

O'REILLY®



原书第7版
涵盖.NET标准2



C# 7.0

核心技术指南

C# 7.0 in a Nutshell: The Definitive Reference



机械工业出版社
China Machine Press

[美] Joseph Albahari Ben Albahari 著
刘夏 译

原书第7版

C# 7.0 核心技术指南

C# 7.0 in a Nutshell

[美] 约瑟夫·阿坝哈瑞 (Joseph Albahari)

本·阿坝哈瑞 (Ben Albahari) 著

刘夏 译



Beijing • Boston • Farnham • Sebastopol • Tokyo

O'REILLY®

O'Reilly Media, Inc. 授权机械工业出版社出版

机械工业出版社

图书在版编目 (CIP) 数据

C# 7.0 核心技术指南 (原书第 7 版) / (美) 约瑟夫·阿坝哈瑞 (Joseph Albahari), (美) 本·阿坝哈瑞 (Ben Albahari) 著; 刘夏译. —北京: 机械工业出版社, 2019.7 (O'Reilly 精品图书系列)

书名原文: C# 7.0 in a Nutshell: The Definitive Reference
ISBN 978-7-111-63134-7

I. C… II. ①约… ②本… ③刘… III. 语言—程序设计—指南 IV. TP312.8-62
中国版本图书馆 CIP 数据核字 (2019) 第 131514 号

北京市版权局著作权合同登记

图字: 01-2017-8755 号

Copyright © 2018 Joseph Albahari and Ben Albahari. All rights reserved.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and China Machine Press, 2019.
Authorized translation of the English edition, 2018 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由 O'Reilly Media, Inc. 出版 2018。

简体中文版由机械工业出版社出版 2019。英文原版的翻译得到 O'Reilly Media, Inc. 的授权。此简体中文版的出版和销售得到出版权和销售权的所有者——O'Reilly Media, Inc. 的许可。

版权所有, 未得书面许可, 本书的任何部分和全部不得以任何形式重制。

封底无防伪标均为盗版

本书法律顾问

北京大成律师事务所 韩光 / 邹晓东

书 名 / C# 7.0 核心技术指南 (原书第 7 版)

书 号 / ISBN 978-7-111-63134-7

责任编辑 / 陈佳媛

封面设计 / Karen Montgomery, 张健

出版发行 / 机械工业出版社

地 址 / 北京市西城区百万庄大街 22 号 (邮政编码 100037)

印 刷 / 北京诚信伟业印刷有限公司

开 本 / 178 毫米 × 233 毫米 16 开本 67 印张

版 次 / 2019 年 8 月第 1 版 2019 年 8 月第 1 次印刷

定 价 / 259.00 元 (册)

客服热线: (010)88361066 88379833 68326294

华章网站: www.hzbook.com

投稿热线: (010)88379604

读者信箱: hzit@hzbook.com

O'Reilly Media, Inc. 介绍

O'Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自 1978 年开始，O'Reilly 一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly 为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了 Make 杂志，从而成为 DIY 革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项 O'Reilly 的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar 博客有口皆碑。”

——Wired

“O'Reilly 凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference 是聚集关键思想领袖的绝对典范。”

——CRN

“一本 O'Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim 是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照 Yogi Berra 的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去 Tim 似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

译者序

C# 是一门出色的语言，丰富而又平衡，兼顾面向对象和函数式编程的特点。若您接触过 C++ 或者 Java，那么可以快速上手 C#。从诞生至今，C# 没有因功能的扩展而改变初心。但是它从不缺乏激情，在 2.0 版本中引入了对泛型的支持；在 3.0 版本中加入了 Lambda 表达式、LINQ、匿名类型和扩展方法；在 4.0 版本中加入了动态类型……2016 年 .NET Core 开源之后，它的进化更是突飞猛进。C# 官方语言特性库 (<https://github.com/dotnet/csharplang>) 中的提交数已超过 500 个，而讨论组中处于积极讨论状态的主题更是超过了 1000 个。我非常愿意将这样一门实用的、平衡的、富有激情和创造力的语言推荐给各位读者。而本书正是全方位介绍 C# 语言的书籍。

本书兼顾了各种类型的读者。对于初学者，本书不论在介绍基本的语法和类型，还是高级的 .NET Framework 类库和特性时，都采用了讲解和示例结合的方法，从原理和实践方面双管齐下，因此阅读门槛并不高。尤其值得一提的是这些范例都可以直接在 LINQPad 中运行。充分利用这些范例可以帮助读者快速掌握 C# 语言的各种实践。而对于经验丰富的读者，本书的每一章都详尽而系统，是优秀的案头参考书。对于框架的设计者，第 14 章到第 24 章包含了框架设计过程中必备的各种知识。除此之外，得益于作者开发 LINQPad 的经验，本书最后一章对 Roslyn 编译服务的讲解系统而完整，是非常难得的参考资料。

最后，感谢本书的作者 Joseph Albahari。他对我翻译本书给予了热情的帮助。我们就书中的一些技术细节进行了讨论，并对十几处内容进行了修正。作者深厚的技术功底和认真负责的态度令人敬佩。感谢华章公司的编辑们，她们给予了我很多专业的宝贵建议。她们高标准的把关保证了本书的质量。感谢我的家人，她们的支持给予了我不断前行的动力。最后，希望这本书能给大家带来愉快的阅读体验。请进入 C# 的精彩世界吧。

刘夏

2019 年于北京

目录

前言	1
第 1 章 C# 和 .NET Framework 简介	7
1.1 面向对象	7
1.2 类型安全性	8
1.3 内存管理	9
1.4 平台支持	9
1.5 C# 和 CLR	9
1.6 CLR 和 .NET Framework	10
1.7 其他框架	11
1.7.1 遗留框架和小众框架	12
1.7.2 Windows Runtime	12
1.8 C# 简史	13
1.8.1 C# 7.0 新特性	14
1.8.2 C# 6.0 新特性	17
1.8.3 C# 5.0 新特性	18
1.8.4 C# 4.0 新特性	19
1.8.5 C# 3.0 新特性	19
1.8.6 C# 2.0 新特性	20
第 2 章 C# 语言基础	21
2.1 第一个 C# 程序	21
2.2 语法	24
2.2.1 标识符和关键字	24

2.2.2 字面量、标点与运算符	26
2.2.3 注释	26
2.3 类型基础	27
2.3.1 预定义类型示例	27
2.3.2 自定义类型示例	28
2.3.3 转换	30
2.3.4 值类型与引用类型	31
2.3.5 预定义类型分类	34
2.4 数值类型	35
2.4.1 数值字面量	36
2.4.2 数值转换	38
2.4.3 算术运算符	39
2.4.4 自增和自减运算符	39
2.4.5 特殊整数类型运算	39
2.4.6 8 位和 16 位整数类型	41
2.4.7 特殊的 float 和 double 值	41
2.4.8 double 和 decimal 的对比	42
2.4.9 实数的舍入误差	43
2.5 布尔类型和运算符	43
2.5.1 布尔类型转换	43
2.5.2 相等和比较运算符	44
2.5.3 条件运算符	44
2.6 字符串和字符	45
2.6.1 char 转换	46
2.6.2 字符串类型	46
2.7 数组	48
2.7.1 默认数组元素初始化	49
2.7.2 多维数组	49
2.7.3 简化数组初始化表达式	51
2.7.4 边界检查	52
2.8 变量和参数	52
2.8.1 栈和堆	52
2.8.2 明确赋值	54
2.8.3 默认值	54
2.8.4 参数	55

2.8.5 引用局部变量 (C# 7).....	61
2.8.6 引用返回值 (C# 7).....	61
2.8.7 var 隐式类型局部变量	61
2.9 表达式和运算符	62
2.9.1 基础表达式	63
2.9.2 空表达式	63
2.9.3 赋值表达式	63
2.9.4 运算符优先级和结合性	64
2.9.5 运算符表	64
2.10 null 运算符	68
2.10.1 null 合并运算符	68
2.10.2 null 条件运算符 (C# 6)	68
2.11 语句	69
2.11.1 声明语句	69
2.11.2 表达式语句	70
2.11.3 选择语句	71
2.11.4 迭代语句	75
2.11.5 跳转语句	77
2.11.6 其他语句	78
2.12 命名空间	79
2.12.1 using 指令	80
2.12.2 using static 指令 (C# 6)	80
2.12.3 命名空间中的规则	81
2.12.4 类型和命名空间别名	83
2.12.5 高级命名空间特性	83
第 3 章 在 C# 中创建类型	86
3.1 类	86
3.1.1 字段	86
3.1.2 方法	87
3.1.3 实例构造器	89
3.1.4 对象初始化器	92
3.1.5 this 引用	94
3.1.6 属性	94
3.1.7 索引器	97

3.1.8 常量	98
3.1.9 静态构造器	99
3.1.10 静态类	100
3.1.11 终结器	101
3.1.12 分部类型和方法	101
3.1.13 nameof 运算符 (C#6)	102
3.2 继承	103
3.2.1 多态	104
3.2.2 类型转换和引用转换	104
3.2.3 虚函数成员	107
3.2.4 抽象类和抽象成员	108
3.2.5 隐藏继承成员	108
3.2.6 密封函数和类	110
3.2.7 base 关键字	110
3.2.8 构造器和继承	110
3.2.9 重载和解析	112
3.3 object 类型	112
3.3.1 装箱和拆箱	113
3.3.2 静态和运行时类型检查	114
3.3.3 GetType 方法和 typeof 运算符	115
3.3.4 ToString 方法	115
3.3.5 object 的成员列表	116
3.4 结构体	116
3.5 访问权限修饰符	118
3.5.1 示例	118
3.5.2 友元程序集	119
3.5.3 可访问性封顶	119
3.5.4 访问权限修饰符的限制	119
3.6 接口	120
3.6.1 扩展接口	121
3.6.2 显式接口实现	121
3.6.3 虚方法实现接口成员	122
3.6.4 在子类中重新实现接口	122
3.6.5 接口和装箱	124
3.7 枚举类型	125

3.7.1 枚举类型转换	126
3.7.2 标志枚举类型	126
3.7.3 枚举运算符	127
3.7.4 类型安全问题	127
3.8 嵌套类型	128
3.9 泛型	130
3.9.1 泛型类型	130
3.9.2 为什么需要泛型	131
3.9.3 泛型方法	132
3.9.4 声明类型参数	133
3.9.5 typeof 和未绑定泛型类型	133
3.9.6 泛型的默认值	134
3.9.7 泛型的约束	134
3.9.8 继承泛型类型	136
3.9.9 自引用泛型声明	136
3.9.10 静态数据	136
3.9.11 类型参数的转换	137
3.9.12 协变	138
3.9.13 逆变	141
3.9.14 C# 泛型和 C++ 模板对比	142

第 4 章 C# 的高级特性..... 143

4.1 委托	143
4.1.1 用委托书写插件方法	144
4.1.2 多播委托	145
4.1.3 实例目标方法和静态目标方法	147
4.1.4 泛型委托类型	147
4.1.5 Func 和 Action 委托	148
4.1.6 委托和接口	149
4.1.7 委托的兼容性	150
4.2 事件	152
4.2.1 标准事件模式	154
4.2.2 事件访问器	158
4.2.3 事件的修饰符	159
4.3 Lambda 表达式	159

4.3.1 显式指定 Lambda 参数的类型	160
4.3.2 捕获外部变量	161
4.3.3 Lambda 表达式和局部方法的对比	163
4.4 匿名方法	164
4.5 try 语句和异常	165
4.5.1 catch 子句	167
4.5.2 finally 块	168
4.5.3 抛出异常	170
4.5.4 System.Exception 的关键属性	172
4.5.5 常用的异常类型	172
4.5.6 TryXXX 方法模式	173
4.5.7 异常的替代方式	173
4.6 可枚举类型和迭代器	174
4.6.1 可枚举类型	174
4.6.2 集合的初始化器	175
4.6.3 迭代器	176
4.6.4 迭代器语义	177
4.6.5 组合序列	178
4.7 可空类型	180
4.7.1 Nullable<T> 结构体	180
4.7.2 隐式和显式的可空对象转换	180
4.7.3 装箱拆箱可空值	181
4.7.4 运算符优先级提升	181
4.7.5 在 bool? 上使用 & 和 运算符	183
4.7.6 可空类型和 null 运算符	183
4.7.7 可空类型的应用场景	184
4.7.8 可空类型的替代方案	184
4.8 扩展方法	185
4.8.1 扩展方法链	186
4.8.2 二义性与解析	186
4.9 匿名类型	188
4.10 元组 (C# 7)	189
4.10.1 元组元素命名	190
4.10.2 ValueTuple.Create	192
4.10.3 元组的解构	192

4.10.4 元组的比较.....	193
4.10.5 System.Tuple 类.....	193
4.11 特性.....	193
4.11.1 特性类.....	194
4.11.2 命名和位置特性参数.....	194
4.11.3 特性的目标.....	195
4.11.4 指定多个特性.....	195
4.12 调用者信息特性.....	195
4.13 动态绑定.....	197
4.13.1 静态绑定与动态绑定.....	197
4.13.2 自定义绑定.....	198
4.13.3 语言绑定.....	199
4.13.4 RuntimeBinderException.....	200
4.13.5 动态类型的运行时表示.....	200
4.13.6 动态转换.....	201
4.13.7 var 与 dynamic.....	201
4.13.8 动态表达式.....	202
4.13.9 无动态接收者的动态调用.....	202
4.13.10 动态表达式中的静态类型.....	203
4.13.11 不可调用的函数.....	204
4.14 运算符重载.....	205
4.14.1 运算符函数.....	206
4.14.2 重载等号和比较运算符.....	206
4.14.3 自定义隐式和显式转换.....	207
4.14.4 重载 true 和 false.....	208
4.15 不安全的代码和指针.....	209
4.15.1 指针基础.....	209
4.15.2 不安全的代码.....	209
4.15.3 fixed 语句.....	209
4.15.4 指针取成员运算符.....	210
4.15.5 数组.....	210
4.15.6 void*.....	211
4.15.7 指向非托管代码的指针.....	212
4.16 预处理指令.....	212
4.16.1 Conditional 特性.....	213

4.16.2 Pragma 警告	214
4.17 XML 文档	214
4.17.1 标准的 XML 文档标签	215
4.17.2 用户定义标签	217
4.17.3 类型或成员交叉引用	217
第 5 章 框架概述	219
5.1 .NET 标准 2.0	221
5.1.1 旧版本 .NET 标准	222
5.1.2 引用程序集	222
5.2 CLR 和核心框架	223
5.2.1 系统类型	223
5.2.2 文本处理	223
5.2.3 集合	223
5.2.4 查询	224
5.2.5 XML	224
5.2.6 诊断	224
5.2.7 并发与异步	225
5.2.8 流与 I/O	225
5.2.9 网络	225
5.2.10 序列化	225
5.2.11 程序集、反射和特性	226
5.2.12 动态编程	226
5.2.13 安全性	226
5.2.14 高级线程功能	226
5.2.15 并行编程	227
5.2.16 应用程序域	227
5.2.17 原生互操作性与 COM 互操作性	227
5.3 应用技术	227
5.3.1 用户界面 API	227
5.3.2 后台技术	230
5.3.3 分布式系统技术	232
第 6 章 框架基础	234
6.1 字符串与文本处理	234

6.1.1 字符	234
6.1.2 字符串	236
6.1.3 字符串的比较	240
6.1.4 StringBuilder	243
6.1.5 文本编码和 Unicode	244
6.2 日期和时间	248
6.2.1 TimeSpan	248
6.2.2 DateTime 和 DateTimeOffset	249
6.3 日期和时区	255
6.3.1 DateTime 与时区	256
6.3.2 DateTimeOffset 与时区	256
6.3.3 TimeZone 和 TimeZoneInfo	257
6.3.4 夏令时与 DateTime	261
6.4 格式化和解析	262
6.4.1 ToString 和 Parse	263
6.4.2 格式提供器	263
6.5 标准格式字符串与解析标记	268
6.5.1 数字格式字符串	268
6.5.2 NumberStyles	270
6.5.3 Date/Time 格式字符串	272
6.5.4 DateTimeStyles	275
6.5.5 枚举的格式字符串	275
6.6 其他转换机制	275
6.6.1 Convert 类	276
6.6.2 XmlConvert	278
6.6.3 类型转换器	278
6.6.4 BitConverter	279
6.7 全球化	279
6.7.1 全球化检查清单	280
6.7.2 测试	280
6.8 操作数字	281
6.8.1 转换	281
6.8.2 Math	281
6.8.3 BigInteger	282
6.8.4 Complex	283

6.8.5 Random	284
6.9 枚举	285
6.9.1 枚举值转换	285
6.9.2 列举枚举值	287
6.9.3 枚举的工作方式	288
6.10 Guid 结构体	288
6.11 相等比较	289
6.11.1 值相等和引用相等	289
6.11.2 标准等值比较协议	290
6.11.3 相等比较和自定义类型	295
6.12 顺序比较	300
6.12.1 IComparable	300
6.12.2 < 和 >	301
6.12.3 实现 IComparable 接口	302
6.13 实用类	303
6.13.1 Console 类	303
6.13.2 Environment 类	304
6.13.3 Process 类	304
6.13.4 AppContext 类	306
第 7 章 集合	307
7.1 枚举	307
7.1.1 IEnumerable 和 IEnumerator	308
7.1.2 IEnumerable<T> 和 IEnumerator<T>	309
7.1.3 实现枚举接口	311
7.2 ICollection 和 IList 接口	315
7.2.1 ICollection<T> 和 ICollection	316
7.2.2 IList<T> 和 IList	317
7.2.3 IReadOnlyList<T>	318
7.3 Array 类	318
7.3.1 创建和索引	321
7.3.2 枚举	322
7.3.3 长度和维数	323
7.3.4 搜索	323
7.3.5 排序	325

7.3.6 反转数组元素	326
7.3.7 复制数组	326
7.3.8 转换和调整大小	326
7.4 List、Queue、Stack 和 Set	327
7.4.1 List<T> 和 ArrayList	327
7.4.2 LinkedList<T>	330
7.4.3 Queue<T> 和 Queue	332
7.4.4 Stack<T> 和 Stack	332
7.4.5 BitArray	333
7.4.6 HashSet<T> 和 SortedSet<T>	334
7.5 字典	336
7.5.1 IDictionary<TKey, TValue>	336
7.5.2 IDictionary	338
7.5.3 Dictionary<TKey, TValue> 和 Hashtable	339
7.5.4 OrderedDictionary	340
7.5.5 ListDictionary 和 HybridDictionary	340
7.5.6 排序字典	341
7.6 自定义集合与代理	342
7.6.1 Collection<T> 和 CollectionBase	343
7.6.2 KeyedCollection<TKey, TValue> 和 DictionaryBase	345
7.6.3 ReadOnlyCollection<T>	348
7.7 扩展相等比较和排序操作	349
7.7.1 IEqualityComparer 和 EqualityComparer	350
7.7.2 IComparer 和 Comparer	352
7.7.3 StringComparer	353
7.7.4 IStructuralEquatable 和 IStructuralComparable	355

第 8 章 LINQ 查询 356

8.1 入门	356
8.2 流式语法	358
8.2.1 查询运算符链	359
8.2.2 使用 Lambda 表达式	361
8.2.3 原始顺序	364
8.2.4 其他运算符	364
8.3 查询表达式	365

8.3.1 范围变量	367
8.3.2 LINQ 查询语法与 SQL 语法	368
8.3.3 查询语法和流式语法	368
8.3.4 混合查询语法	369
8.4 延迟执行	369
8.4.1 重复执行	370
8.4.2 捕获变量	371
8.4.3 延迟执行的工作原理	372
8.4.4 串联装饰器	373
8.4.5 查询语句的执行方式	375
8.5 子查询	376
8.6 构造方式	379
8.6.1 渐进式查询构造	379
8.6.2 into 关键字	380
8.6.3 查询的包装	381
8.7 映射方式	382
8.7.1 对象初始化器	382
8.7.2 匿名类型	383
8.7.3 let 关键字	384
8.8 解释型查询	385
8.8.1 解释型查询的工作机制	387
8.8.2 综合使用解释型查询和本地查询	389
8.8.3 AsEnumerable 方法	390
8.9 LINQ to SQL 和 Entity Framework	391
8.9.1 LINQ to SQL 的实体类	392
8.9.2 Entity Framework 的实体类	393
8.9.3 DataContext 和ObjectContext	395
8.9.4 关联	399
8.9.5 L2S 和 EF 的延迟执行	400
8.9.6 DataLoadOptions	401
8.9.7 Entity Framework 中的立即加载功能	403
8.9.8 更新	403
8.9.9 L2S 和 EF 的 API 对比	405
8.10 构建查询表达式	406
8.10.1 委托与表达式树	406
8.10.2 表达式树	408