# Azure storage provider

This provider is designed to replace standard Kentico Azure provider to ensure faster loading times and smooth experience when working with larger sets of data such as Media Library files.

## File list

* k10.azurestorageprovider.7z

## Installation

1. Unzip all files into CMS/bin folder (confirm override – this provider is using new versions of Azure libraries)
2. Adjust web.config file with following configuration:

## Web.config keys

* **CMSAzurePublicContainer**
  + True|False

Pick true or false based on Azure container settings.

* **CMSAzureRootContainer**
  + {containerName}

Add name of BLOB container.

* **CMSAzureCDNEndpoint**
  + {endpointURL}

Add endpoint URL.

* **CMSAzureAccountName**
  + {accountName}

Add account name.

* **CMSAzureSharedKey**
  + {accountSharedKey}

Add account shared key.

* **CMSExtenalStorageName**
  + azure

Add external storage name – pick one for this provider, e.g. azure.

* **CMSStorageProviderAssembly**
  + AzureStorageProvider
* **CMSAzureCachePath**
  + {cachePath}

Folder in root of your website to store cached files from BLOB storage.

* **AzureStorageProviderCacheType**
  + None
  + Memory
  + FileSystem

Pick one of the cache storage options to cache **data** of BLOBs

* **AzureStorageProviderCacheClearMinutes**
  + {minutes}

How long should file contents be cached?

* **AzureStorageProviderIgnoreLastWriteTime**
  + True|**False**

Instead of evaluating last write time of folder to max of all blobs last write times, DateTime.MinValue is used. This ensures backwards compatibility with the original provider. Also for Kentico features, this value is not needed, so keep it set to true by default.

* **AzureStorageProviderForceLowercase**
  + True|**False**

Before writing/reading files all file names will be lowercased. This ensures backwards compatibility with the original provider.

* **AzureStorageProviderInitializeAtAppStart**
  + True|**False**  
    If true, provider will fetch all metadata at application start.

### Example:

<add key="CMSAzurePublicContainer" value="true" />

<add key="CMSAzureRootContainer" value="cmsstorage" />

<add key="CMSAzureCDNEndpoint" value="{get from azure}" />

<add key="CMSAzureAccountName" value="{get from azure}" />

<add key="CMSAzureSharedKey" value="{get from azure}" />

<add key="CMSExternalStorageName" value="azure" />

<add key="CMSStorageProviderAssembly" value="AzureStorageProvider" />

<add key="CMSAzureCachePath" value="AzureCache" />

<add key="AzureStorageProviderCacheType" value="None" />

<add key="AzureStorageProviderCacheClearMinutes" value="0" />

<add key="AzureStorageProviderIgnoreLastWriteTime" value="true" />

<add key="AzureStorageProviderForceLowercase" value="false" />

<add key="AzureStorageProviderInitializeAtAppStart" value="false" />

## How does it work?

This provider works lazily and keeps track of files and folders that were requested to be initialized. Metadata of all files required by the website are stored in application memory and live there throughout the whole life cycle of an application pool.

If caching is enabled, provider also stores actual data of files in designated folder. These data are cleared after specified amount of minutes. However, metadata are still kept in memory. For complete refresh of data restart is required.

Provider supports web farms. If a file is changed on one instance, it is reinitialized on all other instances, i.e. all other instances ask BLOB storage for current data. Folders work similarly.

## Limitations

Due to the nature of this provider’s implementation, direct changes on BLOB storage are not supported. These changes may not be detected easily, therefore to see complete set of data after performing any direct changes on BLOB storage you need to restart the application / recycle the application pool.

## Known issues

### Eye icon in Media Libraries

Due to a possible bug in Media Libraries application, following change is required to achieve full functionality.

~/CMSModules/MediaLibrary/Controls/MediaLibrary/MediaView.ascx.cs, line 402  
Comment out condition outside setting isExternal variable to true:

//if (!filePath.StartsWithCSafe(rootPath))

//{

isExternal = true;

//}

This workaround will ensure the View icon in Media Library overview is working.

### Subfolders in Media Selector

This issue occurs when you have many subfolders in any media library. When selecting media file from Media Selector dialog Kentico is trying to acquire URL of folders which is not possible.

~/CMSModules/MediaLibrary/Controls/MediaLibrary/MediaView.ascx.cs, line 370 contains assignment to value ext. Add following condition underneath:

if (ext == "<dir>")

return string.Empty;

## Disclaimer

**This provider is provided as-is without any guarantees. It is available as open-source on Kentico GitHub space.** If you find any issue, please let us know via GitHub Issues.