eHF-Document

Webservice-Documentation

Webservice-Documentation | Version 1.0 | Status Approved
Security Level Public

Imprint

InterComponentWare AG
Industriestraße 41
69190 Walldorf, Germany

Tel.: +49 (0) 6227 385 0
Fax: +49 (0) 6227 385 199

Document Version: 1.0

Document Language: en (US)

Security Level: Public

Last change on: 10/14/2008

Editorial Staff: Sven Geier

Table of Contents

1 Introduction

Document Conventions

Mark up	Explanation	Example
italic	Windows, Dialogs, Sections, Forms	Open the window:: *Options*
bold	Buttons, Fields, Tabs, Check boxes, Options, Drop-down-Menus	The Button **next** lets you navigate through the wizard.
bold and underlined	Link	Click on **http://www.java.com**. The website will be opened.
bold and italic	Emphasis	You will find these information ***only*** on the vendors page.
	Warning	**Warning**: Save your inputs!
	Notice	**Note**: This view shows only data according to the installation process.
	Example	**Example:**
	Path	**Path**: Click File > save as
	Code	**Code:** `chmod –R g+wrx /home/esb` `# chgrp –R icw /home/esb`
<<Variable>>	Variable	The installation directory will be shown << INSTALL DIRECTORY>>
[strg+alt]	Shortcut	

Table 1: Document Conventions

2 Document Web Service

Terminology

This section briefly describes the central terms of the Web Service along with the client classes which provide the client model for the Web Service API.

Document

A document is a file containing arbitrary content. The content can be of binary nature (e.g. pictures) or may be textual (e.g. plain text or xml files). For each document metadata such as a document-ID, MIME-Type, author, version number or creation date is stored. Updating a document's content will result in a new document version. Furthermore a document may have child documents and each of these may have further child documents and so forth.

Style Sheets

A Style sheet is a special kind of a document which is associated with an XML based document such as a CDA (Clinical Document Architecture). Style sheets are provided by the document module. The system retrieving the document is also able to retrieve the corresponding style sheets (one for displaying and one for printing), if available.

Client Classes

`AbstractContext`

Metadata that is relevant for `DocumentContext` and `StyleSheetContext` is stored in an AbstractContext.

`DocumentContext`

The `DocumentContext` class contains the document's metadata. The content of the document is stored via the `DocumentContentContext class`, which subclasses `DocumentContext`.

`StyleSheetContext`

As with the `DocumentContext` the `StyleSheetContext` class contains the style sheet's metadata and its subclass `StyleSheetContentContext` contains the style sheet's content.

`DocumentQualifier`

Document qualifiers are document IDs used for document identification. There are three different qualifiers available which are subclasses of the `DocumentQualifier` class: `InstanceQualifier`, `AliasQualifier`, `GlobalIdQualifier`. For further description of these identifiers see Section 0.

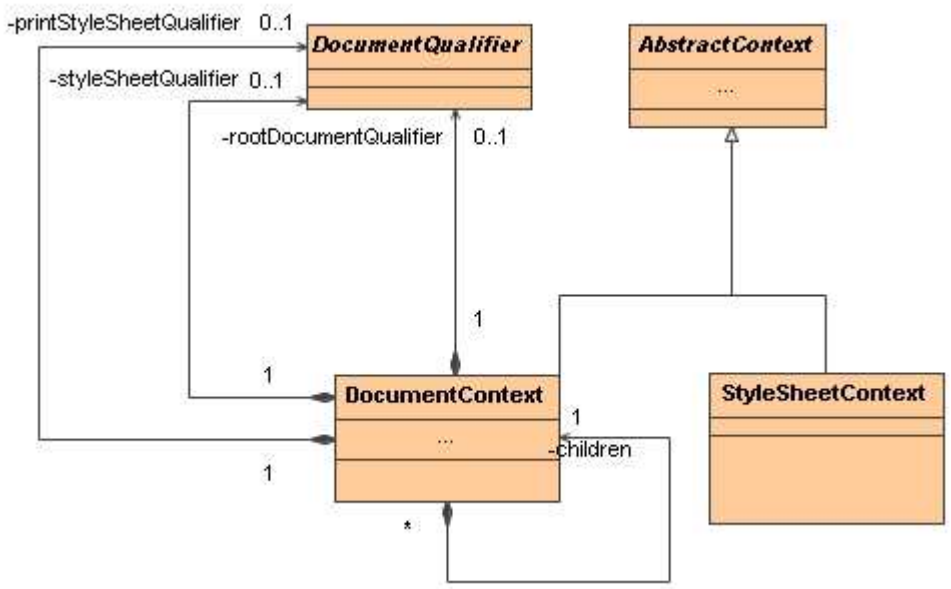

Figure 1: Major Client Classes of the Document Module.

Overview

Web service Description:

The Document Web Service allows uploading, downloading and deleting of documents with arbitrary content. Supported are both binary documents and structured text documents like XML files. The latter ones may be associated with stylesheets, which can be used in XSL transformations to render the XML. The central data types of the Document Web Service are the `DocumentContextXto` and the derived `DocumentContentContextXto`. The first one contains the document's metadata (e.g. author, document version, confidentiality, etc.) and the latter one the actual document content in binary or text form.

Usage Notes:

Here you can see how an upload is done:

Code:

```
File inputFile = new File("path_to_some_file");

DataHandler inputDataHandler = new DataHandler(new File-
DataSource(inputFile));

DocumentContentContextXto inputTransfer = new DocumentContentCon-
textXto();

... set metadata like mime type, author, confidentiality, language,
alias, document type ...

inputTransfer.setDataHandler(inputDataHandler);

// perform the upload

DocumentContextXto result = documentModuleServiceXtoAdapter
.uploadDocument(inputTransfer);

// and get the creates document qualifier from the result

qualifier = result.fetchQualifier();
```

An example for downloading a document you can see here:

```
DocumentContentContextXto downloadTransfer =
documentModuleServiceXtoAdapter. downloadDocument(qualifier, false);
```

Most important methods:

The most important methods are `uploadDocument` and `downloadDocument` to up- and download documents. Others are `loadAllDocumentsByScope` and `loadAll-DocumentsByDocumentTypes`.

Main method parameters:

Parameter	**Description**
`Impl:DocumentQualifierXto`	Abstract superclass for all qualifier types used in the document module. Document qualifiers are used to identify individual documents inside the system. Extending classes can implement their own way of identification.
Impl:InstanceIdentifierQualifierXto	This document qualifier uses an InstanceIdentifierXto to identify the document.
Impl:GlobalIdQualifierXto	The `GlobalIdQualifierXto` is a document qualifier that identifies a document by its individual id which has been assigned to the document by the system.
Impl:AliasQualifierXto	This is a document qualifier that allows a client to identify a document by custom ids the client has provided on creation of the document. Both alias and provider can be custom specified by the client. Only the scope must map to an actual scope in the system.
Impl:AbstractContextXto	Abstract super class for the central domain objects in the Document Web Service. It combines common attributes like author, version and others.
Impl:DocumentContextXto	Represents metadata for a concrete document. Together with the AbstractContext this class delivers the full set of metadata.
Impl:DocumentContentContextXto	Extends the metadata in DocumentContext with the content of document. The file gets handled by the embedded DataHandler.

Impl:StyleSheetContextXto	The StyleSheetContext represents the metadata of a stylesheet. It is used for managing the metadata of stylesheet documents. Stylesheet documents are special documents that live in the context of XML documents. If a XML document is uploaded you can also upload a custom XSL stylesheet which is responsible for the rendering of the uploaded document. You associate this stylesheet by setting a corresponding styleSheetQualifier and printStyleSheetQualifier on the DocumentContextXto of the XML stylesheet. If a client downloads the XML he can also download the associated stylesheet before rendering.
Impl:StyleSheetContentContextXto	This class extends the StyleSheetContext with the actual stylesheet conten

2.1.1 Impl: DocumentQualifierXto

This class does not have any attributes.

2.1.2 Impl:InstanceIdentifierQualifierXto

Attribute	**Description**	**Type**	**Range**
instanceIdentifier	An instanceIdentifier consists of a root OID plus extension. The OID is assigned to the system that has created the document while the extension is a unique Id which was created by the system. Together the OID plus extension form a globally unique identifier for a document	impl:InstanceIdentifierXto	

2.1.3 Impl: GlobalIdQualifierXto

Attribute	**Description**	**Type**	**Range**
globalId	You should set here the global id applied by system on upload. Setting a wrong id here will lead to the document not being found.	xsd:String	Up to 64 characters

2.1.4 Impl: AliasQualifierXto

Attribute	**Description**	**Type**	**Range**
scope	The scope of the subject this document is associated with.	xsd:String	Up to 128 characters
alias	The alias is usually the file name which the document has on the system of the user. But can be any other string, too.	xsd:String	Up to 512 characters
provider	The provider is a string or id which can be supplied by the client. So for example you can specify here the id which is used in the client system to be able to associate it with the data on client side.	xsd:String	Up to 256 characters

2.1.5 Impl: AbstractContextXto

Attribute	**Description**	**Type**	**Range**
documentIdentifier	Defines the instanceIdentifier for this document.	impl:InstanceIdentifierXto	

alias	The alias is usually the file name which the document has on the system of the user. But can be any other String, too. It is usually provided by the client.	xsd:String	Up to 512 characters
commentary	Free text to make comments on the document.	xsd:String	Up to 1024 characters
scope	Scope of the user for whom this document is.	xsd:String	Up to 128 characters
provider	The provider is a string or id which can be supplied by the client. So for example you can specify here the id which is used in the client system to be able to associate it with the data on client side. If this attribute is not specified it gets automatically an identifier set by the system. This attribute is used in the `AliasQualifierXto` to identify a document.	xsd:String	Up to 256 characters
globalId	The global id is provided by the system. Data submitted via web service is ignored here.	xsd:String	Up to 64 characters
author	The author is the person that is responsible for the content of this document.	impl:ParticipantXto	

dataEnterer	The data enterer is the person who actually entered the data into the system. Often the author and the data enterer are equal, but not always. For example the data can be entered by a secretary that is filling in dictated text. Then the author is a different person and the secretary is the data enterer.	impl:ParticipantXto	
contentSize	Size of the content is given in bytes and has to be calculated separately.	xsd:long	
description	The description usually is a short summary of the content of the document and / or its purpose.	xsd:String	Up to 1024 characters
documentVersion	This is usually done internally by the Web Service. The version represents an identifiable state of the document over time.	xsd:int	
encoding	Defines the character set in which the document is encoded (e.g. UTF8)	impl:CodeXto	`Code Set: EXT-GEN-DOCUMENT-CHARSET`
confidentiality	Defines the confidentiality of the document	impl:CodeXto	Code Set: `EXT-GEN-ACT-CONFIDENTIALITY`

documentDate	The document date is the date when the document was written. This is usually the date when the author has created the data of the document. It is not necessarily the date when the document was entered into the system by the data enterer. This date is always specified by the client.	impl:DateXto	
creationDate	The creation date is the point in time when the document was created in the first place. This date is set by the system. Data submitted by the external client is ignored here.	impl:DateXto	

2.1.6 Impl:DocumentContextXto

Attribute	Description	Type	Range
documentType	Specifies the type of a document. This helps the user to classify which content is enclosed in the document. For example if the document is an xray and because of this belongs to medical documents or if it is an email or anything else.	`impl:CodeXto`	Code Set Category: `C-DOCUMENT-CODE`
schemaVersion	Version of a xsd which is used to validate this XML document	xsd:string	Up to 24 characters

mimeType	Specifies the MIME type of the document. This information is important for the client application (browser or web service client) in order to display the content of the document correctly.	impl:CodeXto	Code Set Category: C-DOCUMENT-MIMETYPE
language	Defines the language in which the document is written	impl:CodeXto	Code Set Category: C-LANGUAGE-GUI
children	Children of this document. This way a document tree consisting of a root document and one or more layers of child documents can be constructed. There are seperate methods existing to manipulate only parts of this tree. Children currently can not be deleted as such. They have to be removed from the parent document and then the parent document must be updated.	impl:ArrayOfDocumentContextXto	

print-StyleSheetQualifier	Specifies a reference to a stylesheet for printing. This is for XML documents only. In order to print a XML document in a nice layouted way , a stylesheet is necessary which transfers the XML into a human readable form. With this qualifier you can reference a special document inside the document management, which is then used as a XSL stylesheet for printing the XML document. If no stylesheet is specified by the client the system uses a default stylesheet depending on the document type.	impl:DocumentQualifierXto	

styleSheetQualifier	Specifies a reference to a stylesheet for displaying. This is for XML documents only. In order to display a XML document in a nice layouted way , a stylesheet is necessary which transfers the XML into a human readable form. With this qualifier you can reference a special document inside the document management, which is then used as a XSL stylesheet for displaying the XML document usually as a HTML for the browser of the user. If no stylesheet is specified by the client the system uses a default stylesheet depending on the document type.	impl:DocumentQualifierXto	

rootDocu-mentQualifier	If this document is part of a document tree, then this qualifier provides an eay access to the root document of the whole document tree. This is for convenience as certain operations can only be performed on the root document itself.	impl:Documen ntQualifierXto	

2.1.7 Impl: DocumentContentContextXto

Attribute	Description	Type	Range
dataHandler	The data handler is re-sponsible for the handling (streaming) of the docu-ments content.	apache-soap:DataHa ndler	

2.1.8 Impl: StyleSheetContextXto

Only contains attributes which are set with default values that cannot be al-tered.

2.1.9 Impl: StyleSheetContenContextXto

Attribute	Description	Type	Range
dataHandler	The data handler is responsible for the handling (streaming) of the content of the stylesheet.	apache-soap:DataHandler	

3 Method description

3.1.1 uploadDocument

Description:

Uploads a new document or a new version of an existing document.

Usage Notes:

Uploading a new version of an existing document means applying a DocumentContentContextXto with an existing documentIdentifier. The given DocumentContentContext may have children, which have children ... Note, that in case of an update each these documents in such a hierachy has to be point to an existing child of the tree. Otherwise, those children are just created as new documents and the old documents are orphaned.

Arguments:

Parameter	Description	Data Type	Usage
xto	DocumentContextXto containing document content and its metadata	impl:DocumentContentContextXto	R

Returns:

Description	Data Type
The current version of the document with all server provided fields filled. In particular all qualifier do exist now.	impl:DocumentContextXto

Error codes/Exceptions:

Besides the common errors and exception, the following errors may occur:

Error code/Exception	Reason
`InvalidDocumentContextException`	• one of the mandatory attributes alias, scope, documentType is missing • invalid documentType was submitted
`ServiceException`	the stylesheet specified in the stylesheetQualifier could not be found

3.1.2 uploadDocumentPart

Description:

Upload a part of an existing compound document. The versions of all ancestors of a newly added document part are increased, but not the descendants of the ancestors. Apart from that, it is the same as uploadDocument.

Arguments:

Parameter	Description	Data Type	Usage
part	document part	impl:DocumentContentContextXto	R
parent	the object that defined the parent	impl:DocumentQualifierXto	R

Returns:

Description	Data Type
the version of the document part as property in a DocumentContextXto object	impl:DocumentContextXto

Error codes/Exceptions:

Besides the common errors and exception, the following errors may occur:

Error code/Exception	Reason
`com.icw.ehf.document.exception.InvalidDocumentContextException`	• one of the mandatory attributes alias, scope, documentType is missing. • invalid documentType was submitted.
`ServiceException`	if the stylesheet specified in the stylesheetQualifier could not be found.
`java.lang.RuntimeException`	if the qualified parent container could not be located.

3.1.3 downloadDocument

Description:

Downloads the content of the current version of the document. Depending on `childrenContentIsIncluded` the content of the children is downloaded too.

Arguments:

Parameter	Description	Data Type	Usage
qualifier	The unique identifier of a document.	impl:DocumentQualifierXto	R
childrenContentIsIncluded	If set to `true`, the content of all the children is downloaded. Otherwise, only the content of the root document is returned and for the children only the metadata are returned.	xsd:boolean	R

Returns:

Description	Data Type
The content of the document is embedded as data handler, null if nothing found.	impl:DocumentContentContextXto

3.1.4 deleteDocument

Description:

Deletes the qualified root document and all document tree descendants, if present. If the qualified document is not a root document, or if the document can not be found, an exception is thrown.

Arguments:

Parameter	Description	Data Type	Usage
qualifier	the root document qualifier	impl:DocumentQualifierXto	R

Returns: none

Error codes/Exceptions:

Besides the common errors and exception, the following errors may occur:

Error code/Exception	Reason
`java.lang.RuntimeException`	`if the document specified in the qualifier is not existing, not a root document or the qualifier is empty`

3.1.5 loadAllDocumentsByScope

Description:

Loads the metadata of all the documents which are in the given scope.

Usage Notes: Only root documents will be in the array. Non-root documents are embedded as children in their parents.

Arguments:

Parameter	Description	Data Type	Usage
scope	The given scope to find relative documents for.	xsd:String	R

Returns:

Description	Data Type
array of all documents metadata.	impl:ArrayOfDocumentContextXto

3.1.6 loadAllDocumentsByDocumentTypes

Description:

Loads the metadata of all the documents which are in the given scope and of the given document types.

Usage Notes: Only root documents will be in the array. Non-root documents are embedded as children in their parents.

Arguments:

Parameter	Description	Data Type	Usage
scope	The given scope to find relative documents for.	xsd:String	R
documentTypes	Array of the given types of documents to find relative documents for. If empty, then all documents are returned. Must be C-DOCUMENT-CODE	impl:ArrayOfCodeXto	O

Returns:

Description	Data Type
array of all documents metadata.	impl:ArrayOfDocumentContextXto

3.1.7 loadMetaData

Description:
Loads the meta data of an object.

Arguments:

Parameter	Description	Data Type	Usage
objectQualifier	The object from which the meta data should be loaded.	xsd:QName	R

Returns:

Description	Data Type
The meta data of the given object.	`impl:ClassMetaData`