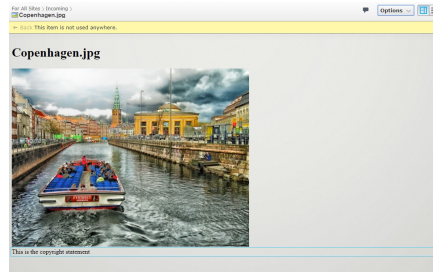


 ALLAN THRAEN | 5 years ago | PDF |

Tips and Tricks CMS Optimizely (EpiServer)

## ON PAGE EDIT FOR IMAGES



**The default image edit mode in EpiServer CMS is a little bit boring - it's nothing but an image tag with the actual image. Why not offer a richer on page edit experience for image media as you typically get for pages and blocks?**

In EpiServer you get pretty used to building `TemplateDescriptor` - typically `Controller` classes that deal with how a specific content type should be controlled and rendered. It can also be other stuff - like full `HttpHandlers` with a `[TemplateDescriptor]` attribute, but that's another story :-)

Anyway - since you typically have to write all of the template descriptors yourself, you might not always realize just how easy it is to replace the few built-in default controllers available. For example, EpiServer comes out-of-the-box with Media Edit controllers (for images, video and other media). They are located in the namespace `EpiServer.Cms.Shell.UI.Controllers.Preview.Internal` and are extremely basic - for images for instance it does nothing but render out the `<img>` tag.

Decompiling it, it looks like this:

```
[TemplateDescriptor(AvailableWithoutTag = false, Inherited = true, TagString = "Edit", TemplateTypeCat
public class ImageEditController : MediaEditController<IContentImage>
{
    private readonly UrlResolver _urlResolver;
    private readonly TemplateResolver _templateResolver;

    public ImageEditController()
        : this(ServiceLocator.Current.GetInstance<UrlResolver>(), ServiceLocator.Current.GetInstance
    }

    public ImageEditController(UrlResolver urlResolver, TemplateResolver templateResolver)
    {
        this._urlResolver = urlResolver;
        this._templateResolver = templateResolver;
    }

    protected override string GetPreviewContent(IContentImage content)
    {
        return new TagBuilder("img")
        {
            Attributes = {
                {
                    "src",
                    content.PreviewUrl(this._urlResolver, this._templateResolver)
                },
                {
                    "alt",
                    content.Name
                }
            }
        }.ToString(TagRenderMode.SelfClosing);
    }
}
```

Note: When dealing with media there are 3 different kinds of template descriptors out the box:

1. The `HttpHandler` that works across all rendering tags and will return the actual binary file (based on `EpiServer.Web.Internal.ContentMediaHttpHandler`)
2. The `HttpHandler` for Preview (which is somewhat similar to #1, but with different caching).
3. The `MediaEditController` (and classes inheriting from it) that will return the HTML for the On Page Edit.

For now, we want to replace #3 with another that actually gives us real on-page-edit capabilities and not just a static image.

Now, since there already exist controllers, to deal with this, we need to override them. Luckily, there is great documentation available that describe in which order template descriptors are picked. Based on this order, we could either hook into an event and pick the handler we want, or we could just make sure it's earlier in the hierarchy. For instance by putting it closer in the inheritance chain; The default one maps to `IContentImage` - we'll instead use `ImageData` (which implements `IContentImage`, and which for instance the `ImageFile` in Alloy is based on).

We'll of course make sure that it's only uses this code for the Edit rendering tag and then we can make a completely standard template controller and view.

```
1 using System.Linq;
2 using System.Web.Mvc;
3 using EpiServer.Core;
4 using EpiServer.Framework.DataAnnotations;
5 using EpiServer.Framework.Web;
6 using SmallExperiments.Business;
7 using SmallExperiments.Models.Pages;
8 using SmallExperiments.Models.ViewModels;
9 using EpiServer.Web;
10 using EpiServer.Web.Mvc;
11 using EpiServer;
12 using EpiServer.Framework.Web.Mvc;
```

```
13 using SmallExperiments.Models.Media;
14 using EPIServer.Shell;
15 using EPIServer.Web.Routing;
16 using EPIServer.ServiceLocation;
17 using EPIServer.Cms.Shell;
18
19 namespace SmallExperiments.Controllers
20 {
21     [TemplateDescriptor]
22     Inherited = true,
23     TemplateTypeCategory = TemplateTypeCategories.MvcController,
24     Tags = new[] { RenderingTags.Edit },
25     AvailableWithoutTag = false, ModelType = typeof(ImageData)]
26     [VisitorGroupImpersonation]
27     [RequireClientResources]
28     public class ImagePreviewController : ControllerBase, IRenderTemplate<ImageData>, IRenderTemplate
29     {
30         private readonly UriResolver _urlResolver;
31         private readonly TemplateResolver _templateResolver;
32
33         public ImagePreviewController()
34             : this(ServiceLocator.Current.GetInstance<UriResolver>(), ServiceLocator.Current.GetInstance<TemplateResolver>())
35         {
36         }
37
38         public ImagePreviewController(UriResolver urlResolver, TemplateResolver templateResolver)
39         {
40             this._urlResolver = urlResolver;
41             this._templateResolver = templateResolver;
42         }
43
44         public ActionResult Index(Content currentContent)
45         {
46             var url = (currentContent as IContentImage).PreviewUrl(_urlResolver, _templateResolver);
47             ViewBag.ImageUrl = url;
48             return View(currentContent);
49         }
50     }
51
52
53 }
```

ImagePreviewController.cs hosted with ❤️ by GitHub

[view raw](#)

```
1 @using SmallExperiments
2 @model SmallExperiments.Models.Media.ImageFile
3 @{
4     Layout = null;
5 }
6
7 <!DOCTYPE html>
8
9 <html>
10 <head>
11     <meta name="viewport" content="width=device-width" />
12     <title></title>
13 </head>
14 <body>
15     <div>
16         <h>@Html.PropertyFor(m => m.Name)</h>
17         
18     </div>
19     @Html.PropertyFor(m -> m.Copyright)
20 </div>
21 </body>
22 </html>
```

Index.cshtml hosted with ❤️ by GitHub

[view raw](#)

[Tips and Tricks](#) | [CMS](#) | [Optimizely \(Episerver\)](#)

RECENT POSTS

## CodeArt ApS

Teknikerbyen 5, 2830 Virum, Denmark

Email: [info@codeart.dk](mailto:info@codeart.dk)

Phone: +45 26 13 66 96

CVR: 39680688



Copyright © 2024