

Glossar

Die im Glossar mit einem Sternchen () gekennzeichneten Begriffe beziehen sich auf Konzepte, die im Rahmen dieses Buches nicht angesprochen wurden, der Vollständigkeit halber aber in das Glossar aufgenommen wurden.*

.NET Framework

Wird zur Ausführung von .NET-Anwendungen benötigt. Die wichtigsten Bestandteile des .NET Frameworks sind die Common Language Runtime und die Bibliotheken.

Argumente

Werte, die beim Aufruf einer Methode an deren Parameter übergeben werden.

Base Class Libraries

Eine der Bibliotheken aus dem .NET Framework. Stellt elementare Funktionalität zur Verfügung, wie sie von den meisten Anwendungen benötigt wird.

Ableitung

Von Ableitung oder Vererbung spricht man, wenn eine neue Klasse auf der Grundlage einer bereits bestehenden Klasse definiert wird. Die neu definierte Klasse bezeichnet man dabei als abgeleitete Klasse, die bereits bestehende Klasse nennt man Basisklasse. Die abgeleitete Klasse erbt die Elemente der Basisklasse.

Abstrakte Klasse*

Als abstrakte Klasse bezeichnet man eine Klasse, die eine oder mehrere abstrakte Methoden (Methoden ohne Definitionsteil) enthält und üblicherweise vor allem als Schnittstellenvorgabe für Klassenhierarchien dient. Von abstrakten Klassen können keine Objekte gebildet werden. Abstrakte Methoden und abstrakte Klassen werden durch das Schlüsselwort `abstract` gekennzeichnet.

Algorithmus

Schrittweiser Ablaufplan, der zur Lösung einer gestellten Aufgabe führt.

Anweisung

Höherer Befehl. Wird vom Compiler in eine Folge von Maschinenbefehlen übersetzt. Anweisungen werden grundsätzlich mit Semikolon abgeschlossen.

Argumente

Werte, die man beim Aufruf einer Methode an deren Parameter übergibt.



Array

Datenstruktur, in der man mehrere Variablen eines Datentyps vereinen kann.

Attribute*

Informationen, die bei der Kompilierung zusammen mit den Metadaten des Programms gespeichert und zur Laufzeit abgefragt werden können.

Ausdruck

Kombination aus Werten, Variablen und Operatoren.

Bibliothek

Sammlung nützlicher Klassen, die man in einem Programm verwenden kann.

Block

Eine oder mehrere Anweisungen, die durch geschweifte Klammern zusammengefasst sind.

Boxing

Als Boxing bezeichnet man in C# die Möglichkeit, einen Werttyp in ein Objekt zu verwandeln. Man kann sich das so vorstellen, dass das Objekt wie eine Box über den Wert gestülpt wird – daher der Begriff Boxing. Den umgekehrten Vorgang, die Rückverwandlung eines »eingeboxten« Werts, bezeichnet man als Unboxing.

Casting

Englischer Begriff für die explizite Typumwandlung.

Common Language Runtime

Laufzeitumgebung des .NET Frameworks. Die Common Language Runtime lädt die Klassen der auszuführenden Anwendung, lässt den Code der Anwendung in Maschinencode übersetzen und überwacht den Speicher.

Compiler

Programm, das den Quelltext eines Programms in binären Code übersetzt.

Datenelement

In einer Klasse definierte Konstanten und Variablen. Letztere werden auch als Felder bezeichnet und in Instanz- und Klassenvariablen unterschieden.

Datentyp

Gibt an, welche Art von Daten in einer Variablen gespeichert werden kann. Hilft dem Compiler außerdem, die korrekte Verwendung von Werten und Objekten sicherzustellen.

Debugger

Programm, das ein anderes Programm schrittweise ausführen kann.

Dekrement

Verringerung des Werts einer Variablen um eine Einheit.

Destruktor*

Spezielle Methode, die bei Auflösung der Objekte der Klasse aufgerufen wird (Pendant zum Konstruktor, muss aber nur selten definiert werden.)

Eigenschaft

Kombination aus get- und/oder set-Block. Gestattet den methodenvermittelten Zugriff auf Felder unter Beibehaltung der Syntax des normalen Feldzugriffs.

Ereignis

Mechanismus, der es einer Klasse ermöglicht, Code auszuführen, der vom Benutzer der Klasse aufgesetzt wurde (also nicht Teil der Klasse ist).

Feld

Variable, die in einer Klasse definiert ist.

Formulare

Von Visual Studio geprägter Oberbegriff für Fenster und Dialogfelder.

Funktionen

C# kennt keine Funktionen. In anderen Programmiersprachen sind Funktionen »Methoden«, die keine Klassenelemente sind.

GUI

Abkürzung für Graphical User Interface (»Grafische Benutzeroberfläche«).

Hauptfenster

Fenster, das beim Start der Anwendung erscheint, den größten Teil der Funktionalität beherbergt und beim Schließen die Anwendung beendet.

IL-Code

Vom C#-Compiler erzeugter, virtueller Zwischencode, der vor der Ausführung erst noch von einem Interpreter in Maschinencode übersetzt werden muss.

Information Hiding

Zielsetzung des Klassen-Designs, die besagt, dass Klassen die Details ihrer Implementierung vor dem Programmierer, der die Klassen verwendet, verbergen sollen. Die Klasse soll wie eine Blackbox erscheinen. Je weniger der Programmierer über die Klasse wissen muss, umso einfacher und sicherer kann er sie einsetzen.



Initialisierung

Werden einer Variablen direkt bei der Definition Werte zugewiesen, spricht man von Initialisierung.

Inkrement

Erhöhung des Werts einer numerischen Variablen um 1.

Instanz

Objekt eines Klassentyps.

Instanzbildung

Erzeugung eines Objekts eines Klassentyps. Ist stets mit dem Aufruf eines Konstruktors der Klasse verbunden.

Instanzvariablen

Nicht-statische Felder einer Klasse, die als Kopie an die erzeugten Objekte (Instanzen) weitergegeben werden.

Integrierte Entwicklungsumgebung (IDE)

Bei der Programmerstellung ist der Programmierer auf eine Reihe von Hilfsprogrammen angewiesen (Editor, Compiler, Linker, Debugger). Eine integrierte Entwicklungsumgebung (wie Visual Studio) ist ein Programm, das eine gemeinsame Benutzeroberfläche zur Verfügung stellt, von der aus man diese Programme aufrufen und bedienen kann.

Interpreter

Programm, das nicht-maschinenspezifischen Code schrittweise in maschinenspezifischen Code übersetzt und ausführt.

Iteration

Schleifendurchgang.

JIT-Compiler, Jitter

Just-in-Time-Compiler zur Ausführung von maschinenunabhängigem Pseudo-Code.

Kapselung

Zielsetzung des Klassen-Designs, die besagt, dass Klassen die Details ihrer Implementierung vor dem Programmierer, der die Klassen verwendet, verbergen sollen. Die Klasse soll wie eine Blackbox erscheinen. Je weniger der Programmierer über die Klasse wissen muss, umso einfacher und sicherer kann er sie einsetzen.

Klassen

Klassen sind Beschreibungen für Objekte mit gemeinsamen Merkmalen (Felder) und Verhaltensweisen (Methoden). Gelegentlich werden Klassen auch als reine Methodensammlungen definiert. Von diesen Klassen können dann meist keine Objekte erzeugt werden.

Klassenvariablen

Statische Felder einer Klasse, die nur als Elemente der Klasse existieren und nicht an die erzeugten Objekte (Instanzen) weitergegeben werden.

Komponenten

Software-Bausteine, hinter denen Klassendefinitionen mit öffentlichen Eigenschaften und Ereignisse stehen.

Konkatenation

Aneinanderhängen von Strings.

Konsolenanwendungen

Konsolenanwendungen sind Programme ohne grafische Oberfläche, die ihre Ein- und Ausgabe über die Betriebssystemkonsole (unter Windows die Eingabeaufforderung) abwickeln.

Konstruktor

Spezielle Methode, die bei Einrichtung (Instanzbildung) der Objekte der Klasse aufgerufen wird.

Literal

Konstante, die als Wert direkt in den Quelltext geschrieben wird.

Lokale*

Beschreibung der lokalen, landesspezifischen Umgebung, unter der ein Programm ausgeführt wird.

Maschinencode

Binärer Code, der von dem Prozessor des Rechners verstanden wird. Leider ist der Maschinencode stark prozessorspezifisch.

Methoden

Mit Namen versehene Anweisungsblöcke, die als Teil einer Klasse definiert werden. Durch Aufruf des Methodennamens kann man den Anweisungsblock der Methoden ausführen lassen. Ob eine Methode für Objekte der Klasse oder direkt über den Namen der Klasse aufgerufen wird, hängt davon ab, ob sie als `static` definiert wurde oder nicht.

Namespace

In C# kann man Klassendefinitionen in Namespaces organisieren. Auf diese Weise lassen sich Namenskonflikte durch Verwendung gleicher Klassennamen vermeiden (interessant bei der Erstellung größerer Software-Projekte, an denen gleichzeitig mehrere Programmierer arbeiten oder bei denen mehrere Bibliotheken verwendet werden).



Objekte

Dem Begriff des Objekts kommen in der Programmierung je nach Kontext verschiedene Bedeutungen zu:

In objektorientierten Modellen bezeichnet man als Objekte real existierende oder abstrakte Dinge, mit denen das Programm später arbeiten soll und für die man Klassen definieren wird.

In einem objektorientierten Programm ist ein Objekt eine explizite Manifestierung einer Klasse. Um auf das Objekt zugreifen und mit ihm arbeiten zu können, weist man es einer Variablen vom Typ der Klasse zu.

Objektorientierung

In der objektorientierten Programmierung werden Probleme gelöst, indem man Klassen implementiert und dann mit den Objekten arbeitet, die aus diesen Klassen erzeugt werden. Ein großer Teil des Programmieraufwands fließt dabei in die Implementierung einer entsprechenden Klasse zur Beschreibung der Objekte. Die solide Implementierung der Klasse zahlt sich beim Schreiben des weiteren Programmcodes aus: Der Programmierer kann mit den Objekten der Klasse (und eventuell definierten statischen Elementen) arbeiten und sich auf die korrekte Implementierung der Klasse verlassen! Der resultierende Code ist dank der Klassen leichter zu verstehen, zu warten und wieder-zuverwerten. Darüber hinaus steht die objektorientierte Sichtweise der menschlichen Art und Weise, Dinge zu sehen und zu klassifizieren, näher als der Umgang mit elementaren Datentypen.

Objektvariable

Variable vom Typ einer Klasse.

Operatorenüberladung*

Bezeichnet die Verbindung mehrerer Implementierungen mit einem Operator. Der Compiler kann anhand der Zahl und der Datentypen der Operanden feststellen, welche Implementierung auszuführen ist.

Overloading

Englischer Begriff für Überladung. Wird gelegentlich fälschlicherweise im Kontext von Klassenhierarchien und Polymorphie verwendet. Es ist zwar möglich, geerbte Methoden in der abgeleiteten Klasse zu überladen (zusätzliche Definition mit anderen Parametern), wenn es jedoch um die Implementierung polymorphen Verhaltens geht, muss man die geerbte Methode überschreiben (Overriding), d.h. für die gleichen Parameter neu definieren.

Overriding

Englischer Begriff für das Überschreiben.

Parameter

Variablen einer Methode, die in den runden Klammern hinter dem Methodennamen definiert werden und die beim Aufruf der Methode mit den Werten (Argumenten) initialisiert werden, die der Aufrufer der Methode übergibt.

Polymorphie

Erlaubt es, durch Überschreibung geerbter Methoden auf unterschiedlichen Objekten gleichnamige Operationen auszuführen.

Designer

Abkürzung für Rapid Application Development (»Schnelle Anwendungsentwicklung«), bezieht sich im Falle von Visual Studio auf die komponentengestützte Programmierung und die grafische Erstellung von Benutzeroberflächen.

Schleife

Programmkonstrukt zur mehrfachen Ausführung von Anweisungsblöcken.

Schnittstelle

Datentyp, der eine Sammlung von Eigenschaften und Methoden ohne Anweisungsblöcke definiert. Klassen, die sich von einer Schnittstelle ableiten, verpflichten sich, die Eigenschaften und Methoden der Schnittstelle zu implementieren und somit die in der Schnittstelle vorgegebenen Elemente zum Teil ihrer Schnittstelle zur Außenwelt zu machen.

Der Begriff »Schnittstelle« wird in der Programmierung aber auch häufig im allgemeinen Sinne gebraucht. So bilden die Parameter und der Rückgabewert die Schnittstelle einer Methode und die `public`-Elemente einer Klasse bilden die öffentliche Schnittstelle einer Klasse.

Standardkonstruktor

Konstruktor, der ohne Argumente aufgerufen werden kann. Der vom Compiler bei Bedarf erzeugte Ersatzkonstruktor ist ein Standardkonstruktor.

Stream*

Bezeichnung für einen Datenstrom zwischen einer Ein- oder Ausgabeeinheit und dem Programm.

String

Zeichenkette.

Strukturen*

Strukturen werden im Prinzip wie Klassen definiert, allerdings mit dem Schlüsselwort `struct` (statt `class`). Strukturen kennen keine Vererbung (obwohl auch sie letztendlich auf den Basistyp `Object` zurückgehen). Obwohl Objekte von selbst definierten Strukturen wie Klassenobjekte mit `new` erstellt werden, handelt es sich bei den Strukturen nicht um Referenztypen, sondern um Werttypen.



Überladung

Mit Überladung bezeichnet man die Zuweisung mehrerer Operationen an einen Operator oder die Definition mehrerer Methoden eines Namens. Im Falle der Operatorenüberladung kann der Compiler anhand der Zahl und der Typen der Operanden feststellen, welche Operation auszuführen ist. Im Falle der Methodenüberladung zieht der Compiler die Zahl und Typen der Parameter zur eindeutigen Identifizierung des aufzurufenden Methodenblocks heran.

Überschreibung

Von Überschreibung spricht man, wenn man in einer abgeleiteten Klasse für eine geerbte Methode eine neue, individuelle Implementierung (Anweisungsblock) definiert. Der Sinn ist, dass die abgeleitete Klasse auf einen entsprechenden Methodenaufwurf in spezifischer Weise reagiert. Zur korrekten Implementierung ist es erforderlich, Methoden, die überschrieben werden sollen, in der Basisklasse als `virtual` und in der abgeleiteten Klasse als `override` zu deklarieren.

Variable

Variablen sind Zwischenspeicher, in denen man Werte ablegen kann. Jede Variable verfügt über einen Namen, den man bei der Definition der Variablen angibt und über den man den aktuellen Wert der Variablen abfragen oder ihr einen neuen Wert zuweisen kann.

Vererbung

Klassen lassen sich in Klassenhierarchien zusammenfassen. Abgeleitete Klassen können die Eigenschaften der übergeordneten Klasse übernehmen (erben).

Whitespace

Zeichen, die Leerräume erzeugen: Leerzeichen, Tabulatoren, Zeilenumbruch.

Zugriffsmodifizierer

Die Zugriffsmodifizierer `public`, `protected`, `internal` und `private` kommen bei der Klassendefinition zum Einsatz und regeln die Sichtbarkeit und Verfügbarkeit der Klasse und ihrer Elemente.