Nº40 junio 2019

CompartiMOSS

REVISTA ESPECIALIZADA EN TECNOLOGÍAS MICROSOFT

Entrevista David Rivera

Azure Digital Twins

10 consejos para mejorar la adopción y gobernanza de Microsoft Teams

Primeros Pasos con Azure Sphere

Contenido

Comparti MOSS

Staff

DIRECCIÓN GENERAL

- Gustavo Velez
- Juan Carlos Gonzalez
- Fabian Imaz
- Alberto Diaz

DISEÑO Y DIAGRAMACIÓN

- Santiago Porras Rodríguez

Contacte con nosotros

revista@compartimoss.com
gustavo@gavd.net
jcgonzalezmartin1978@hotmail.com
fabian@siderys.com.uy
adiazcan@hotmail.com

BLOGS
http://www.gavd.net
http://geeks.ms/blogs/jcgonzalez
http://blog.siderys.com
http://geeks.ms/blogs/adiazmartin

REDES SOCIALES
Facebook:
http://www.facebook.com/group.php?gid=128911147140492
LinkedIn:
http://www.linkedin.com/groups/CompartiMOSS-3776291
Twitter:
@CompartiMOSScom

03

Editorial

Este número de la revista es especial, hemos alcanzado las 40 publicaciones y lamentablemente en el mes de Marzo falleció un gran amigo, un hermano, Ricardo Muñoz, el mae como lo llamábamos nosotros. Esta triste noticia nos ha dejado a los que hacemos la revista, muy tristes, ya que “el mae” era un colaborador activo y autor, es por esto que hemos decidido recordarlo publicando la entrevista que le hicimos. Pero sabemos que fue una persona muy querida dentro de la comunidad de habla hispana de SharePoint, dado lo que significaba Ricardo hemos recopilado palabras de agradecimientos y en este número encontraras una sección con todo este contenido, para que puedas leer lo que significaba para todos nosotros.

Desde la redacción de la revista , te damos las gracia mae por todo lo que hiciste por la comunidad, compartir conocimiento y hacerle llegar nuestro servidor favorito, SharePoint a todos los de habla hispna, GRACIAS.....

El Equipo Editorial de CompartiMOSS

4

Hmenaje a Ricardo Muñoz

El pasado 3 de mayo de 2019 Ricardo Muñoz Monje gran amigo, profesional, persona y referente de las Comunidades Técnicas de Microsoft nos dejó de forma inesperada creando una gran conmoción entre todos los que hemos conocido y hemos colaborado con Ricardo desde hace varios años. Desde el Equipo Editorial de CompartiMOSS hemos querido realizar nuestro particular homenaje a la figura de Ricardo y reconocer no solo su labor como referente de las comunidades técnicas Microsoft, sino también como profesional y amigo. En este artículo publicamos la entrevista que le realizamos en el número 9 de septiembre de 2011, además de las palabras de afecto y recuerdo a Ricardo que nos han ido enviando a la revista.

Mi nombre completo es Ricardo Muñoz Monge y nací hace 30 años en mi bello país Costa Rica, mi infancia transcurrió bastante tranquila sin muchos problemas más que las travesuras típicas de los niños y uno que otro accidente ocasionado por dichas travesuras, en esos momentos mi única relación con la tecnología eran los videojuegos en consola o pc (hobbie que mantengo aun hoy en día). En mi adolescencia inicie mi acercamiento a la tecnología tomando clases de programación en Basic y mantenimiento de computadores en el colegio lo cual me hizo darme cuenta que la informática era lo mío, todo eso unido al hecho que mi otra carrera preferida era historia, (no sé dónde estaría ahora si hubiera tomado otra decisión) inicie mis estudios en el Instituto Tecnológico de Costa Rica. Durante mi práctica de especialidad tuve mi primer acercamiento con SharePoint en su versión 2003 y desde ese momento no he dejado de trabajar con ella por ya más de 9 años. MI trabajo actual es como Gerente de Colaboración y Consultor Sénior en Infraestructura SharePoint para Andes IT y mi compañía LatinShare. Cuando el tiempo entre viajes y trabajo me lo permite actualizo mi blog sobre tecnología SharePoint en www.mundomoss.blogspot.com

¿Por qué y cómo empezaste en tecnología?

Bueno creo que inicie en tecnología porque siempre me han atraído dos temas: La historia y todo lo que tenga que ver con adelantos tecnológicos, yo era uno de esos niños que intentaba armar y desarmar sus juguetes para ver sus componentes con el único problema que debes en cuando sobraban algunas piezas. En el colegio tuve el primer contacto con la programación y la infraestructura, gustándome mucho más la segunda, posteriormente durante mis estudios universitarios ingrese a trabajar en un importante partner de Microsoft en Costa Rica, fui contratado desde un inicio con el objetivo de investigar las funcionalidades de SharePoint y Project Server 2003, herramientas de las cuales quede inmediatamente prendido y hoy en día son mi especialidad. Este gran interés por la tecnología SharePoint me llevo a en conjunto con mi gran amiga Vielka Rojas a formar la comunidad de SharePoint Costa Rica, la cual inicio como una idea entre un par de locos y hoy en día es una de las comunidades de SharePoint mayor importancia a nivel latinoamericano, esto nos llevó a realizar grandes eventos como el Simposio Latinoamericano de SharePoint (www.sharepointcostarica.com) que ya va para su sexta edición, todo esto me ha permitido viajar muchísimo y conocer a grandes amigos pero sobre todo a muchísimas personas apasionadas por la tecnología así como por compartir su conocimiento.

¿Cuáles son tus principales actividades tecnologías hoy en día?

Hoy en día me desempeño como gerente del área de colaboración en la empresa Andes IT, empresa especializada en la implementación de soluciones SharePoint y Project Server. Hace mucho tiempo que deje de programar y ahora me dedico exclusivamente a la infraestructura SharePoint como arquitecto de Soluciones. Adicionalmente desde hace unos meses inicie mi faceta como empresario en Costa Rica y tengo en conjunto con otros expertos del ramo nuestra propia compañía llamada LatinShare dedicada al desarrollo, implementación y outsourcing de soluciones de colaboración sobre plataformas Microsoft.

¿Cuáles son tus principales actividades NO tecnologías hoy en día?

A veces no saco el tiempo que debería para actividades NO tecnologías, pero principalmente me encanta viajar, me gusta conocer nuevos lugares siempre que puedo, mis últimos dos viajes han sido a Isla de Pascua y Machu Pichu dos destinos increíbles Adicionalmente me gusta mucho leer, Jugar videojuegos siempre que puedo pero sobre todo pasar tiempo con las personas que quiero mi familia y amigos.

¿Cuáles son tus hobbies?

Bueno creo que ya es obvio que me gusta mucho la tecnología no solo en el trabajo sino también en mi vida diaria, pero adicional a esto uno de mis hobbies ha sido y será la Historia sobre todo la segunda guerra mundial, mi padre me inculco desde niño la pasión por este tema al punto de que siempre viajo con algún libro sobre el tema para mis ratos de lectura, adicionalmente me encanta conocer nuevas culturas y sus comidas soy de la idea de primero hay que probar algo por más extraño que se vea, Soy un también un gran fan de los comics americanos y europeos (sobre todo Asterix y Obelix) así como del cine.

¿Cuál es tú visión de futuro en la tecnología de acá a los próximos años?

La tecnología cambia a veces demasiado rápido, lo cual es parte de lo interesante de nuestra profesión siempre tenemos que actualizarnos y tratar de estar al tanto de todos los avances que se dan día a día. Hoy en día existen avances sorprendentes que haces unos 10 o 15 años nos hubieran parecido de ciencia ficción como el caso del Kinnect o el ipad así como otro conjunto de tecnologías, todos estos grandes avances y cambios nos permitirán hacer más accesible la información a las masas y a las personas que actualmente no tienen dicho acceso por diversas razones económicas y sociales. Poder predecir que nos traerá el futuro de la tecnología es muy complejo, pero sé que será un futuro muy interesante y lleno de muchos retos y oportunidades para todos los que estamos apasionados con la tecnología.

(ENTREVISTA REALIZADA POR JUAN CARLOS GONZÁLEZ PARA COMPARTIMOSS)

Ricardo, a tu lado di mis primeros pasos en SharePoint. Fuimos compañeros de trabajo, compartimos retos y alcanzamos metas juntos. Pero lo mejor fue tener tu amistad. Descansa en paz querido amigo, te extrañaré.

INGRID ARTAVIA DÍAZ

Aún recuerdo cuando empezamos a hacer los Simposium de SharePoint por LATAM, empezando por Costa Rica, quisimos replicar lo mismo en México, Colombia y en otras ubicaciones.
Cuantas anécdotas, noches preparando material, haciendo amistad... fueron momentos increíbles, no solo hicimos amistad, sino que compartimos nuestra pasión por la tecnología, conocimos familias, comida, lugares, etc...
Y pensar la amistad que logramos en su momento con Ricardo y su familia, incluso vino a México a acompañarnos entregando pláticas, así como también nosotros fuimos para allá varias veces....
Ricardo se te estima amigo y estas en nuestros corazones como parte de la gran familia de la comunidad SharePoint LATAM y WW, abrazo fuerte!

LUIS DU SOLIER

Gracias amigo por darme la oportunidad de conocer tu pequeño y lindo país, por ser un gran amigo y estar dispuesto siempre a ayudar sin esperar nada a cambio. Me habría gustado poder despedirme de ti como dos viejos amigos, cada uno tomando su bebida preferida, pero el destino es caprichoso y nos reservará ese encuentro en el futuro. Como amigo te echaré de menos, como miembro de la comunidad también te echaremos mucho de menos. Descansa en paz amigo allá donde estés y ten seguro que nos volveremos a encontrar.

JUAN CARLOS GONZÁLEZ

Ricardo, quedará tu cálida voz en nuestro corazón. Mi amigo y hermano siempre fuiste un ejemplo de líder ejerciendo de forma directa e indirecta en los demás un deseo profundo de ser y hacer las cosas, así como tú las hacías. Gracias hermano por haber sido un ejemplo de vida profesional y personal.

HAARON GONZÁLEZ

7

Azure Event Grid, el pegamento de nuestras arquitecturas orientadas a eventos

Orquestando y centralizando eventos con Servicios Serverless de Azure

Cada vez va a ser más "normal" e incluso "recurrente", que para implementar gran parte de nuestros sistemas tendamos a diseñar piezas que se basen en servicios Serverless. Esto tiene un primer impacto enorme en nuestras decisiones, ya que contra más procesos corramos en Serverless, y en su mayoría de estos casos en segundo plano, deberemos tener un control casi al milímetro tanto de los "eventos" que disparan estos procesos en background, como de retornar el resultado de estos.

¿Qué pasa si un proceso muy potente, casi inexpugnable en cuanto al rendimiento, se detiene, se para o ni siquiera conseguimos arrancarlo?, ¿Cómo se entera la aplicación que hace de interfaz hacia el usuario?

Si decidimos hacer un portal web, que tenga parte de trabajo en procesos Serverless, necesitaremos hacernos preguntas y poner soluciones como las que vamos a ir desarrollando en los siguientes apartados.

Arquitecturas orientadas a eventos, ¿Qué son?

Una arquitectura basada en eventos debe tener una visión muy orientada a "conectar" sistemas, servicios o productos, en base a unos eventos que se van produciendo en cada uno de estos "Productores de eventos".

Debemos pensar que un Productor de eventos, va a emitir un evento en base a una acción o suceso, al cual los "Consumidores" van a tener acceso mediante una Suscripción a un evento.

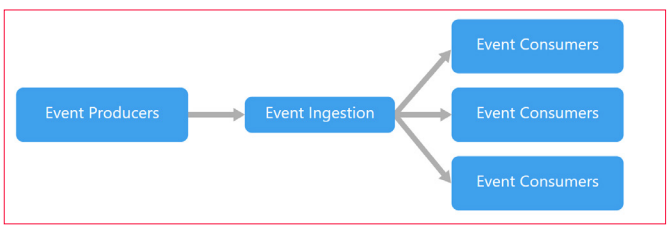


Imagen 1.- Arquitectura basadas en eventos.

Los eventos tienen que ser enviados / recibidos en tiempo real por los Consumidores, y una característica muy especial de estas arquitecturas es que el sistema que produce el Evento, no tiene que controlar que el Consumidor, reciba el evento, o lo que es lo mismo, el proceso que genera un evento, como puede ser el guardado de un dato en una base de datos, o la subida de un documento a un blob, es 100% independiente al hecho de que el evento se propague y que el consumidor reaccione a estas acciones.

Por esto podemos decir que estas arquitecturas implican "procesos" en segundo plano, y que si no se diseñan de forma correcta, con un control de errores muy exhaustivo, puede llegar a provocar perdida de Eventos, y por lo tanto, procesos no ejecutados, que en muchos casos pueden ser capitales para el usuario.

Modelos de la arquitectura, como generar los eventos

Una arquitectura basada en eventos puede seguir dos posibles modelos:

- Modelo Pub/sub: La infraestructura de la mensajería hace un control de las suscripciones. Cuando se publica un evento, se envía el evento a cada suscriptor. Una vez un evento es enviado, no se puedo volver a reproducir, y los nuevos suscriptores a dicho evento no lo reciben.
- Modelo Streaming de Eventos: Los eventos se van "almacenando" en un registro y no se borran. Los eventos sí que llevan un orden, pero los consumidores no se suscriben al flujo de generación de un evento, sino que un cliente accede al registro de eventos, lo lee bajo demanda y es responsable de avanzar su posición en el flujo. La ventaja de este modelo es que cualquier cliente puede unirse en cualquier momento y reproducir eventos pasados.

Modelos de consumo de un evento, ¿Cómo se comporta el consumidor?

Por su lado, el consumidor de un evento tiene varias formas de "procesar" un evento generado:

- Procesamiento sencillo por eventos: Un evento desencadena una acción inmediatamente en el consumidor. Por ejemplo, un Trigger de un Azure Function cuando

se sube un fichero en un Blob Storage.
- Procesamiento de eventos complejos: El consumidor procesa una serie de eventos, y este los filtra en busca de patrones o comportamientos requeridos, por ejemplo, la ingesta de datos y el posterior filtro desde Azure Stream Analytics.

"contra más procesos corramos en Serverless, y en su mayoría de estos casos en segundo plano, deberemos tener un control casi al milímetro"

¿Qué características debe tener nuestro sistema para usar esta arquitectura?

- Hay concurrencia en el consumo de los eventos, es decir varios sistemas o subsitemas necesitan consumir un mismo evento.
- Es un escenario de eventos en "Tiempo real o Retardo Mínimo".
- Procesamiento de eventos complejos
- Gran volumen y alta velocidad de procesamiento

Ventajas de estas arquitecturas

- Nos permiten "desvincular" a los sistemas Productores de eventos de los Consumidores.
- Evitamos las integraciones "Punto a punto", y con ello adaptar nuestros desarrollados para acoger a los nuevos consumidores, lo que facilita añadir nuevos sistemas consumidores en nuestra arquitectura.
- Respuesta en tiempo real para cada consumidor a un evento, y de forma concurrente.
- Arquitectura escalable y muy distribuida
- Cada subsistema puede reaccionar de forma diferente a un evento dentro del flujo de ejecución, lo que nos permite "re-interpretar" el flujo bajo demanda.

¿Qué nos aporta el Serverless en nuestra arquitectura?

Una vez tenemos claro todo lo que nos aporta una arquitectura basada en eventos, y que tenemos alternativas para plantear nuestros eventos, deberemos ajustar el uso y consumo de nuestros servicios Serverless.

"Por esto podemos decir que estas arquitecturas implican "procesos" en segundo plano"

Evidentemente, los servicios Serverless, y en concreto los servicios que nos aporta Azure, nos permiten subir al siguiente nivel de complejidad nuestras arquitecturas, pero ¿Siempre son necesarios incluir un Azure Functions o una Logic App?

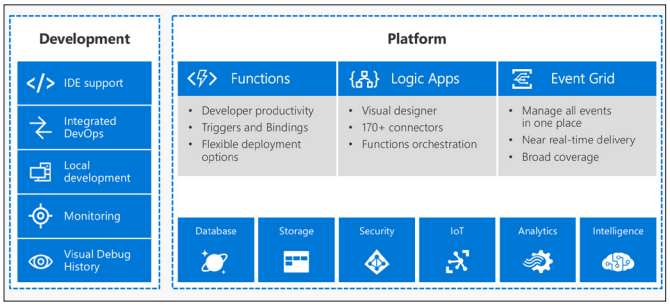


Imagen 2.- Azure Serverless.

La respuesta es NO, muchas veces podemos mantener nuestros modelos clásicos basadas en aplicaciones Monolíticas, que van a tener el mismo o mejor rendimiento que un servicio desplegado en modo Serverless. Es decir, por ejemplo, un App Service bien escalado en algunos casos, puede sernos más útil e incluso más económico.

Una vez puesto el WARNING encima de la mesa, debo decir que, para una arquitectura basada en eventos, nos van a venir ni que al pelo tanto Azure Functions como Logic App.

Podemos pensar que nuestra arquitectura se puede basar en servicios Serverless, siempre y cuando los procesos a implementar cumplan que:

- Son iniciados por un Evento.
- No tienen definido un límite de ESCALADO.
- No conocemos VOLUMENES de ejecución.
- No conocemos el Nº de Usuarios ni la CONCURRENCIA MÁXIMA.
- Debe funcionar el 99% de los casos.
- No está en ALTA DISPONIBILIDAD.
- Preferiblemente para procesos en background.

Podemos decir entonces que con todo lo anterior, ya tenemos una idea más o menos clara de cuando utilizar arquitecturas basadas en Eventos, y, por otro lado, en que escenarios deberíamos plantear el uso de servicios Serverless, pero ¿Cómo arrancamos el diseño sin volvernos locos?

Azure Event Grid, el concentrador de eventos

Siempre que juntemos en nuestros requisitos "Tiempo Real", "Concurrencia", y "Gran Volumen de eventos", no estaría de más pensar en Azure Event Grid.

Para ir situando a este potente servicio de Azure, debemos decir que Event Grid viene a ser un servicio que administra la relación entre los Productores de Eventos y los Consumidores. Pongamos un ejemplo de uso para ver claro donde encajar Event Grid. Imaginemos un escenario en el que dos sistemas esperan a que se suba un fichero a un Blob Storage.

Cada sistema tiene un proceso propio para el re-escalado

de imágenes, ya que el primer sistema muestra las imágenes en un Portal Web a una resolución muy grande de pantallas, y el segundo está orientado a re-escalar estas mismas imágenes, pero para una vista móvil.

Estos dos procesos se van a disparar, una vez se suba la imagen al Blob Storage, y en tiempo real cada uno de ellos generará copias de las imágenes y las dejara en un segundo blob que consume tanto la web, como la app móvil.

Imagen 3.- Ejemplo de Trigger.

Para cubrir este escenario, sin utilizar Event Grid, necesitaríamos 2 funciones y dos Trigger configurados contra el blob storage de origen. O lo que es lo mismo, 2 eventos independientes, y que funcionen de forma totalmente aislada.

Esto conlleva, llevar una monitorización y una gestión de reintentos individuales para cada proceso, lo que lleva a complicar el mantenimiento y el aseguramiento de los procesos.

Llevar un control de eventos es necesario, para no caer en una micro gestión de estos trigger, y para ello deberíamos tender a una arquitectura basada en Eventos y por Suscripciones a eventos.

Azure Event Grid entre otras cosas, viene a "centralizar todos los eventos", y a gestionar esas suscripciones, por eso decimos que va a hace de PEGAMENTO, entre nuestros Productores o Publicadores de eventos, y los Suscriptores o Consumidores.

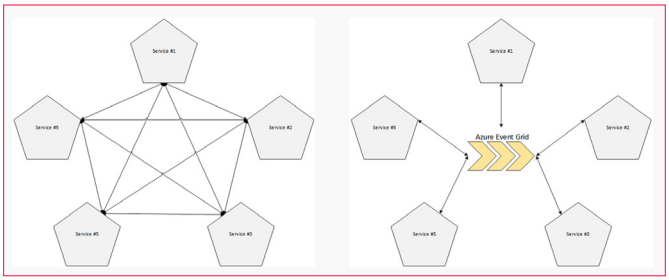


Imagen 4.- El pegamento del serverless, Azure Event Grid.

Como vemos muy gráficamente en la imagen anterior, Azure Event Grid, viene a controlar todos esos eventos que hemos ido creando entre nuestros Servicios Consumidores, y el origen de estos Eventos, que en muchos casos pueden ser sistemas de Almacenamiento, Aplicaciones SaaS o Bases de datos.

Conceptos básicos de Azure Event Grid

Para empezar a manejar Azure Event Grid, tenemos que entender varios conceptos básicos del servicio para que nos sea sencillo utilizarlo. Como ya hemos ido viendo con las Arquitecturas basadas en Eventos, este servicio se basa en conectar al Publicador con el Suscriptor vía un evento, y para ello define los siguientes conceptos:

- Events: Lo que ha sucedido.
- Publisher: Donde ha sucedido el evento, sistema que genera el evento.
- Topics: Canal de envío del evento y categorización del evento, por ejemplo, para un Blob Storage, Upload Blob, es el topic para capturar las subidas de ficheros.
- Suscriptions: Como recibimos el evento, por ejemplo, vía Webhook.
- Handlers: App o servicio que recibe el evento.

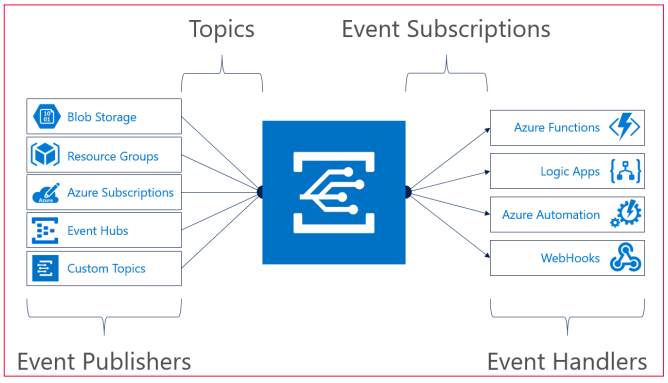


Imagen 5.- Conceptos de Azure Event Grid.

Características del Servicio

Además, a estas características debemos sumar que a nivel de rendimiento:

- Siempre disponible.
- Near real-time event delivery.
- Asegura al menos una vez el envío del evento.
- Escalado dinámico.
- Platform agnostic (WebHook).
- Language agnostic (HTTP protocol).
- Enrutado de eventos.

Un primer ejemplo, Azure Functions y Blob storage conectados por Event Grid por el portal de Azure

Como la mejor forma de terminar de entender algo, es empezar a probarlo vamos a hacer un ejemplo muy sencillo desde el portal de Azure, para que veamos cómo funciona.

Para ello debemos logarnos en el portal y crear dos servicios:

- Azure Storage Account:

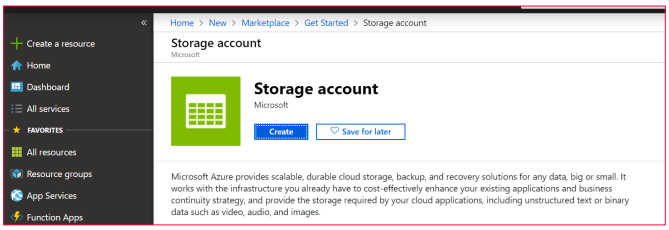


Imagen 6.- Creando un Storage Account.

Crearemos el servicio en un grupo de recursos nuevo llamado CompartimossEventGrid, y configuramos el servicio de la siguiente forma:

- Storage Account Name: storageegdemo
- Location: West Europe
- Performance: Standard
- Account Kind: Storage V2
- Replication: RA-GRS
- Acces tier: Host

Dejamos las configuraciones de red a por defecto, y confirmamos la creación del servicio.

- Azure Funcions App:

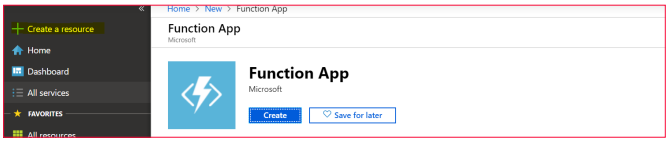


Imagen 7.- Creando un Function App.

Crearemos un servicio Azure Functions App, en el grupo de recursos CompartimossEventGrid, con el nombre fappeventgriddemo, tal y como se ve en la siguiente imagen.

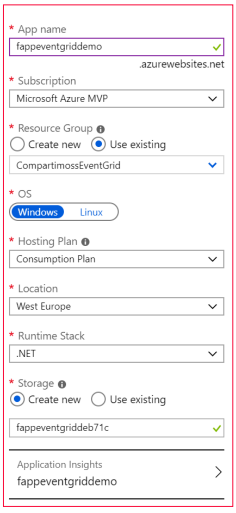


Imagen 8.- Configurando mi plan de Functions.

Si todo ha ido de forma correcta, al acceder al grupo de recursos deberíamos tener los siguientes servicios:

fappeventgriddeb71c	Storage account	West Europe
fappeventgriddemo	Application Insights	West Europe
fappeventgriddemo	App Service	West Europe
storageegdemo	Storage account	West Europe

Imagen 9.- Recursos necesarios para el ejemplo.

Crear una función y conectarla vía Event Grid al Blob

Una vez creado los servicios, accedemos al servicio fappeventgriddemo, y creamos una función de la siguiente forma:

1.– Seleccionamos Nueva Función, en el apartado de Funciones.

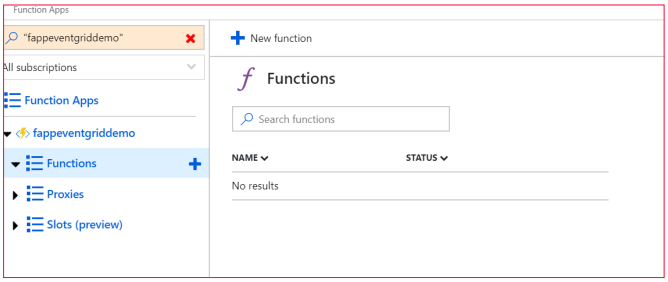


Imagen 10.- Creando la function.

2.– Este segundo punto es importante, ya que necesitamos escoger el desencadenador de la nueva Función, y escogeremos Azure Event Grid Trigger.

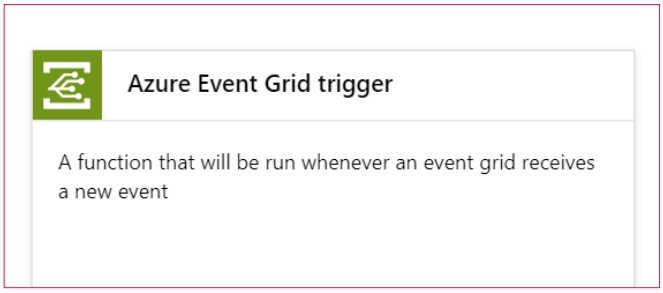


Imagen 11.- El trigger del tipo Event Grid.

3.– Necesitaremos instalar la extensión Microsoft.Azure.WebJobs.Extensions.EventGrid para que funcione el binding de Event Grid, este proceso solo lo deberemos hacer con la primera Función de cada servicio Azure Functions.

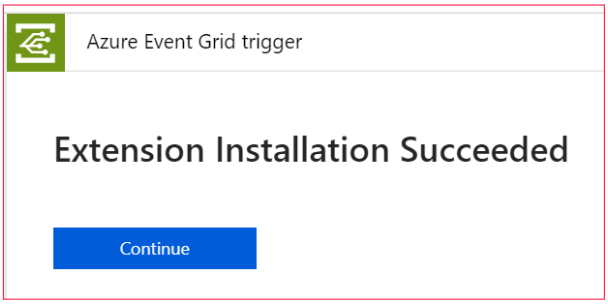


Imagen 12.- Instalando los paquetes de Event Grid.

4.– Una vez terminada la instalación, introducimos como nombre de la función EventGridTriggerDemo, y pulsamos en crear. Cuando el proceso de creación termina en pocos segundos, podremos ver el contenido del fichero run.csx, donde podremos ver el código que se ha autogenerado:

```
#r "Microsoft.Azure.EventGrid"
using Microsoft.Azure.EventGrid.Models;

public static void Run(EventGridEvent eventGridEvent, ILogger
log)
{
   log.LogInformation(eventGridEvent.Data.ToString());
}
```

Sin analizamos el código, tenemos que ver que la función espera un objeto del tipo EventGridEvent, que contiene toda la información del evento. Podemos revisar la definición de la clase en el siguiente enlace.

La función lo que hace es un print por pantalla del evento recibido, por lo que cuando enlacemos con el blob storage esta función podremos ver con más detalle el contenido de un evento, y así analizar la clase EventGridEvent

Una vez el código nos permite recibir eventos de event grid, necesitamos crear la suscripción de Event Grid, como veremos a continuación.

Creando la suscripción

Lo primero que debemos hacer es seleccionar la función recién creada, y seleccionar "Add Event Subscription", que encontramos en el fichero run.csx. Este enlace nos lleva al menú de Event Grid, que nos permite crear una suscripción y que deberemos configurar de la siguiente forma:

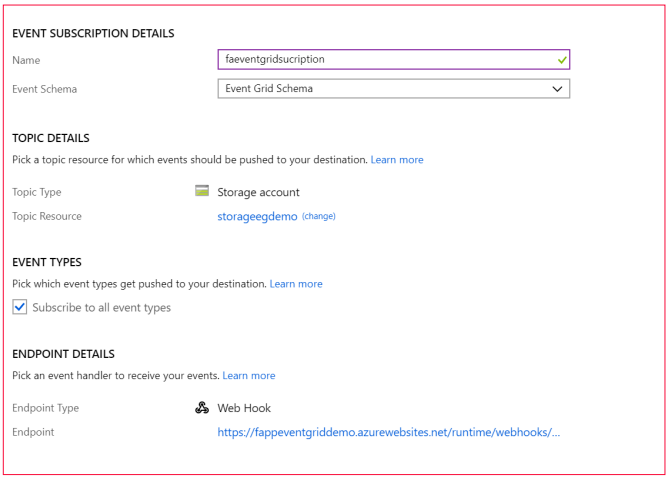


Imagen 13.- Creando la suscripción al blob.

Si analizamos la imagen, veremos que necesitamos dar un nombre a la suscripción, y elegir un esquema que en este caso usaremos el básico de Event Grid para que nos cuadre con la clase EventGridEvent que hemos nombrado en la creación de la función.

Seleccionamos por otro lado el Topic, que no deja de ser el origen del evento, para ello seleccionamos un tipo Storage Account, y seleccionamos la cuenta creada al principio del ejemplo.

Seleccionamos el tipo de Evento, que en caso de un blob pueden ser Blob Created y Blob Deleted, podemos elegir el que más nos guste, en mi caso he seleccionado todos.

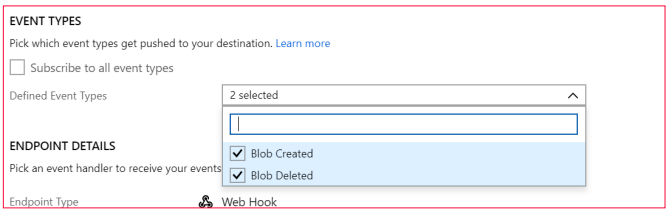


Imagen 14.- Definiendo el topic.

Por último, debemos ver que se nos autogenera una dirección Webhook, que apunta a nuestra función, y que hará de cliente para recibir en tiempo real todos nuestros eventos.

Vamos a omitir en este ejemplo el apartado de Filtros y Características extras que nos aporta una suscripción event grid, que poco nos aporta para un primer ejemplo sencillo.

Probando la suscripción subiendo una imagen al blob

Para poder probar la suscripción, lo primero vamos a crear un blob dentro de la cuenta storagedegdemo, para hacer saltar a Event Grid. Seleccionamos el servicio de almacenamiento, y en el listado de servicios seleccionamos blob, donde seleccionaremos Crear Container.

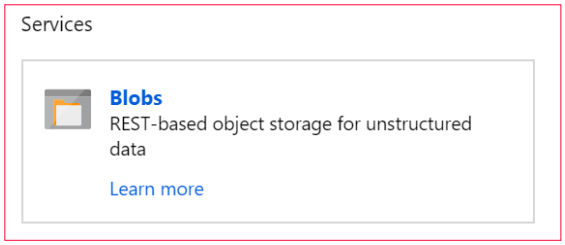


Imagen 15.- Creando el blob.

Una vez creado el contenedor, vamos a abrir en una segunda pestaña la Azure Function que hemos conectado con este servicio de blob. Si seleccionamos la función y justo debajo del código, seleccionamos el tab LOG, en el que podremos ver en tiempo real el imprimir del evento que tiene el código de la función.

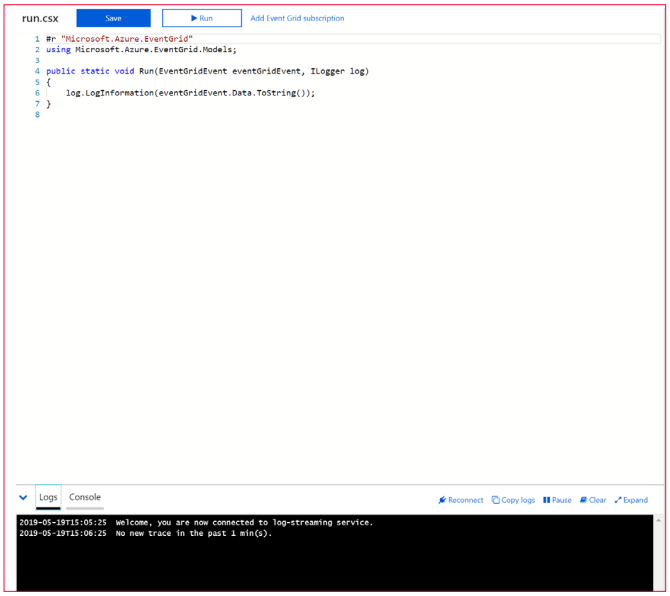


Imagen 16.- Depurando la function.

Volviendo al blob, seleccionamos el blob creado y hacemos un Upload de cualquier fichero que tengamos en nuestra maquina local, para probar nuestra suscripción de Event Grid.

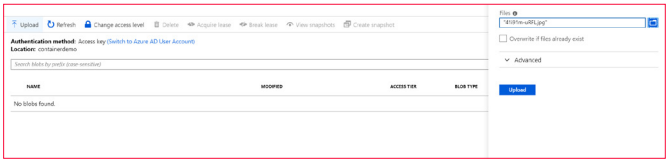


Imagen 17.- Subiendo un fichero al blob.

Una vez que subamos el fichero, en tiempo real, en la consola de Log de nuestra function veremos que se ha lanzado una ejecución, y en consola de log podremos ver todos los datos del Evento subir fichero a un blob.

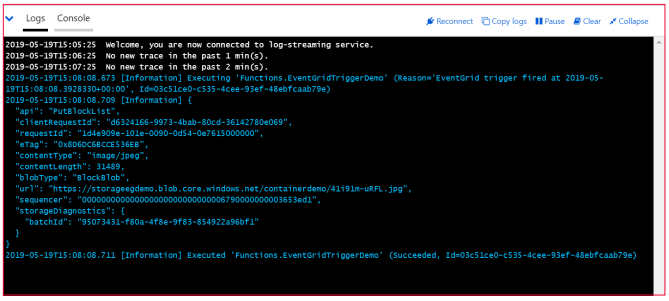


Imagen 18.- Capturando un evento de Event Grid.

A partir de aquí ya tenemos conectado el blob a nuestra Azure Function, y podemos añadir la lógica de negocio necesaria para tratar este evento.

Si analizamos los datos, vemos que recibimos en un JSON los datos relativos al evento, y por destacar:

- Recibimos en un campo URL el fichero que se ha subido.
- Content-type: El tipo de fichero que se ha subido, en este caso yo he subido una imagen.
- Api: nos indica que se ha realizado un put de un fichero

Con estos datos, podemos hacernos a una idea de que fichero se ha subido, y si fuera necesario podríamos transformar la imagen subida o enviársela a un tercer sistema.

Azure Event Grid, un servicio para tener muy en cuenta

Aunque lo hemos visto muy por encima, porque a día de hoy Event Grid está muy evolucionado y nos permite hacer sistemas de suscripción muy complejas para arquitecturas basadas en Eventos, con este artículo nos podemos hacer a la idea de cómo empezar a trabajar con Event Grid, y sobre todo la necesidad o no de usar tanto Event Grid como por ejemplo Azure Function.

Lo más importante cuando diseñamos arquitecturas, es tener 100% claro lo que necesitamos y si los servicios que estamos utilizando son los más idóneos. Nunca debemos utilizar servicios porque nos parezcan potentes o novedosos, a veces arquitecturas más humildes nos pueden llevar al éxito de igual o de mejor forma.

En cualquier caso, este tipo de arquitecturas son muy potentes, muy interesantes para Organizaciones que tengan muchos sistemas distribuidos, y necesidad de conectarlos en tiempo real. En siguientes artículos veremos cómo estos servicios Serverless se convierten en imprescindibles por ejemplo para extender Office 365 tanto para Dynamics como para Sharepoint, ya que Event Grid se convierte en nuestra puerta de entrada a todo lo que pasa en nuestras aplicaciones.

Sin duda tenemos las herramientas, aun necesitamos la pausa para poner todo en orden, y combinar nuestro Serverless con otros servicios como Azure Search o Azure Service Bus, y construir sistemas escalables, seguros concurrentes, y sobre todo muy rápidos; “el tiempo real es una realidad hoy día”.

SERGIO HERNANDEZ MANCEBO
Principal Team Leader en Encamina
Azure MVP
@shmancebo

13

Azure Digital Twins

Azure Digital Twins es un nuevo servicio de Azure con el que podremos que crea modelos completos del entorno físico, pudiendo crear grafos de inteligencia espacial para modelar las relaciones y las interacciones entre personas, espacios y dispositivos.

En este articulo nos centraremos en como poder crear un nuevo servicio de Azure Digital Twins a través de una suscripción de Azure, consultar los datos de los sensores simulados y comprobar si la habitación esta disponible para su uso usando los datos extraídos de los sensores.

Para la creación de la instancia de Digital Twins en nuestra suscripción de Azure deberemos de entrar en el portal de Azure -> Crear un Recurso -> Digital Twins

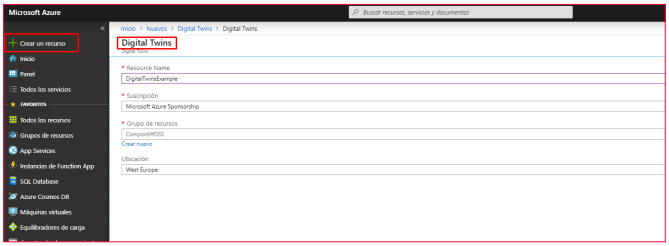


Imagen 1.- Creación del recurso de Digital Twins en el portal de Azure.

Tened en cuenta que solo se puede crear una única instancia de Digital Twins por suscripción.

Una vez que ya tenemos aprovisionada la instancia del servicio Digital Twins podremos ver la siguiente información general, donde encontraremos la url del Management API. Desde esta url nos mostrará todas las capacidades que ha aprovisionado el servicio de Digital Twins y que podremos usar.

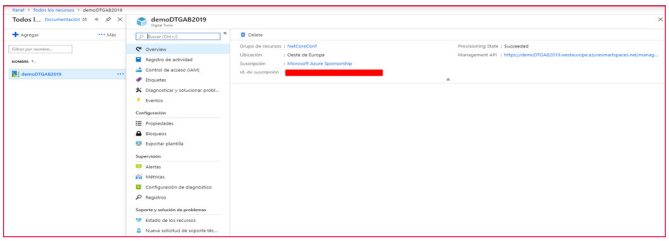


Imagen 2.- Información general del servicio.

La url que nos muestra por defecto el servicio es la que contiene toda la información de la API REST de Azure Digital Twins que se aplica a la instancia, normalmente suele tener el siguiente formato: https://yourDigitalTwinsName.yourLocation.azuresmartspaces.net/management/swagger

Sin embargo, para poder tener acceso a la instancia del servicio deberemos de modificar la url: https://yourDigitalTwinsName.yourLocation.azuresmartspaces.net/management/api/v1.0/

(esta url podemos guardarla en un fichero de texto para poder usarlo en los pasos posteriores)

Ahora deberemos de definir los permisos para nuestra instancia de Digital Twins, por lo que deberemos de entrar en el portal de Azure -> Registros de Aplicaciones -> Nuevo registro de aplicaciones.

Le asignaremos un nombre, seleccionaremos para este ejemplo la opción solo las cuentas de este directorio organizativo del aparatado Tipos de Cuentas compatibles he informaremos la url de nuestra web. Por último, registraremos la aplicación.

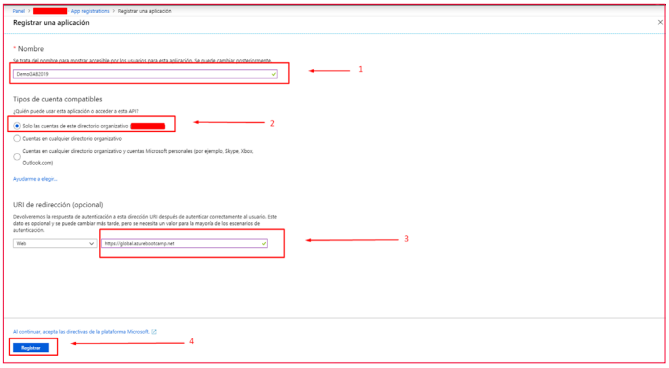


Imagen 3.- Registro de una nueva aplicación en AD.

Una vez que hemos registrado la aplicación podemos revisar en la información general el id de aplicación, Id de directorio (esta información podemos guardarla en un fichero de texto para poder usarlo en los pasos posteriores para provisionar y consultar los datos de la instancia). Debemos irnos al apartado Permisos de API y añadir permisos.

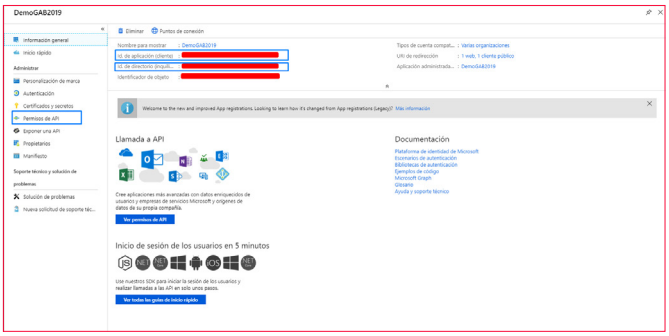


Imagen 4.- Información general de la nueva aplicación-

Muy importante que cuando añadamos el permiso busquemos en API usadas en mi organización con el nombre Azure Digital Twins. Una vez seleccionado le daremos los permisos de Lectura/Escritura y para finalizar deberemos de Conceder Permisos.

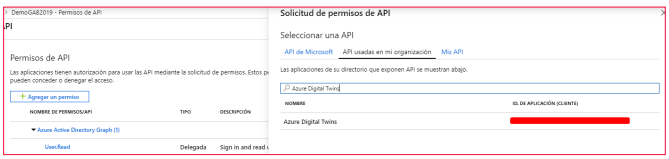


Imagen 5.- Permisos para la API Digital Twins

Por último, deberemos de ir a portal de Azure -> Azure Active Directory -> Propiedades donde podremos consultar el Id de Directorio (esta información podemos guardarla en un fichero de texto para poder usarlo en los pasos posteriores).

"Azure Digital Twins es una plataforma que proporciona a las organizaciones la base para poder realizarla"

Con estos pasos ya tenemos provisionado nuestro servicio de Azure Digital Twins para poder trabajar con él. Ahora vamos a usar un ejemplo ya preparado por el equipo de Microsoft que podréis descargar desde GitHub en el siguiente enlace (https://github.com/Azure-Samples/digital-twins-samples-csharp/) para poder "provisionar una estructura completa".

Una vez que nos descarguemos el ejemplo y lo hayamos descomprimido en una carpeta en nuestro sistema, por ejemplo C:\samples\digital-twins-sample, os recomendaría abrirlo con Visual Studio Code y de esta forma tener una visión del proyecto además de la terminal en una sola ventana.

Cuando entramos en la carpeta descomprimida observamos dos proyectos netcore que son occupancy-quickstart (nos permitirá consultar los datos en tiempo real) y device-connectivity (nos permitirá aprovisionar nuestro servicio con sensores simulados).

Nos centraremos primero en el aprovisionamiento, para ello recordad que en los pasos anteriores os pedía que guardarais cierta información, ahora es el momento de usarla. Iremos a la ruta occupancy-quickstart\src\ appSettings.json

```
{
  "AADInstance": "https://login.microsoftonline.com/",
  "ClientId": "Id registro aplicación de AD",
  "Tenant": "Id del directorio",
  "BaseUrl": "Nuestra URL modificada de Azure Digital Twins"
}
```

Imagen 6.- Fichero appSettings.json.

Una vez guardemos el fichero con los datos solicitados, desde consola realizaremos los siguientes comandos:

- cd occupancy-quickstart\src
- dotnet restore
- dotnet run ProvisionSample

La primera vez que lo hacemos nos pedirá que nos loguemos en una url con un código determinado que se genera aleatoriamente para identificar el servicio. Al ser un servicio que está actualmente en preview se solicitara cada 24h.

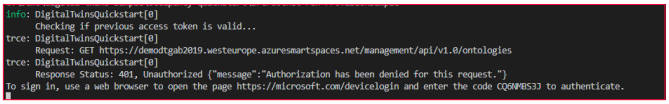


Imagen 7. Terminal con la petición de autorización para aprovisionar el servicio.

Una vez que hemos autorizado el servicio a través de los pasos anteriores, empezara a generar el aprovisionamiento que se ha definido concretamente para tener más información es en el archivo digital-twins-sample\occupancy-quickstart\src\actions\ provisionSample.yaml

Sabremos que ha finalizado cuando encontremos al final la instrucción "Completed Provisioning" además ahora deberemos de buscar la clave ConnectionString y copiaremos su valor en un archivo de texto para usarlo en el proyecto device-connectivity para poder observar los datos.

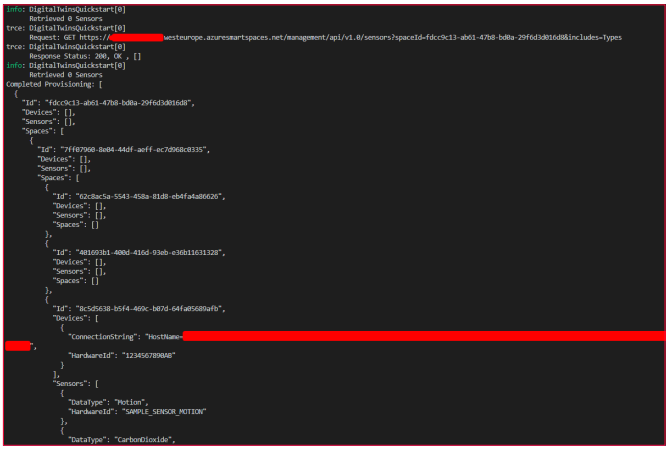


Imagen 8.- Teminal con el resultado del aprovisionamiento completado-

Deberemos de ir al fichero digital-twins-sample\device-connectivity\appsettings.json y copiar el valor que hemos obtenido en el aprovisionamiento del paso anterior, como se puede observar aquí es donde también definimos los dispositivos que queremos observar:

```
{
  "Settings": {
    "DeviceConnectionString": "HostName=[redacted]",
    "MessageIntervalInSeconds": 5,
    "Sensors": [{
      "DataType": "Motion",
      "HardwareId": "SAMPLE_SENSOR_MOTION"
    },{
      "DataType": "CarbonDioxide",
      "HardwareId": "SAMPLE_SENSOR_CARBONDIOXIDE"
    }]
  }
}
```

Imagen 9.- Fichero appSettings.json.

Ahora desde una terminar realizaremos los siguientes comandos:

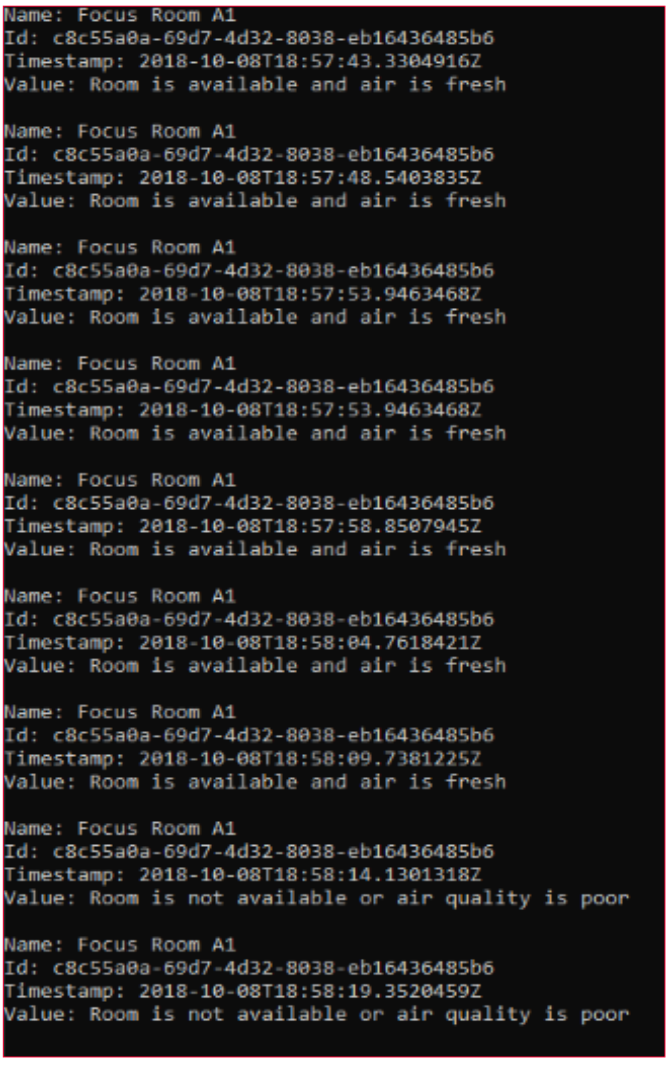
- cd device-connectivity
- dotnet restore
- dotnet run

Imagen 10.- Terminal con los resultados en tiempo real de los datos de los sensores-

Podemos observar como nos devuelve los resultados del sensor en tiempo real. Para poder tener una visión completa de Azure Digital Twins abrimos una nueva terminal en paralelo y lanzamos los siguientes comandos:

- cd occupancy-quickstart\src
- dotnet run GetAvailableAndFreshSpaces

Si colocamos las dos ventanas en el escritorio podremos ver que en la primera estamos obteniendo los datos en tiempo real de los sensores y en la otra terminal estamos tratando esos datos y mostrándonos los espacios disponibles con aire fresco que tenemos en ese momento. Esta lógica se ha definido en el fichero digital-twins-sample\occupancy-quickstart\src\actions\userDefinedFunctions\availability.js



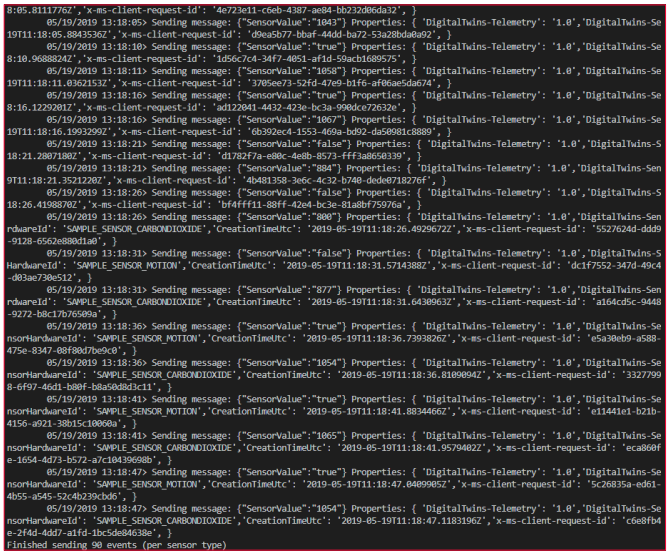


Imagen 11.- Terminal con los datos en tiempo real y tratados.

Para poder tener una visión completa de todo los dispositivos, estancias que tenemos aprovisionadas podemos usar un visor de grafos espacial (https://github.com/Azure/azure-digital-twins-graph-viewer) con el que podremos ver de una forma más visual nuestro aprovisionamiento que hemos realizado en este ejemplo:

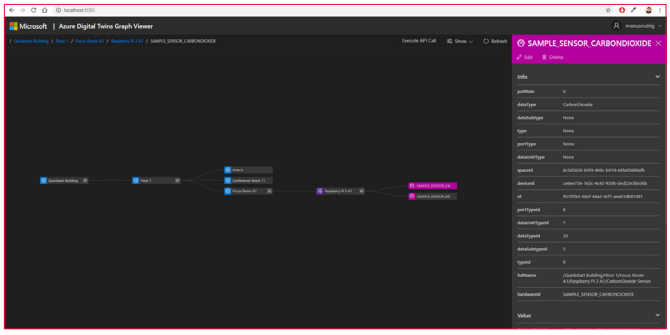


Imagen 12.- Vista del aprovisionamiento

Conclusiones

Sinceramente pienso que el nuevo servicio de Azure Digital Twins es muy útil para poder representar el mundo físico y sus muchas relaciones, con ello puede ayudar a simplificar el modelado, procesamiento de datos, control de eventos y el seguimiento de dispositivos IoT. Con ello el poder llevar la transformación digital a las diferentes industrias se puede conseguir de una forma segura, fácil y sobre todo unificada.

Por otro lado, debemos de tener en cuenta que actualmente es un servicio en preview y por tanto tiene algunas limitaciones. Aunque se espera que para final de año se libere completamente.

MANUEL SÁNCHEZ RODRÍGUEZ
Manuss20@gmail.com
@manuss20
https://manuss20.com

16

Run Multiple Instances of Microsoft Teams

Being able to execute multiple instances of Microsoft Teams is a long-waited feature that is affecting negatively all the users that belong to multiple organizations. If you are doing multiple tenant swaps during the day to keep up with everything that is happening in all your organizations this article is for you, even though Microsoft Teams does not support multiple instances you will be able to execute the web version as standalone app and you will be able to be signed in in all the organizations ate the same time.

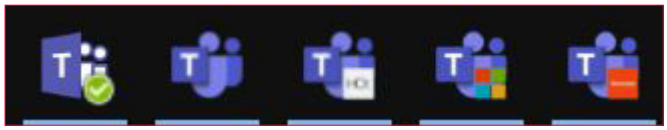
"all the users that belong to multiple organizations. If you are doing multiple tenant swaps during the day"

In order to accomplish the steps described in this article you will need to install Node.js, instructions and the installation packages can be found here.

Once Node.js is installed in your system open the command line and execute the command below.

npm install -g nativefier

Nativefier is a command-line tool to easily create a desktop application for any web site with succinct and minimal configuration. Apps are wrapped by Electron in an OS executable (.app, .exe, etc.) for use on Windows, macOS and Linux. More information about this project can be found in the GitHub repository here.



With all the requirements installed follow the steps bellow to create your own Microsoft Teams App.

1.– Create a folder where the new app will be created

2.– Open the console and navigate to the folder

3.– Execute the command below:

```
nativefier.cmd https://teams.microsoft.com/_?tenantID=-YOURTENANTID --name "YOUR NAME" --internal-urls .* --icon "Logo.ico"
```

- Provide a name for your new Microsoft Teams Application.
- Get the tenant ID to the organization where you want to connect, detailed instructions can be found down below.
- Copy the logo for your new Microsoft Teams application to the folder where the app will be created. (Note: On Windows the icon needs to be in the *.ico format).

4.– Once the execution ends open the folder and look for the *.exe file

5.– Open the application, provide your credentials and then select Use the Web App Instead

How to get Microsoft Teams tenant ID

Your Microsoft Teams tenant ID is a unique identifier that is different than the organization domain where you are connected. You can use two different methods to get this identifier, one using Teams graphic user interface and the other one using Teams PowerShell.

Get Tenant ID using GUI

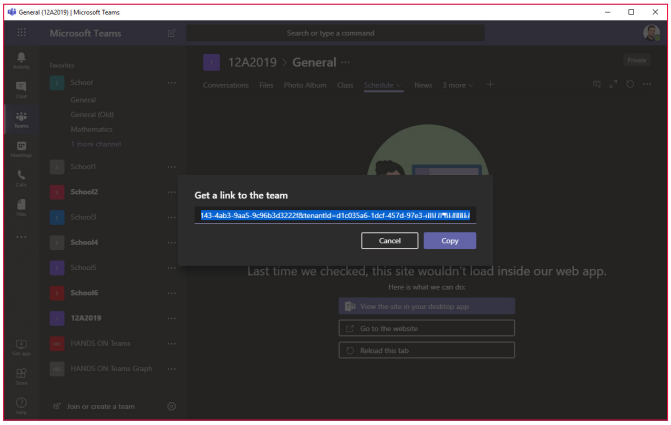
1.– Open one of the Teams where you belong inside the Teams Application.

2.– Click on the … next to the Team title.

3.– Click on Get link to team.

4.– Copy the link to the team and grab the tenant ID form the URL.

"is a command-line tool to easily create a desktop application for any web site with succinct"



Get Tenant ID using PowerShell

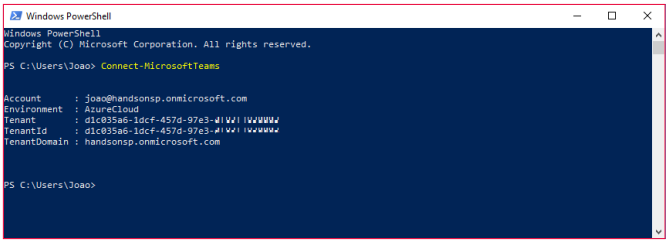
In order to follow this process, you need to install Teams PowerShell, it's available here.

1.– Open the PowerShell window.

2.– Execute the command Connect-MicrosoftTeams.

3.– Provide your login credentials.

4.– Copy the tenant ID from the console:



JOAO FERREIRA
Office Development MVP
SharePoint Team Lead at BindTuning

18

"Library Components" en SharePoint Framework 1.8

A finales de marzo tuvimos una nueva major release del spfx, la 1.8.0, que trajo algunas novedades interesantes, y que fue actualizada cerca de un mes después a la 1.8.1 (que simplemente corrige algunos de los bugs que se colaron en la 1.8). Si queréis conocer todas las novedades incluidas en la 1.8.0, os aconsejo visitar este artículo de Andrew Connell, que hace un resumen muy completo:

http://www.andrewconnell.com/blog/sharepoint-framework-v1-8-0-what-s-in-the-latest-update-of-spfx

En este artículo nos vamos a centrar en una novedad de las más interesantes, llamada "Library Components", y que nos va a permitir crear código compartido (a modo de Librearías) entre diferentes soluciones SPFx. Seguro que tienes funciones de validaciones que copias en cada proyecto, utilidades para llamadas REST, o pequeños (dumb) componentes React, que pintan información con tu formato favorito (fecha en formato largo, imagen de un usuario con bordes redondeados, etc.). Si todavía no es tu caso, lo será en cuanto desarrolles más soluciones SPFx, ¡así que te aseguro que este artículo te interesa!

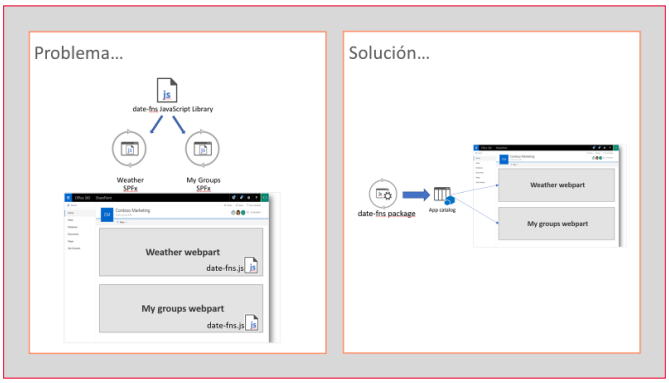


Imagen 1.- Overview de los Library Componentes en SPFx.

El problema, y la regular solución con "externals"

Si hasta ahora te limitas a copiar el código en cada solución SPFx, que sepas que, si tienes varios WebParts de diferentes soluciones, cargados en la misma página de SharePoint, ese código va a estar duplicado, penalizando el rendimiento en la carga de la página.

Antes de la 1.8, había una forma de solucionar este problema de código compartido, pero no era muy friendly, y requería de configuración extra. Dicha solución se basaba en crear tu propia librería de javascript, es decir, que te configurabas un proyecto con tu webpack, typescript compilation, tslint, etc, que daba como resultado final un fichero JavaScript .js. El siguiente paso era subir ese fichero a algún CDN, pudiendo ser el mismo de Office 365, que de nuevo has de configurar a mano. Y finalmente, en nuestro proyecto spfx, debemos configurar el nodo externals del fichero config.json:

```
"externals": {
  "contoso-utilities-library": "https://publiccdn.sharepointonline.com/contoso.sharepoint.com/sites/appcatalog/Library-Components/contoso-utilities-library-component.js"
},
```

Con esto todavía no es suficiente, y tendremos que hacer algunas cosas con npm link para poder utilizar la librería desde nuestro proyecto spfx. Tenéis más detalle de como hacerlo en este artículo: https://spblog.net/post/2019/03/26/a-new-beast-in-sharepoint-framework-development-library-component

Como veis, no es muy sencillo solucionar este problema sin las mejoras venidas en la versión 1.8.0

Solución, "Components Library"

Para ilustrar el artículo, vamos a crear nuestra librería de componentes, donde tendremos una función que formateará una fecha usando la librería date-fns. Dicha función será utilizada en dos soluciones spfx diferentes.

Primero crearemos un proyecto spfx haciendo uso de la plantilla yeoman que ya todos conocemos.

```
C:\Temp\compartimoss\inheritscloud-spfx-library
λ yo @microsoft/sharepoint --plusbeta
```

Nota: Esta característica está todavía en beta, así que tendremos que añadir el parámetro –plusbeta para poderla utilizar.

Al ser una librería compartida por cualquier solución SPFx, debemos contestar Y a la pregunta de "allow solution to be deployed to all sites immediately."

En un momento, el asistente de yeoman nos pregunta el tipo de proyecto, y seleccionaremos la opción "Library"

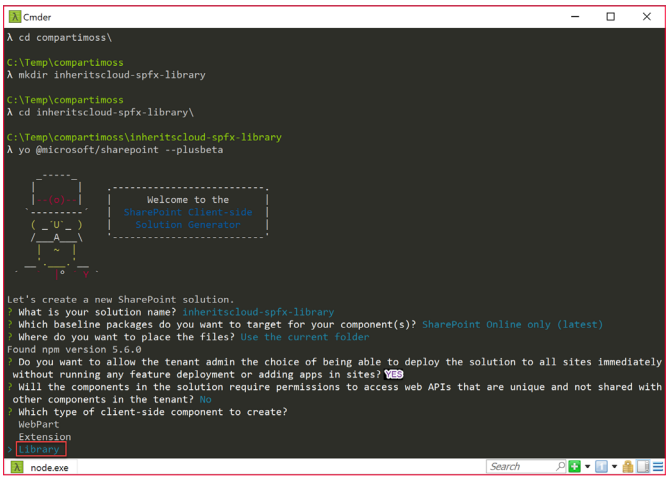


Imagen 2.- Especificando el tipo de proyecto.

Luego daremos un nombre y descripción a nuestra Librería, y finalizaremos el proceso de creación del proyecto.

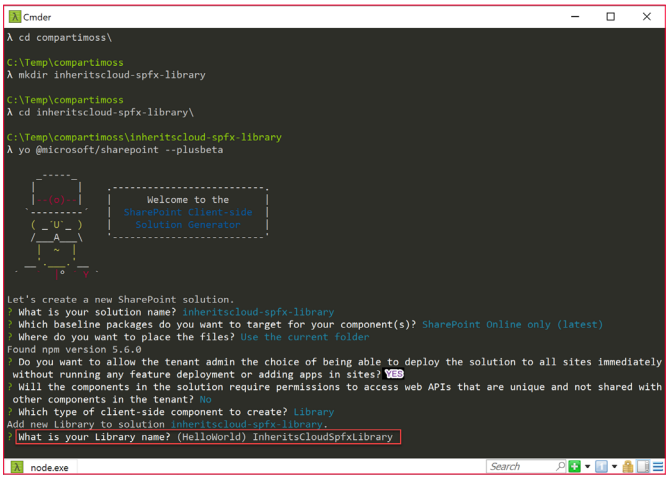


Imagen 3.- Especificando el nombre de la Library.

Una vez creado el proyecto, podemos ver como el package.json contiene una referencia a la dependencia sp-core-library

```
"dependencies": {
  "@microsoft/sp-core-library": "1.8.1-plusbeta",
  "@types/webpack-env": "1.13.1",
  "@types/es6-promise": "0.0.33"
},
```

También vemos como el manifest de la librería, tiene un atributo componentType con el valor Library

```
{
  "id": "17e2d1af-9b9a-4173-9a0d-a37b4921ada9",
  "alias": "InheritsCloudCoreLibrary",
  "componentType": "Library",

  // The "*" signifies that the version should be taken from the package.json
  "version": "*",
  "manifestVersion": 2
}
```

Lo siguiente que haremos será incluir nuestra función reutilizable dentro del código de la librería. Como hemos dicho anteriormente, vamos a tener una función que va a formatear una fecha con formato longdate y además nos dirá cuánto tiempo ha pasado desde esa fecha, hasta hoy (el típico: "hace 3 días"). Para ello vamos a hacer uso de una librería muy famosa que facilita el trabajo con fechas, llamada date-fns (quizá conozcas moment.js, esta otra es similar a moment, pero algo más ligera y modular). Al ser una dependencia, tenemos primero que instalarla a nuestro proyecto, así que ejecutamos el siguiente comando npm:

```
npm install date-fns --save
```

Una vez instalado, añadimos el siguiente código a la Librería:

```
import { format, distanceInWords } from 'date-fns';

export default class InheritsCloudCoreLibrary {
  public name(): string {
    return 'InheritsCloudCoreLibrary';
  }

  public formatDateAsCoolString(date: Date): string {
    const timeAgo: string = distanceInWords(
      date,
      new Date(),
      {addSuffix: true}
    );

    const dateFormatted = format(date, "Do MMM YYYY");

    return `${dateFormatted} (${timeAgo})`;
  }
}
```

Ya tenemos el código de nuestra librería preparada, así que la siguiente parte es desplegarla en el App Catalog, como cualquier solución spfx. Si necesitas los pasos detallados, los tienes aquí:

https://docs.microsoft.com/en-us/sharepoint/dev/spfx/library-component-tutorial#how-to-deploy-and-consume-a-3rd-party-spfx-library-from-tenant-app-catalog

Probando nuestra Librería de componentes

El siguiente paso será crear un proyecto SPFx de WebPart siguiendo el método ya conocido. Una vez creado el proyecto, debemos añadir una dependencia a nuestra Library. Esto lo haremos editando el package.json y añadiendo un dependencia más:

```
{
  "name": "inheritscloud-spfx-library-clientone",
  "version": "0.0.1",
  "private": true,
  "engines": {
    "node": ">=0.10.0"
  },
  "scripts": {
    "build": "gulp bundle",
    "clean": "gulp clean",
    "test": "gulp test"
  },
  "dependencies": {
    "inheritscloud-spfx-library": "0.0.1", // here we added the reference to the library
    "react": "16.7.0",
    "react-dom": "16.7.0",
    "@types/react": "16.4.2",
```

El nombre y la versión lo encontramos en el package.json del proyecto Librería:

```
{} package.json ×
1  {
2    "name": "inheritscloud-spfx-library",
3    "version": "0.0.1",
4    "private": false,
```

Llegado a este punto, si intentamos hacer un import de nuestra librería para usarla en nuestro código, algo como:

```
import { InheritsCloudCoreLibrary } from 'inheritscloud-sp-fx-library';
```

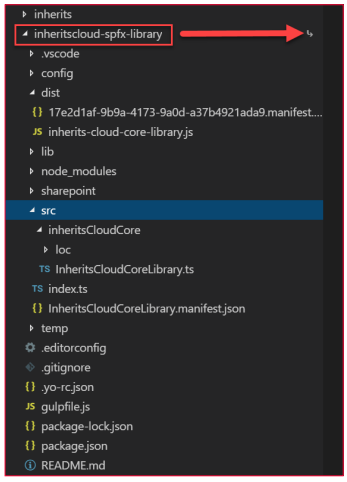
Veremos cómo VS Code nos dice que no sabe nada de ese módulo. Tenemos que, de alguna manera, exponer la Librería. Para ello, vamos a la carpeta del proyecto Librería, y ejecutamos:

```
npm link
```

Ahora, volvemos a la carpeta de nuestro proyecto WebPart SPFx, y ejecutamos:

```
npm link inheritscloud-spfx-library
```

Esto va a incluir una referencia dentro de la carpeta node_modules a nuestro proyecto librería:



Ahora sí, ya podemos crear un objeto de nuestra librería, y poder utilizarlo en nuestro WebPart:

```
export default class HelloLibrary extends React.Component<IHelloLibraryProps, {}> {
  public render(): React.ReactElement<IHelloLibraryProps> {

    const library = new InheritsCloudCoreLibrary();
    const testDate = new Date(2019, 5, 3);
    const dateFormatted = library.formatDateAsCoolString(testDate);

    return (
      <div className={ styles.helloLibrary }>
        <div className={ styles.container }>
          <div className={ styles.row }>
            <div className={ styles.column }>
              <span className={ styles.title }>Webpart using shared code from Library!</span>
              <p className={ styles.description }>Date formatted: {dateFormatted}</p>
            </div>
          </div>
        </div>
      </div>
    );
  }
}
```

Para acabar, podemos desplegar nuestro WebPart SPFx al App Catalog, instalarlo en un site de SharePoint, y añadirlo a una página.

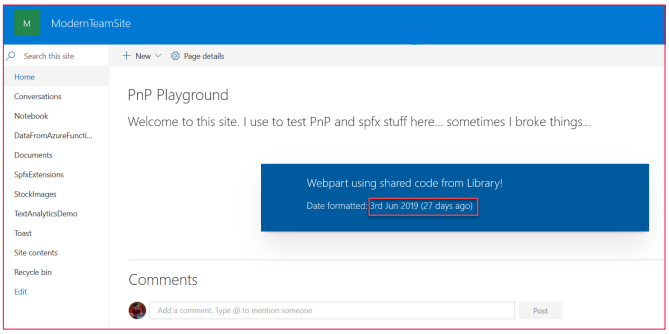


Imagen 4.- WebPart añadido a la página.

Lo interesante viene si añadimos una nueva instancia del WebPart a la página, y observamos las descargas, veremos como se descarga un .js que es el código de nuestra librería, y como se descarga una sola vez, y es reutilizado por las dos instancias del WebPart:

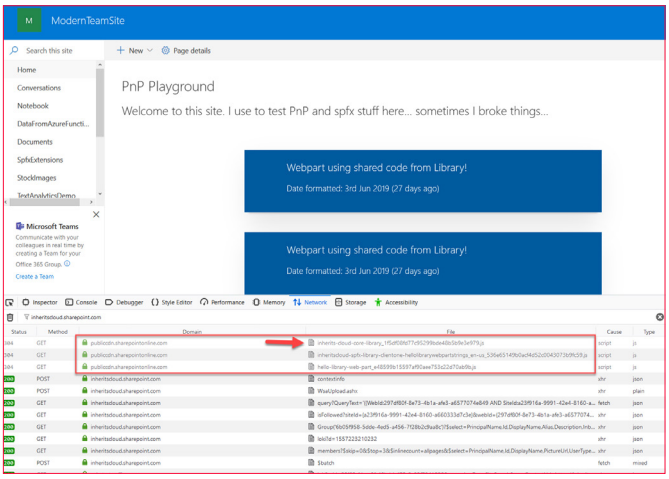


Imagen 5.- Análisis de las descargas en la página.

Nota: cabe destacar, que, si en algún momento necesitáis hacer un npm install en vuestro proyecto WebPart SPFx, la referencia a vuestra librería se va a perder, por lo que tendréis que volver a lanzar el comando npm link inheritscloud-spfx-library

Para acabar, tenéis el código disponible del ejemplo en mi repositorio de GitHub:

https://github.com/luismanez/sp-dev-fx-library-sample

Hasta el próximo número!

LUIS MAÑEZ
SharePoint/Cloud Solutions Architect en ClearPeople LTD
@luismanez
https://medium.com/inherits-cloud

21

10 consejos para mejorar la adopción y gobernanza de Microsoft Teams

Administrar una organización que incluya Microsoft Teams no es una tarea fácil. Asegurarse de controlar el crecimiento incontrolado, mantener segura la información interna y hacer que todo funcione de la manera más eficiente posible para los usuarios requiere algunas medidas importantes de control y gobernanza. En este artículo, comentaremos las 10 recomendaciones más importantes sobre el control y gobernanza de Microsoft Teams. ¡Comencemos sin más preámbulos!

1. ¿Quién debería poder crear Grupos de Office 365 y Microsoft Teams y cómo sería ese proceso?

Primero de todo, entendemos que lo que quiere es que sus empleados aprovechen y utilicen Grupos de Office 365 y Microsoft Teams. De hecho, cuantos más usuarios tengan el poder sobre su creación, mejor será su implementación. Sin embargo, la creación de espacios de colaboración necesita ser un proceso controlado, y la capacidad de hacerlo debe ser relegada a las personas apropiadas.

"antes de generar los gráficos es importante tener una idea de lo que queremos comparar."

Lo que vemos en otras empresas es que las personas apropiadas para aprobar suelen ser los propietarios del negocio, pero el equipo de TI también necesita aprobar el tipo, así como distintos aspectos y funcionalidades sobre qué estará disponible para distintos usuarios. Los propietarios de contenido y usuarios entienden sus procesos y la información contenida dentro de sus espacios de colaboración, pero no significa que se pueda confiar ciegamente en ellos para controlar y administrar las diferentes funciones y permisos de aplicaciones más complejas como Microsoft Teams.

Debe existir suficiente control y equilibrio para que el proceso de creación esté en línea con las políticas internas de la organización. Al mismo tiempo, el proceso también debe ser lo suficientemente rápido y simple para que los usuarios finales lo entiendan. Los tickets al departamento de soporte o TI y los correos electrónicos no son realmente la mejor forma de hacerlo, especialmente a gran escala.

2. ¿Con qué finalidad deberían los usuarios crear grupos y equipos?

Al principio, puede parecer una buena idea permitir que todos los usuarios sean capaces de crear estos espacios. Sin embargo, hemos visto organizaciones con 2.000 usuarios que tienen más de 2.500 Grupos de Office 365. Con el tiempo, ese tipo de libertad puede aumentar drásticamente el riesgo, los costos y lo que es peor confundir al usuario para encontrar lo que realmente necesita.

No es común que los usuarios puedan consumir recursos sin ninguna justificación. Es muy importante comunicarse con sus usuarios y establecer criterios sobre cómo y por qué es necesario crear espacios de colaboración.

3. ¿Qué tipo de usuarios son los más apropiados para decidir estos criterios regularmente?

Al igual que con tantas cosas en el mundo de las TI, la vida se vuelve más fácil cuando tenemos usuarios que se involucran y que impulsan una determinada propuesta dentro de la empresa y que nos pueden ayudar a construir políticas de gobernanza. Son los conocidos como "champions". A veces puede ser complicado en nuestro día a día tener que tratar con estos grupos, pero estas personas suelen apreciar estas políticas una vez que se dan cuenta de que pueden ganar respeto y autoridad al coordinarlas e implementarlas.

Es muy importante tener conversaciones con una amplia variedad de usuarios (o al menos establecer una buena relación con alguien que pueda). Cuanto mejor comprenda qué personas pueden ayudarlo a desarrollar los criterios para los diferentes procedimientos, más precisos, duraderos y exitosos serán.

4. ¿Cómo puede su organización gestionar el acceso y la propiedad?

¿Tiene su departamento los recursos adecuados para monitorear y hacer un seguimiento de los roles y el acceso a Grupos o Teams? ¿Entienden el poder que tienen los administradores (nuevos), Propietarios, Miembros e Invi-

tados en Microsoft Teams? Al considerar la pregunta n° 3 anterior, es importante saber que estas funciones pueden brindar a los usuarios finales una gran cantidad de permisos y la posibilidad de conexión a aplicaciones y almacenamientos externos. También debe tener en cuenta que solo se pueden añadir usuarios como miembros en un Team, no a otros grupos de Seguridad de directorio activo o Grupos de Office 365.

Mantener un registro de quién tiene acceso y permisos sobre las características que Grupos de Office 365 y Microsoft Teams proporcionan al usuario es fundamental para la administración de riesgos y recursos. Tanto los usuarios finales como los equipos de TI, legales y de seguridad deben asegurarse de que la información de la organización permanezca guardada en las aplicaciones y espacios de colaboración adecuados. Todos los departamentos deben colaborar para intentar evitar lo que se conoce como "shadow IT", es decir, el uso de aquellas aplicaciones no aprobadas por el equipo de IT y que se utilizan sin su conocimiento.

5. ¿Qué aplicaciones y servicios podrán crear los usuarios?

Ahora es posible controlar qué aplicaciones e integraciones se pueden agregar a nivel de Team en Microsoft Teams. Hay una amplia variedad de aplicaciones que pueden mejorar el uso y la conexión de las aplicaciones y plataformas en la nube, pero también es importante tener en cuenta la posibilidad de permitir ese acceso al "shadow IT", y el mayor riesgo que conlleva la habilitación de esos servicios.

"antes de generar los gráficos es importante tener una idea de lo que queremos comparar."

Un escenario bastante común se produce cuando el equipo de TI no habilita la posibilidad de compartir con usuarios externos. ¿Deberían los usuarios poder conectar sus Microsoft Teams a otras soluciones de almacenamiento en la nube? La planificación de políticas y un proceso que las aplique es muy importante cuando se trata de administrarlo.

6. ¿Cómo estructurará y aplicará las propiedades y las directivas de nomenclatura?

Controlar la forma en que las personas nombran Grupos de O365 y Teams y el poder aplicar las propiedades en función de cómo las personas utilizan Microsoft Teams y otros espacios de colaboración es clave cuando se trata de mantener el ciclo de vida de la información. No puede aplicar políticas y controlar el ciclo de vida de la información si no conoce por qué existe un determinado Team y qué tipo de información se encuentra almacenada en sus canales, conversaciones o ficheros.

Puede ser correcto dar a los usuarios mucha libertad a corto plazo, pero en el futuro nadie quiere terminar con un escenario donde cada usuario tenga uno o más Teams para cada persona de la empresa. Esto solo conducirá a situaciones difíciles donde discutir qué es lo que se puede guardar, archivar o borrar.

Cuanto más pueda automatizar las propiedades, las directivas de nomenclatura y la gestión del ciclo de vida, menos carga tendrá el equipo de TI para realizar un seguimiento y control de los objetos creados.

7. ¿Qué contenido se guardará, archivará o eliminará en cada Teams y después de cuánto tiempo?

Es importante entender que, aunque Microsoft Teams es un nuevo formato para la colaboración, la gente seguirá compartiendo archivos y documentos de un lado a otro, y esos archivos aún se almacenarán en los sitios de SharePoint que se crean automáticamente con cada Team. Esto significa que las preguntas clásicas sobre el ciclo de vida del contenido, la gestión de registros y la protección de datos / DLP aún deberán tenerse en cuenta.

Es importante tener un plan para asegurarse de que los documentos se etiqueten correctamente, que la seguridad y permisos, la taxonomía y la disposición a nivel de contenido se apliquen y, si es posible, se hagan de manera automática, y que las etiquetas y la clasificación reflejen con precisión la información dentro de los documentos y archivos.

Office 365 tiene algunas herramientas como el "Centro de Seguridad y Cumplimiento" y "SharePoint Records Management Center" que pueden abordar algunos de estos problemas, pero muchas organizaciones contactan con empresas especializadas para recibir consejos para automatizar y aplicar estas políticas en todos los sistemas de colaboración.

8. ¿Cómo reemplazarán y mejorarán los procesos de negocio los servicios e integraciones en Microsoft Teams?

Microsoft Teams pronto reemplazará por completo a Skype Empresarial. Los canales y la funcionalidad de chat persistente en Microsoft Teams hacen un gran trabajo al reducir las conversaciones por correo electrónico en Microsoft Exchange. A medida que los empleados colaboren más en Microsoft Teams y comprendan mejor su funcionamiento, comenzarán a aprovechar más funcionalidades como ChatBots, Conectores o IA, así como las integraciones con flujos

de trabajo o PowerPoint para aumentar la productividad.

A medida que avance el tiempo, cobrará mayor importancia contar con una estrategia de gestión del cambio para proporcionar a los usuarios una formación en los nuevos modelos de negocio que aumentarán la eficiencia reduciendo el riesgo. A medida que cambien las necesidades del negocio, y sea necesario añadir nuevas aplicaciones en intervalos cada vez más cortos, será fundamental contar con una estrategia para implementar nuevos procesos para mantenerse al día con el nuevo ritmo.

9. ¿Cómo formarán sus unidades de negocio y departamentos a los nuevos usuarios en estos procesos?

Es de vital importancia tener una estrategia, pero pensar en cómo se implementará la formación y la gestión del cambio es extremadamente importante en sí mismo. Desarrollar relaciones con los usuarios que puedan convertirse en expertos para nuevos proyectos e implementaciones y ayudarle a comprender las necesidades de los usuarios en su día a día, puede hacer la vida mucho más fácil a todos los involucrados.

10. ¿Cómo pueden los conectores, Microsoft Flow, Microsoft Forms y otras aplicaciones integradas reducir la carga de usuarios y de TI?

Comprender cómo los servicios existentes para usuarios se pueden integrar con Microsoft Teams y otros servicios de Office 365 es clave para una transición sin problemas.

Cuando se trata de formar a los usuarios e implementar procesos de negocios, otorgarles el control sobre cómo pueden interactuar de manera segura con la tecnología reduce la carga de los equipos de TI para comprender y adaptarse a cada flujo de trabajo empresarial.

"Los canales y la funcionalidad de chat persistente en Microsoft Teams hacen un gran trabajo"

Aplicaciones como Flow, Forms y Sway hacen que sea aún más fácil para los usuarios crear simples flujos de trabajo, distribuir información, y colaborar tanto interna como externamente. Sin embargo, estas aplicaciones solo son útiles si se implementan como parte de un plan. Cuando se incorporan con Microsoft Teams, son aún más fáciles de utilizar, con una experiencia sencilla y simple para los usuarios.

Una planificación de manera efectiva puede reducir las micro-implementaciones, o, dicho de otra manera, la necesidad de compras de pequeños servicios en la empresa, lo cual puede ahorrar tiempo, esfuerzo y dinero en la organización.

Con todas las herramientas que nos proporciona Office 365, la implementación de controles de seguridad, permisos y gobernanza es el colofón para facilitar una estrategia de adopción y gobernanza escalable y administrada con éxito.

ROBERTO VÁZQUEZ DELGADO
EMEA Senior Solution Architect at AvePoint

24

Entrevista David Rivera

Soy David Rivera Nací en Sabadell (Barcelona), aunque en el año 1982 junto a mis padres y hermanos emigramos a Venezuela allí tuve mi primer ordenador y realice los estudios universitarios y empezó mi vida profesional. En el año 1992 regrese a España y continúe hasta el día de hoy.

Me he desempeñado como analista de sistemas y arquitecto de Soluciones Microsoft, he liderado proyectos en Venezuela España, Argentina, Estados Unidos, Francia, Singapur, Andorra, México.

He sido galardonado como MVP en Microsoft Cloud and Datacenter Management 2014-2019.

Actualmente me encuentro en Clevertask liderando proyectos con las Soluciones de Microsoft.

¿Por qué y cómo empezaste en el mundo de la tecnología?

En el 1985, empecé con un Sinclair ZX y Basic, hice mis primeros pasos en el mundo de la informática, empecé mis estudios de Ing de Sistemas, donde pasaba horas y días metido en el laboratorio, trabajando con un pc Epson Equity con MS-DOS Gw-Basic, Cobol Dbase III+ y seguí con Clipper, Pascal, C++ etc. Estando en la universidad conseguí mi primer trabajo unido a la informática, trabajaba dando clases de MS-DOS, Printmaster etc., trabajaba de día y estudiaba de tarde/noche.

Me ofrecieron ser el responsable del departamento de ensamblaje de ordenadores y empecé a montar Clones, a montar mis primeras redes y empecé trabajar con diferentes sistemas operativos Xenix, Dr-Dos, Novell NetWare y los primeros pasos de Windows.

En el año 1992 junto a mi familia regreso a España y continúo trabajando de informático como técnico de sistemas, trabajando con redes Novell, MS-DOS, Windows 3,11 Windows NT, 95 hasta el día de hoy.

Las soluciones de Microsoft me acompañado toda mi vida profesional desde mis primeros pasos hasta el día de hoy he trabajado con una gran variedad de las soluciones de Microsoft Sistemas Operativos, Isa Server, TMG, SharePoint, SQL Server, System Center, Office 365, y los últimos años estoy muy centrado con Azure y todo lo que implica, cada día que pasa estoy más unido a las soluciones de Microsoft.

¿Cuáles son tus principales actividades tecnológicas hoy en día?

Actualmente soy arquitecto de soluciones Microsoft en Clevertask, estoy implicado en varios proyectos en los cuales están implicados diferentes soluciones de Microsoft (System Center Configuration Manager, Intune, Windos Server (DirectAccess, ADFS, Directorio Activo etc), Azure (Azure AD, Azure Security Center, Azure monitor, etc.), Office 365) etc.

¿Cuáles son tus principales actividades NO tecnológicas hoy en día?

Soy un Geek, mis principales actividades son tecnológicas, pero de vez en cuando para desconectarme un poco me gusta salir a pasear, caminar, leer libros, ver películas, series y escuchar música, realizar salidas familiares para conocer lugares.

¿Cuáles son tus hobbies?

Yo tengo la suerte, que trabajo con lo que me gusta, y que también es mi hobby, como hobby soy miembro Cofundador de la comunidad DTC2MOBILITY comunidad que anteriormente se llamaba Comunidad de System Center, como miembro de la comunidad nuestra finalidad es dar una mirada de 360º a las tecnologías de Microsoft, partiendo de la gestión del DataCenter con System Center (SCSM, SCOM, SCVMM, SCDPM, SCORCH) y pasando por las soluciones de movilidad con SCCM e Intune para llegar hasta la nube de la mano de Azure, Microsoft 365.

¿Cuál es tu visión de futuro en la tecnología de acá a los próximos años?

Actualmente estamos en un momento muy interesante tanto para los que llevamos tiempo trabajando con la tecnología o como los que están empezando, ya que se nos está abriendo un gran abanico de nuevas tecnologías, conocimientos y nuevas formas de trabajar. Para los que somos ITPros se nos presenta los DevOps o los que trabajamos con SCCM vemos que por la evolución de la tecnología, o por los requerimientos de la tecnología, clientes y nuestras infraestructuras necesitamos de nuevas características que pasando al Co-management SCCM con Intune nos da una nueva forma de administrar nuestros equipos o desplegar los sistemas operativos, aplicaciones etc.

Azure que nos da un gran abanico de servicios, que a su vez nos da una cantidad de herramientas que nos ayuda en nuestro trabajo, también me pongo a pensar que nos viene o lo que estamos empezando a ver los IOT vemos los dispositivos conectados (Autos, electrodomésticos, etc.) o la inteligencia artificial que se empezando aplicar a todo nuestro alrededor.

La realidad Mixta con las HoloLens vemos lo que se puede hacer he visto Operaciones quirúrgicas, educación con mapas virtuales interactivos, también para la industria de la construcción que nos permitirá ver nuestro piso, aunque lo estén construyendo o pero Yo me pongo a pensar en el futuro hacia donde vamos Azure, Realidad Mixta, inteligencia artificial, la sociedad más conectada, ciudades inteligentes, automóviles con conducción automática o mejor dicho sin conductor, si ya los robots están realizando operaciones que será en el futuro.

Estaría escribiendo mucho más del futuro de la tecnología y del gran abanico de nuevas tecnologías, conocimientos que se abre.

DAVID RIVERA
MVP Microsoft Cloud and Datacenter Management
@darifer66

26

SharePoint y Azure: Crear servicios REST para SharePoint con Azure API Apps

Cuando se desarrolla software es muy importante considerar la posibilidad de exponer su funcionalidad al mundo exterior, de tal forma que otros programas puedan acceder a la información y procesamiento de datos ofrecidos por el software. Las rutinas, protocolos de comunicación y herramientas para exponer la funcionalidad se conocen como APIs, Application Programming Interfaces. Aunque existen varios Frameworks para definir APIs, el mundo moderno se ha prácticamente unificado en el uso de APIs REST (REpresentational State Transfer). REST es popular debido a su simplicidad y el hecho de que se basa en los sistemas y características existentes del protocolo HTTP de Internet con el fin de lograr sus objetivos, a diferencia de la creación de nuevos estándares, Frameworks y tecnologías. Para encontrar más información sobre REST y SharePoint, revise los artículos publicados anteriormente en CompartiMOSS número 22 "REST, WebAPI 2 y SharePoint 2013 - Introducción" (http://www.compartimoss.com/revistas/numero-22/rest-webapi-2-sharepoint-2013-introduccion) y CompartiMOSS número 23 "REST, WebAPI 2 y SharePoint 2013 – WebAPI y Odata" (http://www.compartimoss.com/revistas/numero-23/rest-webapi-2-y-sharepoint-2013-webapi-y-odata).

El servicio de Azure API Apps ofrece toda la infraestructura de PaaS para alojar APIs de REST, de tal forma que solamente es necesario pensar en el diseño y desarrollo del código y no en la creación y mantenimiento de hardware. El servicio ofrece, fuera del hosting, soporte de autenticación por medio de Azure Active Directory y OAuth, acepta APIs escritos utilizando .NET, PHP, Node.js, Java o Python, proporciona soporte para CORS (Cross-Origin Resource Sharing), integración con Azure API Management y Azure Logic Apps y uso de Swagger para generar cross-platform SDKs (Software Development Kits).

"es muy importante considerar la posibilidad de exponer su funcionalidad al mundo exterior"

Aunque Office 365 y, especialmente, SharePoint Online disponen de un set completo de APIs REST que cubre prácticamente toda la funcionalidad base, es necesario frecuentemente crear APIs personalizados para exponer funcionalidad especializada que no se encuentra por defecto en el sistema, para dar acceso a procesos específicos para una empresa, o para permitir la ejecución automatizada de métodos dentro de SharePoint inicializados por otros sistemas (integración con sistemas externos).

Creación del Servicio de Azure API Apps

Para poder utilizar el servicio es necesario primero crear una instancia en su suscripción de Azure. Utilice credenciales de administrador en el portal de Azure (https://portal.azure.com) y cree o reutilice un Grupo de Recursos. Dentro del Grupo, agregue un "App Service Plan" indicando su nombre, sistema operativo (Windows), localización y nivel de precios, el que determina las características y costos de los recursos asociados a la aplicación. Existen múltiples planes para escoger, incluyendo uno gratis que ofrece infraestructura compartida, 1 GB de memoria y 60 minutos de tiempo de computación al día sin costos agregados, ideal para utilizar en el proceso de desarrollo.

Cuando el plan de servicios ha terminado de ser creado, agregue un servicio del tipo "API App" indicando su nombre y el plan de servicios creado anteriormente. Opcionalmente se le puede acoplar un servicio de "Application Insights" para colectar datos de monitoreo extendidos sobre el uso del API App; el servicio de API App recopila alguna información básica por defecto, y la muestra en la ventana de manejo del servicio:

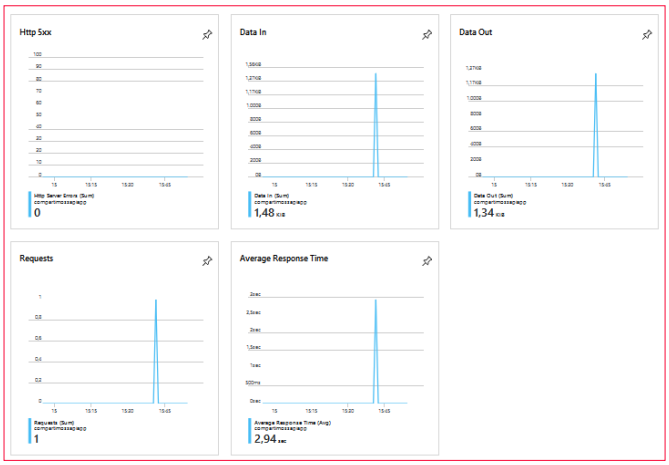


Imagen 1.- Estadísticas de uso del Servicio de API Apps.

Programación de una API REST

Para programar una API REST utilice Visual Studio (versión 2019 utilizada en el siguiente ejemplo). Abra Visual Studio

y cree una nueva solución del tipo “ASP.NET Web Application (.NET Framework)” que se puede encontrar filtrando las plantillas de proyectos por “C#”, “Windows” y “Web”:

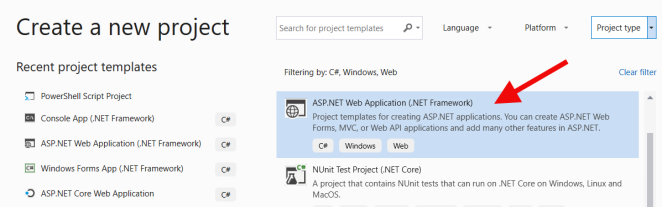


Imagen 2.- Creación del proyecto en Visual Studio.

Asígnele un nombre y localización a la Solución y luego seleccione “Web API” en el tipo de aplicaciones a elegir. La plantilla de Visual Studio crea una aplicación completamente funcional que se puede compilar y hacer funcionar sin cambiar nada de código.

En el siguiente ejemplo se va a crear un API que permite clonar Listas de SharePoint. Esta es funcionalidad que no existe por defecto en SharePoint: copiar todos los elementos de una Lista de origen a un destino. El ejemplo es básico, simplemente copia elementos de una Lista a otra sin hacer ningún tipo de chequeo de validación y no copia historial de versiones ni autorización. La cuenta que ejecuta el código debe tener permisos de leer/escribir en ambas Listas, y las Listas deben tener la misma estructura (y nombres) de campos y estar localizadas en la misma Colección de Sitios y sitio Web raíz de la Colección. Como se ha dicho, el ejemplo es muy básico y no dispone de código para atrapar errores, pero se puede tomar como punto de partida y extenderlo para crear software más robusto.

Una vez creada la Solución, añada el NuGet “SharePointPnPCoreOnline” para obtener los componentes necesarios para programar contra SharePoint Online, y el NuGet “Swashbuckle” que agrega los componentes para Swagger (más adelante se explica su uso).

En el proyecto de Visual Studio abra el panel de “Solution Explorer”, expanda el nodo “Controllers” y agregue un nuevo controlador del tipo “Web API 2 Controller - Empty” llamado “ClonListController.cs”. En la ventana de edición del nuevo controlador comience definiendo directivas a “using Microsoft.SharePoint.Client;” y “using System.Security;” al principio del código. Luego reemplace todo el código fuente dentro de la clase “ClonListController” que fue creado por defecto, con el siguiente código:

```
[HttpGet]
[System.Web.Http.Route(“Api/CopyItems”)]
public void CopyItems(string SourceListName, string DestinationListName)
{
  ClientContext spContext = null;

  string myUserName = “usuario@dominio.onmicrosoft.com”;
  string myPassword = “ClaveMuySegura”;
    string BaseUrl = “https://dominio.sharepoint.com/sites/Test_AzureApiApps”;

  SecureString securePassword = new SecureString();
  foreach (char oneChar in myPassword) securePassword.AppendChar(oneChar);
    SharePointOnlineCredentials myCredentials = new SharePointOnlineCredentials(myUserName, securePassword);

  spContext = new ClientContext(BaseUrl);
  spContext.Credentials = myCredentials;

  Web myWeb = spContext.Site.RootWeb;
  List sourceList = myWeb.Lists.GetByTitle(SourceListName);
    List destList = myWeb.Lists.GetByTitle(DestinationListName);
  ListItemCollection sourceItems = sourceList.GetItems(CamlQuery.CreateAllItemsQuery());
  FieldCollection sourceFields = sourceList.Fields;
  spContext.Load(sourceList);
  spContext.Load(destList);
  spContext.Load(sourceItems);
  spContext.Load(sourceFields);
  spContext.ExecuteQuery();

  foreach (ListItem oneSourceItem in sourceItems)
  {
    ListItemCreationInformation itemInfo = new ListItemCreationInformation();
    ListItem oneDestItem = destList.AddItem(itemInfo);
    foreach (Field oneSourceField in sourceFields)
    {
      if (!oneSourceField.ReadOnlyField && !oneSourceField.Hidden &&
          oneSourceField.InternalName != “Attachments” && oneSourceField.InternalName != “ContentType”)
      {
        oneDestItem[oneSourceField.InternalName] = oneSourceItem[oneSourceField.InternalName];
        oneDestItem.Update();
      }
    }
  }

  spContext.ExecuteQuery();
}
```

El método “CopyItems” comienza definiendo variables conteniendo el usuario, clave y URL de la Colección de Sitios en SharePoint donde están las dos Listas. Note que exponer la cuenta del usuario de esta forma NO es recomendable, y se hace aquí solo por ser código de prueba. Estos datos se utilizan para logarse en la Colección de Sitios de SharePoint y obtener el contexto utilizando el Modelo de Objetos de Cliente (CSOM). En una sola llamada “ExecuteQuery” se obtienen objetos conteniendo la Lista fuente, la de destino, los elementos en la Lista fuente y sus campos.

“el siguiente ejemplo se va a crear un API que permite clonar Listas de SharePoint”

Inmediatamente después se ejecuta un bucle para obtener los elementos en la Lista fuente, se inicializa un objeto del tipo “ListItemCreationInformation” para crear un nuevo Item en la Lista de destino, y por medio de otro bucle se obtienen los valores para cada campo del elemento en la Lista fuente y se le asignan al Item de la Lista de destino. Finalmente, una llamada “ExecuteQuery” serializa toda la información en SharePoint.

Una vez compilado el proyecto para comprobar que no tiene errores, es necesario publicarlo en el servicio de Azure. Seleccione el proyecto en el Explorador de Soluciones de Visual Studio, abra el menú contextual (botón derecho del

ratón) y haga clic sobre “Publish”. En la ventana que abre automáticamente selección “App Service” - “Select Existing” y utilice el botón de “Publish”. La nueva ventana permite seleccionar una suscripción de Azure o configurar una nueva si es necesario, y muestra una lista con los Servicios de API Apps disponibles:

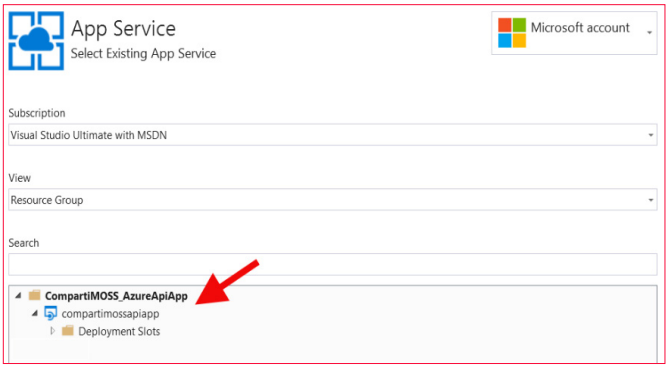


Imagen 3.- Publicando el API en Azure.

Manejo y uso del API

Una vez publicado el proyecto en el Servicio de Azure, se abre automáticamente una ventana de navegador en el sitio de la App configurada. Este es el sitio que se ha programado desde la plantilla de Visual Studio, que incluye una página “Home” con contenido, y una página “API” que muestra los métodos definidos en los controladores del código:

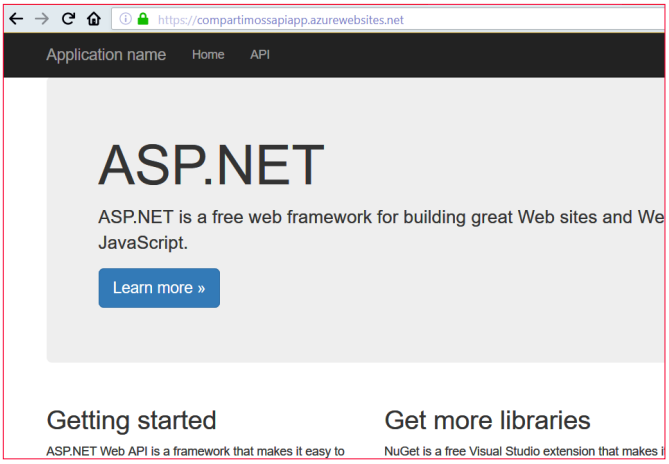


Imagen 4.- Página Home de la App.

“el proyecto para comprobar que no tiene errores, es necesario publicarlo en el servicio de Azure”

Tanto el contenido como la organización y menús del sitio se pueden modificar fácilmente desde el código, o eliminar todo si es necesario. La sección de “API” muestra los APIs definidos en los controladores del código; note que aun existe un método “Values” que es el controlador creado por defecto por la plantilla de Visual Studio:

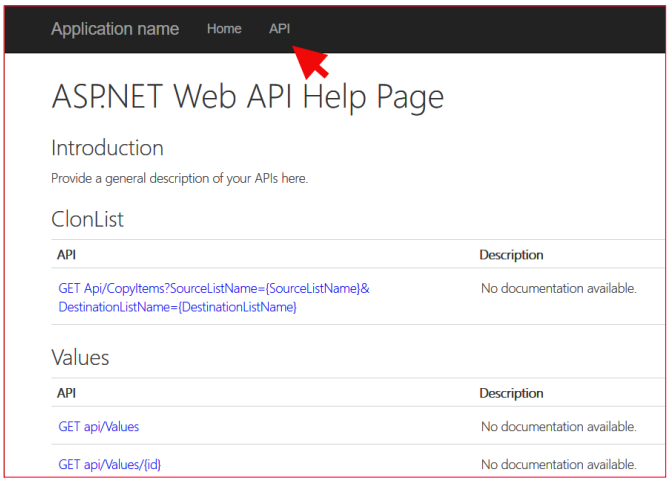


Imagen 5.- Página de definición de los APIs.

Haciendo clic sobre el vínculo del API “ClonList” se abre la página que muestra los parámetros de entrada, la respuesta que se puede esperar, y el URL que ha utilizar para llamar el método REST:

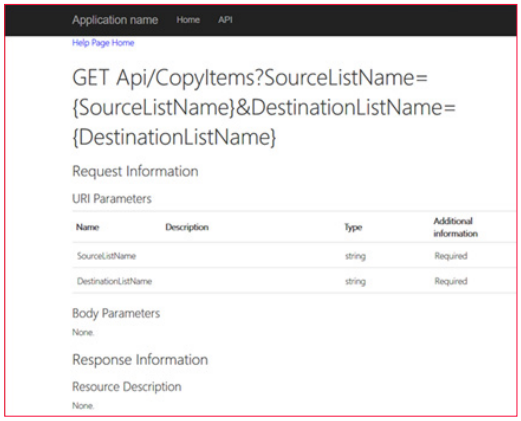


Imagen 6.- Página de definición de uno de los APIs.

Una mejor forma para ver la definición de los APIs, sus parámetros de llamada y las respuestas, es utilizando Swagger. Swagger es un framework de software de código abierto que ayuda a los desarrolladores a diseñar, compilar, documentar y consumir servicios web REST. El Framework proporciona una interfaz de usuario, un conjunto de herramientas para crear documentación automatizada, generación de código y generación de casos de prueba. Cuando se inició el proyecto de Visual Studio, se agregó el NuGet “Swashbuckle” que instala todo el Framework en la Solución y permite utilizar Swagger sin necesidad de más configuraciones. Para ver la interfaz de usuario de Swagger, agréguele “\swagger” al URL del Servicio utilizado anteriormente, de tal forma que debe quedar en la forma “https://dominio.azurewebsites.net/swagger”:

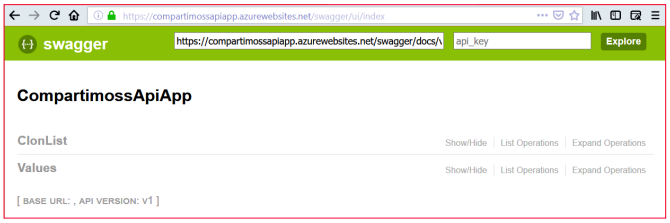


Imagen 7.- Página de Swagger con la definición de los APIs.

Haciendo clic sobre el nombre del API “ClonList” se puede ver su definición, métodos, parámetros e iniciar una llamada REST utilizando el botón “Try it out!”:

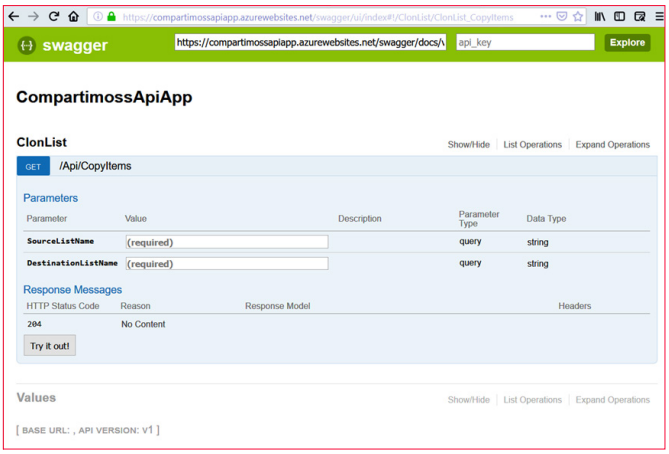


Imagen 8.- Página de Swagger con la definición de uno de los APIs.

Por supuesto que el método se puede testear también desde cualquier navegador o herramienta utilizando el URL (porque es un API REST). En el ejemplo, utilice el URL:

http://compartimossapiapp.azurewebsites.net/Api/CopyItems?SourceListName=TestListSource&DestinationListName=TestListDestination

“es necesario exponer funcionalidad personalizada de SharePoint por medio de un API.”

Conclusiones

Con frecuencia es necesario exponer funcionalidad personalizada de SharePoint por medio de un API. En el estado actual de tecnologías Web, la mejor forma para hacerlo es por medio de un API REST. El servicio de Azure API Apps permite alojar y hacer disponibles este tipo de APIs, sin necesidad de tener que manejar el hardware e infraestructura que lo hacen funcionar. Además, Visual Studio ofrece una plantilla para crear y programar APIs de forma rápida y sencilla.

GUSTAVO VELEZ
MVP Office Apps & Services
gustavo@gavd.net
http://www.gavd.net

30

Como diseñar Apps Multitenant en Azure

Glosario de términos

Antes de empezar a desglosar todos los aspectos que se necesitan para implementar una aplicación multitenant, nos centraremos en explicar un glosario de términos muy importante para poder entender este tipo de arquitectura.

- Tenant => Se puede definir como un grupo de usuarios que comparten acceso común con privilegios específicos a la instancia de software. Es decir, esta única instancia de software e infraestructura únicamente atiende a un único cliente. Este tipo de desarrollo está vinculado al desarrollo tradicional (arquitecturas monolíticas, on-premise, poco aprovechamiento de los recursos y muy personalizado).
- Multitenant => Principio de arquitectura de software en la cual una sola instancia de la aplicación se ejecuta en el servidor, pero sirviendo a múltiples clientes u organizaciones. Una sola base de código se ejecuta en todos sus clientes. Las aplicaciones multitenant no son más que una evolución en el tiempo en la lucha de las empresas por el ahorro de costes:
  - (1960) Timesharing => Las empresas alquilaban CPU entre varias para procesar los datos de sus clientes para reducir costes.
  - (1975) Hosted Application => Los proveedores de servicios de aplicaciones tradicionales (ASP) alojaron aplicaciones (en ese momento) en nombre de sus clientes.
  - (1993) Web Application => Aplicaciones web populares orientadas al consumidor (como Hotmail) desarrolladas con una única instancia de aplicación que sirve a todos los clientes

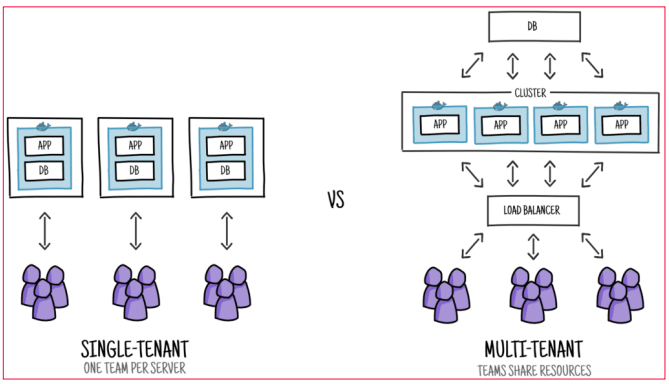


Imagen 1.- Diferencia Tenant vs MultiTenant.

¿Qué consideraciones debemos de tener para hacer una aplicación multitenant?

El principal motivo que nos planteamos para crear una aplicación multitenant es porque nuestra aplicación va a ser consumida por varios clientes. Si por el contrario nuestra aplicación es únicamente para un cliente con su propia lógica no tiene ningún sentido plantearse este tipo de arquitectura. El motivo: No se aprovechan ninguna de las ventajas que nos ofrecen el multitenant.

Ventajas:

- Ahorro de costes, al tener los costes distribuidos entre varios clientes naturalmente el precio se disminuye por cliente. Al hacer uso del cloud y su modelo de pago (siempre que se pueda en los servicios necesarios) todavía es una ventaja todavía más importante. Pensar en el caso de tuviéramos que comprar un servidor y ¿qué tamaño tenemos que poner? Con el cloud todo esto es configurable y escalable a nuestras necesidades con lo que es posible crear pequeños MVP (Minimum Viable Product) lanzarlos al mercado ver si resultan y en el caso de fracaso la inversión realizada será mucha menor.
- Actualización. Al tener una única aplicación ejecutando, actualizando una actualizamos a todos nuestros clientes. Pensar en SharePoint Online cada vez que lanzan una actualización si tuvieran que actualizar todas las granjas de forma manual no habría tiempo casi material para hacerla. El caso de Office 365/SharePoint Online dado la gran cantidad de clientes que tienen e incluso infraestructura que se utiliza tienen un modelo de despliegue un tanto diferente, pero en este caso es que nuestra aplicación ha tenido muchísimo éxito.
- Seguridad de la información de cada cliente.
- Optimización en el uso de los recursos de la infraestructura.

Desventajas:

- Menos personalización, a la hora de hacer un producto si cada cliente tiene una versión de este, sería una verdadera locura. Aun así, es posible, sin embargo, por norma general las aplicaciones multitenant cada cliente tiene poca personalización respecto a lo que es el

producto original.

- Single point of failure => Fallos más dramáticos, cuando se produce un fallo en el sistema este es más sonoro debido a que no solamente se entera un único cliente, sino que todos tus clientes es posible que no dispongan del servicio. Pensar los fallos en Azure, SharePoint Online, Amazon o Netflix.

Una vez ya tenemos claro en que circunstancias se desarrollan las aplicaciones multitenant vamos a ver que debemos tener en cuenta para poder hacer una aplicación multitenant y optimizar al máximo las ventajas que nos ofrece Azure. Desde el punto de vista de arquitectura deberemos tener en cuenta aspectos como la autenticación, el almacenamiento de datos o el lugar donde se va a ejecutar nuestra aplicación. Centrémonos en cada una de esas partes.

"cuando hablamos de desarrollar aplicaciones para la nube, podemos cometer el error de no tener en cuenta."

Autenticación

Dentro de Azure, esta más que extendido el uso del servicio de Azure Active Directory en casi cualquier aplicación que esta alojada en Office 365. En el caso de que nuestra aplicación pueda ser una aplicación dentro del ecosistema Microsoft y/o Office 365 el hacer uso de este servicio puede ser un punto a favor de la adopción de nuestro desarrollo dentro de nuestros clientes. En el caso de que nuestra App necesite una personalización y no este dentro de Office 365 lo más normal es optar por un modelo de autenticación Ad-hoc siguiendo los estándares que marca OpenId.

¿Como podemos hacer usar nuestra Aplicación el servicio de Azure Active Directory Multitenant?

1.– En el registro de nuestra aplicación indicar que la aplicación es multientant, en ese momento dicha aplicación se va a poder utilizar en cualquier tenant de Azure que exista (siempre que el administrador de dicho tenant consienta instalarla naturalmente).

2.– Desde el punto de vista del FrontEnt o de la capa Vista tendremos que modificar los endpoint en los que solicitamos el acceso. Por regla general estamos acostumbramos a autenticarnos contra una url de la siguiente forma:

```
https://login.microsoftonline.com/contoso.onmicrosoft.com
```

Para que nuestra aplicación sea multitenant tendremos que modificar "contoso.onmicrosoft.com" por common. Un ejemplo de esto haciendo uso de Adal.JS sería el siguiente código:

```
export const adalConfig: AdalConfig = {
 tenant: 'common',
 clientId: process.env.REACT_APP_CLIENTID,
 endpoints: {
  api: process.env.REACT_APP_CLIENTID,
  graphApi: 'https://graph.microsoft.com/'
 },
 cacheLocation: 'localStorage',
};
  runWithAdal(authContext, () => {
    renderApp();
  }, false);
```

3.– Si nuestra aplicación autentica la parte servidora también haciendo uso del Azure Active Directory tendríamos que modificar los valores de este para que de esta forma se autentique independientemente del inquilino que se este conectando a nuestra aplicación. Un ejemplo de como hacerlo en .NET Core es implementar un middleware como el siguiente:

```
public static AuthenticationBuilder AddAzureAdBearerMulti-
Tenant(this AuthenticationBuilder builder, Action<AzureAdOp-
tions> configureOptions)
    {
      builder.Services.Configure(configureOptions);
        builder.Services.AddSingleton<IConfigureOptions<Jwt-
BearerOptions>, ConfigureAzureOptions>();
      builder.AddJwtBearer(j =>
      {
        j.TokenValidationParameters = new TokenValidationPa-
rameters()
        {
          ValidateIssuer = false
        };
      });
      return builder;
    }
```

4.– Por último, el Administrador del tenant que va a utilizar la App debe de instalarlo en su "tenant". Para ello hay una url de consentimiento que un usuario administrador debe de aceptar para poder hacer uso de nuestra aplicación. El flujo para dicha aprobación es como el que se muestra en el siguiente dibujo. En el caso de que un administrador, ya no quiera que la aplicación se use más en su tenant puede revocarlo desde su Portal de Azure quitando los permisos.

La Base de datos: SQL Azure/Cosmos DB

Una vez ya tenemos clara que autenticación va a tener nuestra aplicación, el siguiente eslabón que se ha de tratar es el almacenamiento de los datos. Dentro de los servicios que nos proporciona Azure podemos elegir tanto base de datos relacional como SQL Azure o bien base de datos no relacional como Cosmos DB. La elección de una u otra no depende de que nuestra aplicación sea multitenant, pero sí que debemos tener claro qué tipo de arquitectura vamos a tener para que la optimizar los accesos a datos.

¿De que depende el patrón que va a utilizar nuestra apli-

cación?

- Dependiendo de la escalabilidad: Número de tenants que consumen nuestra aplicación, almacenamiento de cada uno de ellos, carga de trabajo.
- Aislamiento de tenants: aislamiento de los datos y rendimiento (si la carga de trabajo de un tenant afecta a otros). No todos los clientes tienen el mismo volumen de datos. También es posible que nuestros clientes tengan flanjas horarias diferentes, por lo que se podemos aprovechar los servicios que nos proporciona el cloud,
- Costo por tenant: dependiendo de que patrón escojamos los costos de la base de datos se pueden incrementar, de cara a la viabilidad de nuestra aplicación es algo que debemos de tener muy presente.
- Complejidad de desarrollo: Cambios en el esquema. Cambios en las consultas (requeridos por el patrón).
- Complejidad operativa: Supervisión y administración del rendimiento., administración del esquema. restauración de un tenant, recuperación ante desastres.
- Capacidad de personalización: facilidad para admitir las personalizaciones del esquema que son específicas de un tenant en particular.

"de los servicios que nos proporciona Azure podemos elegir tanto base de datos relacional como SQL Azure"

Posibles patrones:

- Multi-tenant app with database-per-tenant.

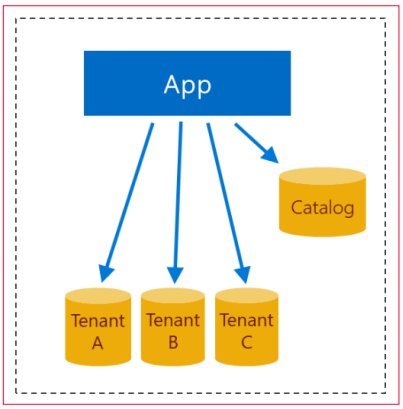


Imagen 2.- Patrón de App multi-tenant con BD Exclusiva.

En este patrón cada tenant tiene su propia base de datos en exclusiva para él. Su principal beneficio es el aislamiento de los datos entre cada uno de los tenant, así como los procesos que se ejecutan en cada uno de ellos. Por otro lado, su principal desventaja es que cada base de datos tiene un coste muy elevado.

- Multi-tenant app with database-per-tenant. (ELASTIC POOL)

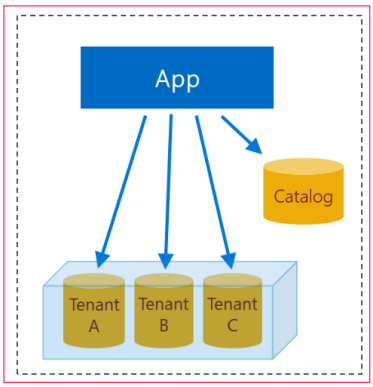


Imagen 3.- Patrón de App multi-tenant con BD Elastic.

Este patrón es una variante, del primer patrón, pero con la ventaja de que el utilizar una base de datos "Elastic" es un único servidor de base de datos lo cual puede proporcionar un ahorro de costos, debido a que se pueden compartir alguno de los recursos.

- Multi-tenant app with multi-tenant databases.

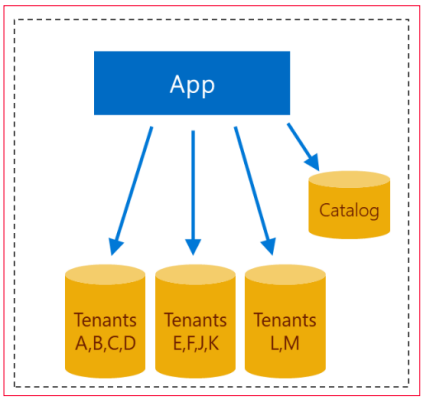


Imagen 4.- Patrón de App multi-tenant con BDs multi-tenant.

Utilizando este patrón deberemos de tener una o varias columnas de identificadores del tenant dentro del esquema de nuestra base de datos. Con esta opción se sacrifica el aislamiento de los datos ya que residen físicamente en la misma base de datos. También se comparte recursos de proceso y almacenamiento entre todos los tenants. Todo ello implica que es la de menos coste.

- Multi-tenant app with sharded multi-tenant databases.

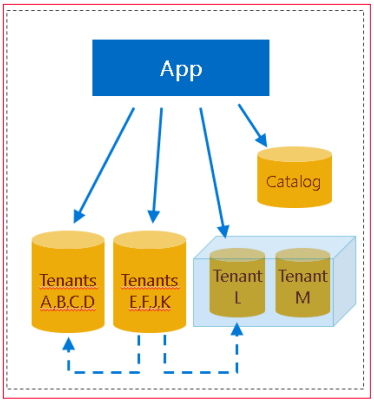


Imagen 5.- Patrón de App multi-tenant con BDs multi-tenant compartidas

Es una variante del modelo anterior, pero los datos de un tenant pueden estar distribuidos a través de varias bases de datos o particiones. El escalado es casi ilimitado, bases de datos más pequeñas que se administran con más facilidad. Pueden usarse Elastic pools. Aquí es clave el identificador del inquilino.

¿Como podemos implementar estos patrones en Entity Framework Core?

En primer lugar, nos crearemos un TenantProvider cuya finalidad será obtener el tenant id en este caso vamos a a obtener el tenantID de la petición http que realice las peticiones. De esta forma podemos filtrar dependiendo de que tenant nos llama. Su implementación seria como la siguiente:

```
public class DatabaseTenantProvider : ITenantProvider
  {
    private string tenantId;

      public DatabaseTenantProvider(TenantsDbContext context, IHttpContextAccessor accessor)
    {
       var tenantIdRequest = accessor.HttpContext.User.FindFirst(c => c.Type == “http://schemas.microsoft.com/identity/claims/tenantid”)?.Value;

      context.Database.EnsureCreated();


      if (!string.IsNullOrWhiteSpace(tenantIdRequest))
      {
        tenantId = tenantIdRequest;
      }
    }

    public string GetTenantId()
    {
      return tenantId;
    }
```

Una vez ya tenemos el patrón que vamos a implementar vamos a indicarle a nuestro contexto de base de datos que cada vez que se realice alguna consulta a la base de datos esta se filtre por el tenant que realiza la consulta. Para ello bastaría con el siguiente ejemplo:

```
public class PastryDbContext : DbContext
  {
    public DbSet<Pastry> Pastry { get; set; }

    private readonly ITenantProvider tenantProvider;

    public PastryDbContext(DbContextOptions<PastryDbContext> options,
                ITenantProvider tenantProvider) : base(options)
    {
      this.tenantProvider = tenantProvider;
    }

      protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
      base.OnModelCreating(modelBuilder);
       modelBuilder.Entity<Pastry>().HasQueryFilter(p => p.TenantId == tenantProvider.GetTenantId());
    }
```

Este seria una implementación de uno de los patrones dependiendo de cada caso sería posible implementar el patrón bien utilizando EF Core.

Alojamiento de la Aplicación

Por regla general nuestra aplicación la tendremos alojada en una WebAPP, que se pueda escalar dependiendo de las necesidades. Esta opción sería para aplicaciones sencillas que no requieran de una alta escalabilidad. En Apps en la que la escalabilidad sea un elemento clave habría que optar por el uso de Kubernetes o cualquier orquestador de Microservicios, de forma que nuestra aplicación pueda escalar en los servicios que más lo requieran y de esta forma dar un servicio óptimo y adaptado a los requerimientos que demanda el negocio. En este articulo solamente es un overview para poder pensar sobre todo los artefactos o elementos que necesitamos para crear una App multitenant.

Conclusiones

El crear Aplicaciones Multitenant no es algo tan trivial como activar una opción y nuestra aplicación ya la pueden utilizar todos nuestros clientes de una forma óptima y adaptada el cloud. Hay que pensar en los aspectos que he indicado en el artículo y en muchos que no hemos abordado como por ejemplo: Ley de protección de datos, Geolocalización, Performance, Actualización, etc…

Azure no es únicamente el Cloud donde los clientes pueden empezar migrar su infraestructura, o consumir servicios, Azure también es la base de muchas de las aplicaciones empresariales/ apps/ startups que están en el mercado. Todo esto es debido a la gran cantidad de servicios por lo que desde el punto de vista tecnológico y de negocio es bastante importante el conocer al máximo la potencia que nos puede ofrecer Azure y de esta forma poder implementar aplicaciones más complejas, mantenibles y escalables.

ADRIÁN DIAZ CERVERA
Architect Software Lead at Encamina
MVP Office Development
http://blogs.encamina.com/desarrollandosobresharepoint
adiaz@encamina.com
@AdrianDiaz81

34

Manejando Microsoft Teams con PowerShell

Tras mucho tiempo en beta y preview, recientemente se ha liberado la versión GA del modulo PowerShell para Microsoft Teams

Instalando el módulo PowerShell para Microsoft Teams

Lo primero que tenemos que hacer es instalar el módulo PowerShell para Teams, para ello abre Windows PowerShell como administrador y ejecuta:

```
Install-Module MicrosoftTeams
```

Una vez instalado, podemos comprobarlo ejecutando:

```
Get-Module MicrosoftTeams -ListAvailable
```

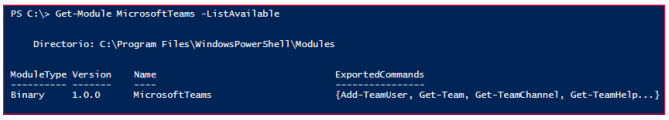


Imagen 1.- Instalación correcta del módulo de PowerShell para Teams.

Para ver los comandos disponibles, podemos ejecutar:

```
Get-Command -Module MicrosoftTeams
```

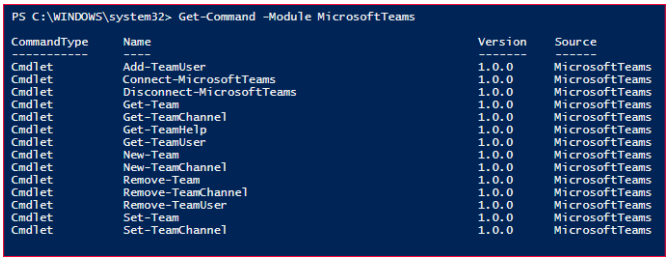


Imagen 2.- Listado de los comandos disponibles.

O bien mostrar la ayuda con el comando Get-TeamHelp:

"Lo primero que tenemos que hacer es instalar el módulo PowerShell para Teams"

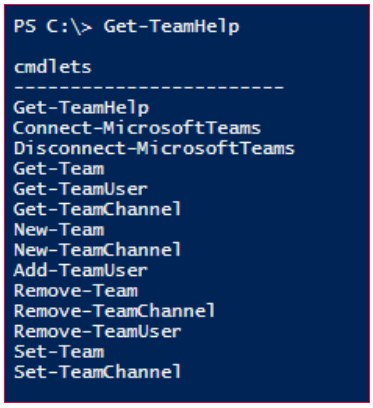


Imagen 3.- Cmdlet de ayuda para obtener los cmdlets de Microsoft Teams.

Conectando a Microsoft Teams

Como es normal primero vamos a tener que conectarnos a Teams, para ello ejecutamos:

```
Connect-MicrosoftTeams
```

Una vez introducidas las credenciales nos mostrará información sobre nuestro tenant:

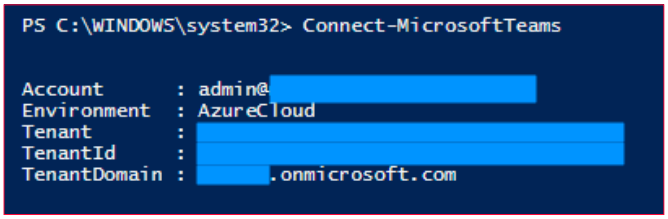


Imagen 4.- Resultado de la conexión a Teams con PowerShell.

Creando un nuevo Team

Para crear un nuevo Team utilizaremos el comando New-Team, podemos crear un Team indicando el nombre del Team en el parámetro DisplayName, o bien asignando el Team a un grupo de Office 365 existente con el parámetro GroupId.

Por ejemplo, para crear un Team que se llame "Team from PowerShell" ejecutaremos:

```
New-Team -DisplayName "Team from PowerShell"
```

```
PS C:\WINDOWS\system32> New-Team -DisplayName "Team from PowerShell"

GroupId                              DisplayName       Visibility  Archived  MailNickName      Description
-------                              -----------       ----------  --------  ------------      -----------
8511ccf1-41f2-41d6-a7dc-5f31d6a8703a Team from Power... Private     False     msteams_fd51e2
```

Imagen 5.- Resultado de la creación de un nuevo Team.

Como vemos en la imagen por defecto nos crea el Team como privado, pero podemos personalizar la creación modificando los parámetros

```
New-Team
  -DisplayName <String>
  [-Description <String>]
  [-MailNickName <String>]
  [-Classification <String>]
  [-Visibility <String>]
  [-Template <String>]
  [-Owner <String>]
  [-AllowGiphy <Boolean>]
  [-GiphyContentRating <String>]
  [-AllowStickersAndMemes <Boolean>]
  [-AllowCustomMemes <Boolean>]
  [-AllowGuestCreateUpdateChannels <Boolean>]
  [-AllowGuestDeleteChannels <Boolean>]
  [-AllowCreateUpdateChannels <Boolean>]
  [-AllowDeleteChannels <Boolean>]
  [-AllowAddRemoveApps <Boolean>]
  [-AllowCreateUpdateRemoveTabs <Boolean>]
  [-AllowCreateUpdateRemoveConnectors <Boolean>]
  [-AllowUserEditMessages <Boolean>]
  [-AllowUserDeleteMessages <Boolean>]
  [-AllowOwnerDeleteMessages <Boolean>]
  [-AllowTeamMentions <Boolean>]
  [-AllowChannelMentions <Boolean>]
  [<CommonParameters>]
```

Para poder listar los Teams que tenemos utilizaremos el comando Get-Team, que nos mostrará información de todos los Teams. O si lo que queremos es obtener las propiedades de un Team en particular podemos ejecutar

```
Get-Team -GroupId 3011ee13-ed98-4faf-9150-155eb07d4d57
```

```
PS C:\> Get-Team -GroupId 3011ee13-ed98-4faf-9150-155eb07d4d57

GroupId                              DisplayName       Visibility  Archived  MailNickName      Description
-------                              -----------       ----------  --------  ------------      -----------
3011ee13-ed98-4faf-9150-155eb07d4d57 Demo              Private     False     Demo              Demo
```

Imagen 6.- Uso de Get-Team para acceder a la información de un Team concreto.

O bien ejecutarlo de esta manera para obtener todas las propiedades del Team:

```
Get-Team -GroupId 3011ee13-ed98-4faf-9150-155eb07d4d57 | Format-List
```

```
PS C:\> Get-Team -GroupId 3011ee13-ed98-4faf-9150-155eb07d4d57 | Format-List

GroupId                           : 3011ee13-ed98-4faf-9150-155eb07d4d57
DisplayName                       : Demo
Description                       : Demo
Visibility                        : Private
MailNickName                      : Demo
Classification                    :
Archived                          : False
AllowGiphy                        : True
GiphyContentRating                : moderate
AllowStickersAndMemes             : True
AllowCustomMemes                  : True
AllowGuestCreateUpdateChannels    : False
AllowGuestDeleteChannels          : False
AllowCreateUpdateChannels         : True
AllowDeleteChannels               : True
AllowAddRemoveApps                : True
AllowCreateUpdateRemoveTabs       : True
AllowCreateUpdateRemoveConnectors : True
AllowUserEditMessages             : True
AllowUserDeleteMessages           : True
AllowOwnerDeleteMessages          : True
AllowTeamMentions                 : True
AllowChannelMentions              : True
```

Imagen 7.- Propiedades de un Team.

Manejando canales

Ahora que ya tenemos creado nuestro Team vamos a agregar algún canal, esto lo hacemos con el comando New-TeamChannel:

```
New-TeamChannel -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a -DisplayName “New Channel”
```

```
PS C:\WINDOWS\system32> New-TeamChannel -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a -DisplayName "New Channel"

Id                                                DisplayName Description
--                                                ----------- -----------
19:26deee559c294a49b008953f9e321a10@thread.skype New Channel
```

Imagen 8.- Creación de un nuevo canal en un Team.

“cuando t engamos el Team creado vamos a poder configurarlo con el comando”

Y si vamos a la aplicación de Teams veremos nuestro canal:

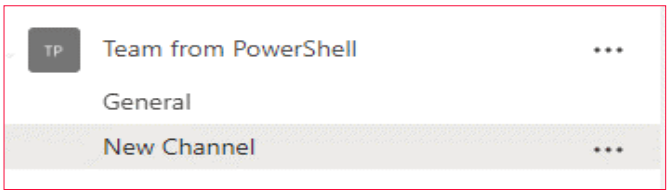


Imagen 9.- Visualización del canal creado.

También podemos modificar el nombre de un canal con el comando Set-TeamChannel

```
Set-TeamChannel -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a -CurrentDisplayName “New Channel” -NewDisplayName “Channel name updated”
```

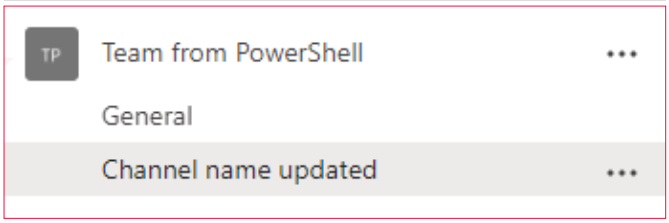


Imagen 10.- Canal actualizado.

Manejando miembros del Team

Desde PowerShell también vamos a poder gestionar los miembros del equipo, por ejemplo, para listar los miembros de un Team ejecutaremos el comando Get-TeamUser

```
Get-TeamUser -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a
```

```
PS C:\> Get-TeamUser -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a

UserId                               User                Name        Role
------                               ----                ----        ----
aee6ab6f-2ffc-4f02-8df9-802d49191fea admin@[redacted]    Ruben Ramos owner
f571c18b-e10f-4e80-94c1-fa9b67b6de99 usertest@[redacted] User Test   member
```

Imagen 11.- Acceso a los integrantes de un Team.

También vamos a poder añadir y eliminar miembros de un

Team:

```
Add-TeamUser -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a -User usertest@tenant.onmicrosoft.com -Role Member

Remove-TeamUser -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a -User usertest@tenant.onmicrosoft.com
```

Personalizando un Team

Una vez que tengamos el Team creado vamos a poder configurarlo con el comando Set-Team. Tenemos disponibles las siguientes opciones:

```
Set-Team
  -GroupId <String>
  [-DisplayName <String>]
  [-Description <String>]
  [-MailNickName <String>]
  [-Classification <String>]
  [-Visibility <String>]
  [-AllowGiphy <Boolean>]
  [-GiphyContentRating <String>]
  [-AllowStickersAndMemes <Boolean>]
  [-AllowCustomMemes <Boolean>]
  [-AllowGuestCreateUpdateChannels <Boolean>]
  [-AllowGuestDeleteChannels <Boolean>]
  [-AllowCreateUpdateChannels <Boolean>]
  [-AllowDeleteChannels <Boolean>]
  [-AllowAddRemoveApps <Boolean>]
  [-AllowCreateUpdateRemoveTabs <Boolean>]
  [-AllowCreateUpdateRemoveConnectors <Boolean>]
  [-AllowUserEditMessages <Boolean>]
  [-AllowUserDeleteMessages <Boolean>]
  [-AllowOwnerDeleteMessages <Boolean>]
  [-AllowTeamMentions <Boolean>]
  [-AllowChannelMentions <Boolean>]
  [<CommonParameters>]
```

Por ejemplo, podemos desactivar la opción para añadir gifs ejecutando:

```
Set-Team -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a -AllowGiphy $false
```

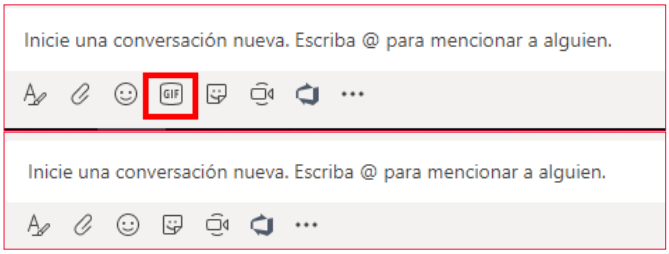


Imagen 12.- Desactivando la opción de GIFs.

Eliminando canales y Teams

Podemos eliminar canales de un Team con el comando Remove-TeamChannel, por ejemplo

```
Remove-TeamChannel -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a -DisplayName "Channel name updated"
```

Y también podemos eliminar Teams con Remove-Team, por ejemplo:

```
Remove-Team -GroupId 8511ccf1-41f2-41d6-a7dc-5f31d6a8703a
```

Conclusión

En esta versión 1.0 del módulo de PowerShell para Teams nos encontramos con diferencias con la última versión 0.96, por ejemplo, se han unificado los comandos para configurar las distintas opciones del Team en Set-Team.

Aunque estamos en la versión 1.0 del módulo, echo de menos comandos para poder manejar tabs y apps, esperemos que Microsoft lo vaya incorporando en futuras versiones. De todas formas, siempre podemos utilizar Graph.

RUBÉN RAMOS MATEO
Technical Architect en Ricoh
ruben_rm@outlook.com
@rubenr79
https://rubenrm.com/

37

Posibilidades de recuperación frente a desastres en SharePoint Online

Una pregunta muy recurrente que he recibido en repetidas ocasiones es la relativa a que opciones proporciona de serie SharePoint Online (SPO) para hacer frente a "desastres" en lo que a perdida de información se refiere. Como os podéis imaginar, en realidad la cuestión de fondo es si Microsoft proporciona algún tipo de mecanismo de Copias de Seguridad y Restauración en SPO. La respuesta a esa pregunta es que "No de forma directa", aunque de forma indirecta si tenemos una serie de herramientas que nos permiten poder recuperar información almacenada en sitios de SPO y en OneDrive For Business (ODFB) en caso de desastre. En este artículo vamos a hacer un repaso a las distintas opciones disponibles.

¿Por qué Microsoft no proporciona características nativas de copia de seguridad y restauración?

La pregunta del millón como se suele decir y para la que Microsoft siempre da la misma respuesta: Office 365 y SPO son unos servicios tan fiables, que no es necesario disponer de una característica nativa de copia de seguridad y restauración a disposición de los clientes. Microsoft proporciona una disponibilidad garantizada del servicio de SPO como parte de Office 365 en base al SLA de Microsoft con sus clientes de Office 365: Microsoft se compromete a garantizar un 99,9 % de disponibilidad en los servicios provistos en la plataforma Office 365 tal y como se refleja en el siguiente enlace: https://docs.microsoft.com/en-us/office365/servicedescriptions/office-365-platform-service-description/service-level-agreement

Versionado de documentos

Por defecto, toda Biblioteca de Documentos en SPO y ODFB tienen habilitado el versionado por lo que si se ha producido un problema / error en una versión de un documento podemos recuperar alguna de sus versiones anteriores. Esta recuperación es posible mediante las siguientes técnicas:

La interfaz de usuario de SPO y ODFB, de forma que cuando se selecciona un documento en una Biblioteca o en ODFB podemos visualizar las versiones que se han ido generando del mismo, abrir una versión previa y restaurar, si es necesario, una versión previa que sobrescriba la actual.

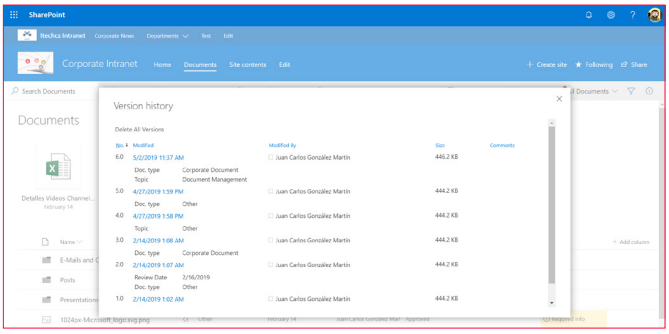


Imagen 1.- Historial de versiones para un documento seleccionado.

En el caso de documentos Office, cuando los abrimos con los clientes de escritorio podemos visualizar también las versiones previas que se han generado del documento y abrirlas.

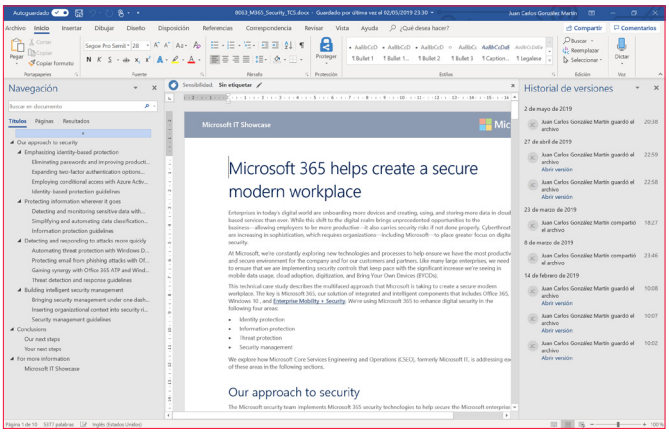


Imagen 2.- Historial de versiones de un documento en el cliente Office (Word).

De forma programática mediante el uso de las APIs de SPO podemos acceder al historial de versiones de un documento almacenado en una Biblioteca de PSO o en ODFB. Por ejemplo, en este artículo se puede ver cómo hacer uso del Modelo de Objetos en Cliente (CSOM) para acceder las versiones de un documento.

Papelera de Reciclaje

Todo Sitio de SPO y espacio de ODFB cuenta con una papelera de reciclaje de 2 niveles:

En el primer nivel de la papelera de reciclaje, cualquier usuario de un Sitio de SPO con al menos permisos de colaborador puede recuperar información eliminada en el sitio durante un período máximo de 90 días. Esta regla aplica también al especio de ODFB, de manera que cada usuario

puede recuperar en el primer nivel de la papelera de reciclaje de su ODDFB cualquier documento eliminado durante la ventana de tiempo indicada:

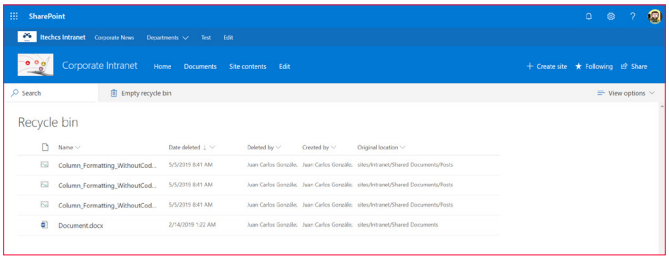


Imagen 3.- Primer nivel de la papelera de reciclaje de un Sitio de SPO.

El segundo nivel de la papelera de reciclaje permite recuperar durante esos 90 días a un usuario administrador cualquier documento que haya sido eliminado del primer nivel. Lógicamente, en el espacio de ODFB cada usuario es administrador del mismo por lo que también puede acceder a ese segundo nivel para recuperar elementos eliminados.

"Office 365 y SPO son unos servicios tan fiables, que no es necesario disponer de una característica nativa de copia de seguridad"

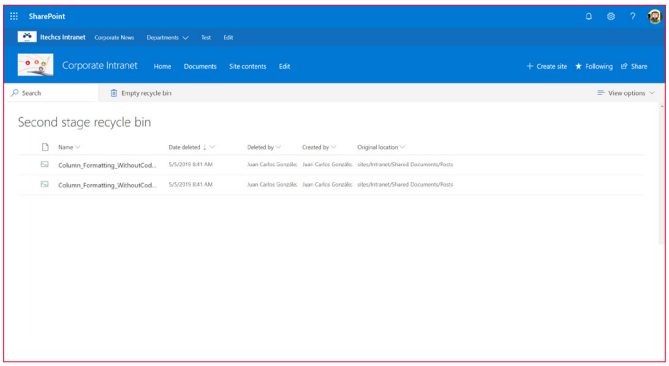


Imagen 4.- Segundo nivel de la papelera de reciclaje de un Sitio de SPO.

Al igual que en el caso de las versiones de un documento, por medio de las APIs de SPO se puede acceder al contenido de la papelera de reciclaje y realizar su restauración. Un ejemplo sobre cómo acceder a la papelera de reciclaje de un sitio de SPO mediante CSOM se puede encontrar en este enlace.

Restaurar ODFB / Bibliotecas de Documentos

A nivel de Biblioteca de Documentos y de ODFB también tenemos la posibilidad de realizar un Restore completo o parcial de su contenido a través de la funcionalidad de Restaurar esta Biblioteca / Restaurar Mi OneDrive. En ambos casos, se puede restaurar el estado de la Biblioteca /Espacio de ODFB de forma completa o bien de ciertos documentos en una fecha anterior a la actual siempre y cuando no excedamos una ventana de 30 días. La característica de "Restaurar esta biblioteca" aparece como una opción más en el menú de configuración:

Esta opción está disponible para el usuario a través del menú de configuración. En el caso de un Sitio de SPO, la acción de "Restore this library" estará disponible para usuarios administradores. En el caso de ODFB, cualquier usuario, como administrador de su ODFB, podrá hacer uso de la opción de "Restore my OneDrive".

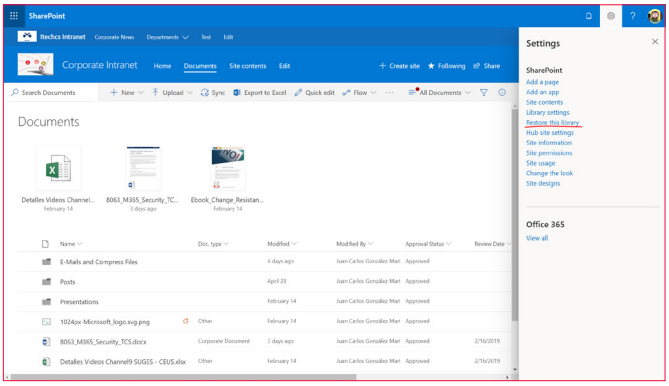


Imagen 5.- Acceso a la opción "Restore this library" en un Sitio de SPO.

A continuación, se muestra la página de "Restaurar esta biblioteca" que es idéntica a la de "Restaurar mi OneDrive". Desde la misma podremos elegir restaurar la biblioteca de acuerdo con las siguientes opciones:

- Ayer.
- Hace una semana.
- Hace tres semanas.
- Un rango de fechas personalizado.

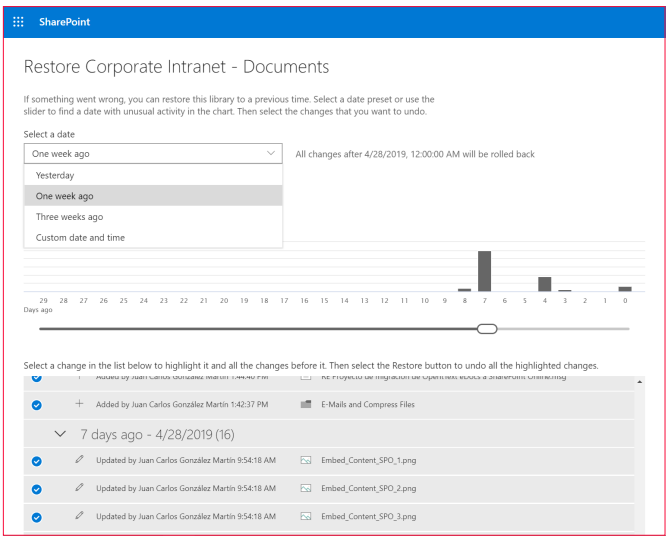


Imagen 6.- Página de restauración de Biblioteca de documentos.

Una vez el administrador selecciona el período de tiempo, puede decidir si restaura todo el contenido de la Biblioteca o bien contenido concreto.

Copias de seguridad y restauración realizadas por Microsoft

Microsoft realiza de manera diaria 2 copias de seguridad de los sitios de SPO y espacios de ODFB de un tenant. Cada copia de seguridad realizada es mantenida por Microsoft un

máximo de 14 días y es posible solicitar una restauración a nivel de Colección de Sitios a través de tickets de soporte. Como se puede deducir, Microsoft no proporciona Copias de seguridad y Restauración con una mayor granularidad.

"toda Biblioteca de Documentos en SPO y ODFB tienen habilitado el versionado"

Herramientas de Backup

Como sección final de este artículo, además de las opciones por defecto para poder recuperarnos frente a pérdidas de información en SPO y ODFB, existen soluciones de terceros que permiten realizar copias de seguridad y restauraciones haciendo uso de las APIs de SPO. Esto implica que estas herramientas no van a proporcionar nunca la flexibilidad a la hora de hacer copias de seguridad que tradicionalmente se ha podido realizar OnPremises y que la velocidad y rendimiento para realizar dichas operaciones dependen de las posibilidades de las APIs.

Conclusión

Microsoft no proporciona de forma nativa herramientas / funcionalidades para que los administradores de Office 365 y/o del servicio de SPO puedan realizar de forma sencilla copias de seguridad / restauraciones. Aunque no existan esas herramientas, si existen distintos mecanismos para facilitar la recuperación de información en caso de desastre.

JUAN CARLOS GONZÁLEZ MARTÍN
Office 365 SME
Office Apps & Services MVP
@jcgm1978

40

Primeros Pasos con Azure Sphere

Después de estar un tiempo trabajando con estas placas multifunción como son Arduino, Raspberry, y algunas otras, la salida al mercado de la Azure Sphere despertó mi curiosidad. Por ello y luego de hacerme con una me ha parecido bueno resumir en este articulo los primeros pasos con esta tarjeta para compartir la experiencia de hacerla funcionar.

Una vez te has hecho con el kit encontrarás en la caja la tarjeta propiamente dicha y un cable USB, imprescindible este último para hacer funcionar la tarjeta las primeras veces, aunque como verás más adelante podrás prescindir de él, en lo que a comunicarse con la placa se refiere.

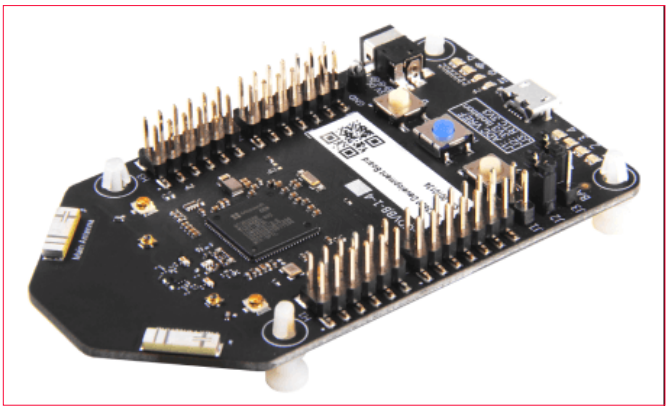


Imagen 1.- Azure Sphere.

Lo primero que debes hacer es conectar la placa con el fin de que se descarguen los drivers, y es importante mencionar que para que todo vaya bien necesitas Windows 10 Anniversary Update instalado en tu ordenador como mínimo. Una vez que la hayas conectado, en el Gestor de Dispositivos verás que se han agregado una serie de Puertos USB cuya asignación COM puede variar de ordenador a ordenador, pero debería ser similar a algo como esto:

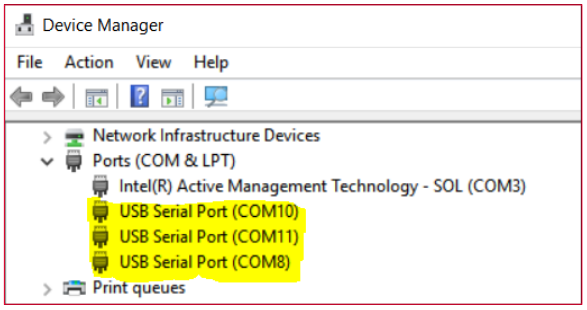


Imagen 2.- Puertos USB agregados.

Si Windows no encuentra los drivers, ni siquiera conectándose a Windows Update, prueba descargarlos desde Future Technology Devices International (FTDI) en la url https://www.ftdichip.com/Drivers/VCP.htm.

Acto seguido toca descargarse el SDK, software que te permitirá la comunicación con la placa, el interfaz de desarrollo con Visual Studio y la operación del hardware directamente desde la línea de comandos (cosas que utilizarás muchas veces al principio). Este software es el Azure Sphere SDK Preview for Visual Studio, descargable desde https://aka.ms/AzureSphereSDKDownload.

A partir de este momento el resto de los pasos de configuración están harto documentados, pero dado que hay uno que es muy importante y que no tiene marcha atrás, te los resumiré aquí con el objetivo de tenerlos a todos en cuenta.

"con el kit encontrarás en la caja la tarjeta propiamente dicha y un cable USB"

Todos los comandos que vamos a ejecutar se realizan a través de la consola de comandos de la Azure Sphere, llamada Azure Sphere Developer Command Prompt (de aquí en adelante ASDcp), que en realidad es la ventana de comandos de toda la vida lanzada con un script previo de configuración llamado "InitializeCommandPrompt.cmd". Para no complicarse la vida y tener que estar buscándose ese script mejor lanzarlo siempre ejecutando el ASDcp mencionado anteriormente y listo.

Paso 1/5: Actualización del S.O.

La mayoría de las Sphere que están listas para su distribución te llegarán con el sistema operativo sin actualizar, por lo que es imprescindible actualizarlo antes de realizar cualquier otra acción. Para hacerlo, con la Sphere conectada mediante el cable USB a tu ordenador, abres una ventana de ASDcp y lanzas una "recuperación" manual, lo que forzará la actualización del sistema operativo, con el siguiente comando: azsphere device recover. Se paciente a hasta que termine, irás viendo en pantalla mensajes como estos:

Starting device recovery. Please note that this may take up to 10 minutes.
Board found. Sending recovery bootloader.
Erasing flash.
Sending images.
Sending image 1 of 16.
Sending image 2 of 16.
. . .
Sending image 16 of 16.
Finished writing images; rebooting board.
Device ID: <GUID>
Device recovered successfully.

Command completed successfully in 00:02:37.3011134.

Paso 2/5: Iniciando sesión en la Sphere

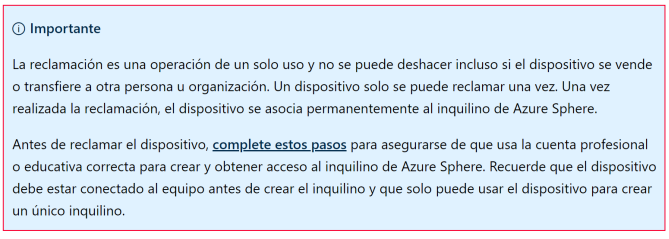
El inicio de sesión es sencillo: desde una ventana de ASDcp lanza el siguiente comando: azsphere login. Si toda va bien, saldrá una ventana donde se te piden credenciales, que deben ser credenciales con acceso a un Tenant de Azure (preferiblemente credenciales con rol de admin en el Tenant). Te recomiendo, por los pasos que verás a continuación, que la cuenta sea una cuenta de pago, especialmente porque más adelante la Sphere quedará asociada definitivamente a tu Tenant y el acceso es algo que no deberías perder si no quieres perder también la Sphere. En la ventana de ASDcp un mensaje del tipo:

"Successfully logged in with the selected AAD user. This authentication will be used for subsequent commands. Command completed successfully in 00:00:22.4007290."

Confirma que todo ha ido bien.

Paso 3/5: Reclamación del Dispositivo

Este paso es el paso más importante de la configuración, ya que es un paso que una vez realizado no tendrá marcha atrás y tu Sphere quedará asociada definitivamente al Tenant sobre el que has iniciado sesión. Las advertencias de Microsoft al respecto están por todos lados, pero en las propias páginas de configuración te encuentras con este mensaje:


ⓘ Importante

La reclamación es una operación de un solo uso y no se puede deshacer incluso si el dispositivo se vende o transfiere a otra persona u organización. Un dispositivo solo se puede reclamar una vez. Una vez realizada la reclamación, el dispositivo se asocia permanentemente al inquilino de Azure Sphere.

Antes de reclamar el dispositivo, complete estos pasos para asegurarse de que usa la cuenta profesional o educativa correcta para crear y obtener acceso al inquilino de Azure Sphere. Recuerde que el dispositivo debe estar conectado al equipo antes de crear el inquilino y que solo puede usar el dispositivo para crear un único inquilino.

Para iniciar el proceso de reclamación (cosa imprescindible para poder utilizar tu Sphere) debes ejecutar en una ventana de ASDcp el comando: azsphere device claim. Una vez ejecutado el comando deberías obtener un resultado similar a este:

Claiming device.
Claiming attached device ID 'Aqui el ID del device' into tenant ID 'Tenant ID'.
Successfully claimed device ID ' Aqui el ID del device' into tenant ID 'Tenant ID'.

Command completed successfully in 00:00:05.5459143.

"La mayoría de las Sphere que están listas para su distribución te llegarán con el sistema operativo desactualizado"

Paso 4/5: Conexión a WiFi

La conexión a una red Wifi es muy sencilla. Desde una ventana de ASDcp ejecutas el siguiente comando (reemplazando yourSSID por el nombre de tu red, y yourNetworkKey por la clave de tu red wifi): azsphere device wifi add --ssid <yourSSID> --key <yourNetworkKey>. Los parámetros se pasan sin comillas.

Si tienes algún problema recuerda que la Sphere solo soporta redes 802.11b/g/n y encriptación WPA/WPA2. Puedes verificar que la conexión es correcta ejecutando el comando azsphere device wifi show-status.

Paso 5/5: Preparar el dispositivo para el desarrollo y depuración

Antes de poder desarrollar y ejecutar aplicaciones en el dispositivo tendrás que habilitarlo para este fin, ya que viene bloqueado por defecto. Lo lograrás ejecutando desde la ventana ASDcp el comando: azsphere device prep-debug. Este comando en realidad hace dos cosas, en primer lugar, acepta que un PC le envíe aplicaciones para ser ejecutadas y depuradas, y en segunda lugar inicia un Debug Server en el dispositivo. Al mismo tiempo pone el dispositivo en modo OTA updates off (OTA=over the air) lo que hace que todo el desarrollo y despliegue se realice con y desde tu PC. El comando contrario a este modo de trabajo es azsphere device prep-field.

Implementación OTA

No voy a entrar en los detalles sobre cómo utilizar OTA en este articulo ya que su extensión sería demasiada, pero si es importante conocer que existe este modelo para gestionar y distribuir aplicaciones. Básicamente el modo OTA (over the air updates) hace que el dispositivo reciba las actualizaciones desde el mismo Azure y no desde tu PC. Debes elegir entre un modo u otro, es decir, o las aplicaciones se envían desde el PC (OTA desactivado) o desde Azure (OTA activado). En realidad la implementación OTA ofrece una aplicación a través de una fuente (un feed) a los dispositivos pertenecientes a un grupo de dispositivos deter-

minado y que coincida con la referencia (SKU) de destino para ese feed. Es muy útil cuando tienes varios dispositivos conectados y desperdigados entre varias ubicaciones, y deseas tanto monitorizarlos como actualizarlos desde un único lugar. En mi opinión, OTA es el mejor modelo de distribución y actualización una vez que tu aplicación está lista, siendo NO OTA el modelo ideal mientras estás en desarrollo y necesitas hacer pruebas, depurar, y hacer despliegues constantemente.

"saldrá una ventana donde se te piden credenciales, que deben ser credenciales con acceso a un Tenant de Azure"

Primera Aplicación

Ejecutar tu primera aplicación es realmente muy sencillo y como muestra tenemos la aplicación Blink, que hará que uno de los leds integrados en la Sphere parpadee a intervalos regulares. Para ello no tienes más que abrir tu Visual Studio y crear un nuevo proyecto, seleccionando la nueva plantilla según ves en esta imagen:

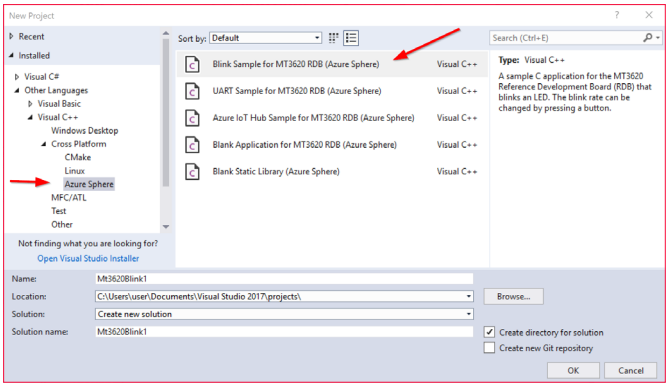


Imagen 3.- Creación de un proyecto para Azure Sphere.

Una vez hayas completado el campo de nombre y ubicación del proyecto, das OK y se habrá creado todo el código. A partir de ese momento solo tienes que ejecutarlo (no hay que tocar nada en el código) y verás que el primer led de la Sphere comienza a parpadear. Cada vez que presiones el primero de los botones de la Sphere (Botón A) verás que la frecuencia de parpadeo cambia. Te invito a que cambies lo necesario para hacer que el botón B cambie el color en el que parpadea el led, lo descubrirás simplemente estudiando un poco el código.

Conclusión

Como conclusión y totalmente personal debo decir que la Azure Sphere me ha impresionado para bien. Lo mejor para hacerse una idea propia es probar el dispositivo durante un tiempo y desarrollar aplicaciones según nuestras necesidades, pero a modo de pequeño resumen os dejo el siguiente cuadro:

Contras:

- La asignación única (reclamación) es un proceso que une al dispositivo a un Tenant de forma irremediable, por lo que desde mi punto de vista es un punto para revisar por el equipo de producto.
- En materia de sensores y otros "artilugios" conectables al dispositivo aún no hay mucha información. Hay muchos ejemplos prácticos que utilizan un kit adicional que yo no adquirí y que os recomiendo: Grove Starter Kit, aunque está claro que con el tiempo habrá más y más ejemplos chulos que no necesiten de este kit.

Ventajas:

- El modelo de seguridad y la conexión nativa a Azure es sin duda una gran ventaja a favor de este dispositivo.
- La capacidad de interconexión promete ser interminable.
- La facilidad de instalación.
- El funcionamiento en modo OTA permite la gestión de innumerables dispositivos desde un punto único, controlado y seguro, y que en realidad es el punto en el que más se ha pensado al momento de diseñar este dispositivo. Para muestra y final adjunto este pequeño diagrama de interconexión y update para múltiples dispositivos.

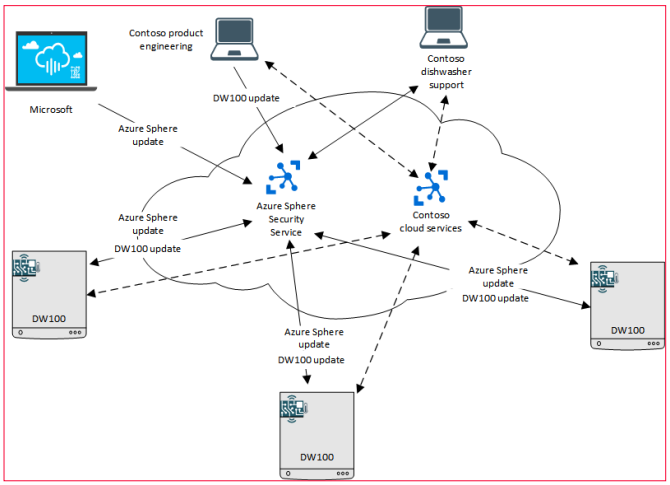


Imagen 4.- Diagrama de interconexión con múltiples dispositivos.

"para poder desarrollar y ejecutar aplicaciones en el dispositivo tendrás que habilitarlo para este fin, ya que viene bloqueado por defecto"

JAVIER MENENDEZ PALLO
Disruptive Solutions Manager en Insight

Mentoring

Comparti*MOSS*

Un servicio experto alrededor de su SharePoint

CompartiMOSS le puede ayudar a través de su programa de Mentoring!

Contacte con nosotros y le enviaremos los planes de mentoring que tenemos disponibles para SharePoint.

44

Add a user as an admin on all associated SharePoint Sites on a Hub

When we are starting or doing development on SharePoint online most of the time, and since the introduction of the Hub Site (hubify), projects are done with this new structure. Adding a new user as admin on each site of the Hub will be very time consuming. With the PowerShell Script shared here, this task can be done in a few seconds.

In order to run this script successfully, you require Global SharePoint administrator rights. Furthermore, a site belonging to the Hub needs to be added, as well as the email of the person to be added as an admin on your Hub.

Make sure that you also have all the SharePoint Online Management Shell installed on your desktop. You can open your PowerShell as an Administrator and paste the next line to install all the required cmdlets.

```
Install-Module -Name Microsoft.Online.SharePoint.PowerShell
```

The following Power Shell script will make a user an admin of your Hub (replace the required values).

```
$adminCenter = "https://contoso-admin.sharepoint.com" #SharePoint admin center
$hubSite = "https://contoso.sharepoint.com/sites/hubcontonso" #Hub Site root
$username = "contoso@contoso.com" #Your email
$password = "password" #Your password
$adminEmail = "contosoAdmin@contoso.com" #New Admin

$encpassword = convertto-securestring -String $password -AsPlainText -Force
$cred = new-object -typename System.Management.Automation.PSCredential -argumentlist $username, $encpassword

Connect-SPOService -Url $adminCenter -Credential $cred
$hub = Get-SPOHubSite $hubSite
$sites = Get-SPOSite -Limit All
$sitesFromHub = New-Object System.Collections.ArrayList

Write-Host ("Searching {0} sites" -f $hub.Title) -BackgroundColor Gray -ForegroundColor Black
foreach ($site in $sites){
	try{
		$hubSiteID = Get-SPOHubSite $site.Url
		if($hubSiteID.ID -eq $hub.ID){
			Write-Host ("* {0} - {1} ... " -f $site.Title, $site.Url) -NoNewline
			$sitesFromHub.add($site) | Out-Null
			Write-Host "Done" -BackgroundColor Green -ForegroundColor Black
		}
	}
	catch{
		continue
	}
}
Write-Host Write-Host ("Adding {0} as Admin:" -f $adminEmail) -BackgroundColor Gray -ForegroundColor Black
foreach ($hubSiteAssociated in $sitesFromHub){
Write-Host ("* {0} - {1} ... " -f $hubSiteAssociated.Title, $hubSiteAssociated.Url) -NoNewline
try{
	Set-SPOUser -Site $hubSiteAssociated -LoginName $adminEmail -IsSiteCollectionAdmin $true | Out-Null
	Write-Host "Done" -BackgroundColor Green -ForegroundColor Black
}
catch{
	Write-Host "No Permissions" -ForegroundColor Black -BackgroundColor Red
 }
}
Write-Host "All Done"
Write-Host "Press any key to Close..."
$Host.UI.RawUI.ReadKey("NoEcho,IncludeKeyDown")
```

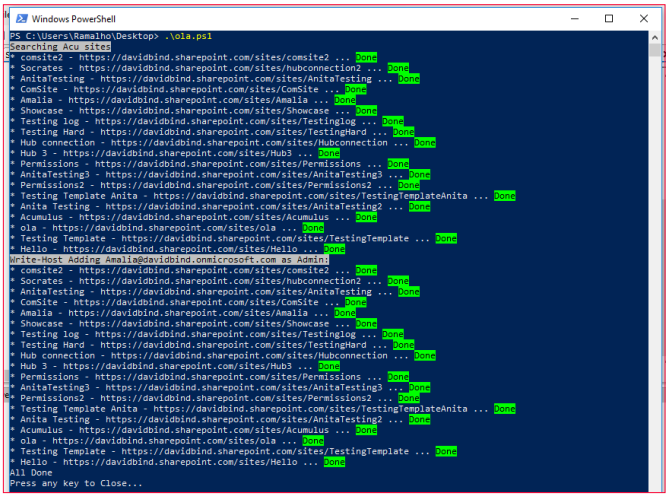


Image 1.- Output generated by the Script execution.

Conclusion

This script will add the user as an admin on the SharePoint sites that are associated with the Hub. This script can be useful when you're developing the Hub sites and keeping the content updated on SharePoint sites.

DAVID RAMALHO
SharePoint Developer at BindTuning

45

Azure App Configuration

Microsoft sigue invirtiendo en Azure con nuevos servicios que ayudan, en este caso, a gobernar las configuraciones de nuestras aplicaciones. App Configuration es un nuevo servicio que nos permite almacenar los parámetros de nuestras aplicaciones de una forma rápida y escalable. Un servicio optimizado para separar totalmente la configuración del código que gestiona los diferentes pares de clave-valor que usamos en las aplicaciones junto con funcionalidades extras de etiquetado, versionado en el tiempo, seguridad basada en identidades administradas de servicios y/o encriptación en transporte y en almacenamiento.

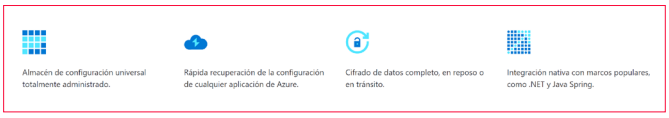


Imagen 1.– Características del servicio App Configuration

"App Configuration es un nuevo servicio que nos permite almacenar los parámetros de nuestras aplicaciones de una forma rápida"

Configuration Explorer

Cuando accedemos al servicio, el explorador de configuración ofrece una interfaz jerárquica para administrar todas las configuraciones que hemos creado.

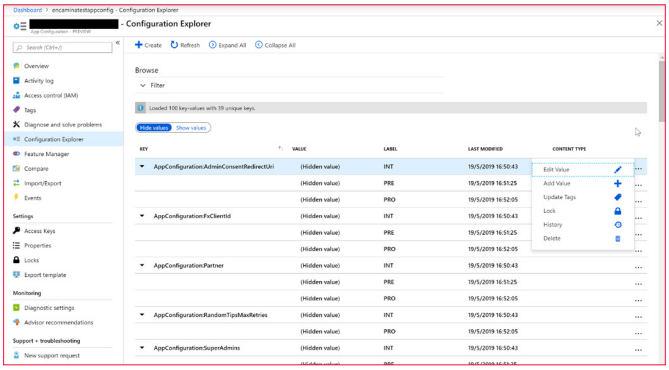


Imagen 2.– Configuration Explorer

La jerarquía se construye usando las etiquetas (LABEL) que podemos asociar a cada una de las claves que creamos, con esto, podemos tener diferentes configuraciones para cada par de clave-valor, por ejemplo, para representar los diferentes valores que podamos tener por entorno de ejecución, Integración, Preproducción y Producción, pero tenemos la libertad de gestionar esas etiquetas como necesitemos, otro ejemplo podría ser usarla para diferenciar los valores en las diferentes regiones geográficas de Azure donde ejecutamos la aplicaciones, North Europe, West Europe, West US, …

"obtenemos todas las claves que no tienen etiquetas junto a las que pertenecen al valor de la variable de nuestro entorno"

Librerías de cliente

Para usar el servicio tenemos diferentes librerías, en función del tipo de aplicación que estamos desarrollando:

FRAMEWORK	COMO CONECTAR	LIBRERÍA
.NET Core and ASP.NET Core	Proveedor de App Configuration para .NET Core	Microsoft.Extensions.Configuration.AzureAppConfiguration https://www.nuget.org/packages/Microsoft.Extensions.Configuration.AzureAppConfiguration
.NET and ASP.NET	Constructor de App Configuration para .NET	Microsoft.Configuration.ConfigurationBuilders.AzureAppConfiguration https://www.nuget.org/packages/Microsoft.Configuration.ConfigurationBuilders.AzureAppConfiguration
Java Spring	Cliente de App Configuration para Spring Cloud	spring-cloud-starter-azure-appconfiguration-config https://mvnrepository.com/artifact/com.microsoft.azure/spring-cloud-starter-azure-appconfiguration-config
Otros	API REST	https://github.com/Azure/AppConfiguration#rest-api-reference

Cómo usarlo en ASP.NET Core

Una vez que hemos creado el servicio y creamos las claves, bien manualmente o bien con el proceso de importación, lo primero que tenemos que hacer para usarlo en nuestra aplicación ASP.NET Core es instalar el paquete Nuget de

configuración.

```
dotnet add package Microsoft.Extensions.Configuration.AzureAppConfiguration --version 1.0.0-preview-008520001
```

Ahora modificamos el proceso de carga de nuestra aplicación para que use el servicio de App Configuration cuando no estamos en el entorno de desarrollo (Development).

"centralizamos las configuraciones de las aplicaciones, además de añadir seguridad al acceso a la mismas"

```
namespace DemoAppConfig
{
    public class Program
    {
        public static IConfiguration Configuration { get; set; }

        public static void Main(string[] args)
        {
            var builder = new ConfigurationBuilder()
                .SetBasePath(Directory.GetCurrentDirectory())
                .AddJsonFile("appsettings.json", optional: false, reloadOnChange: false)
                .AddEnvironmentVariables();

            if (Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT") != "Development")
            {
                builder.AddAzureAppConfiguration(options => {
                    options.ConnectWithManagedIdentity("https://[redacted].azconfig.io")
                        .Use(KeyFilter.Any, LabelFilter.Null)
                        .Use(KeyFilter.Any, Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT"));
                });
            }

            Configuration = builder.Build();

            CreateWebHostBuilder(args).Build().Run();
        }

        public static IWebHostBuilder CreateWebHostBuilder(string[] args) =>
            WebHost.CreateDefaultBuilder(args)
                .UseConfiguration(Configuration)
                .UseStartup<Startup>();
    }
}
```

Imagen 3.- Proceso de carga de la aplicación con Azure App Configuration.

Hay que tener en cuenta que, en el ejemplo anterior, el método de autenticación del servicio de App Configuración usa una identidad administrada del servicio de Azure donde se va a ejecutar la aplicación. En la siguiente imagen, vemos cómo podemos realizar esta conexión usando una clave de acceso del servicio App Configuration.

```
if (Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT") != "Development")
{
    builder.AddAzureAppConfiguration(options => {
        options.Connect("Endpoint=https://[redacted].azconfig.io;Id=2-l9-s0:yy5BYWzWr4MEalZpEEOm;Secret=[redacted]")
            .Use(KeyFilter.Any, LabelFilter.Null)
            .Use(KeyFilter.Any, Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT"));
    });
}
```

Imagen 4.- Conexión usando la cadena de conexión.

El método Use nos permite realizar filtros sobre las claves que son accesibles por la aplicación, en nuestro caso, obtenemos todas las claves que no tienen etiquetas junto a las que pertenecen al valor de la variable de nuestro entorno, por ejemplo, INT, PRE o PRO.

El resto de nuestro código básicamente es el mismo, si ya estamos usando la implementación de .NET Core IConfiguration. Por ejemplo, en la siguiente línea de código, leemos una sección de configuración llamada DevOpsTeams que se almacenará en el contenedor de dependencias.

```
services.Configure<Models.DevOpsTeamConfig>(Configuration.GetSection("DevOpsTeam"));
```

Conclusiones

Con Azure App Configuration centralizamos las configuraciones de las aplicaciones, además de añadir seguridad al acceso a la mismas, conseguimos tener una herramienta para administrar las claves de nuestras aplicaciones.

ALBERTO DIAZ MARTIN
MVP Azure
adiazcan@hotmail.com
@adiazcan
http://blogs.encamina.com/por-una-nube-sostenible/

47

Asistentes con .net – Parte II (gactions)

En el anterior artículo vimos cómo era la estructura de un asistente virtual, de que partes estaba formado, recuerda que todos tienen una estructura similar. Te voy a ser sincero, la idea era que en este artículo te iba a enseñar cómo hacer que Dialogflow consultase a tu API y empezar a crear el bot, pero como yo no soy ningún genio y también estoy en proceso de aprendizaje me he estado pegando con Dialogflow hasta que llegué a un punto en el que rompí nuestra relación. Te explico, Google nos dice que la forma de trabajar es la que os conté en el otro artículo, Google Assistant se conecta a Dialogflow y nosotros le indicamos a Dialogflow que en ciertos puntos llame a nuestro endpoint (webhook) para desarrollar más una intención. Google también nos dice que nos brinda todas las herramientas necesarias, nos da ejemplos y nos da el Dialogflow SDK en un montón de lenguajes:

- Node.js
- Python
- Java
- Go
- Ruby
- C#
- PHP

Hasta aquí todo pintaba bien, los ejemplos solo los dan en Node.js y Java, pero bueno, no tiene que ser muy distinto para C#... ¡ERROR! Una vez me puse a desarrollar me di cuenta de que solo proporcionan un SDK real para Node.js y Java, el resto son estructuras de clases para deserializar el request que llega de Dialogflow y el response que tendremos que enviar, puede parecer interesante, pero no lo es debido a que no contempla ninguna de las reglas que tiene la API, por ejemplo este SDK no controla que nuestro primer mensaje en el response sea del tipo "SimpleResponse" como indica su documentación, tampoco ofrece soporte para poder usar las built-in intents del sistema, tenemos un problema.

Como es open source podemos pensar en hacerlo nosotros y hacer un pull request, pero ya lo dicen los archivos que podemos ver en el GitHub[1] estos ficheros son autogenerados desde las especificaciones de la API y en futuros cambios se dañara todo. Esto solo me dejaba una salida (Do It Yourself), así que manos a la obra.

Como comenté más arriba lo interesante es poder acceder a las built-in ya que son necesarias para cosas como pedir los datos del usuario, no hay otra forma a no ser que se los pidas directamente al usuario, pero te podría mentir y no creo que sea la forma adecuada. Llegados a este punto empecé a tener problemas, la documentación no parece actualizada en todos los puntos, algunas respuestas desde Dialogflow no tienen ningún mensaje del error, se limitan a un "Malformed Response" y no estaba avanzando correctamente. Tocaba tomar un nuevo rumbo y volví a investigar.

Aquí descubrí "gactions CLI", la interfaz en línea de comandos para proyectos en Actions. Se encuentra disponible para Windows, Mac y Linux y sin duda es el mejor camino si quieres hacer un bot sin emplear Dialogflow como NLP (Natural Language Processing). Y te podrías preguntar "Diego ¿Y por qué no lo hiciste con Java o NodeJS?", yo porque soy raro y quería usar LUIS[2]. Volviendo a gactions el uso en Windows es muy sencillo, descargamos el .exe desde la web de la documentación[3] y lanzamos el comando "gactions init" para crear el fichero action.json que necesitaremos luego para enviar a Google Actions la configuración del proyecto.

Este action.json tendrá la siguiente estructura mínima:

```
{
 "actions": [
  {
   "description": "Default Welcome Intent",
   "name": "MAIN",
   "fulfillment": {
    "conversationName": "{PROJECT-NAME}"
   },
   "intent": {
    "name": "actions.intent.MAIN",
    "trigger":{
     "queryPatterns":[
      "talk to DEMO"
     ]
    }
   }
  }
 ],
 "conversations": {
  "{PROJECT-NAME}": {
   "name": "{PROJECT-NAME}",
   "url": "{URL-ENDPOINT}",
   "fulfillmentApiVersion":2
  }
 }
}
```

1 https://github.com/googleapis/google-cloud-dotnet/tree/master/apis/Google.Cloud.Dialogflow.V2

2 https://www.luis.ai/home

3 https://developers.google.com/actions/tools/gactions-cli

En este json indicaremos los actions con los que contara nuestra Action (menuda redundancia en los nombres), un Google Action puede tener distintos actions, por ejemplo, para cada idioma. En este caso se indica un action built-in llamado “actions.intent.MAIN” que corresponde a la intención de llamar o iniciar un Action, con el mensaje “talk to DEMO”. Posteriormente indicamos cual será el endpoint del Action y aquí está el punto interesante, si registramos el action en la web solo nos permitirá registrar un endpoint de Dialogflow, pero al hacerlo con gactions podremos registrar otro endpoint (nuestra API en Azure, por ejemplo) y olvidarnos de Dialogflow.

“Como es open source podemos pensar en hacerlo nosotros”

Como dije en esta serie Dialogflow no nos aporta nada porque mi intención es usar LUIS como servicio NLP, pero podrías usar Dialogflow como NLP y usar .net para extender, por ejemplos para consultar servicios como el tiempo, una API personal de stock… pero repito que la idea es abstraernos completamente de servicios externos y tener la mínima dependencia posible de la plataforma.

El fichero no tiene por qué llamarse “action”, una buena práctica es indicar el idioma del action, en este caso podría ser “action.es.json”.

Ya tenemos nuestro fichero actions.json, pero primero debemos de crear un Actions en la “Actions on Google console” en la dirección https://console.actions.google.com *(Nota: las cuentas empresariales de GSuite pueden tener limitaciones a la hora de probar o desplegar Actions)*, crear un Actions es tan sencillo como ponerle un nombre e indicar un idioma inicial y tu región.

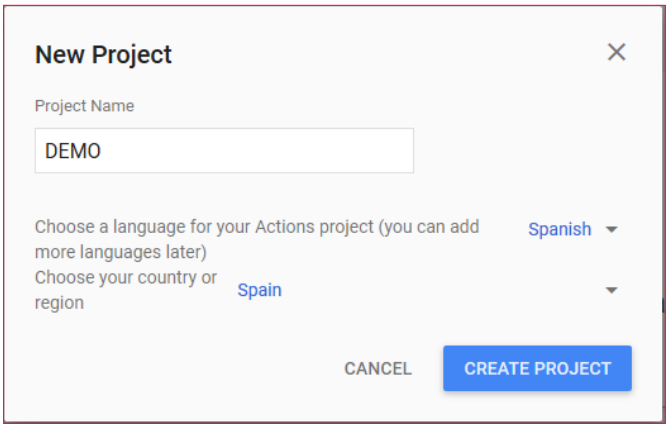


Imagen 1.- Creación del proyecto de Actions.

Después te dan varios ejemplos de plantillas que puedes usar, pero nosotros vamos a indicar que usaremos el “Actions SDK” (que te recuerdo que no dan para C#, pero ya te diré lo que vamos a hacer).

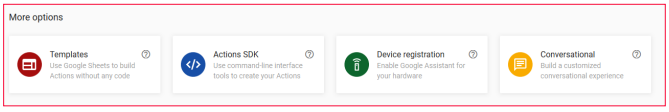


Imagen 2.- Plantillas disponibles para crear Actions.

Aquí ya te habla sobre gactions y te enlaza con la documentación.

En la siguiente página nos indica los pasos mínimos que debemos de completar como:

- Indicar como se invocará nuestra Action.
- Construir Actions.
- Probarlo en el simulador.
- Pasos para desplegar en producción.

Los dos primeros puntos lo haremos con el fichero action.json. Nos dirigimos a “Project settings” (la interfaz está en inglés) y necesitamos nuestro “Project ID” para gactions.

“PACKAGE_NAME es el nombre de nuestro action.json”

Volvemos a gactions.exe y usaremos el comando:

```
gactions update --action_package PACKAGE_NAME --project PROJECT_ID
```

Donde “PACKAGE_NAME” es el nombre de nuestro action.json (o la ruta si no está en la misma carpeta). Gactions nos pedirá acceso para terceros en nuestra cuenta de Actions on Google a través de una URL para autentificarnos y realizará la subida del fichero, si hubiese cualquier error con el json nos lo indicará y en el momento podremos ver en la web como se han creado los actions que pusiésemos en el json.

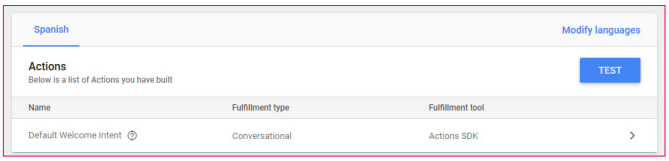


Imagen 3.- Proyecto creado.

Ahora ya tenemos nuestro Assistant haciendo llamadas directamente a nuestra API, esto es válido para usar con cualquier lenguaje, incluso podrías usar VB.NET si quisieras.

En el próximo artículo vamos a recibir las llamadas desde Assistant y ver la estructura de la información que recibimos.

DIEGO ZAPICO
Desarrollador en Encamina

49

CosmosDB Integration Testing con AzureDevOps

En todo desarrollo de software aparte de realizar Unit Test, también realizamos Intgration Tests para verificar que las conexiones que tiene que hacer nuestra aplicación con terceras partes funcionan y que con las nuevas versiones no se ha roto esta integración.

Integration Tests contra CosmosDB

CosmosDB es el servicio para NoSql que nos provee Azure, y es uno de los servicios más usados en estos momentos. Hasta ahora si queríamos hacer Integration Testing de nuestra aplicación contra CosmosDB teníamos que levantar el servicio de CosmosDB, ejecutar nuestras pruebas y después eliminarlo, con el coste que conlleva tanto de tiempo como económico.

"CosmosDB es el servicio para NoSql que nos provee Azure"

Desde hace unos meses esto ya no es necesario ya que Microsoft ha implementado una tarea en AzureDevOps que nos levanta en un container el emulador de CosmosDB, lo que nos permite ejecutar nuestras pruebas de integración sobre el emulador y sin coste alguno.

En el siguiente link os podéis bajar el código del ejemplo que vamos a seguir: https://dev.azure.com/bermejo/CosmosDBEmulator/_git/CosmosEmulator

El código consiste en un pequeño proyecto de consola en .NET Core que lo único que hace es añadir un documento a una colección de CosmosDB.

Además, hay un proyecto de test que lo que hace es hacer el test de integración con CosmosDB.

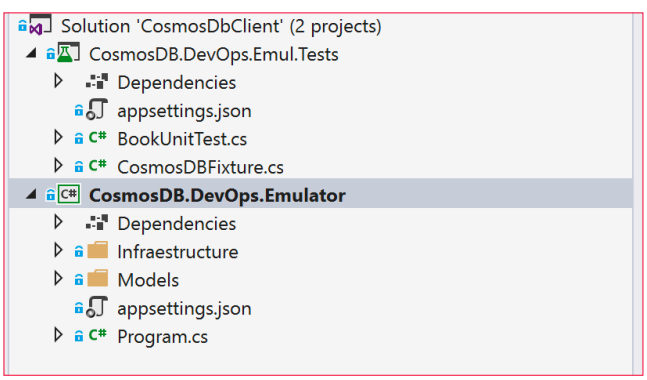


Imagen 1.- Emulador de CosmosDB en Visual Studio.

El archivo de configuración de nuestro proyecto de test debe tener la siguiente forma:

De este archivo es importante dos cosas:

1.– La AuthorizationKey es la key del emulador de CosmosDB, debe ser esta para funcionar.

2.– Y el valor del EndPointUrl será un valor que más adelante modificaremos con el EndPoint que nos dé el contenedor del emulador de CosmosDB.

Añadiendo la tarea de CosmosDB Emulator al pipeline

Para añadir en nuestro pipeline de build el emulador de CosmosDB, vamos a añadir task, buscamos Cosmos DB Emulator y lo añadimos en nuestro pipeline.

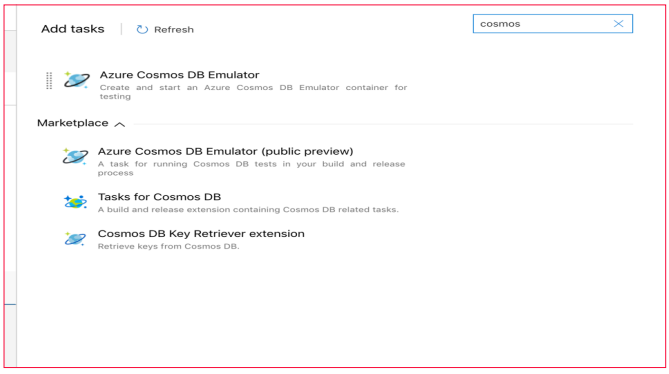


Imagen 2.- Añadiendo la tarea de CosmosDB Emulator al Pipeline.

Una vez la tenemos añadida, debemos ir a la opción Output Variables y añadiremos una descripción para la variable de salida, esta variable nos servirá para saber el endpoint que expondrá el contenedor.

"añadiremos una tarea de powershell que modifica el valor de la variable EndPointUrl del archivo de configuración de nuestros test"

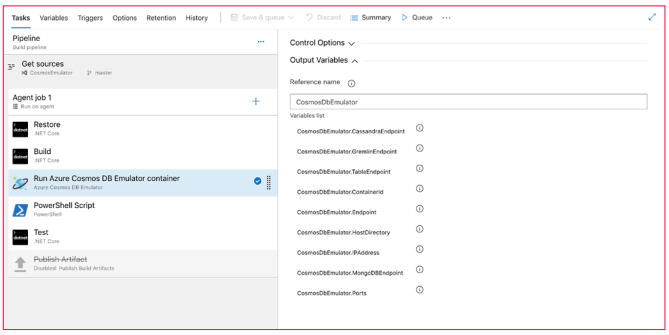


Imagen 3.- Añadiendo la descripción de la variable de salida.

Una vez tenemos la tarea añadida, añadiremos una tarea de powershell que modifica el valor de la variable EndPointUrl del archivo de configuración de nuestros test, por el endpoint de emulador de CosmosDB, para ello ejecutaremos el siguiente script:

```
(Get-Content .\CosmosDB.DevOps.Emul.Tests\appsettings.json) -Replace 'CosmosEndpointUrl', '$(CosmosDbEmulator.Endpoint)' | Set-Content .\CosmosDB.DevOps.Emul.Tests\appsettings.json
```

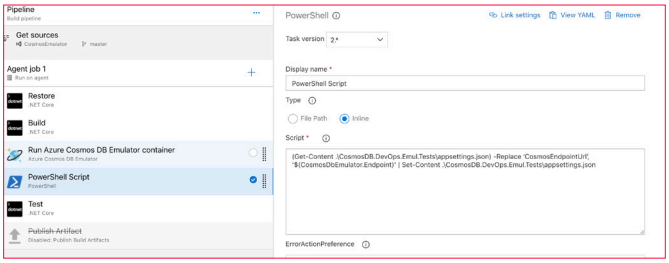


Imagen 4.- Añadiendo la tarea PowerShell que modifica la variable.

Una vez tenemos esta configuración ya podemos lanzar nuestro pipeline de build con nuestros Integration Tests contra CosmosDB.

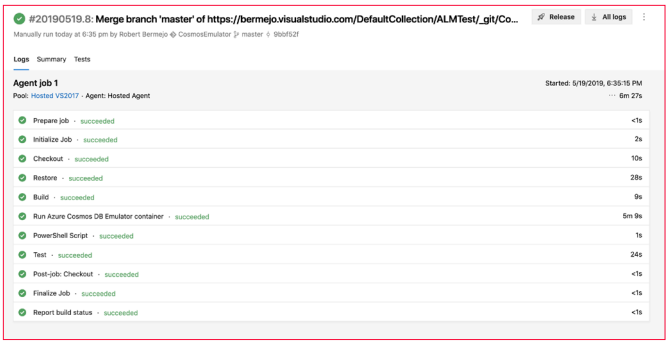


Imagen 5.- Build con los Integration Tests contra CosmosDB.

Conclusiones

La llegado del emulador dentro de nuestros pipelines es una gran noticia ya que nos ahorra dinero y tiempo de configuración de entornos.

En contra debemos tener los siguientes puntos:

- Sino tenemos un self hosted agent, solo funciona con el Hosted VS2017 que nos proporciona AzureDevOps.
- El proceso de build es muy lento ya que tiene que bajar el contenedor instalarlo, configurarlo y levantarlo.

Viendo los puntos anteriores lo más recomendable es ejecutar estos pipelines en un self hosted agent y de esta forma tendremos en cache el contenedor y la velocidad será mayor.

ROBERT BERMEJO BLASCO
BackEnd Technical Lead in SCRM – Lidl Digital Hub
MVP Azure
bermejoblasco@live.com

51

Entrevista Tokiota

TOKIOTA es una compañía nacida en Barcelona, con oficinas en Madrid y A Coruña. Contamos con más de 90 personas, entre consultores internos y consultores asociados, cada uno de ellos experto en su área de especialización. Flexible a cada cliente y escenario, pero procedimental, para garantizar la calidad y el éxito de los proyectos. Orientada a proporcionar soluciones de Negocio con base Tecnológica.

東京

TOKIOTA

¿Cuál es el origen del nombre de la empresa?

Aunque nacimos en Barcelona, tenemos una filosofía de trabajo que potencia valores heredados de la cultura japonesa: apostamos por la eficiencia, la calidad y la honestidad, estando a la cabeza de la vanguardia tecnológica.

¿Por qué y cómo empezó en el mundo de la tecnología?

TOKIOTA ha arrancado en la era Cloud. Cuando abrimos, hace ya algo más de 6 años, muchos eran los que hablaban de los beneficios de la nube. Sin embargo, tenían que seguir manteniendo sus negocios convencionales de Datacenter y Administración de entornos. Nosotros hemos tenido la suerte de no vernos lastrados por un pasado legacy y hemos apostado desde el principio por un negocio basado en el Cloud, siendo pioneros en el posicionamiento e implantación de soluciones de Azure.

¿Cuáles son las principales actividades tecnológicas hoy en día?

Nuestro negocio está orientado a Cloud e Inteligencia Artificial, aportando soluciones tecnológicamente innovadoras basadas en los siguientes pilares: Intelligent Apps, DevOps, IoT, Analítica Avanzada y Predictiva, y Modern Workplace.

Y actualmente estamos consolidando nuestra práctica de seguridad, especializada para entornos Cloud y puesto de trabajo.

¿Cuáles son las principales actividades NO tecnológicas hoy en día?

Nuestro objetivo principal es hacer la vida mejor a todas las personas con las que nos relacionamos en nuestro día a día: a nuestros clientes, a nuestros partners, y, sobre todo, a nuestros consultores, pudiendo presumir de un equipo humano espectacular. Gente buena y buena gente.

¿Cuáles son las actividades que realiza en la comunidad técnica?

Creemos que las comunidades son uno de los principales dinamizadores de la economía digital, muchas veces un termómetro tecnológico más potente que las grandes firmas de investigación y asesoría. Desde TOKIOTA apoyamos un gran número de eventos en diferentes ciudades durante el año, estando presentes como ponentes o patrocinando, y dando especial soporte a aquellos que se estrenan con sus primeras charlas.

¿Cuál es la visión de futuro en la tecnología de acá a los próximos años?

Pensamos que, aunque nuestro mundo está virando muy rápidamente hacia la digitalización, la automatización, la robotización… esto realmente va de personas. La tecnología nos tiene que ayudar a tomar decisiones, nos tiene que hacer la vida más fácil, nos tiene que ayudar a superar barreras, pero son las personas quienes tienen que gobernar su vida.

52

Microsoft Flow vs Azure Logic Apps

Introducción

La constante evolución de los servicios de Microsoft ha hecho que Flow sea ahora mismo una excelente opción para afrontar tareas que impliquen automatizaciones e integraciones, y no solo para usuarios finales, si no incluso para usuarios de TI.

Entonces, ¿Cuándo debemos usar Azure Logic Apps? Aunque en primera instancia podríamos pensar que ambos servicios comparten multitud de características, únicamente podemos constatar que comparten el mismo motor de ejecución y el mismo editor visual. ¡Nada más! Y si es así… ¿En qué se diferencian?

Perfil del usuario

Mientras que Microsoft Flow está enfocado al Citizen Developer, Azure Logic Apps está destinado a usuarios avanzados o de TI, que usualmente están involucrados en proyectos de mayor complejidad técnica, con lo que se hacen necesarias herramientas de monitorización avanzada, uso de best practices o técnicas de integración continua.

Generalmente para poder usar Microsoft Flow necesitamos una cuenta de Office 365 y/o Dynamics 365 (exceptuando si queremos usar Microsoft Flow Free[1], versión gratuita y de aprendizaje) mientras que para usar Logic Apps necesitamos una cuenta de Azure. De esta manera, se puede comprobar como ya de entrada, el enfoque de ambos productos es totalmente distinto.

Funcionalidad

Precisamente el enfoque de ambos servicios hace que tengamos distintas funcionalidades. En Logic Apps disponemos de características de integración empresarial (por ejemplo, transformar un fichero XML), mientras que en Flow disponemos de algunas funcionalidades más de usuario final (por ejemplo, desencadenar un flujo desde un botón de en una aplicación móvil).

Curiosamente, para un mismo criterio de búsqueda sobre conectores en ambos servicios, obtenemos resultados distintos, como podemos observar en la siguiente figura:

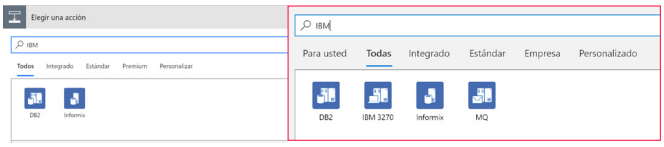


Imagen 1.- Conectores en Flow vs Logic Apps.

En la siguiente tabla se detallan las funcionalidades que podemos encontrar en ambos servicios:

FUNCIONALIDAD	FLOW	LOGIC APPS	OBSERVACIONES
App móvil (Botón)	Sí	No	
Aprobaciones	Sí	No	
Blockchain	Sí	No	
Creación de Flows con Visio	Sí	No	Solo disponible con licencia de Visio Online Plan 2
Ejecución de código Javascript	No	Sí	En fase de prueba
Integración con Forms	Sí	Sí	
Redes Sociales	Sí	Sí	
Sincronización con OneDrive, DropBox, Box, …	Sí	Sí	
Uso de ficheros XML	No	Sí	
Uso de formatos de fichero EDI	No	Sí	

Tabla 1.- Funcionalidades de Flow vs Logic Apps.

Así, podemos concluir que según las necesidades que tengamos en nuestro desarrollo, deberemos elegir una opción u otra, o en el caso que se pueda realizar con ambas, deberemos valorar otros aspectos que veremos a continuación.

Monitorización

Tanto en Flow como en Logic Apps es posible consultar el log de ejecuciones de nuestras apps, aunque el segundo proporciona algunas opciones más para buscar en el histórico y analizarlo.

En la siguiente captura podemos el histórico de ejecución de un flujo concreto en Flow, en el que únicamente podemos filtrar por el resultado de la ejecución, pero no por fechas (para ello deberíamos exportar el histórico a un fichero csv y trabajarlo desde éste):

1 https://flow.microsoft.com/es-es/pricing/

> *"incluido en Office 365 y Dynamics 365, a su vez está basado en Azure Logic Apps, con lo que en determinadas situaciones"*

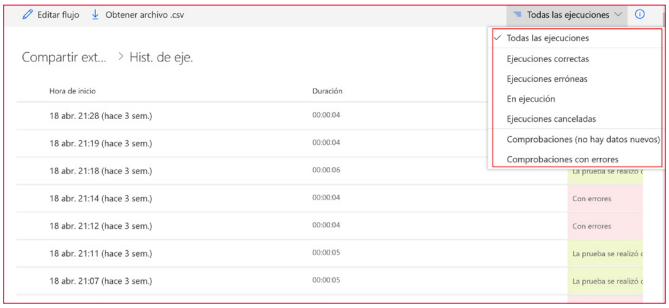


Imagen 2.- Histórico de ejecuciones de una aplicación en Flow.

Si disponemos de una licencia de Flow Plan 2 podemos acceder a un panel de gráficos con datos de uso y análisis sobre nuestras Apps, tal y como podemos ver en la siguiente figura:

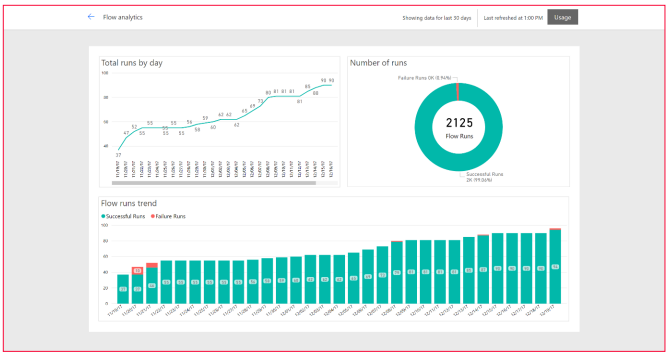


Imagen 3.- Gráfico de análisis sobre una App de Flow.

Por lo contrario, en Logic Apps sí podemos filtrar por fechas e incluso por el identificador de la ejecución, tal y como se puede ver en la siguiente figura:

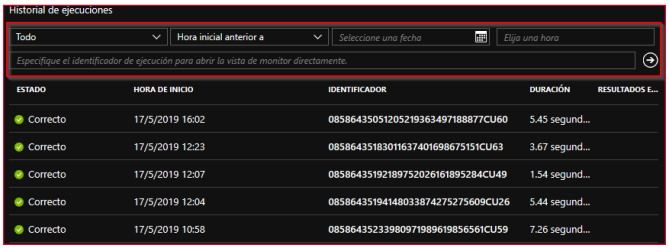


Imagen 4.- Histórico de ejecuciones en una aplicación de Logic Apps.

> *"únicamente podemos constatar que comparten el mismo motor de ejecución y el mismo editor visual"*

Adicionalmente en Logic Apps también podemos consultar información mediante gráficos, como podrían ser el número de acciones facturables o el tiempo medio de ejecución de una aplicación para un determinado periodo de tiempo, como podemos ver seguidamente:

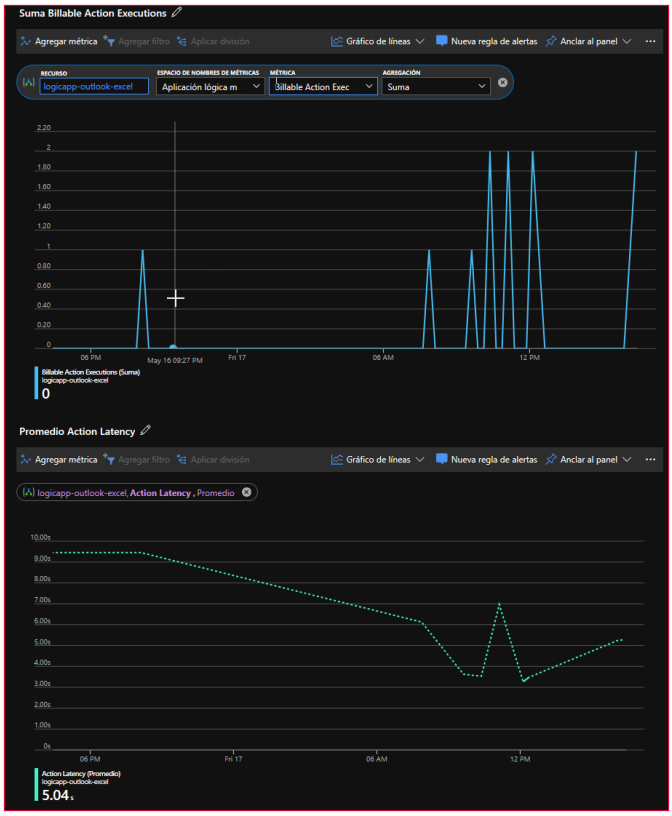


Imagen 5.- Información en formato gráfico en Logic Apps.

Aun así, lo más interesante es que en Azure Logic Apps podemos configurar alertas basadas en distintas métricas de monitorización. Por ejemplo, podríamos crear una alerta para que recibamos un correo cuando el tiempo medio de ejecución correcta de una Logic App específica sea superior a 6 segundos, tal y como podemos ver en la siguiente figura:

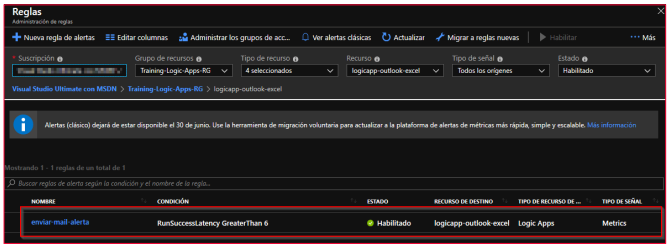


Imagen 6.- Definición de alertas para una Logic App.

Podemos concluir este apartado afirmando rotundamente que las posibilidades de monitorización y análisis son mucho mejores en Logic Apps respecto a Flow, con lo qué es un aspecto de peso que también deberemos tener en cuenta al elegir una opción u otra.

Experiencia de desarrollo y programación

Aunque en ambos servicios podemos trabajar con un editor visual mediante un navegador web, en el caso de Azure Logic Apps tenemos la opción de trabajar con Visual Studio. Concretamente podemos importar cualquier Logic App en un proyecto de tipo Azure Resource Groups en Visual Studio, lo que nos permite realizar tareas como control de código fuente e integración continua.

Si tenemos instalado en nuestro equipo las Azure Logic Apps Tools for Visual Studio[2] podemos conectarnos a nuestra suscripción de Azure y ver cualquiera de nuestras Logic Apps desde el editor, tal y como podemos ver en la siguiente figura:

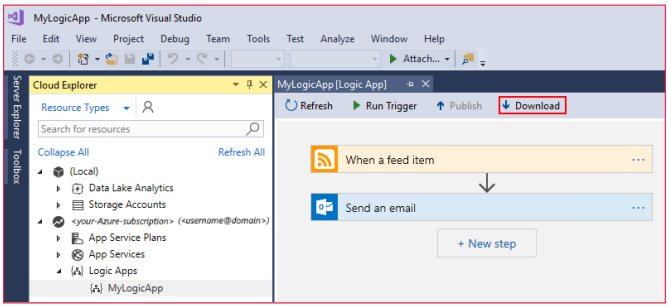


Imagen 7.- Editor visual de Logic Apps en Visual Studio.

Desde el propio editor también podríamos ejecutar la App, habilitarla o deshabilitarla, ver el histórico de ejecuciones o ver y modificar el código fuente de la misma.

Imagen 8.- Edición de una Logic App con Visual Studio.

Así, vemos como las posibilidades de Logic Apps en cuanto a experiencia de desarrollo son también totalmente distintas respecto a Flow.

Precios

Las características de facturación de ambos servicios son totalmente diferentes, haciendo que sea un factor de decisión con un peso importante. En Flow el modelo de facturación se basa en el numero de ejecuciones de nuestros flows, independientemente de las acciones que incluyan.

Imagen 9.- Planes de Flow.

Por otra parte, en Logic Apps el modelo de facturación se basa en las acciones que se ejecutan en cada App. Por lo tanto, se basa en un modelo totalmente granular, haciendo que apps simples puedan ser muy económicas, mientras que las más complejas puedan tener unos costes más altos.

2 *https://github.com/MicrosoftDocs/azure-docs/blob/master/articles/logic-apps/manage-logic-apps-with-visual-studio.md*

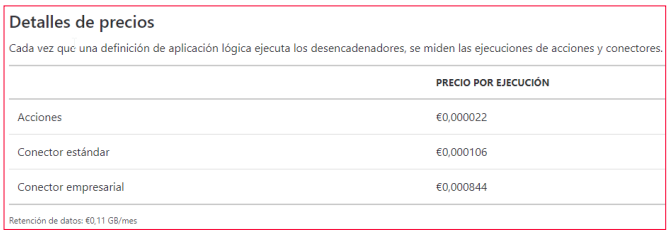


Imagen 10.- Costes de Logic Apps.

En Logic Apps también deberíamos contabilizar si usamos cuentas de integración (procesamiento de EDI y XML) o entornos de servicio de integración (entornos aislados para conectarnos de forma segura a aplicaciones locales).

Por lo tanto, si estamos trabajando con Logic Apps recomendaríamos la creación de Apps pequeñas y granulares, de forma que se pudiesen reutilizar las más veces posibles, ya que eso no afectaría al coste. Justamente en Flow seria todo lo contrario, ya que, si un Flow llama a otro, son 2 ejecuciones que se contabilizan.

Escenarios

A partir de lo que hemos visto en los apartados anteriores, podríamos resumir el uso de un servicio u otro en los siguientes puntos:

- Flow:
  - Tenemos una suscripción de Office 365 y/o Dynamics 365, pero no de Azure.
  - Necesitamos poder usar flujos de aprobación.
  - Necesitamos poder ejecutar una app desde el teléfono móvil.
  - No necesitamos usar acciones complejas de integración (EDI o XML).
  - No tenemos conocimiento de desarrollo o éste es muy básico.
  - No necesitamos control de código fuente y versionado.
  - No necesitamos monitorización avanzada.
- Logic Apps:
  - Tenemos una suscripción de Azure.
  - Queremos usar una experiencia de desarrollo con Visual Studio.
  - Necesitamos desarrollar Apps con cierta complejidad, así como trabajar con EDI o XML.
  - Necesitamos control sobre el código fuente y las versiones de nuestra App.
  - Necesitamos capacidades avanzadas de monitorización.

Conviene tener presente que cualquier aplicación desarrollad con Flow siempre se puede exportar a Logic Apps, con lo que la elección del servicio más básico no supone la exclusión del más avanzado.

Conclusión

En este artículo hemos visto las capacidades que nos ofrecen Microsoft Flow y Azure Logic Apps para desarrollar procesos de automatización de tareas e integración de datos. En ambos casos hemos visto las ventajas e inconvenientes que presentan, y, en definitiva, qué escenarios son los ideales para su uso.

Jugando a ser adivinos, creemos que probablemente, y en un futuro no muy lejano, Microsoft Flow incorporará nuevas funcionalidades o mejoras sobre las que ya están incluidas en Logic Apps, aunque quedará en las manos de Microsoft decidir qué tipo de funcionalidad es fácilmente usable para un Citizen User y cual no.

FERRAN CHOPO GARCIA
IT Consultant & Trainer
ferran@ferranchopo.com
@fchopo
http://www.ferranchopo.com

56

Dataflows – Desde lo básico, y más allá

Desde hace unos meses, y encontrándose en etapa preliminar apareció en el servicio online de Power BI la capacidad de Dataflows. Hoy en día ya contamos con la disponibilidad general de esta tecnología en el servicio nube de Power BI.

Para adentrarnos en el tema comencemos con los conceptos más importantes detrás de esta tecnología.

Se trata de brindarle al usuario de negocio la capacidad de conectar directamente a sus fuentes de datos frecuentemente utilizados, permitiendo no solo extraer información desde ellas sino vincularla a otros sistemas, y más importante aún, contar con la capacidad de transformación, limpieza y manipulación sin necesitar una herramienta de escritorio para lograr dicha tarea.

"Uno de los pilares detrás de Dataflows es también la conexión a Azure Data Lake Gen2 con capacidad de storage"

Uno de los pilares detrás de Dataflows es también la conexión a Azure Data Lake Gen2 con capacidad de storage. Esta capacidad la tenemos tanto utilizando una cuenta Power BI PRO como la capacidad Premium de Power BI (en la que contaremos además con refrescos incrementales). A su vez como dato importante, también contamos con la posibilidad de contar con la capacidad de conectar con nuestro propio Azure Data Lake Gen2 utilizando nuestra propia suscripción de Azure.

Arquitectura de Dataflows

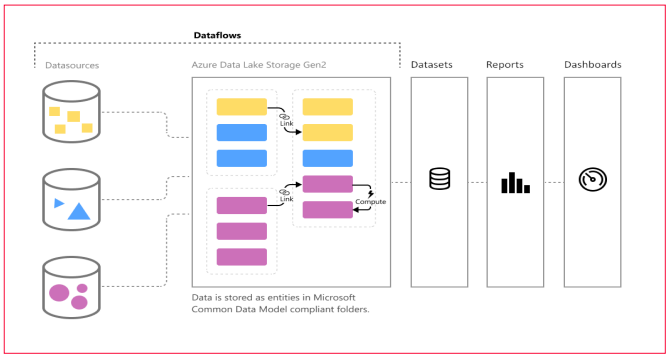


Imagen 1.- Arquitectura de Dataflows.

El concepto de Common Data Model no es menor ya que el roadmap trazado por Microsoft permitirá a todo el stack de lo que denominamos Power Platform compartir un storage en común, y asimismo entidades que podamos utilizar para compartir información entre sistemas. Esto permite un salto muy importante al momento de integrar información desde Dynamics 365, Office 365, Power BI, PowerApps, Flow.

A su vez el concepto de entidades no solo permite conexión entre sistemas sino realizar modificaciones sobre modelos de datos en formato JSON, así como cargar metadatos sobre entidades de nuestros Dataflows

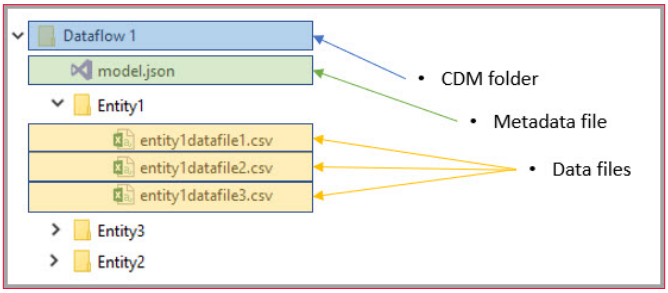


Imagen 2.- Estructura de un Dataflow.

Para comenzar con la implementación de un Dataflow lo que necesitamos es acceder directamente al servicio de online de Power BI, a través de https://app.powerbi.com con nuestras credenciales. Accedemos a un espacio de trabajo al que pertenezcamos, y desde la opción Crear en el punto superior derecho de la ventana podremos dar clic en Crear un nuevo Dataflow:

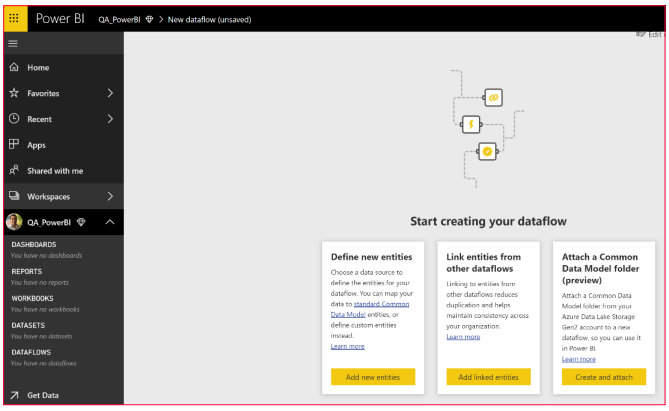


Imagen 3.- Acceso al espacio de trabajo en Power BI.

En esta ventana como podemos ver contamos con la capacidad de crear nuevas entidades a nuestro Dataflows,

generar un vinculo a otras entidades provenientes de otros Dataflows que hayamos generado, así como también incorporar un CDM (Common Data Model) proveniente de nuestra propia suscripción de Azure a través de nuestro propio Azure Data Lake Gen2.

Las fuentes de datos a las que podemos acceder desde Dataflows es muy variada, y no se concentra solo en fuentes Online, sino que además cuenta con la capacidad de conexión a fuentes On Premises (como SQL Server, Oracle, IBM DB2, Teradata, entre otras). No olvidemos que en caso de querer contar con la capacidad de refresco automático debemos configurar un Enterprise Gateway que cumpla con dicho propósito.

Fuentes de Datos (hoy en día… siguen sumándose cada vez más con el tiempo)

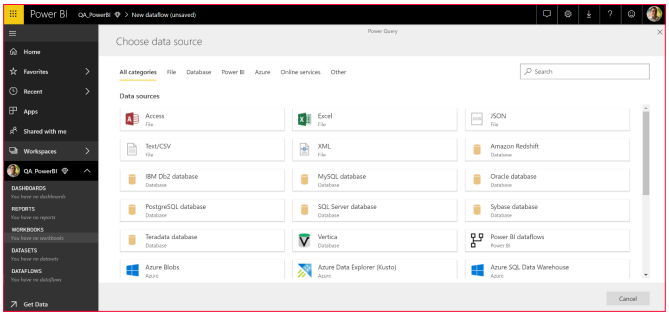


Imagen 4.- Orígenes de datos en Power BI.

Una vez conectamos con nuestro/s sistemas generando las entidades en Dataflows contamos con la capacidad de utilizar el lenguaje Power Query en toda su plenitud para realizar las transformaciones, limpieza y manejo de información necesaria para generar nuestro modelo de datos. Aquí es donde viene una de las preguntas mas frecuentes al momento de realizar entrenamientos. ¿Es Dataflows un sistema que reemplaza a los Datasets en Power BI? No.

Dataflows se utiliza como paso previo a la generación de un dataset. Dataflows nos ayuda al momento de la conexión a fuentes, pero no es dataflow el encargado de contener nuestro modelo de datos.

"Common Data Model no es menor ya que el roadmap trazado por Microsoft"

En el siguiente ejemplo podemos ver un dataflow generado a través de una conexión a una Web API de Azure DevOps para obtener información de varios endpoints de dicha API:

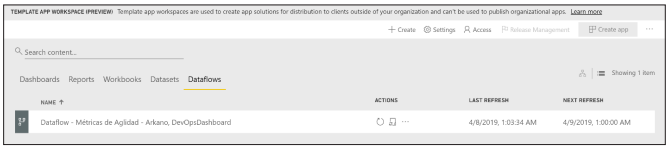


Imagen 5.- Conexión a una WebAPI de Azure DevOps.

Las entidades en este ejemplo son:

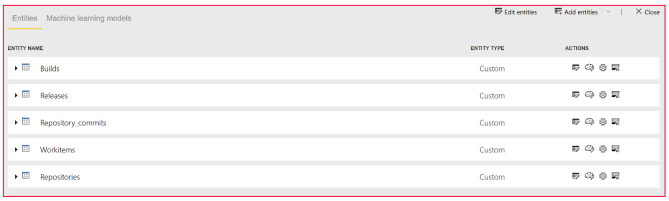


Imagen 6.- Entidades de Ejemplo.

Si editamos una de estas entidades accederemos al lenguaje Power Query que nos mostrara en cada entidad los pasos que se dieron para lograr el resultado final. Sin dudas que, si el caso nos lo permite, tendremos mucho mas simple la tarea utilizando la interfaz grafica de Power Query, pero si lo que buscamos es potencia, y transformaciones complejas, el editor avanzado de Power Query nos provee de una capacidad prácticamente ilimitada para el manejo de datos.

Interfaz Power Query (edición de entidades en un dataflow)

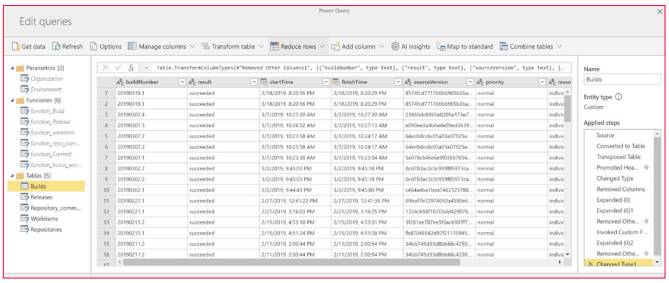


Imagen 7.- Interfaz de Power Query.

Editor Avanzado de Power Query

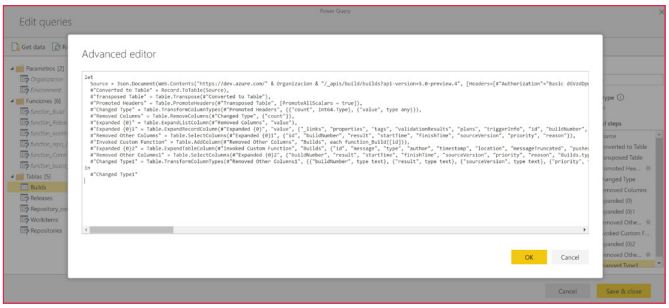


Imagen 8.- Editor avanzado de Power Query.

Una vez que tenemos preparadas las fuentes de información, y hemos transformado los datos, y consecuentemente hemos realizado una limpieza de los mismos, el dataflow genera automáticamente un dataset (modelo de datos) que nos permitirá a posteriori generar reportes ya sea utilizando el servicio nube de Power BI como también conectarnos desde nuestra herramienta de escritorio: Power BI Desktop.

A continuación, mostramos que desde Power BI Desktop, con nuestra opción de Obtener Datos se nos da la posibilidad ahora de conexión directo a dataflows:

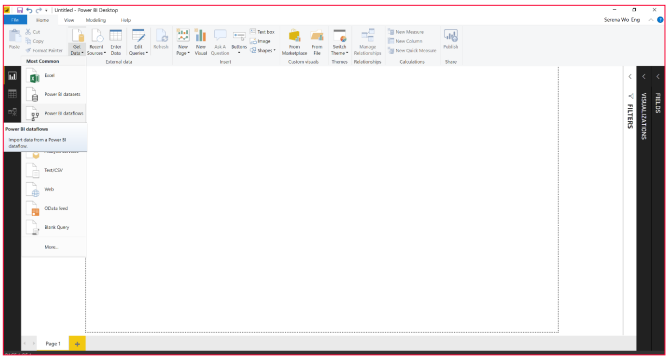


Imagen 8.- Conexión directa a Dataflows en Power BI.

AI y más…

Sin lugar a duda, dataflows está en los primeros pasos, pero lo vemos como una tecnología con amplias posibilidades, y que sigue agregando adeptos con el tiempo. De hecho, hace muy poco tiempo contamos con la posibilidad de utilizar Servicios Cognitivos de Azure desde dataflows para realizar un análisis de sentimientos a través del análisis de un campo de feedback de clientes, así como también el uso de reconocimiento de imágenes para agregar metadata sobre un campo que contenga la imagen de uno de nuestros productos (como ejemplo). Y esto sin tener la obligación de contar con una suscripción de Azure.

Y esto no queda en estos ejemplos, sino que también podemos escalar aún más, utilizando directamente modelos de Machine Learning creados por nuestro equipo de DataScientists y realizar análisis de nuestras entidades con la finalidad de hacer predicciones, o manejar forecastings.

Sigamos expectantes de esta tecnología que viene avanzando a pasos agigantados en el mundo del análisis de datos.

¡Hasta pronto!

GASTON CRUZ
Microsoft MVP Dataplatform
Slalom LLC - Seattle

Alberto Diaz

Alberto Díaz es SharePoint Team Lead en ENCAMINA, liderando el desarrollo de software con tecnología Microsoft.

Para la comunidad, ha fundado TenerifeDev (www.tenerifedev.com) con otros colaboradores, un grupo de usuarios de .NET en Tenerife, y coordinador de SUGES (Grupo de Usuarios de SharePoint de España, www.suges.es) y colaborador con otras comunidades de usuarios. Microsoft MVP de SharePoint Server desde el año 2011 y asiduo conferenciante en webcast y conferencias de tecnología de habla hispana.

Sitio Web: http://blogs.encamina.com/negocios-sharepoint/
Email: adiazcan@hotmail.com
Blogs: http://geeks.ms/blogs/adiazmartin
Twitter: @adiazcan

Fabián Imaz

Fabián Imaz, MVP de SharePoint Server trabaja en el mundo del desarrollo de software desde hace más de 10 años, teniendo la suerte de trabajar en distintas arquitecturas y tecnologías Microsoft. Pertenece a la firma Siderys, http://www.siderys.com empresa de desarrollo de Software especializada en SharePoint 2007/2010/2013 y en desarrollo de soluciones inteligentes.

Desde los comienzos Fabián ha trabajado en distintitas comunidades donde organiza y promueve eventos locales para la difusión de tecnología dentro de los miembros de las mismas. Es director de la carrera SharePoint 2010 y SharePoint 2013 en Microsoft Virtual Academy, http://www.mslatam.com/latam/technet/mva2/ Home.aspx y cuenta con un sitio en CodePlex con varios desarrollos http://siderys.codeplex.com.

Sitio Web: http://www.siderys.com
Email: fabiani@siderys.com.uy
Blogs: http://blog.siderys.com
Twitter: @fabianimaz

Gustavo Velez

Gustavo Velez es Ingeniero Mecánico y Electrónico; trabaja en la arquitectura, diseño e implementación de sistemas de IT basados en tecnologías de Microsoft, especialmente SharePoint, Office 365 y Azure.

Propietario del sitio especializado en información sobre SharePoint en español http://www. gavd.net, autor de ocho libros sobre SharePoint y sus tecnologías y numerosos articulos y conferencias sobre el tema.

Sitio Web: http://www.gavd.net
Email: gustavo@gavd.net
Blogs: http://geeks.ms/blogs/gvelez/

Juan Carlos González Martín

Ingeniero de Telecomunicaciones por la Universidad de Valladolid y Diplomado en Ciencias Empresariales por la Universidad Oberta de Catalunya (UOC). Cuenta con más de 14 años de experiencia en tecnologías y plataformas de Microsoft diversas (SQL Server, Visual Studio, .NET Framework, etc.), aunque su trabajo diario gira en torno a las plataformas SharePoint & Office 365. Juan Carlos es MVP de Office Apps & Services y co-fundador del Grupo de Usuarios de SharePoint de España (SUGES, www.suges.es), delGrupo de Usuarios de Cloud Computing de España (CLOUDES) y de la Comunidad de Office 365. Hasta la fecha, ha publicado 11 libros sobre SharePoint & Office 365, así como varios artículos encastellano y en inglés sobre ambas plataformas.

Email: jcgonzalezmartin1978@hotmail.com
Blogs: http://geeks.ms/blogs/jcgonzalez &
http://jcgonzalezmartin.wordpress.com/

Santiago Porras

Innovation Team Leader en ENCAMINA, lidera el desarrollo de productos mediante tecnologías Microsoft. Se declara un apasionado de la tecnología, destacando el desarrollo para dispositivos móviles y web, donde ya cuenta con 16 años de experiencia.

Microsoft MVP in Developer Technologies, colabora con las comunidades de desarrolladores desde su blog personal **http://geeks.ms/santypr** y ocasionalmente en **CompartiMOSS.com**. Además, es uno de los coordinadores de TenerifeDev, grupo de usuarios de .NET en Tenerife (**http://www.tenerifedev.com**)

Sitio Web: http://www.santiagoporras.es
Email: santiagoporras@outlook.com
Blogs: http://geeks.ms/santypr
Twitter: @saintwukong

Coordinadores de sección

GASTÓN CRUZ
Coordinador de PowerBi
gastoncruz@gmail.com

¿Desea colaborar con CompartiMOSS?

La subsistencia del magazine depende de los aportes en contenido de todos. Por ser una revista dedicada a información sobre tecnologías de Microsoft en español, todo el contenido deberá ser directamente relacionado con Microsoft y escrito en castellano. No hay limitaciones sobre el tipo de articulo o contenido, lo mismo que sobre el tipo de tecnología.
Si desea publicar algo, por favor, utilice uno de los siguientes formatos:

- Artículos de fondo: tratan sobre un tema en profundidad. Normalmente entre 2000 y 3000 palabras y alrededor de 4 o 5 figuras. El tema puede ser puramente técnico, tanto de programación como sobre infraestructura, o sobre implementación o utilización.

- Artículos cortos: Artículos cortos: Máximo 1000 palabras y 1 o 2 figuras. Describen rápidamente una aplicación especial de alguna tecnología de Microsoft, o explica algún punto poco conocido o tratado. Experiencias de aplicación en empresas o instituciones puede ser un tipo de artículo ideal en esta categoría.

- Ideas, tips y trucos: Algunos cientos de palabras máximo. Experiencias sobre la utilización de tecnologías de Microsoft, problemas encontrados y como solucionarlos, ideas y trucos de utilización, etc. Los formatos son para darle una idea sobre cómo organizar su información, y son una manera para que los editores le den forma al magazine, pero no son obligatorios. Los artículos deben ser enviados en formato Word (.doc o .docx) con el nombre del autor y del artículo.

Si desea escribir un artículo de fondo o corto, preferiblemente envíe una proposición antes de escribirlo, indicando el tema, aproximada longitud y número de figuras. De esta manera evitaremos temas repetidos y permitirá planear el contenido de una forma efectiva.

Envíe sus proposiciones, artículos, ideas y comentarios a la siguiente dirección:
revista@compartimoss.com adiazcan@hotmail.com
fabiani@siderys.com.uy jcgonzalezmartin1978@hotmail.com
gustavo@gavd.net


Comparti
MOSS