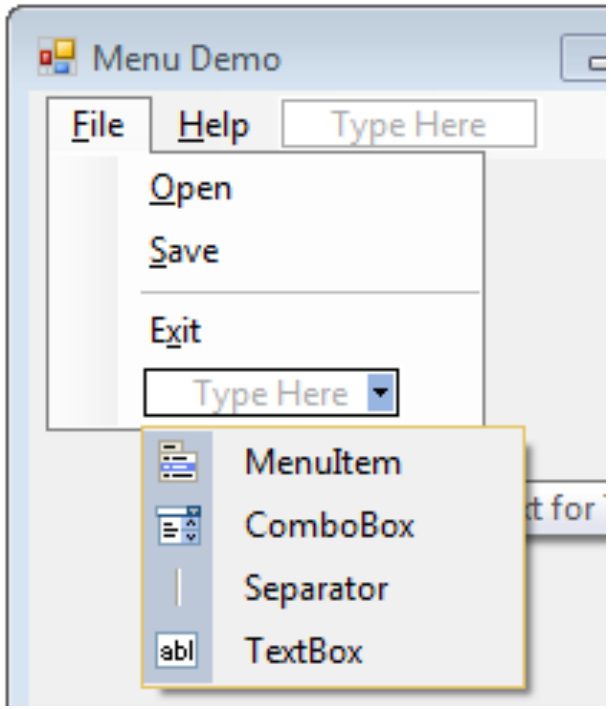
Zie oefeningen

In dit hoofdstuk …

- Tekstbestanden
- Lezen en schrijven
- `MessageBox`
- Standaard dialoogvensters
- Menu's
- Mappen (Directories)

Een menu creëren

- Menu in Toolbox selecteren
- Via XAML opbouwen



XAML Menu

```
<DockPanel>
   <Menu DockPanel.Dock =" Top">
         <MenuItem Header="_File">
             <MenuItem Header="_Open"/>
             <MenuItem Header="_Save"/>
             <Separator />
             <MenuItem Header="E_xit"
                       Name= "exitItem" Click ="exitItem_Click"/>
          </MenuItem>
          <MenuItem Header="_Help">
             <MenuItem Header="_About"/>
          </MenuItem>
    </Menu>

<Grid DockPanel.Dock="Bottom">
</Grid>
</DockPanel>
```

ALT F is shortcut

Menu event

```
private void exitItem_Click(object sender,
                                        RoutedEventArgs e)
{
    Environment.Exit(0);
}
```

- Gebruik `Environment.Exit(0)` om een programma af te sluiten
- Eveneens ok: `Application.Exit()`

Menu Open event

```
private void openMenuItem_Click(object sender,
                                RoutedEventArgs e)
{
    OpenFileDialog dialog = new OpenFileDialog();
    StreamReader ins;
   string startdir = Environment.GetFolderPath(
                        Environment.SpecialFolder.Desktop);
     dialog.InitialDirectory = startdir;
     if (dialog.ShowDialog() == true){
       string cf = dialog.FileName;
       ins = File.OpenText(cf);
       textBox.Text = ins.ReadToEnd();
       ins.Close();
     }
}
```

- Pas XAML aan
  Maak TextBox en zet tekst uit file erin

In dit hoofdstuk …

- Tekstbestanden
- Lezen en schrijven
- `MessageBox`
- Standaard dialoogvensters
- Menu's
- Mappen (Directories)

De klasse `Directory`

- Faciliteiten voor mappen/bestanden
  - Niet de inhoud, maar bv namen van bestanden veranderen of laten zien
- `using System.IO;`
- `GetFiles(path)`
  - Retourneert een array van strings; de bestandsnamen van files in directory `path`
- `GetDirectories(path)`
  - Retourneert een array van strings; de namen van de directories in directory `path`

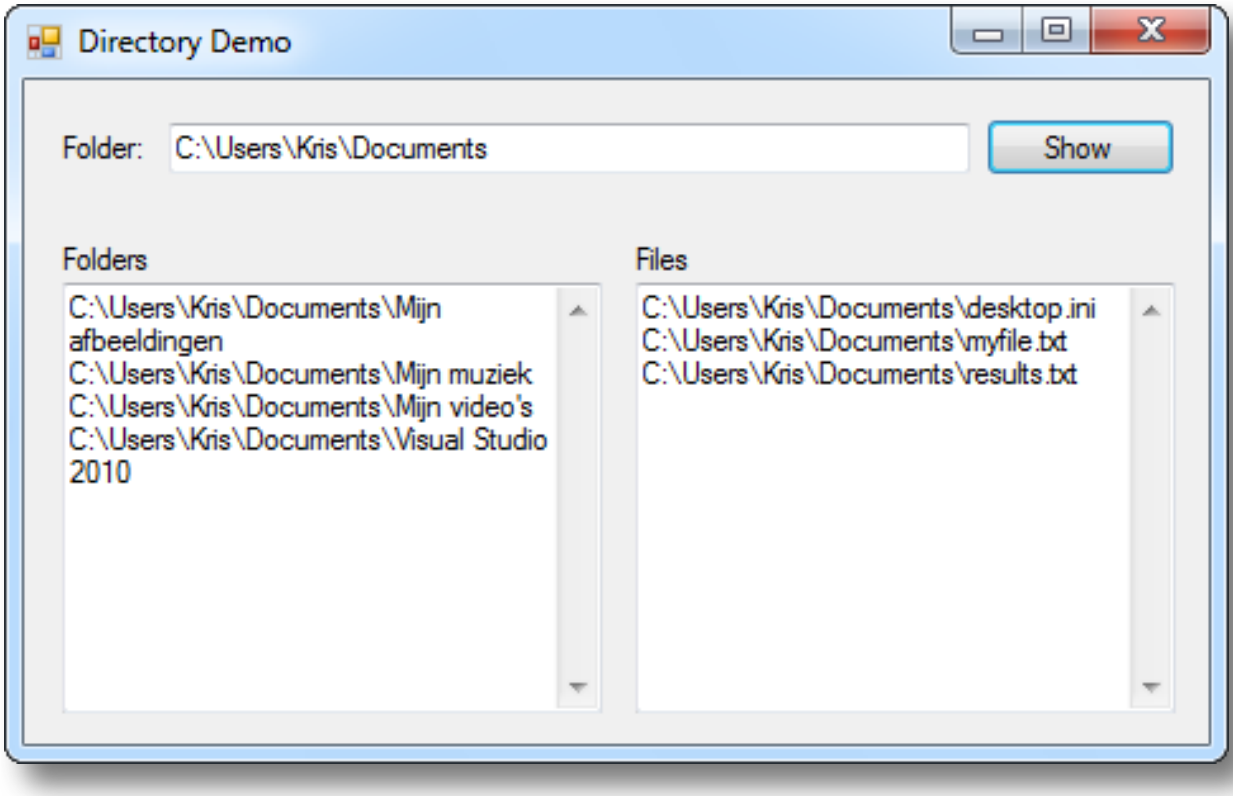
zelf

Demo Directory

```
private void button_Click(object sender, EventArgs e){
        textBox.Clear();
        textBox1.Clear();
        string map = Environment.GetFolderPath(Environment.SpecialFolder.Desktop);
           // Display all file names
        string[] files = Directory.GetFiles(map);
        foreach (string file in files){
           textBox.AppendText(file+"\r\n");
        }

           // Display all folder names
        string[] dirs = Directory.GetDirectories(map);
        foreach (string dir in dirs){
           textBox1.AppendText(dir+"\r\n");
        }
}
```

De klasse `Directory`

- Programma Directory Demo
- *Laat folders en files zien binnen bepaalde Directory*



SAMENVATTING

Samenvatting

- Schrijven van en naar een bestand met klassen StreamReader en StreamWriter
- Gebruik van Exceptions
- Dialoogvensters
- Menu
- Klasse Directory

Oefening

- 18.2: Schrijf een programma dat het aantal regels telt in een tekst
  - Componenten:
    - Browse knop
    - Tekstvak met file naam
    - Regel tel knop
    - Tekstvak met resultaat
  - Browse event: voor openDialog
  - Regel tel event met exceptions
  - Gebruik een “mooie” MessageBox.Show in de exception afhandeling

Pas aan Oefening

- Tel nu het aantal woorden
  - Extra knop (woordTel)
  - Gebruik ReadToEnd methode van StreamReader object
  - Zoek op internet
    - Bv. Site: www.dotnetperls.com/word-count

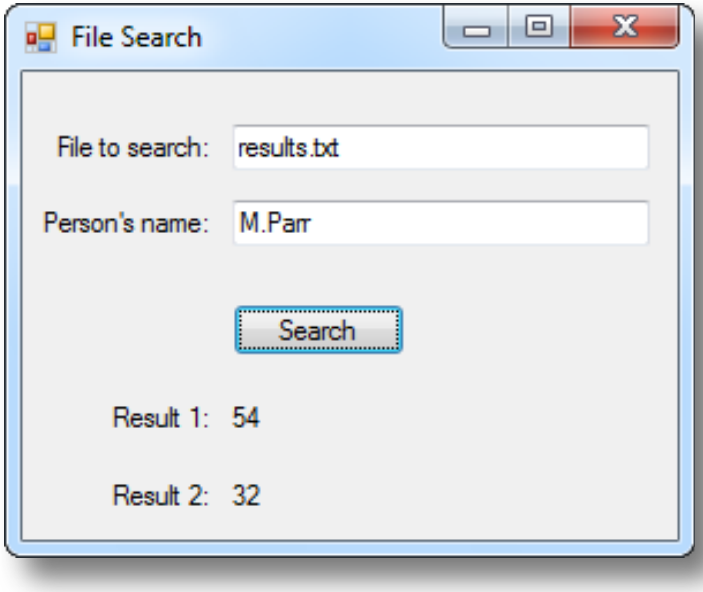
Open bestand

Bestanden doorzoeken

- Doorzoeken van bestand op bepaalde criteria
- Programma File Search
  - Doorzoek bestand met examenuitslagen van de vorm

    P.Doe, 43, 68
    D.Bell, 87, 99
    …

  - Programma toon examencijfers van een opgegeven student



Voorbeeld: Bestand doorzoeken

```
found = false
while(line not null && not found){
        read line;
        split line in 3 fields
        if (field 1 == name student){
                found == true
                display other 2 corresponding fields
        }
}
if (not found) display warning
```