Microsoft

Microsoft Azure 自習書シリーズ

アプリケーション開発–PaaS 基礎編

この自習書では、Microsoft が提供するパブリッククラウドサービスである Microsoft Azure を利用し、Azure App Services を利用したアプリケーションの作成から操作までの一連の流れをハンズオン形式で学習体験します。

発行日 : 2019 年 1 月 28 日

更新履歴

版数	発行日	更新履歴
第 1 版	2017 年 7 月 21 日	初版発行
第 1.1 版	2017 年 8 月 24 日	App Service – Premium v2, Isolated レベルを追加
第 1.2 版	2017 年 9 月 4 日	画像を一部差し換え
第 1.3 版	2017 年 10 月 30 日	Azure Storage の作成メニュー更新
第 1.4 版	2017 年 11 月 29 日	Azure Storage 、 Azure SQL Database の作成メニュー更新 Visual Studio 上の Application Insights メニュー更新
第 1.5 版	2017 年 12 月 26 日	Storage Account v2 一般提供開始に伴い更新
第 1.6 版	2018 年 1 月 26 日	Azure 無料評価版-価格変更に伴い更新
第 1.7 版	2018 年 2 月 26 日	Azure Storage 仮想ネットワーク GA に伴いメニュー更新
第 1.8 版	2018 年 3 月 26 日	Visual Studio 2017 更新 (15.6) に伴い画像を差し替え
第 1.9 版	2018 年 5 月 28 日	Azure App Service – Portal メニュー更新に伴い更新
第 1.10 版	2018 年 11 月 07 日	Azure ポータル更新に伴い更新
第 1.11 版	2019 年 1 月 28 日	Azure ポータル更新に伴い更新

目次

1. はじめに

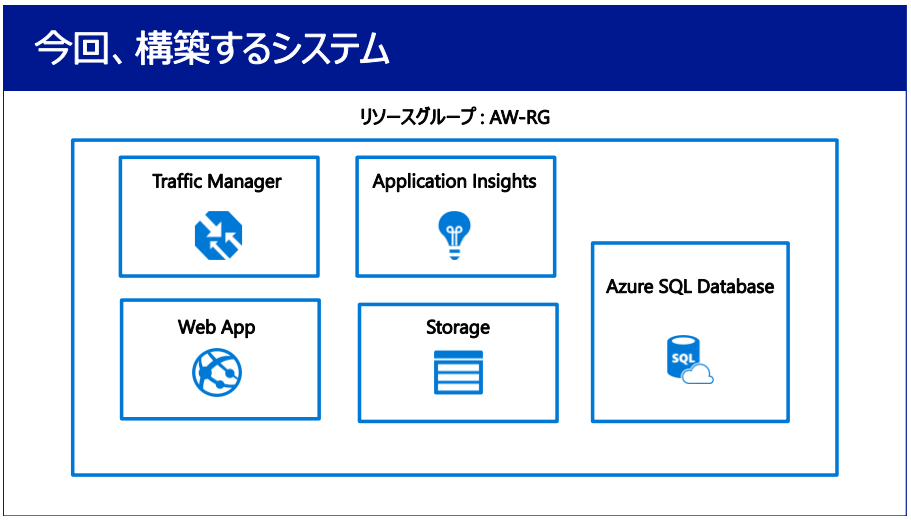
本自習書をご利用いただきありがとうございます。この自習書では、Microsoft が提供するパブリッククラウドサービスである Microsoft Azure を利用し、Azure 環境で動作する社内アプリケーションの作成から操作までの一連の流れをハンズオン形式で体験学習します。

自習書において、あなたは Adventure Works Cycles 社（AW 社）に勤めている IT 管理者です。AW 社は、架空の大規模な多国籍製造企業です。この企業は、北米、ヨーロッパ、およびアジアのマーケットを対象に、金属製自転車やカーボン製自転車の製造および販売を行っています。従業員 290 人の米国ワシントン州ボセルの拠点に加え、自社のマーケット基盤全体にわたって複数の地域販売チームを配置しています。

Adventure Works Cycles とは

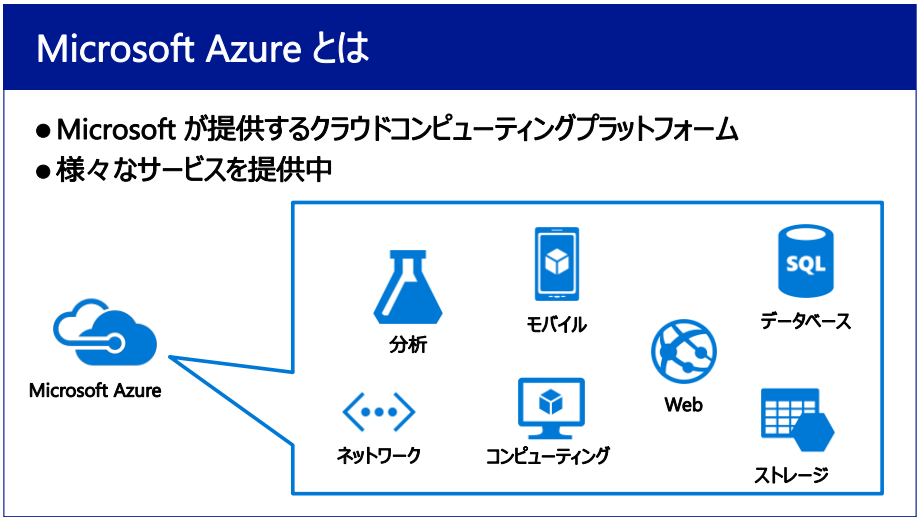
- **架空の企業です。**
- **この企業は、北米、ヨーロッパ、およびアジアのマーケットを対象に、金属製やカーボン製の自転車の製造および販売を行っています。**
- **従業員 290 人の米国ワシントン州ボセルの拠点に加え、自社のマーケット基盤全体にわたって複数の地域販売チームを配置しています。**

あなたは、Microsoft Azure の App Services を利用し、営業部向けの Web システムをテスト運用するため、そのインフラを構築しなければなりません。この Web システムは、Azure Web App と Azure Storage、SQL Database で実行されるアプリケーションサーバーから構成されています。また、可用性確保のため、Traffic Manager を使用します。合わせて、全体の可用性を Application Insights で監視します。

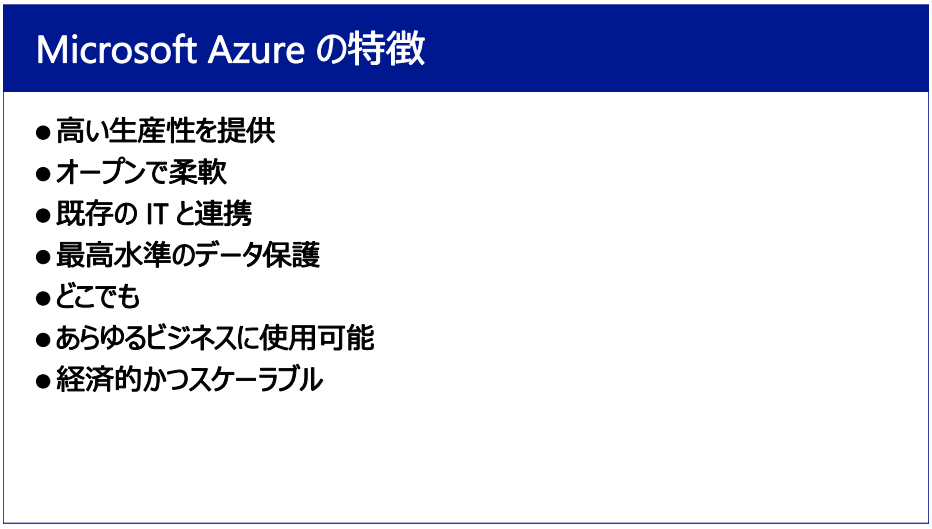


2. Microsoft Azure の概要

まず、基礎知識として、Microsoft Azure と App Services の概要を学習します。



Microsoft Azure は、Microsoft が提供するオープンで柔軟なエンタープライズレベルのクラウドコンピューティングプラットフォームです。Microsoft Azure では、Web から、モバイル、コンピューティング、データベース、分析に至るまで、様々なクラウドサービスを提供しています。企業は、Microsoft Azure を利用することで、より早く、より多くのビジネス目標を達成でき、より経費を節約できます。



Microsoft Azure の特徴は、次のとおりです。

●高い生産性を提供

開発者や IT 管理者は、Microsoft Azure の統合された管理ツール、テンプレートによる展開、管理されたサービスを利用することで、より高い成果を生むことができます。

●オープンで柔軟

Microsoft Azure では、OS、プログラム言語、フレームワーク、ツール、データベース、デバイスにおいて、オープンで幅広い選択肢を用意しています。App Services では PaaS (Platform as a Service)環境として、Web アプリケーションのホスティング環境を提供していますが、.NET アプリケーションだけでなく、Java, Node.js, Python など、幅広い実行環境を提供しています。

●最高水準のデータ保護

Microsoft Azure は、データの保護とプライバシーに関する業界最高水準のコミットメントを受けています。Microsoft は、クラウド プライバシーに関する新たな国際標準である ISO 27018 を採用した最初のメジャークラウドプロバイダーです。

●どこでも

Microsoft Azure は、世界 54 の場所でデータセンターを稼働させています。また、今後もデータセンターの数は増える予定です。日本においては、東日本（埼玉）と西日本（大阪）の 2 つのデータセンターが稼働中であり、日本国内からの利用時に、高いパフォーマンスが確保されます。

●あらゆるビジネスに使用可能

Microsoft Azure は、予算に合わせて利用することができます。例えば、小規模なテスト環境から大規模なオンラインゲーム環境の提供まで幅広く利用可能です。

●経済的かつスケーラブル

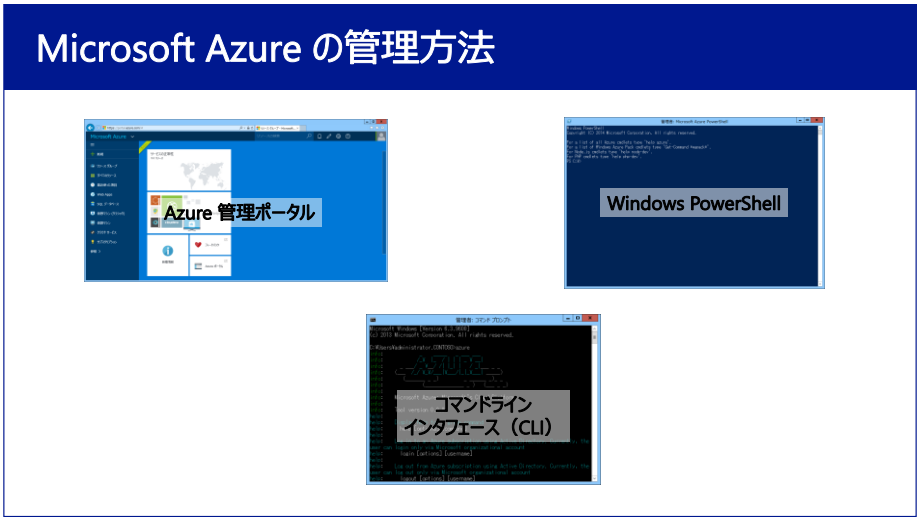
Microsoft Azure は従量課金です。そのため、支払いは使用した分だけとなります。例えば、仮想マシンの実行は、分単位で課金されるため、負荷に合わせて、適切なスケールアップやスケールダウンをおこなえば、パフォーマンスを低下することなく、経費を節約できます。

App Services

- **Azure で提供される PaaS 環境**
- **Web アプリケーションをホスティングする Web Apps**
- **モバイルアプリのバックエンド機能を提供する Mobile Apps**
- **RESTful なAPIを提供するための API Apps**
- **コードを記述せずに自動化・データ統合を行うための Logic Apps**

Web Apps は、Microsoft Azure で提供される代表的なクラウドサービスであり、Microsoft Azure のデータセンターで実行される Web アプリケーションのホスティング環境です。OS のインストール、IIS などの Web サーバーの設定などをすることなく、短時間で Web アプリケーションをホストすることが可能です。

また、Web アプリケーションのホスティングだけではなく、Azure Active Directory との認証連携、自動スケーリング、運用監視などのサービスも合わせて提供されています。



Microsoft Azure を管理する主な方法として、次の 3 つがあります。

●Azure 管理ポータル

Web ベースの管理ポータルです。Azure リソースマネージャーによるリソース管理の概念が取り入れられています。また、シンプルでカスタマイズ性に優れています。https://portal.azure.com からアクセス可能です。

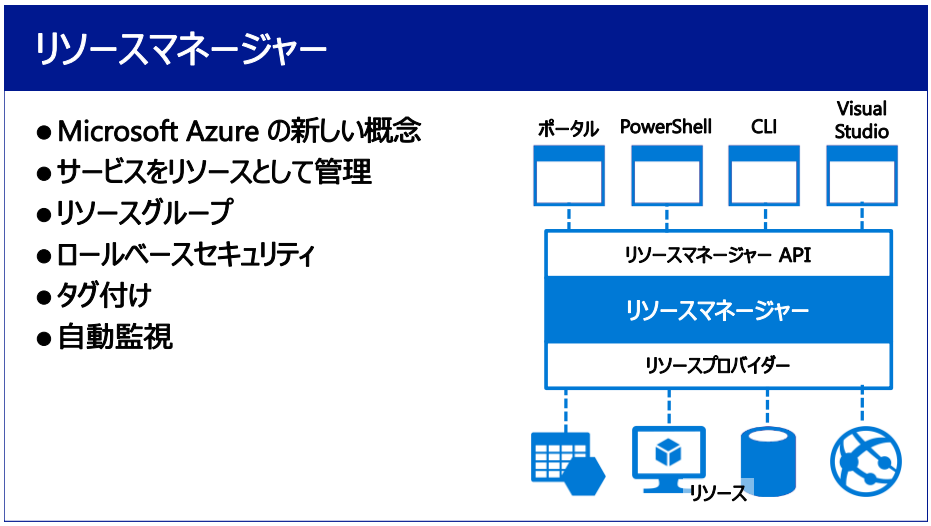
管理ポータルは HTML 5 で記述されているため、Internet Explorer をはじめ、Firefox、Chrome、Safari などの主要な Web ブラウザおよびデバイスで動作します。

●Windows PowerShell

スクリプトによる自動処理を実現する Windows PowerShell コマンドレット群です。一部の機能は、Windows PowerShell からのみ操作可能です。Windows PowerShell は、https://azure.Microsoft.com/ja-jp/downloads より入手可能です。

●コマンドラインインタフェース（CLI）

Microsoft Azure が操作可能なコマンドラインプログラムです。マルチプラットフォームに展開されており、Windows だけでなく、Mac や Linux 用も提供されています。コマンドラインインタフェースは、https://azure.Microsoft.com/ja-jp/downloads より入手可能です。



リソースマネージャーは、Microsoft Azure の新しい概念であり、クラウドサービスの構成要素をリソースとして分解し、個々に管理します。例えば、仮想マシンを作成すると、仮想マシンのリソースだけでなく、ネットワークインターフェイスやパブリック IP アドレスなどのリソースも一緒に作成されます。また、Azure リソースマネージャーでは、リソースに対して、次の管理機能も提供します。

●リソースグループ

複数のリソースをまとめてグループ化し、一元的に展開、管理、監視をおこないます。

●ロールベースセキュリティ

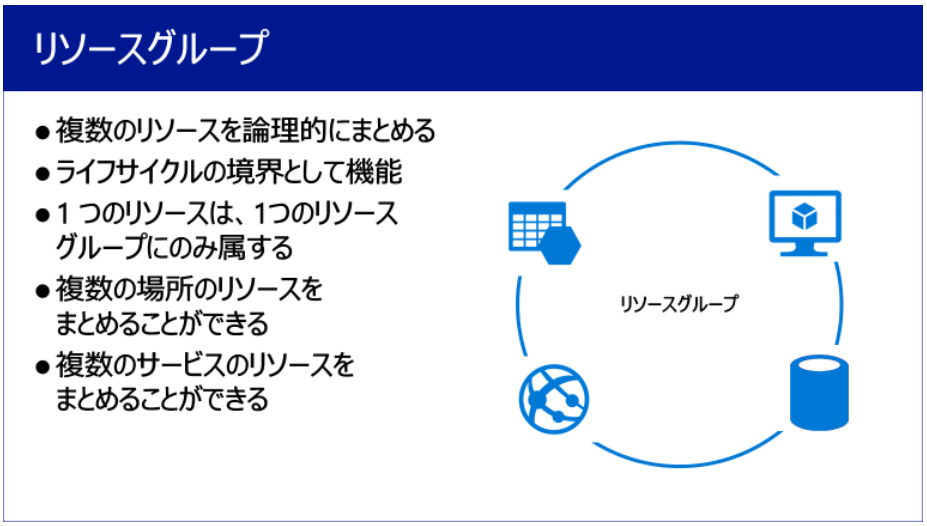
ユーザーやグループに対して、リソースまたはリソースグループごとに役割ベースのアクセス制御を提供します。

●タグ付け

タグは、キーと値のペアです。リソースにタグを付けることで、リソースを整理できます。最大 15 個のタグを付けることができます。特にタグ付けは、課金や管理の目的で使用されます。

●自動監視

パフォーマンスメトリックや各種イベントログを監視し、ストレージアカウントに記録します。また、仮想マシンのブート時の動作の監視（Windows のブート画面の定期的なキャプチャや Linux のブートログの記録）も可能です。



リソースグループは、リソースマネージャーが提供する機能であり、複数のリソースを論理的にまとめるグループです。リソースグループは、展開、更新、削除に至るライフサイクル管理の境界として機能します。リソースグループには次の決まり事があります。

・1 つのリソースは、1 つのリソースグループにのみ追加できます。1 つのリソースを複数のリソースグループに追加することはできません。

・データセンターの異なる複数の場所のリソースをまとめて、1 つのリソースグループに追加することができます。

・Web Apps やストレージアカウントなど異なるクラウドサービスのリソースをまとめて、1 つのリソースグループに追加することができます。

App Services で提供される機能

- **複数の言語をサポートするWeb アプリケーションの実行環境**
- **高可用性を備えた柔軟なスケーリング**
- **Azure Active Directory 等との認証統合**
- **運用環境におけるテスト**
- **利用状況のモニタリング**

App Services は、アプリケーションの実行環境を提供しますが、アプリケーションの開発・運用のための機能も合わせて提供されます。

これらの機能および提供されるリソースは、選択した SKU により異なります。

SKU は Free (無料)、Shared (共有)、Basic、Standard、Premium の 5 種類が提供されています。App Services の SKU は、作成後に変更することもできます。

OS は Windows , Linux から選択することができます。Linux を選択した場合には、以下のランタイムスタックを選択することができます。

- .NET Core
- Java
- Node.js
- PHP
- Python (Preview)
- Ruby

なお、Linux を選択した場合には、Free, Shared は選択できません。

●Free（無料）

試験的に利用する場合、開発・テストを行う場合に最適です。Free レベルでは SLA は提供されません。Free レベルでは CPU 時間、メモリ、帯域幅などに制限があります。制限値を超過した場合には、すべてのリクエストに対して HTTP 403 Forbidden が返却されます。Linux では選択できません。

●Shared（共有）- Preview

試験的、あるいはワークロードの低いアプリケーションに適しています。Shared レベル以上でカスタムドメインの使用がサポートされます。Shared レベルでは SLA は提供されませんが、Preview 割引が適用されます。Linux では選択できません。

●Basic

トラフィック要件が低く、自動スケール等を必要としないアプリケーションに適しています。

Basic レベル以上で 99.95%の SLA が提供されます。

●Standard

実稼働アプリケーションに適しています。高度な自動スケール、自動バックアップ、Traffic Manager を使用した地理分散、VPN ハイブリッド接続などの機能がサポートされます。

●Premium v2

Standard に加え、多数のスケールインスタンス、高頻度の自動バックアップが提供されます。

また、Premium では仮想ネットワークで完全に分離される App Service Environment を作成することもできます。

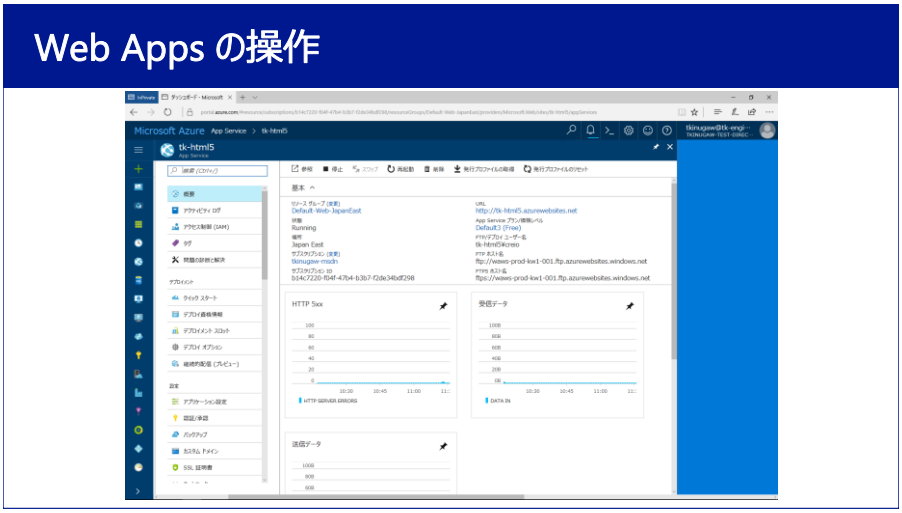
●Premium (Windows) Container

Windows コンテナとしてデプロイされた実稼働アプリのみを対象として、優れたパフォーマンスを提供するよう設計されています。

●Isolated（分離）

ミッションクリティカルなアプリケーションの実行環境として設計されています。より高速な CPU と SSD ストレージを搭載し、他のテナントとは分離された専用環境として提供されます。最大インスタンス数も大幅に増加され、100 インスタンスまで利用できます。この Isolated プランでは、App Service Environment として仮想ネットワークによる分離が提供されます。

インスタンス	コア	メモリサイズ	ストレージ
Free	共有 – 60 分/日	1GB	1GB
Shared	共有 – 240 分/日	1GB	1GB
Basic 1	1	1.75GB	10GB
Basic 2	2	3.5GB	10GB
Basic 3	4	7GB	10GB
Standard 1	1	1.75GB	50GB
Standard 2	2	3.5GB	50GB
Standard 3	4	7GB	50GB
Premium 1 v2	1	3.5GB	250GB
Premium 2 v2	2	7GB	250GB
Premium 3 v2	4	14GB	250GB
Premium (Windows) Container PC2	2	8GB	
Premium (Windows) Container PC3	4	16GB	
Premium (Windows) Container PC4	8	32GB	
Isolated 1	1	3.5GB	1TB
Isolated 2	2	7GB	1TB
Isolated 3	4	14GB	1TB



アプリケーションの停止や SSL 証明書の適用、スケールの設定など、App Services を管理する基本操作は、Azure 管理ポータルから行うことができます。Azure 管理ポータルでは次の操作が可能です。

●停止

アプリケーションの稼働を停止します。App Services ではインスタンスが停止状態でも課金は継続されます。課金を停止させるためにはアプリケーションを削除してください。

●開始

インスタンスが停止している場合、インスタンスを開始できます。

●スワップ

デプロイメントスロットを使用している場合、運用環境とデプロイメントスロット環境をシームレスに入れ替えます。デプロイメントスロットは Standard 以上で利用可能です。

●再起動

インスタンスを再起動します。

●削除

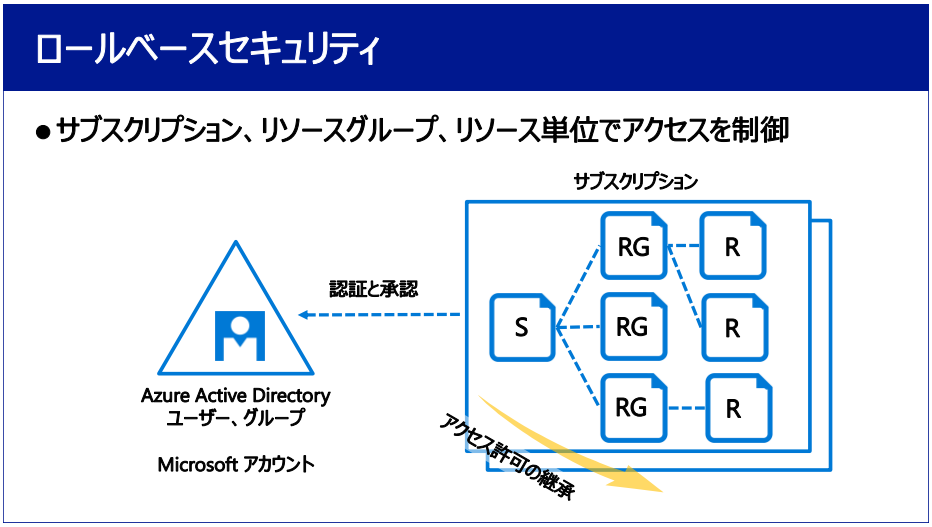
インスタンスを削除します。実行中のインスタンスを削除することもできます。

●発行プロファイルの取得

Visual Studio 等からアプリケーションを発行する際に必要な資格情報が含まれたファイルをダウンロードします。ダウンロードしたファイルは Visual Studio からインポート可能です。

●発行プロファイルのリセット

アプリケーションの発行に必要な資格情報をリセットします。これにより、「発行プロファイルの取得」でダウンロード済みの発行プロファイル、Visual Studio 等に保存された発行プロファイル情報は無効となります。



リソースマネージャーでは、Azure Active Directory のユーザーやグループ、Microsoft アカウントに対して、役割ごとのアクセス許可を割り当てることができます。これを「ロールベースセキュリティ（Role-based access control、RBAC）」と言います。

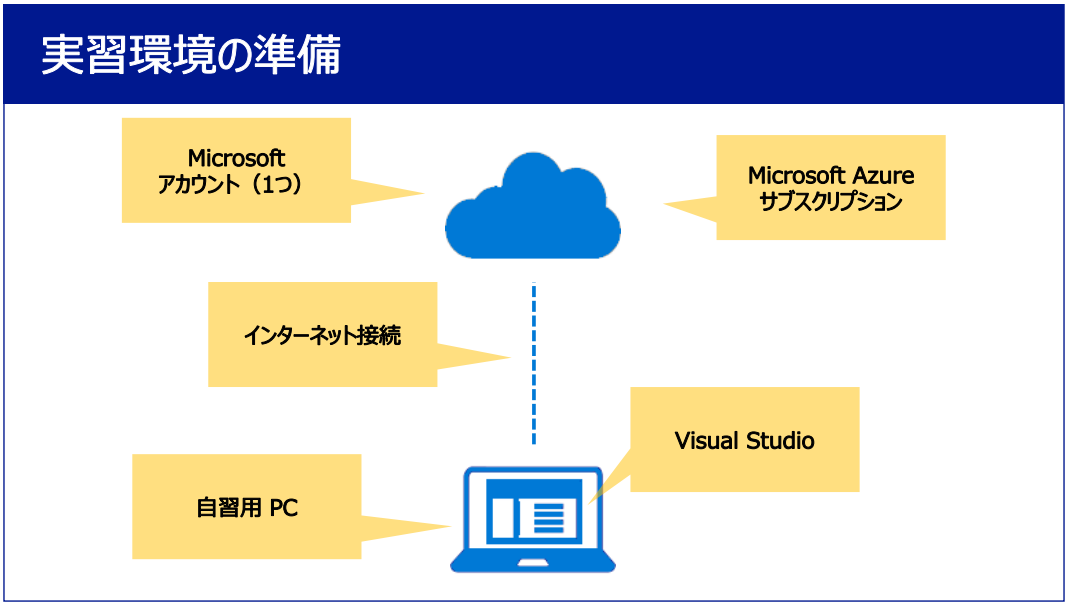
ロールベースセキュリティで割り当て可能な役割は、リソースの種類により異なりますが、例えば、App Services の場合、次の役割があります。

役割	説明
所有者	リソースへのアクセスを含め、すべてを管理できます。
共同作成者	すべてを管理できますが、リソースへはアクセスできません。
閲覧者	すべてを表示できますが、変更はできません。

役割ごとのアクセス許可は、リソースだけでなく、リソースグループやサブスクリプションにも割り当て可能です。リソースグループやサブスクリプションにアクセス許可を割り当てた場合、サブスクリプション、リソースグループ、リソースの順にアクセス許可が継承されます。

3. 実習環境の準備

この自習書の手順は、次の実習環境を準備することで、実際に試すことができ、理解を深めることができます。



●自習用 PC

Windows 10 などの自由に操作可能な Windows コンピューター（物理コンピューター）が 1 台必要です。また、Web ブラウザとして Internet Explorer または Edge Browser を使用します。

●Visual Studio

Windows コンピューターには、展開するアプリケーションを作成するために、Visual Studio をインストールしておきます。本自習書では、Visual Studio 2017 を前提として解説いたします。Edition は問いません。Visual Studio をお持ちでない方は https://www.visualstudio.com/ja/downloads/ よりダウンロード可能です。

●インターネット接続

Windows コンピューターは、インターネットに接続されている必要があります。企業内で実習をおこなう場合は特に注意が必要です。多くの企業のインターネット接続では、ファイアウォールが介在します。その場合は実習で必要な HTTP/s 接続のプロトコルがブロックされていないことを確認しておく必要があります。

●Microsoft Azure サブスクリプション

Microsoft Azure サブスクリプションは、Microsoft Azure を使用するための権利です。すでに Microsoft Azure サブスクリプションをお持ちの場合は、そのサブスクリプションを利用することができます。まだ、Microsoft Azure サブスクリプションをお持ちでない場合は、実習用に

https://azure.microsoft.com/ja-jp/pricing/free-trial/ より、1 か月間の無料評価版をサインアップし、使用することができます。

ワンポイント
Microsoft Azure の 1 か月間の無料評価版のサインアップには、本人確認のため、電話番号（固定電話または携帯電話）およびクレジットカードの情報が必要です。なお、無料評価版の利用は 1 回までとなっており、過去すでに利用された方は無料評価版にサインアップいただけません。有償のサブスクリプションに切り替えていただきますようお願いいたします。

●Microsoft アカウント

Microsoft アカウントは、マイクロソフトが提供するクラウドサービスを利用するための ID です。この自習書では、Microsoft Azure サブスクリプションの利用権を Microsoft アカウントに割り当てます。Microsoft アカウントは http://www.microsoft.com/ja-jp/msaccount/default.aspx より、無償で登録できます。

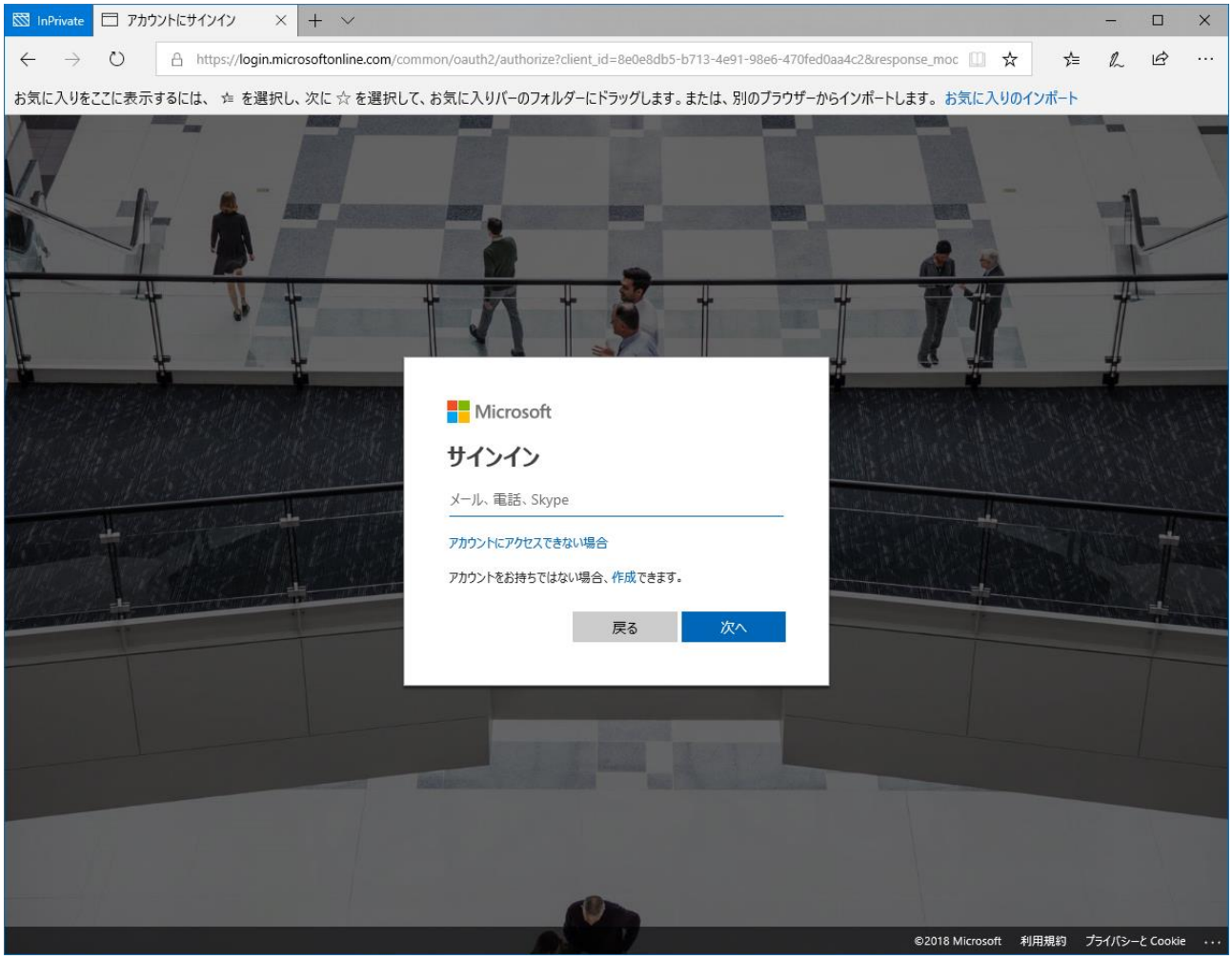
ワンポイント
新しく、Microsoft Azure の 1 か月間の無料評価版のサブスクリプションをサインアップする場合、同時に Microsoft アカウントを登録することもできます。

4. （参考）Microsoft Azure 無料評価版のサインアップ

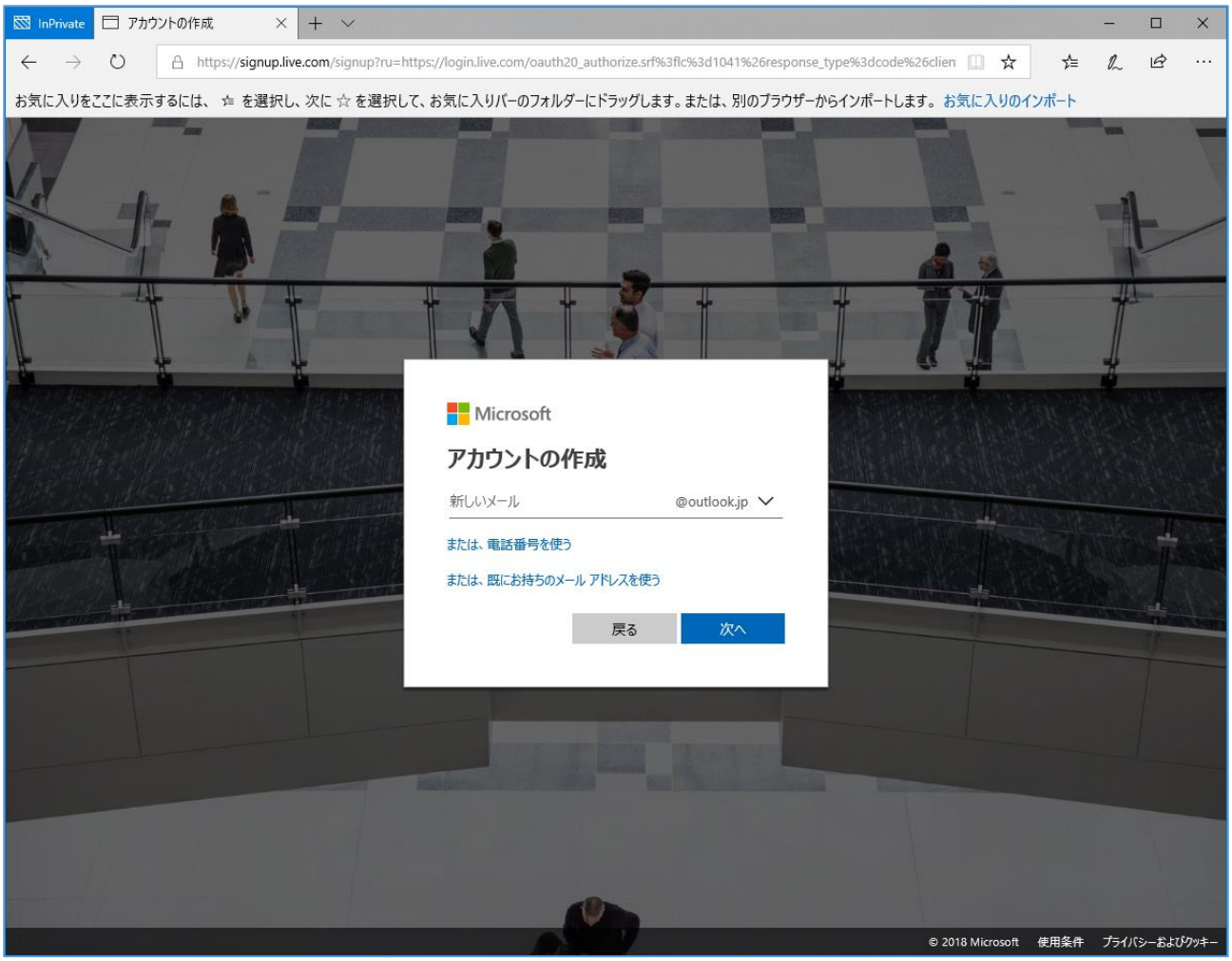
この手順はオプションです。Microsoft Azure サブスクリプションをお持ちでない場合、次の手順を実行し、Microsoft Azure の 1 ヶ月間の無料評価版のサブスクリプションを取得してください。

1. Web ブラウザを起動し、https://azure.microsoft.com/ja-jp/pricing/free-trial/ にアクセスします。
2. ［Azure の無料アカウントを今すぐ作成しましょう］が表示されます。［無料で始める］をクリックします。

3. ［サインイン］が表示されます。Microsoft アカウントのメールアドレスとパスワードを入力し、［サインイン］をクリックします。まだ、Microsoft アカウントをお持ちでない場合は、［作成］をクリックします。



4. ［作成］をクリックした場合、［アカウントの作成］が表示されます。[または、すでにお持ちのメールアドレスを使う]をクリックし、自分のメールアドレス（自分が日常的に使用しているメールアドレス）と任意のパスワードを指定し、［アカウントの作成］をクリックします。利用できるメールアドレスを持ちでない場合、[新しい Outlook メール（～@outlook.jp）を作成することもできます。



ワンポイント

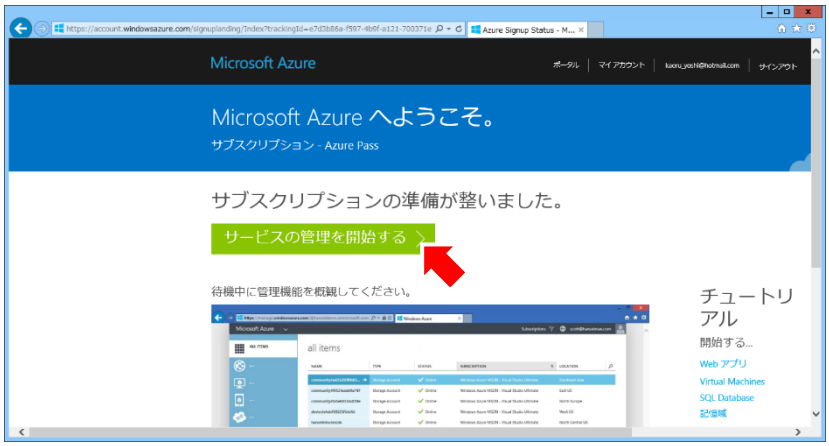
ここで登録した Microsoft アカウントが、Microsoft Azure サブスクリプションの管理者（サブスクリプション所有者）になります。

5. ［サインアップ］が表示されます。サインアップに必要な情報を指定します。なお、サインアップには、ショートメッセージ（SMS）またはコールバックを利用した電話による確認とクレジットカードの登録による確認が必要です。確認が完了したら、［サインアップ］をクリックします。

ワンポイント

登録したクレジットカードが勝手に使用されることはありません。

6. ［Microsoft Azure へようこそ］の［準備が完了するまでしばらくお待ちください］が表示されます。しばらく待つと、［サブスクリプションの準備が整いました］が表示されます。［サービスの管理を開始する］をクリックします。



ワンポイント

Microsoft Azure の 1 か月間の無料評価版では、22,500 円相当のクレジットが利用可能です。なお、無料評価版の使用制限に達した場合には、利用している仮想マシンとクラウドサービスの開放がおこなわれます。もし制限にかかった場合には、翌請求月になるまでは利用が制限されます。ストレージサービスについては、読み取り専用としてアクセスが可能です。

5. Visual Studio からの Web App の作成

はじめに Azure 上に展開する Web アプリケーションを作成します。Visual Studio から新しい Web アプリケーションを作成します。

- Web App とは

Web App は Azure 上で提供される PaaS (Platform as a Service) 環境であり以下の特徴を持っています。

- 必要に応じて自動スケール可能
- 自動バックアップ、デプロイメントスロットの機能も提供
- 最安値無料から利用可能
- 非常に迅速な展開：デプロイ作業が秒単位で可能
- Azure DevOps, Git, GitHub, Dropbox 等からの自動発行も可能
- 実行環境を Windows / Linux から選択可能
  Linux の場合には、ランタイムスタックを以下から選択可能
  .NET Core, Java, Node.js, PHP, Python, Ruby

Web App を利用することにより、Web アプリケーションを手軽かつ迅速に展開する事が可能となります。

Web App で提供されるサービスプランは以下の通りです。

- Free

無料で利用可能 165MB/日 の帯域制限（Linux では選択できません。）

- Shared

SLA 対象外 (Preview)
Shared 以上でカスタムドメインを利用可能（Linux では選択できません。）
（デフォルトでは *.azurewebsites.net となる）

- Basic, Standard, Premium, Isolated

本番運用向け - SLA (99.95%) 利用可能
Standard 以上で自動スケール・自動バックアップをサポート
(Basic では手動スケール)

Basic 以上のプランでは、以下のスケールが提供されます。

	コア数	メモリ
B1/S1/P1	1	1.75GB
B2/S2/P2	2	3.50GB
B3/S3/P3	4	7GB

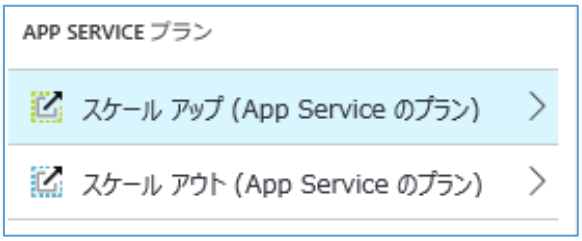
Premium v2, Isolated では以下のスケールが提供されます。

	コア数	メモリ
P1 v2, I1	1	3.5GB
P2 v2, I2	2	7GB
P3 v2, I3	4	14GB

利用可能な最大インスタンス数は以下の通りです。

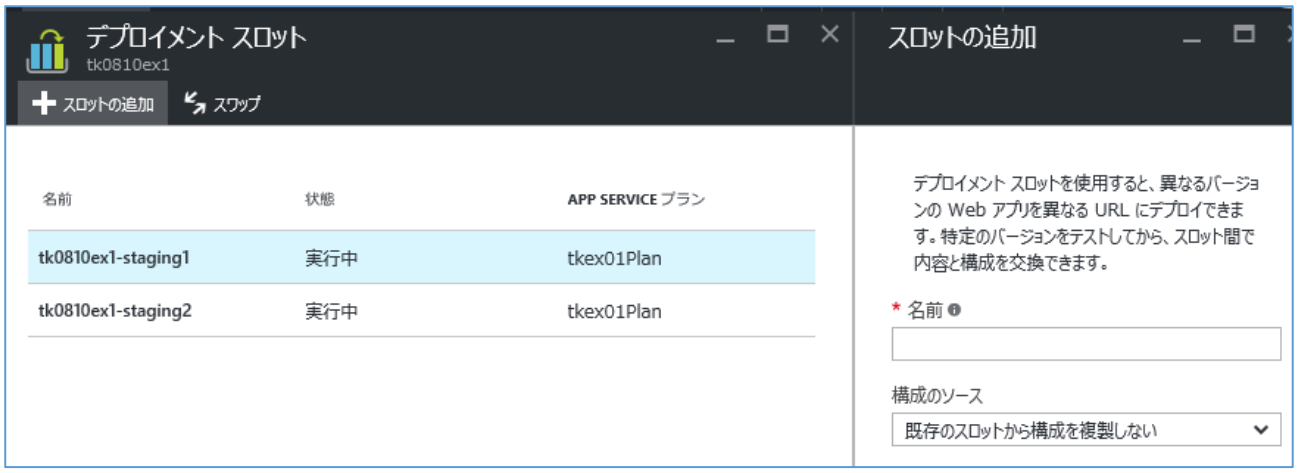
	ローカル ストレージ	最大 インスタンス数	
Basic	10GB	3 (手動)	
Standard	50GB	10 (自動)	自動バックアップ、デプロイメントスロット、地理冗長など利用可能
Premium	250GB	20(自動)	
Isolated	1TB	100 (自動)	リクエストにより 100 以上も利用可能

これらのサービスプランは、Azure 管理ポータルからいつでも変更可能です。



[スケールアップ]によりサービスプランを変更可能です。[スケールアウト]により、インスタンス数を変更可能です。Free, Shared ではスケールアウトはサポートされません。

Standard 以上で提供される「デプロイメントスロット」は、運用環境以外の動作環境を作成することが出来る機能です。例えば“contoso.azurewebsites.net” という Web App に対して “contoso-staging1.azurewebsites.net” という環境を作成することが出来ます。これらの環境は[スワップ]操作により、稼働状態のまま入れ替えを行うことが出来ます。



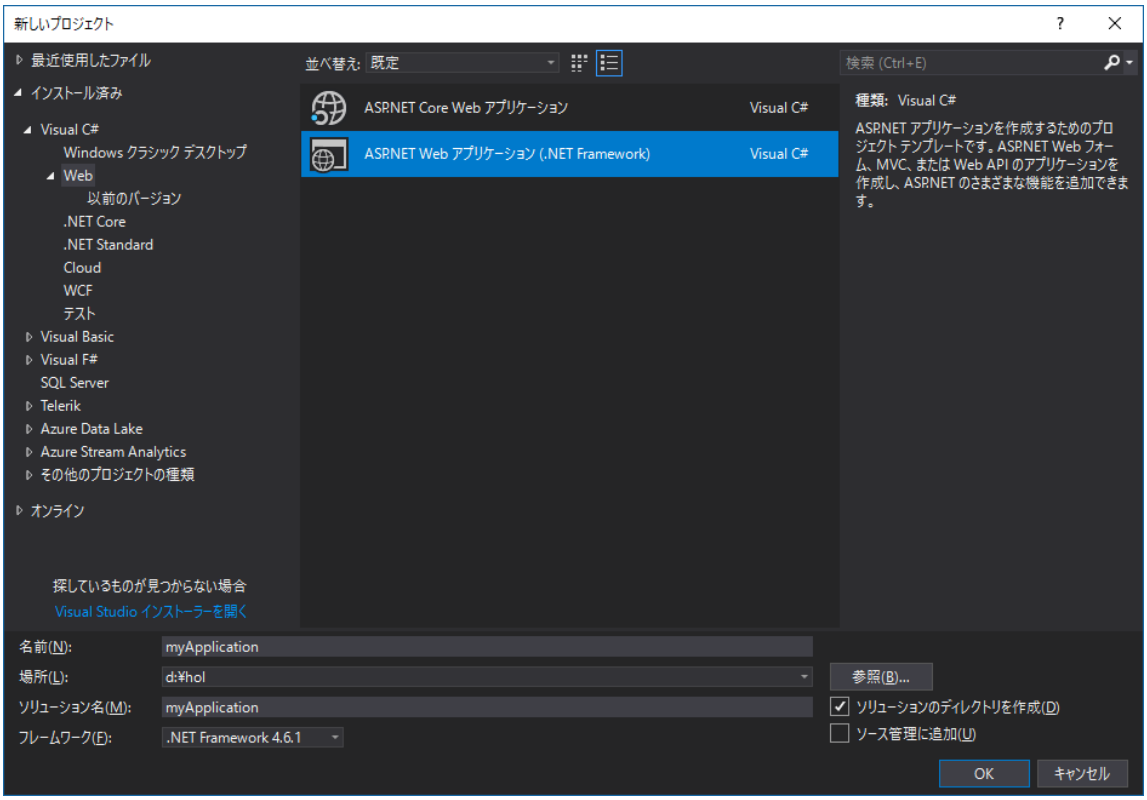
これらの機能を利用することにより、Web アプリケーションの展開・テスト・運用にかかる負荷を大きく軽減することが可能となります。

Web App で稼働する Web アプリケーションの開発は、これまでのオンプレミスでの Web アプリケーション開発の手順とほぼ同じ手順となります。しかしながら、以下に挙げるような Azure 環境一般の注意事項には留意する必要があります。

・Azure 内では、時計は UTC で動作している（日本時間にするには 9 時間加算する必要がある）
・動作環境は Neutral Culture であるため、文字列比較・通貨・日付文字列には注意が必要。

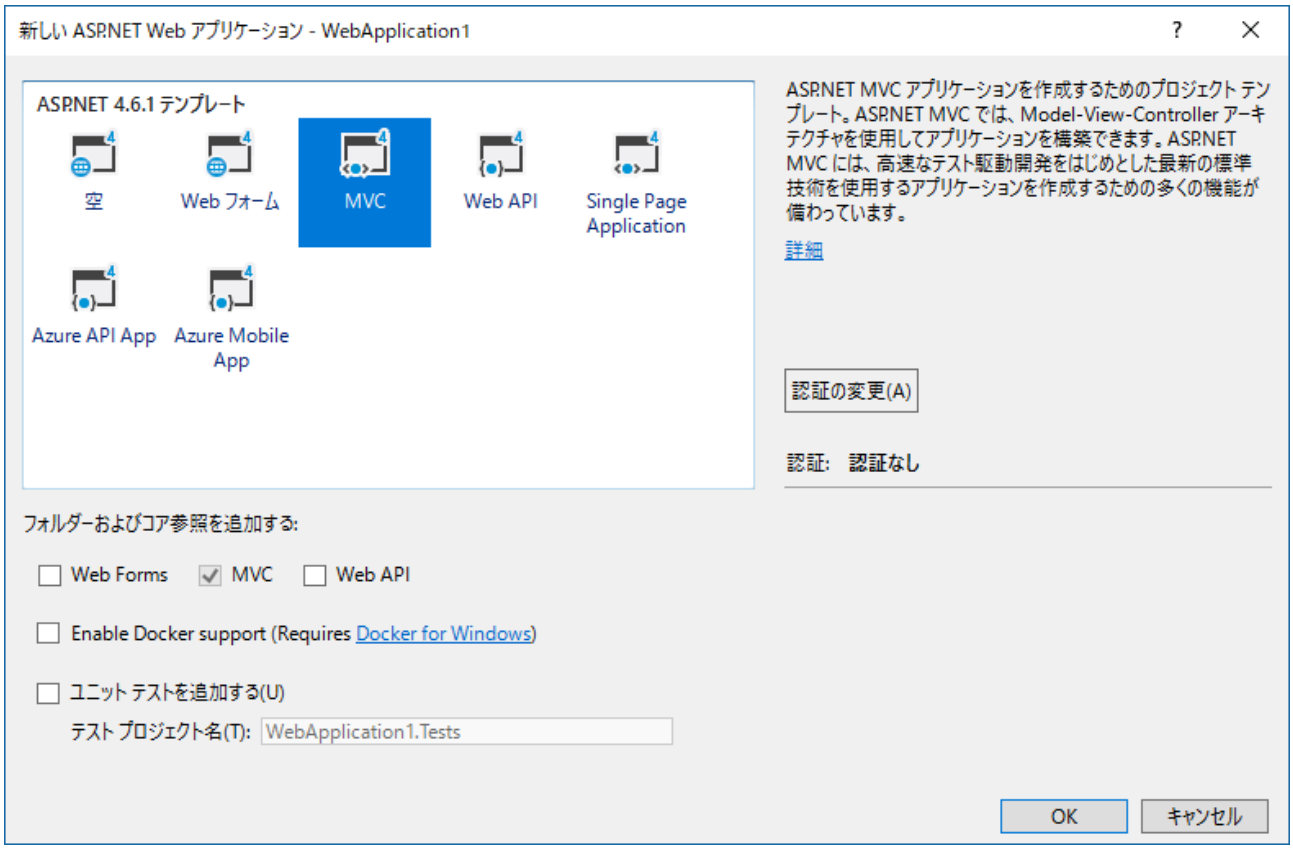
この Web App を開発し、Azure 環境に展開するまでを実際に演習します。

1. Visual Studio を起動し、新しい Web アプリケーションを作成します。Visual Studio 起動後、「新しいプロジェクトの作成…」より新しいプロジェクトを作成します。



パラメーター	説明	値
選択するテンプレート	作成するプロジェクトの種類	“Visual C#”-“Web”から「ASP.NET Web アプリケーション(.NET Framework)」
名前	プロジェクト名	（任意の名称）
場所	ソリューションを作成するフォルダー	（任意の場所）
ソリューション名	ソリューション名	＜規定値＞
フレームワーク	使用する.NET Framework のバージョン	.NET Framework 4.6.1

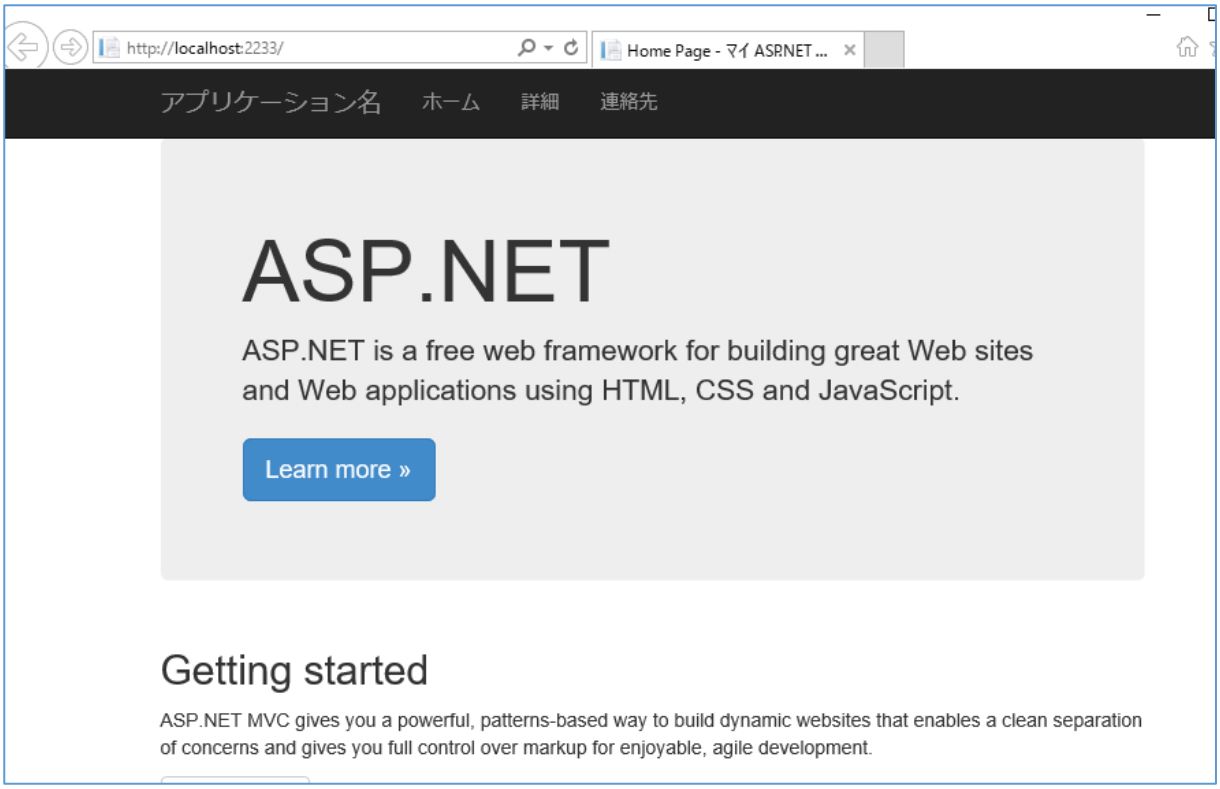
2. 「新しい ASP.NET Web アプリケーション」のダイアログが表示されます。ここでは「MVC」を選択します。



パラメーター	説明	値
テンプレート	作成する Web アプリケーションの種類を選択します	MVC
認証	アプリケーションが必要とする認証を選択します	認証無し（デフォルト）
フォルダーおよびコア参照を追加する	使用するフレームワークと参照設定を追加します	MVC （デフォルト）のみチェック
Enable Docker support	Docker for Windows でのサポートを追加します	チェックしない（デフォルト）
ユニットテストを追加する	単体テストプロジェクトを追加します	チェックしない（デフォルト）

設定後、[OK]をクリックしてください。

3. プロジェクトが作成された状態で、キーボードから[F5]キーを押してデバッグ実行を行います。下図のようにブラウザ上にアプリケーションが表示されます。



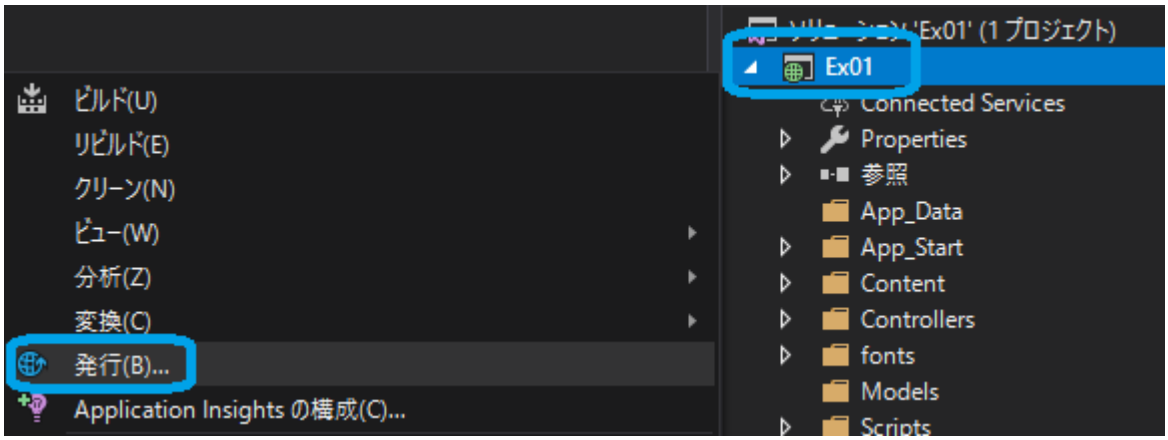
表示を確認したところでブラウザを閉じ、デバッグを終了します。

※デバッグ実行後、Edge browser が起動した場合には、ブラウザを閉じたのち、手動で Visual Studio のデバッグを終了させる必要があります。(Internet Explorer の場合には、自動的に Visual Studio のデバッグが終了します。)

次の手順に進む前に、Visual Studio のデバッグが終了していることを再度確認してください。

4. ソリューションエクスプローラーよりプロジェクトを選択して右クリックします。

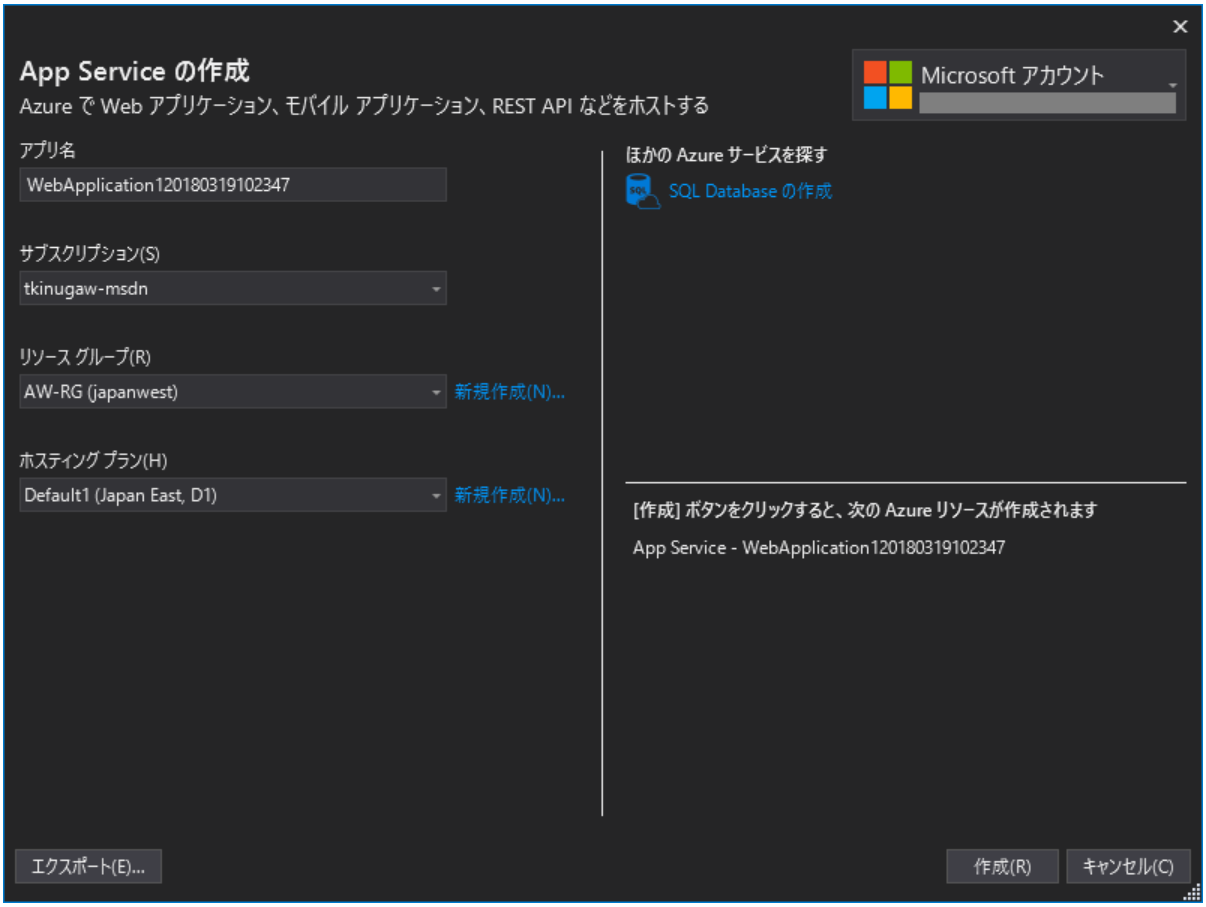
表示されたメニューより[発行(B)...]を選択します。



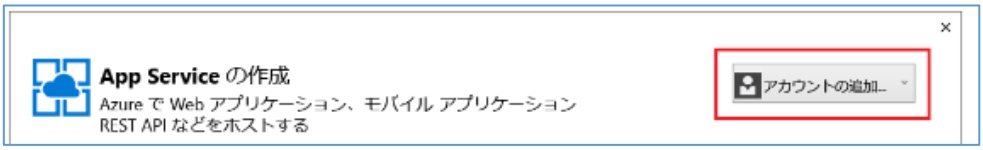
5. Visual Studio 上に発行画面が表示されます。ここからウィザードに従って、Web アプリケーションを Azure Web App として発行します。

[Microsoft Azure App Service] の [新規作成]を選択し、[発行]ボタンをクリックします。

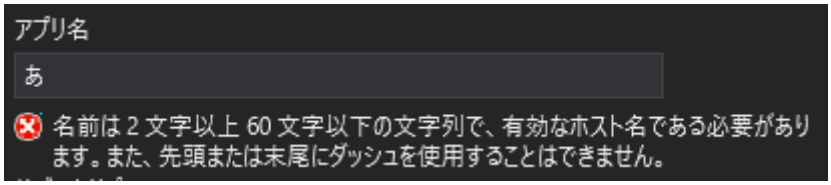
6. 新しい App Service 作成のためのパラメーター入力画面が表示されます。



※Azure にサインインしていない場合には、右上のアカウントの部分に「アカウントの追加」が表示されます。「アカウントの追加」をクリックしてサインインしてください。

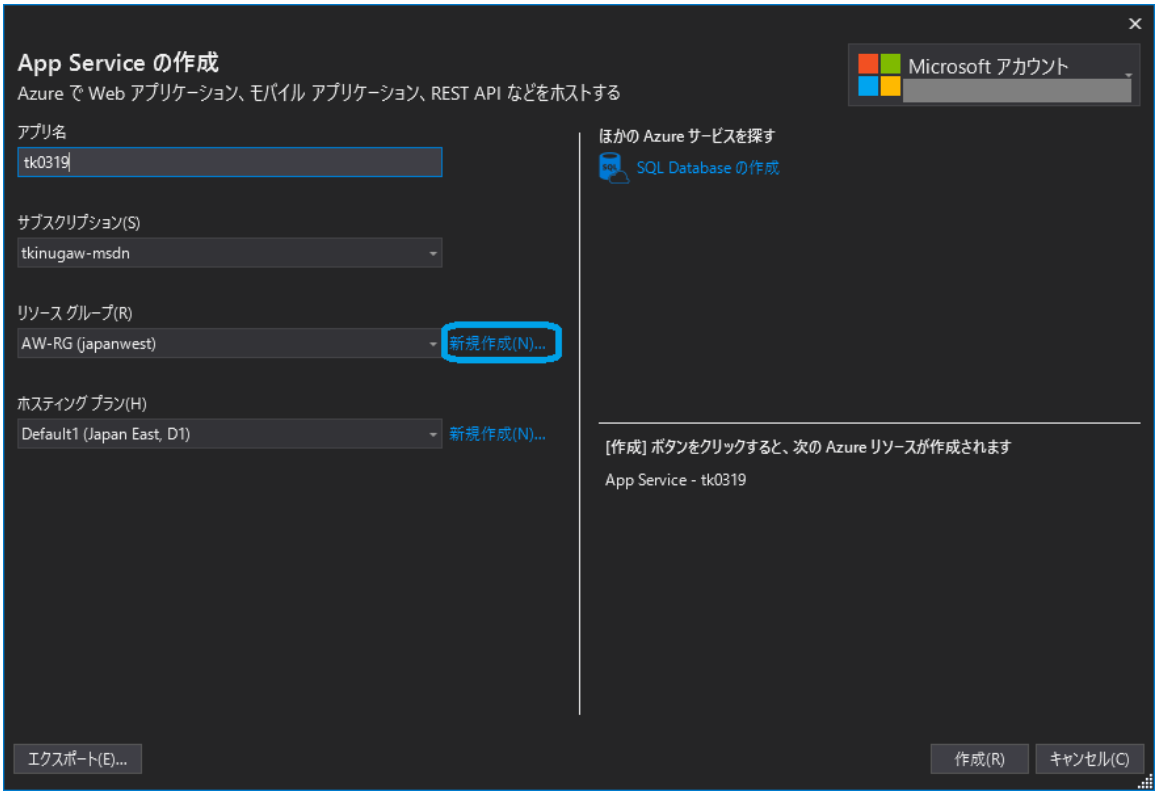


適切な任意の Web App の名前を入力してください。Web App の名前は、2 文字から 60 文字の英数字とハイフンが使用できます（先頭と末尾にハイフンは使用できません。）
使用できない Web App の名前であれば赤の❌が表示されます。

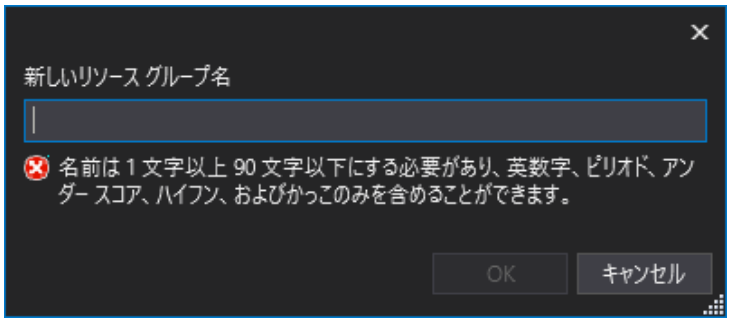


パラメーター	説明	値
アプリ名	URL に使用される名称です。 *.azurewebsites.net となります。	（任意のわかりやすい名称）
サブスクリプション	使用するサブスクリプションです。	<既定値>

7. 展開するリソースグループを作成します。リソースグループの[新規作成(N)...]ボタンをクリックします。



リソースグループ名を入力するダイアログが表示されます。リソースグループ名は 1 文字から 60 文字の英数字、ハイフン、アンダースコア、カッコ、ピリオドを使用できます。

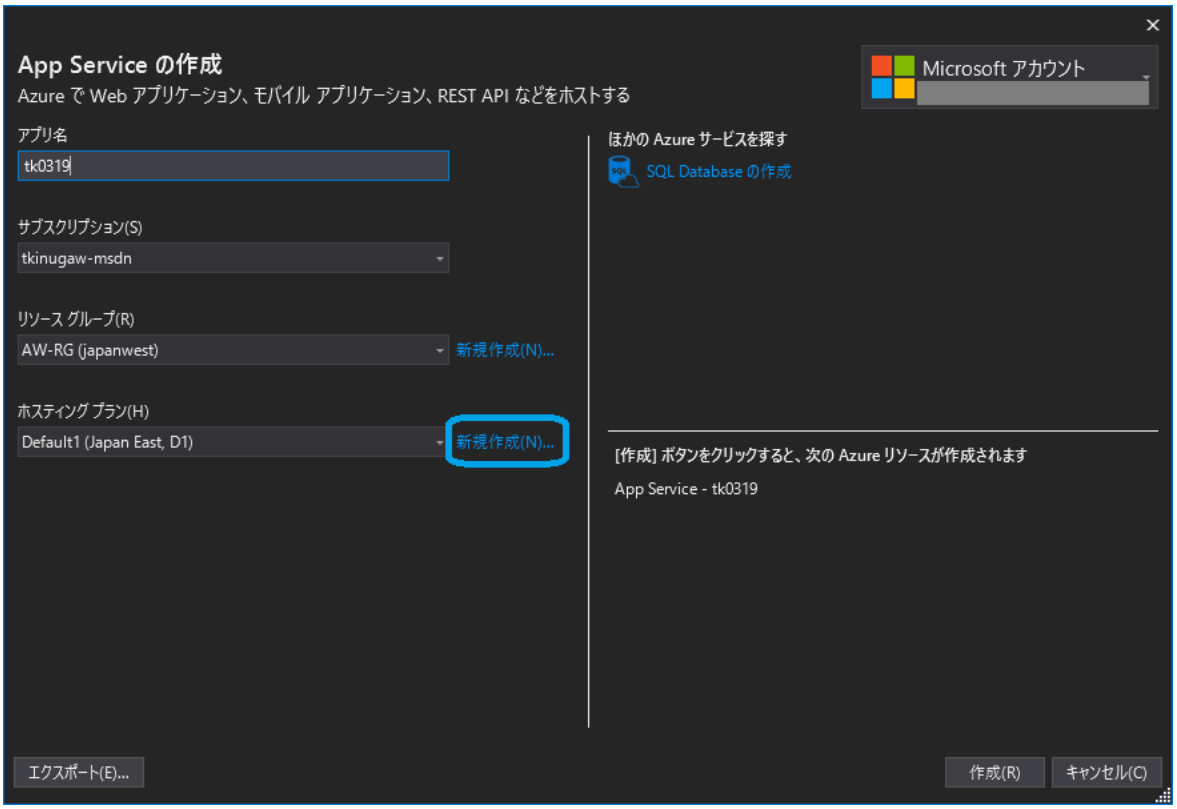


パラメーター	説明	値
新しいリソースグループ名	リソースグループにつける任意の名前です。	AW-RG

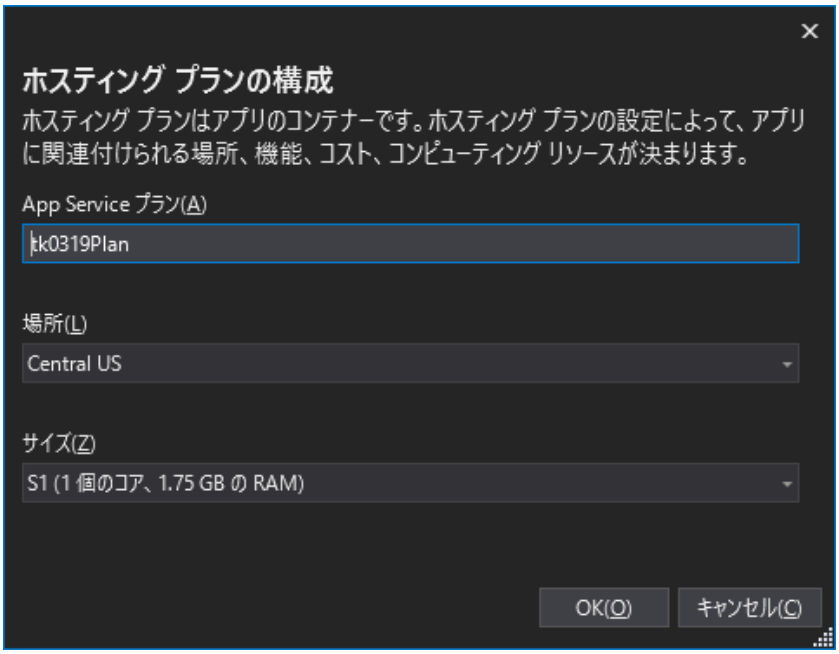
入力後[OK]ボタンクリックによりダイアログを閉じます。

リソースグループ名はこの後の実習でも使用しますので、忘れないようにしてください。

8. 同様に、展開する Web App のサービスプランを作成します。App Service プランの[新規作成(N)...]ボタンをクリックします。



App Service プランの構成ダイアログが表示されます。



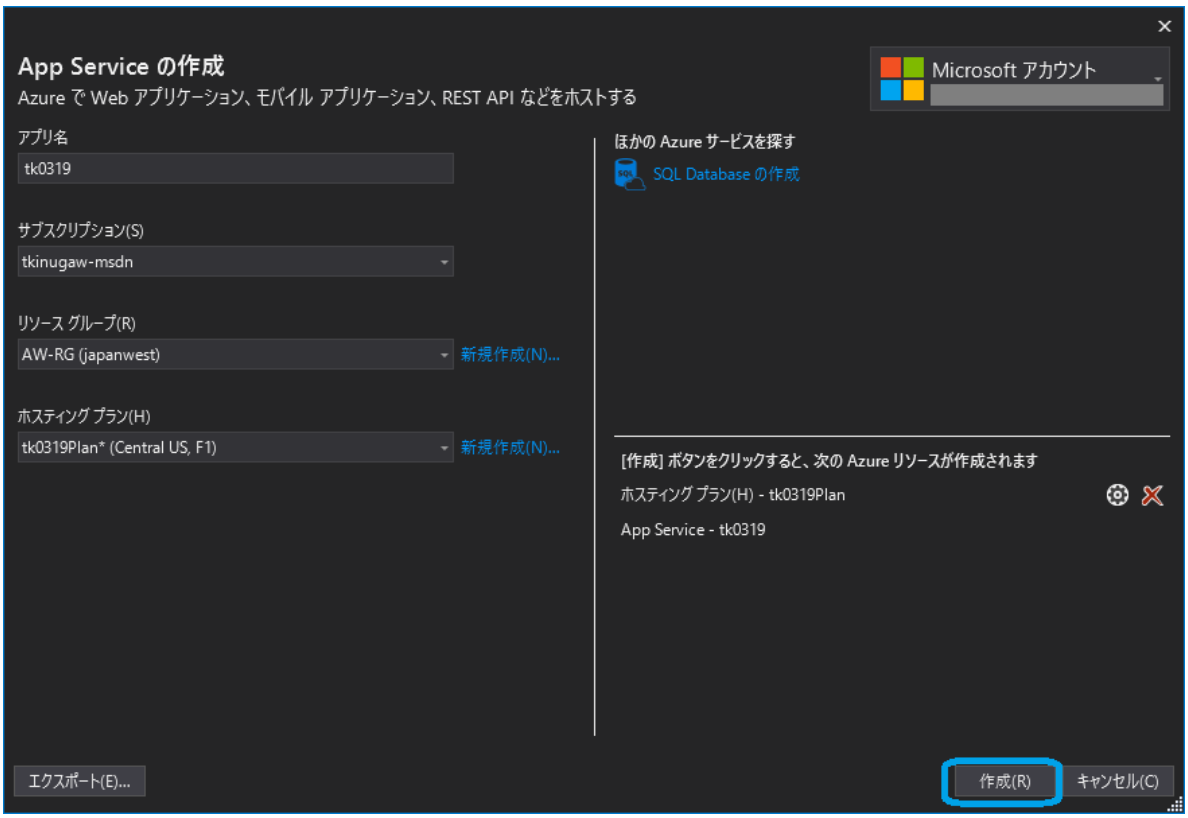
ここで作成する Web アプリケーションの配置先、サービスレベルを設定します。

パラメーター	説明	値
App Service プラン	サービスプランにつける任意の名前です。	（任意の名称）
場所	展開するデータセンターです。	Japan East または Japan West
サイズ	使用するサービスレベルです。	無料

デフォルトでは上図のように場所が “South Central US”となっていますが、”Japan East”などに適宜変更してください。

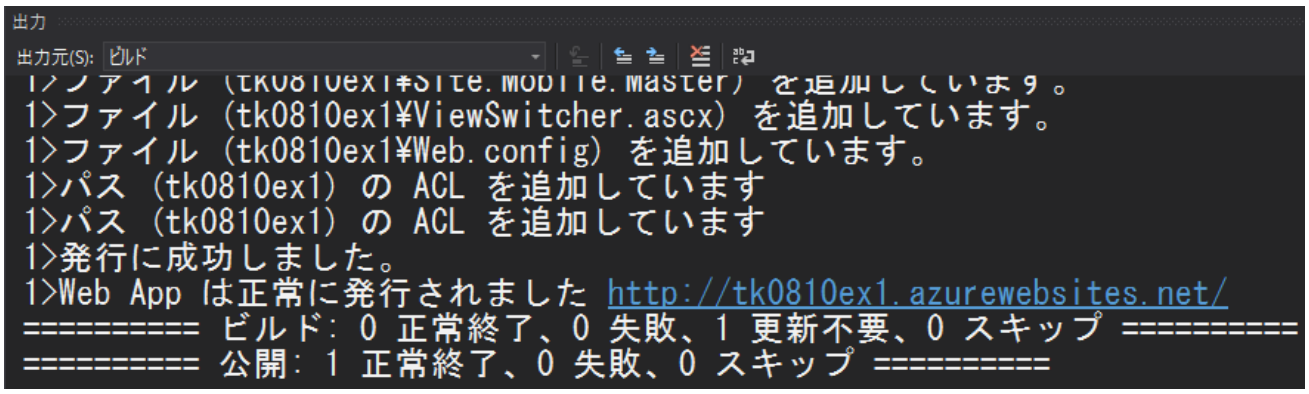
設定後[OK]をクリックしてください。[OK]をクリックすると、このダイアログを閉じます。

9. 入力完了後、[作成(C)]ボタンをクリックします。クリックできない場合には入力エラーが存在しないか確認してください。



10. [作成]ボタンをクリックすることにより Azure 上に Web App が作成され、自動的にデプロイされます。

[発行(P)]ボタンをクリックすることにより、WebApp への発行が行われます。出力ウィンドウに状況が表示されます。



発行完了後、ブラウザが起動し、発行された Web App が表示されることを確認してください。

Visual Studio 上では、下図のように発行プロファイルが表示されます。

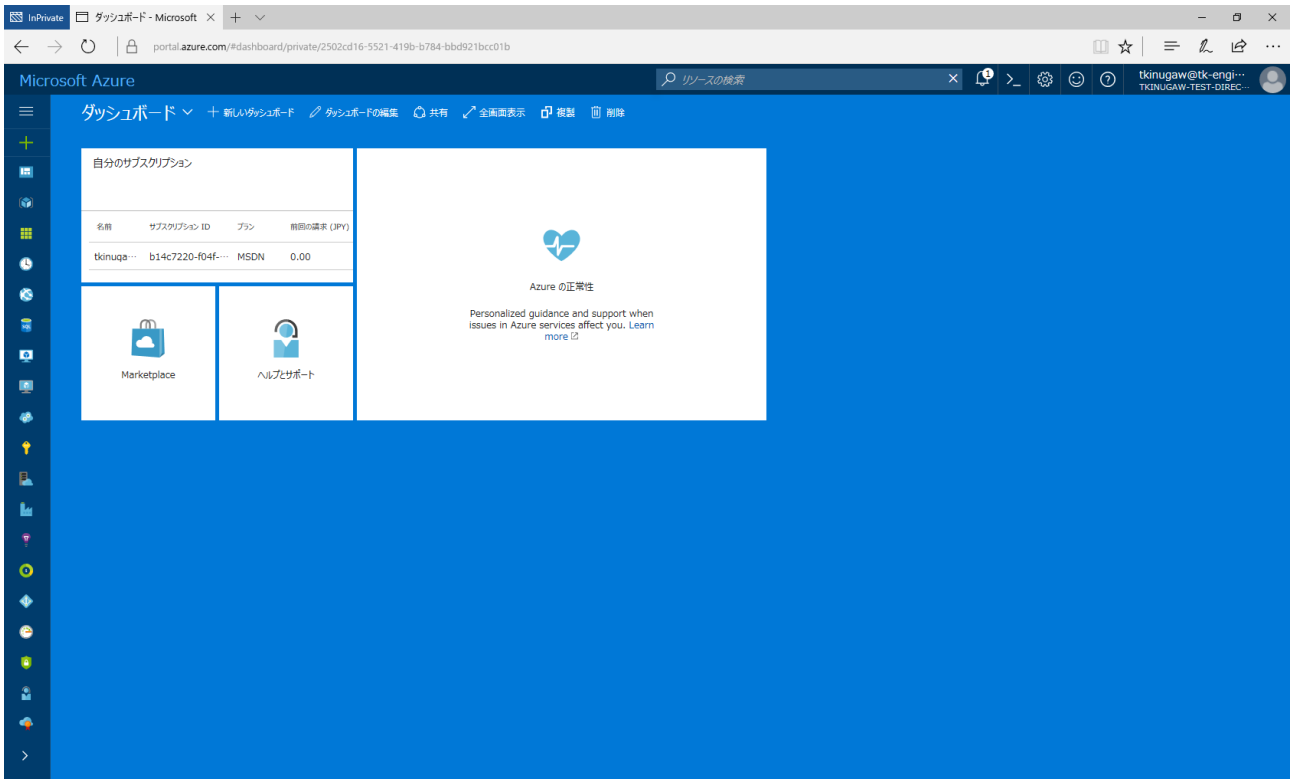
プロジェクトの発行プロファイルは、この画面より管理可能となります。

6. Web App のスケーリング設定 - スケールアップとスケールアウト

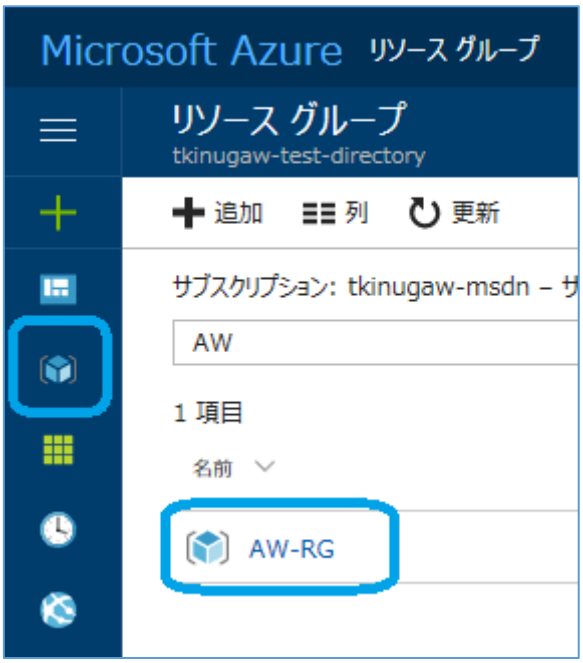
Azure 上に展開されている Web App のスケールアップとスケールアウトの設定を行います。この操作は Azure 管理ポータル上から行います。

Standard 以上で提供される自動スケーリングにより、CPU 負荷率などを基準としてインスタンス数を自動的に増減させることが可能になります。スケジュールによるルール設定も可能です。
従って、負荷の集中により自動的にインスタンス数を増加させ、負荷の減少に伴い自動的にインスタンス数を減少させることが可能になります。

1. ブラウザを起動し、https://portal.azure.com にアクセスします。お持ちのアカウントでサインインしてください。サインインすると Azure 管理ポータルの[スタート画面]が表示されます。



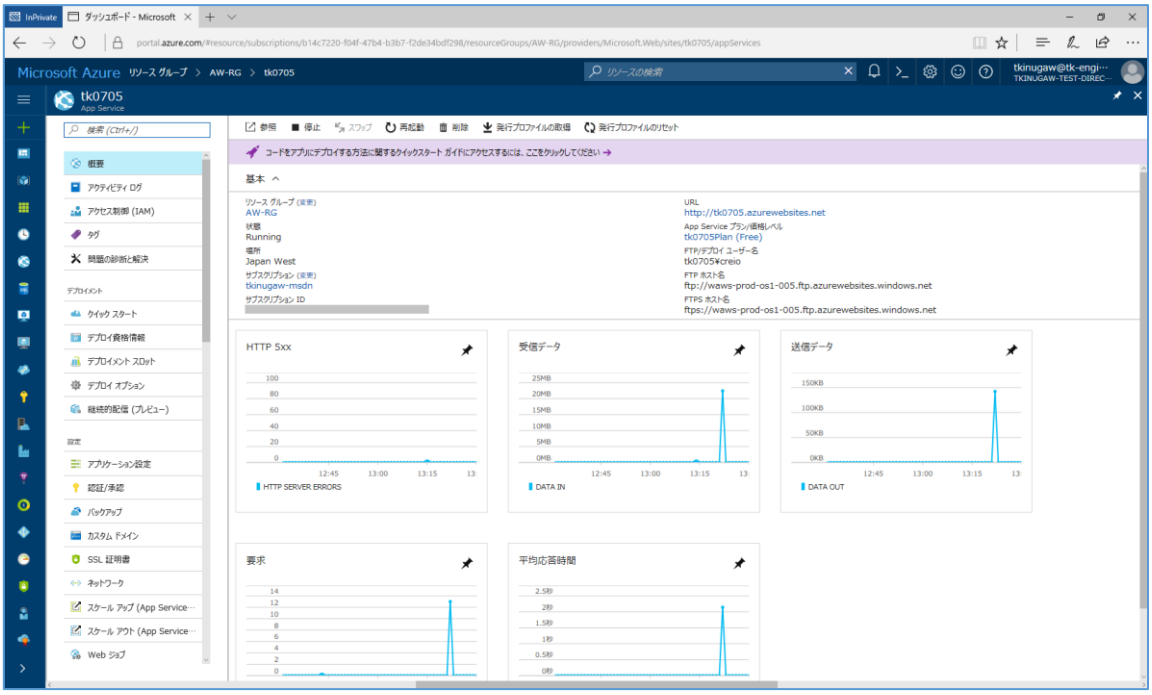
2. 画面左側のメニューより[リソースグループ]を選択します。表示されたリソースグループ一覧より、先ほどの演習で作成したリソースグループを選択します。



3. リソースグループを選択することにより、リソースグループに含まれる内容が表示されます。先ほどの演習で作成した Web App と Service プランが表示されていますので、Web App を選択します。

4. 展開した Web App の概要が表示されます。

Azure 管理ポータル上のこの画面で、Web App に対する各種設定を行うことができます。

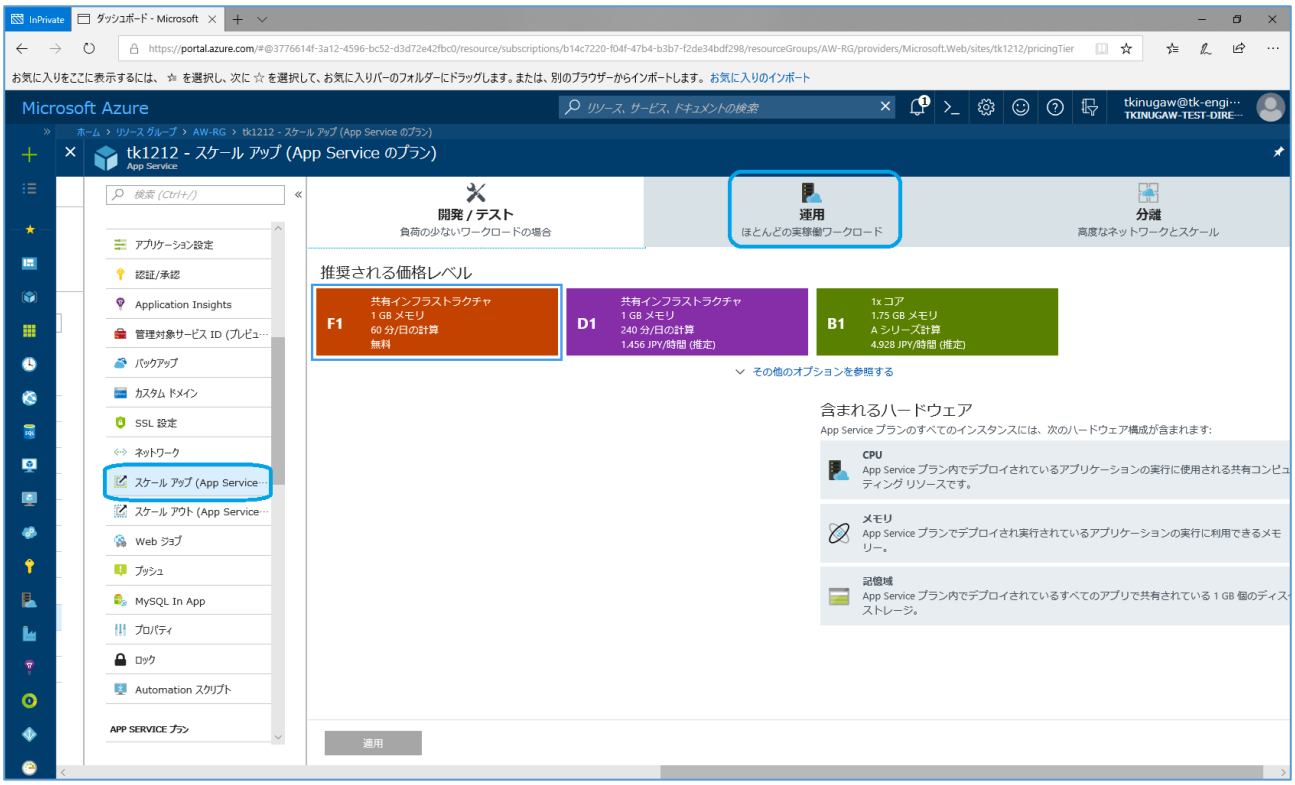


5. 現在 Free で動作していますので、この状態ではスケールアウトの設定を行うことができません。Free, Shard でスケールアウトの設定を行うと、下図のようにエラーが表示されます。

この演習では、Standard レベルに変更した上で自動スケール設定を行います。

Web App を選択している状態で左側メニューより[設定]→[スケール アップ (App Service のプラン)]を選択します。

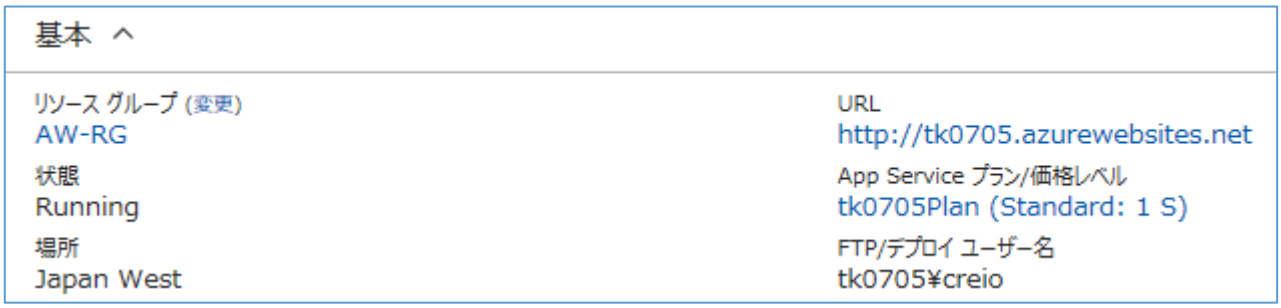
価格レベルの一覧が表示されます。ここでは[運用]をクリックします。



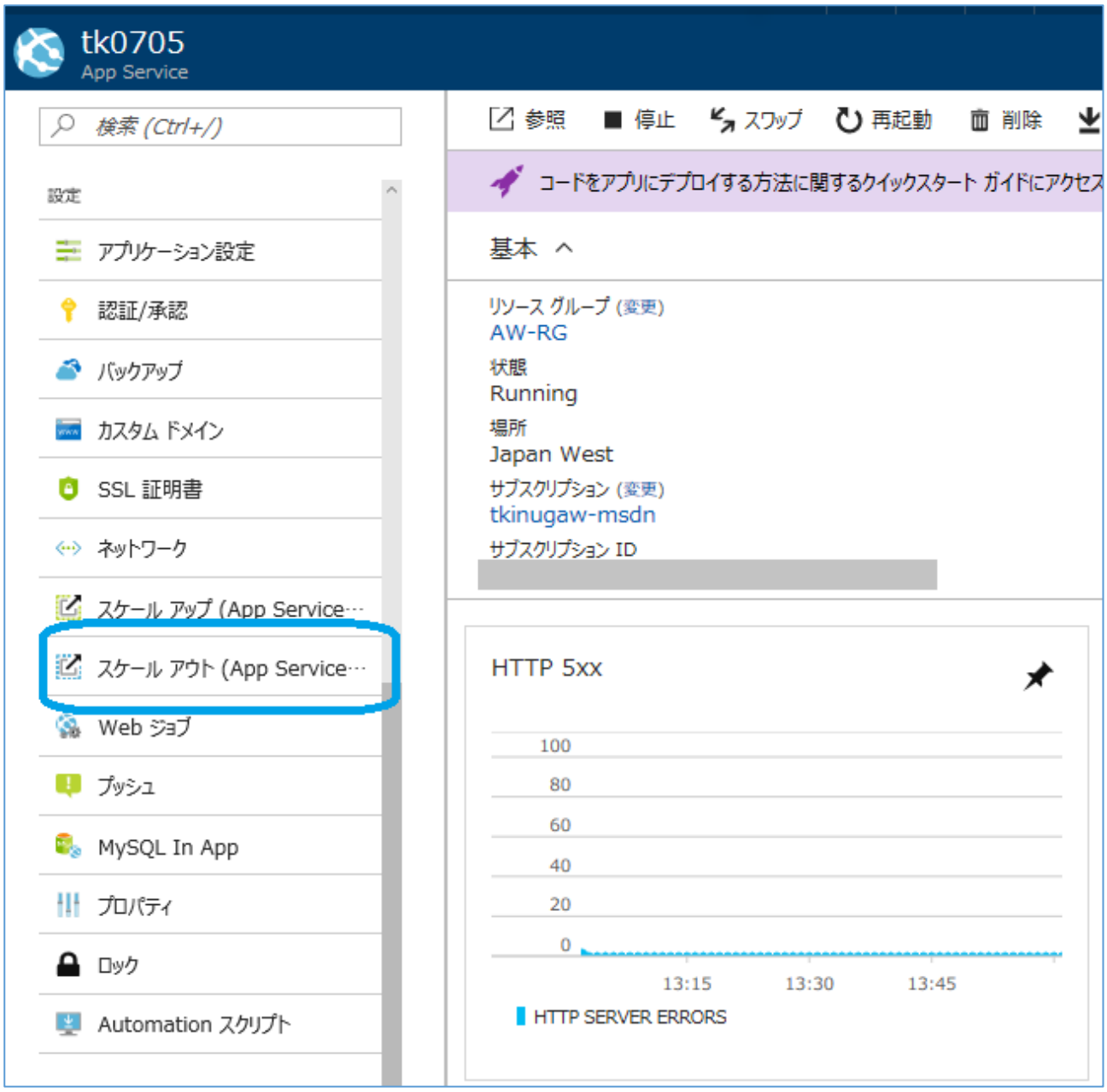
[運用]をクリックすることにより、S,P レベルの価格レベルが表示されます。ここでは[S1]を選択し[適用]をクリックしてください。

適用後ただちにスケールアップが行われます。スケールアップが完了すると、通知エリアに完了メッセージが表示されます。

この状態で[概要]に戻ると、[App Service プラン/価格レベル]の箇所に「Standard: 1 S」と表示されます。これにより、正常にサービスレベルが変更されたことが確認できます。

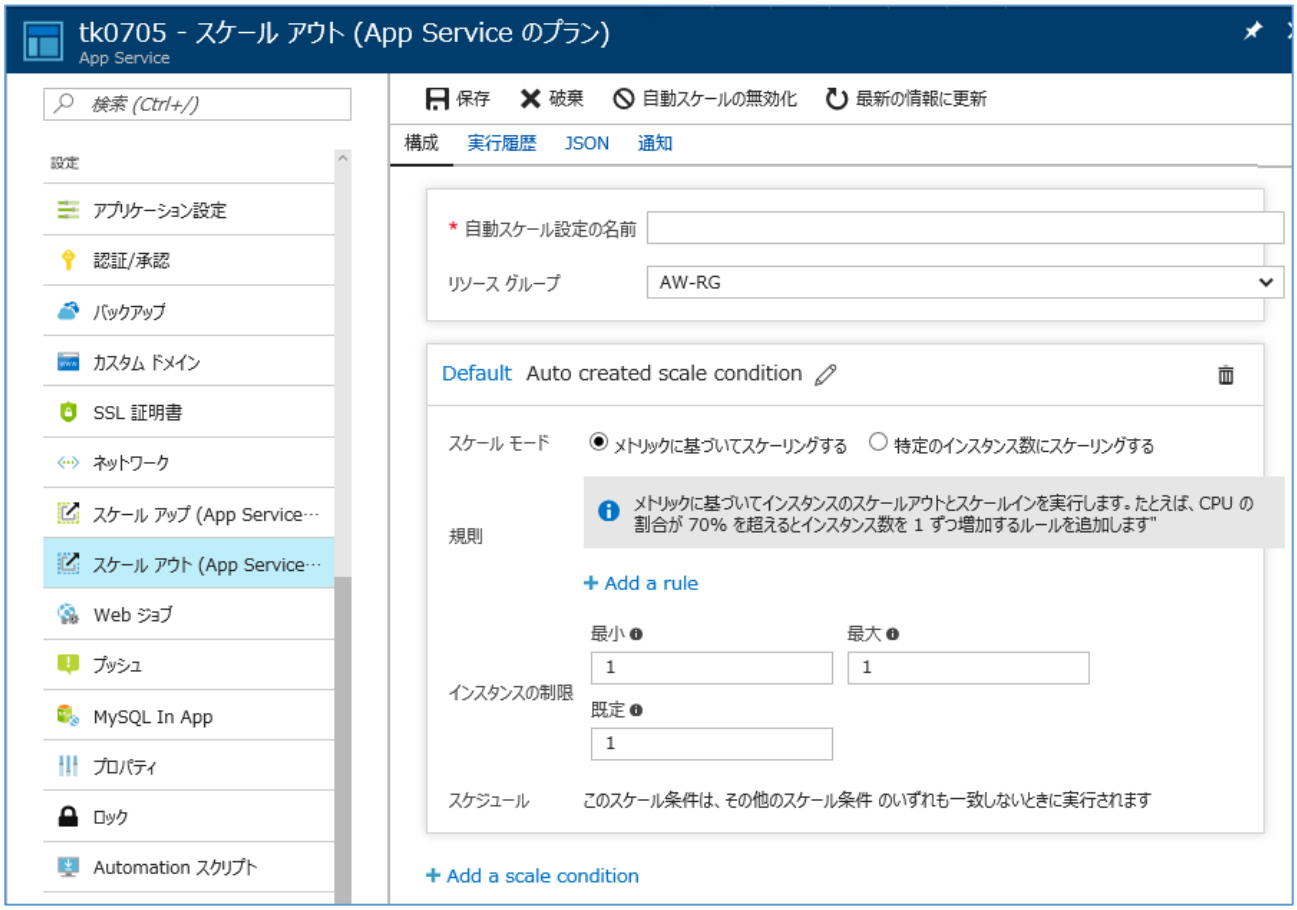


6. 続いて、スケールアウトの設定を行います。スケールアップと同様、左側メニューより[スケールアウト(App Service のプラン)]を選択します。



選択すると以下のようにスケールアウト設定画面が表示されます。ここでは自動スケール設定を行います。[自動スケールの有効化]ボタンをクリックしてください。

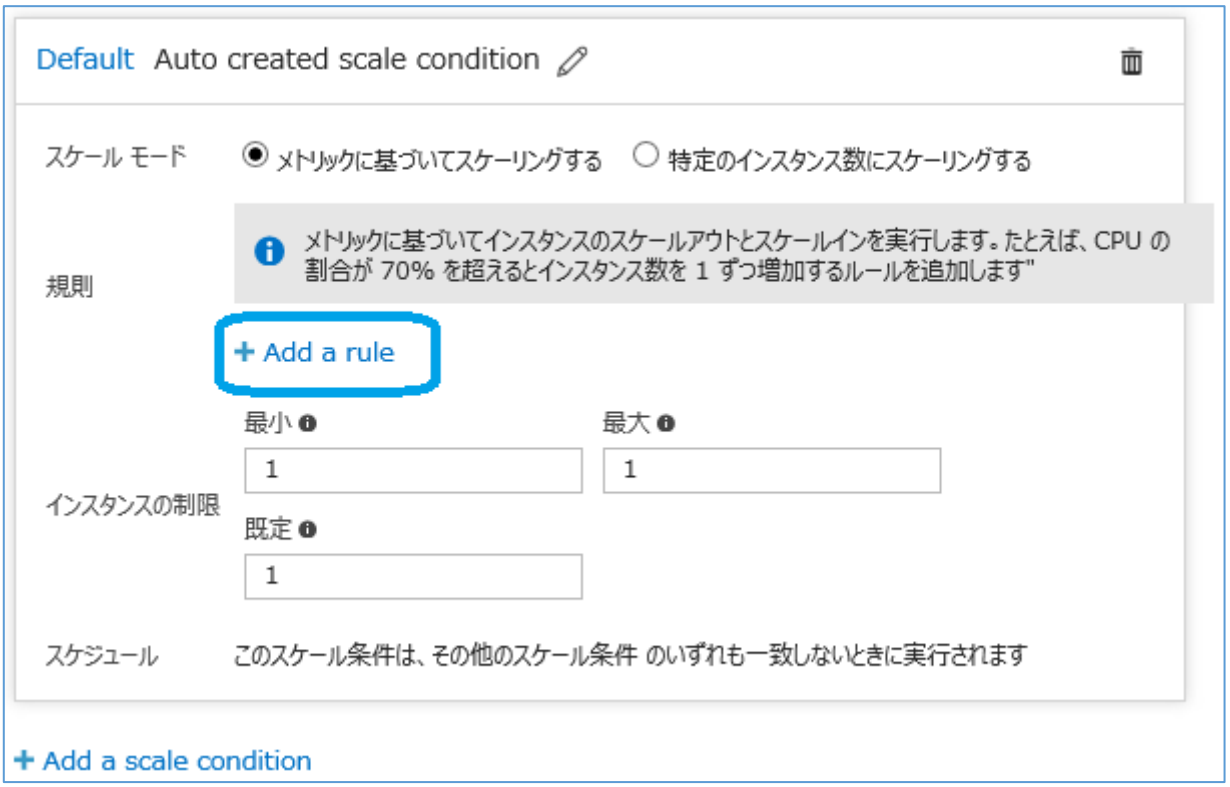
自動スケールの設定画面が表示されます。



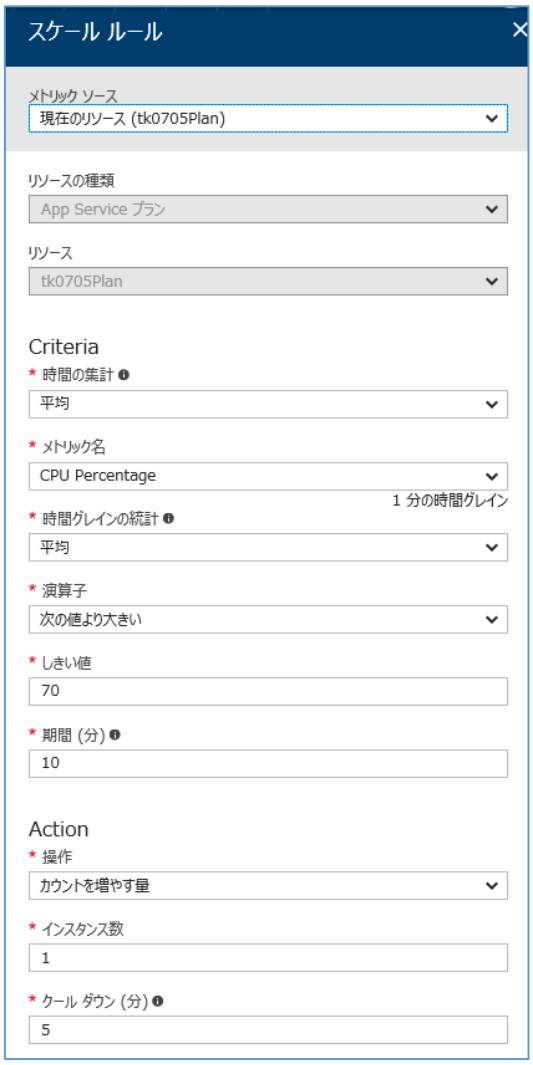
7. 自動スケールの設定を行います。ここでは以下の条件でスケール設定を行います。

・CPU 使用率が 80%を上回った場合、1 インスタンス増加
・CPU 使用率が 20%を下回った場合、1 インスタンス減少
・デフォルト 1 インスタンス、上限 3 インスタンス

表示されている Default ルールから「+Add a rule」をクリックしてください。



スケールルールのブレードが表示されます。

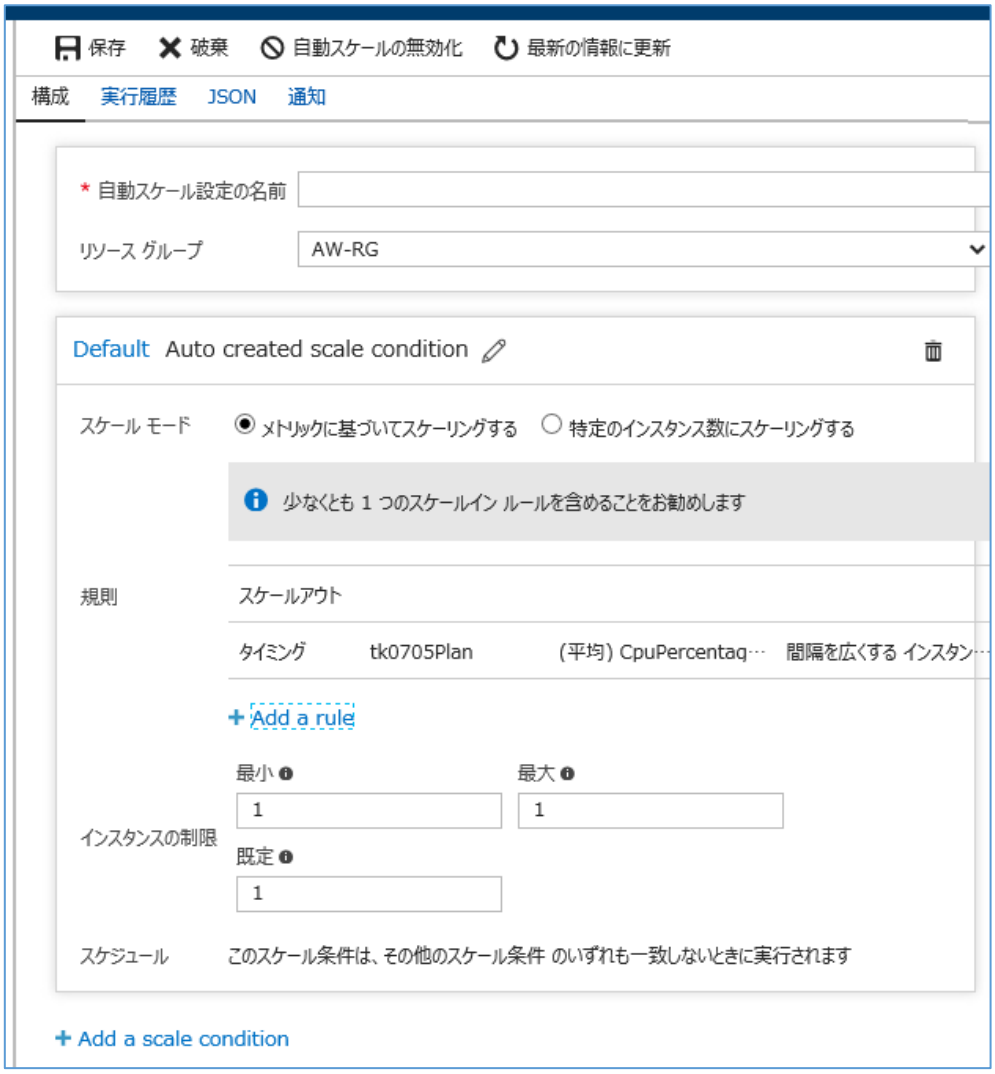


設定する内容は以下の通りです。

パラメーター	説明	値
時間の集計	指定された期間での統計量を指定します。 ここでは 1 分毎の平均 CPU 使用率の 10 分間(期間で指定された間隔での)平均を指定します。	平均
メトリック名		CPU　Percentage
時間グレインの統計		平均
演算子	-	次の値より大きい
しきい値	-	80
期間(分)	-	10
操作	-	カウントを増やす量
インスタンス数	-	1
クールダウン（分）	スケール操作後に次の操作を行うまでの最小間隔を指定します。	5

※グレイン（grain）：「粒」

設定後、[追加]ボタンをクリックします。ルールが追加され、以下のように表示されます。



同様にして、スケールイン（減少方向）のルールを作成します。設定する内容は以下の通りです。

パラメーター	説明	値
時間の集計	指定された期間での統計量を指定します。 ここでは 1 分毎の平均 CPU 使用率の 10 分間(期間で指定された間隔での)平均を指定します。	平均
メトリック名		CPU　Percentage
時間グレインの統計		平均
演算子	-	より小さい
しきい値	-	20
期間(分)	-	10
操作	-	カウントを減らす量
インスタンス数	-	1
クールダウン（分）	スケール操作後に次の操作を行うまでの最小間隔を指定します。	5

追加後、以下のように表示されます。

8. [インスタンスの制限]により、最大・最小のインスタンス数を設定します。

ここでは上限を 3 までに制限します。

パラメーター	説明	今回の設定
最小	最小インスタンス数	1
最大	最大インスタンス数	3
既定	デフォルトのインスタンス数	1

上記設定が終わったところで、[自動スケール設定の名前]を入力し、[保存]ボタンをクリックします。

パラメーター	説明	今回の設定
自動スケール設定の名前		任意のわかりやすい名前

これにより自動スケールの設定が保存され、有効化されます。

7. Storage の作成と管理

Azure 上に展開されている Web App からデータの保存・読み込みを行うために、Storage を利用します。

- ストレージとは

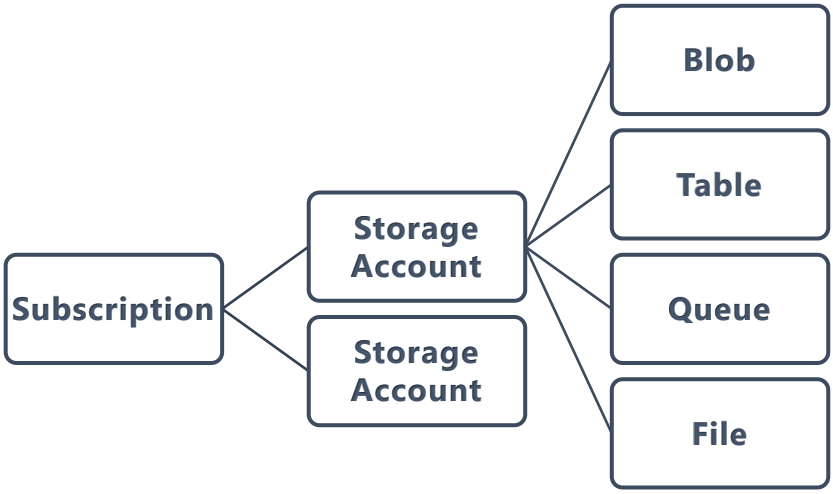
ストレージサービスは、データを永続的に保存するためのサービスです。Web App のローカルストレージではなく、独立したサービスとしてのストレージであるため、以下の用途に適しています。

・ Web アプリケーションの静的データ、コンテンツ
・ オンプレミスを含む他システムとのデータ連係
・ ログの出力、バックアップの保持

一般的に SQL Database 等よりも安価に利用できるため、例えば、従来データベースアプリケーションでログテーブルに出力していたデータをストレージに出力することで、運用コストを抑えることが可能になります。
また、データセンター内で最低でも三重化されますが、加えて地理冗長構成を選択可能なので、可用性も確保されます。

- ストレージアカウント

ストレージを利用するためには、「ストレージアカウント」を作成する必要があります。ストレージアカウントはサブスクリプション内に複数作成することが可能です。このストレージアカウント内、前述の Blob, Table, Queue, File の各サービスが展開されます。



ストレージアカウント毎にアクセスキーが発行されるので、このキーを持って認証します。
（キーは 2 つ発行されますが、機能に違いはありません。）
ストレージアカウントは Azure 管理ポータルから作成可能です。

- ストレージの種類

ストレージは以下の種類が提供されます。

・ Blob (Binary Large Object)

ファイルシステムと同様、非構造化データを取り扱うことが出来ます。

・ Table

表構造のデータを格納可能な Key Value ストアです。

・ Queue

First In First Out のメッセージ配信ストレージです。システム間連係に利用されます。

・ File

ファイル共有サービスです。

- Blob

Blob には以下 3 種類の Blob があります。

・ Block Blob

Streaming に最適化され、最大 4.75TB までのファイルを格納可能。

・ Page Blob

ランダムアクセスに最適化され、最大 8TB までのファイルを格納可能。

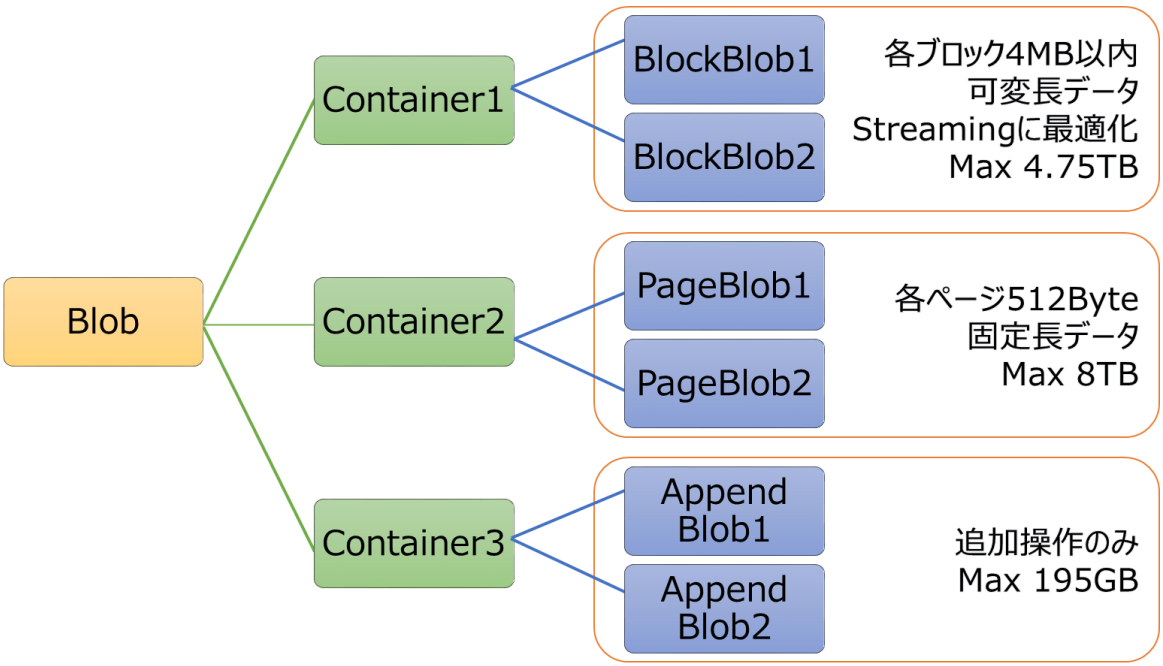
仮想マシンの仮想ハードディスク等に利用される。

・ Append Blob (追加 Blob)

追加操作のみ可能。約 195GB までのファイルを格納可能。

ログ保存などに最適。

Blob の構造を図式化すると下図のようになります。



各 Blob は以下のような URL を持ちます。

http://<アカウント名>.blob.core.windows.net/<コンテナ名>/<Blob 名>

命名規則としては以下の通りです。

- コンテナ名
  文字 [a-z] または数字 [0-9] と”-” (ハイフン)のみ使用可能
  ハイフンは先頭・末尾には使用できない
  ハイフンを連続して使用できない
  全て小文字
  3～63 文字
- Blob 名
  1～1024 文字 任意の文字が使用可能だが、適切にエスケープする必要はある
  ※”/”（スラッシュ）も使用可能。これを利用して仮想ディレクトリのような利用も可能です。

● Table

Table は Excel のような表形式のデータストアです。ストレージアカウント内に複数の Table を作成することが出来、Table 内には複数のエンティティ（Excel で言う「行」）で構成されます。
エンティティは複数のプロパティ（Excel で言う「列」）で構成されます。
このエンティティは固定のスキーマではありません。そのため、Database Table のように「スキーマを厳密に決定する」アプローチとは異なる点に注意してください。

テーブルのエンティティは、以下のシステムプロパティが含まれます。

- ParitionKey
  文字列型

- RowKey
  文字列型
- Timestamp
  システムで自動生成：読み取り専用

上記以外のプロパティは自由に設定することが出来ますが、以下の制限があります。

- プロパティ名は最大 255 文字
- 1 エンティティ当たりのデータサイズは 1MB まで（プロパティ名、型情報を含む）
- 利用できるデータ型は以下の通りです。
  Binary (バイト配列), Bool, Datetime, Double, GUID, Int, Int64, String

Table を利用する際には、上記システムプロパティの"PartitionKey"と"RowKey"で一意になるように設計してください。このうち、PartitionKey の取り扱いには若干注意が必要です。
データセンター内のストレージでパーティーション分割が行われる際、PartitionKey 毎に編成されます。従って、異なる PartitionKey をまたがった検索には注意が必要です。

PartitionKey と RowKey を含まない検索にも十分な注意が必要です。Database のような Index 機構は提供されませんので、件数によっては深刻なパフォーマンス劣化を引き起こす恐れがあります。

しかしながら適切な設計 – Index Table Pattern など – を用いることにより、パフォーマンス劣化を防ぎつつ Table の利点を享受することが可能です。
"Index Table pattern"については以下を参照してください。
[Azure – Architecture – Cloud Design Patterns – Index Table pattern]
https://docs.microsoft.com/en-us/azure/architecture/patterns/index-table

- Queue

Queue は First In First Out (先入れ先出し)のメッセージ配信ストレージです。フロントエンドとなる Web App と Web Job 等のバックエンドの処理、あるいはオンプレミス等の他システムとの連携に使用される場合が多いです。

- File

File は SMB3.0 を使用したファイル共有サービスを提供します。仮想マシン、あるいはオンプレミスからのネットワーク共有を可能にします。

ここでは、アプリケーションのアクセスログを Azure Table に保存する演習を行います。

1. まずストレージアカウントの作成を行います。[Azure 管理ポータル]の画面左上の[+]→[Storage]→［ストレージアカウント – BLOB、File、Table、Queue］をクリックします。

2. [ストレージアカウントの作成]ブレードが表示されます。以下のように設定し[作成]をクリックしてください。

パラメーター	値
サブスクリプション	お使いのサブスクリプションを選択してください
リソースグループ	AW-RG
ストレージアカウント名	（任意のわかりやすい名前）
場所	作成した Web Apps と同じ場所
パフォーマンス	Standard
アカウントの種類	StorageV2 (汎用 v2)
レプリケーション	ローカル冗長ストレージ(LRS)
アクセスレベル	ホット
[詳細]-安全な転送が必須	有効

[詳細]-仮想ネットワーク	すべてのネットワーク

「アカウントの種類」は「Storage (汎用 v1)」「StorageV2 (汎用 v2)」と「Blob ストレージ」の三つが選択可能です。本演習では「StorageV2 (汎用 v2)」を利用します。「Blob ストレージ」は Blob のみ使用可能なストレージアカウントとなり、価格体系も異なります。「StorageV2 (汎用 v2)」と「Blob ストレージ」には「ホット層」と「クール層」があります。

・ ホット層

　　頻繁に読み書きされることを想定したストレージ

　　ストレージコストは高め、トランザクション料金は安め

・ クール層

　　読み書きなどの操作が少ないことを想定したストレージ

　　ストレージコストは安め、トランザクション料金は高め

例えばバックアップデータを保持する場合には、「Blob ストレージ」の「クール層」が適しています。

「パフォーマンス」は「Premium」と「Standard」が用意されています。

・ Premium

　　SSD ベースのストレージです。契約容量単位で課金となります。

・ Standard

　　HDD ベースのストレージです。使用した容量とトランザクションで課金となります。

「レプリケーション」は冗長性に関する選択です。「ローカル冗長(LRS)」「ゾーン冗長(ZRS)」「ジオ冗長(GRS)」「読み取りアクセスジオ冗長(RA GRS)」が用意されています。

・ ローカル冗長ストレージ (LRS)
　　単一リージョンの単一データセンター内で三重化されます。
・ ゾーン冗長ストレージ (ZRS)
　　1 or 2 つのリージョン内の複数データセンター内で三重化されます。
　　ゾーン冗長が適用されるのは ブロック BLOB のみです。(Table, Queue, File は利用できません。)
　　アカウントの種類で StorageV2 (General purpose v2)を選択した場合には、ゾーン冗長は選択できません。
・ ジオ冗長レプリケーション (GRS)
　　プライマリリージョンで三重化され、セカンダリリージョンでも三重化されます (計 六重化)。
　　プライマリリージョンで障害が発生すると、セカンダリにフェールオーバーされます。
・ 読み取りアクセスジオ冗長(RA GRS)
　　GRS + セカンダリからの読み取りアクセス　(計 六重化) が提供されます。
　　通常運用時にも明示的にセカンダリから読み取りを行うことが可能です。
　　99.99% の読み取り可用性を保証します。

以上を確認の上「作成」ボタンをクリックしてください。

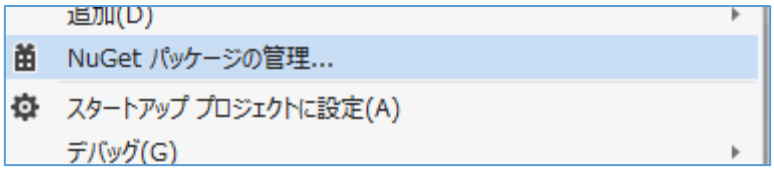
3. ストレージアカウントが作成されます。画面右上の［通知］アイコンをクリックし、一覧に［デプロイメントが成功しました］が表示されれば、作成は終了です。

4. 作成したストレージを扱う Web アプリケーションを作成します。Visual Studio から、先ほどの演習で使用したプロジェクトを開きます。

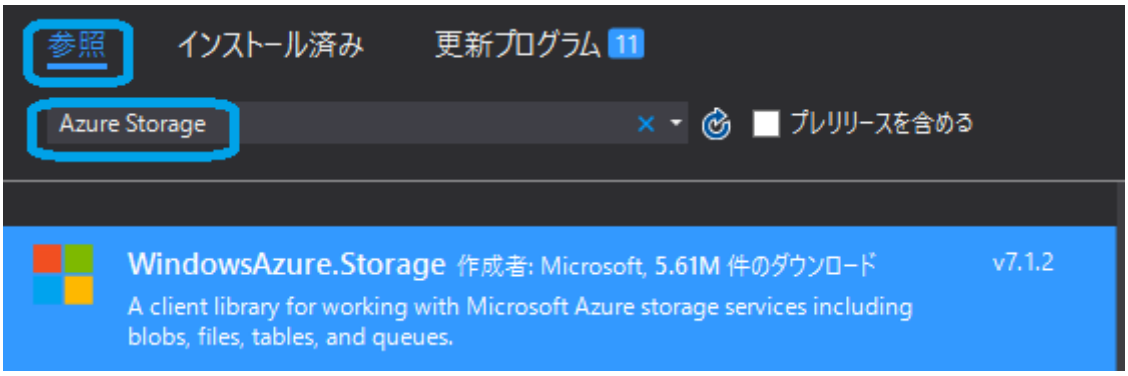
5. Cloud Explorer 上に使用しているサブスクリプションが表示されており、かつ、[Storage Accounts]を展開すると先ほど作成したストレージアカウントが表示されていることを確認してください。表示されていない場合には、サーバーエクスプローラーの[Azure]アイコンを右クリックし、[Microsoft Azure サブスクリプションへの接続]からサインイン操作を行ってください。

これよりアプリケーションからストレージへのアクセスを実装していきます。
ストレージは REST API を提供していますが、.NET 環境からは REST をラップするクライアントライブラリ “Storage Client API”が提供されていますので、これを使用します。

6. ソリューションエクスプローラーよりプロジェクトを右クリックし、[NuGet パッケージの管理]をクリックします。



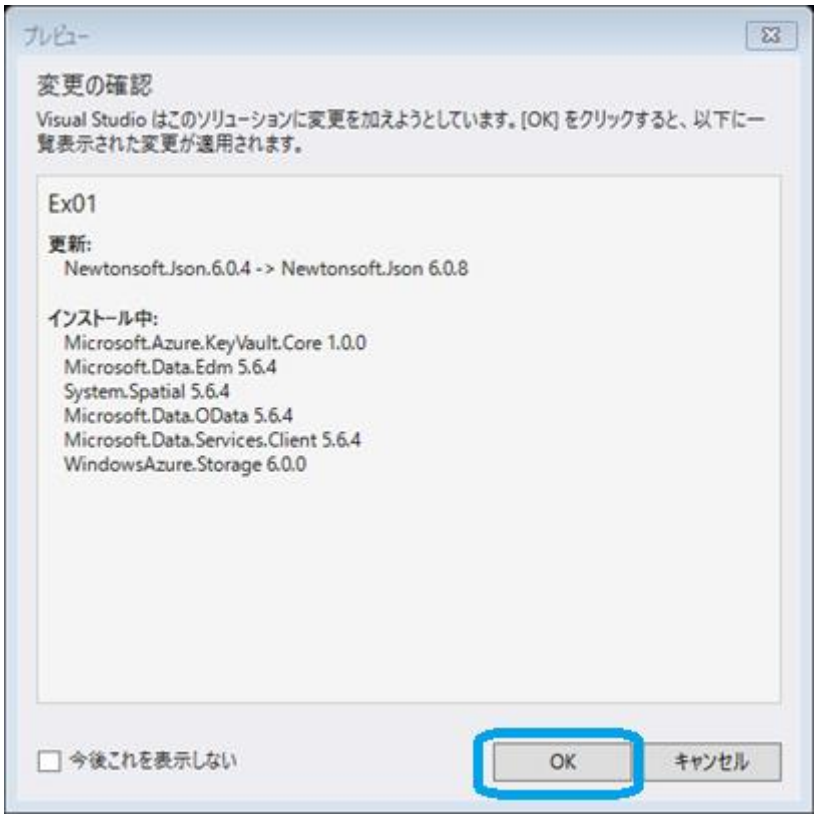
[NuGet パッケージマネージャー]が表示されます。「参照」を選択したうえで、「検索」に”Azure Storage”を入力すると、Azure Storage に関するパッケージが検索されます。



表示された結果から「WindowsAzure.Storage」を選択し、[インストール]ボタンをクリックします。
※表示されない場合には、「参照」を選択していることを確認してください。

※実際に表示されるバージョン番号は上図と異なる可能性があります。

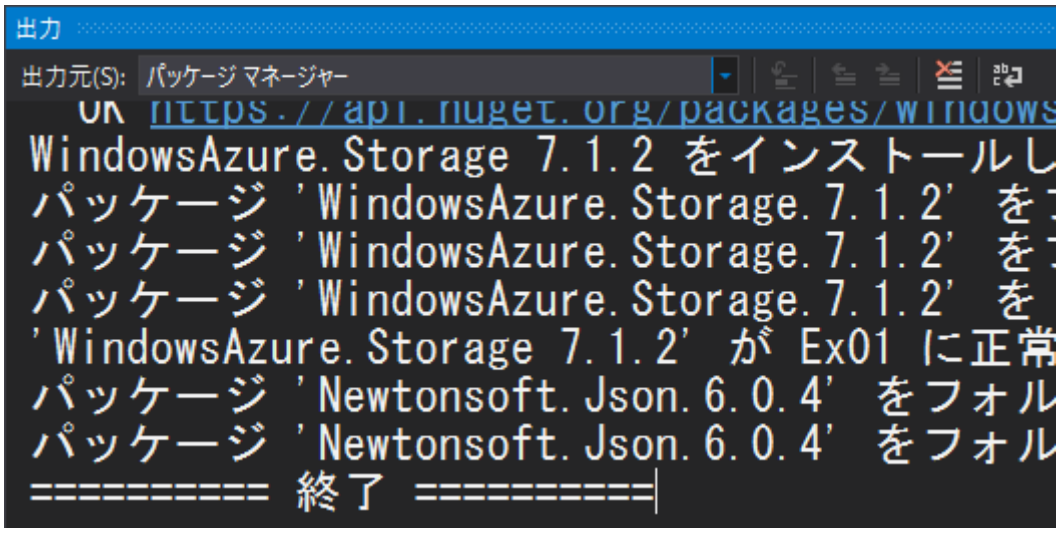
[変更の確認]ダイアログが表示されますので、[OK]をクリックします。



※表示されているモジュール名、バージョン番号は、実際と異なる可能性があります。

[ライセンスへの同意]ダイアログが表示されますので、[同意する]をクリックします。

[同意する]をクリックすると、出力ウィンドウに以下のように表示され、関連するモジュールの追加が行われます。



※実際に表示される内容は上記と異なる可能性があります。

7. Azure ストレージへの接続文字列を設定します。ソリューションエクスプローラーより、web.config ファイルを開きます。

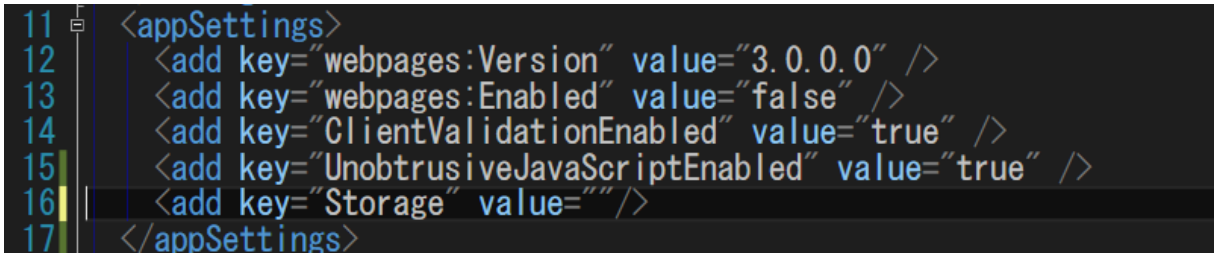
8. Web.config の/configuration 内 appSettings 要素に項目を追加します。

11 行目付近にすでに appSettings が追加されていますので、その中に記述します。

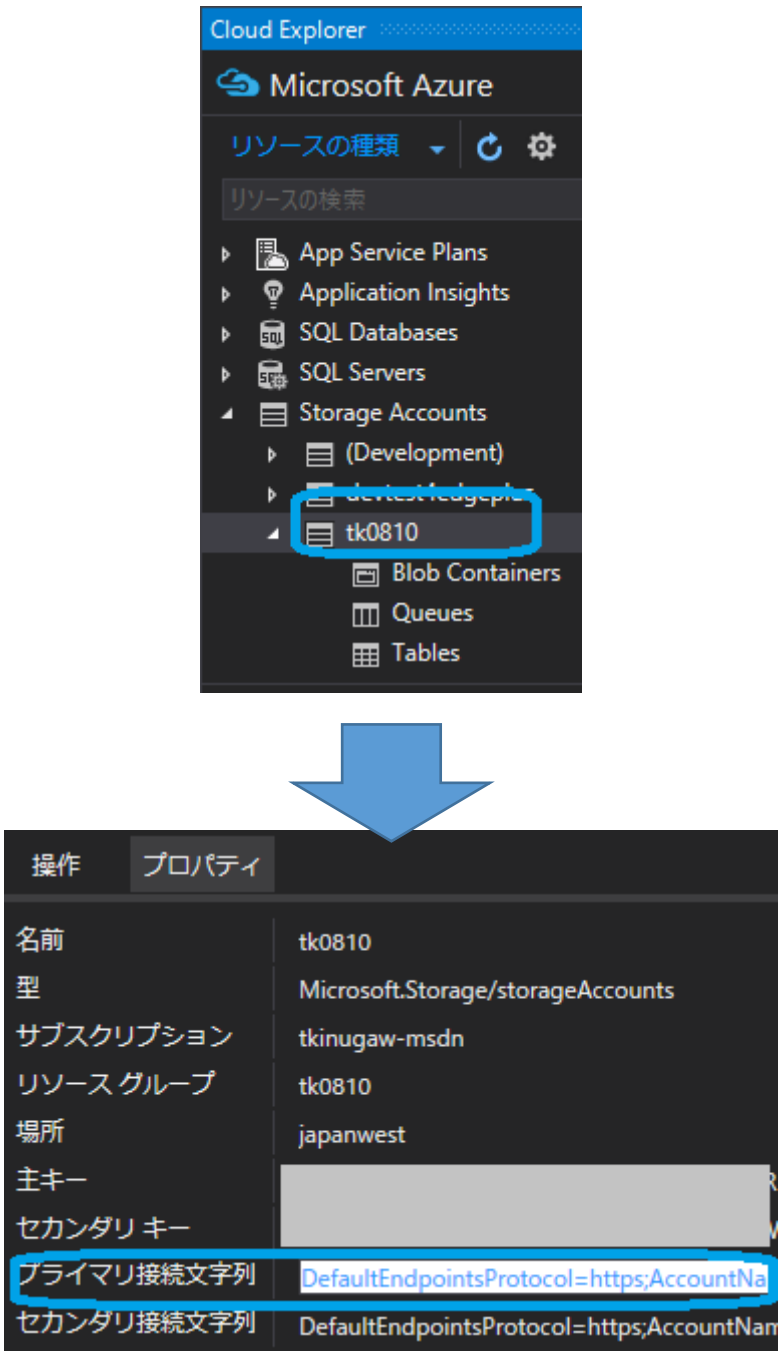
```
<add key="Storage" value=""/>
```

※この Exercise では"Storage"という名前で追加します。

Visual Studio 上では以下のように表示されます。



9. 実際に使用するストレージの接続文字列を取得します。[Cloud Explorer]より [Storage Account]を展開し、使用するストレージアカウントを選択します。選択後、プロパティの[プライマリ接続文字列]を選択します。

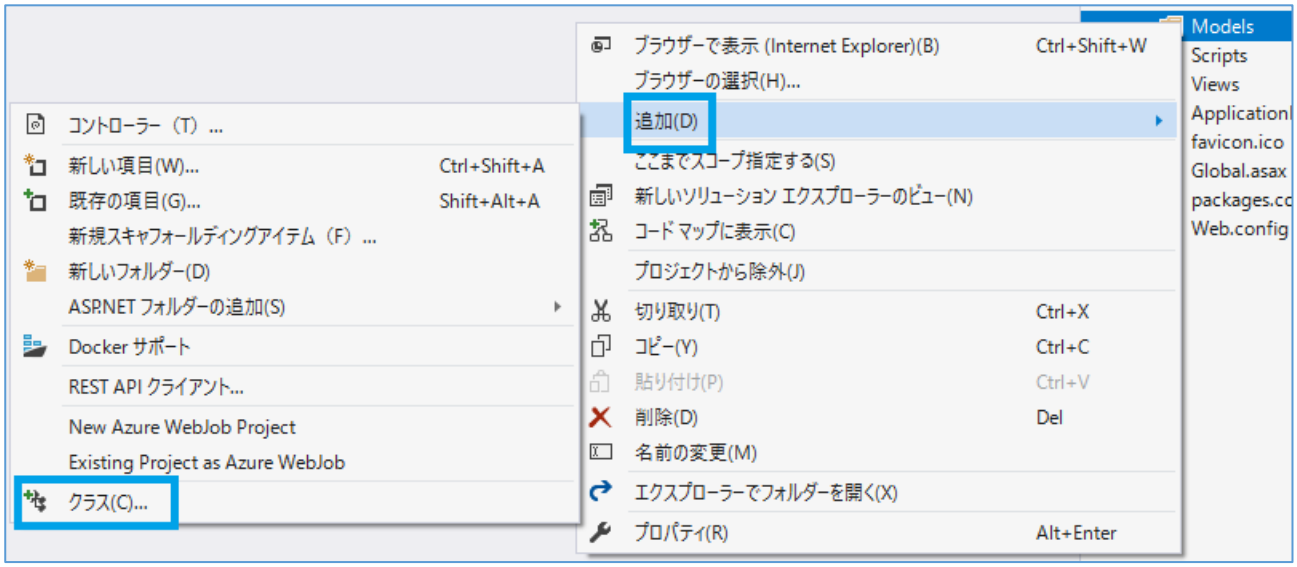


[Ctrl]+[A] で全選択し、[Ctrl]+[C] でコピーすると便利です。

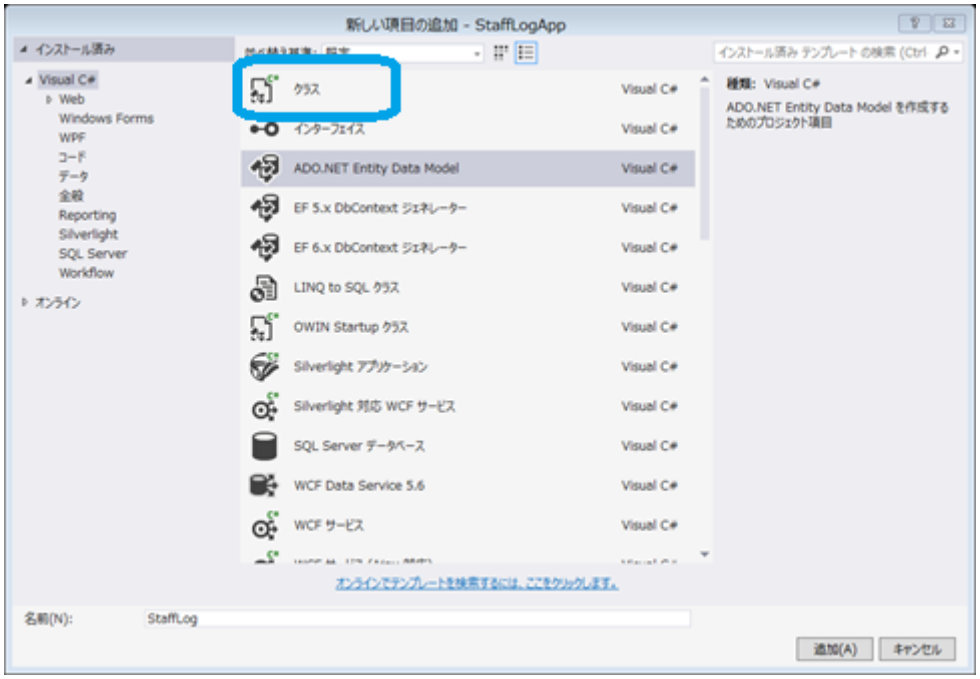
※ここではプライマリ接続文字列を選択しましたが、セカンダリ接続文字列でも以下同じ結果となります。

10. Web.config に戻り、先ほど追加した<add key="Storage" … の value=""の二重引用符内にペーストし、接続文字列を設定します。

11. [ソリューションエクスプローラー]-[（プロジェクト）]-[Models]を右クリックします。表示されたコンテキストメニューより[追加]-[クラス]の順で選択します。



選択すると[新しい項目の追加]ウィンドウが表示されます。[クラス]を選択し、[名前(N)]には「AccessLog」と入力し[追加(A)]をクリックします。



12. 作成された AccessLog クラスに対して編集を行います。まず、コードの先頭に以下の using を追加します。

```
using Microsoft.WindowsAzure.Storage.Table;
```

13. クラスの中身を記述します。以下のように記述してください。

※ TableEntity クラスを継承する点に注意してください。

```
Public class AccessLog : TableEntity
{
        public DateTime LogDateTime { get; set; }
        public string LogMessage { get; set; }

        public AccessLog()
        {
            PartitionKey = nameof(AccessLog);
            RowKey = Guid.NewGuid().ToString();
            LogDateTime = DateTime.UtcNow;
        }
}
```

14. Controller 内の HomeController.cs をダブルクリックして開きます。

先ほどと同様に、以下のように using を追加してください。

```
using Microsoft.WindowsAzure.Storage;
using Microsoft.WindowsAzure.Storage.Table;
using System.Configuration;
using [デフォルト名前空間名].Models;
```

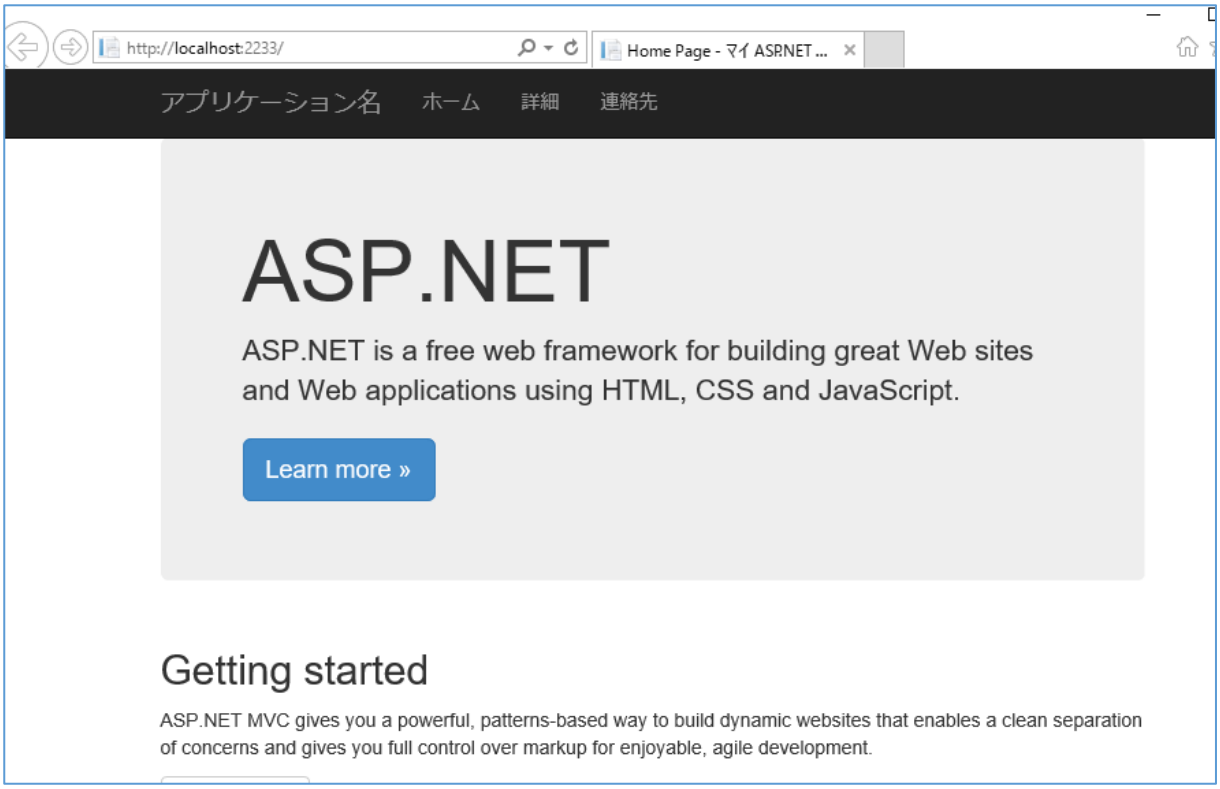
※[デフォルト名前空間名]は、特に設定を変更していない場合にはプロジェクト名になります。

15. Index メソッド内に以下のように記述します。

```
public ActionResult Index()
{
    var conn = ConfigurationManager.AppSettings["Storage"];
    var account = CloudStorageAccount.Parse(conn);
    var client = account.CreateCloudTableClient();

    var table = client.GetTableReference("accesslog");
    table.CreateIfNotExists();

    var msg = new AccessLog(){LogMessage =
                    $"{nameof(Index)} - GET でアクセスしました" };
    var insert = TableOperation.Insert(msg);
    table.Execute(insert);

    return View();
}
```

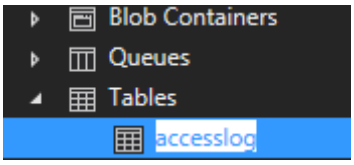
※紙面の都合上一部折り返しております。

16. キーボードから[F5]キーを押してデバッグ実行を行います。下図のようにブラウザ上にアプリケーションが表示されます。

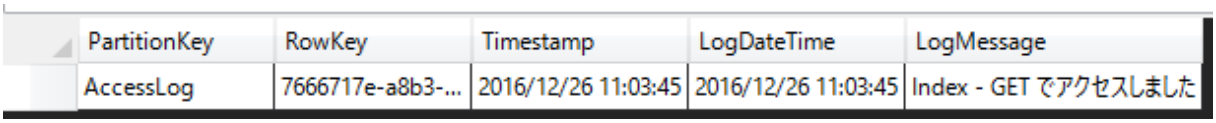


表示されたことを確認した後、デバッグを停止してかまいません。

17. Cloud Explorer より、作成した Storage Account の Tables 内を確認してください。



上図のように “accesslog”テーブルが作成されているので、ダブルクリックで内容を表示します。以下のように table に情報が出力されていることを確認してください。



PartitionKey	RowKey	Timestamp	LogDateTime	LogMessage
AccessLog	7666717e-a8b3-...	2016/12/26 11:03:45	2016/12/26 11:03:45	Index - GET でアクセスしました

18. 先ほどの演習と同様に、Azure 上にアプリケーションを展開します。先ほどの演習で正しく展開できていれば、Visual Studio が展開に必要な情報を保持していますので、先ほどの演習の手順の途中から再開できます。

19. 展開後、デバッグ時と同様にストレージの accesslog テーブルを Cloud Explorer で開きます。同様に情報が出力されていることを確認してください。また、[F5]キーで再読み込みをするたびにエンティティが増加していくことを確認してください。

8. SQL Database の作成

Azure 上に展開されている Web App で使用する SQL Database を作成します。

- SQL Database とは

SQL Database とは Azure 上で提供される Database Service です。時に"DBaaS" (Database as a Service) とも言われます。高可用性構成、バックアップ、地理冗長構成、障害復旧などはサービスとして提供されています。従って、仮想マシン上に SQL Server をインストールする場合と異なり、細かな設定・管理を行う必要はありません。SQL Database と仮想マシン上の SQL Server の違いについては、以下も参照してください。

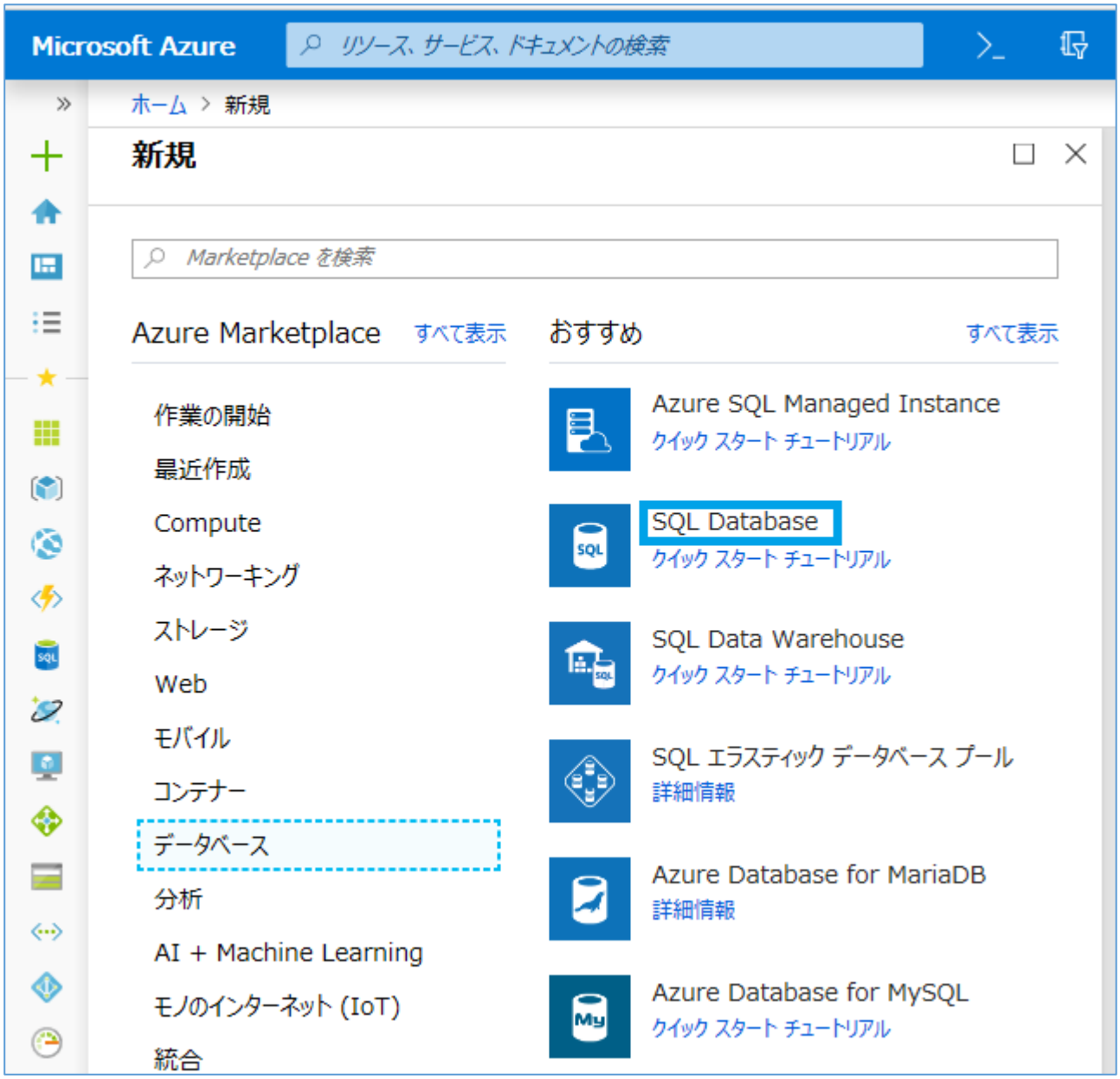
[Azure – SQL Database – クラウド SQL Server オプションの選択：Azure SQL (PaaS) Database または Azure VM (IaaS) の SQL Server]

https://docs.microsoft.com/ja-jp/azure/sql-database/sql-database-paas-vs-sql-server-iaas

アプリケーション開発者から見た際には、SQL Database のテクノロジは SQL Server と同一となります。従って、SSMS (SQL Server Management Studio) または Visual Studio からの操作・コーディングにおいては同一です。但し、Azure 環境一般の注意事項（Time zone, Culture）については留意する必要があります。

また、SQL Database は IP アドレスベースのホワイトリストにより Firewall を設定しています。特に開発中グローバル IP アドレスの変更による接続不可が発生する可能性がありますので注意してください。

1. [Azure 管理ポータル]の[ダッシュボード]が表示されますので、画面左の[新規]をクリックして、[データベース]から[SQL Database]をクリックします。

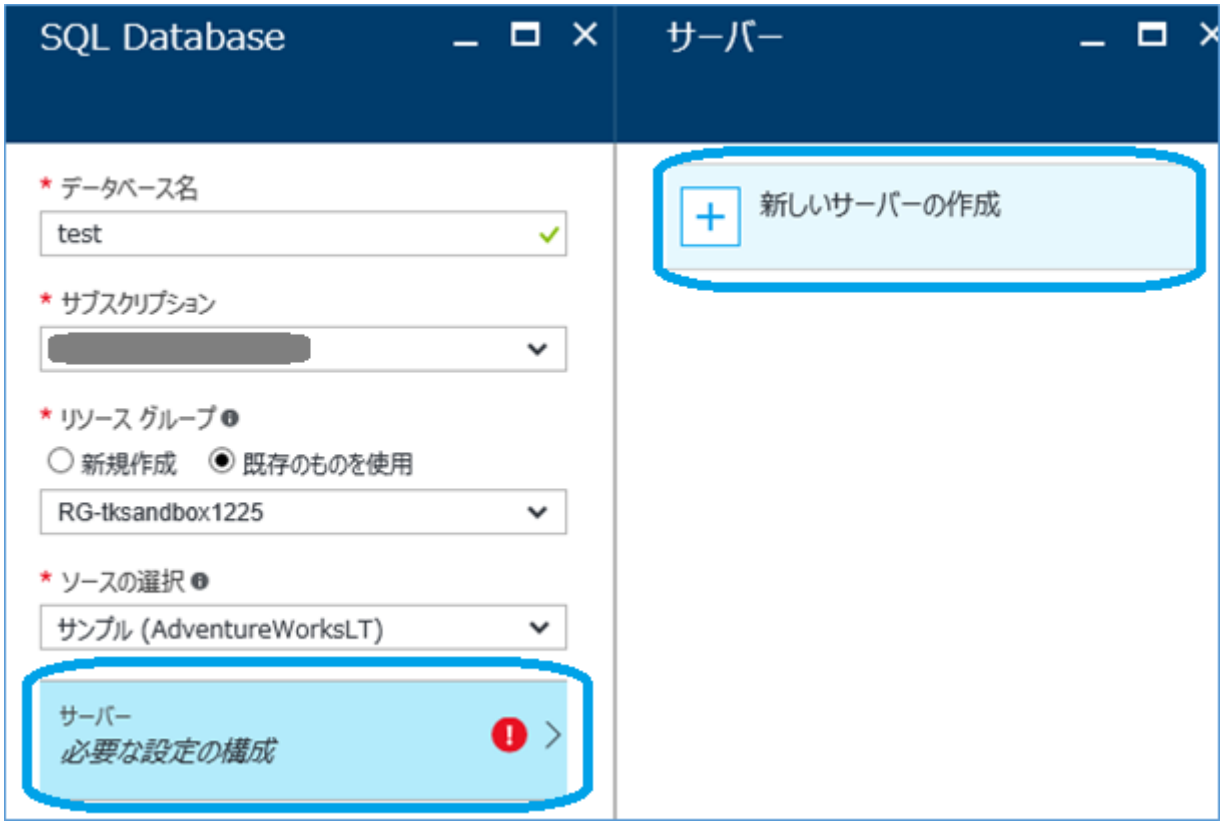


2. SQL Database の新規作成に必要なパラメーターを入力します。

入力内容は以下の通りです。

パラメーター	値
データベース名	（任意のわかりやすい名前）
サブスクリプション	お使いのサブスクリプションを選択してください
リソースグループ名	[既存のものを使用] AW-RG
ソースの選択	サンプル(AdventureWorksLT)
SQL エラスティックプールを使用しますか？	後で
価格レベル	Basic
ダッシュボードにピン留めする	☑

3. [サーバー]欄をクリックして新しいサーバーを作成します。サーバーの選択ブレードが表示されますので、[新しいサーバーの作成]をクリックしてください。



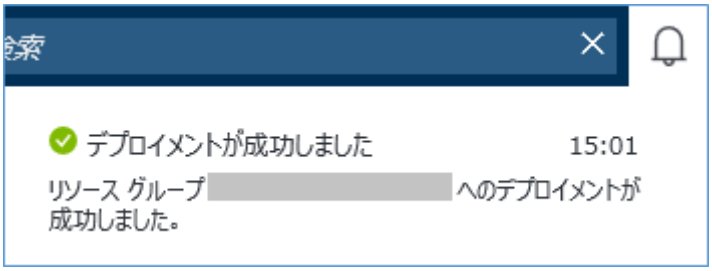
作成するサーバーのパラメーターを入力します。

パラメーター	値
サーバー名	（任意のわかりやすい名前）
サーバー管理者ログイン	Dbadmin
パスワード	Pa$$w0rd1234
パスワードの確認	Pa$$w0rd1234
場所	作成した Web Apps と同じ場所
Azure サービスにサーバーへのアクセスを許可する	☑

上記入力後[選択]をクリックしてください。

4. [SQL Database]のブレードに戻って[作成]をクリックしてください。これにより SQL Database が作成されます。しばらく時間がかかります（SQL Database の作成が完了すると画面右上の［通知］アイコンをクリックした際、一覧に［デプロイメントが成功しました］が表示されます)。

   デプロイが成功すると、通知ステータスに以下のように表示されます。



5. Azure Portal のダッシュボード上に作成された SQL Database がピン留めされています。ピン留めされていない場合には、左メニューの[SQL Database]を選択し、一覧から作成された Database を選択してください。

   作成された SQL Database に対して、サーバーファイアウォールの設定を行います。

   表示されている[概要]のブレード上部に[サーバーファイアウォール]の項目がありますので、ここをクリックしてください。

6. [ファイアウォール設定]のブレードが表示されますので、[クライアント IP の追加]をクリックしてください。

クリックすると現在のクライアント IP アドレスが追加されますので、[保存]をクリックしてください。

これにより、SQL Database が作成され、かつ、開発環境から接続できるようになりました。

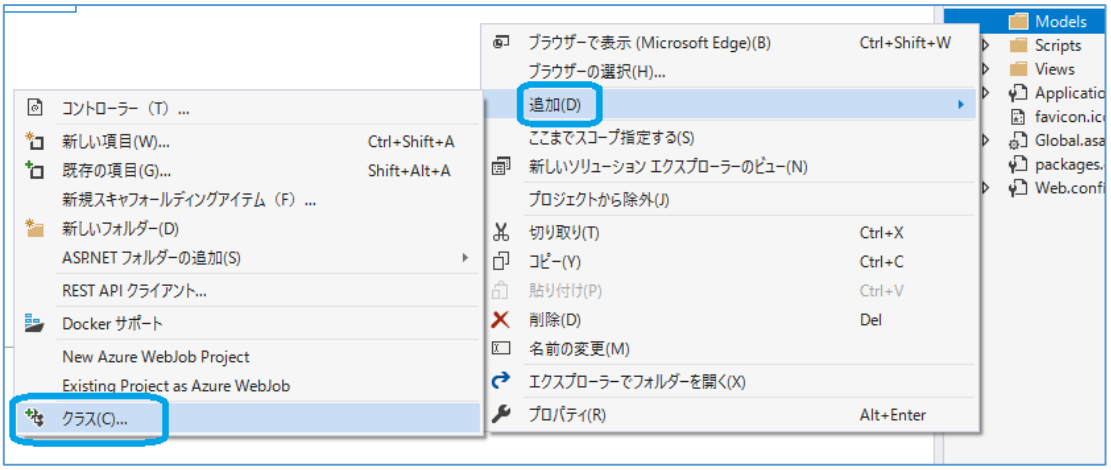
9. アプリケーションからの SQL Database の参照とデプロイ

作成した SQL Database に対して、アプリケーションから接続を行います。

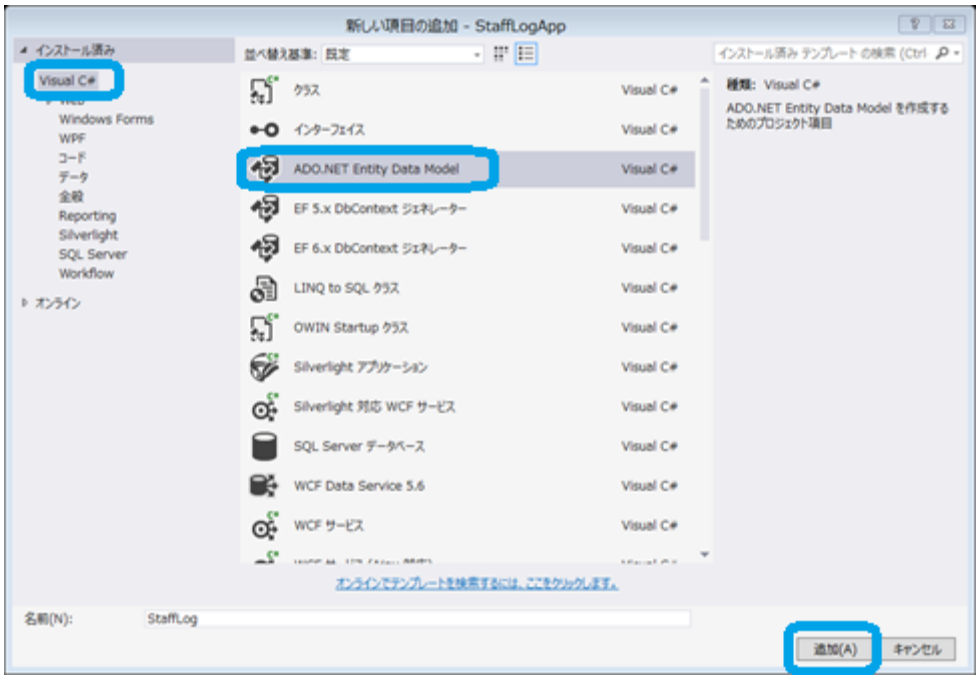
1. Visual Studio から、先ほどの演習で使用したプロジェクトを開きます。

2. サーバーエクスプローラー上に[Azure]が表示されており、かつ、[SQL データベース]を展開すると先ほど作成したデータベースが表示されていることを確認してください。表示されていない場合には、サーバーエクスプローラーの[Azure]アイコンを右クリックし、[Microsoft Azure サブスクリプションへの接続]からサインイン操作を行ってください。

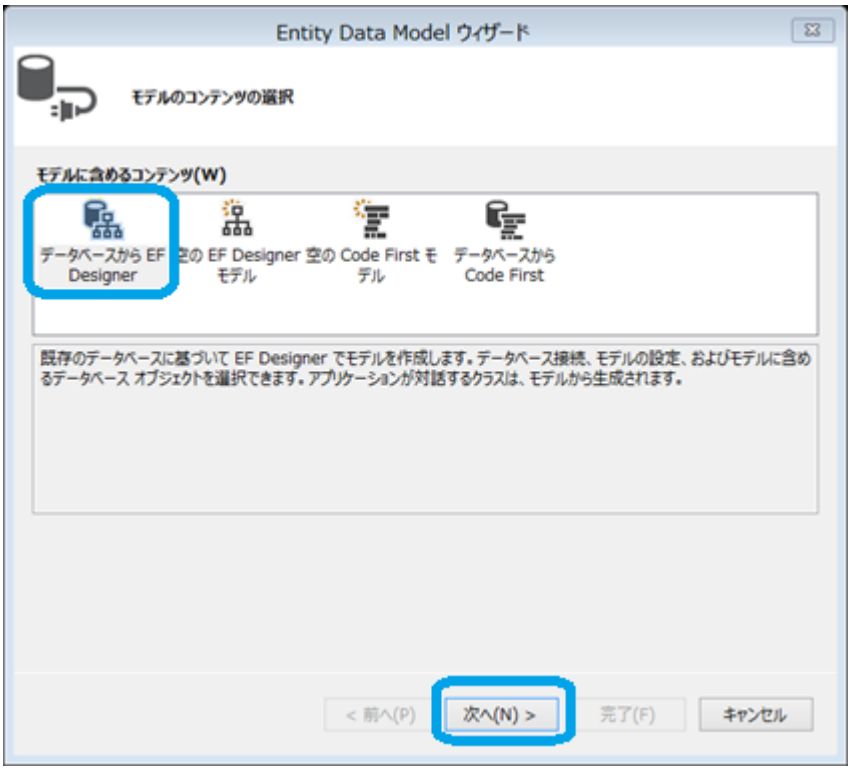
3. [ソリューションエクスプローラー]-[（プロジェクト）]-[Models]を右クリックします。表示されたコンテキストメニューより[追加]-[クラス]の順で選択します。



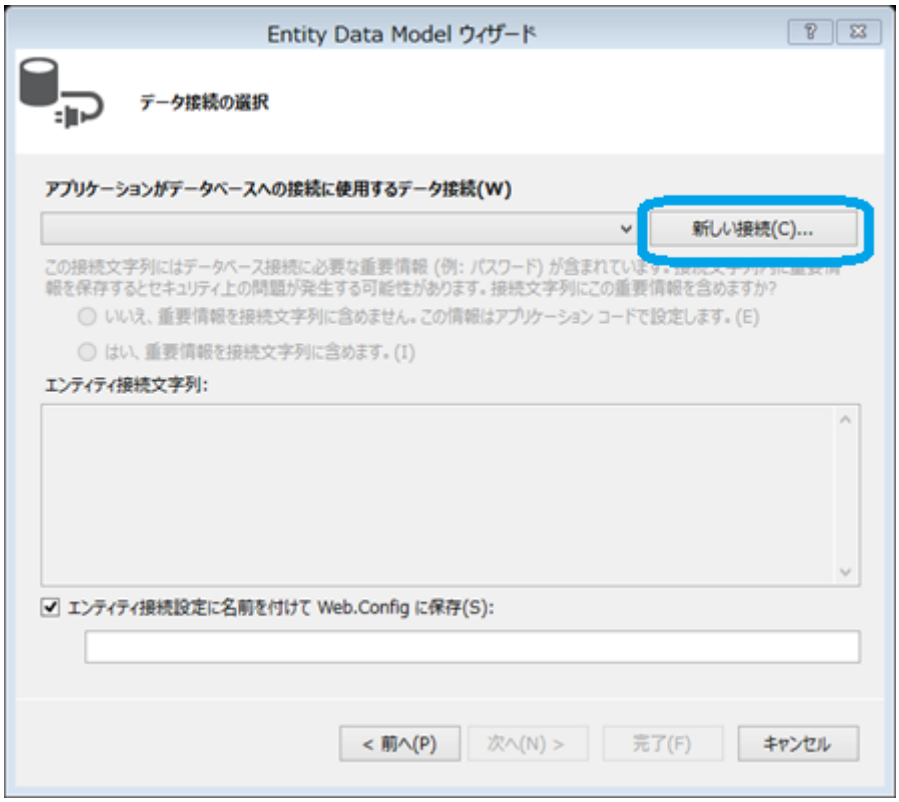
選択すると[新しい項目の追加]ウィンドウが表示されます。[ADO.NET Entity Data Model]を選択し、[名前(N)]には「Product」と入力し[追加(A)]をクリックします。



[Entity Data Model ウィザード]が起動します。このウィザードでは、既存のデータベースを元に Data Model を作成します。この画面では[データベースからの EF Designer]を選択し[次へ]をクリックします。



[データ接続の選択]画面に遷移します。初期状態では登録されているデータ接続は存在しませんので、[新しい接続]をクリックします。



[接続の追加]画面が表示されます。

パラメーター	値
サーバー名	先ほど作成したサーバー名
認証	SQL Server 認証
ユーザー名	Dbadmin
パスワード	Pa$$w0rd1234
データベース名	先ほど作成したデータベース名

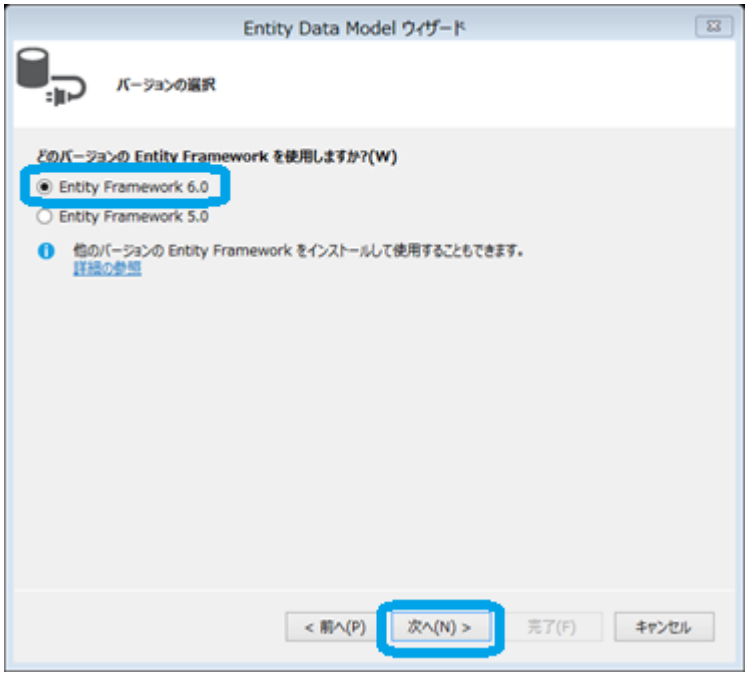
入力後[テスト接続(T)]をクリックし、接続が確立できることを確認してください。

確認後、[OK]をクリックしてください。先ほどの「データ接続の選択」画面に戻ります。

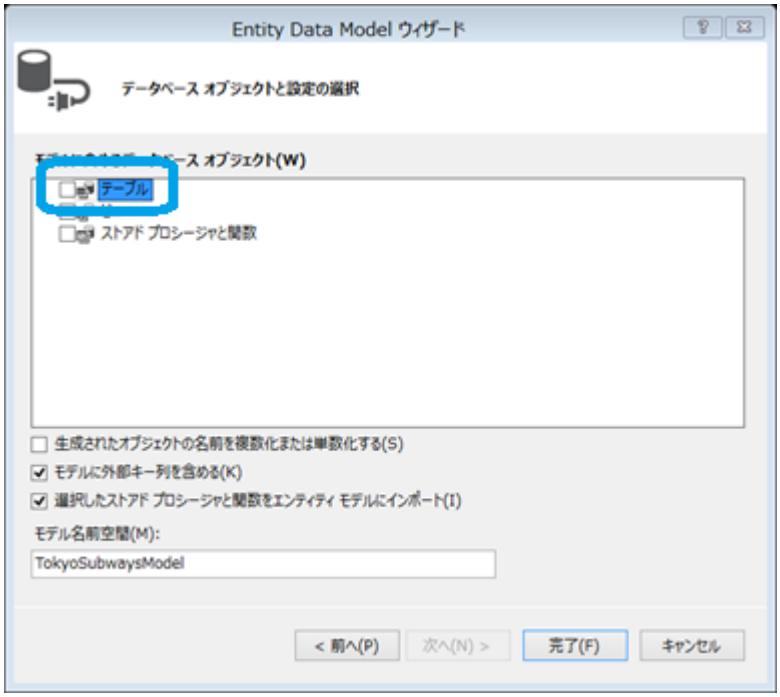
[エンティティ接続文字列]および[エンティティ接続設定に名前を付けて Web.Config に保存(S)]の項目が以下のように表示されます。（下図は例です。）

ラジオボタンの選択が求められます。ここでは[はい、重要情報を接続文字列に含めます(I)]を選択します。選択したのち「次へ」をクリックします。

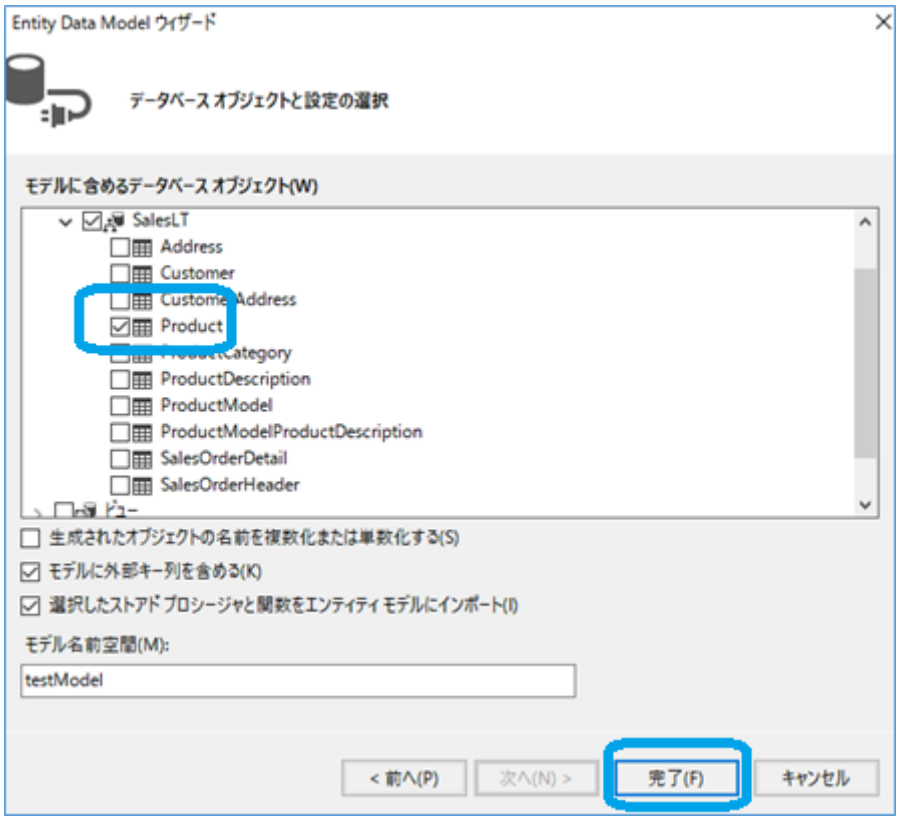
バージョンの選択画面が表示されますので、ここでは [Entity Framework 6.0]を選択し、[次へ]をクリックします。



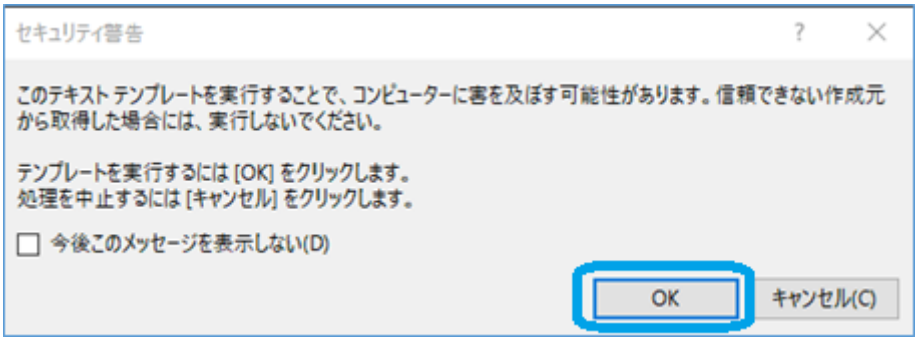
[データベースオブジェクトと設定の選択]画面に遷移します。



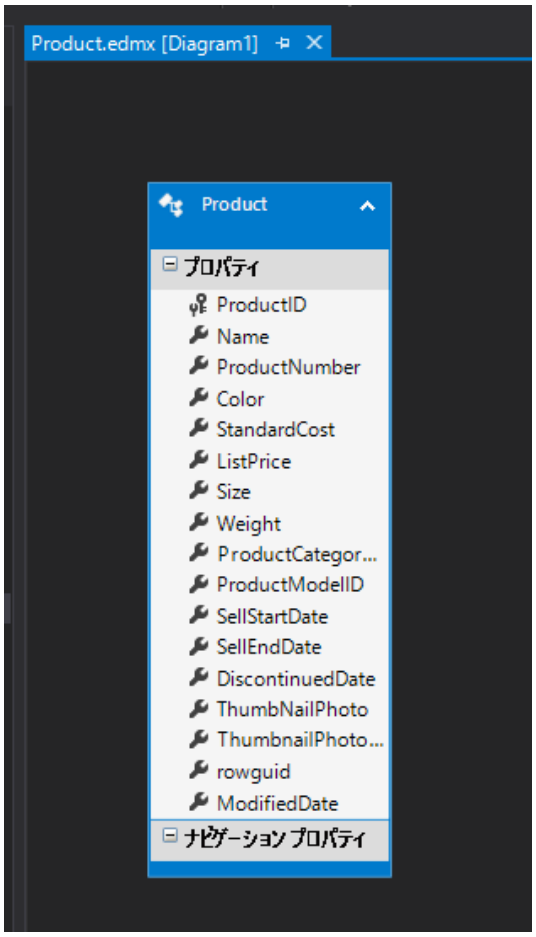
[テーブル]を展開し、[SalesLT]の[Product]をチェックしてください。チェックした後[完了]をクリックしてください。



※[セキュリティ警告]のダイアログが複数回表示されますが、[OK]をクリックしてください。



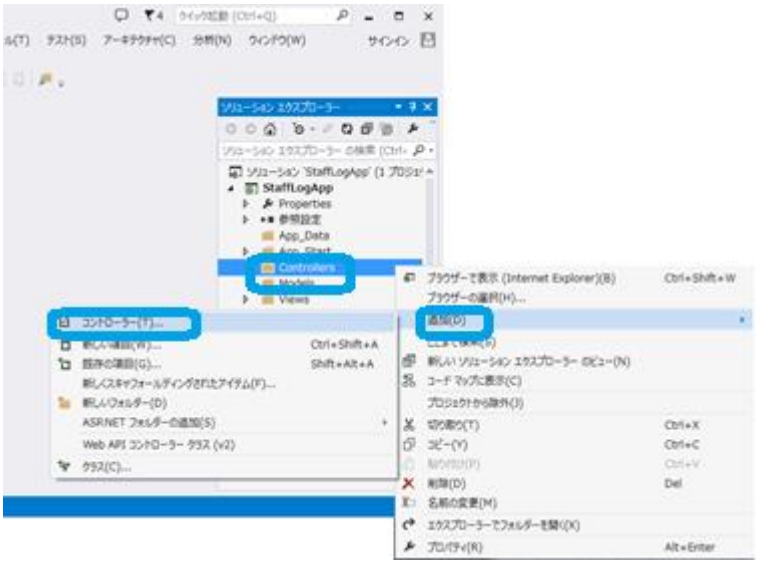
ウィザード画面が閉じ、以下のように表示されれば、Model の作成は完了です。



ここまで出来たところで、一旦ビルドします。

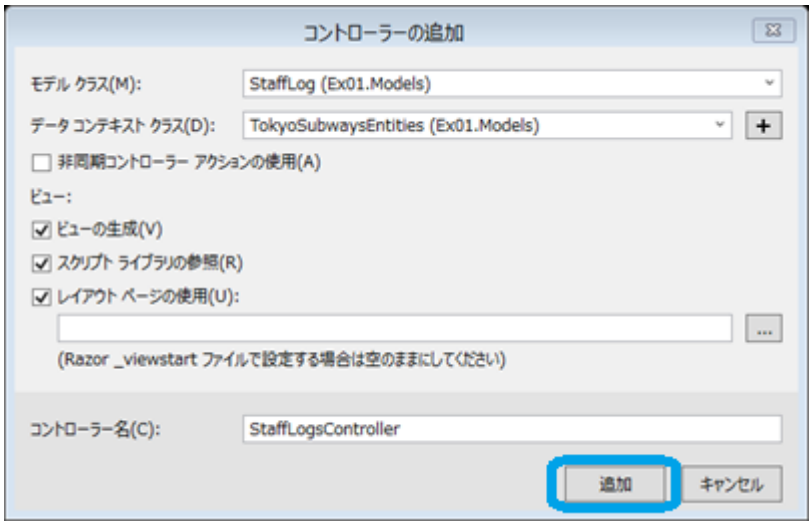
4. Model が作成できましたので、Controller と View を作成します。

[ソリューションエクスプローラー]-[Controllers]を右クリックし、表示されるコンテキストメニューより[追加(D)]-[コントローラー(T)]を選択します。



[スキャフォールディングの追加]ダイアログが表示されます。ここでは[MVC]から[Entity Framework を使用した、ビューがある MVC 5 コントローラー]を選択し、[追加] をクリックします。

[追加]後、「コントローラーの追加」ダイアログが表示されますので、以下のように入力します。

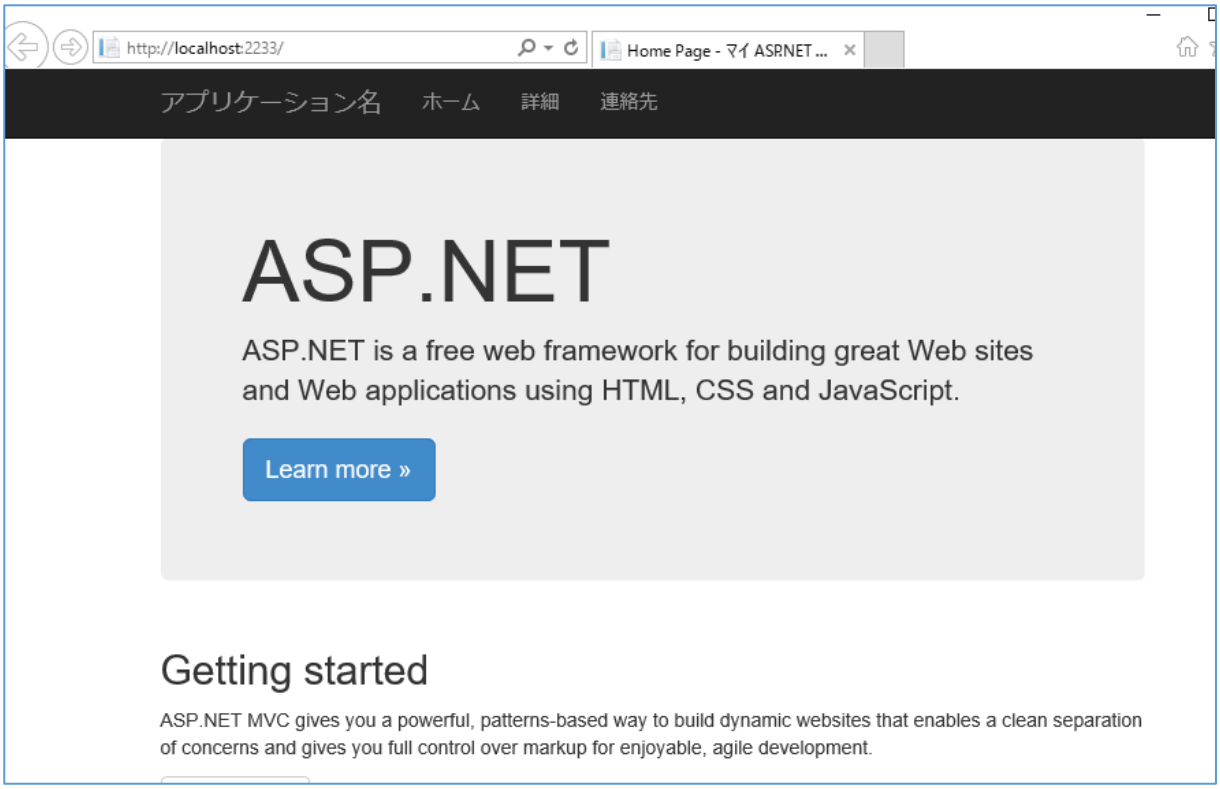


パラメーター	値
モデルクラス	Product ({プロジェクト名}.Models)
データコンテキストクラス	{データベース名}Entities ({プロジェクト名.Models}
非同期コントローラーアクションの使用	チェックしない（デフォルト値）
ビューの生成	チェック（デフォルト値）
スクリプトライブラリの参照	チェック（デフォルト値）
レイアウトページの使用	チェック（デフォルト値）
（レイアウトページの参照）	空白（デフォルト値）
コントローラー名	ProductsController (デフォルト値)

[追加]ボタンクリック後、ダイアログが閉じ、コードが表示されます。この状態で一旦ビルドします。

これにより、Controller と View が作成されました。

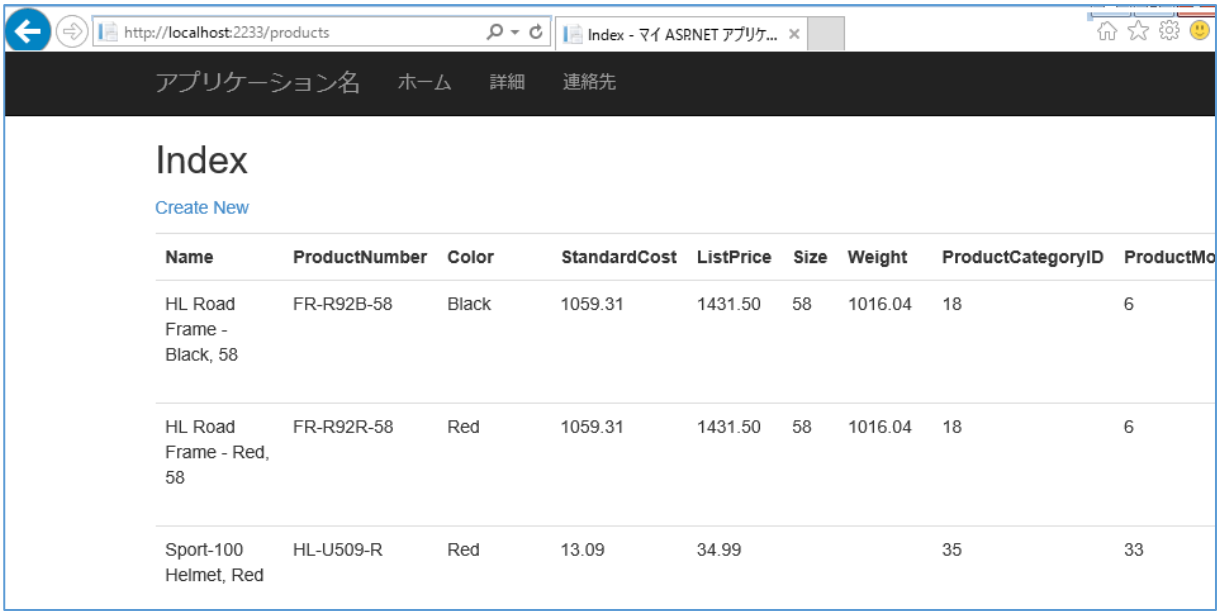
5. プロジェクトが作成された状態で、キーボードから[F5]キーを押してデバッグ実行を行います。下図のようにブラウザ上にアプリケーションが表示されます。



アドレスバーに表示されているサーバー名の後ろに products/ と入力します。

例： 表示されている内容が http://localhost:2233/ ならば、 http://localhost:2233/products/ となります。

その結果、以下のように表示されることを確認してください。



※表示される内容は異なる場合があります。

6. （任意項目）画面上部のメニューに、上記 /products へのリンクを追加します。

Visual Studio のソリューションエクスプローラーより、/Views/Shared/_Layout.cshtml を開きます。

_Layout.cshtml 内、23 行目付近に以下のようなソースコードが記述されています。

```
<ul class="nav navbar-nav">
    <li>@Html.ActionLink("ホーム", "Index", "Home")</li>
    <li>@Html.ActionLink("詳細", "About", "Home")</li>
    <li>@Html.ActionLink("問い合わせ", "Contact", "Home")</li>
</ul>
```

ここでは「詳細」のリンクを変更して"Products"へのリンクとし、表示文言を「製品」と変更します。上記ソースコードの 4 行目に当たる部分を以下のように変更します。（ここでは上記で示したソースコード全体を記載しています。）

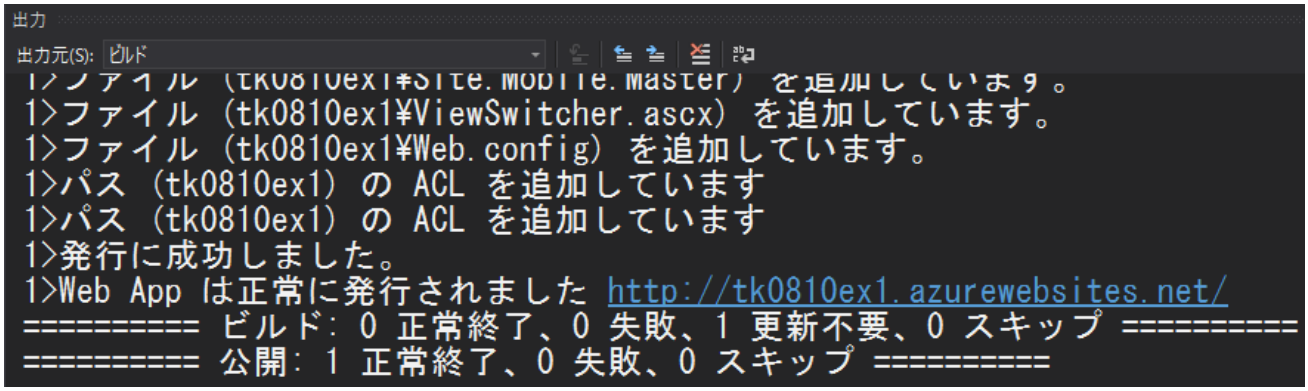
```
<ul class="nav navbar-nav">
    <li>@Html.ActionLink("ホーム", "Index", "Home")</li>
    <li>@Html.ActionLink("製品", "products", "Home")</li>
    <li>@Html.ActionLink("問い合わせ", "Contact", "Home")</li>
</ul>
```

変更行

これにより、トップページに下図のようにリンクが表示されます。

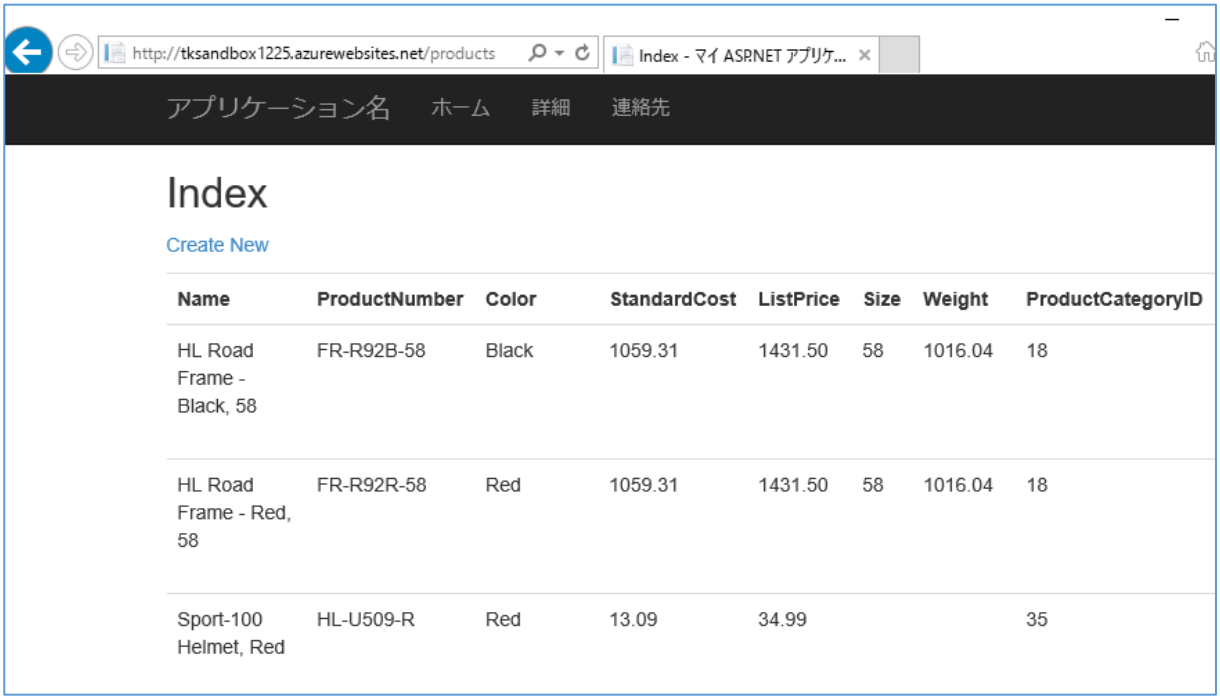
Azure 上にアプリケーションを展開します。先ほどの演習で正しく展開できていれば、Visual Studio が展開に必要な情報を保持していますので、先ほどの演習の手順の途中から再開できます。内容を確認後、[発行]ボタンをクリックしてください。

7. [発行(P)]ボタンをクリックすることにより、WebApp への発行が行われます。出力ウィンドウに状況が表示されます。

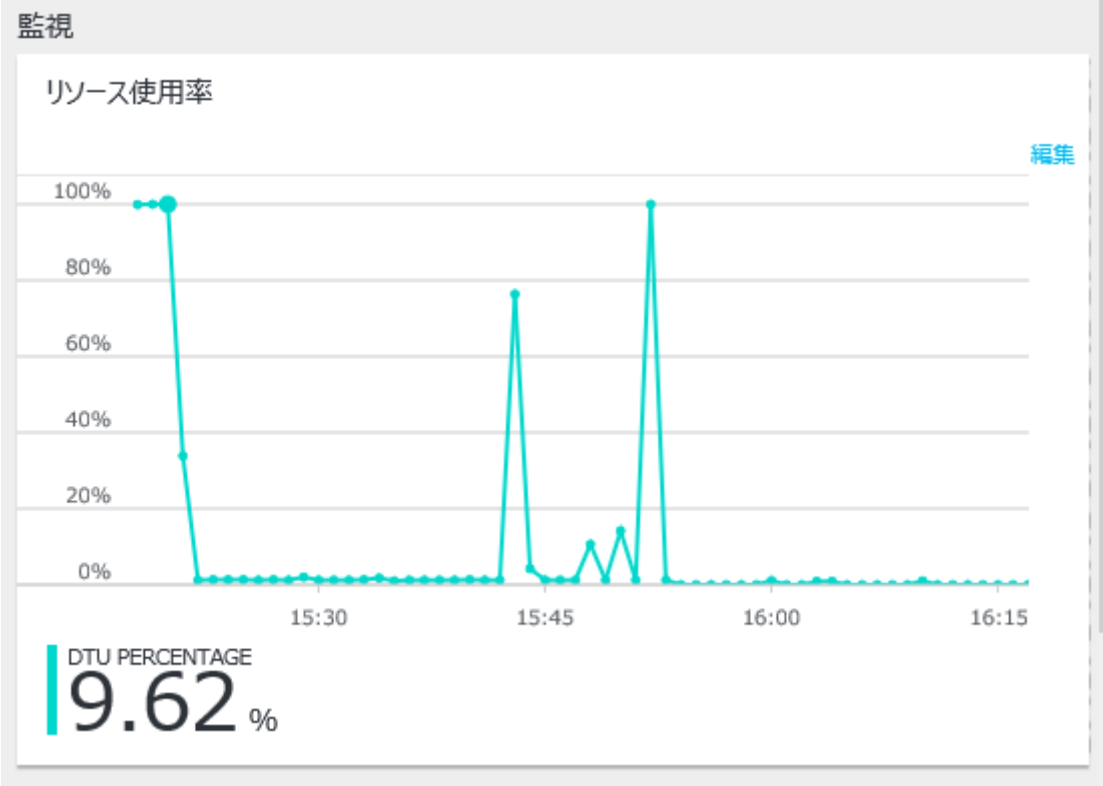


発行完了後、ブラウザが起動し、発行された WebApp が表示されることを確認してください。

8. 表示されたブラウザのアドレスバーに、先ほどのデバッグ時と同様に products/ を追加します。結果として、デバッグ時と同様の内容が表示されていることを確認してください。



9. Azure 管理ポータルにアクセスし、SQL Database の[概要]で表示される[監視]の[リソース使用率]を確認してください。アクセスがあり、グラフが変化していることがわかります。



※実際に表示される内容は異なります

10. Traffic Manager の設定

展開した Web アプリケーションの可用性を確保するために、Traffic Manager の設定を行います。

- 可用性と SLA

Azure で提供されている各サービスでは、SLA (サービスレベルアグリーメント)が提供されています。これは各サービスが接続可能な状態が何パーセント以上であることを保証するものです。ここまで見てきました App Service (Web App) 、SQL Database、Storage の各サービスの SLA は以下のようになっています。

サービス	稼働率	備考
App Service	99.95%	Free, Shared を除く
SQL Database	99.99%	
Storage	99.9%	読み取りアクセスジオ冗長の場合、99.99%

この稼働率を年間・月間のダウンタイムに換算すると、下図のようになります。

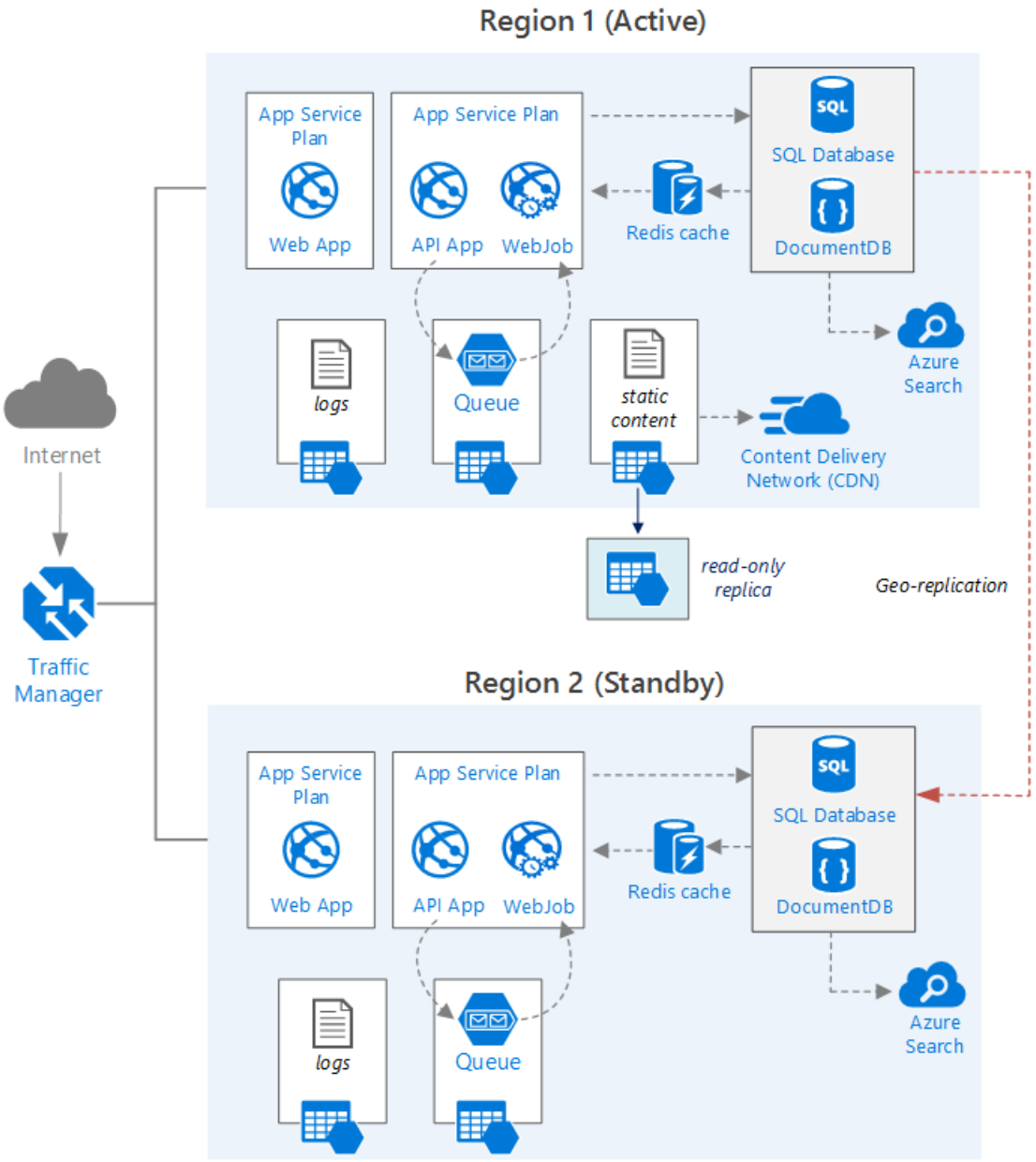
可用性	年間ダウンタイム	月間ダウンタイム
99%	3.65 日	7.2 時間
99.9%	8.76 時間	43.2 分
99.95%	4.38 時間	21.6 分
99.99%	52.56 分	4.32 分
99.999%	5.26 分	25.9 秒

注意すべきなのは、SLA が満たされなかった場合についてです。SLA が満たされなかった場合には、サービスクレジットによる返金が行われます。企業内アプリケーションにおいて、障害によるサービス停止が許容されない場合はどのようにすべきでしょうか。

PaaS アプリケーションの場合には、単一リージョンで障害が発生しても影響を受けないよう、マルチリージョン構成を検討します。App Service (Web App)、SQL Database、Storage におけるマルチリージョン構成のための構成は以下のようになります。

App Service (Web App)	Traffic Manager による切替
SQL Database	Geo レプリケーション構成
Storage	ジオ冗長・読み取りアクセスジオ冗長構成 CDN を併用

これらにより、例えば東日本のデータセンターで障害が発生しても、最小のダウンタイムで西日本のデータセンターでサービスを継続することが可能となります。全体構成としては、例えば下図のようになります。

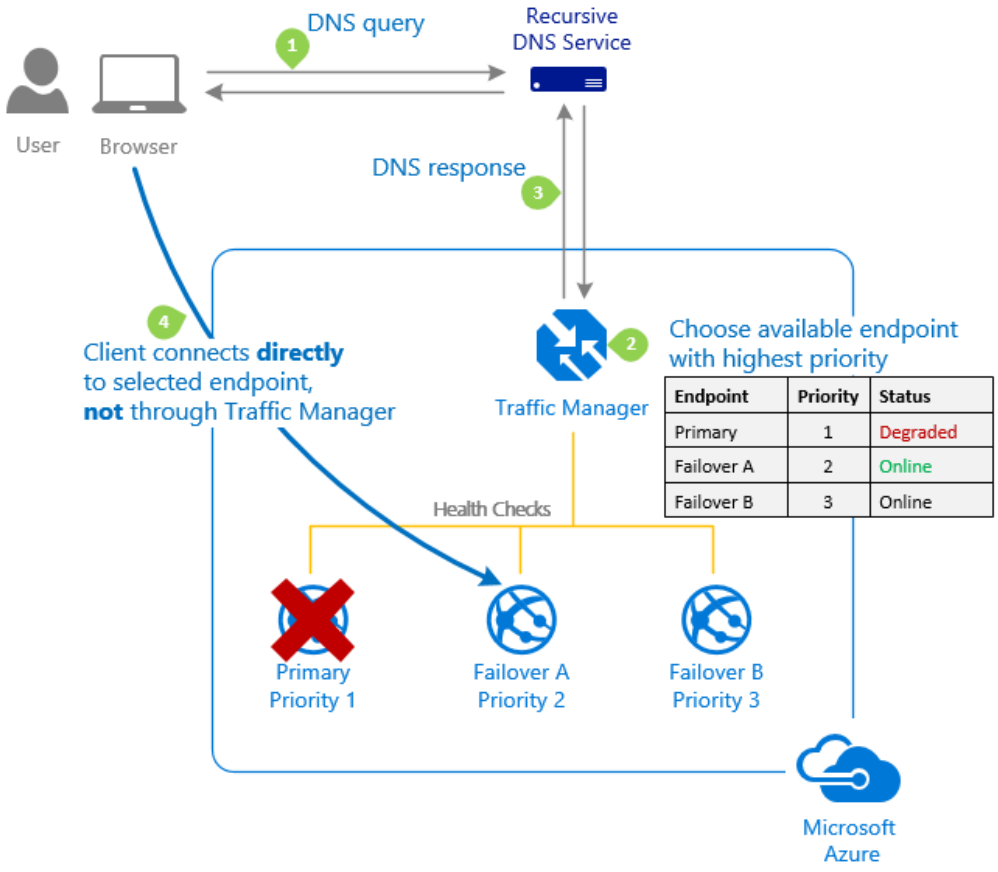


[Azure – Architecture – Reference Architecture – Run a web application in multiple regions]
https://docs.microsoft.com/ja-jp/azure/architecture/reference-architectures/managed-web-app/multi-region-web-app

ここでは、このうち Traffic Manager について見ていきます。

- Traffic Manager とは

Traffic Manager は着信トラフィックをルーティングするサービスです。その際、優先順位、重み付け、パフォーマンス、地理条件による設定が可能です。
合わせて、正常性監視と自動フェールオーバーを提供します。



[Azure – Traffic Manager – Traffic Manager のルーティング方法]
https://docs.microsoft.com/ja-jp/azure/traffic-manager/traffic-manager-routing-methods

上図で Traffic Manager が DNS に作用している点に注意してください。クライアントからの DNS クエリに対して、Traffic Manager がルーティング先をレスポンスします。その後、クライアントはレスポンスされたアドレスに対して直接アクセスを行います。

この Traffic Manager が提供する可動性監視と自動フェールオーバーにより、例えば東日本の Web アプリケーションが停止した場合でも、自動的に西日本の Web アプリケーションにルーティングされ、サービスは継続されます。

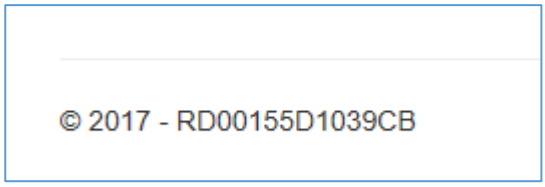
Traffic Manager を使用するためには、Web App が二つ以上のリージョンで Standard 以上のサービスレベルで稼働していることが必要となります。

（任意項目）：振り分けられた先が確認出来るように、Web アプリケーション上に稼働しているサーバー名・IP アドレスなどを表示する、あるいは色を変更するなどすると、なお良いでしょう。
例えばサーバー名を表示する場合、以下のようにします。

Visual Studio で /Views/Shared/_Layout.cshtml を開きます。
表示された _Layout.cshtml の 35 行目付近に footer タグがあります。この footer タグ内を以下のように変更します。

```
<footer>
    <p>&copy; @DateTime.Now.Year - @Environment.MachineName</p>
</footer>
```

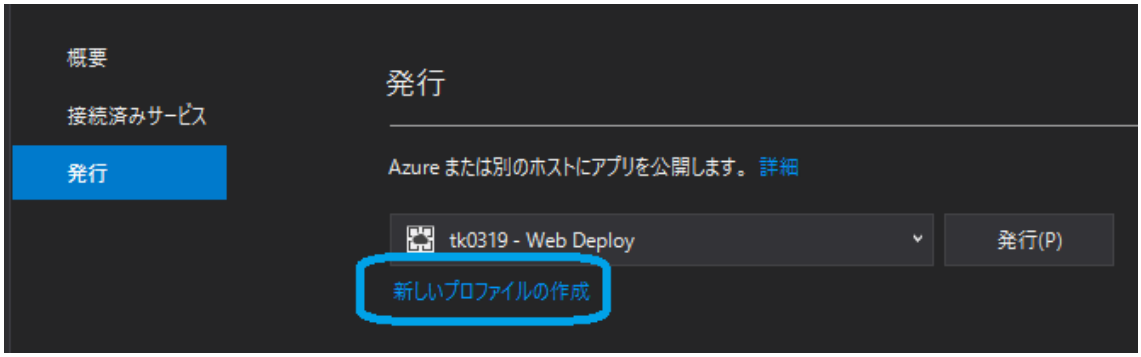
これにより、Web ページの最下部に、下図のようにサーバー名が表示されます。



1. 先程展開した Web App の価格レベルを確認します。Standard 以上になっていることを確認してください。

2. 先ほどの演習と同様の手順で、Web App を異なるリージョンに展開します。

Visual Studio から「発行」を行います。その際、既存の発行プロファイルではなく、新規で作成してください。



設定すべき内容はこれまでの内容と同一ですが、以下が異なります。

パラメーター	説明	今回の設定
Web App の名前	重複できないため異なる名称を使用します。	（作成済みの名称）2

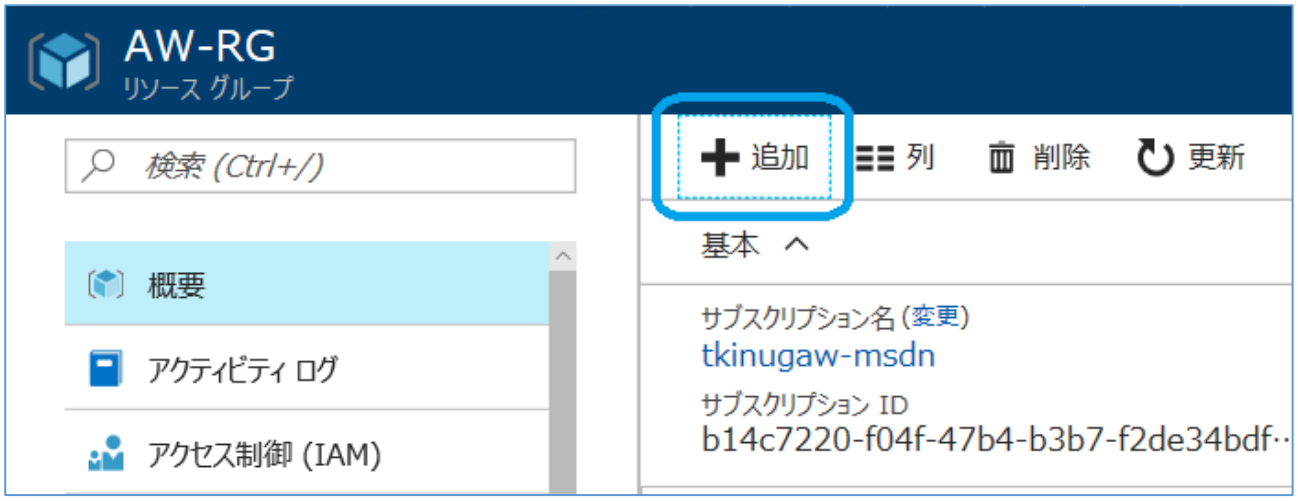
App Service プランも新たに作成します。

パラメーター	説明	今回の設定
App Service プラン	サービスプランにつける任意の名前です。	（任意の名称）2
場所	展開するデータセンターです。 展開済みの Web App と異なる場所を選択してください。	例：Japan East に展開済みならば Japan West
サイズ	使用するサービスレベルです。	S1

作成後、新しい設定で発行してください。

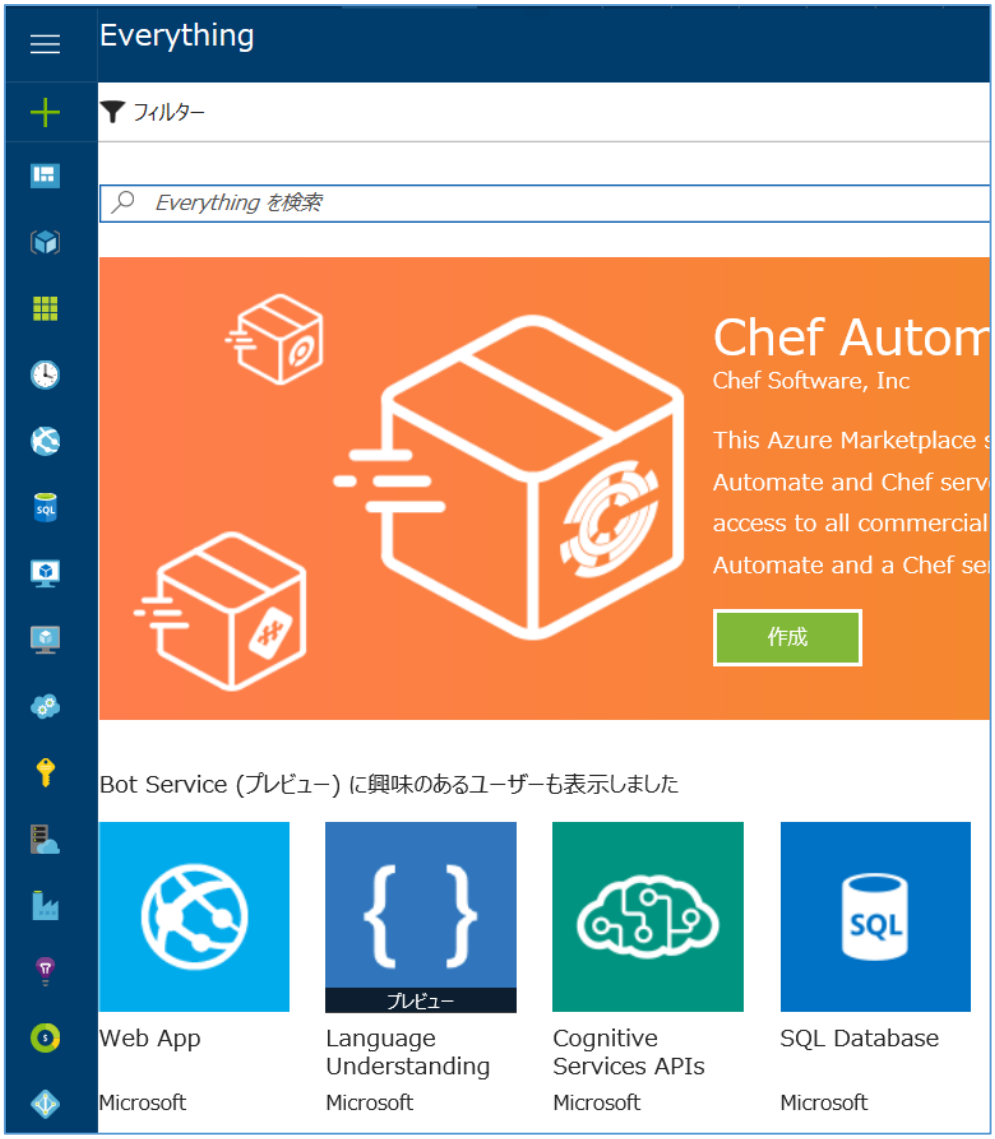
これにより、同じ Web アプリケーションが二つのデータセンター上に存在している状態になります。

3. リソースグループ内に Traffic Manager を追加します。Azure 管理ポータルから、リソースグループ[AW-RG]を開きます。

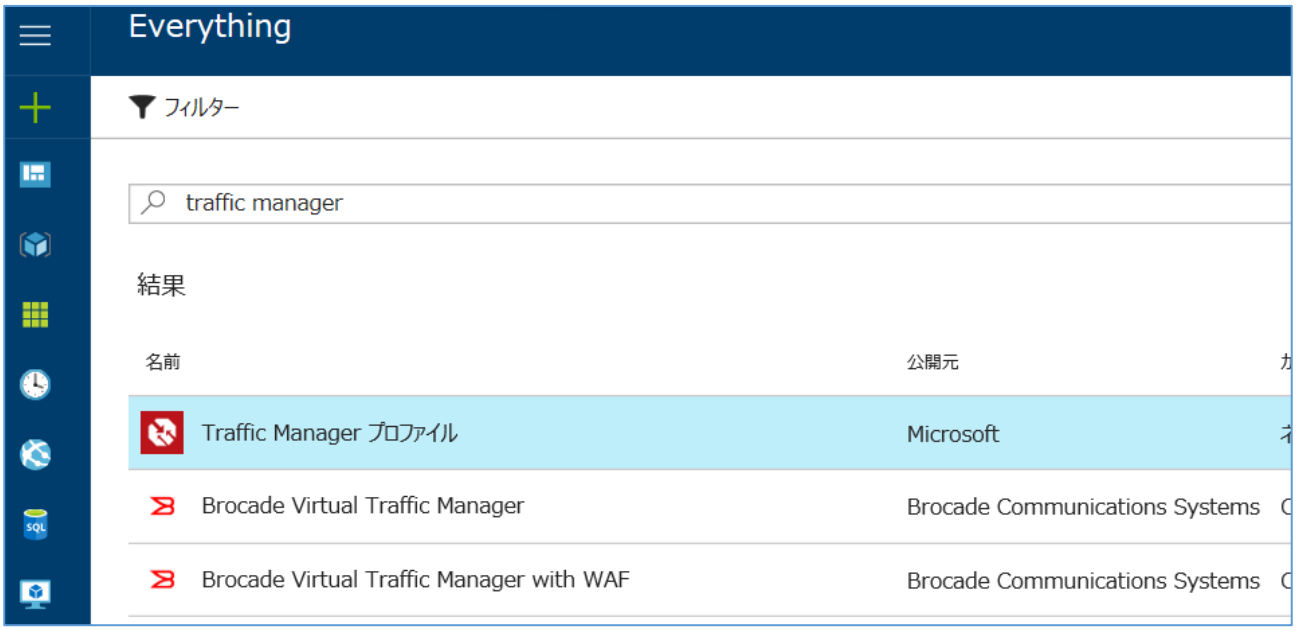


[追加]ボタンをクリックし、リソースを追加します。

4. 追加するリソースメニューが表示されます。



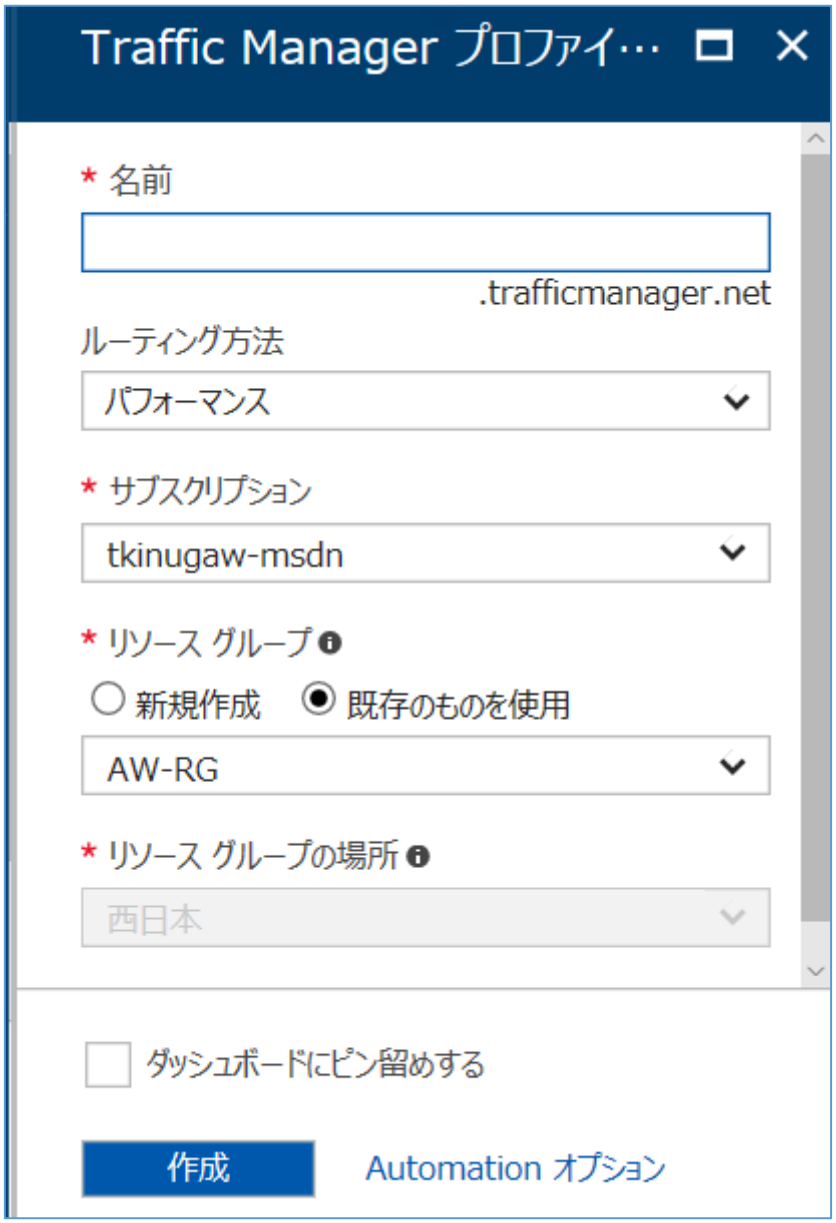
上部の[Everything を検索]の箇所に"traffic manager"と入力して検索を行います。



表示された[Traffic Manager プロファイル]をクリックして下さい。

画面下部の[作成]ボタンをクリックします。

5. Traffic Manger の作成パラメーターを入力します。



入力するパラメーターは以下の通りです。

パラメーター	説明	値
名前	サービスのエンドポイント名です。	（任意のわかりやすい名称）
ルーティング方法	トラフィック分散のルールです。	重み付け
サブスクリプション	使用するサブスクリプション	＜規定値＞
リソースグループ	使用するリソースグループ	[既存のものを使用]を選択し”AW-RG”を選択

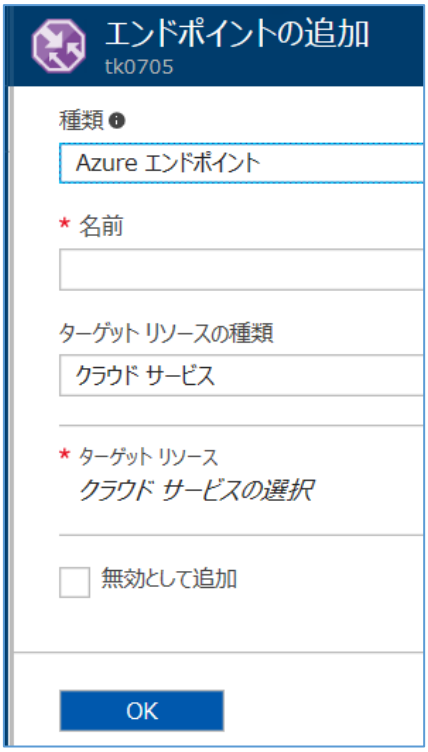
入力後、[作成]ボタンをクリックして下さい。

作成が完了した後、先程の「リソースグループ」のブレードを[更新]してください。作成した Traffic Manager が追加されていることを確認します。

6. 作成した Traffic Manger を開きます。

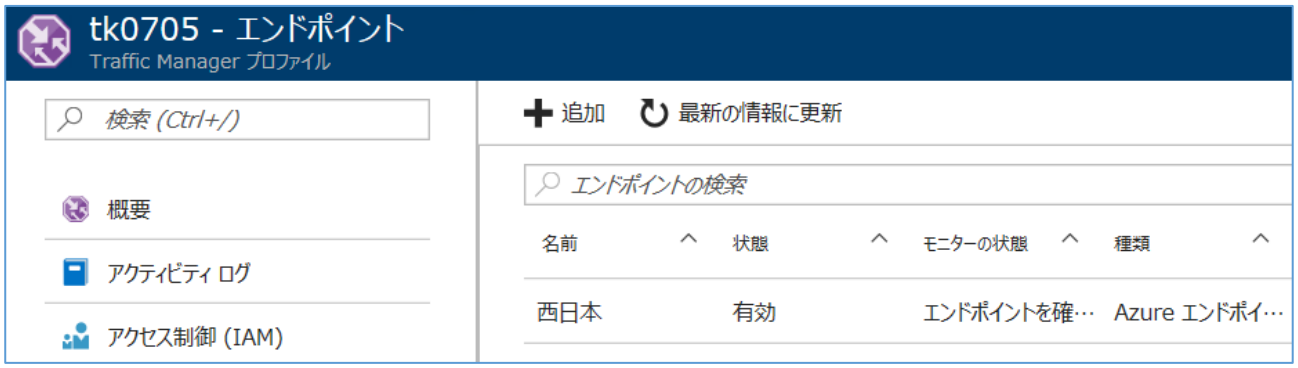
作成した状態ではまだエンドポイントは登録されていません。[エンドポイント]をクリックして、トラフィックによって分散されるエンドポイントを追加します。

[追加]をクリックすると、[エンドポイントの追加]が表示されます。



パラメーター	値
種類	Azure エンドポイント
名前	(任意のわかりやすい名称)
ターゲットリソースの種類	App Service
ターゲットリソース	作成済みの Web App を選択します

入力後[OK]をクリックします。クリックすると、[エンドポイント]に以下のように表示されます。

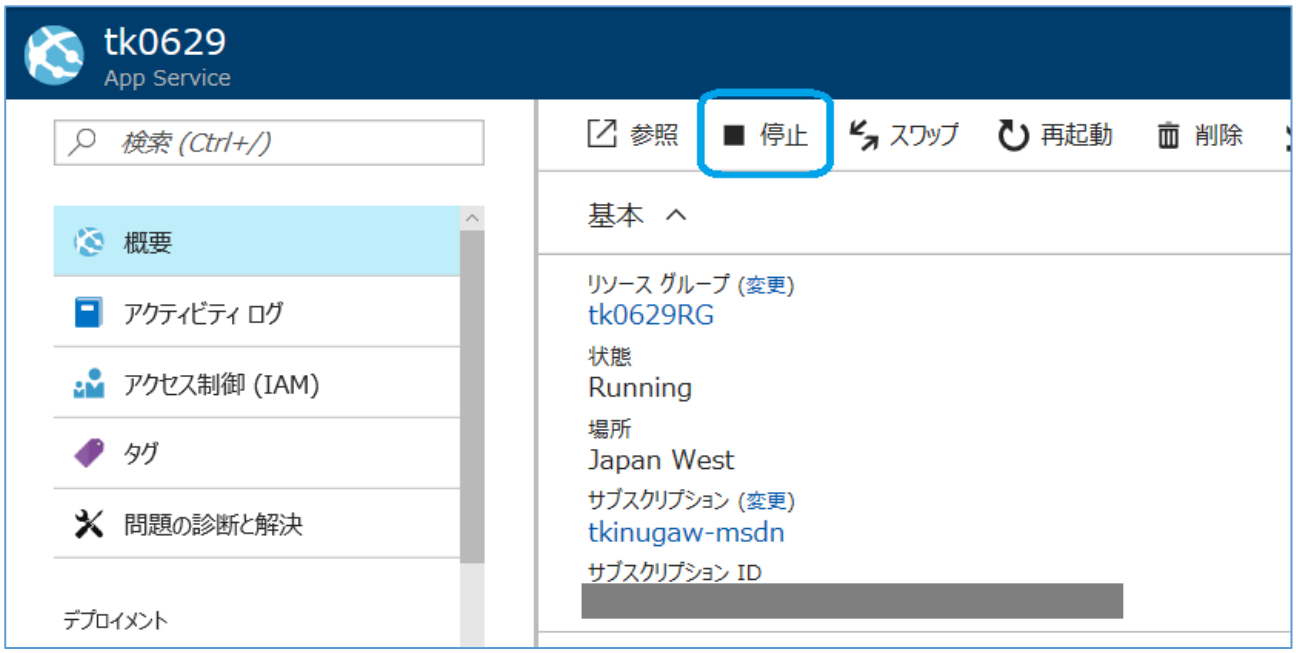


同様の手順で、先程追加した Web App のエンドポイントを追加します。これにより、Traffic Manager から 2 つのエンドポイントを監視している状態になります。

7. Traffic Manger の[概要]から"DNS 名"のハイパーリンクをクリックします。

ブラウザの新しいタブで、Web アプリケーションが表示されることを確認してください。

8. Traffic Manger で振り分けられているエンドポイントの片側の Web App を停止させます。



停止状態になった後も、先程の Traffic Manager の DNS 名でアクセスすると正常に表示されることを確認してください。

確認が終わった後、停止したアプリを再度[開始]してください。

11. Application Insights の設定

可用性を監視するために、Application Insights の設定を行います。

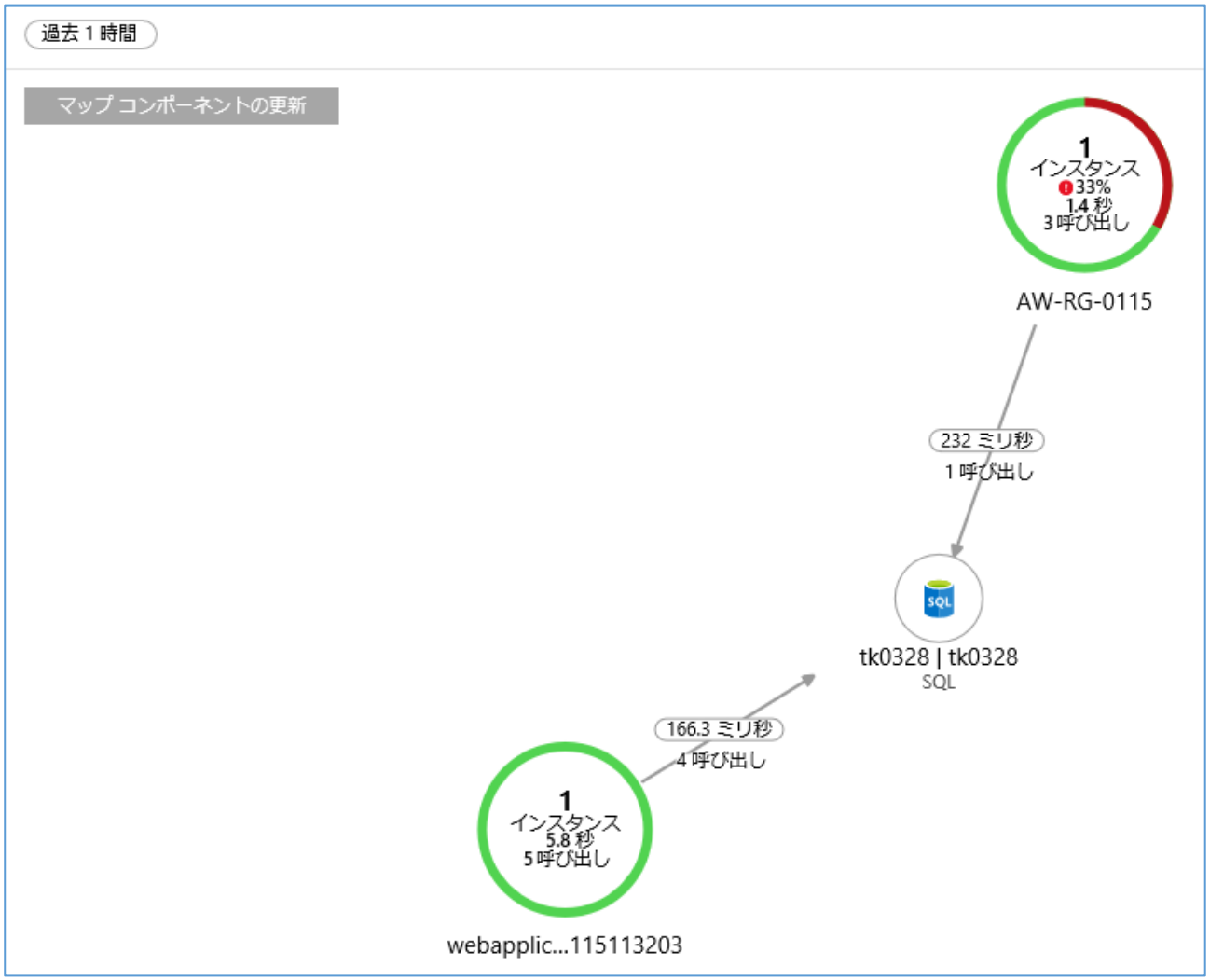
- Application Insights とは

Application Insights は Web アプリケーションに対するテレメトリーサービスを提供します。
稼働状況、エラーの発生状況などを把握することが可能で、最安値無料から利用できます。
注意）Application Insights のお支払いには MSDN Subscription 特典を充てることはできません。

Application Insights は展開済みの Web App に管理ポータルから適用することも可能ですが、より詳細なテレメトリーには、Visual Studio から Application Insights Telemetry の追加を行います。これによりクライアントサイド(ブラウザ)の処理状況、あるいはカスタムテレメトリの取得も可能となります。

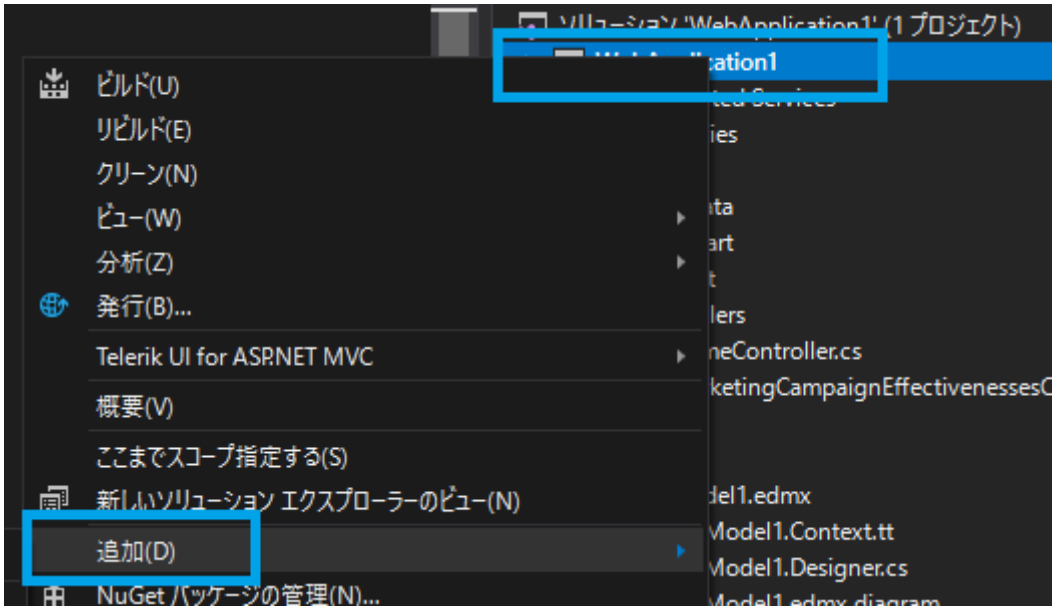
合わせて依存関係を追跡してのテレメトリーも提供されます。具体的には以下の通りです。

- Web アプリケーションからストレージへの呼び出し
- Web アプリケーションから SQL Database への呼び出し
- クライアントからの Ajax 呼び出し

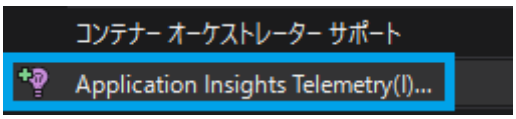


1. これまでの演習で作成したソリューションを Visual Studio で開きます。

2. ソリューションエクスプローラーのプロジェクトを右クリックし、[Application Insight の構成]を選択します。

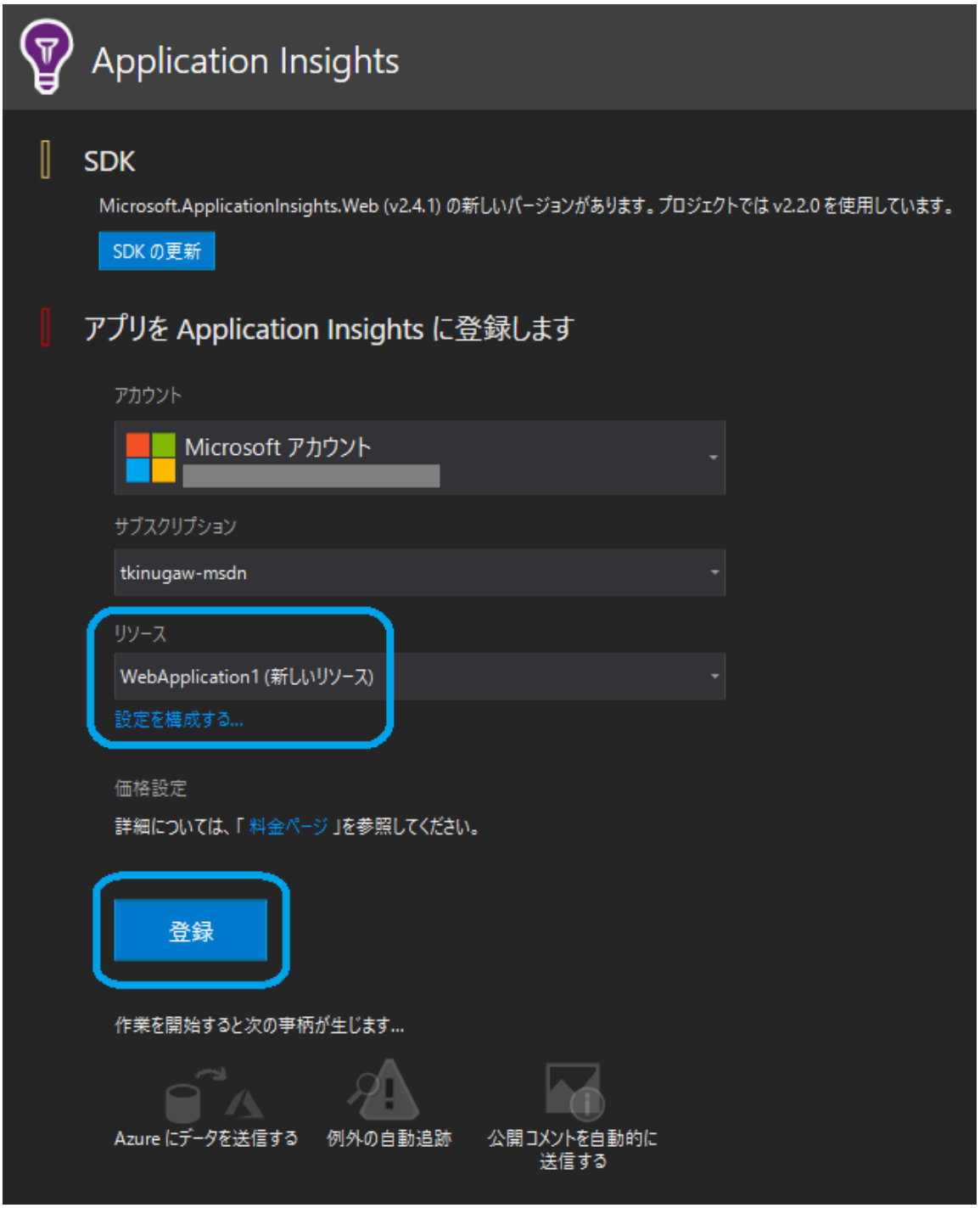


「追加」で表示された内容より「Application Insights Telemetry(I)」をクリックしてください。



3. [Application Insights の構成]が表示されます。[作業の開始]をクリックして下さい。

4. [Application Insights の構成]が表示されます。



リソースは先ほどの Exercise で作成した Application Insights リソースを選択します。

パラメーター	値
リソース	先ほどの Exercise で作成した Application Insights リソース

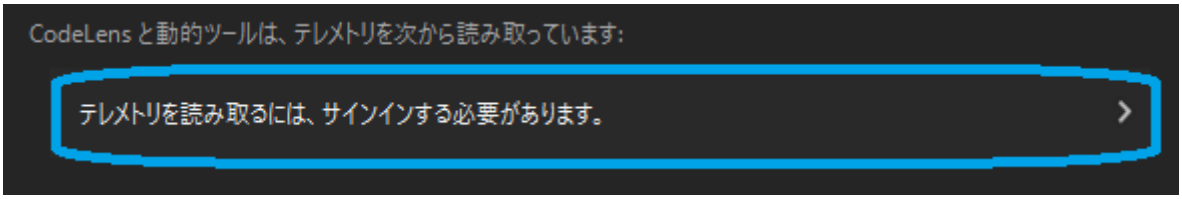
上記を設定後、[登録]をクリックして下さい。

5. [登録ボタン]をクリックすると、必要なライブラリなどが追加されます。[完了]ボタンが表示されるまでお待ち下さい。[完了]が表示されたら、クリックして閉じて下さい。

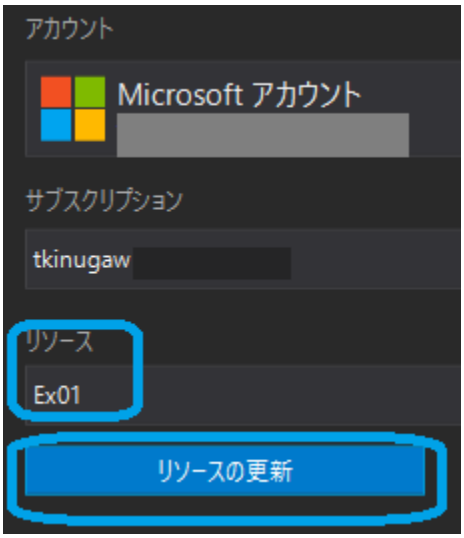

Application Insights
アプリ マップ
コンポーネントの依存性についての説明
スマート検出
異常を検出して通知します
VS 検索
VS でテレメトリを検索して分析します
分析
ほんの数秒でテラバイト単位のデータのクエリを実行します
CodeLens
VS でコードにメソッドのパフォーマンスをインラインで表示する
Live Metrics
リアルタイムで Web アプリの効果を表示します
詳細を表示...
完了

6. [Application Insights の構成]に戻ります。[トレースの収集を有効にします]が表示されている場合、[System.Diagnostics からのトレースの収集]をクリックし、NuGet パッケージを追加します。

7. 【Visual Studio Community では利用できないので、以下の設定は不要です。】
   同様に、Code Lends の設定を追加します。



   展開するとサインイン情報が表示されます。内容を確認して下さい。



   このとき、[リソース]が正しいものであるか確認してください。

   確認後、[リソースの更新]をクリックして下さい。

8. 例外発生時の状況を Application Insights で確認します。Controller 内の HomeController.cs をダブルクリックして開きます。

9. 表示された Controller の先頭に、以下の using を追加します。

```
using Microsoft.ApplicationInsights;
```

10. HomeController クラスのメンバーフィールドとして、TelemetryClient を追加します。以下のコードを記述して下さい。

```
private readonly TelemetryClient telemetry = new TelemetryClient();
```

11. About メソッド内で故意に例外を発生させます。Abount メソッドを以下のように変更して下さい。

```
public ActionResult About()
{
    try
    {
        throw new NotImplementedException("Exceptionの実験");

        ViewBag.Message = "Your application description page.";

        return View();

    }
    catch (Exception ex)
    {
        telemetry.TrackException(ex);
        throw;
    }

}
```

※故意に例外を発生させ、Application Insight に例外情報を送信しています。

12. Index メソッド内に telemetry.TrackEvent を追加します。以下のコードを Index メソッド内に追加します。

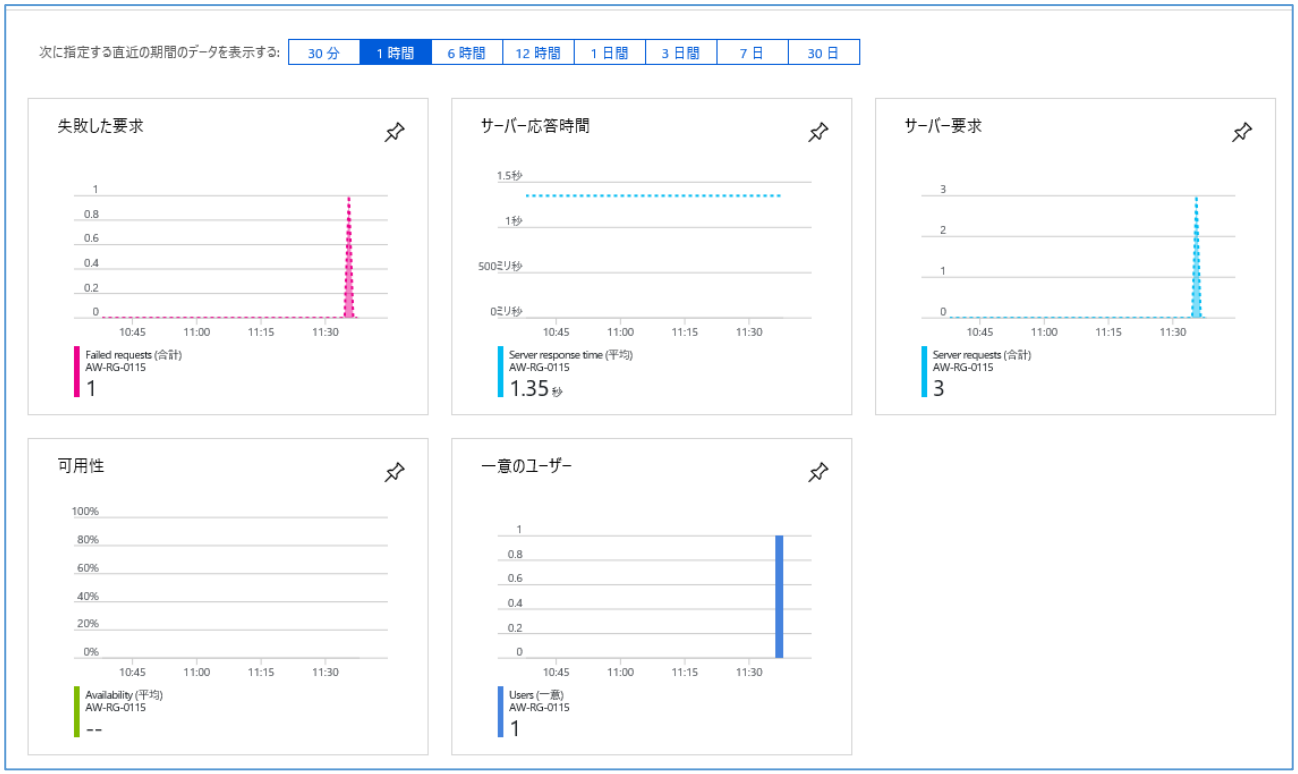
```
telemetry.TrackEvent("Index Requested");
```

13. 先ほどの Exercise と同様に、Azure 上にアプリケーションを展開します。先ほどの Exercise で正しく展開できていれば、Visual Studio が展開に必要な情報を保持していますので、先ほどの Exercise の手順の途中から再開できます。

14. この間に、展開済みの Web アプリケーションの各ページにアクセスします。その際、以下のページに複数回アクセスするようにして下さい。

    ・SQL Database の Exercise で作成した /products
    ・Storage の Exercise で table へのアクセスを追加したトップページ
    ・例外を発生させるようにした /Home/About (上部の「詳細」からアクセス可能)

15. データが収集されると、以下のように表示されます。



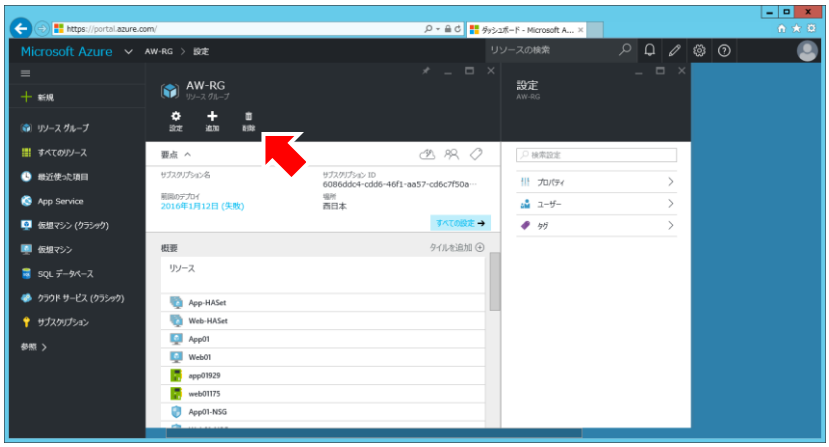
「調査」にある各機能を使用して、それぞれの統計情報を確認してください。

12. リソースグループの削除

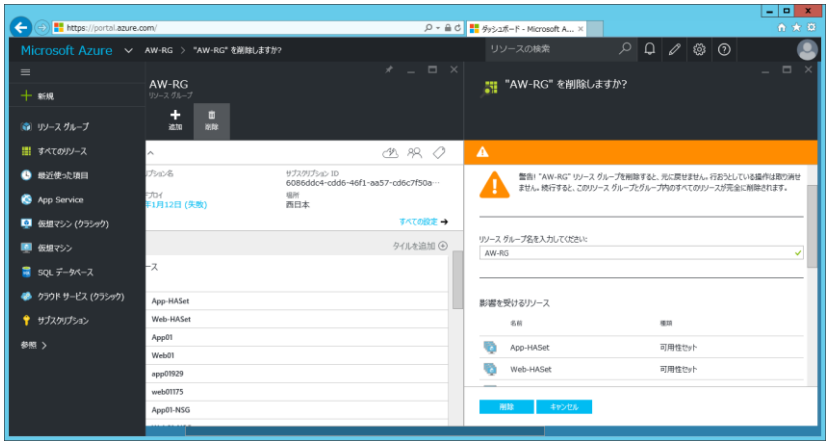
最後の手順では、リソースグループを削除します。単純に Web App を削除すると、Web アプリケーションだけが削除され、関連するストレージアカウント、Application Insights などのリソースが残存しますが、リソースグループを削除すると、そのリソースグループ内のすべてのリソースが削除されます。

次の手順ではリソースグループ「AW-RG」を削除し、その中にあるリソースをすべて削除します。なお、この手順を実行すると、実習で構築したシステムが削除されるため、まだ、検証を続けたい場合は、検証の完了後におこなってください。

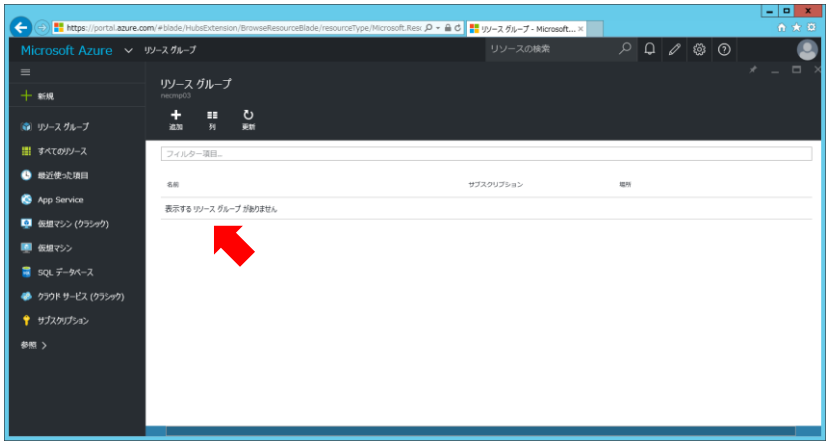
1. Azure 管理ポータルの［スタート画面］より［AW-RG］のタイルをクリックします。
2. ［AW-RG］と［設定］が表示されます。コマンドの［削除］をクリックします。



3. ［"AW-RG"を削除しますか？］が表示されます。［リソースグループ名を入力してください］に「AW-RG」と入力し、［削除］をクリックします。



4. リソースグループとその中のすべてのリソースが削除されます。画面右上の［通知］をクリックし、リソースグループが削除するまで待機します。リソースグループが削除されるまで約 10 分、時間が掛かります。
5. Azure 管理ポータルの画面左のジャンプバーより［リソースグループ］をクリックします。
6. ［リソースグループ］が表示されます。一覧にリソースグループ「AW-RG」がないことを確認します。



13. Microsoft Azure に関する情報の入手元

Microsoft Azure に関する最新の情報は、次の Web サイトから入手できます。

●Azure の公式ページ（各国共通）

製品情報、価格、技術情報など、Azure に関するすべての情報への入口です。Azure をお使いのお客様は、右上の「ポータル」をクリックすると Azure のポータルにログインすることができます。ポータルでは、Azure 上で構築したアプリケーションの管理、課金状況の確認などができます。

http://azure.microsoft.com/ja-jp/

●日本のお客様向けのサイト

上記サイトの内容に加え、日本のお客様用に作成されたコンテンツが満載です。

http://aka.ms/jp/azure

●サポートエンジニアによるブログ サイト

よくお問い合わせをいただく技術・課金・サポートに関する内容をまとめたものです。サポートに問い合わせる前に、まずはここをご参照ください。

http://blogs.msdn.com/b/dsazurejp/
http://blogs.msdn.com/b/jpsql/ （SQL データベース / SQL Server）

●MSDN フォーラムの Azure フォーラム

Azure に関する技術的な質問に対して、これまでの投稿から情報を入手するのみならず、自分の質問を投稿し、他のユーザーや MVP (Most Valuable Professional) からの回答を得られることが期待できます。開発者（DEVELOPER）サポート / 標準（STANDARD）サポート契約ではカバーしていない

“How To” や “仕様” に関する質問も、このフォーラムをご活用ください。

https://social.msdn.microsoft.com/Forums/ja-JP/home?category=azure

14. Microsoft Azure のお問合せ

Microsoft Azure では、お問い合わせ内容に応じて窓口を用意しております。お問い合わせの内容に応じて各窓口をご利用ください。なお、Premier のお客様は、Premier 窓口をご利用いただけます。

お問合せ内容	料金	窓口名	連絡方法	お問い合わせ方法
製品や機能の概要、価格、ライセンスなどの情報の収集や購入前相談	無償	Cloud Direct	電話	「Cloud Direct」で検索してください。
課金、サブスクリプションに関するお問い合わせ、請求書払いへの変更、クォータ増加の依頼	無償	Microsoft Azure ポータル	Azure ポータル※1	※3
技術的なお問い合わせ	有償	Microsoft Azure ポータル※2		

※1：電話窓口はありません。

※2：有償のサポートプランが必要です。

※3：お問い合わせ方法につきましては、次の Blog をご参照ください。

サポートにお問い合わせする方法について http://blogs.msdn.com/b/dsazurejp/archive/2013/10/31/10462044.aspx

なお、ポータルサイトにアクセスできない場合は、次のどちらかの方法でお問い合わせください。

窓口名 / 連絡方法	お問い合わせ方法
カスタマー インフォメーションセンター 電話番号：0120-41-6755	営業時間：平日 9:00 – 17:30 窓口担当者に、「Azure について問い合わせがしたい」 とお伝えください。Azure 担当者より折り返しご連絡いたします。