

NextLabs CADRMX Client SDK

v1.0

1 CONFIDENTIALITY NOTICE	1
2 Introduction	3
2.1 Overview	3
3 Platform and Environment Support	5
3.1 Operating System	5
3.2 Runtime Environment	5
3.3 Development Environment	5
4 Getting Started	7
4.1 Setting up Development Environment	7
4.2 Using APIs to implement Rights Management features	7
4.2.1 Example 1: Intialize RMX	7
4.2.2 Example 2: Handle RPM folder	8
4.2.3 Example 3: Handle file protection	8
4.3 General Notes	8
5 Class Index	11
5.1 Class List	11
6 File Index	13
6.1 File List	13
7 Class Documentation	15
7.1 IRMXInstance Class Reference	15
7.1.1 Detailed Description	15
7.1.2 Member Function Documentation	16
7.1.2.1 IsInitFinished()	16
7.1.2.2 GetLoginUser()	16
7.1.2.3 AddRPMDir()	16
7.1.2.4 RemoveRPMDir()	17
7.1.2.5 IsRPMFolder()	17
7.1.2.6 RegisterApp()	18
7.1.2.7 UnregisterApp()	18
7.1.2.8 IsAppRegistered()	19
7.1.2.9 NotifyRMXStatus()	19
7.1.2.10 EditCopyFile()	19
7.1.2.11 EditSaveFile()	20
7.1.2.12 RPMCopyFile()	21
7.1.2.13 RPMDDeleteFile()	21
7.1.2.14 GetFileRights()	21
7.1.2.15 GetFileStatus()	22
7.1.2.16 ReadFileTags()	22

7.1.2.17 SetViewOveraly()	23
7.1.2.18 ClearViewOverlay()	24
7.2 IRMXUser Class Reference	24
7.2.1 Detailed Description	24
7.2.2 Member Function Documentation	24
7.2.2.1 GetName()	25
7.2.2.2 GetEmail()	25
7.2.2.3 GetUserID()	25
7.2.2.4 ProtectFile()	25
7.2.2.5 GetResourceRightsFromCentralPolicies()	26
7.2.2.6 GetDefaultPolicyBundle()	26
7.2.2.7 GetSystemProjectTenantId()	27
7.2.2.8 GetDefaultTokenGroupName()	27
7.2.2.9 MergeTags()	27
7.3 RMX_CENTRAL_RIGHT Struct Reference	28
7.3.1 Detailed Description	28
7.3.2 Member Data Documentation	28
7.3.2.1 right	28
7.3.2.2 obligations	28
7.4 RMX_EVAL_ATTRIBUATE Struct Reference	29
7.4.1 Detailed Description	29
7.4.2 Member Data Documentation	29
7.4.2.1 key	29
7.4.2.2 value	29
7.5 RMX_OBLIGATION_INFO Struct Reference	29
7.5.1 Detailed Description	30
7.5.2 Member Data Documentation	30
7.5.2.1 name	30
7.5.2.2 options	30
7.6 RMX_VIEWOVERLAY_PARAMS Struct Reference	30
7.6.1 Detailed Description	31
7.6.2 Member Data Documentation	31
7.6.2.1 hTargetWnd	31
7.6.2.2 vecTags	31
7.6.2.3 vecCtxAttrs	31
7.6.2.4 displayOffsets	31
7.7 RMXResult Class Reference	32
7.7.1 Detailed Description	32
7.7.2 Constructor & Destructor Documentation	32
7.7.2.1 RMXResult() [1/3]	32
7.7.2.2 RMXResult() [2/3]	32
7.7.2.3 ~RMXResult()	33

7.7.2.4 RMXResult() [3/3]	33
7.7.3 Member Function Documentation	33
7.7.3.1 operator=()	33
7.7.3.2 operator bool()	34
7.7.3.3 operator==()	34
7.7.3.4 operator!=()	34
7.7.3.5 GetErrCode()	35
7.7.3.6 GetErrMsgeage()	35
7.7.4 Member Data Documentation	35
7.7.4.1 m_code	35
7.7.4.2 m_message	35
8 File Documentation	37
8.1 RMXGlobal.h File Reference	37
8.1.1 Detailed Description	37
8.1.2 Function Documentation	37
8.1.2.1 RMX_GetLibVersion()	37
8.1.2.2 RMX_GetSDWLibVersion()	38
8.1.2.3 RMX_GetCurrentLoggedInUser()	38
8.1.2.4 RMX_DeleteRMXInstance()	38
8.2 RMXLInc.h File Reference	39
8.2.1 Detailed Description	39
8.3 RMXLInstance.h File Reference	39
8.4 RMXLResult.h File Reference	39
8.4.1 Typedef Documentation	40
8.4.1.1 RMXErrCode_t	40
8.5 RMXLTypeDef.h File Reference	40
8.5.1 Detailed Description	40
8.5.2 Enumeration Type Documentation	40
8.5.2.1 RMX_RPMFolderRelation	40
8.5.2.2 RMXFileRight	41
8.6 RMXLUser.h File Reference	42
Index	43

Chapter 1

CONFIDENTIALITY NOTICE

THIS DOCUMENT IS CONFIDENTIAL AND PROPRIETARY TO NEXTLABS, INC. AND MAY NOT BE REPRODUCED, PUBLISHED OR DISCLOSED TO OTHERS WITHOUT COMPANY AUTHORIZATION.

© 2009-2020 NextLabs, Inc. All rights reserved. The information in this document is subject to change without notice.

To provide feedback on this document, email the documentation team at techpubs@nextlabs.com.

TRADEMARKS

NextLabs, the NextLabs Logo, Compliant Enterprise, the Compliant Enterprise Logo, Deep Event Inspection, 360 Degree Enforcement, and ACPL are trademarks or registered trademarks of NextLabs, Inc. in the United States. All other brands or product names used herein are trademarks or registered trademarks of their respective owners.

LICENSE AGREEMENT

This documentation and the software described in this document are furnished under a license agreement or nondisclosure agreement. The documentation and software may be used or copied only in accordance with the Desktop for Windows of those agreements. No part of this document may be reproduced, stored in a retrieval system or transmitted in any form or by any means, either electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's use, without the prior written permission of NextLabs, Inc.

The content of this document is provided for informational and instructional use only. It is subject to change without notice, and should not be construed as a commitment by NextLabs, Inc. NextLabs, Inc. assumes no responsibility or liability for any inaccuracies or technical errors that may appear in the content of this document.

Published in San Mateo, CA, by NextLabs, Inc.

<https://www.nextlabs.com>

info@nextlabs.com

support@nextlabs.com

650.577.9101

Chapter 2

Introduction

2.1 Overview

The CADRMX Client SDK provides object-oriented C++ APIs to develop Rights Management eXtension (RMX) for CAD applications.

The following table describes the terminology and components.

Terminology/Component	Description
Rights Management eXtension (RMX)	A plugin of CAD applications.
Rights Protection Manager (RPM)	A Windows driver running in the kernel.
RPM Folder	A secure folder to let native application handle NXL file transparent.
NXL file	The NextLabs encrypt file format with .nxl extension.
Tag	The meta data persistent in NXL file.

As a developer, you can use CADRMX Client SDK to accomplish the following tasks:

- Protect file with tags.
- Evaluate rights for the rights-projected file.
- Access and edit rights-protected files.
- Manage RPM folders.
- Manage RPM trusted applications.
- Create view overlay with dynamic watermark.
- Evaluate resource rights from central policies.

Next topics:

[Platform and Environment Support](#)

[Getting Started](#)

Chapter 3

Platform and Environment Support

3.1 Operating System

- Windows 7 64-bit
- Windows 10 64-bit

3.2 Runtime Environment

- NextLabs SkyDRM Desktop for Windows
- NextLabs SkyDRM Rights Protection Manager
- NextLabs SkyDRM Rights Management Server
- NextLabs Control Center

3.3 Development Environment

- Microsoft Visual Studio 2015 with Update 3
- CADRMX Client SDK

Next topics:

[Getting Started](#)

Chapter 4

Getting Started

4.1 Setting up Development Environment

- Create a Visual Studio C++ project for RMX module.
- Configure the following Studio project properties:
 - **Include Directories:** {SDK_INSTALLDIR}/include
 - **Additional Library Directories:** {SDK_INSTALLDIR}/bin/x64/release
 - **Additional Dependencies:** CADRMXLib.lib;
- Implement RMX functionalities via CADRMX Client SDK and CAD application SDK.
- Build and deploy RMX plugin on the target application machine.
- Deploy the {SDK_INSTALLDIR}/bin/x64/release/libeay32.dll file in the installation directory of the target application.

4.2 Using APIs to implement Rights Management features

See below examples which illustrate how to use APIs.

4.2.1 Example 1: Initialize RMX

```
#include <RMXLib.h>
IRMXInstance* pInst;
IRMXUser* pUser;
//
// Initialize RMX instance with current logged users via RMD
//
RMXResult result = RMX_GetCurrentLoggedInUser(pInst, pUser);
if (!result) {
    std::wcout << result.GetErrorMessage() << std::endl;
    return;
}
wchar_t szProcessName[MAX_PATH];
GetModuleFileName(NULL, &szProcessName[0], MAX_PATH);
//
// Register caller app as trusted app
//
bool registered;
if (pInst->IsAppRegistered(szProcessName, registered) == 0 && !registered) {
    result = pInst->RegisterApp(szProcessName);
    if (!result) {
        std::wcout << result.GetErrorMessage() << std::endl;
    }
}
```

```

        return;
    }
}
//
// Notify RPM driver the RMX gets started to run
//
result = pInst->NotifyRMXStatus(true);
if (!result) {
    std::wcout << result.GetErrorMessage() << std::endl;
    return;
}
// ....Do other operations
//
// Uninitialize RMX instance when app or plugin terminates
//
RMX_DeleteRMXInstance(pInst);

```

4.2.2 Example 2: Handle RPM folder

```

//
// Check if specified directory is RPM folder
//
bool isRPMDir = false;
const wchar_t* dir = L"c:\\users\\rpmtest";
RMXResult result = pInst->IsRPMFolder(dir, isRPMDir);
if (!result) {
    std::wcout << result.GetErrorMessage() << std::endl;
    return;
}

//
// Add new RPM folder
//
if (!isRPMDir) {
    result = pInst->AddRPMDir(dir, 1);
    if (!result) {
        std::wcout << result.GetErrorMessage() << std::endl;
        return;
    }
}

```

4.2.3 Example 3: Handle file protection

```

//
// Protect file
//
std::wstring newNXFile;
const wchar_t* filePath = L"c:\\users\\rpmtest\\HelloWorld.txt";
const char* tags = "{\\ip_classification\\":[\\\"secret\\\"]}";
RMXResult result = pUser->ProtectFile(filePath, L"c:\\users\\rpmtest", tags, newNXFile);
if (!result) {
    std::wcout << result.GetErrorMessage() << std::endl;
    return;
}
//
// Evaluate file rights
//
std::vector<RMXFileRight> rights;
result = pInst->GetFileRights(newNXFile, rights);
if (!result) {
    std::wcout << result.GetErrorMessage() << std::endl;
    return;
}
//
// Check if edit operation is allowed or not
//
bool allowEdit = false;
for (const auto& r : rights) {
    if (r == RMX_RIGHT_EDIT) {
        allowEdit = true;
        break;
    }
}

```

4.3 General Notes

- APIs return the [RMXResult](#) that contains integer code and error message. The error code is 0 if the operation is successful.

- Must call [RMX_DeleteRMXInstance](#) to terminate the RMX session when the RMX plugin is unloaded.

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

IRMXInstance	15
IRMXUser	24
RMX_CENTRAL_RIGHT	28
RMX_EVAL_ATTRIBUATE	29
RMX_OBLIGATION_INFO	29
RMX_VIEWOVERLAY_PARAMS	30
RMXResult	32

Chapter 6

File Index

6.1 File List

Here is a list of all documented files with brief descriptions:

RMXLGlobal.h	37
RMXLInc.h	39
RMXLInstance.h	39
RMXLResult.h	39
RMXLTypeDef.h	40
RMXLUser.h	42

Chapter 7

Class Documentation

7.1 IRMXInstance Class Reference

```
#include <RMXLInstance.h>
```

Public Member Functions

- virtual [RMXResult IsInitFinished](#) (bool &finished)=0
- virtual [RMXResult GetLoginUser](#) ([IRMXUser](#) *&pUser)=0
- virtual [RMXResult AddRPMDir](#) (const std::wstring &dirPath, uint32_t option=1)=0
- virtual [RMXResult RemoveRPMDir](#) (const std::wstring &dirPath, bool force=true)=0
- virtual [RMXResult IsRPMFolder](#) (const std::wstring &dirPath, bool &isRPMDir, [RMX_RPMFolderRelation](#) *relation=nullptr)=0
- virtual [RMXResult RegisterApp](#) (const std::wstring &appPath)=0
- virtual [RMXResult UnregisterApp](#) (const std::wstring &appPath)=0
- virtual [RMXResult IsAppRegistered](#) (const std::wstring &appPath, bool ®istered)=0
- virtual [RMXResult NotifyRMXStatus](#) (bool running)=0
- virtual [RMXResult EditCopyFile](#) (const std::wstring &nxlFilePath, std::wstring &destPath)=0
- virtual [RMXResult EditSaveFile](#) (const std::wstring &filePath, const std::wstring &originalNXlfilePath=L"", bool deleteSource=false, uint32_t exitEdit=0)=0
- virtual [RMXResult RPMCopyFile](#) (const std::wstring &existingNxIFilePath, const std::wstring &newNxIFile↵Path, bool deleteSource=false)=0
- virtual [RMXResult RPMDeleteFile](#) (const std::wstring &filePath)=0
- virtual [RMXResult GetFileRights](#) (const std::wstring &filePath, std::vector< [RMXFileRight](#) > &rights)=0
- virtual [RMXResult GetFileStatus](#) (const std::wstring &filePath, bool &isProtected, [RMX_RPMFolderRelation](#) *dirRelation=nullptr)=0
- virtual [RMXResult ReadFileTags](#) (const std::wstring &filePath, std::string &tags)=0
- virtual [RMXResult SetViewOveraly](#) (const [RMX_VIEWOVERLAY_PARAMS](#) ¶ms)=0
- virtual [RMXResult ClearViewOverlay](#) (void *hTargetWnd)=0

7.1.1 Detailed Description

This interface defines the RMX instance

Definition at line 20 of file [RMXLInstance.h](#).

7.1.2 Member Function Documentation

7.1.2.1 IsInitFinished()

```
virtual RMXResult IRMXInstance::IsInitFinished (
    bool & finished ) [pure virtual]
```

Checks if the initialization of RMX instance has finished

Parameters

out	<i>finished</i>	true if initialization has finished.
-----	-----------------	--------------------------------------

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

7.1.2.2 GetLoginUser()

```
virtual RMXResult IRMXInstance::GetLoginUser (
    IRMXUser *& pUser ) [pure virtual]
```

Gets current login user in the session

Parameters

out	<i>pUser</i>	Pointer to IRMXUser .
-----	--------------	---------------------------------------

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

7.1.2.3 AddRPMDir()

```
virtual RMXResult IRMXInstance::AddRPMDir (
    const std::wstring & dirPath,
    uint32_t option = 1 ) [pure virtual]
```

Adds RPM secure folder

Parameters

in	<i>dirPath</i>	Directory full path.
in	<i>option</i>	0 When copying a NXL file to this RPM folder, do not allow to overwrite unprotected file if exists 1 Set OVERWRITE_NORMAL flag on the new RPM folder, to allow overwriting unprotected file.

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

7.1.2.4 RemoveRPMDir()

```
virtual RMXResult IRMXInstance::RemoveRPMDir (
    const std::wstring & dirPath,
    bool force = true ) [pure virtual]
```

Removes RPM secure folder

Parameters

in	<i>dirPath</i>	Directory path.
in	<i>force</i>	True to force to remove even though the trusted process is still running. Otherwise, remove will fail.

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

7.1.2.5 IsRPMFolder()

```
virtual RMXResult IRMXInstance::IsRPMFolder (
    const std::wstring & dirPath,
    bool & isRPMDir,
    RMX\_RPMFolderRelation * relation = nullptr ) [pure virtual]
```

Checks if the specified directory is a RPM secure folder

Parameters

in	<i>dirPath</i>	Directory path.
out	<i>isRPMDir</i>	True if it is RPM folder
out	<i>relation</i>	RMX_RPMFolderRelation type

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

See also

[RMX_RPMFolderRelation](#)

7.1.2.6 RegisterApp()

```
virtual RMXResult IRMXInstance::RegisterApp (  
    const std::wstring & appPath ) [pure virtual]
```

Registers an application as rpm trusted app

Parameters

in	<i>appPath</i>	Full path of an executable image.
----	----------------	-----------------------------------

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

7.1.2.7 UnregisterApp()

```
virtual RMXResult IRMXInstance::UnregisterApp (  
    const std::wstring & appPath ) [pure virtual]
```

Unregisters an application as rpm trusted app

Parameters

in	<i>appPath</i>	Full path of an executable image.
----	----------------	-----------------------------------

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

7.1.2.8 IsAppRegistered()

```
virtual RMXResult IRMXInstance::IsAppRegistered (
    const std::wstring & appPath,
    bool & registered ) [pure virtual]
```

Checks if an application is registered as rpm trusted application

Parameters

in	<i>appPath</i>	Full path of an executable image.
out	<i>registered</i>	True if the executable image is registered as rpm trusted application.

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

7.1.2.9 NotifyRMXStatus()

```
virtual RMXResult IRMXInstance::NotifyRMXStatus (
    bool running ) [pure virtual]
```

Notifies RMX status in the caller process to RPM driver

Precondition

[RegisterApp](#) is called to register caller application as RPM trusted application.

Parameters

in	<i>running</i>	True to notify RMX starts to run. False to notify RMX gets stopped.
----	----------------	---

Returns

[RMXResult::GetCode\(\)](#) returns 0 if succeeded. Otherwise, returns error code.

7.1.2.10 EditCopyFile()

```
virtual RMXResult IRMXInstance::EditCopyFile (
    const std::wstring & nxlFilePath,
    std::wstring & destPath ) [pure virtual]
```

Copies NXL file to an RPM folder for editing with native file name.

Parameters

in	<i>nxlFilePath</i>	File path of the NXL file to be edited.
in, out	<i>destPath</i>	As input, specify an RPM folder path. If empty, use RPM secret directory. As output, return the generated NXL file path without .nxl extension.

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

Note

[EditCopyFile](#) is used to copy NXL file into RPM folder, so that native application can edit protected file transparently.

7.1.2.11 EditSaveFile()

```
virtual RMXResult IRMXInstance::EditSaveFile (
    const std::wstring & filePath,
    const std::wstring & originalNXLfilePath = L"",
    bool deleteSource = false,
    uint32_t exitEdit = 0 ) [pure virtual]
```

Re-protects the protected file in RPM folder back to original NXL file

Parameters

in	<i>filePath</i>	Native file path to be updated.
in	<i>originalNXLfilePath</i>	File path of original NXL file. If empty, search local mapping or check current folder.
in	<i>deleteSource</i>	True to delete original NXL file.
in	<i>exitEdit</i>	0: Not exit and save 1: Not exit but save 2: Exit but not save 3: Exit and save

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

Note

[EditSaveFile](#) is used to re-protect the protected file to have recent modification.

7.1.2.12 RPMCopyFile()

```
virtual RMXResult IRMXInstance::RPMCopyFile (
    const std::wstring & existingNxIFilePath,
    const std::wstring & newNxIFilePath,
    bool deleteSource = false ) [pure virtual]
```

Copies NxI file in RPM folder

Parameters

in	<i>existingNxIFilePath</i>	The name of an existing NXI file to be copied.
in	<i>newNxIFilePath</i>	The name of the new NXI file. If the specified new file already exists, the function overwrites the existing file and succeeds.
in	<i>deleteSource</i>	True to delete the source NXI file.

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

Note

[RPMCopyFile](#) is used to copy nxI file by trusted process in RPM folder. By default, using Windows API to copy file by trusted process will copy the corresponding plain file.

7.1.2.13 RPMDeleteFile()

```
virtual RMXResult IRMXInstance::RPMDeleteFile (
    const std::wstring & filePath ) [pure virtual]
```

Deletes NXI file in RPM folder

Parameters

in	<i>filePath</i>	File path to be deleted.
----	-----------------	--------------------------

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

Note [RPMDeleteFile](#) must be called to let RPM driver delete NXI file in RPM folder clearly, including cache.

7.1.2.14 GetFileRights()

```
virtual RMXResult IRMXInstance::GetFileRights (
    const std::wstring & filePath,
    std::vector< RMXFileRight > & rights ) [pure virtual]
```

Gets rights for the specified file

Parameters

in	<i>filePath</i>	File path to be evaluated.
out	<i>rights</i>	Vector of RMXFileRight .

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

See also

[RMXFileRight](#)

7.1.2.15 GetFileStatus()

```
virtual RMXResult IRMXInstance::GetFileStatus (
    const std::wstring & filePath,
    bool & isProtected,
    RMX\_RPMFolderRelation * dirRelation = nullptr ) [pure virtual]
```

Gets status for the specified file, such as if it is in RPM folder, and if it is protected.

Parameters

in	<i>filePath</i>	File path to be evaluated.
out	<i>isProtected</i>	True if file is protected.
out	<i>dirRelation</i>	RMX_RPMFolderRelation type.

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

See also

[RMX_RPMFolderRelation](#)

7.1.2.16 ReadFileTags()

```
virtual RMXResult IRMXInstance::ReadFileTags (
    const std::wstring & filePath,
    std::string & tags ) [pure virtual]
```

Gets tags for the specified file

Parameters

in	<i>filePath</i>	File path to be evaluated.
out	<i>tags</i>	File tags(json-format string), such as {"ip_classification":["secret"],"gov_classification":["test"]}

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

7.1.2.17 SetViewOveraly()

```
virtual RMXResult IRMXInstance::SetViewOveraly (
    const RMX\_VIEWOVERLAY\_PARAMS & params ) [pure virtual]
```

Create a view overlay if dynamic watermark can be found for specified tags parameters.

Parameters

in	<i>params</i>	RMX_VIEWOVERLAY_PARAMS object, containing all below parameters: <ul style="list-style-type: none">• <i>hTargetWnd</i> Top window handler to show view overlay.• <i>vecTags</i> Vector of tags to be evaluated for dynamic watermark.• <i>vecCtxAttrs</i> Vector of context attributes to be evaluated for dynamic watermark, like <i>appPath</i>, <i>fileName</i>, <i>username</i>, etc.• <i>displayOffsets</i> Integer array to specify the offset effect in overlay display area, with {left,upper,right,bottom} sequence.
----	---------------	---

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

Note

Call [ClearViewOverlay](#) to remove view overlay

See also

[RMX_VIEWOVERLAY_PARAMS](#)
[ClearViewOverlay](#)

7.1.2.18 ClearViewOverlay()

```
virtual RMXResult IRMXInstance::ClearViewOverlay (
    void * hTargetWnd ) [pure virtual]
```

Removes view overlay

Parameters

in	<i>hTargetWnd</i>	Top window handler to show view overlay
----	-------------------	---

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

The documentation for this class was generated from the following file:

- [RMXLInstance.h](#)

7.2 IRMXUser Class Reference

```
#include <RMXLUser.h>
```

Public Member Functions

- virtual const std::wstring [GetName](#) ()=0
- virtual const std::wstring [GetEmail](#) ()=0
- virtual uint32_t [GetUserID](#) ()=0
- virtual [RMXResult](#) [ProtectFile](#) (const std::wstring &filePath, const std::wstring &targetDir, const std::string &tags, std::wstring &newNxIFile)=0
- virtual [RMXResult](#) [GetResourceRightsFromCentralPolicies](#) (const std::wstring &resourceName, const std::wstring &resourceType, const std::vector< [RMX_EVAL_ATTRIBUATE](#) > &attrs, std::vector< [RMX_CENTRAL_RIGHT](#) > ¢ralRights)=0
- virtual bool [GetDefaultPolicyBundle](#) (std::string &policyBundle)=0
- virtual const std::string [GetSystemProjectTenantId](#) ()=0
- virtual const std::string [GetDefaultTokenGroupName](#) ()=0
- virtual [RMXResult](#) [MergeTags](#) (const std::string &unionTags, std::string &mergedTags)=0

7.2.1 Detailed Description

This interface provides information about a RMX user.

Definition at line 17 of file [RMXLUser.h](#).

7.2.2 Member Function Documentation

7.2.2.1 GetName()

```
virtual const std::wstring IRMXUser::GetName ( ) [pure virtual]
```

Gets current login user name

Returns

User name string.

7.2.2.2 GetEmail()

```
virtual const std::wstring IRMXUser::GetEmail ( ) [pure virtual]
```

Gets current login user email address

Returns

User email string.

7.2.2.3 GetUserID()

```
virtual uint32_t IRMXUser::GetUserID ( ) [pure virtual]
```

Get current login user id.

Returns

User id.

7.2.2.4 ProtectFile()

```
virtual RMXResult IRMXUser::ProtectFile (
    const std::wstring & filePath,
    const std::wstring & targetDir,
    const std::string & tags,
    std::wstring & newNxIFile ) [pure virtual]
```

Protects a local file

Parameters

in	<i>filePath</i>	File path to be protected.
in	<i>targetDir</i>	Target directory path to place new protected file.
in	<i>tags</i>	Document classification metadata.
out	<i>newNxIFile</i>	File path of NXI file to be created.

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

7.2.2.5 GetResourceRightsFromCentralPolicies()

```
virtual RMXResult IRMXUser::GetResourceRightsFromCentralPolicies (
    const std::wstring & resourceName,
    const std::wstring & resourceType,
    const std::vector< RMX\_EVAL\_ATTRIBUTE > & attrs,
    std::vector< RMX\_CENTRAL\_RIGHT > & centralRights ) [pure virtual]
```

Gets the resource rights with their associated obligations from central policies

Parameters

in	<i>resourceName</i>	Name of resource
in	<i>resourceType</i>	Type of resource (e.g. "fso", "portal")
in	<i>attrs</i>	Resource attributes to be evaluated
out	<i>centralRights</i>	Rights assigned to the resource and their associated obligations to be returned.

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

7.2.2.6 GetDefaultPolicyBundle()

```
virtual bool IRMXUser::GetDefaultPolicyBundle (
    std::string & policyBundle ) [pure virtual]
```

Gets policy boudle for default tenant

Parameters

out	<i>policyBundle</i>	Policy bundle to be returned.
-----	---------------------	-------------------------------

Returns

True if success, else return false.

7.2.2.7 GetSystemProjectTenantId()

```
virtual const std::string IRMXUser::GetSystemProjectTenantId ( ) [pure virtual]
```

Gets system project tenantId

Returns

System project tenantId

7.2.2.8 GetDefaultTokenGroupName()

```
virtual const std::string IRMXUser::GetDefaultTokenGroupName ( ) [pure virtual]
```

Gets token group name of default tenant

Returns

Token group name of default tenant

7.2.2.9 MergeTags()

```
virtual RMXResult IRMXUser::MergeTags (
    const std::string & unionTags,
    std::string & mergedTags ) [pure virtual]
```

Merges tags in case tags come from multiple source NXL files, according to the priority obligation defined in central policy.

Parameters

in	<i>unionTags</i>	Multiple tags strings grouped by "{}", such as {"ip_classification":["secret"]}{"ip_classification":["supper-secure"]}.
out	<i>mergedTags</i>	Single group of tags to be returned, such as {"ip_classification":["supper-secure"]}.

Note

The algorithm to merge tags groups are:

- Duplicated value should be removed for same key.
- Different values are combined as multiples values for same key.
- Lower priorit values will be removed, and leave higher one as single value for same key, if priority obligation is found in central policy.

Returns

[RMXResult](#) object. Either call `RMXResult::GetCode()` to check error code, or call `bool()` to check if error code is 0.

The documentation for this class was generated from the following file:

- [RMXUser.h](#)

7.3 RMX_CENTRAL_RIGHT Struct Reference

```
#include <RMXLTypeDef.h>
```

Public Attributes

- [RMXFileRight](#) `right`
- `std::vector< RMX_OBLIGATION_INFO >` `obligations`

7.3.1 Detailed Description

Structure to store a central right and obligations

Definition at line 85 of file `RMXLTypeDef.h`.

7.3.2 Member Data Documentation

7.3.2.1 `right`

```
RMXFileRight RMX_CENTRAL_RIGHT::right
```

File right

Definition at line 87 of file `RMXLTypeDef.h`.

7.3.2.2 `obligations`

```
std::vector<RMX\_OBLIGATION\_INFO> RMX_CENTRAL_RIGHT::obligations
```

List of obligations

Definition at line 88 of file `RMXLTypeDef.h`.

The documentation for this struct was generated from the following file:

- [RMXLTypeDef.h](#)

7.4 RMX_EVAL_ATTRIBUATE Struct Reference

```
#include <RMXLTypeDef.h>
```

Public Attributes

- std::wstring [key](#)
- std::wstring [value](#)

7.4.1 Detailed Description

Structure to store an attribute to be evaluated

Definition at line 56 of file RMXLTypeDef.h.

7.4.2 Member Data Documentation

7.4.2.1 key

```
std::wstring RMX_EVAL_ATTRIBUATE::key
```

Attribute name

Definition at line 58 of file RMXLTypeDef.h.

7.4.2.2 value

```
std::wstring RMX_EVAL_ATTRIBUATE::value
```

Attribute value

Definition at line 59 of file RMXLTypeDef.h.

The documentation for this struct was generated from the following file:

- [RMXLTypeDef.h](#)

7.5 RMX_OBLIGATION_INFO Struct Reference

```
#include <RMXLTypeDef.h>
```

Public Attributes

- std::wstring [name](#)
- std::vector< [RMX_EVAL_ATTRIBUTE](#) > [options](#)

7.5.1 Detailed Description

Structure to store obligation information defined in central policy

Definition at line 77 of file RMXLTypeDef.h.

7.5.2 Member Data Documentation

7.5.2.1 name

```
std::wstring RMX_OBLIGATION_INFO::name
```

Name of obligation name

Definition at line 78 of file RMXLTypeDef.h.

7.5.2.2 options

```
std::vector<RMX\_EVAL\_ATTRIBUTE> RMX_OBLIGATION_INFO::options
```

Attributes of obligation

Definition at line 79 of file RMXLTypeDef.h.

The documentation for this struct was generated from the following file:

- [RMXLTypeDef.h](#)

7.6 RMX_VIEWOVERLAY_PARAMS Struct Reference

```
#include <RMXLTypeDef.h>
```

Public Attributes

- void * [hTargetWnd](#)
- std::vector< std::string > [vecTags](#)
- std::vector< [RMX_EVAL_ATTRIBUTE](#) > [vecCtxAttrs](#)
- int [displayOffsets](#) [4]

7.6.1 Detailed Description

Structure to store parameters required to set view overlay

Definition at line 66 of file RMXLTypeDef.h.

7.6.2 Member Data Documentation

7.6.2.1 hTargetWnd

```
void* RMX_VIEWOVERLAY_PARAMS::hTargetWnd
```

Top window handler to show view overlay

Definition at line 68 of file RMXLTypeDef.h.

7.6.2.2 vecTags

```
std::vector<std::string> RMX_VIEWOVERLAY_PARAMS::vecTags
```

Vector of tags to be evaluated for dynamic watermark

Definition at line 69 of file RMXLTypeDef.h.

7.6.2.3 vecCtxAttrs

```
std::vector<RMX_EVAL_ATTRIBUATE> RMX_VIEWOVERLAY_PARAMS::vecCtxAttrs
```

Vector of context attributes to be evaluated for dynamic watermark, like appPath, fileName, username, etc

Definition at line 70 of file RMXLTypeDef.h.

7.6.2.4 displayOffsets

```
int RMX_VIEWOVERLAY_PARAMS::displayOffsets[4]
```

Integer array to specify the offset effect in overlay display area, with {left,upper,right,bottom} sequence

Definition at line 71 of file RMXLTypeDef.h.

The documentation for this struct was generated from the following file:

- [RMXLTypeDef.h](#)

7.7 RMXResult Class Reference

```
#include <RMXLResult.h>
```

Public Member Functions

- [RMXResult](#) ()
- [RMXResult](#) ([RMXErrCode_t](#) code, const std::wstring &errorMessage)
- virtual [~RMXResult](#) ()
- [RMXResult](#) (const [RMXResult](#) &other)
- [RMXResult](#) & [operator=](#) (const [RMXResult](#) &other)
- [operator bool](#) () const
- bool [operator==](#) (int code) const
- bool [operator!=](#) (int code) const
- [RMXErrCode_t](#) [GetErrCode](#) () const
- std::wstring [GetErrorMessage](#) () const

Protected Attributes

- [RMXErrCode_t](#) [m_code](#)
- std::wstring [m_message](#)

7.7.1 Detailed Description

This class provides information about the result of API call.

Definition at line 20 of file [RMXLResult.h](#).

7.7.2 Constructor & Destructor Documentation

7.7.2.1 RMXResult() [1/3]

```
RMXResult::RMXResult ( ) [inline]
```

Default constructor

Definition at line 24 of file [RMXLResult.h](#).

```
24 : m\_code(0) {};
```

7.7.2.2 RMXResult() [2/3]

```
RMXResult::RMXResult (
    RMXErrCode\_t code,
    const std::wstring & errorMessage ) [inline], [explicit]
```

Constructor

Parameters

in	<i>code</i>	0 if succeeded. Otherwise, error code.
in	<i>errMessage</i>	Error message if failed.

Definition at line 33 of file RMXLResult.h.

```

34         : m_code(code), m_message(errMessage)
35     {
36     }
```

7.7.2.3 ~RMXResult()

```
virtual RMXResult::~~RMXResult ( ) [inline], [virtual]
```

Destructor

Definition at line 38 of file RMXLResult.h.

```
38 {}
```

7.7.2.4 RMXResult() [3/3]

```
RMXResult::RMXResult (
    const RMXResult & other ) [inline]
```

Copy constructor

Definition at line 45 of file RMXLResult.h.

```

46         : m_code(other.m_code), m_message(other.m_message)
47     {
48     }
```

7.7.3 Member Function Documentation

7.7.3.1 operator=()

```
RMXResult& RMXResult::operator= (
    const RMXResult & other ) [inline]
```

Assignment operator

Definition at line 55 of file RMXLResult.h.

```

56     {
57         if (this != &other)
58         {
59             m_code = other.m_code;
60             m_message = other.m_message;
61         }
62         return *this;
63     }
```

7.7.3.2 operator bool()

```
RMXResult::operator bool ( ) const [inline]
```

Operator to check if the result indicates failure with error

Returns

True if no error occurs.

Definition at line 72 of file RMXLResult.h.

```
72 { return (0 == m_code); }
```

7.7.3.3 operator==()

```
bool RMXResult::operator== (
    int code ) const [inline]
```

Operator to check if they are same error

Parameters

<i>code</i>	The error code.
-------------	-----------------

Returns

True if error code is same.

Definition at line 82 of file RMXLResult.h.

```
82 { return (code == m_code); }
```

7.7.3.4 operator!=()

```
bool RMXResult::operator!= (
    int code ) const [inline]
```

Operator to check if they are same error

Parameters

<i>code</i>	The code.
-------------	-----------

Returns

False if error code is same.

Definition at line 91 of file RMXLResult.h.

```
91 { return (code != m_code); }
```

7.7.3.5 GetErrCode()

```
RMXErrCode_t RMXResult::GetErrCode ( ) const [inline]
```

Gets error code

Returns

The error code.

Definition at line 99 of file RMXLResult.h.

```
99 { return m_code; }
```

7.7.3.6 GetErrMessage()

```
std::wstring RMXResult::GetErrMessage ( ) const [inline]
```

Gets error message

Returns

The error message.

Definition at line 107 of file RMXLResult.h.

```
107 { return m_message; }
```

7.7.4 Member Data Documentation

7.7.4.1 m_code

```
RMXErrCode_t RMXResult::m_code [protected]
```

The code

Definition at line 111 of file RMXLResult.h.

7.7.4.2 m_message

```
std::wstring RMXResult::m_message [protected]
```

The message

Definition at line 113 of file RMXLResult.h.

The documentation for this class was generated from the following file:

- [RMXLResult.h](#)

Chapter 8

File Documentation

8.1 RMXLGlobal.h File Reference

```
#include <wtypes.h>
#include "RMXLResult.h"
#include "RMXLTypeDef.h"
```

Functions

- CADDRMX_API DWORD [RMX_GetLibVersion](#) (void)
- CADDRMX_API DWORD [RMX_GetSDWLibVersion](#) (void)
- CADDRMX_API [RMXResult](#) [RMX_GetCurrentLoggedInUser](#) ([IRMXInstance](#) *pInstance, [IRMXUser](#) *pUser)
- CADDRMX_API void [RMX_DeleteRMXInstance](#) ([IRMXInstance](#) *pInstance)

8.1.1 Detailed Description

This head file contains all gloal functions

8.1.2 Function Documentation

8.1.2.1 [RMX_GetLibVersion\(\)](#)

```
CADDRMX_API DWORD RMX_GetLibVersion (
    void )
```

Returns the version number of the CADDRMX Client SDK

Returns

The format of version is AABBCCCC major.minor.build.

8.1.2.2 RMX_GetSDWLibVersion()

```
CADRMX_API DWORD RMX_GetSDWLibVersion (
    void )
```

Returns the version number of the SkyDRM Client SDK

Returns

The format of version is AABBCCCC major.minor.build.

8.1.2.3 RMX_GetCurrentLoggedInUser()

```
CADRMX_API RMXResult RMX_GetCurrentLoggedInUser (
    IRMXInstance *& pInstance,
    IRMXUser *& pUser )
```

Gets current login user

Parameters

out	<i>pInstance</i>	IRMXInstance pointer.
out	<i>pUser</i>	IRMXUser pointer

Returns

[RMXResult](#) object. Either call [RMXResult::GetCode\(\)](#) to check error code, or call [bool\(\)](#) to check if error code is 0.

Note

Call [RMX_DeleteRMXInstance](#) to delete instance.

8.1.2.4 RMX_DeleteRMXInstance()

```
CADRMX_API void RMX_DeleteRMXInstance (
    IRMXInstance * pInstance )
```

Deletes an [IRMXInstance](#) instance

Precondition

[RMX_GetCurrentLoggedInUser](#) is called to get the [IRMXInstance](#) instance.

Parameters

in	<i>pInstance</i>	IRMXInstance pointer.
----	------------------	---------------------------------------

Returns

void

8.2 RMXLInc.h File Reference

```
#include "RMXLInstance.h"
#include "RMXLGlobal.h"
#include "RMXLUser.h"
```

8.2.1 Detailed Description

This header file includes all headers of SDK.

8.3 RMXLInstance.h File Reference

```
#include "RMXLResult.h"
#include "RMXLTypeDef.h"
```

Classes

- class [IRMXInstance](#)

8.4 RMXLResult.h File Reference

```
#include <string>
#include "RMXLTypeDef.h"
```

Classes

- class [RMXResult](#)

Typedefs

- typedef unsigned long [RMXErrCode_t](#)

8.4.1 Typedef Documentation

8.4.1.1 RMXErrCode_t

```
typedef unsigned long RMXErrCode_t
```

Error code

Definition at line 12 of file RMXLResult.h.

8.5 RMXLTypeDef.h File Reference

```
#include <stdint.h>
#include <string>
#include <vector>
```

Classes

- struct [RMX_EVAL_ATTRIBUATE](#)
- struct [RMX_VIEWOVERLAY_PARAMS](#)
- struct [RMX_OBLIGATION_INFO](#)
- struct [RMX_CENTRAL_RIGHT](#)

Macros

- #define [CADRMX_API](#) __declspec(dllimport)

Enumerations

- enum [RMX_RPMFolderRelation](#) { [RMX_NonRPMFolder](#), [RMX_RPMFolder](#), [RMX_RPMAncestralFolder](#), [RMX_RPMInheritedFolder](#) }
- enum [RMXFileRight](#) {
[RMX_RIGHT_VIEW](#) = 1, [RMX_RIGHT_EDIT](#) = 2, [RMX_RIGHT_PRINT](#) = 4, [RMX_RIGHT_CLIPBOARD](#) = 8,
[RMX_RIGHT_SAVEAS](#) = 0x10, [RMX_RIGHT_DECRYPT](#) = 0x20, [RMX_RIGHT_SCREENCAPTURE](#) =
0x40, [RMX_RIGHT_SEND](#) = 0x80,
[RMX_RIGHT_CLASSIFY](#) = 0x100, [RMX_RIGHT_SHARE](#) = 0x200, [RMX_RIGHT_DOWNLOAD](#) = 0x400,
[RMX_RIGHT_WATERMARK](#) = 0x40000000 }

8.5.1 Detailed Description

Declarations used throughout the CADRMX Client SDK.

8.5.2 Enumeration Type Documentation

8.5.2.1 RMX_RPMFolderRelation

```
enum RMX_RPMFolderRelation
```

RPM folder relation type

Enumerator

RMX_NonRPMFolder	Not RPM folder
RMX_RPMFolder	RPM folder
RMX_RPMAncestralFolder	Ancestor of RPM folder
RMX_RPMInheritedFolder	Descendant of PRM folder

Definition at line 26 of file RMXLTypeDef.h.

```

27 {
28     RMX_NonRPMFolder,
29     RMX_RPMFolder,
30     RMX_RPMAncestralFolder,
31     RMX_RPMInheritedFolder,
32 } RMX_RPMFolderRelation;
```

8.5.2.2 RMXFileRight

```
enum RMXFileRight
```

RMX file right type

Enumerator

RMX_RIGHT_VIEW	Right to view a protected file
RMX_RIGHT_EDIT	Right to edit a protected file
RMX_RIGHT_PRINT	Right to print a protected file
RMX_RIGHT_CLIPBOARD	Right to copy content of a protected file to clipboard
RMX_RIGHT_SAVEAS	Right to save copy of a protected file
RMX_RIGHT_DECRYPT	Right to remove protection
RMX_RIGHT_SCREENCAPTURE	Right to capture screen
RMX_RIGHT_SEND	Right to send a protected file
RMX_RIGHT_CLASSIFY	Right to reclassify a protected file
RMX_RIGHT_SHARE	Right to share a protected file
RMX_RIGHT_DOWNLOAD	Right to download a protected file
RMX_RIGHT_WATERMARK	Right to display a watermark on a protected file

Definition at line 37 of file RMXLTypeDef.h.

```

38 {
39     RMX_RIGHT_VIEW = 1,
40     RMX_RIGHT_EDIT = 2,
41     RMX_RIGHT_PRINT = 4,
42     RMX_RIGHT_CLIPBOARD = 8,
43     RMX_RIGHT_SAVEAS = 0x10,
44     RMX_RIGHT_DECRYPT = 0x20,
45     RMX_RIGHT_SCREENCAPTURE = 0x40,
46     RMX_RIGHT_SEND = 0x80,
47     RMX_RIGHT_CLASSIFY = 0x100,
48     RMX_RIGHT_SHARE = 0x200,
49     RMX_RIGHT_DOWNLOAD = 0x400,
50     RMX_RIGHT_WATERMARK = 0x40000000
51 } RMXFileRight;
```

8.6 RMXLUser.h File Reference

```
#include "RMXLResult.h"  
#include "RMXLTypeDef.h"
```

Classes

- class [IRMXUser](#)

Index

~RMXResult
 RMXResult, [33](#)

AddRPMDir
 IRMXInstance, [16](#)

ClearViewOverlay
 IRMXInstance, [23](#)

displayOffsets
 RMX_VIEWOVERLAY_PARAMS, [31](#)

EditCopyFile
 IRMXInstance, [19](#)

EditSaveFile
 IRMXInstance, [20](#)

GetDefaultPolicyBundle
 IRMXUser, [26](#)

GetDefaultTokenGroupName
 IRMXUser, [27](#)

GetEmail
 IRMXUser, [25](#)

GetErrCode
 RMXResult, [35](#)

GetErrMsg
 RMXResult, [35](#)

GetFileRights
 IRMXInstance, [21](#)

GetFileStatus
 IRMXInstance, [22](#)

GetLoginUser
 IRMXInstance, [16](#)

GetName
 IRMXUser, [24](#)

GetResourceRightsFromCentralPolicies
 IRMXUser, [26](#)

GetSystemProjectTenantId
 IRMXUser, [26](#)

GetUserID
 IRMXUser, [25](#)

hTargetWnd
 RMX_VIEWOVERLAY_PARAMS, [31](#)

IRMXInstance, [15](#)
 AddRPMDir, [16](#)
 ClearViewOverlay, [23](#)
 EditCopyFile, [19](#)
 EditSaveFile, [20](#)
 GetFileRights, [21](#)
 GetFileStatus, [22](#)
 GetLoginUser, [16](#)
 IsAppRegistered, [18](#)
 IsInitFinished, [16](#)
 IsRPMFolder, [17](#)
 NotifyRMXStatus, [19](#)
 ReadFileTags, [22](#)
 RegisterApp, [18](#)
 RemoveRPMDir, [17](#)
 RPMCopyFile, [20](#)
 RPMDelFile, [21](#)
 SetViewOveraly, [23](#)
 UnregisterApp, [18](#)

IRMXUser, [24](#)
 GetDefaultPolicyBundle, [26](#)
 GetDefaultTokenGroupName, [27](#)
 GetEmail, [25](#)
 GetName, [24](#)
 GetResourceRightsFromCentralPolicies, [26](#)
 GetSystemProjectTenantId, [26](#)
 GetUserID, [25](#)
 MergeTags, [27](#)
 ProtectFile, [25](#)

IsAppRegistered
 IRMXInstance, [18](#)

IsInitFinished
 IRMXInstance, [16](#)

IsRPMFolder
 IRMXInstance, [17](#)

key
 RMX_EVAL_ATTRIBUATE, [29](#)

m_code
 RMXResult, [35](#)

m_message
 RMXResult, [35](#)

MergeTags
 IRMXUser, [27](#)

name
 RMX_OBLIGATION_INFO, [30](#)

NotifyRMXStatus
 IRMXInstance, [19](#)

obligations
 RMX_CENTRAL_RIGHT, [28](#)

operator bool
 RMXResult, [33](#)

operator!=

- RMXResult, 34
- operator=
 - RMXResult, 33
- operator==
 - RMXResult, 34
- options
 - RMX_OBLIGATION_INFO, 30
- ProtectFile
 - IRMXUser, 25
- ReadFileTags
 - IRMXInstance, 22
- RegisterApp
 - IRMXInstance, 18
- RemoveRPMDir
 - IRMXInstance, 17
- right
 - RMX_CENTRAL_RIGHT, 28
- RMX_CENTRAL_RIGHT, 28
 - obligations, 28
 - right, 28
- RMX_DeleteRMXInstance
 - RMXLGlobal.h, 38
- RMX_EVAL_ATTRIBUATE, 29
 - key, 29
 - value, 29
- RMX_GetCurrentLoggedInUser
 - RMXLGlobal.h, 38
- RMX_GetLibVersion
 - RMXLGlobal.h, 37
- RMX_GetSDWLibVersion
 - RMXLGlobal.h, 37
- RMX_NonRPMFolder
 - RMXLTypeDef.h, 41
- RMX_OBLIGATION_INFO, 29
 - name, 30
 - options, 30
- RMX_RIGHT_CLASSIFY
 - RMXLTypeDef.h, 41
- RMX_RIGHT_CLIPBOARD
 - RMXLTypeDef.h, 41
- RMX_RIGHT_DECRYPT
 - RMXLTypeDef.h, 41
- RMX_RIGHT_DOWNLOAD
 - RMXLTypeDef.h, 41
- RMX_RIGHT_EDIT
 - RMXLTypeDef.h, 41
- RMX_RIGHT_PRINT
 - RMXLTypeDef.h, 41
- RMX_RIGHT_SAVEAS
 - RMXLTypeDef.h, 41
- RMX_RIGHT_SCREENCAPTURE
 - RMXLTypeDef.h, 41
- RMX_RIGHT_SEND
 - RMXLTypeDef.h, 41
- RMX_RIGHT_SHARE
 - RMXLTypeDef.h, 41
- RMX_RIGHT_VIEW

- RMXLTypeDef.h, 41
- RMX_RIGHT_WATERMARK
 - RMXLTypeDef.h, 41
- RMX_RPMAncestralFolder
 - RMXLTypeDef.h, 41
- RMX_RPMFolder
 - RMXLTypeDef.h, 41
- RMX_RPMFolderRelation
 - RMXLTypeDef.h, 40
- RMX_RPMInheritedFolder
 - RMXLTypeDef.h, 41
- RMX_VIEWOVERLAY_PARAMS, 30
 - displayOffsets, 31
 - hTargetWnd, 31
 - vecCtxAttrs, 31
 - vecTags, 31
- RMXErrCode_t
 - RMXLResult.h, 40
- RMXFileRight
 - RMXLTypeDef.h, 41
- RMXLGlobal.h, 37
 - RMX_DeleteRMXInstance, 38
 - RMX_GetCurrentLoggedInUser, 38
 - RMX_GetLibVersion, 37
 - RMX_GetSDWLibVersion, 37
- RMXLInc.h, 39
- RMXLInstance.h, 39
- RMXLResult.h, 39
 - RMXErrCode_t, 40
- RMXLTypeDef.h, 40
 - RMX_NonRPMFolder, 41
 - RMX_RIGHT_CLASSIFY, 41
 - RMX_RIGHT_CLIPBOARD, 41
 - RMX_RIGHT_DECRYPT, 41
 - RMX_RIGHT_DOWNLOAD, 41
 - RMX_RIGHT_EDIT, 41
 - RMX_RIGHT_PRINT, 41
 - RMX_RIGHT_SAVEAS, 41
 - RMX_RIGHT_SCREENCAPTURE, 41
 - RMX_RIGHT_SEND, 41
 - RMX_RIGHT_SHARE, 41
 - RMX_RIGHT_VIEW, 41
 - RMX_RIGHT_WATERMARK, 41
 - RMX_RPMAncestralFolder, 41
 - RMX_RPMFolder, 41
 - RMX_RPMFolderRelation, 40
 - RMX_RPMInheritedFolder, 41
 - RMXFileRight, 41
- RMXLUser.h, 42
- RMXResult, 32
 - ~RMXResult, 33
 - GetErrCode, 35
 - GetErrMsg, 35
 - m_code, 35
 - m_message, 35
 - operator bool, 33
 - operator!=, 34
 - operator=, 33

- operator==, [34](#)
 - RMXResult, [32](#), [33](#)
- RPMCopyFile
 - IRMXInstance, [20](#)
- RPMDeleteFile
 - IRMXInstance, [21](#)
- SetViewOveraly
 - IRMXInstance, [23](#)
- UnregisterApp
 - IRMXInstance, [18](#)
- value
 - RMX_EVAL_ATTRIBUATE, [29](#)
- vecCtxAttrs
 - RMX_VIEWOVERLAY_PARAMS, [31](#)
- vecTags
 - RMX_VIEWOVERLAY_PARAMS, [31](#)