Quark XML Author adapter for Quark Publishing Platform System Administration Guide

Contents

Quark XML Author for Quark Publishing Platform

The staff of Quark Software Inc. would like to thank you for selecting Quark® XML Author to be your XML authoring solution.

Quark XML Author is a mature XML editing solution in intelligence, pharmaceuticals, European governments, and now DITA.

Quark XML Author supports the following display languages within Microsoft Word: English, French, Japanese and Spanish.

Introduction

The Quark XML Author Adapter for Quark Publishing Platform provides the structured content authoring environment of Quark XML Author integrated with Quark Publishing Platform. This enables SMEs to contribute structured content and to re-use components from the Quark Publishing Platform such as:

- XML Components
- Images
- Microsoft® Excel® charts and tables
- Multimedia

You can create aggregated documents using components available in Quark Publishing Platform, see live previews of those documents with publishing templates and publish them.

About this document

Configuring and integrating **Quark XML Author** involves several processes. This manual provides instructions for:

- Configuring **Quark XML Author** to work in conjunction with the Quark XML Author Adapter for Quark Publishing Platform.
- Configuring the Word **File** menu and Word ribbon to include only those that conform to a valid XML output.
- Configuring access to the Extensibility Interface, a utility linking to an external process for the retrieval of defined values.
- Defining the messages displayed in Microsoft Word as they relate to the document.

- Defining the structure of a document class using the **Quark XML Author Structure** (*.xas).

Intended Audience

This guide has been prepared for persons responsible for configuring and integrating **Quark XML Author for Microsoft Word.** The reader is expected to have knowledge of XML structure, syntax, and standard terminology, as well as of Microsoft Word.

Concepts

Content Types

Every document has a content type. The **content type** concept is a unified mechanism for associating metadata, workflows, relationships, privileges, and rendering and publishing actions with various types of content. The Platform server can automatically detect a variety of content types - including **pictures**, **QuarkXPress projects**, **DITA** topics, and **DITA for Business Documents.**

Assigning content types to assets allows Platform to apply different lifecycles, workflows, and publishing requirements to different types of content.

Content types are hierarchical, with child content types inheriting from their parents for easy and logical configuration. Child content types can be fine-tuned by associating specific metadata and publishing activities with them.

In addition to the standard set of content types, Platform allows you to define new content types and provides an auto-detection mechanism so that they can be automatically recognized.

Attributes

Assets can have attributes, that are containers for metadata that model the intrinsic properties of those assets. The selection of attributes for an asset is determined by its content type. You can use attributes to drive custom workflows and publishing processes, and to reflect a system-managed state. Attributes are created globally and can be applied to one or more content types.

Relationships

A **relationship** links two assets with one another, with one asset being the parent and the other being the child, and has some associated metadata. There are different types of relationships, with different sets of associated metadata. Relationships can be specific to a particular version of a child asset, or can apply to all versions. Relationships enable component-content management use cases. There are predefined relationships between **QuarkXPress** projects and article components, and between **QuarkCopyDesk** article components and pictures. There is a predefined relationship for XML component references. You can also create your own relationships.

Component management and referencing features are available both for XML content and for **QuarkXPress** or **QuarkCopyDesk** components. In this context, an asset can be a single topic, a concept, an image, or a media file. Aggregated documents (including **DITA** maps and **QuarkXPress** layouts) are also modeled as assets.

The content type of an asset determines its role. **Platform** uses asset relationships to model content-component references. For cases of content reuse, **Platform** creates multiple relationships, which define attributes such as a component's location, its update status, and so forth. You can selectively add content when you check in the content to **Platform**, for easy reuse. You can pin content to a particular version, or be automatically updated when the master version of the content changes.

Templates

The adapter allows users to create new documents based on existing server templates. When the user chooses this option, the asset picker dialog is shown with the server templates configured that the user has access to.

Out-of-the-box, the configuration section **ContentTypeMappings > ContentTypeMapping**, where **@name** is **XML Document Template** determines which content type is configured as the content type for server templates.

For example:

```
<ContentTypeMapping name="XML Document Template" type="XML"
class="Business Document"
filter="includeChildContentTypes=true"/>
```

Specify `class` to specify the content type to be used for server templates

Specify `includeChildContentTypes` to set if the child content types of the specified content type needs to be included or not.

The configuration supports combining multiple `ContentTypeMapping` settings for varying server template types.

Global ID

Platform has a predefined server attribute called **Global ID** which is configured to store the document's unique identifier. As an example, for a Business Document, the document's ID attribute **ID7b7c726d-00a2-4a6a-bbf3-17353c695b9a**.

The adapter associates a check out of a document with a specific asset in the Platform Server by searching the document's unique identifier on the server. If there is a conflict, the adapter will bring up a conflict dialog to allow the user to associate the document with a server asset.

Also see the configuration **sections DocSettings > AttributeMapping > ComponentTypes > ComponentType > Attributes > Attribute**, for mapping of the document's unique identifier against the server attribute Global ID. For example:

```
<Attribute name="Global ID" xpath="./@id"
indexingOption="ALL_VERSIONS"/>
```

The SearchAttribute configuration is used to configure the Platform search for association, e.g.

```
<SearchAttribute name="Global ID" xpath="./@id" filter="" />
```

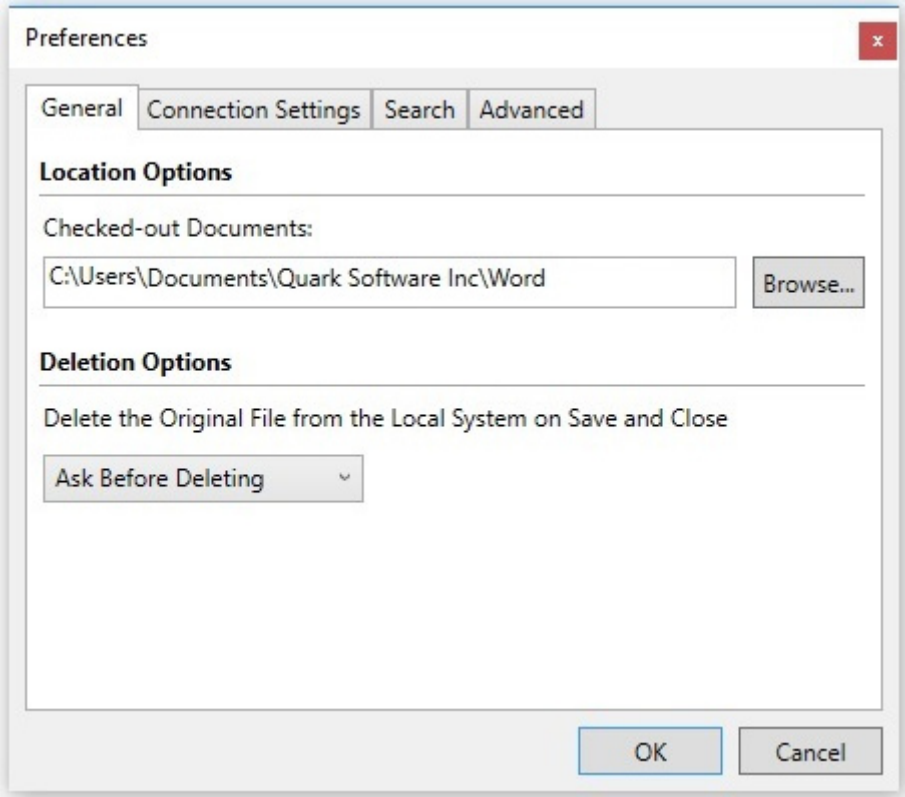
User and application preferences

By default, all preferences are picked up from **Quark.CMSAdapters.config**. It acts as default values for all user and application preferences, which are shown in the preferences dialog.

An example would be the checkout location for documents, it is picked from the following **appSettings** configuration:

```
<!— Defines check out location.—>

<add key="CheckOutLocation" value="Quark Software Inc.\Word"/>
```

This is shown under **Preferences > General > Location Options > Checked-out Documents**:



When the user opens the **Preferences** dialog, this value is shown and honored. Once the user closes the preferences dialog the adapter, captures them inside an encrypted file on disk, such as **Quark.CMSAdapters.QPP.Preferences.dat**.

The user can change any preference to his desired preference value, Quark.CMSAdapters.config is meant to be read-only, and when the user edits the preferences and they get stored in the user's %APPDATA%\Quark\1.0\Word folder.

After this, if the system administrator updates the installer or the configuration files, the system administrator or user needs to:

1. Delete the files under **%APPDATA%\Quark\<VERSION>\Word**, e.g. **%APPDATA%\Quark\13.2.0.0\Word** and then open Microsoft Word again.
2. If the system administrator pushes or remotely changes Quark.CMSAdapters.config and the user uses the **Advanced** preferences feature under **Preferences > Advanced > Reset to Default**

This resets all customized preferences to default values, which are picked from **Quark.CMSAdapters.config**

Adapter Configuration

The Adapter Configuration can be classified into three different categories:

1. CMSAdapter Configuration

In general, the adapter configuration is organized as application level settings and document level settings in **Quark.CMSAdapters.config**. This configuration file is available in the application directory, usually at **C:\Program Files (x86)\Quark\Quark XML Author**

1. Application level Configuration

Application configurations are managed via appconfig.xml. This configuration file is available in the application directory at **C:\Program Files (x86)\Quark\Quark XML Author\en**

1. Document level Configuration

Document level settings are managed in config.xml. This configuration file is available in the BUSDOCS folder within the application directory, usually at **C:\Program Files (x86)\Quark\Quark XML Author\en\BUSDOCS**

CMSAdapter Configuration

AppSettings

The **<appSettings>** element is used to specify various application-level settings.

Check Out Location

Specifies the location for a document on the local system when it is checked out from the repository. Special folder based configuration is also supported. By default, the check-out location is set to **Quark Software Inc.**, which defaults to **%USERPROFILE%\Documents\Quark Software Inc**.

App Data Path

Defines the app data location where the preferences will be stored. The default value is: **%APPDATA%\Quark\1.0\Word**

Log File Path

Specifies the location that the system should use for storing log files. By default, the log file path is **%APPDATA%\Quark\XML Author\Logs\CMS Adapter Log.txt**

AppConfig File Path

Specifies the location that the system should pick up the app.config, this is required in case the administrator would like to configure external run time parameters. You can also inject the custom app.config as the application config. Runtime, network, web settings can be configured and injected using this file.

HttpWebRequestTimeOut

Specifies the HttpWebRequest timeout value in seconds. By default, this value is set to empty, which means using the default configuration.

File Deletion Option

Specifies the delete action for a local document when it is being saved to the server. Applies to the document and references.

Specify **YES** to delete the document.

Specify **NO** to not delete.

Specify **ASK** to prompt the user whether or not the document should be deleted.

By default, this value is set to **ASK**.

Multi Value Separator

Defines the separator that should be used between multiple values in a single string. By default, this value is set to ';'.

UniqueIdentifier

Defines qualified name of the attribute to uniquely identify the element. By default, this value is set to 'id'.

IXMLProcessor

Defines assembly name implementing **IXMLProcessor** interface. A specialization of the **IXMLProcessor** interface can implement the ApplyTransform method to apply a transform on all XML component references.

Common use cases could be truncating long components, such as chapters in books, or hiding disclaimer notices in the canvas.

As part of the build shuttle, an XSLTProcessor specialization is also built, which applies an XSLT transform on all resolved references.

IContentManager

Defines the assembly name that is implementing the IContentManager interface.

IComponentParser

Defines the assembly name that is implementing the IComopnentParser interface.

IConfigurationManager

Defines assembly name implementing the IConfigurationManager interface.

ITaskPaneView

Defines comma separated assembly names implementing the **ITaskPaneView** interface. This is for multiple task pane views in the Smart Document Pane. The following Task Pane view assemblies are configured by default:

- Quark.CMSAdapters.Core.XA.UI.TaskPane.ReferenceView
- Quark.CMSAdapters.Core.XA
- Quark.CMSAdapters.Core.XA.UI.TaskPane.UsagePaneView
- Quark.CMSAdapters.Core.XA.Publishing.QPP

IPreferenceView

Defines the separator that should be used between multiple values in a single string. This is for the multiple preference view panes in the preferences dialog.

Show Advanced Preferences

Defines whether the advanced preferences tab has to be shown or not.

Specify 0 to not display the advanced tab

Specify 1 to display the advanced tab

By default, this value is set to '1'.

AutoCloseRuleEvaluator

Defines whether Rule Evaluator dialog should auto close after all the rules validated successfully. Values supported are 0 (NO) and 1 (YES). The default value is 1 (YES).

EnableSearchWhileOpening

Defines whether the document should be searched on the server based on searchAttribute while opening a local document. Matching server assets are searched based on the SearchAttribute in the ComponentType/SearchAttribute configuration. Values supported are 0 (NO) and 1 (YES). The default value is 1 (YES).

ShowReferencesToOpenedDocument

Defines whether "References" pane should display references to the currently opened document. Values supported are 0 (NO) and 1 (YES). The default value is 0 (NO).

DefaultNameSpace

Defines the default namespace of an XML document in the following format:

```
<add key="DefaultNamespace" value="uri=[Namespace URI]"/>
```

You can additionally specify an element name prefix for the xpath to select elements of this document in the following format:

```
    <add key="DefaultNamespace" value="uri=[namespace
uri],prefix=[namespace prefix]"/>
```

For example, if a prefix has not been specified, then the xpath to select an element appears as follows:

```
      /*[local-name()='topic' and namespace-
uri()='http://quark.com']
```

If 'quark' bas been specified as a prefix, then the xpath becomes

```
/quark:topic —>
```

Quick Search

For specified search criteria, specifies what should be searched by the Search feature. Used in the 'Open' and 'Save' dialogs.

Specify **NAME** to search the file name.

Specify **CONTENT** to search the content of the document.

Specify **NAME_AND_CONTENT** to search both the file name and the content of the document.

By default, this value is set to **NAME_AND_CONTENT**.

Revision Comments Display Option

Defines whether the revision comments need to be displayed or not.

Specify **ALWAYS** to always show the revision comments when opening document from server.

Specify **NEVER** to never show the revision comments.

Specify **NEW_ASSIGNMENTS_ONLY** to show the revision comments when opening newly assigned documents from server.

By default, this value is set to '**ALWAYS**'.

Show Attribute Form

Defines whether the save dialog displays attribute form in the more options.

Specify 0 to not display the attributes form

Specify 1 to display the attributes form

By default, this value is set to '1'.

Refresh Collections

Specifies whether or not the Save and Open dialogs should retrieve the latest collections information from the repository when it displays the list of collections in the dialog.

Specify 1 to retrieve the latest collections information each time, the dialog is invoked.

Specify 0 to not retrieve the latest and use the cached information only. This cached information is updated each time a new session is created, such as closing all documents and opening a new document.

By default, this value is set to '1'. You can configure it to 0, if you feel that the collections hierarchy does not change very frequently, or an application launch is acceptable to the user in such a case.

Honor Saved Display

Specifies whether or not the Open and Save dialogs should honor the search result display as saved in the server for saved searches and collections.

Specify 1 to honor the server display which includes display view whether list or snippet and also the columns to be shown in the list view.

Specify 0 to not honor the server display and use the display specified in the **DisplaySettings > Columns** section.

By default, this value is set to '1'.

ImageHorizontalResolution

Defines horizontal resolution of images used in Office component elements like slides or Visio.

ImageVerticalResolution

Defines vertical resolution of images used in Office component elements like slides or Visio.

ReviewDocumentRelationName

Defines the relation to be used for the Word Review process.

CleanReferenceStrategy

Specifies the reference cleanup strategy.

Specify **Name** to continue to set the reference attribute as the asset name. This is the default value

Specify **AbsolutePath** to set the reference attribute as absolute path of the image.

Specify **RelativePath** to set the reference attribute as a relative path from the document.

By default, this value is set to **'Name'**.

Hierarchy

The adapter configuration file is installed on the client and contains the following hierarchy:

```
<configuration>
  <configSections>
    <sectionGroup name="DocSettings">...</sectionGroup>
    <sectionGroup name="Platform">...</sectionGroup>
  </configSections>
  <appSettings>...</appSettings>
  <DocSettings>
    <AttributeMapping>
      ...
      ...
      <ComponentTypes>...</ComponentTypes>
    </AttributeMapping>
    <ReferenceManagement>
      ...
      <ReferenceModels>...</ReferenceModels>
      ...
      <RelationTypes>...</RelationTypes>
      ...
      <LocalReferences>...</LocalReferences>
      ...
      <CrossReference captionBuilderXslt="ComponentCaption.xslt" openReadOnly="true" />
    </ReferenceManagement>
    <ContentManagement>
      ...
      ...
      <ContentTypeMappings>...</ContentTypeMappings>
      ...
      ...
      <ContentValidationRules>...</ContentValidationRules>
    </ContentManagement>
    <DisplaySettings displayMode="SNIPPET">
      ...
      <Columns>...</Columns>
    </DisplaySettings>
    <LivePreviewSettings>
      ...
      <ChannelMappings>...</ChannelMappings>
    </LivePreviewSettings>
  </DocSettings>
  <Platform>
    ...
    <ConnectionSettings serverName="" port="61400" useHttps="false" />
  </Platform>
</configuration>
```

DocSettings

DocSettings - Attribute Mapping

For each document type and product line (componentRoot=”topic” productLine=”busdoc), specify the document attributes that should be mapped to CMS attributes. Document types include media files such as images. For each attribute, you can specify the value as either a static value or you can specify the XPath to the value within the document. You can specify when that attribute should be indexed. Specify **INITIAL_VERSION** to trigger indexing at initial check in. Specify **ALL_VERSIONS** to trigger indexing each time the document is checked in.

To copy an attribute value from the template, set **inheritValueFromTemplate** to **true**. This overrides the local attribute value.

E.g. `<Attribute name=”Due date” xpath=”“ inheritValueFromTemplate=”true” indexingOption=”ALL_VERSIONS”/>`.

The ‘Due Date’ attribute will be picked up from the template attribute values on indexing and precedence will be given to template value inheritance.

To index an attribute bi-directionally, set **bidirectionalIndexing** to true. This configuration is honored only if valid XPath is specified, and the value denoted by XPath is updated while opening the server document.

To ignore attribute values that are blank during indexing, set **ignoreBlankValue** to **true**.

Specify the following value patterns for various attribute types. For **DATE** attribute types, specify the Short date pattern. For **DATETIME** attributes, specify the combination of the short date and long time pattern, separated by a space. For **TIME** attributes, specify the long time pattern. For **BOOLEAN** attributes specify either "true" or "false".

For **DATE**, **TIME** and **DATETIME** attributes, specify enhanced configuration to optionally take a format, such as 'MMddyyyy'. For advanced configurations, **isInvariantCultureFormat** also supported, set **TRUE** or **FALSE**

Additionally, in the case of bidirectional indexing, also specify the **ChildNodeInsertOrder** element of the appropriate **ComponentType** to represent a list of predecessor-successor pair of elements. If the elements specified by the **XPath** are not present in the document then these elements are created and element insertion is done based on **ChildNodeInsertOrder**.

Use **SearchAttribute** to specify which attribute should be used for searching assets in the repository. You also specify the **XPath** of the attribute and any desired filter.

Use transform to specify the transformation to be applied on the resultant nodes of the xpath attribute.

The Platform adapter supports the following keywords as attribute values:

- `@me:` This attribute denotes the current logged on user
- `@TD:` This attribute denotes the current date and time

Attribute Names- The attribute names should be the internal attribute names in Platform, irrespective of the current language or locale. The following **REST URL** can be used to get the internal attribute names:

```
http://[server
name]:[port]/rest/service/admin/attributes?loginname=[user
name]&loginpassword=[password]
```

From the REST response, use the **attributeInfoList/attributeInfo/name** in the configuration. Attribute names are case sensitive.

➡ The attribute names should be the internal attribute names in the Platform Server, irrespective of the current language or locale.

The following REST URL can be used to get the internal attribute names:

```
http://[server
name]:[port]/rest/service/admin/attributes?loginname=[user
name]&loginpassword=[password]
```

From the REST response, use the **attributeInfoList/attributeInfo/name** in the configuration. The attribute names are case sensitive.

DocSettings - Reference Management

The referencing models for the various content component types are defined in `<ContentManagement>`.

Valid content types include: **XML**, **AUDIO**, **VIDEO**, **PICTURE** and **OTHER**.

Specify the contentType of the reference as either **ATTRIBUTE** based or "reference" **ELEMENT** based. For **ATTRIBUTE**, **referenceAttribute** specifies the qualified name of the attribute to refer for the external component. For **ELEMENT**, **typeName** specifies the name of the container element. Specify the namespace in case the reference attribute also has a namespace prefix.

Additionally, specify the reference element attributes. The attribute value can also be mapped to server attribute value by specifying server attribute name with prefix @ in the value field.

DocSettings - Relation Types

This section defines relation types between various component types. For each **RelationType** specify the name as the name of the relation between parent and child component. Specify the **xpath** signifying the **XPath** to locate related component.

Specify additional attributes to configure relation attributes which need to be set on the server, such as the image metadata which is instance specific.

```
<RelationType name="DitaImageref" xpath="//*[local-name() =
'image']/@href">
  <Attributes>
    <Attribute name="width" value="200"/>
    <Attribute name="height" value="100"/>
     <Attribute name="resolution" xpath="//*[local-name() =
'image']/@widthdpi"/>
  </Attributes>
</RelationType>
```

Local Reference

Use <LocalReferences> to specify how references to non-repository items are processed. When a document is checked in non-repository items are added to the repository. To specify where non-repository items are added, set saveLocationChoice to one of the following: **PARENT_LOCATION** (same location as parent document), **MAPPED_LOCATION** (location specified in the attribute mappings), **RELATIVE_LOCATION** (location relative to parent document).

- In **disableXPath**, specify the XPath of the elements which you would like to be ignored from processing. A typical use case could be an attribute which denoted that this local reference would only be checked in by the user or burst on demand, and not automatically when the parent document is checked in.

- In **saveAsMinorVersion**, specify whether or not the non-repository item is saved as a minor version. Set to 1 for true or 0 for false.
- In **fileDeletionOption**, specify the delete action for a non-repository item when it is being saved to the repository. It applies to the document and references. Specify YES to delete the document. Specify **NO** to not delete.
- In **relinkOption**, specify the relinking options for local references on check-in. Specify **LINK** to link the local reference with the matching server asset. This is the default relinking option. Specify **REPLACE** to link local reference to the matching server asset and also replace the content of the matching asset by creating new version. The **saveAsMinorVersion** setting defines whether new version would be a minor or major version. Specify **NONE** if you do not want to link the local reference with the existing server asset. The matching server assets are searched based on the **SearchAttribute** in the **ComponentType >SearchAttribute** configuration.
- In **searchLocationChoice**, specify the save location search values when searching for server assets. The following values can be specified:
  - **GLOBAL** - Set this value if you wish to search for matching server assets across all save locations. This is the default value of **searchLocationChoice**.
  - **TARGET** - Set this value if you wish to search for matching server asset in the target save location.

Cross-reference Settings

captionBuilderXslt

Specifies the XSLT filename used to build the cross-reference caption.

openReadOnly

Specifies whether or not the cross-reference document is opened read-only. Set to true to open read-only, otherwise set to false.

DocSettings - ContentManagement

The `<ContentManagement>` tag is used to specify the mapping of document content types to their asset class equivalent in the Platform server for enhanced filtering of content. It is applicable for all flows where the asset picker is shown, including:

- Open documents from server
- Reference documents from server
- Create documents based on serve templates

Use `<ContentTypeMapping>` to specify the Content Type mappings. Use name to specify the name of content type mapping. This name should be unique across the content type mappings defined.

Use type to denote the content type, you can specify the following values - **XML, PICTURE, AUDIO, VIDEO and OTHER**

Use class to specify the Content Type server specific name to identify assets of specified type, such as Business Documents, DITA Map, DITA Topic, Research Report etc. Use filter to additionally specify a filter on the content type, such as filter by workflow or status, routed to or any other server evaluated criteria.

The adapter also supports Platform attribute value conditions. The following attribute value conditions are supported, by specify the filter in the following format: `filter="<Attribute Name><operator><Attribute Value>", e.g. filter="Is checked out=true"`, which denotes assets matching criteria "**Is checked out**".

You can specify multiple filters using ";" as a separator:

```
filter="<Attribute Name><operator><Attribute Value> ;< Attribute Name><operator><Attribute Value>", e.g. filter="Is checked out=true; Created=@TD; Routed to=@me"
```

This denotes that assets matching criteria "Is checked out AND Created is today AND Routed to logged on User"

You can specify multiple values for an attribute using "," as a separator: `filter=<Attribute Name><operator><Attribute Value1, Attribute Value2>`, e.g. `filter="Routed to=User1, User2, User3; Created=@TD`, which denotes assets matching criteria **Routed to User1 OR User2 OR User3 AND Created is today.**

The adapter supports filters based on predefined criteria and Platform attribute value conditions.

Use **includeChildContentTypes** whether or not to include assets of child content types while browsing collections or showing results.

Use **@TD** to specify date attribute filters based on 'Today', expressions such as **@TD+1, @TD-7** are also supported, e.g. **filter="Created=@TD; Routed to=@me**

Use **@Me** to specify current user based criteria, e.g. **filter="Routed to=@me"**

Use **@Now** to specify time attribute filters based on '**Now**', expressions such as **@Now-1h**, **@Now-60m** or **@Now+3600s** are also supported.

➡ The attribute names should be the internal attribute names in the Platform Server, irrespective of the current language or locale.

The following REST URL can be used to get the internal attribute names:

```
http://[server name]:[port]/rest/service/admin/attributes?loginname=[user name]&loginpassword=[password]
```

From the REST response, use the **attributeInfoList/attributeInfo/name** in the configuration. The attribute names are case sensitive.

DocSettings - DisplaySettings

This section defines default display mode and default set of Columns for search display. Specify the **displayMode** as the default display mode. Use **LIST** to specify a list based column view, with one column shown for each attribute configured. Use **SNIPPET** to specify a snippet based column view, with the asset thumbnail and few columns configured for display. The columns configured in the snippet view are not configurable. These settings are applicable for all flows where the asset picker is shown.

Use <Column> to specify your custom display column configuration, applicable to all list views including search results. This configuration would be picked up if th**e AppSettings > HonorSavedDisplay** is set to **NO**.

Use **id** to specify a unique identifier for column. The value of the id could be adapter supported default columns or the Platform Server based attribute id. For the server based attributes, the adapter displays the localized display names.

The adapter supports the following default column ids:

- **COLUMN_NAME**, to denote the **Name** column
- **COLUMN_TYPE**, to denote the **Content Type** column
- **COLUMN_VERSION**, to denote the **Version** column, comprised of the Major and Minor version, such as 1.0 or 2.1
- **COLUMN_LAST_MODIFIED**, to denote the **Last Modified Date** column
- **COLUMN_LAST_MODIFIED_BY**, to denote the **Last Modified By** column
- **COLUMN_CHECKED_OUT_BY**, to denote the **Checked Out By** column
- **COLUMN_REVISION_COMMENTS**, to denote the **Revision Comments** column

Specify additional attributes to configure relation attributes which need to be set on the server, such as the image metadata which is instance specific.

DocSettings - ContentValidationRules

For each document type and optionally product line, e.g. componentRoot=”topic” productLine=”busdoc, specify the content validation rules that need to be evaluated on the various trigger events.

Specify **INITIAL_VERSION** to trigger indexing at initial check in. Specify **ALL_VERSIONS** to trigger indexing each time the document is checked in.

This section defines content validation rules for various component types.

- **id** - signifies unique identifier of the rule.
- **trigger** - can have following values - **SAVE, EXPORT, OUTPUT or CUSTOM**
  - Specify **SAVE** to evaluate rules before saving document to the server.
  - Specify **EXPORT** to evaluate rules before saving document fragment to the server.
  - Specify **OUTPUT** to evaluate rules before publishing the document.

- Specify **CUSTOM** to evaluate rules using custom criteria e.g. through EI or task pane.

- **condition** - specifies CMS specific condition to evaluate rule. Rule is always evaluated if no condition is specified. The Platform adapter supports evaluation of rules based on the workflow statuses. Specify the status in the following format: **condition="Status=[Workflow Name]/[Status Name].**

Specify that the rule has to be evaluated on multiple statuses using "," as a separator, e.g. **condition="Status=W1/S1, W2/S2.**

Specify that the rule has to be evaluated on all statuses using "All" as a status value, i.e. **condition="Status=All**

- **policy** - can have following values - **PROHIBIT, WARN**
- **externalMethodIds** - specifies comma separated list of external method id

After the completion of rule evaluation external method must return object of type Quark.CMSAdapters.Core.Data.RuleEvaluationResult. You can also provide detailed information e.g. display name, description etc. of the specified external method id.

For this, you need to define another external method returning object of type Quark.CMSAdapters.Core.Data.RuleInfo and external method id should have following pattern: external method id] Info" e.g. if external method id is "CharacterCountRule" then information external method id should be "CharacterCountRuleInfo".

DocSettings - LivePreviewSettings / Publishing Preview

Both Publishing and Publishing Preview use the Publishing Channels established at the Quark Publishing Platform.

Channels may be:

- QuarkXPress Server
- DITA Open Toolkit
- Antenna House
- Custom renderers via Publishing Service

Publishing parameters are configured using the <LivePreviewSettings> element.

For each document type and product line (**componentRoot="topic" productLine="busdoc**), specify each publishing channel that should be available to the user in the Preview tab of the Word task pane. The **displayName** contains the channel name displayed in the user interface. The **id** signifies the Publishing Channel Id as defined in the Platform Server.

The `outputFormat` of the channel can have following values, specify **IMAGE_ARCHIVE** when the Publishing Channel supports an image archive for the published output, which will be rendered inside a web page. Specify

HTML_ARCHIVE when the Publishing Channel supports an HTML archive for the published output, which will be rendered as is, pointing to the specified **outputStarterFile**. The **outputStarterFile** of Channel specifies starter file name present in the HTML Archive.

The published output URI which will be rendered as is can be configured using the URI. The **downloadMethodId** signifies the external method id to invoke on invoking the download button for selected channel.

Specify any number of channel parameters as shown below. The **name** specifies the Channel Parameter name as supported by the Publishing Channel. The **value** could be any value accepted by the Publishing Channel Parameter.

```
<Channel id="channelId" outputFormat="format"
outputStarterFile="">
  <ChannelParameters>
    <ChannelParameter name="parameterName" value="parameterValue"
/>
    <ChannelParameter name="parameterName" value="parameterValue"
/>
  </ChannelParameters>
</Channel>
```

Adapter features

Connection preferences

When launching Quark XML Author for the first time, Quark XML Author uses the values specified in <ConnectionManagement> in Quark.CMSAdapters.config.

Multiple sets of connection credentials can now be created and saved by adding entries to Quark.CMSAdapters.config.

ConnectionManagement parameters

Parameter	Required	Definition
Name	yes	String. Specifies the connection name.
serverName	yes	String. Specifies the server machine name.
port	yes	String. Specifies the port number.
useHttps	Yes	Boolean. Specifies whether the communication protocol to be used is https or not. Defaults to false.

Configuration Example

```
<ConnectionManagement>

  <ConnectionInfo Name=”connection1” serverName=”Server1” port=”61400” useHttps=”false”/>

</ConnectionManagement>
```

Preferences

The Preferences dialog provides the facility to add connection preferences and other settings.

The User Interface

The following changes need to be made to configure the user interface of Quark XML Author for displaying the **Preferences** dialog. This will add an entry in the Backstage menu.

If you're uising office 2010, In **./<Languagefolder>/Appconfig.xml**, add the following under <backstage>:

```
<button id="CMSPreferences" visible="true"
insertAfterMso="ApplicationOptionsDialog" isDefinitive="true">
  <ExtensibilityMethod id="ShowPreferences"/>
</button>
```

If you're using Office 2013 or later, add the following under **<backstage2013>** in **./<Languagefolder>/Appconfig.xml>**:

```
<button id="CMSPreferences" image="none" visible="true"
insertAfterMso="ApplicationOptionsDialog" isDefinitive="true">
  <ExtensibilityMethod id="ShowPreferences"/>
</button>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id=ShowPreferences" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ShowPreferences"></Method>
```

Server operations on documents

New Document from Server Template

A new Quark XML Author document can be created using the New Document from Server Template option. This opens a copy of an existing server document that the user chooses and opens it to be edited. This document requires to be checked in as a new document and has no connection with the existing document.

The user interface:

To add this option, please add the following task. For Office 2010 installations, add this in **backstage** and for Office 2013 or later, add this in **backstage2013.**

- For BUSDOCS, In the **./<Languagefolder>/Appconfig.xml**, add the task under TabFileNew **<tab id="TabFileNew" insertBeforeMso="TabNew" visible="true" columnWidthPercent="40">** in the **<firstColumn>** under the taskGroup **<taskGroup id="TemplateTaskGroup" allowedTaskSizes="largeMediumSmall">** in the category **<category id="BUSDOCCategory">**:

```
<task id="CMSNewBUSDOCTemplate" imageMso="FileNew" visible="true"
isDefinitive="true">
  <ExtensibilityMethod id="CreateDocument"/>
</task>
```

The extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="CreateDocument" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="CreateDocument">
  <Argument type="Tokens">
    <Token>Type=XML Document Template</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.

- For DITA,
- In the **./<Languagefolder>/Appconfig.xml**, add the task under TabFileNew **<tab id="TabFileNew" insertBeforeMso="TabNew" visible="true" columnWidthPercent="40">** in the **<firstColumn>** under the taskGroup **<taskGroup id="TemplateTaskGroup"**

allowedTaskSizes="largeMediumSmall"> in the category **<category id="BUSDOCCategory">**:

```
<task id="CMSNewBUSDOCTemplate" imageMso="FileNew" visible="true"
isDefinitive="true">
  <ExtensibilityMethod id="CreateDocument"/>
</task>
```

The extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="CreateDocument" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="CreateDocument">
  <Argument type="Tokens">
    <Token>Type=XML Document Template</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.

Opening a document from the server

This invokes the **Open from Server** dialog using the **CheckOutDocument** extensibility method.

The user interface:

To configure the user interface of Quark XML Author, changes need to be made to **./<Language Folder>/AppConfig.xml** depending upon the version of Microsoft Word. For Word 2010, make the changes listed below to **<backstage>**. For Word 2013 or later, make these changes to **<backstage2013>**

```
<button id="CMSOpen" image="CMSOpen_image"
insertAfterMso="FileOpen" visible="true" isDefinitive="true">
  <ExtensibilityMethod id="CheckOutDocument"/>
</button>
```

The Extensibility Interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

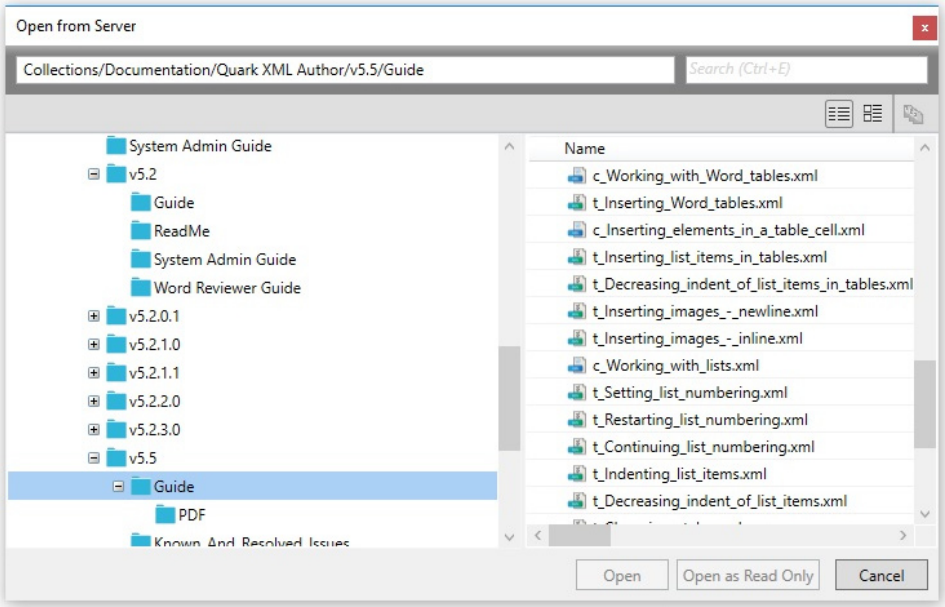
```
<Method id="CheckOutDocument"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="CheckOutDocument">
  <Argument type="Tokens">
    <Token>Type=XML Document</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens:

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. **Type** is the name of the **ContentTypeMapping** element defined in **Quark.CMSAdapters.config**. You can specify multiple document types using "," as a separator.

Token	Definition
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last location browsed. The browse location value must be as shown in the path field of the server browser.



Open from Server Dialog

Saving documents

Save Document

The `SaveDocument` feature checks in a document to the server. This feature invokes the **Check in** dialog. The dialog can be suppressed if desired.

The user interface:

To configure the user interface of Quark XML Author, changes need to be made to **./<Language Folder>/AppConfig.xml** depending upon the version of Microsoft Word.

For Word 2010, make the changes listed below to **<backstage><tab id="CMSInfoTab">.** For Word 2013 or later, make these changes to **<backstage2013><tab id="CMSInfoTab">.**

```
<group id="CMSQPPAdapter">

  <topItems>

    <layoutContainer id="CMSCheckInLayout1"
layoutChildren="horizontal">
```

```
      <button id="CMSInfoTabCheckIn" image="CMSCheckIn_image"
style="large" accessMode="Revise,Author" getEnabled="GetEnabled"
isDefinitive="true">

        <ExtensibilityMethod id="SaveDocument"/>

      </button>

      <layoutContainer id="CMSCheckInDescriptionLayout"
layoutChildren="vertical">

        <labelControl id="CMSCheckInDescriptionLabelControl"/>

        <layoutContainer id="CMSCheckInDescriptionLayout2"
layoutChildren="horizontal">

          <imageControl id="CMSCheckInBullet1Icon"
imageMso="ColorSilver"/>

          <labelControl id="CMSCheckInBullet1LabelControl"/>

        </layoutContainer>

        <layoutContainer id="CMSCheckInBullet2Layout"
layoutChildren="horizontal">

          <imageControl id="CMSCheckIn2Icon"
imageMso="ColorSilver"/>

          <labelControl id="CMSCheckInBullet2LabelControl"/>

        </layoutContainer>

      </layoutContainer>

    </layoutContainer>

  </topItems>

</group>
```

For Word 2010, make the changes listed below to <backstage>. For Word 2013 or later, make these changes to <backstage2013>.

```
<button id="CMSCheckIn" image="CMSCheckIn_image"
insertAfterMso="FileSave" visible="XA" accessMode="Revise,Author"
getEnabled="GetEnabled" isDefinitive="true">
<ExtensibilityMethod id="SaveDocument"/>

</button>
```

The extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="SaveDocumentRevision"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ExportDocument">

  <Argument type="XomRoot"/>
```

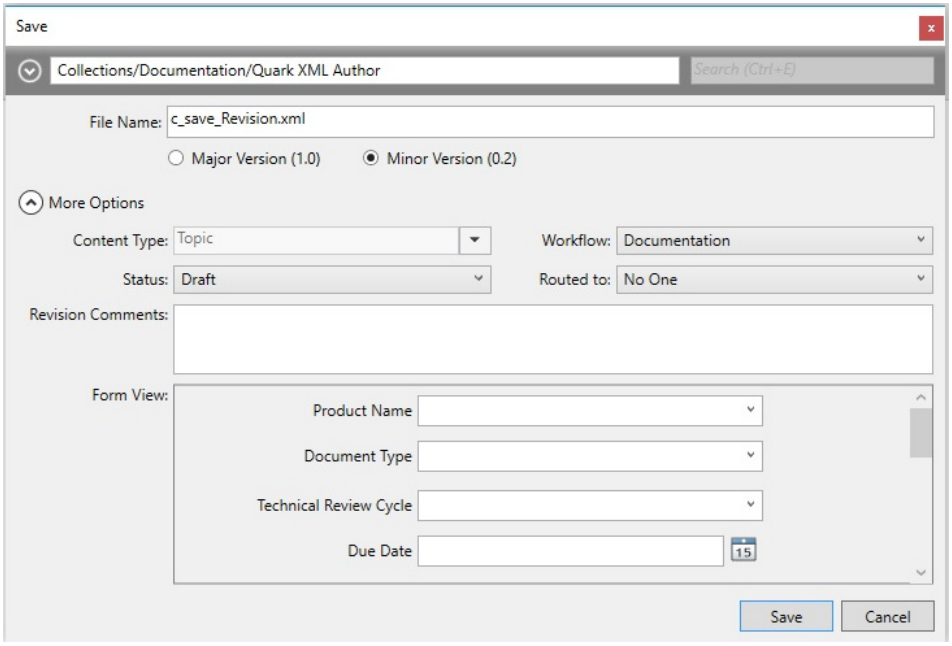
```
  <Argument type="Filename"/>
  <Argument type="Tokens">
    <Token>Type=SaveRevision</Token>
    <Token>SaveSilently=false</Token>
    <Token>SaveAsMinorVersion=true</Token>
    <Token>xslt=</Token>
    <Token>KeepMoreOptionsExpanded=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomRoot	yes	The XOM node corresponding to the current document root.
Filename	yes	The name of the file.
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	String. Specifies type of the save modes. Supported values are: **CheckIn** - the document is saved to the server and closed. **SaveRevision** - the document is saved to the server without closing it.
SaveSilently	String. Specifies whether document is saved without showing save dialog. Values supported are **false** (displays the Save dialog) and **true** (Does not display the Save dialog).
SaveAsMinorVersion	Specifies whether the document is saved as a minor version or not. Values supported are **false** and **true**.
xslt	Specifies the transform to be applied on the exported XML.

Token	Definition
KeepMoreOptionsExpanded	Specifies whether **More Options** should be expanded by default. Values supported are **false** and **true**. If not specified, the **Save** dialog remembers the last expanded state.



The Check in dialog.

Saving a document revision

Saving a document revision ensures that the current state of the document is saved to the server as a revision, while keeping the document open for further edits. This functionality is achieved using the SaveDocumentRevision extensibility method. The **Check in** dialog can be suppressed if desired.

The User Interface:

To configure the user interface of Quark XML Author, changes need to be made to **./<Language Folder>/AppConfig.xml** depending upon the version of Microsoft Word. For Word 2010, make the changes listed below to **<backstage><tab id="CMSInfoTab">.** For Word 2013 or later, make these changes to **<backstage2013><tab id="CMSInfoTab">.**

```
<group id="CMSGroupSaveRevision">

  <topItems>

    <layoutContainer id="CMSSaveRevisionLayout1"
layoutChildren="horizontal">

      <button id="CMSSaveRevision" image="CMSSaveRevision_image"
style="large" accessMode="Revise,Author" getEnabled="GetEnabled"
isDefinitive="true">

        <ExtensibilityMethod id="SaveDocumentRevision"/>

      </button>
```

```
      <layoutContainer id="CMSSaveRevisionDescriptionLayout"
layoutChildren="vertical">

        <labelControl
id="CMSSaveRevisionDescriptionLabelControl"/>

        <layoutContainer id="CMSSaveRevisionDescriptionLayout2"
layoutChildren="horizontal">

          <imageControl id="CMSSaveRevisionBullet1Icon"
imageMso="ColorSilver"/>

          <labelControl id="CMSSaveRevisionBullet1LabelControl"/>

        </layoutContainer>

        <layoutContainer id="CMSSaveRevisionBullet2Layout"
layoutChildren="horizontal">

          <imageControl id="CMSSaveRevision2Icon"
imageMso="ColorSilver"/>

          <labelControl id="CMSSaveRevisionBullet2LabelControl"/>

        </layoutContainer>

      </layoutContainer>

    </topItems>

</group>
```

The Extensibility Interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

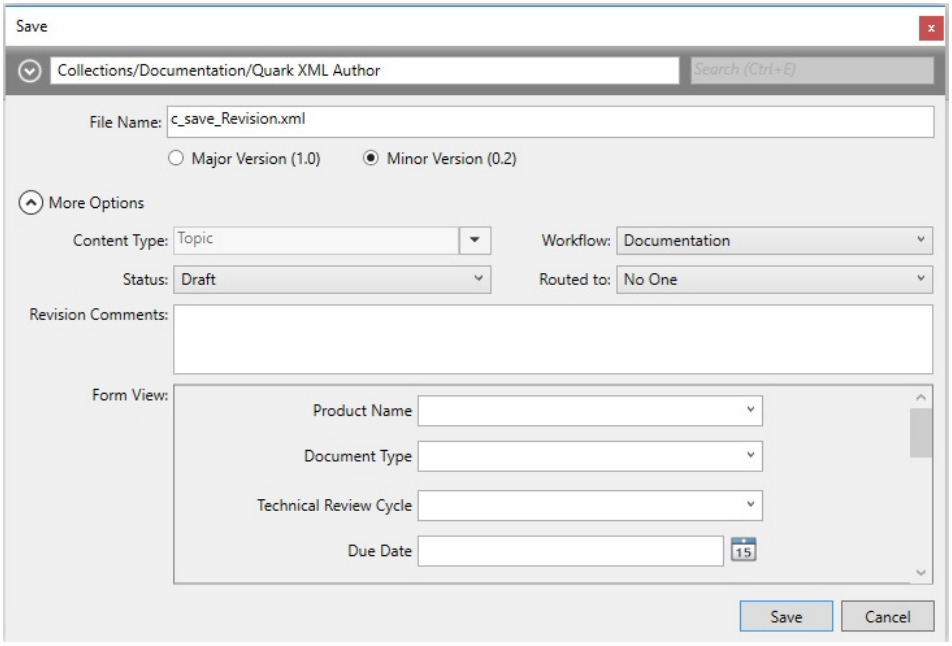
```
<Method id="SaveDocumentRevision"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ExportDocument">

  <Argument type="XomRoot"/>

  <Argument type="Filename"/>

  <Argument type="Tokens">

    <Token>Type=SaveRevision</Token>

    <Token>SaveSilently=false</Token>

    <Token>SaveAsMinorVersion=true</Token>

    <Token>xslt=</Token>

    <Token>KeepMoreOptionsExpanded=</Token>

  </Argument>

</Method>
```

Parameters	Required	Definition
XomRoot	yes	The XOM node corresponding to the current document root.
Filename	yes	Name of the open document
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Required	Definition
Type	yes	String. Specifies type of the save modes. Supported values are: **CheckIn** - the document is saved to the server and closed. **SaveRevision** - the document is saved to the server without closing it.
SaveSilently	No	String. Specifies whether document is saved without showing save dialog. Values supported are **false** (displays the Save dialog) and **true** (Does not display the Save dialog).
SaveAsMinorVersion	yes	Specifies whether the document is saved as a minor version or not. Values supported are **false** and **true**.
xslt	no	Specifies the transform to be applied on the exported XML.
KeepMoreOptionsExpanded	no	Specifies whether **More Options** should be expanded by default. Values

Token	Required	Definition
		supported are **false** and **true**. If not specified, the **Save** dialog remembers the last expanded state.



The Check in dialog.

Discard Changes

Discard changes is a feature that reverts a checked-out document to its original state. All changes made to the document are discarded and the checked-out flag is removed from the document.

The Discard Changes user interface

To configure the user interface of Quark XML Author, changes need to be made to **./<Language Folder>/AppConfig.xml** depending upon the version of Microsoft Word. For Word 2010, make the changes listed below to **<backstage><tab id="CMSInfoTab">.** For Word 2013 or later, make these changes to **<backstage2013><tab id="CMSInfoTab">.**

```
<group id="CMSGroupCancelCheckout">

  <topItems>

    <layoutContainer id="CMSCancelCheckoutLayout1"
layoutChildren="horizontal">

      <button id="CMSCancelCheckout" keytip="IM"
image="CMSCancelCheckout_image" style="large" accessMode="Revise"
getEnabled="GetEnabled" isDefinitive="true">

        <ExtensibilityMethod id="CancelCheckoutDocument"/>

      </button>
```

```
      <layoutContainer id="CMSCancelCheckoutDescriptionLayout"
layoutChildren="vertical">

        <labelControl
id="CMSCancelCheckoutDescriptionLabelControl"/>

        <layoutContainer id="CMSCancelCheckoutDescriptionLayout2"
layoutChildren="horizontal">

          <imageControl id="CMSCancelCheckoutBullet1Icon"
imageMso="ColorSilver"/>

          <labelControl
id="CMSCancelCheckoutBullet1LabelControl"/>

        </layoutContainer>

        <layoutContainer id="CMSCancelCheckoutBullet2Layout"
layoutChildren="horizontal">

          <imageControl id="CMSCancelCheckout2Icon"
imageMso="ColorSilver"/>

          <labelControl
id="CMSCancelCheckoutBullet2LabelControl"/>

        </layoutContainer>

      </layoutContainer>

    </layoutContainer>

  </topItems>

</group>
```

The CancelCheckoutDocument EI is invoked on selecting **Discard Changes** in the backstage menu or by clicking Close.

The Extensibility Interface

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="CancelCheckoutDocument"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="CancelCheckoutDocument">

  <Argument type="XomRoot"/>

  <Argument type="Filename"/>

  <Argument type="Tokens">

    <Token>CancelCheckoutSilently=false</Token>

    <Token>Recursive=false</Token>

  </Argument>
```

```
</Method>
```

Parameter	Required	Definition
XomRoot	yes	Xom Node
Filename	yes	The name of the file.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
CancelCheckoutSilently	Specifies whether cancel checkout is done without displaying the confirmation dialog. Values supported are **false** (displays the dialog) and **true** (suppresses the dialog).
Recursive	Specifies whether this method should cancel checkout all nested inline references along with the document.

Edit

Documents opened as read-only can be edited using **Edit.** The document is checked out and the read-only flag is removed. The read-only version of the document is closed and an editable copy of the same document is checked out from the server and opened. The status of the document changes to **Checked-out.**

The CheckOutOpenedDocument EI is invoked on selecting **Edit** in the backstage menu.

The Edit user interface

To configure the user interface of Quark XML Author, changes need to be made to **./<Language Folder>/AppConfig.xml** depending upon the version of Microsoft Word. For Word 2010, make the changes listed below to <backstage><tab id="CMSInfoTab">. For Word 2013 or later, make these changes to <backstage2013><tab id="CMSInfoTab">.

```
<group id="CMSGroupCheckout">
  <topItems>
    <layoutContainer id="CMSCheckoutLayout1" layoutChildren="horizontal">
      <button id="CMSCheckout" keytip="IM" image="CMSOpen_image" style="large" accessMode="Review" getEnabled="GetEnabled" isDefinitive="true">
        <ExtensibilityMethod id="CheckOutOpenedDocument"/>
      </button>
```

```
        <layoutContainer id="CMSCheckoutDescriptionLayout"
layoutChildren="vertical">
            <labelControl id="CMSCheckoutDescriptionLabelControl"/>
            <layoutContainer id="CMSCheckoutDescriptionLayout2"
layoutChildren="horizontal">
                <imageControl id="CMSCheckoutBullet1Icon"
imageMso="ColorSilver"/>
                <labelControl id="CMSCheckoutBullet1LabelControl"/>
            </layoutContainer>
            <layoutContainer id="CMSCheckoutBullet2Layout"
layoutChildren="horizontal">
                <imageControl id="CMSCheckout2Icon" imageMso="ColorSilver"/>
                <labelControl id="CMSCheckoutBullet2LabelControl"/>
            </layoutContainer>
        </layoutContainer>
    </layoutContainer>
  </topItems>
</group>
```

The Extensibility Interface

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="CheckOutOpenedDocument"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="CheckOutDocument">
    <Argument type="XomRoot"/>
    <Argument type="Filename"/>
</Method>
```

Parameter	Required	Definition
XomRoot	yes	The XOM node corresponding to the current document root.
Filename	yes	The name of the file.

Send for review

Send for review is a feature that allows existing XML documents that have already been checked-in to the Quark Publishing Platform Server to be checked out and sent for review. New XML documents created in Quark XML Author require being checked-in to the Quark Publishing Platform Server prior to being sent out for review. This section explains how to configure Send for review.

Send for review overview

Send for review is accomplished through the use of an Extensibility Interface method.

Here is an overview of what happens when an author sends the document out for review:

1. The file to be sent for review is checked out and the necessary changes made.
2. The author clicks on File > Info and chooses Save to Server & Send as Word document for review in the backstage view.
3. This invokes the SendForReview EI. SendForReview initiates the .docx creation and document save and check in process. A relation is then created between the .docx and the XML Document.

The SendForReview user interface

To configure the user interface of Quark XML Author for the review process, changes need to be made to **./<Language Folder>/AppConfig.xml** depending upon the version of Microsoft Word. For Word 2010, make the changes listed below to <backstage><tab id="CMSInfoTab">. For Word 2013 or later, make these changes to <backstage2013><tab id="CMSInfoTab">.

```
<group id="CMSGroupReview">
  <topItems>
    <layoutContainer id="CMSReviewLayout1"
layoutChildren="horizontal">
      <button id="CMSInfoTabSendForReview"
image="CMSCheckIn_image" style="large" accessMode="Revise"
getEnabled="GetEnabled" isDefinitive="true">
        <ExtensibilityMethod id="SendForReview"/>
      </button>
      <layoutContainer id="CMSReviewDescriptionLayout"
layoutChildren="vertical">
        <labelControl id="CMSReviewDescriptionLabelControl"/>
```

```
        <layoutContainer id="CMSReviewDescriptionLayout2"
layoutChildren="horizontal">
          <imageControl id="CMSReviewBullet1Icon"
imageMso="ColorSilver"/>
          <labelControl id="CMSReviewBullet1LabelControl"/>
        </layoutContainer>
        <layoutContainer id="CMSReviewBullet2Layout"
layoutChildren="horizontal">
          <imageControl id="CMSReview2Icon"
imageMso="ColorSilver"/>
          <labelControl id="CMSReviewBullet2LabelControl"/>
        </layoutContainer>
      </layoutContainer>
    </layoutContainer>
  </topItems>
</group>
```

The Extensibility interface:

The InitReviewFeature EI is added in the connect event to initialize the assembly Quark.CMSAdapters.Features.XA.QPP.WordReview when Quark XML Author is launched.

In **./<Language Folder>**/AppConfig.xml add the following EI in <ExtensibilityInterface>

<Method id="InitReviewFeature" assembly="Quark.CMSAdapters.Features.XA.QPP.WordReview" class="Quark.CMSAdapters.Features.XA.QPP.WordReview.ExtensibilityMethods" method="Init"/>

In **./<Language Folder>**/AppConfig.xml add the following in <ExtensibilityInterface><Connect>

<!— Platform Adapter Feature - Word Review —>

<ExtensibilityMethod id="InitReviewFeature"/>

<!— Platform Adapter Feature - Word Review —>

The SendForReview EI is invoked on clicking the Save to Server & Send as Word document for review button.

In **./<Language Folder>/AppConfig.xml** add the following EI in <ExtensibilityInterface>

```
<Method id="SendForReview"
assembly="Quark.CMSAdapters.Features.XA.QPP.WordReview"
class="Quark.CMSAdapters.Features.XA.QPP.WordReview.Extensibility
Methods" method="SendForReview">
  <Argument type="XomRoot"/>
  <Argument type="Tokens">
    <Token>SaveEI=SaveDocument</Token>
    <Token>UIRestrictionsFile=en/BUSDOCS/customUI14.xml</Token>
    <Token>ProtectionType=0</Token>
    <Token>AttributeMappingComponentType=reviewdocument</Token>
    <Token>SearchAttribute=Global ID</Token>
    <Token>SaveSilently=false</Token>
    <Token>SaveAsMinorVersion=</Token>
    <Token>KeepMoreOptionsExpanded=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomRoot	yes	Xom Node.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
SaveEI	String. Specifies the method id to save active document. Defaults to **SaveDocument**
UIRestrictionsFile	String. Specifies the xml file path containing desired ribbon configuration when review document is opened.
ProtectionType	String. Specifies the allowed actions in the review document.Values supported are "0" (comments and changes) and "1" (comments only)
AttributeMappingComponentType	String. Specifies the root of the ComponentType in the Quark.CMSAdapters.config. This defines the attribute mapping for the document to be reviewed.
SearchAttribute	String. Global ID
SaveSilently	String. Specifies whether the document sent for review

Token	Definition
	is saved without displaying the Save dialog. Values supported are “false” (With Save Dialog) and “true” (Without Save Dialog).
SaveAsMinorVersion	String. Specifies whether the review document is saved as minor version or not. Values supported are “false” and “true”.
KeepMoreOptionsExpanded	String. Specifies whether more options should be in expanded state by default. Values supported are “false” and “true”. In case this token is not specified, save dialog remembers last expanded state.

Import review comments

Import review comments is a feature that facilitates importing comments made by reviewers in the Word file generated during the review process. This section explains how to configure Quark XML Author to import comments made by reviewers.

Overview

The import of review comments from Word is accomplished through the use of an Extensibility Interface method. Here is an overview of what happens when comments are imported from the Word document:

1. The Quark XML Author document is checked out by the user.
2. This invokes the BeforeOpenEventHandler EI which checks if a reviewed document exists.
3. The user is prompted to either import the comments if any, or discard the comments and open the document.

Extensibility Interface

When a reviewed document is opened, the EI checks if a related reviewed .docx file exists for this document. If the docx file exists, the user is prompted to import comments if any.

```
For all Namespaces defined in AppConfig.xml, add the highlighted
<Argument> in <BeforeOpen>

 <BeforeOpen assembly=”Quark.CMSAdapters.Core.XA”
class=”Quark.CMSAdapters.Core.XA.ExtensibilityMethods”
method=”BeforeOpenEventHandler” >

  <Argument type=”ExportedRoot”></Argument>

  <Argument type=”Filename”></Argument>

  <Argument type=”OpenSettings”></Argument>
```

```
  <!– Platform Adapter Feature - Word Review –>
  <Argument type=”Tokens”>
    <Token>Type=Reviewed Word Document</Token>
  </Argument>
  <!– Platform Adapter Feature - Word Review –>
</BeforeOpen>
```

Token	Definition
Type	specifies the type of the document sent for review. Type value is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config.

Closing a server document

The close operation for documents opened from the server in the **Edit** mode varies slightly from documents opened from the local machine. On closing the document, changes made to server documents need to be saved to the server, saved as a draft, or discarded. The document can also be closed, saving it locally and later reopened.

The BeforeDocClosedEventHandler EI handles the close operations of server documents opened for editing.

The BeforeDocClosedEventHandler user interface

Choosing **Close** in a document opened in the **Edit** mode displays the close document dialog, prompting the user to either save the document to the server, discard the changes made to the document or close the document. Choosing **Cancel** preserves changes made to the document locally.

The Extensibility Interface

In **./<Languagefolder>/BUSDOCS/config.xml**, add the following:

```
 <Method id=”BeforeDocClosedEventHandler”
assembly=”Quark.CMSAdapters.Core.XA”
class=”Quark.CMSAdapters.Core.XA.ExtensibilityMethods”
method=”BeforeDocClosedEventHandler”>
       <Argument type=”XomRoot”/>
       <Argument type=”Tokens”>
         <Token>CloseOption=ASK</Token>
         <Token>SAVE_CHANGES_EI=SaveDocument</Token>
         <Token>DISCARD_CHANGES_EI=CancelCheckoutDocument</Token>
       </Argument>
 </Method>
```

Parameter	Required	Definition
XomRoot	yes	Xom Node
Tokens	yes	A string array of tokens used in the config.xml file

Tokens

Token	Definition
CloseOption	Options available in the CloseOption token are SAVE_CHANGES, DISCARD_CHANGES, CLOSE and ASK.
SAVE_CHANGES_EI	The ID of the EI that saves the changes to the server.
DISCARD_CHANGES_EI	The ID of the EI that discards any changes made to the document.

Save Draft to Server

The **Save Draft to Server** feature enables saving changes made to a document onto the server without creating a revision of the document. The version number of the document remains unchanged.

Save Draft to Server overview

Save Draft to Server is accomplished through the use of an Extensibility Interface method.

Here is an overview of what happens when the draft of a document is saved:

1. The version of the checked-out document remains the same. If the document is closed without checking in, a local copy of the document is saved and reopened on subsequent check-outs on the same machine. The SaveDraft EI is invoked.
2. If the Save Draft to Server functionality is made available to the user without invoking a user interface, the AfterSave event is called after a Save or an AutoSave event, which saves a draft of the document to the server.

The Save Draft to Server user interface

To configure the user interface of Quark XML Author, changes need to be made to **./<Language Folder>/AppConfig.xml** depending upon the version of Microsoft Word. For Word 2010, make the changes listed below to **<backstage><tab id="CMSInfoTab">.** For Word 2013 or later, make these changes to **<backstage2013><tab id="CMSInfoTab">.**

```
<group id="CMSGroupSaveDraft">
```

```
            <topItems>
              <layoutContainer id="CMSSaveDraftLayout1"
layoutChildren="horizontal">
                <button id="CMSSaveDraft"
image="CMSSaveDraft_image" style="large" accessMode="Revise"
getEnabled="GetEnabled" isDefinitive="true">
                  <ExtensibilityMethod id="SaveDraft"/>
                </button>
                <layoutContainer
id="CMSSaveDraftDescriptionLayout" layoutChildren="vertical">
                  <labelControl
id="CMSSaveDraftDescriptionLabelControl"/>
                  <layoutContainer
id="CMSSaveDraftDescriptionLayout2" layoutChildren="horizontal">
                    <imageControl id="CMSSaveDraftBullet1Icon"
imageMso="ColorSilver"/>
                    <labelControl
id="CMSSaveDraftBullet1LabelControl"/>
                  </layoutContainer>
                  <layoutContainer
id="CMSSaveDraftBullet2Layout" layoutChildren="horizontal">
                    <imageControl id="CMSSaveDraft2Icon"
imageMso="ColorSilver"/>
                    <labelControl
id="CMSSaveDraftBullet2LabelControl"/>
                  </layoutContainer>
                </layoutContainer>
              </layoutContainer>
            </topItems>
          </group>
```

The extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="SaveDraft" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="SaveDraft">
        <Argument type="XomRoot"/>
        <Argument type="Filename"/>
        <Argument type="SavedState"/>
```

```
</Method>
```

Parameter	Required	Definition
XomRoot	yes	The XOM node corresponding to the current document root.
Filename	yes	If specified, following argument will ensure to save document before saving draft. Remove this argument if this EI is hooked to AfterSave event.
SavedState	yes	The current state of the document.

Publishing

Documents created in Quark XML Author can be published to various formats. The type of output formats available depend upon the flavor of Quark XML Author. The available publishing channels on Platform server can easily be configured in Quark XML Author using the **Publish** method of class **Quark.CMSAdapters.Core.XA.Publishing.QPP.ExtensibilityMethods** present in **Quark.CMSAdapters.Core.XA.Publishing.QPP** assembly.

Server operations on components

Add Reference

Various types of assets available on the Platform server can be inserted in a document as references. The GetElement extensibility method has been extended to provide the capability to insert various reference types. The following reference types are currently supported:

- Media references: This includes image, audio and video references.
- Structured table references: This includes Structured table from Server.
- Document references: This includes Quark XML Author documents checked onto the Server
- Excel references: This includes Excel tables and Excel charts.
- PowerPoint references: This includes PowerPoint slides
- Visio references: This includes Visio drawings

Adding media references

Media references like images, video and audio can be added to Quark XML Author documents.

Adding a Video reference overview

The ReferenceVideoFromServer extensibility method is used to insert the selected video reference.

The user interface

To configure the user interface of Quark XML Author for inserting a video from server, the following changes need to be made in the XAS files.

Adding the element definition for Video:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

For DITA: **./<Language Folder>/DITA/common-elems.xas**

```
<ElementDef name="video" friendly="Video from server"
style="Media" referenceAttribute="conref"
excludeFromChangeToContextMenu="true"
excludeFromContextMenu="true"
externalMethodId="ReferenceVideoFromServer"
externalMethodFriendly="Video from server">
  <Section>
    <Para/>
  </Section>
  <Attributes>
    <AttributeType name="outputclass"/>
  </Attributes>
  &cmsextensibility;
</ElementDef>
```

Adding the context menu entry for Video:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

For DITA: **./<Language Folder>/DITA/topic.xas**

Add the following sectionType in the ElementDef of body:

<SectionType name="video" readonly="true"/>

The Extensibility interface for video:

In **./<Languagefolder/Appconfig.xml**, add the following:

```
<Method id="ReferenceVideoFromServer"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="GetElement">
  <Argument type="Tag"/>
```

```
  <Argument type="Tokens">
    <Token>Type=Video Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tag	yes	Tag is a special Argument type that supplies the label of an item selected in the user interface..
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Adding an Audio reference overview

The ReferenceAudioFromServer extensibility method is used to insert the selected audio reference.

The User Interface

To configure the user interface of Quark XML Author for inserting an audio from server, the following changes need to be made in the XAS files.

Adding the element definition for Audio:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

For DITA: **./<Language Folder>/DITA/common-elems.xas**

```
 <ElementDef name="audio" friendly="Audio from server"
style="Media" referenceAttribute="conref"
excludeFromChangeToContextMenu="true"
```

```
excludeFromContextMenu="true"
externalMethodId="ReferenceAudioFromServer"
externalMethodFriendly="Audio from server">
  <Section>
    <Para/>
  </Section>
  <Attributes>
    <AttributeType name="outputclass"/>
  </Attributes>
  &cmsextensibility;
</ElementDef>
```

Adding the context menu entry for Audio:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

For DITA: **./<Language Folder>/DITA/topic.xas**

Add the following sectionType in the ElementDef of body:

```
<SectionType name="audio" readonly="true"/>
```

The Extensibility interface for Audio:

In **./<Languagefolder/Appconfig.xml**, add the following:

```
<Method id="ReferenceAudioFromServer"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="GetElement">
  <Argument type="Tag"/>
  <Argument type="Tokens">
    <Token>Type=Audio Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tag	yes	Tag is a special Argument type that supplies the label of an item selected in the User Interface.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using “,” as a separator.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Adding an Image reference overview

The User Interface

The following changes need to be made to configure the user interface of Quark XML Author for inserting an Image from Server. This will add an entry in the context menu:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

For DITA: **./<Language Folder>/DITA/common-elems.xas**

Add the externalMethodID and its externalMethodFriendly in the ElementDef of Image.

Example:

```
<ElementDef name=”image” friendly=”Image” visible=”false”
style=”XA_Heading 8” referenceAttribute=”conref”
tooltipTransform=”DITA/element-tooltips.xsl”
externalMethodId=”ReferenceImageFromServer”
externalMethodFriendly=”Image from server”>
```

Also add **&cmsextensibility;** before the end of the ElementDef.

Adding a button in the ribbon

The following changes need to be made to add an entry in the Ribbon. This will split the Picture button in the Insert tab and add **Picture from Server**.

For BUSDOCS: **./<Language Folder>/BUSDOCS/config.xml**

For DITA: **./<Language Folder>/DITA/sme-config.xml**

Add the following under **GroupIllustrations**

```
<splitButton id=”PictureMenu” size=”large”>
  <button id=”InsertElementPicture”
imageMso=”PictureInsertFromFile” >
    <InternalClass name=”InsertElement”
displayName=”Figure[Picture[image(*)]]”/>
```

```
    </button>
    <menu id="Picture_XA_Menu">
    <button id="CMSPicture" imageMso="PictureInsertFromFile">
      <InternalClass name="InsertElement"
displayName="Picture[image(*)]"/>
        </button>
        <button id="CMSPictureFromServer"
imageMso="PictureInsertFromFile">
      <InternalClass name="InsertElement"
useExternalMethodId="true" displayName="Picture[image(*)]"/>
        </button>
  </menu>
</splitButton>
```

The Extensibility interface for Image:

In **./<Languagefolder/Appconfig.xml**, add the following:

```
<Method id="ReferenceImageFromServer"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="GetElement">
  <Argument type="Tag"/>
  <Argument type="Tokens">
    <Token>Type=Picture Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
    <Token>Figure/Picture=<fig><image href="{0}"/></fig></Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tag	yes	Tag is a special Argument type that supplies the label of an item selected in the User Interface.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
Figure/Picture	This token specifies the XML string format corresponding to the Tag value.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Adding Structured table references

Structured tables can be added to Quark XML Author documents just like images, video and audio.

Adding a Structured table reference overview

The ReferenceTableFromServer extensibility method facilitates adding a structured table to the Quark XML Author document.

The User Interface

The following changes need to be made to configure the user interface of Quark XML Author for inserting a structured table from Server. This will add an entry in the context menu:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

Add the externalMethodID and its externalMethodFriendly in the ElementDef of CalsTable.

Example:

<ElementDef name="CalsTable" friendly="Table" xmlname="table" referenceAttribute="conref" **externalMethodId="ReferenceTableFromServer" externalMethodFriendly="Table from server"**>

Also add **&cmsextensibility;** before the end of the ElementDef.

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="ReferenceTableFromServer"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="GetElement">

  <Argument type="Tag"/>
```

```
  <Argument type="Tokens">
    <Token>Type=Structured Table Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tag	yes	Tag is a special Argument type that supplies the label of an item selected in the User Interface.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Adding document references

Existing documents checked in to the server can be added to a Quark XML Author document as section references. Up to six levels of section references can be added to a document, as long as the hierarchical levels never exceed 6.

As an example, a document containing 3 levels of section references cannot be referred as a section in another document at the hierarchical level 4 as the total hierarchical count will then exceed 6. This document can, however, be referred at level 3.

The user interface:

Make the following changes in **./<Languagefolder>/BUSDOCS/BUSDOCS.xas:**

Add the **externalMethodID** and its **externalMethodFriendly** in the **ElementDef** of **section1**, **section2**, **section3**, **section4**, **section5** and **section6.**

For Example:

```
<ElementDef name="section1" friendly="Section 4" xmlname="topic"
visible="false" referenceAttribute="conref"
externalMethodId="GetElement" externalMethodFriendly="Section 1
from server">

<ElementDef name="section2" friendly="Section 4" xmlname="topic"
visible="false" referenceAttribute="conref"
externalMethodId="GetElement" externalMethodFriendly="Section 2
from server">

<ElementDef name="section3" friendly="Section 4" xmlname="topic"
visible="false" referenceAttribute="conref"
externalMethodId="GetElement" externalMethodFriendly="Section 3
from server">

<ElementDef name="section4" friendly="Section 4" xmlname="topic"
visible="false" referenceAttribute="conref"
externalMethodId="GetElement" externalMethodFriendly="Section 4
from server">

<ElementDef name="section5" friendly="Section 4" xmlname="topic"
visible="false" referenceAttribute="conref"
externalMethodId="GetElement" externalMethodFriendly="Section 5
from server">

<ElementDef name="section6" friendly="Section 4" xmlname="topic"
visible="false" referenceAttribute="conref"
externalMethodId="GetElement" externalMethodFriendly="Section 6
from server">
```

In each of the above **ElementDef** add "**&cmsextensibility;**" before the end of the **ElementDef**.

The user interface:

Adding a button in the ribbon

The following changes need to be made to add an entry in the **Ribbon**. This will split the **Insert Section** button in the **Insert** tab and add Section from Server.

For BUSDOCS: Make the changes in the **./<Language Folder>/BUSDOCS/config.xml**

```
<group id="GroupStyles">

  <splitButton id="Section1Menu" size="large">

    <button id="Section1_XA">

      <InternalClass name="InsertElement" displayName="Section
1[Heading 1(<placeholder>%#ls%#Section 1
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>

    </button>

    <menu id="Section1_XA_Menu">

      <button id="Section1_XA_1" image="Section1_XA_image"
label="Section 1">
```

```
        <InternalClass name="InsertElement" displayName="Section
1[Heading 1(<placeholder>%#ls%#Section 1
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>

      </button>

      <button id="CMSSection1FromServer"
image="Section1_XA_image" >

        <InternalClass name="InsertElement"
useExternalMethodId="true" displayName="Section 1[Heading
1(<placeholder>%#ls%#Section 1
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>

      </button>

    </menu>

  </splitButton>

  <splitButton id="Section2Menu" size="large">

    <button id="Section2_XA">

      <InternalClass name="InsertElement" displayName="Section
2[Heading 2(<placeholder>%#ls%#Section 2
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>

    </button>

    <menu id="Section2_XA_Menu">

      <button id="Section2_XA_1" image="Section2_XA_image"
label="Section 2">

        <InternalClass name="InsertElement" displayName="Section
2[Heading 2(<placeholder>%#ls%#Section 2
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>

      </button>

      <button id="CMSSection2FromServer"
image="Section2_XA_image" >

        <InternalClass name="InsertElement"
useExternalMethodId="true" displayName="Section 2[Heading
2(<placeholder>%#ls%#Section 2
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>

      </button>

    </menu>

  </splitButton>

  <splitButton id="Section3Menu" size="large">

    <button id="Section3_XA">

      <InternalClass name="InsertElement" displayName="Section
3[Heading 3(<placeholder>%#ls%#Section 3
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>
```

```
    </button>
    <menu id="Section3_XA_Menu">
      <button id="Section3_XA_1" image="Section3_XA_image"
label="Section 3">
        <InternalClass name="InsertElement" displayName="Section
3[Heading 3(<placeholder>%#ls%#Section 3
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>
      </button>
      <button id="CMSSection3FromServer"
image="Section3_XA_image" >
        <InternalClass name="InsertElement"
useExternalMethodId="true" displayName="Section 3[Heading
3(<placeholder>%#ls%#Section 3
Title%#rs%#<%#fs%#placeholder>)/Section Body]"/>
      </button>
    </menu>
  </splitButton>
</group>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="GetElement" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="GetElement">
  <Argument type="Tag"/>
  <Argument type="Tokens">
    <Token>Type=XML Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tag	yes	Tag is a special Argument type that supplies the label of an item selected in the user interface.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Adding Excel references

Excel tables and charts can be added to Quark XML Author documents as references.

Adding Excel references overview

The functionality is achieved using the Quark.CMSAdapters.Core.XA.ChartsAndTables.dll.

The following settings are defined in the appSettings tag of Quark.CMSAdapters.Core.XA.ChartsAndTables.dll.config file that facilitates the insertion of Excel tables and charts in Quark XML Author.

Key	Value	Description
IDataSource	Quark.CMSAdapters.Core.XA.ChartsAndTables	This defines comma separated assembly names that implement the iDataSource interface.
ISourceReader	Quark.CMSAdapters.Core.XA.Publishing.QPP	This defines comma separated assembly names that implement the ISourceReader interface.
IFormatter	Quark.CMSAdapters.Core.XA.Publishing.QPP	This defines comma separated assembly names that implement the IFormatterinterface.
ClipboardTransform	SP_Clipboard.xslt	Defines the relative path for the exe to the clipboard transform.
FilterTransform	SP_Filter.xslt	Defines the relative path for the exe to the filter transform.

Key	Value	Description
SmartPasteTransform	SmartPaste.xslt	Defines the relative path for the exe to the SmartPaste transform.
DataTransform		Defines the relative path for the exe to the Data transform.
ImageVerticalResolution	Value = the required resolution of the image in pixels. e.g. value=”300”	Defines the vertical resolution of the image.
ImageHorizontalResolution	Value = the required resolution of the image in pixels. e.g. value=”300”	Defines the horizontal resolution of the image.

Excel Table and Table Image

The user interface

The following element definitions need to be added in **./<LanguageFolder>/BUSDOCS/BUSDOCS.xas**

```
<!— - - - - - - - - - - - - - - Excel Table - - - - - - - - - - - -
- - - - —>

<ElementDef name=”ExcelTable” friendly=”Table from Excel”
xmlname=”table” referenceAttribute=”conref” visible=”false”
excludeFromContextMenu=”true”
excludeFromChangeToContextMenu=”true”
externalMethodId=”ReferenceTable” externalMethodFriendly=”Table
from Excel”>

  <Table defaultColumnCount=”3”>

    <BeforeTable>

      <Sequence minOccurs=”0”>

        <SectionType name=”title” friendly=”Table Title”/>

      </Sequence>

      <Sequence minOccurs=”0”>

        <SectionType name=”desc” friendly=”Table Description”/>

      </Sequence>

    </BeforeTable>

    <TableGroupType name=”tgroup”/>

    <TableStyles default=”Default”>
```

```
        <TableStyle naefault”>


          <Table>
            <Section name= “ backColor=”LightGray”/>
            <Section name=”tbody”/>
          </Table>
        </TableStyle>
      </TableStyles>
    </Table>
    <Attributes>
      <AttributeType name=”id”/>
      <AttributeType name=”conref”/>
      <AttributeType name=”frame”/>
      <AttributeType name=”rowsep”/>
      <AttributeType name=”colsep”/>
      <AttributeType name=”tabledef” fixed=”excel”/>
    </Attributes>
        &cmsextensibility;
</ElementDef>
<!— - - - - - - - - - - - - - - - Excel Table Image - - - - - - - -
- - - - - - - - —>
<ElementDef name=”tableimage” friendly=”Table Image”
xmlname=”image” referenceAttribute=”conref” visible=”false”
excludeFromContextMenu=”true”
excludeFromChangeToContextMenu=”true”
externalMethodId=”ReferenceTableImage”
externalMethodFriendly=”Table Image from Excel”>
    <Reference>
      <Media xmlname=”image” required=”true”
excludeFromComponentContextMenu=”true”>
        <Mimetypes>
          <Mimetype type=”image/bmp”/>
          <Mimetype type=”image/dib”/>
          <Mimetype type=”image/emf”/>
          <Mimetype type=”image/emz”/>
          <Mimetype type=”image/eps”/>
```

```
        <Mimetype type="image/gif"/>
        <Mimetype type="image/jfif"/>
        <Mimetype type="image/jpe"/>
        <Mimetype type="image/jpeg"/>
        <Mimetype type="image/jpg"/>
        <Mimetype type="image/png"/>
        <Mimetype type="image/rle"/>
        <Mimetype type="image/tif"/>
        <Mimetype type="image/tiff"/>
        <Mimetype type="image/wmf"/>
        <Mimetype type="image/wmz"/>
      </Mimetypes>
    </Media>
  </Reference>
  <Attributes>
    <AttributeType name="href"/>
    <AttributeType name="width"/>
    <AttributeType name="height"/>
    <AttributeType name="widthdpi"/>
    <AttributeType name="heightdpi"/>
    <AttributeType name="scale"/>
    <AttributeType name="scalefit"/>
    <AttributeType name="imagedef" fixed="exceltable"/>
  </Attributes>
      &cmsextensibility;
</ElementDef>
```

Adding a button in the ribbon

The following changes need to be made to add an entry in the Ribbon. This will split the Table button in the Insert tab to add Table from Excel and Table Image from Excel

For BUSDOCS: **./<Language Folder>/BUSDOCS/config.xml**

Add the following under **GroupTables** in the **TabInsert**

```
<splitButton id="TableMenu" size="large">
  <button id="InsertTable" imageMso="TableInsertDialogWord" >
```

```
    <InternalClass name="InsertTable"/>
    <ShortcutKey key="T" shift="false" ctrl="true"/>
  </button>
  <menu id="Table_XA_Menu">
    <button id="CMSTable" imageMso="TableInsertDialogWord">
      <InternalClass name="InsertTable"/>
    </button>
    <!– Platform Adapter –>
    <button id="CMSTableFromExcel" imageMso="TableInsertExcel">
      <InternalClass name="InsertElement"
useExternalMethodId="true" displayName="Table from Excel(~)" />
    </button>
    <button id="CMSTableImageFromExcel">
      <InternalClass name="InsertElement"
useExternalMethodId="true" displayName="Table Image[image(*)]" />
    </button>
    <!– Platform Adapter –>
  </menu>
</splitButton>
```

Adding the context menu entry:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

Add the following in the ElementDef of body:

<TableType name="ExcelTable"/>

<ReferenceType name="tableimage" />

The Extensibility interface:

Add the following extensibility method in the **./<Language Folder>/AppConfig.xml**

```
<Method id="ReferenceTable"
assembly="Quark.CMSAdapters.Core.XA.ChartsAndTables"
method="ReferenceRange"
class="Quark.CMSAdapters.Core.XA.ChartsAndTables.ExtensibilityMet
hods">
  <Argument type="XomCurrentNode"/>
  <Argument type="XomPreviousNode"/>
  <Argument type="Tokens">
    <Token>Type=Excel Table</Token>
    <Token>Filter=Excel Files|*.xlsx;*.xlsm;*.xlsb</Token>
```

```
    <Token>DefaultBrowseLocation=</Token>
    <Token>BuilderXSL=TableBuilder.xslt</Token>
    <Token>OutputFormat=application/xml-cals</Token>
    <Token>PreviewFormat=image/png</Token>
    <Token>AutoUpload=false</Token>
    <Token>PreventAutoUploadChange=false</Token>
    <Token>ExcelTableType=Microsoft Excel Table</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	Current Xom node.
XomPreviousNode	yes	Tag is a special Argument type that supplies the label of an item selected in the user interface.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
Filter	Specifies the filter for the local file browser.
PreviewFormat	Specifies the preview data format.
AutoUpload	Specifies whether the local references should be automatically uploaded while saving the parent document to the server. Supported values are **false** and **true**.
OutputFormat	Specifies the output data format.
BuilderXSL	Specifies the transform to build the table element. You can specify an absolute or a relative path to the Quark XML Author installation directory
PreventAutoUploadChange	Specifies whether users should be prevented from changing the auto upload value. Supported values are **true** and **false**.

Token	Definition
ExcelTableType	Specifies the type of Excel tables shown in the server browser.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

```
 <Method id="ReferenceTableImage"
assembly="Quark.CMSAdapters.Core.XA.ChartsAndTables"
method="ReferenceRange"
class="Quark.CMSAdapters.Core.XA.ChartsAndTables.ExtensibilityMethods">
  <Argument type="XomCurrentNode"/>
  <Argument type="XomPreviousNode"/>
  <Argument type="Tokens">
    <Token>Type=Excel Table</Token>
    <Token>Filter=Excel Files|*.xlsx;*.xlsm;*.xlsb</Token>
    <Token>DefaultBrowseLocation=</Token>
    <Token>BuilderXSL=TableBuilder.xslt</Token>
    <Token>OutputFormat=image/png</Token>
    <Token>PreviewFormat=image/png</Token>
    <Token>AutoUpload=false</Token>
    <Token>PreventAutoUploadChange=false</Token>
    <Token>ExcelTableType=Microsoft Excel Table</Token>
  </Argument>
 </Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	Current Xom node.
XomPreviousNode	yes	Tag is a special Argument type that supplies the label of an item selected in the user interface.
Tokens	yes	A string array of tokens used in the AppConfig

Parameter	Required	Definition
		file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
Filter	Specifies the filter for the local file browser.
PreviewFormat	Specifies the preview data format.
AutoUpload	Specifies whether the local references should be automatically uploaded while saving the parent document to the server. Supported values are **false** and **true**.
OutputFormat	Specifies the output data format.
BuilderXSL	Specifies the transform to build the table element. You can specify an absolute or a relative path to the Quark XML Author installation directory
PreventAutoUploadChange	Specifies whether users should be prevented from changing the auto upload value. Supported values are **true** and **false**.
ExcelTableType	Specifies the type of Excel tables shown in the server browser.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Excel Charts

The user interface

The following element definitions need to be added in **./<LanguageFolder>/BUSDOCS/BUSDOCS.xas**

```
<!– - - - - - - - - - - - - - - - Excel Chart - - - - - - - - - - - - - - - - –>

<ElementDef name=”chart” friendly=”Chart” xmlname=”image”
visible=”false” excludeFromContextMenu=”true”
excludeFromChangeToContextMenu=”true”
externalMethodId=”ReferenceChart” externalMethodFriendly=”Chart
from Excel” referenceAttribute=”conref”>

  <Reference>
```

```
    <Media xmlname="image" required="true"
excludeFromComponentContextMenu="true">
      <Mimetypes>
        <Mimetype type="image/bmp"/>
        <Mimetype type="image/dib"/>
        <Mimetype type="image/emf"/>
        <Mimetype type="image/emz"/>
        <Mimetype type="image/eps"/>
        <Mimetype type="image/gif"/>
        <Mimetype type="image/jfif"/>
        <Mimetype type="image/jpe"/>
        <Mimetype type="image/jpeg"/>
        <Mimetype type="image/jpg"/>
        <Mimetype type="image/png"/>
        <Mimetype type="image/rle"/>
        <Mimetype type="image/tif"/>
        <Mimetype type="image/tiff"/>
        <Mimetype type="image/wmf"/>
        <Mimetype type="image/wmz"/>
      </Mimetypes>
    </Media>
  </Reference>
  <Attributes>
    <AttributeType name="href"/>
    <AttributeType name="width"/>
    <AttributeType name="height"/>
    <AttributeType name="widthdpi"/>
    <AttributeType name="heightdpi"/>
    <AttributeType name="scale"/>
    <AttributeType name="scalefit"/>
    <AttributeType name="imagedef" fixed="excelchart"/>
  </Attributes>
      &cmsextensibility;
</ElementDef>
```

Adding a button in the ribbon

The following changes need to be made to add an entry in the Ribbon. This will add the Chart from Excel button in the Insert tab.

For BUSDOCS: **./<Language Folder>/BUSDOCS/config.xml**

<button id="CMSChartFromExcel" size="large">

<InternalClass name="InsertElement" useExternalMethodId="true" displayName="Chart[image(*)]"/>

</button>

Adding the context menu entry:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

Add the following in the ElementDef of body:

<ReferenceType name="chart"/>

The Extensibility interface:

The following EI needs to be added in **./<Languagefolder>/Appconfig.xml**

```
<Method id="ReferenceChart"
assembly="Quark.CMSAdapters.Core.XA.ChartsAndTables"
method="ReferenceChart"
class="Quark.CMSAdapters.Core.XA.ChartsAndTables.ExtensibilityMethods">
  <Argument type="XomCurrentNode"/>
  <Argument type="XomPreviousNode"/>
  <Argument type="Tokens">

    <Token>Type=Excel Chart</Token>
    <Token>Filter=Excel Files|*.xlsx;*.xlsm;*.xlsb</Token>
    <Token>DefaultBrowseLocation=</Token>
    <Token>BuilderXSL=ChartBuilder.xslt</Token>
    <Token>OutputFormat=image/png</Token>
    <Token>PreviewFormat=image/png</Token>
    <Token>AutoUpload=false</Token>
    <Token>PreventAutoUploadChange=false</Token>
    <Token>ExcelChartType=Microsoft Excel Chart</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	Current Xom node.
XomPreviousNode	yes	Tag is a special

Parameter	Required	Definition
		Argument type that supplies the label of an item selected in the user interface.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
Filter	Specifies the filter for the local file browser.
PreviewFormat	Specifies the preview data format.
AutoUpload	Specifies whether the local references should be automatically uploaded while saving the parent document to the server. Supported values are **false** and **true**.
OutputFormat	Specifies the output data format.
BuilderXSL	Specifies the transform to build the table element. You can specify an absolute or a relative path to the Quark XML Author installation directory
PreventAutoUploadChange	Specifies whether users should be prevented from changing the auto upload value. Supported values are **true** and **false**.
ExcelTableType	Specifies the type of Excel tables shown in the server browser.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Excel Table holder and Chart holder

The following element definitions need to be added in **./<LanguageFolder>/BUSDOCS/BUSDOCS.xas**

```
<!– - - - - - - - - - - - - - - - Excel Table Holder - - - - - - -
- - - - - - - - - –>
```

```
 <ElementDef name="tableholder" friendly="Table Holder from
Excel" visible="false" referenceAttribute="conref"
excludeFromContextMenu="true"
excludeFromChangeToContextMenu="true"
externalMethodId="ReferenceTableHolder"
externalMethodFriendly="Table Holder from Excel">

  <Section>

    <Sequence minOccurs="0" maxOccurs="1">

      <SectionType name="title" default="[Title]" friendly="Table
Title" style="Caption (XA)"/>

    </Sequence>

    <Sequence minOccurs="0" maxOccurs="1">

      <SectionType name="desc" default="[Description]"
friendly="Table Description"/>

    </Sequence>

    <Sequence minOccurs="0" maxOccurs="unbounded">

      <SectionType name="note" friendly="Table Note"/>

    </Sequence>

    <Choice minOccurs="0" maxOccurs="1">

      <TableType name="ExcelTable"/>

    </Choice>

  </Section>

  <Attributes>

    <AttributeType name="id"/>

    <AttributeType name="conref"/>

  </Attributes>

       &cmsextensibility;

 </ElementDef>

<!— - - - - - - - - - - - - - - - Excel embedchart - - - - - - - -
- - - - - - - - —>

<ElementDef name="embedchart" friendly="Embedded Chart"
xmlname="image" visible="false">

  <Media xmlname="image" required="true"
excludeFromComponentContextMenu="true">

    <Mimetypes>

      <Mimetype type="image/bmp"/>

      <Mimetype type="image/dib"/>

      <Mimetype type="image/emf"/>
```

```
        <Mimetype type="image/emz"/>
        <Mimetype type="image/eps"/>
        <Mimetype type="image/gif"/>
        <Mimetype type="image/jfif"/>
        <Mimetype type="image/jpe"/>
        <Mimetype type="image/jpeg"/>
        <Mimetype type="image/jpg"/>
        <Mimetype type="image/png"/>
        <Mimetype type="image/rle"/>
        <Mimetype type="image/tif"/>
        <Mimetype type="image/tiff"/>
        <Mimetype type="image/wmf"/>
        <Mimetype type="image/wmz"/>
      </Mimetypes>
    </Media>
    <Attributes>
      <AttributeType name="filename"/>
      <AttributeType name="width"/>
      <AttributeType name="height"/>
      <AttributeType name="widthdpi"/>
      <AttributeType name="heightdpi"/>
      <AttributeType name="scale"/>
      <AttributeType name="scalefit"/>
      <AttributeType name="imagedef" fixed="excelchart"/>
    </Attributes>
  </ElementDef>
   <!— - - - - - - - - - - - - - - - Excel Chart Holder - - - - - - -
  - - - - - - - - - —>
   <ElementDef name="chartholder" friendly="Chart Holder from
  Excel" visible="false" referenceAttribute="conref"
  excludeFromContextMenu="true"
  excludeFromChangeToContextMenu="true"
  externalMethodId="ReferenceChartHolder"
  externalMethodFriendly="Chart Holder from Excel">
    <Section>
      <Sequence minOccurs="0" maxOccurs="1">
```

```
      <SectionType name="title" default="[Title]" friendly="Chart
Title" style="Caption (XA)"/>
    </Sequence>
    <Sequence minOccurs="0" maxOccurs="1">
      <SectionType name="desc" default="[Description]"
friendly="Chart Description"/>
    </Sequence>
    <Sequence minOccurs="0" maxOccurs="unbounded">
      <SectionType name="note" friendly="Chart Note"/>
    </Sequence>
    <Choice minOccurs="0" maxOccurs="1">
      <MediaType name="embedchart"/>
    </Choice>
  </Section>
  <Attributes>
    <AttributeType name="id"/>
    <AttributeType name="conref"/>
  </Attributes>
       &cmsextensibility;
 </ElementDef>
```

Adding the context menu entry:

For BUSDOCS: **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

Add the following in the ElementDef of body:

```
<ReferenceType name="tableholder"/>
<ReferenceType name="chartholder"/>
```

The Extensibility interface:

The following EI needs to be added in **./<Languagefolder>/Appconfig.xml**

```
<Method id="ReferenceTableHolder"
assembly="Quark.CMSAdapters.Core.XA.ChartsAndTables"
method="ReferenceRange"
class="Quark.CMSAdapters.Core.XA.ChartsAndTables.ExtensibilityMet
hods">
  <Argument type="XomCurrentNode"/>
  <Argument type="XomPreviousNode"/>
  <Argument type="Tokens">
    <Token>Type=Excel Document</Token>
```

```
    <Token>Filter=Excel Files|*.xlsx;*.xlsm;*.xlsb</Token>
    <Token>DefaultBrowseLocation=</Token>
    <Token>BuilderXSL=TableBuilder.xslt</Token>
    <Token>OutputFormat=application/xml-cals</Token>
    <Token>PreviewFormat=image/png</Token>
    <Token>AutoUpload=false</Token>
    <Token>PreventAutoUploadChange=false</Token>
    <Token>IsComplexElement=True</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	Current Xom node.
XomPreviousNode	yes	Tag is a special Argument type that supplies the label of an item selected in the user interface.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
Filter	Specifies the filter for the local file browser.
PreviewFormat	Specifies the preview data format.
AutoUpload	Specifies whether the local references should be automatically uploaded while saving the parent document to the server. Supported values are **false** and **true**.
OutputFormat	Specifies the output data format.
BuilderXSL	Specifies the transform to build the table element. You can specify an absolute or a relative path to the Quark XML Author installation directory
PreventAutoUploadChange	Specifies whether users should be prevented from changing the auto upload value. Supported values are

Token	Definition
	true and **false**.
ExcelTableType	Specifies the type of Excel tables shown in the server browser.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

```
<Method id="ReferenceChartHolder"
assembly="Quark.CMSAdapters.Core.XA.ChartsAndTables"
method="ReferenceChart"
class="Quark.CMSAdapters.Core.XA.ChartsAndTables.ExtensibilityMethods">
  <Argument type="XomCurrentNode"/>
  <Argument type="XomPreviousNode"/>
  <Argument type="Tokens">
    <Token>Type=Excel Document</Token>
    <Token>Filter=Excel Files|*.xlsx;*.xlsm;*.xlsb</Token>
    <Token>DefaultBrowseLocation=</Token>
    <Token>BuilderXSL=ChartBuilder.xslt</Token>
    <Token>OutputFormat=image/png</Token>
    <Token>PreviewFormat=image/png</Token>
    <Token>AutoUpload=false</Token>
    <Token>PreventAutoUploadChange=false</Token>
    <Token>IsComplexElement=True</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	Current Xom node.
XomPreviousNode	yes	Tag is a special Argument type that supplies the label of an item selected in the user interface.
Tokens	yes	A string array of tokens

Parameter	Required	Definition
		used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
Filter	Specifies the filter for the local file browser.
PreviewFormat	Specifies the preview data format.
AutoUpload	Specifies whether the local references should be automatically uploaded while saving the parent document to the server. Supported values are **false** and **true**.
OutputFormat	Specifies the output data format.
BuilderXSL	Specifies the transform to build the table element. You can specify an absolute or a relative path to the Quark XML Author installation directory
PreventAutoUploadChange	Specifies whether users should be prevented from changing the auto upload value. Supported values are **true** and **false**.
ExcelTableType	Specifies the type of Excel tables shown in the server browser.
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Adding PowerPoint slide references

PowerPoint slides can also be added to Quark XML Author documents just like Media references.

Adding a PowerPoint slide reference overview

The ReferenceSlide extensibility method is used to insert the selected PowerPoint slide reference.

The User Interface

The following changes need to be made to configure the User Interface of Quark XML Author for inserting a PowerPoint slide from server.

Adding the element definition for PowerPoint slides:

Add the following to **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

```
<ElementDef name="slide" friendly="Slide" xmlname="image"
visible="false" excludeFromContextMenu="true"
excludeFromChangeToContextMenu="true"
externalMethodId="ReferenceSlide" externalMethodFriendly="Slide
from PowerPoint" referenceAttribute="conref">
  <Reference>
    <Media xmlname="image" required="true"
excludeFromComponentContextMenu="true">
      <Mimetypes>
        <Mimetype type="image/bmp"/>
        <Mimetype type="image/dib"/>
        <Mimetype type="image/emf"/>
        <Mimetype type="image/emz"/>
        <Mimetype type="image/eps"/>
        <Mimetype type="image/gif"/>
        <Mimetype type="image/jfif"/>
        <Mimetype type="image/jpe"/>
        <Mimetype type="image/jpeg"/>
        <Mimetype type="image/jpg"/>
        <Mimetype type="image/png"/>
        <Mimetype type="image/rle"/>
        <Mimetype type="image/tif"/>
        <Mimetype type="image/tiff"/>
        <Mimetype type="image/wmf"/>
        <Mimetype type="image/wmz"/>
      </Mimetypes>
    </Media>
  </Reference>
  <Attributes>
    <AttributeType name="href"/>
    <AttributeType name="width"/>
    <AttributeType name="height"/>
    <AttributeType name="widthdpi"/>
    <AttributeType name="heightdpi"/>
    <AttributeType name="scale"/>
```

```
    <AttributeType name="scalefit"/>
    <AttributeType name="imagedef" fixed="powerpointslide"/>
  </Attributes>
     &cmsextensibility;
</ElementDef>
```

Adding the context menu entry:

Add the following ReferenceType in the ElementDef of body:

```
<ReferenceType name="slide"/>
```

Adding a button in the ribbon

The following changes need to be made to add an entry in the Ribbon. This will add the **Slide from PowerPoint** button in the **Insert** tab.

Add the following entry in **./<Language Folder>/BUSDOCS/config.xml** to the group **id="GroupIllustrations"**

```
<button id="CMSSlideFromPowerPoint" size="large">
  <InternalClass name="InsertElement" useExternalMethodId="true"
displayName="Slide[image(*)]"/>
</button>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="ReferenceSlide"
assembly="Quark.CMSAdapters.Core.XA.Publishing.QPP"
class="Quark.CMSAdapters.Core.XA.Publishing.QPP.ExtensibilityMeth
ods" method="ReferenceSlide" >
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=PowerPoint Document</Token>
          <Token>DefaultBrowseLocation=</Token>
          <Token>BuilderXSL=OfficeComponentBuilder.xslt</Token>
    <Token>OutputFormat=image/png</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tag	yes	Tag is a special Argument type that supplies the label of an item selected in the user

Parameter	Required	Definition
		interface..
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
BuilderXSL	Specifies the transform to build the element. You can specify an absolute or relative path to the Quark XML Author installation directory.
OutputFormat	Specifies output data format. E.g. OutputFormat=image/png
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Adding Visio drawings

Visio drawings can also be added to Quark XML Author documents just like Media references like images, video and audio.

Adding a Visio drawing reference overview

The ReferenceVisioPage extensibility method is used to insert the selected Visio page reference.

The User Interface

The following changes need to be made to configure the User Interface of Quark XML Author for inserting a Visio page from server.

Adding the element definition for Visio pages:

Add the following to **./<Language Folder>/BUSDOCS/BUSDOCS.xas**

```
<ElementDef name="visio" friendly="Visio Page" xmlname="image"
visible="false" excludeFromContextMenu="true"
excludeFromChangeToContextMenu="true"
externalMethodId="ReferenceVisioPage"
externalMethodFriendly="Page from Visio"
referenceAttribute="conref">

  <Reference>
```

```
    <Media xmlname="image" required="true"
excludeFromComponentContextMenu="true">
      <Mimetypes>
        <Mimetype type="image/bmp"/>
        <Mimetype type="image/dib"/>
        <Mimetype type="image/emf"/>
        <Mimetype type="image/emz"/>
        <Mimetype type="image/eps"/>
        <Mimetype type="image/gif"/>
        <Mimetype type="image/jfif"/>
        <Mimetype type="image/jpe"/>
        <Mimetype type="image/jpeg"/>
        <Mimetype type="image/jpg"/>
        <Mimetype type="image/png"/>
        <Mimetype type="image/rle"/>
        <Mimetype type="image/tif"/>
        <Mimetype type="image/tiff"/>
        <Mimetype type="image/wmf"/>
        <Mimetype type="image/wmz"/>
      </Mimetypes>
    </Media>
  </Reference>
  <Attributes>
    <AttributeType name="href"/>
    <AttributeType name="width"/>
    <AttributeType name="height"/>
    <AttributeType name="widthdpi"/>
    <AttributeType name="heightdpi"/>
    <AttributeType name="scale"/>
    <AttributeType name="scalefit"/>
    <AttributeType name="imagedef" fixed="visiopage"/>
  </Attributes>
      &cmsextensibility;
</ElementDef>
```

Adding the context menu entry:

Add the following ReferenceType in the ElementDef of body:

<ReferenceType name="visio"/>

Adding a button in the ribbon

The following changes need to be made to add an entry in the Ribbon. This will add the **Page from Visio** button in the **Insert** tab.

Add the following entry in./Language Folder>/BUSDOCS/config.xml to the **group id="GroupIllustrations"**

```
<button id="CMSPageFromVisio" size="large">
  <InternalClass name="InsertElement" useExternalMethodId="true"
displayName="Visio Page[image(*)]"/>
</button>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="ReferenceVisioPage"
assembly="Quark.CMSAdapters.Core.XA.Publishing.QPP"
class="Quark.CMSAdapters.Core.XA.Publishing.QPP.ExtensibilityMeth
ods" method="ReferenceVisioPage" >
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=Visio Document</Token>
    <Token>DefaultBrowseLocation=</Token>
    <Token>BuilderXSL=OfficeComponentBuilder.xslt</Token>
    <Token>OutputFormat=image/png</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
Tag	yes	Tag is a special Argument type that supplies the label of an item selected in the user interface.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
BuilderXSL	Specifies the transform to build the element. You can specify an absolute or relative path to the Quark XML Author installation directory.
OutputFormat	Specifies output data format. E.g. OutputFormat=image/png
DefaultBrowseLocation	String. Specifies the default browse location. This location is preselected on opening the server browser. In case the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path.

Changing references

References available in a document can be replaced by another server reference of the same type. The core Quark XML Author ChangeElement extensibility method has been extended to provide the capability to replace various reference types.

Changing a media reference

Media references in a Quark XML Author document can be changed with another server asset of the same content type. For example, an image referred in a document can be replaced with another image from the server.

Changing a video reference

The ChangeVideoElement extensibility interface is used to change referred videos.

The user interface

For BUSDOCS:

To add an entry in the context menu of the element that needs to be changed, add the following. For BUSDOCS, add to **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas** For DITA, add to./**<Languagefolder>/DITA/CMSExtensibilityDefinitions.xas**

```
<ExtensibilityMethod id="ChangeVideoElement" friendly="Change Reference" showXPath="self::*[(local-name()='video') and (string-length(@href) > 0 or string-length(@conref) > 0)]"/>
```

The Extensibility interface:

The following EI needs to be added in **./<Languagefolder>/Appconfig.xml**

```
<Method id="ChangeVideoElement"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement">
```

```
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=Video Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.

Changing an audio reference

The ChangeAudioElement extensibility interface is used to change referred audio files.

The user interface:

To add an entry in the context menu of the element that needs to be changed, add the following.For BUSDOCS, add to **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas** For DITA, add to**./<Languagefolder>/DITA/CMSExtensibilityDefinitions.xas**

```
  <ExtensibilityMethod id="ChangeAudioElement" friendly="Change
Reference" showXPath="self::*[(local-name()='audio') and (string-
length(@href) > 0 or string-length(@conref) > 0)]"/>
```

For DITA:

in **./<Languagefolder>/DITA/CMSExtensibilityDefinitions.xas**

The Extensibility interface:

The following EI needs to be added in **./<Languagefolder>/Appconfig.xml**

```
<Method id="ChangeAudioElement"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement">
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=Audio Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.

Changing an image reference

The ChangeImageElement extensibility interface is used to change referred images.

The user interface:

To add an entry in the context menu of the element that needs to be changed, add the following. For BUSDOCS, add to **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas** For DITA, add to**./<Languagefolder>/DITA/CMSExtensibilityDefinitions.xas**

```
<ExtensibilityMethod id="ChangeImageElement" friendly="Change Reference" showXPath="self::*[(local-name()='image') and (string-length(@href) > 0 or string-length(@conref) > 0)]"/>
```

The Extensibility interface:

The following EI needs to be added in **./<Languagefolder>/Appconfig.xml**

```
<Method id="ChangeImageElement"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement">

  <Argument type="XomCurrentNode"/>

  <Argument type="Tokens">

<Token>Type=Picture Reference</Token>

    <Token>DefaultBrowseLocation=</Token>

  </Argument>

</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.

Changing a Structured table reference

Structured table references in a Quark XML Author document can be changed with another server asset of the same content type.

Changing a Structured table reference

The ChangeStructuredTable extensibility interface is used to change referred structured tables.

The user interface:

To add an entry in the context menu of the element that needs to be changed, add the following in **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas**

```
<ExtensibilityMethod id="ChangeStructuredTable" friendly="Change
Reference" showXPath="self::*[(local-name()='CalsTable') and
(string-length(@href) > 0 or string-length(@conref) > 0)]"/>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="ChangeStructuredTable"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement">
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=Structured Table Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".

Token	Definition
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.

Changing a PowerPoint slide reference

PowerPoint slide references in a Quark XML Author document can be changed with another server asset of the same content type.

Changing a PowerPoint slide reference

The ChangeSlide extensibility interface is used to change referred PowerPoint slides.

The user interface:

To add an entry in the context menu of the element that needs to be changed, add the following in **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas**

```
<ExtensibilityMethod id="ChangeSlide" friendly="Change Reference"
showXPath="self::*[(local-name()='slide') and (string-
length(@href) > 0 or string-length(@conref) > 0)]"/>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="ChangeSlide" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement" >
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=PowerPoint Document</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig

Parameter	Required	Definition
		file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.

Changing a Visio drawing reference

Visio drawing references in a Quark XML Author document can be changed with another server asset of the same content type.

Changing a Visio drawing reference

The **ChangeVisioPage** extensibility interface is used to change referred Visio drawing.

The user interface:

To add an entry in the context menu of the element that needs to be changed, add the following in **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas**

```
<ExtensibilityMethod id="ChangeVisioPage" friendly="Change
Reference" showXPath="self::*[(local-name()='visio') and (string-
length(@href) > 0 or string-length(@conref) > 0)]"/>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="ChangeVisioPage" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement" >

  <Argument type="XomCurrentNode"/>

  <Argument type="Tokens">
```

```
    <Token>Type=Visio Document</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.

Changing Excel references

Excel references in a Quark XML Author document can be changed with another server asset of the same content type.

Changing an Excel chart reference

The ChangeChart extensibility interface is used to change referred Excel charts.

The user interface:

To add an entry in the context menu of the element that needs to be changed, add the following in **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas**

<ExtensibilityMethod id="ChangeChart" friendly="Change Reference" showXPath="self::*[(local-name()='chart') and (string-length(@href) > 0 or string-length(@conref) > 0)]"/>

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following

```
<Method id="ChangeChart" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement">
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=Excel Chart</Token>
    <Token>DefaultBrowseLocation=</Token>
    <Token>Filter=Excel Files|*.xlsx;*.xlsm;*.xlsb</Token>
    <Token>PreviewFormat=image/png</Token>
    <Token>AutoUpload=false</Token>
    <Token>PreventAutoUploadChange=false</Token>
    <Token>ExcelChartType=Microsoft Excel Chart</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.
Filter	Specifies the filter for the local file browser.
PreviewFormat	Specifies the preview data format.
AutoUpload	Specifies whether local references

Token	Definition
	should be auto uploaded while saving parent documents to the server. Values supported are **false** and **true.**
PreventAutoUploadChange	Specifies whether users should be allowed to change the auto upload value. Values supported are **false** and **true.**
ExcelChartType	Specifies the type of the Excel charts shown in the server browser.

Changing an Excel table reference

The ChangeTable extensibility interface is used to change referred Excel tables.

The user interface:

To add an entry in the context menu of the element that needs to be changed, add the following in **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas**

<ExtensibilityMethod id="ChangeTable" friendly="Change Reference" showXPath="self::*[((local-name()='ExcelTable') or (local-name()='tableimage')) and (string-length(@href) > 0 or string-length(@conref) > 0)]"/>

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="ChangeTable" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement">
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=Excel Table</Token>
    <Token>DefaultBrowseLocation=</Token>
    <Token>Filter=Excel Files|*.xlsx;*.xlsm;*.xlsb</Token>
    <Token>PreviewFormat=image/png</Token>
    <Token>AutoUpload=false</Token>
    <Token>PreventAutoUploadChange=false</Token>
    <Token>ExcelTableType=Microsoft Excel Table</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node

Parameter	Required	Definition
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
Type	Specifies the type of the documents to be shown in the server browser. Type value is the name of the ContentTypeMapping element defined in the Quark.CMSAdapters.config. You can specify multiple values using ",".
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser. If the default browse location is not specified, the server browser remembers the last browsed location. The browse location value must be as shown in the path field of the server browser.
Filter	Specifies the filter for the local file browser.
PreviewFormat	Specifies the preview data format.
AutoUpload	Specifies whether local references should be auto uploaded while saving parent documents to the server. Values supported are **false** and **true.**
PreventAutoUploadChange	Specifies whether users should be allowed to change the auto upload value. Values supported are **false** and **true.**
ExcelTableType	Specifies the type of the Excel tables shown in the server browser.

Changing document references

All server items included in a Quark XML Author document as section references can be changed with other assets on the server.

This feature is accomplished using the **ChangeElement** extensibility method.

The user interface:

For BUSDOCS, make the following changes to:
./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas.

For DITA, make the following changes to:
./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas.

```
<ExtensibilityMethod id="ChangeElement" friendly="Change
Reference" showXPath="self::*[not(local-name()='audio') and
      not(local-name()='video') and
      not(local-name()='image') and
      not(local-name()='chart') and
      not(local-name()='chartholder') and
      not(local-name()='ExcelTable') and
      not(local-name()='tableimage') and
      not(local-name()='tableholder') and
      not(local-name()='slide') and
      not(local-name()='visio') and
      not(local-name()='CalsTable') and
      ((string-length(@href) > 0 and not(contains(@href,'#'))) or
      (string-length(@conref) > 0 and
not(contains(@conref,'#'))))]"/>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="ChangeElement" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ChangeElement">
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>Type=XML Reference</Token>
    <Token>DefaultBrowseLocation=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The XOM node corresponding to the user's current selection.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Type	String. Specifies the type of the documents to be shown in the server browser. The value of type is the name of the ContentTypeMapping element defined in Quark.CMSAdapters.config. You can specify multiple values using "," as a separator.
DefaultBrowseLocation	Specifies the default browse location. This location is preselected on opening the server browser.

Edit Inline

The **Edit Inline** feature allows topics referred in a document as a child topic from the server to be edited inline.

Edit Inline overview

Edit inline is accomplished through the use of the **EditInline** Extensibility interface method.

The user interface:

Make the following changes to add an entry in the context menu of the element that needs to be changed:

For BUSDOCS, make the following changes to:
./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas.

For DITA, make the following changes to:
./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas.

```
<ExtensibilityMethod id="EditInline" friendly="Edit Component
(Inline)" showXPath="self::*[not(local-name()='audio') and

      not(local-name()='video') and

      not(local-name()='image') and

      not(local-name()='chart') and

      not(local-name()='chartholder') and

      not(local-name()='ExcelTable') and

      not(local-name()='tableimage') and

      not(local-name()='tableholder') and

      not(local-name()='slide') and

      not(local-name()='visio') and

      ((contains(@href,'qpp://') and
not(contains(@href,'majorversion')) and

      not(contains(@href,'#')) and
not(contains(@href,'isdatadoc=true'))) or

      (contains(@conref,'qpp://') and
not(contains(@conref,'majorversion')) and
```

```
      not(contains(@conref,'#')) and
not(contains(@conref,'isdatadoc=true')))]"/>
```

The Extensibility interface:

The following EI needs to be added in /Languagefolder/Appconfig.xml

```
 <Method id="EditInline" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="EditInline">

  <Argument type="XomCurrentNode"/>

 </Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node

Edit Component

Edit component facilitates opening referred components for editing in a new instance of Quark XML Author (Microsoft Word).

Edit component overview

Edit component is accomplished through the use of the **EditElement** Extensibility interface method.

The user interface:

Add the following to add an entry in the context menu of the element that needs to be changed:

For BUSDOCS, make the following changes to:
./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas.

For DITA, make the following changes to:
./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas.

```
<ExtensibilityMethod id="EditElement" friendly="Edit Component"
showXPath="self::*[not(local-name()='audio') and

      not(local-name()='video') and

      not(local-name()='image') and

                  not(local-name()='CalsTable') and

      ((contains(@href,'qpp://') and
not(contains(@href,'isdatadoc=true'))) or

      (contains(@conref,'qpp://') and
not(contains(@conref,'isdatadoc=true')))]"/>
```

The Extensibility interface:

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="EditElement" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="OpenElement">
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
    <Token>OpenReadOnly=false</Token>
  </Argument>
</Method>
```

Arguments

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
OpenReadOnly	Specifies whether the document has been opened in **Review** or **Revise** mode. Values supported are **True** or **False.**

Save Component changes

Save Component changes saves changes made to inline components using the SaveElement extensibility method.

The user interface:

For BUSDOCS, make the following changes to:

./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas.

For DITA, make the following changes to:
./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas.

```
<ExtensibilityMethod id="SaveElement" friendly="Save Changes"
howXPath="self::*[@cmsId]"/>
```

The extensibility interface:

Add the following to **./<Languagefolder>/Appconfig.xml**

```
<Method id="SaveElement" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="ExportElement">
  <Argument type="XomCurrentNode"/>
  <Argument type="Filename"/>
```

```
  <Argument type="Tokens">
    <Token>SaveSilently=false</Token>
    <Token>SaveAsMinorVersion=true</Token>
    <Token>xslt=</Token>
    <Token>ProductLine=busdoc</Token>
    <Token>KeepMoreOptionsExpanded=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The XOM node corresponding to the user's current selection.
Filename	yes	String. The document's full name, including path. Defaults to null for the current document.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
SaveSilently	Specifies whether the document is saved without displaying the **Save** dialog. Values supported are "**false**" (display the dialog) and "**true**" (do not display the dialog).
SaveAsMinorVersion	Specifies whether the document is saved as minor version. Values supported are "**false**" and "**true**".
xslt	Specifies the transform to be applied on exported XML.
ProductLine	Specifies the product line to be applied on the exported XML
KeepMoreOptionsExpanded	Specifies whether more options should be in expanded state by default

Save All components

This extensibility method will save all referred components that are open for inline editing.

The CMSSaveAll extensibility interface method is used to save all referred components.

The user interface:

Adding the Save all components button:

In **./<Languagefolder>/BUSDOCS/config.xml**, add the following in **TabHome/GroupEditing**

```
<button id="CMSSaveAll" keytip="IM" image="CMSCheckIn_image">
  <ExtensibilityMethod id="SaveAllComponents"
enableXPath="/node()//*[string-length(@cmsId) > 0]"/>
</button>
```

The extensibility interface

In **./<Languagefolder>/BUSDOCS/config.xml**, add the following

```
<Method id="SaveAllComponents"
assembly="Quark.CMSAdapters.Features.XA.QPP.Actions"
class="Quark.CMSAdapters.Features.XA.QPP.Actions.ExtensibilityMet
hods" method="ExportAllElements">
  <Argument type="XomRoot"/>
  <Argument type="Filename"/>
  <Argument type="Tokens">
    <Token>EvaluateRulesSilently=true</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomRoot	yes	The XOM node corresponding to the current document root
Filename		String. Specifies the name of a file to be closed rather than the currently active document.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
EvaluateRulesSilently	Specifies whether rule evaluation is done silently. Values supported are **false** (with dialog) and **true** (without dialog). The default value is true.

Discard Component changes

Discard component changes discards changes made to inline components using the CancelCheckoutElement.extensibility interface method.

The user interface:

For BUSDOCS, make the following changes to:
./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas.

For DITA, make the following changes to:
./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas.

```
<ExtensibilityMethod id="CancelCheckoutElement" friendly="Discard
Changes" showXPath="self::*[@cmsId]"/>

</Extensibility definition>
```

The extensibility interface

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="CancelCheckoutElement"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="CancelCheckoutElement">

  <Argument type="XomCurrentNode"/>

  <Argument type="Tokens">

    <Token>CancelCheckoutSilently=false</Token>

    <Token>Recursive=false</Token>

  </Argument>

</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The XOM node corresponding to the user's current selection.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
Recursive	Specifies whether this method should Cancel the checkout of all nested inline references along with the selected element.
CancelCheckoutSilently	specifies whether Cancel Checkout is done without displaying the confirmation dialog. Values supported are **false** (with dialog) and **true** (without dialog).

Discard all component changes

The CMSRevertAll extensibility method is used to discard all referred components changes done after editing the component inline.

The user interface:

Adding the Discard all components button:

In **./<Languagefolder>/BUSDOCS/config.xml**, add the following in **TabHome/GroupEditing**

```
<button id="CMSRevertAll" keytip="IM" image="CMSCancelCheckout_image">
    <ExtensibilityMethod id="RevertAllComponents"
enableXPath="/node()//*[string-length(@cmsId) > 0]"/>
</button>
```

The extensibility interface

In **./<Languagefolder>/BUSDOCS/config.xml**, add the following:

```
<Method id="RevertAllComponents" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="CancelCheckoutAllElements">
    <Argument type="XomRoot"/>
    <Argument type="Tokens">
        <Token>CancelCheckoutSilently=false</Token>
    </Argument>
</Method>
```

Parameter	Required	Definition
XomRoot	yes	The XOM node corresponding to the current document root.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
CancelCheckoutSilently	Specifies whether the cancel checkout dialog should be displayed. Values supported are **true** and **false**.

Open as Read-only

This option opens a read-only version of the selected Quark XML Author component in a new window.

Open as Read-only overview

The OpenElement extensibility method is used to enable the Open as Read-only functionality.

The user Interface:

The following changes need to be made to configure the user interface of Quark XML Author:

For BUSDOCS, make the entry in **./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas.** For DITA, make the entry in **./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas.**

```
  <ExtensibilityMethod id="OpenElement" friendly="Open as Read
Only" showXPath="self::*[not(local-name()='CalsTable') and
(contains(@href,'qpp://') or contains(@conref,'qpp://'))]"/>
```

The Extensibility interface

In **./<Languagefolder>/Appconfig.xml**, add the following:

```
<Method id="OpenElement" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="OpenElement">

  <Argument type="XomCurrentNode"/>

  <Argument type="Tokens">

    <Token>OpenReadOnly=true</Token>

  </Argument>

</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node.
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
OpenReadOnly	This specifies whether the document is to be opened in "Review" or "Revise" mode. The supported values are **false** (Revise Mode) and **true** (Review Mode).

Convert to local reference

Server references in documents opened from the server can be converted into local references. Further actions supported for local references can be performed on such elements.

Convert to local reference overview

Convert to local reference is accomplished through the use of the **ConvertToLocalReference** Extensibility interface method.

The User Interface:

To add an entry in the context menu of the element that needs to be converted, add the following. For BUSDOCS, make the entry in **./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas**. For DITA, make the entry in **./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas**.

```
  <ExtensibilityMethod id="ConvertToLocalReference"
friendly="Convert to Local Reference"
showXPath="self::*[not(local-name()='audio') and

      not(local-name()='video') and

      not(local-name()='chart') and

      not(local-name()='chartholder') and

      not(local-name()='ExcelTable') and

      not(local-name()='tableimage') and

      not(local-name()='tableholder') and

      not(local-name()='slide') and

      not(local-name()='visio') and

      (contains(@href,'qpp://') or
contains(@conref,'qpp://'))]"/>
```

The Extensibility interface:

The following EI needs to be added in **./<Languagefolder>/Appconfig.xml**

```
 <Method id="ConvertToLocalReference"
assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="DetachElement">

         <Argument type="XomCurrentNode"/>

         <Argument type="Tokens">

  <Token>DetachAction=DOWNLOAD</Token>

  <Token>xslt=</Token>

         </Argument>

</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	Xom Node
Tokens	yes	A string array of tokens used in the AppConfig file

Tokens

Token	Definition
DetachAction	Specifies the detach action to be applied on the element. Supported values are:
XSLT	Specifies the transform to be applied on the downloaded XML in case of an inline action.

Edit Original

Edit Original facilitates opening the Microsoft Excel document from which the referred component has been exported.

The Edit Original functionality is achieved using the EditOriginal extensibility interface.

The user interface:

To add an entry in the context menu of the element that needs to be edited, add the following in **./<Languagefolder>/BUSDOCS/CMSExtensibilityDefinitions.xas**

```
<ExtensibilityMethod id="EditOriginal" friendly="Edit Original"
showXPath="self::*[((local-name()='image') or
      (local-name()='CalsTable') or
      (local-name()='chart') or
      (local-name()='chartholder') or
      (local-name()='ExcelTable') or
      (local-name()='tableimage') or
      (local-name()='tableholder')) and
      ((contains(@href,'qpp://') and
contains(@href,'isexportedfromlocalfile=true')) or
      (contains(@conref,'qpp://') and
contains(@conref,'isexportedfromlocalfile=true')))]"/>
```

The EditOriginal Extensibility interface

The InitEditOriginalFeature EI is added in the connect event to initialize the assembly Quark.CMSAdapters.Features.XA.QPP.EditOriginal.dll when Quark XML Author is launched.

In **./<Languagefolder>/AppConfig.xml** add the following in **<ExtensibilityInterface>** in **<Connect>:**

```
<ExtensibilityMethod id="InitEditOriginalFeature"/>
```

In **./<Languagefolder>/AppConfig.xml** add the following EIs in **<ExtensibilityInterface>**

```
<Method id="InitEditOriginalFeature"
assembly="Quark.CMSAdapters.Features.XA.QPP.EditOriginal"
class="Quark.CMSAdapters.Features.XA.QPP.EditOriginal.Extensibili
tyMethods" method="Init">
  <Argument type="Tokens">
    <Token>
            ShowXPath=self::*[((local-name()='image') or
            (local-name()='CalsTable') or
            (local-name()='chart') or
            (local-name()='chartholder') or
            (local-name()='ExcelTable') or
            (local-name()='tableimage') or
            (local-name()='tableholder')) and
            ((contains(@href,'qpp://') and
contains(@href,'isexportedfromlocalfile=true')) or
            (contains(@conref,'qpp://') and
contains(@conref,'isexportedfromlocalfile=true')))]
    </Token>
  </Argument>
</Method>
```

```
<Method id="EditOriginal"
assembly="Quark.CMSAdapters.Features.XA.QPP.EditOriginal"
class="Quark.CMSAdapters.Features.XA.QPP.EditOriginal.Extensibili
tyMethods" method="EditOriginal">
  <Argument type="XomCurrentNode"/>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node

Pin and Unpin

Image elements referred in a Quark XML Author document will usually get updated once the document is updated along with image references. The **Pin** feature allows image references to get pinned to the document so that the pinned version of the image remains associated with the document until the image is **unpinned** and updated.

Pin and Unpin overview

Both Pin and Unpin are accomplished through the use of context menu options.

When an element is pinned the current version of the referred asset is associated with the document on check-in.

The Pin and Unpin user interface

The contextual menu entries for Pin and Unpin are governed by entries made to the following folders. For BUSDOCS, make the entry in **./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas.**

For DITA, make the entry in **./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas.**

Context menu entry for Pin

```
<ExtensibilityMethod id="PinElement" friendly="Pin Reference"
showXPath="self::*[(contains(@href,'qpp://') and
not(contains(@href,'majorversion'))) or
(contains(@conref,'qpp://') and
not(contains(@conref,'majorversion')))]"/>
```

Context menu entry for Unpin

```
<ExtensibilityMethod id="UnpinElement" friendly="Unpin Reference"
showXPath="self::*[(contains(@href,'qpp://') and
contains(@href,'majorversion')) or (contains(@conref,'qpp://')
and contains(@conref,'majorversion'))]"/>
```

The Pin and Unpin Extensibility interface

In **./<Language Folder>/AppConfig.xml** add the following EI in <ExtensibilityInterface>

```
<Method id="PinElement" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="PinElement">

        <Argument type="XomCurrentNode"/>

</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The XOM node corresponding to the user's current selection.

```
 <Method id="UnpinElement" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="UnpinElement">

        <Argument type="XomCurrentNode"/>

 </Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The XOM node corresponding

Parameter	Required	Definition
		to the user's current selection.

Make Content Inline

All content referred in a Quark XML Author document can be converted into inline content. This content is now a part of the document.

MakeInline overview

The Make Content Inline feature is accomplished through the **MakeInline** Extensibility interface method.

The user interface:

To configure the user interface for **MakeInline**, changes need to be made to the following folders. For BUSDOCS, make the entry in **./<LanguageFolder>/BUSDOCS/CmsExtensibilityDefintions.xas**. For DITA, make the entry in **./<LanguageFolder>/DITA/CmsExtensibilityDefintions.xas.**

```
<ExtensibilityMethod id="MakeInline" friendly="Make Content
Inline" showXPath="self::*[not(local-name()='audio') and
      not(local-name()='video') and
      not(local-name()='image') and
      not(local-name()='chart') and
      not(local-name()='chartholder') and
      not(local-name()='ExcelTable') and
      not(local-name()='tableimage') and
      not(local-name()='tableholder') and
      not(local-name()='slide') and
      not(local-name()='visio') and
      (contains(@href,'qpp://') or
contains(@conref,'qpp://'))]"/>
```

The MakeInline Extensibility interface

In **./<Language Folder>/AppConfig.xml** add the following:

```
 <Method id="MakeInline" assembly="Quark.CMSAdapters.Core.XA"
class="Quark.CMSAdapters.Core.XA.ExtensibilityMethods"
method="DetachElement">
  <Argument type="XomCurrentNode"/>
  <Argument type="Tokens">
```

```
    <Token>DetachAction=INLINE</Token>
    <Token>xslt=</Token>
  </Argument>
</Method>
```

Parameter	Required	Definition
XomCurrentNode	yes	The current Xom node
Tokens	yes	A string array of tokens used in the AppConfig file.

Tokens

Token	Definition
DetachAction	Specifies the detach action. The following values are supported: INLINE - Deletes the reference but retains the content. DOWNLOAD - Converts the server reference to a local reference.
xslt	Specifies the transform to be applied on the downloaded XML in case of INLINE action.

Changelog

Configuration changes

AppConfig.xml

Method id=”ChangeChart” *New

Added a new method to Change Reference for Microsoft Excel elements, with the following tokens:

@Type - specifies the type of the documents to be shown in the server browser, set to “Excel Chart”

@DefaultBrowseLocation - specifies the default browse location, set to empty

@Filter - specifies filter for the local file browser, set to “Excel Files|*.xlsx;*.xlsm;*.xlsb”

@PreviewFormat - specifies preview data format, set to “image/png”

@AutoUpload - specifies whether local references should auto upload while saving parent document to the server, set to false

@PreventAutoUploadChange - specifies whether UI should prevent users to change auto upload value, set to false

@ExcelChartType - specifies the type of the Excel charts shown in the server browser, set to “Microsoft Excel Chart”

Method id=”ChangeTable” *New

Added a new method to Change Reference for Microsoft Table elements, with the following tokens:

@Type - specifies the type of the documents to be shown in the server browser, set to “Excel Table”

@DefaultBrowseLocation - specifies the default browse location, set to empty

@Filter - specifies filter for the local file browser, set to “Excel Files|*.xlsx;*.xlsm;*.xlsb”

@PreviewFormat - specifies preview data format, set to “image/png”

@AutoUpload - specifies whether local references should auto upload while saving parent document to the server, set to false

@PreventAutoUploadChange - specifies whether UI should prevent users to change auto upload value, set to false

@ExcelChartType - specifies the type of the Excel charts shown in the server browser, set to “Microsoft Excel Table”

6

Method id=”ChangeSlide” *New

Added a new method to Change Reference for PowerPoint slide elements, with the following tokens:

@Type - specifies the type of the documents to be shown in the server browser, set to “PowerPoint Document”

@DefaultBrowseLocation - specifies the default browse location, set to empty

Method id=”ChangeVideoElement” *New

Added a new method to Change Reference for video elements, with the following tokens:

@Type - specifies the type of the documents to be shown in the server browser, set to “Video Reference”

@DefaultBrowseLocation - specifies the default browse location, set to empty

Method id=”ChangeAudioElement” *New

Added a new method to Change Reference for audio elements, with the following tokens:

@Type - specifies the type of the documents to be shown in the server browser, set to “Audio Reference”

@DefaultBrowseLocation - specifies the default browse location, set to empty

Method id=”ChangeImageElement” *New

Added a new method to Change Reference for image elements, with the following tokens:

@Type - specifies the type of the documents to be shown in the server browser, set to “Picture Reference”

@DefaultBrowseLocation - specifies the default browse location, set to empty

Method id=”ChangeElement” *New

Added a new method to Change Reference for section elements, with the following tokens:

@Type - specifies the type of the documents to be shown in the server browser, set to “XML Reference”

@DefaultBrowseLocation - specifies the default browse location, set to empty

Method id=”ChangeStructuredTable” *New

Added a new method to Change Reference for table elements, with the following tokens:

@Type - specifies the type of the documents to be shown in the server browser, set to “Structured Table Reference”

@DefaultBrowseLocation - specifies the default browse location, set to empty

Method id=”ChangeVisioPage” *New

Added a new method to Change Reference for Microsoft Visio diagram elements, with the following tokens:

7

@Type - specifies the type of the documents to be shown in the server browser, set to “Visio Document”

@DefaultBrowseLocation - specifies the default browse location, set to empty

Method id=”ReferenceChart” *Updated

Added a new token ExcelChartType to specify the type of the Excel charts shown in the server browser.

@ExcelChartType: “Microsoft Excel Chart”

Changed the token Type to Excel Chart instead of generic Excel Document

@Type=Excel Chart

Method id=”ReferenceTable” *Updated

Added a new token ExcelTableType to specify the type of the Excel charts shown in the server browser.

@ExcelTableType: “Microsoft Excel Table”

Changed the token Type to Excel Table instead of generic Excel Document

@Type=Excel Table

Method id=”ReferenceTableImage” *Updated

Added a new token ExcelChartType to specify the type of the Excel charts shown in the server browser.

@ExcelTableType: “Microsoft Excel Table”

Changed the token Type to Excel Table instead of generic Excel Document

@Type=Excel Table

Method id=”InitEditOriginalFeature” *Updated

Modified the token ShowXPath to handle the new URN attribute isexportedfromlocalfile

config.xml

Method/@id=”PasteFromServer” *New

Added a new Method to Paste from Server with the following tokens:

@ExcelChartType: “Microsoft Excel Chart”

@ChartBuilderXSL: “ChartBuilder.xslt”

@ChartOutputFormat: “image/png”

@ExcelTableType: “Microsoft Excel Table”

@TableOutputFormat: “application/xml-cals”

button/@id="CMSPasteFromServer" *New

Added a new button to Paste from Server, when the accessMode is "Revise, Author"

8

button/@id="CMSTableFromExcel" *Updated

Changed the displayName

@displayName: Changed to "Table from Excel(~)"

Quark.CMSAdapters.config

key="EnableTaskPaneViewHibernation" *New

Defines whether task pane views should hibernate when task pane is not visible. Hibernation allows views to partially clean up resources when task pane is in hidden state. This key is applicable to views additionally implementing Quark.CMSAdapters.Core.Interfaces.IHibernatingView.

Values supported are 0 (NO) and 1 (YES). The default value is 0 (NO)

ContentTypeMapping /@name="Excel Chart" *New

Added new ContentTypeMapping for Microsoft Excel Chart references, with the following configuration:

@name="Excel Chart"

@type="OTHER"

@class="Microsoft Excel, Microsoft Excel Chart"

@filter="includeChildContentTypes=false;File extension=xlsx,xlsm,xlsb"

ContentTypeMapping /@name="Excel Table" *New

Added new ContentTypeMapping for Microsoft Excel Table references, with the following configuration:

@name="Excel Table"

@type="OTHER"

@class="Microsoft Excel, Microsoft Excel Table"

@filter="includeChildContentTypes=false;File extension=xlsx,xlsm,xlsb"

ContentTypeMapping /@name="PowerPoint Document" *Updated

Added new ContentTypeMapping for PowerPoint Document references, with changes to the following configuration. Added support for .pptm files

@filter="includeChildContentTypes=true;File extension=pptx,pptm"

ConnectionManagement section *New

Replaced the existing ConnectionSettings section to now support multiple connection settings configuration. This section can contain multiple ConnectionInfo configurations. Also, see below

ConnectionInfo section *New

This section defines various connection parameters to connect to the Quark Publishing Platform server.

@Name - specifies the connection name

@serverName - specifies the server machine name.

9

@port - specifies the port number.

@useHttps - specifies whether https protocol is used for the communication. Values supported are “true” and “false”. The default value is “false”

ConnectionSettings section *Removed

See above

BUSDOCS.xas

ElementDef/@name=”CalsTable” *Updated

Table elements have grossly been simplified in the out-of-the-box configuration. While the configuration removes the surfacing of these elements on the canvas, the user experience is aimed to be simpler. Hence the widely-used table element CalsTable is retained used in the config.

@friendly: Simplified and changed to “Table”

@externalMethodFriendly: Simplified and changed to “Table from server”

Language translations are updated correspondingly.

ElementDef/@name=”ExcelTable” *Updated

@friendly: Changed to “Table from Excel”

Language translations are updated correspondingly.

ElementDef/@name=”htmlcelltext” *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name=”HtmlHeadingCell” *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name=”HtmlBodyCell” *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name=”HtmlBodyRow” *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name=”HtmlColumn” *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name=”HtmlHeading” *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name="HtmlBody" *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

10

ElementDef/@name="HtmlTable" *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name="stentry" *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name="sthead" *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name="strow" *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name="simpletable" *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name="OLEWordDocument" *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

ElementDef/@name="WordTable" *Removed

Commented it out from the out-of-the-box configuration, support still provided in Quark XML Author

11

CmsExtensibilityDefintions.xas

ExtensibilityMethod id="EditOriginal" *Updated

@showXPath: Updated the conditions when the EditOriginal command can be shown, since for tables where edit original can be shown, a new URN parameter for conref and href values, isexportedfromlocalfile will be set to true

ExtensibilityMethod id="SaveElement" *Updated

@friendly: Changed to "Save Changes"

Language translations are updated correspondingly.

ExtensibilityMethod id="CancelCheckoutElement" *Updated

@friendly: Changed to "Discard Changes"

Language translations are updated correspondingly.

ExtensibilityMethod id=”PinElement” *Updated

@friendly: Changed to “Pin Reference”

Language translations are updated correspondingly.

ExtensibilityMethod id=”UnpinElement” *Updated

@friendly: Changed to “Unpin Reference”

Language translations are updated correspondingly.

ExtensibilityMethod id=”ChangeAudioElement” *New

Added new ExtensibilityMethod to provide options to Change Reference for audio elements

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeAudioElement” *New

Added new ExtensibilityMethod to provide options to Change Reference for audio elements

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeVideoElement” *New

Added new ExtensibilityMethod to provide options to Change Reference for video elements

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeImageElement” *New

Added new ExtensibilityMethod to provide options to Change Reference for image elements

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeStructuredTable” *New

Added new ExtensibilityMethod to provide options to Change Reference for table elements

12

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeChart” *New

Added new ExtensibilityMethod to provide options to Change Reference for Excel chart elements

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeTable” *New

Added new ExtensibilityMethod to provide options to Change Reference for Excel table elements

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeSlide” *New

Added new ExtensibilityMethod to provide options to Change Reference for PowerPoint slide elements

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeVisioPage” *New

Added new ExtensibilityMethod to provide options to Change Reference for Visio diagram elements

Language translations have been added correspondingly.

ExtensibilityMethod id=”ChangeElement” *New

Added new ExtensibilityMethod to provide options to Change Reference for all other section type elements

Language translations have been added correspondingly.

MapExtensibilityDefintions.xas

ExtensibilityMethod id=”PinElement” *Updated

@friendly: Changed to “Pin Reference”

Language translations are updated correspondingly.

ExtensibilityMethod id=”UnpinElement” *Updated

@friendly: Changed to “Unpin Reference”

Language translations are updated correspondingly.

Contacting Quark

The support portal allows you tolog support tickets, track tickets, receive status notifications,chat with a technical support representative, search the Knowledge Base, and access product documentation.

With direct access to documentation acrossall Quark software - from QuarkXPress and App Studio to QuarkEnterprise Solutions - you can find answers to your questions atyour convenience. Our support team is also available to help,either through our support portal, or via phone for our maintenancecontract customers.

If you are a Quark customer and have a currentmaintenance or support contract your account has already been created for youusing your registered email address. If you do not have a supportcontract you can purchase a single support incident to get yourproblem resolved. If you have purchased or registered a supportedproduct, you are eligible for free support for the first 90days.

In the Americas

For more details, please check out our support website.

Outside the Americas

For countries outside the Americas, please visit the following sites to access your support account:

Support Website

- France - *www.quark.com/fr/support*
- Germany - *www.quark.com/de/support*

Legal notices

©2018 Quark Software Inc. and itslicensors. All rights reserved.

Quark, the Quark logo, and Quark XML Authorare trademarks or registered trademarks of Quark Software Inc. and itsaffiliates in the U.S. and/or other countries. All other marks arethe property of their respective owners.

Contact Information:

Denver Corporate Address: 1225 17th Street Suite 2050 Denver, CO 80202