INTERNATIONAL STANDARD

ISO/IEC 23270

Third edition
2018-12

Information technology — Programming languages — C#

Technologies de l'information — Langages de programmation — C#

Reference number
ISO/IEC 23270:2018(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Table of Contents

Foreword .... **xix**

Introduction .... **xxi**

1. Scope **1**

2. Normative references **3**

3. Terms and definitions **5**

4. Acronyms and abbreviations **7**

5. General description **9**

6. Conformance **11**

7. Lexical structure **13**

- 7.1 Programs 13
- 7.2 Grammars 13
  - 7.2.1 General 13
  - 7.2.2 Grammar notation 13
  - 7.2.3 Lexical grammar 14
  - 7.2.4 Syntactic grammar 15
  - 7.2.5 Grammar ambiguities 15
- 7.3 Lexical analysis 16
  - 7.3.1 General 16
  - 7.3.2 Line terminators 16
  - 7.3.3 Comments 17
  - 7.3.4 White space 18
- 7.4 Tokens 19
  - 7.4.1 General 19
  - 7.4.2 Unicode character escape sequences 19
  - 7.4.3 Identifiers 20
  - 7.4.4 Keywords 21
  - 7.4.5 Literals 22
    - 7.4.5.1 General 22
    - 7.4.5.2 Boolean literals 23
    - 7.4.5.3 Integer literals 23
    - 7.4.5.4 Real literals 24
    - 7.4.5.5 Character literals 25
    - 7.4.5.6 String literals 26
    - 7.4.5.7 The null literal 28
  - 7.4.6 Operators and punctuators 28
- 7.5 Pre-processing directives 28
  - 7.5.1 General 28
  - 7.5.2 Conditional compilation symbols 30
  - 7.5.3 Pre-processing expressions 30
  - 7.5.4 Definition directives 31
  - 7.5.5 Conditional compilation directives 32

7.5.6 Diagnostic directives 34
7.5.7 Region directives 35
7.5.8 Line directives 35
7.5.9 Pragma directives 36

8. Basic concepts 37

8.1 Application startup 37
8.2 Application termination 38
8.3 Declarations 38
8.4 Members 41
8.4.1 General 41
8.4.2 Namespace members 41
8.4.3 Struct members 41
8.4.4 Enumeration members 42
8.4.5 Class members 42
8.4.6 Interface members 42
8.4.7 Array members 42
8.4.8 Delegate members 42
8.5 Member access 42
8.5.1 General 42
8.5.2 Declared accessibility 42
8.5.3 Accessibility domains 43
8.5.4 Protected access 46
8.5.5 Accessibility constraints 47
8.6 Signatures and overloading 48
8.7 Scopes 49
8.7.1 General 49
8.7.2 Name hiding 52
8.7.2.1 General 52
8.7.2.2 Hiding through nesting 52
8.7.2.3 Hiding through inheritance 53
8.8 Namespace and type names 54
8.8.1 General 54
8.8.2 Unqualified names 56
8.8.3 Fully qualified names 56
8.9 Automatic memory management 57
8.10 Execution order 59

9. Types 61

9.1 General 61
9.2 Reference types 61
9.2.1 General 61
9.2.2 Class types 62
9.2.3 The object type 62
9.2.4 The dynamic type 63
9.2.5 The string type 63
9.2.6 Interface types 63
9.2.7 Array types 63
9.2.8 Delegate types 63
9.3 Value types 63
9.3.1 General 63

9.3.2 The System.ValueType type ... 64
9.3.3 Default constructors ... 64
9.3.4 Struct types ... 65
9.3.5 Simple types ... 65
9.3.6 Integral types ... 66
9.3.7 Floating-point types ... 67
9.3.8 The decimal type ... 68
9.3.9 The bool type ... 69
9.3.10 Enumeration types ... 69
9.3.11 Nullable value types ... 69
9.3.12 Boxing and unboxing ... 70
9.4 Constructed types ... 70
9.4.1 General ... 70
9.4.2 Type arguments ... 71
9.4.3 Open and closed types ... 71
9.4.4 Bound and unbound types ... 72
9.4.5 Satisfying constraints ... 72
9.5 Type parameters ... 73
9.6 Expression tree types ... 73
9.7 The dynamic type ... 74

10. Variables ... 77

10.1 General ... 77
10.2 Variable categories ... 77
10.2.1 General ... 77
10.2.2 Static variables ... 77
10.2.3 Instance variables ... 77
10.2.3.1 General ... 77
10.2.3.2 Instance variables in classes ... 78
10.2.3.3 Instance variables in structs ... 78
10.2.4 Array elements ... 78
10.2.5 Value parameters ... 78
10.2.6 Reference parameters ... 78
10.2.7 Output parameters ... 79
10.2.8 Local variables ... 79
10.3 Default values ... 80
10.4 Definite assignment ... 80
10.4.1 General ... 80
10.4.2 Initially assigned variables ... 81
10.4.3 Initially unassigned variables ... 81
10.4.4 Precise rules for determining definite assignment ... 81
10.4.4.1 General ... 81
10.4.4.2 General rules for statements ... 82
10.4.4.3 Block statements, checked, and unchecked statements ... 82
10.4.4.4 Expression statements ... 82
10.4.4.5 Declaration statements ... 83
10.4.4.6 If statements ... 83
10.4.4.7 Switch statements ... 83
10.4.4.8 While statements ... 83
10.4.4.9 Do statements ... 83

10.4.4.10 For statements 84
10.4.4.11 Break, continue, and goto statements 84
10.4.4.12 Throw statements 84
10.4.4.13 Return statements 84
10.4.4.14 Try-catch statements 85
10.4.4.15 Try-finally statements 85
10.4.4.16 Try-catch-finally statements 85
10.4.4.17 Foreach statements 86
10.4.4.18 Using statements 86
10.4.4.19 Lock statements 87
10.4.4.20 Yield statements 87
10.4.4.21 General rules for constant expressions 87
10.4.4.22 General rules for simple expressions 88
10.4.4.23 General rules for expressions with embedded expressions 88
10.4.4.24 Invocation expressions and object creation expressions 88
10.4.4.25 Simple assignment expressions 89
10.4.4.26 && expressions 89
10.4.4.27 || expressions 90
10.4.4.28 ! expressions 91
10.4.4.29 ?? expressions 91
10.4.4.30 ?: expressions 91
10.4.4.31 Anonymous functions 92
10.5 Variable references 92
10.6 Atomicity of variable references 92

11. Conversions 93

11.1 General 93
11.2 Implicit conversions 93
11.2.1 General 93
11.2.2 Identity conversion 94
11.2.3 Implicit numeric conversions 94
11.2.4 Implicit enumeration conversions 94
11.2.5 Implicit nullable conversions 94
11.2.6 Null literal conversions 94
11.2.7 Implicit reference conversions 95
11.2.8 Boxing conversions 95
11.2.9 Implicit dynamic conversions 97
11.2.10 Implicit constant expression conversions 97
11.2.11 Implicit conversions involving type parameters 97
11.2.12 User-defined implicit conversions 98
11.2.13 Anonymous function conversions and method group conversions 98
11.3 Explicit conversions 98
11.3.1 General 98
11.3.2 Explicit numeric conversions 99
11.3.3 Explicit enumeration conversions 101
11.3.4 Explicit nullable conversions 101
11.3.5 Explicit reference conversions 101
11.3.6 Unboxing conversions 102
11.3.7 Explicit dynamic conversions 103
11.3.8 Explicit conversions involving type parameters 103

11.3.9 User-defined explicit conversions 105
11.4 Standard conversions 105
11.4.1 General 105
11.4.2 Standard implicit conversions 105
11.4.3 Standard explicit conversions 105
11.5 User-defined conversions 105
11.5.1 General 105
11.5.2 Permitted user-defined conversions 105
11.5.3 Evaluation of user-defined conversions 106
11.5.4 User-defined implicit conversions 107
11.5.5 User-defined explicit conversions 107
11.6 Conversions involving nullable types 109
11.6.1 Nullable Conversions 109
11.6.2 Lifted conversions 109
11.7 Anonymous function conversions 109
11.7.1 General 109
11.7.2 Evaluation of anonymous function conversions to delegate types 111
11.7.3 Evaluation of anonymous function conversions to expression tree types 112
11.8 Method group conversions 112

12. Expressions 115

12.1 General 115
12.2 Expression classifications 115
12.2.1 General 115
12.2.2 Values of expressions 116
12.3 Static and Dynamic Binding 116
12.3.1 General 116
12.3.2 Binding-time 117
12.3.3 Dynamic binding 117
12.3.4 Types of subexpressions 118
12.4 Operators 118
12.4.1 General 118
12.4.2 Operator precedence and associativity 118
12.4.3 Operator overloading 119
12.4.4 Unary operator overload resolution 121
12.4.5 Binary operator overload resolution 121
12.4.6 Candidate user-defined operators 121
12.4.7 Numeric promotions 122
12.4.7.1 General 122
12.4.7.2 Unary numeric promotions 122
12.4.7.3 Binary numeric promotions 122
12.4.8 Lifted operators 123
12.5 Member lookup 124
12.5.1 General 124
12.5.2 Base types 125
12.6 Function members 126
12.6.1 General 126
12.6.2 Argument lists 128
12.6.2.1 General 128
12.6.2.2 Corresponding parameters 129

12.6.2.3 Run-time evaluation of argument lists 130
12.6.3 Type inference 131
12.6.3.1 General 131
12.6.3.2 The first phase 132
12.6.3.3 The second phase 133
12.6.3.4 Input types 133
12.6.3.5 Output types 133
12.6.3.6 Dependence 133
12.6.3.7 Output type inferences 133
12.6.3.8 Explicit parameter type inferences 134
12.6.3.9 Exact inferences 134
12.6.3.10 Lower-bound inferences 134
12.6.3.11 Upper-bound inferences 134
12.6.3.12 Fixing 135
12.6.3.13 Inferred return type 135
12.6.3.14 Type inference for conversion of method groups 137
12.6.3.15 Finding the best common type of a set of expressions 137
12.6.4 Overload resolution 137
12.6.4.1 General 137
12.6.4.2 Applicable function member 138
12.6.4.3 Better function member 139
12.6.4.4 Better conversion from expression 140
12.6.4.5 Better conversion from type 140
12.6.4.6 Better conversion target 140
12.6.4.7 Overloading in generic classes 140
12.6.5 Compile-time checking of dynamic member invocation 141
12.6.6 Function member invocation 142
12.6.6.1 General 142
12.6.6.2 Invocations on boxed instances 143
12.7 Primary expressions 143
12.7.1 General 143
12.7.2 Literals 144
12.7.3 Simple names 144
12.7.3.1 General 144
12.7.3.2 Invariant meaning in blocks 146
12.7.4 Parenthesized expressions 146
12.7.5 Member access 147
12.7.5.1 General 147
12.7.5.2 Identical simple names and type names 149
12.7.6 Invocation expressions 149
12.7.6.1 General 149
12.7.6.2 Method invocations 150
12.7.6.3 Extension method invocations 151
12.7.6.4 Delegate invocations 153
12.7.7 Element access 154
12.7.7.1 General 154
12.7.7.2 Array access 154
12.7.7.3 Indexer access 155
12.7.8 This access 156
12.7.9 Base access 156

12.7.10 Postfix increment and decrement operators 157
12.7.11 The new operator 158
12.7.11.1 General 158
12.7.11.2 Object creation expressions 158
12.7.11.3 Object initializers 160
12.7.11.4 Collection initializers 162
12.7.11.5 Array creation expressions 163
12.7.11.6 Delegate creation expressions 165
12.7.11.7 Anonymous object creation expressions 166
12.7.12 The typeof operator 168
12.7.13 The sizeof operator 169
12.7.14 The checked and unchecked operators 170
12.7.15 Default value expressions 172
12.7.16 Anonymous method expressions 173
12.8 Unary operators 173
12.8.1 General 173
12.8.2 Unary plus operator 173
12.8.3 Unary minus operator 173
12.8.4 Logical negation operator 174
12.8.5 Bitwise complement operator 174
12.8.6 Prefix increment and decrement operators 175
12.8.7 Cast expressions 176
12.8.8 Await expressions 176
12.8.8.1 General 176
12.8.8.2 Awaitable expressions 177
12.8.8.3 Classification of await expressions 177
12.8.8.4 Run-time evaluation of await expressions 177
12.9 Arithmetic operators 178
12.9.1 General 178
12.9.2 Multiplication operator 178
12.9.3 Division operator 179
12.9.4 Remainder operator 180
12.9.5 Addition operator 181
12.9.6 Subtraction operator 183
12.10 Shift operators 185
12.11 Relational and type-testing operators 186
12.11.1 General 186
12.11.2 Integer comparison operators 187
12.11.3 Floating-point comparison operators 188
12.11.4 Decimal comparison operators 188
12.11.5 Boolean equality operators 189
12.11.6 Enumeration comparison operators 189
12.11.7 Reference type equality operators 189
12.11.8 String equality operators 191
12.11.9 Delegate equality operators 191
12.11.10 Equality operators between nullable value types and the null literal 192
12.11.11 The is operator 192
12.11.12 The as operator 193
12.12 Logical operators 194
12.12.1 General 194

12.12.2 Integer logical operators ... 194
12.12.3 Enumeration logical operators ... 195
12.12.4 Boolean logical operators ... 195
12.12.5 Nullable Boolean & and | operators ... 195
12.13 Conditional logical operators ... 196
12.13.1 General ... 196
12.13.2 Boolean conditional logical operators ... 197
12.13.3 User-defined conditional logical operators ... 197
12.14 The null coalescing operator ... 198
12.15 Conditional operator ... 198
12.16 Anonymous function expressions ... 199
12.16.1 General ... 199
12.16.2 Anonymous function signatures ... 201
12.16.3 Anonymous function bodies ... 201
12.16.4 Overload resolution ... 202
12.16.5 Anonymous functions and dynamic binding ... 203
12.16.6 Outer variables ... 203
12.16.6.1 General ... 203
12.16.6.2 Captured outer variables ... 203
12.16.6.3 Instantiation of local variables ... 204
12.16.7 Evaluation of anonymous function expressions ... 206
12.16.8 Implementation Exmple ... 206
12.17 Query expressions ... 209
12.17.1 General ... 209
12.17.2 Ambiguities in query expressions ... 210
12.17.3 Query expression translation ... 210
12.17.3.1 General ... 210
12.17.3.2 select and group … by clauses with continuations ... 211
12.17.3.3 Explicit range variable types ... 211
12.17.3.4 Degenerate query expressions ... 212
12.17.3.5 From, let, where, join and orderby clauses ... 212
12.17.3.6 Select clauses ... 216
12.17.3.7 Group clauses ... 216
12.17.3.8 Transparent identifiers ... 216
12.17.4 The query-expression pattern ... 218
12.18 Assignment operators ... 219
12.18.1 General ... 219
12.18.2 Simple assignment ... 220
12.18.3 Compound assignment ... 222
12.18.4 Event assignment ... 223
12.19 Expression ... 223
12.20 Constant expressions ... 223
12.21 Boolean expressions ... 225

13. Statements ... 227

13.1 General ... 227
13.2 End points and reachability ... 227
13.3 Blocks ... 229
13.3.1 General ... 229
13.3.2 Statement lists ... 229

13.4 The empty statement 230
13.5 Labeled statements 230
13.6 Declaration statements 231
13.6.1 General 231
13.6.2 Local variable declarations 231
13.6.3 Local constant declarations 233
13.7 Expression statements 233
13.8 Selection statements 234
13.8.1 General 234
13.8.2 The if statement 234
13.8.3 The switch statement 234
13.9 Iteration statements 238
13.9.1 General 238
13.9.2 The while statement 238
13.9.3 The do statement 239
13.9.4 The for statement 239
13.9.5 The foreach statement 240
13.10 Jump statements 243
13.10.1 General 243
13.10.2 The break statement 244
13.10.3 The continue statement 245
13.10.4 The goto statement 245
13.10.5 The return statement 246
13.10.6 The throw statement 247
13.11 The try statement 248
13.12 The checked and unchecked statements 251
13.13 The lock statement 251
13.14 The using statement 252
13.15 The yield statement 254

14. Namespaces 257

14.1 General 257
14.2 Compilation units 257
14.3 Namespace declarations 257
14.4 Extern alias directives 259
14.5 Using directives 259
14.5.1 General 259
14.5.2 Using alias directives 260
14.5.3 Using namespace directives 264
14.6 Namespace member declarations 265
14.7 Type declarations 266
14.8 Qualified alias member 266
14.8.1 General 266
14.8.2 Uniqueness of aliases 268

15. Classes 269

15.1 General 269
15.2 Class declarations 269
15.2.1 General 269
15.2.2 Class modifiers 269

15.2.2.1 General ... 269
15.2.2.2 Abstract classes ... 270
15.2.2.3 Sealed classes ... 270
15.2.2.4 Static classes ... 271
15.2.3 Type parameters ... 272
15.2.4 Class base specification ... 272
15.2.4.1 General ... 272
15.2.4.2 Base classes ... 272
15.2.4.3 Interface implementations ... 274
15.2.5 Type parameter constraints ... 275
15.2.6 Class body ... 280
15.2.7 Partial declarations ... 280
15.3 Class members ... 281
15.3.1 General ... 281
15.3.2 The instance type ... 283
15.3.3 Members of constructed types ... 283
15.3.4 Inheritance ... 284
15.3.5 The new modifier ... 285
15.3.6 Access modifiers ... 285
15.3.7 Constituent types ... 286
15.3.8 Static and instance members ... 286
15.3.9 Nested types ... 287
15.3.9.1 General ... 287
15.3.9.2 Fully qualified name ... 287
15.3.9.3 Declared accessibility ... 287
15.3.9.4 Hiding ... 288
15.3.9.5 this access ... 288
15.3.9.6 Access to private and protected members of the containing type ... 289
15.3.9.7 Nested types in generic classes ... 290
15.3.10 Reserved member names ... 291
15.3.10.1 General ... 291
15.3.10.2 Member names reserved for properties ... 291
15.3.10.3 Member names reserved for events ... 292
15.3.10.4 Member names reserved for indexers ... 292
15.3.10.5 Member names reserved for finalizers ... 292
15.4 Constants ... 292
15.5 Fields ... 294
15.5.1 General ... 294
15.5.2 Static and instance fields ... 295
15.5.3 Readonly fields ... 295
15.5.3.1 General ... 295
15.5.3.2 Using static readonly fields for constants ... 296
15.5.3.3 Versioning of constants and static readonly fields ... 296
15.5.4 Volatile fields ... 297
15.5.5 Field initialization ... 298
15.5.6 Variable initializers ... 298
15.5.6.1 General ... 298
15.5.6.2 Static field initialization ... 299
15.5.6.3 Instance field initialization ... 300
15.6 Methods ... 301

15.6.1 General 301
15.6.2 Method parameters 303
15.6.2.1 General 303
15.6.2.2 Value parameters 304
15.6.2.3 Reference parameters 305
15.6.2.4 Output parameters 305
15.6.2.5 Parameter arrays 306
15.6.3 Static and instance methods 309
15.6.4 Virtual methods 309
15.6.5 Override methods 311
15.6.6 Sealed methods 313
15.6.7 Abstract methods 314
15.6.8 External methods 315
15.6.9 Partial methods 315
15.6.10 Extension methods 318
15.6.11 Method body 319
15.7 Properties 319
15.7.1 General 319
15.7.2 Static and instance properties 321
15.7.3 Accessors 321
15.7.4 Automatically implemented properties 326
15.7.5 Accessibility 326
15.7.6 Virtual, sealed, override, and abstract accessors 328
15.8 Events 329
15.8.1 General 329
15.8.2 Field-like events 331
15.8.3 Event accessors 332
15.8.4 Static and instance events 333
15.8.5 Virtual, sealed, override, and abstract accessors 334
15.9 Indexers 334
15.10 Operators 338
15.10.1 General 338
15.10.2 Unary operators 339
15.10.3 Binary operators 340
15.10.4 Conversion operators 340
15.11 Instance constructors 343
15.11.1 General 343
15.11.2 Constructor initializers 344
15.11.3 Instance variable initializers 344
15.11.4 Constructor execution 345
15.11.5 Default constructors 346
15.12 Static constructors 347
15.13 Finalizers 349
15.14 Iterators 351
15.14.1 General 351
15.14.2 Enumerator interfaces 351
15.14.3 Enumerable interfaces 351
15.14.4 Yield type 351
15.14.5 Enumerator objects 351
15.14.5.1 General 351

15.14.5.2 The MoveNext method 352
15.14.5.3 The Current property 353
15.14.5.4 The Dispose method 353
15.14.6 Enumerable objects 354
15.14.6.1 General 354
15.14.6.2 The GetEnumerator method 354
15.15 Async Functions 354
15.15.1 General 354
15.15.2 Evaluation of a task-returning async function 355
15.15.3 Evaluation of a void-returning async function 355

16. Structs 357

16.1 General 357
16.2 Struct declarations 357
16.2.1 General 357
16.2.2 Struct modifiers 357
16.2.3 Partial modifier 358
16.2.4 Struct interfaces 358
16.2.5 Struct body 358
16.3 Struct members 358
16.4 Class and struct differences 358
16.4.1 General 358
16.4.2 Value semantics 359
16.4.3 Inheritance 360
16.4.4 Assignment 360
16.4.5 Default values 360
16.4.6 Boxing and unboxing 361
16.4.7 Meaning of this 361
16.4.8 Field initializers 362
16.4.9 Constructors 363
16.4.10 Static constructors 364
16.4.11 Automatically implemented properties 364

17. Arrays 365

17.1 General 365
17.2 Array types 365
17.2.1 General 365
17.2.2 The System.Array type 365
17.2.3 Arrays and the generic collection interfaces 366
17.3 Array creation 367
17.4 Array element access 367
17.5 Array members 367
17.6 Array covariance 367
17.7 Array initializers 368

18. Interfaces 371

18.1 General 371
18.2 Interface declarations 371
18.2.1 General 371
18.2.2 Interface modifiers 371
18.2.3 Variant type parameter lists 372