Programmieren II – Schlüsselwörter

**Zugriffsmodifizierer** sind Schlüsselwörter, die verwendet werden, um die deklarierte Zugriffsart eines Members oder Typs anzugeben.

* [public](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/public): Der Zugriff ist nicht beschränkt.
* [protected](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/protected): Der Zugriff ist auf die enthaltende Klasse oder auf Typen beschränkt, die von der enthaltenden Klasse abgeleitet sind.
* [internal](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/internal): Der Zugriff ist auf die aktuelle Assembly (C# Projekt) beschränkt.
* [protected internal](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/protected-internal): Der Zugriff ist auf die aktuelle Assembly oder auf Typen beschränkt, die von der enthaltenden Klasse abgeleitet sind.
* [private](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/private): Der Zugriff ist auf den enthaltenden Typ beschränkt.
* [private protected](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/private-protected): Der Zugriff ist auf die enthaltende Klasse oder auf Typen beschränkt, die von der enthaltenden Klasse innerhalb der aktuellen Assembly abgeleitet sind.

struct: Ein struct-Typ ist ein ein Werttyp, der in der Regeln verwendet wird, um eine kleine Gruppe verwandter Variablen zusammenzufassen, z. B. Koordinaten eines Rechtecks oder die Merkmale eines Lagerartikels. Strukturen können eine Schnittstelle implementieren, aber nicht von einer anderen Struktur erben. Aus diesem Grund können Strukturmember nicht als protected deklariert werden.

abstract: Der abstract-Modifizierer gibt an, dass dem modifizierten Objekt eine Implementierung fehlt oder dass diese unvollständig ist. Der abstract-Modifizierer kann für Klassen, Methoden, Eigenschaften, Indexer und Ereignisse verwendet werden. Verwenden Sie den abstract-Modifizierer in einer Klassendeklaration, um anzugeben, dass die Klasse nur die Basisklasse für eine andere Klasse sein soll. Als abstrakt markierte Member oder Member in einer abstrakten Klasse **müssen** von Klassen, die von der abstrakten Klasse abgeleitet wurden, implementiert werden. Verwendung: Sicherstellen das diese Methode implementiert wird, obwohl Implementierung zu aktuellem Zeitpunkt nicht festliegt.

Abstrakte Klassen weisen die folgenden Funktionen auf:

* Eine abstrakte Klasse darf nicht instanziiert werden.
* Eine nicht abstrakte Klasse, die von einer abstrakten Klasse abgeleitet wurde, muss Implementierungen aller geerbten abstrakten Methoden und Accessoren enthalten.

Abstrakte Methoden weisen die folgenden Funktionen auf:

* Abstrakte Methodendeklarationen sind nur in abstrakten Klassen zulässig.
* Es gibt keinen Methodenkörper, da eine abstrakte Methodendeklaration keine Implementierungen bietet; die Methodendeklaration enden ganz einfach mit einem Semikolon; auf die Signatur folgen keine geschweiften Klammern ({ }). Die Implementierung wird von der Methode [override](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/override) zur Verfügung gestellt, die ein Member einer nicht abstrakten Klasse ist.
* Es ist unzulässig, die Modifizierer [static](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/static) oder [virtual](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/virtual) in einer abstrakten Methodendeklaration zu verwenden.

virtual: Das Schlüsselwort virtual wird zum Ändern einer Methoden-, Eigenschaften-, Indexer- oder Ereignisdeklaration verwendet, und lässt zu, dass sie in einer abgeleiteten Klasse außer Kraft gesetzt werden. Diese Methode **kann** z.B. von jeder Klasse, die sie erbt, überschrieben werden. Die Implementierung eines virtuellen Members **kann** durch einen [overriding member](https://docs.microsoft.com/de-de/dotnet/csharp/language-reference/keywords/override) (überschreibender Member) in einer abgeleiteten Klasse geändert werden. Verwendung: Ändern oder Erweitern einer Methode in abgeleiteten Klassen.

Themen:

* Operatorüberladungen
* Verkettete Listen
* Interfaces
* Exceptions
* Generics
* Delegates