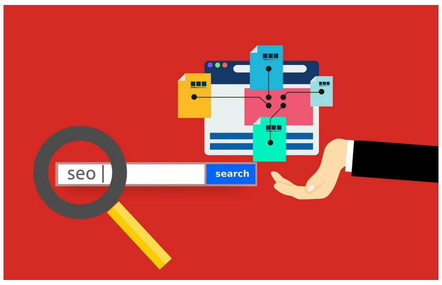
ALLAN THRAEN | 6 years ago | PDF |

Tips and Tricks Optimizely (Episerver)

SITE MAP GENERATION WITH CUSTOM FILTER



There's a handful of great add-ons I use in almost every Episerver project - and several of them are from Geta. Here is a useful hint if you, like me, use the Geta SiteMap generator.

An easy way to have the classic sitemap.xml generated for you, on your Episerver site, is of course to use the de-facto standard: Geta's open source Sitemap generator.

https://github.com/Geta/SEO.Sitemaps/tree/master/src/Geta.SEO.Sitemaps

Even though it's been around for quite a while I continue to be amazed at how well engineered it is - and how you can solve pretty much every need with it.

Recently I was looking for a way to have a property on the individual pages manage the visibility in the sitemap - and sure enough - that's an out-of-the-box feature.

And today I needed to write a bit of code to dynamically decide which pages should appear in the sitemap, based on some rather complex business rules. At first I assumed that I would probably have to fork the project to add this functionality, but luckily, it turns out that it implements it's own version of IContentFilter, managed by Episervers dependency injection. The default implementation does a lot of great filtering, so I simply opted to inherit from it and then extend on it with custom filtering logic.

```
[ServiceConfiguration(typeof(IContentFilter))]
public class CustomSiteMapContentFilter : ContentFilter
{

    public override bool ShouldExcludeContent(IContent content)
                                                        {
        bool rt= base.ShouldExcludeContent(content);
        if (!rt)
        {
            //Do custom logic
            rt = (bool) content.Name.ToLower().StartsWith("a");
        }

        return rt;
    }
}
```

Note, that to make the above example easily readable I replaced my custom logic with a simple example to remove content with names starting with "a". Useless, sure - but would be fun to see if it's caught in test (in which case you might be testing too much) :-)

In the above example it's registered for dependency injection using attributes - however since only 1 IContentfilter is applied, it's important that it's registered last - so I later moved it to an IConfigurable initialization module set to run after the Sitemap had initialized.

Tips and Tricks Optimizely (Episerver)

CodeArt ApS
Teknikerbyen 5, 2830 Virum, Denmark
Email: info@codeart.dk
Phone: +45 26 13 66 96
CVR: 39680688