Ditt första C#-program med Visual Studio

Upphovsrätt för detta verk

Detta verk är framtaget i anslutning till kursen Inledande programmering med C# vid Linnéuniversitetet.

Du får använda detta verk så här:

Allt innehåll i verket Ditt första program med Visual Studio av Mats Loock, förutom Screen Beans samt Linnéuniversitetets logotyp och symbol, är licensierad under:

Creative Commons Erkännande-IckeKommersiell-DelaLika 2.5 Sverige licens.
http://creativecommons.org/licenses/by-nc-sa/2.5/se/

Det betyder att du i icke-kommersiella syften får:

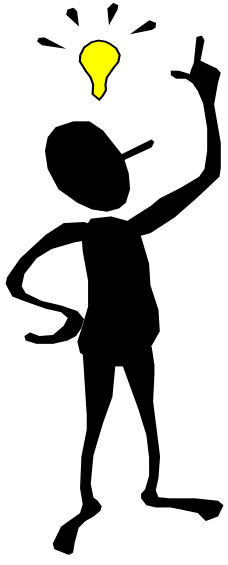
- kopiera hela eller delar av innehållet
- sprida hela eller delar av innehållet
- visa hela eller delar av innehållet offentligt och digitalt
- konvertera innehållet till annat format
- du får även göra om innehållet

Om du förändrar innehållet så ta inte med Screen Beans samt Linnéuniversitetets logotyp och symbol i din nya version!

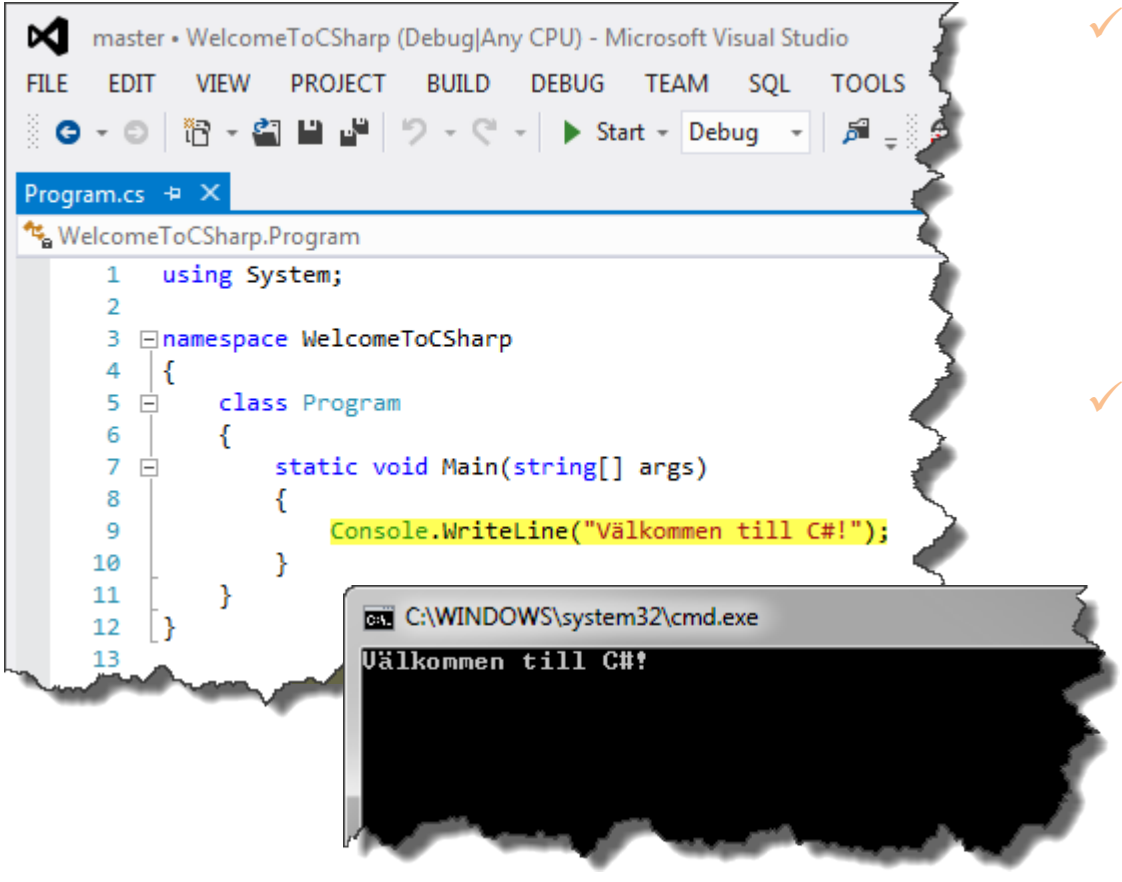
Vid all användning måste du ange källan: ”Linnéuniversitetet – Inledande programmering med C#” och en länk till https://coursepress.lnu.se/kurs/inledande-programmering-med-csharp och till Creative Common-licensen här ovan.

Hur skriver jag ut text i konsolfönstret?

- ✓ Problem
  - Skriv ett C#-program som skriver ut "Välkommen till C#!" i konsolfönstret.
- ✓ Analys
  - Ett C#-program ska skrivas.
  - Källkod måste skrivas och sparas i en textfil med filändelsen `.cs`.
  - En klass måste skapas och innehålla metoden `Main`.
  - Texten "Välkommen till C#!" ska skrivas ut i konsolfönstret.
  - Källkoden måste kompileras, d.v.s. översättas till CIL-kod.
  - Programmet måste köras.
- ✓ Algoritm
  - Skriv ut en rad med strängen "Välkommen till C#!".



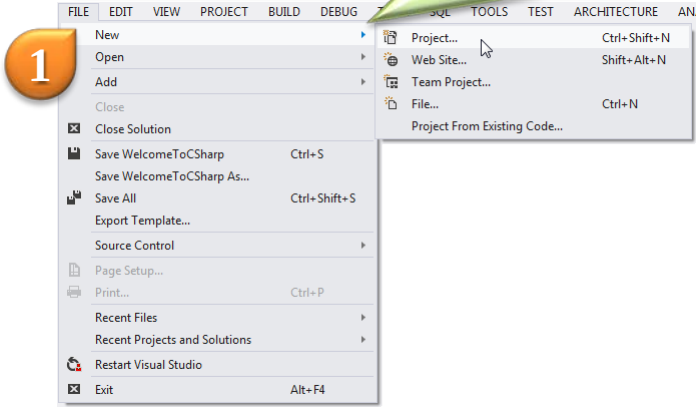
Det färdiga programmet



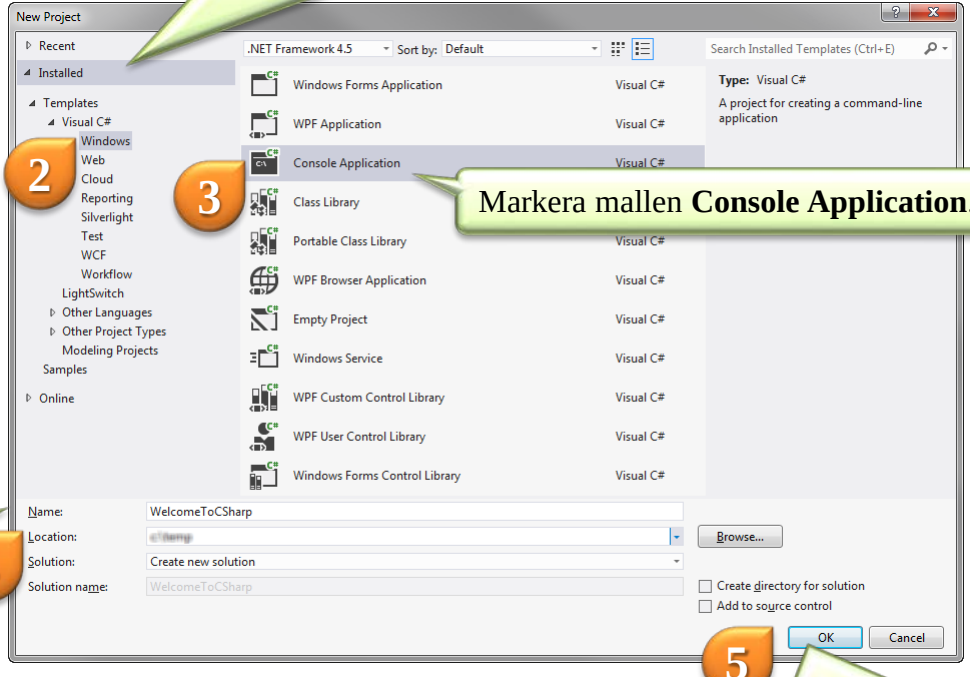
- Ett C#-program är ett datorprogram som körs då du skriver namnet på programmet så att CLR:n laddar och ”jittar” programmet för att sedan exekvera det.
- Det är den markerade raden som gör jobbet i programmet – nämligen att presentera meddelandet `Välkommen till C#!` i ett konsolfönster.

Så skapar du ett nytt projekt

Du startar Visual Studio och väljer **File**, **New** och **Project…**.



Dialogrutan **New Project** visas. Under **Installed, Templates** expanderar du **Visual C#** och markerar **Windows**.

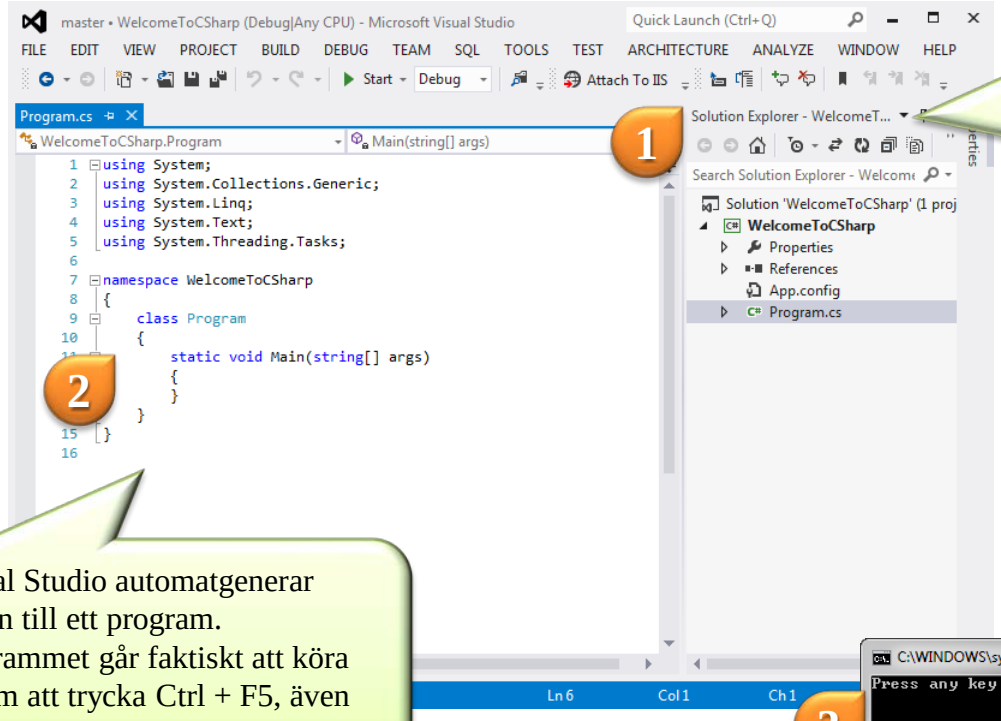


Markera mallen **Console Application**.

Du skriver in namnet på projektet vid **Name** och väljer i vilken katalog du vill projektet ska skapas vid **Location**.

Sedan stänger du dialogrutan genom att klicka på knappen **OK**.

Detta får du



- I fönstret **Solution Explorer** hittar du filer som tillhör projektet.
- Överst ser du projektets namn **WelcomeToCSharp**. Det är det namn som den exekveringsbara filen (programmet) kommer att få.
- Projektet har en fil som innehåller källkod, `Program.cs`. Det är den filen som är öppen och kan redigeras.

Visual Studio automatgenerar början till ett program. Programmet går faktiskt att köra genom att trycka Ctrl + F5, även om…

…det inte är mycket som händer.

Program.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace WelcomeToCSharp
{
    class Program
    {
        static void Main(string[] args)
        {
        }
    }
}
```

Metoden `Main` är speciell, den är programmets startpunkt.

Det är här du placera koden som skriver ut ”Välkommen till C#!”.

(Det som är suddigt är inte helt ovidkommande. Du kommer att lära dig mer om detta senare…)

Program.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace WelcomeToCSharp
{
    class Program
    {
        static void Main(string[] args)
        {
            Console
        }
    }
}
```

Du använder klassen `Console` för att mata ut text i ett konsolfönster.

`Console` är en inbyggd klass som innehåller (statiska) metoder för att presentera text i ett konsolfönster och ta emot inmatning från tangentbordet.

Program.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace WelcomeToCSharp
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine();
        }
    }
}
```

Metoden `WriteLine` använder du då du vill skriva ut något i ett konsolfönster, som t.ex. ett meddelande.

Det du vill mata ut placerar du mellan parenteserna.

`WriteLine` finns i flera versioner för presentation, eller utmatning, av olika typer av data, t.ex. bokstäver och tal av olika slag.

Program.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace WelcomeToCSharp
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Välkommen till C#!");
        }
    }
}
```

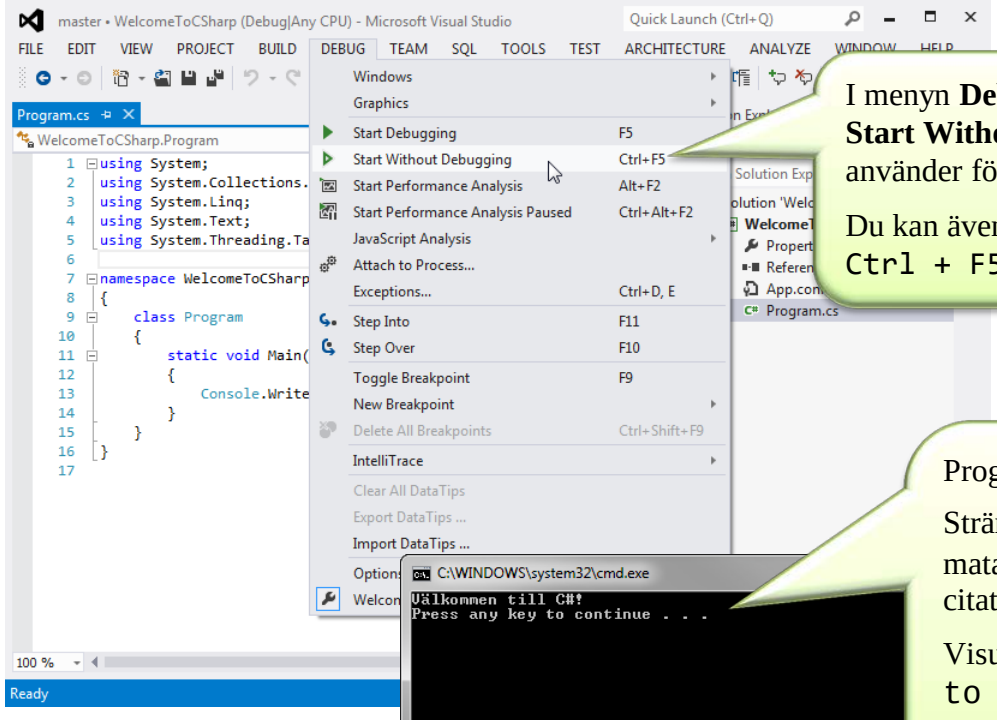
Strängen `"Välkommen till C#!"` skriver du mellan parenteserna.

Det som står mellan parenteserna kommer att matas ut i konsolfönstret.

Hela raden, som avslutas med `;`, kallas en sats. Alla satser avslutas med `;`.

- ✓ …och nu är programmet färdigt för att köras!

Så här kör du programmet



I menyn **Debug** hittar du kommandot **Start Without Debugging**, som du använder för att köra programmet.

Du kan även använda kortkommandot `Ctrl + F5`.

Programmet körs i ett konsolfönster.

Strängen `Välkommen till C#!` har matats ut av programmet. Lägg märke till att citattecknen `"` inte matats ut.

Visual Studio lägger till `Press any key to continue`.... Då du trycker på en tangent på tangentbordet kommer konsolfönstret att stängas.

…men vad sa du nu att Main var?

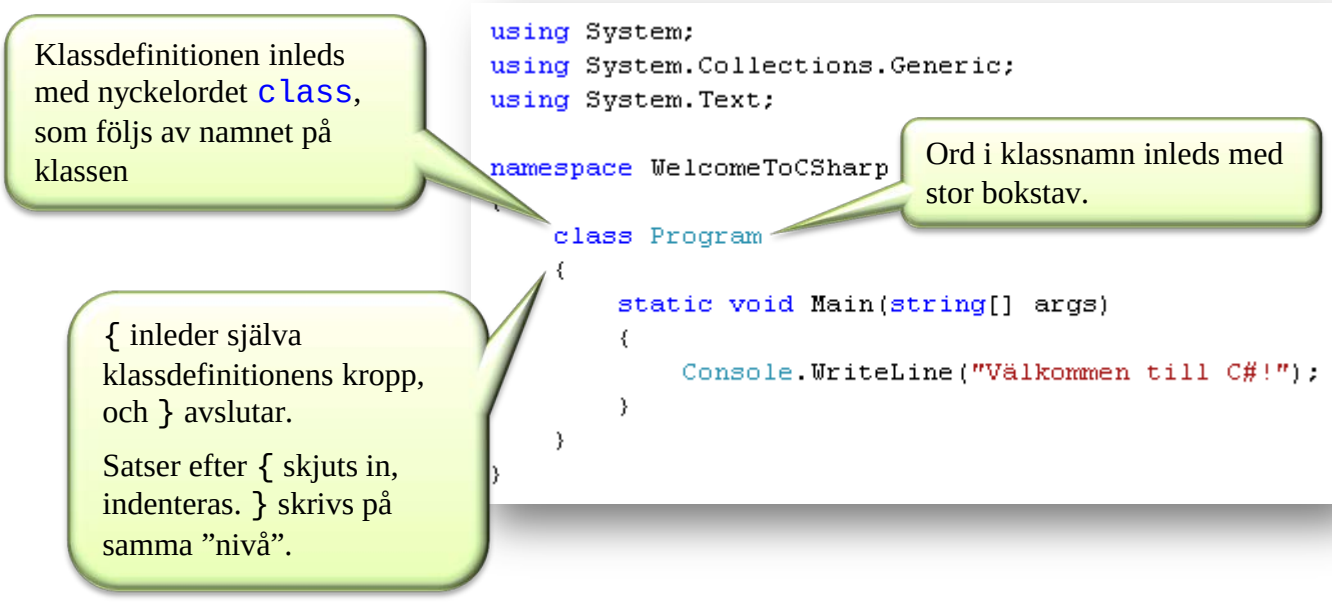
```
using System;
using System.Collections.Generic;
using System.Text;

namespace WelcomeToCSharp
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Välkommen till C#!");
        }
    }
}
```

`Main` är metoden där exekveringen av programmet börjar.

- Alla C#-program måste innehålla en metod som heter `Main`, annars kan inte programmet starta.
- `Main`-metodens första rad måste se ut exakt som den gör. (Vad `static`, `void`, `string[]` och `args` betyder kommer du att lära dig senare.)
- `{` inleder metodkroppen, och `}` avslutar den.

…men vad innebär class?



- ✓ Alla C#-program består av minst en klassdefinition…
- ✓ …som innehåller en metod som kallas `Main`, annars kommer inte programmet att starta.

…men varför är klassen placerad i en namnrymd (*namespace*)?

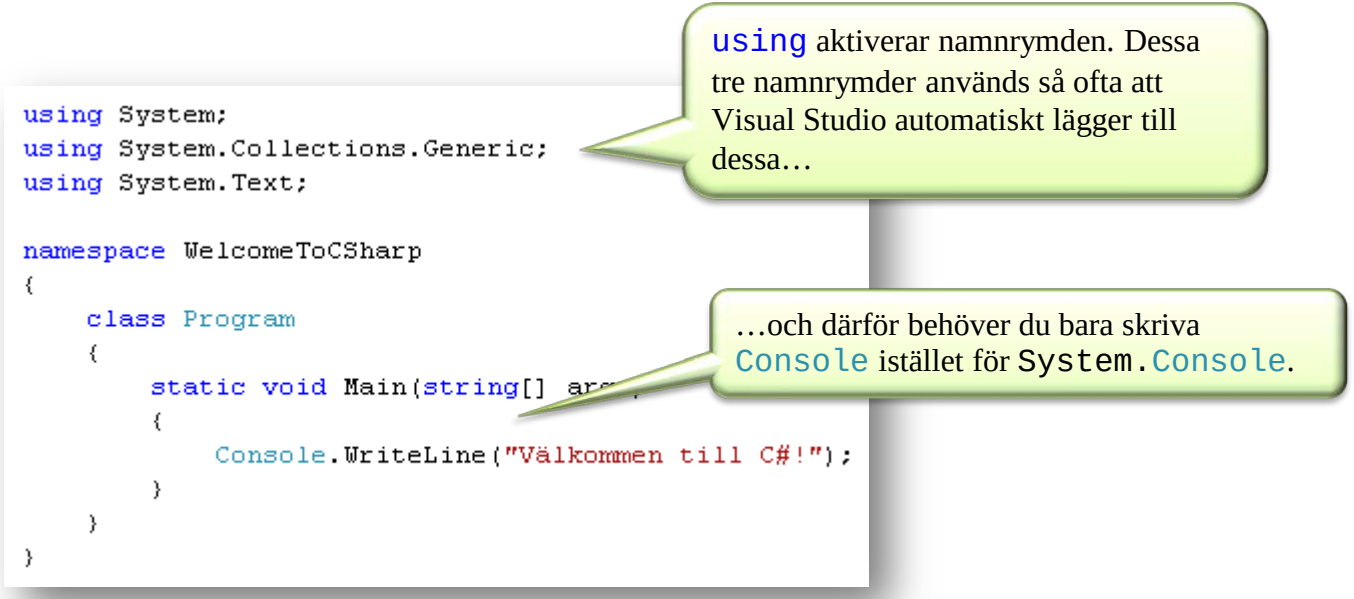
```
using System;
using System.Collections.Generic;
using System.Text;

namespace WelcomeToCSharp
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Välkommen till C#!");
        }
    }
}
```

Klassen `Program` skapas i namnrymden `WelcomeToCSharp` (samma namn som projektet!).

- Namnrymden (*namespace*) är till för att du ska slippa namnkonflikter. Två klasser med samma namn kan inte förväxlas om de finns i olika namnrymder. Alla klasser ska placeras i en namnrymd.
- Du kan referera till klassen `Program` med `WelcomeToCSharp.Program` och skulle fler klasser skapas som också heter `Program` kommer programmet fortfarande att fungera eftersom klassen `WelcomeToCSharp.Program` används.

…och varför används using?



- Genom att använda `using System;` kan du skriva

```
Console.WriteLine("Välkommen till C#!");
```

istället för

```
System.Console.WriteLine("Välkommen till C#!");
```

där `System.Console` är det fullständiga namnet. Du behöver helt enkelt inte skriva så mycket!

Sammanfattning

- ✓ Alla C#-program består av minst en klassdefinition.

```
class Program { … }
```

- ✓ Det måste finnas en metod som heter **`Main`** för att det ska gå att köra programmet.

```
static void Main(string[] args) { … }
```

- ✓ Med `Console.WriteLine` kan du skriva ut en rad med text i konsolfönstret.

```
Console.WriteLine("Välkommen till C#!");
```