

# Azure Functions を試してみた

Azure AzureFunctions

この記事は最終更新日から1年以上が経過しています。

## Azure Functions とは？

---

Microsoftのクラウドサービスのこと。

サーバーレスと呼ばれているものですが、別にサーバーが無い、サーバーが要らないということじゃなくて、サーバーの存在をほとんど意識せずにWebサービスを作ることが出来るものです。

WebAPIをととても簡単に作ることが出来るフレームワークと言った方が分かり易いでしょうか。

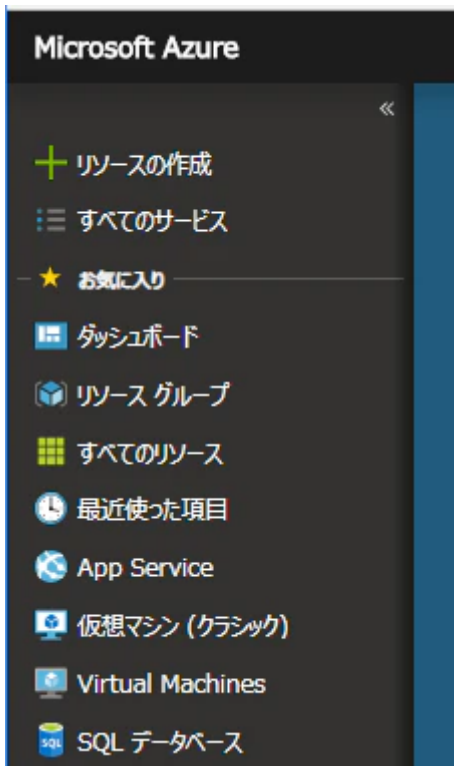
Microsoft Azure

## 試しにサービスを作ってみる

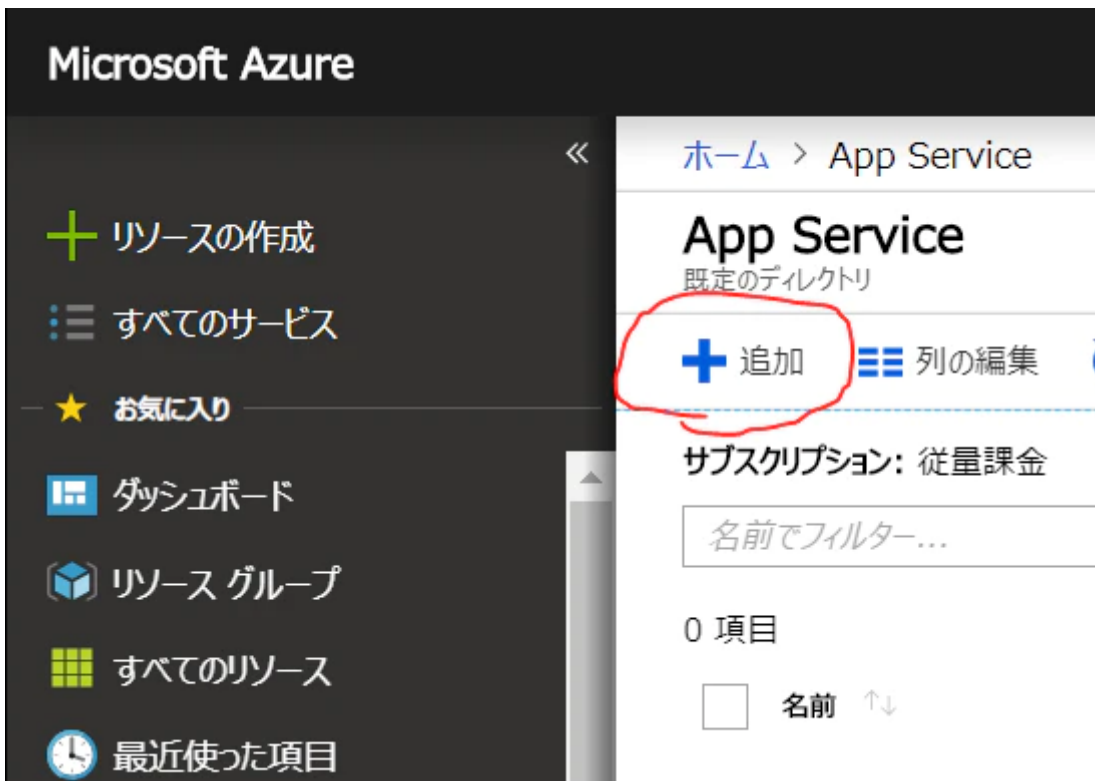
---

# ポータルから App Service を作成する

ポータルメニューの「すべてのサービス」もしくは「お気に入り」から「App Service」を選択します。

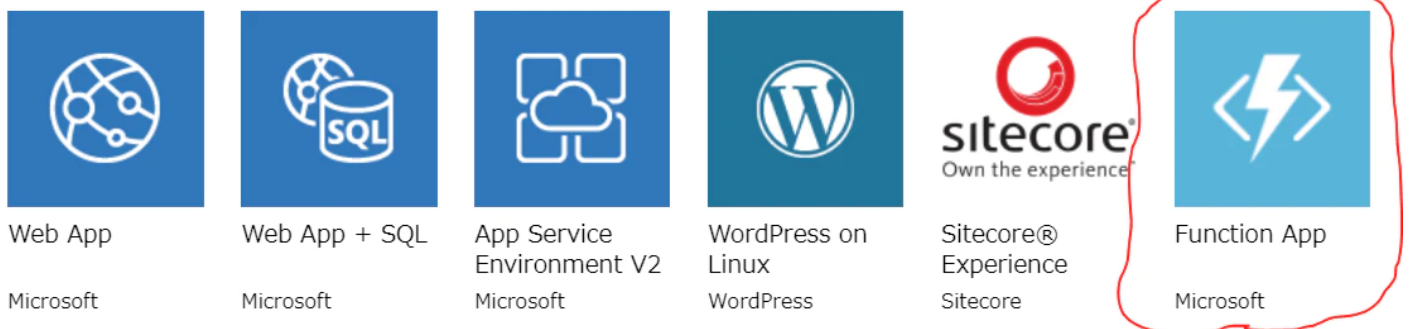


「追加」ボタンを押します。

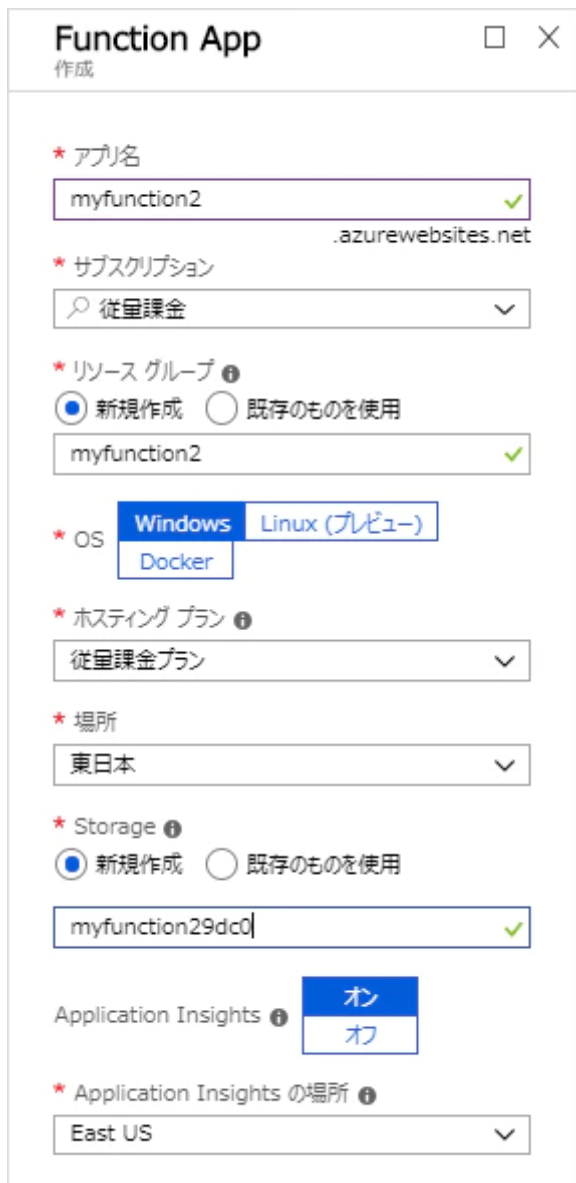


## Function App を選択し、サービスを「作成」します。

### Web Apps



## 作成するアプリ名を設定



The screenshot shows the 'Function App' creation window with the following settings:

- アプリ名**: myfunction2 (with a green checkmark and .azurewebsites.net domain)
- サブスクリプション**: 従量課金 (with a search icon and dropdown arrow)
- リソース グループ**:
  - ☒ 新規作成 ☐ 既存のものを使用
  - myfunction2 (with a green checkmark)
- OS**: Windows (selected), Linux (プレビュー), Docker
- ホスティング プラン**: 従量課金プラン (with a dropdown arrow)
- 場所**: 東日本 (with a dropdown arrow)
- Storage**:
  - ☒ 新規作成 ☐ 既存のものを使用
  - myfunction29dc0 (with a green checkmark)
- Application Insights**: オン (selected), オフ
- Application Insights の場所**: East US (with a dropdown arrow)

アプリ名：他の利用者と被らないアプリ名(ホスト名)を付けます。

サブスクリプション：無料の人は無料のアカウントを選択します。


リソースグループ：1つのアプリは複数のサービスから構成されるので、それらのグループ名です。

場所：日本であれば「東日本」か「西日本」を選択すれば良いです。Azureのデータセンターが日本に2か所あります。

Storage：Azure FunctionsがStorageを使用しますので、そのアカウントも併せて作ります。

Application Insights：アプリの動作状況を監視する仕組みです。

作成したアプリはAzureメニューの「すべてのリソース」や「リソースグループ」から辿ることが出来ます。

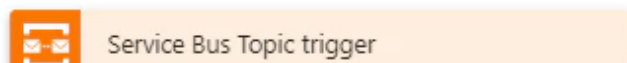
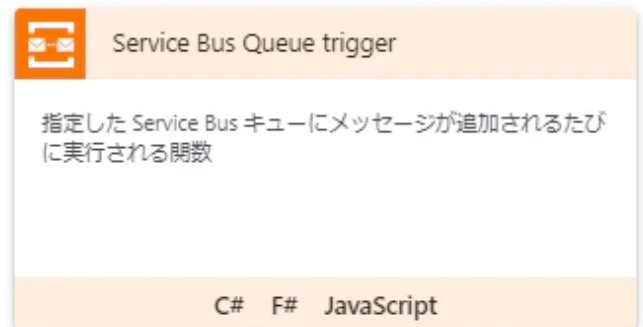
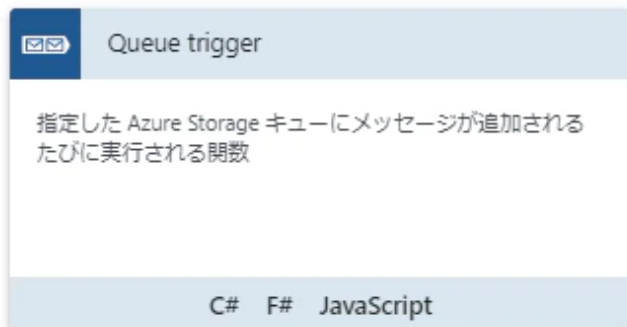
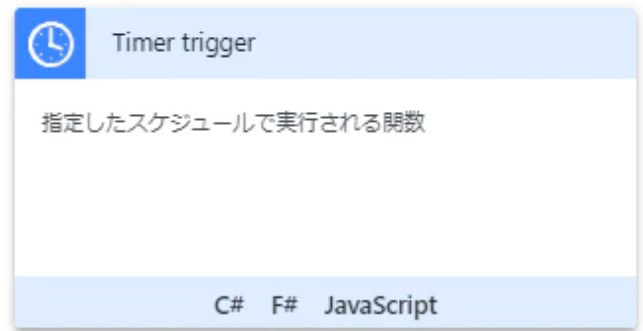
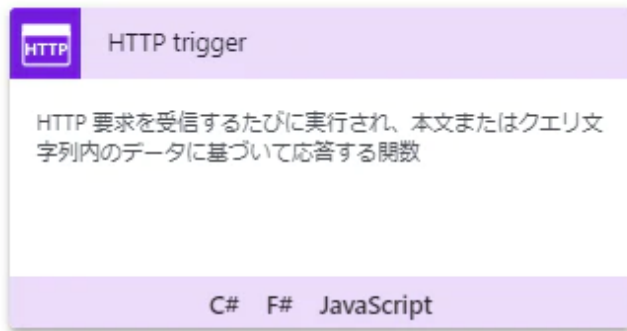
<input type="checkbox"/> 名前 ↑↓	種類 ↑↓	リソース グループ ↑↓	場所 ↑↓
<input type="checkbox"/>  JapanWestPlan	App Service プラン	myfunction2	西日本
<input type="checkbox"/>  myfunction2	Application Insights	myfunction2	米国東部
<input type="checkbox"/>  myfunction2	App Service	myfunction2	西日本
<input type="checkbox"/>  myfunction28d3a	ストレージ アカウント	myfunction2	西日本

「種類」が「App Service」となっているところで実装します。

## 関数を実装する



「関数」から「新しい機能」を選択します。



色んなパターンのテンプレートが用意されています。

HTTPのリクエストに対して応答を返す標準的なものや、POSTされたデータをテーブルに保存するもの、さらにはタイマーで起動するものなど。様々な入力パターンをトリガーとして、様々な出力パターンの組み合わせが選択出来ます。後からカスタマイズも可能です。これらを連携させてアプリを構築していきます。

# HTTP triggerのテンプレートから作成してみる

HTTP要求に対して応答を返す機能です。

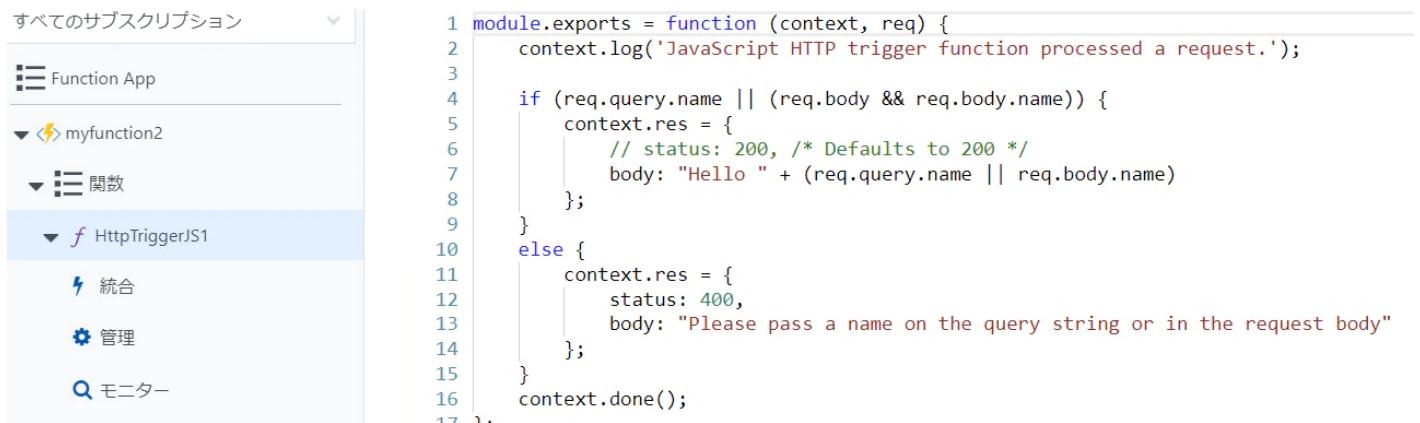
標準で、C#, F# JavaScriptが開発言語として選べます。

オプションでPhthonやTypeScriptなども可能なようです。

ここでは、言語をJavaScriptで作成してみます。

「承認レベル」はFunctionにしておきます。

以下がテンプレートとして生成されます。



The screenshot displays the Azure Functions portal interface. On the left, a sidebar shows the navigation menu with 'Function App' selected, followed by 'myfunction2' and '関数' (Functions). Under '関数', 'HttpTriggerJS1' is highlighted. Below this, there are links for '統合' (Integrate), '管理' (Manage), and 'モニター' (Monitor). The main area shows the JavaScript code for the 'HttpTriggerJS1' function template. The code is as follows:

```
1 module.exports = function (context, req) {
2     context.log('JavaScript HTTP trigger function processed a request.');
```

```
3
4     if (req.query.name || (req.body && req.body.name)) {
5         context.res = {
6             // status: 200, /* Defaults to 200 */
7             body: "Hello " + (req.query.name || req.body.name)
8         };
9     }
10    else {
11        context.res = {
12            status: 400,
13            body: "Please pass a name on the query string or in the request body"
14        };
15    }
16    context.done();
17 }
```

既に、「実行」ボタンを押すと「ログ」が出力されて「出力」欄にHTTP応答の内容が出力される状態です。

実装方法など、ドキュメントは以下に有ります。

[Functionsのドキュメント](#)



言語毎のドキュメントは以下です。

サポートされている言語

## 入出力をカスタマイズする

左側メニューで「統合」を選びます。

トリガー ⓘ	入力 ⓘ	出力 ⓘ
HTTP (req)	+ 新しい入力	HTTP (res)
		+ 新しい出力

---

HTTP trigger ✕ 削除

許可されている HTTP メソッド ⓘ

All methods ▼

モード ⓘ

Standard ▼

「トリガー」：この関数が動くきっかけです。この場合はHTTPのリクエストが有った場合です。

「入力」：この関数で参照したいストレージデータです。

「出力」：処理結果の出力先です。この場合はHTTP応答です。テーブルやキューなどを出力先に追加することも出来ます。

それぞれ設定されている欄を選択することで、詳細を設定します。

## 最後に

---

Azure Functions は、WebAPI、Webサービスをとても簡単に作ることができます。

HTTPも、フレームワーク側で要求を処理してくれたものを変数で受け取って、応答も変数にセットするだけです。

開発者はその間の処理を書くだけで良いです。

ストレージへの入出力についても、バインドされた変数への読み書きだけなので非常に簡単です。

アプリに大量のリクエストが来た時もサーバー台数を増やすなんてことも考えなくていいです。Azureが自動的にやってくれます。

サーバーのハード障害も心配ありません。自動的に切り替わります。

ユーザはもうサーバーの面倒を見る必要は無く、ただアプリロジックを書けばいいだけなのです。