ABSTRAKTE KLASSEN UND METHODEN

WAS SIND ABSTRAKTE KLASSEN?

In diesem Modul ging es um Abstrakte Klassen und Methoden. Um das Thema nochmal richtig zu verinnerlichen, wird in diesem Dokument alles nochmal zusammengefasst. Fangen wir also an mit der Frage: *Was sind Abstrakte Klassen eigentlich?*

Eine abstrakte Klasse ist **eine nicht instanziierbare Klasse**, welche nur dazu dient geerbt zu werden. Man kann also keine Objekte von einer abstrakten Klasse erstellen. Wie Interfaces auch schon dienen abstrakte Klassen nur als Vorlagen, die normalen Klassen erben können. Eine abstrakte Klasse dient meist als eine sogenannte **Basisklasse**.

Hier ist mal ein Beispiel für eine einfache abstrakte Klasse:

```
abstract class Arbeiter
{
    //Eigenschaften
    public string Name { get; set; }
    public double Gehalt { get; set; }

    //Methoden
    public abstract void ArbeitVerrichten();
}
```

Diese Klasse beschreibt ganz einfach einen Arbeiter. Ein Arbeiter hat 2 Eigenschaften (Name und Gehalt) und eine Methode die ihn seine Arbeit verrichten lässt. Wie du siehst ist die Methode „ArbeitVerrichten()“ eine abstrakte Methode und sie enthält keinerlei Code. Darauf kommen wir gleich noch genauer zu sprechen.

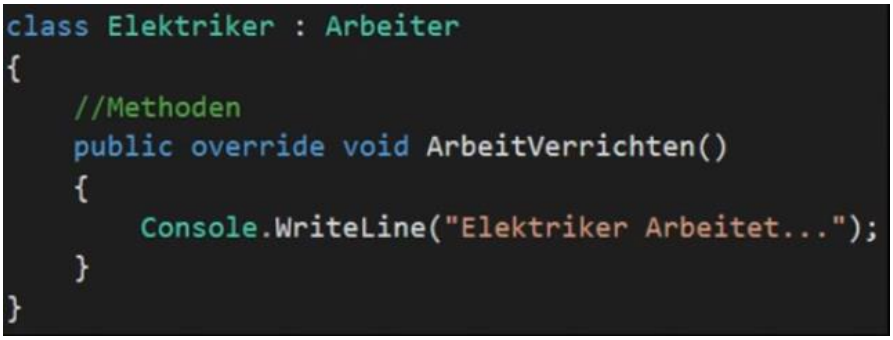
Du siehst allerdings, dass eine abstrakte Klasse, wie ein Interface auch, eine Vorlage für andere Klassen darstellt. Da stellt man sich doch die Frage wo der Unterschied zwischen Interfaces und abstrakten Klassen ist oder? Keine Sorge, das ist eigentlich ganz einfach: **Während ein Interface eine nicht auscodierte Vorlage darstellt, sind abstrakte Klassen vollfunktionsfähig. Man kann in abstrakten Klassen also Methodenkörper schreiben, Zugriffsmodifizierer nutzen, usw... Sie sind im Kern ganz normale Klassen, mit der Besonderheit, dass man von diesen keine Objekte erstellen kann.**

WAS SIND ABSTRAKTE METHODEN?

Wir haben im obigen Codeschnipsel schonmal kurz die „ArbeitVerrichten()“-Methode angesprochen. Diese beinhaltet in der Definition das Schlüsselwort „abstract“ und ist somit eine abstrakte Methode.
Abstrakte Methoden sind im Grunde einfach nur Methoden, welche erst von der erbenden Klasse implementiert werden. Im Beispiel unserer „Arbeiter“-Klasse ist es ja so, dass jeder Arbeiter natürlich seine Arbeit verrichten muss. Wie die verschiedenen Arten von Arbeiter ihre Arbeit verrichten ist allerdings unterschiedlich. Es gibt verschiedene Berufe und dementsprechend auch verschiedene Klassen die von der abstrakten „Arbeiter“-Klasse erben könnten. Dementsprechend wäre es unlogisch, die „ArbeitVerrichten()“-Methode schon in dieser Klasse selbst zu definieren, da diese Methode in jeder Arbeiter-Klasse anders ausfallen wird.

VON ABSTRAKTEN KLASSEN ERBEN

Um von einer abstrakten Klasse zu erben müssen wir lediglich den „:“ bei der Klassendefinition setzen und dahinter die zu erbende abstrakte Klasse schreiben.



```
class Elektriker : Arbeiter
{
    //Methoden
    public override void ArbeitVerrichten()
    {
        Console.WriteLine("Elektriker Arbeitet...");
    }
}
```

Wie du siehst ist das ganze also ziemlich einfach. In diesem Codeschnipsel sehen wir eine weitere Beispielklasse namens „Elektriker“. „Elektriker“ erbt von der abstrakten Klasse „Arbeiter“ und beinhaltet somit automatisch die Eigenschaften „Name“ und „Gehalt“. Außerdem siehst du in diesem Codeschnipsel wie die „ArbeitVerrichten()“-Methode überschrieben wird (also voll auscodiert wird) mithilfe des „override“-Schlüsselwortes. Dieses muss man immer dann setzen wenn man eine als „abstract“ markierte Methode voll implementieren möchte.