Publishing Dependencies Playbook

Version 0.92

Document History

Author/Editor	Date	Reason for Change	Version
Greg Strickler	12/20/2013	Draft	0.90
Fahd Shaaban	01/02/2014	Edits and Formatting	0.91
Greg Strickler	02/25/2014	Modifications/Playbook Discussion	0.92

Table of Contents

Overview

The publisher has the ability to only publish the current asset and not all the assets that the CMS system deems a dependency. This feature is currently available to all customer instances, and can be disabled at the instance level.

Note: At this time, this capability cannot be enabled or disabled at the group level or user permissions.

Dependencies Overview

Dependencies are relationships between content created by links or by using the API. These dependencies are page-level dependencies that are computed when you publish an asset. An asset using methods in Appendix A, when published, will generate dependencies between itself and the assets involved and will be stored in the dependency table. These dependencies are bi-directional, meaning if either asset is published, the CMS will know to send the other asset to the queue.

Note: Every time an asset is published, that portion of the dependency table will be regenerated prior to the actual publish. This insures that all depending assets are sent to the publishing queue.

The CrownPeak CMS computes dependencies automatically. This is the best approach since adding them manually often leads to a huge unmaintainable and wide structure for example, too many connections and it is hard to maintain. However, there are some API functions to help tune dependencies in certain situations, but these functions not produce the result you want, either too many or too few dependencies.

For example, with a press release asset/template and a press index asset/template, the press index has a child dependency on the press release folder and all the press releases in it since any new release or change could affect the index. The releases themselves have a parent dependency on the index. The home page may have a link to the press index, so any change to a release could affect the press index link (label or filename), which could affect the home page.

API Methods that create dependencies

- GetField
- GetContentFields
- GetCmsUtilityLink
- GetMetaField
- CreateNewAsset
- GetPanels
- GetPanelsFromFolderChildren
- GetFolderList
- GetContent
- GetLink
- AddDependencyTo
- GetMeta
- Show
- GetFileList
- GetFilterList
- Rename

API Methods and Properties

AddDependencyTo(Asset asset)

- Adds a dependency on this asset to the provided asset. So when this asset publishes, it will check the asset or its children and see if they need to be published.
- This makes a bidirectional dependency
- If the user publishes without dependencies, all manually added dependencies will be skipped.
- Asset dependentAsset = Asset.Load("/Site/Press Releases/");
- Asset indexAsset = Asset.Load("/Site/Index");
- indexAsset.AddDependencyTo(dependentAsset);

GetContentFields(List<String>)

Gets a dictionary of fields depending on the fieldNames argument. This function is called on the current asset for all templates before running to load all DB fields. Will look for both fields equaling the name and equaling "upload#" plus the name unless the specified name already starts with upload#. If you don't provide this param, or pass null, or pass an empty list or pass a list that contains a * element, then all fields in the db are returned. Found fields are added to this asset instance's cache so there will be no more DB hits when you access them. If the option to load all fields is taken, then any other field's lookups on the asset instance will not go back to the DB.

OutputContext.IsGeneratingDependencies

This is default set to true, setting to false will disable that asset generating dependencies.

Deleting Dependencies

Clean up the dependency table is necessary at times to remove unused dependencies that can cause the CMS publishing to slow down. The dependency table should remove the connection between the two when an asset is republished and links to the old asset are removed, but this is not always the case. If you want a fresh dependency table you can request CrownPeak Engineering to delete your dependency table by submitting a JIRA ticket *to help@crownpeak.com*. By doing this, you will have a fresh table and when an asset is published it will create all the correct dependencies and store them in the table.

Note: At this time you cannot manually delete a single dependency.

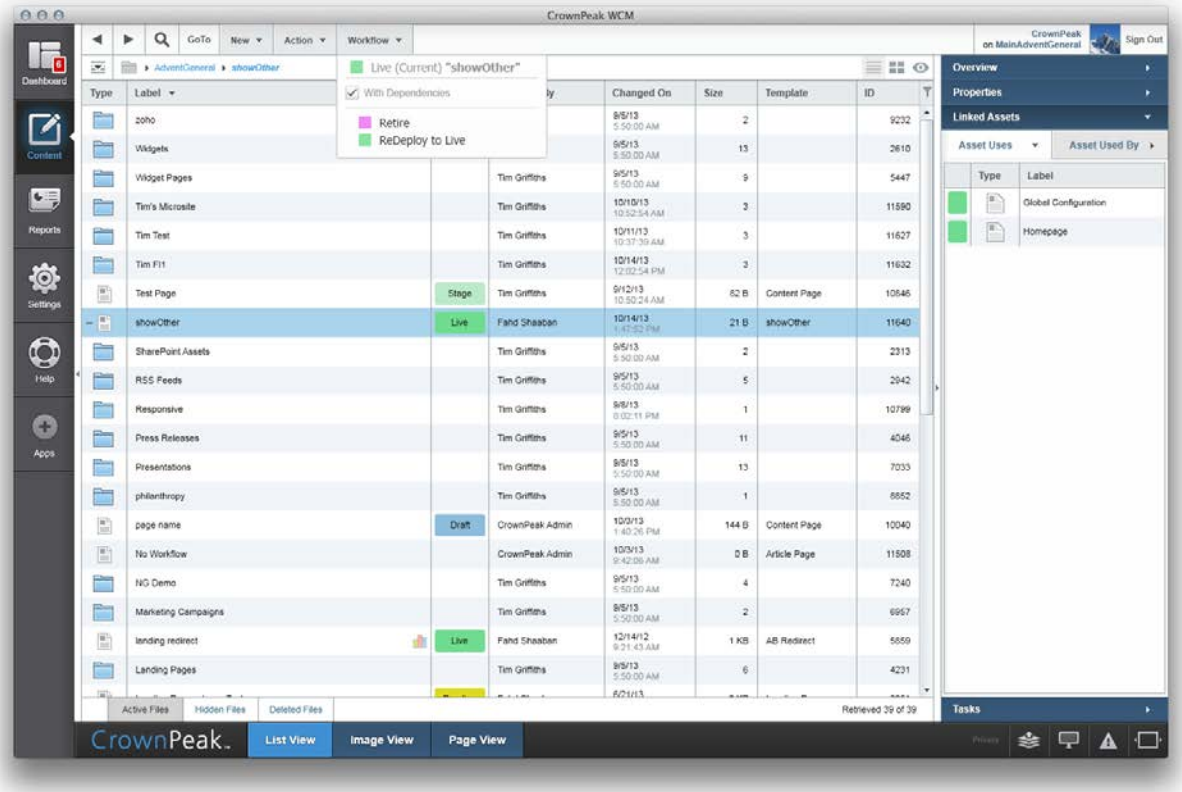
Benefits

Correctly using dependencies tells the CMS only to publish the asset and nothing else. This will speed up the publishing as it will be the only asset being sent to the queue to be published. However, this could cause issues to other linked assets if they did need to be published as well.

This feature is very beneficial when an asset's content is changed but nothing else. That way the only asset that needs to be published can, and all associated assets that do not need to be republished will not be sent to the queue.

Using Dependencies

The With Dependencies checkbox is selected by default. If you remove the checkbox publishing an asset will tell the CMS to ignore publishing all dependencies associated with the asset. Leaving the box checked will publish all dependencies as usual.



Disabling Dependencies

This feature cannot be disabled or enabled at the group level. However, it can be disabled for the whole instance. If you want this feature disabled for the whole instance, submit a JIRA ticket to *help@crownpeak.com* requesting from engineering to disable the ability to uncheck the publishing with dependencies option in the CMS.

Notes

- Post publish should not be used to generate dependencies. All manually added dependencies should be added in the output. This is because you want all dependencies generated before publishing, adding them after publishing will not cause those assets to be published.
- When an asset is published, the CMS builds an asset tree. The depth of nesting can be changed to account for more assets. The depth can be set from 0 to 3, the default is 1, to give 4 levels using the "PublishMaxDependencyDepth" flag. To change this value, the user will need to send a JIRA ticket into *help@crownpeak.com*.

  Note: this is for the whole CMS instance.

- When adding a new press release, on the initial publish, you may consider adding the AddDependencyTo() to the index asset. This will build the relationship so that when that asset is published that the index asset is republished as well. You could also, republish the index asset on its own and it would build the dependency between itself and all releases.

Additional Information

More information can be found in

- *http://connect.crownpeak.com/blogs/developers/dependenciesoverview*
- *http://connect.crownpeak.com/blogs/developers/dependencieswhydomorearticlesgooursometimes*