

C# - Generics

Vi låtsas att vi ska skapa 3 olika klasser. En som ska spara namn(*strings*) och kunna skriva ut dessa på kommando, en för att spara åldrar(*int*) och skriva ut dessa på kommando och en annan för att spara olika vikter(*float*) och skriva ut dessa på kommando. Med hjälp av generics behöver vi bara skapa en klass och återanvända den genom att vid skapandet av objektet specificera vilken typ vi vill att den ska vara.

Generiska klasser har "Type" parametrar ("*<T>*") som läggs till efter klassens namn. Exempel: `class GeneriskKlass<T> { }`

Denna parameter ändrar vi till den typ vi vill använda när vi skapar ett objekt av klassen. Ett exempel på en generisk klass är *List<T>*. *List<T>* används för att dynamiskt lagra variabler och när vi skapar ett objekt av *List<T>* så måste vi byta ut "T" mot den typ vi vill att listan ska lagra, exempelvis "*List<int>*" för en lista som lagrar integer.

Att skapa en generisk klass

I detta exempel skapar vi en generisk klass som tar in vilken typ som helst och har en metod för att skriva ut typens värde samt vilken typ det är.

```
class SimpleGenerics<T>
{
    T anyType;

    public SimpleGenerics(T anyType)
    {
        this.anyType = anyType;
    }

    public void PrintValue()
    {
        Console.WriteLine($"Värdet är {anyType} och " +
            $"är av en {anyType.GetType()}");
    }
}
```

I bilden ovan har vi skapat den generiska klassen "*SimpleGenerics*". Den har en generisk instansvariabel som vi kallar *anyType* som får ett värde i konstruktorn som tar in ett generiskt värde. Klassen har en metod, *PrintValue()*, vars enda uppgift är att skriva ut värdet på *anyType* och skriva ut vilken typ värdet består av.

Att använda en generisk klass

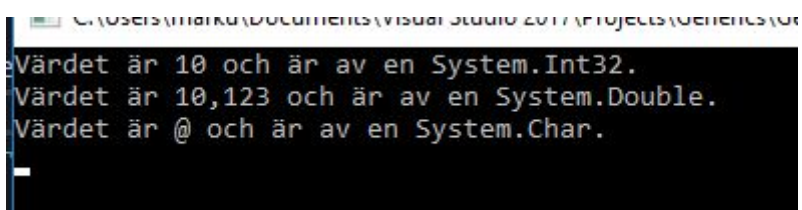
Nu när vi har skapat vår generiska klass kan vi använda den. Vi skapar tre objekt för tre olika typer, *int*, *double* och *char*, tilldelar värde i konstruktorn och anropar klassens enda metod.

```
static void Main(string[] args)
{
    // Skapa objekt av vår generiska klass med olika typsättningar
    SimpleGenerics<int> intKlass = new SimpleGenerics<int>(10);
    SimpleGenerics<double> doubleKlass = new SimpleGenerics<double>(10.123);
    SimpleGenerics<char> charKlass = new SimpleGenerics<char>('@');

    intKlass.PrintValue();
    doubleKlass.PrintValue();
    charKlass.PrintValue();

    Console.ReadLine();
}
```

Resultat:



```
C:\Users\marku\Documents\visual studio 2017\Projects\Generics\Ge
Värdet är 10 och är av en System.Int32.
Värdet är 10,123 och är av en System.Double.
Värdet är @ och är av en System.Char.
```

Slutsas

Med hjälp av generiska klasser har vi nu kunnat skapa tre olika objekt med olika typer från en och samma klass.