

AZ-204: Create Azure App Service web apps

Azure App Service Intro

Типи планів (оплата)

- Shared - використовуєш CPU на машинах, які shared з іншими клієнтами
- Consumption - тільки для functions з динамічним скейлінгом
- Dedicated - Azure виділяє VMs, на яких ти раниш свої апки
- Isolated - це розширення для dedicated плану, але також виділяється окрема ізольована мережа

В рамках одного плану Azure деплоїть всі апки на всі VMs. Якщо потрібно коригувати навантаження одного сервісу, тоді краще винести його в окремий план.

App Service підтримує автоматичний та мануальний деплоймент.

Automated deployment

- Azure DevOps
- GitHub
- Bitbucket

Manual deployment

- Git - пушиш зміни в Git, воно автоматично підтягує в App Service
- CLI - 'az webapp up *'
- Zip deploy - відправити zip з кодом по HTTP (наприклад, curl)
- FTP/S

Security/Auth

App Service з коробки підтримує інтеграцію з різними Identity Providers. Це вбудовано в фреймворк і не потребує додаткового коду. Це працює як federated identity (Auth0 по такій же схемі реалізований).

Identity providers

- Microsoft - /.auth/login/aad - <https://learn.microsoft.com/en-us/azure/app-service/configure-authentication-provider-aad>
- Facebook - /.auth/login/facebook - <https://learn.microsoft.com/en-us/azure/app-service/configure-authentication-provider-facebook>
- Google - /.auth/login/google - <https://learn.microsoft.com/en-us/azure/app-service/configure-authentication-provider-google>
- Twitter - /.auth/login/twitter - <https://learn.microsoft.com/en-us/azure/app-service/configure-authentication-provider-twitter>
- Any OpenID Connect provider - /.auth/login/<providerName> - <https://learn.microsoft.com/en-us/azure/app-service/configure-authentication-provider-openid-connect>

Це працює як HTTP Proxu в окремому контейнері. Кожний запит на сервіс спочатку проходить через модуль авторизації, а потім вже відправляється на апку.

Authentication flow

- Sign user in - redirect to `/.auth/login/<provider>`
- Post-authentication - redirect to `/.auth/login/<provider>/callback` with token
- Establish authenticated session - App Service додає куки в відповідь
- Serve authenticated content - браузер автоматично додає куки в кожний запит (X-ZUMO-AUTH заголовок)

Authorization behavior

- Allow unauthenticated requests - дозволяє анонімні запити, але так же може оброблювати авторизованих юзерів
- Require authentication - редірект на авторизацію або HTTP 401/403 статус код

App Service roles

- Front ends - обробляє HTTP запити
- Workers - відповідають за навантаження

App Service Multi-tenant Networking

App Service включає в себе багато модулів, яким потрібно підключення до мережі конкретного плану. Тому by design немає можливості конфігурувати мережу на лоу левелі. Система надає певний API, щоб налаштовувати комунікацію між сервісами.

Inbound communication features:

- App–assigned address
- Access restrictions
- Service endpoints
- Private endpoints

Outbound communication features:

- Hybrid Connections
- Gateway-required virtual network integration
- Virtual network integration

Examples

- Support IP-based SSL -> App-assigned address
- Support unshared dedicated inbound address -> App-assigned address
- Restrict access to your app from a set of well-defined addresses -> Access restrictions

Default networking behavior

Shared план хостить апки в multi-tenant workers за замовчуванням. Більш дорогі плани мають власних workers. Якщо потрібно скейлити систему, то ми створюємо в рамках плану ще один worker, а він реплікує всі сервіси. Налаштування мережі працюють в рамках одного worker.

Outbound addresses

Різні плани використовують різні типи VMs. Тому вихідні IP адреси можуть змінюватись. Це потрібно враховувати при міграції з одного плану на інший.


Ці IP адреси можна знайти в налаштуваннях кожного сервісу на Azure Portal, або викликати наступну команду:

```
az webapp show \
  --resource-group <group_name> \
  --name <app_name> \
  --query outboundIpAddresses \
  --output tsv
```

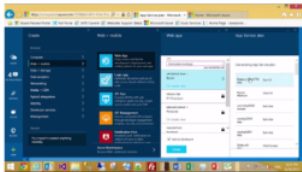
Також можна знайти можливі IP адреси для всіх типів планів:

```
az webapp show \
  --resource-group <group_name> \
  --name <app_name> \
  --query possibleOutboundIpAddresses \
  --output tsv
```

Practice



Azure App Service - Sample Static HTML Site



Azure App Service Web Apps

App Service Web Apps is a fully managed compute platform that is optimized for hosting websites and web applications. This platform-as-a-service (PaaS) offering of Microsoft Azure lets you focus on your business logic while Azure takes care of the infrastructure to run and scale your apps.

```

telamarfolen [ ~ ]$ mkdir htmlapp
telamarfolen [ ~ ]$ cd htmlapp
telamarfolen [ ~/htmlapp ]$ git clone https://github.com/Azure-Samples/html-docs-hello-world.git
Cloning into 'html-docs-hello-world'...
remote: Enumerating objects: 50, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 50 (delta 0), reused 2 (delta 0), pack-reused 46
Receiving objects: 100% (50/50), 343.04 KiB | 14.29 MiB/s, done.
Resolving deltas: 100% (8/8), done.
telamarfolen [ ~/htmlapp ]$ resourceGroup=$(az group list --query "[].{id:name}" -o tsv)
telamarfolen [ ~/htmlapp ]$ appName=az204app$RANDOM
telamarfolen [ ~/htmlapp ]$ cd html-docs-hello-world
telamarfolen [ ~/htmlapp/html-docs-hello-world ]$ az webapp up -g $resourceGroup -n $appName --html
The webapp 'az204app9345' doesn't exist
Creating AppServicePlan 'telamarfolen_asp_0530' ...
Creating webapp 'az204app9345' ...
Configuring default logging for the app, if not already enabled
Creating zip with contents of dir /home/telamarfolen/htmlapp/html-docs-hello-world ...
Getting scm site credentials for zip deployment
Starting zip deployment. This operation can take a while to complete ...
Deployment endpoint responded with status code 202
You can launch the app at http://az204app9345.azurewebsites.net
Setting 'az webapp up' default arguments for current directory. Manage defaults with 'az configure --scope local'
--resource-group/-g default: learn-def02263-99d6-4fdf-b36b-d0d953c321ca
--sku default: F1
--plan/-p default: telamarfolen_asp_0530
--location/-l default: centralus
--name/-n default: az204app9345
{
  "URL": "http://az204app9345.azurewebsites.net",
  "appserviceplan": "telamarfolen_asp_0530",
  "location": "centralus",
  "name": "az204app9345",
  "os": "Windows",
  "resourcegroup": "learn-def02263-99d6-4fdf-b36b-d0d953c321ca",
  "runtime_version": "-",
  "runtime_version_detected": "-",
  "sku": "FREE",
  "src_path": "//home//telamarfolen//htmlapp//html-docs-hello-world"
}

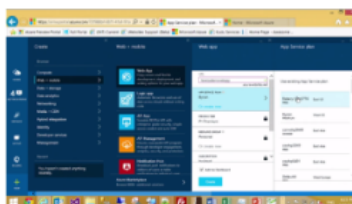
```

```

telamarfolen [ ~/htmlapp/html-docs-hello-world ]$ code index.html
telamarfolen [ ~/htmlapp/html-docs-hello-world ]$ az webapp up -g $resourceGroup -n $appName --html
Webapp 'az204app9345' already exists. The command will deploy contents to the existing app.
Creating AppServicePlan 'telamarfolen_asp_0530' ...
Creating zip with contents of dir /home/telamarfolen/htmlapp/html-docs-hello-world ...
Getting scm site credentials for zip deployment
Starting zip deployment. This operation can take a while to complete ...
Deployment endpoint responded with status code 202
You can launch the app at http://az204app9345.azurewebsites.net
Setting 'az webapp up' default arguments for current directory. Manage defaults with 'az configure --scope local'
--resource-group/-g default: learn-def02263-99d6-4fdf-b36b-d0d953c321ca
--sku default: F1
--plan/-p default: telamarfolen_asp_0530
--location/-l default: centralus
--name/-n default: az204app9345
{
  "URL": "http://az204app9345.azurewebsites.net",
  "appserviceplan": "telamarfolen_asp_0530",
  "location": "centralus",
  "name": "az204app9345",
  "os": "Windows",
  "resourcegroup": "learn-def02263-99d6-4fdf-b36b-d0d953c321ca",
  "runtime_version": "-",
  "runtime_version_detected": "-",
  "sku": "FREE",
  "src_path": "//home//telamarfolen//htmlapp//html-docs-hello-world"
}

```

Refactored by Telamar



Azure App Service Web Apps

App Service Web Apps is a fully managed compute platform that is optimized for hosting websites and web applications. This platform-as-a-service (PaaS) offering of Microsoft Azure lets you focus on your business logic while Azure takes care of the infrastructure to run and scale your apps.

Configure web app settings

Application settings

App Service може налаштовувати environment variables, які потім вичитає апка. Це можна зробити на UI.

my-core-app - Configuration

App Service

Search (Ctrl+/)

Security

Deployment

Quickstart

Deployment slots

Deployment Center

Settings

Configuration

Application settings (Classic)

Authentication / Authorization

Application Insights

Identity

Backups

Custom domains

SSL settings

Networking

Scale up (App Service plan)

Scale out (App Service plan)

Save Discard

Click here to upgrade to a higher SKU and enable additional features.

Application settings

General settings

Default documents

Path mappings

Application settings

Application settings are encrypted at rest and transmitted over an encrypted channel. You can choose to display them in plain text in your browser by using the controls below. Application Settings are exposed as environment variables for access by your application at runtime. [Learn more](#)

+ New application setting

Show values

Advanced edit

Filter

Name	Value	deployment...
(no application settings to display)		

Connection strings

Connection strings are encrypted at rest and transmitted over an encrypted channel.

+ New connection string

Show values

Advanced edit

Filter

Name	Value	Type	deployment...
(no connection strings to display)			

Можна додавати налаштування по одному або балком (Advanced edit button).

Add/Edit application setting

Name	Debug
Value	false
<input type="checkbox"/>	deployment slot setting

UpdateCancel

```
JSON
[
  {
    "name": "<key-1>",
    "value": "<value-1>",
    "slotSetting": false
  },
  {
    "name": "<key-2>",
    "value": "<value-2>",
    "slotSetting": false
  },
  ...
]
```

Connection strings

Деякі Azure DB потребують, щоб в App Service був обов'язково збережений connection до них. Так можна додати новий app settings та вказати тип для нього (наприклад, `SQLServer`).

Connection strings зберігаються зашифрованими.

```
[
  {
    "name": "name-1",
    "value": "conn-string-1",
    "type": "SQLServer",
    "slotSetting": false
  },
  {
    "name": "name-2",
    "value": "conn-string-2",
    "type": "PostgreSQL",
    "slotSetting": false
  },
  ...
]
```

General settings

Для того, щоб використовувати більш дорогі цінові плани, потрібно цілий ряд налаштувань.

- Stack settings (дані про фреймворк, який використовує апка)

Stack settings

Stack

Python


Major version

Python 3

Minor version

Python 3.8

Startup Command

 Provide an optional startup command that will be run as part of container startup. [Learn more](#)

- Platform settings
 - Bitness
 - WebSocket protocol
 - Always On (якщо апка 20 хвилин не використовується, її можна вимкнути)
 - Managed pipeline version (для IIS)
 - HTTP version
 - ARR affinity (якщо декілька інстансів апки, то можна налаштувати роутінг юзера до того самого інстансу типу сесія)
- Debugging (можна віддалено дебажити апку)
- Incoming client certificates (потрібно для mutual authentication)

Path mappings

Для контейнерів можна налаштувати своє сховище (аля docker volume).

Кнопка New Azure Storage Mount.

Параметри:

- Name - display name
- Configuration options - Basic or Advanced
- Storage accounts - приєднати контейнер до конкретного storage account
- Storage type - Blobs або Files
- Storage Container - використовувати конкретний контейнер
- Share name - the file share name
- Access key
- Mount path - абсолютний шлях по якому контейнер буде зчитувати файли

Diagnostic logging

App Service має вбудовану діагностику. Різні типи логування:

Application logging	Windows/Linux	App Service file system or Azure Storage blobs	Зберігає логи в файлі, які генерує фреймворк та клієнтський код
Web server logging	Windows	App Service file system or Azure Storage blobs	Логує в файл кожен HTTP запит
Detailed error logging	Windows	App Service file system	Ми можемо відправляти юзеру html сторінки з помилками 400+. Але на проді такого не повинно бути, тому app service може зберігати ці сторінки у себе
Failed request tracing	Windows	App Service file system	Зберігає trace для HTTP запиту, включаючи всі дані про IIS (handlers)
Deployment logging	Windows/Linux	App Service file system	Автоматичне логування деплоя. Працює без налаштування

Логи можна переглядати (realtime streaming) в Azure Portal, Azure CLI, Local Console.
az webapp log tail --name appname --resource-group myResourceGroup

Щоб викачати логи з App Service file system потрібно в браузері перейти за посиланням:

- Windows -> <https://<app-name>.scm.azurewebsites.net/api/dump>
- Linux -> <https://<app-name>.scm.azurewebsites.net/api/logs/docker/zip>

Якщо логи в Storage Container, то підійде будь-який клієнт (наприклад, Azure Storage Explorer).

Security certificates

Сертифікати зберігаються в webspace. Це місце прив'язане до resource group та region, тож всі апки задеплойені в цій комбінації мають доступ до цих сертифікатів.

App Service certificate options:

- Create a free App Service managed certificate - безкоштовний приватний сертифікат
- Purchase an App Service certificate - приватний сертифікат, який менеджить Azure
- Import a certificate from Key Vault
- Upload a private certificate - завантажити сертифікат з third party провайдера
- Upload a public certificate - потрібно для випадків, коли апка потребує доступ до віддалених ресурсів

Private certificate requirements

- PFX file з паролем зашифрований в triple DES

- Приватний ключ не менше 2046 бітів
- Містить проміжні сертифікати в цепочці

Для TLS додаткові вимоги:

- Містить Extended Key Usage for server authentication (OID = 1.3.6.1.5.5.7.3.1)
- Підписаний довіреним авторитетом

Free managed certificate

Доступний лише для Basic, Standard, Premium та Isolated планів. Azure повністю сам менеджить сертифікат для SSL. Оновляє його кожні 6 місяців.

Але є обмеження:

- Не підтримує wildcard сертифікати
- Не підтримує використання сертифікату як certificate thumbprint
- Не можна викачати
- Не підтримується в App Service Environment (ASE)
- Не підтримується в доменах інтегрованих з traffic manager
- Якщо сертифікат для CNAME-mapped domain, CNAME повинен бути прив'язаний напряму до <app-name>.azurewebsites.net

Import App Service Certificate

Якщо купити сертифікат у Azure, то платформа робить наступні шаги:

- Заменеджить процесс покупки в GoDaddy
- Провалідує домен
- Заменеджить сертифікат в Azure Key Vault
- Заменеджить поновлення
- Автоматично синхронізує сертифікат з копіями в App Service

Upload a private certificate

Якщо сертифікат згенерований через OpenSSL, то потрібно згенерувати PFX файл з приватним ключем. Також потрібно замінити заглушки в файлі реальними даними.

Це все можна зробити командою:

```
openssl pkcs12 -export -out myserver.pfx -inkey <private-key-file> -in <merged-certificate-file>
```

Enforce HTTPs

Весь HTTP трафік краще перенаправити на HTTPs. Це можна зробити в налаштуваннях.

TLS/SSL settings

×

App Service

Search (Ctrl+ /)

«

Settings

Configuration

Authentication / Authorization

Application Insights

Identity

Backups

Custom domains

TLS/SSL settings

Networking

Scale up (App Service plan)

Scale out (App Service plan)

WebJobs

Push

MySQL In App

Properties

Locks

Refresh

Delete bindings

Buy Certificate

Troubleshoot

FAQs

Bindings

Private Key Certificates (.pfx)

Public Key Certificates (.cer)

⚙️

Protocol Settings

Protocol settings are global and apply to all bindings defined by your app.

HTTPS Only: ⓘ

Off

On

Minimum TLS Version ⓘ

1.0

1.1

1.2

🔒

TLS/SSL bindings

Bindings let you specify which certificate to use when responding to requests to a specific hostname over HTTPS. TLS/SSL Binding requires valid private certificate (.pfx) issued for the specific hostname.

[Learn more](#)

+ Add TLS/SSL Binding

☐

Host name

Private Certificate Thumbprint

TLS/SSL Type

No TLS/SSL bindings configured for the app.

App Features

Це звичайний механізм, який дозволяє увімкнути або вимкнути якусь фічу. Цей механізм підтримує більш складні фільтри, щоб менеджити фічі на основі якихось даних (група юзера, тип браузера, географічна локація чи щось інше). Менеджер фічей підтримує appsettings.json файл.

Є спеціальний синтаксис, щоб описувати ці флаги та фільтри.

```

"FeatureManagement": {
  "FeatureA": true, // Feature flag set to on
  "FeatureB": false, // Feature flag set to off
  "FeatureC": {
    "EnabledFor": [
      {
        "Name": "Percentage",
        "Parameters": {
          "Value": 50
        }
      }
    ]
  }
}

```

Autoscaling

Autoscaling потребує багато ресурсів, щоб моніторити навантаження на систему. Це зручно, але ціна більш висока.

Autoscaling - це функція плану, тож механізм працює в рамках його налаштування (типи VMs, обмеження, фільтри).

Autoscaling conditions

- Metric based - кількість HTTP запитів чи черга на запис на диск
- Schedule - налаштувати графік, коли запускати чи зупиняти додаткові машини

Ці два механізми можна комбінувати для більш складних сценаріїв.

Metrics for autoscale rules

Використовується середнє значення по всіх інстансах

- CPU Percentage
- Memory Percentage
- Disk Queue Length
- Http Queue Length
- Data In
- Data Out

Також є метрики, які залежні від інших Azure сервісів. Наприклад, кількість повідомлень в ServiceBus.

How autoscale rule analyzes metrics

- Спочатку система збирає всі метрики за +- одну хвилину (time grain). Time grain для кожної метрики окремий, тому є агреговане значення time aggregation. На основі метрик вивисляються параметри average, min, max, sum, last, count
- Оскільки одна хвилинка - це дуже мало. То система на другому кроці порівнює отримані параметри зі статистичними даними за більш довгий період (duration time). Порівняння параметрів може відбуватись для декількох time grain, щоб краще розуміти статистику
- Порівняння йде на основі тих правил, які ми описуємо в налаштуваннях

Combining autoscale rules

Одна умова може містити декілька правил. Наприклад:

- If the HTTP queue length exceeds 10, scale out by 1
- If the CPU utilization exceeds 70%, scale out by 1
- If the HTTP queue length is zero, scale in by 1
- If the CPU utilization drops below 50%, scale in by 1

Enable autoscaling

Azure Portal -> App Service -> Settings -> Scale out

Configure Run history JSON Notify Diagnostic settings

Autoscale is a built-in feature that helps applications perform their best when demand changes. You can choose to scale your resource manually to a specific instance count, or via a custom Autoscale policy that scales based on metric(s) thresholds, or schedule instance count which scales during designated time windows. Autoscale enables your resource to be performant and cost effective by adding and removing instances based on demand. [Learn more about Azure Autoscale](#) or [view the how-to video](#).

Choose how to scale your resource

Manual scale
Maintain a fixed instance count

Custom autoscale
Scale on any schedule, based on any metrics

Manual scale

Override condition

Instance count

1

Після того, як механізм активований, можна налаштувати умови та правила.
Є ще default condition. Він буде використаний, якщо не налаштувати інші умови.

Default* Auto created default scale condition

Scale mode

☐ Scale based on a metric ☒ Scale to a specific instance count

Instance count*

1

Schedule

This scale condition is executed when none of the other scale condition(s) match

Auto created scale condition 1

Scale mode

☒ Scale based on a metric ☐ Scale to a specific instance count

Rules

No metric rules defined; click [Add a rule](#) to scale out and scale in your instances based on rules. For example: 'Add a rule that increases instance count by 1 when CPU percentage is above 70%'. If you save the setting without any rules defined, no scaling will occur.

[+ Add a rule](#)

Instance limits

Minimum Maximum Default

Schedule

☒ Specify start/end dates ☐ Repeat specific days

Timezone

(UTC-08:00) Pacific Time (US & Canada)

Start date

07/17/2021 12:00:00 AM

End date

07/17/2021 11:59:00 PM

Є кнопка Add a rule. Вона відкриє наступне вікно для налаштування правил.

Scale rule

Metric source

Current resource

Resource type

App Service plans

Resource

ASP-game-89c8

Criteria

Time aggregation

Average

Metric namespace

App Service plans standard metrics

Metric name

CPU Percentage

1 minute time grain

Dimension Name

Instance

Operator

=

Dimension Values

All values

Add

If you select multiple values for a dimension, autoscale will aggregate the metric across the selected values, not evaluate the metric for each values individually.

CpuPercentage (Average)

1.93 %

Enable metric divide by instance count

Operator

Greater than

Metric threshold to trigger scale action

70

%

Duration (in minutes)

10

Add

Scale rule

Resource group

game

Instance count

1

Default

Auto created default scale condition

Scale mode

Scale based on a metric

Scale to a specific instance count

Instance count

1

Schedule

This scale condition is executed when none of the other scale condition(s) match

Auto created scale condition 1

Scale mode

Scale based on a metric

Scale to a specific instance count

Rules

No metric rules defined; click Add a rule to scale out and scale in your instances based on example: Add a rule that increases instance count by 1 when CPU percentage is above 70% and the scaling without any rules defined, no scaling will occur.

Add a rule

Instance limits

Minimum

1

Maximum

2

Default

1

Schedule

Specify start/end dates

Repeat specific days

Timezone

(UTC-08:00) Pacific Time (US & Canada)

Start date

07/17/2021

12:00:00 AM

End date

07/17/2021

11:59:00 PM

Monitor autoscaling activity

Search (Ctrl+/)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Security

Deployment

Quickstart

Deployment slots

Deployment Center

Settings

Application settings

Configuration (Preview)

Authentication / Authorizati...

Application Insights

Identity

Backups

Custom domains

SSL settings

Networking

Scale up (App Service plan)

Scale out (App Service plan)

Save

Discard

Disable autoscale

Refresh

Configure

Run history

JSON

Notify

Observed instance count - this chart plots the instance count as observed by the auto scale engine. If the chart is empty it either means auto scale is in cool down period or auto scale was disabled over a period of time or auto scale was not configured.

Show data for last

1 hour

6 hours

12 hours

1 day

7 days

Custom

Pin to dashboard

Observed Capacity (Avg)

ScaleOnCPU

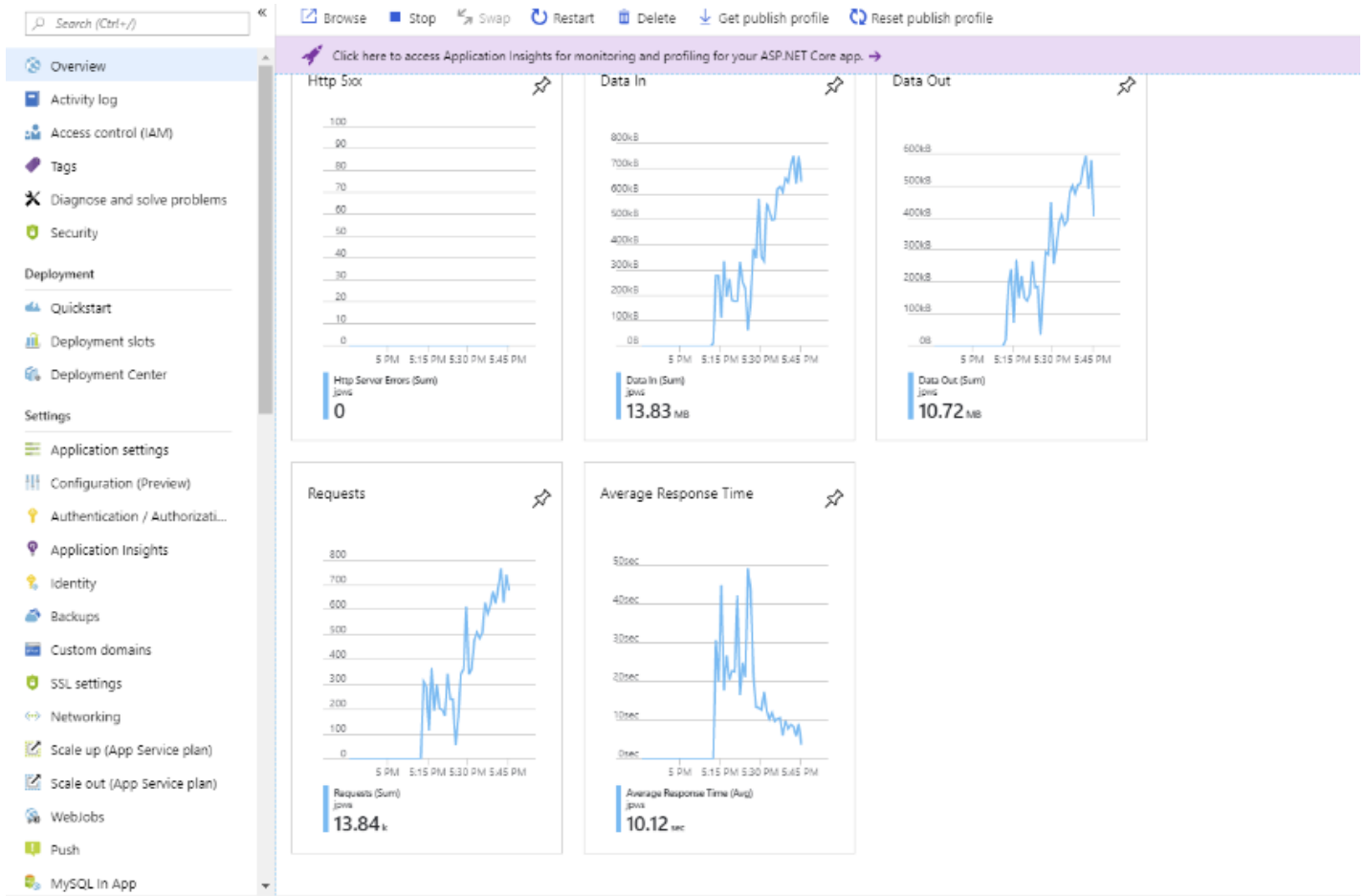
2.22

Autoscale events for this time range

View more details in the Activity Log

OP...	STATUS	EVENT CATEG...	TIME	TIME STAMP	SUB...	EVENT INITIATED BY	RES...	RES...
1	Succeeded	Autoscale	20 min ago	Sun Mar 17 2...	Azu...	Microsoft.Insights/autoscale...	Mic...	serv...
1	Succeeded	Autoscale	20 min ago	Sun Mar 17 2...	Azu...	Microsoft.Insights/autoscale...	Mic...	serv...
1	Succeeded	Autoscale	26 min ago	Sun Mar 17 2...	Azu...	Microsoft.Insights/autoscale...	Mic...	serv...
1	Succeeded	Autoscale	26 min ago	Sun Mar 17 2...	Azu...	Microsoft.Insights/autoscale...	Mic...	serv...
1	Succeeded	Autoscale	32 min ago	Sun Mar 17 2...	Azu...	Microsoft.Insights/autoscale...	Mic...	serv...

Run history можна комбінувати з метриками на Overview сторінці.



Autoscaling best practices

Мінімальна та максимальні кількість інстансів повинна бути різною.

Потрібно правильно вибрати, яку величину контролювати для метрики (min, max, average, total). Зазвичай використовують average.

Краще зробити проміжок між мінімальною та максимальною межею. Це дозволить менше штормити систему. Наприклад, додавати машини лише, коли CPU Usage > 70%, а відключати, коли < 50%. Є зазор між 50 та 70.

Підняття нових машин відбувається, коли будь-яке одне з правил відпрацювало. А зменшення кількості машин відбувається лише, коли всі правила для scale in проходять.

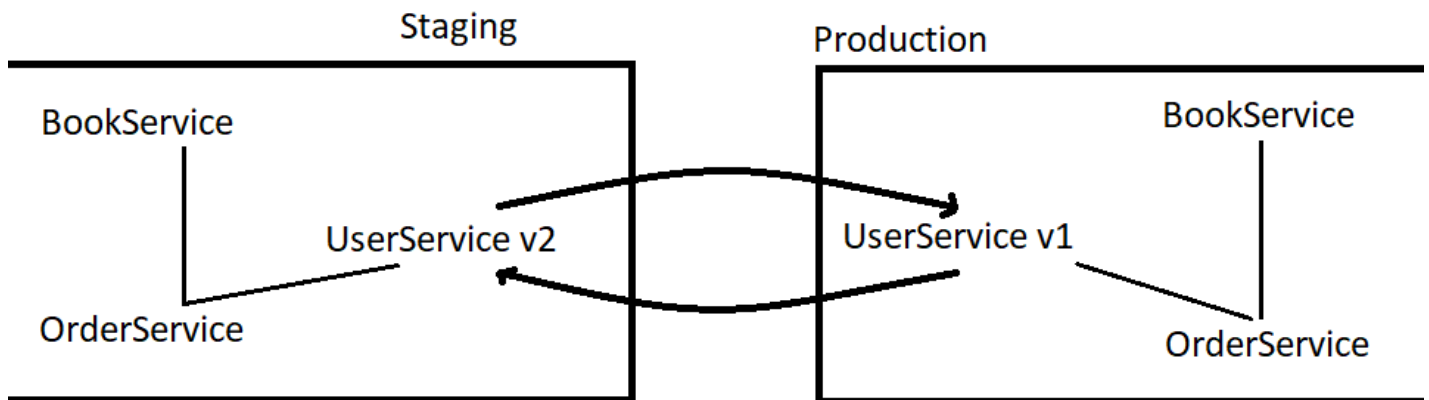
Потрібно правильно обирати initial/safe кількість машин.

Потрібно налаштувати нотифікації, щоб контролювати все, що відбувається.

Deployment slots

Staging environments

Є можливість створити слоти для різних енів. У кожного слота буде свій host name. Також є можливість підмінювати слоти. Наприклад, задеплоїти мікросервіс в слот staging, а потім обміняти його з production слотом.



Slot swapping

App Service виконує такі шаги під час обміну:

- Замінює app settings, connection strings, continuous deployment settings, authentication settings
- Чекає, поки пройде рестарт всіх сервісів в source слоті
- Якщо налаштовано локальний кеш, то App Service робить HTTP запит по рутовій адрес (" /"), щоб ініціалізувати кеш. Це також тригерить рестарт сервісу
- Якщо налаштовано auto swap, то App Service робить health check (HTTP запит) по заданій адресі або руту
- Якщо всі інстанси живі, замінює правила маршрутизації, щоб staging слот став production слотом
- Далі всі ті ж самі дії відбуваються для слотів, які раніше були production

Поки відбувається підготовка staging слоту до підміни, production продовжує працювати.

Деякі налаштування слотів автоматично змінюються під час обміну:

- General settings, such as framework version, 32/64-bit, web sockets
- App Settings
- Connection strings
- Handler mappings
- Public certificates
- WebJobs content
- Hybrid connections *
- Virtual Network Connections *

- Service endpoints *
- Azure Content Delivery Network *

Налаштування з * плануються бути незмінними під час обміну.

Налаштування, які не змінюються під час обміну:

- Publishing endpoints
- Custom domain names
- Non-public certificates and TLS/SSL settings
- Scale settings
- WebJobs schedulers
- IP restrictions
- Always ON
- Diagnostic log settings
- Cross-origin resource sharing (CORS)

Щоб налаштування були swappable, потрібно додати app setting `WEBSITE_OVERRIDE_PRESERVE_DEFAULT_STICKY_SLOT_SETTINGS` до кожного слоту зі значенням 0 або false.

Manually swapping deployment slots

Інструкція:

- 1) Відкрити Deployment slots сторінку та обрати Swap
- 2) Обрати Source та Target слоти
- 3) На табі з налаштуваннями потрібно перевірити, що все правильно
- 4) Все)

Swap with preview (multi-phase swap)

App Service може налаштувати сервіси в слоті, але не обмінювати їх з іншим слотом. В такому випадку можна провалідувати результат та лише потім продовжити процес.

Інструкція:

- 1) Ті ж самі шаги, що й в manual
- 2) Обрати Perform swap with preview
- 3) Обрати Start Swap
- 4) Після налаштування сервіс доступний за шляхом
`https://<app_name>-<source-slot-name>.azurewebsites.net`
- 5) Якщо все окей, то обрати Complete Swap, або Cancel Swap

Auto swap

Авто своп потрібний, коли потрібно деплоїти сервіси з zero cold starts та zero downtime. Цей механізм інтегровано з Azure DevOps. Кожен раз, коли код потрапляє в бранчу source слоту, запускається механізм.

Механізм не працює з web apps on Linux.

Налаштувати:

1. App's resource page -> Deployment slot -> Configuration -> General settings
2. Auto swap enabled - On
3. Обрати source та target слоти
4. Запушити код до брэнчі source слоту

Custom warm-up (health check)

<https://ruslany.net/2017/11/most-common-deployment-slot-swap-failures-and-how-to-fix-them/>

Some apps might require custom warm-up actions before the swap. The `applicationInitialization` configuration element in `web.config` lets you specify custom initialization actions. The swap operation waits for this custom warm-up to finish before swapping with the target slot. Here's a sample `web.config` fragment.

```
<system.webServer>
  <applicationInitialization>
    <add initializationPage="/" hostName="[app hostname]" />
    <add initializationPage="/Home/About" hostName="[app hostname]" />
  </applicationInitialization>
</system.webServer>
```

Копіювати

For more information on customizing the `applicationInitialization` element, see [Most common deployment slot swap failures and how to fix them](#).

You can also customize the warm-up behavior with one or both of the following app settings:

- `WEBSITE_SWAP_WARMUP_PING_PATH`: The path to ping to warm up your site. Add this app setting by specifying a custom path that begins with a slash as the value. An example is `/statuscheck`. The default value is `/`.
- `WEBSITE_SWAP_WARMUP_PING_STATUSES`: Valid HTTP response codes for the warm-up operation. Add this app setting with a comma-separated list of HTTP codes. An example is `200,202`. If the returned status code isn't in the list, the warmup and swap operations are stopped. By default, all response codes are valid.

Rollback and monitor

Інформація про статус процесу доступний в activity log.

Якщо потрібно відкотити зміни, то потрібно повторити механізм ще раз.

Resource Page -> Activity Log -> Swap Web App Slots query

Route traffic in App Service

За замовчуванням всі запити до `http://<app_name>.azurewebsites.net` йдуть на production слот. Також є можливість перенаправити частину трафіку до іншого слоту.

Route production traffic automatically

Є можливість перенаправити автоматично лише частину трафіку до nonproduction слоту.

1. App's resource page -> Deployment slots
2. Traffic % column для конкретного слоту
3. Вказати значення в відсотках
4. Save

App Service також прив'язує юзера до конкретного слоту в рамках однієї сесії. Це чим-то нагадує hashing load balancing.

В хедерах в браузері можна побачити куки x-ms-routing-name з ім'ям слоту.

Route production slot manually

В client-side кодї можна вказати вручну назву слоту використовуючи той же самий параметр x-ms-routing-name.

`<a href="<webappname>.azurewebsites.net/?x-ms-routing-name=self">Go back to production app`