Windows Azure Blob

GVHD: Phan Trung Hiếu Sinh viên thực hiện:

Nguyễn Minh Tuấn – 07520450

Nguyễn Anh Duy – 07520058

Table of Contents

[1 Introduction 2](#_Toc290113181)

[2 Giao tiếp REST với Blob 3](#_Toc290113182)

[2.1 Phiên bản 4](#_Toc290113183)

[3 Một Blob là một danh sách các block 4](#_Toc290113184)

[3.1 Trừu tượng hóa dữ liệu Block 5](#_Toc290113185)

[3.2 Blobs – Azure Storage Client v1.0 6](#_Toc290113186)

[3.2.1 CloudBlobClient 9](#_Toc290113187)

[3.2.2 CloudBlobContainer 10](#_Toc290113188)

[3.2.3 CloudBlob 11](#_Toc290113189)

[3.2.4 CloudBlockBlock 14](#_Toc290113190)

[3.2.5 CloudPageBlob 16](#_Toc290113191)

[3.3 Các chiến lược tải block lên 17](#_Toc290113192)

# Introduction

Windows Azure là nền tảng đám mây của Microsoft. Nó được gọi là “hệ điều hành đám mây”, nó cung cấp những khối kiến trúc cần thiết để các lập trình viên mở rộng và nâng cấp những dịch vụ sẵn có. Windows Azure cung cấp:

* Tính toán ảo hóa.
* Lưu trữ mở rộng.
* Quản lí tự động.
* Bộ phát triển SDK phong phú.

Bộ lưu trữ Windows Azure cho phép các lập trình viên lưu dữ liệu của họ trên đám mây. Ứng dụng có thể truy xuất dữ liệu từ bất kì nơi đâu, tại bất kì thời điểm nào, lưu trữ lượng dữ liệu tùy ý trong khoảng thời gian bất kì và nó đảm bảo rằng dữ liệu được lưu trữ bền vững và không bị thất lạc. Bộ lưu trữ Windows Azure cung cấp một bộ trừu tượng hóa dữ liệu phong phú:

* Windows Azure Blob – cung cấp lưu trữ cho các mẫu dữ liệu lớn.
* Windows Azure Table – cung cấp bộ lưu trữ có cấu trúc để lưu trữ tình trạng sử dụng.
* Windows Azure Queue – cung cấp điều phối công việc không đồng bộ để khởi động dịch vụ liên lạc.

Phần tài liệu này mô tả Windows Azure Blob, và cách thức sử dụng Windows Azure Blob để cho phép ứng dụng lưu trữ những đối tượng có kích thước lên đến 50GB trong đám mây. Nó hỗ trợ hệ thống blob có khả năng mở rộng cực đại, và các blob nóng sẽ được cung ứng bởi nhiều máy chủ để đáp ứng nhu cầu lưu thông của ứng dụng. Hơn nữa, hệ thống có tính sẵn sàng cao và bền lâu. Bạn có thể truy vấn dữ liệu của bạn từ bất kì nơi đâu từ bất kì thời điểm nào, và dữ liệu được sao ra ít nhất 3 lần để đảm bảo độ bền. Ngoài ra, tính nhất quán mạnh được cung cấp để đảm bảo rằng các đối tượng có thể truy cập ngay lập tức một khi nó mới được thêm vào hoặc được cập nhật; lần đọc tiếp theo ngay lập tức sẽ thấy những thay đổi mới nhất của đối tượng.

Giới thiệu các thành phần của Windows Azure Blob.

* **Tài khoản lưu trữ** – Tất cả truy xuất đến bộ lưu trữ Windows Azure được thực hiện thông qua tài khoản lưu trữ.
  + Một tài khoản có thể có nhiều Container.

Hình 1Khái niệm bộ lưu trữ Blob

* + **Blob Container** – Một container bao gồm nhiều blob.
  + Chính sách chia sẻ được thiết lập ở cấp độ container. Hiện tại hai thuộc tính "Public READ" và "Private" được hỗ trợ. Khi một container có thuộc tính là “Public READ”,thì tất cả nội dung của nó có thể được đọc bởi bất kì người dùng mà không yêu cầu chứng thực. Khi một container có thuộc tính là “Private”, thì chỉ có người sở hữu tương ứng mới có thể truy xuất blob trong container với truy xuất được chứng thực.
  + Một container có thể có metadata kết hợp với chúng. Metadata ở dạng cặp <tên, giá trị>, và chúng lên đến 8KB kích thước với mỗi container.
* **Blob** – Blob được lưu trữ trong Container. Kích thước mỗi blob có thể lên đến 50GB. Một blob có một tên duy nhất trong một container. Blob có thể có metadata kết hợp với chúng, cặp <tên, giá trị>, và chúng lên đến 8KB kích thước cho mỗi blob.

Đường dẫn URI cho một blob được cấu trúc như sau:

http://<account>.**blob**.core.windows.net/<container>/<blobname>

# Giao tiếp REST với Blob

Tất cả các giao tiếp đến Windows Azure Blob được thực hiện thông qua giao tiếp HTTP REST PUT/GET/DELETE.

Các lệnh HTTP/REST được hỗ trợ để thực thi các thao tác blob bao gồm :

* PUT Blob – Thêm một blob mới hoặc ghi đè blob đã tồn tại.
* GET Blob – Lấy một thực thể blob, hoặc lấy một dãy các byte trong blob sử dụng thao tác GET chuẩn HTTP.
* DELETE Blob – Xóa một blob đã tồn tại.
* Copy Blob – Sao chép một blob từ blob nguồn đến blob đích trong cùng tài khoản lưu trữ. Nó sẽ sao chép toàn bộ blob, bao gồm metadata, thuộc tính, và blocklist. Bạn có thể sử dụng CopyBlob cùng với DeleteBlob để đổi tên một blob, để di chuyển một blob giữa các containers, hoặc tạo bảng sao dự phòng cho blob đang tồn tại của bạn.
* Get Block List - Lấy danh sách các block đã được tải lên như một phần của blob. Có hai loại danh sách block:
  + Danh sách block đã commit – Đây là danh sách block đã được commit thành công qua PutBlockList vào một blob.
  + Danh sách block chưa commit – Đây là danh sách các block đã được tải lên cho một blob những chưa được commit vào blob.

Bạn có thể tải một blob có kích thước lên đến 64MB chỉ sử dụng một request PUT blob . Để tải lên các blob có kích thước lên đến 50GB, bạn cần phải ghép các block lại với nhau.

## Phiên bản

Với tất cả các giải pháp của bộ lưu trữ Windows Azure, chúng tôi đã giới thiệu header HTTP mới gọi được gọi là “x-ms-version”. Tất cả những thay đổi đến API sẽ được đặt tên phiên bản trên header này. Điều này cho phép các phiên bản trước của các lệnh xử lí trên hệ thống lưu trữ tiếp tục làm việc, và chúng ta mở rộng khả năng của các lệnh hiện tại và giới thiệu những lệnh mới.

x-ms-version nên được chỉ rõ cho tất cả các request đến bộ lưu trữ Windows Azure. Nếu có một request anonymous mà không có phiên bản, thì phiên bản cũ nhất của lệnh sẽ được xử lí trên hệ thống lưu trữ.

Phiên bản được hỗ trợ hiện tại là “x-ms-version: 2009-04-14”. Phiên bản này có thể được sử dụng cho tất cả các lệnh và request gửi đến bộ lưu trữ Windows Azure. Chức năng mới chúng tôi giới thiệu cho Windows Azure Blob với phiên bản này là CopyBlob và phiên bản mới của GetBlockList.

# Một Blob là một danh sách các block

Một trong những chiến lược chính cho Windows Azure Blob là cho phép tải lên các blob có kích thước lên đến nhiều GB. Điều này được cung cấp bởi Windows Azure Blob qua những bước sau:

* Cắt Blob cần tải lên thành nhiều khối liên tiếp nhau. Vd, Một phim 10GB có thẻ cắt thành 2500 khối, mỗi khối có kích thước 4MB, khối đầu tiên đại diện cho byte thứ 1 đến 4194304, khối thứ 2 đại diện cho byte 4194305 đến 8388608.
* Đặt mỗi khối một Tên/ ID duy nhất. ID duy nhất này được giới hạn bởi tên blob được tải lên. Ví dụ, block đầu tiên được gọi là “Block 0001”, block thứ hai “Block 0002”, …
* PUT mỗi block vào đám mây.
* Sau khi tất cả các block đã được lưu trữ trong bộ lưu trữ Windows Azure, sau đó chúng ta commit danh sách các block đã tải lên vào blob mà chúng thuộc về. Điều này được thực hiện với một lênh PUT chỉ định URL ở trên với truy vấn xác định rằng đây là lệnh Blocklist. Sau đó, header HTTP chứa danh sách các block đã được commit cho khối này. Khi thao tác này thực hiện thành công, danh sách các block, theo thứ tự mà chúng được liệt kê, bây giờ đại diện cho phiên bản có thể đọc được của các blob. Blob sau đó có thể được đọc bằng cách sử dụng lệnh GET blob.

Hình dưới đây kết hợp các khối vào khái niệm dữ liệu Windows Azure Blob.

Hình 2 Khái niệm lưu trữ Blob – Thêm Block

Như mô tả trước đó, blob có thể được truy xuất qua PUT và GET bằng cách sử dụng URL sau:

http://<account>.**blob**.core.windows.net/<container>/<blobname>

Ví dụ hình 2, một lệnh PUT có thể được sử dụng để đặt các ảnh theo URL sau:

http://sally.**blob**.core.windows.net/pictures/IMG001.JPG

http://sally.**blob**.core.windows.net/pictures/IMG002.JPG

Khi sử dụng một lệnh PUT đơn, block có kích thước lên đến 64MB có thể được lưu trữ. Để lưu trữ các blob lớn hơn 64MB và lên đến 50GB, đầu tiên cần PUT tất cả các block, và sau đó PUT blocklist để hợp thành phiên bản có thể đọc được của blob này. Trong hình 2 ở trên,chỉ sau khi các block đã được put và commit thành blob, thì blob có thể được đọc theo URL sau:

http://sally.**blob**.core.windows.net/pictures/MOV1.AVI

## Trừu tượng hóa dữ liệu Block

Mỗi block được xác định bởi Block ID và nó được giới hạn bởi tên blob. Vì thế những blob khác nhau có thể có các block với cùng ID. Mỗi block có kích thước lên đến 4MB, và các block trong cùng blob có thể có kích thước khác nhau. Windows Azure Blob cung cấp các thao tác cấp độ Block như sau:

* PUT block – tải một block vào một blob. Chú ý một block được tải lên thành công với thao tác PUT block không trở thành một phần của blob chỉ khi nào nó được commit vào blocklist với thao tác PUT blocklist.
* PUT blocklist – commit một blob bằng cách xác định danh sách các ID block tạo nên blob. Các block được xác định trong hoạt động này phải được tải lên thành công bằng cách sử dụng lời gọi PUT block. Thứ tự các block trong thao tác PUT blocklist, sẽ hợp thành phiên bản có thể đọc của blob.
* GET blocklist – lấy danh sách block đã được commit trước đó cho blob bằng thao tác PUT blocklist. Danh sách block được trả ra định rõ ID và kích thước của mỗi block. Chức năng này có thể được sử dụng để lấy các danh sách block chưa được commit.

Chú ý : ID Block có thể được thấy qua một phần của metadata mà bạn có thể lưu vết cho mỗi block.

Một ví dụ tận dụng ID block là có được ID block ID qua giá trị băm của nội dung dữ liệu qua mỗi block. Trong phương pháp này, đây là một trong những cách có thể kiểm tra tính toàn vẹn dữ liệu của mỗi block dữ liệu nhận được từ hệ thống.

## Blobs – Azure Storage Client v1.0

Mặc dù REST API và các thao tác trong REST API được đọc một cách dễ dàng. Nhưng để thuận tiện hơn, nhóm Windows Azure SDK đã tạo ra thư viện hỗ trợ: Microsoft.WindowsAzure.StorageClient. Thư viện này được sử dụng để gọi REST API của dịch vụ lưu trữ Windows Azure.

Azure Storage Client v1.0 cung cấp các lớp hỗ trợ chức năng Azure Blob với những lớp chính sau:

* *CloudStorageAccount*
* CloudBlob
* CloudBlobClient
* CloudBlobContainer
* CloudBlobDirectory
* CloudBlockBlob
* CloudPageBlob

*CloudStorageAccount* Là một lớp hỗ trợ lấy thông tin tài khoản từ tập tin cấu hình hoặc tạo ra một thực thể của đối tượng tài khoản lưu trữ từ các thông số.

*CloudBlob* là lớp cơ sở cho *CloudBlobDirectory, CloudBlockBlob* và *CloudPageBlob.*Nó cung cấp hầu hết các chức năng để xử lí blob bao gồm : tạo và xóa blob; và duy trì các đặc tính blob bao gồm thuộc tính và metadata.

*CloudBlobDirectory* thêm các chức năng truy xuất blob với các tên được cấu trúc như hệ thống cây thư mục.

*CloudBlockBlob* thêm các chức năng đặc trưng cho Block Blob trong khi *CloudPageBlob* thêm các chức năng đặc trưng cho Page Blob.

*CloudBlobClient* cung cấp chức năng lấy danh sách các blob và tăng thêm truy vấn trực tiếp đến các kiểu blob khác nhau. Nó cũng cung cấp cách thực thuận tiện để kết nối với đối tượng CloudStorageAccount cung cấp quyền truy xuất đến các thông tin xác thực được lưu trữ trong cấu hình Azure. *CloudBlobContainer* cung cấp các chức năng để quản lí container bao gồm tạo, xóa và lấy danh sách.

Ngoài lớp chính trên, còn có những lớp khác như BlobProperty và BlobContainerProperties, CloudPageBlob và CloudBlockBlob…

Các bước để lập trình một ứng dụng Blob đơn giản với các lớp trên như sau:

1. Thêm lệnh sau vào lớp C# của bạn:

using Microsoft.WindowsAzure.StorageClient;

2. Tạo một đối tượng lớp CloudStorageAccount từ tập tin cấu hình:

CloudStorageAccount storageAccountInfo =

CloudStorageAccount.FromConfigurationSetting(configurationSettingName);

3. Hoặc, tạo một đối tượng lớp CloudStorageAccount sử dụng thông tin tài khoản:

CloudStorageAccount storageAccountInfo = new CloudStorageAccount(new

StorageCredentialsAccountAndKey(accountName, accountKey), new Uri(blobEndpointURI), new

Uri(queueEndpointURI), new Uri(tableEndpointURI));

4. Tạo một đối tượng CloudBlobClient:

CloudBlobClient blobStorageType = storageAccountInfo.CreateCloudBlobClient();

5. Khi bạn có một đối tượng lớp CloudBlobClient, bạn có thể xử lí các thao tác trên dịch vụ Blob Storage:

Lấy container :

IList<CloudBlobContainer> containers = new

List<CloudBlobContainer>(this.blobStorageType.ListContainers(prefix,

ContainerListingDetails.All));

Tạo một container:

blobStorageType.GetContainerReference(containerName).CreateIfNotExist();

Tạo một container với các quyền:

CloudBlobContainer container = blobStorageType.GetContainerReference(containerName);

BlobContainerPermissions perm = new BlobContainerPermissions();

perm.PublicAccess = accessType;

container.SetPermissions(perm);

container.Metadata.Add(new NameValueCollection());

container.CreateIfNotExist();

Tạo một blob bằng cách tải lên các mảng byte:

blobStorageType.GetContainerReference(containerName).GetBlobReference(blobName).

UploadByteArray(blobContents);

Tạo một blob bằng cách tải lên text:

blobStorageType.GetContainerReference(containerName).GetBlobReference(blobName)

.UploadText(blobContents);

Tạo một blob bằng cách tải lên một stream:

blobStorageType.GetContainerReference(containerName).GetBlobReference(blobName)

.UploadFromStream(blobContents);

Tạo một blob bằng cách tải lên một tập tin:

blobStorageType.GetContainerReference(containerName).GetBlobReference(blobName).

UploadFile(fileName);

Lấy một blob bằng cách tải xuống một mảng byte:

blobStorageType.GetContainerReference(containerName).GetBlobReference(blobName).

DownloadByteArray();

Lấy một blob bằng cách tải xuống text:

blobStorageType.GetContainerReference(containerName).GetBlobReference(blobName).

DownloadText();

Lấy một blob bằng cách tải xuống một stream:

blobStorageType.GetContainerReference(containerName).GetBlobReference(blobName).

DownloadToStream(outputStream);

Lấy một blob bằng cách tải xuống một file:

blobStorageType.GetContainerReference(containerName).GetBlobReference(blobName).

DownloadToFile(outputFileName);

### CloudBlobClient

Đối tượng CloudBlobClient có thể được tạo hoặc sử dụng một trong các hàm contructor hoặc gọi phương thức mở rộng :

*public static CloudBlobClient CreateCloudBlobClient(CloudStorageAccount account);*

Ví dụ :

CloudStorageAccount cloudStorageAccount = CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
CloudBlobClient cloudBlobClient = new

CloudBlobClient(cloudStorageAccount.BlobEndpoint.ToString(), cloudStorageAccount.Credentials);

Hoặc

CloudStorageAccount cloudStorageAccount = CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

CloudBlobClient cung cấp nhiều phương thức đồng bộ để lấy danh sách các container, cũng như các phương thức không đồng bộ mà còn hỗ trợ phân trang theo danh sách. Chú ý hậu tố Segmented được sử dụng suốt Azure Storage Client để hỗ trợ phân trang. Đối tượng ResultSegment<CloudBlobContainer> được trả về từ phương thức EndListContainersSegmented(). Lớp ResultSegment<TElement> cung cấp các phương thức

BeginGetNext() và EndGetNext() có thể được sử dụng đánh số trang qua danh sách các danh sách các container. Phương thức ListBlobWithPrefix() lấy danh sách các container trong container $root. GetContainerReference() cung cấp tham chiếu đến một container.

Ví dụ sau cho thấy cách sử dụng của GetBlockBlob():

protected void GetBlockBlobStream(String blobAddress)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlockBlob cloudBlockBlob = cloudBlobClient.GetBlockBlob(blobAddress);

    using( MemoryStream memoryStream = new MemoryStream())  
    {  
        cloudBlockBlob.DownloadToStream(memoryStream);  
        Int64 streamSize = memoryStream.Length;  
        memoryStream.Seek(0, SeekOrigin.Begin);  
        using (StreamReader streamReader = new StreamReader(memoryStream))  
        {  
            String blobText = streamReader.ReadToEnd();  
        }  
    }  
}

Ví dụ tạo ra một tham chiếu đến đối tượng CloudBlockBlob và sau đó tải vào [MemoryStream](http://msdn.microsoft.com/en-us/library/system.io.memorystream.aspx). Chú ý rằng trước đó memoryStream đã được sử dụng phương thức Seek() để thiết lập vị trí hiện tại ở đầu stream. Cho một container có tên music chứa một blob có tên British/StoneRoses ,phương thức có thể được gọi :

GetBlockBlobStream("music/British/StoneRoses")

### CloudBlobContainer

Lớp CloudBlobContainer đại diện một container blob. Nó cung cấp các phương thức hỗ trợ tạo và xóa container cũng như cung cấp các chức năng bảo trì như gán và lấy các thuộc tính, metadata và thiết lập bảo mật cho container. Lớp CloudBlobContainer cũng cung cấp các phương thức cho phép truy xuất trực tiếp.

Một CloudBlobContainer được tạo bằng cách sử dụng một trong các constructor và được cung cấp tên container. Ví dụ :

protected void CreateContainer(String containerName)  
{  
    CloudStorageAccount cloudStorageAccount = CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = new CloudBlobContainer(containerName, cloudBlobClient);  
    cloudBlobContainer.Metadata["MetadataName"] = "MetadataValue";  
    cloudBlobContainer.Create();  
}

ListBlobs() có thể được sử dụng để liệt kê các blob trong container :

protected void GetBlobs(String containerName)  
{  
    CloudStorageAccount cloudStorageAccount = CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName);

    IEnumerable<IListBlobItem> blobList = cloudBlobContainer.ListBlobs();  
    foreach (IListBlobItem item in blobList)  
    {  
        Uri uri = item.Uri;  
    }  
}

Thuộc tính container có thể được gán bằng cách sử dụng thuộc tính [Attributes](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblobcontainer.attributes.aspx). Metadata có thể được gán bằng cách sử dụng [Metadata](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblobcontainer.metadata.aspx) [NameValueCollection](http://msdn.microsoft.com/en-us/library/system.collections.specialized.namevaluecollection.aspx) như ví dụ CreateContainer() ở trên. [SetMetadata()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblobcontainer.setmetadata.aspx) có thể được sử dụng để cập nhật Metadata trong một container.

Không có phương thức để kiểm tra một container có tồn tại hay không.

### CloudBlob

Lớp [CloudBlob](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblob.aspx) đại diện một blob và thực thi các chức năng không định rõ cho các lớp đại diện cho block blob và page blob.

##### Blob Addressing

Một blob tồn tại trong một thùng chứ và một phương pháp đơn giản nhất để đánh địa chỉ một blob là sử dụng mẫu containername/blobname, với một URL đầy đủ:

https://myaccount.blob.core.windows.net/containername/blobname

Tuy nhiên tên blob có thể bao gồm dấu phân cách, tiêu biểu như ‘/’, có thể được sử dụng để mô phỏng sự phân cấp cây thư mục trong namespace của các blob. Ví dụ, dưới đây là tất cả các địa chỉ hợp lệ cho blob trong container được đặt tên *music*:

* music/American/BobDylan/PositivelyFourthStreet
* music/American/StevieWonder/InnerVisions
* music/British/Portishead/SourTimes
* music/British/Portishead/WanderingStar

Giản đồ tên thư mục mô phỏng American và British như thư mục cấp cao nhất với BobDylan, StevieWonder và Portishead là thư mục cấp thấp. Thư mục *Portishead* chứa hai blob được xác định bởi SourTimes và WanderingStar mặc dù tên đầy đủ của chúng là British/Portishead/SourTimes và British/Portishead/WanderingStar. Khi gọi

GetDirectoryList(“music/British”, “Portishead”);

Mã bên dưới mô tả cách liệt kê hai blob này:

protected void GetDirectoryList(String topLevelDirectoryName, String subDirectoryName)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");

    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobDirectory topLevelDirectory = cloudBlobClient.GetBlobDirectoryReference(topLevelDirectoryName);

    CloudBlobDirectory subDirectory = topLevelDirectory.GetSubdirectory(subDirectoryName);

    IEnumerable<IListBlobItem> blobItems = subDirectory.ListBlobs();  
    foreach (IListBlobItem blobItem in blobItems)  
    {  
        Uri uri = blobItem.Uri;  
    }  
}

Đối tượng CloudBlobClient có thể được tạo ra sử dụng một trong các hàm constructor hoặc gọi phương thức CloudBlobClient:

public CloudBlob GetBlobReference(String blobAddress);

Các hàm này không tạo ra một blob trong dịch vụ lưu trữ Azure mà chỉ là một đại diện trong bộ nhớ. Việc tạo ra thực tế đòi hỏi tải lên nội dung blob thông qua một trong những phương pháp tải lên. Ví dụ:

protected void UploadText(String containerName, String blobName)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName);

    CloudBlob cloudBlob = cloudBlobContainer.GetBlobReference(blobName);  
    cloudBlob.Metadata[“MetadataName”] = “MetadataValue”;  
    cloudBlob.UploadText("Stately, plump Buck Mulligan came from the stairhead");  
}

Một blob có thể được tạo ra trong dịch vụ lưu trữ bằng cách sử dụng [OpenWrite()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblob.openwrite.aspx) để tạo một đối tượng [BlobStream](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.blobstream.aspx) và sau đó viết vào stream:

protected void WriteToBlobStream(String containerName, String blobName)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName);

    CloudBlob cloudBlob = cloudBlobContainer.GetBlobReference(blobName);

    String blobText = "*bearing a bowl of lather on which a mirror and a razor lay crossed*";

    UTF8Encoding utf8Encoding = new UTF8Encoding();  
    Byte[] bytes = utf8Encoding.GetBytes(blobText);

    using (BlobStream blobStream = cloudBlob.OpenWrite())  
    {  
        blobStream.Write(bytes, 0, bytes.Count<Byte>());  
    }  
}

Khi BlobStream được đóng, nó tải các block và sau đó commit các block.

Nội dung của blob có thể được lấy sử dụng một trong những phương pháp tải về hoặc sử dụng [OpenRead()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblob.openread.aspx) để tạo một đối tượng BlobStream và sau đó đọc từ stream. Ví dụ:

protected void DownloadText(String containerName, String blobName)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName);  
    CloudBlob cloudBlob = cloudBlobContainer.GetBlobReference(blobName);  
    String blobText = cloudBlob.DownloadText();  
}

Và :

protected void ReadBlobStream(String containerName, String blobName)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName);  
    CloudBlob cloudBlob = cloudBlobContainer.GetBlobReference(blobName);

    Byte[] bytes = new Byte[1000];  
    using ( BlobStream blobStream = cloudBlob.OpenRead())  
    {  
        blobStream.Read(bytes, 0, bytes.Count<Byte>());  
    }  
}

Blob có thể được xóa khỏi Dịch vụ Lưu trữ Azure với phương thức Delete() và [DeleteIfExists()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblob.deleteifexists.aspx). Sau đó là một số phương thức thuận tiện xác nhật sự tồn tại của blob trước khi blob được xóa.

Blob có thể được sao chép trực tiếp trên Dịch vụ Lưu trữ Azure sử dụng phương thức [CopyFromBlob()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblob.copyfromblob.aspx)

protected void CopyFromBlob(String containerName, String blobName)  
{  
    CloudStorageAccount cloudStorageAccount =  
      CloudStorageAccount.FromConfigurationSetting("DataConnectionString");

    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = CloudBlobClient.GetContainerReference(containerName);

    CloudBlockBlob cloudBlockBlob = cloudBlobContainer.GetBlockBlobReference(blobName);

CloudBlockBlob copyCloudBlockBlob = cloudBlobContainer.GetBlockBlobReference("Copy" + blobName);

    copyCloudBlockBlob.CopyFromBlob(cloudBlockBlob);  
}

### CloudBlockBlock

Lớp [CloudBlockBlob](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudblockblob.aspx) có nguồn gốc từ lớp CloudBlob và thêm các chức năng riêng biệt cho block blob.

Chức năng CloudBlockBlob được cung cấp cả dạng đồng bộ và không đồng bộ.

Block blob bao gồm một chuỗi liên tục các block được tải lên một cách động lập nhờ phương thức PutBlock() và sau đó commit vào blob sử dụng PutBlockList(). Các Block được xác định bằng block ID. Danh sách block có thể được hiển thị sử dụng DownloadBlockList(). Phần code dưới đây cho thấy một block đơn được tải lên blob qua [MemoryStream](http://msdn.microsoft.com/en-us/library/system.io.memorystream.aspx) :

protected void PutBlockFromStream(String containerName, String blobName, Int32 blockId)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");

    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName)

CloudBlockBlob cloudBlockBlob = cloudBlobContainer.GetBlockBlobReference(blobName);

    String blobText = new String('b', 1000);  
    UTF8Encoding utf8Encoding = new UTF8Encoding();  
    using (MemoryStream memoryStream = new MemoryStream(utf8Encoding.GetBytes(blobText)))  
    {  
        cloudBlockBlob.PutBlock(Convert.ToBase64String(System.BitConverter.GetBytes(blockId)), memoryStream, null);  
    }  
}

Phần dưới hiển thị danh sách block được commit để tạo một blob block trong Dịch vụ Lưu trữ Azure:

protected void PutBlockList(String containerName, String blobName, Int32 numBlocks)  
{  
    CloudStorageAccount cloudStorageAccount   
      CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName);  
    CloudBlockBlob cloudBlockBlob = cloudBlobContainer.GetBlockBlobReference(blobName);

    String[] blockIds = new String[numBlocks];  
    for (Int32 i = 0; i < numBlocks; i++)  
    {  
        blockIds[i] = Convert.ToBase64String(System.BitConverter.GetBytes(i)); ;  
    }

    cloudBlockBlob.PutBlockList(blockIds);  
}

Ví dụ này giả định rằng blockID là các con số tuần tự từ 0 đến nhỏ hơn số lượng các block.

### CloudPageBlob

Một page blob bao gồm một dãy tuần tự các page và định địa chỉ cho tối ưu hóa việc đọc và ghi ngẫu nhiên. Cả page và page blob đều có kích thước tối đa là 1 TB. Còn Block blobs được tối ưu hóa cho stream.

Chỉ có một cách viết vào page blob là qua phương thức [WritePages()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudpageblob.writepages.aspx) và phiên bản không đồng bộ của nó [BeginWritePages()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudpageblob.beginwritepages.aspx) và [EndWritePages()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudpageblob.endwritepages.aspx).

Một page blob được tạo trong Dịch vụ Lưu trữ Azure sử dụng phương thức [Create()](http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.cloudpageblob.create.aspx). Dữ liệu được viết vào các trang riêng lẻ trong một page blob sử dụng WritePages(). Bất kì phương thức tải dữ liệu của CloudBlob đều có thể sử dụng để lấy dữ liệu từ page blob. Khi tải về, bất kì trang nào không được viết sẽ được thiết lập là 0.

Sau đây cho thấy việc tạo một page blob và tải một vài dữ liệu lên page thứ nhất và thứ ba trong bốn page :

protected void CreatePageBlob(String containerName, String blobName)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName);

    CloudPageBlob cloudPageBlob = cloudBlobContainer.GetPageBlobReference(blobName);

    const Int32 pageSize = 512;  
    const Int32 numPages = 4;  
    cloudPageBlob.Create(numPages \* pageSize);

    String blobText = new String('z', pageSize);  
    UTF8Encoding utf8Encoding = new UTF8Encoding();  
    using (MemoryStream memoryStream = new MemoryStream(utf8Encoding.GetBytes(blobText)))  
    {  
        cloudPageBlob.WritePages(memoryStream, 0);  
        memoryStream.Seek(0, SeekOrigin.Begin);  
        cloudPageBlob.WritePages(memoryStream, 2 \* pageSize);  
    }  
}

Sau đây là lấy danh sách của các dãy page trong page blob cũng như tải dữ liệu page blob vào MemoryStream:

protected void GetPageBlob(String containerName, String blobName)  
{  
    CloudStorageAccount cloudStorageAccount =  
       CloudStorageAccount.FromConfigurationSetting("DataConnectionString");  
    CloudBlobClient cloudBlobClient = cloudStorageAccount.CreateCloudBlobClient();

    CloudBlobContainer cloudBlobContainer = cloudBlobClient.GetContainerReference(containerName);

    CloudPageBlob cloudPageBlob = cloudBlobContainer.GetPageBlobReference(blobName);

    IEnumerable<PageRange> pageRanges = cloudPageBlob.GetPageRanges();  
    foreach (PageRange pageRange in pageRanges)  
    {  
        Int64 startOffset = pageRange.StartOffset;  
        Int64 endOffset = pageRange.EndOffset;  
    }

    using (MemoryStream memoryStream = new MemoryStream())  
    {  
        cloudPageBlob.DownloadToStream(memoryStream);  
        Int64 streamSize = memoryStream.Length;  
    }  
}

## Các chiến lược tải block lên

Tải lên một blob theo danh sách các block chúng ta có được các lợi ích sau:

* Sự nối tiếp – khi mỗi block được tải lên thành công điều này có thể chắc chắn thành công của khối này và thử lại khối nếu có thất bại và tiếp tục từ điểm đó.
* Tải lên song song – điều này có thể tải các block lên song song làm giảm thời gian tải của các blob rất lớn.
* Tải lên không theo thứ tự - Bạn có thể tải các block lên mà không theo thứ tự.
* Kết hợp BlockID như Metadata cho mỗi block.

**PutBlock**(BlockId1);

**PutBlock**(BlockId3);

**PutBlock**(BlockId4);

**PutBlock**(BlockId2);

**PutBlock**(BlockId4);

**PutBlockList**(BlockId2, BlockId3, BlockId4);

Figure 4 Chiến lược tải block lên

Ví dụ hình 4 sẽ được sử dụng để giải thích các chiến lược khác nhau có thể gặp khi sử dụng giao tiếp block để tải lên các blob. Ở đó là:

* Tải lên các block với cùng Block ID – Khi các block với cùng ID được tải lên cùng blob, block được úp lên gần nhất cùng với ID sẽ được sử dụng khi commit blob với PUT blocklist. Trong ví dụ trên, hai block với ID = BlockId4 được tải lên và block sau cùng sẽ được sử dụng trong block list commit cho blob.
* Tải các block lên không theo thứ tự - Block có thể được tải lên theo thứ tự khác nhau như trong danh sách block tải lên blob. Trong ví dụ ở trên, danh sách block được commit cuối cùng là những block theo thứ tự BlockId2, BlockId3, BlockId4, nhưng những block này được upload theo thứ tự khác nhau. Dữ liệu blob có thể đọc được (qua GET) được sắp xếp liên quan đến danh sách được đặt tả trong PUT blocklist. Do đó, nếu bạn lấy một blob và đọc blob từ lúc bắt đầu đến khi kết thúc bạn sẽ thấy nội dung của BlockId2, theo sau la BlockId3, sau nữa là Block ID 4.

Các Block không sử dụng – Hơn nữa, một vài block có thể không bao giờ lấy được commit vào danh sách block cuối cùng trong blob. Các block này sẽ được dọn dẹp bởi hệ thống. BlockId1 và Block với ID BlockId4 đầu tiên trong ví dụ sẽ được thu dọn.