# C#- Kommentarer och dokumentation

## Dokumentation

Att dokumentera sin kod är bland det viktigaste en utvecklare bör göra, dels för att själva hålla reda på vad ens stora kodblock gör men även för att underlätta för andra utvecklare som är delaktiga i samma projekt att veta vad som försiggår. Att dokumentera sin kod är även väldigt bra för nybörjare att enklare förstå sig på sin kod då denne får genom sina kommentarer förklara för sig själv vad koden gör.

## Kommentarer

För att dokumentera vår kod använder vi oss av kommentarer. När vi skapar en kommentar berättar vi för programmet att koden inuti kommentaren inte ska ses som kod och ska ignoreras.

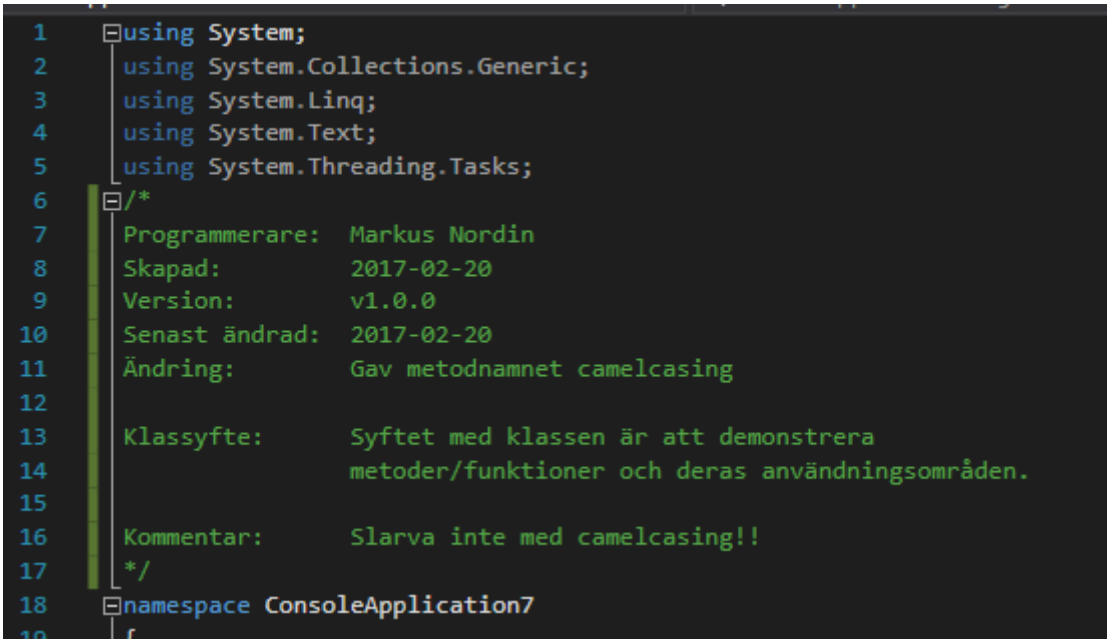
### Olika typer av kommentarer

- // - Kommentar som reserverar en hel rad
- /* */ - Kommenterar allt inom ”/*” och ”*/”
- /// - XML-tag kommentering.

Den vanligaste typen av kommentar ni kommer att använda är ”//” då den är enklast att använda. ”/* */”-kommentering är bra att använda sig av när man behöver använda många rader. ”///” är lite mer ovanlig då man kan använda sig av XML-taggar för att kommentera, detta kommer jag inte att gå in på, men det är bra att lära sig.

## Klasskommentar

Jag börjar alltid mina klasser med klasskommentarer. I dessa skriver jag vem som utvecklar klassen, vilket datum den skapades, vilket datum den ändrades senast, vad som ändrades, version, klassens syfte samt eventuella kommentarer. Detta underlättar väldigt mycket då man slipper kolla igenom rad för rad vad klassen gör och kan istället bara läsa klasskommentaren. För dessa kommentarer använder jag ”/* */” kommentaren. Klasskommentarer lägger jag längst upp mellan alla using-deklarationer och början på namespace.



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
/*
Programmerare:  Markus Nordin
Skapad:         2017-02-20
Version:        v1.0.0
Senast ändrad:  2017-02-20
Ändring:        Gav metodnamnet camelcasing

Klassyfte:      Syftet med klassen är att demonstrera
                metoder/funktioner och deras användningsområden.

Kommentar:      Slarva inte med camelcasing!!
*/
namespace ConsoleApplication7
```

## Metodkommentar

Varje metod bör ha en kommentar som förklarar vad syftet med den är.

```
// Metoden tar in två variabler av typen int
// returnerar värdet av a + b
private static int secondMethod(int a, int b)
{
    return a + b;
}

// Metod med returtyp string,
// returnerar en simpel sträng
public static string thirdMethod()
{
    return "Jag kommer från den tredje metoden!";
}

// Metod med returtyp double
// returnerar summan av två flyttal
private static double fourthMethod()
{
    return 1.234 + 5.678;
}
```

### Line-kommentar

Har vi kod som bör förklaras ytterligare så kommenterar vi den koden var för sig. Detta är väldigt bra för nybörjare att göra då man i kommentarerna förklarar koden för sig själv.

```
// Metod utan returtyp
// Metoden ansvarar för att anropa de andra metoderna
// och skriva ut värdena till vår konsol.
public static void firstMethod()
{
    // Anropar secondMethod med argumenten 2 och 4
    // Sedan skriver vi ut resultatet i konsolen.
    Console.WriteLine(secondMethod(2, 4));

    // Anropar vår tredje metod och skriver ut till konsolen
    Console.WriteLine(thirdMethod());

    // Anropar vår privata metod och skriver ut resultatet.
    Console.WriteLine(fourthMethod());
}
```

Dyker man på ny kod är det väldigt bra om man förklarar noggrant vad koden gör så att man lättare lär sig.

Exempel:
```
// Skapa random-objekt
Random rng = new Random();
// Slumpa ett tal mellan 1 till 10 och spara
// talet i slumpatTal
int slumpatTal = rng.Next(1,10);
```

## Kommentera hellre för mycket än för lite!!!