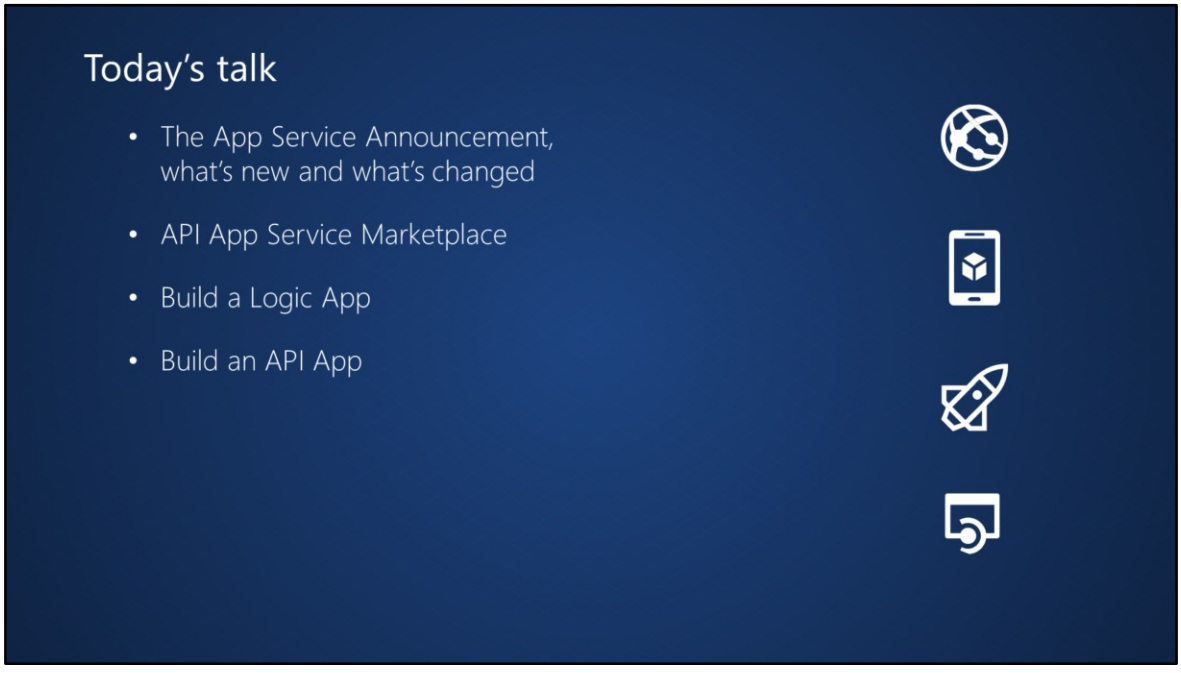
Azure App Service

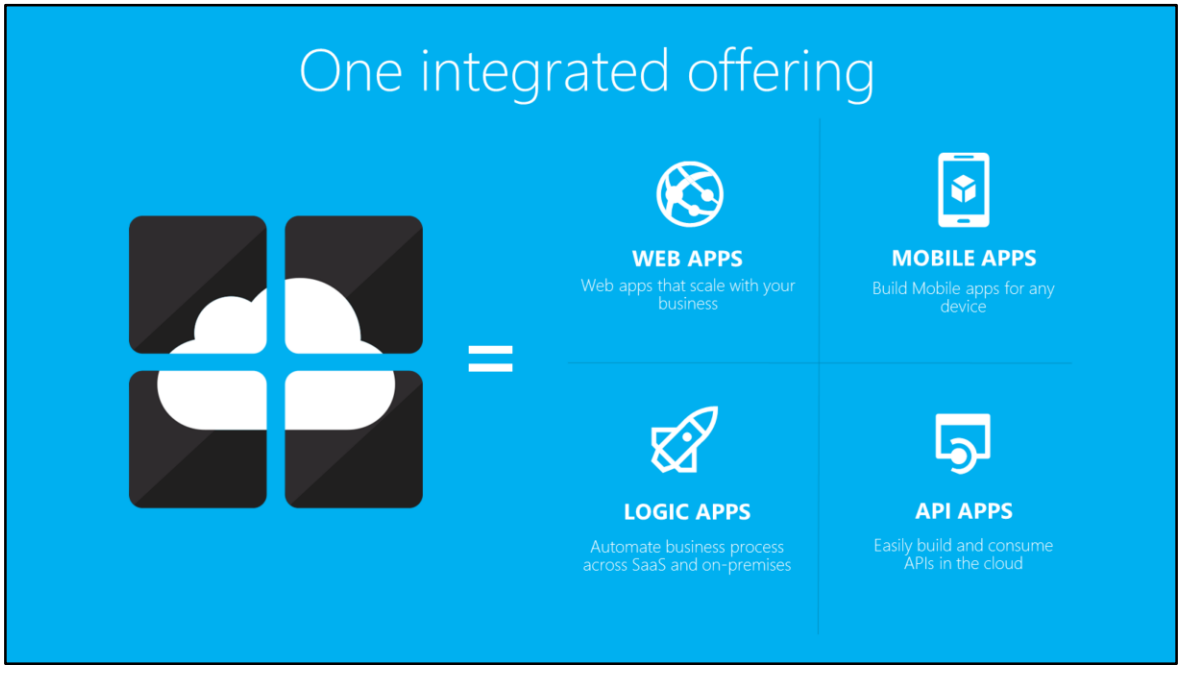
@DanielLarsenNZ

- Good afternoon everyone, my name is Daniel Larsen. I work for Fraedom, the Expense management software company formerly known as Spendvision, where I am a DevOps Practice Lead.
- Today I am very happy to be talking to you about Azure App Service, Microsoft's new Web and Mobile app PaaS offering on Azure.



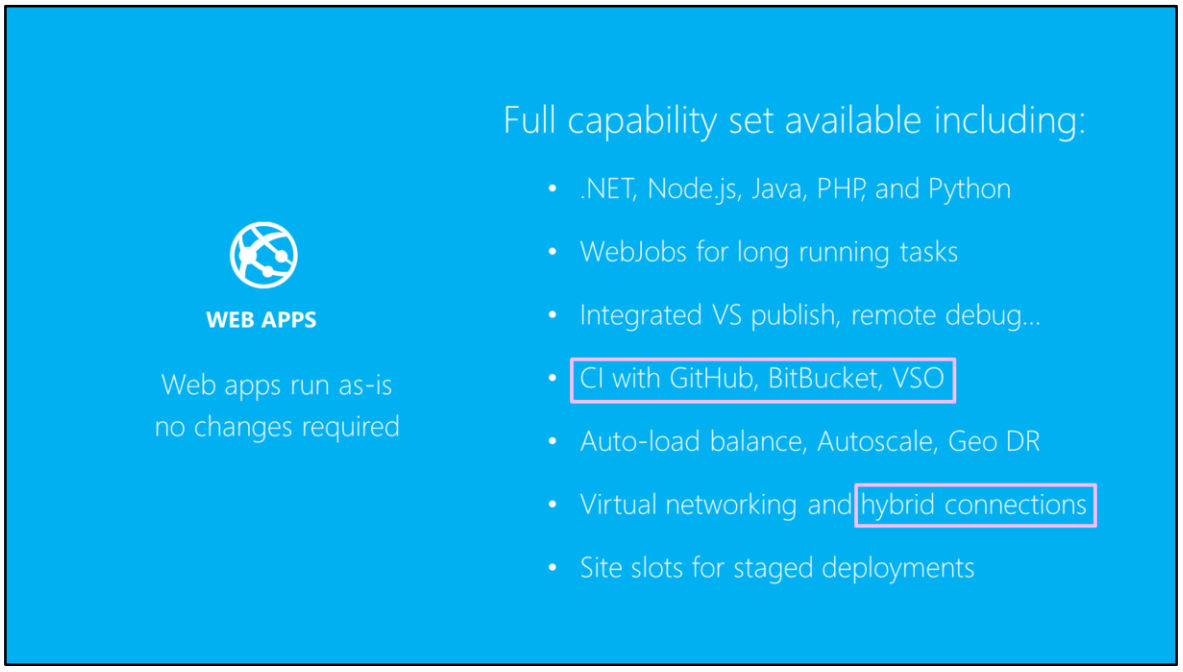
- In Today's talk I will be moving very quickly through the announcement; talking about the new Azure service types, and how the old ones have changed to fit the new App Service concept.
- I will take you on a tour of the API App Service Marketplace and discuss what this means for Integrators.
- We will take a moment to build a simple Logic App from scratch so that you can see what is involved.
- And finally we will build our own API App to incorporate in to our Logic App workflow.
- That's a lot to cover so I will talk fast – yell out if you would like me to stop and drill in to any topic at any stage.

- OK, so chances are (since you are at an Azure Meetup) that you are familiar with one or more of these services.
- I have been working with Azure Websites for more than a year and I love the simplicity of deployment and management, as well as the power of its scale-out model and test-on-prod features.
- Mobile Services and BizTalk Services have been PaaS offerings in their own right for some time now.
- But the challenge has been that these three services have been quite isolated from each other in several ways including affinity and also pricing.
- Integrating these services has felt a bit clunky and does not spring to mind as a viable option.
- There have been improvements over the past year, but now the new Azure App Service is all about bring these three services as first-class citizens of a PaaS ecosystem.

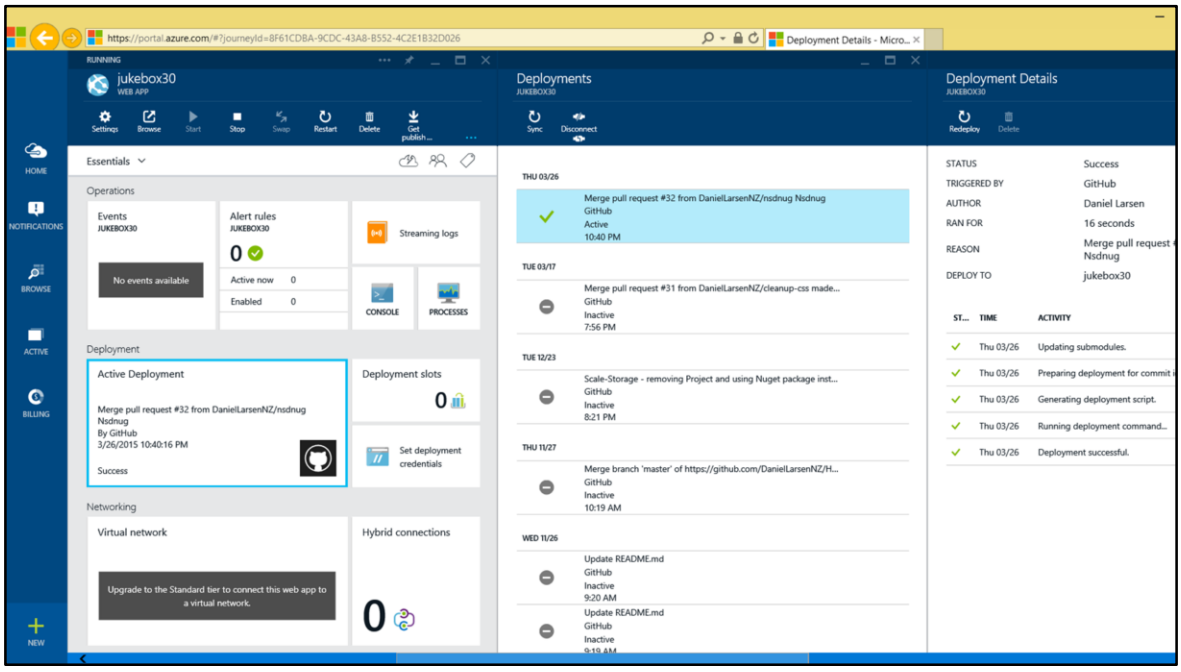


- So now Azure Websites becomes “Web Apps”, Mobile Services becomes “Mobile Apps”, “Logic Apps” takes over the Workflow previously provided by BizTalk Services, with API Apps as a pluggable LOB Connectors that can be authored in several different languages and shared with the community via a Marketplace.
- All of this adds up to a new integrated PaaS offering, standing on the shoulders of Azure Websites, which reinforces PaaS as a major play for Microsoft Azure.

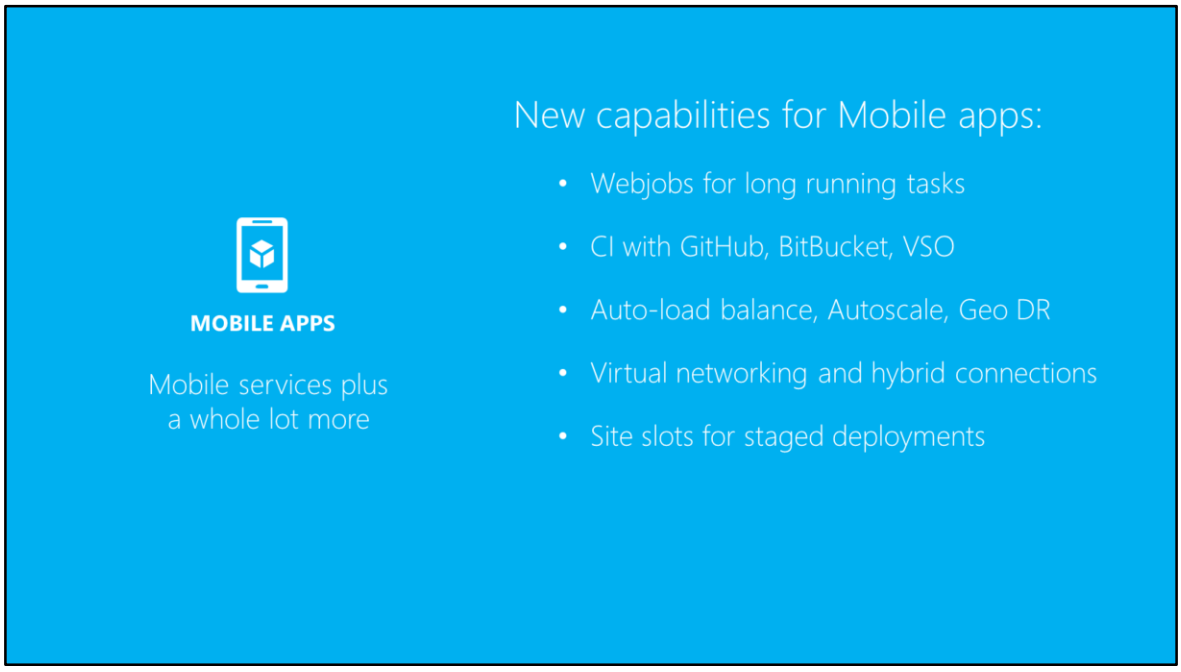
- I’ve spent a couple of weeks digging in to the announcement and under the hood of the new offering and can confirm, this is not just a name change.
- As a Developer I see this as a paradigm shift for the way we think about building Apps that is following Lean, Continuous Delivery trends.
- Let’s drill into each of the services one by one.



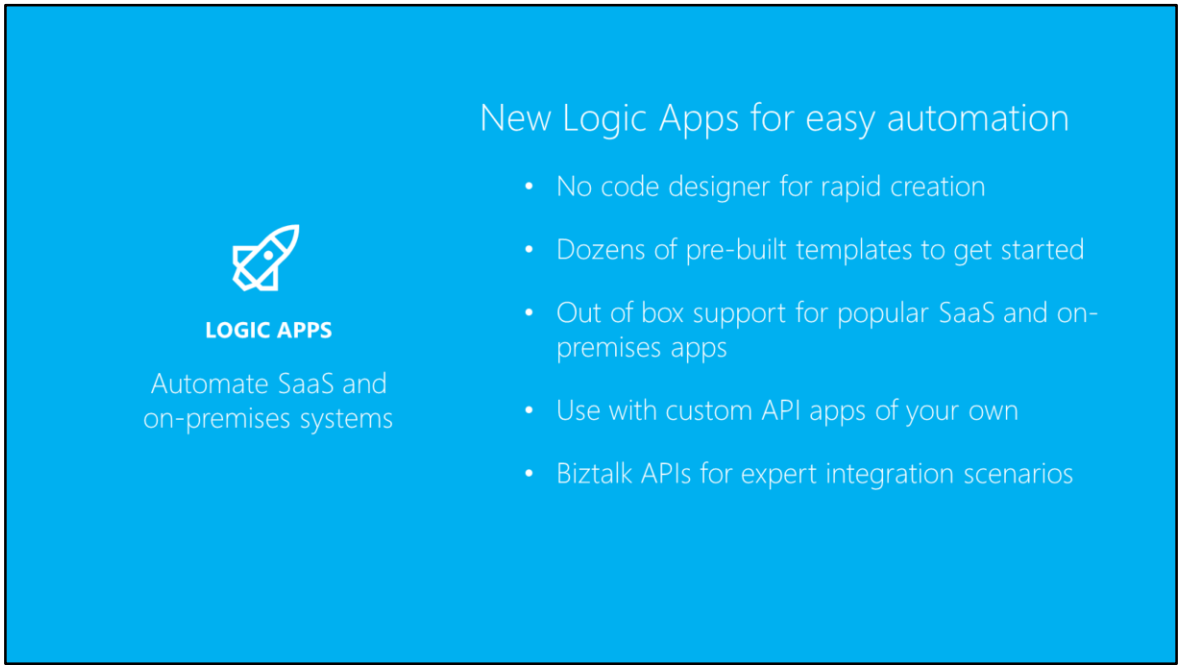
- Web Apps keep all of the announcements from the past year or so including support for Node.js, Java, PHP, Python.
- The New (Preview) Portal gets a major upgrade and deserves a shoutout – especially from those of us deploying from GitHub: GitHub and BitBucket finally join VSO as first-class deployment citizens in the new portal.
- The other exciting announcement (which I have not had time to investigate further) is Hybrid Connections to on-premise resources (like on-prem SQL Server for example).



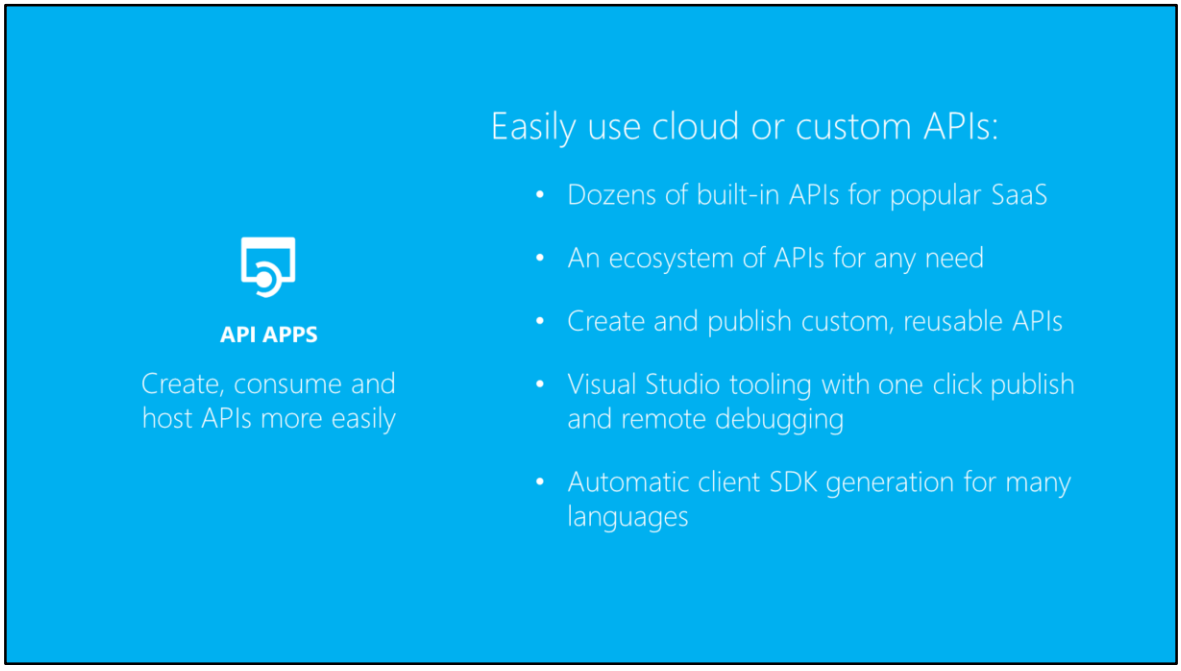
Here's a quick shot of the New Portal with GitHub deployment history and details.



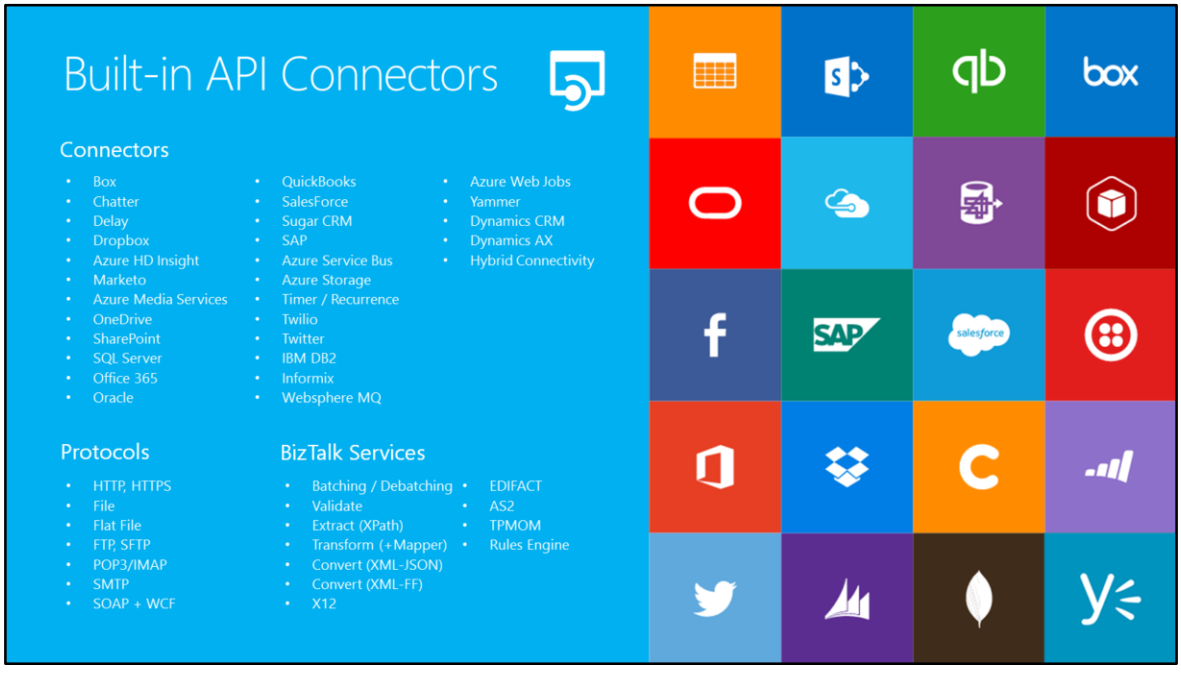
- TBH I have not spent any time in Mobile Services or with the New Mobile Apps, but I am told that Mobile Apps is a new beast.
- Mobile Apps' code runs on a Web App, and you have full control over the web app and how it operates, and features which were previously unavailable to Mobile Services, such as Traffic Manager and Deployment Slots, can now be used.
- Because you have more control over the app, any version of any NuGet package can now be deployed without worry about dependency conflicts.
- When you create a Mobile App you get a Mobile App resource (which includes some new features like SaaS API connectors, etc), a Mobile App Code site to run your Web API project using the Mobile App Server SDK, and an App Service Gateway to handle logic and assists with adding App Service API Apps to your application.



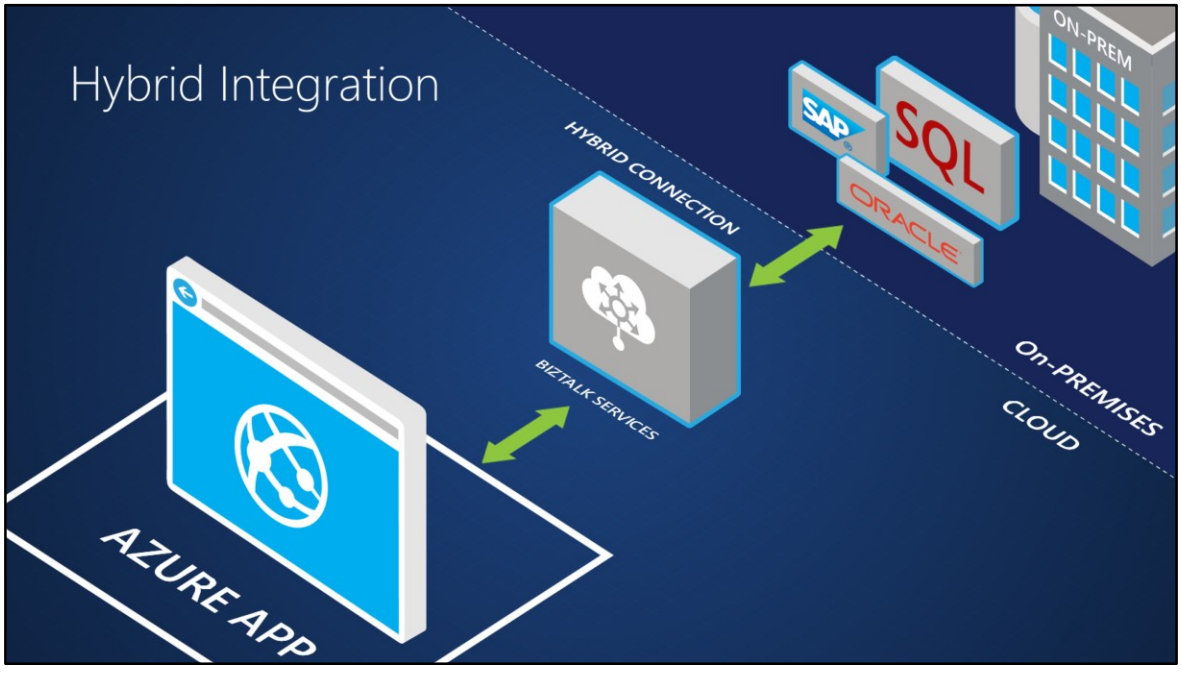
Logic Apps are new. They are easiest explained as a show and tell, so I will open the box shortly. But think “Lego” for integrators.



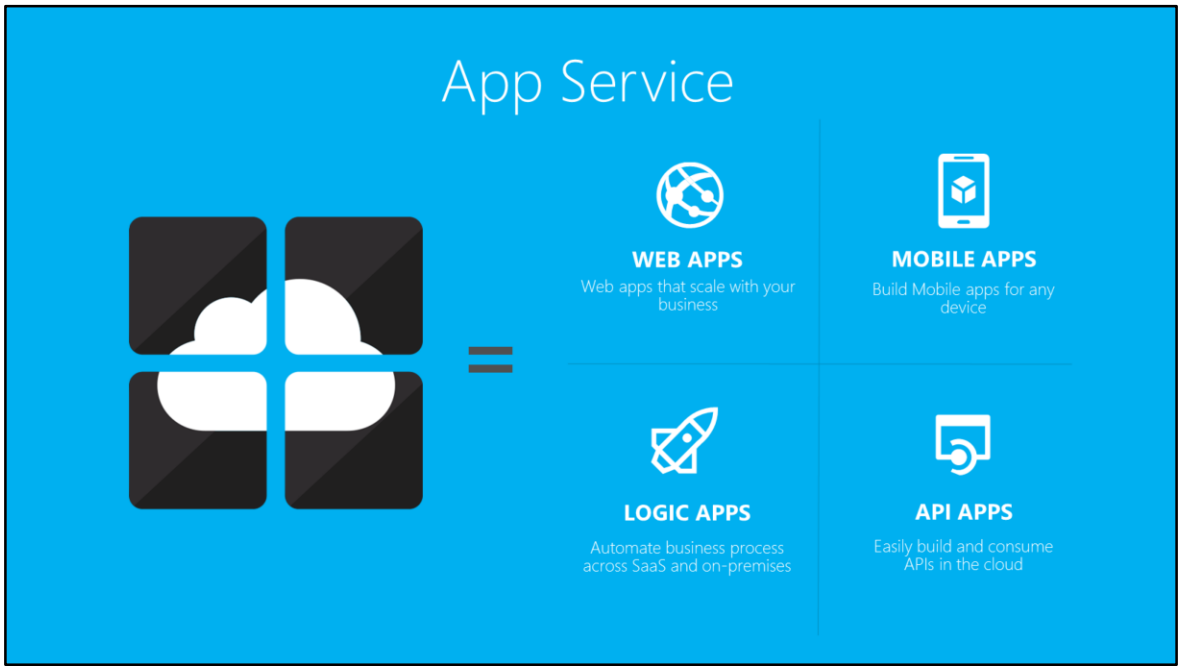
- And if Logic Apps are Lego then API Apps are the building blocks.
- Each one is a Web API App which means you can easily create your own from scratch or from existing code.
- The Azure Marketplace now includes pre-built API Apps that you can pick and choose for your own custom (Logic App) workflows.
- Let's take a look:



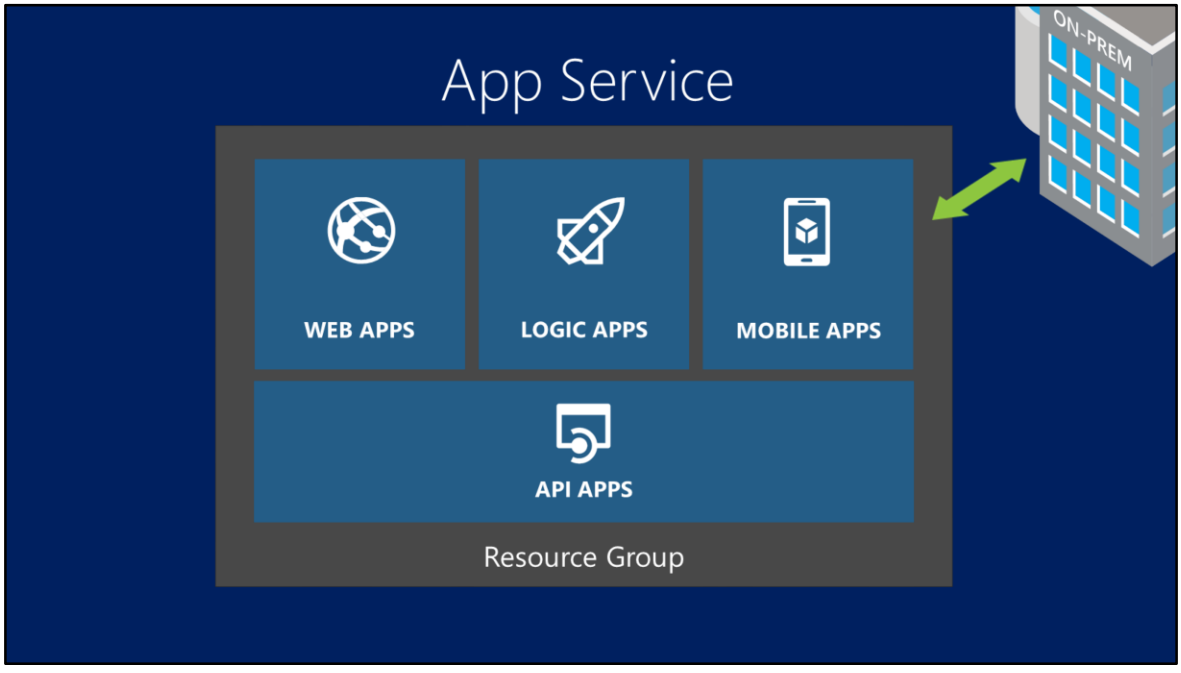
- Over 40 connectors and growing all the time with the ability for any 3rd party to contribute.
- This is a fascinating aspect of the Announcement for mine which deserves special attention, so we will build a Logic App in a minute.



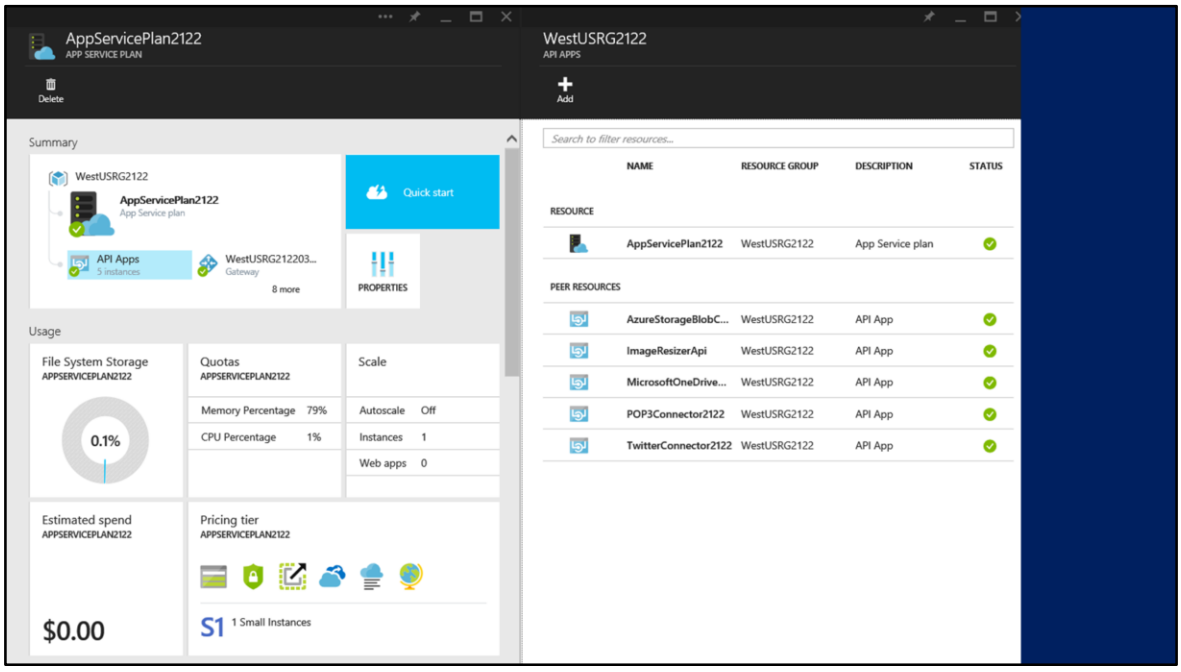
- Hybrid connections are interesting. Apparently these have been available via BizTalk Services for some time but now App Services have them too.
- This is going to unlock some interesting scenarios for integrating with a System of Record or other on premise systems that don't make sense to move to the cloud.



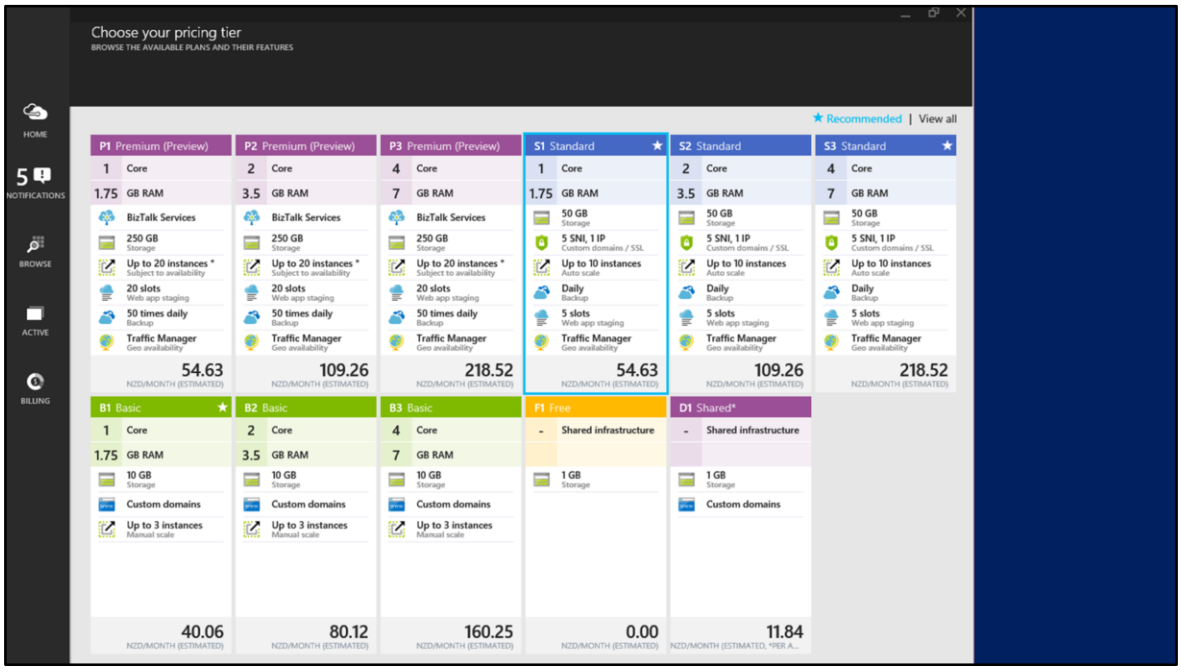
- So there we have it, the four App types that make up the new App Service.
- Are there any questions before I go under the hood?



- OK, this is one of Hanselman's slides which I have borrowed and extended. He sees Web Apps as logically building on API Apps which is an interesting way of thinking of them. This is almost a micro-services architectural pattern where API services underlie everything.
- Another important aspect is that instances of these App Service Apps all co-exist in a Resource Group which mean low latency between the Web, Logic and Mobile Apps and the API Apps that they build on.
- Resource Group is a logical group for networking which means Virtual Networks and Hybrid Connections extend off of this container.



- All Apps in your App Service exist in the same App Service Plan too. Remember Web Service plans for Azure Websites? These have been renamed to App Service Plans to which all App Service Apps belong.
- This is a fundamental shift in the PaaS pricing model in your favour – each Web, Mobile, Logic and API App instance counts towards your Plan's total instances.
    - Previously Mobile Services and BizTalk Services would have been charged separately.



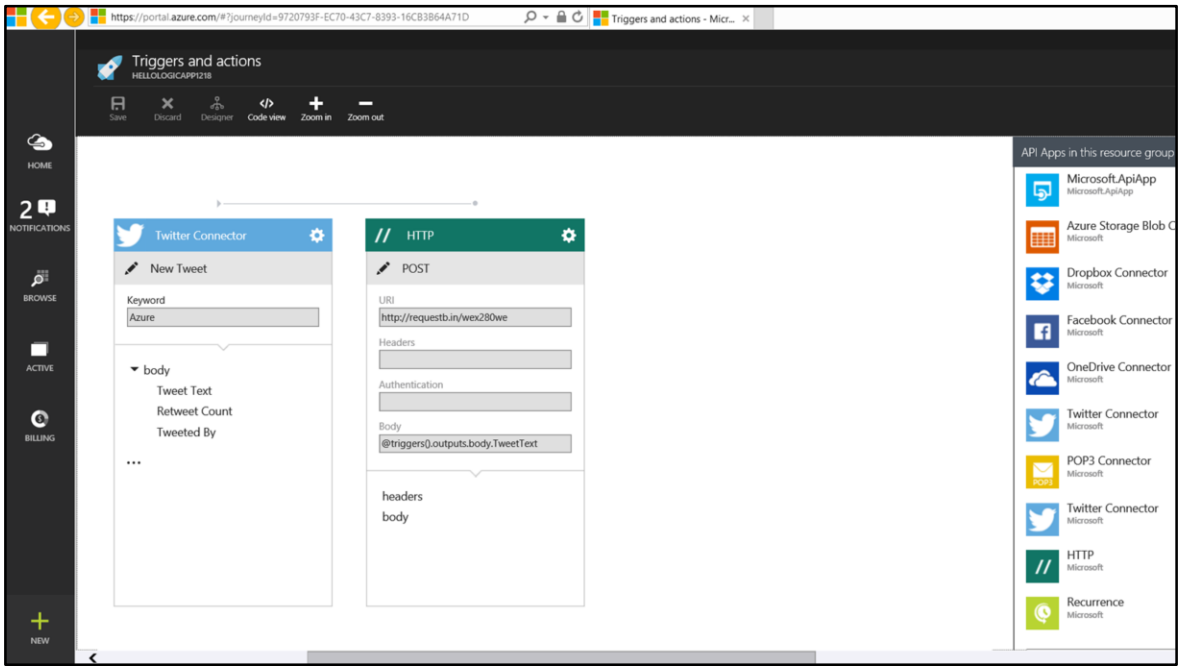
- For my demos I am using an S1 Standard Plan which supports up to 10 instances of which I am using 5. If I need more power I can turn up the Cores. I can also auto-scale my Apps as required.
- At $55 NZD per month I think this represents pretty good value.

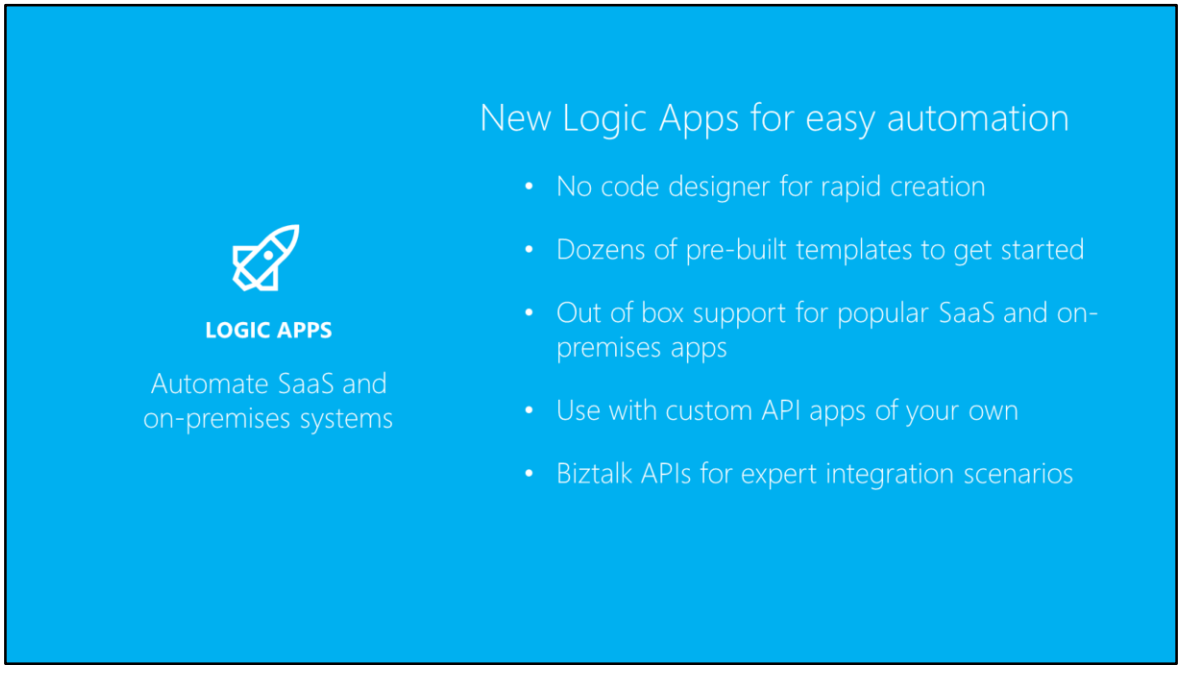
OK let's build a Logic App.

1. Marketplace -> API Apps
    1. (browse the offerings)
    2. Choose Twitter and create it
2. New > Web + Mobile > Logic App
    1. Choose Twitter App
    2. Click Authorize
    3. New Tweet Trigger
    4. Keyword @@DanielLarsenNZ
    5. Choose HTTP Action
    6. POST
    7. http://requestb.in/
    8. Body Tweet Text
    9. OK
    10. Create

    You get the idea…

    1. Edit in Code View

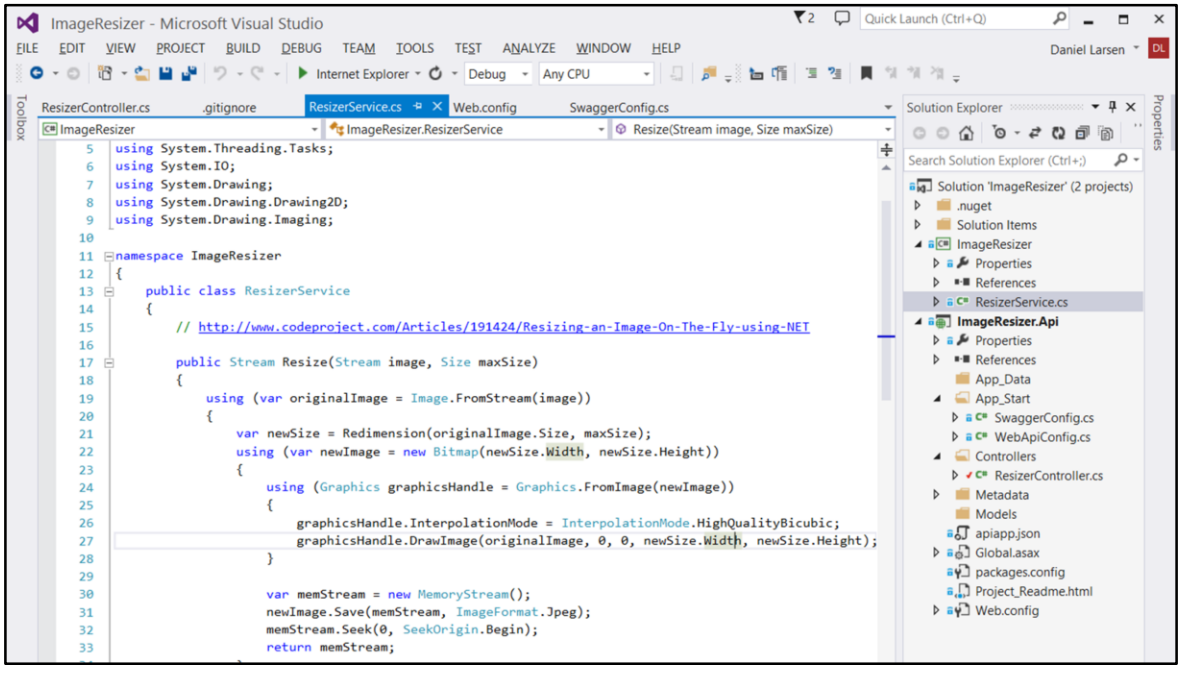

https://portal.azure.com/#?journeyId=9720793F-EC70-43C7-8393-16CB3B64A71D
Triggers and actions - Micr...
Triggers and actions
HELLOLOGICAPP1218
Save
Discard
Designer
Code view
Zoom in
Zoom out
HOME
2
NOTIFICATIONS
BROWSE
ACTIVE
BILLING
NEW
Twitter Connector
New Tweet
Keyword
Azure
body
Tweet Text
Retweet Count
Tweeted By
HTTP
POST
URI
http://requestb.in/wex280we
Headers
Authentication
Body
@triggers().outputs.body.TweetText
headers
body
API Apps in this resource group
Microsoft.ApiApp
Microsoft
Azure Storage Blob C
Dropbox Connector
Facebook Connector
OneDrive Connector
Twitter Connector
POP3 Connector
Twitter Connector
HTTP
Recurrence

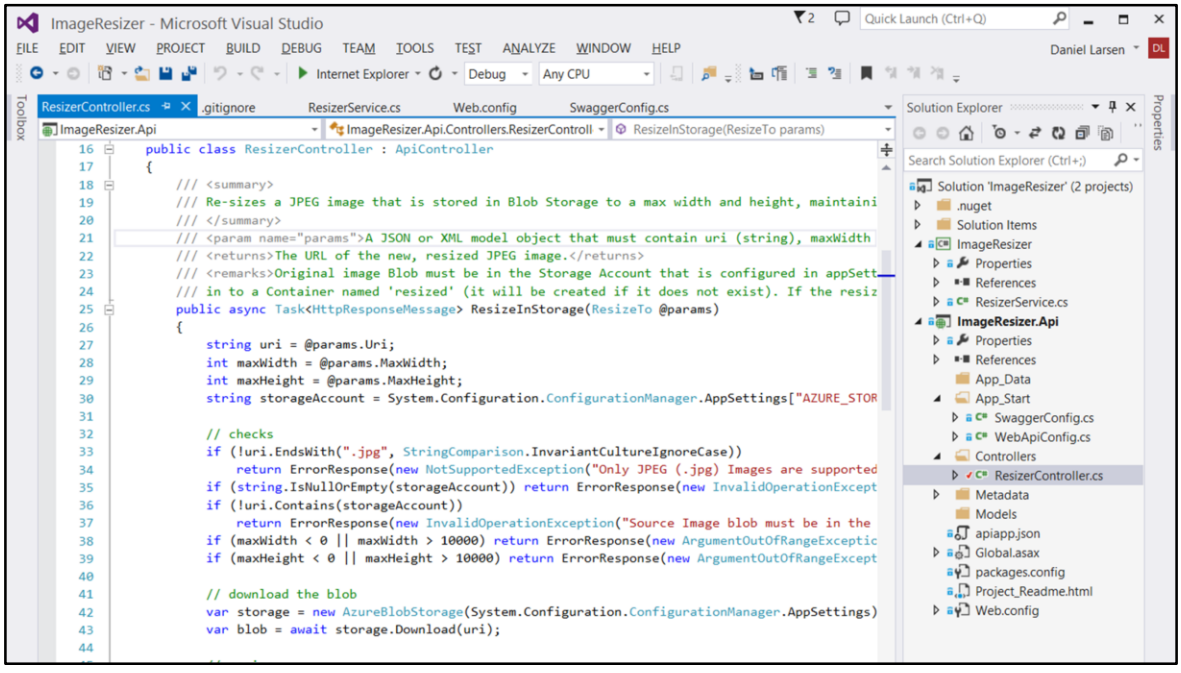

LOGIC APPS
Automate SaaS and on-premises systems
New Logic Apps for easy automation
No code designer for rapid creation
Dozens of pre-built templates to get started
Out of box support for popular SaaS and on-premises apps
Use with custom API apps of your own
Biztalk APIs for expert integration scenarios

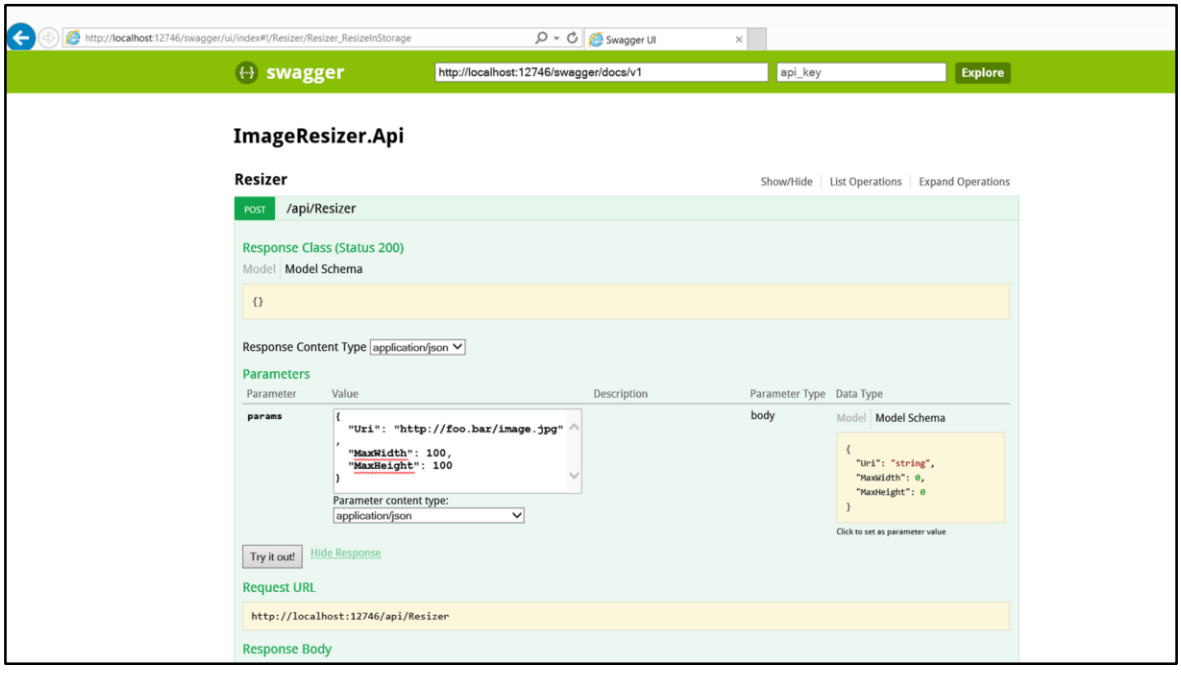
- Let's take a look at an API App now.
- In the App Service announcement Scott Hanselman does a killer demo of a Car Service company workflow – check it out.
- My favourite but is when he reuses an old VIN Decoder class from an old service in a Legacy app.
    - To me this is the opportunity – exploiting old code to create micro API Apps that can contribute to new workflows, or old workflows in new ways.
- My demo is very simple, but reinforces this point.
- (ImageResizer.sln)
- ImageResizer.Resizer service is a simple Service that takes an image as stream and returns a resized image as Stream. It is old, well worn and tested code that does one thing well.
    - I have extracted it from a horrible system that stores the Images as Blobs in an old SQL Server DB
    - I am going to create a new workflow that stores the original and resized images in Azure Blob Storage – a much cheaper an easier to manage option.
- I create ImageResizer.API – A new Azure API App project (now available in Azure

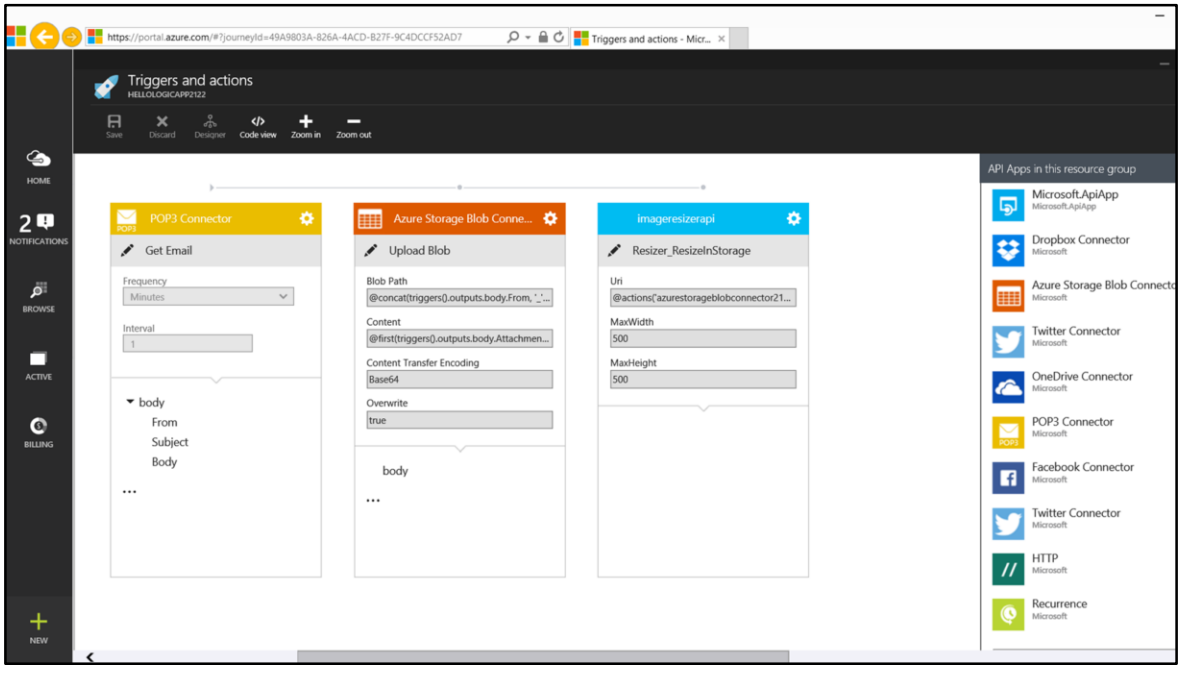
SDK 2.5.1).

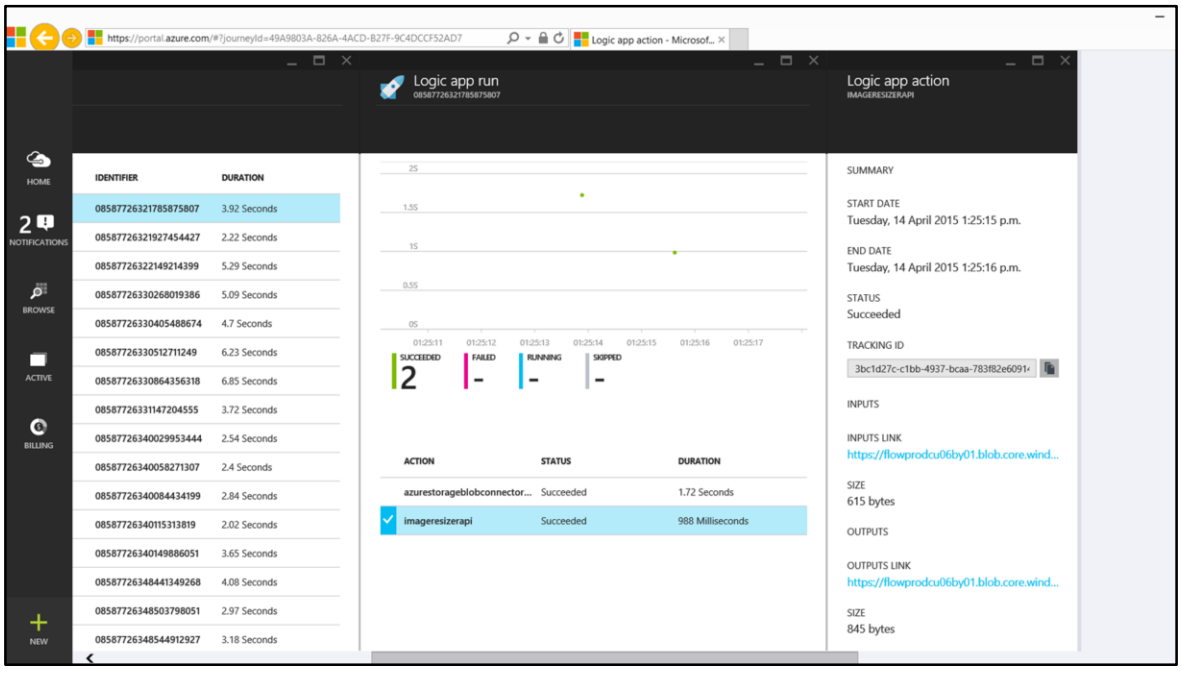
- I write a Web API controller to download the Image as Blob from Storage, pass the Resize Service, and upload the resized version to storage.
- I enable Swagger!
- I compile and F5 Run
- Check out /swagger – definition language for REST with a beautiful docs UI
- I Publish to Azure using the Publish wizard.
- It is now available to my workflow.
- This unlocks extreme possibilities in terms of building a library of Integration service Apps that you can share with your Organisation, or the community (or even sell).

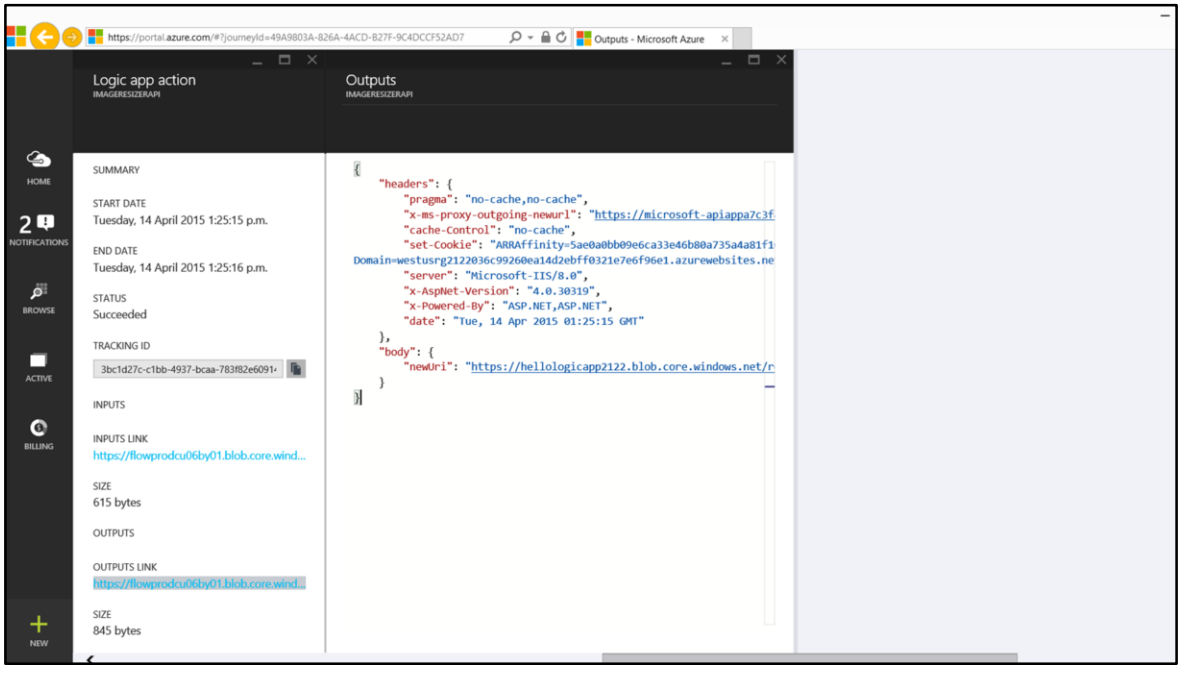

ImageResizer - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
Daniel Larsen
Internet Explorer
Debug
Any CPU
Toolbox
ResizerController.cs
.gitignore
ResizerService.cs
Web.config
SwaggerConfig.cs
ImageResizer
ImageResizer.ResizerService
Resize(Stream image, Size maxSize)
using System.Threading.Tasks;
using System.IO;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Drawing.Imaging;
namespace ImageResizer
{
public class ResizerService
{
// http://www.codeproject.com/Articles/191424/Resizing-an-Image-On-The-Fly-using-NET
public Stream Resize(Stream image, Size maxSize)
{
using (var originalImage = Image.FromStream(image))
{
var newSize = Redimension(originalImage.Size, maxSize);
using (var newImage = new Bitmap(newSize.Width, newSize.Height))
{
using (Graphics graphicsHandle = Graphics.FromImage(newImage))
{
graphicsHandle.InterpolationMode = InterpolationMode.HighQualityBicubic;
graphicsHandle.DrawImage(originalImage, 0, 0, newSize.Width, newSize.Height);
}
var memStream = new MemoryStream();
newImage.Save(memStream, ImageFormat.Jpeg);
memStream.Seek(0, SeekOrigin.Begin);
return memStream;
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'ImageResizer' (2 projects)
.nuget
Solution Items
ImageResizer
Properties
References
ResizerService.cs
ImageResizer.Api
Properties
References
App_Data
App_Start
SwaggerConfig.cs
WebApiConfig.cs
Controllers
ResizerController.cs
Metadata
Models
apiapp.json
Global.asax
packages.config
Project_Readme.html
Web.config
Properties

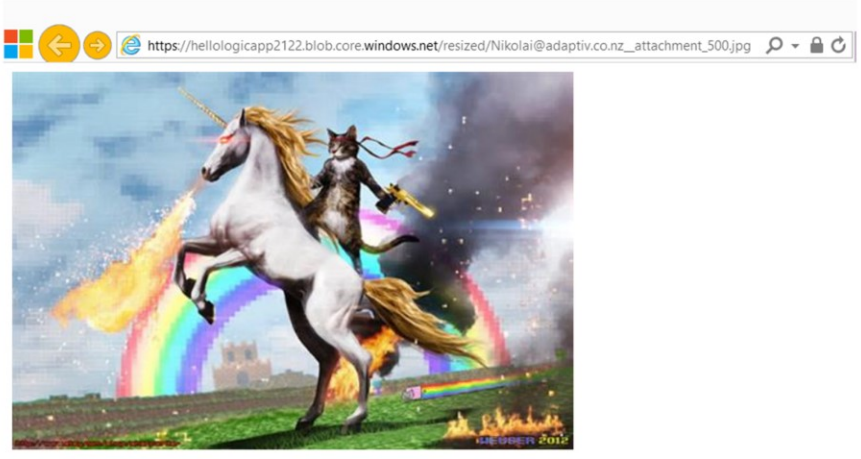

ImageResizer - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
Daniel Larsen
Internet Explorer
Debug
Any CPU
Toolbox
ResizerController.cs
.gitignore
ResizerService.cs
Web.config
SwaggerConfig.cs
ImageResizer.Api
ImageResizer.Api.Controllers.ResizerControll
ResizeInStorage(ResizeTo params)
public class ResizerController : ApiController
{
/// <summary>
/// Re-sizes a JPEG image that is stored in Blob Storage to a max width and height, maintaini
/// </summary>
/// <param name="params">A JSON or XML model object that must contain uri (string), maxWidth
/// <returns>The URL of the new, resized JPEG image.</returns>
/// <remarks>Original image Blob must be in the Storage Account that is configured in appSett
/// in to a Container named 'resized' (it will be created if it does not exist). If the resiz
public async Task<HttpResponseMessage> ResizeInStorage(ResizeTo @params)
{
string uri = @params.Uri;
int maxWidth = @params.MaxWidth;
int maxHeight = @params.MaxHeight;
string storageAccount = System.Configuration.ConfigurationManager.AppSettings["AZURE_STOR
// checks
if (!uri.EndsWith(".jpg", StringComparison.InvariantCultureIgnoreCase))
return ErrorResponse(new NotSupportedException("Only JPEG (.jpg) Images are supported
if (string.IsNullOrEmpty(storageAccount)) return ErrorResponse(new InvalidOperationExcept
if (!uri.Contains(storageAccount))
return ErrorResponse(new InvalidOperationException("Source Image blob must be in the
if (maxWidth < 0 || maxWidth > 10000) return ErrorResponse(new ArgumentOutOfRangeExceptic
if (maxHeight < 0 || maxHeight > 10000) return ErrorResponse(new ArgumentOutOfRangeExcept
// download the blob
var storage = new AzureBlobStorage(System.Configuration.ConfigurationManager.AppSettings)
var blob = await storage.Download(uri);
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'ImageResizer' (2 projects)
.nuget
Solution Items
ImageResizer
Properties
References
ResizerService.cs
ImageResizer.Api
Properties
References
App_Data
App_Start
SwaggerConfig.cs
WebApiConfig.cs
Controllers
ResizerController.cs
Metadata
Models
apiapp.json
Global.asax
packages.config
Project_Readme.html
Web.config
Properties


http://localhost:12746/swagger/ui/index#!/Resizer/Resizer_ResizeInStorage
Swagger UI
swagger
http://localhost:12746/swagger/docs/v1
api_key
Explore
ImageResizer.Api
Resizer
Show/Hide
List Operations
Expand Operations
POST
/api/Resizer
Response Class (Status 200)
Model
Model Schema
{}
Response Content Type
application/json
Parameters
Parameter
Value
Description
Parameter Type
Data Type
params
{
"Uri": "http://foo.bar/image.jpg"
,
"MaxWidth": 100,
"MaxHeight": 100
}
Parameter content type:
application/json
body
Model
Model Schema
{
"Uri": "string",
"MaxWidth": 0,
"MaxHeight": 0
}
Click to set as parameter value
Try it out!
Hide Response
Request URL
http://localhost:12746/api/Resizer
Response Body


https://portal.azure.com/#?journeyId=49A9803A-826A-4ACD-B27F-9C4DCCF52AD7
Triggers and actions - Micr...
Triggers and actions
HELLOLOGICAPP2122
Save
Discard
Designer
Code view
Zoom in
Zoom out
HOME
2
NOTIFICATIONS
BROWSE
ACTIVE
BILLING
NEW
POP3 Connector
Get Email
Frequency
Minutes
Interval
1
body
From
Subject
Body
Azure Storage Blob Conne...
Upload Blob
Blob Path
@concat(triggers().outputs.body.From, '_'...
Content
@first(triggers().outputs.body.Attachmen...
Content Transfer Encoding
Base64
Overwrite
true
body
imageresizerapi
Resizer_ResizeInStorage
Uri
@actions('azurestorageblobconnector21...
MaxWidth
500
MaxHeight
500
API Apps in this resource group
Microsoft.ApiApp
Microsoft.ApiApp
Dropbox Connector
Microsoft
Azure Storage Blob Connecto
Microsoft
Twitter Connector
Microsoft
OneDrive Connector
Microsoft
POP3 Connector
Microsoft
Facebook Connector
Microsoft
Twitter Connector
Microsoft
HTTP
Microsoft
Recurrence
Microsoft


https://portal.azure.com/#?journeyId=49A9803A-826A-4ACD-B27F-9C4DCCF52AD7
Logic app action - Microsof...
Logic app run
08587726321785875807
Logic app action
IMAGERESIZERAPI
HOME
2
NOTIFICATIONS
BROWSE
ACTIVE
BILLING
NEW
IDENTIFIER
DURATION
08587726321785875807 3.92 Seconds
08587726321927454427 2.22 Seconds
08587726322149214399 5.29 Seconds
08587726330268019386 5.09 Seconds
08587726330405488674 4.7 Seconds
08587726330512711249 6.23 Seconds
08587726330864356318 6.85 Seconds
08587726331147204555 3.72 Seconds
08587726340029953444 2.54 Seconds
08587726340058271307 2.4 Seconds
08587726340084434199 2.84 Seconds
08587726340115313819 2.02 Seconds
08587726340149886051 3.65 Seconds
08587726348441349268 4.08 Seconds
08587726348503798051 2.97 Seconds
08587726348544912927 3.18 Seconds
25
1.5S
1S
0.5S
0S
01:25:11
01:25:12
01:25:13
01:25:14
01:25:15
01:25:16
01:25:17
SUCCEEDED
2
FAILED
-
RUNNING
-
SKIPPED
-
ACTION
STATUS
DURATION
azurestorageblobconnector... Succeeded 1.72 Seconds
imageresizerapi Succeeded 988 Milliseconds
SUMMARY
START DATE
Tuesday, 14 April 2015 1:25:15 p.m.
END DATE
Tuesday, 14 April 2015 1:25:16 p.m.
STATUS
Succeeded
TRACKING ID
3bc1d27c-c1bb-4937-bcaa-783f82e6091
INPUTS
INPUTS LINK
https://flowprodcu06by01.blob.core.wind...
SIZE
615 bytes
OUTPUTS
OUTPUTS LINK
https://flowprodcu06by01.blob.core.wind...
SIZE
845 bytes


https://portal.azure.com/#?journeyId=49A9803A-826A-4ACD-B27F-9C4DCCF52AD7
Outputs - Microsoft Azure
Logic app action
IMAGERESIZERAPI
Outputs
IMAGERESIZERAPI
HOME
2
NOTIFICATIONS
BROWSE
ACTIVE
BILLING
NEW
SUMMARY
START DATE
Tuesday, 14 April 2015 1:25:15 p.m.
END DATE
Tuesday, 14 April 2015 1:25:16 p.m.
STATUS
Succeeded
TRACKING ID
3bc1d27c-c1bb-4937-bcaa-783f82e6091
INPUTS
INPUTS LINK
https://flowprodcu06by01.blob.core.wind...
SIZE
615 bytes
OUTPUTS
OUTPUTS LINK
https://flowprodcu06by01.blob.core.windows...
SIZE
845 bytes
{
"headers": {
"pragma": "no-cache,no-cache",
"x-ms-proxy-outgoing-newurl": "https://microsoft-apiappa7c3f
"cache-Control": "no-cache",
"set-Cookie": "ARRAffinity=5ae0a0bb09e6ca33e46b80a735a4a81f1
Domain=westusrg2122036c99260ea14d2ebff0321e7e6f96e1.azurewebsites.ne
"server": "Microsoft-IIS/8.0",
"x-AspNet-Version": "4.0.30319",
"x-Powered-By": "ASP.NET,ASP.NET",
"date": "Tue, 14 Apr 2015 01:25:15 GMT"
},
"body": {
"newUri": "https://hellologicapp2122.blob.core.windows.net/r
}
}


https://hellologicapp2122.blob.core.windows.net/resized/Nikolai@adaptiv.co.nz__attachment_500.jpg

API APPS

Create, consume and host APIs more easily

Easily use cloud or custom APIs:

- Dozens of built-in APIs for popular SaaS
- An ecosystem of APIs for any need
- Create and publish custom, reusable APIs
- Visual Studio tooling with one click publish and remote debugging
- Automatic client SDK generation for many languages

Wrap-up

- Azure App Service is a single integrated offering for building rich Web and Mobile Apps backed by composable Integration connectors and API Apps
- API App Service Marketplace presents a library of ready made connectors and the opportunity to contribute more.
- Logic Apps – Drag-and-drop Lego for Integrators
- API Apps – Quickly create pluggable Interfaces from new code or upgrade your old LOB apps so that they can be exploited by other workflows.




Get Started w/ App Service!

- Web Apps generally available today!
- Mobile, API, Logic Apps in preview
- Get started for **free** @ azure.com

Questions?

@DanielLarsenNZ
https://github.com/DanielLarsenNZ/Azure-App-Service-Talk
http://daniellarsen.nz