Nimクイックスターターガイド

目次

[第１章 Nimのインストール 2](#_Toc143004367)

[第１節 Choosenimのインストール 2](#_Toc143004368)

[第２章 VSCodeでNimの開発環境をつくる 3](#_Toc143004369)

[第１節 なぜVisual Studio Codeを用いるのか 3](#_Toc143004370)

[第２節 Visual Studio Codeのインストール 3](#_Toc143004371)

[第３節 Visual Studio Codeの日本語化 3](#_Toc143004372)

[第４節 Nim開発用拡張機能のインストール 4](#_Toc143004373)

[第５節 デバッガのインストール 4](#_Toc143004374)

[第３章 Nimプログラムをパッケージ化 5](#_Toc143004375)

[第１節 パッケージのビルド 5](#_Toc143004376)

[第１項 Nimble Build 5](#_Toc143004377)

[第２項 Nimble Build（Debug） 5](#_Toc143004378)

[第２節 テスト 6](#_Toc143004379)

[第１項 Nimble Test 6](#_Toc143004380)

[第２項 Nimble Test(Debug) 6](#_Toc143004381)

[第３項 Launch(F5キー) 6](#_Toc143004382)

[第３節 パッケージのドキュメント生成 6](#_Toc143004383)

# Nimのインストール

この章では、Nim言語開発キットのインストール手順を解説します。

Nim単体でインストールすることも可能ですが、Nimのバージョン管理ツールであるchoosenimをインストールすると管理が楽になります。

## Choosenimのインストール

Choosenim(<https://github.com/dom96/choosenim.git>)

上記URLの「Release」から最新バージョンの

「choosenim-installer-X.X.X\_windows\_amd64-download-this-not-the-exe.zip」

をダウンロードしましょう。

ダウンロードしたzipファイルを解凍し、中にある

「runme.bat」

を実行してください。

以上でNimのインストールは完了です。

# VSCodeでNimの開発環境をつくる

この章では、Visual Studio Codeというコードエディター上でNim言語を用いた開発ができる環境の構築手順を解説します。

既にVisual Studio Codeを使っていて、不要な項目があるかもしれませんが、ご容赦ください。

## なぜVisual Studio Codeを用いるのか

有志がVisual Studio Code上でNimの開発ができるように拡張機能を公開してくださっており、構文チェック機能、サジェスト機能等が利用可能です。

またLLDBという高性能なデバッガも簡単に利用できるためです。

## Visual Studio Codeのインストール

VisualStudioCode（<https://azure.microsoft.com/ja-jp/products/visual-studio-code>）

上記URLの「Visual Studio Codeをダウンロードする」をクリックし、

画面遷移後に「Windows」ボタンをクリックし、

「VSCodeUserSetup-x64-X.X.X.exe」をダウンロードしましょう。

ダウンロードしたファイルを実行し、インストールしてください。

以上でVisual Studio Codeのインストールは完了です。

## Visual Studio Codeの日本語化

Visual Studio Codeはデフォルトで英語なので日本語化します。

ウィンドウ左側の「Extensions(拡張機能)」をクリックし、

検索欄に「Japanese」と入力すると

「Japanese Language Pack for Visual Studio Code」

という拡張機能がヒットするので

「Install」ボタンをクリックしてください。

Visual Studio Codeの再起動が促されるので、

再起動するとVisual Studio Codeの日本語化は完了です。

## Nim開発用拡張機能のインストール

Visual Studio CodeはデフォルトではNimに対応していませんので、

対応用の拡張機能をインストールします。

ウィンドウ左側の「拡張機能」をクリックし、

検索欄に「Nim」と入力すると

「Nim」

という拡張機能が沢山ヒットするので、たくさん利用されている、前回の更新日や最終更新日が一番最近のものを選び、

「Install」ボタンをクリックしてください。

以上でNim開発拡張機能のインストールは完了です。

## デバッガのインストール

Nimのデバッグ用にLLDBの拡張機能をインストールします。

ウィンドウ左側の「拡張機能」をクリックし、

検索欄に「LLDB」と入力すると

「CodeLLDB」

という拡張機能がヒットするので

「Install」ボタンをクリックしてください。

以上でNim開発拡張機能のインストールは完了です。

# Nimプログラムをパッケージ化

この章では、Nim言語を用いたプログラムをパッケージ化、テストする手順を解説します。

以下の操作は、「.vscode」フォルダ内に格納されている

「tasks.json」および「launch.json」で定義されています。

主に使用するのは、

* Nimble Build(Debug)
* Nimble Build
* Launch(F5キー)
* Nimble Test(Debug)
* Nimble Test
* Nimble Doc

になると思います。

## パッケージのビルド

プログラムをパッケージ化するためのタスクには、

「Nimble Build(Debug)」、「Nimble Build」

があります。

### Nimble Build

こちらはリリース用のパッケージングタスクです。

実行ファイルは「output」フォルダ内に配置されます。

### Nimble Build（Debug）

こちらはテスト用のパッケージングタスクです。

デバッグ用情報を付加した状態でビルドします。

実行ファイルは「tests」フォルダ内に配置されます。

「Launch(F5キー)」のタスク前にも実行されるので、こちら単体で実行する必要はありません。

## テスト

プログラムをテストするためのタスクには、

「Nimble Test」、「Nimble Test(Debug)」、「Launch(F5キー)」

があります。

### Nimble Test

こちらはtestsフォルダ内のNimファイルをテストするためのタスクです。

### Nimble Test(Debug)

こちらはtestsフォルダ内のNimファイルをテストするためのタスクです。

デバッグ用情報を付加した状態でテストします。（おそらくビルドも同時にする）

### Launch(F5キー)

こちらはtestsフォルダ内の実行ファイルをテストするためのタスクです。

LLDBを用いて状態を監視しながら、ブレークポイントが適用された状態でテストできます。

このタスクを行う前に自動で「Nimble Build(Debug)」タスクが実行されます。

## パッケージのドキュメント生成

プログラムの構造を自動でドキュメント化するタスクとして

「Nimble Doc」があります。

本来は、ライブラリのAPIをドキュメント化する機能であるため、もしかしたらうまく動かないかもしれません。