

Hur hanterar jag textfiler?

Inledning

Klasserna `StreamWriter` och `StreamReader` har du användning för då du behöver läsa strängar från eller skriva strängar till textfiler. Det finns ett flertal klasser du kan använda för att hantera filer. `StreamWriter` och `StreamReader` kommer du finna enkla att använda, och fullt tillräckliga.

Skriva till en textfil

Genom att använda ett `StreamWriter`-objekt kan du skriva text till en ny fil, på följande sätt:

```
static void Main(string[] args)
{
    Console.WriteLine("***** Exempel med StreamWriter *****");

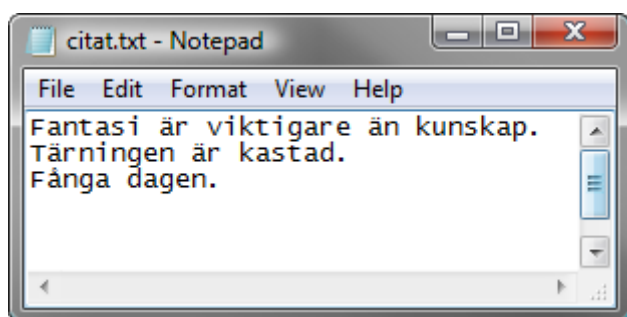
    // Skapar en StreamWriter-objekt och skriver strängar till
    // filen citat.txt.
    StreamWriter writer = new StreamWriter("citat.txt");
    writer.WriteLine("Fantasi är viktigare än kunskap. ");
    writer.WriteLine("Tärningen är kastad.");
    writer.WriteLine("Fånga dagen.");

    // Lägger till två nya tomma rader.
    writer.Write(writer.NewLine);
    writer.WriteLine();

    // Vid stängning anropas automatiskt Flush() och data som
    // behöver skrivas till filen skrivs till den.
    writer.Close();

    Console.WriteLine("Skapade en fil med några citat...");
}
```

Då programmet körs skapas en ny fil:



Figur 1. Innehållet i filen citat.txt.

Programmet ovan har en del brister. Felhantering saknas och programmet behöver kompletteras med en `try-catch`-sats. Ett annat problem som måste åtgärdas är att säkerställa att filen som öppnas stängs oavsett vad som inträffar. Ett enkelt sätt att säkerställa detta är att använda det reserverade ordet `using` på detta sätt:

```
static void Main(string[] args)
{
    Console.WriteLine("***** Exempel med StreamWriter *****");

    try
    {
        // Skapar en StreamWriter-objekt och skriver strängar.
        using (StreamWriter writer = new StreamWriter("citatt.txt"))
        {
            writer.WriteLine("Fantasi är viktigare än kunskap.");
            writer.WriteLine("Tärningen är kastad.");
            writer.WriteLine("Fånga dagen.");

            // Lägger till två nya rader.
            writer.Write(writer.NewLine);
            writer.WriteLine();
        }

        Console.WriteLine("Skapade en fil med några citat...");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Ett oväntat fel inträffade.\n{0}",
            ex.Message);
    }
}
```

Att använda using-satsen är ett säkert sätt att kontrollera ett StreamWriter-objekts livstid. StreamWriter-objektet förstörs när blocket med using-satsen avslutas. Innan StreamWriter-objektet förstörs anropas automatiskt Close varför du själv inte behöver stänga filen.

Läsa från en textfil

Klassen StreamReader, som du använder för att läsa data från en text, ärver en del av sin funktionalitet från den klassen TextReader.

Medlem	Kommentar
Peek()	Returnerar nästa tillgängliga tecken utan att ändra läsarens position. Värdet -1 indikerar att du är i slutet av strömmen (filen).
Read()	Läser data från strömmen.
ReadBlock()	Läser ett maximalt antal tecken från strömmen och skriver tecknen till en buffert, vid angivet index.
ReadLine()	Läser en rad med data från strömmen och returnerar data i form av en sträng. null indikerar EOF (end of file).
ReadToEnd()	Läser alla tecken från aktuell position till slutet av strömmen och returnerar dem som en enskild sträng.

Genom att använda ett StreamReader-objekt kan du nu läsa textdata från filen citatt.txt på följande sätt:

```
static void Main(string[] args)
{
    Console.WriteLine("***** Exempel med StreamReader *****");
```

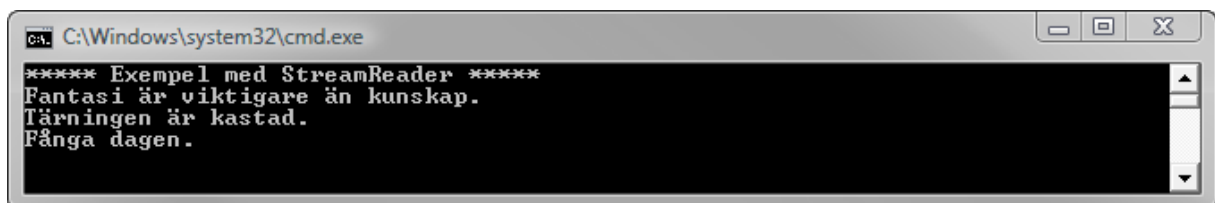
```
try
{
    // Skapar en StreamReader-objekt och läser rader.
    using (StreamReader reader = new StreamReader("citatt.txt"))
    {
        // Läser filen rad för rad.
        string line = null;
        while ((line = reader.ReadLine()) != null)
        {
            // Citatet skrivs ut i konsolfönstret.
            Console.WriteLine(line);
        }
    }
}
catch (Exception ex)
{
    Console.WriteLine("Ett oväntat fel inträffade.\n{0}",
        ex.Message);
}
```

Satsen

```
while ((line = reader.ReadLine()) != null)
```

läser en rad från textfilen och lagrar den i string-variabeln `line`. Då det inte finns flera rader att läsa returnerar `ReadLine` `null` och "while"-satsen bryts.

Då programmet körs visas följande i konsolfönstret:



Figur 2.