

# Newman Run Report

## 仕様

## 動作

実行すると `newman` フォルダが作成され、結果が保存されていきます

`newman` ディレクトリにはhtml、json形式の結果

### Newman Report

CollectionPotato\_Run

TimeFri Feb 01 2019 15:44:34 GMT+0900 (GMT+09:00)

Exported withNewman v4.3.1

	Total	Failed
Iterations	1	0
Requests	63	0
Prerequisite Scripts	63	0
Test Scripts	108	0
Assertions	142	0

Total run duration6.5s

Total data received56KB (approx)

Average response time70ms

Total Failures0

### Requests

アクセストークン取得	
Method	POST
URL	https://admin-api-dev.██████████.net/access_tokens
Mean time per request	261ms
Mean size per request	269B

`newman\log` ディレクトリにはcsv形式の結果とlog形式のスク립トログ

Time	TotalRunDuration	ResponseAverage	PassTests	FailTests	PassTestScripts	FailTestScripts
2019/2/1 15:36	6.499	69.38	63	0	108	0
2019/2/1 15:36	6.363	69.48	63	0	108	0
2019/2/1 15:37	6.247	68.51	63	0	108	0
2019/2/1 15:37	6.436	71.7	63	0	108	0
2019/2/1 15:37	6.717	73.56	63	0	108	0
2019/2/1 15:38	6.489	71.6	63	0	108	0
2019/2/1 15:38	6.37	69.49	63	0	108	0
2019/2/1 15:39	6.667	73.21	63	0	108	0
2019/2/1 15:39	6.46	69.52	63	0	108	0
2019/2/1 15:39	6.29	67.54	63	0	108	0
2019/2/1 15:40	6.323	69.35	63	0	108	0
2019/2/1 15:40	6.653	71.21	63	0	108	0

```
newman-run-report.log - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
Host Application: C:\Windows\System32\WindowsPowerShell\v1.0\powershell
Process ID: 3240
PSVersion: 5.1.14409.1018
PSEdition: Desktop
PSCompatibleVersions: 1.0, 2.0, 3.0, 4.0, 5.0, 5.1.14409.1018
BuildVersion: 10.0.14409.1018
CLRVersion: 4.0.30319.42000
WSManStackVersion: 3.0
PSRemotingProtocolVersion: 2.3
SerializationVersion: 1.1.0.1
*****
Transcript started, output file is .\newman\log\newman-run-report.log

-----02/12/2019 12:22:11: テスト開始-----

@{iterations=; items=; scripts=; prerequisites=; requests=; tests=; asse
Total run duration : 12.261s
responseAverage    : 156.79ms
Fail アクセストークン取得: Response time is less than 200ms
    Message expected 398 to be below 200
Fail アカウント登録: Response time is less than 200ms
    Message expected 277 to be below 200
Fail アカウント更新: Response time is less than 200ms
    Message expected 235 to be below 200
Fail 組織全件取得: TypeError
    Message Cannot read property 'name' of undefined
Fail 組織登録: Response time is less than 200ms
    Message expected 321 to be below 200
Fail デバイス全件取得: Response time is less than 200ms
    Message expected 331 to be below 200
Fail ibeacon更新: Response time is less than 200ms
    Message expected 201 to be below 200
-----02/12/2019 12:22:32: 全テスト終了-----
```

さらに、任意のSlackチャンネルに結果の通知が送られます



**Newman-Run-Report** アプリ 14:24

---02/12/2019 14:24:05開始テストレポート---

全テスト終了



エラーなし



**Newman-Run-Report** アプリ 12:59

---02/08/2019 12:59:11開始テストレポート---

全テスト終了

リクエスト:0回失敗, テストスクリプト:3回失敗



アクセストークン取得: Response time is less than 200ms

Message expected 426 to be below 200



組織全件取得: TypeError

Message Cannot read property 'name' of undefined

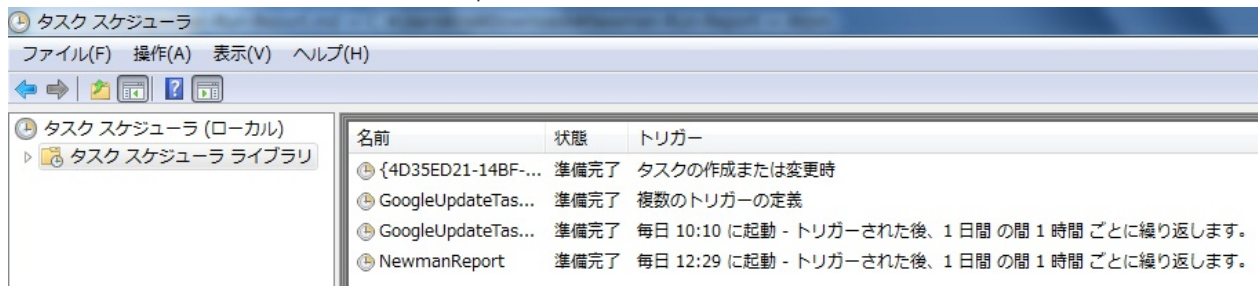


デバイス全件取得: Response time is less than 200ms

Message expected 239 to be below 200

file:///C:/Users/irie/Downloads/newman/newman-run-report-2019-02-08-03-59-19-829-0.html

また、WindowsタスクスケジューラにNewman Run Reportを登録するとテストを好みのタイミングかつ、バックグラウンドで行うことができます



## 必要条件

- Windows PowerShell 3.0以上
- Node.js via package manager v6以上
- Newman v4

## 使用方法

### 流れ

1. Postmanをインストール
2. Postmanにて Collections を作成し、json形式で出力
3. Newmanをインストール
4. Slack Incoming Webhook URL を取得
5. Newman-Run-Report に入っている Report.ps1 を編集し、同じディレクトリに#2で作成したjsonファイルを配置
6. (オプション) WindowsタスクスケジューラにNewman Run Reportを登録する

## 1.Postmanをインストール

まだインストールしていない場合は[こちら](#)からインストールしてください

## 2.Postman

### Collectionsを作成

1. 画面左側の + を押す



2. 名前を適当についたら右下の Create で作成

CREATE A NEW COLLECTION

Name

Collection Name

Description Authorization Pre-request Scripts Tests Variables

This description will show in your collection's documentation, along with the descriptions of its folders and requests.

Adding a description makes your docs better

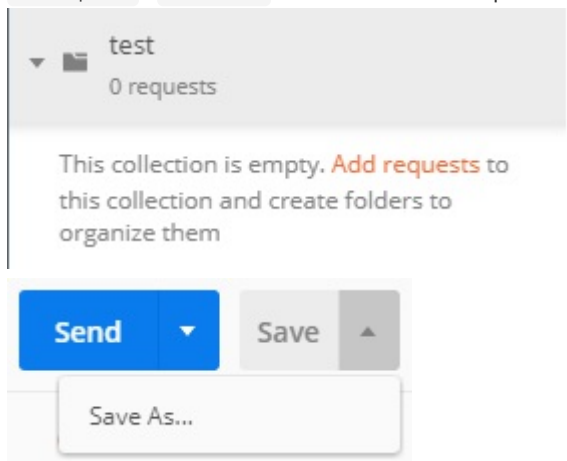
Descriptions support Markdown

Cancel Create

他の項目は後から変更できます

## Requestsを作成

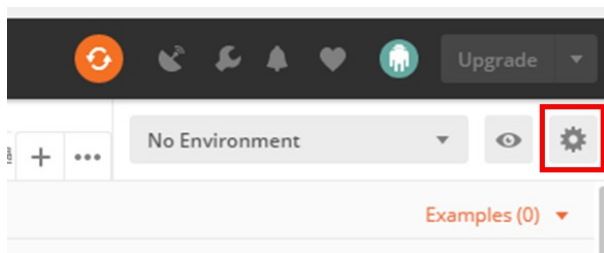
Add requests や Save As... などからCollectionsにRequestsを追加



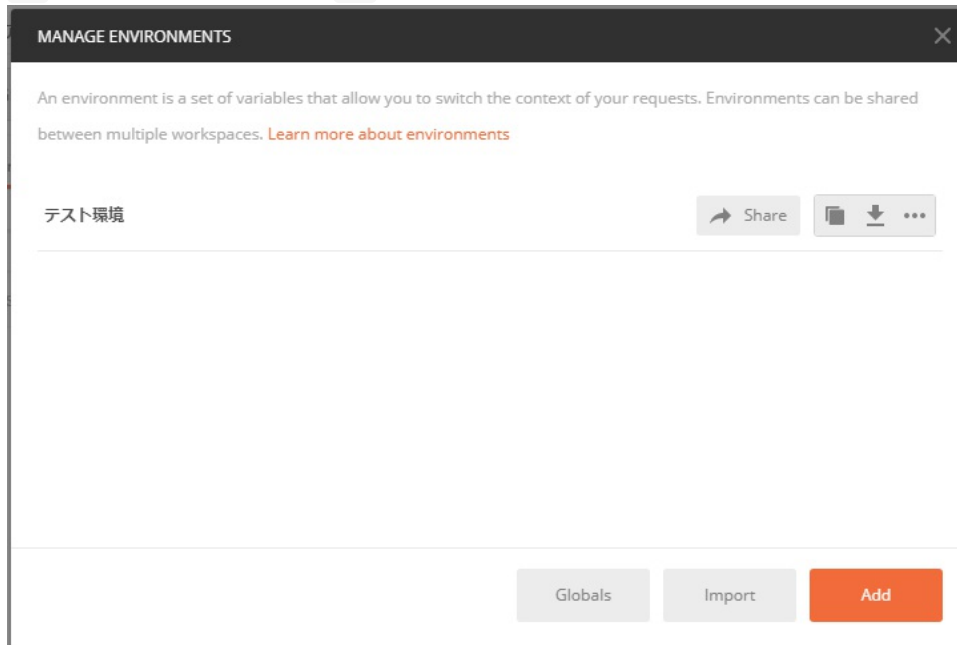
お好みでRequests内容を編集してください

## トークン設定

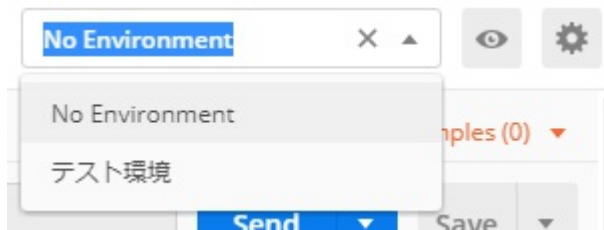
1. 画面右上の 歯車 を押す



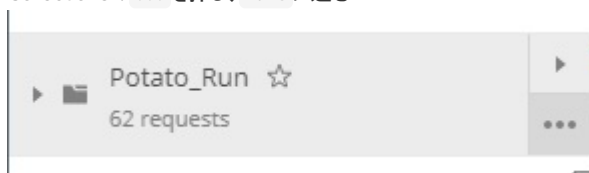
2. Add を押し、適当な名前をつけて再び Add



3. 画面右上の No Environment を押し、作成した環境を選択



4. Collectionsの... を押し、Edit に進む



5. Authorization タブのTYPEを任意のものに変更し、Tokenを {{token}} とする

EDIT COLLECTION

Name

testCollection

Description

Authorization

Pre-request Scripts

Tests

Variables

This authorization method will be used for every request in this collection. You can override this by specifying one in the request.

TYPE

Bearer Token

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. [Learn more about variables](#)

Token

{{token}}

Cancel

Update

## 6. Bodyにtokenが返るRequestsの場合

▶ アクセストークン取得

POST

https://admin-api-dev.beaconnect-plus.net/access\_tokens

Params

Authorization

Headers (1)

Body

Pre-request Script

Tests

none

form-data

x-www-form-urlencoded

raw

binary

JSON (application/json)

1 {

2 "organization\_id": 192701,

3 "email": "shuto.kawashima@asia-quest.jp",

4 "password": "Kawashima12321"

5 }

Body

Cookies

Headers (9)

Test Results (1/2)

Status: 201 Created

Time: 421 ms

Size: 557 B

Save

Download

Pretty

Raw

Preview

Auto

🔍

1 {

"id": "jFe365", "organization\_id": "192701", "email": "shuto.kawashima@asia-quest.jp", "name": "川嶋", "note": "川嶋管理者アカウント", "end\_time": "2030-11-29T15:52:40.592247+09:00", "role": "ADMIN", "tutorial\_flg": "false", "token": "347a6d23-a665-46e3-a4ce-d4359eb588bd" }

Tests に以下の記述をするとPostman内の環境変数 token にBody内のtokenが代入されます

```
var data = JSON.parse(responseBody);
postman.setEnvironmentVariable("token", data.token);
```

▶ アクセストークン取得

POST    https://admin-api-dev.beaconnect-plus.net/access\_tokens

Params    Authorization    Headers (1)    Body ●    Pre-request Script    Tests ●

```
1 var data = JSON.parse(responseBody);
2 tests["トークン取得" + data.token] = true;
3
4 postman.setEnvironmentVariable("token", data.token);
```

## Tests設定

もし アカウント登録 → アカウント削除(更新) にランダムなアカウントIDを用いていた場合、一連の流れでテストを行うことができません。そこで、先ほどのトークン設定で用いた環境変数に代入を利用します。

アカウント登録のリクエストを送ると以下のようなBodyが返ってくるので

Body    Cookies    Headers (10)    Test Results (1/1)

Pretty    Raw    Preview    JSON ▼    ≡

```
1 {
2   "id": "vM7GrD",
3   "organization_id": "192701",
4   "email": "account8@example.com",
5   "name": "アカウント 8",
6   "note": "アカウント 8 の備考です",
7   "end_time": "2019-12-28T12:53:51.364189+09:00",
8   "role": "USER",
9   "tutorial_flg": "false",
10  "favorite_smartcube": []
11 }
```

Testsタブに以下の記述を入力するとBodyに返ってきた連想配列内の `id` がPostmanの環境変数に `id` として登録されます

```
var data = JSON.parse(responseBody);
postman.setEnvironmentVariable("id", data.id);
```

POST    https://admin-api-dev.beaconnect-plus.net/accounts

Params    Authorization    Headers (3)    Body ●    Pre-request Script    Tests ●

```
1 var data = JSON.parse(responseBody);
2 postman.setEnvironmentVariable("id", data.id);
3
```

環境画面を見るとVARIABLEに `id` が追加されていることが確認できます



MANAGE ENVIRONMENTS

Environment Name

テスト環境

	VARIABLE	INITIAL VALUE ⓘ	CURRENT VALUE ⓘ	...	Persist All	Reset All
<input checked="" type="checkbox"/>	id		DEmkXD			

次にアカウント更新のURLを次のように変更します

▶ アカウント更新

PUT

https://admin-api-dev.beaconnect-plus.net/accounts/{{id}}

Postmanでは{{環境変数名}}と入力すれば基本的にどこでも環境変数に置き換えられます。そのため アカウント作成 のTestsで環境変数 id に代入された値がURLに代入され、ランダムなアカウントIDに対応することができます。

これまででは変数代入のためにTestsを利用しましたが、このTestsにはその名の通りテストしたい内容を記述してテストすることができます。  
例えば

```
pm.test("Response time is less than 200ms", function () {
  pm.expect(pm.response.responseTime).to.be.below(200);
});
```

と記述すればレスポンス200ms以上の場合、Failedとなります

Params
Authorization
Headers (4)
Body
Pre-request Script
Tests
Cookies
Code
Comments (0)

```

1 pm.test("Response time is less than 200ms", function () {
2   pm.expect(pm.response.responseTime).to.be.below(200);
3 });

```

Test scripts are written in JavaScript, and are run after the response is received.

SNIPPETS
Response body: Is equal to a string
Response headers: Content-Type header check
Response time is less than 200ms
Status code: Successful POST request
Status code: Code name has string
Response body: Convert XML body to a JSON Object
Use Tiny Validator for JSON data

Body
Cookies
Headers (10)
Test Results (0/1)
Status: 200 OK Time: 432 ms Size: 558 B

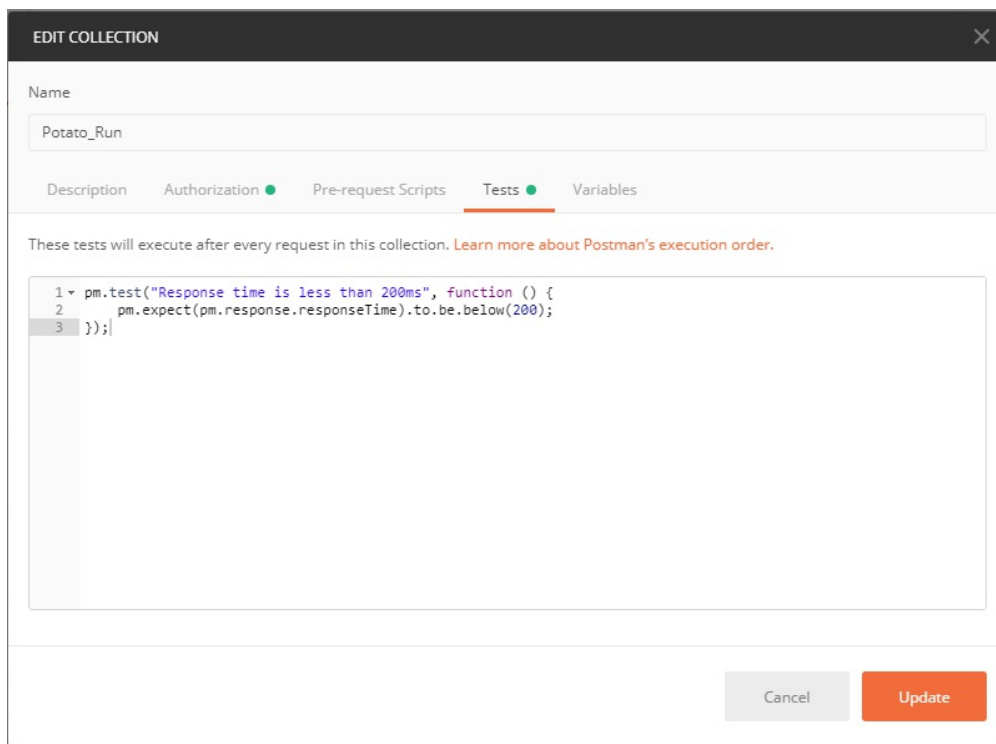
All Passed Skipped Failed

FAIL

Response time is less than 200ms | AssertionError: expected 432 to be below 200

このTestsはCollectionsごとにも管理できます



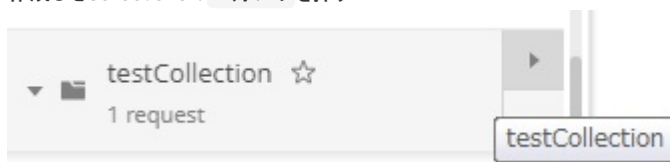


テストスクリプトはjavascriptで記述でき、画面右の **SNIPPETS** から追加できる例文や**公式ドキュメント**を参考にカスタマイズするとさらなる効率化が図れます

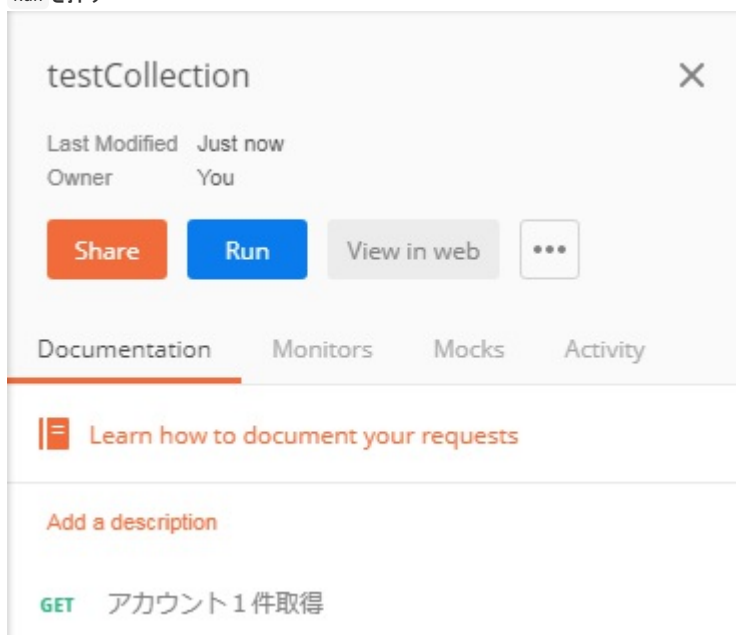
## Runnerで確認

作成したCollectionsを一括でテストしてみましょう

1. 作成したCollectionsの 三角マークを押す



2. Runを押す



Runnerが起動します

### 3. 左下の Run Collection を押す

Choose a collection or folder

testCollection

GET アカウント 1 件取得

Environment No Environment

Iterations 1

Delay 0 ms

Log Responses For all requests

Data Select File

☐ Keep variable values

Run testCollection

### 4. 自動でコレクション内のテストが始まり、結果が出ます

Collection Runner

Run Results

My Workspace

Run In Command Line

Docs

0 PASSED

0 FAILED

testCollection テスト環境

just now

Run Summary

Export Results

Retry

New

Iteration 1

GET アカウント 1 件取得

https://admin-api-dev.be...

testCollection / アカウント 1 件取得

200 OK

124 ms

222 B

This request does not have any tests.



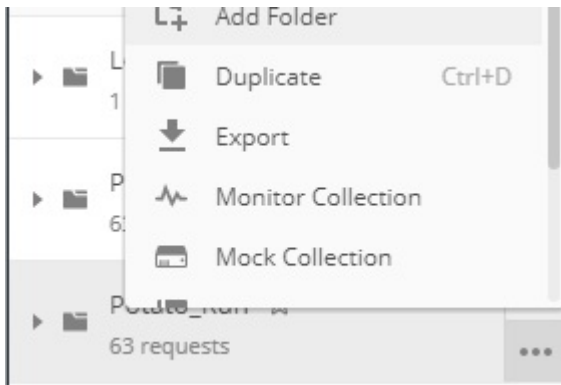
作成した場合、Testsの結果も出ます



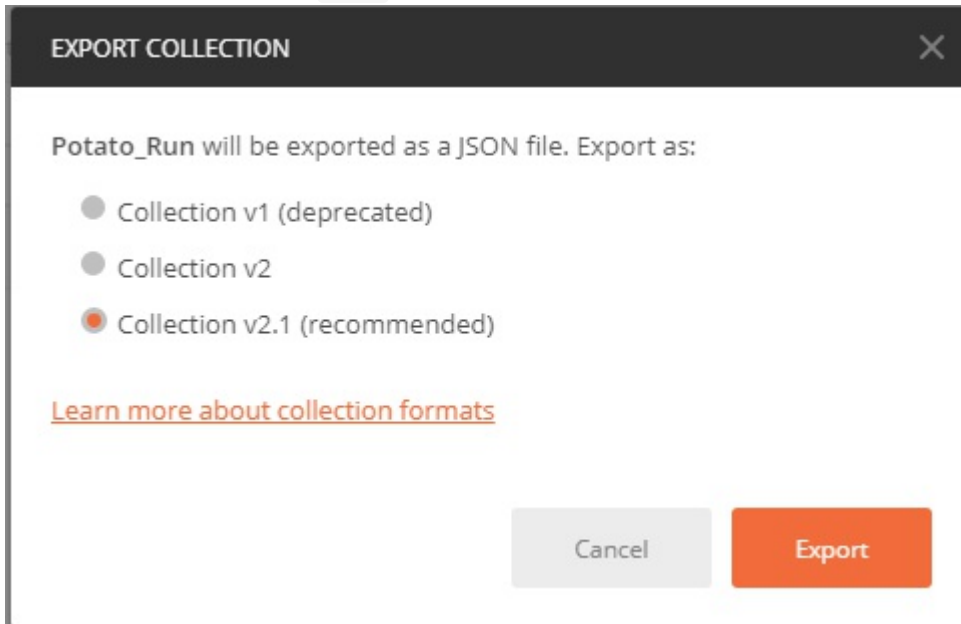
## json出力

ここまでがうまくいっているのであればコレクションをjson形式で出力します

1. Collectionsの ... から Export を押す



2. v2.1が選択されていることを確認し、Export を押す



### 3.Newmanをインストール

1. New manのインストールにはNode.js via package manager v6 以上が必要です  
無い場合 [ここ](#)から最新版をインストールしてください
2. コマンドラインで `npm install -g newman` と入力
3. (オプション)コマンドラインで `newman -v` と入力し、バージョンが返ってくるか確認

### 4.Slack Incoming Webhook URLを取得

1. Slackの通知を受けたいワークスペースにサインインした状態で[Slack Incoming Webhook](#)にアクセス
2. 通知したいチャンネルを選択し、インテグレーションの追加を押す

### チャンネルへの投稿

まず受信 Webフックがメッセージを投稿するチャンネルを選択します。

チャンネルを選択

または 新しいチャンネルを作成する

### Incoming Webhook インテグレーションの追加

着信 Web フックを作成することで、Slack API サービス利用規約に同意したものとみなされます。

3. Webhook URLに記述されているURLをコピー

### Webhook URL

この URL に JSON ペイロードを送信してください。

[セットアップの手順を表示](#)

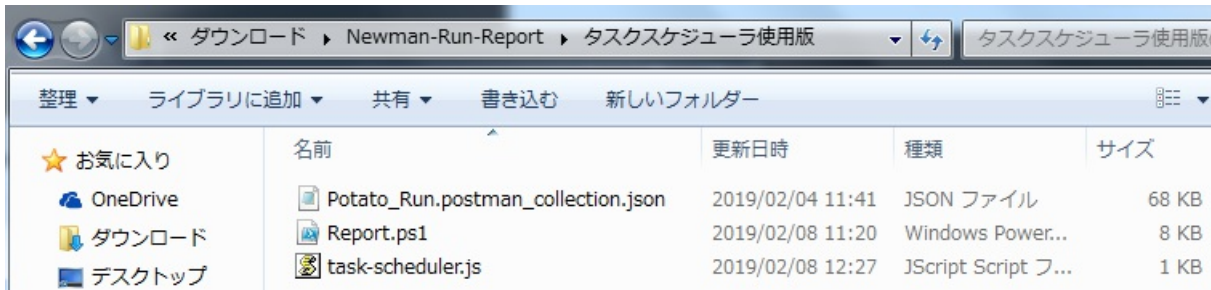
https://hooks.slack.com/services/

[URL をコピーする](#) • [再生成する](#)

## 5.Newman Run Report

### 準備1 - スクリプト編集

1. Newman-Run-Report\taskscheduler使用版 or 未使用版 内のReport.ps1と同じディレクトリに「Postman」で作成した collection.json を配置



2. Report.ps1 をメモ帳などで開き、# で囲まれた部分を編集する

newman run example.postman\_collection.json -r html,json の部分のjson名を「Postman」で作成した collection.json の名前に編集

```
Start-Transcript -path ".\newman¥log¥newman-run-report.log" -Append #テキストログ出力
$startDate = Get-date
$nowDate = Get-date
Write-Host "r n-----${nowDate}: テスト開始-----" -ForegroundColor Green
```

```
#####
```

```
newman run example.postman_collection.json -r html,json #PostmanのCollectionファイルを指定
```

```
#####
```

```
# .\newman¥最新json 読み込み
$files = @(Get-ChildItem -Path .\newman -Filter *.json)
```

\$webhook = "https://hooks.slack.com/services/XXXXXXXX/XXXXXXXX/XXXXXXXXXXXXXXXXXXXX の部分を「Slack Incoming Webhook」で  
取得したURLに編集

```
#####
$webhook = "https://hooks.slack.com/services/XXXXXXXX/XXXXXXXX/XXXXXXXXXXXXXXXXXXXX" #Slack Webhookを入力
#####
$json = ConvertTo-Json $payload
$body = [System.Text.Encoding]::UTF8.GetBytes($json)
Invoke-RestMethod -Uri $webhook -Method Post -Body $body
```

## (オプション) 準備2 - Powershell設定

New man-Run-Reportを使用するには Windows Powershell 3.0以上 が必要ですが、OSごとにデフォルトでインストールされているバージョンが違います。バージョンが気になる方はPowershellに `Get-Host` と入力するとバージョンが確認できます。

Windows OSバージョン	デフォルト	導入可能
Windows Server 2016	5.1	(6.0)
Windows 10 RS1	5.1	(6.0)
Windows 10	5.0	5.1/(6.0)
Windows Server 2012 R2	4.0	5.0/5.1/(6.0)
Windows 8.1	4.0	5.0/5.1/(6.0)
Windows 2012	3.0	4.0/5.0/5.1/(6.0)
Windows 8	3.0	4.0/5.0/5.1/(6.0)
Windows Server 2008 R2	2.0	3.0/4.0/5.0/5.1/(6.0)
Windows 7 SP1	2.0	3.0/4.0/5.0/5.1/(6.0)
Windows Server 2008 SP2	1.0	2.0/3.0/4.0/5.0/5.1/(6.0)

デフォルトのバージョンが2.0以下の場合はPowershellのバージョンアップを行ってください

Windows Manage Framework (WMF) をバージョンアップするとWMFに含まれるPowershellもバージョンアップされます。今回は比較的新しく、安定しているPowershell5.1にアップデートします。

[WMF5.1のインストールと構成](#)からご使用のOSに合うパッケージをダウンロード

しかしこのままではうまく実行できません。デフォルトでは Powershellスクリプト(.ps1) の実行禁止という実行ポリシー<sup>^1</sup>になっているからです。なので、実行ポリシーを変更します。

### Powershellの実行ポリシーの変更

管理者として実行しているPowershell上で `Set-ExecutionPolicy AllSigned` と入力

```
PS C:\#hoge> Set-ExecutionPolicy AllSigned
```

#### 実行ポリシーの変更

実行ポリシーは、信頼されていないスクリプトからの保護に役立ちます。実行ポリシーを変更すると、`about_Execution_Policies` のヘルプ トピックで説明されているセキュリティ上の危険にさらされる可能性があります。実行ポリシーを変更しますか？

[Y] はい(Y) [N] いいえ(N) [S] 中断(S) [?] ヘルプ (既定値は "Y"): y  
PS C:\#hoge>

すると画像のように質問されるので Y を入力

再び、Powershellバージョンアップ作業にもどります

1. ダウンロードしたzipファイルを解凍し、`Install-WMF5.1.ps1` を管理者としてPowershellで実行
2. 実行するか聞かれるので R を入力

```
C:\#hoge> Install-WMF5.1.ps1
```

この信頼されていない発行元からのソフトウェアを実行しますか？

ファイル C:\#hoge\Install-WMF5.1.ps1 の発行元は CN=Microsoft Corporation,  
OU=MOPR, O=Microsoft Corporation, L=Redmond, S=Washington, C=US

であり、このシステムで信頼されていません。信頼された発行元からのスクリプトのみを実行してください。

[V] 常に実行しない(V) [D] 実行しない(D) [R] 一度だけ実行する(R) [A] 常に実行する(A) [?] ヘルプ (既定値は "D"):

あとは表示に従って進めていくとバージョンアップができます

## スクリプト実行

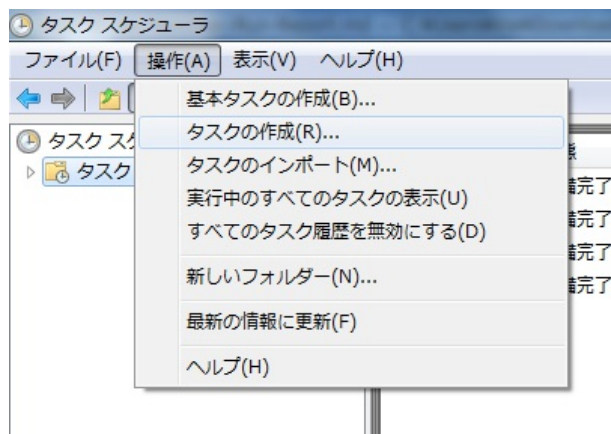
`Report.ps1` をPowershellで実行すると `newman` フォルダが作成され、その中にログがたまっていきます。また、Slackの指定したチャンネルに結果が通知されます。

## 6.(オプション) WindowsタスクスケジューラにNewman Run Reportを登録する

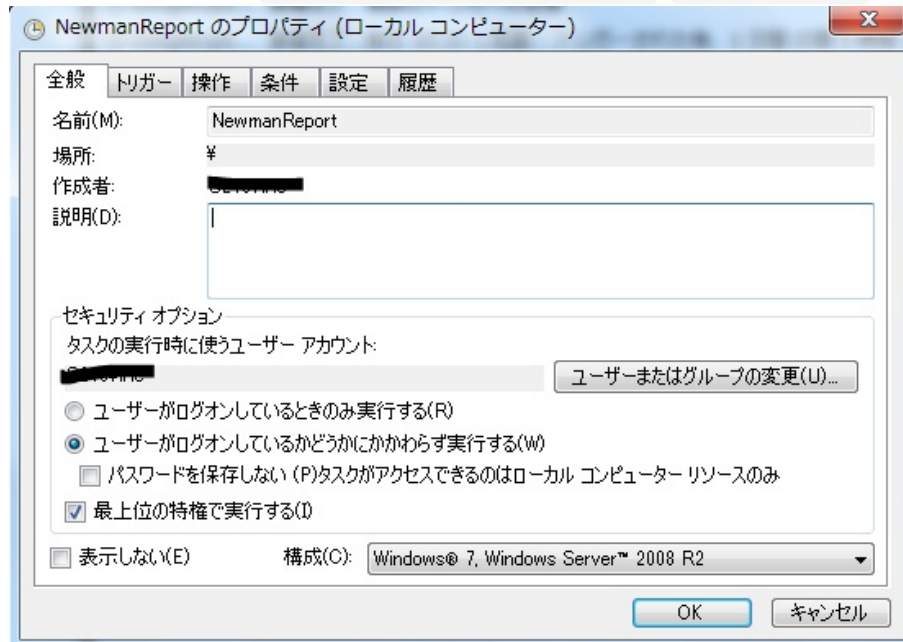
テストを一定期間ごとに行いたい場合はWindowsタスクスケジューラにNewman Run Reportを登録します

1. タスクスケジューラを起動
2. 操作 → タスクの作成 をクリック

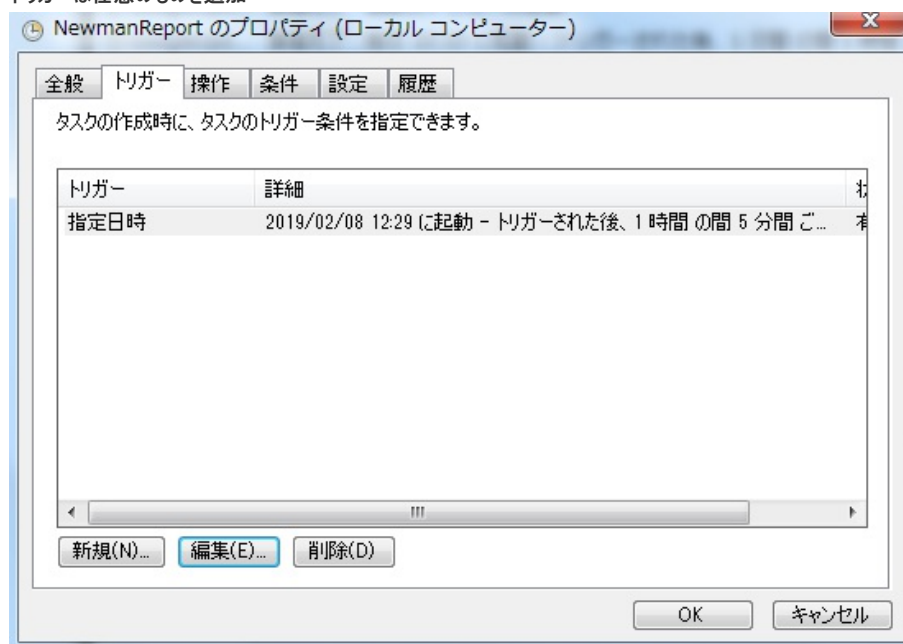




3. 全般の名前は任意のものを、ユーザーがログオンしているかどうかにかかわらず実行すると 最上位の特権で実行する にチェック



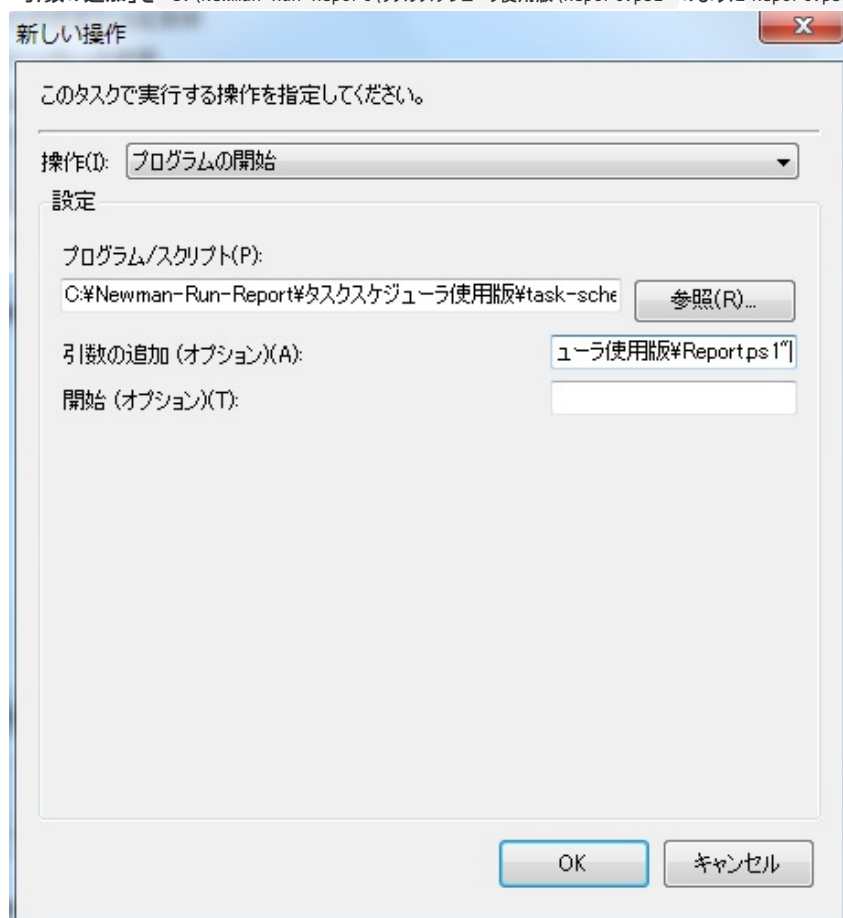
4. トリガーは任意のものを追加



5. 操作の 新規 から新しい操作を作成

6. 「プログラム/スクリプト」を `C:\Newman-Run-Report\taskscheduler使用版\task-scheduler.js` のように `task-scheduler.js` のパスを指定

7. 「引数の追加」を "`C:\Newman-Run-Report\taskscheduler使用版\Report.ps1`" のように `Report.ps1` のパスを指定



8. そのほかの項目はお好みで変更し `OK` を押すとトリガー条件を満たしたときにNew man Run Reportが実行されます

Name: New man Run Report

License: This softw are is released under the MIT License.

Created date: 2019/02/12

Author: Kouki Ooe