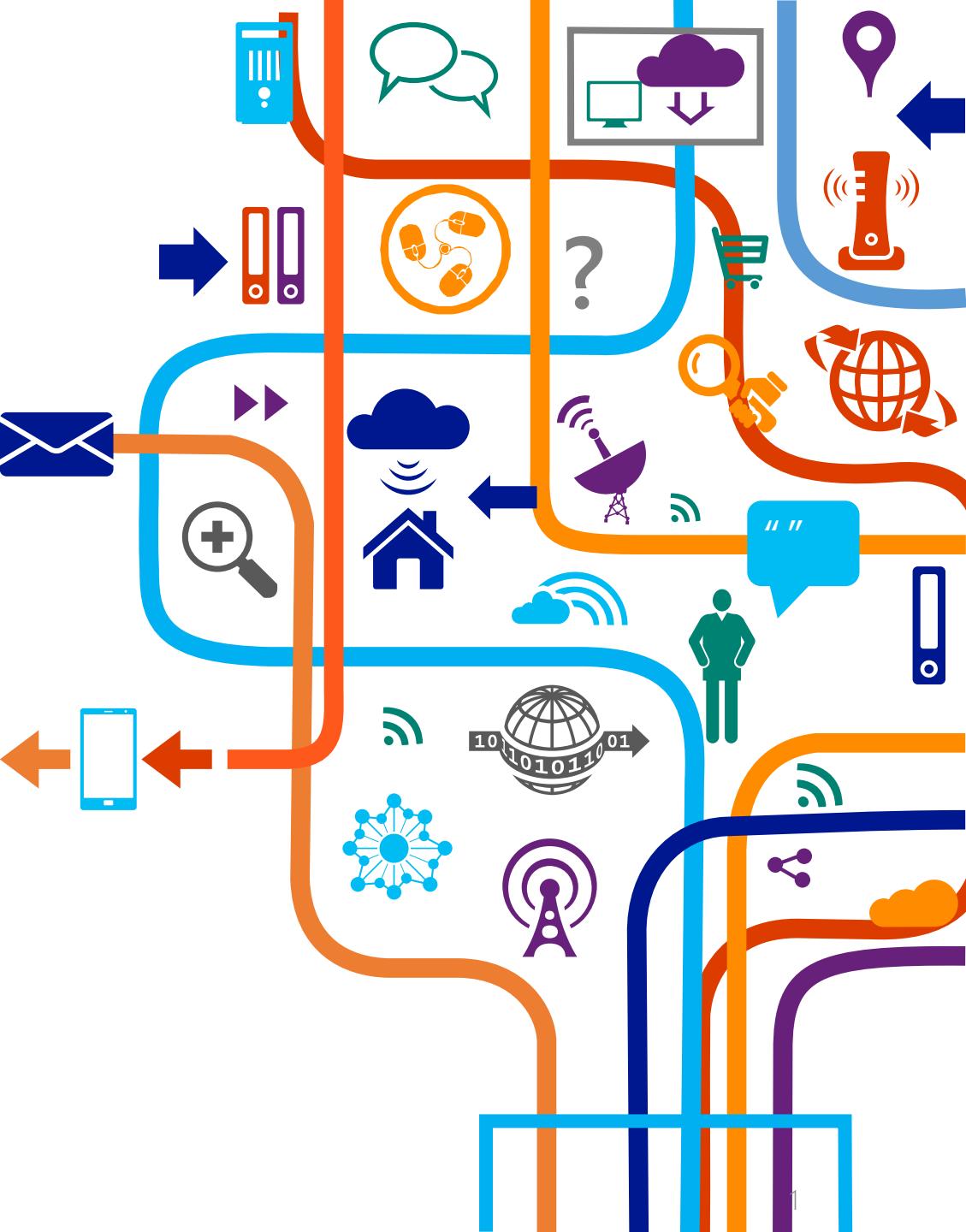


Internet of Things IoT Kit ハンズオントレーニング

Windows 10 IoT Core for Raspberry PI 2!!

2015/6/3版

日本マイクロソフト株式会社
デベロッパー＆エンジニアリング統括本部
テクニカルエンジニアリスト
太田 寛



目次

- はじめに
 - IoT – Internet of Things
 - Windows 10 IoT Core
- 実習
 - Lチカ
 - Win10 RP2をクラウドにつなげる
 - クラウドからLチカを制御する
 - クラウドにデータをアップロードする
- ネクストステップ

確認です

- ✓ 開発用PC持ってきましたか？
- ✓ Windows 10 Technical Previewはインストール済みですか？
- ✓ Visual Studio 2015 RCはいってますか？
- ✓ Raspberry PI2用のマイクロSDカードは作成してきましたか？
- ✓ Azureの契約はしてきましたか？
- ✓ Azure Tools SDKはインストールしてますか？
- ✓ Raspberry PI2 + 実習キット一式ありますね？
- ✓ 説明資料とアンケート、ありますか？

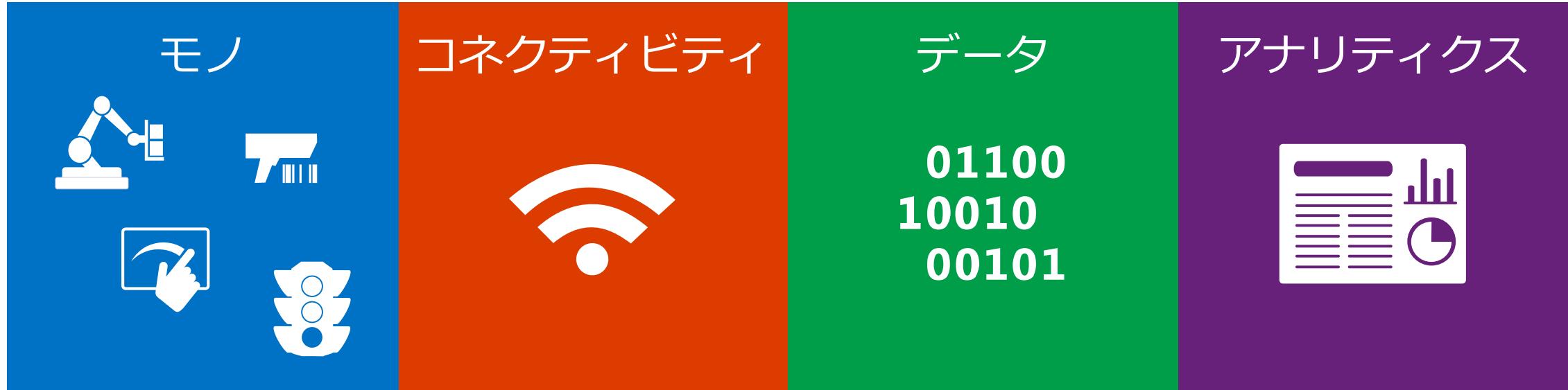
はじめに

- 目的
 - Windows 10 IoT Core で動く Raspberry PI2を使って開発実習を行います。
 - 折角なので、Microsoft Azure とつないで、“Internet of Things”の開発に必要なスキルの、基本中の基本を学びます

- 実習内容

「チカして、
クラウドとつないで、
制御・蓄積する

IoT (Internet of Things) とは

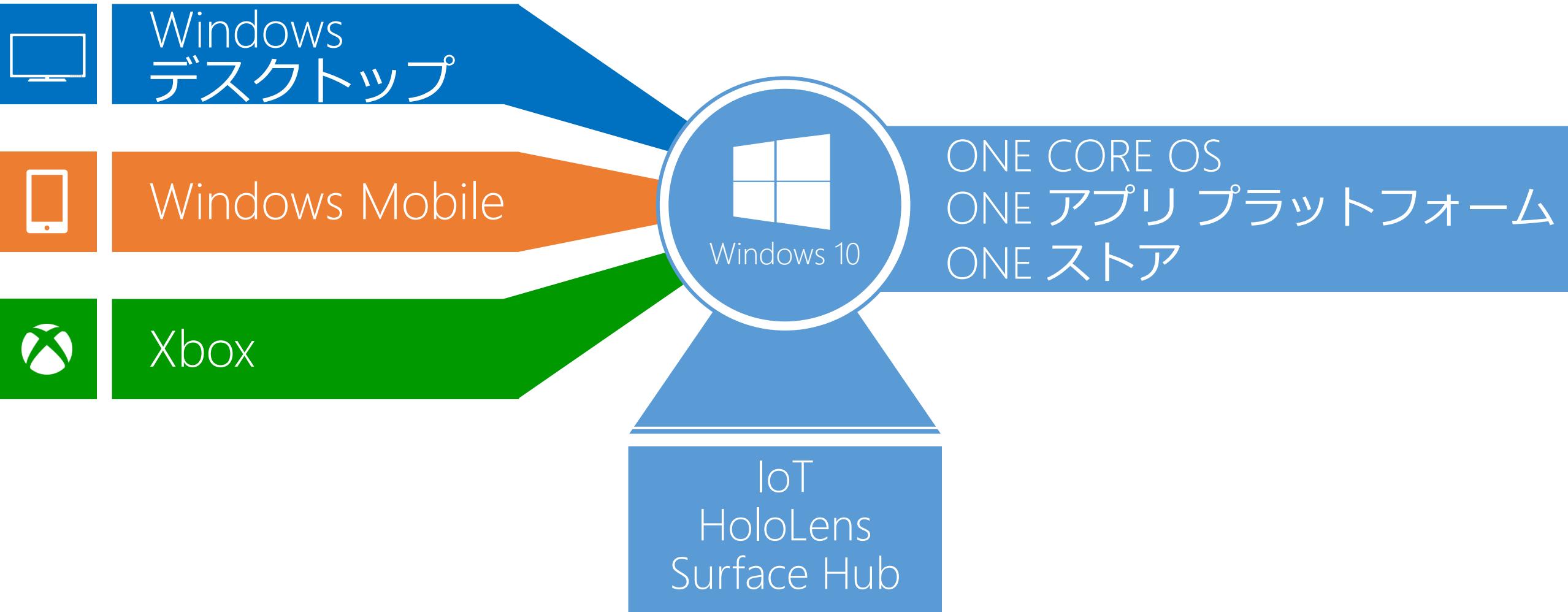


“「Connected World」のソリューションとは、センサーやテクノロジーを組み合わせ、“もの”や“インフラ”が、監視・分析・制御システムなどとインターネット上で対話可能になること

Source: Forrester



Windows 10



Windows 10 IoT

Windows 10 IoT for industry devices

デスクトップシェル、Win32 アプリ、ユニバーサルアプリ、
ユニバーサルドライバー
1 GB RAM、16 GB ストレージ
X86/x64

Windows 10 IoT for mobile devices

モダンシェル、ユニバーサルアプリ、
ユニバーサルドライバー
512 MB RAM、4 GB ストレージ
ARM

Windows IoT Core

シェルなし、ユニバーサルアプリ、
ユニバーサルドライバー
256 MB RAM、2GB ストレージ
X86 または、ARM



新たな Windows
アップデートと
サービス



セキュリティと
認証



Visual Studio &
UWP



新しいユーザー
インターフェース



周辺デバイスと
の接続と活用の
強化



Microsoft
Azure IoT

Windows 10 IoT Editions

デスクトップアプリ
(Win32,.NET,WPF) を使いたい

モバイル用途、シェルが必要で、複数のアプリも使いたい

その他 :UWPのみ
(ユニバーサルアプリ)

Industry



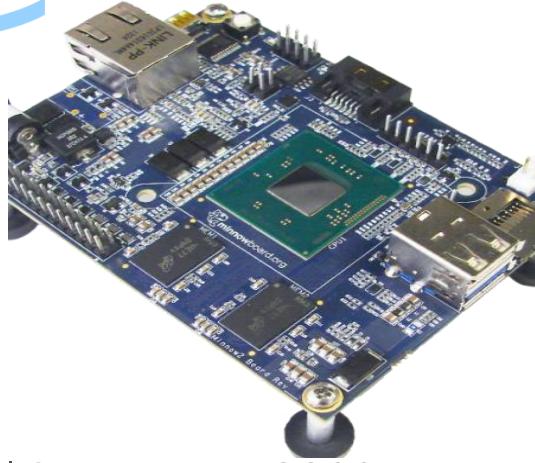
Mobile



IoT Core



Windows 10 IoT Core



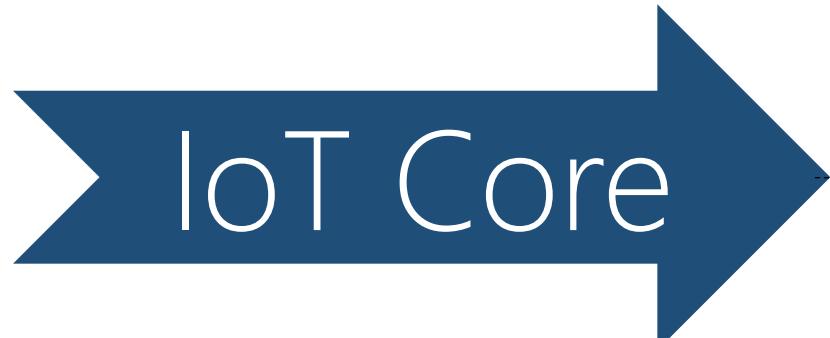
Intel® Atom™ E3800 processor



Broadcom 2836 900 MHz quad-core ARM Cortex-A7 CPU



Qualcomm Snapdragon 410
(APQ8016)



X86/ARM
UWP対応
専用装置向け
UI/ストア/MS アプリなし
256 MB RAM, 2GB storage

セットアップ

<http://www.WindowsOnDevices.com>

Microsoft | Windows デベロッパーセンター

ホーム Windows の新しい時代 ドキュメント ダウンロード サンプル コミュニティ プログラム ダッシュボード

モノのインターネット

モノのインターネット (IoT) は、デバイス、センサー、クラウド、データ、そして想像力を自分にとって最も重要なものを作りましょう。

すぐに作業を開始

作業の開始 プロジェクト ドキュメントとサンプル 上く寄せられる質問 タンク ハードウェア コミュニティ Build 2015

OS Image 作成方法 開発環境構築方法

Microsoft | Windows Dev Center

Home Explore Docs Downloads Samples Community Programs Dashboard

Docs and Samples

Download code samples to get started with Windows on Devices. Also, read docs to help you use tools and resources to help you develop.

Did you set up your environment?

We assume you already [set up your environment](#), have [Windows 10 IoT Core](#) installed, and have a device running Windows IoT Core (MinnowBoard Max, Raspberry Pi 2, or VM).

Questions/Suggestions

Remember, you can always send [contact us](#) for help and suggestions!

Get Started

Select your device to get started. If you already have a device running Windows 10 IoT Core Insider Preview, then start building with [PowerShell](#) now.

Learn more about Windows IoT devices

1. Select Your Device
2. Set up your Device
3. Set up your PC
4. Develop

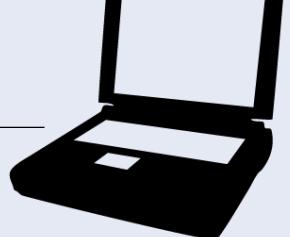
Raspberry Pi 2 MinnowBoard Max Galileo

IoT Core の開発環境



Raspberry PI2

Local Network



Windows IoT Core Watcher でリストアップ

Windows IoT Core Watcher

BoardName	MacAddress	IpAddress	LastPing	Online	OsVersion	Bid
minwinpc	b8:27:eb:5c:aa:02:00:00	10.168.132.52	5/18/2015 11:28:23 AM	✓	Windows 10 IoT Core	

Windows B Web Management
AppX Manager

Status Apps Processes Performance Debugging ETW Realtime Perf Tracing Device Manager Networking

Installed apps

DefaultApp_1.0.0.0_neutral_cw5n1h2txyewy Remove Start

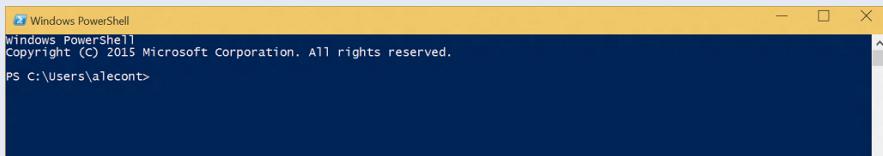
Running apps

PID	Name
180	IoTCoreDefaultApp
2000	ZWaveBackgroundService

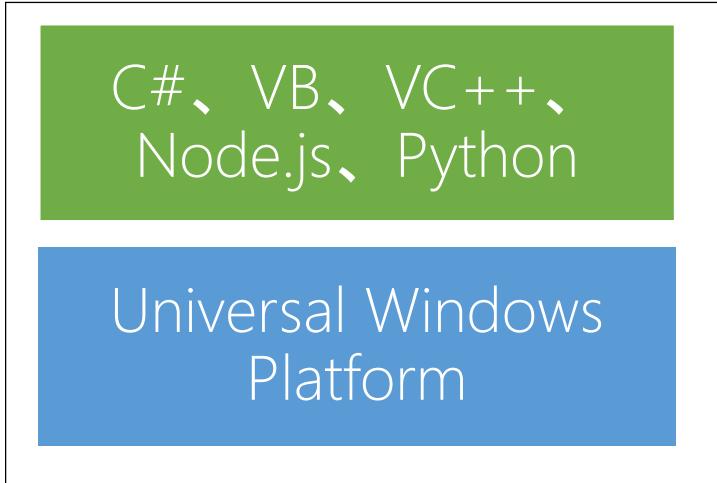
Install app

アプリ管理

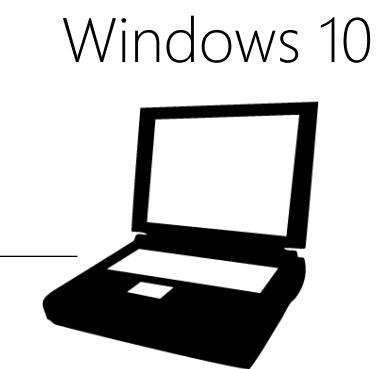
Power Shell で管理



Universal Windows Apps を開発する

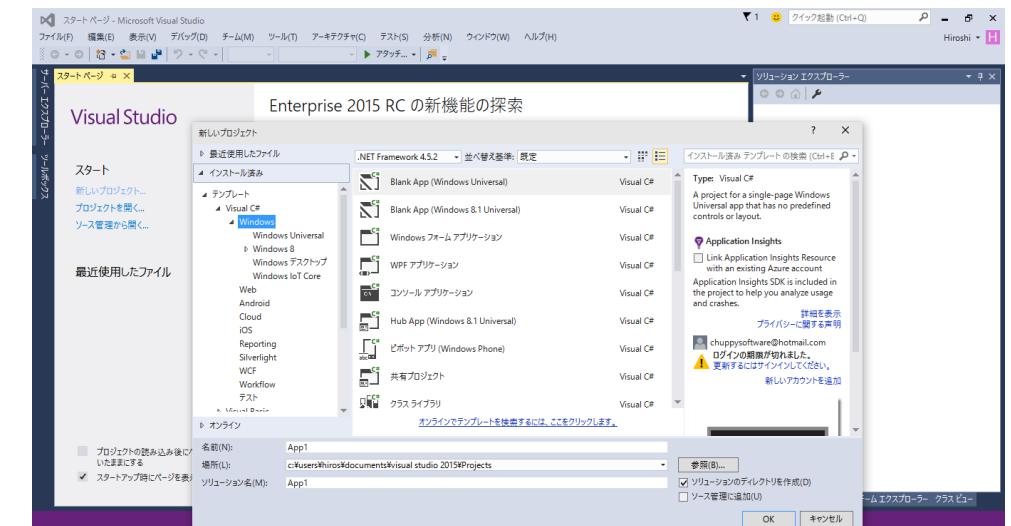


Raspberry PI2



開発用PC

Visual Studio 2015 で開発
WindowsDeveloperProgramForIoT.msi
を追加インストール



アプリの配置
リモートデバッグ

実習内容

1.
ルチカ実習

2.
クラウドへの
接続テスト

Win10 for Raspberry PI 2



+
LED
サーミスター

実習範囲

3.
Azure Storage 作成

4.
Azure Web API 作成

取得
送信

Web API

蓄積 & 参照

Microsoft Azure

5.
LED遠隔操作

6.
温度通知

1. Lチカ実習



実習内容

1.
ルチカ実習

2.
クラウドへの
接続テスト

Win10 for Raspberry PI 2



5.
LED遠隔操作

実習範囲

3.
Azure Storage 作成

4.
Azure Web API 作成

取得
送信

Web API

蓄積 & 参照

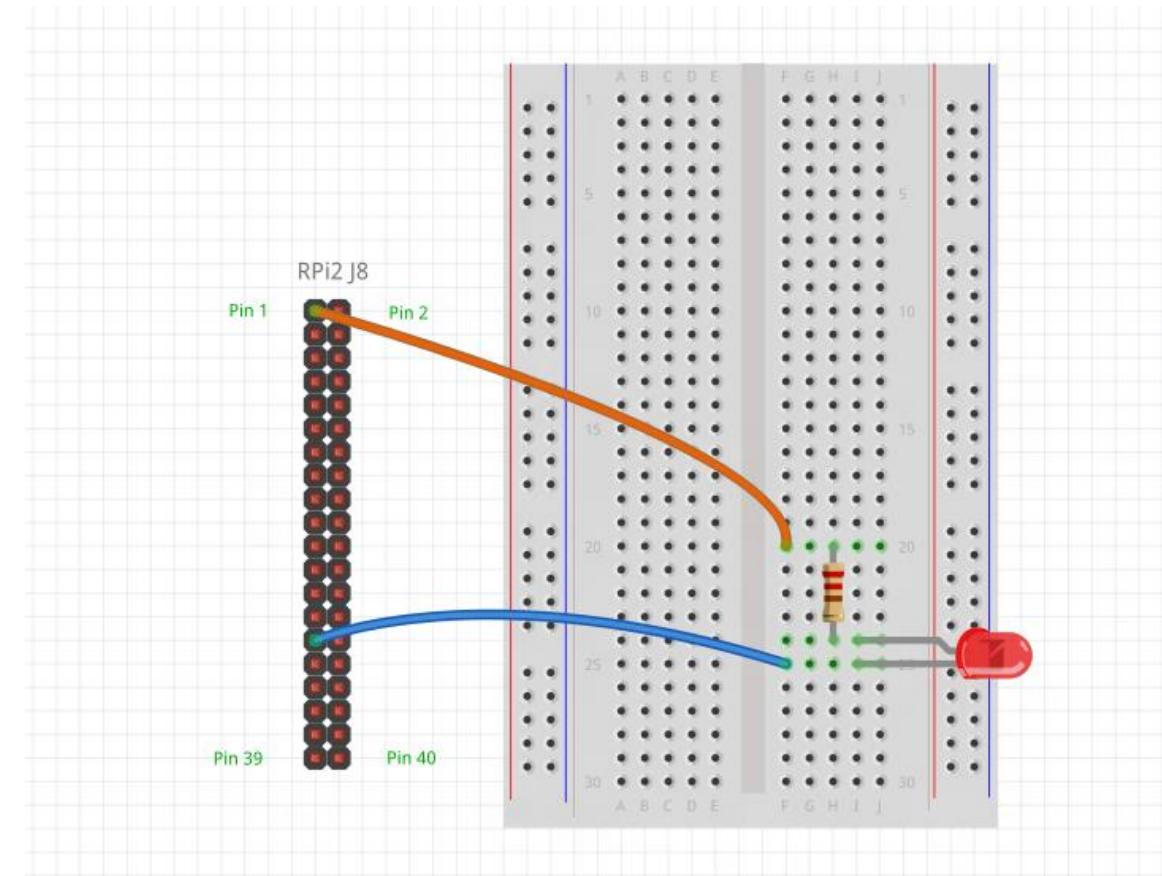
Microsoft Azure

6.
温度通知

LEDなどをつなぐ

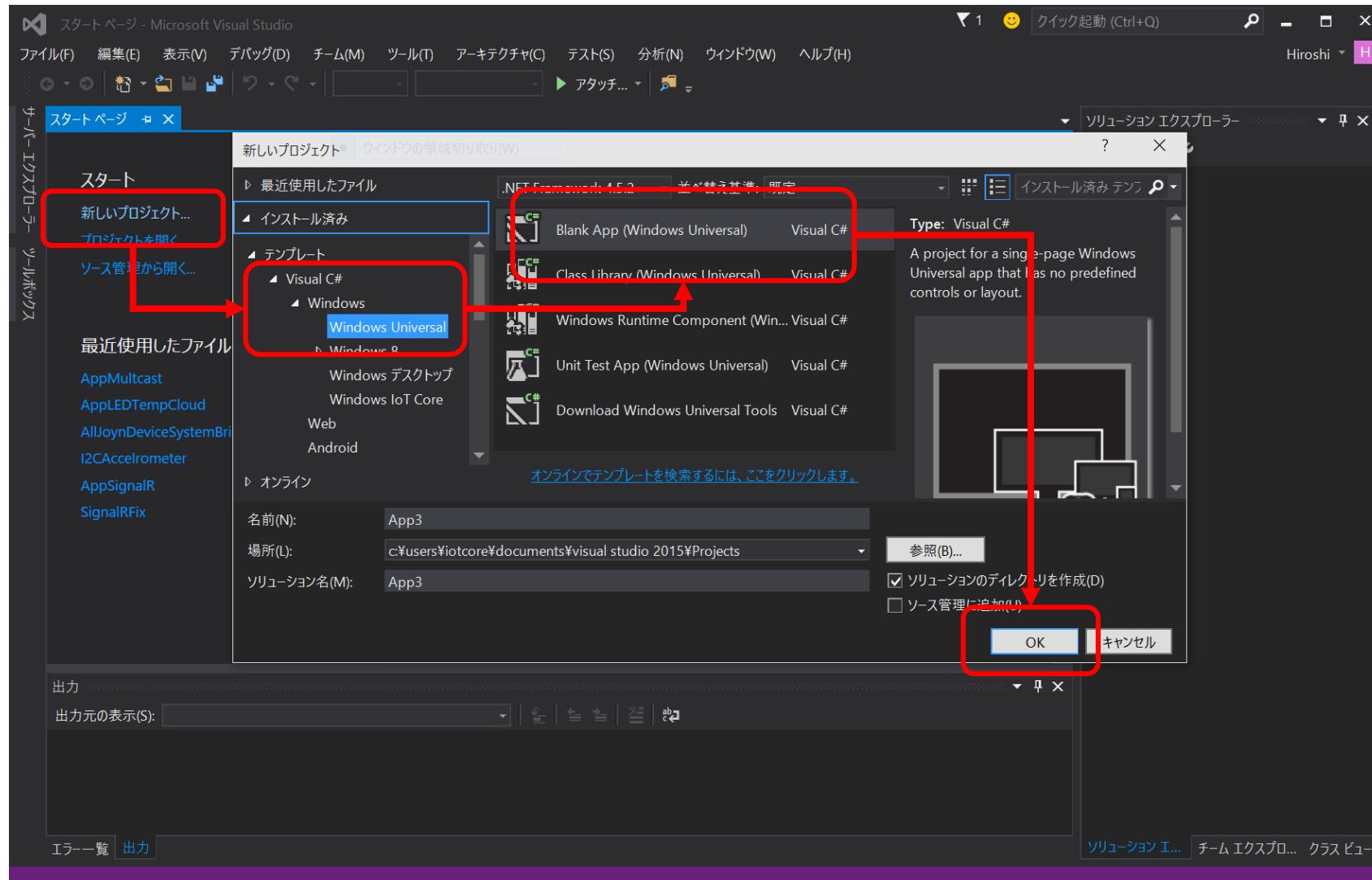


3.3V PWR	1	2	5V PWR
I2C1 SDA	3	4	5V PWR
I2C1 SCL	5	6	GND
GPIO 4	7	8	Reserved
GND	9	10	Reserved
SPI1 CS0	11	12	GPIO 18
GPIO 27	13	14	GND
GPIO 22	15	16	GPIO 23
3.3V PWR	17	18	GPIO 24
SPI0 MOSI	19	20	GND
SPI0 MISO	21	22	GPIO 25
SPI0 SCLK	23	24	SPI0 CS0
GND	25	26	SPI0 CS1
Reserved	27	28	Reserved
GPIO 5	29	30	GND
GPIO 6	31	32	GPIO 12
GPIO 13	33	34	GND
SPI1 MISO	35	36	GPIO 16
GPIO 26	37	38	SPI1 MOSI
GND	39	40	SPI1 SCLK



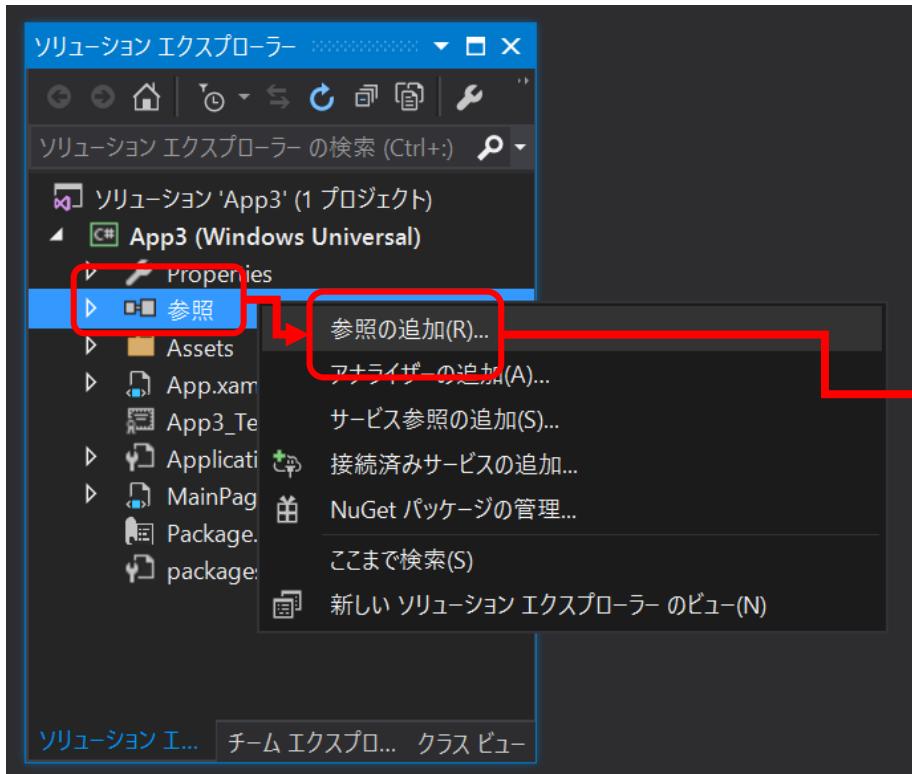
「チカアプリを開発する

1. Visual Studio 2015 RCを起動する
2. “新しいプロジェクト”をクリック
3. “Visual C#”→“Windows”→“Windows Universal”を選択
4. “Blank App”を選択
5. “OK”をクリック

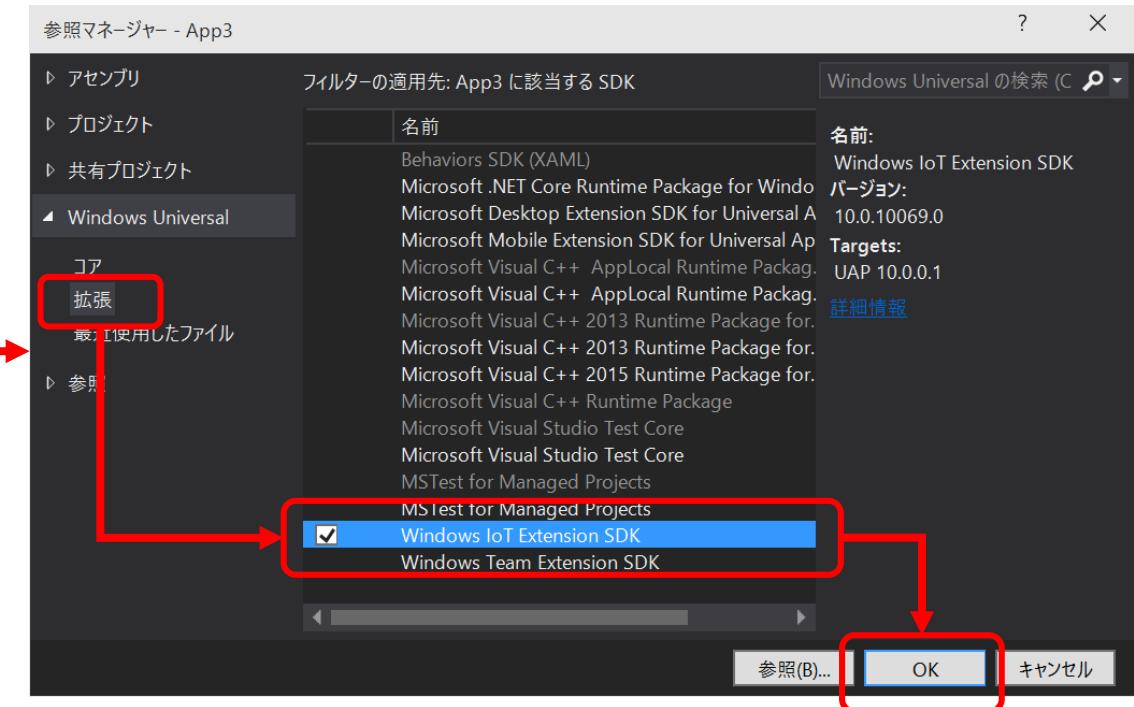


IoT Core Extension SDKを追加

1. “参照”を右クリックし、“参照の追加”を選択



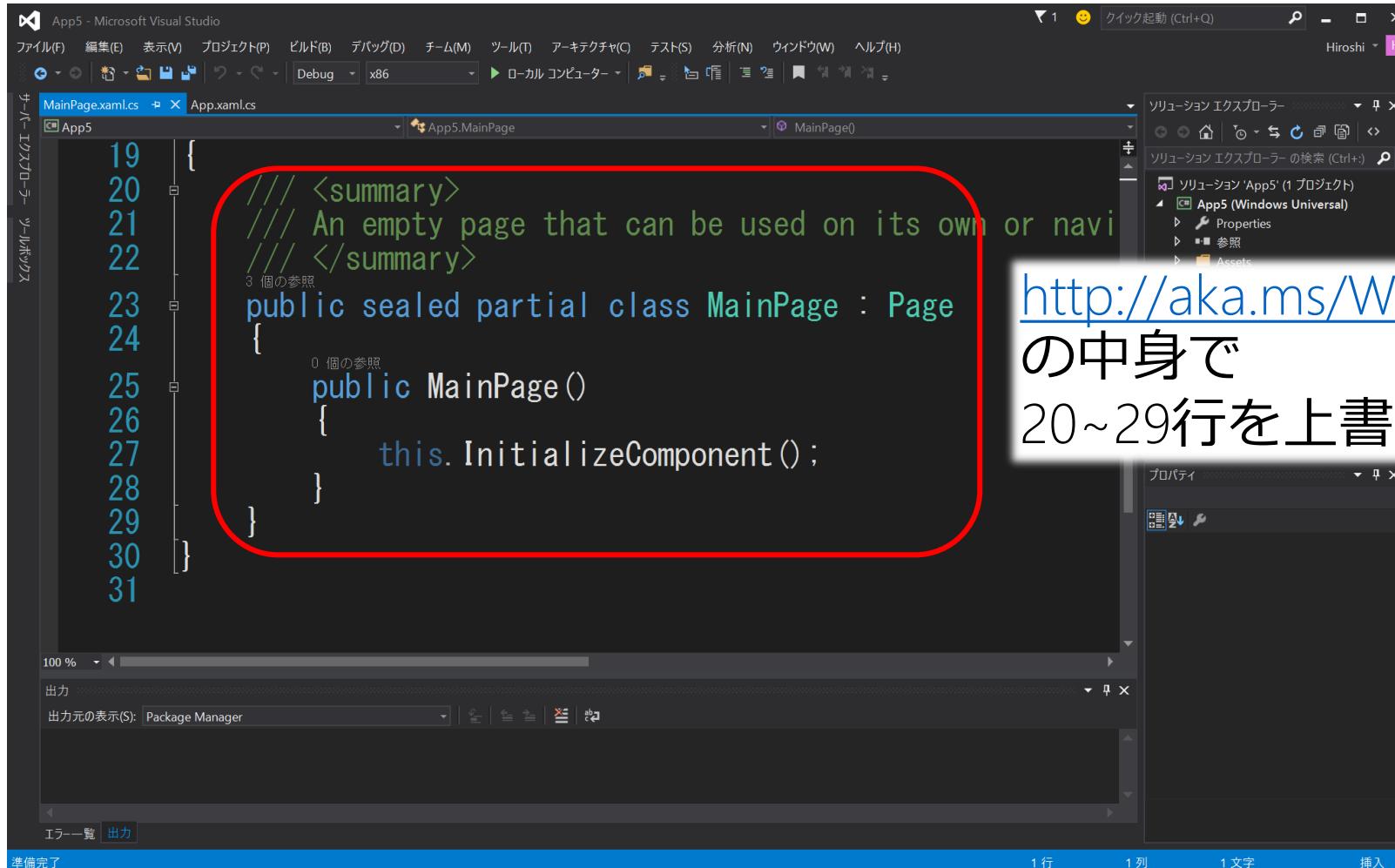
2. “Windows Universal”→“拡張”を選択



3. “Windows IoT Extension SDK”を選択

Lチカラジック追加

MainPage.xaml.cs を開く

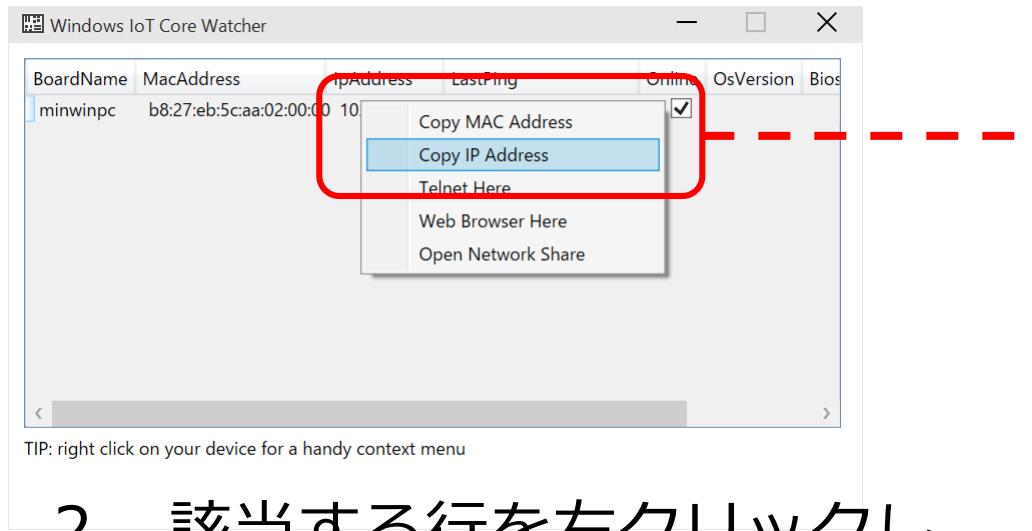


```
19 {
20     /// <summary>
21     /// An empty page that can be used on its own or navi
22     /// </summary>
23     public sealed partial class MainPage : Page
24     {
25         public MainPage()
26         {
27             this.InitializeComponent();
28         }
29     }
30 }
31 
```

<http://aka.ms/Win10LChikaCode>
の中身で
20~29行を上書きする

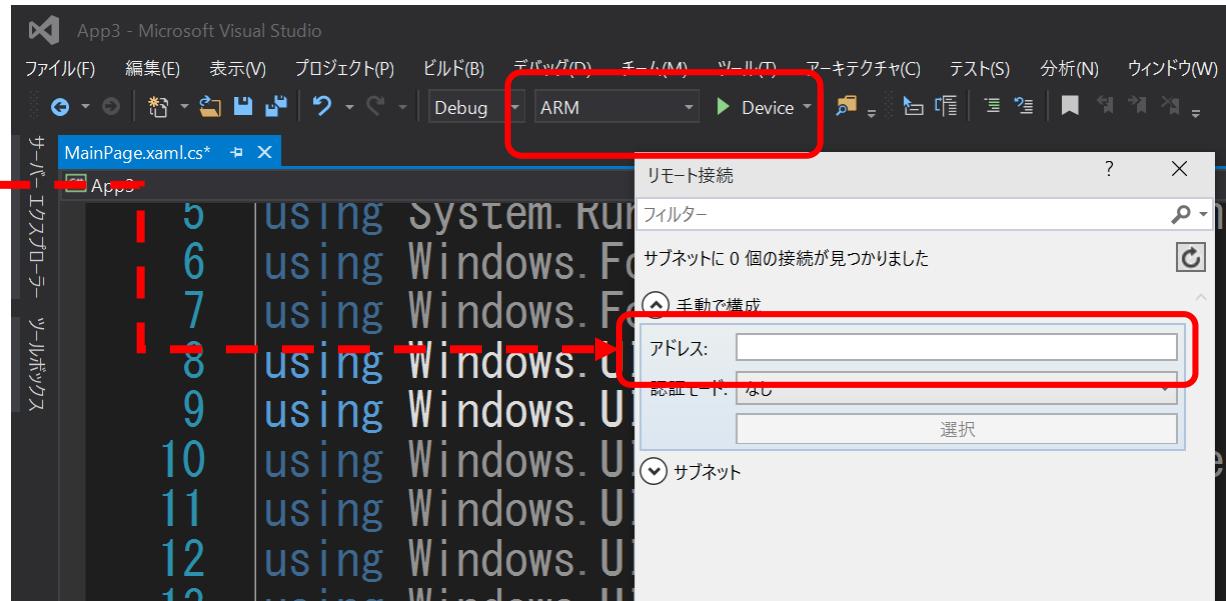
「チカアプリ」の実行

1. “ARM”、“リモートコンピュータ”を選択

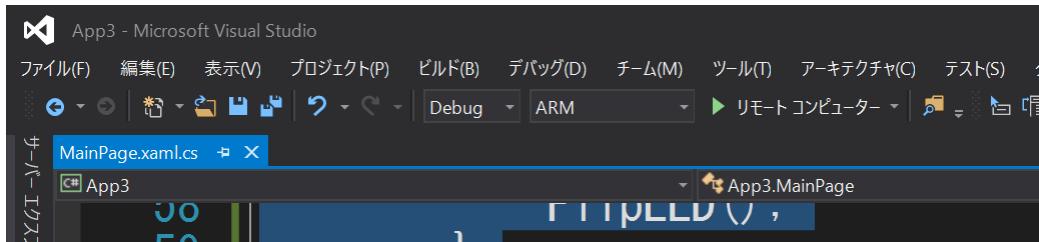


2. 該当する行を右クリックし
“Copy IP Address”を選択

4. “▶リモートコンピュータ”をクリック
→実行開始



3. “アドレス”にペースト
“認証モード”を“なし”に設定
“選択”をクリック



2. クラウドへの 接続テスト



実習内容

1.
力実習

2.
クラウドへの
接続テスト

Win10 for Raspberry PI 2



5.
LED遠隔操作

実習範囲

3.
Azure Storage 作成

4.
Azure Web API 作成

Web API

取得
送信



蓄積 & 参照



Microsoft Azure

6.
温度通知

コメントを外して実行

MainPage.xaml.csファイルで
StartFlipLED()の前に“//”を入力
TestConnect().Wait()の前の“//”を削除

```
60  |  private void MainPage_Loaded(object
61  |  {
62  |  // TestConnect().Wait();
63  |  StartFlipLED();
64  |  // LEDRemoteControl();
65  |  // StartThmtSensing();
66  |
67  }
```

```
60  |  private void MainPage_Loaded(object
61  |  {
62  |  TestConnect().Wait();
63  |  // StartFlipLED();
64  |  // LEDRemoteControl();
65  |  // StartThmtSensing();
66  |
67  }
```

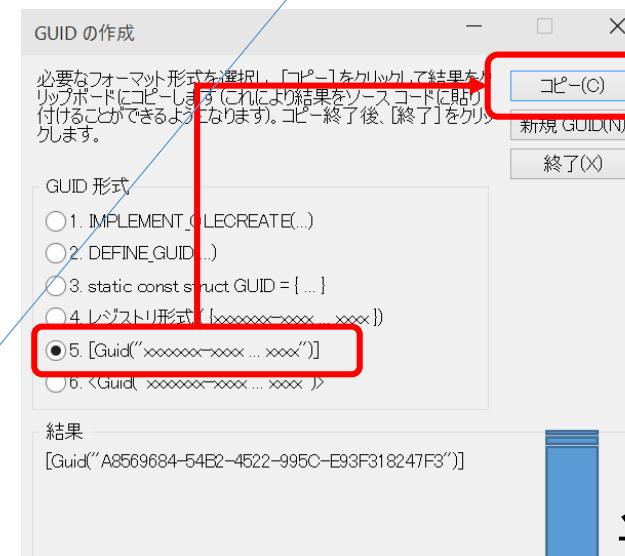
deviceId の設定

MainPage.xaml.cs

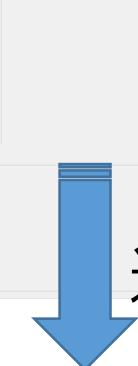
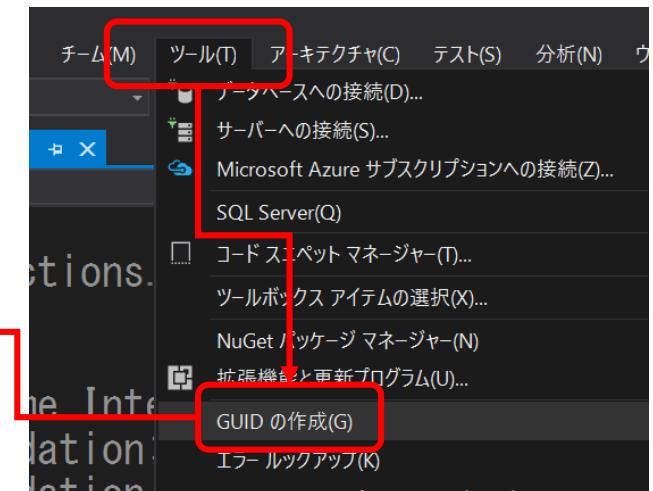
```
32 |     private string webApiEndPoint = "[WEB API URL]";  
33 |     Guid deviceId = new Guid();  
34 |     double thresholdTemperature = 25;
```

```
= new Guid("31E4AD72-4E7D-48C5-  
thresholdTemperature = 25;
```

文字列をコピー



“ツール”→“GUIDの作成”を選択

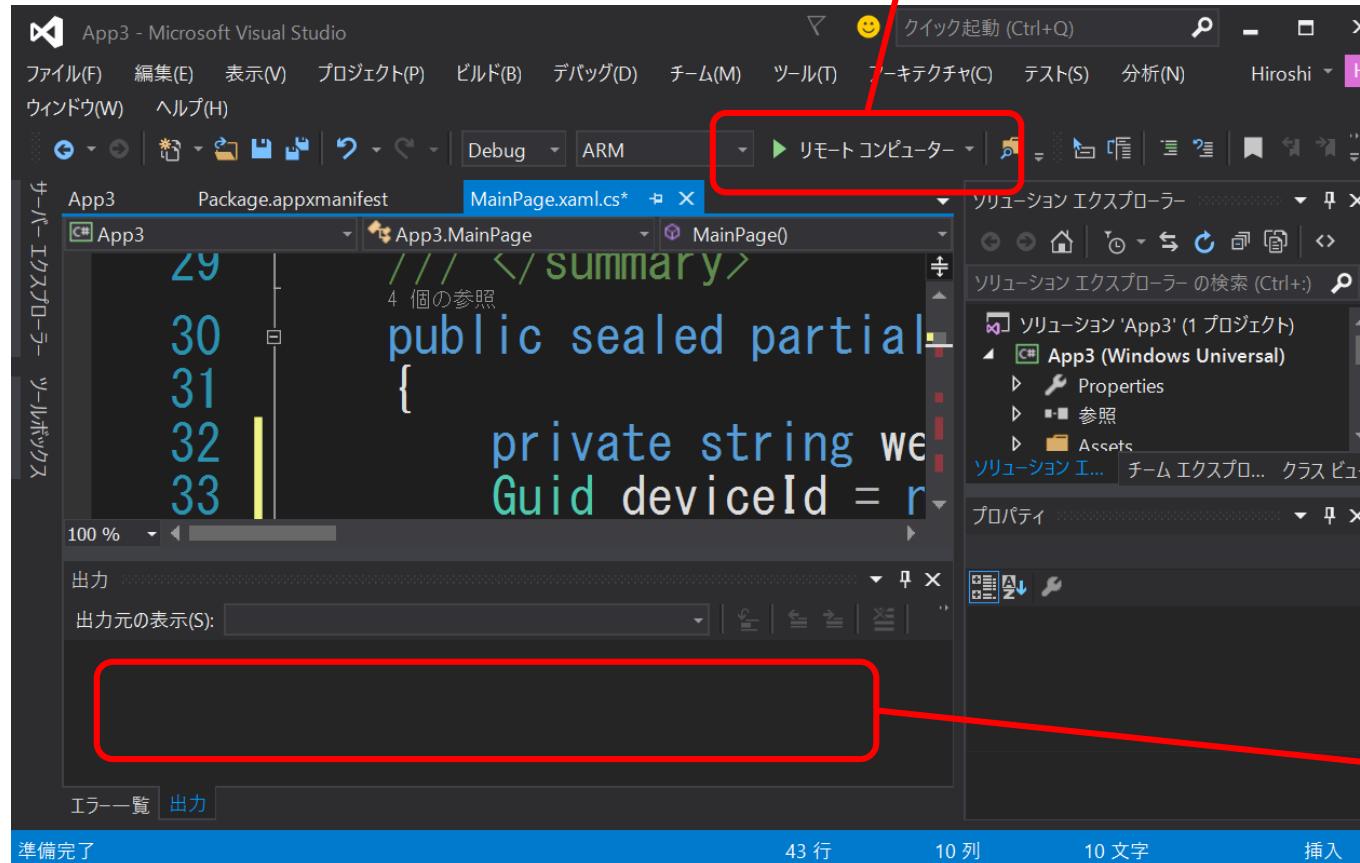


適当な場所にペースト

```
[Guid("31E4AD72-4E7D-48C5-B2CD- • • • • ")]
```

実行

クリックして実行



“出力”ビューに
“Received –Hello ...”
と表示されたらOK

3. Azure Storage 作成



実習内容

1.
ルチカ実習

2.
クラウドへの
接続テスト

Win10 for Raspberry PI 2



5.
LED遠隔操作

実習範囲

3.
Azure Storage 作成

4.
Azure Web API 作成

取得
送信

Web API

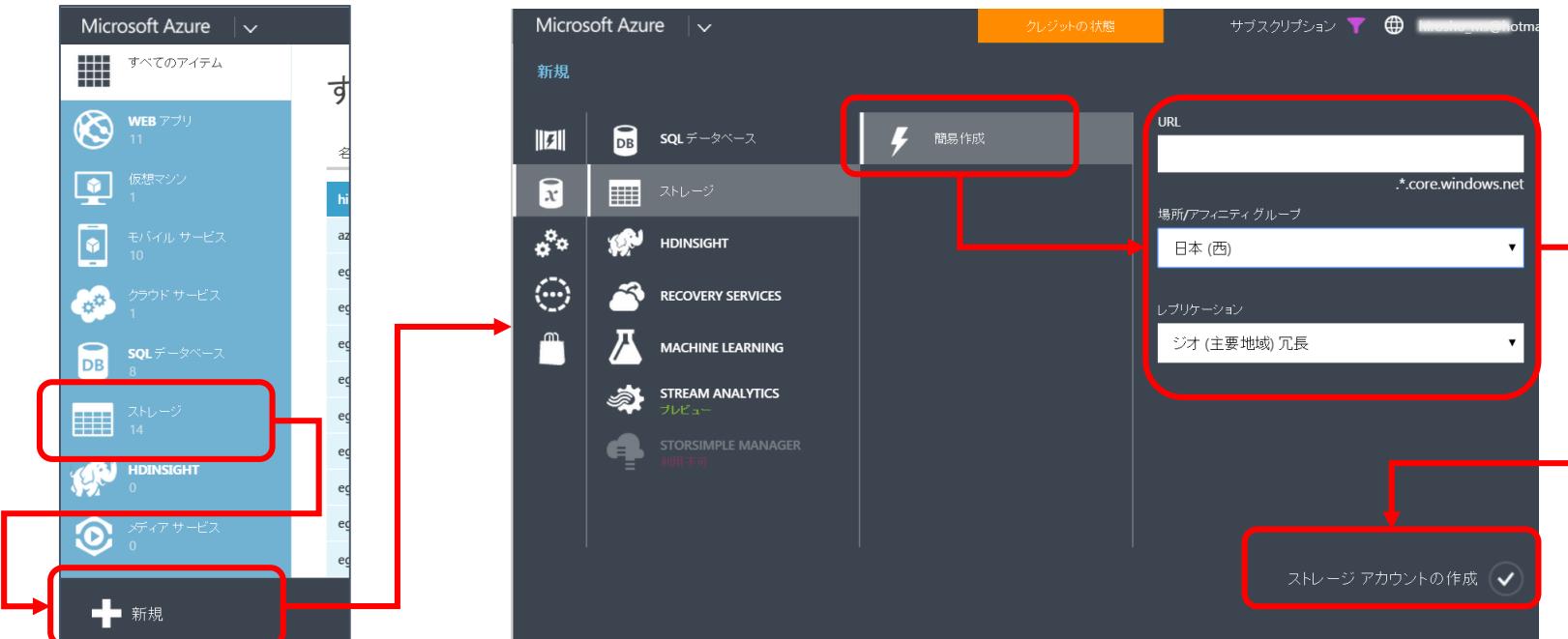
蓄積 & 参照

Microsoft Azure

6.
温度通知

Azure Storage の作成

- ・ブラウザで <http://manage.windowsazure.com> を開く



“ストレージ”を選択し、
“+新規”をクリック

“簡易作成”を選択

“URL”に適当な名前を入力
ここでは、“iotstorage”と仮にしておきます
“場所”は“日本（西）”を選択し、右下の“✓”をクリック

アクセスキーの確認

出来上がった“ストレージ”をダブルクリック

“アクセスキーの管理”をクリック
表示されたダイアログの赤枠で括ったアイコンをクリックすると、“プライマリアクセスキー”がクリップボードにコピーされる

4. Azure Web API 作成



実習内容

1.
力実習

2.
クラウドへの
接続テスト

Win10 for Raspberry PI 2



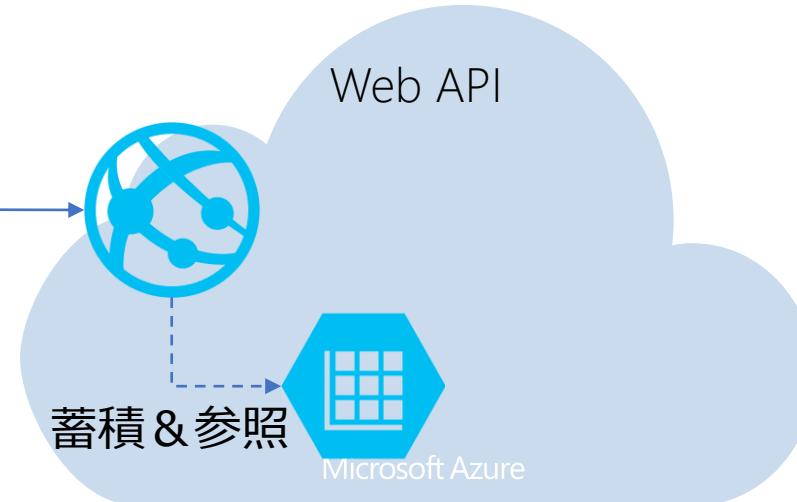
5.
LED遠隔操作

実習範囲

3.
Azure Storage 作成

4.
Azure Web API 作成

取得
送信

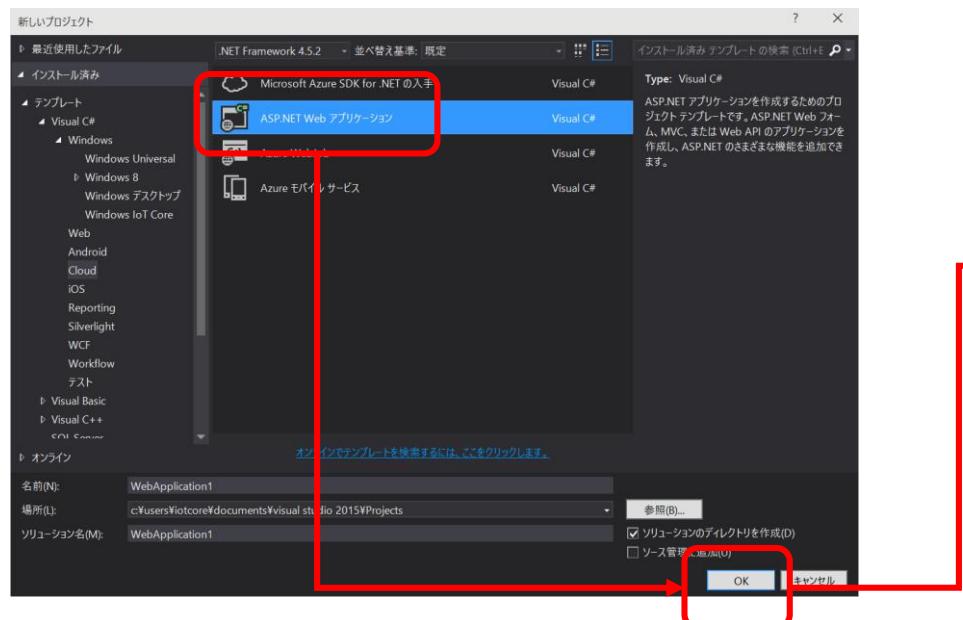


6.
温度通知

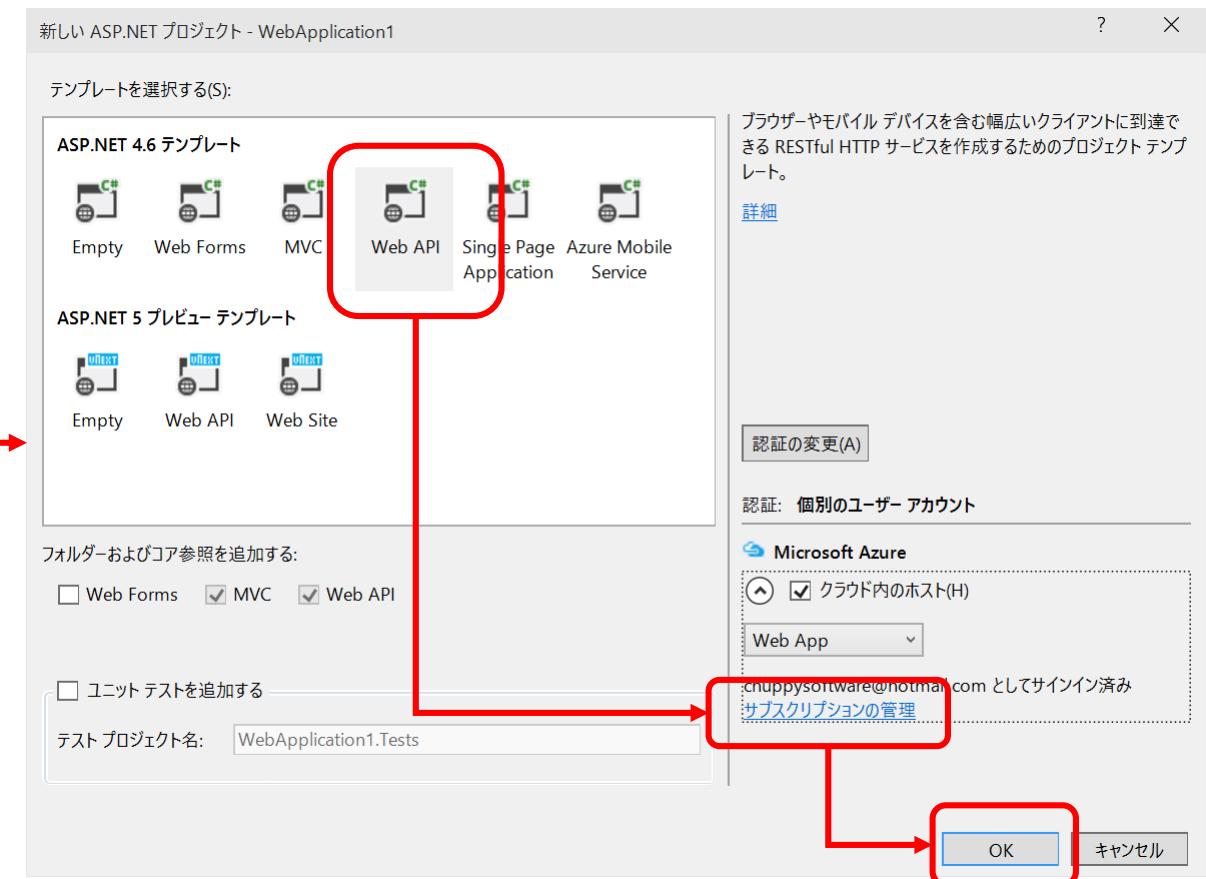
ASP.NET Web アプリケーションプロジェクト

Visual Studio 2015 RCを起動
“新しいプロジェクト”→・・・

C#→Cloud の“ASP.NET Webアプリケーション”を選択

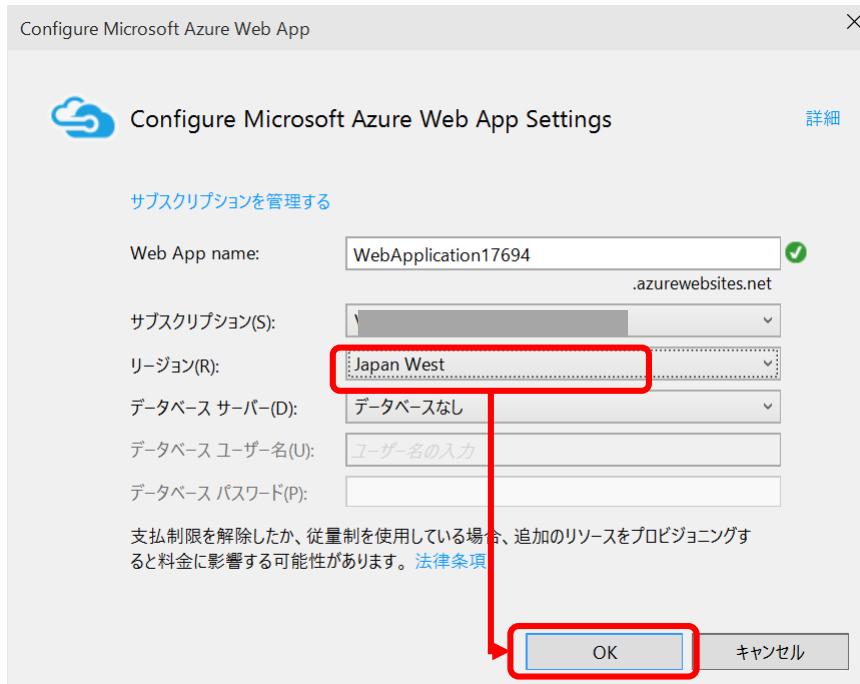


“Web API”を選択→“サブスクリプションの管理”で各自のアカウントでサインイン



続き

“リージョン”：“Japan West”を選択し、“OK”クリック

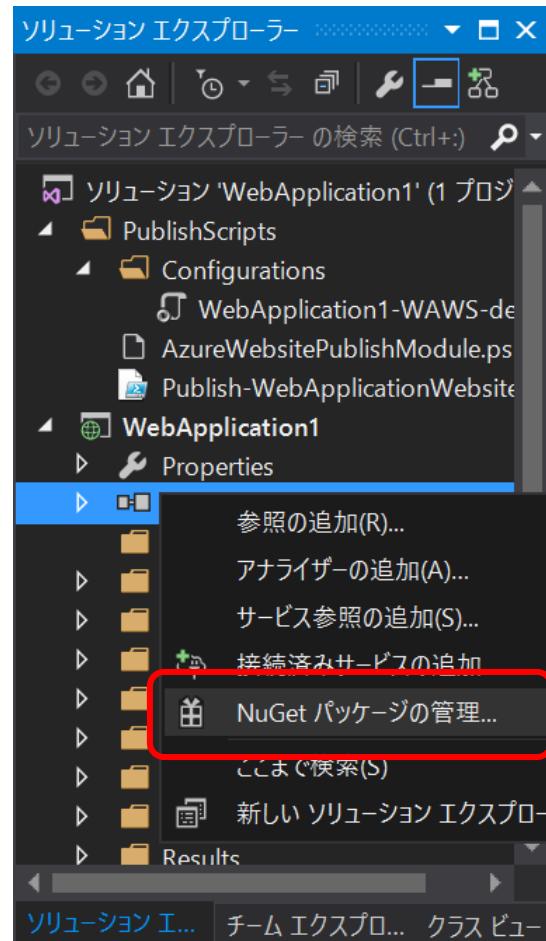


“Web App name”は後で使うので、記録
※名前は各自違います

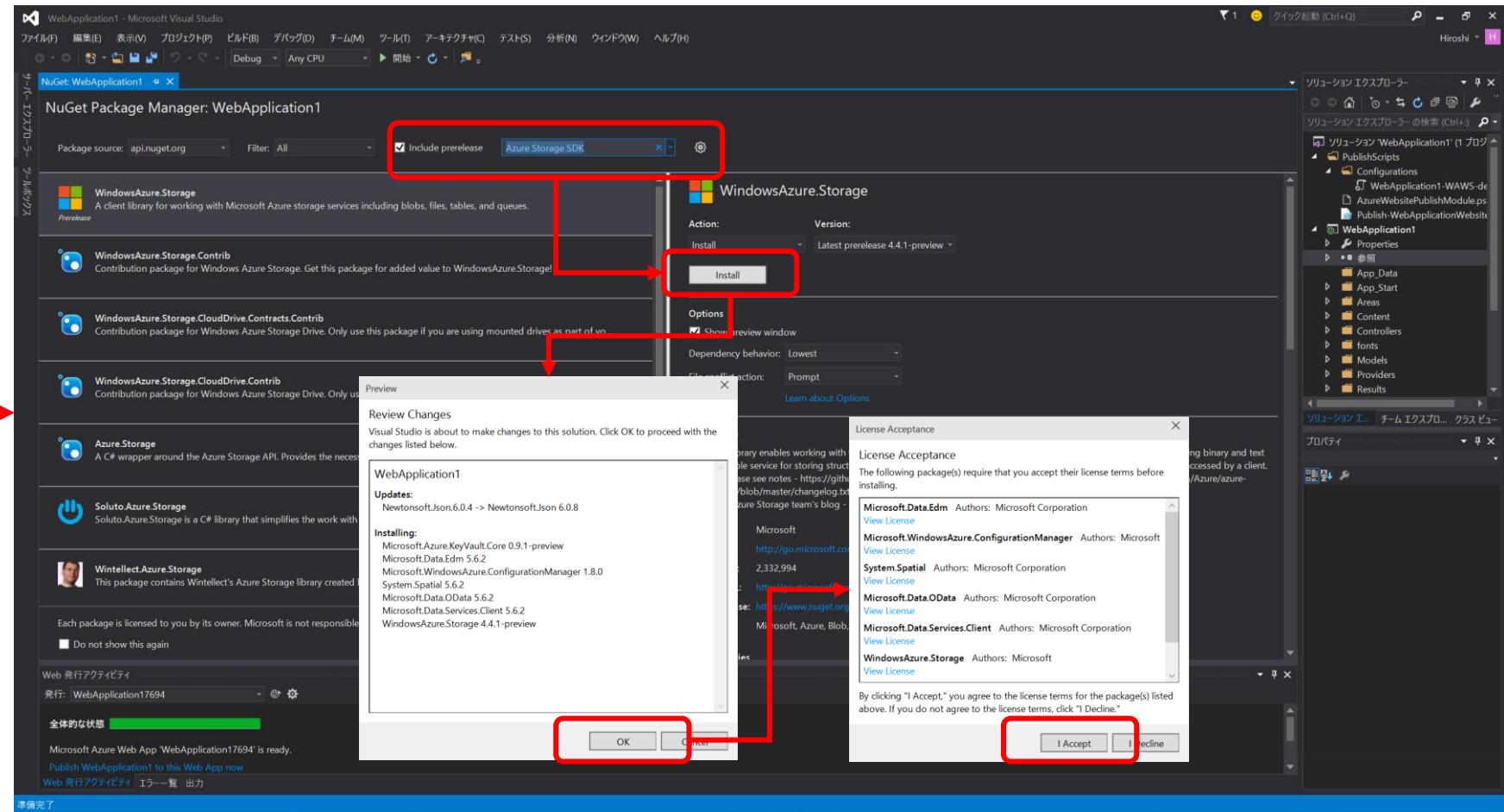
Visual Studio のWebアプリプロジェクト作成
Microsoft Azure上でのWebアプリ作成

Azure Storage SDKのインストール

“参照”を右クリックし、
“NuGetパッケージの管理”を選択



“検索”窓に“Azure Storage SDK”と入力し、
表示された項目の“Install”をクリック



Web.configへの設定

Visual Studio で、 Web.configを開く

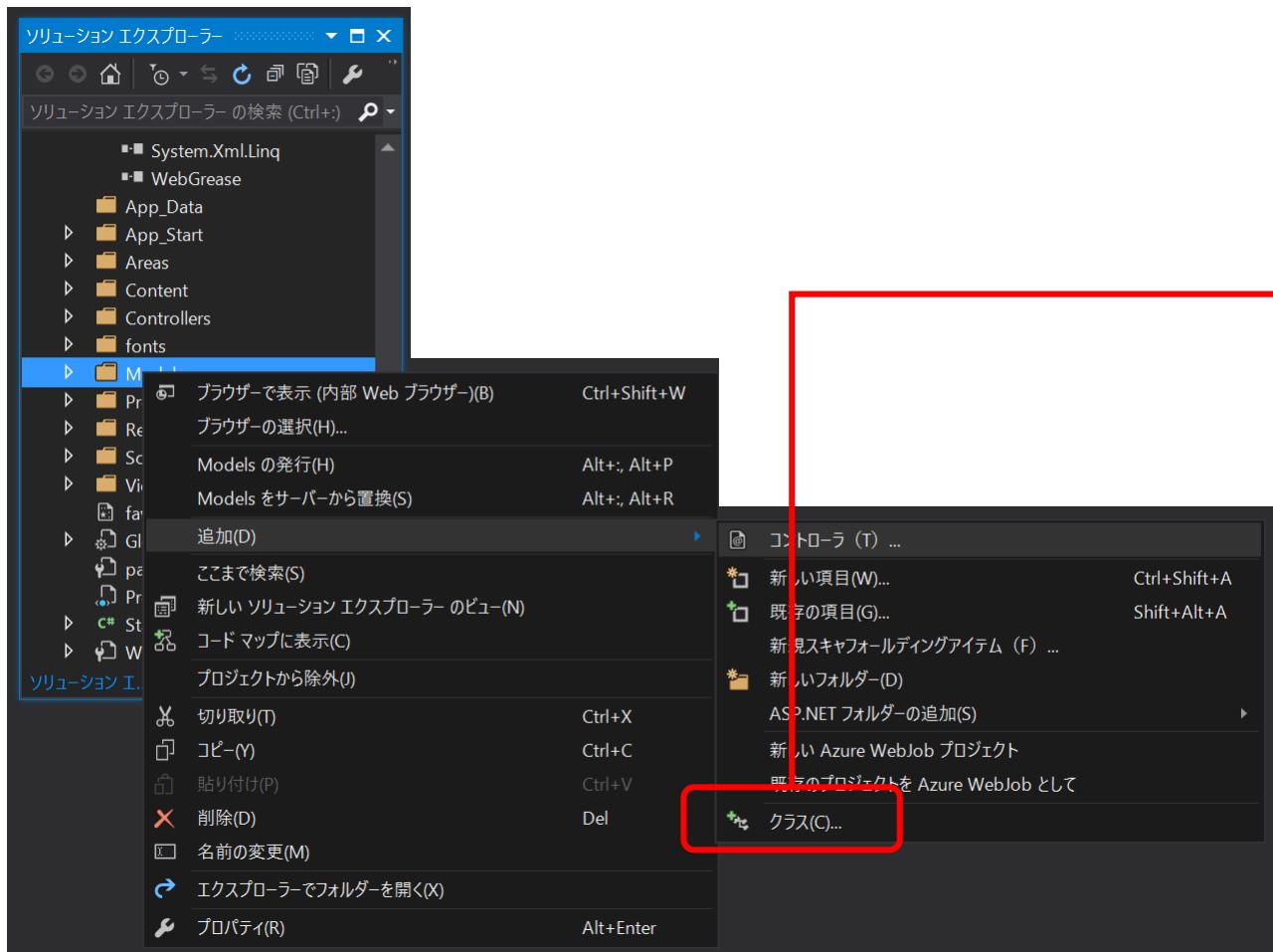
Web.configファイルに、 StorageConnectionStringの設定を追加
ソリューションエクスプローラーで、 Web.configをダブルクリックし、 エディタで表示

```
<appSettings>
<add key="StorageConnectionString" value="DefaultEndpointsProtocol=https;AccountName=[account-name];AccountKey=[access-key]" />
</appSettings>
```

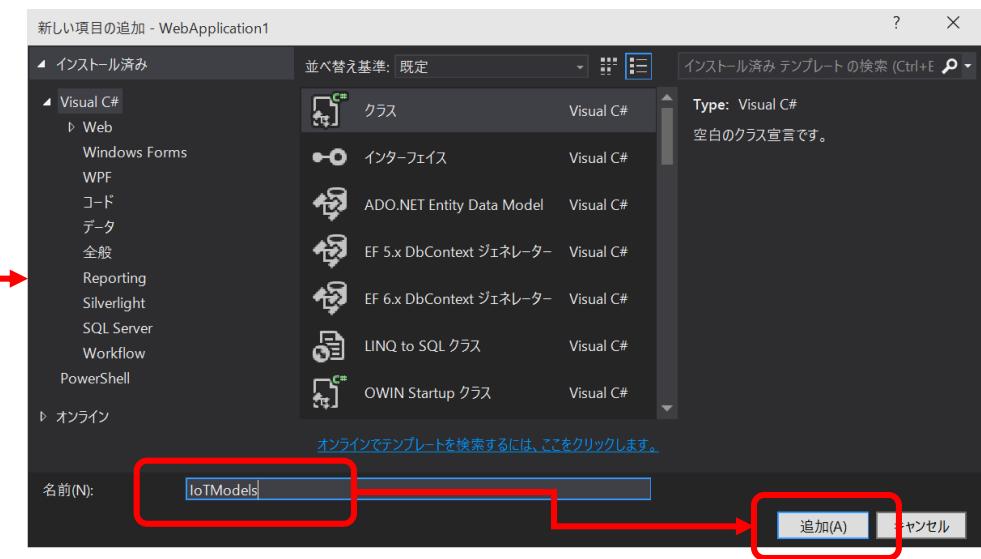
太字で示した、 "<add key="StorageConnectionString" .../>" の行を追加
[account-name]、 [access-key]を、 Azureの管理ポータルで作成したストレージの
ストレージアカウント名とアクセスキーで、 "["、 "]" 毎置き換えてください。

データモデルの追加

ソリューションエクスプローラで、
“Models”フォルダーを右クリックし、
“追加”→“クラス”を選択

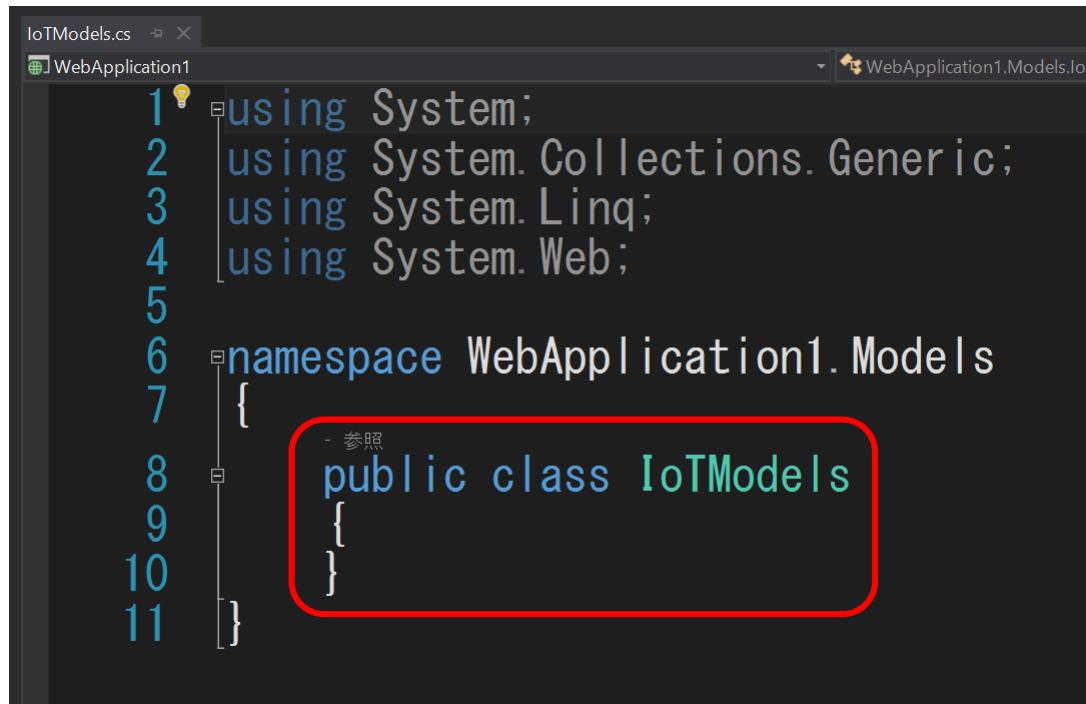


“IoTModels”という名前で
クラスファイルを作成



コードの置き換え

作成したIoTModels.csを開く



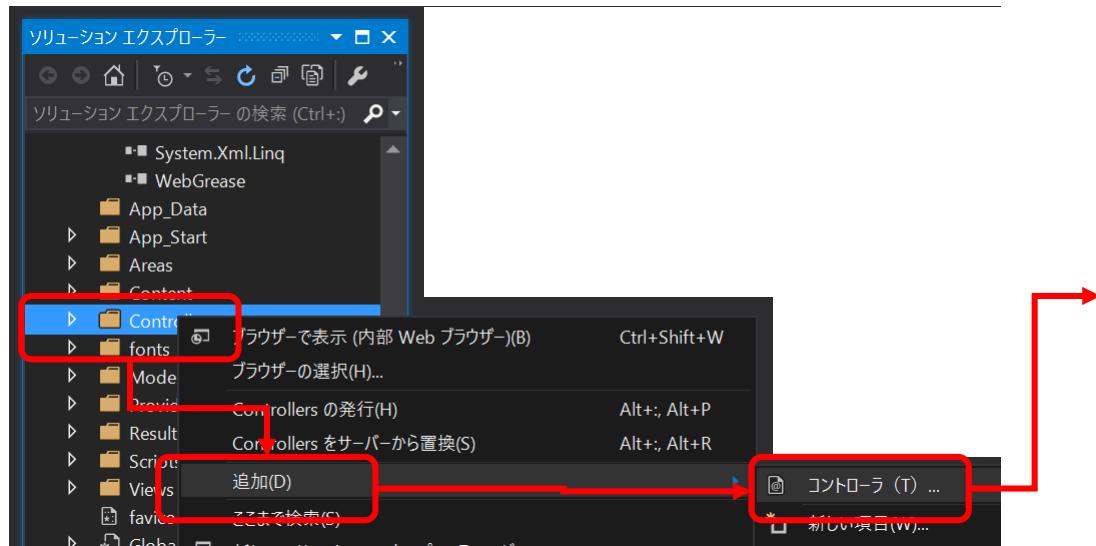
```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5
6 namespace WebApplication1.Models
7 {
8     public class IoTModels
9     {
10    }
11 }
```

public class IoTModels {...}
の部分を、

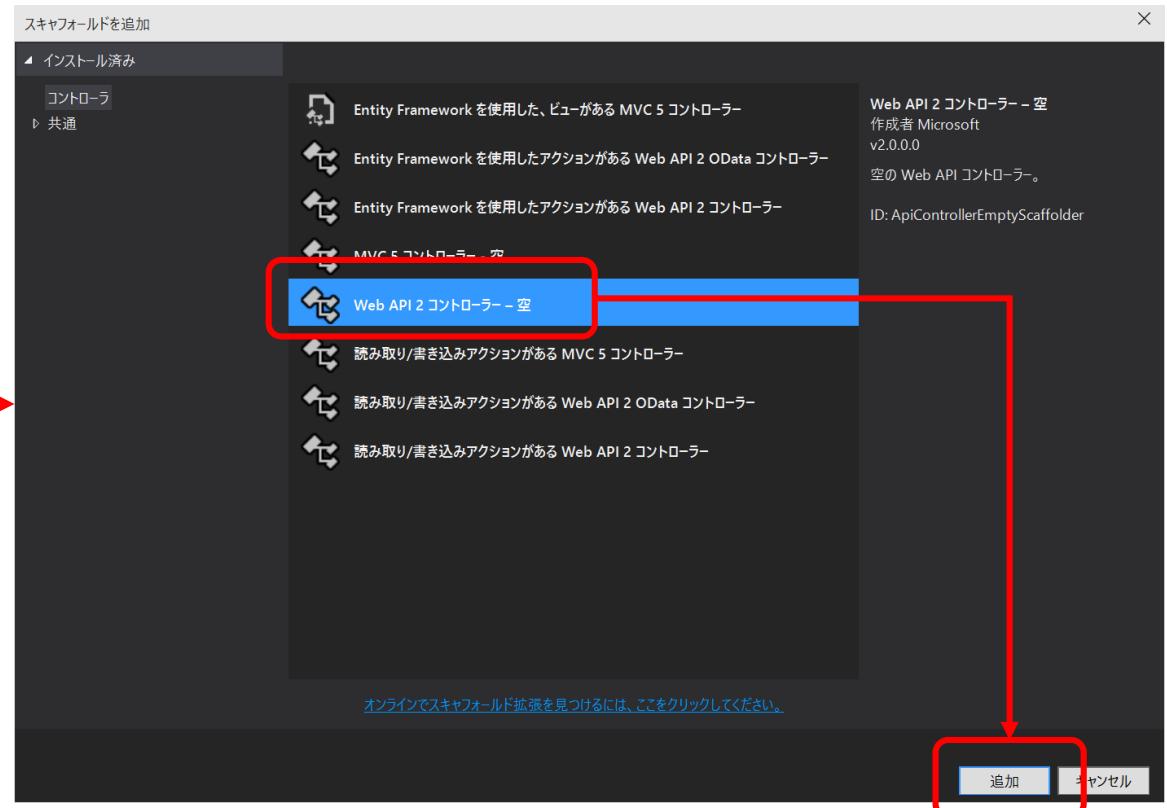
<http://aka.ms/Win10WebModel>
の中身で上書き

API コントローラーの作成

“Controllers”フォルダーを右クリック
“追加”→“コントローラ”を選択



“Web API 2 コントローラー – 空”選択
“追加”をクリック



“コントローラー名”を“Win10IoTController”と入力
“追加”をクリック

コードの書き換え

```
namespace WebApplication1.Controllers
{
    public class Win10IoTController : ApiController
    {
    }
}
```

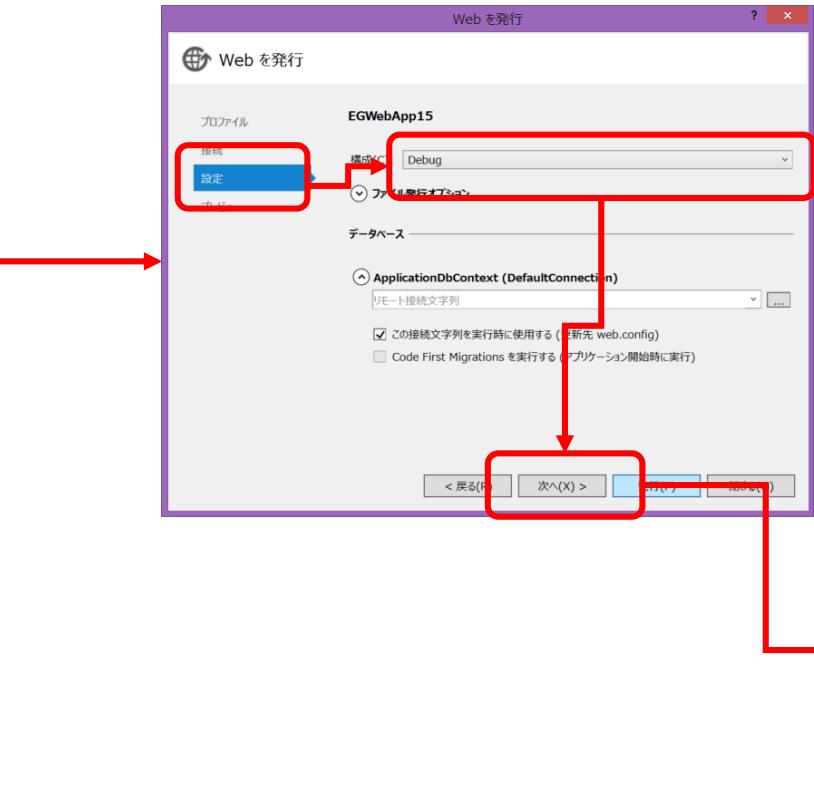
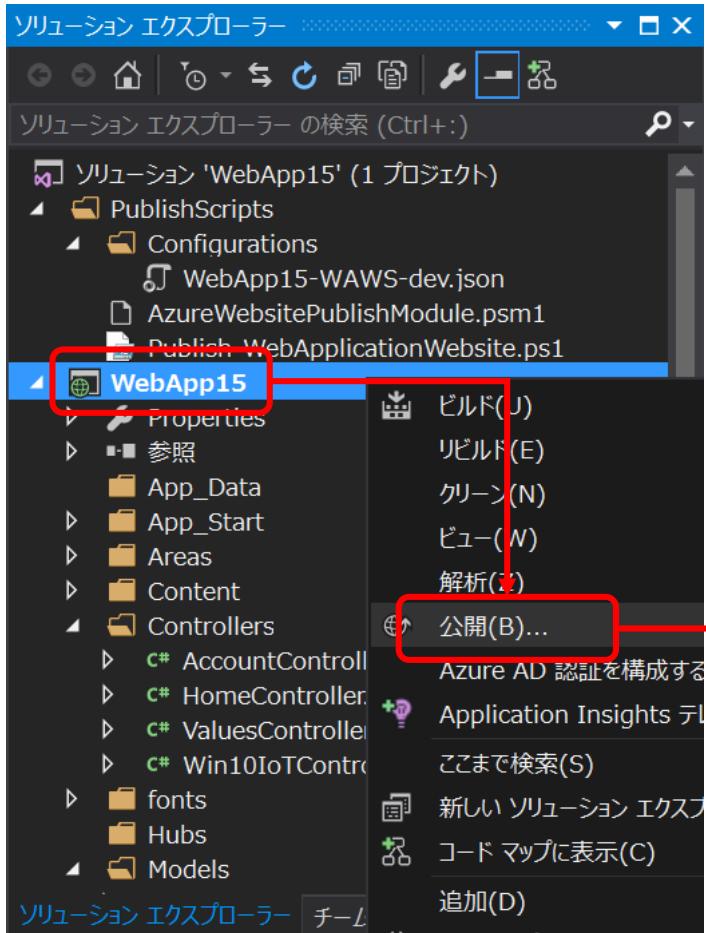
public class Win10IoTController : ...} の部分を、

<http://aka.ms/Win10WebAPI>

の中身で上書き

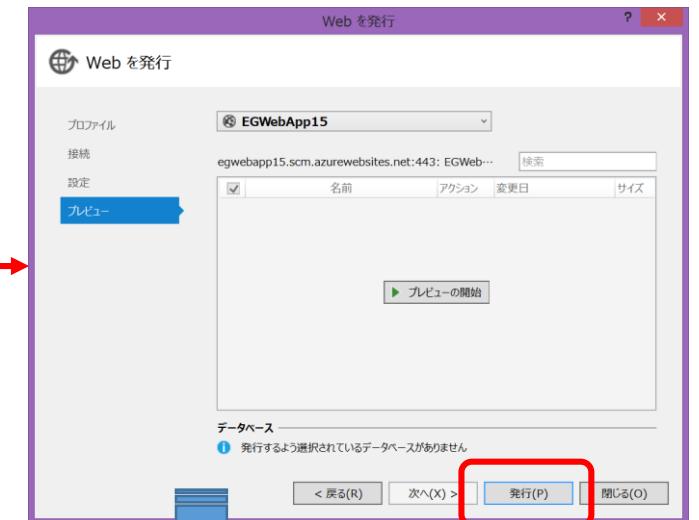
Cloud への発行

プロジェクトを
右クリックし、
“公開”を選択

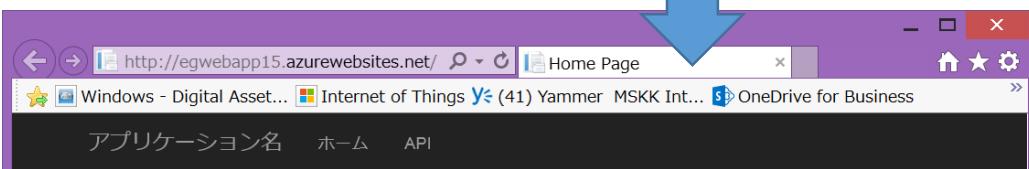


“設定”をクリック
← 構成を“Debug”に変更
“次へ”をクリック

“発行”をクリック



ブラウザが開く



5. LED遠隔操作



実習内容

1.
ルチカ実習

2.
クラウドへの
接続テスト

Win10 for Raspberry PI 2



実習範囲

3.
Azure Storage 作成

4.
Azure Web API 作成

取得
送信

Web API

蓄積 & 参照

Microsoft Azure

5.
LED遠隔操作

6.
温度通知

Web APIのエンドポイント設定

Raspberry PI2のVisual Studio で実施

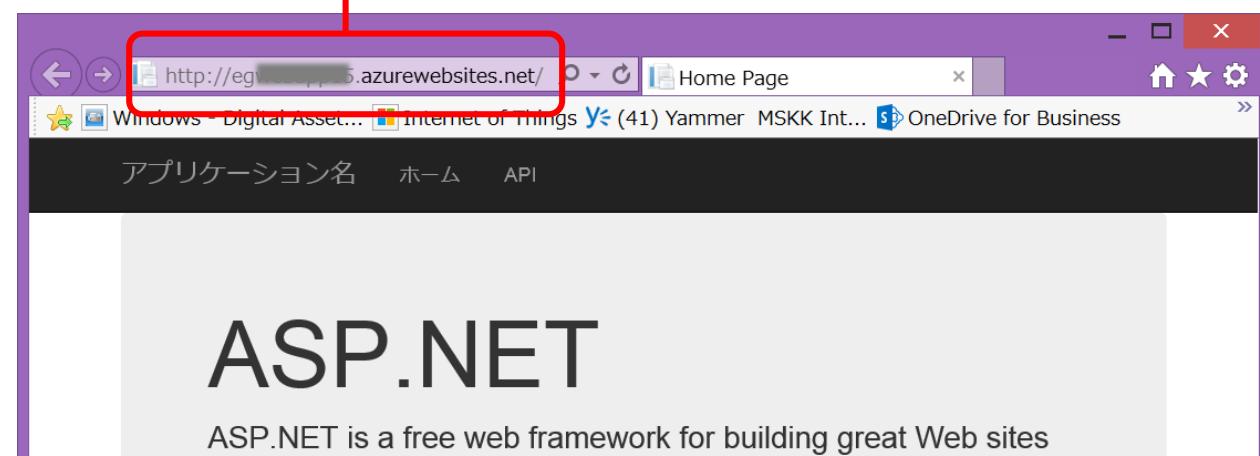
MainPage.xaml.csの

webApiEndPointの値を、前ステップで作ったWeb APIのURLに変更する

```
32 |     private string webApiEndPoint = "[WEB API URL]"  
33 |     Guid deviceId = new Guid("1E5570BD-350F-4C30-99E3")
```

Cloudへの発行の際、表示された
ブラウザのURLをコピー

この設定で、RP2から
Azure上のWeb APIアプリへの
アクセスが可能になる



コメントアウトを切り替える

LEDRemoteControl()の前の“//”を削除

The diagram illustrates the refactoring of a C# code snippet. On the left, the original code has several lines starting with double slashes (//) followed by method names. A large blue arrow points to the right, indicating the transformation. On the right, the code is shown with the double slashes removed, meaning the lines are now executable.

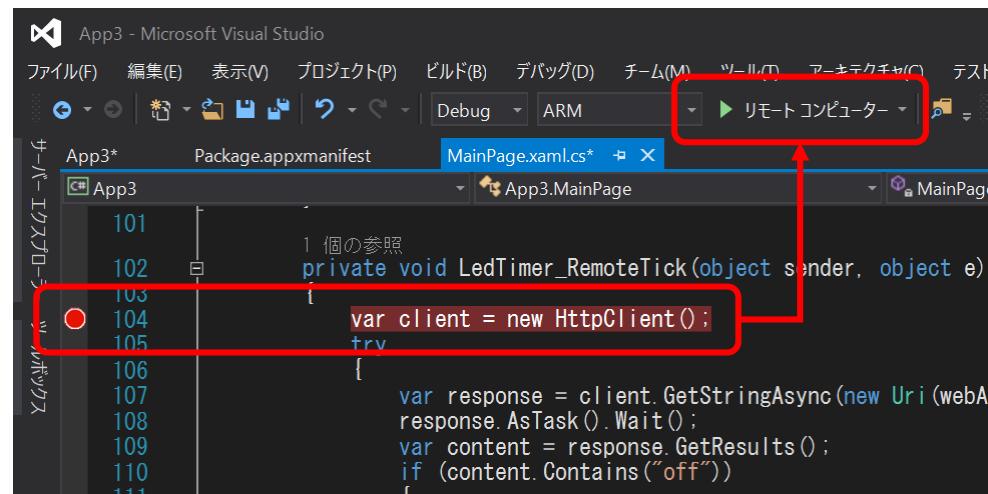
```
60     private void MainPage_Loaded(object sender, RoutedEventArgs e)
61     {
62         TestConnect().Wait();
63         // StartFlipLED();
64         // LEDRemoteControl();
65         // StartThmtSensing();
66     }
67 
```

```
60     private void MainPage_Loaded(object sender, RoutedEventArgs e)
61     {
62         TestConnect().Wait();
63         StartFlipLED();
64         LEDRemoteControl();
65         StartThmtSensing();
66     }
67 
```

Visual StudioでRaspberry Pi2とクラウドのアプリを実行確認

Raspberry PI2とWebアプリの実行確認

Win10 RP2側のVS2015



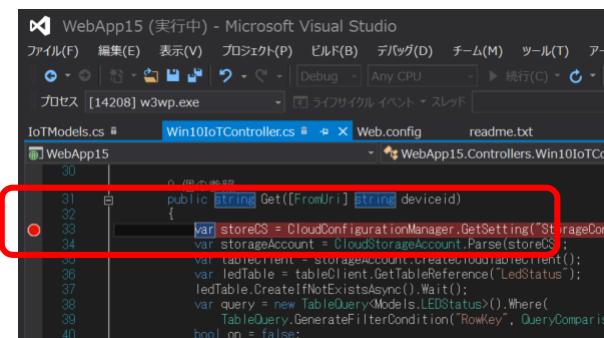
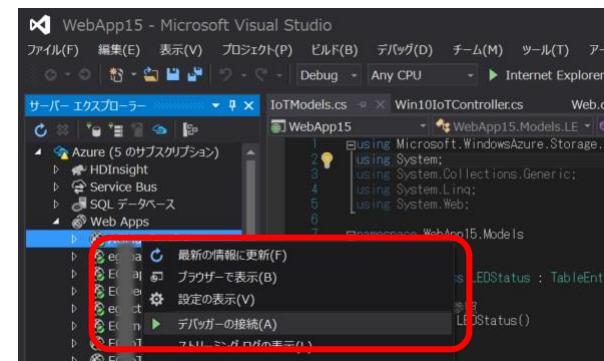
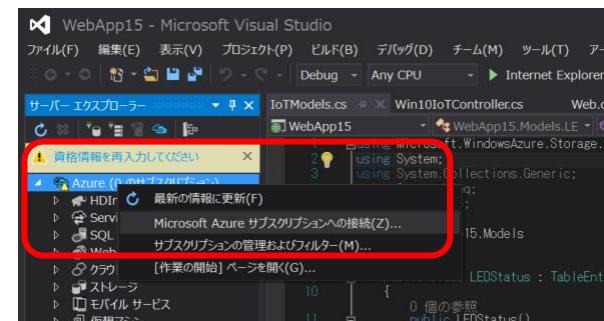
```
App3 - Microsoft Visual Studio
...
リモートコンピューター
...
private void LedTimer_RemoteTick(object sender, object e)
{
    var client = new HttpClient();
    try
    {
        var response = client.GetStringAsync(new Uri(webAp
        response AsTask().Wait();
        var content = response.GetResults();
        if (content.Contains("off"))
    }
}
...

```

LedTimer_RemoteTickメソッドの
"var client = new HttpClient()"の行に
ブレークポイントを設定し、起動

※ブレークポイント設定
図の左端の赤丸付近をマウスでクリック
もしくは、カーソルを行に移動し、F9を押す

Azure Webアプリ側のVS2015



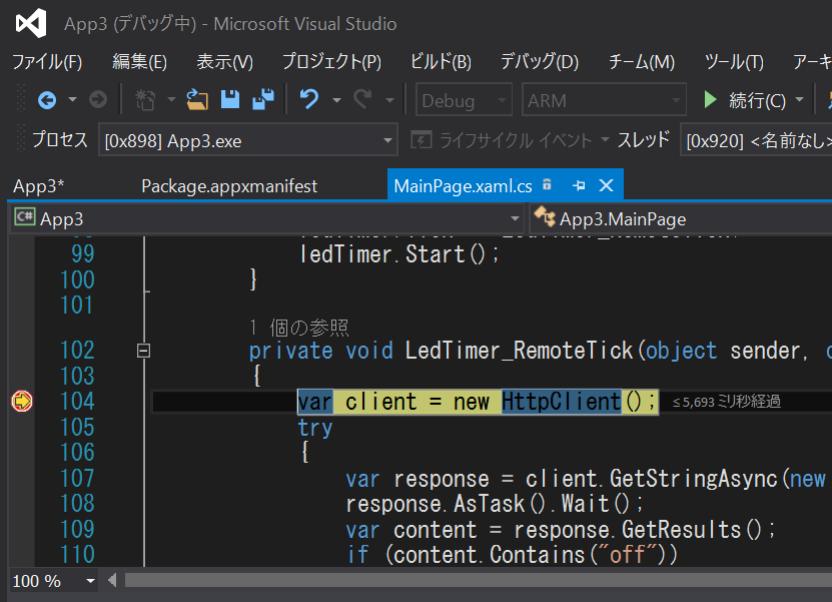
サーバーエクスプローラーで
Azureの接続を更新
"Microsoft Azure サブスクリプ
ションへの接続"を選択

"Azure"→"Web Apps"を展開し、
作成したWebアプリを右クリック
し、"デバッガーの接続"を選
択

"Win10IoTController.cs"ファイル
の、Getメソッドの
"var storeCS=Cloud..."の行に
ブレークポイントを設定

動作実行確認

Win10 RP2側のVS2015



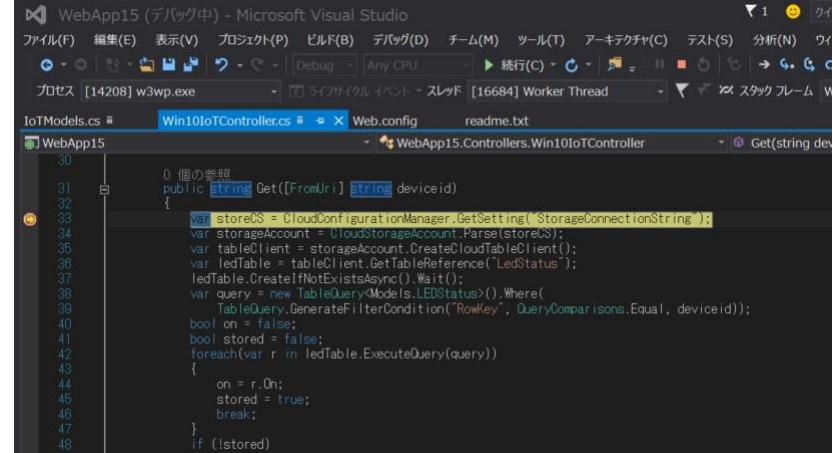
```
App3 (デバッグ中) - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) アーキテクチャ(C) テスト(S) 分析(N) ウィンドウ(W)
プロセス [0x898] App3.exe ライフサイクルイベント スレッド [0x920] <名前なし>
App3* Package.appxmanifest MainPage.xaml.cs
App3
    99     ledTimer.Start();
    100
    101
    102     private void LedTimer_RemoteTick(object sender, EventArgs e)
    103     {
    104         var client = new HttpClient();
    105         try
    106         {
    107             var response = client.GetStringAsync(new Uri("http://192.168.1.10:8080/api/ledstatus?deviceid=1"));
    108             response.AsTask().Wait();
    109             var content = response.GetResults();
    110             if (content.Contains("off"))
    111             {
    112                 // LEDをオフにする操作
    113             }
    114         }
    115     }
    116 }
```

ブレークしたら、F10 キーでステップ実行

"var response = client.GetStringAsync..."
の行でF10キーを押下すると

"response.AsTask().Wait()"でブロックされた制御が戻り、
続きのコードが実行される
"content"変数にLEDの点灯指示が格納される。
If文の条件判定を確認し、F5で継続実行→5秒後にブレーク

Azure Webアプリ側のVS2015



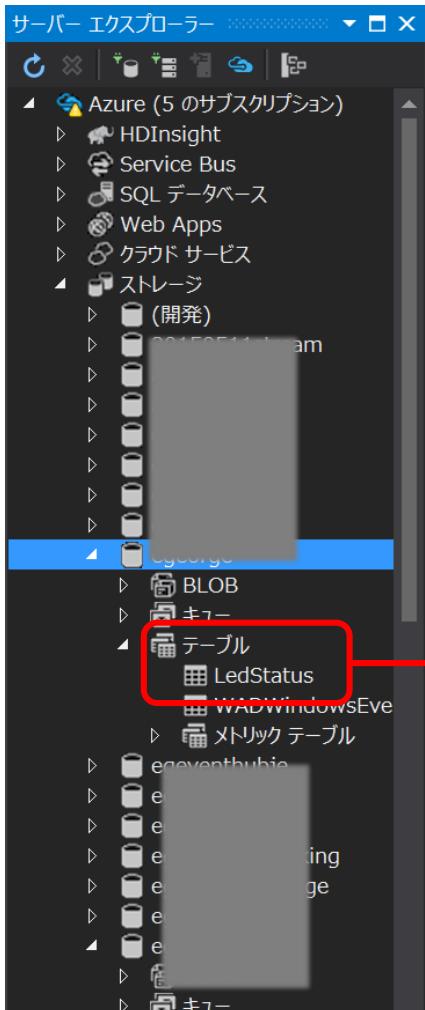
```
WebApp15 (デバッグ中) - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) アーキテクチャ(C) テスト(S) 分析(N) ウィンドウ(W)
プロセス [14208] w3wp.exe ライフサイクルイベント スレッド [16684] Worker Thread
IoTModels.cs WebApp15.Controllers.Win10IoTController Web.config readme.txt
WebApp15
    30     0 個の参照
    31     public string Get([FromUri] string deviceId)
    32     {
    33         var storeConnectionString = CloudConfigurationManager.GetSetting("StorageConnectionString");
    34         var storageAccount = CloudStorageAccount.Parse(storeConnectionString);
    35         var tableClient = storageAccount.CreateCloudTableClient();
    36         var ledTable = tableClient.GetTableReference("LedStatus");
    37         ledTable.CreateIfNotExistsAsync().Wait();
    38         var query = new TableQuery<Models.LEDstatus>().Where(ledgerTable.Query.GenerateFilterCondition("RowKey", QueryComparisons.Equal, deviceid));
    39         bool on = false;
    40         bool stored = false;
    41         foreach(var r in ledTable.ExecuteQuery(query))
    42         {
    43             on = r.On;
    44             stored = true;
    45         }
    46         if (!stored)
    47         {
    48             // LEDを初期化する操作
    49         }
    50     }
    51 }
```

Getメソッドのブレークポイントで実行停止
F10キーで、Getメソッドの最後までF10キーで
ステップ実行していく。全ての行が実行される
最後まで行くと...

※CreateIfNotExistAsyncメソッド
実行でLedStatusテーブル作成

「チカの遠隔制御

サーバーエクスプローラーで、
ストレージアカウントを展開し、
“LEDStatus”テーブルをクリック



PartitionKey	RowKey	Timestamp	On
LED	83c3d97a-e9…	2015/05/30 1…	False

“On”を“True”、“False”で変える

RP2のLEDが、それに合わせて
点滅する

6. 温度通知



実習内容

1.
Iチカ実習

2.
クラウドへの
接続テスト

Win10 for Raspberry PI 2



5.
LED遠隔操作

実習範囲

3.
Azure Storage 作成

4.
Azure Web API 作成

取得
送信

Web API

蓄積 & 参照



Microsoft Azure

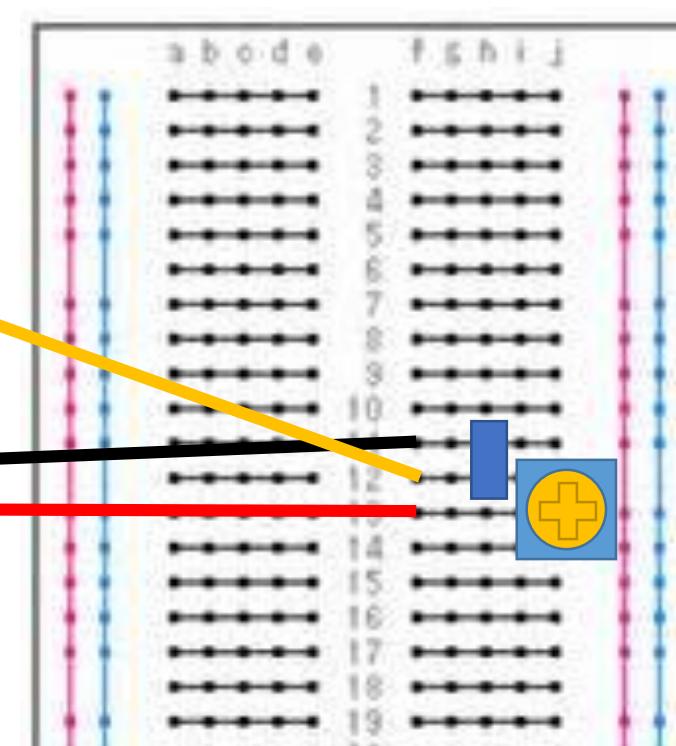
6.
温度通知

サーミスターの接続

ある温度を超えるとLow、下回るとHighを検知
半固定抵抗を調整することにより温度を調整可能



3.3V PWR	1
I2C1 SDA	3
I2C1 SCL	5
GPIO 4	7
GND	9
SPI1 CS0	11
GPIO 27	13
GPIO 22	15
3.3V PWR	17
SPI0 MOSI	19
SPI0 MISO	21
SPI0 SCLK	23
GND	25
Reserved	27
GPIO 5	29
GPIO 6	31
GPIO 13	33
SPI1 MISO	35
GPIO 26	37
GND	39



#30 GND
#12 GPIO 18
#2 5 V PWR

LEDは外してください

コメントアウトを外す

InitThmtGPIO()の先頭の“//”を外し、
InitLEDGPIO()の先頭に“//”を付与する

```
36     public MainPage()
37     {
38         this.InitializeComponent();
39
39         this.Loaded += MainPage_Loaded;
40         InitLEDGPIO();
41         // InitThmtGPIO();
42     }
43 }
```



```
36     public MainPage()
37     {
38         this.InitializeComponent();
39
40         // this.Loaded += MainPage_Loaded;
41         // InitLEDGPIO();
42         InitThmtGPIO();
43     }
44 }
```

Raspberry PI2のVisual Studio で実施

StartThmtSensing()の先頭の“//”を外し、
LedRemoteControl()の先頭に“//”を付与する

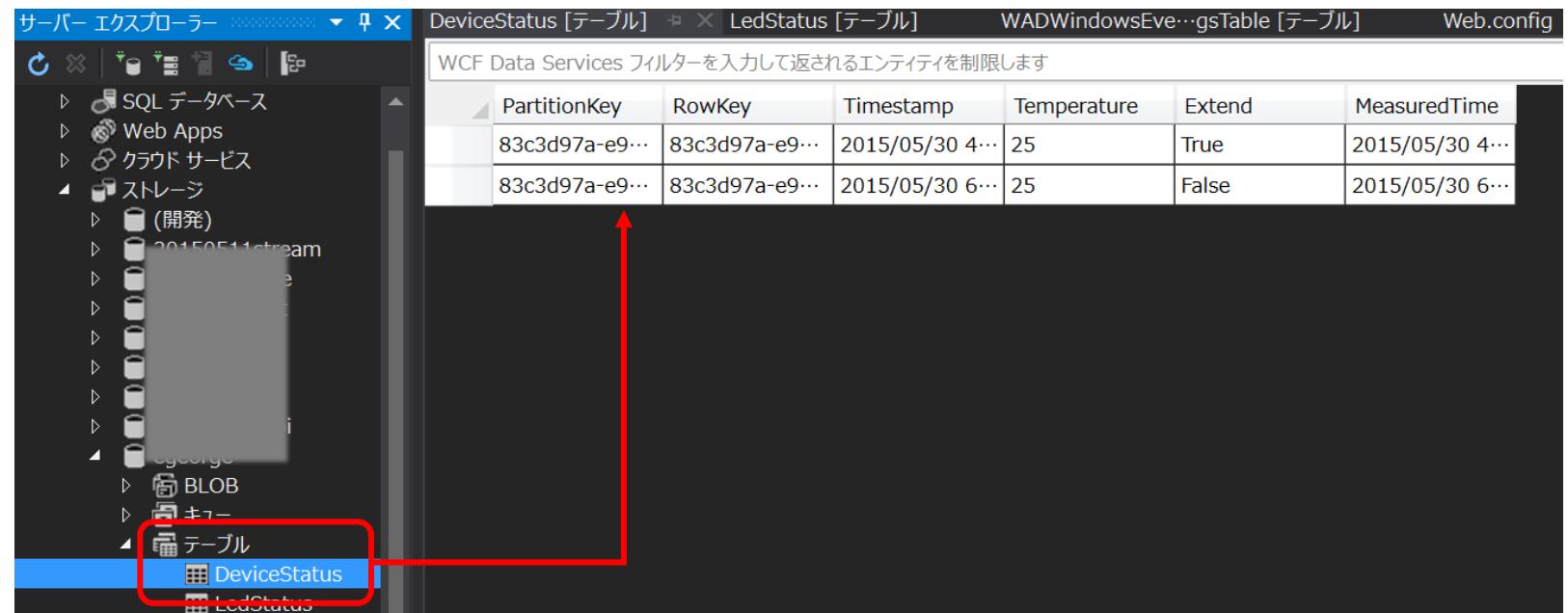
```
60     private void MainPage_Loaded(object send
61     {
62         TestConnect().Wait();
63         // StartFlipLED();
64         LEDRemoteControl();
65         // StartThmtSensing();
66     }
67 }
```



```
60     private void MainPage_Loaded(object send
61     {
62         TestConnect().Wait();
63         // StartFlipLED();
64         // LEDRemoteControl();
65         StartThmtSensing();
66     }
67 }
```

Visual Studio で送信された内容を確認

Visual Studio で
アプリを実行



Visual Studio で送信データ確認

実習内容

1.
Iチカ実習

2.
クラウド接続
Wi-Fi

実習範囲

3.
Azure Storage 作成

4.
Azure Web API 作成

- Iチカ
- クラウド上の設定 → Iチカ制御
- 温度の上下 → クラウド上の蓄積

+
LED
サーミスター



5.
LED遠隔操作

6.
温度通知

継続学習を是非

Microsoft Internet of Things Windows for IoT IoT Kit Hands-on Home Downloads Contact Us

Internet of Things キットハンズオントレーニング

IoT 実践に必要なスキルを獲得!!
本物のセンサー搭載ハードウェア (.NET Micro Framework) と
クラウド (Microsoft Azure) で、ソフトウェアを
Visual Studio で開発しながら、体系的に開発スキルを獲得。
※ 学習コンテンツ公開中!! ※

<http://aka.ms/IoTKitHoL>

.NET Micro Framework + Sensors Hardware Kit

Microsoft Azure



事前準備

トレーニング概要

さあ、始めよう

ハードウェア購入、開発環境準備

トレーニングの内容と学習方法

トレーニングを始める。

参考URL

- <http://www.WindowsOnDevices.com>
 - Windows IoT Core ポータル
- <http://blogs.msdn.com/hirosho>
 - 講師のブログ
- <https://www.facebook.com/groups/ioytjp/>
 - IoTあるじゃんサイト

お疲れ様でした

- ・アンケートをお願いします
- ・IoTあるじゃんよろしくお願ひします
- ・ハンズオン是非!!