GENERICS

WAS SIND GENERICS?

In den letzten Videos ging es um das Thema Generics. Diese machen unseren Code um einiges dynamischer und in diesem Dokument fassen wir das Ganze nochmal zusammen.

In der Programmierung dienen Variablen als Platzhalter für Daten und jeder Wert muss in C# von einem bestimmten Datentyp sein. Generics dienen auch als Platzhalter, allerdings nicht für Werte, sondern für Datentypen. Sie kommen dann zum Einsatz, wenn man zum Zeitpunkt der Entwicklung noch nicht weiß, welchen Datentyp man für einen bestimmten Wert verwenden möchte. Ein gutes Beispiel hierfür sind die Collections-Klassen, welche wir im letzten Kapitel schon kennengelernt haben. Bei diesen konnten wir über die spitzen Klammern bestimmen, welcher Datentyp für die Collection verwendet wird. Und genau das sind Generics!

WIE DEFINIERT MAN EINEN GENERIC

Generics definiert man schon direkt bei der Klassendefinition, ganz oben am Bezeichner.

```
class WerteBehälter<T>
{

}
```

Direkt neben dem Klassenbezeichner wurde ein spitzes Klammerpaar gesetzt, welches den Generic definiert. Das T ist der Bezeichner für unser Generic (T steht in diesem Fall für Type).

Innerhalb der Klassendefinition kann man diesen Generic nun verwenden und wie einen Datentyp behandeln.

```
class WerteBehälter<T>
{
    private T meinWert;
}
```

Wie du siehst hat die Variable „meinWert“ als Datentyp den Generic T angegeben. Später wenn wir ein Objekt von dieser Klasse erstellen, müssen wir einen Datentyp für den Generic wählen und „meinWert“ hat dann genau diesen Typ.

Man kann den Generic auch bei Methoden als Parameter nutzen wie du am Beispiel von diesem Konstruktor siehst:

```
class WerteBehälter<T>
{
    //Variable
    private T meinWert;

    //Konstruktor
    public WerteBehälter(T _wert)
    {
        meinWert = _wert;
    }
}
```

Wenn du den Wert innerhalb von einer Methode nun in der Konsole ausgeben möchtest, dann musst du unbedingt die Methode „ToString()“ verwenden, da das Objekt sonst nicht als Text dargestellt werden kann.

```
//Methode
public void Ausgabe()
{
    Console.WriteLine(meinWert.ToString());
}
```

Das erstellen eines Objekts von einer generischen Klasse funktioniert dabei so, wie du das schon von den Collections gewohnt bist. Neben dem Typ (dem Klassennamen) muss bei der Deklaration einer Variable auch der Datentyp für den Generic angegeben werden. Dies geschieht zwischen den spitzen Klammern.

```
static void Main(string[] args)
{
    WerteBehälter<string> name = new WerteBehälter<string>("Janek");
    name.Ausgabe();

    Console.WriteLine();
}
```