Microsoft Azure для веб-разработчика

О том, зачем веб-разработчику облако, в каких задачах оно помогает, а в каких нет, написано уже много — в основном выделяют возможность быстро масштабироваться.

Не автоматически, но с определённой степенью гибкости — для веб-сайтов это обычно значение загрузки ЦП, превышая которую система инициирует вызов внутренних API для развертывания реплики.

Второй по популярности — оплата по факту использования. Сколько система отработала, за столько и пришёл счет.

Можно назвать ещё много причин, но перейдём к сути — посмотрим, что есть на платформе Microsoft Azure для веб-разработчика.

Когда встаёт тема веб-разработки на Microsoft Azure, обычно смотрят сразу на виртуальные машины. Выбор правильный, однако не всегда самый экономичный — за счёт предоставления клиенту практически полного контроля над происходящим с экосистемой вокруг проекта возникает как необходимость в настройке, управлении и дальнейшей поддержке (настройке отказоустойчивых ферм веб-серверов или СУБД, например), так и связанных с этим затрат.

По опыту — выбор виртуальных машин обоснован в двух случаях:

• Нужно перенести систему, у которой столько разных версий, что вопрос изменения кодовой базы может даже не рассматриваться. Вариант «виртуализовали готовый сервер с приложением и экосистемой и перенесли в облако на виртуальную машину» здесь самый очевидный;

• Нужен тюнинг экосистемы.

В Microsoft Azure есть сервис виртуальных машин, которые официально поддерживают образы, подготовленные для Windows Server, начиная с 2008 R2, и некоторых дистрибутивов Linux. Во время развертывания виртуальной машины в нее инжектируются специальные сервисы, которые и делают все бенефиты облака реальными, общаясь с системными сервисами платформы и при необходимости производя «лечение» машины, перенос данных и др. операции.

Делать с ВМ можно практически всё, что можно делать с локально-созданной, однако нужно учитывать то, что все ресурсы виртуализованы, а диски расположены по-разному — диск C:/ находится в Azure Storage, что накладывает ограничения на производительность файловой системы, так как диск, по сути, находится на расстоянии, а диск D:/ локален для сервера, и тоже имеет свою особенность — он временный. При перезагрузке или любой другой операции с ВМ этот диск будет очищен.

**Второй вариант** — Cloud Services, который был по сути родоначальником платформенных сервисов Azure. Суть CS сводится к тому, что веб-разработчик берёт (или начинает новый ) веб-проект на .NET или Java , разделяет его на фронтенд (Web-роль) и бэкенд (Worker-роль) и кладёт эти проекты в решение.

Указывает для Web-роли и Worker-роли ресурсы, которые будут выделяться, и загружает решение в Azure.

Контроль над происходящим в CS не такой, как с виртуальной машиной, и состояние хранить здесь сложнее, однако многих это устраивает — например, когда мы понимаем, что на веб-интерфейс у нас идет большая нагрузка, а на обработчик нет, смысла в масштабировании всей системы нет, поэтому мы просто увеличиваем количество экземпляров Web-роли.

Можно это сделать и с виртуальными машинами, но ручного труда будет гораздо больше (плюс это решение будет дороже).

Кстати, помимо значения CS как контейнера для приложений, CS ещё используются для размещения в них виртуальных машин. Они выступают как сетевая граница, на которой открываются/закрываются порты и др. Поэтому не удаляйте CS, используемый для виртуальной машины — он, во-первых, бесплатный, во-вторых, это гарантированный способ нарушить функционирование системы.

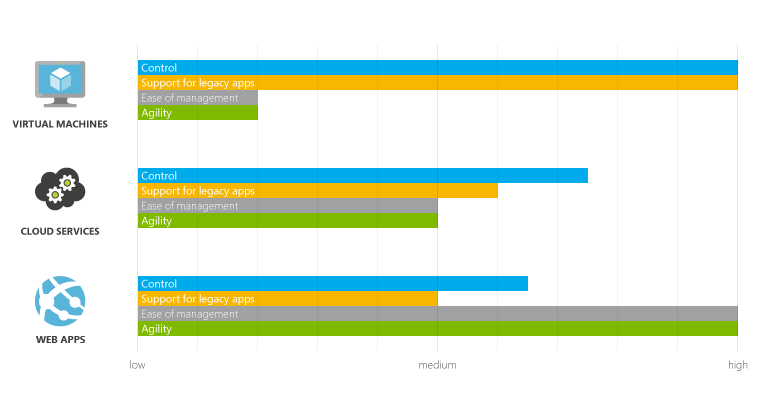
**Третий вариант** — Web Apps. Раньше назывались Web Sites.

Самое простое решение для размещения веб-приложений. Отличие от ВМ и CS — минимум контроля и настроек.

Отличие от CS — всё идёт в одном пакете, без возможности масштабировать отдельно слои приложения.

Удобно, когда нужно развернуть веб-сайт без изменений (за исключением некоторых ситуаций, например, использования старых версий PHP Runtime) кодовой базы или просто в быстром режиме. Если же у вас нет веб-сайта, но нужно быстро развернуть и посмотреть какое-то решение a-la Drupal, этот сервис тоже может помочь — есть галерея образов различного ПО.

Наглядно:

[](https://channel9.msdn.com/Events/DevCon/Russia-2015/Latest-Services-Azure-to-help-make-your-decisions-faster-more-reliable-and-safer)

Помимо этого сравнения, на этой странице много сценарных подходов. Например, My application depends on highly customized Windows or Linux environments and I want to move it to the cloud.

А дальше? Разместили...

Развернули сайт, настроили развертывание из GitHub, научились масштабировать. Что дальше?

С ростом проекта может расти количество его компонентов и модулей.

В Azure, как облачной платформе, есть набор сервисов, каждый из которых может быть практически применён для решения той или иной задачи вместо самостоятельной реализации и поддержки.

Сервисы Microsoft Azure делятся на три логических слоя – **слой вычислений**, на базе сервисов которого работают практически все остальные сервисы, **слой интеграции** – для мобильных приложений, для обеспечения коммуникаций между различными компонентами проекта, для защиты и аутентификации и др., **и слой работы с данными** – все сервисы, которые позволяют разными способами взаимодействовать и менять состояние данных. Посмотрим на всё это вкратце.

[](https://channel9.msdn.com/Events/DevCon/Russia-2015/Latest-Services-Azure-to-help-make-your-decisions-faster-more-reliable-and-safer)

**Слой доступа к данным**

Слой доступа к данным содержит нереляционные хранилища данных: **таблицы, диски, очереди, хранение двоичных объектов, DocumentDB + реляционное хранилище данных в виде SQL Database.**

Выбор опции хранилища, разумеется, зависит от задач, и каждый из сервисов имеет свою специфику.

Например, типичным использованием таблиц является хранение логов (сама Microsoft Azure при активации логирования приложения складывает данные именно туда), SQL Database актуален, когда есть необходимость быстро развернуть базу данных или протестировать решение.

• **Таблицы** — В таблице хранятся структурированные нереляционные данные.

• **Очереди** обеспечивают обмен сообщениями между приложениями.

• **Блобы** дают возможность хранить большие объёмы неструктурированных текстовых или двоичных данных.

• **SQL Database** — реляционная база данных — это масштабируемая облачная служба базы данных, построенная на основе технологий SQL Server. Повторюсь — если нужно быстро развернуть базу данных, это отличный и самый быстрый вариант. Также внутри этого сервиса есть различная интересная функциональность, например, инструмент для создания non-clustered индексов (подробнее от sitox).

Помимо SQL Database, на платформе есть еще несколько предложений — MySQL и MongoDB, аналогично доступные как сервисы.

**Azure Files** дает возможность обращаться к данным хранилища Azure Storage как к сетевому ресурсу по протоколу SMB, что позволяет осуществлять привычный доступ к данным из виртуальных машин через сетевое взаимодействие. Что значит, что можно использовать стандартные Windows API.

**Caching** — распределенный кэш в памяти, с помощью которого вместо медленного дискового хранилища приложения получают высокоскоростной доступ к данным, хранящимся в оперативной памяти, с возможностью масштабирования.

Cache на базе Redis — cервис Azure Redis Cache представляет собой готовое redis-хранилище с требуемым размером для задач кеширования данных.

[](http://habrahabr.ru/post/262125/)

**Слой интеграции**

***Media Services*** включают в себя облачные версии многих существующих технологий платформы мультимедиа Microsoft и партнеров, в том числе для просмотра, кодирования, преобразования формата и защиты контента, а также потоковой передачи по запросу и в реальном времени. Поддерживает Live Streaming.

Этот сервис рекомендуется использовать при задачах обработки мультимедиа контента. Можно делать это и на виртуальных машинах (например, ставить VoIP или медиа-сервер), однако в этом случае можно столкнуться с различными ограничениями (такими, как ограниченное количество портов, трафик и др.).

***Mobile Services*** предлагает облачную инфраструктуру для всех популярных мобильных платформ: Windows 8+, Windows Phone 8+, iOS, Android, HTML 5.

На основе сервиса можно построить облачный бэкенд, на который перенести задачи по хранению данных, аутентификации и Push-уведомлений. Поддерживается Xamarin. Используется многими клиентами.

***Identity*** — служба аутентификации обеспечивает управление удостоверениями и доступом к приложениям, с помощью службы Microsoft Azure Active Directory (бывший Access Control Service) можно обеспечить единый вход, повышенную безопасность и простое взаимодействие с уже развернутыми в Active Directory приложениями, а также выполнить интеграцию с другими провайдерами аутентификации (Live ID, Google, Facebook и т. п.).

Azure Active Directory позволяет решать задачи единой авторизации пользователей для множества сервисов (Single Sign On), вести единый каталог пользователей, синхронизировать данные каталога с Active Directory на предприятии и т. д.

Azure Active Directory — это полноценная реализация каталога в облаке. Сервис поддерживает популярные открытые стандарты обеспечения федераций: SAML 2.0, OData, WS-FED, OAuth 2.0/OpenID.

***Service Bus*** предоставляет возможности ретрансляции и безопасного обмена сообщениями и позволяет создавать распределенные и слабосвязанные приложения в облаке, а также гибридные приложения, размещенные одновременно в частных и общедоступных облачных службах. Гораздо мощнее и функциональнее, нежели Storage Queues, следовательно, сложнее в поддержке и выборе правильных технологий и подходов. Кроме обычной очереди, содержит специальную очередь Event Hubs, которая актуальна при использовании в концепции Интернета Вещей, когда много (очень) устройств, посылающих большое количество небольших сообщений.

***Traffic Manager*** — трафик-менеджер обеспечивает балансировку нагрузки по входящему трафику между несколькими размещенными службами Windows Azure независимо от того, работают ли они в одном центре обработки данных или распределены по нескольким.

***API Management*** предлагает разработчикам собственных API возможность получить окружение по управлению, мониторингу и администрированию своего API, размещенного в любом месте, как в облаке, так и на любом хостинге, включая собственную инфраструктуру.

[](http://habrahabr.ru/post/262125/)

Помимо этих сервисов есть еще, например***, Application Insights*** — интегрируете агента в приложение или на сервер, и имеете полноценную историческую картину происходящего с вашим проектом. В целом же, если требуется выбрать правильный сервис, можно воспользоваться сайтом-конструктором.