Av **Erik ”Zkall” Svensson**
@esdev.se

C# Class

Vad är en klass? Enligt wikipedia:

Klass är i objektorienterad programmering ett avsnitt programkod som samlar en mängd relaterade attribut och funktioner, även kallat metoder. Det är ett viktigt kännetecken för en klass att dess inre struktur inte är tillgänglig utanför klassen, utan den kan enbart manipuleras genom ett specificerat gränssnitt. Fenomenet att en klass privata delar ej är tillgängliga utanför kallas inkapsling (eng. encapsulation). En klass kan sägas vara en användardefinierad datatyp, som alltså kompletterar de fördefinierade datatyperna, i C++ till exempel **int** och **char**. För att de klasserna skall likna just användardefinierade datatyper använder man i vissa språk överlagring av operatorer för klasser.

Varför klass?

Enkelt, smidigt, renare kod. Gör koden mer läsbar.

Hur ser en klass ut?

```
   public class ChattVM
    {         public virtual int Id { get; set; }
public virtual UserVM Från { get; set; }

        public virtual string Meddelande { get; set; }

        public virtual UserVM Till { get; set; }

        public virtual DateTime datum { get; set; }
}
```

```
Så kan en klass se ut, denna använder bara propertys och är en så kallad viewmodel.
Men en class kan innehålla metoder och diverse annat med!
```

Skapar objekt

Varje class är ett skapat objekt som är konfigurerat efter behov. I exemplet ovan ser ni att den refererar till andra classer som variabel typ. UserVM i detta fall. Och jepp det går att göra.

Du kan även skapa interfaces från en klass genom att högerklicka refactor -> extract -> extract interface.

Exempel klass med metoder

```
static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
[STAThread]         static
void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
```

Taget ur en windowsform app.

Arv

För att signalera att en class ärver från en annan class skriver man

`public class DB : DbContext` klassnamn : arvklassnamn

Implementation av interface

Skrivs precis likadant som ett arv, dock **EJ** samma sak. `public class ChattBusinessLogicLayer : IchattBusinessLogicLayer`