

INDEX

Symboles

!= operator, 74
- operator, 51
-- operator, 57
π, 61
! operator, 76, 177
#define, 387
#elif, 386
#else, 386
#endif, 386
#endregion, 385
#error, 385
#if, 386
#region, 385
#undef, 387
#warning, 385
& operator, 369, 382
&& operator, 76
&= operator, 383
* operator, 51, 368
.. operator, 93
/ operator, 51
?. operator, 176
?? operator, 177
?[] operator, 176
@ symbol, 66
[] operator, 90
^ operator, 93, 382
^= operator, 383
| operator, 382
|| operator, 76
|= operator, 383
~ operator, 382
~= operator, 383
+ operator, 51
< operator, 74
<< operator, 381
<<= operator, 383

<= operator, 74
== operator, 70
=> operator, 81, 304
> operator, 74
-> operator, 369
>= operator, 74
>> operator, 381
>>= operator, 383
.cs file, 15
.csproj file, 15, 409
.dll, 456
.NET, 9, 10, 404, 452
.NET Core, 405, 452
.NET Framework, 404, 452
.NET MAUI, 407
.NET Multi-platform App User Interface, 407
.sln file, 409

0

0-based indexing, 91, 452

A

absolute value, 61
abstract class, 208, 452
abstract keyword, 209
abstract method, 208, 452
abstraction, 159, 452
accessibility level, 156, 452
accessibility modifier, 156
acquiring a lock, 349
Action (System), 294
add keyword, 301
addition, 51
Address Of operator, 369
ahead-of-time compilation, 403, 452
algorithm, 51
alias, 266
alignment, 66

allocating memory, 110
and keyword, 321
and pattern, 321
 Android, 10, 407
 anonymous type, 169, 453
 AOT compilation. *See* ahead-of-time compilation
 app model, 407, 453
 architecture, 402, 453
 argument, 102, 453
 arm, 453
 array, 90, 453
as keyword, 204
ascending keyword, 336
 ASP.NET, 407, 453
 assembler, 401, 453
 assembly, 401, 453
 assembly language, 453
 assignment, 453, 460, 464
 associative array, 453
async keyword, 355
 asynchronous programming, 351, 453
 attribute, 376, 453
 defining, 378
 auto property, 166
 auto-implemented property, 166
 automatic memory management, 122, 453
 auto-property, 453
await keyword, 355
 awaitable, 358, 453

B

backing field, 165, 453
 backing store, 165
 base class, 198, 453
 Base Class Library, 10, 22, 247, 403, 406, 453
base keyword, 202
 BCL. *See* Base Class Library
 binary, 400, 453, 454
 binary literal, 454
 binary operator, 51
BinaryReader (System.IO), 314
BinaryWriter (System.IO), 314
 binding, 20
 bit, 38, 400, 454
 bit field, 380, 454
 bit manipulation, 380, 454
 bitshift operator, 381
 bitwise operator, 454
 Blazor, 407
 block, 454
 block body, 105, 454
 block scope, 72
 block statement, 70
 body. *See* method body
bool, 45
 Boolean, 454
Boolean (System), 224
 boxing, 225, 454
 boxing conversion, 225
break keyword, 87
 breakpoint, 449, 454
 conditions and actions, 451
 build configuration, 27, 409
 built-in type, 38, 454
 built-in type alias, 224
by keyword, 339
 byte, 38, 40, 400, 454

Byte (System), 224

C

C, 10
 C#, **9**
 C++, 10, 454
 callback, 352, 454
 camelCase, 37
 captured variable, 306
 case guard, 319
case keyword, 81
 casting, 59, 454
 catch block, 454
 catching exceptions, 281
char, 42
Char (System), 224
 character, 454
 checked context, 391, 454
checked keyword, 391
 child class, 198
 CIL, 402, 405
 clamp, 62
 class, 21, 130, 144, 145, 454
 compared to structs, 220
 creating instances, 147
 default field values, 149
 defining, 145
 defining constructors, 148
 sealing, 204
class keyword, 145
 class library, 406
 clause (query expressions), 334
 ClickOnce, 411
 closure, 306, 455
 CLR. *See* Common Language Runtime
 Code Editor window, 18
 code library, 394
 code map, 20
 Code Window, 436, 455
 collaborator, 181
 collection initializer syntax, 93, 455
 command-line arguments, 387, 455
 comment, 29, 455
 Common Intermediate Language, 402, 405, 455
 Common Language Runtime, 402, 405, 455
 compiler, 18, 401, 455
 compiler error, 27, 442, 455
 suggestions for fixing, 443
 compiler warning, 442, 455
 compile-time constant, 272
 compiling, 18, 399
 composite type, 138, 455
 composition, 138
 compound assignment operator, 57, 455
 concrete class, 209, 455
 concurrency, 343, 455
 concurrency issue, 347
 condition, 70
 conditional compilation symbol, 386, 455
 conditional operator, 77
 const, 375
const keyword, 375
 constant, 375, 455
 constant pattern, 317
 constructor, 147, 148, 455
 default parameterless constructor, 455
 parameterless, 150
 context switch, 344, 455

continuation clause, 339
continue keyword, 87
 contravariant, 390
Convert (System), 47
 cosine, 62
 covariance, 390
 CRC card, 455
 CRC cards, 181
 critical section, 348, 456
 curly braces, 456
 custom conversion, 329, 456

D

dangling pointer, 456
 dangling reference, 122
 data structure, 456
DateTime (System), 249
 deadlock, 349, 456
 deallocating memory, 110
 debug, 456
 debugger, 447, 456
 debugging, 27, 447
decimal, 43
Decimal (System), 224
 declaration, 98, 456
 declaration pattern, 318
 deconstruction, 141, 228, 456
 deconstructor, 276
 decrement, 456
 decrement operator, 57
default keyword, 81, 239
 default operator, 239
 deferred execution, 340, 456
 delegate, 291, 456
 delegate chain, 295
delegate keyword, 292
 dependency, 456
 dependency (project), 394
 derived class, 198, 456
 deriving from classes, 198
descending keyword, 336
 deserialization, 456
 design, 144, 178, 456
 desktop development, 407
 dictionary, 456
Dictionary<TKey, TValue> (System.Collections.Generic), 256
 digit separator, 41, 456
 directed graph, 117
Directory (System.IO), 312
 discard, 142, 456
 discard pattern, 317
 Discord, 5
 divide and conquer, 456
 division, 51
 division by zero, 55, 456
DLLImport (System.Runtime.InteropServices), 372
do/while loop, 86
 dot operator, 20
dotnet command-line interface, 13, 412
double, 43
Double (System), 224
 downcasting, 203
dynamic keyword, 362
 dynamic object, 361, 457
 dynamic objects, 361
 dynamic type checking, 361, 457
DynamicObject (System.Dynamic), 364

E

E notation, 457
 early exit, 104, 457
 Edit and Continue, 451
else if statement, 73
else statement, 73
 encapsulation, 146, 457
 entry point, 23, 268, 457
 enum. *See* enumeration
enum keyword, 133
 enumeration, 132, 457
 equality operator, 70
equals keyword, 338
Equals method, 199
 Error List, 440, 457
 escape sequence, 65
 evaluation, 457
 event, 296, 457
 custom accessors, 301
 leak, 300
 null, 299
 raising, 297
 subscribing, 298
event keyword, 297
 event leak, 457
EventHandler (System), 300
EventHandler<EventArgs> (System), 300
 exception, 280, 457
 guidelines for using, 285
 rethrow, 288
Exception (System), 281
 exception filter, 289
 exception handler, 281
 EXE, 457
ExpandoObject (System.Dynamic), 363
 explicit, 457
 explicit conversion, 58
explicit keyword, 329
 exponent, 61
 expression, 24, 457
 evaluating, 24
 expression body, 105, 457
 extending classes, 198
 extension method, 457
extern keyword, 371

F

F#, 10, 402, 405
 factory method, 172
false keyword, 45
 field, 145, 457
 default value, 149
 initialization, 150
File (System.IO), 308
 files, 308
FileStream (System.IO), 314
 finally block, 284
finally keyword, 284
 fire (event), 297
fixed statement, 369, 457
 fixed-size array, 370, 457
 fixed-size buffer, 370
 flags enumeration, 383
float, 43
 floating-point division, 54, 458
 floating-point type, 43, 458

for loop, 86
 frame. *See* stack frame
 framework-dependent deployment, 412, 458
from clause, 334
from keyword, 334
 fully qualified name, 264, 265, 458
Func (System), 294
 function, 98, 458

G

game development, 408
 garbage collection, 122, 406, 458
 garbage collector, 123
 generic method, 237
 generic type, 232, 235, 458
 inheritance, 236
 generic type argument, 235, 458
 generic type constraint, 458
 generic type constraints, 237
 generic type parameter, 235, 458
 multiple, 236
 generic variance, 389, 458
 generics, 232
 constraints, 237
 inheritance, 389
 motivation for, 232
get keyword, 164
GetHashCode method, 257
 get-only property, 167
 getter, 157, 164, 458
GetType method, 203
global keyword, 266
 global namespace, 264, 458
 global state, 171, 458
 global using directive, 266
goto keyword, 388
 graph, 117
group by clause, 339
 guard expression, 319
Guid (System), 251

H

hash code, 257, 458
 heap, 115, 458
 hexadecimal, 458
 hexadecimal literal, 42

I

IAsyncEnumerable<T> (**System.Collections.Generic**), 359, 375
 IDE. *See* integrated development environment
 identifier, 20
IDisposable (System), 384
IDynamicMetaObjectProvider (System.Dynamic), 363
IDynamicMetaObjectProvider interface, 362
IEnumerable<T> (**System.Collections.Generic**), 255, 334
 if statement, 69
 IL, 402
 immutability, 167, 459
 implicit, 459
 implicit conversion, 58
implicit keyword, 329
in keyword, 276

increment, 459
 increment operator, 57
 index, 91, 459
 index initializer syntax, 328
 indexer, 327, 459
 indexer operator, 91
 indirection operator, 369
 infinite loop, 85, 459
 infinity, 54
 information hiding, 155
 inheritance, 197, 459
 constructors, 201
 inheritance hierarchy, 201
 inheritance relationship, 198
init keyword, 168
 initialization, 459
 inner exception, 289
 input parameter, 276
 instance, 130, 145, 459
 instance variable. *See* field
 instruction set architecture, 401, 459
int type, 34
Int16 (System), 224
Int32 (System), 224
Int64 (System), 224
 integer, 34
 integer division, 54, 459
 integer type, 39
 integral type, 39, 459
 integrated development environment, 11, 459
 IntelliSense, 437, 459
 interface, 211, 459
 and base classes, 214
 default methods, 215
 defining, 212
 explicit implementation, 214
 implementing, 213
interface keyword, 212
internal keyword, 160
into clause, 339
into keyword, 339
 iOS, 10
is keyword, 204, 322
 ISA, 402
 iterator, 374, 460

J

jagged array, 96, 460
 Java, 10, 460
 JetBrains Rider, 12
 JIT compiler, 403
 jitter, 403
join clause, 338
join keyword, 338
 Just-in-Time compilation, 406
 Just-in-Time compiler, 403, 460

K

keyword, 26, 460

L

label, 388
 labeled statement, 388
 lambda expression, 303, 460

lambda statement, 305
 Language Integrated Query, 333, 460
 lazy evaluation, 460
let clause, 339
let keyword, 339
 library, 315, 394, 406
 LINQ, 333
 LINQ to SQL, 341
 Linux, 10
List<T> (**System.Collections.Generic**), 252
 listener, 297
 literal, 20
 literal value, 460
 local function, 98, 307, 460
 local variable, 101, 460
lock keyword, 348
 logical operator, 76, 460
long, 40
 loop, 84, 460
 lowerCamelCase, 37

M

macOS, 10
 main method, 23, 457, 460
Main method, 23, 268
 managed code, 460
 managed memory, 460
 math, 50
Math (**System**), 61
MathF (**System**), 62
 MAUI, 407
 maximum, 62
 member, 20, 460
 member access operator, 20
 memory address, 32, 461
 memory allocation, 461
 memory leak, 122, 461
 memory management, 109
 memory safety, 406, 461
 method, 21, 97, 98, 271, 461

- calling, 99
- calling methods, 21
- return type, 98
- returning data, 21
- scope, 100

 method body, 98
 method call, 21, 461
 method call syntax, 336, 461
 method group, 105, 461
 method implementation, 461
 method invocation, 21
 method overload, 104, 461
 method scope, 72
 method signature, 461
 Microsoft Developer Network, 431
 minimum, 62
 mobile development, 407
 Mono, 404, 461
 MonoGame, 408
 MSBuild, 409
 MSIL, 402
MulticastDelegate (**System**), 295
 multi-dimensional array, 95, 461
 multiplication, 51
 multi-threading, 343, 461
 mutex, 348
 mutual exclusion, 348, 461
 MVC, 407

N

name, 20
 name binding, 20
 name collision, 461
 name conflict, 266
 name hiding, 151, 152, 461
 named argument, 272, 461
nameof operator, 379
 namespace, 21, 264, 446, 461
namespace keyword, 267
 namespaces, 267
 NaN, 54, 462
 narrowing conversion, 58, 462
 native code, 367, 462
 native integer types, 371
 nested pattern, 320
 nested type, 379
 nesting, 77, 88, 462
new keyword, 209
 new method, 209
nint, 371
not keyword, 321
not pattern, 321
 noun extraction, 180, 462
 NuGet, 396
 NuGet Package Manager, 462
nuint, 371
 null, 92
 null check, 176
 null conditional operator, 176
null keyword, 174
 null reference, 174, 462
 nullable type, 462
Nullable<T> (**System**), 258
 null-coalescing operator, 177

O

object, 130, 144, 198, 462
Object (**System**), 198, 224
 object initializer syntax, 168
object keyword, 198
 object-initializer syntax, 462
 object-oriented design, 144, 153, 178, 462

- rules, 184

 object-oriented programming, 129, 462
 observer, 297
ObsoleteAttribute (**System**), 376
on keyword, 338
 operation, 50, 462
 operator, 50, 462

- binary, 454
- ternary, 466
- unary, 466

 operator associativity, 52, 462
operator keyword, 326
 operator overloading, 325, 462
 operator precedence, 52, 462
 optional arguments, 271
 optional parameter, 271, 462
or keyword, 321
or pattern, 321
 order of operations, 52, 462
orderby clause, 336
orderby keyword, 336
out keyword, 275, 390
 out-of-order execution, 462

output parameter, 275
 overflow, 60, 462
 overload. *See* method overload
 overload resolution, 105, 463
 overloading, 463

P

P/Invoke, 371
 package, 396, 463
 package manager, 396
 parameter, 101, 149, 463
 variable number of, 272
 parameterful property, 327
 parameterless constructor, 150
params keyword, 272
 parent class. *See* base class
 parentheses, 463
 parse, 463
Parse methods, 48
 parsing, 48
 partial class, 387, 463
partial keyword, 387
 partial method, 388
 PascalCase, 37
 passing, 102
 passing by reference, 273, 463
 passing by value, 273
Path (System.IO), 313
 pattern matching, 82, 316, 463
 pi, 61
 pinning, 369
 Platform Invocation Services, 371, 463
 pointer member access operator, 369
 pointer type, 368, 463
 polymorphism, 206, 463
 positional pattern, 321
 postfix notation, 57
 power (math), 61
 PowerShell, 10
Predicate (System), 294
 prefix notation, 57
 preprocessor directive, 385, 463
 primitive type. *See* built-in type
 print debugging, 448, 463
private keyword, 156
private protected accessibility level, 380
Program class, 23
 program order, 464
 programming language, 9, 401
 project, 464
 project configuration, 15
 project template, 15
 Properties Window, 440
 property, 163, 464
 property pattern, 319
protected accessibility modifier, 204
protected internal accessibility level, 380
protected keyword, 204
 pseudo-random number generation, 248
public keyword, 156
 publish profile, 410
 publishing, 409

Q

query expression, 333, 464
 query syntax, 464

Quick Action, 438

R

raise (event), 297
Random (System), 248
 range operator, 93
 range variable, 335
 Razor Pages, 407
readonly keyword, 167
 read-only property, 167
 record, 227, 464
 struct-based, 230
 rectangular array, 96, 464
 recursion, 107, 464, *See* recursion
ref keyword, 274
ref local variable, 276
ref return, 276
 refactor, 464
 refactoring, 189
 reference, 116, 464
 reference semantics, 121, 464
 reference type, 118, 464
 reflection, 378, 464
 relational operator, 74, 464
 remainder, 55
remove keyword, 301
 requirements, 179, 464
 responsibility, 181
 rethrowing exceptions, 288
 return, 21, 26, 103, 464
return keyword, 103
 return type, 464
 returning early, 104
 Rider. *See* JetBrains Rider
 roundoff error, 60, 464
 runtime, 10, 402, 464

S

sbyte, 40
SByte (System), 224
 scheduler, 344, 465
 scope, 72, 100, 465
 SDK. *See* Software Development Kit
 sealed class, 204, 465
sealed keyword, 204
 seed, 248
select clause, 334
select keyword, 334
 self-contained deployment, 412
 serialization, 310
set keyword, 164
 setter, 157, 164
short, 40
 SignalR, 407
 signed type, 40, 465
 sine, 62
Single (System), 224
sizeof operator, 370
 software design, 144, 178
 Software Development Kit, 10, 406
 solution, 465
 Solution Explorer, 18, 439, 465
 source code, 15, 465
Span<T> (System), 276
 square brackets, 465

square root, 61
 stack, 110, 465
 stack allocation, 370, 465
 stack frame, 111, 465
 stack trace, 288, 465
stackalloc keyword, 370
 standard library, 406, 465
 statement, 23, 465
 static, 170, 465
 static class, 173
 static constructor, 172
 static field, 170
static keyword, 170
 static method, 172
 static property, 171
 static type checking, 361, 465
 static using directive, 266
 stream, 313
Stream (System.IO), 314
StreamWriter (System.IO), 314
string, 42, 465
String (System), 224
 string formatting, 67
 string interpolation, 66
 string manipulation, 310
string type, 20
 string literal, 20
StringBuilder (System.Text), 259
 struct, 218, 465
 compared to classes, 220
 constructors, 219
 memory, 219
struct keyword, 218
 subclass, 198
 subtraction, 51
 superclass, 198
 switch, 79, 466
 switch arm, 79
 switch expression, 81
 guard, 319
switch keyword, 80
 switch statement, 80
 symbol, 386
 synchronization context, 357
 synchronization issue, 347
 synchronous programming, 351, 466
 syntax, 19
System namespace, 21

T

tangent, 62
 task, 353, 466
Task (System.Threading.Tasks), 353
Task<T> (System.Threading.Tasks), 353
 ternary operator, 51, 77
this keyword, 152
 thread, 343, 466
Thread (System.Threading), 344
 thread pool, 356, 466
 thread safety, 347, 348, 466
Thread.Sleep, 346
 threading, 343
 threading issue, 347
ThreadPool (System.Threading), 356
throw keyword, 283
 throwing exceptions, 281
TimeSpan (System), 250
 top-level statement, 268, 466

ToString method, 199
 trigonometric functions, 62
true keyword, 45
try keyword, 281
 tuple, 137
 deconstruction, 141
 element names, 139
 equality, 142
 in parameters and return types, 139
 type, 130, 466
Type (System), 378
 type inference, 46, 466
 type pattern, 318
 type safety, 406, 466
 typecasting, 466
typeof keyword, 203

U

uint, 40
UInt16 (System), 224
UInt32 (System), 224
UInt64 (System), 224
ulong, 40
 UML, 181
 unary operator, 51
 unboxing, 225, 466
 unboxing conversion, 225
 unchecked context, 466
unchecked keyword, 391
 underlying type, 136, 466
 Unicode, 42
 Unified Modeling Language, 181
 Unity game engine, 408
 Universal Windows Platform, 408, 466
 unmanaged code, 367, 466
 unmanaged type, 368
 unpacking, 141, 466
 unsafe code, 367, 466
 unsafe context, 368, 466
unsafe keyword, 368
 unsigned type, 40, 466
 unverifiable code, 368
 UpperCamelCase, 37
 user-defined conversion, 467
ushort, 40
using directive, 22, 265, 467
using statement, 384, 467
 UWP, 408

V

value keyword, 164
 value semantics, 121, 228, 467
 value type, 118, 467
ValueTask (System.Threading.Tasks), 359
ValueTask<T> (System.Threading.Tasks), 359
var, 46
var pattern, 322
 variable, 25, 32, 467
 assignment, 33
 declaration, 25, 33
 initialization, 33
 naming, 36
 variance, 390
 verbatim string literal, 66
 virtual machine, 402, 467
 virtual method, 467

Visual Basic, 10, 402, 405, 467
Visual Studio, 12, 18, 435, 467
 Community Edition, 11
 Enterprise Edition, 12
 installing, 13
 Professional Edition, 12
Visual Studio Code, 12, 467
Visual Studio for Mac, 12
volatile field, 392, 467
volatile keyword, 392

W

Web API, 407
web development, 407
where clause, 335
where keyword, 335
while keyword, 84
while loop, 84

whitespace, 23
widening conversion, 58
Windows, 10
Windows Forms, 407, 467
Windows Presentation Foundation, 407, 467
WinForms, 407
with keyword, 228
WPF, 407

X

Xamarin Forms, 407, 467
XML Documentation Comment, 106, 467

Y

yield keyword, 374

THE C# PLAYER'S GUIDE

IT'S DANGEROUS TO CODE ALONE!
TAKE THIS.



The book in your hands is a **different kind of programming book**. Like an entertaining video game, programming is an often challenging but always rewarding experience. This book shakes off the dusty, dull, dryness of the typical programming book, replacing it with something more exciting and flavorful: a bit of humor, a casual tone, and examples involving dragons and asteroids instead of bank accounts and employees.

And since you learn to program by doing instead of just reading, this book contains **over 100 hands-on programming challenges**. You will be building software instead of just reading about it. By completing the challenges, you'll earn experience points, level up, and become a True C# Programmer!

This book covers the C# language from the ground up. It doesn't assume you've been programming for years, but it also doesn't hold back on exciting, powerful language features.

- The journey begins by getting you set up to program in C#.
- We will then explore the **basic mechanics of C#**: statements, expressions, variables, if statements, loops, and methods.
- Next, we dive deep into a powerful and central feature of C#: **object-oriented programming**, which is an essential tool needed to build larger programs.
- We then look at the **advanced C# features** that make the language unique, elegant, and powerful.

With this book as your companion, you will soon be off to save the world (or take it over) with your own C# programs!