

**Update
inside**

Jürgen KOTZ
Christian WENZ

C# und .NET 6

GRUNDLAGEN, PROFIWISSEN UND REZEPTE



Bonuskapitel und Beispiele unter
plus.hanser-fachbuch.de

HANSER

HANSER

Jürgen Kotz
Christian Wenz

C# und .NET 6

Grundlagen, Profiwissen und Rezepte

**Ihr Plus – digitale
Zusatzinhalte!**

Auf unserem Download-Portal
finden Sie zu diesem Titel
kostenloses Zusatzmaterial.

Geben Sie auf [plus.hanser-
fachbuch.de](https://plus.hanser-fachbuch.de) einfach diesen
Code ein:

plus-43qr9-pKvN2

Die Autoren:

Jürgen Kotz, München

Christian Wenz, München

Alle in diesem Buch enthaltenen Informationen, Verfahren und Darstellungen wurden nach bestem Wissen zusammengestellt und mit Sorgfalt getestet. Dennoch sind Fehler nicht ganz auszuschließen. Aus diesem Grund sind die im vorliegenden Buch enthaltenen Informationen mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autor und Verlag übernehmen infolgedessen keine juristische Verantwortung und werden keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Art aus der Benutzung dieser Informationen – oder Teilen davon – entsteht.

Ebenso übernehmen Autor und Verlag keine Gewähr dafür, dass beschriebene Verfahren usw. frei von Schutzrechten Dritter sind. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Buch berechtigt deshalb auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Dieses Werk ist urheberrechtlich geschützt.

Alle Rechte, auch die der Übersetzung, des Nachdruckes und der Vervielfältigung des Buches, oder Teilen daraus, vorbehalten. Kein Teil des Werkes darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren) – auch nicht für Zwecke der Unterrichtsgestaltung – reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

© 2022 Carl Hanser Verlag München, www.hanser-fachbuch.de

Lektorat: Sylvia Hasselbach

Copy editing: Walter Saumweber, Ratingen

Umschlagdesign: Marc Müller-Bremer, München, www.rebranding.de

Umschlagrealisation: Max Kostopoulos

Titelmotiv: © thinkstockphotos.de/liuzishan

Gesamtherstellung: Kösel, Krugzell

Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702

Print-ISBN: 978-3-446-46930-3

E-Book-ISBN: 978-3-446-47349-2

E-Pub-ISBN: 978-3-446-47423-9

Update inside.

Mit unserem kostenlosen Update-Service zum Buch erhalten Sie aktuelle Infos zu C# und dem .NET Framework.

Und so funktioniert es:

1. Registrieren Sie sich unter:

www.hanser-fachbuch.de/Csharp-update

2. Geben Sie diesen Code ein:

5UKz-TaNroZp-mR5n

Der Update-Service läuft bis März 2024.

Als registrierter Nutzer werden Sie in diesem Zeitraum persönlich per E-Mail informiert, sobald ein neues Buch-Update zum Download verfügbar ist.

Wenn Sie Fragen haben, wenden Sie sich gerne an:

Csharp-update@hanser.de

Inhalt

Titelei

Impressum

Inhalt

Vorwort

Zusatzmaterial online

Teil I: Grundlagen

1 .NET 6

1.1 Microsofts .NET-Technologie

1.1.1 Zur Geschichte von .NET

1.1.2 .NET-Features und Begriffe

1.2 .NET Core

1.2.1 Geschichte von .NET Core

1.2.2 LTS – Long Term Support und zukünftige Versionen

1.2.3 .NET Standard

1.3 Features von .NET 6

2 Einstieg in Visual Studio 2022

2.1 Die Installation von Visual Studio 2022

2.1.1 Überblick über die Produktpalette

2.1.2 Anforderungen an Hard- und Software

2.2 Unser allererstes C#-Programm

2.2.1 Vorbereitungen

2.2.2 Quellcode schreiben

2.2.3 Programm kompilieren und testen

2.2.4 Einige Erläuterungen zum Quellcode

2.2.5 Konsolenanwendungen sind out

2.3 Die Windows-Philosophie

[2.3.1 Mensch-Rechner-Dialog](#)

[2.3.2 Objekt- und ereignisorientierte Programmierung](#)

[2.3.3 Programmieren mit Visual Studio 2022](#)

[**2.4 Die Entwicklungsumgebung Visual Studio 2022**](#)

[2.4.1 Neues Projekt](#)

[2.4.2 Die wichtigsten Fenster](#)

[2.4.3 Projektvorlagen in Visual Studio 2022 – Minimal APIs](#)

[**2.5 Praxisbeispiele**](#)

[2.5.1 Unsere erste Windows-Forms-Anwendung](#)

[2.5.2 Umrechnung Euro-Dollar](#)

[**3 Grundlagen der Sprache C#**](#)

[**3.1 Grundbegriffe**](#)

[3.1.1 Anweisungen](#)

[3.1.2 Bezeichner](#)

[3.1.3 Schlüsselwörter](#)

[3.1.4 Kommentare](#)

3.2 Datentypen, Variablen und Konstanten

3.2.1 Fundamentale Typen

3.2.2 Wertetypen versus Verweistypen

3.2.3 Benennung von Variablen

3.2.4 Deklaration von Variablen

3.2.5 Typsuffixe

3.2.6 Zeichen und Zeichenketten

3.2.7 object-Datentyp

3.2.8 Konstanten deklarieren

3.2.9 Nullable Types

3.2.10 Typinferenz

3.2.11 Gültigkeitsbereiche und Sichtbarkeit

3.3 Konvertieren von Datentypen

3.3.1 Implizite und explizite Konvertierung

3.3.2 Welcher Datentyp passt zu welchem?

3.3.3 Konvertieren von string

3.3.4 Die Convert-Klasse

3.3.5 Die Parse-Methode

3.3.6 Boxing und Unboxing

3.4 Operatoren

3.4.1 Arithmetische Operatoren

3.4.2 Zuweisungsoperatoren

3.4.3 Logische Operatoren

3.4.4 Rangfolge der Operatoren

3.5 Kontrollstrukturen

3.5.1 Verzweigungsbefehle

3.5.2 Schleifenanweisungen

3.6 Benutzerdefinierte Datentypen

3.6.1 Enumerationen

3.6.2 Strukturen

3.7 Nutzerdefinierte Methoden

3.7.1 Methoden mit Rückgabewert

3.7.2 Methoden ohne Rückgabewert

[3.7.3 Parameterübergabe mit ref](#)

[3.7.4 Parameterübergabe mit out](#)

[3.7.5 Methodenüberladung](#)

[3.7.6 Optionale Parameter](#)

[3.7.7 Benannte Parameter](#)

[3.8 Praxisbeispiele](#)

[3.8.1 Vom PAP zur Konsolenanwendung](#)

[3.8.2 Ein Konsolen- in ein Windows-Programm verwandeln](#)

[3.8.3 Schleifenanweisungen verstehen](#)

[3.8.4 Benutzerdefinierte Methoden überladen](#)

[3.8.5 Anwendungen von Visual Basic nach C# portieren](#)

[4 OOP-Konzepte](#)

[4.1 Kleine Einführung in die OOP](#)

[4.1.1 Historische Entwicklung](#)

[4.1.2 Grundbegriffe der OOP](#)

[4.1.3 Sichtbarkeit von Klassen und ihren Mitgliedern](#)

[4.1.4 Allgemeiner Aufbau einer Klasse](#)

[4.1.5 Das Erzeugen eines Objekts](#)

[4.1.6 Einführungsbeispiel](#)

[**4.2 Eigenschaften**](#)

[4.2.1 Eigenschaften mit Zugriffsmethoden kapseln](#)

[4.2.2 Berechnete Eigenschaften](#)

[4.2.3 Lese-/Schreibschutz](#)

[4.2.4 Property-Accessoren](#)

[4.2.5 Statische Felder/Eigenschaften](#)

[4.2.6 Einfache Eigenschaften automatisch implementieren](#)

[**4.3 Methoden**](#)

[4.3.1 Öffentliche und private Methoden](#)

[4.3.2 Überladene Methoden](#)

[4.3.3 Statische Methoden](#)

[**4.4 Ereignisse**](#)

[4.4.1 Ereignis hinzufügen](#)

4.4.2 Ereignis verwenden

4.5 Arbeiten mit Konstruktor und Destruktor

4.5.1 Konstruktor und Objektinitialisierer

4.5.2 Destruktor und Garbage Collector

4.5.3 Mit using den Lebenszyklus des Objekts kapseln

4.6 Vererbung und Polymorphie

4.6.1 Method-Overriding

4.6.2 Klassen implementieren

4.6.3 Implementieren der Objekte

4.6.4 Ausblenden von Mitgliedern durch Vererbung

4.6.5 Allgemeine Hinweise und Regeln zur Vererbung

4.6.6 Polymorphes Verhalten

4.6.7 Die Rolle von System.Object

4.7 Spezielle Klassen

4.7.1 Abstrakte Klassen

4.7.2 Versiegelte Klassen

4.7.3 Partielle Klassen

4.7.4 Statische Klassen

4.8 Schnittstellen (Interfaces)

4.8.1 Definition einer Schnittstelle

4.8.2 Implementieren einer Schnittstelle

4.8.3 Abfragen, ob Schnittstelle vorhanden ist

4.8.4 Mehrere Schnittstellen implementieren

4.8.5 Schnittstellenprogrammierung ist ein weites Feld

4.9 Datensatztypen – Records

4.9.1 Definition eines Record

4.9.2 Mutable Properties

4.9.3 Nicht-destruktive Änderung

4.9.4 Dekonstruktion

4.10 Praxisbeispiele

4.10.1 Eigenschaften sinnvoll kapseln

4.10.2 Eine statische Klasse anwenden

[4.10.3 Vom fetten zum schlanken Client](#)

[4.10.4 Schnittstellenvererbung verstehen](#)

[4.10.5 Rechner für komplexe Zahlen](#)

[4.10.6 Sortieren mit IComparable/IComparer](#)

[4.10.7 Einen Objektbaum in generischen Listen abspeichern](#)

[4.10.8 OOP beim Kartenspiel erlernen](#)

[4.10.9 Eine Klasse zur Matrizenrechnung entwickeln](#)

[4.10.10 Vererbung von Records](#)

[5 Arrays, Strings, Funktionen](#)

[5.1 Datenfelder \(Arrays\)](#)

[5.1.1 Array deklarieren](#)

[5.1.2 Array instanzieren](#)

[5.1.3 Array initialisieren](#)

[5.1.4 Zugriff auf Array-Elemente](#)

[5.1.5 Zugriff mittels Schleife](#)

[5.1.6 Mehrdimensionale Arrays](#)

[5.1.7 Zuweisen von Arrays](#)

[5.1.8 Arrays aus Strukturvariablen](#)

[5.1.9 Löschen und Umdimensionieren von Arrays](#)

[5.1.10 Eigenschaften und Methoden von Arrays](#)

[5.1.11 Übergabe von Arrays](#)

[5.2 Verarbeiten von Zeichenketten](#)

[5.2.1 Zuweisen von Strings](#)

[5.2.2 Eigenschaften und Methoden von String-Variablen](#)

[5.2.3 Wichtige Methoden der String-Klasse](#)

[5.2.4 Die StringBuilder-Klasse](#)

[5.3 Datums- und Zeitberechnungen](#)

[5.3.1 Die DateTime-Struktur](#)

[5.3.2 Wichtige Eigenschaften von DateTime-Variablen](#)

[5.3.3 Wichtige Methoden von DateTime-Variablen](#)

[5.3.4 Wichtige Mitglieder der DateTime-Struktur](#)

[5.3.5 Konvertieren von Datumstrings in DateTime-Werte](#)

[5.3.6 Die TimeSpan-Struktur](#)

[5.3.7 DateOnly und TimeOnly](#)

5.4 Mathematische Funktionen

5.4.1 Überblick

5.4.2 Zahlen runden

5.4.3 Winkel umrechnen

5.4.4 Potenz- und Wurzeloperationen

5.4.5 Logarithmus und Exponentialfunktionen

5.4.6 Zufallszahlen erzeugen

5.4.7 Kreisberechnung

5.5 Zahlen- und Datumsformatierungen

5.5.1 Anwenden der ToString-Methode

5.5.2 Anwenden der Format-Methode

5.5.3 Stringinterpolation

5.6 Praxisbeispiele

5.6.1 Zeichenketten verarbeiten

5.6.2 Zeichenketten mit StringBuilder addieren

5.6.3 Methodenaufrufe mit Array-Parametern

6 Weitere Sprachfeatures

6.1 Namespaces (Namensräume)

6.1.1 Ein kleiner Überblick

6.1.2 Einen eigenen Namespace einrichten

6.1.3 Die using-Anweisung

6.1.4 Namespace Alias

6.1.5 Globale using-Anweisungen

6.2 Operatorenüberladung

6.2.1 Syntaxregeln

6.2.2 Praktische Anwendung

6.3 Collections (Auflistungen)

6.3.1 Die Schnittstelle IEnumerable

6.3.2 ArrayList

6.3.3 Hashtable

6.3.4 Indexer

6.4 Generics

6.4.1 Generics bieten Typsicherheit

6.4.2 Generische Methoden

6.4.3 yield – Iteratoren

6.5 Generische Collections

6.5.1 List-Collection statt ArrayList

6.5.2 Vorteile generischer Collections

6.5.3 Constraints

6.6 Das Prinzip der Delegates

6.6.1 Delegates sind Methodenzeiger

6.6.2 Einen Delegate-Typ deklarieren

6.6.3 Ein Delegate-Objekt erzeugen

6.6.4 Anonyme Methoden

6.6.5 Lambda-Ausdrücke

6.6.6 Lambda-Ausdrücke in der Task Parallel Library

6.6.7 Action<> und Func<>

6.7 Dynamische Programmierung

6.7.1 Wozu dynamische Programmierung?

6.7.2 Das Prinzip der dynamischen Programmierung

6.7.3 Optionale Parameter sind hilfreich

6.7.4 Kovarianz und Kontravarianz

6.8 Weitere Datentypen

6.8.1 BigInteger

6.8.2 Complex

6.8.3 Tuple<>

6.8.4 SortedSet<>

6.9 Praxisbeispiele

6.9.1 ArrayList versus generische List

6.9.2 Generische IEnumerable-Interfaces implementieren

6.9.3 Delegates, Func, anonyme Methoden, Lambda Expressions

7 Einführung in LINQ

7.1 LINQ-Grundlagen

[7.1.1 Die LINQ-Architektur](#)

[7.1.2 Anonyme Typen](#)

[7.1.3 Erweiterungsmethoden](#)

[**7.2 Abfragen mit LINQ to Objects**](#)

[7.2.1 Grundlegendes zur LINQ-Syntax](#)

[7.2.2 Zwei alternative Schreibweisen von LINQ-Abfragen](#)

[7.2.3 Übersicht der wichtigsten Abfrageoperatoren](#)

[**7.3 LINQ-Abfragen im Detail**](#)

[7.3.1 Die Projektionsoperatoren Select und SelectMany](#)

[7.3.2 Der Restriktionsoperator Where](#)

[7.3.3 Die Sortierungsoperatoren OrderBy und ThenBy](#)

[7.3.4 Der Gruppierungsoperator GroupBy](#)

[7.3.5 Verknüpfen mit Join](#)

[7.3.6 Aggregat-Operatoren](#)

[7.3.7 Verzögertes Ausführen von LINQ-Abfragen](#)

[7.3.8 Konvertierungsmethoden](#)

[7.3.9 Abfragen mit PLINQ](#)

7.4 Praxisbeispiele

7.4.1 Die Syntax von LINQ-Abfragen verstehen

7.4.2 Aggregat-Abfragen mit LINQ

7.4.3 LINQ im Schnelldurchgang erlernen

7.4.4 Strings mit LINQ abfragen und filtern

7.4.5 Duplikate aus einer Liste entfernen

7.4.6 Arrays mit LINQ initialisieren

7.4.7 Arrays per LINQ mit Zufallszahlen füllen

7.4.8 Einen String mit Wiederholmuster erzeugen

7.4.9 Mit LINQ Zahlen und Strings sortieren

7.4.10 Mit LINQ Collections von Objekten sortieren

7.4.11 Where – Deep Dive

8 Neuerungen von C# im Überblick

8.1 C# 4.0 – Visual Studio 2010

8.1.1 Datentyp dynamic

8.1.2 Benannte und optionale Parameter

8.1.3 Kovarianz und Kontravarianz

8.2 C# 5.0 – Visual Studio 2012

8.2.1 Async und Await

8.2.2 CallerInfo

8.3 Visual Studio 2013

8.4 C# 6.0 – Visual Studio 2015

8.4.1 String Interpolation

8.4.2 Schreibgeschützte AutoProperties

8.4.3 Initialisierer für AutoProperties

8.4.4 Expression Body Member

8.4.5 using static

8.4.6 Bedingter Nulloperator

8.4.7 Ausnahmenfilter

8.4.8 nameof-Ausdrücke

8.4.9 await in catch und finally

8.4.10 Indexinitialisierer

8.5 C# 7.0 – Visual Studio 2017

[8.5.1 out-Variablen](#)

[8.5.2 Tupel](#)

[8.5.3 Mustervergleich](#)

[8.5.4 Discards](#)

[8.5.5 Lokale ref-Variablen und Rückgabetypen](#)

[8.5.6 Lokale Funktionen](#)

[8.5.7 Mehr Expression Body Member](#)

[8.5.8 throw-Ausdrücke](#)

[8.5.9 Verbesserung der numerischen literalen Syntax](#)

[**8.6 C# 7.1 bis 7.3 – Visual Studio 2019**](#)

[8.6.1 C# 7.1](#)

[8.6.2 C# 7.2](#)

[8.6.3 C# 7.3](#)

[8.6.4 Visual Studio 2019 – Live Share](#)

[**8.7 C# 8.0**](#)

[8.7.1 Standardschnittstellenmethoden](#)

[8.7.2 Vereinfachung von switch-Ausdrücken](#)

[8.7.3 Eigenschaftenmuster](#)

[8.7.4 Vereinfachte using-Ressourcenschutzblöcke](#)

[8.7.5 Null-Coalescing-Zuweisungen](#)

[8.7.6 Nullable Referenztypen](#)

[8.7.7 Indizes und Bereiche](#)

[8.7.8 Weitere Sprachneuerungen](#)

[**8.8 C# 9.0**](#)

[8.8.1 Records](#)

[8.8.2 Init-only Setter](#)

[8.8.3 Weitere Verbesserungen in Musterausdrücken](#)

[8.8.4 Weitere Sprachneuerungen](#)

[**8.9 C# 10**](#)

[8.9.1 Datensatzstrukturen](#)

[8.9.2 Globale using-Anweisungen](#)

[8.9.3 Weitere Sprachneuerungen](#)

[**Teil II: Desktop-Anwendungen**](#)

9 Einführung in WPF

9.1 Einführung

9.1.1 Was kann eine WPF-Anwendung?

9.1.2 Die eXtensible Application Markup Language

9.1.3 Unsere erste XAML-Anwendung

9.1.4 Zielplattformen

9.1.5 Applikationstypen

9.1.6 Vor- und Nachteile von WPF-Anwendungen

9.1.7 Weitere Dateien im Überblick

9.2 Alles beginnt mit dem Layout

9.2.1 Allgemeines zum Layout

9.2.2 Positionieren von Steuerelementen

9.2.3 Canvas

9.2.4 StackPanel

9.2.5 DockPanel

9.2.6 WrapPanel

9.2.7 UniformGrid

[9.2.8 Grid](#)

[9.2.9 ViewBox](#)

[9.2.10 TextBlock](#)

[**9.3 Das WPF-Programm**](#)

[9.3.1 Die App-Klasse](#)

[9.3.2 Das Startobjekt festlegen](#)

[9.3.3 Kommandozeilenparameter verarbeiten](#)

[9.3.4 Die Anwendung beenden](#)

[9.3.5 Auswerten von Anwendungsereignissen](#)

[**9.4 Die Window-Klasse**](#)

[9.4.1 Position und Größe festlegen](#)

[9.4.2 Rahmen und Beschriftung](#)

[9.4.3 Das Fenster-Icon ändern](#)

[9.4.4 Anzeige weiterer Fenster](#)

[9.4.5 Transparenz](#)

[9.4.6 Abstand zum Inhalt festlegen](#)

[9.4.7 Fenster ohne Fokus anzeigen](#)

[9.4.8 Ereignisfolge bei Fenstern](#)

[9.4.9 Ein paar Worte zur Schriftdarstellung](#)

[9.4.10 Ein paar Worte zur Darstellung von Controls](#)

[9.4.11 Wird mein Fenster komplett mit WPF gerendert?](#)

[10 Übersicht WPF-Controls](#)

[10.1 Allgemeingültige Eigenschaften](#)

[10.2 Label](#)

[10.3 Button, RepeatButton, ToggleButton](#)

[10.3.1 Schaltflächen für modale Dialoge](#)

[10.3.2 Schaltflächen mit Grafik](#)

[10.4 TextBox, PasswordBox](#)

[10.4.1 TextBox](#)

[10.4.2 PasswordBox](#)

[10.5 CheckBox](#)

[10.6 RadioButton](#)

10.7 ListBox, ComboBox

10.7.1 ListBox

10.7.2 ComboBox

10.7.3 Den Content formatieren

10.8 Image

10.8.1 Grafik per XAML zuweisen

10.8.2 Grafik zur Laufzeit zuweisen

10.8.3 Bild aus Datei laden

10.8.4 Die Grafikskalierung beeinflussen

10.9 Slider, ScrollBar

10.9.1 Slider

10.9.2 ScrollBar

10.10 ScrollViewer

10.11 Menu, ContextMenu

10.11.1 Menu

10.11.2 Tastenkürzel

10.11.3 Grafiken

10.11.4 Weitere Möglichkeiten

10.11.5 ContextMenu

10.12 ToolBar

10.13 StatusBar, ProgressBar

10.13.1 StatusBar

10.13.2 ProgressBar

10.14 Border, GroupBox, BulletDecorator

10.14.1 Border

10.14.2 GroupBox

10.14.3 BulletDecorator

10.15 Expander, TabControl

10.15.1 Expander

10.15.2 TabControl

10.16 Popup

10.17 TreeView