SAS® Drug Development
SAS API Macros 1.2
User's Guide

SAS® Documentation

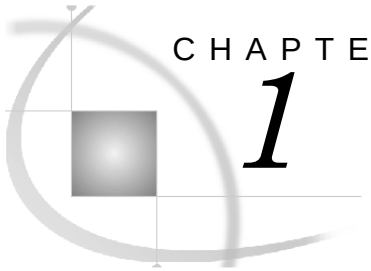
SAS® Drug Development 4.3: Macros User's Guide

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

2nd electronic book, July 2013

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at **support.sas.com/publishing** or call 1-800-727-3228.

Contents



CHAPTER 1

Introduction

Audience

This guide is intended for users who want to develop applications with the SAS Drug Development macros.

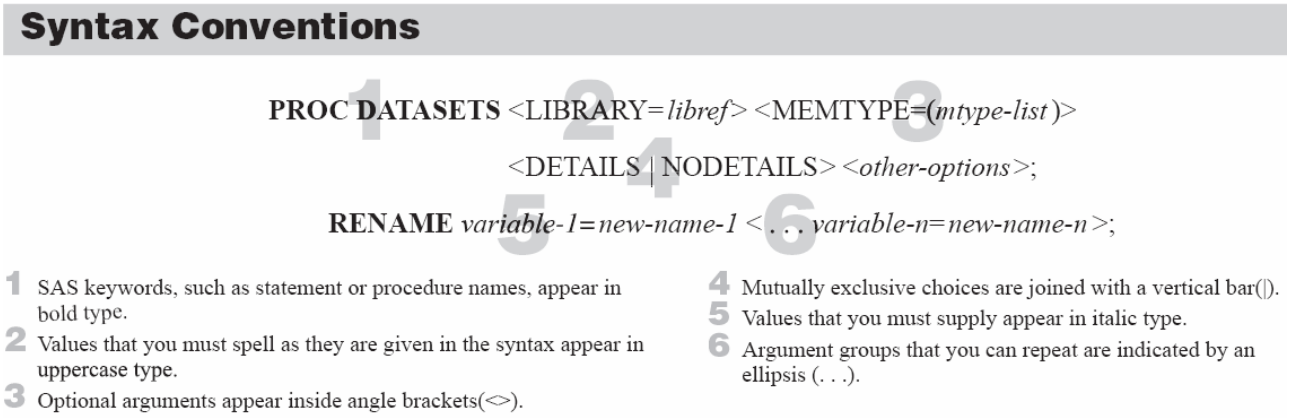
You must be familiar with SAS Drug Development functionality, such as type definitions, containers, files, and access permissions. For reference information on SAS Drug Development functionality, see the SAS Drug Development online Help and user's guide.

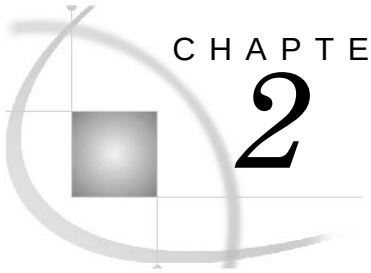
Typographic Conventions Used in This Guide

Throughout this document you'll see the following typographic conventions:

Convention	Description
`monospace` font	denotes code, such as a code example
`monospace bold` font	denotes text that you type, such as an object name
`monospace bold italics` font	denotes a value that you specify, such as your name

The following graphic explains the syntax for SAS code used in this document:





CHAPTER 2

Installing the Macros

Overview

This document describes how to install the SAS Drug Development API Macros, which are distributed in the sdd-sas-macro-1.2.zip.

Requirements

The following software is required by the SAS Drug Development Macros:

- SAS 9.3+
- Java Runtime Environment Version 1.6.0+
- The corresponding version of the SAS Drug Development Remote API client

Installation Instructions for Microsoft Windows

1. Follow the instructions in the *Getting Started with the SAS Drug Development Java API* doc to install the API client. Be sure to note the location of the lib directory. Typically this would be:

   ```
   C:\sdd-java-api-client-1.6\lib
   ```

2. Unzip the contents of sdd-sas-macro-1.2.zip to C:\.

   This creates the following files and folders in C:\:

   - sdd-sas-macro-1.2\lib

     This folder contains the sas.hls.drug.api.macro.jar
   - sdd-sas-macro-1.2\sasmacros

     This is the folder that contains the SAS Drug Development macros as .sas files.

3. Determine where you have SAS installed and where the config file is. In a typical Windows English Language installation, **!sasroot** would point to:

   ```
   C:\Program Files\SAS\SASFoundation\9.3\nls\en
   ```

You need to edit the file **!sasroot\sasv9.cfg** and add the following lines near the top, right before the comment box with the "WARNING:" label in it:

```
/* define the location of the SAS Drug Development API Macros   */
-insert sasautos "C:\sdd-sas-macro-1.2\sasmacros"

/* put both the macro and api client jars on the classpath      */
-JREOPTIONS (-Dsas.app.class.dirs=C:\sdd-sas-macro-1.2\lib;C:\sdd-java-api-
client-1.5\lib)
```

Caution: Back up this file before you edit it. Use extreme care when editing this file and modify only these options. Ensure that you do not insert any carriage returns in the **sas.app.class.dirs** option. If you have any questions, concerns, or problems, contact your on-site SAS support personnel.

4 If you are connecting to an instance of SAS Drug Development that is hosted by SAS, and you are behind a firewall and proxy servers, you might need to address proxy requirements or restrictions.

If HTTPS traffic is proxied, specify the following Java system properties in the -JREOPTIONS statement to configure the Java runtime environment:

```
-Dhttps.proxyHost=<proxy-host-name> -Dhttps.proxyPort=<port-number>
```

The -JREOPTIONS statement (when using a proxy server) will look similar to the following when you are finished:

```
-JREOPTIONS=(
-Dsas.app.class.dirs=C:\sdd-sas-macro-1.2\lib;C:\sdd-java-api-client-1.5\lib
-Dhttps.proxyHost=yourProxyServer.sas.com
-Dhttps.proxyPort=8080)
```

5 Once you have made these changes to the sasv9.cfg file, save the file and start a new SAS session.

Verifying the Installation

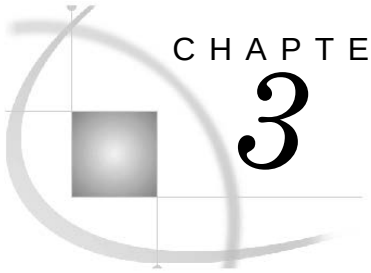
The following SAS code displays the settings for the JREOPTIONS and verifies that the JRE is configured properly. It also verifies that the SAS Drug Development macros are installed and functioning as expected. Replace ***sdd-instance***, ***sdd-user-ID***, and ***sdd-password*** with values that reflect your instance of SAS Drug Development.

```
/* verify the JRE settings                          */

options mprint;

proc javainfo;

run;


/* initiate a connection to SAS Drug Development     */

%sasdrugdev_login(sdd_url=%str(https://sdd-instance), sdd_userid=%str(sdd-user-ID),
sdd_password=%str(sdd-password)) ;


/* print information about the version of the API    */

%sasdrugdev_getsddapiversion() ;


/* List the contents of the root folder in SDD       */

%sasdrugdev_getchildren(sdd_path=%str(/SAS/Files)) ;

proc print;
  title "List of Objects in /SAS/Files";

run;


/* terminate the connection to SAS Drug Development  */

%sasdrugdev_logout() ;
```

The code should generate a list of the contents of the root folder in the SAS Drug Development repository.

Note: By default, the root folder in the SAS Drug Development repository is **/SAS**. If the root of your SAS Drug Development repository is different, change **/SAS** in the code above.

The SAS log will contain information that might be useful for debugging the installation of the SAS Drug Development macros.



CHAPTER 3

SAS® Drug Development Macros

Introduction

The SAS Drug Development macros enable you to use familiar SAS macro syntax to perform operations against SAS Drug Development.

Macro Return Codes

After you execute a macro, the global macro variable _SDDRC_ will contain a return code reflecting the success or failure of the operation; the global macro variable _SDDMSG_ will contain text information regarding the success or the cause of the failure

Here are the specific values that can be returned by a macro:

SDDRC Value	Explanation
-99	There is no return code.
-70	HTTP session has expired.
-69	No explicit permissions were set for a new acl entry.
-68	The ACL principal type is invalid.
-67	The ACL owner was not updated.
-66	The ACL type is invalid.
-65	The privilege is already assigned.
-64	An unknown error occurred while updating permissions.
-60	The privilege was not found.
-59	Invalid role assignment.
-58	Login and logout macros are not allowed within an active SAS DRUG DEVELOPMENT session.
-57	The specified path is not a valid repository path.
-56	The specified path does not exist within the organization.
-55	Cannot unassign a role that was defined at the same context level.
-54	An invalid value was specified for the principal type.
-53	An invalid value was specified for including inheritance.
-52	The role or group was not found in the context.
-51	Could not establish a connection to the ACL service.
-50	The role context was not valid.
-49	An invalid member was specified.
-48	The specified path is not a container.
-47	The member is not available for the action.
-46	The specified group context was not valid.
-45	The specified member was not found.
-40	The specified path is not a valid context value.
-39	The target folder is invalid.
-38	The path is missing.
-37	The source equals the target.
-36	The version setting is invalid.
-35	The local path does not exist.
-34	An invalid parameter was passed.
-33	A new entry was not created.
-32	The object was not versioned.
-30	There is an invalid recursion.

SDDRC Value	Explanation
-29	There are multiple sessions.
-28	The content source is invalid.
-25	An entry exists.
-14	The user account could not be authenticated.
-10	The user account is invalid.
-9	The group is invalid.
-8	An unexpected error was encountered.
-4	The node is invalid.
-3	The object type is invalid for this macro.
-2	The session does not exist.
-1	The session is no longer valid.
0	The macro executed without error.
1	The entity code is valid.
3	The member was found.
4	The entity was found within the given context.
5	Path is a container.
6	The privilege was found.

Note: A macro always returns a code, but any single macro will not return all of these codes.

The Macros

Using the Ampersand Character (&) in URLs

For a macro with a parameter that specifies a URL, such as a macro that sets properties, you cannot embed the ampersand character (&) in the URL. The ampersand character is a special character in SAS. If you embed an ampersand character, SAS will attempt to resolve the subsequent text as a macro variable.

The Proper Case for Parameter Values

Although SAS is case insensitive, the parameter values passed to SAS Drug Development may be case sensitive. This chapter presents the parameter values in the case that they must be passed to the macros.

Quoting Parameter Values

In order to ensure consistent results, it is recommended that the parameters of type String be wrapped using one of the string functions, e.g. %str(), %nrbquote(). Using double quotes will result in a SAS system syntax error.

SASDRUGDEV_ADDGROUPMEMBER

Description

Adds a user or a group as a member of a group in the specified context.

Syntax

%SASDRUGDEV_ADDGROUPMEMBER(SDD_PATH=*sdd-path,* SDD_GROUP=*sdd-name,* SDD_MEMBER=*sdd-member-identifier* <, SDD_TYPE=*sdd-type,* SDD_GROUP_CONTEXT=*sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) where the group is defined.

sdd-name is the name or the group to be added as a member.

sdd-member-identifier is the id of the user or the name of the group being added.

sdd-type indicates the type of member being added. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which the group being added is defined as a path (e.g., /SAS). This parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

Example Code:

SAS Drug Development Context	Given Name	Content Path
Organization	SAS	/SAS
Project	Project1	/SAS/Project1
Analysis	Analysis1	/SAS/Project1/Analysis1

Add User as a Group member at the project level:

```
%sasdrugdev_AddGroupMember(sdd_path=%str(/SAS/Project1),
sdd_group=%str(PrjGrp1), sdd_member=%str(User1),
sdd_type=%str(USER));
```

Add Group as a Group member at the project level:

```
%sasdrugdev_AddGroupMember(sdd_path=%str(/SAS/Project1),
sdd_group=%str(PrjGrp1), sdd_member=%str(OrgGrp1),
sdd_type=%str(GROUP), sdd_group_context=%str(/SAS));
```

Add User as a Group member at the analysis level:

```
%sasdrugdev_AddGroupMember(sdd_path=%str(/SAS/Project1/Analysis1), sdd_group=%str(AnaGrp1), sdd_member=%str(User1), sdd_type=%str(USER));
```

Add Project Group as a Group member at the analysis level:

```
%sasdrugdev_AddGroupMember(sdd_path=%str(/SAS/Project1/Analysis1), sdd_group=%str(AnaGrp1),
sdd_member=%str(PrjGrp1), sdd_type=%str(GROUP),
sdd_group_context=%str(/SAS/Project1));
```

Add Organization Group as a Group member at the analysis level:

```
%sasdrugdev_AddGroupMember(sdd_path=%str(/SAS/Project1/Analysis1), sdd_group=%str(AnaGrp1),
sdd_member=%str(OrgGrp1), sdd_type=%str(GROUP),
sdd_group_context=%str(/SAS));
```

SASDRUGDEV_ADDMEMBER

Description

Adds a user or a group as a member of the specified context.

Syntax

%SASDRUGDEV_ADDMEMBER(SDD_PATH=*sdd-path,* SDD_MEMBER=*sdd-member-identifier,* <SDD_TYPE=*sdd-type,* *SDD_GROUP_CONTEXT=sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) where the member is being added.

sdd-member-identifier is the id of the user or the name of the group being added.

sdd-type indicates the type of member being added. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which the group is defined as a path (e.g., /SAS). Parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

Example Code:

Adding User as a Member at the project level;

```
%sasdrugdev_AddMember(SDD_PATH=%str(/SAS/Project1),
SDD_MEMBER=%str(User1));
```

Adding Group as a Member at the project level;

```
%sasdrugdev_AddMember(sdd_path=%str(/SAS/Project1),
sdd_member=%str(OrgGrp1), sdd_type=%str(GROUP),
sdd_group_context=%str(/SAS));
```

Adding User as a Member at the analysis level;

```
%sasdrugdev_AddMember(SDD_PATH=%str(/SAS/Project1/Anal
ysis1), SDD_MEMBER=%str(User1));
```

Adding Group defined at Org Level as a Member at the analysis level;

```
%sasdrugdev_AddMember(sdd_path=%str(/SAS/Project1/Analysi
s1), sdd_member=%str(OrgGrp1), sdd_type=%str(GROUP),
sdd_group_context=%str(/SAS));
```

SASDRUGDEV_ADDROLEMEMBER

Description

Adds a user or a group as a member of a role in the specified context.

Syntax

%SASDRUGDEV_ADDROLEMEMBER(SDD_PATH=*sdd-path*, SDD_ROLE=*sdd-name*, SDD_ROLE_CONTEXT=sdd-role-context, SDD_MEMBER=*sdd-member-identifier* <, SDD_TYPE=*sdd-type, SDD_GROUP_CONTEXT=sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) where the role is to be assigned.

sdd-name is the name of the role that the member is to be added to.

sdd-role-context specifies the context in which the role is defined as a path (e.g., /SAS).

sdd-member-identifier is the id of the user or the name of the group being added.

sdd-type indicates the type of member being added. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which the group is defined as a path (e.g., /SAS). Parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

Example Code:

SAS Drug Development Context	Given Name	Content Path
Organization	SAS	/SAS
Project	Project1	/SAS/Project1
Analysis	Analysis1	/SAS/Project1/Analysis1

Adding USER-User1 to DEFINED Project Role - PrjRole1 Assignment;

```
%sasdrugdev_AddRoleMember(sdd_path=%str(/SAS/Project1),
sdd_role=%str(PrjRole1),
sdd_role_context=%str(/SAS/Project1),sdd_member=%str(User1),
sdd_type=%str(USER));
```

Adding OrgGrp1 group defined at Organization Level to DEFINED Project Role - PrjRole1 Assignment;

```
%sasdrugdev_AddRoleMember(sdd_path=%str(/SAS/Project1),
sdd_role=%str(PrjRole1),
sdd_role_context=%str(/SAS/Project1),sdd_member=%str(OrgGr
p1), sdd_type=%str(GROUP), sdd_group_context=%str(/SAS));
```

Adding USER-User1 to INHERITED Project Role - OrgRole1 Assignment;

```
%sasdrugdev_AddRoleMember(sdd_path=%str(/SAS/Project1),
sdd_role=%str(OrgRole1),
sdd_role_context=%str(/SAS),sdd_member=%str(User1),
sdd_type=%str(USER));
```

Adding OrgGrp1 group defined at Organization Level to INHERITED Project Role - OrgRole1 Assignment;

```
%sasdrugdev_AddRoleMember(sdd_path=%str(/SAS/Project1),
sdd_role=%str(OrgRole1),
sdd_role_context=%str(/SAS),sdd_member=%str(OrgGrp1),
sdd_type=%str(GROUP), sdd_group_context=%str(/SAS));
```

SASDRUGDEV_ADDROLEPRIVILEGE

Description

Adds a privilege to a role in the specified context.

Syntax

%SASDRUGDEV_ADDROLEPRIVILEGE(SDD_PATH=*sdd-path,* SDD_ROLE=*sdd-name,* SDD_PRIVILEGE=*sdd-privilege-name*);

sdd-path	is the path (starting at the root) of the context object (organization, project, or analysis) where the role is defined.
sdd-name	is the name of the role to add to the privilege.
sdd-privilege-name	is the name of the privilege being added.

Example Code:

Use %sasdrugdev_getContextPrivileges to fetch a list of privileges from a defined role at a specified context. This macro returns a SAS data set work.sddgetcontextprivileges. Use one of the PrivilegeName values to add a privilege to a Role.

```
%sasdrugdev_getContextPrivileges(sdd_path=%str(/SAS/Project1
));

%sasdrugdev_AddRolePrivilege(sdd_path=%str(/SAS/Project1),
sdd_role=%str(PrjRole1),
sdd_privilege=%str(PRIVILEGE_SIGN_FILE));
```

SASDRUGDEV_ASSIGNEDROLEEXISTS

Description

Determines whether a role exists (as defined or inherited) for the specified context. A message will be printed to the log stating the existence status of the role. The macro variable _sddAssignedRoleExists_ will be set to 1 if the role is present, 0 if it is not.

Syntax

%SASDRUGDEV_ASSIGNEDROLEEXISTS (SDD_PATH=*sdd-path,* SDD_ROLE=*sdd-name, SDD_ROLE_CONTEXT=sdd-role-context*);

sdd-path	is the path (starting at the root) of the context object (organization, project, or analysis).
sdd-name	is name of the role.
sdd-role-context	specifies the context in which the group is defined as a path (e.g., /SAS).

SASDRUGDEV_ASSIGNROLE

Description

Assigns a role to the specified context as an inherited role.

Syntax

%SASDRUGDEV_ASSIGNROLE(SDD_PATH=*sdd-path*, SDD_ROLE=*sdd-role, SDD_ROLE_CONTEXT=sdd-role-context*);

sdd-path	is the path (starting at the root) of the context object (project or analysis) to assign the role to.
sdd-role	is the name of the role being assigned.
sdd-role-context	specifies the context in which the role is defined as a path (e.g., /SAS).

SASDRUGDEV_COPY

Description

Copies a file or folder to the specific target location. The copy of versioned files will not be versioned. The children of the copied folder will also be copied to the new location in their original directory structure.

Syntax

%SASDRUGDEV_COPY(SDD_SOURCE=*sdd-source, SDD_TARGET=sdd-target);*

sdd- source	is the path (starting at the root) and the name of the object in SAS Drug Development to be copied.
sdd-target	is the path (starting at the root) in SAS Drug Development where the object is to be copied. The target folder must already exist.

SASDRUGDEV_CREATEANALYSIS

Description

Creates an analysis in the SAS Drug Development repository. The parent project must already exist.

Syntax

%SASDRUGDEV_CREATEANALYSIS(SDD_PATH=*sdd-path*) ;

sdd-path is the path (starting at the root) of the analysis to be created in the SAS Drug Development repository. The path must be a valid path structure.

SASDRUGDEV_CREATEFILE

Description

Creates an object in the SAS Drug Development repository. All parent folders needed to create the full path for the file will be created if they do not already exist.

Syntax

%SASDRUGDEV_CREATEFILE(LOCAL_PATH=*local-path*, SDD_PATH=*sdd-path* < *SDD_VERSIONING=sdd-versioning*, ,SDD_COMMENT=*sdd_comment, SDD_VERSION=sdd- version*>) ;

local-path is the absolute path and name of the file on the local computer.

sdd-path is the path (starting at the root) and the name of the content object to be created in the SAS Drug Development repository. The path may not be a container.

sdd_versioning is an optional text string to indicate whether an object being created in the repository should be versioned. Valid values are 0 and 1. The default value is 0, which will create a non-versioned object. A value of 1 will create a versioned object.

sdd-comment is an optional text string to be inserted as the comment for a versioned object in the repository. This option is ignored for non-versioned objects in the repository.

sdd-version is an optional text string to indicate the version number to assign to an object being created in the repository. The value must be in the valid format including a decimal point. This option is ignored when creating non-versioned objects in the repository.

SASDRUGDEV_CREATEFOLDER

Description

Creates a folder in the SAS Drug Development repository. All parent folders needed to create the full path for the folder will be created if they do not already exist.

Syntax

%SASDRUGDEV_CREATEFOLDER(SDD_PATH=*sdd-path*) ;

sdd-path is the path (starting at the root) of the folder to be created in the SAS Drug Development repository. The path must be a valid path structure.

SASDRUGDEV_CREATEGROUP

Description

Creates a group in the specified context.

Syntax

%SASDRUGDEV_CREATEGROUP (SDD_PATH=*sdd-path*, SDD_GROUP=*sdd-name* <, *SDD_DESCRIPTION=sdd-description*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis).

sdd-name is the name of the new group.

sdd-description is a description for the new group.

SASDRUGDEV_CREATEPROJECT

Description

Creates a project in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_CREATEPROJECT(SDD_PATH=*sdd-path*) ;

sdd-path is the path (starting at the root) of the project to be created in the SAS Drug Development repository. The path must be a valid path structure.

SASDRUGDEV_CREATEROLE

Description

Creates a role in the specified context.

Syntax

%SASDRUGDEV_CREATEROLE (SDD_PATH=*sdd-path,* SDD_ROLE=*sdd- name* <, *SDD_DESCRIPTION=sdd-description*>);

sdd-path	is the path (starting at the root) of the context object (organization, project, or analysis).
sdd-name	is the name of the new role.
sdd-description	is a description for the new role.

SASDRUGDEV_DELETEANALYSIS

Description

Deletes an analysis from the SAS Drug Development repository.

Syntax

%SASDRUGDEV_DELETEANALYSIS (SDD_PATH=*sdd-path*);

sdd-path	is the path (starting at the root) of the analysis to be deleted from the SAS Drug Development repository. The path must be a valid path structure.

SASDRUGDEV_DELETEGROUP

Description

Deletes a group from the specified context.

Syntax

%SASDRUGDEV_DELETEGROUP (SDD_PATH=*sdd-path,* SDD_GROUP=*sdd-name*);

sdd-path	is the path (starting at the root) of the context object (organization, project, or analysis).
sdd-name	is the name of the group to be deleted.

SASDRUGDEV_DELETEOBJECT

Description

Deletes a content object from the SAS Drug Development repository.

Syntax

%SASDRUGDEV_DELETEOBJECT(SDD_PATH=*sdd-path*);

sdd-path is the path (starting at the root) and name of the content object in the SAS Drug Development repository to delete.

SASDRUGDEV_DELETEPROJECT

Description

Deletes a project from the SAS Drug Development repository.

Syntax

%SASDRUGDEV_DELETEPROJECT (SDD_PATH=*sdd-path*);

sdd-path is the path (starting at the root) of the project to be deleted from the SAS Drug Development repository. The path must be a valid path structure.

SASDRUGDEV_DELETEROLE

Description

Deletes a role from the specified context.

Syntax

%SASDRUGDEV_DELETEROLE (SDD_PATH=*sdd-path*, SDD_ROLE=*sdd-name*);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis).

sdd-name is the name of the role to be deleted.

SASDRUGDEV_GETACLS

Description

Returns a SAS data set that contains the access control list for objects in the SAS Drug Development repository. For container objects, both the DEFAULT and CURRENT access control lists are returned. For non-container objects, only the CURRENT access control lists is returned.

Syntax

%SASDRUGDEV_GETACLS(SDD_PATH=*sdd-path*, <SAS_DSNAME=*SAS-data-set,* SDD_RECURSIVE=*recursion-level* >);

sdd-path is the path (starting at the root) and the name of the object in the SAS Drug Development repository. This path is case sensitive.

recursion_level is the level of recursion when the path is a container. Valid values are 0, 1, 2, or 99 as described below. The default value is 0. For files, the value will be ignored.

- 0 - returns acls for the input container only.
- 1 - returns acls for the container and all objects within the container, but not sub containers.
- 2 - returns acls for the container, all objects within the container, and one level of sub containers.
- 99 - returns acls for the container and all of the sub containers and objects (it traverses the entire tree).

SAS-data-set is the name of the output SAS data set that will contain the metadata for the access control lists. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETACL**.

The data set contains a row for each access control entry and columns the following names. The data set will be sorted by fullPath.

- fullPath: the fully qualified name of the object
- path: the path to the object without the object name
- ACEname: the name of the object (either file or container) that the acl is set on. Also called the principal name.
- objectType: describes the type of the object (folder, data set, program, etc.)
- isContainer: a numeric flag that indicates whether the object is a container object (0=no, 1=yes)
- Owner: the name of the user who owns the object at the fullpath (Current) location.
- AclType: will be either Default or Current. Files only have Current acls. Containers have both Default and Current acls.
- principalName: the identifier for a user or the name for a group.
- principalType: specifies if the principal is a USER, GROUP, ACLOWNER, or ACLMEMBERS.
- grpSrcCtxt: for group principals, specifies the context path where the group was defined.
- adminPermission: principal's permission to administer the object.
- readPermission: principal's permission to read the object.

- writePropPerm: principal's permission to update the object properties.
- writeContentPerm: principal's permission to update the object content.
- deletePermission: principal's permission to delete the object.

- Permission values are specified as follows.
  - 1 = Allowed
  - -1 = Denied
  - 0 = not specified (Inherited)

SASDRUGDEV_GETASSIGNEDMEMBERS

Description

Returns a SAS data set that contains the metadata for all of the members assigned to a context object in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETASSIGNEDMEMBERS(SDD_PATH=*sdd-path* <,SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) of the context object in the SAS Drug Development repository.

SAS-data-set is the name of the output SAS data set that contains the metadata for the members. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETASSIGNEDMEMBERS**.

The data set contains a row for each member in the context object and columns with the following names. The data set will be sorted by member type and principalId.

- contextPath: The full path of the context object in the SAS Drug Development Repository.
- type: The member type. Valid values are USER and GROUP.
- principalId: The name of the member. The returned values will be either a group name or a user id.
- descriptor: The description of the object. The returned values will be either a group name or a user display name.
- grpSrcCtxt: The context path in which the group member was defined.

SASDRUGDEV_GETCHILDREN

Description

Returns a SAS data set that contains the metadata for objects within a container in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETCHILDREN(SDD_PATH=*sdd-path*
<SDD_RECURSIVE=*recursion-level*,SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) and name of the container object in the SAS Drug Development repository.

recursion_level is the level of recursion. Valid values are 0, 1, 2, or 99. The default value is 0.

- 0 - Returns the metadata for the container only.
- 1 - Returns the metadata for the container and all objects within the container, but not sub containers.
- 2 - Returns the metadata for the container, all objects within the container, and one level of sub containers.
- 99 - Returns the metadata for the container, all sub containers and all objects (it traverses the entire tree).

SAS-data-set is the name of the output SAS data set that will contain the metadata for the objects. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETCHILDREN**.

The data set contains a row for each object in the container object and columns with the following names. The data set will be sorted by path and then name within path.

- name: The name of the object.
- path: The path to the object without the object name.
- fullPath: The fully qualified name of the object.
- description: The description of the object.
- isContainer: A numeric flag that indicates whether the object is a container object (0=no, 1=yes).
- isVersioned: A numeric flag that indicates whether the object is currently versioned (0=no, 1=yes).
- versionLabel: The label associated with this specific version.
- isCheckedOut: A numeric flag that indicates whether the object is currently checked out (0=no, 1=yes).
- isLocked: A numeric flag that indicates whether the object is currently locked (0=no, 1=yes).

- size: The size of the object in bytes represented as a string (will be 0 for containers).
- contentType: The content or MIME type of object.
- objectType: Describes the type of the object (folder, data set, program, etc).
- createdBy: The userid that created the object.
- createdOn: The full date and time when the object was created. This value is represented as a String.
- dateCreated: The date when the object was created as a SAS Datetime format.
- lastModifiedBy: The userid that last modified the object.
- lastModifiedOn: The date when the object was last modified. This value is represented as a String.
- dateLastModified: The date when the object was last modified as a SAS Datetime format.
- propsLastModifiedBy: The userid that last modified the properties for the object.
- state: This variable is for an Analysis or Project object only. It indicates if the state is DEV, PROD, or CLOSED.
- stateComment: This variable is for Analysis or Project only and provides any comments attached by a user when changing the state of a project or analysis.
- isSigned: Indicates if the object has electronic signatures attached to it.

SASDRUGDEV_GETCONTENTSOURCE

Description

Returns the current setting for the Macro API content source (as set by sasdrugdev_setcontentsource()). The value of the current setting is written to the log and also stored in the macro variable _sddContentSource_.

Syntax

%SASDRUGDEV_GETCONTENTSOURCE() ;

SASDRUGDEV_GETCONTEXTPRIVILEGES

Description

Returns a SAS data set that contains the metadata for all of the privileges available at the given context level in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETCONTEXTPRIVILEGES(SDD_PATH=*sdd-path* <,SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) of the context object in the SAS Drug Development repository.

SAS-data-set is the name of the output SAS data set that contains the metadata for the privileges. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETCONTEXTPRIVILEGES**.

The data set contains a row for each privilege defined at the context and columns with the following names. The data set will be sorted by privilegeId.

- contextPath: The full path of the context object in the SAS Drug Development Repository.
- privilegeId: The identifier of the privilege.
- privilegeName: The name of the privilege.

SASDRUGDEV_GETGROUPMEMBERS

Description

Returns a SAS data set that contains the metadata for all of the members assigned to a group within a context object in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETGROUPMEMBERS(SDD_PATH=*sdd-path*, SDD_GROUP=*sdd-group* <,SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) of the context object in the SAS Drug Development repository where the group is defined.

sdd-name is the name of the group.

SAS-data-set is the name of the output SAS data set created that contains the metadata for the members. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETGROUPMEMBERS**.

The data set contains a row for each member in the group and columns with the following names. The data set will be sorted by member type and principalId.

- contextPath: The full path of the context object in the SAS Drug Development Repository.
- type,: The member type. Valid values are USER and GROUP.

- principalId: The name of the member. Valid values are group name or user id
- descriptor: The description of the object.. Valid values are group name or user display name
- grpSrcCtxt: The context in which the of group member was defined.

SASDRUGDEV_GETGROUPS

Description

Returns a SAS data set that contains the metadata for all of the groups defined within a context object in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETGROUPS(SDD_PATH=*sdd-path* <,SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) of the context object in the SAS Drug Development repository.

SAS-data-set is the name of a SAS data set to create that will contain the metadata for the groups. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETGROUPS**.

The data set contains a row for each group in the context object and columns with the following names. The data set will be sorted by group name.

- contextPath: The full path of the context object in the SAS Drug Development Repository.
- name: The name of the group.
- description: The group description.

SASDRUGDEV_GETOWNER

Description

Returns the current owner of an object within the organization in the SAS Drug Development repository. The macro variable _sddOwner_ will be set with the owner's user id. The value of _sddOwner_ will be <creator> for the DEFAULT permissions on a containers with no specific user designated as the owner.

Syntax

For files:

%SASDRUGDEV_GETOWNER(SDD_PATH =*sdd-path* <, SDD_ACLTYPE =*sdd-acl-type* >);

For container objects:

%SASDRUGDEV_GETOWNER(SDD_PATH =*sdd-path*, SDD_ACLTYPE =*sdd-acl-type*);

sdd-path is the path (starting at the root) and name of the content object in the SAS Drug Development repository. This can be either a file or a container.

sdd-acl-type is the type of permissions being returned. The valid values are case-insensitive DEFAULT and CURRENT. Files only have CURRENT permissions; therefore the value is not required when *sdd-path* is a file. For container objects the value must be specified. The operation will fail if DEFAULT is requested for a file path.

SASDRUGDEV_GETPROPERTIES

Description

Returns a SAS data set that contains the properties and attributes for the object in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETPROPERTIES(SDD_PATH=*sdd-path*< SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) and name of the object in the SAS Drug Development repository.

SAS-data-set is the name of the output SAS data set that will contain the metadata for the object. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETPROPERTIES**.

The data set will contain two variables representing the name/value pair for each element of metadata for the object. The data set will be sorted by the NAME variable.

The following metadata will be represented in the data set. Other values will vary based on the type of object specified in *sdd-path*. Extended attributes are included but the names are represented by the system identifier.

- objectName: The name of the object without the path.
- objectPath,: The fully qualified name of the object.

- objectParentPath,: The path to the object, excluding the object name.
- objectType: The type of object (i.e., project, container).

SASDRUGDEV_GETROLEMEMBERS

Description

Returns a SAS data set that contains the metadata for all of the members assigned to a role within a context object in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETROLEMEMBERS(SDD_PATH=*sdd-path*, SDD_ROLE=*sdd-name*, SDD_ROLE_CONTEXT=sdd-role-context, <,SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) of the context object in the SAS Drug Development repository where the role is assigned.

sdd-name is the name of the role.

sdd-role-context specifies the context in which the role is defined as a path (e.g., /SAS).

SAS-data-set is the name of the output SAS data set that contains the metadata for the members. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETROLEMEMBERS**.

The data set contains a row for each member in the role and columns with the following names. The data set will be sorted by member type and principalId.

- contextPath: The full path of the context object in the SAS Drug Development repository.
- Type: The member type. Valid values are user or group.
- principalId: The name of the member. Valid values are group name or user id.
- Descriptor: The description of the object. Valid values are group name or user display name
- grpSrcCtxt: The context in which the of group member was defined.

SASDRUGDEV_GETROLEPRIVILEGES

Description

Returns a SAS data set that contains the metadata for all of the privileges assigned to a role within a context in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETROLEPRIVILEGES(SDD_PATH=*sdd-path*, SDD_ROLE=*sdd-name* <,SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) of the context object in the SAS Drug Development repository where the role is defined.

sdd-name is the name of the role.

SAS-data-set is the name of the output SAS data set that contains the metadata for the privileges. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETROLEPRIVILEGES**.

The data set contains a row for each privilege in the role and columns with the following names. The data set will be sorted by privilegeId.

- contextPath: The full path of the context object in the SAS Drug Development repository.
- roleName: The name of the role.
- privilegeId: The identifier of the privilege.
- privilegeName: The name of the privilege.

SASDRUGDEV_GETROLES

Description

Returns a SAS data set that contains the metadata for all of the roles defined and, if specified, inherited within a context object in the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETROLES(SDD_PATH=*sdd-path* <,SAS_DSNAME=*SAS-data-set, SDD_GETINHERITED=inheritance-flag*>);

sdd-path is the path (starting at the root) of the context object in the SAS Drug Development repository.

SAS-data-set is the name of the output SAS data set that will contain the metadata for the roles. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETROLES**.

The data set contains a row for each role in the context object as well as columns with the following names. The data set will be sorted by role name.

- contextPath: The full path of the context object in the SAS Drug Development repository.
- Name: The name of the role.
- Description: The role description.
- roleSrcCtxt: The context in which the of role was defined.

- isInherited: A flag indicating whether or not the role is inherited. The valid values are 0 and 1. This variable is not included if SDD_GETINHERITED is not valid or set to 0.

inheritance-flag indicates whether or not to include inherited roles. The valid values are 0 and 1 to exclude or include inherited roles respectively. The default is 1.

SASDRUGDEV_GETSDDAPIVERSION

Description

Reports the version and build number for the current SAS Drug Development client API version. The value is printed in the SAS log.

Syntax

%SASDRUGDEV_GETSDDAPIVERSION();

SASDRUGDEV_GETVERSIONS

Description

Returns a SAS data set that contains the metadata for all of the versions of an object within the SAS Drug Development repository.

Syntax

%SASDRUGDEV_GETVERSIONS(SDD_PATH=*sdd-path,* SAS_DSNAME=*SAS-data-set*>);

sdd-path is the path (starting at the root) and name of the object in the SAS Drug Development repository.

SAS-data-set is the name of the output SAS data set that will contain the metadata for the versions. Specify *SAS-data-set* as *libref.dataset*. The default value is **WORK.SDDGETVERSIONS**.

The data set contains a row for each version of the object and columns with the following names. The most recent version will be the first observation in the data set.

- name: The name of the object.
- Path: The path to the object , excluding the object name.
- fullPath: The fully qualified name of the object.
- description: The description of the object.
- isContainer: A numeric flag that indicates whether the object is a container object (0=no, 1=yes).

- isVersioned: A numeric flag that indicates whether the object is currently versioned (0=no, 1=yes).
- versionLabel: The label associated with this specific version.
- isCheckedOut: A numeric flag that indicates whether the object is currently checked out (0=no, 1=yes).
- isLocked: A numeric flag that indicates whether the object is currently locked (0=no, 1=yes).
- Size: The size of the object in bytes represented as a string (will be 0 for containers).
- contentType: The content or MIME type of object.
- objectType: Describes the type of the object (folder, data set, program, etc.).
- created: The userid that created the object.
- createdOn: The full date and time the object was created represented as a String
- dateCreated: The date the object was created as a SAS Datetime format.
- lastModifiedBy: The userid that last modified the object.
- lastModifiedOn: The date the object was last modified represented as a String.
- dateLastModified: The date the object was last modified as a SAS Datetime format.
- propsLastModifiedBy: The userid that last modified the properties for the object.
- State: For an Analysis or Project object only. Indicates the state as being Dev, Prod, or Closed.
- stateComment: For Analysis or Project only. This is a comment that was entered when the state was changed.
- isSigned: Indicates if the object has electronic signatures attached to it.

SASDRUGDEV_GROUPEXISTS

Description

Determines whether a group is defined for the specified context. A message will be printed to the log stating the existence status of the group. The macro variable _sddGroupExists_ will be set to 1 if the group is defined, 0 if it is not.

Syntax

%SASDRUGDEV_GROUPEXISTS (SDD_PATH=*sdd-path,* SDD_GROUP=*sdd-name*);

sdd-path	is the path (starting at the root) of the context object (organization, project, or analysis).
sdd-name	is name of the group.

SASDRUGDEV_ISCONTAINER

Description

Returns a SAS macro variable _sddIsContainer_ that contains a value of 1 when the specified object is a container object. Containers objects include folders, projects, analyses, and the organization.

Syntax

%SASDRUGDEV_ISCONTAINER(SDD_PATH=*sdd-path*);

sdd-path is the path (starting at the root) and name of the object in the SAS Drug Development repository

SASDRUGDEV_ISGROUPMEMBER

Description

Determines whether a user or group is a member of a group in the specified context. A message will be printed to the log stating the membership status. The macro variable _sddIsGroupMember_ will be set to 1 if the user or group is a member of the group, 0 if it is not.

Syntax

%SASDRUGDEV_ISGROUPMEMBER(SDD_PATH=*sdd-path,* SDD_GROUP=*sdd-name*, SDD_MEMBER=*sdd-member-identifier,* < SDD_TYPE=*sdd-type, SDD_GROUP_CONTEXT=sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) where the group is assigned.

sdd-name is the name of the group.

sdd-member-identifier is the user id or group name.

sdd-type indicates the type of member. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which the member group is defined as a path (e.g., /SAS). Parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

SASDRUGDEV_ISMEMBER

Description

Determines whether a user or group is a member of the specified context. A message will be printed to the log stating the membership status. The macro variable _sddIsMember_ will be set to 1 if the user or group is a member, 0 if it is not. Groups defined at the context level will not be considered as members of that context.

Syntax

%SASDRUGDEV_ISMEMBER(SDD_PATH=*sdd-path ,* SDD_MEMBER=*sdd-member-identifier,* <SDD_TYPE=*sdd-type, SDD_GROUP_CONTEXT=sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis).

sdd-member-identifier is a user id or group name.

sdd-type indicates the type of member. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which a group member is defined as a path (e.g., /SAS). This parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

SASDRUGDEV_ISROLEMEMBER

Description

Determines whether a user or group is a member of a role in the specified context. A message will be printed to the log stating the membership status. The macro variable _sddIsRoleMember_ will be set to 1 if the user or group is a member of the role, 0 if it is not.

Syntax

%SASDRUGDEV_ISROLEMEMBER(SDD_PATH=*sdd-path,* SDD_ROLE=*sdd-name,* SDD_ROLE_CONTEXT=sdd-role-context, SDD_MEMBER=*sdd-member-identifier,* < SDD_TYPE=*sdd-type, SDD_GROUP_CONTEXT=sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) where the role is assigned.

sdd-name is the name of the role.

sdd-role-context specifies the context in which the role is defined as a path (e.g., /SAS).

sdd-member-identifier is the user id or group name.

sdd-type indicates the type of member. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which the group is defined as a path (e.g., /SAS). This parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

SASDRUGDEV_ISROLEPRIVILEGE

Description

Determines whether a privilege is assigned to a role in the specified context. A message will be printed to the log stating the membership status. The macro variable _sddIsRolePrivilege_ will be set to 1 if the user or group is a member of the role, 0 if it is not.

Syntax

%SASDRUGDEV_ISROLEPRIVILEGE(SDD_PATH=*sdd-path,* SDD_ROLE=*sdd-name*, SDD_PRIVILEGE=*sdd-privilege-name*);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) where the role is assigned.

sdd-name is the name of the role.

sdd-privilege-name is the name of the privilege.

SASDRUGDEV_ISVERSIONED

Description

Returns a SAS macro variable _sddIsVersioned_ that contains a value of 1 when the specified file in the repository has versioning enabled.

Syntax

%SASDRUGDEV_ISVERSIONED(SDD_PATH=*sdd-path*);

sdd-path is the path (starting at the root) and name of the object in the SAS Drug Development repository.

SASDRUGDEV_LOGIN

Description

Creates a connection to the SAS Drug Development instance. This macro is only used when running the macros from a PC SAS session. It must be called before any other SDD macros in order to establish your SDD session.

When running the macros from within an active SDD web browser session, this macro call is not required and will result in an error.

Syntax

%SASDRUGDEV_LOGIN(SDD_URL=*url*, SDD_USERID=*user-ID*, SDD_PASSWORD=*password*) ;

url	is the URL of the SAS Drug Development instance to use in the form **https://sddname.sas.com**
user-ID	is the SAS Drug Development ID for the user account.
Password	is the SAS Drug Development password for the user account.

Note: The parameters passed to the login macro are often strings containing special characters which may cause problems when resolved by SAS. You will probably want to wrap each of the parameters in a SAS function call to prevent any issues.

For example:

```
%sasdrugdev_login(sdd_url=%str(https://sddinstance),
                  Sdd_userid=%str(sdd-user-ID),
                  Sdd_password=%str (sddpassword ));
```

Or if your password may contain a "%" or "&":

```
%sasdrugdev_login(sdd_url=%str(https://sddinstance),
                  Sdd_userid=%str(sdd-user-ID),
                  Sdd_password=%nrbquote(sddpassword ));
```

SASDRUGDEV_LOGOUT

Description

Closes the open connection to SAS Drug Development. This should be the last macro you call in any program. This macro is only used when running the macros from a PC SAS session.

When running the macros from within an active SAS Drug Development web browser session, this macro call is not required and will result in an error.

Syntax

%SASDRUGDEV_LOGOUT() ;

SASDRUGDEV_OBJECTEXISTS

Description

Determines whether an object exists in the SAS Drug Development Repository. A message will be printed to the log stating the existence status of the object. The macro variable _sddObjectExists_ will be set to 1 if the object exists, 0 if it does not.

Syntax

%SASDRUGDEV_OBJECTEXISTS(SDD_PATH =*sdd-path*);

sdd-path is the path (starting at the root) and name of the content object in the SAS Drug Development repository. This can be either a file or a container.

SASDRUGDEV_REMOVEGROUPMEMBER

Description

Removes a user or a group as a member of a group in the specified context.

Syntax

%SASDRUGDEV_REMOVEGROUPMEMBER(SDD_PATH=*sdd-path,* SDD_GROUP=*sdd-name*, SDD_MEMBER=*sdd-member-identifier*, < SDD_TYPE=*sdd-type*, *SDD_GROUP_CONTEXT=sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) where the group is defined.

sdd-name is the name of the group that the member is to be removed from.

sdd-member-identifier is the id of the user or the name of the group being removed.

sdd-type indicates the type of member being removed. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which the group being removed is defined as a path (e.g., /SAS). This parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

SASDRUGDEV_REMOVEMEMBER

Description

Removes a user or a group from membership of the specified context.

Syntax

%SASDRUGDEV_REMOVEMEMBER(SDD_PATH=*sdd-path* SDD_MEMBER=*sdd-member-identifier*, <SDD_TYPE=*sdd-type, SDD_GROUP_CONTEXT=sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) to remove the member from.

sdd-member-identifier is the id of the user or the name of the group being removed.

sdd-type indicates the type of member being removed. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which the group is defined as a path (e.g., /SAS). This parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

SASDRUGDEV_REMOVEROLEMEMBER

Description

Removes a user or a group as a member of a role in the specified context.

Syntax

%SASDRUGDEV_REMOVEROLEMEMBER(SDD_PATH=*sdd-path,* SDD_ROLE=*sdd-name*, SDD_ROLE_CONTEXT=sdd-role-context, SDD_MEMBER=*sdd-member-identifier*, < SDD_TYPE=*sdd-type, SDD_GROUP_CONTEXT=sdd-group-context*>);

sdd-path is the path (starting at the root) of the context object (organization, project, or analysis) where the role is assigned.

sdd-name is the name of the role that the member is to be removed from.

sdd-role-context specifies the context in which the role is defined as a path (e.g., /SAS).

sdd-member-identifier is the id of the user or the name of the group being removed.

sdd-type indicates the type of member being removed. Valid values are USER and GROUP (non-case-specific). The default value is USER.

sdd-group-context specifies the context in which the group is defined as a path (e.g., /SAS). This parameter is required if SDD_TYPE=GROUP. Otherwise, it will be ignored.

SASDRUGDEV_REMOVEROLEPRIVILEGE

Description

Removes a privilege from a role in the specified context.

Syntax

%SASDRUGDEV_REMOVEROLEPRIVILEGE(SDD_PATH=*sdd-path,* SDD_ROLE=*sdd-name,* SDD_PRIVILEGE=*sdd-privilege-name*);

sdd-path	is the path (starting at the root) of the context object (organization, project, or analysis) where the role is defined.
sdd-name	is the name of the role to remove from the privilege.
sdd-privilege-name	is the name of the privilege being removed.

SASDRUGDEV_ROLEEXISTS

Description

Determines whether a role is defined for the specified context. A message will be printed to the log stating the existence status of the role. The macro variable _sddRoleExists_ will be set to 1 if the role is defined, 0 if it is not.

Syntax

%SASDRUGDEV_ROLEEXISTS (SDD_PATH=*sdd-path*, SDD_ROLE=*sdd-name*);

sdd-path	is the path (starting at the root) of the context object (organization, project, or analysis).
sdd-name	is name of the role.

SASDRUGDEV_UNASSIGNROLE

Description

Unassigns a role from the specified context as an inherited role.

Syntax

%SASDRUGDEV_UNASSIGNROLE(SDD_PATH=*sdd-path,* SDD_ROLE=*sdd-role*, *SDD_ROLE_CONTEXT=sdd-role-context*);

sdd-path	is the path (starting at the root) of the context object (project or analysis) from which to unassign the role.
sdd-role	is the name of the role being unassigned.
sdd-role-context	specifies the context in which the role is defined as a path (e.g., /SAS).

SASDRUGDEV_UPDATEACLS

Description

This macro sets access permissions on repository objects using a SAS data set as input. The permission lists (known as Access Control Lists or ACLs) for multiple objects can be modified using a single data set. Access permissions are applied only to objects in the Repository.

There are two types of access permission lists: Current and Default. The *Current* access permissions are applied to an individual object. The *Default* access permissions apply only to container objects (e.g. folders, projects,) and define the permissions that will be inherited by any new objects added to the container. In the context of this macro the access permission type is referred to as *aclType*. All records in the input data set that pertain to the same repository object and aclType will be processed together as a single ACL.

A set of permissions for a specific repository object for an individual principal within an access permission type are referred to as an Access Control Entry (ACE). An ACE is represented as a single row in the input data set.

An ACE is defined by the following characteristics:

- Fullpath – Full path of the repository object (file or container) to which the permissions are being assigned
- aclType – which access permission type is being assigned (default or current)
- principal name – the name of the user or group to assign the permission to
- principal type – identifies the type of the principal. The principal can be either a user, group, members, or owner.
- set of permissions – the specific permissions being assigned (e.g. administration, read, properties write, content write, delete)

The output data set from the sasdrugdev_getACLs macro contains all the necessary variables and data needed as the input data set for the ***sasdrugdev_updateACLs*** macro. Modify the data set with any changes prior to a call the sasdrugdev_updateACLs.

- To add a principal with specific access permissions to a repository object, add a new row to the data set for that principal.
- To modify the access permissions for a principal already associated with a repository object, edit the appropriate row in the data set.
- To remove a principal with specific access permissions to a repository object, delete the appropriate row from the data set. Note that the Owner and

Members access control entries cannot be removed from the object. Removing them from the data set will cause a failure.

- To maintain the existing permissions for a principal already associated with a repository object, leave the record as is in the data set. The record must remain otherwise the access permissions for this principal will be removed.

Error and Message Handling

The macro performs a series of validation procedures to determine that the input data set has the necessary structure for processing. If any of these validation procedures fail, an appropriate error message is printed to the log and the macro stops additional processing.

In addition to other modifications, the macro sorts the input data set and stores the modified data in a temporary data set called work.__SASMacro_acltemp__. In the event that an error occurs during processing this temporary data set is retained as a reference. If all updates are performed successfully, this data set will be deleted from the work library. A message will be printed to the log reporting whether the data set was retained or not. Regardless of success or failure of previous runs, this data set will be deleted at the beginning of each new run of the macro. The data values and sort order of the original input data set are maintained.

All records pertaining to the same repository object and aclType are processed as a single access control list by the macro. Messages will be printed to the log when a complete list has been processed reporting the success or failure of updating the object's ACL. If the update of the ACL fails, the macro will stop processing.

If an error occurs while processing a specific access control entry (i.e. a principal), the macro will stop processing and an error message will be written to the log including the observation number for the entry that failed. The observation number refers to the observation in the interim data set (work__SASMacro_acltemp__) created by the macro. Successful updates to ACLs made prior to the observation containing the error will be maintained in SAS Drug Development. Changes made to access permissions for entries that are part of the same ACL that contained the observation with the error will be lost.

Syntax

%SASDRUGDEV_UPDATEACLS(SAS_DSNAME=*SAS-data-set*);

SAS-data-set is the name of a SAS data set containing the permission information for all the objects to be modified. Specify *SAS-data-set* as *libref.dataset*.

The data set must contain, at minimum, the variables listed below. All required variables must be of the type character. Any additional variables will be ignored.

- fullPath: the fully qualified case-sensitive path of the object including object name
- aclType: indicates which ACL to modify for the object. For files objects, the variable must be present but the value will be ignored. For container objects, the value must be either DEFAULT or CURRENT (case-insensitive).
- principalType: indicates the type of ACE being updated. Valid values are USER, GROUP, ACLOwner, and ACLMembers (case-insensitive).
- principalName: the name of the principal ACE being updated or added. For principalType USER, this must be a valid userid; for principalType GROUP, this must be a valid group name. For principalType ACLOwner, this must have the value OWNER. For principalType ACLMembers, this must have the value MEMBERS. For principalTypes GROUP, the principalName is case-sensitive, otherwise it is not.
- grpSrcCtxt: the case-sensitive path of the context (organization, project, or analysis) where the group indicated by principalType and principalName is defined. For principal type GROUP, this value is required, otherwise the variable must be present but the value will be ignored.
- adminPermission: the permission level being set for the ACE's ability to administer the object. Valid values are described below.
- readPermission: the permission level being set for the ACE's ability to view the object. Valid values are described below.
- writePropPerm: the permission level being set for the ACE's ability to update the object's properties. Valid values are described below.
- writeContentPerm: the permission level being set for the ACE's ability to write to the object's content. Valid values are described below.
- deletePermission: the permission level being set for the ACE's ability to the delete object. Valid values are described below.

Valid values for permissions are as follows. All values must be of type character.

- 1 for GRANT
- -1 for DENY
- 0 for inherited.

See also

- sasdrugdev_getacls()
- sasdrugdev_updateowner()
- sasdrugdev_getowner()

Example Code:

```
* Get Permissions for Folder;
        %sasdrugdev_getACLs(sdd_path=%str(/SAS/Study1/Files/test2),
        sdd_recursive=0);


         data UpdateACLS;
         set sddGetACLS end=eof;


         output;


         if eof then do;
           * Adding User Permission under DEFAULT;
           principalName = "user1";   * Name of Principal;
           principalType = "User";     * Principal Type;
           grpSrcCtxt = "";              * No Group Context Required for
       User;
           aclType="Default";            * Permission Type;
           adminPermission  = "0";    * Admin Permission;
           readPermission   = "1";      * Read Permission;
           writePropPerm    = "1";     * Write Properties Permission;
           writeContentPerm = "1";    * Write Content Permission;
           deletePermission = "1";      * Delete Permission;
           output;


           * Adding Group Permission under CURRENT;
           principalName = "ogroup1";  * Name of Principal;
           principalType = "Group";      * Principal Type;
           grpSrcCtxt = "/SAS";          * No Group Context Required for
       User;
           aclType="Current";            * Permission Type;
           adminPermission  = "0";      * Admin Permission;
           readPermission   = "1";       * Read Permission;
           writePropPerm    = "1";       * Write Properties Permission;
           writeContentPerm = "1";      * Write Content Permission;
           deletePermission = "1";       * Delete Permission;
```

```
    output;
  end;

run;

%sasdrugdev_updateACLS(sas_dsname=UpdateACLS);
```

Note: Access Control Entries (ACEs) with values of 0 for all permission variables will be removed from or will not be added to the ACL.

Note: Users and groups being added as access control entries must be members in the context (organization, project, analysis) containing the object being updated.

SASDRUGDEV_UPDATEFILE

Description

Updates the contents of an existing object in the SAS Drug Development repository. If the object has versioning enabled, a new minor version of the object will be created. If the object is not being versioned, the existing contents will be replaced.

Syntax

%SASDRUGDEV_UPDATEFILE(LOCAL_PATH=*local-path*, SDD_PATH=*sdd-path* <, SDD_COMMENT=*sdd-comment*>, SDD_VERSION=*sdd-version-type)* ;

local-path	is the absolute path and name of the file on the local computer.
sdd-path	is the path (starting at the root) and name of the content object in the SAS Drug Development repository to update. The object must already exist and may not be a container.
sdd-comment	is an optional text string to be inserted as the comment for a versioned file in the repository. This value is ignored for non-versioned objects.
sdd-version-type	is an optional text string to designate the type of version to be created if the object to be updated has versioning enabled. Valid values are MAJOR or MINOR. The default is MINOR. This value is ignored for non-versioned objects.

SASDRUGDEV_UPDATEOWNER

Description

Updates the current owner of an object within the organization in the SAS Drug Development repository.

Syntax

For files:

%SASDRUGDEV_UPDATEOWNER(SDD_PATH =*sdd-path,* SDD_USERID=*user-ID* <, SDD_ACLTYPE =*sdd-acl-type* >);

For container objects:

%SASDRUGDEV_UPDATEOWNER(SDD_PATH =*sdd-path,* SDD_USERID=*user-ID* , SDD_ACLTYPE =*sdd-acl-type*);

sdd-path	is the path (starting at the root) and name of the content object in the SAS Drug Development repository. This can be either a file or a container.
sdd-acl-type	is the type of permissions being updated. The valid values are case-insensitive DEFAULT and CURRENT. Files only have CURRENT permissions; therefore the value is not required when *sdd-path* is a file. For container objects the value must be specified. The operation will fail if DEFAULT or an invalid value is provided for a file path.
user-id	is the SAS Drug Development ID for the user account that will be the new owner. In order to set the owner for a container's DEFAULT permissions to be the creator of the objects, specify the value <*creator*> as the value for user-id.