# Chapter 4

# Windows Azure Storage Part I – Blobs

Windows Azure Storage is a scalable, highly available, and durable service to store any kind of application data.

The Storage service provides you with the ability to store data in three different types of storage types: *blobs, queues,* and *tables*. Depending on the application requirements, you can choose the appropriate storage type for your data. You can also use multiple storage types within the same application.

* The **Blob** service is designed to store large binary objects with associated metadata like documents, pictures, videos, and music files.
* The **Queue** is a reliable asynchronous message delivery and storage type. Cloud services as well as on-premises applications can use queues for asynchronous cross-application communications.
* The **Table** storage type provides structured storage capability to store billions of lightweight data objects occupying terabytes of data.

Table 4-1 lists the Windows Azure storage types and some of their properties.

|  |  |  |  |
| --- | --- | --- | --- |
| **Table 4-1. Windows Azure Storage** | | | |
| **Feature** | **Blob** | **Queue** | **Table** |
| URL schema | http://[Storage Account].**blob**.core.windows.net/**[Container Name]**/[Blob Name] | http://[Storage Account].**queue**.core.windows.net/[Queue Name] | http://[Storage Account].**table**.core.windows.net/[Table Name]**?$filter=[Query]** |
| Max size | 50GB/blob | 8KB (string) | Designed for terabytes of data |
| Recommended usage | Designed for large binary data types | Designed for cross-service message communication | Designed to store smaller structured objects like the user state across sessions |
| API reference | <http://msdn.microsoft.com/en-us/library/dd135733.aspx> | <http://msdn.microsoft.com/en-us/library/dd179363.aspx> | <http://msdn.microsoft.com/en-us/library/dd179423.aspx> |

## Storage Service Architecture

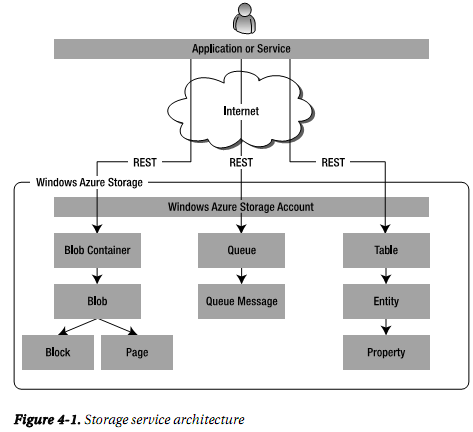
The Windows Azure Storage service allows users to store application data in the cloud and access it from anywhere, anytime.

The open architecture of the Storage service lets you design your applications and services to store and retrieve data using REST APIs. Each storage type in the Storage service has an independent REST programming API.

Windows Azure Storage types are scoped at the account level. The Blob, Queue, and Table services all have REST API functions that let you interact with them.

* A blob account is a collection of *containers*. A container consists of number of *blobs*. A blob can be further composed of a series of *blocks*.
* A queue account is a collection of *queues*. A queue is composed of *queue messages* sent by the message sending applications.
* The table storage type supports access via REST as well as the *ADO.NET Data Services API*. A table consists of a set of *entities* that represent runtime objects or data. Entities are similar to the rows of data in a relational database. They have *properties*, which are similar to the database fields in a relational database table.

The table storage type isn’t a relational database table; it follows the entity model.



**What is REST ?**

**Representation State Transfer (REST)** is an interface architecture for systems deployed and accessible over the network.

The system entry points are abstracted into web resources. In REST, each resource has metadata and is uniquely identified by a URL. The operations of the resource are also exposed via URL. Each URL interaction with the resource returns a representation that can be any document or a binary object.

The URI scheme for addressing the Storage services is:

<http|https>://<account-name>.<storage service name>.core.windows.net/<resource-path>

* <http|https> is the protocol used to access Storage services.
* <account-name> is the unique name of your storage account.
* <storage service name> is the name of the storage service you’re accessing (blob, queue, or table).
* <resource-path> is the path of the underlying resource in the storage services that you’re accessing. It can be a blob container name, a queue name, or a table name.

## The Blob Service

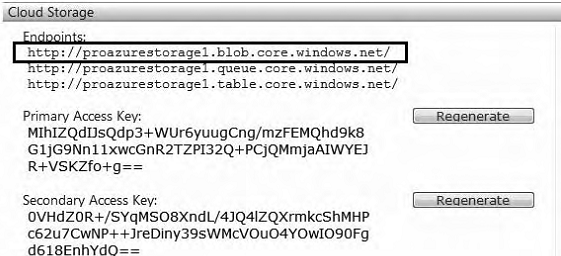
The Blob service provides scalable and highly available storage for any kind of entities, such as binary files and documents.

The Blob service achieves its scalability and high availability by distributing blob files across multiple servers and replicating (tái tạo) them at least three times.

It provides a REST API to store named files along with their metadata. The Blob REST API provides consistency-checking features for concurrent operations.

**Note**: *Windows Azure Blob,* *blob storage*, *Blob service*, and *Blob Storage service* all mean the same thing. The blob object in the Blob service points to the actual file stored in the Blob service.

The Blob service is scoped at the account level. When you create an account on the Azure Services Developer Portal, you get access to the Blob service. Figure 4-2 shows the Azure Services Developer Portal page for the created storage account and the URL endpoint for the Blob service:



**Figure 4-2. Blob endpoint URL**

The account endpoint for the Blob service is *<account name>*.blob.core.windows.net, where *<account name>* is the unique name you created for your storage account.

The secret key associated with the account provides security for accessing the storage account. You can use the secret key for create an HMAC-SHA256 signature for each request. The storage server uses the signature to authenticate the request.

**Note:** **HMAC** stands for **Hash Message Authentication Code**, which is a message-authentication code calculated from the secret key using a special cryptographic hash function like MD5, SHA-1, or SHA256.

The Windows Azure Storage service expects a SHA256 hash for the request. SHA256 is a 256-bit hash for the input data.

### Blob Limitations and Constraints

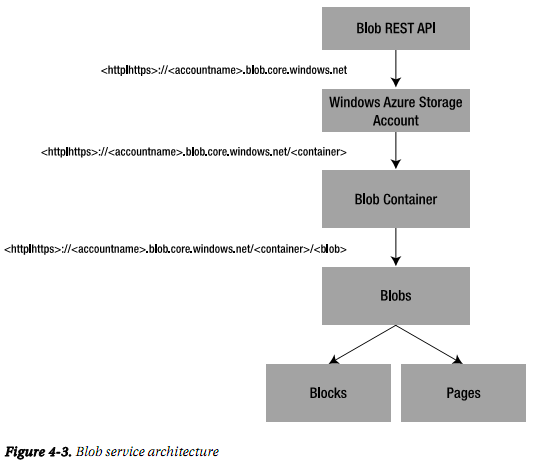
The limitations of the Blob service are:

* The maximum size of each block blob is 200GB and each page blob is 1TB (per version 2009-09-19 of the storage service API).
* You can upload blobs that are less than or equal to 64MB in size using a single PUT operation. Blobs more than 64MB in size must be uploaded as a set of blocks, with each block not greater than 4MB in size.
* The development Blob service supports blob sizes only up to 2GB.

### Blob Architecture

The blob architecture consists of a four-level hierarchical structure: account, containers, blobs, blocks and pages.

Windows Azure storage account is the entry point to the Blob service via the REST API.



#### Windows Azure Storage Account

The Windows Azure storage account encompasses the blob, queue, and table storage types.

The URI scheme to access the Blob service via storage account is:

*<http|https>://<account name>.blob.core.windows.net*

where *<account name>* is the unique name you created for your storage account. The *<account name>* must be globally unique.

#### Containers

A container is a logical grouping for a set of blobs.

Containers can have metadata in the form of name-value pairs.

Blob is the only storage type that supports public (for read-only purposes without authentication) and private access (are visible only to the account owner.); the queue and table storage types support only private access.

You can access a container the following URI:

*<http|https>://<account name>.blob.core.windows.net/<container>*

where *<container>* is the name of the container you want to access.

The naming constraints on a container are as follows:

* Container names must be unique within an account.
* Container names must start with a letter or a number.
* Container names can’t contain any special characters other than the dash (-) character.
* The dash (-) character must be immediately followed by a character or a number.
* All the characters in the container name must be lowercase.
* Container names can’t be fewer than 3 or more than 63 characters in length.

If a container name or the URI violates the naming convention, an HTTP status code 400 (Bad Request) is returned by the server.

#### Blobs

Blobs, which are the actual entities in the Blob service, are stored in containers.

A blob name must be unique within the scope of a container.

A blob can also have metadata in the form of name-value pairs.

You can access a blob using the following URI:

*<http|https>://<accountname>.blob.core.windows.net/<container>/<blob>*

where *<blob>* is a unique name of the blob within the specified container.

A blob name can’t be more than 1,024 characters long. Blob doesn’t support creation of folder hierarchies to store files; you can store files only in a flat structure. To facilitate creation of a virtual folder structure, you to add a delimiter to a blob’s name.

**Types of Blobs**

In version 2009-09-19 of the storage service API, two types of blobs are available: page blobs and block blobs.

**Page Blobs**

Page block are optimized for read/write access and provide you with the ability to copy a series of bytes into a blob.

A page is represented by its start offset from the start of the blob. Writes to page blobs are immediately committed to the blob storage.

You can store up to 1TB of data per page.

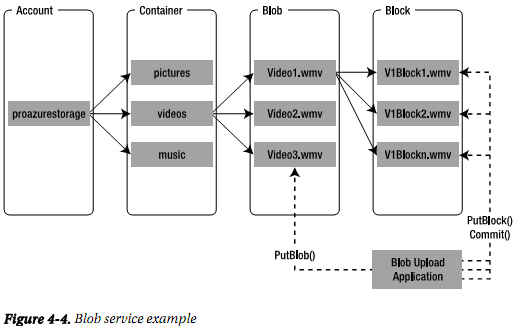
Page blobs are ideal for applications requiring quick read/write access to binary data like images, videos, documents, and so on. The Windows Azure Storage Client API provides two operations on page blobs: *Put Page* and *Get Page Regions*.

**Block Blobs**

Each block can be a maximum of 4MB in size.

After all the blocks are uploaded, they can be committed to a particular blob. There is no URI to access blocks in a blob: after blocks are committed to a blob, you can only retrieve that complete blob. So, you can execute the *GET* operation only to the blob level.

Uploading blocks and committing blocks to a blob are two separate operations.You may upload multiple blocks in parallel in any random sequence, but when you execute the commit operation, you must specify the correct list for the block sequence representing the readable blob.

****