

C#

Notes for Professionals



700+ pages
of professional hints and tricks

Contents

About	1
Chapter 1: Getting started with C# Language	2
Section 1.1: Creating a new console application (Visual Studio)	2
Section 1.2: Creating a new project in Visual Studio (console application) and Running it in Debug mode	4
Section 1.3: Creating a new program using .NET Core	7
Section 1.4: Creating a new program using Mono	9
Section 1.5: Creating a new query using LingPad	9
Section 1.6: Creating a new project using Xamarin Studio	12
Chapter 2: Literals	18
Section 2.1: uint literals	18
Section 2.2: int literals	18
Section 2.3: sbyte literals	18
Section 2.4: decimal literals	18
Section 2.5: double literals	18
Section 2.6: float literals	18
Section 2.7: long literals	18
Section 2.8: ulong literal	18
Section 2.9: string literals	19
Section 2.10: char literals	19
Section 2.11: byte literals	19
Section 2.12: short literal	19
Section 2.13: ushort literal	19
Section 2.14: bool literals	19
Chapter 3: Operators	20
Section 3.1: Overloadable Operators	20
Section 3.2: Overloading equality operators	21
Section 3.3: Relational Operators	22
Section 3.4: Implicit Cast and Explicit Cast Operators	24
Section 3.5: Short-circuiting Operators	25
Section 3.6: ?: Ternary Operator	26
Section 3.7: ?. (Null Conditional Operator)	27
Section 3.8: "Exclusive or" Operator	27
Section 3.9: default Operator	28
Section 3.10: Assignment operator '='	28
Section 3.11: sizeof	28
Section 3.12: ?? Null-Coalescing Operator	29
Section 3.13: Bit-Shifting Operators	29
Section 3.14: => Lambda operator	29
Section 3.15: Class Member Operators: Null Conditional Member Access	31
Section 3.16: Class Member Operators: Null Conditional Indexing	31
Section 3.17: Postfix and Prefix increment and decrement	31
Section 3.18: typeof	32
Section 3.19: Binary operators with assignment	32
Section 3.20: nameof Operator	32
Section 3.21: Class Member Operators: Member Access	33
Section 3.22: Class Member Operators: Function Invocation	33

Section 3.23: Class Member Operators: Aggregate Object Indexing	33
Chapter 4: Conditional Statements	34
Section 4.1: If-Else Statement	34
Section 4.2: If statement conditions are standard boolean expressions and values	34
Section 4.3: If-Else If-Else Statement	35
Chapter 5: Equality Operator	36
Section 5.1: Equality kinds in c# and equality operator	36
Chapter 6: Equals and GetHashCode	37
Section 6.1: Writing a good GetHashCode override	37
Section 6.2: Default Equals behavior	37
Section 6.3: Override Equals and GetHashCode on custom types	38
Section 6.4: Equals and GetHashCode in IEqualityComparator	39
Chapter 7: Null-Coalescing Operator	41
Section 7.1: Basic usage	41
Section 7.2: Null fall-through and chaining	41
Section 7.3: Null coalescing operator with method calls	42
Section 7.4: Use existing or create new	43
Section 7.5: Lazy properties initialization with null coalescing operator	43
Chapter 8: Null-conditional Operators	44
Section 8.1: Null-Conditional Operator	44
Section 8.2: The Null-Conditional Index	44
Section 8.3: Avoiding NullReferenceExceptions	45
Section 8.4: Null-conditional Operator can be used with Extension Method	45
Chapter 9: nameof Operator	47
Section 9.1: Basic usage: Printing a variable name	47
Section 9.2: Raising PropertyChanged event	47
Section 9.3: Argument Checking and Guard Clauses	48
Section 9.4: Strongly typed MVC action links	48
Section 9.5: Handling PropertyChanged events	49
Section 9.6: Applied to a generic type parameter	49
Section 9.7: Printing a parameter name	50
Section 9.8: Applied to qualified identifiers	50
Chapter 10: Verbatim Strings	51
Section 10.1: Interpolated Verbatim Strings	51
Section 10.2: Escaping Double Quotes	51
Section 10.3: Verbatim strings instruct the compiler to not use character escapes	51
Section 10.4: Multiline Strings	52
Chapter 11: Common String Operations	53
Section 11.1: Formatting a string	53
Section 11.2: Correctly reversing a string	53
Section 11.3: Padding a string to a fixed length	54
Section 11.4: Getting x characters from the right side of a string	55
Section 11.5: Checking for empty String using String.IsNullOrEmpty() and String.IsNullOrWhiteSpace()	56
Section 11.6: Trimming Unwanted Characters Off the Start and/or End of Strings	57
Section 11.7: Convert Decimal Number to Binary,Octal and Hexadecimal Format	57
Section 11.8: Construct a string from Array	57
Section 11.9: Formatting using ToString	58
Section 11.10: Splitting a String by another string	59

Section 11.11: Splitting a String by specific character	59
Section 11.12: Getting Substrings of a given string	59
Section 11.13: Determine whether a string begins with a given sequence	59
Section 11.14: Getting a char at specific index and enumerating the string	59
Section 11.15: Joining an array of strings into a new one	60
Section 11.16: Replacing a string within a string	60
Section 11.17: Changing the case of characters within a String	60
Section 11.18: Concatenate an array of strings into a single string	61
Section 11.19: String Concatenation	61
Chapter 12: String.Format	62
Section 12.1: Since C# 6.0	62
Section 12.2: Places where String.Format is 'embedded' in the framework	62
Section 12.3: Create a custom format provider	62
Section 12.4: Date Formatting	63
Section 12.5: Currency Formatting	64
Section 12.6: Using custom number format	65
Section 12.7: Align left/ right, pad with spaces	65
Section 12.8: Numeric formats	66
Section 12.9: ToString()	66
Section 12.10: Escaping curly brackets inside a String.Format() expression	67
Section 12.11: Relationship with ToString()	67
Chapter 13: String Concatenate	68
Section 13.1: + Operator	68
Section 13.2: Concatenate strings using System.Text.StringBuilder	68
Section 13.3: Concat string array elements using String.Join	68
Section 13.4: Concatenation of two strings using \$	69
Chapter 14: String Manipulation	70
Section 14.1: Replacing a string within a string	70
Section 14.2: Finding a string within a string	70
Section 14.3: Removing (Trimming) white-space from a string	70
Section 14.4: Splitting a string using a delimiter	71
Section 14.5: Concatenate an array of strings into a single string	71
Section 14.6: String Concatenation	71
Section 14.7: Changing the case of characters within a String	71
Chapter 15: String Interpolation	73
Section 15.1: Format dates in strings	73
Section 15.2: Padding the output	73
Section 15.3: Expressions	74
Section 15.4: Formatting numbers in strings	74
Section 15.5: Simple Usage	75
Chapter 16: String Escape Sequences	76
Section 16.1: Escaping special symbols in string literals	76
Section 16.2: Unicode character escape sequences	76
Section 16.3: Escaping special symbols in character literals	76
Section 16.4: Using escape sequences in identifiers	76
Section 16.5: Unrecognized escape sequences produce compile-time errors	77
Chapter 17: StringBuilder	78
Section 17.1: What a StringBuilder is and when to use one	78
Section 17.2: Use StringBuilder to create string from a large number of records	79

Chapter 18: Regex Parsing	80
Section 18.1: Single match	80
Section 18.2: Multiple matches	80
Chapter 19: DateTime Methods	81
Section 19.1: DateTime Formatting	81
Section 19.2: DateTime.AddDays(Double)	82
Section 19.3: DateTime.AddHours(Double)	82
Section 19.4: DateTime.Parse(String)	82
Section 19.5: DateTime.TryParse(String, DateTime)	82
Section 19.6: DateTime.AddMilliseconds(Double)	83
Section 19.7: DateTime.Compare(DateTime t1, DateTime t2)	83
Section 19.8: DateTime.DaysInMonth(Int32, Int32)	83
Section 19.9: DateTime.AddYears(Int32)	84
Section 19.10: Pure functions warning when dealing with DateTime	84
Section 19.11: DateTime.TryParseExact(String, String, IFormatProvider, DateTimeStyles, DateTime)	84
Section 19.12: DateTime.Add(TimeSpan)	86
Section 19.13: Parse and TryParse with culture info	86
Section 19.14: DateTime as initializer in for-loop	87
Section 19.15: DateTime.ParseExact(String, String, IFormatProvider)	87
Section 19.16: DateTime.ToString, ToShortDateString, ToLongDateString and ToString formatted	88
Section 19.17: Current Date	88
Chapter 20: Arrays	89
Section 20.1: Declaring an array	89
Section 20.2: Initializing an array filled with a repeated non-default value	89
Section 20.3: Copying arrays	90
Section 20.4: Comparing arrays for equality	90
Section 20.5: Multi-dimensional arrays	91
Section 20.6: Getting and setting array values	91
Section 20.7: Iterate over an array	91
Section 20.8: Creating an array of sequential numbers	92
Section 20.9: Jagged arrays	92
Section 20.10: Array covariance	94
Section 20.11: Arrays as IEnumerable<> instances	94
Section 20.12: Checking if one array contains another array	94
Chapter 21: O(n) Algorithm for circular rotation of an array	96
Section 21.1: Example of a generic method that rotates an array by a given shift	96
Chapter 22: Enum	98
Section 22.1: Enum basics	98
Section 22.2: Enum as flags	99
Section 22.3: Using << notation for flags	101
Section 22.4: Test flags-style enum values with bitwise logic	101
Section 22.5: Add and remove values from flagged enum	102
Section 22.6: Enum to string and back	102
Section 22.7: Enums can have unexpected values	103
Section 22.8: Default value for enum == ZERO	103
Section 22.9: Adding additional description information to an enum value	104
Section 22.10: Get all the members values of an enum	105
Section 22.11: Bitwise Manipulation using enums	105
Chapter 23: Tuples	106

Section 23.1: Accessing tuple elements	106
Section 23.2: Creating tuples	106
Section 23.3: Comparing and sorting Tuples	106
Section 23.4: Return multiple values from a method	107
Chapter 24: Guid	108
Section 24.1: Getting the string representation of a Guid	108
Section 24.2: Creating a Guid	108
Section 24.3: Declaring a nullable GUID	108
Chapter 25: BigInteger	110
Section 25.1: Calculate the First 1,000-Digit Fibonacci Number	110
Chapter 26: Collection Initializers	111
Section 26.1: Collection initializers	111
Section 26.2: C# 6 Index Initializers	111
Section 26.3: Collection initializers in custom classes	112
Section 26.4: Using collection initializer inside object initializer	113
Section 26.5: Collection Initializers with Parameter Arrays	114
Chapter 27: An overview of C# collections	115
Section 27.1: HashSet<T>	115
Section 27.2: Dictionary<TKey, TValue>	115
Section 27.3: SortedSet<T>	116
Section 27.4: T[] (Array of T)	116
Section 27.5: List<T>	117
Section 27.6: Stack<T>	117
Section 27.7: LinkedList<T>	117
Section 27.8: Queue	118
Chapter 28: Looping	119
Section 28.1: For Loop	119
Section 28.2: Do - While Loop	120
Section 28.3: Foreach Loop	120
Section 28.4: Looping styles	121
Section 28.5: Nested loops	122
Section 28.6: continue	122
Section 28.7: While loop	123
Section 28.8: break	123
Chapter 29: Iterators	125
Section 29.1: Creating Iterators Using Yield	125
Section 29.2: Simple Numeric Iterator Example	126
Chapter 30: IEnumerable	127
Section 30.1: IEnumerable with custom Enumerator	127
Section 30.2: IEnumerable<int>	128
Chapter 31: Value type vs Reference type	129
Section 31.1: Passing by reference using ref keyword	129
Section 31.2: Changing values elsewhere	130
Section 31.3: ref vs out parameters	131
Section 31.4: Assignment	132
Section 31.5: Difference with method parameters ref and out	132
Section 31.6: Passing by reference	133
Chapter 32: Built-in Types	134
Section 32.1: Conversion of boxed value types	134

This is a sample, click download link to get the full Tutorial

CLICK BELOW

