



Documentation for SDK for .NET 1.2.2

Table of Contents

Table of Contents	2
SDK for .NET 1.2.2	5
License	5
Compatibility	5
Download	5
Requirements	6
Visual studio project configuration	6
Sample client	9
Basic concepts	10
Authentication	10
Accessing API	11
Class Hierarchy	13
Auth samples	15
Basics	15
Methods	15
getGrantedRoles	15
getGrantedUsers	16
getMail	17
getName	17
getRoles	18
getRolesByUser	19
getUsers	19
getUsersByRole	20
revokeRole	21
revokeUser	22
grantRole	23
grantUser	24
Conversion samples	25
Methods	25
doc2pdf	25
imageConvert	26
Document samples	28
Basics	28
Methods	28
createDocumentSimple	28
deleteDocument	29
getDocumentProperties	29
getContent	30
getContentByVersion	31
getDocumentChildren	32
renameDocument	33
setProperties	33
checkout	34
cancelCheckout	35
forceCancelCheckout	36
isCheckedOut	37
checkin	38
getDocumentVersionHistory	38
lock	39
unlock	40
forceUnlock	41
isLocked	41
getLockInfo	42
purgeDocument	43
moveDocument	44
copyDocument	44
restoreVersion	45

purgeVersionHistory	46
getVersionHistorySize	47
isLocked	48
getDocumentPath	48
isValidDocument	49
Folder samples	51
Basics	51
Methods	51
createFolder	51
createFolderSimple	52
getFolderProperties	52
deleteFolder	53
renameFolder	54
moveFolder	54
getFolderChildren	55
isValidFolder	56
getFolderPath	57
Note samples	58
Basics	58
Methods	58
addNote	58
getNote	59
deleteNote	59
setNote	60
listNotes	61
Property samples	63
Basics	63
Methods	63
addCategory	63
removeCategory	64
addKeyword	64
removeKeyword	65
setEncryption	66
unsetEncryption	67
setSigned	68
PropertyGroup samples	69
Basics	69
Methods	69
addGroup	69
removeGroup	70
getGroups	71
getAllGroups	72
getPropertyGroupProperties	72
getPropertyGroupForm	73
setPropertyGroupProperties	74
setPropertyGroupPropertiesSimple	76
hasGroup	77
getPropertyGroupPropertiesSimple	78
Repository samples	80
Methods	80
getRootFolder	80
getTrashFolder	80
getTemplatesFolder	81
getPersonalFolder	82
getMailFolder	83
getThesaurusFolder	83
getCategoriesFolder	84
purgeTrash	85
getUpdateMessage	86
getRepositoryUuid	86
hasNode	87
getNodePath	88
getNodeUuid	89
getAppVersion	89

executeSqlQuery	90
executeHqlQuery	91
executeScript	92
getConfiguration	93
Search samples	95
Basics	95
Methods	98
findByContent	98
findByName	98
findByKeywords	99
find	100
findPaginated	101
findSimpleQueryPaginated	102
findMoreLikeThis	104
getKeywordMap	105
getCategorizedDocuments	106
saveSearch	107
updateSearch	108
getSearch	109
getAllSearchs	110
deleteSearch	110

SDK for .NET 1.2.2

OpenKM SDK for .NET is a set of software development tools that allows for the creation of applications for OpenKM. The OpenKM SDK for .NET includes a Webservices library.

This Webservices library is a complete API layer to access OpenKM through REST Webservices and provides complete compatibility between OpenKM REST Webservices versions minimizing the changes in your code.

License



SDK for .NET is licensed under the terms of the [EULA - OpenKM SDK End User License Agreement](#) as published by OpenKM Knowledge Management System S.L.

This program is distributed WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the [EULA - OpenKM SDK End User License Agreement](#) for more details.

Compatibility



SDK for .NET version 1.2.2 should be used:

- From OpenKM Community version 6.3.7 and upper.

Download

Download the [OKMRest-1.2.2.zip](#) file.

Requirements

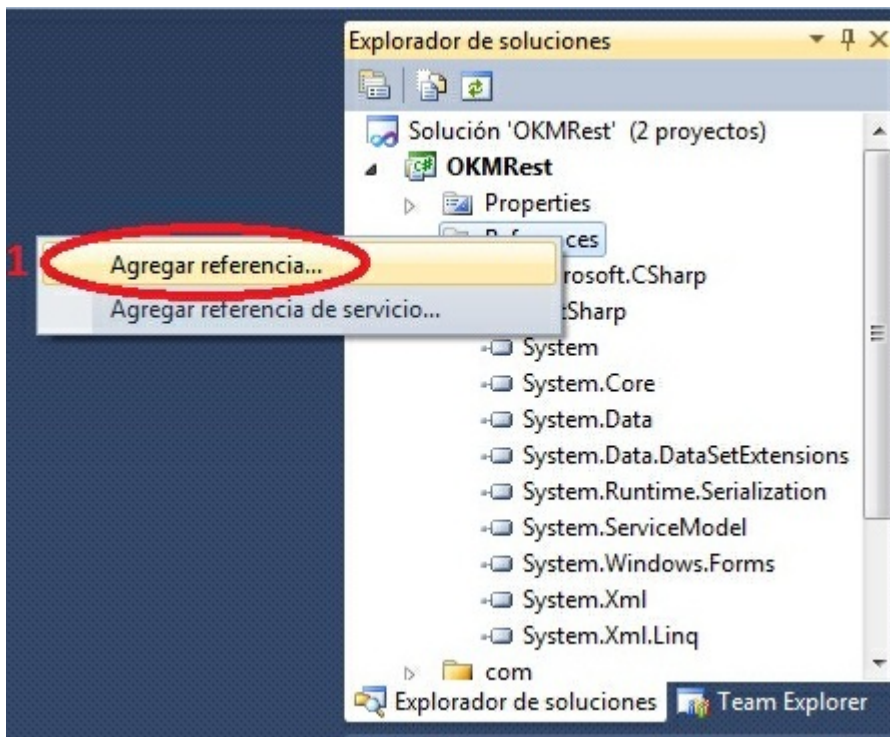
- NetFramework 4.5.2
- SDK for .NET needs RestSharp.dll library version v105.2.3.0.



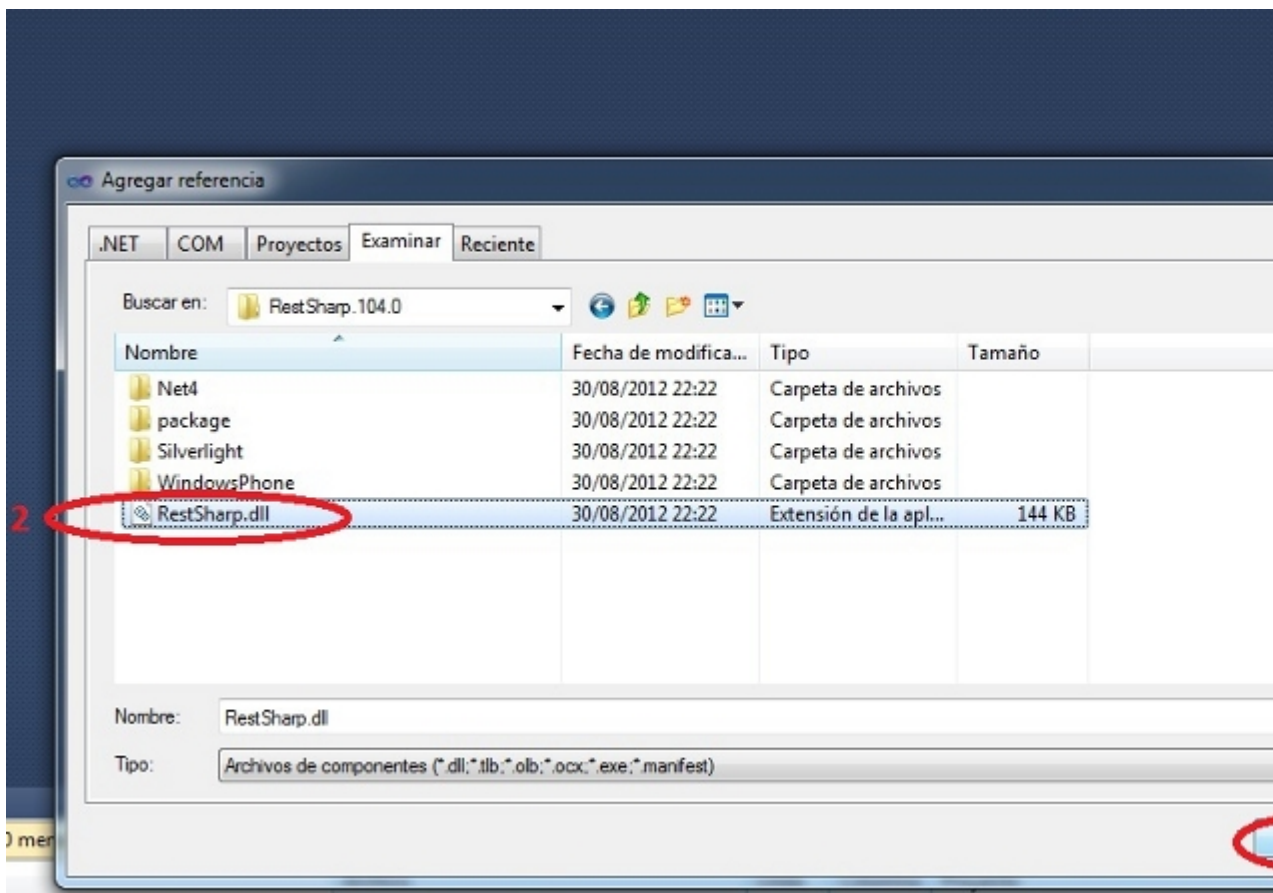
More information about RestSharp at <http://restsharp.org/>.

Visual studio project configuration

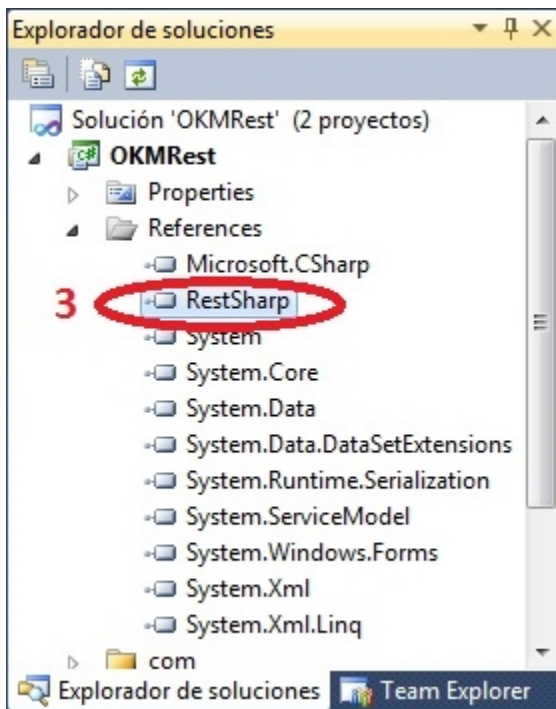
- Download the "**RestSharp.dll**" from [here](#).
- Right click on "**References**" and in contextual menu choose "**add reference**".



- Choose the "**RestSharp.dll**" from your file system.



- Click on "Accept" button.



The dll library is yet configured in your .NET project.

Sample client

Your first class:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    class Program
    {
        /**
         * Sample OpenKM SDK client
         */
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach(Folder fld in ws.getFolderChildren("/okm:root"))
                {
                    System.Console.WriteLine("Folder -> " + fld.path);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }

            System.Console.ReadKey();
        }
    }
}
```

Basic concepts

Authentication

The first lines in your .NET code should be used to create the Webservices object.

We suggest using this method:

```
OKMWebservices ws = OKMWebservicesFactory.newInstance(host, username, password);
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.exception;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try {
                System.Console.WriteLine(ws.getAppVersion().getVersion());
            } catch (RepositoryException e) {
                System.Console.WriteLine(e.ToString());
            } catch (DatabaseException e) {
                System.Console.WriteLine(e.ToString());
            } catch (UnknowException e) {
                System.Console.WriteLine(e.ToString());
            } catch (WebserviceException e) {
                System.Console.WriteLine(e.ToString());
            }

            System.Console.ReadKey();
        }
    }
}
```

Also is possible doing the same from each API class implementation.



We do not suggest this way.

For example with this method:

```
RepositoryImpl repositoryImpl = new RepositoryImpl(host, username, password);
```

```
using System;
using System.Collections.Generic;
```

```

using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.exception;
using com.openkm.sdk4csharp.impl;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            RepositoryImpl repositoryImpl = new RepositoryImpl(host, username, password);

            try {
                System.Console.WriteLine(repositoryImpl.getAppVersion().getVersion());
            } catch (RepositoryException e) {
                System.Console.WriteLine(e.ToString());
            } catch (DatabaseException e) {
                System.Console.WriteLine(e.ToString());
            } catch (UnknownException e) {
                System.Console.WriteLine(e.ToString());
            } catch (WebserviceException e) {
                System.Console.WriteLine(e.ToString());
            }

            System.Console.ReadKey();
        }
    }
}

```

Accessing API

OpenKM API classes are under com.openkm package, as can shown at this [javadoc API summary](#).



At main url <http://docs.openkm.com/apidoc/> you'll see all available javadoc documentation.

At the moment of writing this page the actual OpenKM version was 6.3.0 what can change on time.



There is a direct correspondence between the classes and methods into, implemented at com.openkm.api packages and available from SDK for .NET.

OpenKM API classes:

OpenKM API class	Description	Supported	Implementation	Interface
OKMAuth	Manages security and users. For example add or remove grants on a node, create or modify users or getting the profiles.	Yes	AuthImpl.cs	BaseAuth.cs


OKMBookmark	Manages the user bookmarks.	No		
OKMDashboard	Manages all data shown at dashboard.	No		
OKMDocument	Manages documents. For example creates, moves or deletes a document.	Yes	DocumentImpl.cs	BaseDocument.cs
OKMFolder	Manages folders. For example creates, moves or deletes a folder.	Yes	FolderImpl.cs	BaseFolder.cs
OKMMail	Manages mails. For example creates, moves or deletes a mail.	No	MailImpl.cs	BaseMail.cs
OKMNote	Manages notes on any node type. For example creates, edits or deletes a note on a document, folder, mail or record.	Yes	NoteImpl.cs	BaseNote.cs
OKMNotification	Manages notifications. For example adds or removes subscriptions on a document or a folder.	No		
OKMProperty	Manages categories and keywords. For example adds or removes keywords on a document, folder, mail	Yes	PropertyImpl.cs	BaseProperty.cs

	or record.			
OKMPropertyGroup	Manages metadata. For example adds metadata group, sets metadata fields.	Yes	PropertyGroupImpl.cs	BasePropertyGroup.cs
OKMRepository	A lot of stuff related with repository. For example it gets the properties of main root node (/okm:root).	Yes	RepositoryImpl.cs	BaseRepository.cs
OKMSearch	Manage search feature. For example it manages saved queries or perform a new query to the repository.	Yes	SearchImpl.cs	BaseSearch.cs
OKMStats	General stats of the repository.	No		
OKMUserConfig	Manages user home configuration.	No		

Class Hierarchy

Packages detail:

Name	Description
com.openkm.sdk4csharp	<p>The com.openkm.OKMWebservicesFactory that returns an com.openkm.OKMWebservices object which implements all interfaces.</p> <pre>OKMWebservices ws = OKMWebservicesFactory.newInstance(host, username, password);</pre>
com.openkm.sdk4csharp.bean	Contains all classes result of unmarshalling REST objects.

com.openkm.sdk4csharp.definition	<p>All interface classes:</p> <ul style="list-style-type: none"> • com.openkm.sdk4csharp.definition.BaseAuth • com.openkm.sdk4csharp.definition.BaseDocument • com.openkm.sdk4csharp.definition.BaseFolder • com.openkm.sdk4csharp.definition.BaseNote • com.openkm.sdk4csharp.definition.BaseProperty • com.openkm.sdk4csharp.definition.BasePropertyGroup • com.openkm.sdk4csharp.definition.BaseRepository • com.openkm.sdk4csharp.definition.BaseSearch
com.openkm.sdk4csharp.impl	<p>All interface implementation classes:</p> <ul style="list-style-type: none"> • com.openkm.sdk4csharp.impl.AuthImpl • com.openkm.sdk4csharp.impl.DocumentImpl • com.openkm.sdk4csharp.impl.FolderImpl • com.openkm.sdk4csharp.impl.NoteImpl • com.openkm.sdk4csharp.impl.PropertyGroupImpl • com.openkm.sdk4csharp.impl.PropertyImpl • com.openkm.sdk4csharp.impl.RepositoryImpl • com.openkm.sdk4csharp.impl.SearchImpl
com.openkm.sdk4csharp.util	<p>A couple of utilities.</p> <div>  <p>The class com.openkm.sdk4csharp.util.ISO8601 should be used to set and parse metadata date fields.</p> </div>
com.openkm.sdk4csharp.exception	<p>All exception classes.</p>

Auth samples

Basics

The class **com.openkm.sdk4csharp.bean.Permission** contains permission values (READ, WRITE, etc.). You should use it in combination with methods that are changing or getting security grants.



To set READ and WRITE access you should do:

```
int permission = Permission.READ + Permission.WRITE;
```

To check if you have permission access you should do:

```
// permission is a valid integer value
if ((permission | Permission.WRITE) = Permission.WRITE) {
    // Has WRITE grants.
}
```

On almost methods you'll see parameter named "**nodeId**". The value of this parameter can be some valid node **UUID** (folder, document, mail, record) or node **path**.



Example of nodeId:

- Using UUID -> "**c41f9ea0-0d6c-45da-bae4-d72b66f42d0f**";
- Using path -> "**/okm:root/sample.pdf**"

Methods

getGrantedRoles

Description:

Method	Return values	Description
getGrantedRoles(String nodeId)	Dictionary<String, int>	Return the granted roles of a node.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
```

```

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Dictionary<String, int> grants = ws.getGrantedRoles("/okm:root");
                foreach (String role in grants.Keys)
                {
                    System.Console.WriteLine("role ->" + role);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getGrantedUsers

Description:

Method	Return values	Description
getGrantedUsers(String nodeId)	Dictionary<String, int>	Returns the granted users of a node.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Dictionary<String, int> grants = ws.getGrantedUsers("/okm:root");
                foreach (KeyValuePair<string, int> kvp in grants)
                {

```



```
                Console.WriteLine("{0} -> {1}", kvp.Key, kvp.Value);
            }
        }
    }
}
}
```

getMail

Description:

Method	Return values	Description
getMail(String user)	String	Returns the mail of a valid user.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getMail("okmAdmin"));
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getName

Description:

Method	Return values	Description
getName(String user)	String	Returns the name of a valid user.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getName("okmAdmin"));
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
            System.Console.ReadKey();
        }
    }
}
```

getRoles

Description:

Method	Return values	Description
getRoles()	List<String>	Returns the list of all the roles.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);
```

```

        try
        {
            foreach (String role in ws.getRoles())
            {
                System.Console.WriteLine(role);
            }
        } catch (Exception e) {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

getRolesByUser

Description:

Method	Return values	Description
getRolesByUser(String user)	List<String>	Returns the list of all the roles assigned to a user.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (String role in ws.getRolesByUser("okmAdmin"))
                {
                    System.Console.WriteLine(role);
                }
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getUsers

Description:

Method	Return values	Description
--------	---------------	-------------

getUsers()	List<String>	Returns the list of all the users.
-------------------	---------------------------	------------------------------------

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (String user in ws.getUsers())
                {
                    System.Console.WriteLine(user);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getUsersByRole

Description:

Method	Return values	Description
getUsersByRole(String role)	List<String>	Returns the list of all the users who have assigned a role.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
```

```

{
    static void Main(string[] args)
    {
        String host = "http://localhost:8080/OpenKM";
        String username = "okmAdmin";
        String password = "admin";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

        try
        {
            foreach (String user in ws.getUsersByRole("ROLE_ADMIN"))
            {
                System.Console.WriteLine(user);
            }
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

revokeRole

Description:

Method	Return values	Description
revokeRole(String nodeId, String role, int permissions, boolean recursive)	void	Removes a role grant on a node.
<p>The parameter recursive only makes sense when the nodeId is a folder or record node.</p> <p>When parameter recursive is true, the change will be applied to the node and descendants.</p>		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

```

```

        try
        {
            // Remove ROLE_USER write grants at the node but not descendants
            ws.revokeRole("/okm:root", "ROLE_USER", Permission.ALL_GRANTS, false)

            // Remove all ROLE_ADMIN grants to the node and descendants
            ws.revokeRole("/okm:root", "ROLE_ADMIN", Permission.ALL_GRANTS, true)
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

revokeUser

Description:

Method	Return values	Description
revokeUser(String nodeId, String user, int permissions, boolean recursive)	void	Removes a user grant on a node.
<p>The parameter recursive only makes sense when the nodeId is a folder or record node.</p> <p>When parameter recursive is true, the change will be applied to the node and descendants.</p>		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Remove john write grants at the node but not descendants
                ws.revokeUser("/okm:root", "john", Permission.ALL_GRANTS, false);

                // Remove all okmAdmin grants at the node and descendants
            }
        }
    }
}

```

```

        ws.revokeUser("/okm:root", "okmAdmin", Permission.ALL_GRANTS, true);
    } catch (Exception e) {
        System.Console.WriteLine(e.ToString());
    }
}
}
}

```

grantRole

Description:

Method	Return values	Description
grantRole(String nodeId, String role, int permissions, boolean recursive)	void	Adds a role grant on a node.
<p>The parameter recursive only makes sense when the nodeId is a folder or record node.</p> <p>When a parameter recursive is true, the change will be applied to the node and descendants.</p>		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Add ROLE_USER write grants at the node but not descendants
                ws.grantRole("/okm:root", "ROLE_USER", Permission.ALL_GRANTS, false);

                // Add all ROLE_ADMIN grants to the node and descendants
                ws.grantRole("/okm:root", "ROLE_ADMIN", Permission.ALL_GRANTS, true);
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

grantUser

Description:

Method	Return values	Description
grantUser(String nodeId, String user, int permissions, boolean recursive)	void	Adds a user grant on a node.
The parameter recursive only makes sense when the nodeId is a folder or record node.		
When a parameter recursive is true, the change will be applied to the node and descendants.		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Add john write grants at the node but not descendants
                ws.grantUser("/okm:root", "john", Permission.ALL_GRANTS, false);

                // Add all okmAdmin grants at the node and descendants
                ws.grantUser("/okm:root", "okmAdmin", Permission.ALL_GRANTS, true);
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```


Conversion samples


Methods

doc2pdf

Description:

Method	Return values	Description
doc2pdf(FileStream is, String fileName)	Stream	Retrieve the uploaded document converted to PDF format.

The parameter fileName is the document file name. The application uses this parameter to identify by document extension the document MIME TYPE.

 The OpenOffice service must be enabled in OpenKM server to get it running.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.util;
using System.IO;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);
            try
            {
                BeanHelper beanHelper = new BeanHelper();
                FileStream fileInput = new FileStream("C:\\Documents\\test.docx", FileMode.Open);
                Stream stream = ws.doc2pdf(fileInput, "test.docx");
                Byte[] data = beanHelper.ReadToEnd(stream);
                FileStream fileStream = new FileStream("C:\\Documents\\out.pdf", FileMode.Create);
                foreach (byte b in data)
                {
                    fileStream.WriteByte(b);
                }
                fileStream.Close();
            }
            catch { }
        }
    }
}
```



```

        fileInput.Close();
    }
    catch (Exception e)
    {
        System.Console.WriteLine(e.ToString());
    }
}
}

```

imageConvert

Description:

Method	Return values	Description
imageConvert(FileStream fs, String fileName, List<String> params, String dstMimeType)	Stream	Retrieve the uploaded image with transformation.
<p>The variable fileName is the document file name. The application uses this variable to identify by document extension the document MIME TYPE.</p> <p>The parameter dstMimeType is the expected document MIME TYPE result.</p> <div>  Using this method you are really executing on the server side the ImageMagick convert tool. You can set a lot of parameters - transformations - in the params variable. Take a look at ImageMagick convert tool to get a complete list of them. </div> <div>  The image convert tool must be enabled in OpenKM server to get it running. When params value is not empty always must contain ends with the chain "\${fileIn} \${fileOut}". Ensure there is only a white space as a separator between two parameters. When building your integrations, we suggest installing ImageMagick software locally and check your image transformations first from your command line. </div>		

Example:

```

using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.util;
using System.IO;

namespace OKMRest
{
    public class Program

```

```
{
    static void Main(string[] args)
    {
        String host = "http://localhost:8080/OpenKM";
        String username = "user1";
        String password = "pass1";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);
        try
        {
            FileStream filestream = new FileStream("C:\\Documents\\test.png", FileMode.Open);
            List<string> param = new List<string>();
            param.Add("-resize 50% ${fileIn} ${fileOut}");
            Stream stream = ws.imageConvert(filestream, "test.png", param, "image/png");
            BeanHelper beanHelper = new BeanHelper();
            Byte[] data = beanHelper.ReadToEnd(stream);
            FileStream fileStream = new FileStream("C:\\Documents\\test.png", FileMode.Create);
            foreach (byte b in data)
            {
                fileStream.WriteByte(b);
            }
            fileStream.Close();
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}
```

Document samples

Basics

On most methods you'll see parameter named "**docId**". The value of this parameter can be some valid document **UUID** or **path**.



Example of docId:

- Using UUID -> "f123a950-0329-4d62-8328-0ff500fd42db";
- Using path -> "/okm:root/logo.png"

Methods

createDocumentSimple

Description:

Method	Return values	Description
createDocumentSimple(String docPath, FileStream fs)	Document	Creates a new document and returns as a result an object Document with the properties of the created document.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;
using System.IO;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                FileStream fileStream = new FileStream("E:\\logo.png", FileMode.Open);
                ws.createDocumentSimple("/okm:root/logo.png", fileStream);
                fileStream.Dispose();
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

```


    }
}

```

deleteDocument

Description:

Method	Return values	Description
deleteDocument(String docId)	void	Deletes a document.

 When a document is deleted is automatically moved to /okm:trash/userId folder.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.deleteDocument("/okm:root/logo.png");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }

            System.Console.ReadKey();
        }
    }
}

```

getDocumentProperties

Description:

Method	Return values	Description
getDocumentProperties(String docId)	Document	Returns the document properties.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getDocumentProperties("/okm:root/logo.png"));
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getContent

Description:

Method	Return values	Description
getContent(String docId)	Stream	Retrieves a document content - binary data - of the actual document version



In case you sent the file across a Servlet response we suggest set the content length with:

```
Document doc = ws.getDocumentProperties("/okm:root/logo.png");
response.setContentLength(new Long(doc.getActualVersion().getSize()).intValue());
```



We've found wrong size problems while using:

```
response.setContentLength(is.available());
```

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using System.IO;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Stream s = ws.getContent("/okm:root/logo.png");
                FileStream fs = new FileStream(@"D:\Download\logo.png", FileMode.Open);
                s.CopyTo(fs);
                fs.Close();
                s.Close();
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getContentByVersion

Description:

Method	Return values	Description
getContentByVersion(String docId,String versionId)	Stream	Retrieve document content (binary data) of some specific version.



In case you sent the file across a Servlet response we suggest set the content length with:

```
Document doc = ws.getDocumentProperties("/okm:root/logo.png");
response.setContentLength(new Long(doc.getActualVersion().getSize()).intValue());
```



We've found wrong size problems while using:

```
response.setContentLength(is.available());
```

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using System.IO;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Stream st = ws.getContentByVersion("/okm:root/logo.png", "1.1");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getDocumentChildren

Description:

Method	Return values	Description
getDocumentChildren(String fldId)	List<Document>	Returns a list of all documents which their parent is fldId.
The parameter fldId can be a folder or a record node.		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
```



```

        String password = "admin";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

        try
        {
            foreach (Document doc in ws.getDocumentChildren("/okm:root"))
            {
                System.Console.WriteLine(doc);
            }
        } catch (Exception e) {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

renameDocument

Description:

Method	Return values	Description
renameDocument(String docId, String newName)	document	Changes the name of a document.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Document doc = ws.renameDocument("f123a950-0329-4d62-8328-0ff500fd42", "NewName");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```


setProperties

Description:

Method	Return values	Description
setProperties(Document doc)	void	Changes a document properties.

Variables allowed to be changed:

- Title
- Description
- Language
- Associated categories
- Associated keywords

 Only not null and not empty variables will take on consideration.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Document doc = ws.getDocumentProperties("f123a950-0329-4d62-8328-0ff5");
                doc.description = "some description";
                doc.keywords.Add("test");
                ws.setProperties(doc);
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

checkout

Description:

Method	Return values	Description
checkout(String docId)	void	Marks the document for edition.

Only one user can modify a document at the same time.

Before starting edition must do a checkout action that locks the edition process for other users and allows only to the user who has executed the action.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.checkout("/okm:root/logo.png");
                // At this point the document is locked for other users except for the user who executed the action.
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

cancelCheckout

Description:

Method	Return values	Description
cancelCheckout(String docId)	void	Cancels a document edition.

 This action can only be done by the user who previously executed the checkout action.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // At this point the document is locked for other users except for the user who executed the checkout action.
                ws.cancelCheckout("/okm:root/logo.png");
                // At this point other users are allowed to execute a checkout and modify the document.
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```


forceCancelCheckout

Description:

Method	Return values	Description
forceCancelCheckout(String docId)	void	Cancels a document edition.

This method allows canceling edition on any document.

It is not mandatory to execute this action by the same user who previously executed the checkout action.

 This action can only be done by a superuser (user with ROLE_ADMIN).

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
```

```

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // At this point the document is locked for other users except for the user who is editing it.
                ws.forceCancelCheckout("/okm:root/logo.png");
                // At this point other users are allowed to execute a checkout and modify the document.
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

isCheckedOut

Description:

Method	Return values	Description
isCheckedOut(String docId)	Boolean	Returns a boolean that indicates if the document is on edition or not.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine("Is the document checkout:"+ws.isCheckedOut("/okm:root/logo.png"));
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

checkin

Description:

Method	Return values	Description
checkin(String docId, String comment, FileStream fs)	Version	Updates a document with a new version and returns an object with new Version values.
 Only the user who started the edition - checkout - is allowed to update the document.		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                FileStream fs = new FileStream("E:\\\\logo.png", FileMode.Open);
                ws.checkin("/okm:root/logo.png", "optional some comment", fs);
                fs.Dispose();
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getDocumentVersionHistory

Description:

Method	Return values	Description
getDocumentVersionHistory(String docId)	List<Version>	Returns a list of all document versions.

Example:


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (Version version in ws.getVersionHistory("/okm:root/logo.png"))
                {
                    System.Console.WriteLine(version);
                }
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

lock

Description:

Method	Return values	Description
lock(String docId)	LockInfo	Locks a document and returns an object with the Lock information.
<div>  <p>Only the user who locked the document is allowed to unlock.</p> <p>A locked document can not be modified by other users.</p> </div>		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
```

```


{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.lockDocument("/okm:root/logo.png");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

unlock

Description:

Method	Return values	Description
unlock(String docId)	void	Unlocks a locked document.
<div>  Only the user who locked the document is allowed to unlock. </div>		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.unlock("/okm:root/logo.png");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```



```
}  
}
```


forceUnlock

Description:

Method	Return values	Description
forceUnlock(String docId)	void	Unlocks a locked document.

This method allows to unlock any locked document.

It is not mandatory to execute this action by the same user who previously executed the checkout lock action.

 This action can only be done by a superuser (user with ROLE_ADMIN).

Example:

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using com.openkm.sdk4csharp;  
  
namespace OKMRest  
{  
    public class Program  
    {  
        static void Main(string[] args)  
        {  
            String host = "http://localhost:8080/OpenKM";  
            String username = "okmAdmin";  
            String password = "admin";  
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);  
  
            try  
            {  
                ws.forceUnlock("/okm:root/logo.png");  
            } catch (Exception e) {  
                System.Console.WriteLine(e.ToString());  
            }  
        }  
    }  
}
```

isLocked

Description:

Method	Return values	Description

isLocked(String docId)	Boolean	Returns a boolean that indicates if the document is locked or not.
-------------------------------	----------------	--

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine("Is document locked:" + ws.isLocked("/okm:root"));
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getLockInfo

Description:

Method	Return values	Description
getLockInfo(String docId)	LockInfo	Returns an object with the Lock information

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);
        }
    }
}
```

```

        try
        {
            System.Console.WriteLine(ws.getLockInfo("/okm:root/logo.png"));
        } catch (Exception e) {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

purgeDocument

Description:

Method	Return values	Description
purgeDocument(String docId)	void	The document is definitely removed from repository.

Usually, you will purge documents into /okm:trash/userId - the personal trash user locations - but is possible to directly purge any document from the whole repository.



When a document is purged it will only be able to be restored from a previous repository backup. The purge action removes the document definitely from the repository.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.purgeDocument("/okm:trash/okmAdmin/logo.png");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

moveDocument

Description:

Method	Return values	Description
moveDocument(String docId, String dstId)	void	Moves a document into some folder or record.
The values of the dstId parameter should be a folder or record UUID or path.		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.moveDocument("/okm:root/logo.png", "/okm:root/test");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

copyDocument

Description:

Method	Return values	Description
copyDocument(String docId, String dstId)	void	Copies a document into some folder or record.
The values of the dstId parameter should be a folder or record UUID or path.		
When parameter newName value is null, the document will preserve the same name.		
 Only the binary data and the security grants are copied to the destination, the metadata, keywords, etc. of the		

document are not copied.

See "**extendedDocumentCopy**" method for this feature.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.copyDocument("/okm:root/logo.png", "/okm:root/temp");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

restoreVersion

Description:

Method	Return values	Description
restoreVersion(String docId, String versionId)	void	Promote a previous document version to the actual version.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {

```

```

static void Main(string[] args)
{
    String host = "http://localhost:8080/OpenKM";
    String username = "okmAdmin";
    String password = "admin";
    OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

    try
    {
        // Actual version is 2.0
        ws.restoreVersion("/okm:root/logo.png", "1.0");
        // Actual version is 1.0
    } catch (Exception e) {
        System.Console.WriteLine(e.ToString());
    }
}
}


```

purgeVersionHistory

Description:

Method	Return values	Description
purgeVersionHistory(String docId)	void	Purges all documents version except the actual version.

This action compact the version history of a document.


This action cannot be reverted.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Version history has version 1.3,1.2,1.1 and 1.0
                ws.purgeVersionHistory("/okm:root/logo.png");
                // Version history has only version 1.3
            }
        }
    }
}

```

```

        } catch (Exception e) {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

getVersionHistorySize

Description:

Method	Return values	Description
getVersionHistorySize(String docId)	long	Returns the sum in bytes of all documents into documents history.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                String[] UNITS = new String[] { "B", "KB", "MB", "GB", "TB", "PB", "EB" };
                long bytes = ws.getVersionHistorySize("/okm:root/logo.png");
                String value = "";

                for (int i = 6; i > 0; i--)
                {
                    double step = Math.Pow(1024, i);
                    if (bytes > step)
                        value = String.Format(Locale.ROOT, "%3.1f %s", bytes / step, UNITS[i]);
                }

                if (value.Equals(""))
                {
                    value = bytes.ToString() + " " + UNITS[0];
                }

                System.Console.WriteLine(value);
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

```
    }
}
```

isLocked

Description:

Method	Return values	Description
isLocked(String docId)	Boolean	Returns a boolean that indicates if the node is a document or not.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Return true
                ws.isValidDocument("/okm:root/logo.png");

                // Return false
                ws.isValidDocument("/okm:root");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getDocumentPath

Description:

Method	Return values	Description
getDocumentPath(String uuid)	String	Converts a document UUID to document path.

Example:


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebService ws = OKMWebServicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getDocumentPath("e339f14b-4d3a-489c-91d3-"));
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

isValidDocument

Description:

Method	Return values	Description
isValidDocument(String docId)	Boolean	Returns a boolean that indicates if the node is a document or not.

Parameters:

docId string type is the uuid or path of the Document

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebbservice ws = OKMWebservicesFactory.newInstance(host, username, pass
```

```
        try
        {
            // Return true
            ws.isValidDocument("/okm:root/logo.png");

            // Return false
            ws.isValidDocument("/okm:root");
        } catch (Exception e) {
            System.Console.WriteLine(e.ToString());
        }
    }
}
```

Folder samples

Basics

On most methods you'll see parameter named "**fldId**". The value of this parameter can be some valid folder **UUID** or **path**.



Example of fldId:

- Using UUID -> "**f123a950-0329-4d62-8328-0ff500fd42db**";
- Using path -> "**/okm:root/test**"

Methods


createFolder

Description:

Method	Return values	Description
createFolder(Folder fld)	Folder	Creates a new folder and return as a result an object Folder.

The variable **path** into the parameter **fld**, must be initialized. It indicates the folder path into OpenKM.

```
Folder fld = new Folder();
fld.path = "/okm:root/test";
```

 The other variables of Folder (fld) will not take any effect on folder creation.
We suggest use the method below createFolderSimple rather this one.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
```

```

        String password = "admin";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

        try
        {
            Folder fld = new Folder();
            fld.path = "/okm:root/test";
            ws.createFolder(fld);
        } catch (Exception e) {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

createFolderSimple

Description:

Method	Return values	Description
createFolderSimple(String fldPath)	Folder	Creates a new folder and returns as a result an object Folder.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.createFolderSimple("/okm:root/test");
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getFolderProperties

Description:

Method	Return values	Description
--------	---------------	-------------

getFolderProperties(String fldId)	Folder	Returns the folder properties.
--	---------------	--------------------------------

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getFolderProperties("/okm:root/test"));
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

deleteFolder

Description:

Method	Return values	Description
deleteFolder(String fldId)	void	Delete a folder.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
```

```

        String username = "okmAdmin";
        String password = "admin";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

        try
        {
            ws.deleteFolder("/okm:root/test");
        } catch (Exception e) {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

renameFolder

Description:

Method	Return values	Description
renameFolder(String fldId, String newName)	Folder	Renames a folder.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Exists folder /okm:root/test
                ws.renameFolder("/okm:root/test", "renamedFolder");
                // Folder has renamed to /okm:root/renamedFolder
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

moveFolder

Description:

Method	Return values	Description
moveFolder(String fldId, String dstId)	void	Moves a folder into some folder or record.
The values of the dstId parameter should be a folder or record UUID or path.		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Exists folder /okm:root/test
                ws.moveFolder("/okm:root/test", "/okm:root/tmp");
                // Folder has moved to /okm:root/tmp/test
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getFolderChildren

Description:

Method	Return values	Description
getFolderChildren(String fldId)	List<Folder>	Returns a list of all folder which their parent is fldId.
The parameter fldId can be a folder or a record node.		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
```

```

using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (Folder fld in ws.getFolderChildren("/okm:root"))
                {
                    System.Console.WriteLine(fld);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

isValidFolder

Description:

Method	Return values	Description
isValidFolder(String fldId)	Boolean	Returns a boolean that indicate if the node is a folder or not.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Return false
                ws.isValidFolder("/okm:root/logo.png");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```



```

        // Return true
        ws.isValidFolder("/okm:root");
    } catch (Exception e) {
        System.Console.WriteLine(e.ToString());
    }
}
}
}

```

getFolderPath

Description:

Method	Return values	Description
getFolderPath(String uuid)	String	Converts a folder UUID to folder path.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getFolderPath("f123a950-0329-4d62-8328-0f"));
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

Note samples

Basics

On most methods you'll see parameter named "**nodeId**". The value of this parameter can be a valid document, folder, mail or record **UUID** or **path**.



Example of nodeId:

- Using UUID -> "f123a950-0329-4d62-8328-0ff500fd42db";
- Using path -> "/okm:root/logo.png"

Methods

addNote

Description:

Method	Return values	Description
addNote(String nodeId, String text)	Note	Adds note to a node and returns an object Note.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.addNote("/okm:root/logo.png", "the note text");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getNote

Description:

Method	Return values	Description
getNote(String noteId)	Note	Retrieves the note.

 The noteId is an UUID.
The object Note has a variable named path, in that case the path contains an UUID.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                List<Note> notes = ws.listNotes("/okm:root/logo.png");
                if (notes.Count > 0)
                {
                    System.Console.WriteLine(ws.getNote(notes[0].path));
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

deleteNote

Description:

Method	Return values	Description
--------	---------------	-------------

deleteNote(String noteId)	Note	Deletes a note.
<div>  The noteId is an UUID. The object Node has a variable named path, in that case the path contains an UUID. </div>		

Example:


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                List<Note> notes = ws.listNotes("/okm:root/logo.png");
                if (notes.Count > 0)
                {
                    System.Console.WriteLine(ws.deleteNote(notes[0].path));
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

setNote

Description:

Method	Return values	Description
setNote(String noteId, String text)	void	Changes the note text.
<div>  The noteId is an UUID. The object Node has a variable named path, in that case the path contains an UUID. </div>		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                List<Note> notes = ws.listNotes("/okm:root/logo.png");
                if (notes.Count > 0)
                {
                    ws.setNote(notes[0].path, "text modified");
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

listNotes

Description:

Method	Return values	Description
listNotes(String nodeId)	List<Note>	Retrieves a list of all notes of a node.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
```

```
{
    static void Main(string[] args)
    {
        String host = "http://localhost:8080/OpenKM";
        String username = "okmAdmin";
        String password = "admin";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

        try
        {
            List<Note> notes = ws.listNotes("/okm:root/logo.png");
            foreach (Note note in notes)
            {
                System.Console.WriteLine(note);
            }
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}
```

Property samples

Basics

On most methods you'll see parameter named "**nodeId**". The value of this parameter can be a valid document, folder, mail or record **UUID** or **path**.



Example of nodeId:

- Using UUID -> "f123a950-0329-4d62-8328-0ff500fd42db";
- Using path -> "/okm:root/logo.png"

Methods

addCategory

Description:

Method	Return values	Description
addCategory(String nodeId, String catId)	void	Sets a relation between a category and a node.
The value of the catId parameter should be a category folder UUID or path.		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.addCategory("/okm:root/logo.png", "/okm:categories/test");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

```

    }
}

```

removeCategory

Description:

Method	Return values	Description
removeCategory(String nodeId, String catId)	void	Removes a relation between a category and a node.
The value of the catId parameter should be a category folder UUID or path.		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.removeCategory("/okm:root/logo.png", "/okm:categories/test");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

addKeyword

Description:

Method	Return values	Description
addKeyword(String nodeId, String keyword)	void	Adds a keyword and a node.

The keyword should be a single word without spaces, formats allowed:

- "test"
- "two_words" (the character "_" is used for the junction).



Also we suggest you to add keyword in lower case format, because OpenKM is case sensitive.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.addKeyword("/okm:root/logo.png", "test");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

removeKeyword

Description:

Method	Return values	Description
removeKeyword(String nodeId, String keyword)	void	Removes a keyword from a node.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```

using com.openkm.sdk4csharp;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.removeKeyword("/okm:root/logo.png", "test");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

setEncryption

Description:

Method	Return values	Description
setEncryption(String nodeId, String cipherName)	void	Marks a document as an encrypted binary data into the repository
<p>The parameter nodeId should be a document node.</p> <p>The parameter cipherName saves information about the encryption mechanism.</p> <div>  <p>This method does not perform any kind of encryption, simply marks into the database that a document is encrypted.</p> </div>		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {

```

```

static void Main(string[] args)
{
    String host = "http://localhost:8080/OpenKM";
    String username = "okmAdmin";
    String password = "admin";
    OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

    try
    {
        ws.setEncryption("/okm:root/logo.png", "phrase");
    }
    catch (Exception e)
    {
        System.Console.WriteLine(e.ToString());
    }
}
}


```

unsetEncryption

Description:

Method	Return values	Description
unsetEncryption(String nodeId)	void	Marks a document as a normal binary data into repository.

The parameter nodeId should be a document node.



This method does not perform any kind of unryption, simply marks into the database that a document has been unrypted.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.unsetEncryption("/okm:root/logo.png");
            }
            catch (Exception e)
            {
            }
        }
    }
}

```


```

        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

setSigned

Description:

Method	Return values	Description
setSigned(String nodeId, boolean signed)	void	Marks a document as signed or unsigned binary data into the repository
The parameter nodeId should be a document node.		
 This method does not perform any kind of digital signature process, simply marks into the database that a document is signed.		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.setSigned("/okm:root/logo.pdf", true);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

PropertyGroup samples

Basics



From older OpenKM version we named "**Metadata Groups**" as "**Property Groups**".

Although we understand this name does not helps a lot to identifying these methods with metadata ones, for historical reason, we continue maintaining the nomenclature.

For more information about [Metadata](#).

On most methods you'll see parameter named "**nodeId**". The value of this parameter can be a valid document, folder, mail or record **UUID** or **path**.



Example of nodeId:

- Using UUID -> "**f123a950-0329-4d62-8328-0ff500fd42db**";
- Using path -> "**/okm:root/logo.png**"



The class `com.openkm.sdk4j.util.ISO8601` should be used to set and parse metadata date fields. The metadata field of type date values are stored into application in ISO-8601 basic format.

To convert retrieved metadata field of type date to a valid date use:

```
DateTime date = ISO8601.parseBasic(metadataFieldValue);
```

To save date value into metadata field of type date use:

```
DateTime date = DateTime.now; // Present date
String metadataFieldValue = ISO8601.formatBasic(date);
// metadataFieldValue can be saved into repository metadata field of type date
```

Methods

addGroup

Description:

Method	Return values	Description
addGroup(String nodeId, String grpName)	void	Adds an empty metadata group to a node.
The grpName should be a valid Metadata group name.		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                ws.addGroup("/okm:root/logo.png", "okg:consulting");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

removeGroup

Description:

Method	Return values	Description
removeGroup(String nodeId, String grpName)	void	Removes a metadata group of a node.
The grpName should be a valid Metadata group name.		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
```

```

        String username = "okmAdmin";
        String password = "admin";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

        try
        {
            ws.removeGroup("/okm:root/logo.png", "okg:consulting");
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

getGroups

Description:

Method	Return values	Description
getGroups(String nodeId)	List<PropertyGroup>	Retrieves a list of metadata groups assigned to a node.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (PropertyGroup pGroup in ws.getGroups("/okm:root/logo.png"))
                {
                    System.Console.WriteLine(pGroup);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getAllGroups

Description:

Method	Return values	Description
getAllGroups()	List<PropertyGroup>	Retrieves a list of all metadata groups set into the application.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (PropertyGroup pGroup in ws.getAllGroups())
                {
                    System.Console.WriteLine(pGroup);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getPropertyGroupProperties

Description:

Method	Return values	Description
getPropertyGroupProperties(String nodeId, String grpName)	List<FormElement>	Retrieve a list of all metadata group elements and its values of a node.
The grpName should be a valid Metadata group name.		



The method is usually used to display form elements with its values to be shown or changed by used.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;
using com.openkm.sdk4csharp.bean.form;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (FormElement fElement in ws.getPropertyGroupProperties("/okm:rest"))
                {
                    System.Console.WriteLine(fElement);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getPropertyGroupForm

Description:

Method	Return values	Description
getPropertyGroupForm(String grpName)	List<FormElement>	Retrieves a list of all metadata group elements definition.

The grpName should be a valid Metadata group name.



The method is usually used to display empty form elements for creating new metadata values.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;
using com.openkm.sdk4csharp.bean.form;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (FormElement fElement in ws.getPropertyGroupForm("okg:consult"))
                {
                    System.Console.WriteLine(fElement);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

setPropertyGroupProperties

Description:

Method	Return values	Description
setPropertyGroupProperties(String nodeId, String grpName, List<FormElement> ofeList)	void	Changes the metadata node.
The grpName should be a valid Metadata group name.		
<p>i Is not mandatory to set into parameter ofeList all FormElement, is enough with the formElements you wish to change.</p>		
<p>i The sample below is based on this Metadata group definition:</p> <pre><?xml version="1.0" encoding="UTF-8"?> <!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.0//EN"</pre>		

```

"http://www.openkm.com/dtd/property-groups"
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Name" type="text" name="okp:consulting.name"/>
    <input label="Date" type="date" name="okp:consulting.date" />
    <checkbox label="Important" name="okp:consulting.important"/>
    <textarea label="Comment" name="okp:consulting.comment"/>
  </property-group>
</property-groups>

```

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;
using com.openkm.sdk4csharp.util;
using com.openkm.sdk4csharp.bean.form;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // Modify with a full FormElement list
                List<FormElement> fElements = ws.getPropertyGroupProperties("/okm:root/logo.pdf", "okg:consulting");
                foreach (FormElement fElement in fElements)
                {
                    if (fElement.name.Equals("okp:consulting.name"))
                    {
                        Input name = (Input) fElement;
                        name.value = "new value";
                    }
                }

                ws.setPropertyGroupProperties("/okm:root/logo.pdf", "okg:consulting", fElements);

                // Same modification with only affected FormElement
                fElements = new List<FormElement>();
                Input n = new Input();
                n.name = "okp:consulting.name";
                n.value = "new value";
                fElements.Add(n);
                ws.setPropertyGroupProperties("/okm:root/logo.pdf", "okg:consulting", fElements);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

setPropertyGroupPropertiesSimple

Description:

Method**setPropertyGroupPropertiesSimple(String nodeId, String grpName, Dictionary<String, String> properties)**

The grpName should be a valid Metadata group name.



Is not mandatory to set into properties parameter all fields values, is enough with the fields you wish to change its v



The sample below is based on this Metadata group definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.0//EN"
    "http://www.openkm.com/dtd/property-groups"
</!DOCTYPE>

<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Name" type="text" name="okp:consulting.name"/>
    <input label="Date" type="date" name="okp:consulting.date" />
    <checkbox label="Important" name="okp:consulting.important"/>
    <textarea label="Comment" name="okp:consulting.comment"/>
  </property-group>
</property-groups>
```



To add several values on a metadata field of type multiple like this:

```
<select label="Multiple" name="okp:consulting.multiple" type="multiple">
  <option label="One" value="one"/>
  <option label="Two" value="two"/>
  <option label="Three" value="three" />
</select>
```

You should do:

```
properties.Add("okp:consulting.multiple", "one;two");
```

Where **"one"** and **"two"** are valid values and character ";" is used as separator.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```

using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;
using com.openkm.sdk4csharp.util;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Dictionary<String, String> properties = new Dictionary<String, String>();
                properties.Add("okp:consulting.name", "new name");

                //The date fields must be saved with basic ISO 8601 format
                properties.Add("okp:consulting.date", ISO8601.formatBasic(DateTime.Now));
                ws.setPropertyGroupPropertiesSimple("/okm:root/logo.pdf", "okg:consulting", properties);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

hasGroup

Description:

Method	Return values	Description
hasGroup(String nodeId, String grpName)	Boolean	Returns a boolean that indicate if the node has or not a metadata group.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";

```

```

        String password = "admin";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

        try
        {
            System.Console.WriteLine("Have metadata group:"+ws.hasGroup("/okm:root/logo.pdf"));
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

getPropertyGroupPropertiesSimple

Description:

Method	Return values	Description
getPropertyGroupPropertiesSimple(String nodeId, String grpName)	Dictionary<String, String>	Retrieves a dictionary- (key,value) pairs - with Metada group values of a node.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Dictionary<String, String> properties = new Dictionary<String, String>();
                properties = ws.getPropertyGroupPropertiesSimple("/okm:root/logo.pdf");

                foreach (KeyValuePair<String, String> pair in properties)
                {
                    System.Console.WriteLine(pair.Key + "-> " + pair.Value);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

```
}  
}
```

Repository samples

Methods

getRootFolder

Description:

Method	Return values	Description
getRootFolder()	Folder	Returns the object Folder of node "/okm:root"

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getRootFolder());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getTrashFolder

Description:

Method	Return values	Description
getTrashFolder()	Folder	Returns the object Folder of node "/okm:trash/{userId}"

The returned folder will be the user trash folder.



For example if the method is executed by "okmAdmin" user then the folder returned will be "/okm:trash/okmAdmin".

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getTrashFolder());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getTemplatesFolder

Description:

Method	Return values	Description
getTemplatesFolder()	Folder	Returns the object Folder of node "/okm:templates"

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getTemplatesFolder());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

```

public class Program
{
    static void Main(string[] args)
    {
        String host = "http://localhost:8080/OpenKM";
        String username = "user1";
        String password = "pass1";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

        try
        {
            System.Console.WriteLine(ws.getTemplatesFolder());
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

getPersonalFolder

Description:

Method	Return values	Description
getPersonalFolder()	Folder	Returns the object Folder of node "/okm:personal/{userId}"

The returned folder will be the user personal folder.



For example if the method is executed by "okmAdmin" user then the folder returned will be "/okm:personal/okmAdmin".

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getPersonalFolder());
            }
        }
    }
}

```

```

        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

getMailFolder

Description:

Method	Return values	Description
getMailFolder()	Folder	Returns the object Folder of node "/okm:mail/{userId}"

The returned folder will be the user mail folder.



For example if the method is executed by "okmAdmin" user then the folder returned will be "/okm:mail/okmAdmin".

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getMailFolder());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getThesaurusFolder

Description:

Method	Return values	Description
getThesaurusFolder()	Folder	Returns the object Folder of node "/okm:thesaurus"

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getThesaurusFolder());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getCategoriesFolder

Description:

Method	Return values	Description
getCategoriesFolder()	Folder	Returns the object Folder of node "/okm:categories"

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getCategoriesFolder());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

```

public class Program
{
    static void Main(string[] args)
    {
        String host = "http://localhost:8080/OpenKM";
        String username = "user1";
        String password = "pass1";
        OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);


        try
        {
            System.Console.WriteLine(ws.getCategoriesFolder());
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}


```

purgeTrash

Description:

Method	Return values	Description
purgeTrash()	void	Definitively remove from repository all nodes into "/okm:trash/{userId}"

 For example if the method is executed by "okmAdmin" user then the purged trash will be "/okm:trash/okmAdmin".

 When a node is purged it will only be able to be restored from a previous repository backup. The purge action remove the node definitely from the repository.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try

```

```

        {
            ws.purgeTrash();
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}


```

getUpdateMessage

Description:

Method	Return values	Description
getUpdateMessage()	String	Retrieves a message when a new OpenKM release is available.

There's an official OpenKM update message service available which is based on your local OpenKM version.


 The most common message is that a new OpenKM version has been released.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8180/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getUpdateMessage());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getRepositoryUuid

Description:

Method	Return values	Description
getRepositoryUuid()	String	Retrieves an installation unique identifier.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getRepositoryUuid());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

hasNode

Description:

Method	Return values	Description
hasNode(String nodeId)	Boolean	Returns a node that indicate if a node exists or not.
The value of the parameter nodeId can be a valid UUID or path .		

Example:

```

using System;
using System.Collections.Generic;

```

```

using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine("Exists node:" + ws.hasNode("064ff51a-b815-4000-8000-000000000000"));
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getNodePath

Description:

Method	Return values	Description
getNodePath(String uuid)	String	Converts a node UUID to path.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getNodePath("e339f14b-4d3a-489c-91d3-05e000000000"));
            }
            catch (Exception e)
            {
            }
        }
    }
}

```



```

        System.Console.WriteLine(e.ToString());
    }
}
}
}

```

getNodeUuid

Description:

Method	Return values	Description
getNodeUuid(String path)	String	Converts a node path to UUID.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getNodeUuid("/okm:root/tmp"));
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getAppVersion

Description:

Method	Return values	Description
getAppVersion()	AppVersion	Returns information about OpenKM version.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                System.Console.WriteLine(ws.getAppVersion());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

executeSqlQuery

Description:

Method	Return values	Description
executeSqlQuery(FileStream fs)	SqlQueryResults	Executes SQL sentences.
<p>The test.sql script used in the sample below:</p> <pre>SELECT NBS_UUID, NBS_NAME FROM OKM_NODE_BASE LIMIT 10;</pre>		
<div>  <p>The SQL script can only contain a single SQL sentence.</p> <p>This action can only be done by a superuser (user with ROLE_ADMIN).</p> </div>		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using System.IO;

```

```

using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                FileStream fs = new FileStream("E:\\test.sql", FileMode.Open);
                SqlQueryResults result = ws.executeSqlQuery(fs);

                foreach (SqlQueryResultColumns row in result.sqlQueryResults)
                {
                    System.Console.WriteLine("uuid:" + row.sqlQueryResultColumn[0]);
                }
                fs.Dispose();
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```


executeHqlQuery

Description:

Method	Return values	Description
executeHqlQuery(FileStream fs)	HqlQueryResults	Execute HQL sentences.

The test.sql script used in the sample below:

```
SELECT uuid, name from NodeBase where name = 'okm:root';
```



The HQL script can only contain a single HQL sentence.

This action can only be done by a superuser (user with ROLE_ADMIN).

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

```

```

using com.openkm.sdk4csharp;
using System.IO;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);
            try
            {
                FileStream fs = new FileStream(@"C:\Desktop\test.sql", FileMode.Open);
                HqlQueryResults result = ws.executeHqlQuery(fs);


                foreach (HqlQueryResultColumns row in result.hqlQueryResults)
                {
                    foreach (string column in row.hqlQueryResultColumn)
                    {
                        System.Console.WriteLine(column);
                    }
                }

                fs.Dispose();
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

executeScript

Description:

Method	Return values	Description
executeScript(FileStream fs)	ScriptExecutionResult	Execute an script.
<p>The local script - test.bsh - used in the sample below:</p> <pre> import com.openkm.bean.*; import com.openkm.api.*; for (Folder fld : OKMFolder.getInstance().getChildren(null, "/okm:root")) { print(fld+"\n"); } // Some value can also be returned as string return "some result"; </pre>		
 This action can only be done by a superuser (user with ROLE_ADMIN).		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using System.IO;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                FileStream fs = new FileStream("E:\\test.bsh", FileMode.Open);
                ScriptExecutionResult result = ws.executeScript(fs);
                System.Console.WriteLine(result.result);
                System.Console.WriteLine(result.stdout);


                if (!result.stderr.Equals(""))
                {
                    System.Console.WriteLine("Error happened");
                    System.Console.WriteLine(result.stderr);
                }

                fs.Dispose();
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getConfiguration

Description:

Method	Return values	Description
getConfiguration(String key)	Configuration	Retrieve the value of a configuration parameter.



If your OpenKM version have the configuration parameter named "**webservices.visible.properties**", will be restricted for non Administrator users what parameters are accessible. That means any non Administrator user who will try accessing across the webservices to configuration parameters not set into the list of values

of **"webservices.visible.properties"** will get an access denied exception.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "user1";
            String password = "pass1";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                Configuration configuration = ws.getConfiguration("system.ocr");
                System.Console.WriteLine(configuration);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

Search samples

Basics

Most methods use QueryParams here there're some clues about how using it.

Variables	Type	Allow wildcards	
domain	long	No.	<p>Available values:</p> <ul style="list-style-type: none">• QueryParams.DOCUMENT• QueryParams.FOLDER• QueryParams.MAIL <p>By default the value is set to QueryParams.DOCUMENT.</p> <p>For searching documents and folders use value:</p> <pre>(QueryParams.DOCUMENT QueryParams.FOLDER</pre>
author	String	No.	The value must be a valid userId.
name	String	Yes.	
keywords	List<String>	Yes.	
categories	List<String>	No.	Values should be categories UUID, not use path value.
content		Yes.	

mimeType		No.	The value should be a valid and registered MIME type. Only can be applied to a documents node.
path		No.	When empty is used by default "/okm:root" node.
lastModifiedFrom	Calendar	No.	
lastModifiedTo	Calendar	No.	
mailSubject	String	Yes.	Only applies to mail nodes.
mailFrom	String	Yes.	Only applies to mail nodes.
mailTo		Yes.	Only applies to mail nodes.
properties	Dictionary<String, String>	Yes on almost.	On metadata field values like "date" cannot be applied wildcards. The dictionary of the properties is composed of pairs: ('metadata_field_name','metada_field_value')

For example:

```
Dictionary<String, String> properties = ne
properties.Add("okp:consulting.name", "nam
```

Filtering by a range of dates:

```
DateTime date = DateTime.Now;// today
DateTime to = new DateTime(date.Year, dat
DateTime from = to.AddDays(-3);// three d
Dictionary<String,String> properties = ne
properties.Add("okp:consulting.date", ISO
```



When filtering by a range of dates you mu

To filtering by a metadata field of type multiple

```
<select label="Multiple" name="okp:consult
  <option label="One" value="one"/>
  <option label="Two" value="two"/>
  <option label="Three" value="three" />
</select>
```

You should do:

```
properties.Add("okp:consulting.multiple",
```

Where **"one"** and **"two"** are valid values and chara



The search operation is done only by AND logic.

Wildcard examples:

Variable	Example	Description
name	test*.html	Any document that starts with the characters "test" and ends with characters ".html"
name	test?.html	Any document that starts with the characters "test" followed by a single character and ends with characters ".html"


name	?test*	Any the documents where the first character doesn't matter but is followed by the characters, "test".
-------------	---------------	---

Methods

findByContent

Description:

Method	Return values	Description
findByContent(String content)	List<QueryResult>	Returns a list of results filtered by the value of the content parameter.



The method only searches among all documents, it does not take in consideration any other kind of nodes.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (QueryResult qr in ws.findByContent("test"))
                {
                    System.Console.WriteLine(qr);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

findByName

Description:

Method	Return values	Description
findByName(String name)	List<QueryResult>	Returns a list of results filtered by the value of the name parameter.

 The method only searches among all documents, it does not take in consideration any other kind of nodes.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);


            try
            {
                foreach (QueryResult qr in ws.findByName("?test*"))
                {
                    System.Console.WriteLine(qr);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

findByKeywords

Description:

Method	Return values	Description

findByKeywords(List<String> keywords)	List<QueryResult>	Returns a list of results filtered by the values of the keywords parameter.
--	--------------------------------	---



The method only searches among all documents, it does not take in consideration any other kind of nodes.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                List<string> keywords = new List<string>();
                keywords.Add("test");

                foreach (QueryResult qr in ws.findByKeywords(keywords))
                {
                    System.Console.WriteLine(qr);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

find

Description:

Method	Return values	Description
find(QueryParams queryParams)	List<QueryResult>	Returns a list of results filtered by the values of the queryParams parameter.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;


namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                QueryParams qParams = new QueryParams();
                qParams.domain = QueryParams.DOCUMENT;
                qParams.name = "test*.html";

                foreach (QueryResult qr in ws.find(qParams))
                {
                    System.Console.WriteLine(qr);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

findPaginated

Description:

Method	Return values	Description
findPaginated(QueryParams queryParams, int offset, int limit)	ResultSet	Returns a list of paginated results filtered by the values of the queryParams parameter.
<div>  <p>The parameter "limit" and "offset" allow you to retrieve just a portion of the results of a query.</p> <ul style="list-style-type: none"> • The parameter "limit" is used to limit the number of results returned. • The parameter "offset" says to skip that many results before the beginning to return results. </div>		



For example, if your query has 1000 results, but you only want to return the first 10, you should use these values:

- limit=10
- offset=0

Now suppose you want to show the results from 11-20, you should use these values:

- limit=10
- offset=10

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                QueryParams qParams = new QueryParams();
                qParams.domain = QueryParams.DOCUMENT;
                qParams.name = "test*.html";
                ResultSet rs = ws.findPaginated(qParams, 20, 10);
                System.Console.WriteLine("Total results:" + rs.total);

                foreach (QueryResult qr in rs.results)
                {
                    System.Console.WriteLine(qr);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

findSimpleQueryPaginated

Description:

Method	Return values	Description
findSimpleQueryPaginated(String statement, int offset, int limit)	ResultSet	Returns a list of paginated results filtered by the values of the statement parameter.

i The **syntax** to use in the statement parameter is the pair '**field:value**'. For example:

- "name:grial" is filtering field name by word grial.

More information about Lucene syntaxis at [Lucene query syntax](#).

✓ The parameter "limit" and "offset" allow you to retrieve just a portion of the results of a query.

- The parameter "limit" is used to limit the number of results returned.
- The parameter "offset" says to skip that many results before the beginning to return results.

i For example, if your query has 1000 results, but you only want to return the first 10, you should use these values:

- limit=10
- offset=0

Now suppose you want to show the results from 11-20, you should use these values:

- limit=10
- offset=10

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);
        }
    }
}
```

```


        try
        {
            ResultSet rs = ws.findSimpleQueryPaginated(20, 10, "name:grial");
            System.Console.WriteLine("Total results:" + rs.total);

            foreach (QueryResult qr in rs.results)
            {
                System.Console.WriteLine(qr);
            }
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

findMoreLikeThis

Description:

Method	Return values	Description
findMoreLikeThis(String uuid, int max)	ResultSet	Returns a list of documents that are considered similar by the search engine.
<p>The uuid is a document UUID.</p> <p>The max value is used to limit the number of results returned.</p> <div>  The method can only be used with documents. </div>		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, pass

```



```

        try
        {
            ResultSet rs = ws.findMoreLikeThis("c5cb1982-1a99-4741-8a07-a319b9ac3
            System.Console.WriteLine("Total results:" + rs.total);

            foreach (QueryResult qr in rs.results)
            {
                System.Console.WriteLine(qr);
            }
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

getKeywordMap

Description:

Method	Return values	Description
getKeywordMap(String[] filter)	Dictionary<String, Integer>	Returns a map of keywords with its count value filtered by other keywords.



Example:

- Doc1.txt has keywords "test", "one", "two".
- Doc2.txt has keywords "test", "one"
- Doc3.txt has keywords "test", "three".

The results filtering by "test" -> "one", "two", "three".

The results filtering by "one" -> "test", "two".

The results filtering by "two" -> "test", "one".

The results filtering by "three" -> "test".

The results filtering by "one" and "two" -> "test".

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

```

```

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // All keywords without filtering
                System.Console.WriteLine("Without filtering");
                Dictionary<String, int> keywords = ws.getKeywordMap(new String[]{"test"});

                foreach (KeyValuePair<string,int> kvp in keywords)
                {
                    System.Console.WriteLine(kvp.Key + " is used :" + kvp.Value);
                }

                // Keywords filtered
                System.Console.WriteLine("Filtering");
                keywords = ws.getKeywordMap(new String[]{"test"});

                foreach (KeyValuePair<string,int> kvp in keywords)
                {
                    System.Console.WriteLine(kvp.Key + " is used :" + kvp.Value);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

getCategorizedDocuments

Description:

Method	Return values	Description
getCategorizedDocuments(String categoryId)	List<Document>	Retrieves a list of all documents related with a category.
The values of the categoryId parameter should be a category folder UUID or path.		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

```

```

using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (Document doc in ws.getCategorizedDocuments("/okm:categories/"))
                {
                    System.Console.WriteLine(doc);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

saveSearch

Description:

Method	Return values	Description
saveSearch(QueryParams params)	Long	Saves search parameters.
The variable queryName of the parameter params, should have to be initialized.		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                // ...
            }
        }
    }
}

```

```

        {
            QueryParams qParams = new QueryParams();
            qParams.domain = QueryParams.DOCUMENT;
            qParams.name = "test*.html";


            foreach (QueryResult qr in ws.find(qParams))
            {
                System.Console.WriteLine(qr);
            }

            // Save the search to be used later
            qParams.queryName = "sample search";
            ws.saveSearch(qParams);
        }
        catch (Exception e)
        {
            System.Console.WriteLine(e.ToString());
        }
    }
}

```

updateSearch

Description:

Method	Return values	Description
updateSearch(QueryParams params)	void	Updates a previously saved search parameters.
 It can only be updated as a saved search created by the same user who's executing the method.		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                foreach (QueryParams qParams in ws.getAllSearchs())
                {

```


```

        if (qParams.queryName.Equals("sample search"))
        {
            // Change some value.
            qParams.name = "admin*.html";
            ws.updateSearch(qParams);
        }
    }
}
catch (Exception e)
{
    System.Console.WriteLine(e.ToString());
}
}
}

```

getSearch

Description:

Method	Return values	Description
getSearch(int qpId)	QueryParams	Gets saved searches parameters.
 It can only be retrieved as a saved search created by the same user who's executing the method.		

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                int qpId = 1; // Some valid search id
                QueryParams qParams = ws.getSearch(qpId);
                System.Console.WriteLine(qParams);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

```


    }
}

```

getAllSearchs

Description:

Method	Return values	Description
getAllSearchs()	List<QueryParams>	Retrieves a list of all saved search parameters.

 It can only retrieve the list of the saved searches created by the same user who's executing the method.

Example:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
using com.openkm.sdk4csharp.bean;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);


            try
            {
                foreach (QueryParams qParams in ws.getAllSearchs())
                {
                    System.Console.WriteLine(qParams);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}

```

deleteSearch

Description:

Method	Return values	Description
deleteSearch(int qpId)	void	Deletes a saved search parameters.

 It can only be deleted as a saved search created by the same user who's executing the method.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;

namespace OKMRest
{
    public class Program
    {
        static void Main(string[] args)
        {
            String host = "http://localhost:8080/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, password);

            try
            {
                int qpId = 1; // Some valid search id
                ws.deleteSearch(qpId);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```