Image variants are user-defined versions of page attachments. You can define each image variant differently so that you have images for various scenarios. For example, you can create image variant definitions for different browser viewports or for various devices such as mobile phones, tablets, desktops, etc.

Each image variant is described by one **image variant definition**. The image variant definition describes what operations need to be applied to page attachments in order to create image variants.

The image operations are defined by filters. How you use the filters in an image variant definition is up to you. For example, you can use multiple filters or use a filter repeatedly in an image variant definition to achieve the desired effect.

By default, no definitions of image variants are included in the system. To add an image variant definition, you need to create a new class in the Kentico project and implement the image variant definition in code.

Example

Creating image variant definitions

The following example describes how to create a sample image variant definition for a low resolution image variant. The definition uses the sample crop filter to modify images.

1. Open your project in Visual Studio (using the **WebSite.sln** or **WebApp.sln** file).
2. Create a new class in the **App_Code** folder (or *Old_App_Code* folder if the project is installed as a web application). You can name it, for example, **TabletImageVariantDefinition.cs**.

   Using the class in your own project

   If you want to use the code file in a separate library or external application, you need to allow the system to detect the class in a custom assembly. In your project, navigate to the *Properties/AssemblyInfo.cs* file and add the *AssemblyDiscoverable* assembly attribute:

   ```
   using CMS;

   [assembly: AssemblyDiscoverable]
   ```

3. Add *using* statements for the following namespaces.

   ```
   using System.Collections.Generic;

   using CMS.DocumentEngine;
   using CMS.ResponsiveImages;
   ```

4. Set the class to inherit from the **ImageVariantDefinition** abstract class.
5. Override the **Identifier** and **Filters** properties defined in the *ImageVariantDefinition* class, and implement their functionality.

   - *Identifier* property – Defines the image variant definition identifier. The identifier name cannot start with a number, and can contain only underscore and alphanumeric characters.
   - *Filters* property – Defines a collection of filters that are used to create an image variant. The order of filters in code is the order in which they are applied to images.

```
public class TabletImageVariantDefinition : ImageVariantDefinition
{
        // Defines the image variant definition identifier
        public override string Identifier
        {
                get
                {
                        return "Tablet";
                }
        }

        // Defines a set of filters applied to an image
    public override IEnumerable<IImageFilter> Filters
    {
        get
        {
            return new IImageFilter[]
            {
                new CropImageFilter(768)
            };
        }
    }
}
```

6. Ensure that the system loads the appropriate class when working with the image variant definition by adding the **RegisterImageVariantDefinition** assembly attribute above the class declaration.

```
// Registers the 'TabletImageVariantDefinition' class within the system
[assembly: RegisterImageVariantDefinition(typeof(TabletImageVariantDefinition))]
```

7. On web application projects, build the solution.

Limiting the scope of image variant definitions

By default, image variant definitions are global and can be applied on any image page attachments. When defining an image variant, you can restrict its scope by specifying context information, i.e., site code name, page type, and node alias path.

To limit the scope of image variant definition, you need to override and implement the *ContextScopes* property. The property can contain multiple context scopes.

For example, to limit the application of image variant definition only to articles on the Dancing Goat site, and news posts on the Corporate Site, you can use the following code in your image variant definition class:

```
// Sets context scopes to restrict application of image variant definition
public override IEnumerable<IVariantContextScope> ContextScopes
{
    get
    {
        return new[]
        {
            new AttachmentVariantContextScope().OnSite("DancingGoat")
                .Type("DancingGoat.Article")
                .Path("/Articles", PathTypeEnum.Children),

            new AttachmentVariantContextScope().OnSite("CorporateSite")
                .Type("CMS.News")
                .Path("/News", PathTypeEnum.Children)
        };
    }
}
```

You can use the *OnSite, Type,* and *Path* methods independently and in any order.

Adding a display name

By default, the image variant display name that appears in the administration interface is the same as the identifier set for the image variant definition (in the **Identifier** property).

To assign a custom display name, you need to create a new resource string. The name of the resource string, i.e. the value in the **Key** field, must be in the following format: *AttachmentVariant.<ImageVariantDefinitionIdentifier>*

With the definition of an image variant prepared, you can use it to generate image variants for your page attachments.