Media libraries represent a data storage for your MVC application. They can store different types of audio, video, image, and document files. For a complete list of the file types supported by default, see Supported file types in Media libraries.

You can retrieve media library files in MVC applications by using the same Kentico API methods as you would when developing custom Kentico features.

By default, content of media libraries is synchronized via web farms and duplicated between Kentico and your MVC application. If you need to store media files in one location, see Configuring custom storage for media libraries.

Retrieving media library files in MVC applications

The following example demonstrates how to retrieve files from Kentico media libraries in MVC applications. The example requires a media library with the code name *SampleMediaLibrary* to work properly.

Tip: To view the full code of a functional example directly in Visual Studio, download the Kentico MVC solution from GitHub and inspect the **LearningKit** project. You can also run the Learning Kit website after connecting the project to a Kentico database.

1. Open your MVC project in Visual Studio.
2. Create a new controller class or edit an existing one.
3. Implement a GET action method to retrieve media files.
   The controller action retrieves .jpg files from the *SampleMediaLibrary* media library using the methods from the *CMS.MediaLibrary* namespace.

```
using System.Collections.Generic;
using System.Web.Mvc;

using CMS.MediaLibrary;
using CMS.SiteProvider;
```

Retrieving media files in a controller action

```
            // Creates an instance of the 'SampleMediaLibrary' media library for
the current site
            MediaLibraryInfo mediaLibrary = MediaLibraryInfoProvider.
GetMediaLibraryInfo("SampleMediaLibrary", SiteContext.CurrentSiteName);

            // Gets a collection of media files with the .jpg extension from the
media library
            IEnumerable<MediaFileInfo> mediaLibraryFiles = MediaFileInfoProvider.
GetMediaFiles()
                .WhereEquals("FileLibraryID", mediaLibrary.LibraryID)
                .WhereEquals("FileExtension", ".jpg");
```

You can now pass the media library files to a view or, optionally, create a view model for media files and pass it instead.

Displaying media library files on MVC sites

Example

In order to display media library files on your MVC site, you need to generate URLs for the media files.

1. Open your MVC application in Visual Studio.
2. Install the **Kentico.MediaLibrary** and **Kentico.Content.Web.Mvc** NuGet integration packages into your MVC project.

3. Create a view that displays the media library files. To resize image files, you can use the *ImageUrl(string path, SizeConstraint constraint)* method.

Media file view example

```
@using Kentico.Content.Web.Mvc
@using Kentico.MediaLibrary
@using Kentico.Web.Mvc
@using CMS.MediaLibrary

@model IEnumerable<MediaFileInfo>

<h2>Media library files listing</h2>

@foreach (MediaFileInfo mediaFile in Model)
{
    @* Gets a direct URL for the media file and limits the file's maximum side
size to 400 px *@
    string url = Url.Kentico().ImageUrl(mediaFile.GetPermanentUrl(),
SizeConstraint.MaxWidthOrHeight(400));

    <img src="@url" alt="@mediaFile.FileTitle" />
}
```

The media library files are shown on the MVC site.

Generating media library file URLs

The **Kentico.MediaLibrary** NuGet integration package adds two extension methods to the *MediaFileInfo* class:

- **GetUrl** – returns a direct URL to the media file, for example, *~/Kentico/DancingGoat/media/Images/sample_image.jpg*.
- **GetPermanentUrl** – returns a permanent URL to the media file, for example, *~/getmedia/0140bccc-9d47-41ea-94a9-ca5d35b2964c/sample_image.*

> You can use both methods to generate URLs for media libraries stored in Kentico.
>
> Use the *GetPermanentUrl* method to allow checking of media library permissions. See Assigning permissions to media libraries for more details.

The *ImageUrl* method in the example uses the *MaxWidthOrHeight* size constraint parameter to limit the maximum side size of the displayed image.

You can use the following size constraints:

- *SizeConstraint.Empty* – leaves the image as-is
- *SizeConstraint.Height(100)* – resizes the image to the specified height and maintains aspect ratio
- *SizeConstraint.MaxWidthOrHeight(100)* – resizes the image so that its width and height do not exceed the specified value (maintains aspect ratio)
- *SizeConstraint.Size(100, 120)* – resizes the image to the specified width and height, and does not maintain aspect ratio
- *SizeConstraint.Width(100)* – resizes the image to the specified width *maintains aspect ratio)

> **Note**: The resized images are never larger than the original.

To force a file download, you can use the *FileUrl(string path, FileUrlOptions options)* method:

```
    @using CMS.MediaLibrary
    @using Kentico.Web.Mvc
    @using Kentico.MediaLibrary
    @using Kentico.Content.Web.Mvc
```

```
    // Gets a direct URL for the media file and forces download
    Url.Kentico().FileUrl(mediaFile.GetUrl(), new FileUrlOptions() {
AttachmentContentDisposition = true })
```

Media libraries in an external storage

To store media library files in an external storage (cloud-based file system), you need to:

1. Configure your MVC application to use the given external storage provider for the media library folder. The instructions for configuring MVC applications are the same as for Kentico projects.
    - For Azure Blob storage, see Configuring Azure storage.
    - For Azure CDN, see Configuring Azure CDN.
    - For Amazon S3, see Configuring Amazon S3.
2. In the views that display the media library files, always generate URLs by calling the **GetPermanentUrl** extension method for *MediaFileInfo* objects.

Note: URLs generated by the *GetUrl* method for external storage files do not work on MVC applications.