

改訂2版

基礎からわかる

Go言語

Furukawa Noboru
古川 昇

今注目されているプログラミング言語「Go」の
基礎から応用、ポイントを丁寧に解説!

Go言語入門の 定番書!

Go 1.4
対応!

 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

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」



改訂2版

基礎からわかる

Go言語

Furukawa Noboru

古川 昇



■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コードの実行結果などは、環境によって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。
- 本書の内容は、2015年6月現在の情報を基に記述しています。

■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、5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、著者及びC&R研究所が所有します。許可なく配布・販売することは強く禁止します。

●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
「改訂2版 基礎からわかる Go言語」サポート係



PROLOGUE

Go言語は、2009年11月に発表されたソフトウェア開発言語です。正式版であるバージョン1がリリースされたのも2012年3月と、比較的新しい開発言語です。

Go言語は、Google社のエンジニア達の手によって生み出されました。最初は仕事の合間に進められた小さなプロジェクトだったそうです。その後、オープンソース化され、Google社の手を離れてからもコミュニティの協力によって改良と機能追加が重ねられ、バージョン1が生み出されました。その後もパフォーマンスの改善など、日々進化しています。

Google社といえば検索エンジンやスマートフォン用OSのAndroidが有名で、優秀なエンジニアが多いことでも知られています。彼らが新しい言語を作ろうと思った背景には、年々、複雑化・肥大化していくソフトウェア開発の開発効率を向上させたいという思いがありました。そこで彼らがとったアプローチは、一言でいえばシンプル化でした。シンプルな構文は学習しやすく、コンパイル速度の向上にも寄与しました。継承という概念を持たないシンプルな型システムは、型の定義を簡単にしました。例外処理も本当にハンドリングが必要なとき以外は、何もする必要がなくなりました。

Go言語はC言語系の流れを組むコンパイラ言語ですが、新規に言語を設計したことで、既存の各種開発言語のよいところだけを取り入れつつ、さまざまな新しいアイデアを加えた結果として、他に類を見ない素晴らしい言語となりました。はじめてGo言語のプログラムを書き始めたときに、その構文のわかりやすさ、書きやすさに驚きました。

まだ生まれて日が浅い言語ですが、最近では実際に開発現場でも使われ始めたようです。多くのプログラマに使われるようになるにはまだまだ時間がかかると思いますが、いずれこの素晴らしい言語がさまざまな場で活躍する日が来るとしています。本書は、Go言語を学ぼうという方のための入門書です。本書ではGo言語の基本的な説明を中心に説明し、CHAPTER 09では実際に開発で役立つサンプルコードを逆引き形式で説明しています。

最後に、本書の執筆・制作にあたって、お世話になったC&R研究所の吉成様はじめスタッフの皆様には心から感謝申し上げます。そして、本書がGo言語の開発に興味を持ち始めた読者の皆様に少しでもお役に立てれば幸いです。

2015年6月

株式会社エイベル 第二システムソリューション部 部長
古川 昇

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。

📖 開発環境について

本書のサンプルコードは、Go 1.4.2を対象としています。また、Go言語を動かす環境として、Linuxを想定しています。環境によって異なる部分がある場合は、その都度、他の環境での解説を記述します。なお、本書では、LinuxやMac OS Xを合わせてUnix系と記述します。Windows 7やWindows 8.1などを合わせてWindows系と記述します。

なお、本書のサンプルコードは、次の環境で実行を確認しています。

- Ubuntu 15.04 / Go 1.4.2
- Mac OS X 10.10.3 / Go 1.4.2
- Windows 7および8.1 Update / Go 1.4.2

📖 コードについて

内容を簡潔にするために、エラー処理の一部を省略しています。また、コードを解説するためのコメントは記述していますが、関数のコメントは記述していません。

📖 コードの実行結果について

本書に記載しているコードの実行結果は、Ubuntu 15.04の結果となります。必要に応じてWindows環境の実行結果も並記しています。カレントディレクトリなどについては実行環境によって異なりますので、お使いの環境に合わせて読み替えてください。

また、コードの実行によってファイルが出力されるサンプルがあります。出力されるファイルは、文字コードがUTF-8(BOMなし)、改行コードがLFとなります。ファイルを開いて内容を確認する場合は、文字コードのUTF-8と改行コードのLFに対応したテキストエディタなどをお使いください。

📖 本書の表記方法

本書の表記についての注意点は、次のようになります。

◆ サンプルコードの中の「\」（バックスラッシュ）と「¥」（円記号）について

環境によっては、「\」（バックスラッシュ）は「¥」（円記号）で表示されます。本書の誌面に掲載しているサンプルコードでは「\」で表記していますので、必要に応じて読み替えてください。なお、コマンドプロンプトへの入力などは「¥」で表記しています。

◆ 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

本書のサンプルデータは、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「178-8」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 **k2kgo**
パスワード **fg4tm**

※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。

🔗 サンプルコードの利用方法

サンプルファイルは、CHAPTERごとのフォルダの中に収録されています。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。

また、サンプルファイルは、文字コードがUTF-8(BOMなし)、改行コードがLFとなります。ファイルを開いて内容を確認する場合や編集する場合は、文字コードのUTF-8と改行コードのLFに対応したテキストエディタなどをお使いください。

なお、サンプルの実行方法については、28ページ、または37ページを参照してください。

CHAPTER-01

Go言語による開発の概要

001	Go言語とは	16
	▶ Go言語の概要	16
	▶ Go言語の特徴	17
	▶ バージョンごとの主要な変更点	18
002	Go言語の開発環境の構築	21
	▶ 開発環境について	21
	▶ Go言語ツールのダウンロード	21
	▶ Microsoft Windowsへのインストール	22
	▶ Mac OS Xへのインストール	25
	▶ LinuxまたはFreeBSDへのインストール	27
003	開発環境の動作確認	28
	▶ 簡単なプログラミング	28
	▶ コンパイルと実行	28

CHAPTER-02

Go言語の基本

004	基本的な文法について	34
	▶ ソースファイルの例	34
	▶ 「Hello.go」の解説	34
	▶ 文末の「:」(セミコロン)は不要	35
	▶ ソースコード内のコメント	35
005	コンパイルと実行について	37
	▶ ソースファイルのコンパイル	37
	▶ コンパイルと実行を同時に行う	37
006	キーワードについて	38
	▶ Go言語のキーワードの一覧	38
007	基本的な型について	39
	▶ 基本的な型	39
	▶ 論理値型	39
	▶ 数値型	40
	▶ 文字列型	41

008	型の宣言について	44
	▶型の宣言	44
009	変換について	45
	▶型の変換	45
010	演算子について	47
	▶Go言語の演算子	47
	▶算術演算子	47
	▶比較演算子	50
	▶論理演算子	51
	▶インクリメント文とデクリメント文	52
	▶演算子の優先順位	53
011	代入について	54
	▶代入とは	54
	▶ブランク識別子について	54
012	変数について	55
	▶Go言語の変数の宣言	55
	▶「var」キーワードを使った基本的な変数の宣言	55
	▶変数の宣言と同時に初期値を与える	55
	▶変数の宣言時に型を省略する	56
	▶省略書式を使った変数の宣言	57
013	リテラルについて	58
	▶Go言語のリテラル	58
014	定数について	61
	▶定数の宣言	61
015	パッケージとインポート・エクスポートについて	64
	▶パッケージ	64
	▶エクスポート	64
	▶インポート	65
016	繰り返し処理について	68
	▶「for」文	68
	▶C言語やJavaの「for」文に相当する書式	68
	▶C言語やJavaの「while」文に相当する書式	69
	▶無限ループ	70
	▶「range」式を使用する書式	71

▶「break」文	73
▶「continue」文	73
□ 17 条件分岐について	75
▶「if」文	75
▶「switch」文	76
□ 18 ポインタについて	81
▶Go言語のポインタ	81
▶ポインタ渡しと値渡し	82
▶「new」組み込み関数によるメモリの割り当て	83
□ 19 ゼロ値について	84
▶ゼロ値	84
□ 20 ラベルについて	86
▶ラベルとは	86
▶ラベルにbreakする	86
▶ラベルにcontinueする	87
▶「goto」文	88

CHAPTER-03

関数とメソッド

□ 21 関数について	90
▶Go言語の関数	90
▶関数の宣言	90
▶多値を返す関数の宣言	91
▶関数から返却された多値を直接、他の関数に渡す	92
▶可変長パラメータ	93
▶戻り値に名前を付ける	94
▶関数リテラル	94
▶関数型	96
□ 22 メソッドについて	98
▶Go言語のメソッド	98
▶メソッドの宣言	98
▶メソッドのレシーバをポインタにする	99
▶メソッド値	100

023	関数の遅延実行について	102
▶	関数の遅延実行	102
▶	遅延実行はリソースの解放に便利	103

CHAPTER-04

構造体

024	構造体について	106
▶	構造体とは	106
▶	構造体型の書式	106
▶	構造体型に名前を付ける	107
025	特殊なフィールドについて	108
▶	ブランクフィールド	108
▶	匿名フィールドと埋め込み	108
026	構造体の初期化について	110
▶	リテラルを使った構造体の初期化	110
▶	構造体リテラルにおける埋め込み	111

CHAPTER-05

インタフェース

027	インタフェースについて	114
▶	Go言語のインタフェース	114
▶	インタフェース型の書式	114
▶	インタフェースは自動的に実装される	116
▶	空インタフェース	116
028	インタフェースの埋め込みについて	119
▶	インタフェースの埋め込み	119
029	型の変換について	120
▶	型アサーション	120
▶	「型switch」文	121

CHAPTER-06

配列・スライス・マップ

030	配列について	126
	▶配列とは	126
	▶配列型の書式	126
	▶配列の長さを取得する	126
	▶配列の要素へのアクセス	127
031	配列の初期化	129
	▶リテラルを使った配列の初期化	129
032	スライスについて	131
	▶スライスとは	131
	▶スライス型の書式	131
	▶スライス式	131
	▶スライスは参照型	132
	▶スライスのキャパシティ	133
	▶文字列のスライス	135
	▶スライスへ要素を追加する	136
	▶スライスの要素をコピーする	137
	▶スライスを可変長パラメータに直接、受け渡す	138
033	スライスの初期化	139
	▶配列とスライスを同時に作成する	139
	▶リテラルによるスライスの初期化	140
034	マップについて	141
	▶マップとは	141
	▶マップ型の書式	141
	▶マップの長さを取得する	141
	▶マップの作成	141
	▶マップへの値の出し入れ	142
	▶マップのキー存在確認	143
	▶マップから要素を削除する	144
035	マップの初期化	145
	▶リテラルを使ったマップの初期化	145

CHAPTER-07

エラーハンドリング

036	戻り値によるエラーハンドリングについて	148
	▶ 戻り値によるエラーハンドリング	148
	▶ 自作関数のエラー処理	149
037	パニックとリカバリ	151
	▶ パニックとは	151
	▶ リカバリとは	153
038	ランタイムパニックについて	155
	▶ ランタイムパニックとは	155

CHAPTER-08

並列処理

039	ゴルーチンについて	158
	▶ ゴルーチンとは	158
	▶ 「go」文の書式	158
	▶ ゴルーチンの中断と再開	160
	▶ ゴルーチンを使用する上での注意点	160
040	ゴルーチン間における通信	163
	▶ チャンネルとは	163
	▶ チャンネル型の書式	163
	▶ チャンネルの作成	163
	▶ チャンネルへの値の送受信	163
	▶ チャンネルのクローズ	164
	▶ チャンネルのキャパシティとバッファリングされている要素数を取得する	165
	▶ チャンネルを利用した同期	166
	▶ セマフォの代用	167
	▶ 値の共用	169
041	「select」文について	172
	▶ 「select」文とは	172
	▶ 「select」文の書式	172

CHAPTER-09

逆引きサンプル

□ 4.2	入出力	176
▶	値を文字列にフォーマットする	176
▶	標準出力に値を出力する	177
▶	標準エラー出力に値を出力する	178
▶	標準入力から値を読み込む	179
▶	ファイルから入出力を行う	180
▶	CSVファイルを出力する	182
▶	CSVファイルを読み込む	183
□ 4.3	ファイルシステム	185
▶	ディレクトリを作成する	185
▶	ファイル/ディレクトリを削除する	185
▶	カレントディレクトリを変更する	186
▶	ファイル/ディレクトリの名前を変更する	187
▶	テンポラリファイルを作成する	188
▶	ディレクトリ内のファイル一覧を取得する	189
□ 4.4	データ処理	190
▶	リストの作成と要素の追加を行う	190
▶	リストの要素数を取得する	190
▶	リストの要素の値を取得・変更する	191
▶	2つのリストを結合する	193
▶	リストを空にする	194
▶	ハッシュ値(SHA-1、MD5)を求める	195
▶	BASE64エンコード/デコードを行う	196
▶	URLエンコード/デコードを行う	197
▶	16進ダンプ出力を行う	198
▶	乱数を取得する	199
□ 4.5	日付・時刻	200
▶	現在時刻を取得する	200
▶	時刻を文字列にフォーマットする	200
▶	日時を指定して「Time」型の値を作成する	201
▶	「Time」型に時間を加算・減算する	202
▶	「Time」型に日付を加算・減算する	203
▶	曜日を取得する	203
▶	時間を指定してスリープする	204

▶経過時間を算出する	205
▶UNIX時間を取得する	206
▶一定時間ごとに処理を行う	207
□46 文字列	209
▶文字列を比較する	209
▶文字列を結合する	210
▶部分文字列を取得する	211
▶文字列をトリムする	211
▶文字列を大文字・小文字にする	212
▶文字列を分割する	213
▶指定した文字列の位置を取得する	213
□47 文字列と数値の相互変換	215
▶文字列を整数型に変換する	215
▶文字列を浮動小数点型に変換する	215
▶整数を文字列に変換する	216
▶浮動小数点を文字列に変換する	217
□48 数学処理	219
▶三角関数を使用する	219
▶小数点以下の切り上げ/切り捨てを行う	220
▶平方根を計算する	221
□49 ネットワーク	222
▶ホスト名からIPアドレスを取得する	222
▶TCP/IP通信を行う	222
▶HTTP通信を行う	224
▶HTTPのPOST通信を行う	225
▶Webサーバを作成する	226
□50 ログ出力	229
▶標準エラー出力にログを出力する	229
▶ログをファイルに出力する	229
□51 その他	231
▶コマンドラインパラメータを取得する	231
▶プログラムを指定した終了コードで終了する	232
▶外部プログラムを実行する	233

COLUMN

■ Go言語のAndroidアプリへの対応	19
■ Go言語のマスコット	20
■ Eclipseを使った開発	29
■ Eclipseのエディタ上でオートコンプリート機能を使う	31
■ Go言語をインストールせずに試してみる	32
■ Go言語の公式サイトはGo言語で動いている	32
■ ソースコードの整形	36
■ 識別子に日本語を使用する	63
■ 独自パッケージを使用する	67
■ 「nil」について	85
■ 「init」関数	97
■ ビルド時に条件を指定する	104
■ インタフェースの「nil」	117
■ インタフェースの命名規則	118
■ コードからドキュメントを作成する	124
■ マップのキーとして使用できる値	144
■ 「nil」マップ	144
■ 環境変数「GOTRACEBACK」による異常終了時の動作変更	156
■ Go言語で作られた「Docker」	162
■ 標準以外のパッケージライブラリを使用する	208
■ 非推奨組み込み関数	218
● 索引	235



CHAPTER

01

Go言語による開発の概要

Go言語とは

Go言語の概要

Go言語とは、Googleが2009年11月に公開したプログラミング言語です。Googleの社員であるロブ・パイク氏やケン・トンプソン氏が開発に携わっています。Go言語はコンパイル言語で、構文としてはC言語を基にしていますが、ほぼ別物といってもいいほどに進化しています。設計方針として、既存の言語のよいところは取り入れ、必要性の低いものは省き、新たなアイデアを詰め込んだ結果、誕生したのがGo言語です。

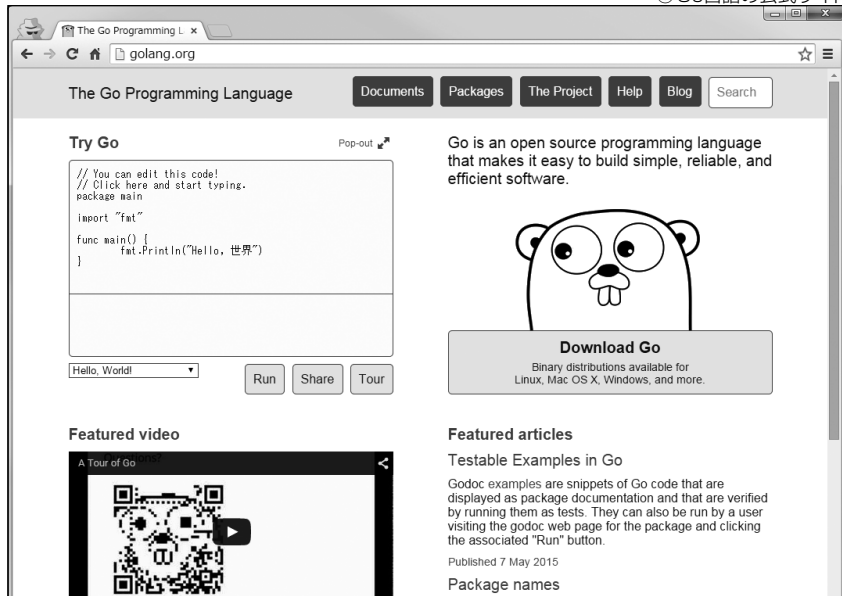
発表された当初は、LinuxとMac OS Xしかサポートしていませんでしたが、2012年3月にリリースされたバージョン1からはWindowsもサポートされるようになりました。また、同時にバイナリも配布されるようになり、それまではソースファイルからコンパイルする必要がありましたが、インストールが簡単になっています。

Go言語の公式サイトは、次のURLになります。

- The Go Programming Language

URL <http://golang.org/>

Go言語の公式サイト



なお、著者が所属する会社で公式サイト日本語訳を行っており、次のURLで公開しています。

- golang.jp - プログラミング言語Goの情報サイト

URL <http://golang.jp/>

●公式サイト日本語訳をまとめているサイト



● Go言語の特徴

Go言語の特徴は、次のようになります。

◆ オープンソース

Go言語はオープンソースプロジェクトであり、「BSDスタイル・ライセンス」(BSD-style license)で公開されています。そのため、Go言語自体のソースを見ることもできますし、必要であれば自由に機能を追加・拡張することもできます。

◆ シンプルな構文

新規に言語をデザインすることで、C言語の構文を基本としながらも、シンプルで新しい構文体系となりました。シンプルさに徹底的にこだわった結果、たとえば、文末の「;」(セミicolon)さえ不要になり、ソースコードの量まで減らしています。

◆ コンパイルが速い

C言語系のコンパイラに比べ、格段にコンパイル速度が速くなっています。これには構文がシンプルになったことも寄与していますが、C言語のようなインクルードファイルを採用しなかったことで、コンパイル時のオーバーヘッドが削減されました。

◆ 並列処理をサポート

並列処理を言語としてサポートしているため、複数の処理を同時に、かつ、効率的に実行することができます。この並列処理は「ゴルーチン」と呼ばれる実行単位で管理され、必要に応じてOSのスレッド上に割り当てられて実行されます。プログラマは、どのプロセッサ、スレッド上で実行されるかを意識することなく、並列処理を記述できます。

◆ ガベージコレクションによるメモリ管理

ガベージコレクションとは、メモリが使用されているかどうかを自動的に判断し、使用されなくなったメモリを自動的に開放する機構です。ガベージコレクションにより、煩雑なメモリ管理用のコードを記述する必要がなくなり、メモリリークの発生も抑えられます。

◆ 安全性が高い

Go言語にはポインタはありますが、不正なメモリアクセスができないように、ポインタの演算はできません。また、型のチェックも厳密に行われるため、バグによってメモリが壊される危険性を回避しています。

◆ 継承がない

Go言語は、一般的なオブジェクト指向言語で用いられる「クラス」がありません。すべての変数は、いずれかの「型」に属します。この「型」にはメソッドを持たせることができますが、継承という概念はありません。継承による複雑さを排除した代わりに、「埋め込み」や「インタフェース」という仕組みが用意されています。

◆ 容易なエラーハンドリング

Go言語の関数は、「複数個の値」(多値)を戻り値として返せます。関数の実行結果と一緒にエラー情報を返すことができるので、エラーハンドリングが容易です。多くの言語で採用されている「try～catch」による例外処理機構ではエラーハンドリングのために不必要に多くのコードを記述しなければなりませんが、Go言語の例外処理機構は必要なところだけにエラーハンドリングを行えば済むようになっています。

🔗 バージョンごとの主要な変更点

Go言語は、バージョン1がリリースされてから3年経ち、現バージョンは1.4となりました。言語仕様としては、バージョン1ではほぼ完成されており、バージョンアップでは主に内部処理の効率化、より多くのプラットフォームへの対応、標準ライブラリの拡充が行われています。

バージョンアップによる主要な変更点は次のようになります。

◆ Go 1.1

Go 1.1での主な変更点は次のようになります。

- 定数0での除算は、コンパイル時に検出するようになった。
- メソッド値が導入された。
- 「return」文が必須となる条件が緩和され、たとえば無限ループでは「return」文が必要ないので、記述しなくてもコンパイルエラーとはならなくなった。
- 64ビット環境上でのint型、uint型のサイズが64ビットに変更された(本来の仕様通りになった)。
- 実際はシステムに依存するが、64ビット環境でのヒープサイズの上限が、数十ギガバイトまで拡張された。
- goコマンドのエラー表示が、より詳細に出力されるようになった。
- 30%～40%程度、パフォーマンスが向上した。

◆ Go 1.2

Go 1.2での主な変更点は次のようになります。

- nilポインタを介してメモリに不正アクセスできていた問題が修正された。
- スライス式にキャパシティが指定可能になった。
- 無限ループにおいてゴルーチンがスレッドを専有してしまう問題が改善された。
- スレッドの上限数(デフォルト10,000)が設けられた。これは「runtime/debug」パッケージの「SetMaxThreads」で変更可能。
- 「go doc」コマンドが廃止された。「godoc」コマンドで代用可能。

◆ Go 1.3

このバージョンでは言語的な変更はなく、スタックのメモリ管理方法、ガベージコレクション、ランタイムが改善され、実行速度が向上しています。

◆ Go 1.4

Go 1.4での主な変更点は次のようになります。

- 「range」式から返される値を使用しない場合でも、ブランク識別子に代入しなければならなかったが、その代入が不要となった。
- 仕様上、許されていないはずの二重ポインタの自動間接参照が、実際には可能だったのが仕様通り禁止された。
- Androidアプリの開発が可能となった。
- C言語またはアセンブラで記述されていたランタイムの大部分がGo言語で書き直された。

◆ 将来、盛り込まれる予定の機能

次のバージョンでは、ガベージコレクションの並行化による処理効率向上や、コンパイラやランタイムなどのツール類がGo言語で書き直されるそうです。



COLUMN

Go言語のAndroidアプリへの対応

Go 1.4からAndroidアプリの作成が可能になりました。Javaプログラムから呼び出されるライブラリとして作成する方法と、Go言語だけでアプリを作成する方法がありますが、後者はまだ機能的な制限があります。

また、現時点ではARMプロセッサにのみ対応しています。

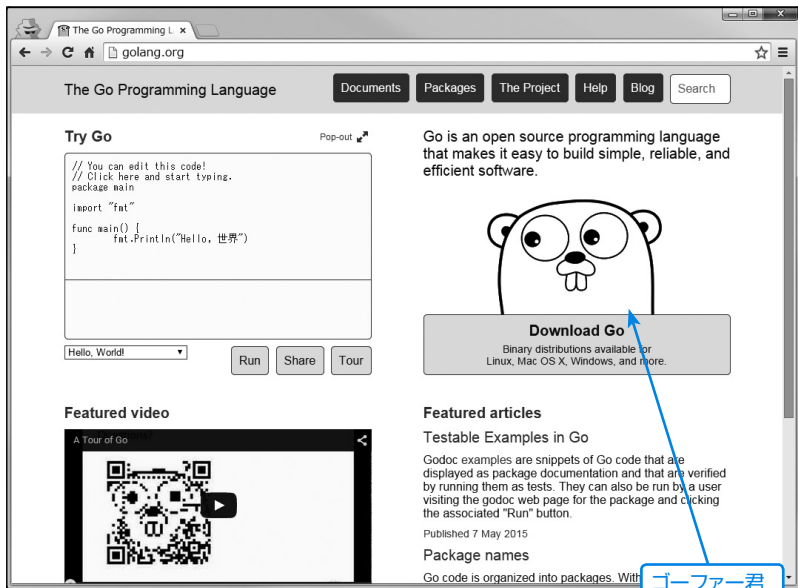


COLUMN

Go言語のマスコット

Go言語の公式サイト(<http://golang.org/>)のトップページでおどけているマスコットキャラの名前は「ゴーフー」君です。作者のルネ・フレンチの夫はGo言語の開発者の1人ロブ・パイクです。

このキャラは「Creative Commons Attribution 3.0」でライセンスされているため、著作権者の表示をすれば誰でも利用可能です。



ゴーフー君

Go言語は、「Microsoft Windows」「Mac OS X」「Linux」「FreeBSD」のいずれかの上で動作します。コンパイル言語であるため、Go言語のプログラム開発にはソースファイルを実行形式ファイルに変換するための「コンパイラ」が必要となります。

Go言語のコンパイラには、「gc」と「GCC」の2種類が用意されています。「gc」はGo言語の標準コンパイラなので、本書では「gc」の開発環境の構築について説明します。もう一方の「GCC」はC言語をはじめ、FortranやJavaなど、さまざまな言語をコンパイルできる多言語コンパイラです。これは古くからあるコンパイラで、GCC 4.7.0からGo言語のコンパイルもできるようになりました。

WebブラウザでGo言語プロジェクトのダウンロードページ(<https://golang.org/dl/>)を表示してください。そこで一覧表示されたリンクの「OS」欄と「Arch」欄を見て、各自の環境と一致する「Go言語ツール」をダウンロードしてください。

オペレーティングシステムが64ビット版であれば「32-bit」「64-bit」のどちらでも使用可能ですが、64ビット版Linuxで32ビット版のGo言語ツールをインストールした場合は、別途、ライブラリが必要になる場合があります。オペレーティングシステムが32ビット版の場合は「32-bit」と付いた方を使用してください。また、WindowsのIEで「.msi」のファイルをダウンロードすると警告が表示される場合がありますが、警告は無視してかまいません。

なお、それぞれの環境ごとのインストール方法は、次のページから解説します。

21

🌐 Microsoft Windowsへのインストール

Windows用のファイルには32ビット/64ビットの違いの他に、ファイルの拡張子が「.zip」と「.msi」の違いもあります。zip版の方は別途、環境変数の設定が必要になりますが、MSI版はインストーラによって簡単にインストールすることができ、環境変数も自動で設定されるので便利です。

◆ zip版

zip版の場合、まずダウンロードしたファイルを解凍ツールを使用して解凍します。解凍先のディレクトリ(フォルダ)は「C:¥Go」が推奨されています。

次に環境変数の設定を行います。次のように操作します。

- ① コントロールパネル内の「システムとセキュリティ」をクリックします。
- ② 「システム」をクリックします。
- ③ 左側の「システムの詳細設定」をクリックします。
- ④ 「環境変数(N)」ボタンをクリックします。
- ⑤ 「環境変数」ダイアログボックスが表示されるので、[システム環境変数(S)]にある「Path」を選択し、[編集(I)]ボタンをクリックします。
- ⑥ [変数値(V)]の末尾に「;」(セミコロン)を入力し、続けて「C:¥Go¥bin」と入力して、[OK]ボタンをクリックします。なお、「C:¥Go」以外に解凍した場合は、解凍したディレクトリに合わせてください。
- ⑦ 「環境変数」ダイアログボックスの[OK]ボタンをクリックして、設定を完了します。

なお、圧縮ファイルを「C:¥Go」に解凍した場合は不要ですが、それ以外のディレクトリに解凍した場合は環境変数「GOROOT」に、ドライブレターも含めて、その解凍先のディレクトリをフルパスで設定する必要があります。具体的には、次のように操作します。

- ① 上記の①～④と同様に操作し、「環境変数」ダイアログボックスを表示します。
- ② [システム環境変数(S)]にある[新規(W)]ボタンをクリックします。
- ③ [変数名(N)]に「GOROOT」と入力し、[変数値(V)]に解凍先のディレクトリをドライブレターも含めてフルパスで入力して、[OK]ボタンをクリックします。
- ④ 「環境変数」ダイアログボックスの[OK]ボタンをクリックします。

◆ MSI(Microsoft Windows Installer)版

MSI版の場合は、次のように操作してインストールします。なお、旧バージョンがインストールされている場合は、コントロールパネルで旧バージョンを削除してからインストールしてください。

- ① ダウンロードした「.msi」のファイルをダブルクリックして実行します。
- ② セキュリティの警告が表示された場合は、[実行(R)]ボタンをクリックします。
- ③ セットアップウィザードが起動するので、[Next]ボタンをクリックします。