ALLAN THRAEN | 5 years ago | PDF |

ElasticSearch Tips and Tricks Optimizely (Episerver)

POWERSLICE: IDENTIFY UNUSED BLOCKS

Powerslice might have a few years on it's back, but it's still a great editorial tool, when you are working with large amounts of content and have access to Episerver Find. Here's an example of a recent slice I made that let's editors easily identify unused blocks.

Powerslice is a nice widget that gives the editors an alternative to the classic tree navigation when working with content.

In the backend it is powered by Episerver Find, and as a developer it's pretty easy to extend with your own views of the content.

Typical extension points are:

1. Slices with different filters for content. For example a slice that lists all ArticlePages that are not 'Shortcut to another page'. Or a list of blocks that was created by the current editor - or maybe a list of all the pages that does not have proper SEO meta-data set?
2. Custom sort orders for the slices. Maybe you want to sort by content type? Or what about showing the oldest published content (that probably needs a rewrite)? Sometimes it could also be interesting to see a list of blocks sorted by which are used in the most places (example below).
3. Finally you can also provide editors with a shortcut to create content of a specific type in a specific location. For example, when they are looking at a slice of all Blog Posts, you can let them create a new blog post in the correct location - straight from the widget.

A client of mine had a specific problem that I think many others can recognize. They have many blocks - placed all over the possible structure - and they need to clean them out a bit - evaluate the unused ones to see if they should be deleted or archived. So - here is one approach I made to this problem:

Initial challenge is that the usages of a block is not stored on the block itself, and as such it's not indexed in Find and cannot be filtered on. But luckily that's easy to overcome.

I began by creating an extension method for IContent that would essentially return the number of times a content was referenced / used in other content, following a reverse lookup in the SoftLink Repository.

Now, before you start yelling "But Allan, how dare you use the ServiceLocator.Current in a public blog post?" just know, that I tend to take a more pragmatic approach than a dogmatic approach - especially when I want to keep it in a pretty, little extensionmethod - you feel free to pick your own way.

Next, the trick is simply during initialization to configure an indexing convention in Find where you specify that the extension method is to be included when indexing objects of the type you want to add this value to.

Finally, it's just a matter of doing a basic Episerver Find filtering query in the right place in the slice using our new number.

For pure bonus, I also included a sample of an All-Blocks slice, that will allow you to sort based on how frequently the blocks are used - as that may also come in handy.

Enjoy!

```
using EPiServer.Core;
using EPiServer.DataAbstraction;
using EPiServer.ServiceLocation;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Demo.Business.Powerslice
{
    public static class BlockSliceExtensions
    {
        public static int UsageCount(this IContent cnt)
        {
            //Count usages
            var _soft = ServiceLocator.Current.GetInstance<IContentSoftLinkRepository>();
            var links = _soft.Load(cnt.ContentLink, true);
            return links.Count();

        }
```

```
    }
}
```

BlockSliceExtensions.cs hosted with ❤ by GitHub view raw

```
using EPiServer.Core;
using EPiServer.Find;
using EPiServer.ServiceLocation;
using EPiServer.Shell.ContentQuery;
using PowerSlice;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Demo.Business.Powerslice
{
    [ServiceConfiguration(typeof(IContentQuery)), ServiceConfiguration(typeof(IContentSlice))]
    public class BlocksSlice : ContentSliceBase<BlockData>
    {
        public override IEnumerable<SortAction<BlockData>> SortActions => base.SortActions.Union(new SortAction<BlockData>[] {
            new SortAction<BlockData>("Most Popular","popular", (a,b) => a.OrderByDescending(blk => (blk as IContent).UsageCount()))
        });

        public override int SortOrder => 20;

        public override string Name
        {
            get { return "All Blocks"; }
        }
    }
}
```

BlocksSlice.cs hosted with ❤ by GitHub view raw

```
using System;
using System.Linq;
using EPiServer.Core;
using EPiServer.Find.ClientConventions;
using EPiServer.Find.Framework;
using EPiServer.Framework;
using EPiServer.Framework.Initialization;
using Demo.Business.Powerslice;

namespace Demo.Business.Initialization
{
    [InitializableModule]
    [ModuleDependency(typeof(EPiServer.Web.InitializationModule))]
    public class FindIndexingConventionsInit : IInitializableModule
    {
        public void Initialize(InitializationEngine context)
        {
            //Ensure the UsageCount is indexed along with the block.
            SearchClient.Instance.Conventions.ForInstancesOf<BlockData>().IncludeField(x => (x as IContent).UsageCount());
        }

        public void Uninitialize(InitializationEngine context)
        {
            //Add uninitialization logic
        }
    }
}
```

FindIndexingConventionsInit.cs hosted with ❤ by GitHub view raw

```
using EPiServer.Cms.Shell.UI.Rest.ContentQuery;
using EPiServer.Core;
using EPiServer.Find;
using EPiServer.ServiceLocation;
using EPiServer.Shell.ContentQuery;
using PowerSlice;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Demo.Business.Powerslice
{
    [ServiceConfiguration(typeof(IContentQuery)), ServiceConfiguration(typeof(IContentSlice))]
    public class UnusedBlocksSlice : ContentSliceBase<BlockData>
    {

        public override int SortOrder => 750;

        protected override ITypeSearch<BlockData> Filter(ITypeSearch<BlockData> searchRequest, ContentQueryParameters parameters)
        {
            return searchRequest.Filter(f => (f as IContent).UsageCount().Match(0));

        }

        public override string Name
        {
            get { return "Unused Blocks"; }
        }
    }
}
```

UnusedBlocksSlice.cs hosted with ❤ by GitHub view raw

ElasticSearch Tips and Tricks Optimizely (Episerver)

PowerSlice

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