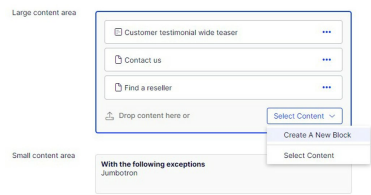


ALLAN THRAEN | 1 years ago | PDF |

Tips and TricksOptimizely (EpiServer)CMS

AUTOMATICALLY HAVE OPTIMIZEZELY CONTENT CREATED IN THE CORRECT PLACE

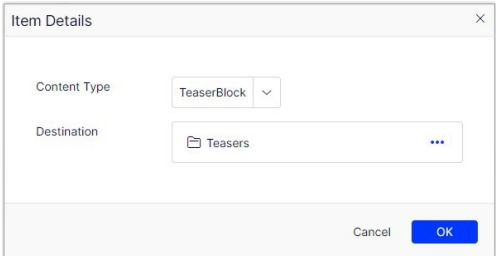


A classic problem is that you want better structure and governance in for example your blocks. Maybe you have a policy to have all Banners or FAQ items in a specific folder so they can easily be found and re-used across the site, but way too often editors take the 'easy' approach and use the "Create a New Block" option in a content area - which by default places the block in the "For this page" folder, rendering re-use impossible. Here is a simple solution to that.

My approach to solving the above problem is pretty straightforward - whenever a content is being created, simply check if it's being created in the right place - and if not, simply change the parent to make sure that it is.

This is easily done in the ContentEvents - and in the CreatingContent event you can modify the content before it's saved for the first time!

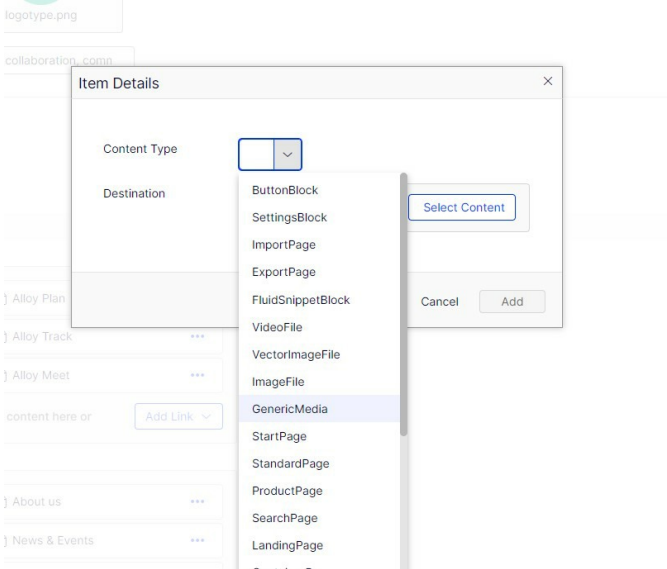
But - in order to make the solution a bit more generic and usable I've included a way to let editors define these rules on a settings page.



Note, this solution doesn't handle the fact that editors can move the content after creation - in which case it seems like a conscious decision. It simply just makes sure that the lazy 'create' button still places content where you want it.

I'd recommend putting the list of rules on a settings content item - in the below example it's on the SiteSettings tab of the start page - but that's for you to decide.

I also included a ContentTypeSelectionFactory. Note, there are many other ContentTypeSelectionFactories in other projects - but none included in the core of Optimizely CMS. It basically just produces a list of the content types for the property when you add a rule. A rule basically just that a certain type of content should go to a certain folder on creation.



```
1 using EpiServer.Plugin;  
2 using EpiServer.Shell.ObjectEditing;  
3 using System.ComponentModel.DataAnnotations;  
4  
5 /// <summary>  
6 /// Rule that specifies where a certain type of content should be put on creation  
7 /// </summary>  
8 public class ContentCreationRule  
9 {  
10     [Display(Name = "Content Type", Order = 10)]  
11     [SelectOne(SelectionFactoryType = typeof(ContentTypeSelectionFactory))]  
12     public virtual string ContentTypeId { get; set; }  
13  
14     [Display(Name = "Destination", Order = 20)]  
15     public virtual ContentReference ParentToMoveTo { get; set; }  
16  
17 }  
18
```

```
19 [PropertyDefinitionTypePlugin]
20 public class ContentCreationRuleList : PropertyList<ContentCreationRule> {}

ContentCreationRule.cs hosted with ♥ by GitHub view raw

1 [ModuleDependency(typeof(InitializationModule))]
2 public class ContentPlacerInit : IInitializableModule
3 {
4     public void Initialize(InitializationEngine context)
5     {
6         var cevents = ServiceLocator.Current.GetInstance<IContentEvents>();
7         cevents.CreatingContent += Cevents_CreatingContent;
8     }
9
10    private void Cevents_CreatingContent(object sender, ContentEventArgs e)
11    {
12        var rules = ServiceLocator.Current.GetInstance<IContentLoader>().Get<Startup>(ContentReference.Startup).ContentCreationRules;
13        var rule = rules.FirstOrDefault(r => r.ContentTypeID == e.Content.ContentTypeID.ToString());
14        if (rule != null)
15        {
16            if (!e.Content.ParentLink.CompareToIgnoreWorkID(rule.ParentToMoveTo))
17            {
18                e.Content.ParentLink = rule.ParentToMoveTo;
19            }
20        }
21    }
22 }
23
24 public void Uninitialize(InitializationEngine context)
25 {
26 }
27 }
```

```
1 using EPIServer.Shell.ObjectEditing;
2
3 public class ContentTypeSelectionFactory : ISelectionFactory
4 {
5     private readonly IContentTypeRepository _repo;
6     public ContentTypeSelectionFactory(IContentTypeRepository repo)
7     {
8         _repo = repo;
9     }
10
11    public IEnumerable<ISelectItem> GetSelections(ExtendedMetadata metadata)
12    {
13        return _repo.List()
14            .Select(r => new SelectItem() { Text = r.Name, Value = r.ID.ToString() });
15    }
16 }
```

```
1 //..
2
3 [Display(GroupName = Globals.GroupNames.SiteSettings)]
4 [EditorDescriptor(EditorDescriptorType = typeof(CollectionEditorDescriptor<ContentCreationRule>))]
5 public virtual IList<ContentCreationRule> ContentCreationRules { get; set; }
6
7 //..

StartupPage.cs hosted with ♥ by GitHub view raw
```

RECENT POSTS