



# LINE API HANDBOOK

by  **L A E**  
LINE API Expert

**#LINE BOT**

**#LINE MESSAGING API**

**#LIFF**

**#LINE CLOVA**

**#LINE THINGS**

**#LINE PAY**

**#LINE LOGIN**

# LINE API HANDBOOK

LINE API Experts 著

技術書典7 (2019 年秋) 新刊  
2019 年 9 月 22 日 ver 1.0

#### ■免責

本書は情報の提供のみを目的としています。

本書の内容を実行・適用・運用したことで何が起こりようとも、それは実行・適用・運用した人自身の責任であり、著者や関係者はいかなる責任も負いません。

#### ■商標

本書に登場するシステム名や製品名は、関係各社の商標または登録商標です。

また本書では、™、®、©などのマークは省略しています。

# まえがき

本書を手にとっていただき、ありがとうございます。

本書は、LINE 株式会社が認定する LINE API Expert および LINE Developer Community の有志メンバーにより執筆、編集された書籍となります。さまざまな LINE API について、基本的な使い方から応用例まで幅広く解説しています。

## 本書で得られること

以下の LINE API に関する知識

- LINE Bot
- LINE Messaging API
- LIFF (LINE Front-end Framework)
- LINE Clova
- LINE Things
- LINE Pay
- LINE ログイン

## 本書のソースコードリポジトリ

本書に記載しているソースコードは、以下の GitHub リポジトリに公開しています。

### ■リポジトリ URL

<https://github.com/line-developer-community/line-api-handbook>

### 正誤表

本書に誤植があった場合、上記リポジトリ内に正誤表を掲載します。

### ■正誤表 URL

<https://github.com/line-developer-community/line-api-handbook/blob/master/errata/v1.0.md>

# 目次

<b>まえがき</b>	<b>i</b>
本書で得られること	i
本書のソースコードリポジトリ	i
<b>第 1 章   まずは各種 LINE の API を利用してみよう！</b>	<b>1</b>
1.1   はじめに	1
1.1.1   作るもの	1
1.1.2   必要なもの	2
1.2   Clova 編	2
1.2.1   スキルのチャンネル作成	2
1.2.2   開発環境	13
1.2.3   動作確認	16
1.3   Bot 編	17
1.3.1   Messaging API のチャンネル作成	17
1.3.2   動作確認	24
1.3.3   LIFP の作成	26
1.3.4   動作確認	28
1.3.5   LINE Pay の設定	31
1.3.6   動作確認	33
1.3.7   クイックリプライの動作確認	34
1.3.8   リッチメニューの作成	35
1.3.9   動作確認	40
1.4   おわりに	40
<b>第 2 章   C# で LINE ボット開発</b>	<b>41</b>
2.1   はじめに	41
2.2   C# SDK と Microsoft Bot Framework	41
2.3   Microsoft Azure	42
2.3.1   体験版の申し込み方法	42
2.4   オウム返しボットの作成	44
2.4.1   Microsoft Bot Framework リソースの作成	44
2.4.2   LINE と繋いでみる	49
2.5   ステート管理を利用した会話機能の追加	52
2.5.1   自動生成されたコードの取得	52
2.5.2   ステート管理機能の追加	53
2.5.3   ダイアログの追加	57
2.5.4   コードの公開	60
2.5.5   検証動作の変更	63
2.6   割り込み処理	65

2.7	LINE 固有機能の利用	67
2.7.1	ChannelData	67
2.7.2	スタンプ	67
2.7.3	ボタンテンプレート	69
2.7.4	Flex Message	71
2.8	自然言語解析	75
2.8.1	LUIS の準備	76
2.8.2	LUIS の組み込み	82
2.9	おわりに	85
<b>第 3 章</b>	<b>Blazor と Azure Functions で作る LIFF アプリケーション</b>	<b>86</b>
3.1	はじめに	86
3.2	対象者	86
3.3	作成するアプリケーションの概要	87
3.4	開発環境	88
3.4.1	サンプルリポジトリをフォーク	89
3.5	Blazor アプリを GitHub ページに公開する	89
3.5.1	GitHub の設定	89
3.5.2	TodoBot.Client プロジェクトの確認	90
3.6	LIFF アプリの作成	92
3.6.1	LIFF のクライアントライブラリを使用する	92
3.6.2	GitHub ページの URL を LIFF アプリとして登録する	93
3.7	API サーバーを Azure Functions で作る	96
3.7.1	Todo Model の作成	96
3.7.2	Todo API の仕様	97
3.7.3	Azure Functions プロジェクトの作成	97
3.7.4	Function クラスの定義	98
3.7.5	データアクセス処理の実装	99
3.7.6	Azure Function のデプロイ	104
3.8	Blazor (client-side) アプリの実装	107
3.8.1	LIFF クライアント SDK の初期化と利用	107
3.8.2	Todo API を利用した Todo データへのアクセス	108
3.8.3	Todo 管理画面の実装	111
3.8.4	Todo 作成画面	112
3.9	おわりに	116
<b>第 4 章</b>	<b>LIFF 活用事例と開発 Tips の紹介</b>	<b>117</b>
4.1	はじめに	117
4.1.1	対象	117
4.1.2	LIFF とは	117
4.2	LIFF 活用事例	118

4.2.1	AJINOMOTO	118
4.2.2	EPARK グルメ	120
4.2.3	福岡市	122
4.2.4	ロクシタン	124
<b>4.3</b>	<b>開発 Tips</b>	<b>126</b>
4.3.1	vConsole	126
4.3.2	liff-cli の活用	129
4.3.3	キャッシュ問題への対策	130
<b>第 5 章</b>	<b>C#と Azure で Clova スキル開発～裏技を添えて～</b>	<b>132</b>
<b>5.1</b>	<b>はじめに</b>	<b>132</b>
5.1.1	① Clova スキル開発のための C# 向け新 SDK の使い方	132
5.1.2	② 裏技「無音無限ループ」を使った応用的なスキル開発方法	132
<b>5.2</b>	<b>Clova スキルの C# での実装</b>	<b>133</b>
5.2.1	前提となる環境	133
5.2.2	CEK C# SDK + Azure Functions での開発手順	133
5.2.3	ASP.NET Core での DI	138
5.2.4	Azure Functions での DI	140
5.2.5	ClovaBase を継承したクラス内でロガーなどを使いたい場合	142
5.2.6	Durable Functions を使用したスキルの拡張	145
<b>5.3</b>	<b>無音無限ループによるスキルの拡張</b>	<b>151</b>
5.3.1	無音無限ループとは	151
5.3.2	実用例の紹介	151
5.3.3	無音無限ループの実装方法	153
5.3.4	ストア公開	158
<b>5.4</b>	<b>おわりに</b>	<b>161</b>
<b>第 6 章</b>	<b>TypeScript × Clova スキル開発</b>	<b>162</b>
<b>6.1</b>	<b>はじめに</b>	<b>162</b>
<b>6.2</b>	<b>TypeScript について</b>	<b>163</b>
6.2.1	TypeScript とは	163
6.2.2	TypeScript のメリット	163
6.2.3	型	163
6.2.4	入力補完	167
6.2.5	最新の ECMAScript 構文を使える	167
6.2.6	JavaScript 完全互換	168
6.2.7	読んでおいたほうがいいもの	168
<b>6.3</b>	<b>JavaScript サンプルスキルの構築</b>	<b>168</b>
6.3.1	サンプルスキル	168
6.3.2	スキル作成	169
6.3.3	対話モデルの構築	169



---

6.3.4	Visual Studio Code の起動	170
6.3.5	ngrok のインストールと実行	170
6.3.6	ngrok URL のコピー	170
6.3.7	エンドポイントの設定	171
6.3.8	npm パッケージのインストールと Node 実行	171
6.3.9	テスト実行	172
6.3.10	ソースコードについて	172
<b>6.4</b>	<b>TypeScript 環境構築</b>	<b>173</b>
6.4.1	ディレクトリの作成とソース移動	173
6.4.2	typescript と typesync のインストール	173
6.4.3	型定義のインストール	173
6.4.4	型定義と@types について	174
6.4.5	TypeScript の初期化	174
6.4.6	tsconfig.json の編集	174
6.4.7	tsconfig.json について	175
6.4.8	index.ts の確認	176
6.4.9	strict の無効化	176
6.4.10	コンパイル	177
6.4.11	Node 実行とテスト	177
6.4.12	strict を戻す	177
<b>6.5</b>	<b>JavaScript から TypeScript への移植</b>	<b>178</b>
6.5.1	require を import に書き換え	178
6.5.2	clovaSkillHandler のエラー	178
6.5.3	ソースの修正	179
6.5.4	残りのエラーを確認	179
6.5.5	diceCount の型	180
6.5.6	responseHelper の型	180
6.5.7	正解の型	180
6.5.8	Context 型アノテーションをつける	181
6.5.9	Context 型を探す	181
6.5.10	Context 型のインポート	182
6.5.11	入力補完を試してみよう	182
6.5.12	SpeechBuilder でも試してみよう	183
6.5.13	すべての responseHelper の修正	183
6.5.14	diceCount のエラー	184
6.5.15	スロットの値は文字列型	184
6.5.16	型エラーの対処	185
6.5.17	throwResult の型	185
6.5.18	型を切り出して宣言	186
6.5.19	ThrowResult 型アノテーションをつける	186
6.5.20	interface での型宣言	187

6.5.21	コンパイル & Node 実行 & テスト	187
6.5.22	【実践】ガイドインテントを実装してみよう	187
6.6	関連ツールについて	188
6.6.1	Linters	188
6.6.2	tsc -w	190
6.6.3	nodemon	190
6.6.4	開発 Tips	190
6.6.5	VSCoDe でのデバッグ	191
6.7	おわりに	194
<b>第 7 章</b>	<b>AWS Lambda を使って LINE ログインを実装してみよう！</b>	<b>195</b>
7.1	対象読者	195
7.2	前提条件	195
7.3	はじめに	195
7.3.1	LINE ログインとは	195
7.3.2	LINE ログインで取得できるもの	196
7.3.3	Social API V2.1 で取得できるもの	196
7.3.4	アーキテクチャ	196
7.3.5	全体の流れ	196
7.3.6	認証フロー	196
7.4	LINE ログインのチャンネルを作成しよう！	197
7.5	フロントエンドの設定をしよう！	199
7.5.1	S3 バケットを作成しよう！	199
7.5.2	S3 バケットに Web ページのファイルをアップロードしよう！	202
7.6	バックエンドの設定をしよう	205
7.6.1	Lambda 関数を作成しよう！	205
7.6.2	モジュールをインストールしよう！	206
7.6.3	Lambda にコードをアップロードしよう！	207
7.6.4	API Gateway の設定をしよう！	208
7.6.5	Lambda の環境変数を設定しよう！	217
7.7	CallBack URL を登録しよう！	219
7.8	login.js の修正をしよう！	221
7.8.1	ローカルでファイルを書き換えよう！	221
7.8.2	S3 にアップロードしよう！	221
7.9	動かしてみよう！	224
7.10	認証フロー解説	225
7.10.1	認証フロー図の①	226
7.10.2	認証フロー図の②	227
7.10.3	認証フロー図の③	227
7.10.4	認証フロー図の④	228

7.11	コード解説	229
7.11.1	バックエンド	229
7.11.2	フロント	234
7.12	おまけ	234
7.12.1	LINE ログインのチャンネルを公開しよう！	234
7.12.2	メールアドレスを取得してみよう！	236
7.13	おわりに	241
<b>第 8 章</b>	<b>LINE Things の基礎と LINE Beacon との違いなどをおさらい</b>	<b>242</b>
8.1	はじめに	242
8.1.1	本章の対象者	242
8.2	LINE Things とは	242
8.2.1	LINE が IoT 領域に切り込むことで解決する課題	243
8.2.2	LINE アプリや LINE BOT から IoT 機器の制御	244
8.3	LINE Things の特徴が生きそうなユースケースイメージ	244
8.3.1	電波が届かなく、電源の共有が厳しい場所	244
8.3.2	データを即時反映させたいケース	244
8.3.3	多くの人が訪れる場所	244
8.4	LINE Beacon	245
8.4.1	LINE Beacon と LINE Things の違い	245
8.4.2	LINE Simple Beacon	245
8.4.3	どちらを選択するべきか	246
8.5	LIFF 版と Message 版の二種類の LINE Things	246
8.5.1	LINE Things LIFF (LIFF 版)	246
8.5.2	LINE Things Message (Message 版)	247
8.6	LINE の API 群と LINE Things や LINE Beacon の立ち位置	248
8.6.1	LINE Beacon や LINE Things の比較	248
8.7	LINE Things Message の無限連続通知	249
8.8	つながらないときの対処	250
8.9	まとめ	250
<b>第 9 章</b>	<b>ハード初心者が LINE Things で準備すること</b>	<b>252</b>
9.1	はじめに	252
9.2	購入する基板を選ぼう	252
9.2.1	LINE Things Starter で提供されている対応基板	252
9.2.2	はじめて購入すべきハードはどれなのか	253
9.2.3	何がいいか選べない	254
9.3	購入はどこですればいいのか	257
9.4	余談 1：用語	258
9.5	余談 2：開発するためのサーバ環境	260

9.5.1	デバックコンソールを使いたい	260
9.5.2	HTML をキャッシュさせないようにしたい	260
9.6	おわりに	261
<b>第 10 章</b>	<b>ちょっとだけセキュアな認証システムを LINE Things でやってみた</b>	<b>263</b>
10.1	仕組み	263
10.2	準備	263
10.3	電話番号の発行	264
10.3.1	リソースを作成する	264
10.3.2	電話番号を取得する	267
10.3.3	発行した電話番号を確認する	269
10.4	問い合わせフロー作成	270
10.4.1	問い合わせフローを作成する	270
10.4.2	ID をメモしておく	278
10.5	AWS Lambda 関数を作成する	279
10.5.1	Lambda 関数を作成する	279
10.5.2	Amazon Connect アクセス権限を追加する	280
10.5.3	プログラムを書き込む	283
10.5.4	環境変数を設定する	285
10.5.5	API Gateway を設定する	286
10.6	LINE Bot を作ろう！	287
10.6.1	チャンネルの作成	287
10.6.2	アクセストークンを発行する	290
10.6.3	Webhook を設定	291
10.6.4	LINE Bot と友だちになる	292
10.6.5	LIFF の設定をする	292
10.7	LINE Things 自動通信機能の設定	294
10.7.1	サービス UUID を発行する	294
10.7.2	LINE アプリの LINE Things を有効化	296
10.8	LIFF アプリを構築する	297
10.8.1	Vue.js でプロジェクトを作成する	297
10.8.2	Vuetify を追加	298
10.8.3	LIFF の SDK を入れる	299
10.8.4	認証コード送信画面を作成する	299
10.8.5	外部公開設定を行う	302
10.8.6	実行する	302
10.9	外部公開をする	303
10.9.1	serveo.net に公開する	303
10.9.2	LIFF の編集をする	303
10.10	M5Stack に書き込む	304
10.10.1	M5Stack にプログラムを書き込む	304

10.10.2	動作確認する	311
<b>第 11 章</b>	<b>LINE Pay 決済と LINE Things に対応したドリンクバーを開発する</b>	<b>313</b>
11.1	はじめに	313
11.2	LINE Things Drink Bar とは	314
11.3	LINE Things Drink Bar での購入・決済の流れ	315
11.4	システム構成	316
11.4.1	サーバサイドの構成	317
11.4.2	フロントサイドの構成	317
11.4.3	ハードウェアの構成	317
11.4.4	ソースコード	318
11.5	LINE Pay API について	319
11.5.1	LINE Pay API を使った決済の流れ	319
11.5.2	決済予約	319
11.5.3	ユーザーによる承認	319
11.5.4	決済実行	319
11.6	LINE Pay の導入	319
11.6.1	LINE Pay の加盟店申請	320
11.6.2	LINE Pay Sandbox の申請と設定	322
11.7	開発	323
11.7.1	サーバ (Heroku) の準備	323
11.7.2	コンテナの準備	324
11.7.3	ドリンクバー画面	325
11.7.4	ドリンク一覧の表示	326
11.7.5	注文情報の登録	329
11.7.6	決済予約の実行	332
11.7.7	決済実行処理の実行	336
11.8	Heroku へのデプロイ	339
11.8.1	環境変数の設定	339
11.8.2	コンテナのデプロイ	340
11.9	おわりに	341
11.10	関連リンク	341
<b>著者紹介</b>		<b>342</b>
	LINE API Expert	344
	LINE Developer Community	344