Microsoft Azure - Starter Kits for Partners

Architecture

Mobile App with backend in Azure

Last Update: September 2015





**MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.**

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The descriptions of other companies’ products in this document, if any, are provided only as a convenience to you. Any such references should not be considered an endorsement or support by Microsoft. Microsoft cannot guarantee their accuracy, and the products may change over time. Also, the descriptions are intended as brief highlights to aid understanding, rather than as thorough coverage. For authoritative descriptions of these products, please consult their respective manufacturers.

© 2015 Microsoft Corporation. All rights reserved. Any use or distribution of these materials without express authorization of Microsoft Corp. is strictly prohibited.

Microsoft and Windows are either registered trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Contents

[Overview 4](#_Toc429519721)

[This Starter Kit Selected Architecture 4](#_Toc429519722)

[Azure Mobile Services Architecture and Infographic 5](#_Toc429519723)

[Choosing a Compute Service 8](#_Toc429519724)

[Azure App Service and Mobile Apps 8](#_Toc429519725)

[App Types 10](#_Toc429519726)

[Migrate your existing Azure Mobile Service to an Azure App Service Mobile App 10](#_Toc429519727)

[Auto Scaling 12](#_Toc429519728)

[What is auto scale? 12](#_Toc429519729)

[How to auto scale an application on Azure? 13](#_Toc429519730)

Architecture for Mobile App back end on Azure

# Overview

The purpose of this document is to complement the lab document to include a broader discussion of concepts relevant to setting up a mobile application backend Environment in Azure.

# This Starter Kit Selected Architecture

The scenario below is illustrated in the Cost Calculator and implemented in the Deployment Guidance (Hand on Labs). However, you may make changes in the architecture and topology as appropriate.

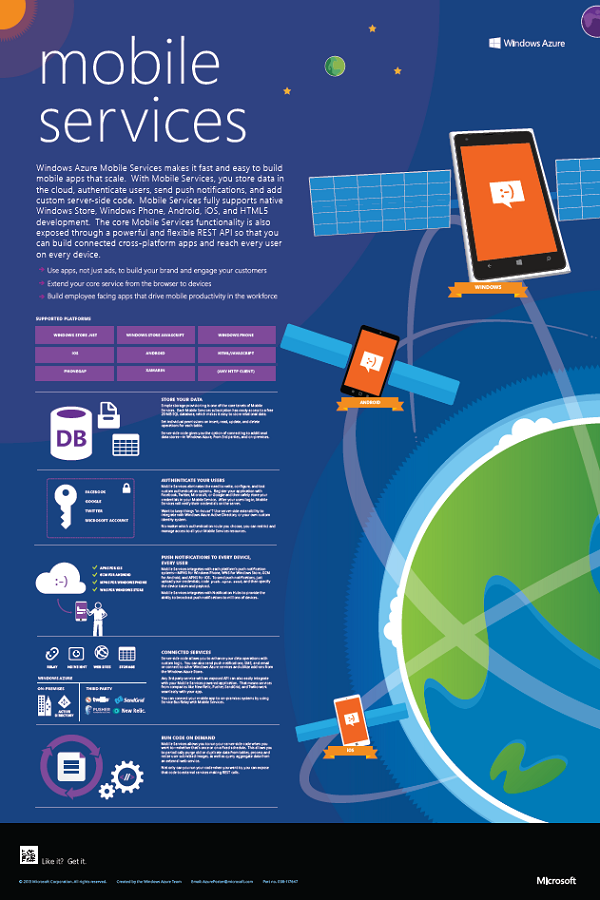
This scenario is purely based on Platform as a Service features. It provides higher agility, scalability and elasticity, which is key for migrating existing new applications to Microsoft Azure.



The rest of this document will discuss the other possible architecture aspects for this scenario.

# Azure Mobile Services Architecture and Infographic

This topic supports the Microsoft Azure Mobile Services infographic shown below. You can download a high-resolution version of this poster from the download center: [Mobile Services infographic](http://go.microsoft.com/fwlink/p/?LinkId=309701).



Mobile Services makes it fast and easy to build mobile apps that scale. With Mobile Services, you store data in the cloud, authenticate users, send push notifications, and add custom server-side code. Mobile Services fully supports native Windows Store, Windows Phone, Android, iOS, and HTML5 development. The core Mobile Services functionality is also exposed through a powerful and flexible REST API so that you can build connected cross-platform apps and reach every user on every device. For more information about the core scenarios supported by Mobile Services, see the Mobile Solutions page on the Azure web site. For general information about Mobile Services, see the [Mobile Services developer center](http://go.microsoft.com/fwlink/p/?LinkId=260441).

[Supported client platforms](javascript:void(0)): Mobile Services provides native support for the following client platforms:

|  |  |
| --- | --- |
| **Windows Store (C#/XAML)**   * [Quickstart](http://go.microsoft.com/fwlink/p/?LinkID=256501) * [Samples](http://go.microsoft.com/fwlink/p/?LinkId=309682) * [How tos](http://go.microsoft.com/fwlink/p/?LinkID=306490) * [API Reference](https://msdn.microsoft.com/en-us/library/azure/jj863454.aspx) | **Windows Store (JavaScript/HTML)**   * [Quickstart](http://go.microsoft.com/fwlink/p/?LinkID=256501) * [Samples](http://go.microsoft.com/fwlink/p/?LinkId=309682) * [How tos](http://go.microsoft.com/fwlink/p/?LinkID=306493) * [API Reference](https://msdn.microsoft.com/en-us/library/azure/jj554207.aspx) |
| **Windows Phone**   * [Quickstart](http://go.microsoft.com/fwlink/p/?LinkID=270906) * [Samples](http://go.microsoft.com/fwlink/p/?LinkId=309683) * [How tos](http://go.microsoft.com/fwlink/p/?LinkID=306490) * [API Reference](https://msdn.microsoft.com/en-us/library/azure/jj863454.aspx) | **iOS**   * [Quickstart](http://go.microsoft.com/fwlink/p/?LinkID=270907) * [Samples](http://go.microsoft.com/fwlink/p/?LinkId=309683) * [How tos](http://go.microsoft.com/fwlink/p/?LinkID=309690) * [API Reference](http://go.microsoft.com/fwlink/p/?LinkId=310146) |
| **Android**   * [Quickstart](http://go.microsoft.com/fwlink/p/?LinkID=282685) * [Samples](http://go.microsoft.com/fwlink/p/?LinkId=309691) * [How tos](http://go.microsoft.com/fwlink/p/?LinkID=301690) * [API Reference](http://go.microsoft.com/fwlink/p/?LinkID=286981) | **HTML5**   * [Quickstart](http://go.microsoft.com/fwlink/p/?LinkID=286662) * [How tos](http://go.microsoft.com/fwlink/p/?LinkID=306493) * [API Reference](https://msdn.microsoft.com/en-us/library/azure/jj554207.aspx) |

REST APIs enable you to access Mobile Services resources from in any client that can send and receive HTTP messages.

**Store your data**

Simple storage provisioning is one of the core tenets of Mobile Services. Each Mobile Services subscription has ready access to a free 20 MB Microsoft Azure SQL Database, which makes it easy to store relational data. For more information, see Work with data.

Set individual permissions on insert, read, update, and delete operations for each table. For more information, see Permissions.

Server-side code gives you the option of connecting to additional data stores—in Azure, from 3rd parties, and on-premises. For more information, see Work with server scripts in Mobile Services.

**Authenticate your users**

Mobile Services eliminates the need to write, configure, and test custom authentication systems. Register your application with Facebook, Twitter, Microsoft, or Google and then safely store your credentials in your Mobile Service. After your users log in, Mobile Services will verify their credentials on the server. For more information, see Get started with authentication (Windows Store / Windows Phone / iOS / Android / HTML).

Want to keep things “in-house”? Use server-side extensibility to integrate with Azure Active Directory or your own custom identity system. For more information, see this post on generating custom authentication tokens.

No matter which authentication route you choose, you can restrict and manage access to all your Mobile Services resources. For more information, see Use scripts to authorize users (Windows Store / Windows Phone / iOS / Android / HTML).

**Push notifications to every device, every user**

Mobile Services integrates with each platform’s push notification systems—MPNS for Windows Phone, WNS for Windows Store, GCM for Android, and APNS for iOS. To send push notifications, just upload your credentials, code push.apns.send, and then specify the device token and payload. For more information, see Get started with push notifications (Windows Store / Windows Phone / iOS / Android).

Mobile Services integrates with Notification Hubs to provide the ability to broadcast push notifications to millions of devices. For more information, see Azure Service Bus Notification Hubs.

**Connected services**

Server-side code allows you to enhance your data operations with custom logic. You can also send push notifications, SMS, and email or connect to other Azure services and utilize add-ons from the Azure Store. For more information, see Work with server scripts in Mobile Services.

Any 3rd party service with an exposed API can also easily integrate with your Mobile Services powered application. That means services from companies like New Relic, Pusher, SendGrid, and Twilio work seamlessly with your app. For more information, see Send email from Mobile Services with SendGrid and Build Real-time Apps with Mobile Services and Pusher.

You can connect your mobile app to on-premises systems by using Service Bus Relay with Mobile Services. For more information, see How to integrate a Mobile Service with a REST Service Bus Relay Service.

**Run code on demand**

Mobile Services allows you to run your server-side code when you want to—whether that’s once or on a fixed schedule. This allows you to periodically purge old or duplicate data from tables, process and resize user submitted images, as well as query aggregate data from an external web service. For more information, see Schedule backend jobs in Mobile Services

Not only can you run your code when you want to, you can expose that code to external services making REST calls. For more information, see Custom API.

# Choosing a Compute Service

Microsoft Azure Platform offers three compute services: Azure Virtual Machines, Azure Webs Apps and Azure Cloud Services (Web and Worker roles). Azure Virtual Machines offer image state persistency as well as the choice of the operating system (Windows Server or LINUX) and therefore are suitable for the majority of the applications.

Web workloads (websites and web services), however, can also be deployed to Azure Web Apps and the Azure Cloud Services (Web and Worker roles). Due the nature of these services, more detailed technical analysis of the application in question is required to determine hosting feasibility such as access to the underlying application server configuration, operating system dependencies, access to domain protected resources, etc.

Because Azure Virtual Machines [offer full control](http://azure.microsoft.com/en-us/documentation/articles/choose-web-site-cloud-service-vm/) over the environment, the rest of the discussion here focuses on this service for the application hosting.

# Azure App Service and Mobile Apps

Azure App Service is the only cloud service that integrates everything you need to quickly and easily build web and mobile apps for any platform and any device. Built for developers, App Service is a fully managed platform with powerful capabilities such as built-in DevOps, continuous integration with Visual Studio Online and GitHub, staging and production support, and automatic patching.

Azure App Service is a single service that includes all of the existing capabilities from Azure Websites, Azure Mobile Services, and Azure Biztalk Services, while adding new capabilities. [Learn more](http://azure.microsoft.com/documentation/services/app-service/) about how App Service relates to these existing services.

**Why App Service?**

Focused on rapid development of web and mobile apps, as well as automating business processes, Azure App Service provides an integrated set of enterprise capabilities through a single development and management experience offering you the following benefits:

* **Build Web and Mobile Apps Fast** - Rapidly build, deploy and manage web and mobile apps for employees or customers using a single back-end. Use your existing languages skills -- .NET, Java, NodeJS, PHP, or Python. Accelerate development with access to a rich gallery of APIs, connectors, and logic available in the Azure Marketplace.
* **Connect to any service and unlock your data** - Connect your web or mobile app to enterprise systems or SaaS in minutes with built-in connectors. Choose from more than 50 connectors for enterprise systems such as SAP, Siebel, and Oracle to popular enterprise SaaS services like Salesforce and Office 365 to popular internet services such as Facebook, Twitter and Dropbox.
* **Integration made simple** - Integrate data across clouds and automate business processes in minutes with a simple visual design experience. Easily integrate your logic it with any mobile or web app via standard REST APIs. Build sophisticated enterprise application integration, B2B solutions using EDI, business policies (rules engine), and more based on the industry leading BizTalk platform.
* **Unparalleled developer productivity** - Optimized for DevOps, with continuous integration support for Visual Studio Online and Github, so you can focus on rapidly improving your apps without worrying about infrastructure. Deploy app updates with built-in staging, roll-back and in-production testing capabilities.
* **Enterprise Ready** - Enjoy peace of mind with enterprise grade security and management. Provide delegated and role-based administration; easily secure and manage data flowing to your mobile apps; and protect your assets with built-in backup and restore capability. Rest confidently knowing App Service is PCI compliant. Stay in control with dedicated environments, and the ability to deploy across public and private clouds. Benefit from over 25 years of Microsoft experience and our enterprise Service Level Agreement.

**App Service Concepts**

The following high-level concepts describe what you can do with App Service.

### App Types

App Service allows you to create the following app types from a single development experience:

* [**Web Apps**](https://azure.microsoft.com/en-us/documentation/articles/app-service-web-overview) - Quickly create and deploy mission critical Web apps that scale with your business.
* [**Mobile Apps**](https://azure.microsoft.com/en-us/documentation/articles/app-service-mobile-value-prop-preview) - Engage employees, partners and customers on any device at any time.
* [**API Apps**](https://azure.microsoft.com/en-us/documentation/articles/app-service-api-apps-why-best-platform) - Easily build and consume Cloud APIs and participate in the API economy.
* [**Logic Apps**](https://azure.microsoft.com/en-us/documentation/articles/app-service-logic-what-are-logic-apps) - Automate the access and use of data across clouds without writing code.

As a single integrated service, App Service makes its easy to compose the above app types into a single solution, allowing you to easily build apps that target both web and mobile clients using the same back-end and integrate with on-premise systems as well as popular services such as Office 365 and salesforce.com.

**App Service Plans**

App Service Plans represent a set of features and capacity that you can share across your apps. App Service Plans support a few pricing tiers (e.g. Free, Shared, Basic, and Standard) where each tier has its own capabilities. You can quickly change which App Service Plan an app is hosted on with no downtime. [Learn more about App Service Plans](https://azure.microsoft.com/en-us/documentation/articles/web-sites-web-hosting-plan-overview/).

# Migrate your existing Azure Mobile Service to an Azure App Service Mobile App

This topic shows you how to migrate an existing application from Azure Mobile Services to a new App Service Mobile App. All existing Mobile Services apps can be easily migrated to a new App Service Mobile app. During a migration, your existing Mobile Services application can continue to operate. Over time, the process for migrating will become even easier, but for those who wish to migrate today, the following steps can be used.

##### NOTE:

Migrations are currently only supported for customers using the Mobile Services .NET backend. Applications using the Node.JS backend will need to stay on Mobile Services at this time.

**Understanding App Service Mobile Apps**

App Service Mobile Apps is a new way to build mobile applications using Microsoft Azure. You can learn more about Mobile Apps in the [What are Mobile Apps?](https://azure.microsoft.com/en-us/documentation/articles/app-service-mobile-value-prop-preview/) topic.

In a migration to Mobile Apps, all existing app functionality (and code) can be preserved. Moreover, new features are available to the application. In the Mobile Apps model, your code actually runs on a Web App (the new version of Azure Web Sites). You have full control over the web app and how it operates. In addition, Web Apps features which were previously unavailable to Mobile Services customers, such as Traffic Manager and Development Slots, can now be used.

The new model also addresses one of the major difficulties of working with Mobile Services. Now, any version of any NuGet package can be deployed without worrying about dependency conflicts. More about the benefits of migrating can be found in the [I already use web sites and mobile services – how does App Service help me?](https://azure.microsoft.com/en-us/documentation/articles/app-service-mobile-value-prop-migration-from-mobile-services-preview) topic.

**Basic migration overview**

The simplest way to migrate is to create a new instance of an App Service Mobile App. In many cases, migrating will be as simple as switching to the new Server SDK and republishing your code onto a new Mobile App. There are, however some scenarios which will require some additional configuration, such as advanced authentication scenarios and working with scheduled jobs. Each of these is covered in the following sections.

##### NOTE:

It is advised that you read and understand the rest of this topic completely before starting a migration. Make note of any features you use which are called out below.

You can move and test your code at your pace. When the Mobile App backend is ready, you can release a new version of your client application. At this point, you will have two copies of your application backend running side by side. You need to make sure any bug fixes you make get applied to both. Finally, once your users have updated to the newest version, you can delete the original Mobile Service.

The full set of steps for this migration is as follows:

1. Create and configure a new Mobile App
2. Address any authentication concerns
3. Release a new version of your client application
4. Delete your original Mobile Services instance

See details at:

<https://azure.microsoft.com/en-us/documentation/articles/app-service-mobile-dotnet-backend-migrating-from-mobile-services-preview/>

# Auto Scaling

## What is auto scale?

One of the key benefits that the Microsoft Azure technology platform delivers is the ability to rapidly scale your application in the cloud in response to changes in demand.

**Scalability is a key feature of Azure.**

When you deploy an application to Microsoft Azure, you deploy roles: web roles for the externally facing portions of your application and worker roles to handle back-end processing. When you run your application in Microsoft Azure, your roles run as role instances (you can think of role instances as virtual machines). You can specify how many role instances you want for each of your roles; the more instances you have, the more computing power you have available for that role, but the more it will cost you. There are, of course, some specific design requirements if your roles are to operate correctly when there are multiple instances of that role, but Microsoft Azure looks after the infrastructure requirements for you. For more information about design requirements, see "[Cloud Development](https://msdn.microsoft.com/en-us/library/ff898430.aspx)."

|  |  |
| --- | --- |
|  | |
|  | Scaling by adding additional instances is often referred to as **scaling ou**t. Microsoft Azure also supports scaling up by using larger role instances instead of more role instances. |

You can specify the size and the number of instances you require for each of your roles when you first deploy an application to Microsoft Azure. You can also add or remove role instances on the fly while the application is running, either manually through the Microsoft Azure portal, or programmatically by using the Microsoft Azure Management API.

By adding and removing role instances to your Microsoft Azure application while it is running, you can balance the performance of the application against its running costs. You can add new instances when demand is high, and remove instances when you no longer need them in order to reduce running costs.

If you rely on manual interventions to scale your application, you may not always achieve the optimal balance between costs and performance; an operator may respond late, or underestimate the number of role instances that you need to maintain throughput.

|  |  |
| --- | --- |
| **Hh680945.note(en-us,PandP.50).gif** | |
|  | You also need to consider the cost of having human operators performing this task, especially if you have hundreds or even thousands of role instances running in Microsoft Azure data centers around the globe. |

An autoscaling solution reduces the amount of manual work involved in dynamically scaling an application. It can do this in two different ways: either preemptively by setting constraints on the number of role instances based on a timetable, or reactively by adjusting the number of role instances in response to some counter(s) or measurement(s) that you can collect from your application or from the Microsoft Azure environment.

You will still need to evaluate the results of your autoscaling solution on a regular basis to ensure that it is delivering the optimal balance between costs and performance. Your environment is unlikely to be static; overall, the numbers of users can change, access patterns by users can change, your application may perform differently as it stores more data, or you may deploy your application to additional Microsoft Azure data centers.

**You should evaluate your autoscaling behavior on a regular basis. Even with autoscaling in place, fire-and-forget is not the best practice.**

Scaling your application by adjusting the number of role instances may not be the best or only way to scale your application. For example, you may want to modify the behavior of your application in some way during bursts in demand, or to alter the number of Microsoft Azure queues, or the size of your SQL Azure database. An autoscaling solution may not be limited to just adjusting the number of role instances.

## How to auto scale an application on Azure?

On the Scale page of the Azure Management Portal, you can manually scale your application or you can set parameters to automatically scale it. You can scale applications that are running Web Roles, Worker Roles, or Virtual Machines. To scale an application that is running instances of Web Roles or Worker Roles, you add or remove role instances to accommodate the work load.

When you scale an application up or down that is running Virtual Machines, new machines are not created or deleted, but are turned on or turned off from an availability set of previously created machines. You can specify scaling based on average percentage of CPU usage or based on the number of messages in a queue.

You should consider the following information before you configure scaling for your application:

* You must add Virtual Machines that you create to an availability set to scale an application that uses them. The Virtual Machines that you add can be initially turned on or turned off, but they will be turned on in a scale-up action and turned off in a scale-down action. For more information about Virtual Machines and availability sets, see [Manage the Availability of Virtual Machines](https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-manage-availability/).
* Scaling is affected by core usage. Larger role instances or Virtual Machines use more cores. You can only scale an application within the limit of cores for your subscription. For example, if your subscription has a limit of twenty cores and you run an application with two medium sized Virtual Machines (a total of four cores), you can only scale up other cloud service deployments in your subscription by sixteen cores. All Virtual Machines in an availability set that are used in scaling an application must be the same size. For more information about core usage and machine sizes, see [Sizes for Virtual Machines](https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-size-specs/).
* You must create a queue and associate it with a role or availability set before you can scale an application based on a message threshold. For more information, see [How to use the Queue Storage Service](http://www.windowsazure.com/en-us/develop/net/how-to-guides/queue-service).
* You can scale resources that are linked to your cloud service. For more information about linking resources, see [How to: Link a resource to a cloud service](http://www.windowsazure.com/en-us/manage/services/cloud-services/how-to-manage-a-cloud-service/#linkresources).
* To enable high availability of your application, you should ensure that it is deployed with two or more role instances or Virtual Machines. For more information, see [Service Level Agreements](https://www.windowsazure.com/en-us/support/legal/sla/).