Content Services Toolkit

Business Automation Workflow Components Guide

9/26/2022

This document describes the view components that are available in the Content Services toolkit released in 4Q 2022. There are a handful of components tailored for two scenarios; general content management and document processing. In most cases there is overlap in functionality and most components can be used in a variety of scenarios.

## [Content List](#_Content_List_Component)

## [Content List Filter](#_2._Content_List)

## [Content Properties](#_Content_Properties)

## [Document Thumbnail](#_4._Document_Thumbnail)

## [Document Reference](#_5._Document_Reference)

## [Document Viewer](#_6._Document_Viewer)

## [File Drop Zone](#_5._File_Drop)

## [Get Content](#_2._Get_Content)

## [Upload Content](#_3._Upload_Content)

## [Batch Content](#_8._Batch_Content)

## [Property Poller](#_9._Property_Poller)

## [Progress Indicator](#_10._Progress_Indicator)

## [Search](#_11._Search)

## [Summary Tile](#_12._Summary_Tile)

## [Add Batch Modal](#_14._Add_Batch)

## [Add Document Modal](#_15._Add_Document)

## [Add](#_12._Summary_Tile) [Folder](#_16._Add_Folder) Modal

## [Delete Object Modal](#_17._Delete_Object)

## [Edit Properties Modal](#_18._Edit_Properties)

## [Upload Version Modal](#_19._Upload_Version)

## [List Annotations Modal](#_20._List_Annotations)

## [List Versions Modal](#_21._List_Versions)

## [Rename Folder Modal](#_22._Rename_Folder)

# 1. Content List

## Description

This is a general-purpose content list component that can be used to display folder contents, search results, batches, batch documents or just a list of specified documents. It includes a breadcrumb for folder navigation, toolbar, context menus and paging controls. The content list, when configured for search results can also display a search property filter control at run time (see Content List Filter for more detail). The Content List is built upon the Service Data Table component that comes from the UI toolkit delivered with BAStudio application development environment. The following is an image of what the content list looks like at runtime:

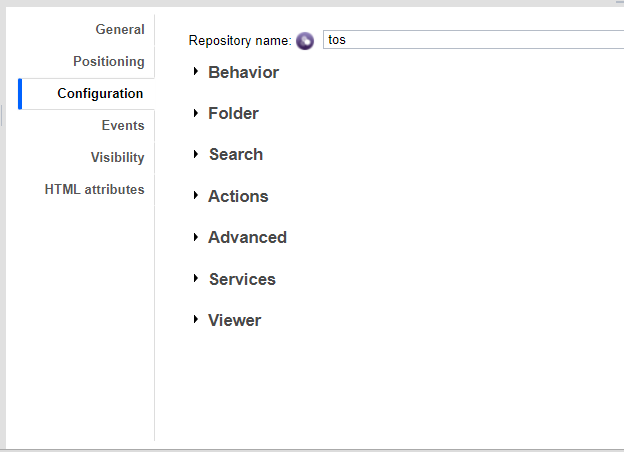


The content list provides a general list of content management actions that can be performed on the documents, folders, or batch objects. They can be invoked either from the Add button on the toolbar or the context menu per row. Both drop-down menus are customizable where custom actions can be added to the menus. See the developers guide for more information about adding custom actions.

The content services toolkit includes many “Actions” services that are wired up internally to the content list to provide the data and behavior it needs to render. These action services make “graphql” requests through the Application Engine server to the Content Engine server in the back end. The graphql requests return response data in JSON format which the toolkit components use to render.

## Configuration

The Properties display of each component has a Configuration tab where all the design time settings are displayed. The Content List has by far the most settings and each section will be discussed below. By default, the Repository setting is required and is the object store name from the Content Engine repository to be used. In addition, the minimum requirement for configuration settings of the content list is the loading mechanism, ie. by folder, search, or document list. See the Folder and Search sections below for the details of those settings. The following image is what a user would see when first opening the Properties action:



There are several categories of settings that will be displayed when opening the sections. The sections are the following:

**Display options** – settings that affect the way the content list view renders.

**Folder** – settings for selecting a folder to populate the list view.

**Search** – settings for specifying query parameters to populate the list view, plus filter configuration.

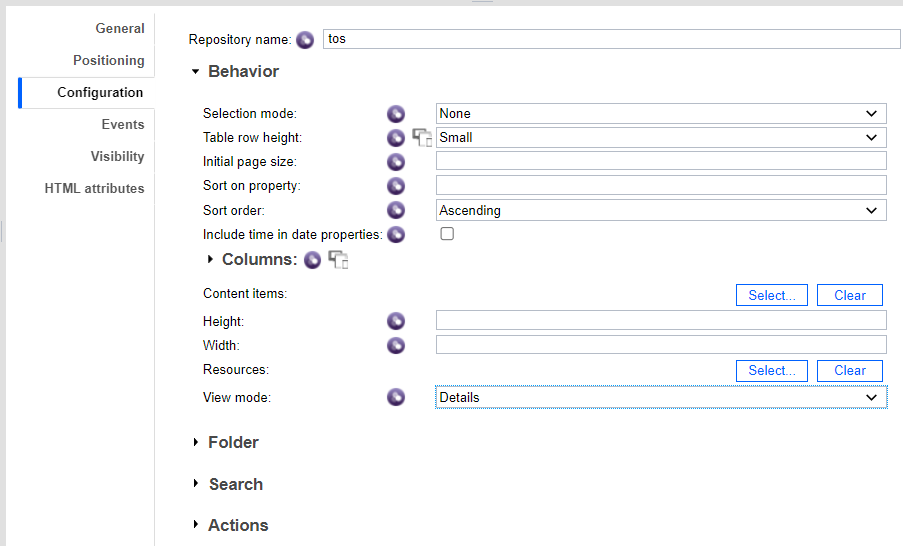
**Actions** – settings for controlling the display of the toolbar and context menu actions.

**Advanced** – settings for miscellaneous and document processing specific options.

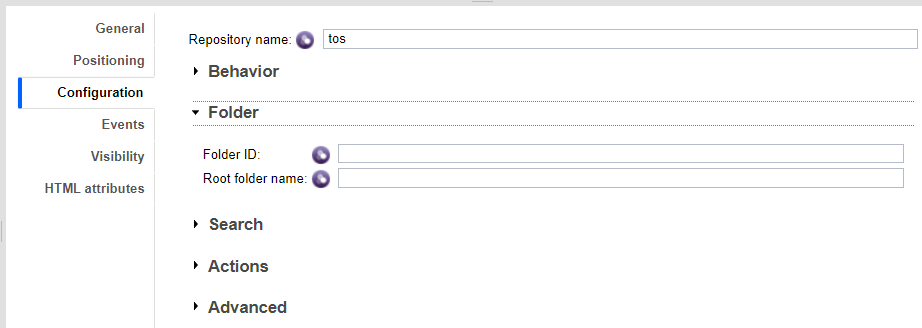
**Services** – settings for graphql service in a Business Automation Workflow environment

**Viewer** – settings for the viewer component when opened from view action on context menu

The following image shows the display options section open where users can set such things as the initial page size (default is 20 if not specified), default column sort property and specify custom columns for the list view (if no columns specified, a default set will be used).

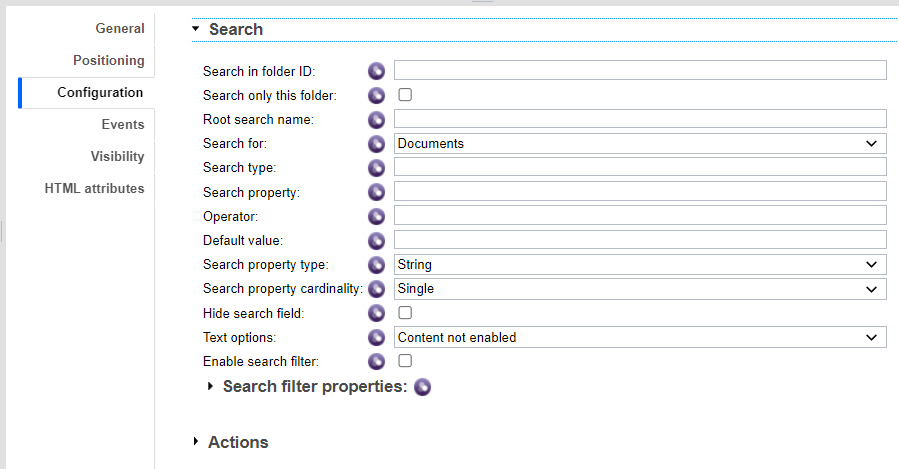


The following image shows the Folder options section open where users can specify a folder to be loaded into the Content List at runtime. The folder can be specified as a folder {guid} or a folder path from the Content Engine. Specifying a folder ID here will have precedence over the Search settings.

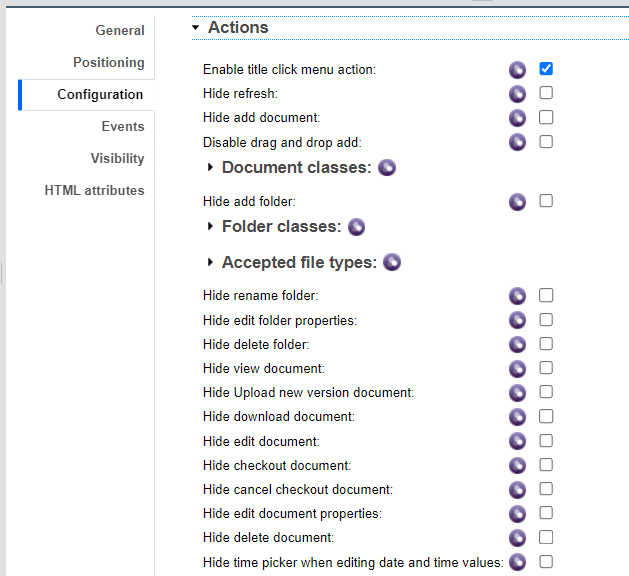


The following image shows the Search options section open where users can set such things as to search for (documents, folders, or abstract objects), search property (specified as the symbolic property name from the Content Engine), operator, and default value to use when populating the list view. Users can also further scope their query down to search in folder and/or for a specific search type (class). In addition, a search filter component can be enabled, and search properties configured, which will add a toolbar button to the content list view which will open the search filter component where the user can specify search property values.

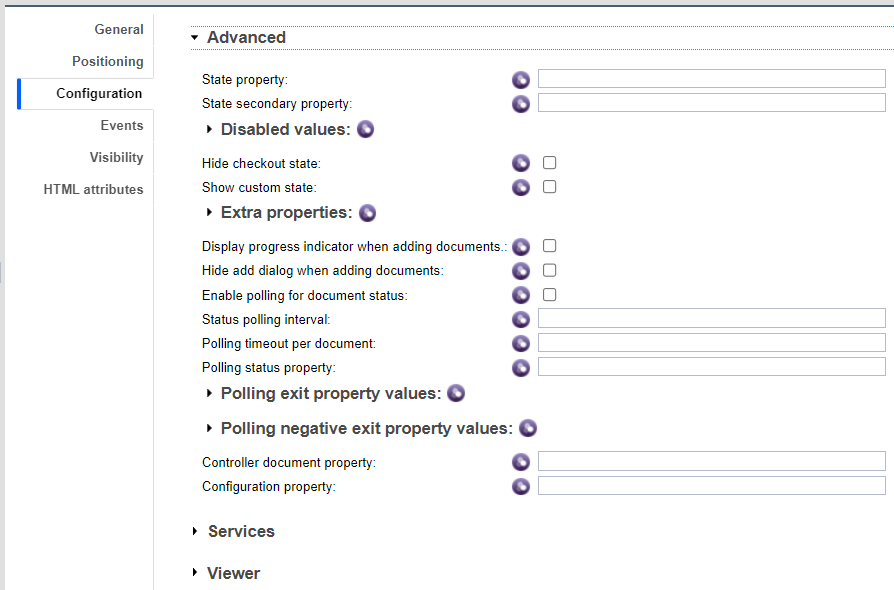
NOTE: Abstract objects are objects of class type “Abstract Persistable” from the repository.



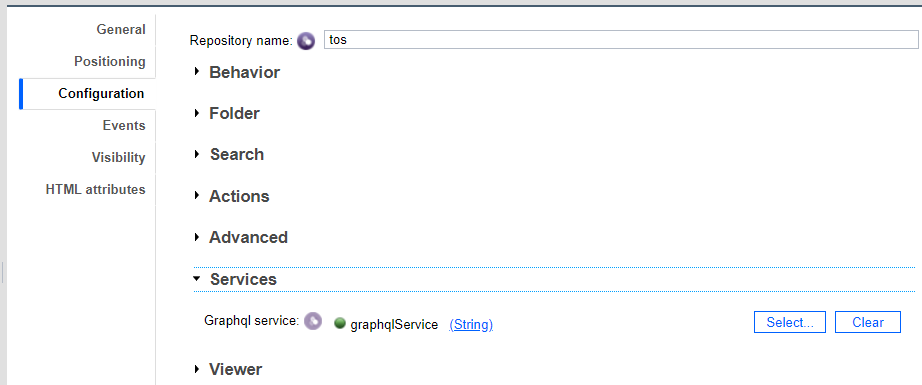
The following image shows the Actions options section open where users can specify whether certain actions show up in the context menu or specify a class or set of classes to be displayed in the add document/folder modal dialogs. In addition, the enable title click menu action will enable the click behavior of the main column field to execute the first action listed in the context menu (by default it is the View action for documents).



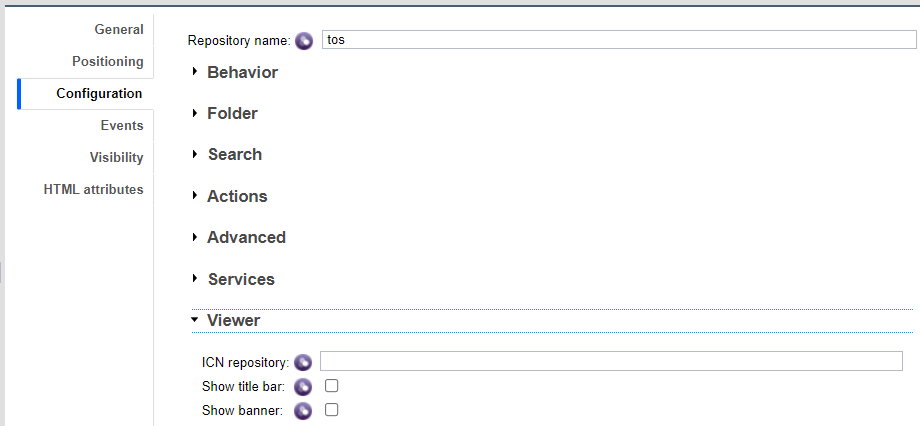
The following image shows the Advanced options section open where users can specify miscellaneous settings such as displaying a custom state column on the left side of the list view, whether to include extra properties in the response when loading the list view (these could be used for custom business logic), plus configuration for a secondary progress indicator when adding documents and setting up a polling mechanism to handle asynchronous state/property changes in the newly added documents in the list view.



The following image shows the Service options section open where the graphql service can be specified when running in the Business Automation Workflow environment (this setting is not needed in the BAStudio environment). In this screen shot, the graphql service is shown dynamically linked to an application Variable. In this way, other components on the page can specify the same service (and the value can be specified in one place).



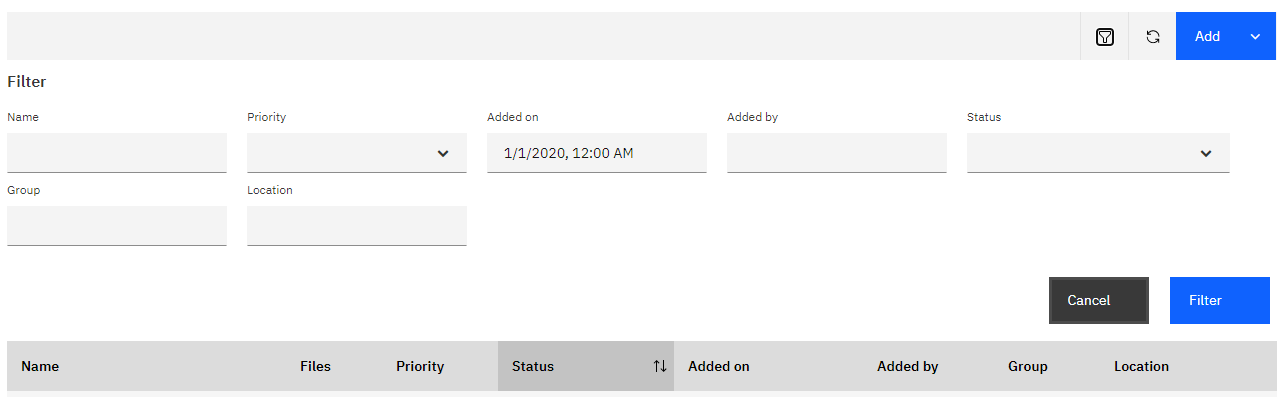
The following image shows the Viewer options section open where the viewer component can be configured. The viewer component is displayed when the View action is run from the context menu. The ICN repository setting is required when the Navigator desktop specified for the application has more than one ICN repository, then the viewer URL will need to specify this repository instead of the global object store setting (repository).



# 2. Content List Filter

## Description

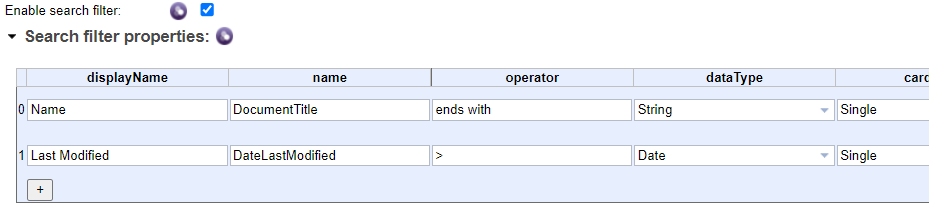
The content list filter component allows the user to specify property values to be used to query the system with to identify the results. This component is not a stand-alone component, it’s part of the content list component and must be configured as part of the search parameter configuration settings of the content list (the filter cannot be used when a folder has been selected to populate the list). The filter can be opened by clicking the funnel icon on the toolbar (and closed the same way). The property controls configured as part of the filter will honor the property data type setting and restrict the user into entering the correct values for that data type or in the case of dates, show the data picker control. The Filter button will execute the query by “AND”ing together the specified property values (“OR”ing values is not supported). Users can cancel the current filter, which will reset the filter values to those that were configured in the content list at design time.



The Filter component configuration settings allow for filter properties to be defined with default values and hidden from the runtime UI. This will force specific query parameters that the end user cannot change (or even see).

## Configuration

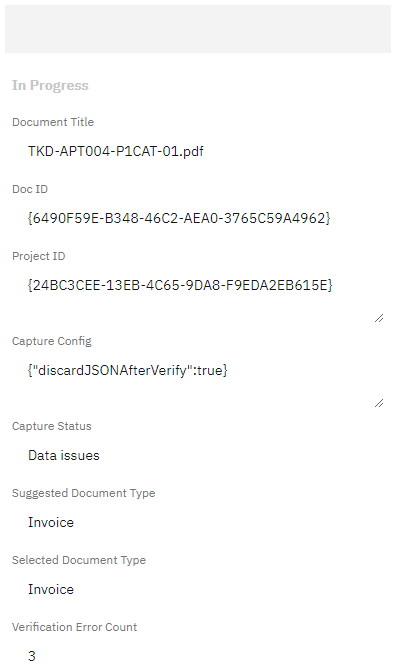
The content list filter configuration settings are part of the content list settings and can be found in the Search section (below the general search parameters). First the filter must be enabled (checkbox). Users can add multiple property filters and optionally specify a default value to use when executing the search. They can also enter multiple default values for a single property by separating them with the semi colon character ‘;’. The config settings provide a “hide” option, where when used, there is a requirement for default value (since users cannot see property at runtime). Hidden filter options will always be added to the query parameters. Finally, all the filter parameters will be ANDed together when performing the query (there is no option to specify ORing the values). The following image shows a couple filter properties specified. The “name” field should be the symbolic property name of the property from the Content Engine.



# 3. Content Properties

## Description

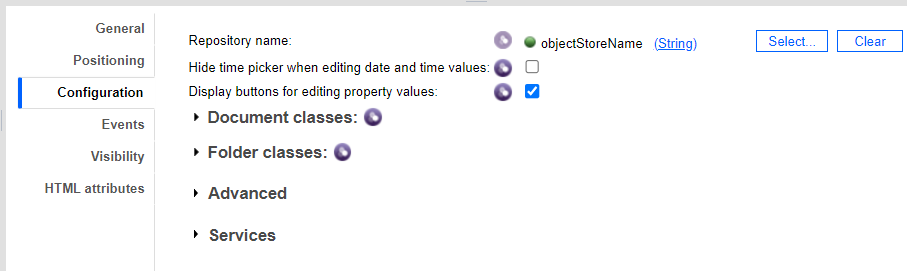
The Content Properties component can be used to display and edit content properties. It will display the custom properties assigned to the class type of the object being investigated. This component utilizes a vertical display for the list of properties with the labels above the values, it’s intended for taller vertical restate areas, and is typically displayed to the right of the content list. Users can configure at design time whether to allow the runtime user to edit the properties, if so, will display appropriate buttons on the toolbar at the top of the component. In edit mode, the property picker controls honor the data type of the properties and present the appropriate picker and validation logic. In none edit mode, the property list will only display properties with values to minimize the display area, in edit mode, all editable properties are displayed.



To setup the loading of the Content Properties component with the selection of an item from the content list, a “binding” must be specified by both the content list and the property list. The binding must specify a CSContentItem object as the source in both components. Then at runtime, when a user selects a row in the list view, the content properties list will pick up the new selection and load itself with the appropriate data. In addition, if the user edits the property values and saves the changes, the content list will pick up the new values and refresh the row (column properties) with the new values.

## Configuration

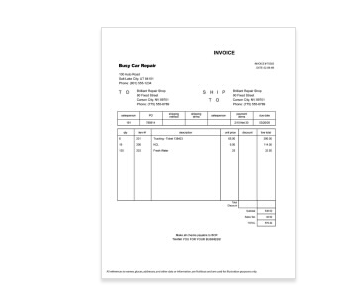
The Content Properties component also has just a few basic settings that can be seen in the following image. The repository name is shown as a dynamic link to an application variable for a single point of source. The flag for displaying the edit buttons on the property list toolbar is probably the most significant option to note here. The Services section contains the graphql service configuration setting that is only required in the Business Automation Workflow environment.



# 4. Document Thumbnail

## Description

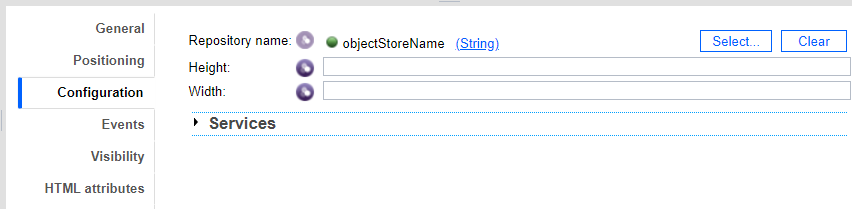
The Document Thumbnail component can be used to display document thumbnail images. The images are generated by the thumbnail “AddOn” in the repository. If not configured or a thumbnail has not been generated yet (it’s an asynchronous process), the mime type icon will be displayed instead. In a typical scenario the thumbnail component would be positioned to right of the list view and above the properties list. This orientation gives the application a standard content management display and general feature set.



The Thumbnail component should also be configured to be bound to the CSContentItem object so that it will react to the list view row selection and change its display based on the item selected.

## Configuration

The document thumbnail has one basic configuration setting for the repository. The height and width can also optionally be set.



# 5. Document Reference

## Description

The Document Reference component can be used to display a specified document in either normal or thumbnail mode (thumbnail mode requires the Thumbnail generation AddOn be install in the repository). The Document Reference can be configured to allow the user to add a document and can also be configured to allow the full set of content management actions off the context menu. This component also includes several events that can be used for customization such as adding custom context menu actions.

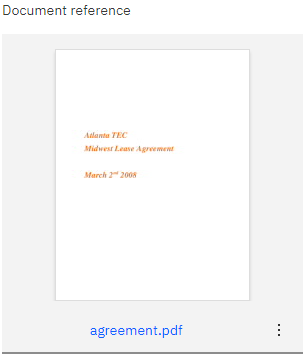
This is the Document reference in default unloaded state, with the ability to add documents turned on in the configuration settings (both the component label and the hint text inside the control are configurable):



This is the Document Reference loaded with a document in normal display mode and the ability to run content management actions turned on in the configuration settings:



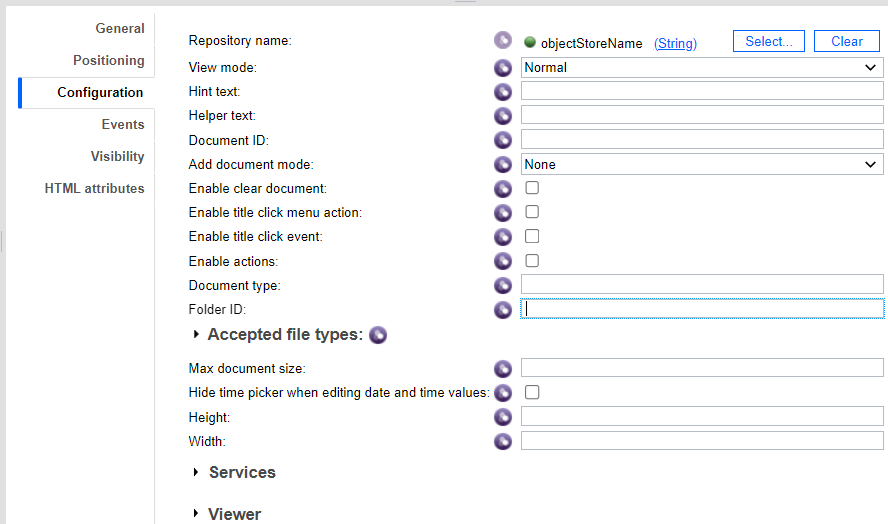
This is the Document reference loaded with a document in thumbnail display mode and the ability to run content management actions turned on in the configuration settings.



## Configuration

The Document Reference component also has a full set of configuration settings. Some of the more notable settings are the ability to turn on the feature to add documents, ether with Properties UI or without (would use the specified Document type if without UI). Also setting the ability to enable actions which turns on the content management context menu actions at runtime (these are the same actions as exposed in the content list component). Plus enabling the title click menu action which would perform the first action listed in the context menu which be default is the view document action. The Document Reference also sends out a on context menu open event which can customize the context menu actions. The first action in the context menu maybe a custom action injected via this event.

The Viewer options section is where the viewer component can be configured. The viewer component is displayed when the standard View action is run from the context menu. The ICN repository setting is required when the Navigator desktop specified for the application has more than one ICN repository, then the viewer URL will need to specify this repository instead of the object store setting (repository).



# 6. Document Viewer

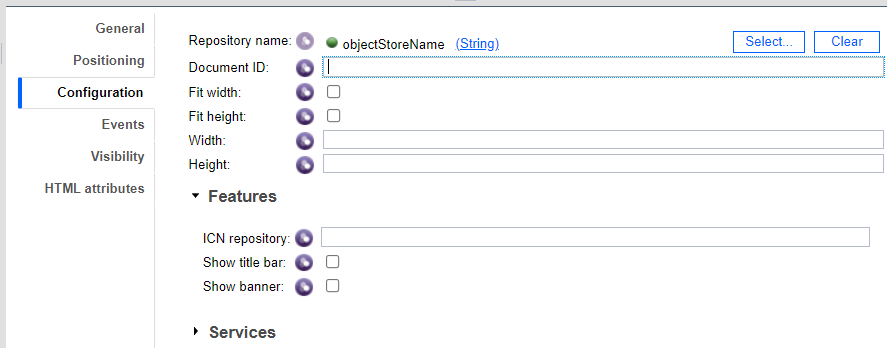
## Description

The Document Viewer component can be used to display documents that are stored in the Content Engine repository. The viewer component is an html iframe element loaded with the Navigator bookmark.jsp URL. Therefore, it requires that the application page hosting the viewer must be run from an application accessed through a Navigator desktop. Since the viewer URL is pointing back into Navigator, the viewer will not run in BAStudio’s Preview mode (since preview is not running inside Navigator). The viewer cannot load documents outside the repository nor documents from the users local file system.

## Configuration

The Document viewer has a minimum set of configuration options, the standard object store setting plus a document Id and height and width settings. The document Id setting can be dynamically linked to an application variable (or instance variable) of a document to be loaded in the viewer at runtime. The Height and Width settings have default values of 480px by 600px, most likely these settings will need to be changes to match the application page design.

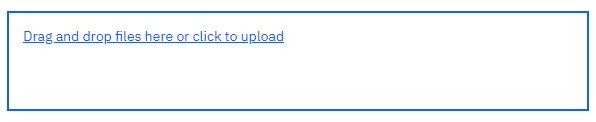
The Features options section is where miscellaneous settings can be configured. The ICN repository setting is required when the Navigator desktop specified for the application has more than one ICN repository, then the viewer URL will need to specify this repository instead of the object store setting (repository). The show title bar and banner bar settings control the display of the Navigator bars that are typically used when running the viewer from Navigator. The title bar settings display the Navigator title bar, and the banner bar shows the viewer banner bar which may include so viewer actions depending on the viewer framework setup in the Navigator desktop.



# 7. File Drop Zone

## Description

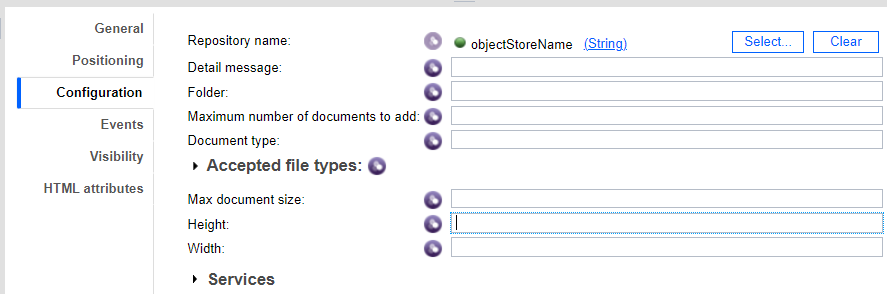
The File Drop Zone component can be used to add documents to the repository. The file drop zone does not display the standard Add Document UI and will perform the add process with whatever setup in the configuration settings. In addition, it will automatically use the file name as the Document Title property value on add. The drop zone supports both adding documents via drag-n-drop or by opening the local file selection control to select from the local file system. This component supports both single select or multi-select adds.



This component can be configured to specify a folder that all the documents will be added too, if not specified, documents will be added to the root folder (unfiled). There is also a configuration setting for specifying a default document class type to use when adding the document. If not specified, the base “Document” class will be used.

## Configuration

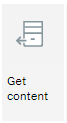
The File Drop Zone component has several configuration settings for controlling the add document process. The configuration allows for a details message to be defined that will be displayed above drop zone area. Users can specify a folder that their documents will be filed in (added to) at runtime. This setting can be specified as a specific folder id (guid) or a folder path. In addition, users can specify a document type (class) to use at runtime. Finally, the drop zone can be configured to support adding only one document or many documents at one time. The add document process will not display the properties ui at runtime.



# 8. Get Content

## Description

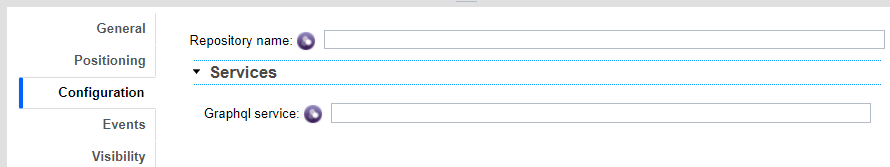
The Get Content component can be used to retrieve document content from the repository. This component has no UI to display and will be hidden in the view at runtime. It can be referenced with JavaScript and will send out an event when the content has been retrieved from the server. This object looks for the content element information in the passed in content item object, however if that information is not found, it will first make a server request to retrieve the document properties which include the content element information necessary to retrieve the content. The retrieved content can be specified to be returned in one of three ways, file, json or response; see the JavaScript developers guide for more details. The following is an image of the component from the designer palette of building blocks:



Designers would place this component on their page or view and reference it with JavaScript from an event handler, for example a on load event or a button click event.

## Configuration

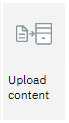
The Get Content component has only one configuration setting for the repository and is accessed via the advanced Configuration tab of the Properties dialog.



# 9. Upload Content

## Description

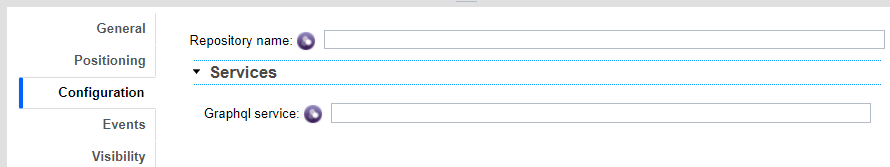
The Upload Content component can be used to upload document content to the repository. This component has no UI to display and will be hidden in the view. It can be referenced with JavaScript and will send out an event when the content has been uploaded to the repository. This component has several functions on it for uploading content. The standard functions take a “file” object as a parameter that comes from the HTML file type=input control. When adding a new document to the system, the api also has parameters for specifying the document class type and the class property values along with the file content. See the JavaScript developers guide for more detail. The following is an image of the component from the designer palette of building blocks:



The Upload Content component fires an “On content added” event once the document content has been uploaded to the repository. This event can be handled in a page or view to pick up the new document object information such as the document name or id from the “results” parameter that is passed along with the event. The results object properties collection contains the set of object properties from the CPE.

## Configuration

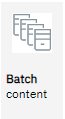
The Upload Content component has only one configuration setting for the repository and is accessed via the advanced Configuration tab of the Properties dialog.



# 10. Batch Content

## Description

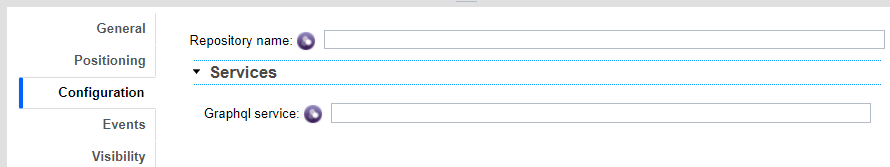
The Batch Content component can be used to perform operations on batch objects. Batch objects are items design and used exclusively for Document Processing applications. This component has no UI to display and will be hidden in the view. This component contains several JavaScript apis used for things such as getting property values, locking, unlocking batches and processing, saving and rolling back change sets (multiple transactional operations on the list of documents in a batch). See the JavaScript developers guide for more detail. The following is an image of the component from the designer palette of building blocks:



This component fires several events once the JavaScript apis that were called have return from performing their operation. These asynchronous events can be used to hook up business logic based on the returned data from the events fired.

## Configuration

The Batch Content component has a primary configuration setting for the repository and is accessed via the advanced Configuration tab of the Properties dialog.



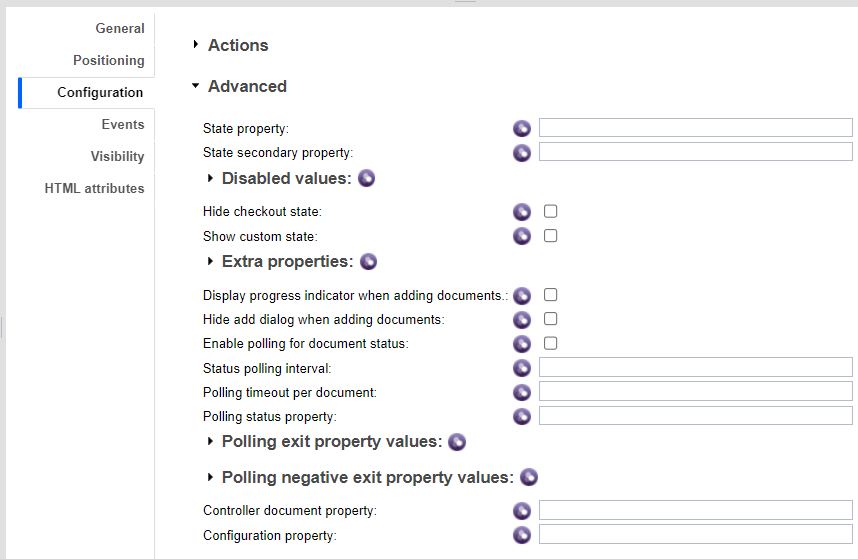
# 11. Property Poller

## Description

The Property Poller component is used to periodically retrieve property information from the repository and update the component view information with the new values, for instance the list view column properties. This component has no UI to display and will be hidden in the view. Also, this component is not a stand-alone component, it’s part of the Content List component and must be configured as part of the advanced configuration settings of the content list.

## Configuration

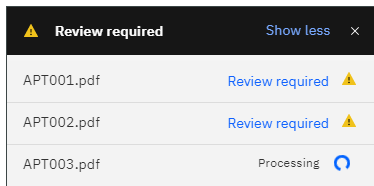
The property poller configuration settings are part of the content list settings and can be found in the Advanced section (below the Extra properties configuration setting). There is a main flag for enabling the property polling and a property selection to be used as the control property for the poller. A polling interval can be specified, usually somewhere in the 5-10 seconds range that determines how often calls are made to the server to retrieve new data. The property poller configuration settings should include specifying when the poller should stop polling. Users can either specify a value or values the polling property has been set to that indicates when to stop or they can specify the negative values (values when the controlling property is not set to).



# 12. Progress Indicator

## Description

The Progress Indicator component displays the current progress of documents being add the repository and then subsequently being processed by the content analyzer. This component is not a stand-alone component, it’s part of the Content List component and must be configured as part of the advanced configuration settings of the content list. When documents are added from the Add menu in the content list, the progress indicator will display itself (if not already open) with the document loading and with an indication of the status.



The progress indicator component can be configured to display a hyper link, that when clicked can execute some custom JavaScript, for instance it could execute a page navigation event. Since the progress indicator is internal to and part of the content list, there is a function in the content list JavaScript API that can be called to add the link information and the click function. See the JavaScript developers guide for more detail.

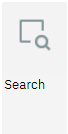
## Configuration

The progress indicator configuration settings are part of the content list settings and can be found in the Advanced section (below the extra properties). There is one main setting for turning on the progress indicator called “Display progress indicator when adding documents” (see the property poller configuration section above for a screen shot of the configuration setting).

# 13. Search

## Description

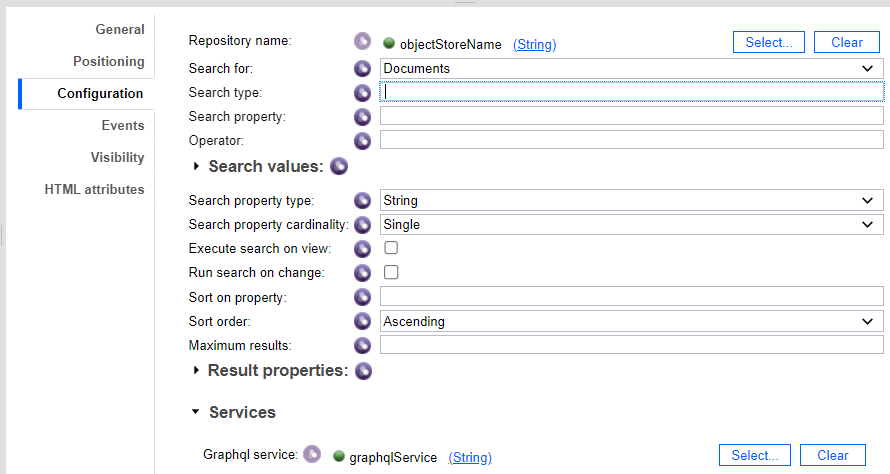
The Search component can be used to execute searches and obtain search results. This component has no UI to display at runtime and will be hidden in the view. The component fires an event when the search results are returned that can be used to send results to other component views. There are an extensive set of search configuration settings that must be specified at design time to create the specified query.



The Search component can be used in conjunction with the Summary Tile component to display the number of items returned from a particular query. The “On results” event that is fired can be handled and reference the summary tile to pass the search results including total count information along.

## Configuration

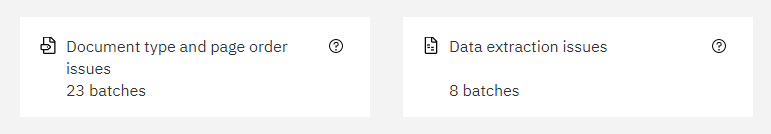
The Search component has several configuration settings and does have a view configurator created for them. Users can specify the kind of object to search for (Documents, Folders or Abstract objects), the search type (if not specified will be the default base class), search property, operator and one or more search Result properties (values) to use as the minimum query parameters.



# 14. Summary Tile

## Description

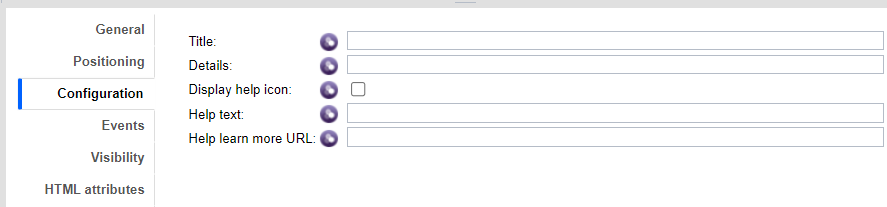
The Summary Tile component can be used to display summary information and will fire an event when the tile is clicked that can be handled and invoke your custom JavaScript, for instance a page navigation event. The summary title and details text string can be set in the configuration settings or set dynamically via a JavaScript API where more information can be retrieved first, for instance the search counts to be included as part of the details string.



The Summary tile component also includes an optional “info” bubble that can be configures to display help text and a link to more documentation, for instance a global knowledge center website that has more extensive online help.

## Configuration

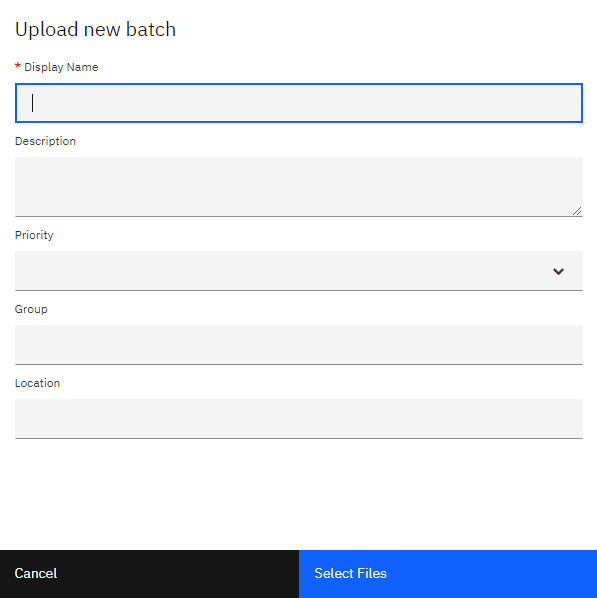
The Summary Tile component has several configuration settings, however, does not have a view configurator created for them. Users will have to open the Properties dialog and navigate to the Configuration tab to see them. Users can specify title and details text and whether to include the help icon. If the help icon is configured, additional help text and a help link can also be included. In addition an optional learn more URL can be specified and will display a learn more link on the tile.



# 15. Add Batch Modal

## Description

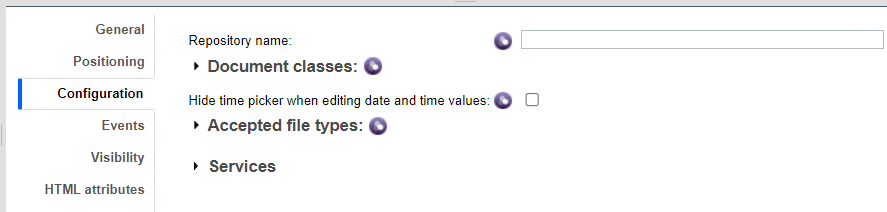
The Add batch modal dialog component can be used to add automated document processing batch objects which includes selecting the list of documents to be included in the batch. This is a very specialized dialog and should not be used outside an ADP application.



The add batch modal dialog uses a fixed document type that has a few custom properties that can be used for querying for your batch later when they are presented in the content list.

## Configuration

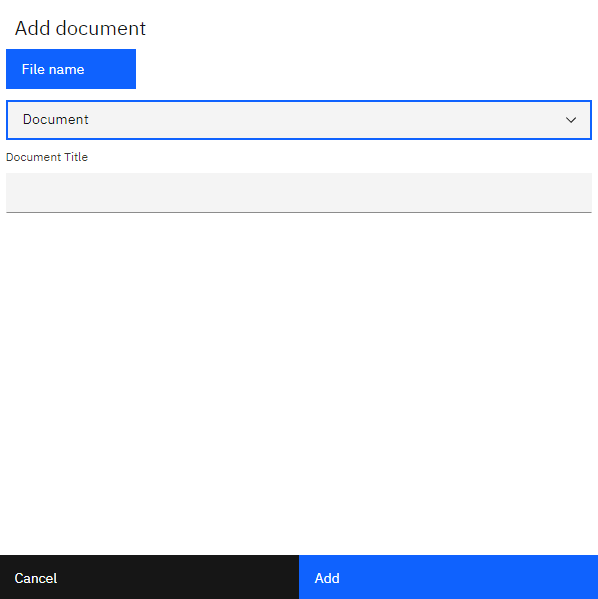
There are only a few configuration settings, most notably in the Accepted files type allowed at the time of adding a batch. Since the system may not support all file types, the dialog can be restricted to specific types that are supported. The accepted file types can be specified as file .dot extensions.



# 16. Add Document Modal

## Description

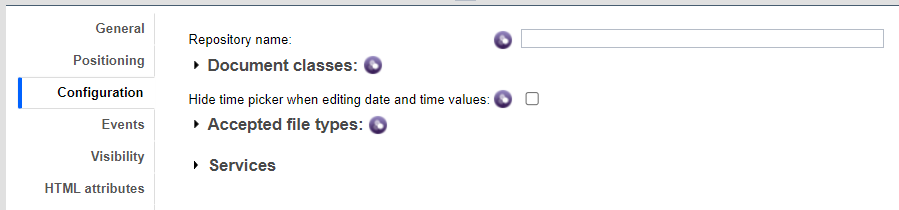
The Add document modal dialog component can be used to add documents into the system which can include setting custom properties on a selected document type.



The dialog presents a local file selection control, document class selector and a properties table for displaying and entering the properties in the selected document type. The properties list will enforce the property meta data for data type and whether the value is required or not.

## Configuration

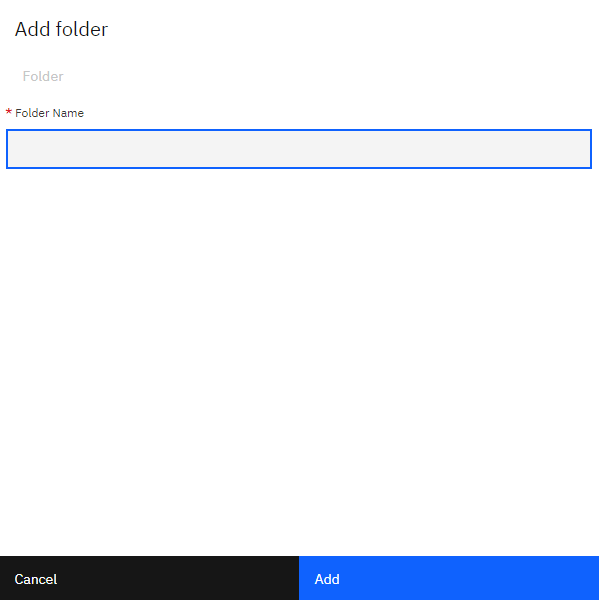
There are only a few configuration settings, most notably in the Document classes list and the Accepted file types list. Setting the Document classes list will restrict the dialog to the list of classes and can simplify the input process for the user. The Accepted file types can restrict which file types can be added by the dialog.



# 17. Add Folder Modal

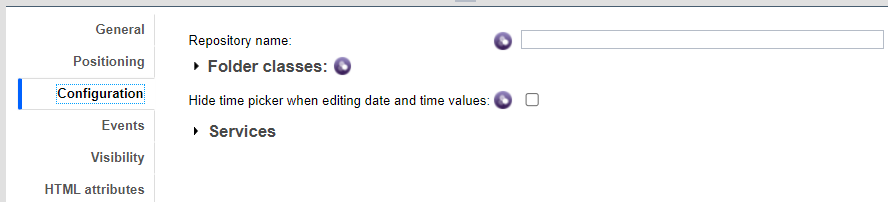
## Description

The Add folder modal dialog component can be used to add folders into the system which can include setting custom properties on a selected folder class. By default, the base Folder class is already selected, and the Folder Name is the only input required.



## Configuration

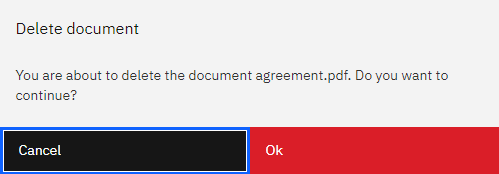
There are only a few configuration settings, most notably in the Folder classes. Setting the Folder classes list will restrict the dialog to the list of classes and can simplify the input process for the user.



# 18. Delete Object Modal

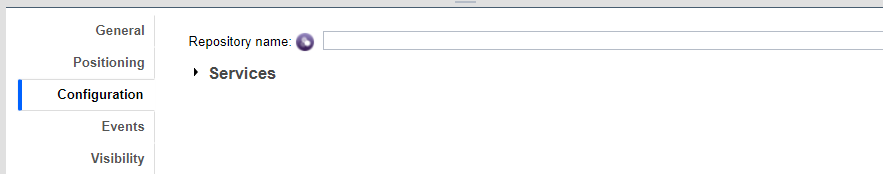
## Description

The Delete object modal dialog component can be used to delete either documents or folders. It presents a second chance dialog to allow the user to continue deleting the object or canceling out.



## Configuration

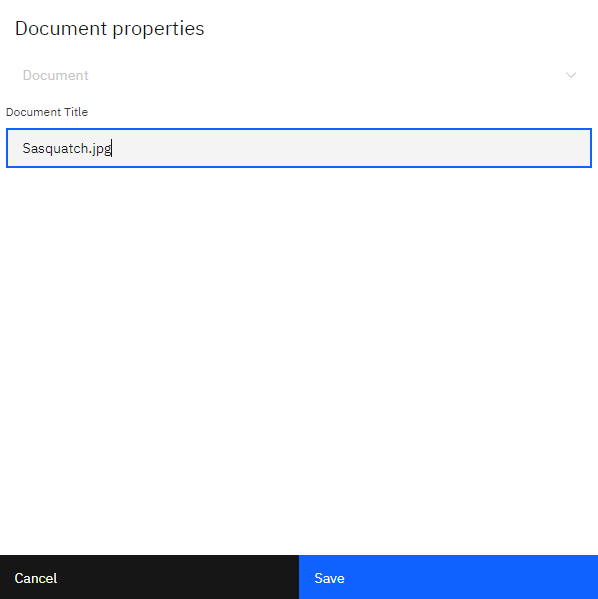
There is only a single configuration setting for the repository name.



# 19. Edit Properties Modal

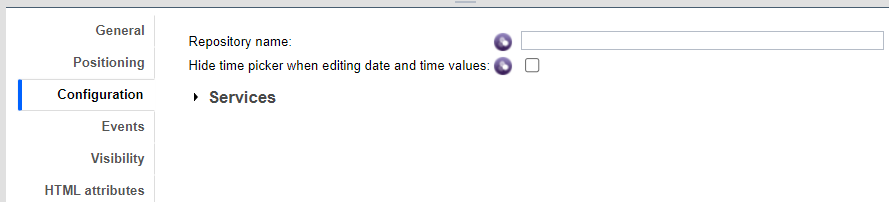
## Description

The Edit properties modal dialog component can be used for editing class properties. It presents the set of custom properties associated with the assigned class and enforces the property data type and whether the value is required or not. The user is not allowed to change the class setting in this dialog.



## Configuration

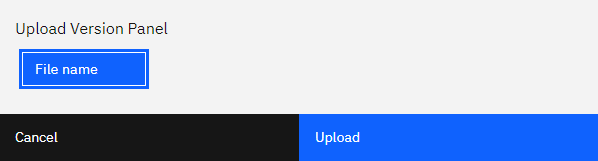
There is a configuration setting for the repository name and a setting for whether to display the time portion when editing date properties.



# 20. Upload Version Modal

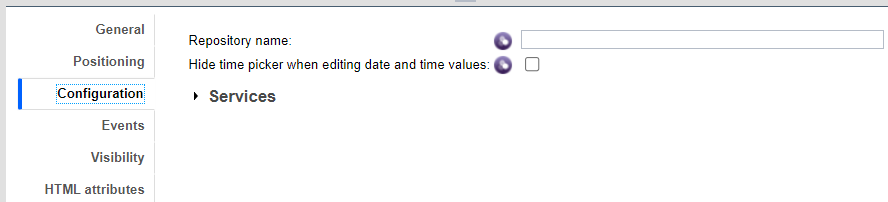
## Description

The Upload version modal dialog allows the user to select a new local file to upload as a new version of a selected document from the repository. No class properties are presented in this dialog.



## Configuration

There is a configuration setting for the repository name and a setting for whether to display the time portion when editing date properties.



# 21. List Annotations Modal

## Description

The List Annotations modal dialog will display the list of annotations associated with a document. The List view also includes a Download action off the context menu so the individual annotations can be retrieved from the system.

The ability to view annotations would typically be implemented as a custom context menu action off a document in the Content List component.

First find the List Annotations Modal component in the palette view and drag it on to the canvas and set the control id to “ListAnnotationsModal”.

Here is a sample of what that code would look like as implemented from the “On context menu open” event of the Content List:

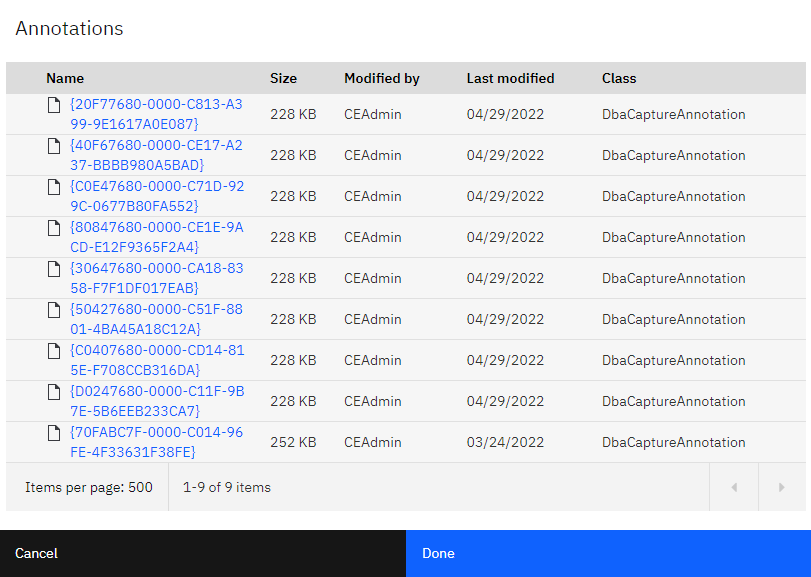
var onListAnnotationsFunc = function() {

${ListAnnotationsModal}.listAnnotations(contentItem);

};

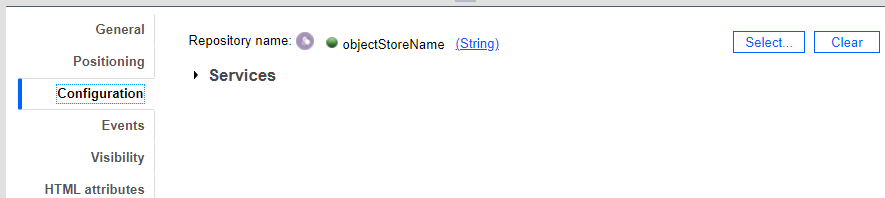
${ContentList}.addContextMenuAction("ListAnnotations", "List Annotations", "document", true, onListAnnotationsFunc);

Where ListAnnotationsModal and ContentList are the control IDs of the content list and the annotations modal dialog respectively.



## Configuration

There is a single configuration setting for the repository name.



# 22. List Versions Modal

## Description

The List Versions modal dialog displays the list of versions (version series) associated with a document. The list view also includes a View and Download action off the context menu so individual versions can be viewed or retrieved from the system.

The ability to view versions would typically be implemented as a custom context menu action off a document in the Content List component.

First find the List Versions Modal component in the palette view and drag it on to the canvas and set the control id to “ListVersionsModal”.

Here is a sample of what that code would look like as implemented from the “On context menu open” event of the Content List:

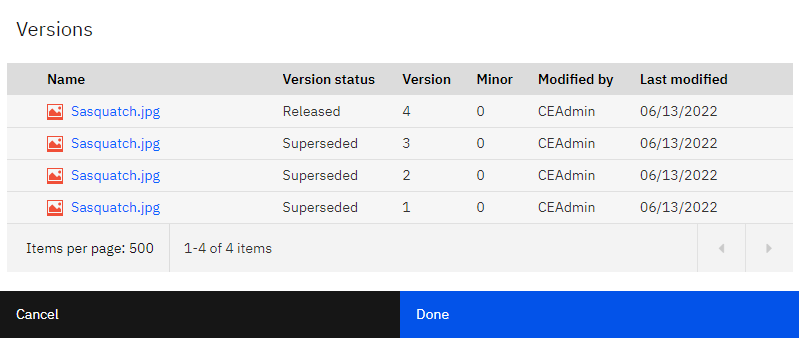
var onListVersionsFunc = function() {

${ListVersionsModal}.listVersions(contentItem);

};

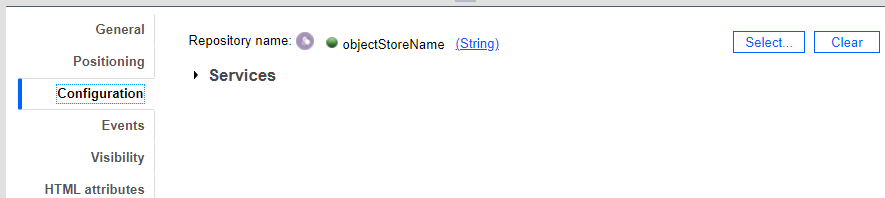
${ContentList}.addContextMenuAction("ListVersions", "List Versions", "document", true, onListVersionsFunc);

Where ListVersionsModal and ContentList are the control IDs of the content list and the versions modal dialog respectively.



## Configuration

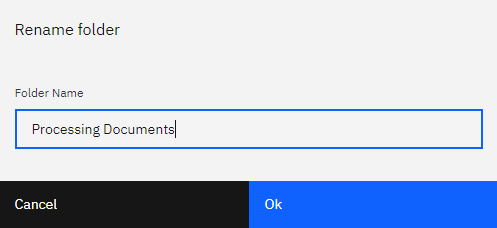
There is a single configuration setting for the repository name.



# 23. Rename Folder Modal

## Description

The Rename folder modal dialog component allows the user to modifying an existing folder name.



## Configuration

There is a single configuration setting for the repository name.

