

C# Advanced Tekst bestanden

Koen Bloemen



**DE HOGESCHOOL
MET HET NETWERK**

Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be





1.7 CSV files

1.8 Fixed width tekst bestanden

1.9 Exceptions

1.10 Directory

1.11 Standaard dialoogvensters

CSV files

- CSV = Comma Separated Value
- Velden of waarden worden gescheiden door een scheidingsteken
 - Komma
 - Puntkomma
 - Eender welk teken
- File extensie
 - .txt
 - .csv
- Wordt gebruikt door heel veel applicaties om gegevens te lezen en/of weg te schrijven

CSV file schrijven

- Tekst wegschrijven

```
using (StreamWriter sw = new StreamWriter("KommaBestand.txt"))
{
    // Gegevens wegschrijven
    sw.WriteLine("Patricia Briers,PCVO Limburg");
    sw.WriteLine("Anne Koninx,VJC Hasselt");
    sw.WriteLine("Tom Quareme,UHasselt Diepenbeek");
    sw.WriteLine("Paul Dox,PXL Hasselt");
}
```

- Denk eraan
 - Dankzij using () {...} moeten we geen sw.Close() doen!

CSV file lezen

- Tekst inlezen en opsplitsen via `String.Split()` volgens scheidingsteken

```
using (StreamReader sr = new StreamReader("KommaBestand.txt"))
{
    // Tekst inlezen regel per regel
    while (!sr.EndOfStream)
    {
        // Splits ingelezen regel op volgens ;
        string[] waarden = sr.ReadLine().Split(",");
        Console.WriteLine($"{waarden[0]} werkt in {waarden[1]}");
    }
}
```

- Denk eraan
 - Dankzij `using () {...}` moeten we geen `sr.Close()` doen!

Fixed-width text files schrijven

- Elke regel heeft vaste breedte (dus een vast aantal karakters).
- Elke regel verder opvullen met spaties tot vaste breedte bereikt is.
- Tekst wegschrijven

```
using (StreamWriter sw = new StreamWriter("VastBestand.txt"))
{
    // Gegevens wegschrijven
    // Het getal vb/ 20 voorziet een ruimte van 20 karakters
    // Negatief getal is links uitlijnen vb/ |Tom |
    // Bij een positief getal was het rechts uitlijnen vb/ | Tom|
    sw.WriteLine($"{ "Quareme",-20}{ "Tom",-10}{ "Borgloon",-15}");
    sw.WriteLine($"{ "Briers",-20}{ "Patricia",-10}{ "Gent",-15}");
    sw.WriteLine($"{ "Dox",-20}{ "Paul",-10}{ "Hasselt",-15}");
}
```

Fixed-width text files lezen

- Tekst lezen met Substring() en spaties wegdoen met Trim()

```
using (StreamReader sr = new StreamReader("VastBestand.txt"))
{
    while (!sr.EndOfStream)
    {
        string lijn = sr.ReadLine();
        string veld1 = lijn.Substring(0, 19).Trim();
        string veld2 = lijn.Substring(20, 9).Trim();
        string veld3 = lijn.Substring(30, 14).Trim();
        Console.WriteLine($"{veld2} {veld1} werkt in {veld3}");
    }
}
```

Exceptions

- In namespace System.IO zitten de volgende exceptions (gaan we opvangen met try-catch!!!)
- IOException:
 - Base class voor de andere IO-gerelateerde exceptions hieronder.
 - Als je niet kan lezen/schrijven naar bestand.
- FileNotFoundException:
 - Afgeleid van IOException.
 - Als het bestand niet gevonden wordt.
- OutOfMemoryException:
 - Afgeleid van IOException.
 - Als er niet genoeg geheugen beschikbaar is.

Exceptions

- Try catch gebruiken om fouten op te vangen

```
try
{ // Opent het bestand om te lezen. In de try kan een fout plaatsvinden
  using (StreamReader sr = File.OpenText("mijnbestand.txt"))
  {
    while (!sr.EndOfStream)
    {
      Console.WriteLine(s);
    }
  }
}
catch (FileNotFoundException ex) // de juiste catch vangt de fout op
{
  MessageBox.Show(ex.Message, "Foutmelding", MessageBoxButtons.OK, MessageBoxIcon.Warning);
}
catch (Exception) // Zet altijd de meest algemene exception onderaan!!!
{
  MessageBox.Show("Kan bestand niet vinden.", "Foutmelding", MessageBoxButtons.OK,
    MessageBoxIcon.Warning);
}
```

Directory

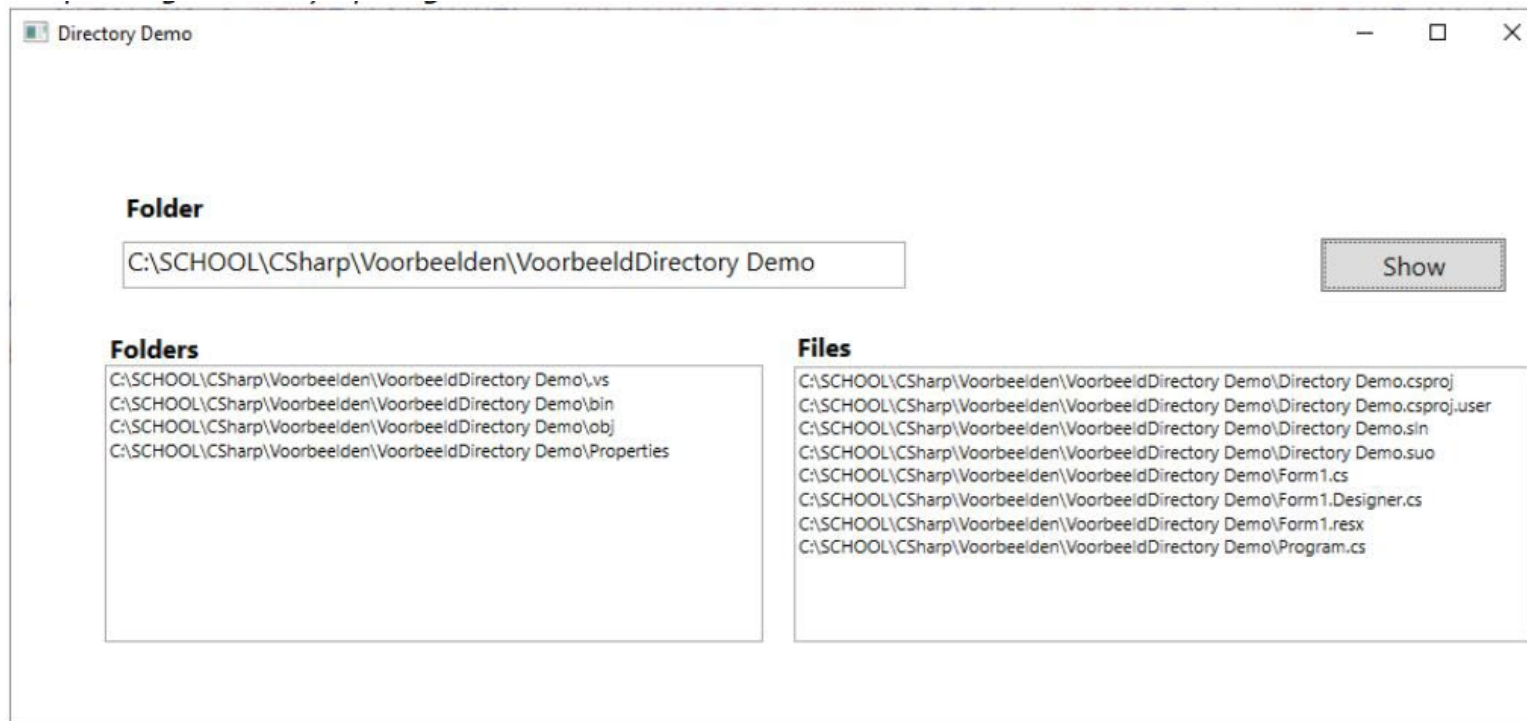
- Bestand opvragen uit speciale folder

```
// Bestand opvragen: het pad is Mijn Documenten (My Documents)  
// en het bestand is mijnbestand.txt en bevindt zich in My Documents  
string pad = Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments);  
string bestand = Path.Combine(pad, "mijnbestand.txt");
```

- Wanneer met Directory werken?
 - Als je bestanden wil opzoeken of naam wil wijzigen.
 - Als je geen bestanden wil bewerken

Directory

- Voorbeeld: een Directory opvragen
 - We geven een map in in de textbox TxtMap
 - We tonen links alle submappen en rechts alle bestandsnamen in de map.



Directory

- Voorbeeld: een Directory opvragen

```
// Toont alle mappen
string[] mappen = Directory.GetDirectories(TxtMap.Text);
foreach (string dir in mappen)
{
    TxtMappen.AppendText(dir);
    TxtMappen.AppendText(Environment.NewLine);
}
// Toont alle bestandsnamen
string[] bestanden = Directory.GetFiles(TxtMap.Text);
foreach (string file in bestanden)
{
    TxtBestanden.AppendText(file);
    TxtBestanden.AppendText(Environment.NewLine);
}
```


Standaard dialoogvensters

- OPMERKING: Dit is Windows-specifiek
⇒ kies voor .NET Framework Project (WPF) en niet .NET Core (WPF/Console)!!!
- OpenFileDialog:
Dialoogvenster om bestanden te gaan openen
- SaveFileDialog:
Dialoogvenster om bestanden te gaan opslaan
- Includeer ook de juiste namespace!!!

```
using Microsoft.Win32;
```

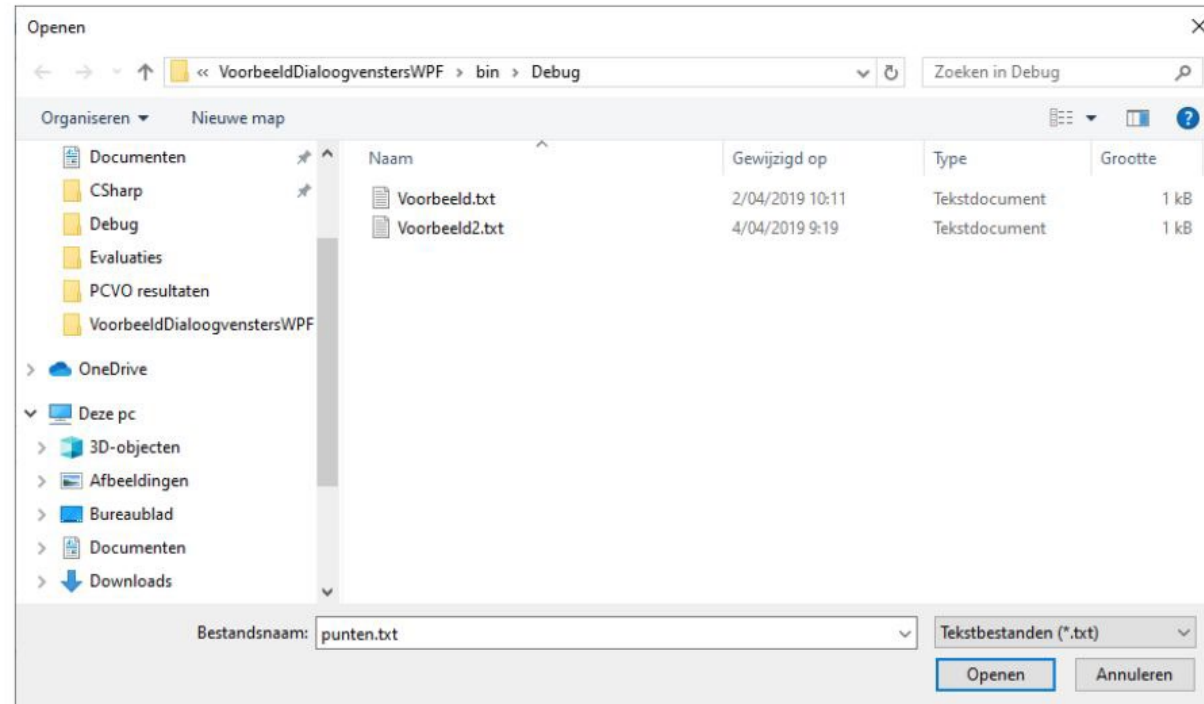
OpenFileDialog

```
OpenFileDialog ofd = new OpenFileDialog
{
    Filter = "Alle bestanden (*.*)|*.*|Tekstbestanden (*.txt) |*.txt",
    FilterIndex = 2, // index start vanaf 1, niet 0 hier! 2 wil zeggen hier
                    // filteren op .txt
    FileName = "punten.txt",
    Multiselect = true, // je kan meerdere bestanden selecteren (true, anders
                        // false)
    InitialDirectory =
        Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) // start
        in My Documents
    // == OF ==
    // InitialDirectory = System.IO.Path.GetFullPath(@"..\..\Bestanden"), //
    // volledig pad
    // == OF ==
    // InitialDirectory = Environment.CurrentDirectory // onder onze \Debug map
};
```

OpenFileDialog

```
if (ofd.ShowDialog() == true) // als de OpenFileDialog getoond kan worden
{
    // volledig pad en bestandsnaam opvragen
    string padEnBestandsnaam = ofd.FileName;
    // enkel pad opvragen
    string pad = System.IO.Path.GetDirectoryName(ofd.FileName);
    // enkel bestandsnaam opvragen
    string bestandsnaam = System.IO.Path.GetFileName(ofd.FileName);
    // Lijst van bestanden opvragen voor een zeker pad
    string[] bestanden = System.IO.Directory.GetFiles(pad);
    // OF: string[] bestanden = ofd.FileNames;
    for (int i = 0; i < bestanden.Length; i++)
    {
        Console.WriteLine($"{bestanden[i]}");
    }
}
```

OpenFileDialog



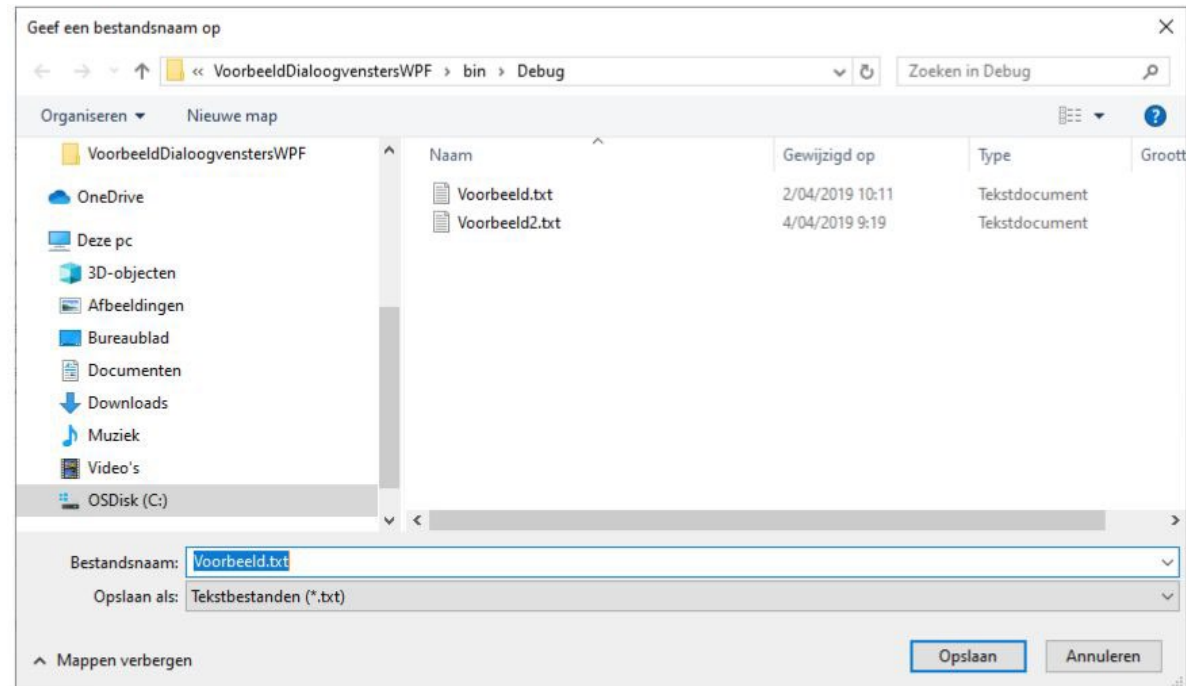
SaveFileDialog

```
SaveFileDialog sfd = new SaveFileDialog()
{
    Filter = "Alle bestanden (*.*)|*.*|Tekstbestanden (*.txt)|*.txt",
    FilterIndex = 2,
    Title = "Geef een bestandsnaam op",
    OverwritePrompt = true, // bevestiging vragen bij overschrijven van een bestand
    AddExtension = true, // extensie wordt toegevoegd
    DefaultExt = "txt", // standaard extensie
    FileName = "Voorbeeld.txt",
    InitialDirectory = Environment.CurrentDirectory // onder onze \Debug map
};
```

SaveFileDialog

```
if (sfd.ShowDialog() == true) // als de SaveFileDialog getoond kan worden
{
    // volledig pad en bestandsnaam opvragen
    string padEnBestandsnaam = sfd.FileName;
    // enkel pad opvragen
    string pad = System.IO.Path.GetDirectoryName(sfd.FileName);
    // enkel bestandsnaam opvragen
    string padEnBestandsnaam = System.IO.Path.GetFileName(sfd.FileName);
    // huidige map opvragen
    string huidigeMap = System.IO.Directory.GetCurrentDirectory();
}
```

SaveFileDialog



Opmerkingen

- In een using blok kan je ook meerdere variabelen gaan declareren

```
using (FileStream fs = new FileStream("tekst.txt", FileMode.Append, FileAccess.Write))  
using (StreamWriter sw = new StreamWriter(fs))  
{  
    sw.WriteLine("Hello!");  
}
```


Opmerkingen

- We stelden onze StreamWriter in op Append door er een FileStream object aan mee te geven

```
using (FileStream fs = new FileStream("tekst.txt", FileMode.Append, FileAccess.Write))  
using (StreamWriter sw = new StreamWriter(fs))  
{  
    sw.WriteLine("Hello!");  
}
```

- We kunnen dit ook zonder

```
using (StreamWriter sw = new StreamWriter("tekst.txt", true))  
{  
    sw.WriteLine("Hello!");  
}
```

- true meegeven: append
- false meegeven (gebeurt standaard al): overschrijven