Documentation for SDK for .NET 1.4.0

Table of Contents

SDK for .NET 1.4.0

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 1.4.0 should be used:

- **From OpenKM Community version 6.3.11 and upper.**

Download

Download the OKMRest-1.4.0.zip file.

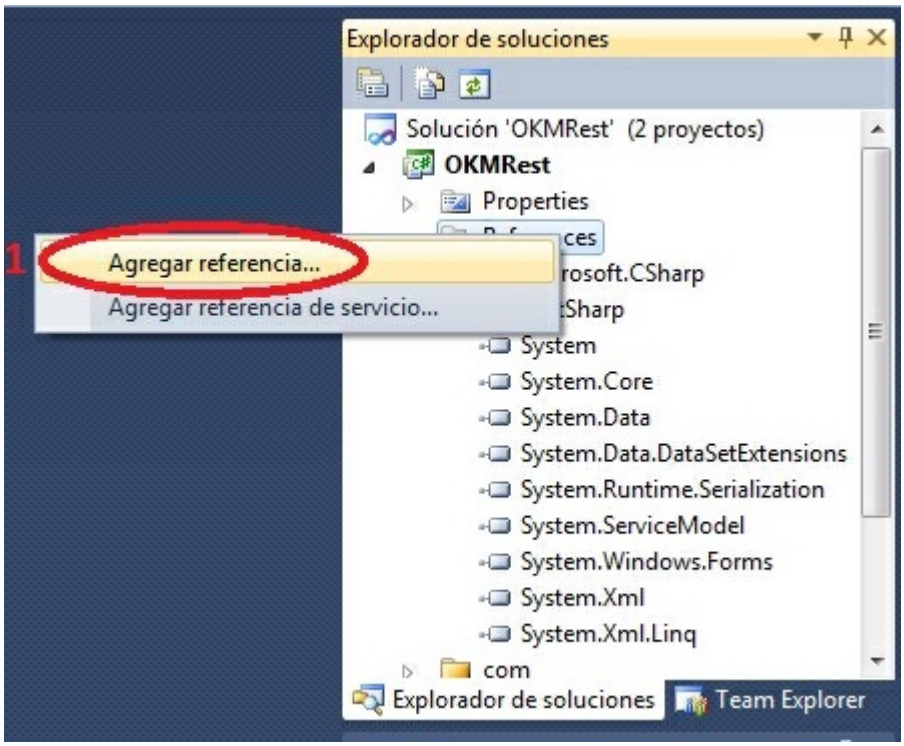
Requirements

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

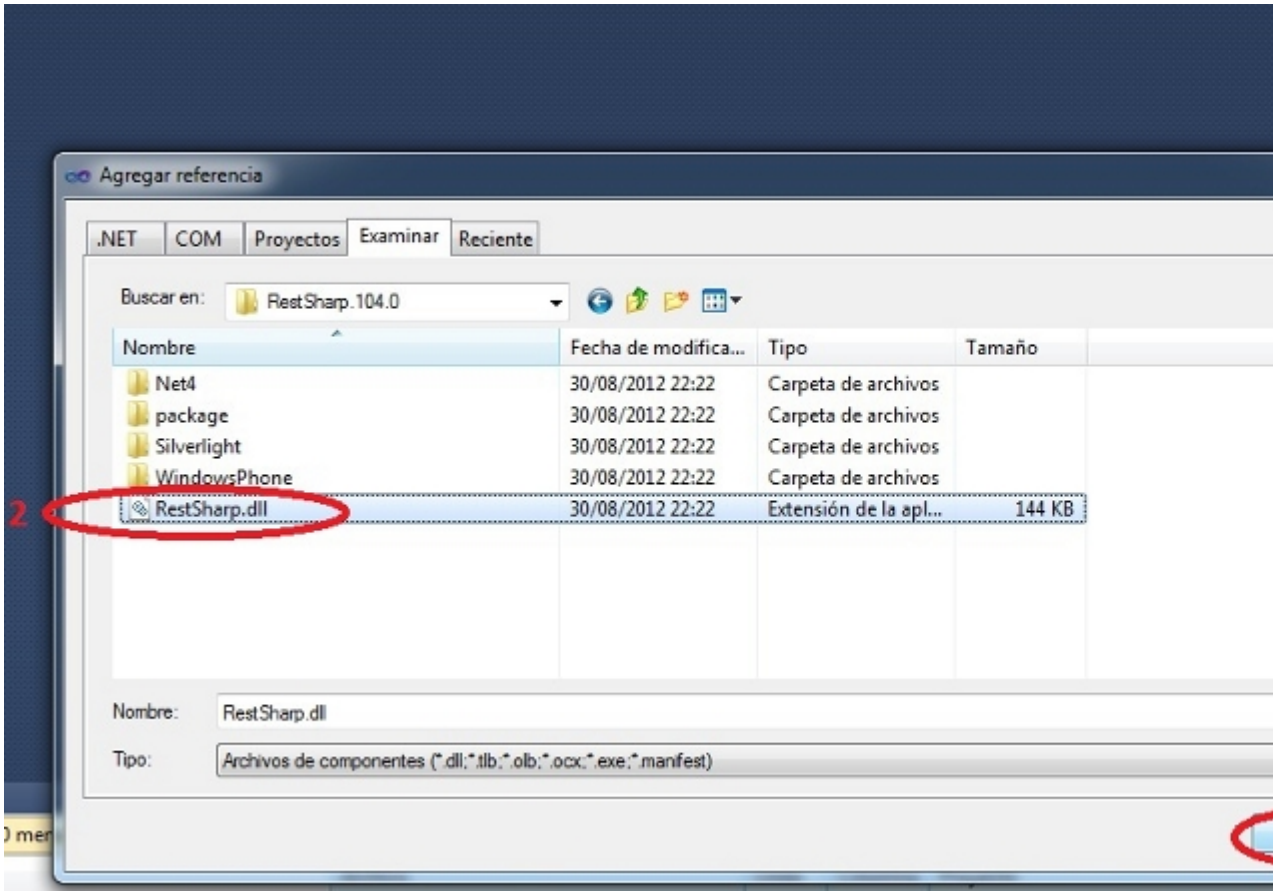
> More information about RestSharp at http://restsharp.org/.

Visual studio project configuration

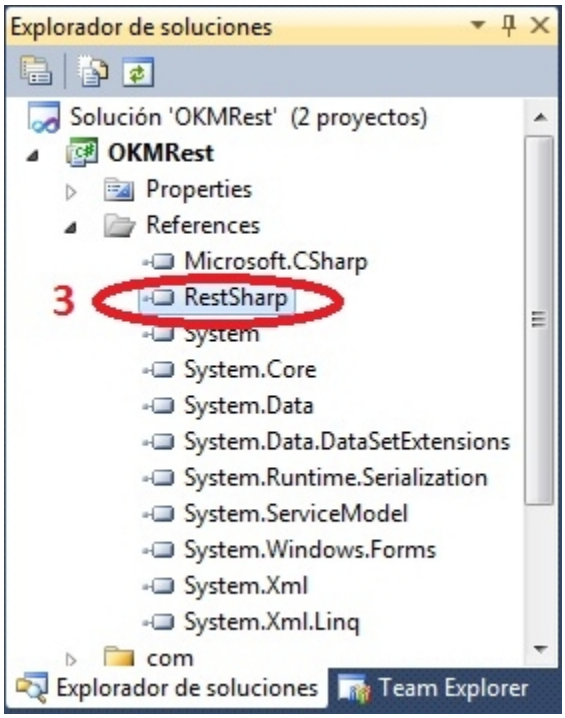
- Download the "**RestSharp.dll**" from here.
- Right click on "**References**" and in contextual menu choose "**add reference**".



- 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, pass

            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, pass

            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, passwo

            try {
                System.Console.WriteLine(repositoryImpl.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();
        }
    }
}
```

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	The **com.openkm.OKMWebservicesFactory** that returns an **com.openkm.OKMWebservices** object which implements all interfaces. `OKMWebservices ws = OKMWebservicesFactory.newInstance(host, username, password);`
com.openkm.sdk4csharp.bean	Contains all classes result of unmarshalling REST objects.

com.openkm.sdk4csharp.definition	All interface classes: • 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	All interface implementation classes: • 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	A couple of utilities. The class com.openkm.sdk4csharp.util.ISO8601 should be used to set and parse metadata date fields.
com.openkm.sdk4csharp.exception	All exception classes.

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, pass

            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, pass

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

```
                    Console.WriteLine("{0} -> {1}", kvp.Key, kvp.Value);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

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, pass

            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, pass

            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, pass
```

```
            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, pass

            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, pass

            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, pass

            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.
The parameter recursive only makes sense when the nodeId is a folder or record node. When 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;
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, pass
```

```
            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.
The parameter recursive only makes sense when the nodeId is a folder or record node. When 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, pass

            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.
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, pass

            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, pass

            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());
            }
        }
    }
}
```

createUser

Description:

Method	Return values	Description

createUser(String user, String password, String email, String name, Boolean active)	**void**	Create a new 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, pass
            try
            {
                ws.createUser("test", "password.2016", "some@mail.com", "User Name",
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

deleteUser

Description:

Method	Return values	Description
deleteUser(String user)	**void**	Delete 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, pass
            try
            {
                ws.deleteUser("test");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

updateUser

Description:

Method	Return values	Description
updateUser(String user, String password, String email, String name, Boolean active)	**void**	Update 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, pass
            try
            {
                ws.updateUser("test", "newpassword", "some@mail.com", "Test", false);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

createRole

Description:

Method	Return values	Description
createRole(String role, boolean active)	**void**	Create a new 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, pass
            try
            {
                ws.createRole("ROLE_TEST", true);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

deleteRole

Description:

Method	Return values	Description
deleteRole(String role)	**void**	Delete 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, pass
            try
            {
                ws.deleteRole("ROLE_TEST");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

updateRole

Description:

Method	Return values	Description
updateRole(String role, boolean active)	**void**	Update 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, pass
            try
            {
                ws.updateRole("ROLE_TEST", true);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

assignRole

Description:

Method	Return values	Description
assingRole(String user, String role)	**void**	Assign a role 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, pass
            try
            {
                ws.assignRole("test", "ROLE_USER");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

removeRole

Description:

Method	Return values	Description
removeRole(String user, String role)	**void**	Remove a role from 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, pass
            try
            {
                ws.removeRole("test", "ROLE_USER");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

changeSecurity

Description:

Method	Return values	Description
changeSecurity(ChangeSecurity changeSecurity)	**void**	Change the security 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, pass
            try
            {
                ChangeSecurity cs = new ChangeSecurity();
                cs.nodeId = "7f3e8715-945d-4d0f-a66c-444c2b0c6dcd";
                GrantedRoleList grList = new GrantedRoleList();
                GrantedRole gr = new GrantedRole();
                gr.role = "ROLE_TEST";
                gr.permissions = Permission.READ | Permission.WRITE;
                grList.grantedRoles.Add(gr);
                cs.grantedRolesList = grList;
                ws.changeSecurity(cs);
            }
            catch (Exception e)
            {
```

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

login

Description:

Method	Return values	Description
login()	**void**	Simulate first login process.

When user login into the application the first time, is called internally a method what creates user specific folders, like /okm:trash/userId etc.

From API point of view, this method should be only executed first time user accessing from API, for creating a specific user folder structure.

Otherwise you can get errors, for example, PathNotExistsException /okm:trash/userId when deleting objects, because the folders have not been created.

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

Bookmarck 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

getUserBookmarks

Description:

Method	Return values	Description
getUserBookmarks()	**List<Bookmark>**	Returns a list of all the bookmarks of a user.

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
            {
                List<Bookmark> list = ws.getUserBookmarks();
                foreach(Bookmark bookmark in list)
                {
                    Console.WriteLine(bookmark.toString());
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
```

```
            }
        }
    }
}
```

createBookmark

Description:

Method	Return values	Description
createBookmark(String nodeId, String name)	**void**	Create a new bookmark.
The value of the **nodeId** is the UUID or PATH of the node (document, folder, mail or record).		

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
            {
                 Bookmark bookmark = ws.createBookmark("/okm:root/myDoc.docx", "test
                 Console.WriteLine(bookmark.toString());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

renameBookmark

Description:

Method	Return values	Description
renameBookmark(int bookmarkId, String name)	Bookmark	Rename a bookmark

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
            {
                int bookmarkId = 1;
                Bookmark bookmark = ws.renameBookmark(bookmarkId, "new name bookmark"
                Console.WriteLine(bookmark.toString());
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

deleteBookmark

Description:

Method	Return values	Description
deleteBookmark(int bookmarkId)	**void**	Delete a bookmark.

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
            {
                 int bookmarkId = 1;
                 ws.deleteBookmark(bookmarkId);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getBookmark

Description:

Method	Return values	Description
getBookmark(int bookmarkId)	Bookmark	get a bookmark

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
            {
                 int bookmarkId = 1;
                 Bookmark bookmark = ws.getBookmark(bookmarkId);
                 Console.WriteLine(bookmark.toString());
            }
            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, pass
            try
            {
                BeanHelper beanHelper = new BeanHelper();
                FileStream fileInput = new FileStream("C:\\Documents\\test.docx", Fil
                Stream stream = ws.doc2pdf(fileInput, "test.docx");
                Byte[] data = beanHelper.ReadToEnd(stream);
                FileStream fileStream = new FileStream("C:\\Documents\\out.pdf", File
                foreach (byte b in data)
                {
                    fileStream.WriteByte(b);
                }
                fileStream.Close();
```

```
                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.

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

The parameter dstMimeType is the expected document MIME TYPE result.

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.

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.

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, pass
            try
            {
                FileStream filestream = new FileStream("C:\\Documents\\test.png", Fil
                List<string> param = new List<string>();
                param.Add("-resize 50% ${fileIn} ${fileOut}");
                Stream stream = ws.imageConvert(filestream, "test.png", param, "image
                BeanHelper beanHelper = new BeanHelper();
                Byte[] data = beanHelper.ReadToEnd(stream);
                FileStream fileStream = new FileStream("C:\\Documents\\test.png", Fil
                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, pass

            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, pass

            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, pass

            try
            {
                 System.Console.WriteLine(ws.getDocumentProperties("/okm:root/logo.pn
            } 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 vers

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()).intVal
```

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, pass

            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 specif 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()).intVal
```

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, pass

            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, pass

            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, pass

            try
            {
                 Document doc = ws.renameDocument("f123a950-0329-4d62-8328-0ff500fd42
            } 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, pass

            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, pass

            try
            {
                ws.checkout("/okm:root/logo.png");
                // At this point the document is locked for other users except for th
            } 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, pass

            try
            {
                // At this point the document is locked for other users except for th
                ws.cancelCheckout("/okm:root/logo.png");
                // At this point other users are allowed to execute a checkout and mo
            } 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, pass

            try
            {
                // At this point the document is locked for other users except for th
                ws.forceCancelCheckout("/okm:root/logo.png");
                // At this point other users are allowed to execute a checkout and mo
            } 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, pass

            try
            {
                System.Console.WriteLine("Is the document checkout:"+ws.isCheckedOut(
            } 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, pass

            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, pass

            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.

Only the user who locked the document is allowed to unlock.

A locked document can not be modified by other users.

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
            {
                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.

Only the user who locked the document is allowed to unlock.

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, pass

            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, pass

            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, pass

            try
            {
                System.Console.WriteLine("Is document locked:" + ws.isLocked("/okm:ro
            } 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, pass
```

```
            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, pass

            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, pass

            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, pass

            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, pass

            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, pass

            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, pass

            try
            {
                String[] UNITS = new String[] { "B", "KB", "MB", "GB", "TB", "PB", "E
                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,
                }

                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, pass

            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, pass

            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";
            OKMWebservice 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());
            }
        }
    }
}
```

extendedDocumentCopy

Description:

Method	Return values	Description
extendedDocumentCopy(String docId, String dstId, String name, boolean categories, boolean keywords, boolean propertyGroups, boolean notes, boolean wiki)	**void**	Copies a document with associated data into a 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. By default, only the binary data and the security grants, the metadata, keywords, etc. of the document are not copied. Additional: • When the category parameter is true the original values of the categories will be copied. • When the keywords parameter is true the original values of the keywords will be copied. • When the property groups parameter is true the original values of the metadata groups will be copied. • When the notes parameter is true the original values of the notes will be copied. • When the wiki parameter is true the original		

values of the wiki will be copied.

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, pass

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

createFromTemplate

Description:

Method	Return values	Description
Document createFromTemplate(String docId, String dstPath, bool categories, bool keywords, bool propertyGroups, bool notes, bool wiki, Dictionary<String, String> properties)	**Document**	Creates a new document from the template and returns a Document.

The dstPath must be a folder.

When the template use metadata groups to fill in fields, then these values are mandatory and must be set into properties parameter.

For more information about Templates and metadata check: Creating templates.

Additional:

- When category parameter is true the original values of the categories will be copied.
- When keywords parameter is true the original values of the keywords will be copied.
- When property Groups parameter is true the original values of the metadata groups will be copied.
- When notes parameter is true the original values of the notes will be copied.
- When wiki parameter is true the original values of the wiki will be copied.

Example:

The example below is based on Creating PDF template sample.

```
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, pass

            try
            {
                Dictionary<String, String> properties = new Dictionary<String, String
                properties.Add("okp:tpl.name", "Some name");
                DateTime date = DateTime.Now;
                // Value must be converted to String ISO 8601 compliant
                properties.Add("okp:tpl.bird_date", ISO8601.formatBasic(date));
                properties.Add("okp:tpl.language", "java");
                // create from template
                Document doc2 = ws.createFromTemplate("fc638b93-0072-49ac-ad88-d4eeb0

            } 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 initializated. 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, pass

            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, pass

            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, pass

            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, pass

            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, pass

            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, pass

            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, pass

            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, pass

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

```
                // 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, pass

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

copyFolder

Description:

Method	Return values	Description
copyFolder(String fldId, String dstId)	**void**	Copies a folder into a 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 folder will preserve the same name.

Only the security grants are copied to the destination, the metadata, keywords, etc. of the folder are not copied.

See "**extendedFolderCopy**" 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, pass

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

extendedFolderCopy

Description:

Method	Return values	Description
extendedFolderCopy(String fldId, String dstId, boolean categories, boolean keywords, boolean propertyGroups, boolean notes, boolean wiki)	**void**	Copies a folder with associated data into some folder or record.
The values of the dstId parameter should be a folder or record UUID or path.		

By default, only the binary data and the security grants, the metadata, keywords, etc. of the folder are not copied.

Additional:

- When the category parameter is true the original values of the categories will be copied.
- When the keywords parameter is true the original values of the keywords will be copied.
- When the property groups parameter is true the original values of the metadata groups will be copied.
- When the node parameter is true the original values of the notes will be copied.
- When wiki parameter is true the original values of the wiki will be copied.

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, pass

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

getContentInfo

Description:

Method	Return values	Description
getContentInfo(String fldId)	**ContentInfo**	Returne an object ContentInfo with information about folder.
The ContentInfo object retrieves information about:		

- The number of folders.
- The number of documents.
- The number of records.
- The number of emails.
- The size in bytes of all objects into the 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, pass

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

purgeFolder

Description:

Method	Return values	Description
purgeFolder(String fldId)	**void**	The folder is definitely removed from the repository.

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

When a folder is purged only will be able to be restored from a previously repository backup. The purge action remove the folder 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, pass

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

createMissingFolders

Description:

Method	Return values	Description
createMissingFolders(String fldPath)	**void**	Create missing folders.

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, pass
            try
            {
               ws.createMissingFolders("/okm:root/missingfld1/missingfld2/missingfld3
```

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

Mail samples

Basics

On almost methods you'll see parameter named "**mailId**". The value of this parameter can be a valid mail **UUID** or **path**.

Example of fldId:

- Using UUID -> "**064ff51a-b815-4f48-a096-b4946876784f**";
- Using path -> "**/okm:root/2937b81d-0b10-4dd0-a426-9acbd80be1c9-some subject**"

Methods

createMail

Description:

Method	Return values	Description
createMail(Mail mail)	**Mail**	Creates a new mail and return as a result an object Mail.

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

Other mandatory variables:

- size (mail size in bytes).
- from (mail from the account).
- reply, to, cc, bcc (mail accounts are optional).
- sendDate (date when mail was sent).
- receivedDate (date when was received).
- subject (mail subject).
- content (the mail content).
- mimeType (HTML or text mime type).
- headers (the mail headers are optional).
- raw (the mail raw are optional).
- origin (msg, eml, api, pop3, imap origin)
- title (mail title)

Mails account allowed formats:

- "\"John King\" <jking@mail.com>"
- "<jking@mail.com>"

Mail path allowed is:

MSGID + "-" + sanitized(subject).

MIME types of values:

- Mail.MIME_TEXT for text mail format.
- Mail.MIME_HTML for html mail format.

Origin values:

- Mail.ORIGIN_MSG for .msg mail format.
- Mail.ORIGIN_EML for .eml mail format.
- Mail.ORIGIN_API for mail format created from API.
- Mail.ORIGIN_POP3 for mail imported from pop3 account.
- Mail.ORIGIN_IMAP for mail imported from imap account.

The other variables of Mail (mail) will not take any effect on mail creation.

Example:

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

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
            {
                    Mail mail = new Mail();

                    // Mail path = msgId + escaped(subject)
                    String msgId = "2937b81d-0b10-4dd0-a426-9acbd80be1c9";
```

```
                    String subject = "some subject";
                    String mailPath = "/okm:root/" + msgId + "-" + escape(subject);
                    mail.path = mailPath;

                    // Other format for mail "some name <no_reply@openkm.com>"
                    mail.from = "<no_reply@openkm.com>";
                    String[] to = new String[] { "anonymous@gmail.com" };
                    mail.to = to;

                    // You should set real dates
                    mail.sentDate = DateTime.Now;
                    mail.receivedDate = DateTime.Now;
                    mail.content = "some content";
                    mail.mimeType = Mail.MIME_TEXT;
                    mail.subject = subject;
                    mail.origin = Mail.ORIGIN_API;

                    // Get only as an approximation of real size for these sample
                    mail.size = mail.toString().Length;
                    ws.createMail(mail);
                }
                catch (Exception e)
                {
                    System.Console.WriteLine(e.ToString());
                }
            }


            private static String escape(String name)
            {
                String ret = cleanup(name);
                // Fix XSS issues
                ret = WebUtility.HtmlEncode(name);
                return ret;
            }

            private static String cleanup(String name)
            {
                String ret = name.Replace("/", "");
                ret = ret.Replace("*", "");
                ret = ret.Replace("\\s+", " ").Trim();
                return ret;
            }
        }
    }
```

getMailProperties

Description:

Method	Return values	Description
getMailProperties(String mailId)	**Mail**	Returns the mail 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, pass

            try
            {
                System.Console.WriteLine(ws.getMailProperties("50b7a5b9-89d2-430e-bbc
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

deleteMail

Description:

Method	Return values	Description
deleteMail(String mailId)	**void**	Deletes a mail.

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, pass

            try
            {
                ws.deleteMail("50b7a5b9-89d2-430e-bbc9-6a6e01662a71");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
```

```
            }
        }
    }
```

purgeMail

Description:

Method	Return values	Description
purgeMail(String mailId)	**void**	Folder is definitely removed from repository.

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

When a mail is purged it will only be able to be restored from a previously repository backup. The purge action removes the mail 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, pass

            try
            {
                ws.purgeMail("50b7a5b9-89d2-430e-bbc9-6a6e01662a71");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

renameMail

Description:

Method	Return values	Description
renameMail(String mailId, String newName)	**void**	Renames a mail.

From OpenKM frontend UI the subject is used to show the mail name at file browser table. That means the change will take effect internally on mail path, but will not be appreciated from default UI.

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, pass

            try
            {
                ws.renameMail("50b7a5b9-89d2-430e-bbc9-6a6e01662a71", "new name");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

moveMail

Description:

Method	Return values	Description
moveMail(String mailId, String dstId)	**void**	Moves a mail into a 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, pass

            try
            {
                ws.moveMail("50b7a5b9-89d2-430e-bbc9-6a6e01662a71", "/okm:root/tmp");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

copyMail

Description:

Method	Return values	Description
public void copyMail(String mailId, String dstId, String newName)	**void**	Copies mail into a folder or record.

The values of the dstId parameter should be a folder or record UUID or path.

When parameter newName value is null, mail will preserve the same name.

Only the security grants are copied to the destination, the metadata, keywords, etc. of the folder are not copied.

See "**extendedMailCopy**" 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, pass

            try
            {
                ws.copyMail("50b7a5b9-89d2-430e-bbc9-6a6e01662a71", "/okm:root/tmp",
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

extendedMailCopy

Description:

Method	Return values	Description
extendedMailCopy(String mailId, String dstId, boolean categories, boolean keywords, boolean propertyGroups, boolean notes, boolean wiki, String newName)	**void**	Copies mail with associated data into a folder or record.

The values of the dstId parameter should be a folder or record UUID or path.

By default, only the binary data and the security grants, the metadata, keywords, etc. of the folder are not copied.

Additional:

- When the category parameter is true the original values of the categories will be copied.
- When the keywords parameter is true the original values of the keywords will be copied.
- When the property groups parameter is true the original values of the metadata groups will be copied.
- When the notes parameter is true the original values of the notes will be copied.
- When the wiki parameter is true the original values of the wiki will be copied.
- When newName is set the mail name is renamed.

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, pass
            try
            {
                ws.extendedMailCopy("fe51797c-be0b-4076-8f4b-b4fffe8fd2bc", "/okm:roo
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getMailChildren

Description:

Method	Return values	Description
getMailChildren(String fldId)	**List<Mail>**	Returns a list of all mails which their parent is fldId.

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
            {
                foreach (Mail mail in ws.getMailChildren("/okm:root/"))
                {
                    System.Console.WriteLine(mail);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

isValidMail

Description:

Method	Return values	Description
isValidMail(String mailId)	**Boolean**	Returns a boolean that indicates if the node is a mail or not.

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
            {
                // Return false
                ws.isValidMail("/okm:root/logo.png");

                // Return true
                ws.isValidMail("50b7a5b9-89d2-430e-bbc9-6a6e01662a71");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getMailPath

Description:

Method	Return values	Description
getMailPath(String uuid)	**String**	Converts a mail 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, pass

            try
            {
                System.Console.WriteLine(ws.getMailPath("50b7a5b9-89d2-430e-bbc9-6a6e
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

importEml

Description:

Method	Return values	Description
importEml(String dstId, String title, FileStream fs)	**Mail**	Import a mail in EML format.
The values of the dstId parameter should be a folder or record UUID or path. The dstId parameter indicates where the mail will be stored in the repository after is sent.		

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, pass
            try
            {
                FileStream filestream = new FileStream("C:\\Documents\\test.eml", Fil
                Mail mail = ws.importEml("/okm:root/test", "Test", filestream);
                System.Console.WriteLine(mail);
                filestream.Close();
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

importMsg

Description:

Method	Return values	Description
importMsg(String dstId, String title, FileStream fs)	**Mail**	Import a mail in MSG format.
The values of the dstId parameter should be a folder or record UUID or path. The dstId parameter indicates where the mail will be stored in the repository after is sent.		

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, pass
            try
            {
                FileStream filestream = new FileStream("C:\\Documents\\test.msg", Fil
                Mail mail = ws.importMsg("/okm:root/test", "Test", filestream);
                System.Console.WriteLine(mail);
                filestream.Close();
            }
            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, pass

            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 Node 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, pass

            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.

> The noteId is an UUID.
>
> The object Node 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, pass

            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.

> The noteId is an UUID.
>
> The object Node 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, pass

            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, pass

            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, pass

            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, pass

            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, pass

            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, pass

            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

The parameter nodeId should be a document node.

The parameter cipherName saves information about the encryption mechanism.

This method does not perform any kind of encryption, simply marks into the database that a document is encrypted.

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, pass

            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 uncryption, simply marks into the database that a document has been uncrypted.

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, pass

            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, pass

            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 metada 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 dat
```

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, pass

            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, pass

            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, pass

            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, pass

            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, pass

            try
            {
                foreach(FormElement fElement in ws.getPropertyGroupProperties("/okm:r
                {
                    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, pass

            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	Descri
setPropertyGroupProperties(String nodeId, String grpName, List<FormElement> ofeList)	**void**	Changes the metadat node.

The grpName should be a valid Metadata group name.

Is not mandatory to set into parameter ofeList all FormElement, is enought with the formElements you wish to char

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
<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, pass

            try
            {
                // Modify with a full FormElement list
                List<FormElement> fElements = ws.getPropertyGroupProperties("/okm:roo
                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",

                // 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",
            }
            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
<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, pass

            try
            {
                Dictionary<String, String> properties = new  Dictionary<String, Strin
                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", p
            }
            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, pass

            try
            {
                System.Console.WriteLine("Have metadata group:"+ws.hasGroup("/okm:roo
            }
            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, passw

           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());
           }
        }
```

```
    }
}
```

getSuggestions

Description:

Method	Return values	Description
getSuggestions(String nodeId, String grpName, String propName)	**List<String>**	Retrieves a list of a suggested me

The propName parameter should be a Metadata Select field type.

More information at Creating your own Suggestion Analyzer and Metadata Select field.

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
<property-groups>
  <property-group label="Technology" name="okg:technology">
    <select label="Type" name="okp:technology.type" type="multiple">
      <option label="Alfa" value="t1"/>
      <option label="Beta" value="t2" />
      <option label="Omega" value="t3" />
    </select>
    <select label="Language" name="okp:technology.language" type="simple">
      <option label="Java" value="java"/>
      <option label="Python" value="python"/>
      <option label="PHP" value="php" />
    </select>
    <input label="Comment" name="okp:technology.comment"/>
    <textarea label="Description" name="okp:technology.description"/>
    <input label="Link" type="link" name="okp:technology.link"/>
  </property-group>
</property-groups>
```

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, pass

            try
            {
                foreach(String value in ws.getSuggestions("/okm:root/logo.png","okg:t
                {
                     System.Console.WriteLine(value);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

registerDefinition

Description:

Method	Return values	Description
registerDefinition(InputStream is)	**void**	Sets the XML Metadata groups definition into the repository..

The method can only be executed by superuser grants (ROLE_ADMIN member user).

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:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, pass

            try
            {
                 FileStream fs = new FileStream("E:\\PropertyGroups.xml",FileMode.Ope
                 ws.registerDefinition(fs);
```

```
                    fs.Dispose();
                }
                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, pass

            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, pass

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

getTrashFolderBase

Description:

Method	Return values	Description
getTrashFolderBase()	**Folder**	Returns the object Folder of node "/okm:trash"

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, pass

            try
            {
                System.Console.WriteLine(ws.getTrashFolderBase());
            }
            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, pass

            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, pass

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

getPersonalFolderBase

Description:

Method	Return values	Description
getPersonalFolderBase()	**Folder**	Return the object Folder of node "/okm:personal"

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, pass

            try
            {
                System.Console.WriteLine(ws.getPersonalFolderBase());
            }
            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, pass

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

getMailFolderBase

Description:

Method	Return values	Description
getMailFolderBase()	**Folder**	Returns the object Folder of node "/okm:mail"

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, pass

            try
            {
                System.Console.WriteLine(ws.getMailFolderBase());
            }
            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, pass

            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, pass

            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, pass
            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, pass
            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, pass

            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, pass

            try
            {
                 System.Console.WriteLine("Exists node:" + ws.hasNode("064ff51a-b815-
            }
            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, pass

            try
            {
                 System.Console.WriteLine(ws.getNodePath("e339f14b-4d3a-489c-91d3-05e
            }
```

```
            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, pass

            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, pass

            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.

The test.sql script used in the sample below:

```
SELECT NBS_UUID, NBS_NAME FROM OKM_NODE_BASE LIMIT 10;
```

The SQL script can only contain a single SQL 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;
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, pass

            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, pass
            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.

The local script - test.bsh - used in the sample below:

```
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";
```

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, pass

            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 use 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, pass

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

copyAttributes

Description:

Method	Return values	Description
copyAttributes(String nodeId, String dstId, boolean categories, boolean keywords, boolean propertyGroups, boolean notes, boolean wiki)	**void**	Copy attributes from a node to other.

The values of the dstId parameter should be a node UUID or path.

- When the category parameter is true the original values of the categories will be copied.
- When the keywords parameter is true the original values of the keywords will be copied.

- When the property groups parameter is true the original values of the metadata groups will be copied.
- When the notes parameter is true the original values of the notes will be copied.
- When the wiki parameter is true the original values of the wiki will be copied.

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, pass

            try
            {
                ws.copyAttributes("/okm:root/invoice.pdf", "/okm:root/cloned_invoice.
            }
            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.	Available values: • QueryParams.DOCUMENT • QueryParams.FOLDER • QueryParams.MAIL By default the value is set to QueryParams.DOCUMENT. For searching documents and folders use value: `(QueryParams.DOCUMENT | QueryParams.FOLDER`
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, pass

            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, pass

            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, pass

            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, pass

            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.

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.

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, pass

            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.

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.

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, pas
```

```
            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.

The uuid is a document UUID.

The max value is used to limit the number of results returned.

The method can only be used with documents.

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, pass

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

                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, pass

            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, pass

            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.

Ir 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, pass

            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, pass

            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, pass

            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, pass

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

findByQuery

Description:

Method	Return values	Description
findByQuery(String query)	**List<QueryResult>**	Returns a list of results filtered by the query parameter.

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 sintaxis at Lucene query syntax.

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
            {
                 List<QueryResult> rs= ws.findByQuery("keyword:test AND name:t*.pdf")

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

findByQueryPaginated

Description:

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

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 sintaxis at Lucene query syntax.

The parameter "limit" and "offset" allows 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.

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, pass

            try
            {
                 ResultSet rs = ws.findByQueryPaginated("name:grial", 0, 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());
            }
        }
    }
}
```

Workflow samples

For most examples, it has been used the Purchase workflow sample.

Methods

registerProcessDefinition

Description:

Method	Return values	Description
registerProcessDefinition(FileStream fs)	**void**	Registers a new workflow.

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:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, pass

            try
            {
                FileStream fs = new FileStream("E:\\Purchase.par",FileMode.Open);
                ws.registerProcessDefinition(fs);
                fs.Dispose();
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

deleteProcessDefinition

Description:

Method	Return values	Description

deleteProcessDefinition(long pdId)	**void**	Deletes a workflow.
The parameter pdId value is a valid workflow process definition.		

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:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, pass

            try
            {
                long pdId = 5; // Valid workflow process definition
                ws.deleteProcessDefinition(pdId);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getProcessDefinition

Description:

Method	Return values	Description
getProcessDefinition(long pdId)	**void**	Returns a workflow process definition.
The parameter pdId value is a valid workflow process definition.		

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, pass

            try
            {
                long pdId = 1; // Valid workflow process definition
                ProcessDefinition pd = ws.getProcessDefinition(pdId);
                System.Console.WriteLine(pd);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

runProcessDefinition

Description:

Method	Return values	Description
runProcessDefinition(long pdId, String uuid, List<FormElement> values)	**ProcessInstance**	Executes a workflow on some node.

The parameter pdId value is a valid workflow process definition.

The parameter uuid can be any document, mail, folder or record UUID.

The parameter values are form element values needed for starting the workflow (not all workflows need form values for starting).

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, pass
            try
            {
                long pdId = 8041; // Some valid workflow process definition id
                List<FormElement> feList = new List<FormElement>();
                Input price = new Input();
                price.name = "price";
                price.value = "1000";
                feList.Add(price);
                TextArea textArea = new TextArea();
                textArea.name = "description";
                textArea.value = "some description here";
                feList.Add(textArea);
                ws.runProcessDefinition(pdId, "7aa523e0-06cf-4733-b8c8-cc74d8b2716d",
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

findAllProcessDefinitions

Description:

Method	Return values	Description
findAllProcessDefinitions()	**List<ProcessDefinition>**	Retrieves a list of all registered workflows definitions.

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, pass

            try
            {
                foreach (ProcessDefinition pd in ws.findAllProcessDefinitions())
```

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

findProcessInstances

Description:

Method	Return values	Description
findProcessInstances(long pdId)	**List<ProcessInstance>**	Retrieves a list of all process instances of some registered workflows definition.
The parameter pdId value is a valid workflow process definition.		

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, pass

            try
            {
                // Get all workflow definitions
                foreach (ProcessDefinition pd in ws.findAllProcessDefinitions())
                {
                    System.Console.WriteLine("WF definition: "+pd);
                    // Get all process of some workflow definition
                    foreach (ProcessInstance pi in ws.findProcessInstances(pd.id))
                    {
                         System.Console.WriteLine("PI: "+pi);
                    }
                }
            }
            catch (Exception e)
            {
```

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

findLatestProcessDefinitions

Description:

Method	Return values	Description
findLatestProcessDefinitions()	**List<ProcessDefinition>**	Retrieves a list of the last workflows definitions.

Can be several versions of the same workflow registered.

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, pass

            try
            {
                    // Get all latest workflow definitions
             foreach (ProcessDefinition pd in ws.findLatestProcessDefinitions())
                    {
                        System.Console.WriteLine("WF definition: " + pd);
             }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

findLastProcessDefinition

Description:

Method	Return values	Description
findLastProcessDefinition(String name)	**ProcessDefinition**	Retrieves the last workflow definition of some specific workflow.

The parameter name identifies a specific workflow definitions group.

Several workflow definition versions can have the same name.

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, pass

            try
            {
                ProcessDefinition pd = ws.findLastProcessDefinition("purchase");
                System.Console.WriteLine("WF definition: " + pd);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getProcessInstance

Description:

Method	Return values	Description
getProcessInstance(long piId)	**ProcessInstance**	Returns the process instance.

The parameter piId is a valid process instance id.

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, pass

            try
            {
                long piId = 5868; // Some valid process instance id
         ProcessInstance pi = ws.getProcessInstance(piId);
                System.Console.WriteLine("PI: " + pi);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

findUserTaskInstances

Description:

Method	Return values	Description
findUserTaskInstances()	**List<TaskInstance>**	Retrieves a list of task instances assigned to the user.

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, pass

            try
            {
                // Get all user task intances
                foreach(TaskInstance ti in ws.findUserTaskInstances())
                {
                     System.Console.WriteLine(ti);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

findTaskInstances

Description:

Method	Return values	Description
findTaskInstances(long piId)	**List<TaskInstance>**	Retrieves a list of task instances of some process instance id.
The parameter piId is a valid process instance id.		

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, pass

            try
            {
                // Get all task instances of some process instance
                long piId = 8108; // Some valid process instance id
```

```
                foreach (TaskInstance ti in ws.findTaskInstances(piId))
                {
                    System.Console.WriteLine(ti);
                }
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

setTaskInstanceValues

Description:

Method	Return values	Description
setTaskInstanceValues(long tiId, String transName, List<FormeElement> valuess)	**void**	Retrieves a list of task intances of some process instance id.

The parameter tiId is a valid task instance id.

The parameter transName is the chosen transaction.

The parameter values are form element values needed for starting the workflow (not all workflow tasks need form 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, pass
            try
            {
                long tiId = 8110; // Some valid task instance id
                List<FormElement> feList = new List<FormElement>();
                ws.setTaskInstanceValues(tiId, "approve", feList);
            }
            catch (Exception e)
```

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

getTaskInstance

Description:

Method	Return values	Description
getTaskInstance(long tiId)	**TaskInstance**	Returns a task instance.
The parameter tiId is a valid task instance id.		

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, pass

            try
            {
                long tiId = 8110; // Some valid task instance id
                TaskInstance ti = ws.getTaskInstance(tiId);
                System.Console.WriteLine(ti);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

startTaskInstance

Description:

Method	Return values	Description

startTaskInstance(long tiId)	**void**	Starts a task instance.
The parameter tiId is a valid task instance id.		

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

            try
            {
                long tiId = 8110; // Some valid task instance id
                ws.startTaskInstance(tiId);
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

setTaskInstanceActorId

Description:

Method	Return values	Description
setTaskInstanceActorId(long tiId, String actorId)	**void**	Starts a task instance.
The parameter tiId is a valid task instance id. The parameter actorId must be a valid OpenKM userId.		

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

            try
            {
                long tiId = 8110; // Some valid task instance id
                ws.setTaskInstanceActorId(tiId, "okmAdmin");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

endTaskInstance

Description:

Method	Return values	Description
endTaskInstance(long tiId, String transName)	**void**	Starts a task instance.
The parameter tiId is a valid task instance id. The parameter transName is a transaction 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:8180/OpenKM";
            String username = "okmAdmin";
            String password = "admin";
            OKMWebservice ws = OKMWebservicesFactory.newInstance(host, username, pass
            try
            {
                long tiId = 8110; // Some valid task instance id
                ws.endTaskInstance(tiId, "end");
            }
            catch (Exception e)
            {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```

getProcessDefinitionForms

Description:

Method	Return values	Description
getProcessDefinitionForms(long pdId)	**Dictionary<String, List<FormElement>>**	Return a map with all the process definition forms.
The parameter pdId value is a valid workflow process definition.		

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using com.openkm.sdk4csharp;
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, pass
            try
            {
                 Dictionary<String, List<FormElement>> forms = ws.getProcessDefinitio
                 foreach (KeyValuePair<String, List<FormElement>> pair in forms)
                 {
                    System.Console.WriteLine("Key:"+ pair.Key);
                        foreach (FormElement fe in pair.Value)
                        {
                            System.Console.WriteLine("Fe:"+ fe);
                        }
                 }
```

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

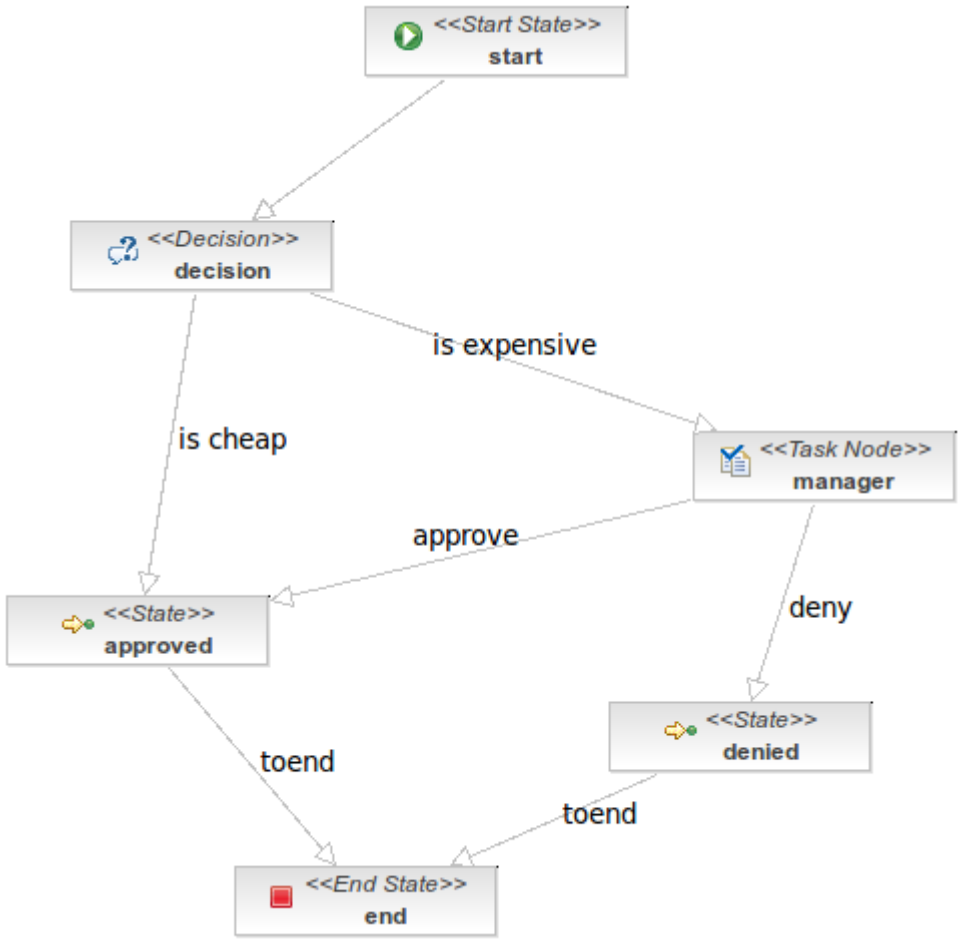
Purchase workflow sample

The tasks will be assigned to a user called "manager" so you need to create this user and log as such to see the task assignment.

Also you can assign this task to another user from the process instance workflow administration.

Download the Purchase.par file.

Process image



Process definition

```
<?xml version="1.0" encoding="UTF-8"?>
<process-definition xmlns="urn:jbpm.org:jpdl-3.2" name="purchase">
  <start-state name="start">
    <transition to="decision"></transition>
  </start-state>

  <decision name="decision">
    <transition to="approved" name="is cheap">
      <condition expression="#{price.value &lt;= 500}"></condition>
```

```
    </transition>
    <transition to="manager" name="is expensive">
      <condition expression="#{price.value &gt; 500}"></condition>
    </transition>
  </decision>

  <task-node name="manager">
    <task name="evaluate price">
      <description>The manager may deny purchase or go ahead.</description>
      <assignment actor-id="manager"></assignment>
    </task>
    <transition to="denied" name="deny"></transition>
    <transition to="approved" name="approve"></transition>
  </task-node>

  <state name="approved">
    <description>The purchase has been approved.</description>
    <timer duedate="15 seconds" name="approved timer" transition="toend">
      <script>print(&quot;From APPROVED Go to END&quot;);</script>
    </timer>
    <transition to="end" name="toend"></transition>
  </state>

  <state name="denied">
    <description>The purchase has been denied.</description>
    <timer duedate="15 seconds" name="denied timer" transition="toend">
      <script>print(&quot;From DENIED Go to END&quot;);</script>
    </timer>
    <transition to="end" name="toend"></transition>
  </state>

  <end-state name="end"></end-state>
</process-definition>
```

Process handlers

None.

Form definition

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.0//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.0.dtd">
<workflow-forms>
  <workflow-form task="run_config">
    <input label="Purchase price" name="price" />
    <textarea label="Purchase description" name="description" />
    <button name="submit" label="Submit" />
  </workflow-form>
  <workflow-form task="evaluate price">
    <input label="Purchase price" name="price" data="price" readonly="true" />
    <textarea label="Purchase description" name="description" data="description" read
    <button name="approve" label="Approve" transition="approve"/>
    <button name="deny" label="Deny" transition="deny"/>
  </workflow-form>
</workflow-forms>
```

Notification samples

Basics

On most methods you'll see parameter named "**nodeId**". 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

notify

Description:

Method	Return values	Description
notify(String nodeId, List<String> users, List<String> mails, String message, boolean attachment)	**void**	Send a mail notification.

The parameter nodeId is the UUID or PATH of the node (document, folder, mail, or record).

The parameter users are a set of OpenKM users to be notified.

The parameter mails are a set of email addresses - usually external mails - to be notified.

The parameter message is the content body of the mail.

When the attachment value is true the node is attached to the mail.

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, pass

            try
            {
                 List<String> users = new List<String>();
                 users.Add("test");

                 List<String> mails = new List<String>();
                 mails.Add("test@none.com");

                 String message = "Body of the message";

                 ws.notify(nodeId, users, mails, message, false);
            } catch (Exception e) {
                System.Console.WriteLine(e.ToString());
            }
        }
    }
}
```