

基礎からわかる

# C#

Makoto Nishimura  
西村 誠

Microsoftが開発したプログラミング言語「C#」の  
基本・ポイントをわかりやすく解説!

# C#の入門に 最適な1冊!

C# 6.0の  
新機能も  
紹介!

 24時間無料でサンプルデータをダウンロードできます。

 C&R研究所

## C&R研究所について

C&R研究所は新潟市にある出版社です。ユニークな社風や教育方針は新聞やテレビなどで紹介されたりします。詳細については、次のWebサイトでご覧いただくことができます。

**[www.c-r.com](http://www.c-r.com)**

また、新潟本社には2代目会社犬「ラッキー」がいます。名刺を持つ正式な社員として広報部に勤務しつつ、セラピードッグとして社内のメンタルヘルスにも貢献しています。

◎会社犬「ラッキー」

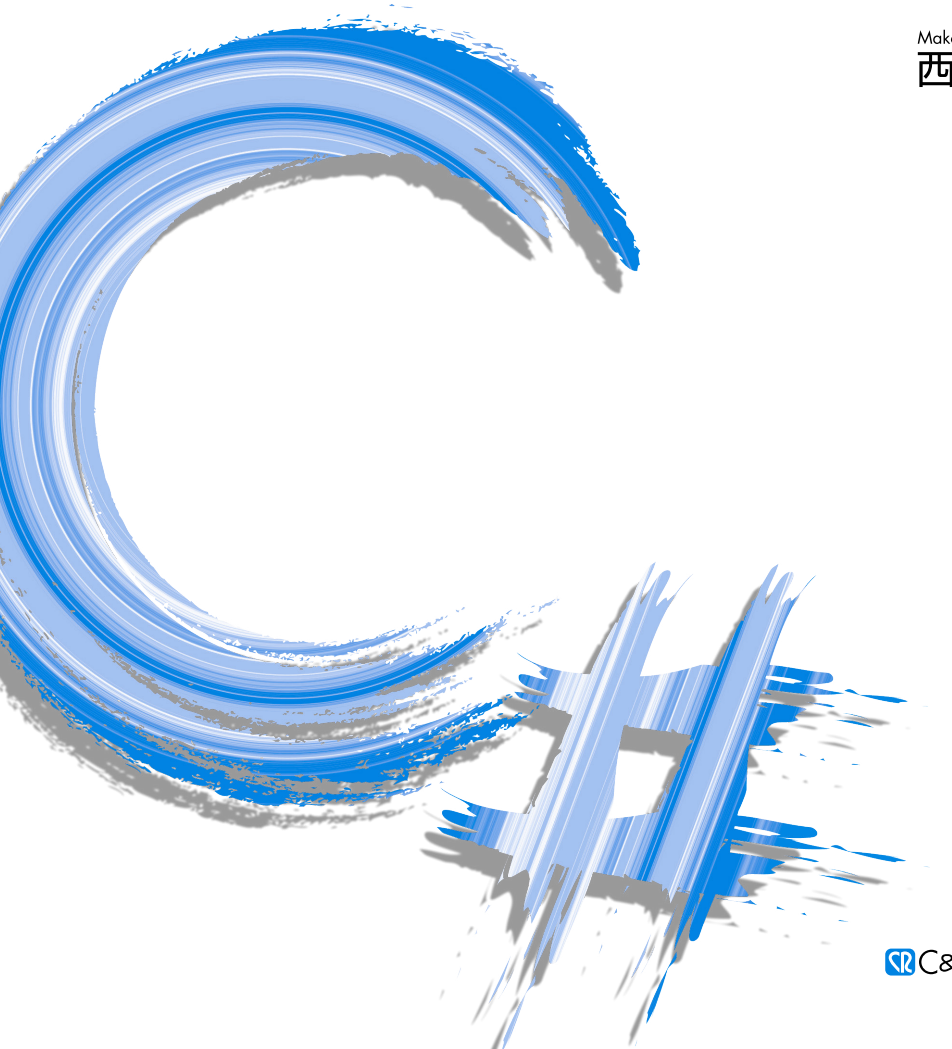


基礎からわかる

# C#

Makoto Nishimura

西村 誠



## ■権利について

- 本書に記述されている社名・製品名などは、一般に各社の商標または登録商標です。
- 本書では™、©、®は割愛しています。

## ■本書の内容について

- 本書は著者・編集者が実際に操作した結果を慎重に検討し、著述・編集しています。ただし、本書の記述内容に関わる運用結果にまつわるあらゆる損害・障害につきましては、責任を負いませんのであらかじめご了承ください。
- 本書で紹介している操作の画面は、Windows 8.1とVisual Studio Community 2015を基本にしています。他の環境では、画面のデザインや操作が異なる場合がございますので、あらかじめご了承ください。
- 本書は2015年7月現在の情報で記述しています。

## ■サンプルについて

- 本書で紹介しているサンプルは、C&R研究所のホームページ (<http://www.c-r.com>) からダウンロードすることができます。ダウンロード方法については、4ページを参照してください。
- サンプルデータの動作などについては、著者・編集者が慎重に確認しております。ただし、サンプルデータの運用結果にまつわるあらゆる損害・障害につきましては、責任を負いませんのであらかじめご了承ください。
- サンプルデータの著作権は、著者およびC&R研究所が所有します。許可なく配布・販売することは堅く禁止します。

### ●本書の内容についてのお問い合わせについて

この度はC&R研究所の書籍をお買い上げいただきましてありがとうございます。本書の内容に関するお問い合わせは、「書名」「該当するページ番号」「返信先」を必ず明記の上、C&R研究所のホームページ(<http://www.c-r.com/>)の右上の「お問い合わせ」をクリックし、専用フォームからお送りいただくか、FAXまたは郵送で次の宛先までお送りください。お電話でのお問い合わせや本書の内容とは直接的に関係のない事柄に関するご質問にはお答えできませんので、あらかじめご了承ください。

〒950-3122 新潟県新潟市北区西名目所4083-6 株式会社 C&R研究所 編集部  
FAX 025-258-2801  
「基礎からわかるC#」サポート係



# PROLOGUE

C#はC++、Javaと似た静的な型を持つ言語です。

これらの言語の良いところを取り入れながら、LINQなどの独自の機能を取り入れた言語として改良を続けてきました。最新のC# 6ではコンパイラを一新し、さらに拡張性を増し、柔軟に時代に合わせて変化する言語となりました。

2015年7月、Windows 10およびVisual Studio 2015の発売とともにC#を利用した開発はデスクトップアプリケーションやスマートフォンのみならず、IoTなどといったWindows 10が搭載された端末に拡大されました。また、Xamarinなどのテクノロジーを利用することでAndroidやiOSに向けたアプリケーションもC#で作成することができます。Unityなど、ゲーム開発の分野でもC#を用いて開発できます。C#はMicrosoft社のマルチプラットフォーム戦略の中核言語として、ますますその価値を増していくでしょう。

本書ではC#の言語仕様を素早く学ぶことができるように書かれています。もちろん、最新のVisual Studio 2015とともに利用可能となったC# 6も含まれます。

C#という言語は静的な言語という性質と高機能なエディタであるVisual Studioを組み合わせることで高い生産性をもたらしてくれます。本書を読み進めながらVisual Studioにも触れてみてください。その優れた特性が体感できるでしょう。

また、C#もバージョン6に至るまで、さまざまな機能追加や変更が加えられてきました。コレクションに対する扱いの変化、マルチタスクのスレッド処理に対する新しい解法、そのような歴史的な経緯も記述するようにしました。

本書が、多くの開発者の皆様がC#を用いて活躍するための一助となることを願います。

最後に、本書の完成にご協力いただきましたC&R研究所の吉成様、技術的なアドバイスをいただきました神守様、赤木様、大田様に心からの感謝の意を表明いたします。

2015年8月

西村 誠

# 本書について

## ▶ 対象読者について

本書は、他のプログラミング言語での開発経験がある方を読者対象としています。本書では、プログラミング言語そのものの基礎知識については解説を省略していますので、ご了承ください。

## ▶ 本書の動作環境について

本書では、Windows 8.1とVisual Studio 2015の環境で動作を確認しています。

## ▶ サンプルコードの中の▼について

本書に記載したサンプルコードは、誌面の都合上、1つのサンプルコードがページをまたがって記載されていることがあります。その場合は▼の記号で、1つのコードであることを表しています。

## ▶ サンプルファイルのダウンロードについて

本書で紹介しているサンプルデータは、C&R研究所のホームページからダウンロードすることができます。本書のサンプルを入手するには、次のように操作します。

- ① 「<http://www.c-r.com/>」にアクセスします。
- ② トップページ左上の「商品検索」欄に「181-8」と入力し、[検索]ボタンをクリックします。
- ③ 検索結果が表示されるので、本書の書名のリンクをクリックします。
- ④ 書籍詳細ページが表示されるので、[サンプルデータダウンロード]ボタンをクリックします。
- ⑤ 下記の「ユーザー名」と「パスワード」を入力し、ダウンロードページにアクセスします。
- ⑥ 「サンプルデータ」のリンク先のファイルをダウンロードし、保存します。

### サンプルのダウンロードに必要な ユーザー名とパスワード

ユーザー名 **csharp**  
パスワード **n8m7v**

※ユーザー名・パスワードは、半角英数字で入力してください。また、「j」と「j」や「k」と「k」などの大文字と小文字の違いもありますので、よく確認して入力してください。

## ▶ サンプルファイルの利用方法について

サンプルファイルは、CHAPTERごとのフォルダの中に、項目番号のフォルダに分かれています。サンプルはZIP形式で圧縮してありますので、解凍してお使いください。

それぞれのフォルダ内には、プロジェクトファイルとソースファイルが保存されています。サンプルの動作を確認するには、拡張子「.sln」のプロジェクトファイルをVisual Studioで開いて「F5」キーを押してください(24ページ参照)。なお、ソースコードを確認する場合は、同様にして拡張子「.sln」のプロジェクトファイルをVisual Studioで開くか、プロジェクト内の「Program.cs」ファイルをテキストエディタで開くことで確認できます。

## CHAPTER 01

## C#の概要

001	C#の概要	14
	▶C#とは	14
	▶C#の歴史	14
	▶.NET Framework	15
002	C#を動かす	17
	▶Visual Studioの導入	17
003	Hello C# World	21
	▶プロジェクトの作成	21
	▶初期コードの説明	22
	▶Hello Worldの記述	23
	▶コードの説明	24
004	C#の基本構文	26
	▶プログラムの構成	26
	▶文	26
	▶式	26
	▶分岐処理	27
	▶繰り返し(ループ)	30
	▶予約語	34

## CHAPTER 02

## 変数

005	変数	36
	▶変数とは	36
	▶基本的な変数の記法(宣言)	36
	▶値を代入する	36
	▶変数の初期化	37
	▶型とは	37
	▶静的と動的な型付けの違い	38
006	組み込み型	41
	▶あらかじめ用意された型	41
	▶int型	41

▶ int以外の整数型	41
▶ float型、double型	41
▶ decimal型	42
▶ string型	43
▶ char型	43
▶ bool型	43
▶ enum型	44
▶ object型	44
<b>007 変数の応用</b>	<b>46</b>
▶ キャスト	46
▶ null	48
▶ 「var」キーワードと型推論	48
▶ 「dynamic」キーワード	49
▶ 値型と参照型	50
▶ ボックス化	51
▶ 定数	51
▶ 「checked」キーワード	52
<b>008 演算子</b>	<b>53</b>
▶ さまざまな演算子	53
▶ 代入演算子(=)	53
▶ 算術演算子	53
▶ ビット演算子	55
▶ 論理演算子	55
▶ 関係演算子	57
▶ 三項演算子	59
▶ null合体演算子(??)	59
▶ 演算子の優先順位	60
▶ 演算子の結合規則	60

## CHAPTER 03

### 文字列処理の基礎

<b>009 string型</b>	<b>62</b>
▶ string型の基本	62
▶ string型の初期化	62
▶ エスケープシーケンス	62
▶ @(逐語的文字列)	63



▶ 文字列の比較	63
▶ 空文字の判定	64
▶ 文字列の連結	65
▶ 文字列の分割	66
▶ 文字列の置換	66
▶ 文字列の前後にある空白を取り除く	66
▶ 文字数を調べる	67
▶ 文字列から数値への変換	67
▶ 文字列の書式を指定する	67
▶ 数字を0パディングする	68
▶ 正規表現	68
▶ C#の文字列は変更できない	68
<b>□ 1 □ char型</b>	<b>70</b>
▶ 1文字を扱うchar型	70
▶ String型からchar型への変換	70
▶ char型からString型への変換	70
▶ 文字コードの取得	71
<b>□ 1 1 StringBuilder型</b>	<b>72</b>
▶ StringBuilder型の用途	72
▶ 文字列の連結	72
▶ StringBuilder型からString型への変換	72

## CHAPTER 04

### 関数

<b>□ 1 2 関数の基礎</b>	<b>74</b>
▶ 関数とは	74
▶ 関数の基本文型	74
▶ 関数を利用する	74
<b>□ 1 3 引数</b>	<b>76</b>
▶ 関数の引数	76
▶ 引数のデフォルト値	76
▶ 関数のオーバーロード	77
▶ 「ref」キーワード	77
▶ 引数が参照型か値型による挙動の違い	79
▶ 「out」キーワード	79
▶ 可変引数	81

014	戻り値	82
	▶ 戻り値とは	82
	▶ 戻り値がない場合	82
	▶ 処理途中の「return」文	82

## CHAPTER 05

# クラス

015	クラス	86
	▶ クラスとは	86
	▶ クラスの書き方	86
	▶ ローカル変数とフィールドの違い	87
	▶ クラスを利用する	89
	▶ クラスの公開範囲	89
	▶ コンストラクタ	90
	▶ デストラクタ	92
016	継承	93
	▶ 継承とは	93
	▶ 継承元への代入	93
	▶ 継承元の同名メソッドを呼び出す	94
	▶ 継承元のコンストラクタを明示的に呼び出す	95
	▶ 継承と公開範囲	96
	▶ 抽象クラス	96
	▶ インターフェイス	98
017	クラスの応用	100
	▶ 「virtual」キーワード	100
	▶ 匿名型	101
	▶ プロパティ	101
	▶ インデクサ	103
	▶ 静的メンバー (static)	104
	▶ 部分クラス (partial)	105
	▶ 「sealed」キーワード	106
	▶ 読み取り専用	107
	▶ 構造体	107

## CHAPTER 06

### 配列とコレクション

018	配列とコレクションの基礎	110
	▶データの集合を扱う	110
019	配列	111
	▶配列の基礎	111
	▶配列のサイズを変更する	112
	▶配列の宣言を簡略化する	112
	▶多次元配列	112
020	非ジェネリックなコレクション	113
	▶非ジェネリックは最新のC#では出番が少ない	113
	▶非ジェネリックなコレクションの問題	113
021	ジェネリックなコレクション	115
	▶ジェネリックなコレクションとは	115
	▶ジェネリックなコレクションの使い方	115
	▶「List」クラス	116
	▶「Dictionary」クラス	116
	▶「SortedList」クラス、「SortedDictionary」クラス	117
	▶「Queue」クラス、「Stack」クラス	118
	▶「LinkedList」クラス	119
	▶コレクションの処理	119
022	LINQ	122
	▶LINQの基礎	122
	▶クエリの連結	122
	▶LINQの別の記法	123
	▶遅延実行	123
	▶LINQの主な機能	124

## CHAPTER 07

### イベント

023	イベント	128
	▶イベントとは	128
	▶イベントを受け取る	128
	▶イベントを発行する	128

▶ EventHandler	129
▶ EventArgs型を変更する	129
▶ 複数のイベントハンドラーを登録する	130
▶ イベントハンドラーの登録がない場合	130
<b>024 デリゲート</b>	<b>131</b>
▶ デリゲートとは	131
▶ 複数の処理を登録する	131
▶ デリゲートとイベントの違い	132
▶ 匿名関数	132
<b>025 ラムダ式</b>	<b>133</b>
▶ ラムダ式とは	133
▶ 「delegate」キーワードが不要になり、「=>」が必要になった	133
▶ 引数が1つの場合は「()」が省略可能	133
▶ 式が1つの場合は「{ }」と「return」文が省略可能	134
▶ 外部変数の利用	134

## CHAPTER 08

### 非同期処理

<b>026 非同期処理</b>	<b>138</b>
▶ 非同期処理とは	138
▶ C#と非同期処理	138
<b>027 Thread</b>	<b>139</b>
▶ 「Thread」クラスを用いた記法	139
▶ 「ThreadPool」クラス	140
<b>028 Task</b>	<b>142</b>
▶ 「Task」クラスの基礎	142
▶ 排他制御	142
<b>029 Parallel</b>	<b>144</b>
▶ ループ処理の並列化	144
▶ 「Parallel.Invoke」メソッド	144
▶ 「Parallel.For」メソッド、「Parallel.ForEach」メソッド	145
<b>030 async/await</b>	<b>146</b>
▶ C# 5.0から追加された新しい記法	146
▶ 「async/await」の基礎	146
▶ async/awaitに対応したメソッド	146

## CHAPTER 09

### その他の要素

<b>031</b>	<b>エラーハンドリング</b> .....	148
	▶トライキャッチ .....	148
	▶finally .....	149
	▶例外をスローする .....	149
<b>032</b>	<b>リフレクションと属性</b> .....	150
	▶リフレクション .....	150
	▶属性 .....	150
	▶リフレクションで属性情報を取得する .....	150
	▶属性を自作する .....	151
<b>033</b>	<b>using</b> .....	153
	▶「using」キーワードのさまざまな使い方 .....	153
	▶名前空間の読み込み .....	153
	▶エイリアス .....	153
	▶リソースを利用する .....	153
<b>034</b>	<b>object型</b> .....	155
	▶すべてのクラスはobject型を継承する .....	155
	▶「Equals」メソッド .....	155
	▶「ToString」メソッド .....	155
	▶「GetType」メソッド .....	156
	▶「GetHashCode」メソッド .....	156
<b>035</b>	<b>ポインタの利用</b> .....	157
	▶C#のポインタ .....	157
	▶「unsafe」キーワード .....	157
	▶「/unsafe」オプションが必要 .....	157
	▶メソッド全体に「unsafe」キーワードを指定する .....	158
<b>036</b>	<b>プリプロセッサ</b> .....	159
	▶プリプロセッサとは .....	159
	▶プリプロセッサのサンプル .....	159
	▶#define .....	160
	▶#if .....	160
	▶#region .....	160
<b>037</b>	<b>C# 6.0の新機能</b> .....	162
	▶C# 6.0の新機能について .....	162

▶ 自動プロパティの初期化 .....	162
▶ ラムダ式による関数の記述 .....	162
▶ null条件演算子 .....	162
▶ 「string.Format」メソッドの簡略記法 .....	162
▶ 「nameof」演算子 .....	163
▶ インデックス初期化子 .....	163
▶ using static .....	163
▶ 例外処理catch内でのawaitの利用 .....	163
▶ 例外処理のフィルタリング .....	164
▶ 文字列リテラル内での変数の埋め込み .....	164

## COLUMN

■ アプリケーションを開発するためには .....	16
■ Communityエディションの利用条件について .....	20
■ 「using」と「dll」について .....	25
■ 予約語を識別子として使いたい場合 .....	34
■ Stringとstring .....	45
■ C#における関数 .....	84
■ 「const」と「readonly」 .....	108
■ タイマーを用いたイベント .....	136
■ 「async/await」とWinRT .....	138
● 索引 .....	165



CHAPTER

01

C#の概要



## C#の概要

## ● C#とは

C#はMicrosoft社の開発したオブジェクト指向型のプログラミング言語です。

その由来から主に同社のテクノロジーであるWindows OSやスマートフォン(Windows Phone)、Webサーバー(IIS)上で動作するアプリケーションを開発するための言語でしたが、最近ではiOSやAndroidのアプリケーションや、Unityによるゲーム開発など、利用できるプラットフォームが増え、さまざまなシーンで活躍できる言語となりました。

LINQ(統合言語クエリ)やラムダ式、非同期処理の可読性を高める「await」キーワード、「async」キーワードなどが特徴的な機能です。これらについては本書でも基礎から解説していきます。

C++やJavaと比較すると後発の言語のため、それらの良さを上手く取り入れ、バランスの良い言語になっています。また、静的型付けと高機能な開発ツール「Visual Studio」の相性が良く、コーディングする上でさまざまな補助を受けることができます。

## ● Visual C#

URL <https://msdn.microsoft.com/ja-jp/library/kx37x362.aspx>

## ● Visual C#

The screenshot shows the Microsoft Developer Network page for Visual C#. The page title is "Visual C#" and it includes a table of contents for the section.

**Visual C#**

このセクションの内容

- Visual C# について
  - C# 言語または Visual Studio を初めて使用するプログラマー向けに、C# の機能を紹介します。また、Visual Studio のヘルプを検索するためのODDリンクも示します。
- Visual C# 開発環境の採用
  - Visual C# 開発環境について紹介します。
- C# プログラムの構造
  - C# 開発環境のインストール方法に関する情報および実際の例を提供します。
- C# リファレンス
  - C# プログラムの構造、キーワード、型、演算子、宣言、リファレンス、フィールド、コンパイラ ステート、およびコンパイルオプションと警告に関する詳細なリファレンスを提供します。
- C# のガバナンス
  - C# のドキュメントと開発された MSDN Code Gallery
- C# チュートリアル
  - C# を使用するプログラマー向けのチュートリアルと、各チュートリアルに関する簡単な説明のリンクをリストアップします。

## ● C#の歴史

C#はC、C++、Javaなどと比較すると新しいプログラミング言語であり、最近登場したGo言語やSwiftなどに比べると古い言語です。今から10年以上前に登場した言語と聞くと、かなり古い印象を持つ方もいらっしゃるでしょうが、バージョンアップを重ねて新しい機能を追加し続け、最新バージョンのC# 6.0が2015年7月に登場しました。



### ◆ C# 1.0

C#は当時のJavaやC++の良いところを取り入れたバランスの良い言語として登場しました。Delphiという言語の開発者がMicrosoftに移籍して開発に携わったという経緯もあり、Delphiに似た部分もありました。

ここからC#はバージョンを重ねるごとに大きく変化し、時代にあったプログラミング言語として成長していきます。

### ◆ C# 2.x

C# 2.xではジェネリクスや静的クラス、「yield」キーワードなどが導入されました。匿名メソッドが登場したのもこのバージョンです。

C# 2.x以前には、静的クラスが利用できなかったというのは驚きです。ジェネリクスや「yield」キーワードの登場でデータの集合(コレクション)がより便利に安全に利用できるようになりました。匿名メソッドはC# 3.xで登場するラムダ式と合わせてC#のメソッドを使いやすく拡張しました。

### ◆ C# 3.x

C# 3.xでは「var」キーワードと型推論、ラムダ式、初期化式の簡略化などが追加されました。そして何より、クエリ式(LINQ)というC#特有の機能が追加されました。これらの機能を利用するようになると、C#の記述が大きく変わります。そういう点ではC#らしい機能が色々と揃ってきたバージョンといえます。

### ◆ C# 4.x

C# 4.xで「dynamic」キーワードが追加されました。

少し前にC#は「静的な型付け」な言語と表現しましたが、厳密にいうと「dynamic」キーワードはC#に動的な型付けの性質を加えることができます。

### ◆ C# 5.x

C# 5.xでは非同期処理の可読性をあげる記法(「async」「await」キーワード)が追加されました。同時に登場したランタイムWinRTでは応答に50ms以上の時間がかかる可能性がある処理は非同期処理で行うように設計されており、「async」「await」キーワードを利用するシーンが多くなっています。

### ◆ C# 6.0

C# 6.0では言語仕様面での大きな変更は加えられておらず、C#がより便利に記述できるような記法の変更などが加えられました。言語仕様上の大きな変更がない代わりに、プログラムを動作可能な形式にコンパイルするコンパイラーが一新されるという大きくその土台が変更されています。

## 🌐 .NET Framework

.NET Frameworkとは、C#で書かれたプログラミングが実行される環境のことです。C#だけでなく、Visual BasicやF#, C++などの言語も実行可能な共通言語ランタイム(共通言語ランタイムは「Common Language Runtime」を略してCLRとも呼ばれる)という実行基盤を持ちます。