**The lecture 14**

**Azure blob storage**

Azure Blob storage is Microsoft's object storage solution for the cloud. Blob storage is optimized for storing massive amounts of unstructured data. Unstructured data is data that doesn't adhere to a particular data model or definition, such as text or binary data.

**About Blob storage**

Blob storage is designed for:

* Serving images or documents directly to a browser.
* Storing files for distributed access.
* Streaming video and audio.
* Writing to log files.
* Storing data for backup and restore, disaster recovery, and archiving.
* Storing data for analysis by an on-premises or Azure-hosted service.

Users or client applications can access objects in Blob storage via HTTP/HTTPS, from anywhere in the world. Objects in Blob storage are accessible via the [Azure Storage REST API](https://docs.microsoft.com/rest/api/storageservices/blob-service-rest-api), [Azure PowerShell](https://docs.microsoft.com/powershell/module/azure.storage), [Azure CLI](https://docs.microsoft.com/cli/azure/storage), or an Azure Storage client library. Client libraries are available for different languages, including:

* [.NET](https://docs.microsoft.com/dotnet/api/overview/azure/storage/client)
* [Java](https://docs.microsoft.com/java/api/overview/azure/storage)
* [Node.js](https://azure.github.io/azure-storage-node)
* [Python](https://docs.microsoft.com/azure/storage/blobs/storage-quickstart-blobs-python)
* [Go](https://github.com/azure/azure-storage-blob-go/)
* [PHP](https://azure.github.io/azure-storage-php/)
* [Ruby](https://azure.github.io/azure-storage-ruby)

**About Azure Data Lake Storage Gen2**

Blob storage supports Azure Data Lake Storage Gen2, Microsoft's enterprise big data analytics solution for the cloud. Azure Data Lake Storage Gen2 offers a hierarchical file system as well as the advantages of Blob storage, including:

* Low-cost, tiered storage
* High availability
* Strong consistency
* Disaster recovery capabilities

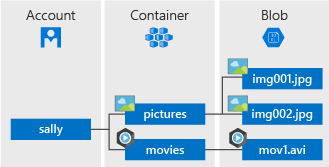
For more information about Data Lake Storage Gen2, see [Introduction to Azure Data Lake Storage Gen2](https://docs.microsoft.com/en-gb/azure/storage/data-lake-storage/introduction).

**Blob storage resources**

Blob storage offers three types of resources:

* The **storage account**.
* A **container** in the storage account
* A **blob** in a container

The following diagram shows the relationship between these resources.



**Storage accounts**

A storage account provides a unique namespace in Azure for your data. Every object that you store in Azure Storage has an address that includes your unique account name. The combination of the account name and the Azure Storage blob endpoint forms the base address for the objects in your storage account.

For example, if your storage account is named *mystorageaccount*, then the default endpoint for Blob storage is:

http://mystorageaccount.blob.core.windows.net

To create a storage account, see [Create a storage account](https://docs.microsoft.com/en-gb/azure/storage/common/storage-account-create). To learn more about storage accounts, see [Azure storage account overview](https://docs.microsoft.com/en-gb/azure/storage/common/storage-account-overview?toc=%2fazure%2fstorage%2fblobs%2ftoc.json).

**Containers**

A container organizes a set of blobs, similar to a directory in a file system. A storage account can include an unlimited number of containers, and a container can store an unlimited number of blobs.

**Blobs**

Azure Storage supports three types of blobs:

* **Block blobs** store text and binary data, up to about 4.7 TB. Block blobs are made up of blocks of data that can be managed individually.
* **Append blobs** are made up of blocks like block blobs, but are optimized for append operations. Append blobs are ideal for scenarios such as logging data from virtual machines.
* **Page blobs** store random access files up to 8 TB in size. Page blobs store virtual hard drive (VHD) files and serve as disks for Azure virtual machines. For more information about page blobs, see [Overview of Azure page blobs](https://docs.microsoft.com/en-gb/azure/storage/blobs/storage-blob-pageblob-overview)

For more information about the different types of blobs, see [Understanding Block Blobs, Append Blobs, and Page Blobs](https://docs.microsoft.com/rest/api/storageservices/understanding-block-blobs--append-blobs--and-page-blobs).

**Move data to Blob storage**

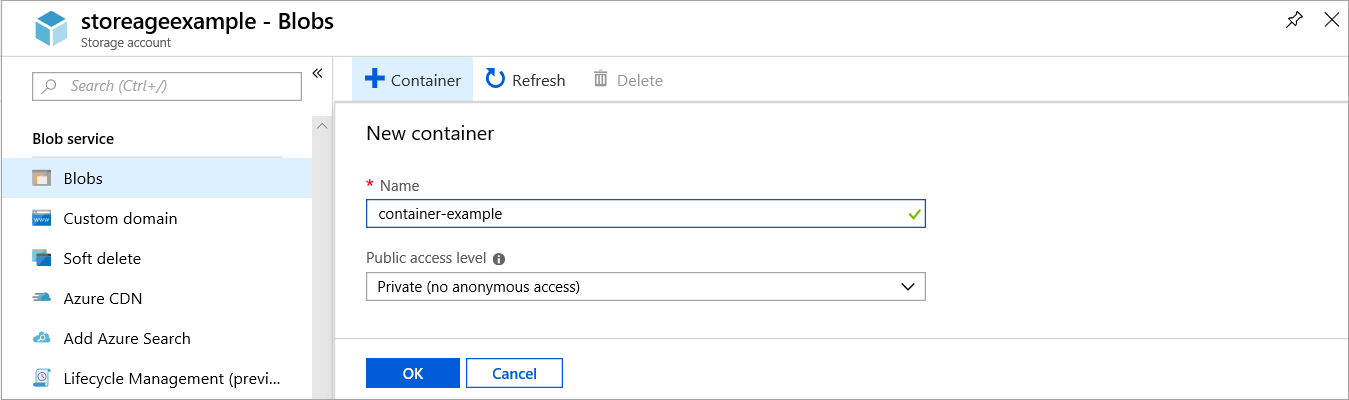
A number of solutions exist for migrating existing data to Blob storage:

* **AzCopy** is an easy-to-use command-line tool for Windows and Linux that copies data to and from Blob storage, across containers, or across storage accounts. For more information about AzCopy, see [Transfer data with the AzCopy v10 (Preview)](https://docs.microsoft.com/en-gb/azure/storage/common/storage-use-azcopy-v10).
* The **Azure Storage Data Movement library** is a .NET library for moving data between Azure Storage services. The AzCopy utility is built with the Data Movement library. For more information, see the [reference documentation](https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.storage.datamovement) for the Data Movement library.
* **Azure Data Factory** supports copying data to and from Blob storage by using the account key, a shared access signature, a service principal, or managed identities for Azure resources. For more information, see [Copy data to or from Azure Blob storage by using Azure Data Factory](https://docs.microsoft.com/azure/data-factory/connector-azure-blob-storage?toc=%2fazure%2fstorage%2fblobs%2ftoc.json).
* **Blobfuse** is a virtual file system driver for Azure Blob storage. You can use blobfuse to access your existing block blob data in your Storage account through the Linux file system. For more information, see [How to mount Blob storage as a file system with blobfuse](https://docs.microsoft.com/en-gb/azure/storage/blobs/storage-how-to-mount-container-linux).
* **Azure Data Box** service is available to transfer on-premises data to Blob storage when large datasets or network constraints make uploading data over the wire unrealistic. Depending on your data size, you can request [Azure Data Box Disk](https://docs.microsoft.com/en-gb/azure/databox/data-box-disk-overview), [Azure Data Box](https://docs.microsoft.com/en-gb/azure/databox/data-box-overview), or [Azure Data Box Heavy](https://docs.microsoft.com/en-gb/azure/databox/data-box-heavy-overview) devices from Microsoft. You can then copy your data to those devices and ship them back to Microsoft to be uploaded into Blob storage.
* The **Azure Import/Export service** provides a way to import or export large amounts of data to and from your storage account using hard drives that you provide. For more information, see [Use the Microsoft Azure Import/Export service to transfer data to Blob storage](https://docs.microsoft.com/en-gb/azure/storage/common/storage-import-export-service).

**Create a container**

To create a container in the Azure portal, follow these steps:

1. Navigate to your new storage account in the Azure portal.
2. In the left menu for the storage account, scroll to the **Blob service** section, then select **Containers**.
3. Select the **+ Container** button.
4. Type a name for your new container. The container name must be lowercase, must start with a letter or number, and can include only letters, numbers, and the dash (-) character. For more information about container and blob names, see [Naming and referencing containers, blobs, and metadata](https://docs.microsoft.com/rest/api/storageservices/naming-and-referencing-containers--blobs--and-metadata).
5. Set the level of public access to the container. The default level is **Private (no anonymous access)**.
6. Select **OK** to create the container.

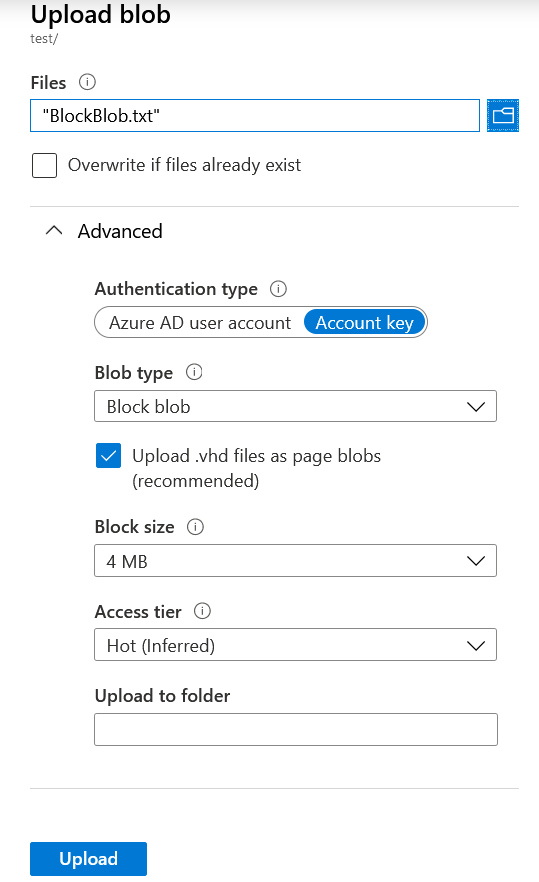


**Upload a block blob**

Block blobs consist of blocks of data assembled to make a blob. Most scenarios using Blob storage employ block blobs. Block blobs are ideal for storing text and binary data in the cloud, like files, images, and videos. This quickstart shows how to work with block blobs.

To upload a block blob to your new container in the Azure portal, follow these steps:

1. In the Azure portal, navigate to the container you created in the previous section.
2. Select the container to show a list of blobs it contains. Since this container is new, it won't yet contain any blobs.
3. Select the **Upload** button to open the upload blade
4. Browse your local file system to find a file to upload as a block blob



1. Optionally, expand the advanced section to define other setting such as authentication type, access tier, or virtual folder path.
2. Select the **Upload** button to commit the upload
3. Upload as many blobs as you like in this way. You'll see that the new blobs are now listed within the container.

**Download a block blob**

You can download a block blob to display in the browser or save to your local file system. To download a block blob, follow these steps:

1. Navigate to the list of blobs that you uploaded in the previous section.
2. Right-click the blob you want to download, and select **Download**.

**Clean up resources**

To remove the resources you created in this quickstart, you can delete the container. All blobs in the container will also be deleted.

To delete the container:

1. In the Azure portal, navigate to the list of containers in your storage account.
2. Select the container to delete.
3. Select the **More** button (**...**), and select **Delete**.
4. Confirm that you want to delete the container.