

# Oracle® Fusion Middleware

## WebLogic Scripting Tool Command Reference for Identity and Access Management



14.1.2.1.0 Release  
G24182-01  
March 2025

ORACLE®

Oracle Fusion Middleware WebLogic Scripting Tool Command Reference for Identity and Access Management,  
14.1.2.1.0 Release

G24182-01

Copyright © 2011, 2025, Oracle and/or its affiliates.

Primary Author: Panendra Puttachar

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

# Contents

## Preface

---

Audience	xvi
Documentation Accessibility	xvi
Related Documents	xvi
Conventions	xvii

## 1 Introduction

---

1.1 Guide to This Document	1-1
1.2 Document Scope and Audience	1-1
1.3 Using WSLT Commands	1-2
1.4 Related Documentation	1-2

## Part I WLST for Oracle Identity and Access Management

---

## 2 Access Manager WLST Commands

---

2.1 Access Manager Commands	2-1
2.1.1 displayAuthZCallBackKey	2-4
2.1.2 updateCustomPages	2-4
2.1.3 createUserIdentityStore	2-5
2.1.4 editUserIdentityStore	2-7
2.1.5 displayUserIdentityStore	2-9
2.1.6 createOAMServer	2-10
2.1.7 editOAMServer	2-11
2.1.8 deleteOAMServer	2-12
2.1.9 deleteUserIdentityStore	2-12
2.1.10 displayOAMServer	2-13
2.1.11 configurePersistentLogin	2-13
2.1.12 configOAMLoginPagePref	2-14
2.1.13 configRequestCacheType	2-16
2.1.14 displayRequestCacheType	2-16
2.1.15 editOssoAgent	2-17

2.1.16	deleteOssoAgent	2-18
2.1.17	displayOssoAgent	2-19
2.1.18	editWebgateAgent	2-19
2.1.19	deleteWebgateAgent	2-21
2.1.20	displayWebgateAgent	2-21
2.1.21	exportPolicy	2-22
2.1.22	importPolicy	2-22
2.1.23	importPolicyDelta	2-23
2.1.24	migratePartnersToProd	2-23
2.1.25	exportPartners	2-24
2.1.26	importPartners	2-24
2.1.27	displayTopology	2-25
2.1.28	configureOAAMPartner	2-25
2.1.29	registerOIFDAPPartner	2-26
2.1.30	registerOIFDAPPartnerIDPMode	2-26
2.1.31	registerThirdPartyTAPPartner	2-27
2.1.32	disableCoexistMode	2-28
2.1.33	enableOamAgentCoexist	2-28
2.1.34	disableOamAgentCoexist	2-28
2.1.35	editGITOValues	2-29
2.1.36	editWebgate11gAgent	2-29
2.1.37	deleteWebgate11gAgent	2-31
2.1.38	displayWebgate11gAgent	2-32
2.1.39	displayOAMMetrics	2-32
2.1.40	updateOIMHostPort (deprecated)	2-33
2.1.41	configureOIM (deprecated)	2-33
2.1.42	updateOSSOResponseCookieConfig	2-34
2.1.43	deleteOSSOResponseCookieConfig	2-35
2.1.44	configureAndCreateIdentityStore	2-35
2.1.45	configAndCreateIdStoreUsingPropFile	2-37
2.1.46	manageAuditEvents	2-37
2.1.47	migrateArtifacts (deprecated)	2-38
2.1.48	displaySimpleModeGlobalPassphrase	2-38
2.1.49	exportSelectedPartners	2-39
2.1.50	oamMigrate	2-39
2.1.51	preSchemeUpgrade	2-40
2.1.52	postSchemeUpgrade	2-40
2.1.53	oamSetWhiteListMode	2-41
2.1.54	oamWhiteListURLConfig	2-41
2.1.55	enableMultiDataCentreMode	2-42
2.1.56	disableMultiDataCentreMode	2-43
2.1.57	setMultiDataCentreClusterName	2-43

2.1.58	setMultiDataCentreLogoutURLs	2-43
2.1.59	updateMultiDataCentreLogoutURLs	2-44
2.1.60	addPartnerForMultiDataCentre	2-44
2.1.61	removePartnerForMultiDataCentre	2-45
2.1.62	addOAMSSOPProvider	2-45
2.1.63	setDiscoveryProvider	2-46
2.1.64	displayDiscoveryProvider	2-46
2.1.65	configurePluginMetadata	2-47

## 3 Identity Federation WLST Commands

---

3.1	Identity Federation Commands	3-1
3.1.1	addWSFed11IdPFederationPartner	3-5
3.1.2	addWSFed11SPFederationPartner	3-5
3.1.3	addOpenID20IdPFederationPartner	3-6
3.1.4	addOpenID20SPFederationPartner	3-6
3.1.5	addOpenID20GoogleIdPFederationPartner	3-7
3.1.6	addOpenID20YahooIdPFederationPartner	3-7
3.1.7	addSAML11IdPFederationPartner	3-8
3.1.8	addSAML11SPFederationPartner	3-8
3.1.9	addSAML20IdPFederationPartner	3-9
3.1.10	addSAML20SPFederationPartner	3-9
3.1.11	addSAML20IdPFederationPartnerWithoutMetadata	3-10
3.1.12	addSAML20SPFederationPartnerWithoutMetadata	3-11
3.1.13	configureIdPPartnerAttributeProfile	3-11
3.1.14	configureSAML20Logout	3-12
3.1.15	configureSAMLBinding	3-12
3.1.16	configureUserSelfRegistration	3-13
3.1.17	configureUserSelfRegistrationAttr	3-14
3.1.18	createAuthnSchemeAndModule	3-14
3.1.19	createIdPPartnerAttributeProfile	3-15
3.1.20	createSPPartnerAttributeProfile	3-15
3.1.21	deleteAuthnSchemeAndModule	3-16
3.1.22	deleteFederationPartner	3-16
3.1.23	deleteFederationPartnerEncryptionCert	3-17
3.1.24	deleteFederationPartnerSigningCert	3-17
3.1.25	deleteIdPPartnerAttributeProfile	3-18
3.1.26	deleteSPPartnerAttributeProfile	3-18
3.1.27	deleteIdPPartnerAttributeProfileEntry	3-18
3.1.28	deleteSPPartnerAttributeProfileEntry	3-19
3.1.29	deletePartnerProperty	3-19
3.1.30	displayIdPPartnerAttributeProfile	3-20

3.1.31	displaySPPartnerAttributeProfile	3-20
3.1.32	getAllFederationIdentityProviders	3-21
3.1.33	getAllFederationServiceProviders	3-21
3.1.34	getFederationPartnerEncryptionCert	3-21
3.1.35	getFederationPartnerSigningCert	3-22
3.1.36	getIdPPartnerBasicAuthCredentialUsername	3-22
3.1.37	getPartnerProperty	3-23
3.1.38	getStringProperty	3-23
3.1.39	isFederationPartnerPresent	3-24
3.1.40	listIdPPartnerAttributeProfileIDs	3-25
3.1.41	listSPPartnerAttributeProfileIDs	3-25
3.1.42	putStringProperty	3-25
3.1.43	setDefaultSSOIdPPartner	3-26
3.1.44	setFederationPartnerEncryptionCert	3-26
3.1.45	setFederationPartnerSigningCert	3-27
3.1.46	setIdPPartnerAttributeProfile	3-27
3.1.47	setIdPPDefaultScheme	3-28
3.1.48	setSPPartnerAttributeProfile	3-28
3.1.49	setIdPPartnerAttributeProfileEntry	3-29
3.1.50	setSPPartnerAttributeProfileEntry	3-29
3.1.51	setSPPartnerAttributeValueMapping	3-30
3.1.52	deleteSPPartnerAttributeValueMapping	3-31
3.1.53	displaySPPartnerAttributeValueMapping	3-32
3.1.54	setIdPPartnerAttributeValueMapping	3-32
3.1.55	deleteIdPPartnerAttributeValueMapping	3-34
3.1.56	displayIdPPartnerAttributeValueMapping	3-34
3.1.57	setSPPartnerAttributeValueFilter	3-35
3.1.58	deleteSPPartnerAttributeValueFilter	3-36
3.1.59	displaySPPartnerAttributeValueFilter	3-37
3.1.60	setIdPPartnerBasicAuthCredential	3-38
3.1.61	setIdPPartnerMappingAttribute	3-38
3.1.62	setIdPPartnerMappingAttributeQuery	3-39
3.1.63	setIdPPartnerMappingNameID	3-39
3.1.64	setPartnerAlias	3-40
3.1.65	setPartnerIDStoreAndBaseDN	3-40
3.1.66	setSPSAMLPartnerNameID	3-41
3.1.67	setSPPartnerAlternateScheme	3-42
3.1.68	setSPPartnerDefaultScheme	3-43
3.1.69	setSPPartnerProfileAlternateScheme	3-43
3.1.70	setSPPartnerProfileDefaultScheme	3-44
3.1.71	updatePartnerMetadata	3-45
3.1.72	updatePartnerProperty	3-45

3.2	Advanced Identity Federation Commands	3-46
3.2.1	configureFederationService	3-49
3.2.2	setFederationStore	3-50
3.2.3	configureIdPAuthnRequest	3-50
3.2.4	configureFedSSOAuthz	3-51
3.2.5	configureFedDigitalSignature	3-51
3.2.6	configureFedSignEncKey	3-52
3.2.7	configureAttributeSharingSPPartnerNameIDMapping	3-53
3.2.8	configureAttributeSharingIdPPartner	3-55
3.2.9	configureAttributeSharingUserDNToIdPPartnerMapping	3-56
3.2.10	configureAttributeSharing	3-56
3.2.11	removeAttributeSharingFromAuthnModule	3-57
3.2.12	configureAttributeSharingPlugin	3-57
3.2.13	insertAttributeSharingInToAuthnModule	3-58
3.2.14	addSPPartnerAuthnMethod	3-59
3.2.15	addSPPartnerProfileAuthnMethod	3-60
3.2.16	addIdPPartnerAuthnMethod	3-61
3.2.17	addIdPPartnerProfileAuthnMethod	3-61
3.2.18	listPartnerAuthnMethods	3-62
3.2.19	listPartnerProfileAuthnMethods	3-62
3.2.20	removePartnerAuthnMethod	3-63
3.2.21	removePartnerProfileAuthnMethod	3-63
3.2.22	setIdPPartnerRequestAuthnMethod	3-64
3.2.23	setIdPPartnerProfileRequestAuthnMethod	3-64
3.2.24	useProxiedFedAuthnMethod	3-65
3.2.25	createFedPartnerProfileFrom	3-66
3.2.26	deleteFedPartnerProfile	3-66
3.2.27	displayFedPartnerProfile	3-67
3.2.28	listFedPartnerProfiles	3-67
3.2.29	listFedPartnersForProfile	3-67
3.2.30	getFedPartnerProfile	3-68
3.2.31	setFedPartnerProfile	3-68
3.2.32	idpinitiatedssoprovideridparam	3-69
3.2.33	idpinitiatedssotargetparam	3-69
3.2.34	subjectconfirmationcheck	3-70

## 4 Identity Directory Service WLST Commands

---

4.1	Identity Directory Service WLST Commands	4-1
4.1.1	activateIDConfigChanges	4-2
4.1.2	addAttributeInEntityConfig	4-3
4.1.3	addAttributePropsInEntityConfig	4-3

4.1.4	addAttributeRefForEntity	4-4
4.1.5	addAttrrefPropsInEntityConfig	4-5
4.1.6	addCommonPropertyForOperationConfig	4-6
4.1.7	addEntity	4-6
4.1.8	addEntityProps	4-7
4.1.9	addEntityRelation	4-8
4.1.10	addIdentityDirectoryService	4-8
4.1.11	addOperationConfig	4-9
4.1.12	addPropertyForOperationConfig	4-10
4.1.13	deleteAttributeInEntityConfig	4-10
4.1.14	deleteAttributePropsInEntityConfig	4-11
4.1.15	deleteAttrrefPropsInEntityConfig	4-11
4.1.16	deleteEntity	4-12
4.1.17	deleteEntityProps	4-12
4.1.18	deleteEntityRelation	4-12
4.1.19	deleteIdentityDirectoryService	4-13
4.1.20	deleteOperationConfig	4-13
4.1.21	listAllAttributeInEntityConfig	4-14
4.1.22	listAllEntityInEntityConfig	4-14
4.1.23	listAllIdentityDirectoryService	4-15
4.1.24	removeAttributeRefForEntity	4-15
4.1.25	removeCommonPropertyForOperationConfig	4-15
4.1.26	removePropertyForOperationConfig	4-16
4.1.27	updateAttributeInEntityConfig	4-16
4.1.28	updateAttributePropsInEntityConfig	4-17
4.1.29	updateAttrrefPropsInEntityConfig	4-17
4.1.30	updateEntity	4-18
4.1.31	updateEntityAttrs	4-19
4.1.32	updateEntityProps	4-19
4.1.33	deleteAttributePropsInEntityConfig	4-20
4.1.34	dumpConnectionPoolStatsForInMemoryConfig	4-20
4.1.35	dumpConnectionPoolStatsForAllInMemoryConfig	4-21
4.1.36	dumpConnectionPoolStatsForAllFileBasedConfig	4-21
4.1.37	dumpConnectionPoolStatsForAllFileBasedConfig	4-22

## 5 Library Oracle Virtual Directory WLST commands

---

5.1	Library Oracle Virtual Directory WLST Commands	5-1
5.1.1	addDNAttribute	5-3
5.1.2	activateLibOVDConfigChanges	5-4
5.1.3	addAttributeExclusionRule	5-4
5.1.4	addAttributeRule	5-5



5.1.5	addDomainExclusionRule	5-5
5.1.6	addDomainRule	5-6
5.1.7	addJoinRule	5-6
5.1.8	addLDAPHost	5-7
5.1.9	addMappingContext	5-8
5.1.10	addPlugin	5-8
5.1.11	addPluginParam	5-9
5.1.12	addToRequestControlExcludeList	5-10
5.1.13	addToRequestControlIncludeList	5-10
5.1.14	assignViewToAdapter	5-11
5.1.15	createJoinAdapter	5-11
5.1.16	createLDAPAdapter	5-12
5.1.17	createLDAPAdapterWithDefaultPlugins	5-13
5.1.18	createView	5-14
5.1.19	deleteAdapter	5-14
5.1.20	deleteAttributeExclusionRule	5-15
5.1.21	deleteAttributeRule	5-15
5.1.22	deleteDomainExclusionRule	5-16
5.1.23	deleteDomainRule	5-17
5.1.24	deleteMappingContext	5-17
5.1.25	deleteView	5-18
5.1.26	getAdapterDetails	5-18
5.1.27	listAdapters	5-19
5.1.28	listAllMappingContextIds	5-19
5.1.29	listAttributeRules	5-19
5.1.30	listDomainRules	5-20
5.1.31	listViews	5-20
5.1.32	modifyLDAPAdapter	5-21
5.1.33	modifySocketOptions	5-24
5.1.34	removeAllRequestControlExcludeList	5-24
5.1.35	removeAllRequestControlIncludeList	5-25
5.1.36	removeDNAttribute	5-25
5.1.37	removeFromRequestControlExcludeList	5-26
5.1.38	removeFromRequestControlIncludeList	5-26
5.1.39	removeJoinRule	5-27
5.1.40	removeLDAPHost	5-27
5.1.41	removePlugin	5-28
5.1.42	removePluginParam	5-28
5.1.43	replacePluginParam	5-29
5.1.44	unassignViewFromAdapter	5-30
5.1.45	listSSLStoreType	5-30
5.1.46	enableKSSForSSL	5-31

5.1.47	enableJKSForSSL	5-31
5.1.48	createKeyStoreAndEnableJKSForSSL	5-32
5.1.49	importTrustedCertificateIntoSSLStore	5-32
5.1.50	migrateAllTrustedCertificatesFromJKSToKSS	5-33
5.1.51	migrateTrustedCertificatesFromJKSToKSS	5-33
5.1.52	changeLDAPHostPort	5-34
5.1.53	removeLDAPHostPort	5-34
5.1.54	setReadOnlyForLDAPHost	5-35
5.1.55	dumpLdapConnectionPoolStats	5-36
5.1.56	addCipherSuite	5-36
5.1.57	removeCipherSuite	5-37

## Part II WLST for Oracle Mobile Security Suite

---

### 6 Mobile Security Access Server WLST Commands

---

6.1	Using the WLST Commands	6-1
6.2	MSAS Configuration Commands	6-1
6.2.1	displayMSASConfiguration	6-2
6.2.2	setMSASConfiguration	6-2
6.2.3	setMSASLogLevel	6-3
6.2.4	getMSASLogLevel	6-4
6.2.5	listMSASLoggers	6-5
6.3	MSAS Identity Store Profile Commands	6-5
6.3.1	createIdentityProfile	6-6
6.3.2	displayIdentityProfile	6-7
6.3.3	selectIdentityProfile	6-7
6.3.4	deleteIdentityProfile	6-8
6.3.5	setIdentityProfileDirectory	6-9
6.3.6	setIdentityProfileUser	6-10
6.3.7	setIdentityProfileGroup	6-10
6.4	Repository Commands	6-11
6.4.1	exportMSASAppMetadata	6-11
6.4.2	importMSASAppMetadata	6-13
6.4.3	migrateMSASAppHostports	6-14
6.4.4	resetWSMPolicyRepository	6-15
6.5	Session Commands	6-16
6.5.1	abortRepositorySession	6-16
6.5.2	beginRepositorySession	6-17
6.5.3	commitRepositorySession	6-17
6.5.4	describeRepositorySession	6-17

6.6	Token Issuer Trust Configuration Commands	6-18
6.6.1	createWSMTokenIssuerTrustDocument	6-19
6.6.2	deleteWSMTokenIssuerTrust	6-19
6.6.3	deleteWSMTokenIssuerTrustAttributeRule	6-20
6.6.4	deleteWSMTokenIssuerTrustDocument	6-21
6.6.5	displayWSMTokenIssuerTrust	6-21
6.6.6	exportWSMTokenIssuerTrustMetadata	6-22
6.6.7	importWSMTokenIssuerTrustMetadata	6-23
6.6.8	listWSMTokenIssuerTrustDocuments	6-23
6.6.9	revokeWSMTokenIssuerTrust	6-24
6.6.10	selectWSMTokenIssuerTrustDocument	6-24
6.6.11	setWSMTokenIssuerTrust	6-25
6.6.12	setWSMTokenIssuerTrustAttributeFilter	6-26
6.6.13	setWSMTokenIssuerTrustAttributeMapping	6-27
6.6.14	setWSMTokenIssuerTrustDisplayName	6-28
6.7	Diagnostic Commands	6-29
6.7.1	checkWSMStatus	6-29

## List of Examples

---

2-1 Examples

2-38

## List of Tables

---

2-1	WLST Access Manager Commands	2-1
2-2	Language Codes For Login Pages	2-15
3-1	WLST Commands for Identity Federation	3-1
3-2	Advanced Identity Federation WLST Commands	3-47
4-1	WLST Identity Directory Service Commands	4-1
4-2	addAttributeInEntityConfig Arguments	4-3
4-3	addAttributePropsInEntityConfig Arguments	4-4
4-4	addAttributeRefForEntity Arguments	4-4
4-5	addAttrrefPropsInEntityConfig Arguments	4-5
4-6	addCommonPropertyForOperationConfig Arguments	4-6
4-7	addEntity Arguments	4-7
4-8	addEntityProps Arguments	4-8
4-9	addEntityRelation Arguments	4-8
4-10	addIdentityDirectoryService Arguments	4-9
4-11	addOperationConfig Arguments	4-9
4-12	addPropertyForOperationConfig Arguments	4-10
4-13	deleteAttributeInEntityConfig Arguments	4-10
4-14	deleteAttributePropsInEntityConfig Arguments	4-11
4-15	deleteAttrrefPropsInEntityConfig Arguments	4-11
4-16	deleteEntity Arguments	4-12
4-17	deleteEntityProps Arguments	4-12
4-18	deleteEntityRelation Arguments	4-13
4-19	deleteOperationConfig Arguments	4-14
4-20	removeAttributeRefForEntity Arguments	4-15
4-21	removeCommonPropertyForOperationConfig Arguments	4-16
4-22	removePropertyForOperationConfig Arguments	4-16
4-23	updateAttributeInEntityConfig Arguments	4-17
4-24	updateAttributePropsInEntityConfig Arguments	4-17
4-25	updateAttrrefPropsInEntityConfig Arguments	4-18
4-26	updateEntity Arguments	4-18
4-27	updateEntityAttrs Arguments	4-19
4-28	updateEntityProps Arguments	4-20
4-29	deleteAttributePropsInEntityConfig	4-20
4-30	dumpConnectionPoolStatsForInMemoryConfig	4-21
4-31	dumpConnectionPoolStatsForAllInMemoryConfig	4-21
4-32	dumpConnectionPoolStatsForAllFileBasedConfig	4-22

4-33	dumpConnectionPoolStatsForAllFileBasedConfig	4-22
5-1	WLST libOVD Commands	5-1
5-2	addDNAttribute Arguments	5-3
5-3	activateLibOVDConfigChanges Arguments	5-4
5-4	addAttributeExclusionRule Arguments	5-4
5-5	addAttributeRule Arguments	5-5
5-6	addDomainExclusionRule Arguments	5-6
5-7	addDomainRule Arguments	5-6
5-8	addJoinRule Arguments	5-7
5-9	addLDAPHost Arguments	5-7
5-10	addMappingContext Arguments	5-8
5-11	addPlugin Arguments	5-8
5-12	addPluginParam Arguments	5-9
5-13	addToRequestControlExcludeList Arguments	5-10
5-14	addToRequestControlIncludeList Arguments	5-10
5-15	assignViewToAdapter Arguments	5-11
5-16	createJoinAdapter Arguments	5-12
5-17	createLDAPAdapter Arguments	5-12
5-18	createLDAPAdapterWithDefaultPlugins Arguments	5-13
5-19	createView Arguments	5-14
5-20	deleteAdapter Arguments	5-15
5-21	deleteAttributeExclusionRule Arguments	5-15
5-22	deleteEntityRelation Arguments	5-16
5-23	deleteEntityRelation Arguments	5-16
5-24	deleteDomainRule Arguments	5-17
5-25	deleteMappingContext Arguments	5-17
5-26	createView Arguments	5-18
5-27	getAdapterDetails Arguments	5-18
5-28	listAdapters Arguments	5-19
5-29	listAllMappingContextIds Arguments	5-19
5-30	listAttributeRules Arguments	5-20
5-31	listDomainRules Arguments	5-20
5-32	listViews Arguments	5-21
5-33	modifyLDAPAdapter Arguments	5-22
5-34	modifySocketOptions Arguments	5-24
5-35	removeAllRequestControlExcludeList Arguments	5-24
5-36	removeAllRequestControlIncludeList Arguments	5-25

5-37	removeDNAttribute Arguments	5-25
5-38	removeFromRequestControlExcludeList Arguments	5-26
5-39	removeFromRequestControlIncludeList Arguments	5-26
5-40	removeJoinRule Arguments	5-27
5-41	removeLDAPHost Arguments	5-28
5-42	removePlugin Arguments	5-28
5-43	removePluginParam Arguments	5-29
5-44	replacePluginParam Arguments	5-29
5-45	unassignViewFromAdapter Arguments	5-30
5-46	listSSLStoreType Arguments	5-30
5-47	enableKSSForSSL Arguments	5-31
5-48	enableJKSForSSL Arguments	5-31
5-49	createKeyStoreAndEnableJKSForSSL Arguments	5-32
5-50	importTrustedCertificateIntoSSLStore Arguments	5-32
5-51	migrateAllTrustedCertificatesFromJKSToKSS Arguments	5-33
5-52	migrateTrustedCertificatesFromJKSToKSS Arguments	5-33
5-53	changeLDAPHostPort Arguments	5-34
5-54	removeLDAPHostPort Arguments	5-35
5-55	setReadOnlyForLDAPHost Arguments	5-35
5-56	dumpLdapConnectionPoolStats Arguments	5-36
5-57	addCipherSuite Arguments	5-36
5-58	removeCipherSuite Arguments	5-37
6-1	MSAS Configuration Commands	6-1
6-2	Identity Profile Management Commands	6-5
6-3	Policy Repository Management Commands	6-11
6-4	Session Management WLST Commands	6-16
6-5	Token Issuer Trust Commands	6-18

# Preface

This preface describes the document accessibility features and conventions used in this guide —*WebLogic Scripting Tool Command Reference for Identity and Access Management*.

## Audience

This document is intended for Administrators who are familiar with:

- Access and Identity Management concepts and administration
- Oracle WebLogic Server concepts and administration
- LDAP server concepts and administration
- Database concepts and administration (for policy and session management data)
- Web server concepts and administration
- WebGate and mod\_osso agents
- Auditing, logging, and monitoring concepts
- Security token concepts
- Integration of the Policy store, Identity store, and familiarity with OIS might be required

## Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

### Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

## Related Documents

You can see the following documents in the Oracle Fusion Middleware 14c (14.1.2.1.0) documentation set:

- *Oracle Access Management 14c (14.1.2.1.0) Release Notes*
- *Installing and Configuring Oracle Identity and Access Management*: Explains how to use the Oracle Universal Installer and the WebLogic Configuration Wizard.
- *Administrator's Guide for Oracle Access Management*: Explains how to manage configuration and policies for Access Manager, Security Token Service, Identity Federation, Mobile and Social, Identity Context and other Oracle Access Management suite services.



- *Developer's Guide for Oracle Access Management*: Explains how to write custom AccessGates and plug-ins that enable programmatic access to extend Access Manager single sign-on and authorization functions.
- *Upgrading Oracle Identity and Access Management*: Explains how to upgrade Oracle Identity and Access Management to 14c (14.1.2.1.0).
- *Tuning Performance*: Explains how to monitor and optimize performance, configure components for optimal performance, and write highly performance applications in the Oracle Fusion Middleware environment.
- *Administering Oracle Fusion Middleware*: Explains how to manage a secure Oracle Fusion Middleware environment, including how to change ports, deploy applications, and back up and recover Oracle Fusion Middleware. This guide also explains how to move data from a test to a production environment.
- *Enterprise Deployment Guide for Oracle Identity and Access Management*: Provides a step-by-step guide to deployment.
- *High Availability Guide*: Provides a high availability conceptual information and also administration and configuration procedures for Administrators, developers, and others whose role is to deploy and manage Oracle Fusion Middleware with high availability requirements.
- *Administering Web Services*: Explains how to administer and secure Web services.
- *WLST Command Reference for Infrastructure Security* : Describes all the commands that are available to use with the WLST for Oracle Platform Security Services and Infrastructure Security.

## Conventions

The following text conventions are used in this document:

Convention	Meaning
<b>boldface</b>	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

# 1

## Introduction

This section describes the contents and organization of this guide—*WebLogic Scripting Tool Command Reference for Identity and Access Management*.

- [Guide to This Document](#)
- [Document Scope and Audience](#)
- [Using WSLT Commands](#)
- [Related Documentation](#)

### 1.1 Guide to This Document

This document is organized as follows:

- This chapter introduces the guide and lists related documentation.
- [WLST for Oracle Identity and Access Management](#) summarizes WebLogic Scripting Tool (WLST) commands and variables for Identity and Access Management.
- [WLST for Oracle Mobile Security Suite](#) provides detailed descriptions for commands and variables for the Oracle Mobile Security Suite gateway component.

A refreshed version of this 11.1.2.3.0 documentation (published in September 2015) was reorganized. For example, commands previously collected in one long chapter in [WLST for Oracle Identity and Access Management](#) are now grouped in several shorter chapters.

### 1.2 Document Scope and Audience

This document describes the Identity and Access Management commands that are available for the WebLogic Scripting Tool (WLST). This document includes WLST commands for WebLogic Server, as well as custom WLST commands that can be used to manage installed Oracle Fusion Middleware Identity and Access Management components.



**Note:**

Custom WLST commands for a given Oracle Fusion Middleware component are available for use only if the component is installed in the `ORACLE_HOME` directory.

This document is written for WebLogic Server administrators and operators who deploy Java EE applications using the Java Platform, Enterprise Edition (Java EE) from Oracle. It is assumed that readers are familiar with Web technologies and the operating system and platform on which WebLogic Server is installed.

## 1.3 Using WSLT Commands

To use the custom WLST commands on WebLogic Server, you must invoke the WLST script from the Oracle Common home. To use the applicable Infrastructure Security custom WLST commands on a WebSphere Server, see the 3rd Party Integration Guide. For other WebLogic Server and Fusion Middleware WLST commands, refer to the WLST Command Reference guide for your installed release of WebLogic Server.

 **Note:**

For additional information about Oracle Platform Security Services, see Introduction to Oracle Platform Security Services in *Securing Applications with Oracle Platform Security Services*.

## 1.4 Related Documentation

WLST is one of several interfaces for managing and monitoring WebLogic Server. See *Understanding the WebLogic Scripting Tool* to understand WebLogic Scripting Tool. For other management interfaces, see:

- [Using Ant Tasks to Configure and Use a WebLogic Server Domain in \*Developing Applications for Oracle WebLogic Server\*](#), describes using WebLogic Ant tasks for starting and stopping WebLogic Server instances and configuring WebLogic domains.
- [Deployment Tools in \*Deploying Applications to Oracle WebLogic Server\*](#) describes several tools that WebLogic Server provides for deploying applications and stand-alone modules.
- [Administration Console Online Help](#) describes a Web-based graphical user interface for managing and monitoring WebLogic domains.
- [Creating WebLogic Domains Using the Configuration Wizard](#) describes using a graphical user interface to create a WebLogic domain or extend an existing one.
- [Creating Templates and Domains Using the Pack and Unpack Commands](#) describes commands that recreate existing WebLogic domains quickly and easily.
- [Developing Custom Management Utilities Using JMX for Oracle WebLogic Server](#) describes using Java Management Extensions (JMX) APIs to monitor and modify WebLogic Server resources.
- [Monitoring Oracle WebLogic Server with SNMP](#) describes using Simple Network Management Protocol (SNMP) to monitor WebLogic domains.
- [Administering Oracle Fusion Middleware](#) describes how to manage Oracle Fusion Middleware, including how to start and stop Oracle Fusion Middleware, how to configure and reconfigure components, and how to back up and recover.

# Part I

## WLST for Oracle Identity and Access Management

Part I describes the Identity and Access Management WLST commands.

- [Access Manager WLST Commands](#)
- [Identity Federation WLST Commands](#)
- [Identity Directory Service WLST Commands](#)
- [Library Oracle Virtual Directory WLST commands](#)

# 2

## Access Manager WLST Commands

Use these custom WebLogic Scripting Tool (WLST) commands to manage Oracle Access Management Access Manager related components such as authorization providers, identity asserters, and SSO providers. These commands allow you to display metrics and deployment topology, manage your server and agent configurations and logger settings.

The [Access Manager Commands](#) section lists the Oracle Access Management Access Manager WLST commands and their details.

### 2.1 Access Manager Commands

**Table 2-1 WLST Access Manager Commands**

Use this command...	To...	Use with WLST...
<a href="#">displayAuthZCallBackKey</a>	Generate and retrieve the key used to hash a resource URL in an authorization policy.	Online
<a href="#">updateCustomPages</a>	Enables and disables custom error and login pages.	Online Offline
<a href="#">createUserIdentityStore</a>	Create a user identity store registration.	Online Offline
<a href="#">editUserIdentityStore</a>	Edit a user identity store registration.	Online Offline
<a href="#">deleteUserIdentityStore</a>	Delete a user identity store registration.	Online Offline
<a href="#">displayUserIdentityStore</a>	Display a user identity store registration.	Online
<a href="#">createOAMServer</a>	Create an entry for an Access Manager Server configuration.	Online Offline
<a href="#">editOAMServer</a>	Edit the entry for an Access Manager Server configuration.	Online Offline
<a href="#">deleteOAMServer</a>	Delete the named Access Manager Server configuration.	Online Offline
<a href="#">displayOAMServer</a>	Display Access Manager Server configuration details.	Online Offline
<a href="#">configurePersistentLogin</a>	Enable or disable the Persistent Login feature.	Online
<a href="#">configOAMLoginPagePref</a>	Configure the Access Manager login page user preferences.	Online
<a href="#">configRequestCacheType</a>	Configure the SSO server request cache type.	Online
<a href="#">displayRequestCacheType</a>	Display the SSO server request cache type entry.	Online Offline
<a href="#">editOssoAgent</a>	Edit OSSO Agent configuration details.	Online Offline

Table 2-1 (Cont.) WLST Access Manager Commands

Use this command...	To...	Use with WLST...
<a href="#">deleteOssoAgent</a>	Delete the named OSSO Agent configuration.	Online Offline
<a href="#">displayOssoAgent</a>	Display OSSO Agent configuration details.	Online Offline
<a href="#">editWebgateAgent</a>	Edit 10g WebGate Agent registration details.	Online Offline
<a href="#">deleteWebgateAgent</a>	Delete the named 10g WebGate Agent configuration.	Online Offline
<a href="#">displayWebgateAgent</a>	Display WebGate Agent configuration details.	Online Offline
<a href="#">exportPolicy</a>	Export Access Manager policy data from a test (source) to an intermediate Access Manager file.	Online
<a href="#">importPolicy</a>	Import Access Manager policy data from the Access Manager file specified.	Online
<a href="#">importPolicyDelta</a>	Import Access Manager policy changes from the Access Manager file specified.	Online
<a href="#">migratePartnersToProd</a>	Migrate partners from the source Access Manager Server to the specified target Access Manager Server.	Online
<a href="#">exportPartners</a>	Export the Access Manager partners from the source to the intermediate Access Manager file specified.	Online
<a href="#">importPartners</a>	Import the Access Manager partners from the intermediate Access Manager file specified.	Online
<a href="#">displayTopology</a>	List the details of deployed Access Manager Servers.	Online Offline
<a href="#">configureOAAMPartner</a>	Configure the Access Manager-Oracle Adaptive Access Manager basic integration.	Online
<a href="#">registerOIFDAPPartner</a>	Register Identity Federation as Delegated Authentication Protocol (DAP) Partner.	Online Offline
<a href="#">registerOIFDAPPartnerIDPMode</a>	Registers Identity Federation in IDP mode.	
<a href="#">registerThirdPartyTAPPartner</a>	Registers any third party as a Trusted Authentication Protocol (TAP) Partner.	Online
<a href="#">disableCoexistMode</a>	Disable the Coexist Mode.	Online
<a href="#">enableOamAgentCoexist</a>	Enables Coexist Mode for the Access Manager agent (enabling the Access Manager 11g server to own the Obsocookie set by 10g WebGate).	Online
<a href="#">disableOamAgentCoexist</a>	Disables Coexist Mode for the Access Manager agent (disabling the Access Manager 11g server from the Obsocookie set by 10g WebGate).	Online
<a href="#">editGITOValues</a>	Edit GITO configuration parameters.	Online
<a href="#">editWebgate11gAgent</a>	Edit an 11g WebGate registration.	Online Offline

**Table 2-1 (Cont.) WLST Access Manager Commands**

Use this command...	To...	Use with WLST...
<a href="#">deleteWebgate11gAgent</a>	Remove an 11g WebGate Agent registration.	Online Offline
<a href="#">displayWebgate11gAgent</a>	Display an 11g WebGate Agent registration.	Online Offline
<a href="#">displayOAMMetrics</a>	Display metrics of Access Manager Servers.	Online Offline
<a href="#">updateOIMHostPort (deprecated)</a>	Update the Oracle Identity Manager configuration when integrated with Access Manager.	Online
<a href="#">configureOIM (deprecated)</a>	Creates an Agent registration specific to Oracle Identity Manager when integrated with Access Manager.	Online
<a href="#">updateOSSOResponseCookieConfig</a>	Updates OSSO Proxy response cookie settings.	Online
<a href="#">deleteOSSOResponseCookieConfig</a>	Deletes OSSO Proxy response cookie settings.	Online
<a href="#">configureAndCreateIdentityStore</a>	Configures an identity store and external user store.	Online
<a href="#">configAndCreateIdStoreUsingProfile</a>	Configures an identity store and external user store using values defined in a file.	Online
<a href="#">migrateArtifacts (deprecated)</a>	Migrates artifacts based on the specified artifact file.	Online
<a href="#">displaySimpleModeGlobalPassphrase</a>	Displays the simple mode global passphrase in plain text from the system configuration.	Online
<a href="#">exportSelectedPartners</a>	Exports selected Access Manager Partners to the intermediate Access Manager file specified.	Online
<a href="#">oamMigrate</a>	Migrates policies, authentication stores, and user stores from OSSO, OAM10g, OpenSSO, or AM 7.1 to OAM11g.	Online
<a href="#">preSchemeUpgrade</a>	Invokes the preSchemeUpgrade operation.	Online
<a href="#">postSchemeUpgrade</a>	Invokes the postSchemeUpgrade operation.	Online
<a href="#">oamSetWhiteListMode</a>	Set to true and the Access Manager Server will redirect to the URLs specified in the WhiteListURL list only.	Online
<a href="#">oamWhiteListURLConfig</a>	Add, update or remove whitelist URL entries from configuration file.	Online
<a href="#">enableMultiDataCentreMode</a>	Enable Multi Data Centre Mode.	Online
<a href="#">disableMultiDataCentreMode</a>	Disable Multi Data Centre Mode.	Online
<a href="#">setMultiDataCentreClusterName</a>	Set the Multi Data Centre Cluster name.	Online
<a href="#">setMultiDataCentreLogoutURLs</a>	Set the Multi Data Centre logout URLs.	Online
<a href="#">addPartnerForMultiDataCentre</a>	Add partner for Multi Data Centre.	Online
<a href="#">removePartnerForMultiDataCentre</a>	Remove partner from Multi Data Centre.	Online
<a href="#">addOAMSSOProvider</a>	Add an OAM SSO provider.	Online

**Table 2-1 (Cont.) WLST Access Manager Commands**

Use this command...	To...	Use with WLST...
<code>setDiscoveryProvider</code>	Set the fully qualified classname for the given discovery provider.	Online
<code>displayDiscoveryProvider</code>	Display the fully qualified classname configured for the discovery provider.	Online
<code>configurePluginMetadata</code>	Add the plugin and plugin-metadata as given in the propFile in the oam-config.xml.	Online

### 2.1.1 displayAuthZCallBackKey

The `displayAuthZCallBackKey` command is an online command that allows generation and retrieval of the key used to hash the resource URL that is returned during authorization when a success or failure URL is configured for the policy.

#### Description

Allows retrieval of the key used to hash the resource URL during authorization if already present. If the key is not present it is created and returned. The scope of this command is an instance only; the scope is not an argument.

#### Syntax

```
displayAuthZCallBackKey()
```



#### Note:

There are no arguments for this command.

#### Example

The following example displays the hash key.

```
displayAuthZCallBackKey()
```

### 2.1.2 updateCustomPages

The `updateCustomPages` command is an online and offline command that enables and disables custom error and login page configuration.

#### Description

Adds a context path and page extension to `oam-config.xml` that points to the WAR containing the custom Error and login pages:

```
<Setting Name="ssoengine" Type="htf:map">
<Setting Name="ErrorConfig" Type="htf:map">
<Setting Name="ErrorMode" Type="xsd:string">EXTERNAL</Setting>
<Setting Name="CustomPageExtension" Type="xsd:string">jsp</Setting>
<Setting Name="CustomPageContext" Type="xsd:string">/SampleApp</Setting>
</Setting>
</Setting>
```



## Syntax

```
updateCustomPages (pageExtension="<fileExtension>", context="<contextPath>")
```

Argument	Definition
<i>context</i>	Specifies the context path to the application; for example, /SampleApp.
<i>pageExtension</i>	Has a default value of "jsp" but can be left blank.

## Example

To enable the Custom Error page functionality, use `updateCustomPages` with the `context` and `pageExtension` parameters. This will modify the `oam-config.xml` file and enable the custom page functionality.

```
updateCustomPages (pageExtension = "jsp", context = "/SampleApp")
```

To disable the Custom Error page functionality, use the command without parameters [`updateCustomPages ()`]. This will undo the modifications made when the command is run with parameters.

## 2.1.3 createUserIdentityStore

The `createUserIdentityStore` command is an online and offline command that creates an identity store registration in the Access Manager system configuration.

### Description

Creates an entry in the system configuration for a new user identity store registered with Access Manager. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
createUserIdentityStore (name="<Name>", principal="<Principal>",
credential="<Credential>", type="<Type>", userAttr="<userAttr>",
ldapProvider="<ldapProvider>", userSearchBase="<userSearchBase>",
ldapUrl="<ldapUrl>", isPrimary="<isPrimary>", isSystem="<isSystem>",
userIDProvider="<userIDProvider>", roleSecAdmin="<roleSecAdmin>",
roleSysMonitor="<roleSysMonitor>", roleAppAdmin="<roleAppAdmin>",
roleSysManager="<roleSysManager>", roleSecAdminGroups="<roleSecAdminGroups>",
roleSecAdminUsers="<roleSecAdminUsers>", groupSearchBase="<groupSearchBase>",
supplementaryReturnAttributes="<supplementaryReturnAttributes>",
domainHome="<domainHome>")
```

Argument	Definition
<i>name</i>	Mandatory. Specifies the unique name of the LDAP identity store being created. Use only upper and lower case alpha characters and numbers.
<i>principal</i>	Mandatory. Specifies the Principal Administrator of the LDAP identity store being created. For example, cn=Admin.
<i>credential</i>	Mandatory. Specifies the password of the Principal for the LDAP identity store being created.

Argument	Definition
<i>type</i>	Mandatory. Specifies the type of the LDAP identity store being created. For this command, the value would be LDAP.
<i>userAttr</i>	Mandatory. Specifies the user attributes of the LDAP identity store being created.
<i>ldapProvider</i>	Mandatory. Specifies the type of the LDAP identity store being created. The value might be ODSEE, AD, OID, OVD, SJS, OUD, and the like. This value is defined when a new user identity store is created using the Access Manager Administration Console and corresponds with Store Type in the user identity store.
<i>userSearchBase</i>	Mandatory. Specifies the node under which user data is stored in the LDAP identity store being created. For example, <code>cn=users</code> .
<i>groupSearchBase</i>	Mandatory. Specifies the node under which group data is stored in the LDAP identity store being created. For example, <code>cn=groups</code> .
<i>ldapUrl</i>	Mandatory. Specifies the URL of the server host (including port number) of the LDAP identity store being created. For example, <code>ldap://localhost:7001</code> .
<i>isPrimary</i>	Optional. Specifies whether the LDAP identity store being created is the primary identity store. Takes true or false as a value.
<i>isSystem</i>	Optional. Specifies whether the LDAP identity store being created is the system store. Takes true or false as a value.
<i>userIDProvider</i>	Optional. Specifies the underlying infrastructure with which to connect to the identity store. Only supported type is OracleUserRoleAPI.
<i>roleSecAdminGroups</i>	Optional. Specifies one or more comma-delimited groups with Access Manager Console Administrator privileges. Needed if it is a System Store in which the <code>IsSystem</code> property is set to true.
<i>roleSecAdminUsers</i>	Optional. Specifies one or more comma-delimited users with Access Manager Console Administrator privileges. Needed if it is a System Store in which the <code>IsSystem</code> property is set to true.
<i>roleSecAdmin</i>	Optional. Specifies the Security Administrator of the LDAP identity store being created.
<i>roleSysMonitor</i>	Optional. Specifies the System Monitor of the LDAP identity store being created.
<i>roleAppAdmin</i>	Optional. Specifies the Application Administrator of the LDAP identity store being created.
<i>roleSysManager</i>	Optional. Specifies the System Manager of the LDAP identity store being created.
<i>supplementaryReturnAttributes</i>	Specifies a comma-delimited list of attributes that need to be retrieved as part of the User object. For example: ORCL_USR_ENC_FIRST_NAME,ORCL_USR_ENC_LAST_NAME,USR_USR_NAME,ORCL_USR_CTY_CODE,ORCL_USR_LANG_CODE_S,ORCL_USR_JROLE_ID_S,ORCL_USR_IND_ID,ORCL_USR_COMP_REL_ID,ORCL_USR_ASCII_IND,ORCL_ORA_UCM_VER,ORCL_ORA_UCM_SRVC
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere.

## Example

The following example registers a new Oracle Internet Directory user identity store definition for use with Access Manager.

```
createUserIdentityStore (name="Name1", principal="Principal1",
credential="Credential1", type="Type1", userAttr="userAttr1",
ldapProvider="ldapProvider", userSearchBase="userSearchBase", ldapUrl="ldapUrl",
isPrimary="isPrimary", isSystem="isSystem", userIDProvider="userIDProvider",
roleSecAdmin="<roleSecAdmin>", roleSysMonitor="<roleSysMonitor>",
roleAppAdmin="<roleAppAdmin>", roleSysManager="<roleSysManager>",
roleSecAdminGroups="<roleSecAdminGroups>",
roleSecAdminUsers="<roleSecAdminUsers>", groupSearchBase="groupSearchBase",
supplementaryReturnAttributes="supplementaryReturnAttributes",
domainHome="domainHome1")
```

## 2.1.4 editUserIdentityStore

The editUserIdentityStore command is an online and offline command that modifies an existing identity store registration for Access Manager.

### Description

Changes one or more attributes of the user identity store registered with Access Manager. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
editUserIdentityStore (name="<Name>", [ principal="<Principal>",
credential="<Credential>", type="<Type>", userAttr="<userAttr>",
ldapProvider="<ldapProvider>", roleSecAdmin="<roleSecAdmin>",
roleSysMonitor="<roleSysMonitor>", roleSysManager="<roleSysManager>" ,
roleAppAdmin="<roleAppAdmin>", roleSecAdminGroups="<roleSecAdminGroups>",
roleSecAdminUsers="<roleSecAdminUsers>", userSearchBase="<userSearchBase>",
ldapUrl="<ldapUrl>", isPrimary="<isPrimary>", isSystem="<isSystem>",
userIDProvider="<userIDProvider>" , groupSearchBase="<groupSearchBase>",
domainHome="<domainHome>", userFilterObjectClasses="<userFilterObjectClasses>",
groupFilterObjectClasses="<groupFilterObjectClasses>",
referralPolicy="<referralPolicy>", searchTimeLimit="<searchTimeLimit>",
minConnections="<minConnections>", maxConnections="<maxConnections>",
connectionWaitTimeout="<connectionWaitTimeout>",
connectionRetryCount="<connectionRetryCount>", groupNameAttr="<groupNameAttr>",
groupCacheEnabled="<groupCacheEnabled>", groupCacheSize="<groupCacheSize>",
groupCacheTTL="<groupCacheTTL>",
supplementaryReturnAttributes="<supplementaryReturnAttributes>" )
```

Argument	Definition
<i>name</i>	Mandatory. Specifies the unique name of the LDAP identity store being modified. Use only upper and lower case alpha characters and numbers.
<i>principal</i>	Specifies the Principal Administrator of the LDAP identity store being modified. For example, cn=Admin.
<i>credential</i>	Specifies the encrypted Password of the Principal Administrator for the LDAP identity store being modified.
<i>type</i>	Specifies the type of the base identity store being modified. For this command, the value would be LDAP.

Argument	Definition
<i>userAttr</i>	Mandatory. Specifies the user attributes of the LDAP identity store being modified.
<i>ldapProvider</i>	Mandatory. Specifies the LDAP type of the LDAP identity store being registered. The value might be ODSEE, AD, OID, OVD, SJS, OUD, and the like. This value is defined when a new user identity store is created using the Access Manager Administration Console and corresponds with Store Type in the user identity store.
<i>roleSecAdminGroups</i>	Optional. Specifies one or more comma-delimited groups with Access Manager Console Administrator privileges. Needed if it is a System Store in which the <code>IsSystem</code> property is set to true.
<i>roleSecAdminUsers</i>	Optional. Specifies one or more comma-delimited users with Access Manager Console Administrator privileges. Needed if it is a System Store in which the <code>IsSystem</code> property is set to true.
<i>roleSecAdmin</i>	Optional. Specifies the Security Administrator of the LDAP identity store being modified.
<i>roleSysMonitor</i>	Optional. Specifies the System Monitor of the LDAP identity store being modified.
<i>roleAppAdmin</i>	Optional. Specifies the Application Administrator of the LDAP identity store being modified.
<i>roleSysManager</i>	Optional. Specifies the System Manager of the LDAP identity store being modified.
<i>userSearchBase</i>	Mandatory. Specifies the node under which user data is stored in the LDAP identity store being modified. For example, <code>cn=users</code> .
<i>groupSearchBase</i>	Mandatory. Specifies the node under which user data is stored in the LDAP identity store being modified. For example, <code>cn=groups</code> .
<i>ldapUrl</i>	Mandatory. Specifies the URL of the server host (including port number) of the LDAP identity store being modified. For example, <code>ldap://localhost:7001</code> .
<i>isPrimary</i>	Optional. Specifies whether the LDAP identity store being modified is the primary identity store. Takes true or false as a value.
<i>isSystem</i>	Optional. Specifies whether the LDAP identity store being modified is the system store. Takes true or false as a value.
<i>userIDProvider</i>	Optional. Specifies the underlying infrastructure with which to connect to the identity store. Only supported type is <code>OracleUserRoleAPI</code> .
<i>supplementaryReturnAttributes</i>	Specifies a comma-delimited list of attributes that need to be retrieved as part of the User object. For example: ORCL_USR_ENC_FIRST_NAME,ORCL_USR_ENC_LAST_NAME,USR_USRNAME,ORCL_USR_CTY_CODE,ORCL_USR_LANG_CODE_S,ORCL_USR_JROLE_ID_S,ORCL_USR_IND_ID,ORCL_USR_COMP_REL_ID,ORCL_USR_ASCII_IND,ORCL_ORA_UCM_VER,ORCL_ORA_UCM_SRVC
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.
<i>userFilterObjectClasses</i>	Mandatory. Specifies a list of user filter object classes (separated by semicolon).

Argument	Definition
<i>groupFilterObjectClasses</i>	Specifies a list of group filter object classes (separated by semicolon).
<i>referralPolicy</i>	Specifies an LDAP referral policy (either "follow", "ignore" or "throw").
<i>searchTimeLimit</i>	Specifies the time limit in seconds for an LDAP Search operation.
<i>minConnections</i>	Specifies the minimum number of connections in the connection pool.
<i>maxConnections</i>	Specifies the maximum number of connections in the connection pool.
<i>connectionWaitTimeout</i>	Specifies the number of seconds to wait for obtaining a connection from the pool.
<i>connectionRetryCount</i>	Specifies the number of attempts to retry when establishing a connection to the identity store.
<i>groupNameAttr</i>	Specifies the name of the attribute to lookup the user groups. For example, <code>ou=people,ou=myrealm,dc=base_domain</code> .
<i>groupCacheEnabled</i>	A boolean that specifies whether to enable the LDAP group cache. Takes true or false as a value.
<i>groupCacheSize</i>	Specifies the number of entries in the LDAP group cache.
<i>groupCacheTTL</i>	Specifies the total time to live for each entry in the LDAP group cache.

### Example

The following example changes the search base values for the registered identity store.

```
editUserIdentityStore(name="IdStore1", userSearchBase="cn=users",
groupSearchBase="cn=groups")
```

## 2.1.5 displayUserIdentityStore

The `displayUserIdentityStore` command is an online command that displays user identity store registration information.

### Description

Displays the information regarding the identity store registered with Access Manager. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
displayUserIdentityStore(name="<name>", domainHome="<domainHome>")
```

Argument	Definition
<i>name</i>	Mandatory. Specifies the name of the LDAP identity store registration to be displayed.
<i>domainhome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere.

### Example

The following example invocation for WebSphere displays registration details of the user identity store. To use this command in online mode with WebLogic, there is no need to specify the domainHome argument.

```
displayUserIdentityStore(name="ID_Store1", domainHome="domainHome1")
```

## 2.1.6 createOAMServer

The createOAMServer command is an online and offline command that creates an Access Manager Server entry in the system configuration.

### Description

Creates an Access Manager Server registration. The details include the host, port, registration name, Access Manager Proxy port, server ID and, optionally, the OAM Proxy shared secret. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
createOAMServer(configurationProfile="<configurationProfile>",
host="<host>",port="<port>", oamProxyPort="<0000>",
oamProxyServerID="<oamProxyServerID>",siteName="<siteName>", domainHome="<domainHome>")
```

Argument	Definition
<i>configurationProfile</i>	Mandatory. Specifies the Configuration Profile of the OAM Server. The profile appears under Server Instances on the System Configuration tab in the Access Manager Administration Console.
<i>host</i>	Mandatory. Specifies the name of the Access Manager Server host.
<i>port</i>	Mandatory. Specifies the listening port of the Access Manager Server host.
<i>oamProxyPort</i>	Mandatory. Specifies the proxy port of the Access Manager Server host.
<i>oamProxyServerID</i>	Mandatory. Specifies the proxy server ID of the Access Manager Server host. The Access Manager Proxy name appears under the Access Manager Proxy sub tab of the server instance in the Access Manager Administration Console.
<i>siteName</i>	Mandatory. Specifies the siteName/serverName for the instance.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

## Example

The following example creates a configuration for `my_host` with listening port 15000. The configuration entry in the Access Manager Administration Console will be `oam_server1`. The Access Manager Proxy port is 3004 and the Access Manager Proxy Server ID is `oamProxyServerID1`.

```
createOAMServer(configurationProfile="oam_server1", host="my_host",
port="15000", oamProxyPort="3004", oamProxyServerID="oamProxyServerID1",
siteName="siteName1", domainHome="domainHome1")
```

## 2.1.7 editOAMServer

The `editOAMServer` command is an online and offline command that enables you to modify the details of an Access Manager Server registration.

### Description

Modifies the specified parameter values of the registration for an Access Manager Server. The details may include the host, port, registration name, Access Manager Proxy port, server ID and, optionally, the Access Manager Proxy shared secret. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
editOAMServer(configurationProfile="<configurationProfile>",
host="<host>",port="<port>", oamProxyPort="<0000>",
oamProxyServerID="<oamProxyServerID>",siteName="<siteName>", domainHome="<domainHome>")
```

Argument	Definition
<i>configurationProfile</i>	Mandatory. Specifies the Configuration Profile of the Access Manager Server. The profile appears under Server Instances on the System Configuration tab in the Access Manager Administration Console.
<i>host</i>	Mandatory. Specifies the name of the Access Manager Server host.
<i>port</i>	Mandatory. Specifies the listening port of the Access Manager Server host.
<i>oamProxyPort</i>	Mandatory. Specifies the proxy port of the Access Manager Server host.
<i>oamProxyServerID</i>	Mandatory. Specifies the proxy server ID of the Access Manager Server host. The Access Manager Proxy name appears under the Access Manager Proxy sub tab of the server instance in the Access Manager Administration Console.
<i>siteName</i>	Mandatory. Specifies the siteName/serverName for the instance.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

You can use any of the optional attributes to change current settings. The following invocation enables you to add the Access Manager Proxy Server ID to the configuration entry `oam_server1`.

```
editOAMServer(configurationProfile="oam_server1", host="my_host",
port="15000", oamProxyPort="3004", oamProxyServerID="oamProxyServerID1",
siteName="siteName1", domainHome="domainHome1")
```

## 2.1.8 deleteOAMServer

The deleteOAMServer command is an online and offline command that enables you to delete the specified Access Manager Server registration.

### Description

Deletes the specified Access Manager Server configuration. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
deleteOAMServer(host="<host>", port="<port>", domainHome="<domainHome>")
```

Argument	Definition
<i>host</i>	Mandatory. Specifies the name of the Access Manager Server host.
<i>port</i>	Mandatory. Specifies the listening port of the Access Manager Server host.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

The following example enables you to delete the `oam_server1` Access Manager Server registration with listening port 15000.

```
deleteOAMServer(host="oam_server1", port="15000", domainHome="domainHome1")
```

## 2.1.9 deleteUserIdentityStore

The deleteUserIdentityStore command is an online and offline command that deletes an existing identity store registration for Access Manager.

### Description

Deletes the identity store registration. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
deleteUserIdentityStore(name="<name>", domainHome="<domainHome>")
```

Argument	Definition
<i>name</i>	Mandatory. Specifies the name of the LDAP identity store registration to be removed.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.



**Example**

The following example can be used on WebSphere and deletes the registration of the named identity store. To use this command in online mode with WebLogic Server, the `domainHome` argument need not be specified.

```
deleteUserIdentityStore (name="identity_store", domainHome="domainHome1")
```

## 2.1.10 displayOAMServer

The `displayOAMServer` command is an online and offline command that displays registration details for the specified Access Manager Server.

**Description**

Displays the registration details of the specified Access Manager Server, including the host, port, registration name, Access Manager Proxy port, server ID and, optionally, the Access Manager Proxy shared secret. The scope of this command is an instance only; the scope is not an argument.

**Syntax**

```
displayOAMServer (host="<host>", port="<port>", domainHome="<domainHome>")
```

Argument	Definition
<i>host</i>	Mandatory. Specifies the name of the Access Manager Server host.
<i>port</i>	Mandatory. Specifies the listening port of the Access Manager Server host.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

**Example**

The following example will list all metrics specific to the `my_host` Access Manager Server.

```
displayOAMServer (host="my_host", port="15000", domainHome="domainHome1")
```

## 2.1.11 configurePersistentLogin

The `configurePersistentLogin` command is an online command used to enable or disable the Persistent Login feature.

**Description**

Enables the Persistent Login feature.

**Syntax**

```
configurePersistentLogin (enable="true/false",  
validityInDays="<#>", maxAuthnLevel="<#>", userAttribute="<userAttr>")
```

Argument	Definition
<i>enable</i>	Mandatory. Specify true or false.
<i>validityInDays</i>	Mandatory. Specifies the number of days that the user login will be persisted for a particular browser instance or device.
<i>maxAuthnLevel</i>	Mandatory. Specifies the maximum Authentication Level allowed after re-authenticating automatically through Persistent Login.
<i>userAttr</i>	Mandatory. Specifies the user attribute with which Persistent Login properties will be stored.

### Example

The following example changes the search base values for the registered identity store.

```
configurePersistentLogin(enable="true", validityInDays="30", maxAuthnLevel="2"
  userAttribute="obPSFTID")
```

## 2.1.12 configOAMLoginPagePref

The configOAMLoginPagePref command is an online command that configures the Access Manager login page user preferences.

### Description

Configures the Access Manager login page user preferences.

### Syntax

```
configOAMLoginPagePref(persistentCookie="true", persistentCookieLifetime=14,
  langPrefCookieDomain="oracle.com", langPrefOrder="serverOverrideLangPref,
  oamPrefsCookie, browserAcceptLanguage, defaultLanguage",
  serverOverrideLanguage="en", defaultLanguage="en",
  applicationSupportedLocales="en, fr")
```

Argument	Definition
<i>persistentCookie</i>	Mandatory. Boolean that defines whether the OAM_LANG_PREF cookie is persistent or non-persistent. Set to true or false.
<i>persistentCookieLifetime</i>	Mandatory. Lifetime of the OAM_LANG_PREF cookie if persistent.
<i>langPrefCookieDomain</i>	Mandatory. Defines the domain of the OAM_LANG_PREF cookie.
<i>langPrefOrder</i>	Mandatory. Decides the order of language precedence. Must be formatted as in the syntax and example. The allowed value set is (serverOverrideLangPref,oamPrefsCookie,browserAcceptLanguage,defaultLanguage). "oamPrefsCookie, browserAcceptLanguage, serverOverrideLangPref"
<i>serverOverrideLanguage</i>	The server side language of Access Manager. Must be defined in language codes and selected from OAM supported languages. Default value is en.

Argument	Definition
<i>defaultLanguage</i>	The default language.
<i>applicationSupportedLocales</i>	Supported languages defined in a comma-delimited list. Setting <code>applicationSupportedLocales="en,fr"</code> insures the OAM Login page will display a list of values containing French and English. The supported language codes are documented in <a href="#">Table 2-2</a> below.

**Table 2-2 Language Codes For Login Pages**

Language Code	Language	Administrators
ar	Arabic	
cs	Czech	
da	Danish	
de	German	German
el	Greek	
en	English	English
es	Spanish	Spanish
fi	Finnish	
fr	French	French
fr-CA	Canadian French	Canadian French
he	Hebrew	
hr	Croatian	
hu	Hungarian	
it	Italian	Italian
ja	Japanese	Japanese
ko	Korean	Korean
nl	Dutch	
no	Norwegian	
pl	Polish	
pt-BR	Brazilian Portuguese	Brazilian Portuguese
pt	Portuguese	
ro	Romanian	
ru	Russian	
sk	Slovak	
sv	Swedish	
th	Thai	
tr	Turkish	
zh-CN	Simplified Chinese	Simplified Chinese
zh-TW	Traditional Chinese	Traditional Chinese

### Example

```
configOAMLoginPagePref (persistentCookie="true", persistentCookieLifetime=14,
langPrefCookieDomain="oracle.com", langPrefOrder="serverOverrideLangPref,
oamPrefsCookie, browserAcceptLanguage, defaultLanguage",
serverOverrideLanguage="en", defaultLanguage="en",
applicationSupportedLocales="en, fr")
```

This next example allows an administrator to revert back to the default behavior in which no language list of values is displayed.

```
configOAMLoginPagePref (persistentCookie="true",
persistentCookieLifetime=14, langPrefCookieDomain="example.com",
langPrefOrder="serverOverrideLangPref, oamPrefsCookie, browserAcceptLanguage,
defaultLanguage", serverOverrideLanguage="",
defaultLanguage="en", applicationSupportedLocales="")
```

## 2.1.13 configRequestCacheType

The `configRequestCacheType` command is an online and offline command that defines the SSO server request cache type in the system configuration.

### Description

Defines the SSO server request cache type in the system configuration. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
configRequestCacheType (type="<requestCacheType>", domainHome="<domainHome>")
```

Argument	Definition
<i>type</i>	Mandatory. Specifies the request cache type. Takes a value of BASIC or COOKIE.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

The following example identifies the request cache type as Cookie:

```
configRequestCacheType (type="COOKIE")
```

## 2.1.14 displayRequestCacheType

The `displayRequestCacheType` command is an online and offline command that displays the SSO server request cache type defined for the specified domain. The request cache type may be BASIC or COOKIE.

### Description

Displays the SSO server request cache type entry defined for the specified domain. The scope of this command is an instance only; the scope is not an argument.

## Syntax

```
displayRequestCacheType (domainHome="<domainHome>")
```

Argument	Definition
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

## Example

The following example will display the request cache type (BASIC or COOKIE) defined for the specified domain home.

```
displayRequestCacheType (domainHome="domainHome1")
```

## 2.1.15 editOssoAgent

The editOssoAgent command is an online and offline command that enables you to modify the details of an OpenSSO (OSSO) Agent registration in the system configuration.

### Description

Modifies OSSO Agent registration details including the Site Token, Success URL, Failure URL, Home URL, Logout URL, Start Date, End Date, Administrator ID, and Administrator Info. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
editOssoAgent (agentName="AgentName", partnerId = "<partnerId>",
siteToken = "<siteToken>", siteName = "<siteName>", successUrl = "<successUrl>",
failureUrl = "<failureUrl>", homeUrl="<homeUrl>", logoutUrl="<logoutUrl>", startDate =
"<startDate>", endDate = "<endDate>", adminId = "<adminId>",
adminInfo = "<AdminInfo>", domainHome="<domainHomeName>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the OSSO Agent entry to be modified. adminId=admin Id of OSSO agent <optional> adminInfo=admin Information of OSSO agent <optional>
<i>partnerId</i>	Optional. Specifies the Agent Name of the OSSO agent instance.
<i>siteToken</i>	Optional. Specifies the Application Token used by the partner when requesting authentication.
<i>siteName</i>	Optional. Specifies the SiteName/ServerName for the OSSO agent instance.
<i>successUrl</i>	Optional. Specifies the redirect URL to be used by the OSSO Agent if authentication is successful.
<i>failureUrl</i>	Optional. Specifies the redirect URL to be used by the OSSO Agent if authentication fails.
<i>homeUrl</i>	Optional. Specifies the redirect URL to be used for the Home page after authentication.

Argument	Definition
<i>logoutUrl</i>	Optional. Specifies the redirect URL to be used when a user is logging out.
<i>startDate</i>	Optional. Specifies the first month, day, and year for which login to the application is allowed by the server.
<i>endDate</i>	Optional. Specifies the final month, day, and year for which login to the application is allowed by the server.
<i>adminId</i>	Optional. Specifies the administrator login ID for the OSSO Agent.
<i>adminInfo</i>	Optional. Specifies an administrator identifier for the OSSO Agent for tracking purpose.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

The following example changes the Administrator ID and information in the registration entry for `OSSOAgent1`.

```
editOssoAgent(agentName = "OSSOAgent1", partnerId = "partnerId",
siteToken = "siteToken", siteName = "siteName", successUrl="successUrl",
failureUrl = "failureUrl", homeUrl="homeUrl", logoutUrl="logoutUrl",
startDate = "2009-12-10", endDate = "2012-12-30", adminId = "345",
adminInfo = "Agent11", domainHome="domainHome1")
```

## 2.1.16 deleteOssoAgent

The `deleteOssoAgent` command is an online and offline command that enables you to remove the specified OSSO Agent registration in the system configuration.

### Description

Removes the specified OSSO Agent registration in the system configuration. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
deleteOssoAgent (agentName="<AgentName>", domainHome="<domainHomeName>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the OSSO Agent entry to be removed.
<i>domainhome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

The following example removes the OSSO Agent registration entry named `OSSOAgent1`.

```
deleteOssoAgent (agentName="OSSOAgent1", domainHome="domainHome1")
```

## 2.1.17 displayOssoAgent

The `displayOssoAgent` command is an online and offline command that displays the details of the specified OSSO Agent entry in the system configuration.

### Description

Displays the details of the specified OSSO Agent entry in the Access Manager Administration Console. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
displayOssoAgent (agentName="<AgentName>", domainHome="<domainHomeName>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the OSSO Agent entry to be displayed.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

The following example displays the `OSSOAgent1` entry details.

```
displayOssoAgent (agentName="OSSOAgent1", domainHome="domainHome1")
```

## 2.1.18 editWebgateAgent

The `editWebgateAgent` command is an online and offline command that enables you to modify a Webgate 10g registration entry in the system configuration.

### Description

Enables you to modify a Webgate 10g registration entry in the system configuration. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
editWebgateAgent (agentName="<AgentName>",  
accessClientPasswd="<accessClientPassword >", state="<state>", preferredHost="<host>",  
aaaTimeOutThreshold="<aaaTimeoutThreshold >",  
security="<security>", primaryCookieDomain="<primaryCookieDomain>",  
maxConnections="<maxConnections>", maxCacheElems="<maxCacheElements >",  
cacheTimeout="<cacheTimeout>",  
cookieSessionTime="<cookieSessionTime >", maxSessionTime="<maxSessionTime>",  
idleSessionTimeout="<idleSessionTimeout >", failoverThreshold="<failoverThreshold >",  
domainHome="<domainHomeName>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the WebGate Agent to be modified.
<i>accessClientPasswd</i>	Optional. Specifies the access client password of WebGate Agent.

Argument	Definition
<i>state</i>	Optional. Specifies whether the WebGate Agent is enabled or disabled with a value of either Enabled or Disabled, respectively.
<i>preferredHost</i>	Optional. Specifies the preferred host of the WebGate Agent. This prevents security holes that can be created if a host's identifier is not included in the Host Identifiers list. For virtual hosting, you must use the Host Identifiers feature.
<i>aaaTimeOutThreshold</i>	Optional. Specifies the number (in seconds) to wait for a response from the Access Manager run-time server. If this parameter is set, it is used as an application TCP/IP timeout instead of the default TCP/IP timeout. Default = -1 (default network TCP/IP timeout is used)
<i>security</i>	Optional. Specifies the level of transport security to and from the Access Manager run-time server. Takes as a value either open, simple, or cert.
<i>primaryCookieDomain</i>	Optional. Specifies the Web server domain on which the Access Manager Agent is deployed. For example, <i>.acompany.com</i>
<i>maxConnections</i>	Optional. Specifies the maximum number of connections that this Access Manager Agent can establish with the Access Manager Server. This number must be the same as (or greater than) the number of connections that are actually associated with this agent. Default = 1
<i>maxCacheElems</i>	Optional. Specifies the maximum number of elements maintained in the cache. Cache elements are URLs or Authentication Schemes. The value of this setting refers to the maximum consolidated count for elements in both of these caches. Default = 10000
<i>cacheTimeout</i>	Optional. Specifies the amount of time cached information remains in the Access Manager Agent cache when the information is neither used nor referenced. Default = 1800 (seconds)
<i>cookieSessionTime</i>	Optional. Specifies the amount of time that the ObSSOCookie persists. Default = 3600 (seconds)
<i>maxSessionTime</i>	Optional. Specifies the maximum amount of time in seconds that a user's authentication session is valid regardless of their activity. At the expiration of this time, the user is re-challenged for authentication. This is a forced logout. A value of 0 disables this timeout setting. Default = 3600 (seconds)
<i>idleSessionTimeout</i>	Specifies the location of the Domain Home. When Offline, a value is mandatory; when online, optional.
<i>failoverThreshold</i>	Optional. Specifies a number representing the point when this Access Manager Agent opens connections to a Secondary Access Manager Server. Default = 1
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

You can alter any or all of the settings. Use the following example to change the Agent ID, state, maximum connections, Access Manager Server timeout, primary cookie domain, cache time out, cookie session timeout, maximum session timeout, idle session timeout, and failover threshold.

```
editWebgateAgent(agentName="WebgateAgent1", accessClientPasswd="welcome1",
state="Enabled", preferredHost="141.144.168.148:2001", aaaTimeOutThreshold = "10",
security="open", primaryCookieDomain="primaryCookieDomain", maxConnections="16",
maxCacheElems="10000", cacheTimeout="1800", cookieSessionTime="3600",
```



```
maxSessionTime="24", idleSessionTimeout="3600", failoverThreshold="1",
domainHome="domainHome1")
```

## 2.1.19 deleteWebgateAgent

The deleteWebgateAgent command is an online and offline command that enables you to delete a Webgate\_agent registration entry in the system configuration.

### Description

Removes the specified Webgate\_agent registration entry from the system configuration. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
deleteWebgateAgent (agentName="<AgentName>", domainHome="<domainHomeName>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the WebGate Agent being deleted.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

The following example removes the WebGate Agent named WebgateAgent1.

```
deleteWebgateAgent (agentName="WebgateAgent1", domainHome="domainHome1")
```

## 2.1.20 displayWebgateAgent

The displayWebgateAgent command is an online and offline command that displays all details of the specified Webgate\_agent registration entry in the Access Manager Administration Console.

### Description

Displays all details of the specified Webgate\_agent registration entry in the Access Manager Administration Console. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
displayWebgateAgent (agentName="<AgentName>", domainHome="<domainHomeName>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the WebGate Agent being displayed.
<i>domainhome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

**Example**

The following example displays entry details for `WebgateAgent1`.

```
displayWebgateAgent(agentName="WebgateAgent1", domainHome="domainHome1")
```

## 2.1.21 exportPolicy

The `exportPolicy` command is an online command that exports Access Manager policy data from a test (source) environment to the intermediate Access Manager file specified.

**Description**

Exports Access Manager policy data from a test (source) environment to the intermediate Access Manager file. The scope of this command is an instance only; the scope is not an argument.

**Syntax**

```
exportPolicy(pathTempOAMPolicyFile="<absoluteFilePath>")
```

Argument	Definition
<i>pathTempOAMPolicyFile</i>	Mandatory. Specifies the absolute path to the temporary Access Manager file.

**Example**

The following example specifies the path to the `tempfile.txt` file used when exporting policy data from a test (source) environment.

```
exportPolicy(pathTempOAMPolicyFile="/expleroot/parent/tempfile.txt")
```

## 2.1.22 importPolicy

The `importPolicy` command is an online command that imports the Access Manager policy data from the specified Access Manager file.

**Description**

Imports the Access Manager policy data from the specified Access Manager file. The scope of this command is an instance only; the scope is not an argument.

**Syntax**

```
importPolicy(pathTempOAMPolicyFile="<absoluteFilePath>")
```

Argument	Definition
<i>pathTempOAMPolicyFile</i>	Mandatory. Specifies the absolute path to the temporary Access Manager file.

**Example**

The following example specifies the path to the `tempfile.txt` file used when importing policy data to a production (target) environment.

```
importPolicy (pathTempOAMPolicyFile="/expleroot/parent/tempfile.txt")
```

## 2.1.23 importPolicyDelta

The `importPolicyDelta` command is an online command that imports the Access Manager policy changes from the specified Access Manager file.

### Description

Imports the Access Manager policy changes from the specified Access Manager file. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
importPolicyDelta (pathTempOAMPolicyFile="<absoluteFilePath>")
```

Argument	Definition
<i>pathTempOAMPolicyFile</i>	Mandatory. Specifies the absolute path to the temporary Access Manager file.

### Example

The following example specifies the path to the `tempfile_delta.txt` file used when importing changed policy data to a production (target) environment.

```
importPolicyDelta (pathTempOAMPolicyFile="/expleroot/parent/tempfile_delta.txt")
```

## 2.1.24 migratePartnersToProd

The `migratePartnersToProd` command is an online command that migrates partners from the current (source) Access Manager Server to the specified (target) Access Manager Server.

### Description

Migrates partners from the current (source) Access Manager Server to the specified (target) Access Manager Server. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
migratePartnersToProd (prodServerHost="<host>", prodServerPort="<port>",  
prodServerAdminUser="<user>", prodServerAdminPwd="<passwd>")
```

Argument	Definition
<i>prodServerHost</i>	Host name of the target Access Manager Server to which partners are to be migrated.
<i>prodServerPort</i>	Port of the target Access Manager Server to which partners are to be migrated.
<i>prodServerAdminUser</i>	Administrator of the target Access Manager Server to which partners are to be migrated.
<i>prodServerAdminPwd</i>	Target Access Manager Server administrator's password.

### Example

The following example specifies the required information for partner migration.

```
migratePartnersToProd(prodServerHost="myhost", prodServerPort="1234",
prodServerAdminUser="weblogic", prodServerAdminPwd="welcome")
```

## 2.1.25 exportPartners

The `exportPartners` command is an online command that exports Access Manager partners from the source to the Access Manager file specified.

### Description

Exports the Access Manager partners from the source to the Access Manager file specified. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
exportPartners (pathTempOAMPartnerFile="<absoluteFilePath>")
```

Argument	Definition
<i>pathTempOAMPolicyFile</i>	Mandatory. Specifies the absolute path to the temporary Access Manager file.

### Example

The following example specifies the absolute path to the Access Manager partners file.

```
exportPartners (pathTempOAMPolicyFile="/explampleroot/parent/tempfile_partners.xml")
```

## 2.1.26 importPartners

The `importPartners` command is an online command that imports Access Manager partners from the specified Access Manager file.

### Description

Imports the Access Manager partners from the specified Access Manager file. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
importPartners (pathTempOAMPartnerFile="<absoluteFilePath>")
```

Argument	Definition
<i>pathTempOAMPartnerFile</i>	Mandatory. Specifies the path to the temporary Access Manager partner file.

### Example

The following example specifies the absolute path to the Access Manager file from which the partners will be imported.

```
importPartners (pathTempOAMPolicyFile="/explampleroot/parent/tempfile_partners.xml")
```

## 2.1.27 displayTopology

The `displayTopology` command is an online and offline command that displays information about all Access Manager Servers in a deployment.

### Description

Lists the topology of deployed Access Manager Servers.

### Syntax

```
displayTopology (domainHome="<domainHomeName>")
```

Argument	Definition
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

The following example lists the details of all deployed Access Manager Servers in the specified domain home.

```
displayTopology (domainHome="domainHome1")
```

## 2.1.28 configureOAAMPartner

The `configureOAAMPartner` command is an online command that configures the basic integration of Access Manager and Oracle Adaptive Access Manager (OAAM).

### Description

Configures the basic integration of Access Manager and OAAM. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
configureOAAMPartner (dataSourceName="<dataSourceName>", hostName="<hostName>",  
port="<port>", serviceName="<serviceName>", userName="<userName>",  
passWord="<passWord>", maxConnectionSize="<maxConnectionSize>",  
maxPoolSize="<maxPoolSize>", serverName="<serverName>")
```

Argument	Definition
<i>dataSourceName</i>	Mandatory. Specifies the name of the data source to be created.
<i>hostName</i>	Mandatory. Specifies the name of the database host.
<i>port</i>	Mandatory. Specifies the database port number.
<i>serviceName</i>	Mandatory. Specifies the database service name.
<i>userName</i>	Mandatory. Specifies the OAAM schema name.
<i>passWord</i>	Mandatory. Specifies the OAAM schema password.
<i>maxConnectionSize</i>	Optional. Specifies the maximum connection reserve time out size.
<i>maxPoolSize</i>	Optional. Specifies the maximum size for the connection pool.

Argument	Definition
<i>serverName</i>	Optional. Specifies the target server for the data source.

### Example

The following example configures a basic integration for Access Manager and OAAM.

```
configureOAAMPartner (dataSourceName="MyOAAMDS", hostName="host.example.com",
port="1521", serviceName="sevice1", userName="username", passWord="password",
maxConnectionSize=None, maxPoolSize=None, serverName="oam_server1")
```

## 2.1.29 registerOIFDAPPartner

The registerOIFDAPPartner command is an online and offline command that registers Oracle Access Management Identity Federation (Identity Federation) as a Delegated Authentication Protocol (DAP) Partner.

### Description

Registers Identity Federation as Delegated Authentication Protocol (DAP) Partner. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
registerOIFDAPPartner (keystoreLocation="/scratch/keystore"
logoutURL="http://<oifhost>:<oifport>/fed/user/splooaamlg?
doneURL=http(s)://<oamhost>:<oamport>/oam/server/pages/logout.jsp",
rolloverTime="nnn")
```

Argument	Definition
<i>keystoreLocation</i>	Mandatory. Specifies the location of the Keystore file (generated at the Identity Federation Server).
<i>logoutURL</i>	Mandatory. Specifies the logout URL for the Identity Federation server.
<i>rolloverTime</i>	Optional. Specifies the amount of time in seconds for which the keys used to encrypt/decrypt SASSO tokens can be rolled over.

### Example

The following example illustrates the use of the parameters.

```
registerOIFDAPPartner (keystoreLocation="/scratch/keystore",
logoutURL="http(s)://oif.mycompany.com:1234/fed/user/splooaamlg?
doneURL=http(s)://oam.mycompany.com:5678/oam/server/pages/logout.jsp",
rolloverTime="500")
```

## 2.1.30 registerOIFDAPPartnerIDPMode

The registerOIFDAPPartnerIDPMode command is an online and offline command that registers Identity Federation as a Delegated Authentication Protocol (DAP) Partner in IDP Mode.

### Description

Registers Identity Federation as Delegated Authentication Protocol (DAP) Partner in IDP Mode. The scope of this command is an instance only; the scope is not an argument.

## Syntax

```
registerOIFDAPPartnerIDPMode (logoutURL="http://<oifhost>:<oifport>/fed/user/sploosso?
doneURL=http://<oamhost>:<oamport>/ngam/server/pages/logout.jsp")
```

Argument	Definition
<i>logoutURL</i>	Mandatory. Specifies the logout URL for the Identity Federation server.

## Example

The following example illustrates the use of the logout URL parameter.

```
registerOIFDAPPartner (
logoutURL="http://oif.oracle.com:1234/fed/user/sploosso?
doneURL=http://oam.oracle.com:5678/ngam/server/pages/logout.jsp")
```

## 2.1.31 registerThirdPartyTAPPartner

The registerThirdPartyTAPPartner command is an online command that registers any third party as a Trusted Authentication Protocol (TAP) Partner.

### Description

Registers any third party as a Trusted Authentication Protocol (TAP) Partner.

### Syntax

```
registerThirdPartyTAPPartner (partnerName="ThirdPartyTAPPartner",
keystoreLocation="/scratch/DAPKeyStore/mykeystore.jks",
password="test", tapTokenVersion="v2.0", tapScheme="TAPScheme",
tapRedirectUrl="http://thirdpartyserverhost:port/loginPage.jsp")
```

Argument	Definition
<i>partnerName</i>	Mandatory. Specifies the name of the partner. Can be any name used to identify the third party partner.
<i>keystoreLocation</i>	Mandatory. Specifies the location of the keystore file.
<i>password</i>	Mandatory. Specifies the password for the keystore file.
<i>tapTokenVersion</i>	Mandatory. Specifies the version of the Trusted Authentication Protocol.
<i>tapScheme</i>	Optional. Specifies the TAPScheme name used to protect the resource - TAPScheme, out of the box.
<i>tapRedirectUrl</i>	Optional. Specifies the TAP challenge URL to which the credential collector will be redirected.

### Example

The following example illustrates the use of the parameters.

```
registerThirdPartyTAPPartner (partnerName = "ThirdPartyTAPPartner",
keystoreLocation="/scratch/DAPKeyStore/mykeystore.jks",
password="test", tapTokenVersion="v2.0", tapScheme="TAPScheme",
tapRedirectUrl="http://thirdpartyserverhost:port/loginPage.jsp")
```

## 2.1.32 disableCoexistMode

The `disableCoexistMode` command is an online command that disables Coexist Mode.

### Description

Disables Coexist Mode. The scope of this command is an instance only; the scope is not an argument. There are no arguments for this command.

### Syntax

```
disableCoexistMode()
```

### Example

The following example disables Coexist Mode.

```
disableCoexistMode()
```

## 2.1.33 enableOamAgentCoexist

The `enableOamAgentCoexist` command is an online command that enables coexist mode for the Access Manager agent (enabling the Access Manager 11g server to own the Obsocookie set by 10g WebGate).

### Description

Enables Coexist Mode for the Access Manager agent. The scope of this command is an instance only; the scope is not an argument. There are no arguments for this command.

### Syntax

```
enableOamAgentCoexist()
```

### Example

The following example enables the Coexist Mode.

```
enableOamAgentCoexist
```

## 2.1.34 disableOamAgentCoexist

The `disableOamAgentCoexist` command is an online command that disables coexist mode for the Access Manager agent.

### Description

Disables the Coexist Mode for the Access Manager agent. The scope of this command is an instance only; the scope is not an argument. There are no arguments for this command.

### Syntax

```
disableOamAgentCoexist()
```

### Example

The following invocation enables the Coexist Mode.

```
disableOamAgentCoexist
```



## 2.1.35 editGITOVAlues

The editGITOVAlues command is an online and offline command that edits GITO configuration parameters.

### Description

Edits GITO configuration parameters. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
editGITOVAlues(gitoEnabled="true", gitoCookieDomain=".abc.com",
gitoCookieName="ABC", gitoVersion="v1.0", gitoTimeout="20",
gitoSecureCookieEnabled="false", domainHome="/abc/def/ijk")
```

Argument	Definition
<i>gitoEnabled</i>	Allows (or denies) user to set GITO enabled property. Takes a value of true or false.
<i>gitoCookieDomain</i>	Mandatory. Specifies the GITO cookie domain.
<i>gitoCookieName</i>	Optional. Specifies the cookie name.
<i>gitoVersion</i>	Optional. Specifies the GITO version. Takes ONLY v1.0 or v3.0.
<i>gitoTimeout</i>	Optional. Specifies the GITO timeout value.
<i>gitoSecureCookieEnabled</i>	Optional. Enables the GITO cookie enabled property. Takes a value of true or false.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

### Example

The following example edits the GITO configuration parameters.

```
editGITOVAlues(gitoEnabled="true", gitoCookieDomain=".abc.com",
gitoCookieName="ABC", gitoVersion="v1.0", gitoTimeout="20",
gitoSecureCookieEnabled="false", domainHome="/abc/def/ijk")
```

## 2.1.36 editWebgate11gAgent

The editWebgate11gAgent command is an online and offline command that edits an 11g Webgate\_entry registration in the system configuration.

### Description

Edits an 11g Webgate\_entry registration in the system configuration. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
editWebgate11gAgent(agentName="<AgentName>",
accessClientPasswd="<accessClientPassword >",state="<state>", preferredHost="<host>",
aaaTimeoutThreshold="<aaaTimeOutThreshold>",
security="<security>",logOutUrls="<logOutUrls>",
maxConnections="<maxConnections>",maxCacheElems="<maxCacheElements>",
```

```
cacheTimeout="<cacheTimeOut>",
logoutCallbackUrl="<logoutCallbackUrl >",maxSessionTime="<maxSessionTime>",
logoutRedirectUrl="<logoutRedirectUrl >",failoverThreshold="<failoverThreshold>",
tokenValidityPeriod="<tokenValidityPeriod>",logoutTargetUrlParamName="<logoutTargetUrlPar
amName>",
domainHome="<domainHome>",allowManagementOperations="<allowManagementOperations>",
allowTokenScopeOperations="<allowTokenScopeOperations>",
allowMasterTokenRetrieval="<allowMasterTokenRetrieval>",
allowCredentialCollectorOperations="<allowCredentialCollectorOperations>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the 11g WebGate Agent to be modified.
<i>accessClientPasswd</i>	Optional. Specifies the unique client password for this WebGate Agent.
<i>state</i>	Optional. Specifies whether the WebGate Agent is enabled or disabled with a value of either Enabled or Disabled, respectively.
<i>preferredHost</i>	Optional. Specifies the preferred host of the WebGate Agent. This prevents security holes that can be created if a host's identifier is not included in the Host Identifiers list. For virtual hosting, you must use the Host Identifiers feature.
<i>aaaTimeoutThreshold</i>	Optional. Specifies the number (in seconds) to wait for a response from the Access Manager run-time server. If this parameter is set, it is used as an application TCP/IP timeout instead of the default TCP/IP timeout. Default = -1 (default network TCP/IP timeout is used)
<i>security</i>	Optional. Specifies the level of transport security to and from the Access Manager run-time server. Takes as a value either open, simple, or cert.
<i>logOutUrls</i>	List of URLs that trigger the logout handler, which removes the ObSSOCookie.
<i>maxConnections</i>	Optional. Specifies the maximum number of connections that this Access Manager Agent can establish with the Access Manager Server. This number must be the same as (or greater than) the number of connections that are actually associated with this agent. Default = 1
<i>maxCacheElems</i>	Optional. Specifies the maximum number of elements maintained in the cache. Cache elements are URLs or Authentication Schemes. The value of this setting refers to the maximum consolidated count for elements in both of these caches. Default = 10000
<i>cacheTimeout</i>	Optional. Specifies the amount of time cached information remains in the Access Manager Agent cache when the information is neither used nor referenced. Default = 1800 (seconds)
<i>logoutCallbackUrl</i>	The URL to oam_logout_success, which clears cookies during the call back. By default, this is based on the Agent base URL supplied during agent registration. For example: http://<host>:<port>
<i>maxSessionTime</i>	Optional. Specifies the maximum amount of time in seconds that a user's authentication session is valid regardless of their activity. At the expiration of this time, the user is re-challenged for authentication. This is a forced logout. A value of 0 disables this timeout setting. Default = 3600 (seconds)
<i>logoutRedirectUrl</i>	Optional. Specifies the URL (absolute path) to the central logout page (logout.html). By default, this is based on the Access Manager Administration Console host name with a default port of 14200.

Argument	Definition
<i>failoverThreshold</i>	Optional. Specifies a number representing the point when this Access Manager Agent opens connections to a Secondary Access Manager Server. Default = 1
<i>tokenValidityPeriod</i>	Optional. Specifies the amount of time in seconds that a user's authentication session remains valid without accessing any Access Manager Agent protected resources.
<i>logoutTargetUrlParamName</i>	Optional. The value for this is the Logout Target URL to be invoked on logout and configured at the OPSS level.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.
<i>allowManagementOperations</i>	Optional. Specifies the Set the flag for Allow Management Operations
<i>allowTokenScopeOperations</i>	Optional. Specifies the Set the flag for Allow Token Scope Operations
<i>idleSessionTimeout</i>	Optional. Specifies the
<i>allowMasterTokenRetrieval</i>	Set flag for Allow Master Token Retrieval
<i>allowCredentialCollectorOperations</i>	Set flag for Allow Credential Collector Operations

### Example

The following example uses all mandatory and optional parameters.

```
editWebgate11gAgent(agentName="WebgateAgent1", accessClientPasswd="welcome1",
state="Enabled", preferredHost="141.144.168.148:2001", aaaTimeoutThreshold="10",
security="open", logOutUrls="http://host1.oracle.com:1234", maxConnections = "16",
maxCacheElems="10000", cacheTimeout="1800",
logoutCallbackUrl="http://host2.oracle.com:1234",
maxSessionTime="24", logoutRedirectUrl="logoutRedirectUrl",
failoverThreshold="1", tokenValidityPeriod="tokenValidityPeriod",
logoutTargetUrlParamName="logoutTargetUrl", domainHome="domainHome1",
allowManagementOperations="false", allowTokenScopeOperations="false",
allowMasterTokenRetrieval="false", allowCredentialCollectorOperations="false")
```

## 2.1.37 deleteWebgate11gAgent

The deleteWebgate11gAgent command is an online and offline command that enables you to remove an 11g Webgate\_agent entry in the system configuration.

### Description

Removes an 11g Webgate\_agent entry in the system configuration. The scope of this command is an instance only; the scope is not an argument.

**Syntax**

```
deleteWebgate11gAgent (agentName="<AgentName>", domainHome="<domainHomeName>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the 11g WebGate Agent to be removed.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

**Example**

The following example removes the 11g Webgate\_agent entry named `my_11gWebGate`.

```
deleteWebgate11gAgent (agentName="my_11gWebGate", domainHome="domainHome1")
```

## 2.1.38 displayWebgate11gAgent

The `displayWebgate11gAgent` command is an online and offline command that enables you to display an 11g Webgate\_agent registration entry.

**Description**

Displays an 11g WebGate Agent registration entry. The scope of this command is an instance only; the scope is not an argument.

**Syntax**

```
displayWebgate11gAgent (agentName="<AgentName>", domainHome="<domainHomeName>")
```

Argument	Definition
<i>agentName</i>	Mandatory. Specifies the name of the 11g WebGate Agent to be modified.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

**Example**

The following example displays the WebGate Agent named `my_11gWebGate`:

```
displayWebgate11gAgent (agentName="my_11gWebGate", domainHome="domainHome1")
```

## 2.1.39 displayOAMMetrics

The `displayOAMMetrics` command is an online and offline command that enables the display of metrics for Access Manager Servers.

**Description**

Enables the display of metrics for Access Manager Servers. The scope of this command is an instance only; the scope is not an argument.

**Syntax**

```
displayOAMMetrics (domainHome="<domainHomeName>")
```

Argument	Definition
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

**Example**

The following example displays the metrics for Access Manager Servers in the specified domain.

```
displayOAMMetrics (domainHome="domainHome1")
```

## 2.1.40 updateOIMHostPort (deprecated)

DEPRECATED - Online only command that updates the Oracle Identity Manager configuration when integrated with Access Manager.

**Description**

Updates the Identity Manager configuration in the system configuration. The scope of this command is an instance only; the scope is not an argument.

**Syntax**

```
updateOIMHostPort (hostName="<host name>", port="<port number>", secureProtocol="true")
```

Argument	Definition
<i>hostName</i>	Name of the Identity Manager host.
<i>port</i>	Port of the Identity Manager host.
<i>secureProtocol</i>	Takes a value of true or false depending on whether communication is through HTTP or HTTPS.

**Example**

The following example illustrates this command.

```
updateOIMHostPort (hostName="OIM.oracle.com", port="7777", secureProtocol="true")
```

## 2.1.41 configureOIM (deprecated)

DEPRECATED - Online only command that registers an agent profile specific to Oracle Identity Manager when integrated with Access Manager.

**Description**

Creates an Agent profile specific to Oracle Identity Manager when integrated with Access Manager. The scope of this command is an instance only; the scope is not an argument.

## Syntax

```
configureOIM(oimHost="<OIM host>", oimPort="<port>",
oimSecureProtocolEnabled="true | false", oimAccessGatePwd="<AccessGatePassword>",
oimCookieDomain="<OIMCookieDomain>", oimWgId="<OIMWebgateID>",
oimWgVersion="<OIMWebgateVersion>")
```

Argument	Definition
<i>oimHost</i>	Name of the Oracle Identity Manager host. In the case of EDG, the front ending LBR hostname of the OIM Cluster.
<i>oimPort</i>	Port of the Oracle Identity Manager Managed Server. In the case of EDG, the front ending LBR port of the OIM Managed Server Cluster.
<i>oimSecureProtocolEnabled</i>	Takes a value of true or false depending on whether communication is through HTTP or HTTPS.
<i>oimAccessGatePwd</i>	If provided, the agent password for Open mode.
<i>oimCookieDomain</i>	Domain in which the cookie is to be set .
<i>oimWgId</i>	Agent registration name.
<i>oimWgVersion</i>	Possible values are 10g or 11g. If not provided, default is 10g.

## Example

The following example illustrates this command.

```
configureOIM(oimHost="oracle.com", oimPort="7777", oimSecureProtocolEnabled="true",
oimAccessGatePwd = "welcome", oimCookieDomain = "domain1",
oimWgId="<OIM Webgate ID>", oimWgVersion="10g")
```

## 2.1.42 updateOSSOResponseCookieConfig

The updateOSSOResponseCookieConfig command is an online and offline command that updates the OSSO Proxy response cookie settings.

### Description

Updates OSSO Proxy response cookie settings. The scope of this command is an instance only; the scope is not an argument.

### Syntax

```
updateOSSOResponseCookieConfig(cookieName="<cookieName>", cookieMaxAge="<cookie age in minutes>",
isSecureCookie="true | false", cookieDomain="<domain of the cookie>",
domainHome="<domainHomeName>")
```

Argument	Definition
<i>cookieName</i>	Optional. Name of the cookie for which settings are updated. If not specified, the global setting is updated.
<i>cookieMaxAge</i>	Maximum age of a cookie in minutes. A negative value sets a session cookie.
<i>isSecureCookie</i>	Boolean flag that specifies if cookie should be secure (sent over SSL channel).
<i>cookieDomain</i>	The domain of the cookie.

Argument	Definition
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

**Example**

The following example illustrates this command.

```
updateOSSOResponseCookieConfig(cookieName = "ORASSO_AUTH_HINT",
cookieMaxAge = "525600", isSecureCookie = "false",
cookieDomain=".example.com", domainHome = "<domain_home>")
```

## 2.1.43 deleteOSSOResponseCookieConfig

The deleteOSSOResponseCookieConfig command is an online and offline command that deletes the OSSO Proxy response cookie settings in the system configuration.

**Description**

Deletes the OSSO Proxy response cookie settings. The scope of this command is an instance only; the scope is not an argument.

**Syntax**

```
deleteOSSOResponseCookieConfig(cookieName="<cookieName>",
domainHome="<domainHomeName>")
```

Argument	Definition
<i>cookieName</i>	Mandatory. Name of the cookie for which settings are being deleted. The global cookie setting cannot be deleted.
<i>domainHome</i>	Specifies the location for the Weblogic Server OR Cell Path for WebSphere. This parameter is mandatory for WebSphere. When Offline, a value is mandatory; when online, optional.

**Example**

The following example illustrates this command.

```
deleteOSSOResponseCookieConfig(cookieName="ORASSO_AUTH_HINT",
domainHome = "<domain_home>")
```

## 2.1.44 configureAndCreateIdentityStore

The configureAndCreateIdentityStore command is an online command that configures the identity store and external user store.

**Description**

Configures the identity store and external user store using the values supplied.

**Syntax**

```
configureOIM(oimHost="<OIM host>", oimPort="<port>",
oimSecureProtocolEnabled="true | false", oimAccessGatePwd="<AccessGatePassword>",
oimCookieDomain="<OIMCookieDomain>", oimWgId="<OIMWebgateID>",
```

```
oimWgVersion="<OIMWebgateVersion>"), nameOfIdStore="<nameOfIdStore>",
idStoreSecurityCredential="<idStoreSecurityCredential>",
userSearchBase="<userSearchBase>", ldapUrl="<ldapUrl>",
groupSearchBase="<groupSearchBase>", securityPrincipal="<securityPrincipal>",
idStoreType="<idStoreType>", ldapProvider="<ldapProvider>",
isPrimary="<isPrimary>", userIDProvider="<userIDProvider>",
userNameAttr="<userNameAttr>"
```

Argument	Definition
<i>oimHost</i>	Name of the Oracle Identity Manager host. In the case of EDG, the front ending LBR hostname of the OIM Cluster.
<i>oimPort</i>	Port of the Oracle Identity Manager Managed Server. In the case of EDG, the front ending LBR port of the OIM Managed Server Cluster.
<i>oimSecureProtocolEnabled</i>	Takes a value of true or false depending on whether communication is through HTTP or HTTPS.
<i>oimAccessGatePwd</i>	If provided, the agent password for Open mode.
<i>oimCookieDomain</i>	Domain in which the cookie is to be set .
<i>oimWgId</i>	Agent registration name.
<i>oimWgVersion</i>	Possible values are 10g or 11g. If not provided, default is 10g.
<i>nameOfIdStore</i>	Mandatory. Specifies the name of the LDAP ID store to be created.
<i>idStoreSecurityCredential</i>	Mandatory. Specifies the password of the Principal for the LDAP identity store being created.
<i>userSearchBase</i>	Mandatory. Specifies the node under which user data is stored in the LDAP identity store being created.
<i>ldapUrl</i>	Mandatory. Specifies the URL for the LDAP host (including port number) of the LDAP identity store being created.
<i>groupSearchBase</i>	Mandatory. Specifies the node under which group data is stored in the LDAP identity store being created.
<i>securityPrincipal</i>	Mandatory. Specifies the Principal Administrator of the LDAP identity store being created.
<i>idStoreType</i>	Mandatory. Specifies the type of the LDAP identity store being created.
<i>ldapProvider</i>	Specifies the LDAP Provider type of the store being created.
<i>isPrimary</i>	Optional. Specifies whether the LDAP identity store being registered is the primary identity store. Takes true or false as a value.
<i>userIDProvider</i>	Specifies the user Identity Provider for the store being created.
<i>userNameAttr</i>	Mandatory. Specifies the user attributes for the store.

## Example

The following example illustrates this command.

```
configureOIM(oimHost="oracle.com", oimPort="7777", oimSecureProtocolEnabled="true",
oimAccessGatePwd = "welcome", oimCookieDomain = "domain1",
oimWgId="<OIM Webgate ID>", oimWgVersion="10g"
nameOfIdStore="nameOfIdStore",
idStoreSecurityCredential="idStoreSecurityCredential",
userSearchBase="userSearchBase", ldapUrl="ldapUrl",
groupSearchBase="groupSearchBase", securityPrincipal="securityPrincipal",
idStoreType="idStoreType", ldapProvider="ldapProvider", isPrimary="true",
userIDProvider="userIDProvider", userNameAttr="userNameAttr")
```



## 2.1.45 configAndCreateIdStoreUsingPropFile

The `configAndCreateIdStoreUsingPropFile` command is an online command that configures the identity store and external user store using the values supplied in a properties file.

### Description

Configures the identity store and external user store using the values supplied in the specified properties file.

### Syntax

```
configAndCreateIdStoreUsingPropFile(path="<path_of_property_file>")
```

Argument	Definition
<i>path</i>	Path to the property file in which the values are defined.

### Example

The following example illustrates this command.

```
configAndCreateIdStoreUsingPropFile(path="/prop_file_directory/values.properties")
```

## 2.1.46 manageAuditEvents

Use the `manageAuditEvents` command to disable the audit of events of a specific type.

### Description

The `manageAuditEvents` is an event type filter that allows you to disable the audit of events of a specific type. The command can be run in online mode (after `domainRuntime()` is run) or offline mode.

### Syntax

```
manageAuditEvents(eventType="<event type>", auditEnabled="<true|false>",  
[eventStatus="<true|false>"], [presetFilter="<All|Medium|Low>"])
```

Argument	Definition
<code>eventType</code>	Mandatory Specifies the event type to be filtered. Its value is as defined in the <code>audit.log</code> file by the <code>EventType</code> field or in the <code>IAU_BASE</code> table by the <code>iau_eventtype</code> field. It is case sensitive.
<code>auditEnabled</code>	Mandatory Set to true to disable the audit. Setting it to true is equivalent to removing the filter.
<code>eventStatus</code>	Optional Enables or Disables audit of events of specific event status. Set to true (false) to disable only the audit of events, which status is true (false).
<code>presetFilter</code>	Optional Activates the filter only for the specified preset filter.

**Example 2-1 Examples**

The following command disables the audit of all `PluginInvocationStart` events, in offline mode.

```
manageAuditEvents(eventType="PluginInvocationStart",auditEnabled="false",domainHome="/MW_HOME/user_projects/domains/oam_domain")
```

The following command disables the audit of successful `Authorization` events, in online mode.

```
manageAuditEvents(eventType="Authorization",auditEnabled="false",eventStatus="true")
```

The following command disables the audit of all the events of type `Authorization`, in online mode.

```
manageAuditEvents(eventType="Authorization",auditEnabled="false")
```

## 2.1.47 migrateArtifacts (deprecated)

DEPRECATED - Migrates artifacts.

**Description**

Migrates artifacts based on the values defined in the input artifact file.

**Syntax**

```
migrateArtifacts(path="<path_to_artifacts_file>", password="<password>", type="OutOfPlace|InPlace", isIncremental="true|false")
```

Argument	Definition
<i>path</i>	Location of the artifacts file
<i>password</i>	Password used while generating original artifacts.
<i>type</i>	Boolean that defines the type of migration and takes as a value <code>InPlace</code> or <code>OutOfPlace</code>
<i>isIncremental</i>	Boolean that takes a value of true or false. If true, an incremental upgrade is done.

**Example**

The following example illustrates this command.

```
migrateArtifacts(path="/expleroot/parent/t", password="welcome", type="InPlace", isIncremental="false")
```

## 2.1.48 displaySimpleModeGlobalPassphrase

The `displaySimpleModeGlobalPassphrase` command is an online command that displays the simple mode global passphrase defined in the system configuration in plain text.

**Description**

Online only command that displays the simple mode global passphrase in plain text. There are no arguments for this command.

**Syntax**

```
displaySimpleModeGlobalPassphrase()
```

**Example**

The following example illustrates this command.

```
displaySimpleModeGlobalPassphrase()
```

## 2.1.49 exportSelectedPartners

The exportSelectedPartners command is an online command that exports selected Access Manager Partners to the specified Access Manager file.

**Description**

Exports selected Access Manager Partners to the specified Access Manager file specified.

**Syntax**

```
exportSelectedPartners (pathTempOAMPartnerFile="<absoluteFilePath>",  
partnersNameList="<comma_separated_partner_names>")
```

Argument	Definition
<i>pathTempOAMPartnerFile</i>	Mandatory. The location of the file to which the information will be exported.
<i>partnersNameList</i>	Mandatory. Specifies a comma separated list of partner ids being exported.

**Example**

The following example illustrates this command.

```
exportSelectedPartners (pathTempOAMPartnerFile="/expleroot/parent/tempfile.extn"  
partnersNameList="partner1,partner2")
```

## 2.1.50 oamMigrate

The oamMigrate command is an online command that migrates policies, authentication stores, and user stores from OSSO, OAM10g, OpenSSO, or AM 7.1 to OAM11g.

**Description**

Invokes the beginMigrate operation of the migration framework mbean.

**Syntax**

```
oamMigrate (oamMigrateType=<migrationType>,  
pathMigrationPropertiesFile="<absoluteFilePath>")
```

Argument	Definition
<i>oamMigrateType</i>	Mandatory. Specifies the type of migration being done. Takes one of the following as a value: OSSO   OpenSSO   OAM10g NOTE: OpenSSO applies to both SAML 7.1 and OpenSSO.

Argument	Definition
<i>pathMigrationPropertiesFile</i>	Mandatory. Specifies the path to the file from which the necessary artifacts for migration are read.

**Example**

The following example illustrates this command.

```
oamMigrate (oamMigrateType=OSSO,
pathMigrationPropertiesFile="/middlewarehome/oam-migrate.properties")
```

## 2.1.51 preSchemeUpgrade

The preSchemeUpgrade command is an online command that invokes the preSchemeUpgrade operation.

**Description**

Invokes the preSchemeUpgrade operation.

**Syntax**

```
preSchemeUpgrade
(pathUpgradePropertiesFile="/middlewarehome/oam-upgrade.properties")
```

Argument	Definition
<i>pathUpgradePropertiesFile</i>	Mandatory. Specifies the path to the file from which the necessary system properties for upgrade are read.

**Example**

The following example illustrates this command.

```
preSchemeUpgrade (pathUpgradePropertiesFile="/exampleroot/parent/tempfile.extn")
```

## 2.1.52 postSchemeUpgrade

The postSchemeUpgrade command is an online command that invokes the postSchemeUpgrade operation.

**Description**

Invokes the postSchemeUpgrade operation.

**Syntax**

```
postSchemeUpgrade
(pathUpgradePropertiesFile="/middlewarehome/oam-upgrade.properties")
```

Argument	Definition
<i>pathUpgradePropertiesFile</i>	Mandatory. Specifies the path to the file from which the necessary system properties for upgrade are read.

**Example**

The following example illustrates this command.

```
postSchemeUpgrade (pathUpgradePropertiesFile="/expleroot/parent/tempfile.extn")
```

## 2.1.53 oamSetWhiteListMode

The `oamSetWhiteListMode` command is an online command that sets the `oamSetWhiteListMode` to true or false.

**Description**

Sets the `oamSetWhiteListMode` to true or false. If true, Access Manager redirects to the last URL requested by the consuming application only if it is configured as a white-list URL.

**Syntax**

```
oamSetWhiteListMode (oamWhiteListMode="true|false")
```

Argument	Definition
<i>oamWhiteListMode</i>	Mandatory. Enables the Access Manager white list mode.

**Example**

The following example illustrates this command.

```
oamSetWhiteListMode (oamWhiteListMode="true")
```

## 2.1.54 oamWhiteListURLConfig

The `oamWhiteListURLConfig` command is an online command that performs actions on whitelist URL entries from the specified file based on the add, update, or remove operations.

**Description**

Add, update or remove whitelist URL entries from the specified file.

This command allows you to enter whitelist URL values having wildcard port/host into the WhiteList config map.

In the value field, if host/port is specified using wildcard characters (\* symbol) then all the host/port belonging to that particular format will be allowed.

On adding the \* symbol, the match will be made for the WhiteList URL based on wild card comparison mechanism.

**Syntax**

```
oamWhiteListURLConfig (Name="xyz", Value="http://xyz.com:1234",  
Operation="Remove|Update")
```

Argument	Definition
<i>Name</i>	Mandatory. A valid string representing the name (key) for this entry.

Argument	Definition
<i>Value</i>	Mandatory. A valid URL in the <protocol>://<host>:<port> format. If the port is not specified, default HTTP/HTTPS ports are assigned accordingly.
<i>Operation</i>	Mandatory. Takes as a value Update or Remove. Not case sensitive.

### Example

The following example illustrates this command:

```
oamWhiteListURLConfig (Name="xyz", Value="http://xyz.com:1234", Operation="Update")
```

The following example illustrates this command using wildcards for Whitelist ports:

```
oamWhiteListURLConfig (Name="xyz", Value="http://xyz.com:*", Operation="Update")
oamWhiteListURLConfig (Name="xyz", Value="http://xyz.com:*", Operation="Remove")
```

The following examples illustrates this command when host/port is specified using wild card characters in value field:

```
oamWhiteListURLConfig (Name="xyz", Value="http://*.com:7777", Operation="Update")
```

The above command will allow URL's such as `http://xyz.com:7777`, `http://abc.com:7777` and so on for redirection.

```
oamWhiteListURLConfig (Name="xyz", Value="http://xyz.com:*", Operation="Update")
```

The above command will allow URL's such as `http://xyz.com:8000`, `http://abc.com:4040` and so on for redirection.

## 2.1.55 enableMultiDataCentreMode

The `enableMultiDataCentreMode` command is an online command that enables Multi Data Center Mode.

### Description

Enables Multi Data Center Mode.

### Syntax

```
enableMultiDataCentreMode (propfile="<absoluteFilePath>")
```

Argument	Definition
<i>propFile</i>	Mandatory. Specifies the absolute path to a file from which the properties to enable the Multi Data Center are read.

### Example

The following example illustrates this command.

```
enableMultiDataCentre (propfile="/middlewarehome/oamMDCProperty.properties")
```

## 2.1.56 disableMultiDataCentreMode

The `disableMultiDataCentreMode` command is an online command that disables Multi Data Center Mode.

### Description

Disables Multi Data Center Mode. This command has no arguments.

### Syntax

```
disableMultiDataCentreMode()
```

### Example

The following example illustrates this command.

```
disableMultiDataCentreMode()
```

## 2.1.57 setMultiDataCentreClusterName

The `setMultiDataCentreClusterName` command is an online command that sets the Multi Data Center cluster name.

### Description

Sets the Multi Data Center cluster name.

### Syntax

```
setMultiDataCentreClusterName(clusterName="MyCluster")
```

Argument	Definition
<i>clusterName</i>	Mandatory. Specifies the name of the cluster.

### Example

The following example illustrates this command.

```
postSchemeUpgrade(clusterName="MyCluster")
```

## 2.1.58 setMultiDataCentreLogoutURLs

The `setMultiDataCentreLogoutURLs` command is an online command that sets the Multi Data Center Partner logout URLs.

### Description

Sets the Multi Data Center Partner logout URLs.

### Syntax

```
setMultiDataCentreLogoutURLs  
(logoutURLs="http://<host>:<port>/logout.jsp,http://<host>:<port>/logout.jsp")
```

Argument	Definition
<i>logoutURLs</i>	Mandatory. Specify a comma separated list of Multi Data Center Partner logout URLs.

**Example**

The following example illustrates this command.

```
setMultiDataCentreLogoutURLs (logoutURLs="http://localhost:6666/logout.jsp,http://localhost:8888/logout.jsp")
```

## 2.1.59 updateMultiDataCentreLogoutURLs

The updateMultiDataCentreLogoutURLs command updates the Multi Data Center Partner logout URLs.

**Description**

Updates the Multi Data Center Partner logout URLs.

**Syntax**

```
updateMultiDataCentreLogoutURLs  
(logoutURLs="http://<host>:<port>/logout.jsp,http://<host>:<port>/logout.jsp")
```

Argument	Definition
<i>logoutURLs</i>	Mandatory. Specify a comma separated list of Multi Data Center Partner logout URLs.

**Example**

The following example illustrates this command.

```
updateMultiDataCentreLogoutURLs (logoutURLs="http://localhost:7777/logout.jsp,http://localhost:9999/logout.jsp")
```

## 2.1.60 addPartnerForMultiDataCentre

The addPartnerForMultiDataCentre command is an online command that adds a partner to a Multi Data Center.

**Description**

Adds a partner to a Multi Data Center. This command is supported only in online mode and adds one partner at a time.

**Syntax**

```
addPartnerForMultiDataCentre (propfile="<absoluteFilePath>")
```

Argument	Definition
<i>propFile</i>	Mandatory. Specifies the absolute path to a file that contains the agent information.



**Example**

The following example illustrates this command.

```
addPartnerForMultiDataCentre(propfile="/middlewarehome/partnerInfo.properties")
```

## 2.1.61 removePartnerForMultiDataCentre

The `removePartnerForMultiDataCentre` command is an online command that removes a partner from Multi Data Center.

**Description**

Removes a partner from Multi Data Center. This command is supported only in online mode and removes one partner at a time.

**Syntax**

```
removePartnerForMultiDataCentre(webgateid="<webgateId")
```

Argument	Definition
<i>webgateid</i>	Mandatory. Specifies the ID of the partner to be deleted.

**Example**

The following example illustrates this command.

```
removePartnerForMultiDataCentre(webgateid="IAMSuite")
```

## 2.1.62 addOAMSSOProvider

The `addOAMSSOProvider` command is an online command that adds an Access Manager SSO provider with the given login URI, logout URI, and auto-login URI.

**Description**

This command modifies the domain `jps-config.xml` by adding an Access Manager SSO service instance with the required properties. In the event of an error, the command returns a `WLSTException`.

**Syntax**

```
addOAMSSOProvider(loginuri, logouturi, autologinuri)
```

Argument	Definition
<i>loginuri</i>	Specifies the URI of the login page. Required.
<i>logouturi</i>	Specifies the URI of the logout page. Optional. If unspecified, defaults to <code>logouturi=NONE</code> . Set to "" to ensure that ADF security calls the OPSS logout service, which uses the implementation of the class <code>OAMSSOServiceImpl</code> to clear the cookie <code>ObSSOCookie</code> . An ADF-secured web application that would like to clear cookies without logging out the user should use this setting.
<i>autologinuri</i>	Specifies the URI of the autologin page. Optional. If unspecified, it defaults to <code>autologin=NONE</code> .

**Example**

The following example illustrates this command.

```
addOAMSSOProvider(loginuri="/${app.context}/adfAuthentication",
  logouturi="/oamssso/logout.html", autologinuri="/example.cgi")
```

## 2.1.63 setDiscoveryProvider

**Description**

This command sets the fully qualified classname for the given discovery provider.

**Syntax**

```
setDiscoveryProvider(name="<Discovery Provider Name>",value="<class name>")
```

Argument	Definition
name	Name of the discovery provider
value	Fully qualified class name value.

**Examples**

The following examples illustrate this command.

```
setDiscoveryProvider('blobdiscovery','oracle.security.fed.jvt.discovery.model.
profilestate.RDBMSBlobDiscoveryProvider')
```

```
setDiscoveryProvider(name='blobdiscovery',value='oracle.security.fed.jvt.disco
very.model.profilestate.RDBMSBlobDiscoveryProvider')
```

## 2.1.64 displayDiscoveryProvider

**Description**

This command displays the fully qualified classname configured for the discovery provider.

**Syntax**

```
displayDiscoveryProvider(name="<Discovery Provider Name>")
```

Argument	Definition
name	Name of the discovery provider

## Examples

The following examples illustrate this command.

```
displayDiscoveryProvider('blobdiscovery')
```

```
displayDiscoveryProvider(name='blobdiscovery')
```

## 2.1.65 configurePluginMetadata

### Description

This command adds the plugin and plugin-metadata as specified in the `propFile` in the `oam-config.xml` file.



#### Note:

This command cannot be used to update an existing plugin.

### Syntax

```
configurePluginMetadata(name="<Plugin Name>",propFile="<path to the properties file>")
```

Argument	Definition
<code>name</code>	Name of the Plugin to be added
<code>propFile</code>	Path to the properties file with plugin metadata

### Examples

The following examples illustrate this command.

```
configurePluginMetadata('OAuthUserSelfRegistrationPlugin','/scratch/data.xml')
```

```
configurePluginMetadata(name='OAuthUserSelfRegistrationPlugin',propFile='/scratch/data.xml')
```

# 3

## Identity Federation WLST Commands

Use these custom WebLogic Scripting Tool (WLST) commands for Oracle Access Management Identity Federation (Identity Federation) to configure federation partners and partner profiles.

The Identity Federation WLST commands are organized into two categories.

- [Identity Federation Commands](#)
- [Advanced Identity Federation Commands](#)

### Note:

Identity Federation WLST commands take attributes specified as key-value pairs or only the value; Oracle Access Management Access Manager takes only key-value pairs. Thus, WLST examples in this document might be defined in either manner. This WLST example uses key-value pairs.

```
setIdPPartnerAttributeProfileEntry(attrProfileID="openid-idp-attribute-profile", messageAttributeName="http://axschema.org/namePerson", oamSessionAttributeName="name", requestFromIdP="true")
```

### 3.1 Identity Federation Commands

Use the WLST commands listed in [Table 3-1](#) to configure federation partners and partner profiles.

### Note:

The Identity Federation command definitions begin with "[addWSFed11IdPFederationPartner](#)."

**Table 3-1 WLST Commands for Identity Federation**

Use this command...	To...	Use with WLST...
<a href="#">addWSFed11IdPFederationPartner</a>	Create a WS-Fed 1.1 IdP partner.	Online
<a href="#">addWSFed11SPFederationPartner</a>	Create a WS-Fed 1.1 SP partner.	Online
<a href="#">addOpenID20IdPFederationPartner</a>	Create an OpenID 2.0 IdP partner.	Online
<a href="#">addOpenID20SPFederationPartner</a>	Create an OpenID 2.0 SP partner.	Online
<a href="#">addOpenID20GoogleIdPFederationPartner</a>	Create a Google OpenID 2.0 IdP partner.	Online
<a href="#">addOpenID20YahooIdPFederationPartner</a>	Create a Yahoo OpenID 2.0 IdP partner.	Online

**Table 3-1 (Cont.) WLST Commands for Identity Federation**

Use this command...	To...	Use with WLST...
<a href="#">addSAML11IdPFederationPartner</a>	Create an IdP federation partner, including metadata, under the SAML 1.1 protocol.	Online
<a href="#">addSAML11SPFederationPartner</a>	Create an SP federation partner, including metadata, under the SAML 1.1 protocol.	Online
<a href="#">addSAML20IdPFederationPartner</a>	Create an IdP federation partner under the SAML 2.0 protocol.	Online
<a href="#">addSAML20SPFederationPartner</a>	Create an SP federation partner under the SAML 2.0 protocol.	Online
<a href="#">addSAML20IdPFederationPartnerWithoutMetadata</a>	Create an IdP federation partner under the SAML 2.0 protocol without importing metadata.	Online
<a href="#">addSAML20SPFederationPartnerWithoutMetadata</a>	Create an SP federation partner under the SAML 2.0 protocol without importing metadata.	Online
<a href="#">configureIdPPartnerAttributeProfile</a>	Configure an IdP partner attribute profile to specify whether incoming attributes that are not part of the profile should be ignored.	Online
<a href="#">configureSAML20Logout</a>	Configure global federation logout for a SAML 2.0 federation partner.	Online
<a href="#">configureSAMLBinding</a>	Configure the preferred binding for a SAML federation partner.	Online
<a href="#">configureUserSelfRegistration</a>	Enable user self registration.	Online
<a href="#">configureUserSelfRegistrationAttr</a>	Sets which attributes from the assertion should be used as email, first name, last name or username during self registration.	Online
<a href="#">createAuthnSchemeAndModule</a>	Create an authentication scheme and module for an IdP partner.	Online
<a href="#">createIdPPartnerAttributeProfile</a>	Create an IdP partner attribute profile for a federation partner.	Online
<a href="#">createSPPartnerAttributeProfile</a>	Create an SP partner attribute profile for a federation partner.	Online
<a href="#">deleteAuthnSchemeAndModule</a>	Delete an authentication scheme and module for an IdP partner.	Online
<a href="#">deleteFederationPartner</a>	Delete a specific federation partner.	Online
<a href="#">deleteFederationPartnerEncryptionCert</a>	Delete the encryption certificate of a federation partner.	Online
<a href="#">deleteFederationPartnerSigningCert</a>	Delete the signing certificate of a federation partner.	Online
<a href="#">deleteIdPPartnerAttributeProfile</a>	Delete the attribute profile of an IdP federation partner.	Online
<a href="#">deleteSPPartnerAttributeProfile</a>	Delete the attribute profile of an SP federation partner.	Online

**Table 3-1 (Cont.) WLST Commands for Identity Federation**

Use this command...	To...	Use with WLST...
<a href="#">deleteIdPPartnerAttributeProfileEntry</a>	Delete an entry from the attribute profile of a federation partner.	Online
<a href="#">deleteSPPartnerAttributeProfileEntry</a>	Delete an entry from the attribute profile of a federation partner.	Online
<a href="#">deletePartnerProperty</a>	Delete a partner-specific property that was added to the partner's configuration.	Online
<a href="#">displayIdPPartnerAttributeProfile</a>	Display an IdP federation partner's attribute profile.	Online
<a href="#">displaySPPartnerAttributeProfile</a>	Display an SP federation partner's attribute profile.	Online
<a href="#">getAllFederationIdentityProviders</a>	List all IdP federation partners.	Online
<a href="#">getFederationPartnerEncryptionCert</a>	Retrieve the encryption certificate for a federation partner.	Online
<a href="#">getFederationPartnerSigningCert</a>	Retrieve the signing certificate for a federation partner	Online
<a href="#">getIdPPartnerBasicAuthCredentialUsername</a>	Retrieve the HTTP basic authentication username for a federation partner.	Online
<a href="#">getPartnerProperty</a>	Retrieve a property for a federation partner.	Online
<a href="#">getStringProperty</a>	Retrieve a string property from a federation partner profile.	Online
<a href="#">isFederationPartnerPresent</a>	Check whether a partner is configured.	Online
<a href="#">listIdPPartnerAttributeProfileIDs</a>	List an IdP partner's attribute profiles.	Online
<a href="#">listSPPartnerAttributeProfileIDs</a>	List an SP partner's attribute profiles.	Online
<a href="#">putStringProperty</a>	Sets an OpenID partner as the default Federation IdP.	Online
<a href="#">setDefaultSSOIdPPartner</a>	Set an IdP partner as the default identity provider for a federation single sign-on.	Online
<a href="#">setFederationPartnerEncryptionCert</a>	Set the encryption certificate for a federation partner.	Online
<a href="#">setFederationPartnerSigningCert</a>	Set the signing certificate for a federation partner.	Online
<a href="#">setIdPPartnerAttributeProfile</a>	Set the attribute profile to use during federated single sign-on with an IdP partner.	Online
<a href="#">setIdPDefaultScheme</a>	Sets the default OAM Authentication Scheme.	Online
<a href="#">setSPPartnerAttributeProfile</a>	Set the attribute profile to use during federated single sign-on with an SP partner.	Online
<a href="#">setIdPPartnerAttributeProfileEntry</a>	Set an entry in an IdP federation partner's profile.	Online
<a href="#">setSPPartnerAttributeProfileEntry</a>	Set an entry in an SP federation partner's profile.	Online

**Table 3-1 (Cont.) WLST Commands for Identity Federation**

Use this command...	To...	Use with WLST...
<a href="#">setSPPartnerAttributeValueMapping</a>	add or update an outgoing attribute value mappings in an SP profile.	Online
<a href="#">deleteSPPartnerAttributeValueMapping</a>	delete one or all the value mappings of an outgoing attribute configured in an SP profile.	Online
<a href="#">displaySPPartnerAttributeValueMapping</a>	display the value mappings of one or all outgoing attributes configured in an SP profile.	Online
<a href="#">setIdPPartnerAttributeValueMapping</a>	add or update an incoming attribute value mappings in an IdP profile.	Online
<a href="#">deleteIdPPartnerAttributeValueMapping</a>	delete one or all the value mappings of an incoming attribute configured in an IdP profile.	Online
<a href="#">displayIdPPartnerAttributeValueMapping</a>	display the value mappings of one or all incoming attributes configured in an IdP profile.	Online
<a href="#">setSPPartnerAttributeValueFilter</a>	add or update an attribute value filter in an SP profile.	Online
<a href="#">deleteSPPartnerAttributeValueFilter</a>	delete one or all the value filters of an attribute configured in an SP profile.	Online
<a href="#">displaySPPartnerAttributeValueFilter</a>	display the value filters of one or all attributes configured in an SP profile.	Online
<a href="#">setIdPPartnerBasicAuthCredential</a>	Update a federation partner's HTTP basic auth credential.	Online
<a href="#">setIdPPartnerMappingAttribute</a>	Set the attribute used for assertion mapping for a federation partner.	Online
<a href="#">setIdPPartnerMappingAttributeQuery</a>	Set the attribute query used for assertion mapping for a federation partner.	Online
<a href="#">setIdPPartnerMappingNameID</a>	Set the assertion mapping nameID value for an IdP federation partner	Online
<a href="#">setPartnerAlias</a>	Update a federation partner's alias name.	Online
<a href="#">setPartnerIDStoreAndBaseDN</a>	Set a federation partner's identity store and base DN.	Online
<a href="#">setSPPartnerAlternateScheme</a>	Configure an alternate Authentication Scheme.	Online
<a href="#">setSPPartnerDefaultScheme</a>	Configure a default Authentication Scheme.	Online
<a href="#">setSPPartnerProfileDefaultScheme</a>	Configure the profile with a default Authentication Scheme.	Online
<a href="#">setSPPartnerProfileAlternateScheme</a>	Configure the profile for an alternate Authentication Scheme.	Online
<a href="#">updatePartnerMetadata</a>	Update a federation partner's metadata.	Online
<a href="#">updatePartnerProperty</a>	Update a property for a federation partner.	Online

### 3.1.1 addWSFed11IdPFederationPartner

The `addWSFed11IdPFederationPartner` command is an online command that creates a WS-Federation 1.1 IdP partner.

#### Description

Creates an IdP partner under the WS-Federation 1.1 protocol. The NameID will be mapped to the LDAP user mail attribute.

#### Syntax

```
addWSFed11IdPFederationPartner(partnerName, ssoURL, providerID, description)
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be created.
<i>ssoURL</i>	The Identity Realm Secure Token URL where users will be redirected at the IdP for WS-Federation 1.1 operations.
<i>providerID</i>	Provider ID/Issuer used in the SAML Assertion.
<i>description</i>	The description of the partner. Optional.

#### Example

```
addWSFed11IdPFederationPartner("testpartner1", "http://idp.com/wsfed11",  
"http://idp.com", description="WS-Fed IdP1")
```

### 3.1.2 addWSFed11SPFederationPartner

The `addWSFed11SPFederationPartner` command is an online command that creates a WS-Federation 1.1 SP partner.

#### Description

Creates an SP partner under the WS-Federation 1.1 protocol.

#### Syntax

```
addWSFed11SPFederationPartner(partnerName, realm, ssoURL, samlVersion,  
msftADFSCompatible, description)
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be created.
<i>realm</i>	The realm identifier for this SP partner. It will be used in the WS-Federation 1.1 protocol exchange.
<i>ssoURL</i>	The Identity Realm Secure Token URL where users will be redirected at the SP for WS-Federation 1.1 operations.
<i>samlVersion</i>	The optional SAML version indicating what kind of Assertion to issue. Takes a value of <code>saml11</code> (default) or <code>saml20</code> .



Argument	Definition
<i>msftADFSCompatible</i>	An optional boolean indicating if the issued SSO Response should be in the Microsoft ADFS compatible format WS-Trust 1.2 or WS-Trust 1.3.
<i>description</i>	The description of the partner. Optional.

**Example**

```
addWSFed11SPFederationPartner("testpartner1", "http://sp.com",
    "http://sp.com/wsfed11", description="Test SP1")
```

### 3.1.3 addOpenID20IdPFederationPartner

The `addOpenID20IdPFederationPartner` command is an online command that creates an OpenID 2.0 IdP partner.

**Description**

Creates an IdP partner under the OpenID 2.0 protocol.

**Syntax**

```
addOpenID20IdPFederationPartner(partnerName, idpSSOURL, discoveryURL, description)
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be created.
<i>idpSSOURL</i>	The initiate SSO URL of the IdP. Can be set to "" if the discovery URL is specified and intended to be used.
<i>discoveryURL</i>	The OpenID discovery URL of the IdP.
<i>description</i>	The description of the partner. Optional.

**Example**

```
addOpenID20IdPFederationPartner("testpartner1", "",
    "http://host:port/discoveryurl", description="Test IdP1")
```

### 3.1.4 addOpenID20SPFederationPartner

The `addOpenID20SPFederationPartner` command is an online that creates an OpenID 2.0 SP partner.

**Description**

Creates an SP partner under the OpenID 2.0 protocol.

**Syntax**

```
addOpenID20SPFederationPartner(partnerName, realm, ssoURL, description)
```

---

Argument	Definition
<i>partnerName</i>	The name of the partner to be created.
<i>realm</i>	The realm for the SP (RP).
<i>ssoURL</i>	The endpoint URL of the SP (RP).
<i>description</i>	The description of the partner. Optional.

---

**Example**

```
addOpenID20SPFederationPartner (partnerName="partnerID",  
    realm="http://realm.domain.com", ssoURL="http://host:port/endpoint",  
    description="some description")
```

### 3.1.5 addOpenID20GoogleIdPFederationPartner

The `addOpenID20GoogleIdPFederationPartner` command is an online command that creates an IdP partner with the name `google`.

**Description**

Creates an IdP partner with the name `google` using a discovery URL `https://www.google.com/accounts/o8/id`.

**Syntax**

```
addOpenID20GoogleIdPFederationPartner ()
```

**Example**

```
addOpenID20GoogleIdPFederationPartner ()
```

### 3.1.6 addOpenID20YahooIdPFederationPartner

The `addOpenID20YahooIdPFederationPartner` command is an online command that creates an IdP partner with the name `yahoo`.

**Description**

create an IdP partner with the name `yahoo` using a discovery URL `https://open.login.yahooapis.com/openid20/user_profile/xrds`.

**Syntax**

```
addOpenID20YahooIdPFederationPartner ()
```

**Example**

```
addOpenID20YahooIdPFederationPartner ()
```

### 3.1.7 addSAML11IdPFederationPartner

The `addSAML11IdPFederationPartner` command is an online command that creates a SAML 1.1 IdP federation partner.

#### Description

Creates a SAML 1.1 IdP federation partner.

#### Syntax

```
addSAML11IdPFederationPartner (partnerName,providerID, ssoURL,
soapURL, succinctID, description)
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be created.
<i>providerID</i>	The providerID of the partner.
<i>ssoURL</i>	The initiate SSO URL of the IdP.
<i>soapURL</i>	The artifact resolution SOAP endpoint URL of the IdP.
<i>succinctID</i>	The succinctID of the provider.
<i>description</i>	The description of the partner. Optional.

#### Example

```
addSAML11IdPFederationPartner (partnerName="partnerID",
providerID="providerA", ssoURL="http://host:port/saml11sso",
soapURL="http://host:port/soapurl", succinctID="1234",
description="somedescription")
```

### 3.1.8 addSAML11SPFederationPartner

The `addSAML11SPFederationPartner` command is an online command that creates a SAML 1.1 SP federation partner.

#### Description

Creates a SAML 1.1 SP federation partner.

#### Syntax

```
addSAML11SPFederationPartner (partnerName,providerID, ssoURL, description)
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be created.

Argument	Definition
<i>providerID</i>	The providerID of the partner.
<i>ssoURL</i>	The initiate SSO URL of the IdP.
<i>description</i>	The description of the partner. Optional.

**Example**

```
addSAML11SPFederationPartner(partnerName="partnerID", providerID="providerA",
ssoURL="http://host:port/saml11sso", description="somedescription")
```

### 3.1.9 addSAML20IdPFederationPartner

The `addSAML20IdPFederationPartner` command is an online command that creates a SAML 2.0 IdP Federation partner.

**Description**

Creates a federation partner as an identity provider for Access Manager under the SAML 2.0 protocol, and loads the partner metadata from a file.

**Syntax**

```
addSAML20IdPFederationPartner(partnerName, metadataFile, description)
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be created.
<i>metadataFile</i>	The location of the metadata file (full path).
<i>description</i>	The description of the partner. Optional.

**Example**

```
addSAML20IdPFederationPartner(partnerName="partnerID",
metadataFile="location_metadata_file", description="somedescription")
```

### 3.1.10 addSAML20SPFederationPartner

The `addSAML20SPFederationPartner` command is an online command that creates a SAML 2.0 SP Federation partner.

**Description**

Creates a federation partner as a service provider for Access Manager under the SAML 2.0 protocol, and loads the partner metadata from a file.

**Syntax**

```
addSAML20SPFederationPartner(partnerName, metadataFile, description)
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be created.
<i>metadataFile</i>	The location of the metadata file (full path).
<i>description</i>	The description of the partner. Optional.

**Example**

```
addSAML20SPFederationPartner (partnerName="partnerID",
metadataFile="location_metadata_file", description="somedescription")
```

### 3.1.11 addSAML20IdPFederationPartnerWithoutMetadata

The `addSAML20IdPFederationPartnerWithoutMetadata` command is an online command that creates a SAML20 IdP federation partner without SAML 2.0 metadata.

**Description**

Creates a SAML20 IdP federation partner without loading SAML 2.0 metadata.

**Syntax**

```
addSAML20IdPFederationPartnerWithoutMetadata (partnerName,
providerID, ssoURL, soapURL, succinctID, description)
```

Argument	Definition
<i>partnerName</i>	The name of the federation partner to be created.
<i>providerID</i>	The providerID of the partner.
<i>ssoURL</i>	The initiate SSO URL of the IdP.
<i>soapURL</i>	The artifact resolution SOAP endpoint URL of the IdP.
<i>succinctID</i>	The succinctID of the provider.
<i>description</i>	The description of the partner. Optional.

**Example**

```
addSAML20IdPFederationPartnerWithoutMetadata (partnerName="partnerName",
providerID="http://host:port", ssoURL="http://host:port/saml/sso", soapURL="http://
host:port/saml/soap",description="some description")
```

### 3.1.12 addSAML20SPFederationPartnerWithoutMetadata

The `addSAML20SPFederationPartnerWithoutMetadata` command is an online command that creates a SAML20 SP federation partner without SAML 2.0 metadata.

#### Description

Creates a SAML20 SP federation partner without loading SAML 2.0 metadata.

#### Syntax

```
addSAML20SPFederationPartnerWithoutMetadata (partnerName,
providerID, ssoURL, description)
```

Argument	Definition
<i>partnerName</i>	The name of the federation partner to be created.
<i>providerID</i>	The providerID of the partner.
<i>ssoURL</i>	The initiate SSO URL of the IdP.
<i>description</i>	The description of the partner. Optional.

#### Example

```
addSAML20SPFederationPartnerWithoutMetadata (partnerName="partnerName",
providerID="http://host:port", ssoURL="http://host:port/saml/sso",
description="somedescription")
```

### 3.1.13 configureIdPPartnerAttributeProfile

The `configureIdPPartnerAttributeProfile` command is an online command that configures an IdP partner attribute profile to process incoming attributes.

#### Description

Configures an IdP partner attribute profile to process or ignore incoming attributes not defined in the profile.

#### Syntax

```
configureIdPPartnerAttributeProfile (attrProfileID, ignoreUnmappedAttributes)
```

Argument	Definition
<i>attrProfileID</i>	The identifier referencing the IdP partner attribute profile to configure.
<i>ignoreUnmappedAttributes</i>	Determines whether incoming attributes that are not defined in the profile should be ignored. Valid values are true (ignore) or (the default) false (process).

**Example**

```
configureIdPPartnerAttributeProfile(attrProfileID="idp-attribute-profile",
ignoreUnmappedAttributes="false")
```

## 3.1.14 configureSAML20Logout

The configureSAML20Logout command is an online command that configures global federation logout for a SAML 2.0 partner.

**Description**

Configures global federation logout for a SAML 2.0 federation partner.

**Syntax**

```
configureSAML20Logout(partnerName, partnerType, enable,
saml20LogoutRequestURL, saml20LogoutResponseURL, soapURL)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>partnerType</i>	Whether the partner is a service provider or identity provider. Valid values are sp, idp.
<i>enable</i>	Enable or disable global logout for that partner. Valid values true (enable), false (disable)
<i>saml20LogoutRequestURL</i>	The SAML 2.0 logout request service URL. Optional if the partner was created using metadata, or if logout is disabled.
<i>saml20LogoutResponseURL</i>	The SAML 2.0 logout response service URL. This is optional if the partner was created using metadata, or if logout is disabled.
<i>soapURL</i>	The SAML 2.0 SOAP Service URL. This is optional if the partner was created using metadata, if logout is disabled, or if SOAP logout is not supported.

**Example**

```
configureSAML20Logout(partnerName="partnerID", partnerType="sp", enable="true",
saml20LogoutRequestURL="http://host:port/saml/logoutrequest",
saml20LogoutResponseURL="http://host:port/saml/logoutresponse",
soapURL="http://host:port/saml/soap")
```

## 3.1.15 configureSAMLBinding

The configureSAMLBinding command is an online command that specifies the binding for a SAML partner.

**Description**

Configures the preferred binding for a SAML Partner.

## Syntax

```
configureSAMLBinding(partnerName, partnerType, binding, ssoResponseBinding="httppost")
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be configured.
<i>partnerType</i>	Indicates whether the partner is a service provider or an identity provider. Valid values are sp, idp.
<i>binding</i>	Specifies the binding to use for messages other than SSO responses (authentication requests, logout messages). Valid options are httppost for HTTP-POST binding and httpredirect for HTTP-Redirect binding.
<i>ssoResponseBinding</i>	This optional attribute defines the binding to use for an SSO response. Valid options are httppost for HTTP-POST binding (the default value), httpredirect for HTTP-Redirect binding or artifact for Artifact binding.

## Example

```
configureSAMLBinding(partnerName="partnerID",
partnerType="sp", binding="httpredirect", ssoResponseBinding="httppost")
```

## 3.1.16 configureUserSelfRegistration

The `configureUserSelfRegistration` command is an online command that enables the user self-registration module.

### Description

Enables the user self-registration module.

### Syntax

```
configureUserSelfRegistration(<enabled>, <registrationURL>,
<regDataRetrievalAuthnEnabled>, <regDataRetrievalAuthnUsername>,
<regDataRetrievalAuthnPassword>, <partnerName>)
```

Argument	Definition
<i>enabled</i>	Indicates if the user self-registration module is enabled. Takes a value of true or false.
<i>registrationURL</i>	The location to which the user will be redirected for self-registration. If <code>partnerName</code> is not specified, and if <code>registrationURL</code> is empty or missing, the current property will be unchanged. If <code>partnerName</code> is specified, and if <code>registrationURL</code> is empty or missing, this property will be removed from the partner's configuration.
<i>regDataRetrievalAuthnEnabled</i>	Indicates if authentication of the registration page is enabled when contacting the server to retrieve registration data.
<i>regDataRetrievalAuthnUsername</i>	Specifies the username the registration page will send to the server when retrieving the registration data from the server.



Argument	Definition
<i>regDataRetrievalAuthnPassword</i>	Specifies the password the registration page will send to the server when retrieving the registration data from the server.
<i>partnerName</i>	Indicates the IdP partner for which to enable user self-registration. If missing, the configuration operation will be global.

**Example**

```
configureUserSelfRegistration("true", regDataRetrievalAuthnEnabled="true",
  regDataRetrievalAuthnUsername="username",
  regDataRetrievalAuthnPassword="password")
```

### 3.1.17 configureUserSelfRegistrationAttr

The `configureUserSelfRegistrationAttr` command is an online command that sets the attributes in an assertion that will be used as email, first name, last name, and username.

**Description**

Sets the attributes in an assertion that will be used as email, first name, last name and username.

**Syntax**

```
configureUserSelfRegistration(<registrationAttrName>, <assertionAttrNames>,
  <partnerName>)
```

Argument	Definition
<i>registrationAttrName</i>	The self-registration page attribute to set. Can be one of the following values: email, firstname, lastname or username.
<i>assertionAttrNames</i>	The possible attributes from the assertion that can be used to populate the self-registration page field specified as the <code>registrationAttrName</code> .
<i>partnerName</i>	Indicates the IdP partner for which to configure user self-registration. If missing, the configuration operation will be global.

**Example**

```
configureUserSelfRegistrationAttr("email", "mail,fed.nameidvalue")
```

The second parameter means that `mail` or `fed.nameidvalue` from the assertion can be used to populate the email attribute in the user's self registration page.

### 3.1.18 createAuthnSchemeAndModule

The `createAuthnSchemeAndModule` command is an online command that creates an authentication scheme that uses an OpenD IdP.

**Description**

Creates an authentication scheme that uses an OpenD IdP to protect resources in Access Manager.

**Syntax**

```
createAuthnSchemeAndModule(partnerName)
```

Argument	Definition
<i>partnerName</i>	The name of the partner for whom the scheme is to be created.

**Example**

```
createAuthnSchemeAndModule("testpartner")
```

### 3.1.19 createIdPPartnerAttributeProfile

The `createIdPPartnerAttributeProfile` command is an online command that creates an IdP attribute profile. This will contain name mapping rules used to process attributes in incoming SAML assertions.

**Description**

Creates an IdP partner attribute profile that will contain name mapping rules used to process attributes in incoming SAML Assertions.

**Syntax**

```
createIdPPartnerAttributeProfile(attrProfileID)
```

Argument	Definition
<i>attrProfileID</i>	The identifier of the IdP attribute profile.

**Example**

```
createIdPPartnerAttributeProfile(attrProfileID="idp-attribute-profile")
```

### 3.1.20 createSPPartnerAttributeProfile

The `createSPPartnerAttributeProfile` command is an online command that creates an SP attribute profile. This will contain name mapping rules used to process attributes in incoming SAML Assertions.

**Description**

Creates an SP partner attribute profile that will contain name mapping rules used to process attributes in incoming SAML Assertions.

**Syntax**

```
createSPPartnerAttributeProfile(attrProfileID)
```

Argument	Definition
<i>attrProfileID</i>	The identifier of the SP attribute profile.

**Example**

```
createSPPartnerAttributeProfile(attrProfileID="sp-attribute-profile")
```

### 3.1.21 deleteAuthnSchemeAndModule

The `deleteAuthnSchemeAndModule` command is an online command that deletes an authentication scheme for an IdP partner.

**Description**

Deletes an authentication scheme for an IdP partner.

**Syntax**

```
deleteAuthnSchemeAndModule(partnerName)
```

Argument	Definition
<i>partnerName</i>	The name of the partner whose scheme is to be deleted.

**Example**

```
deleteAuthnSchemeAndModule("testpartner")
```

### 3.1.22 deleteFederationPartner

The `deleteFederationPartner` command is an online command that deletes a federation partner from Access Manager.

**Description**

Deletes a federation partner from Access Manager.

**Syntax**

```
deleteFederationPartner(partnerName, partnerType)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be deleted.
<i>partnerType</i>	Specifies whether the partner is a service provider or an identity provider. Valid values are sp, idp.

**Example**

```
deleteFederationPartner(partnerName="partnerID", partnerType="idp")
```

### 3.1.23 deleteFederationPartnerEncryptionCert

The deleteFederationPartnerEncryptionCert command is an online command that deletes the encryption certificate of a federation partner.

#### Description

Deletes the encryption certificate of a federation partner.

#### Syntax

```
deleteFederationPartnerEncryptionCert (partnerName, partnerType)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner whose encryption certificate is to be deleted.
<i>partnerType</i>	Specifies whether the partner is a service provider or an identity provider. Valid values are sp, idp.

#### Example

```
deleteFederationPartnerEncryptionCert (partnerName="customPartner", partnerType="idp")
```

### 3.1.24 deleteFederationPartnerSigningCert

The deleteFederationPartnerSigningCert command is an online command that deletes the signing certificate of a federation partner.

#### Description

Deletes the signing certificate of a federation partner.

#### Syntax

```
deleteFederationPartnerSigningCert (partnerName, partnerType)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner whose signing certificate is to be deleted.
<i>partnerType</i>	Specifies whether the partner is a service provider or identity provider. Valid values are sp, idp.

#### Example

```
deleteFederationPartnerSigningCert (partnerName="customPartner", partnerType="idp")
```

### 3.1.25 deleteIdPPartnerAttributeProfile

The `deleteIdPPartnerAttributeProfile` command is an online command that deletes an IdP partner attribute profile.

#### Description

Deletes an IdP partner attribute profile.

#### Syntax

```
deleteIdPPartnerAttributeProfile (attrProfileID)
```

Argument	Definition
<i>attrProfileID</i>	The identifier referencing the IdP partner attribute profile.

#### Example

```
deleteIdPPartnerAttributeProfile (attrProfileID="idp-attribute-profile")
```

### 3.1.26 deleteSPPartnerAttributeProfile

The `deleteSPPartnerAttributeProfile` command is an online command that deletes an SP partner attribute profile.

#### Description

Deletes an SP partner attribute profile.

#### Syntax

```
deleteSPPartnerAttributeProfile (attrProfileID)
```

Argument	Definition
<i>attrProfileID</i>	The identifier referencing the SP partner attribute profile.

#### Example

```
deleteSPPartnerAttributeProfile (attrProfileID="sp-attribute-profile")
```

### 3.1.27 deleteIdPPartnerAttributeProfileEntry

The `deleteIdPPartnerAttributeProfileEntry` command is an online command that deletes an entry from the IdP partner attribute profile.

#### Description

Deletes an attribute from the attribute profile.

**Syntax**

```
deleteIdPPartnerAttributeProfileEntry(attrProfileID,
messageAttributeName)
```

Argument	Definition
<i>attrProfileID</i>	The identifier referencing the IdP partner attribute profile.
<i>messageAttributeName</i>	The name of the attribute to delete, as it appears in the outgoing message.

**Example**

```
deleteIdPPartnerAttributeProfileEntry(attrProfileID="idp-attribute-profile",
messageAttributeName="first_name")
```

### 3.1.28 deleteSPPartnerAttributeProfileEntry

The `deleteSPPartnerAttributeProfileEntry` command is an online command that deletes an entry from the SP Partner attribute profile.

**Description**

Deletes an attribute from the attribute profile.

**Syntax**

```
deleteSPPartnerAttributeProfileEntry(attrProfileID,
messageAttributeName)
```

Argument	Definition
<i>attrProfileID</i>	The identifier referencing the IdP partner attribute profile.
<i>messageAttributeName</i>	The name of the attribute to delete, as it appears in the outgoing message.

**Example**

```
deleteSPPartnerAttributeProfileEntry(attrProfileID="sp-attribute-profile",
messageAttributeName="first_name")
```

### 3.1.29 deletePartnerProperty

The `deletePartnerProperty` command is an online command that deletes a partner-specific property.

**Description**

Deletes a partner-specific property. Use this command only for a property that was added to the partner's configuration.

See [Advanced Identity Federation Commands](#) for information regarding SAML 1.1.

**Syntax**

```
deletePartnerProperty(partnerName,partnerType,propName)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated. By replacing the value of <partnerName> with the partner ID and including the <code>includecertinsignature</code> parameter, the certificate will be included with the signature. See <a href="#">Advanced Identity Federation Commands</a> for information regarding SAML 1.1.
<i>partnerType</i>	Specifies whether the partner is a service provider or an identity provider. Valid values are <code>sp</code> , <code>idp</code> .
<i>propName</i>	The name of the configured property to be removed.

**Example**

```
deletePartnerProperty(partnerName="partner1025", partnerType="sp/idp",
propName="includecertinsignature")
```

### 3.1.30 displayIdPPartnerAttributeProfile

The `displayIdPPartnerAttributeProfile` command is an online command that displays a partner attribute profile.

**Description**

Display the content of an IdP Partner Attribute Profile.

**Syntax**

```
displayIdPPartnerAttributeProfile(attrProfileID)
```

Argument	Definition
<i>attrProfileID</i>	The identifier referencing the IdP partner attribute profile to be displayed.

**Example**

```
displayIdPPartnerAttributeProfile(attrProfileID="idp-attribute-profile")
```

### 3.1.31 displaySPPartnerAttributeProfile

The `displaySPPartnerAttributeProfile` command is an online command that displays an SP partner attribute profile.

**Description**

Display the content of an SP Partner Attribute Profile.

**Syntax**

```
displaySPPartnerAttributeProfile(attrProfileID)
```

---

Argument	Definition
<i>attrProfileID</i>	The identifier referencing the SP partner attribute profile to be displayed.

---

**Example**

```
displaySPPartnerAttributeProfile(attrProfileID="sp-attribute-profile")
```

### 3.1.32 getAllFederationIdentityProviders

The `getAllFederationIdentityProviders` command is an online command that lists all federation identity providers.

**Description**

Displays a list of all federation identity providers for Access Manager.

**Syntax**

```
getAllFederationIdentityProviders()
```

**Example**

```
getAllFederationIdentityProviders()
```

### 3.1.33 getAllFederationServiceProviders

The `getAllFederationServiceProviders` command is an online command that lists all federation service providers.

**Description**

Displays a list of all federation service providers for Access Manager.

**Syntax**

```
getAllFederationServiceProviders()
```

**Example**

```
getAllFederationServiceProviders()
```

### 3.1.34 getFederationPartnerEncryptionCert

The `getFederationPartnerEncryptionCert` command is an online command that retrieves the encryption certificate for a partner.

**Description**

Retrieves the encryption certificate for a federation partner.

**Syntax**



Argument	Definition
<i>partnerName</i>	The ID of the partner for which the encryption certificate will be retrieved.
<i>partnerType</i>	Specifies whether the partner is a service provider or an identity provider. Valid values are sp, idp.

**Example**

```
getFederationPartnerEncryptionCert (partnerName="customPartner", partnerType="idp")
```

### 3.1.35 getFederationPartnerSigningCert

The `getFederationPartnerSigningCert` command is an online command that retrieves the signing certificate for a partner.

**Description**

Retrieves the signing certificate for a federation partner.

**Syntax**

Argument	Definition
<i>partnerName</i>	The ID of the partner for which the signing certificate will be retrieved.
<i>partnerType</i>	Specifies whether the partner is a service provider or an identity provider. Valid values are sp, idp.

**Example**

```
getFederationPartnerSigningCert (partnerName="partnerID1", partnerType="idp")
```

### 3.1.36 getIdPPartnerBasicAuthCredentialUsername

The `getIdPPartnerBasicAuthCredentialUsername` command is an online command that gets a partner's basic authentication username.

**Description**

Retrieves the HTTP basic authentication username for a federation partner.

**Syntax**

```
getIdPPartnerBasicAuthCredentialUsername (partnerName)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner for which the username will be retrieved and displayed.

**Example**

```
getIdPPartnerBasicAuthCredentialUsername (partnerName="partnerID5")
```

### 3.1.37 getPartnerProperty

The `getPartnerProperty` command is an online command that retrieves a partner property.

#### Description

Retrieves a property for a federation partner.

#### Syntax

```
getPartnerProperty(partnerName, partnerType, propName)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner for which the property will be retrieved. By replacing the value of <code>&lt;partnerName&gt;</code> with the partner ID and including the <code>includecertinsignature</code> parameter, the certificate will be included with the signature. See <a href="#">Advanced Identity Federation Commands</a> for information regarding SAML 1.1.
<i>partnerType</i>	Specifies whether the partner is a service provider or an identity provider. Valid values are <code>sp</code> , <code>idp</code> .
<i>propName</i>	The name of the property to configure.

#### Example

```
getPartnerProperty(partnerName="partnerID4", partnerType="sp",  
propName="providertrusted")
```

### 3.1.38 getStringProperty

The `getStringProperty` command is an online command that retrieves a string property for a federation partner profile.

#### Description

Retrieves a string property for a federation partner profile.

If a Partner does not have an Attribute Profile assigned to it, the default Attribute Profile (based on whether the partner is an IdP or SP) will be used. The `defaultattributeprofileidp` and `defaultattributeprofilesp` properties in the `fedserverconfig` file reference the default profiles.

#### Syntax

```
getStringProperty("/fedserverconfig/<propertyName>")
```

Argument	Definition
<i>propertyName</i>	<p>The name of the property to be retrieved.</p> <p>Default Partner Profiles are available after installation and the following properties reference them. Default property values can be retrieved by replacing <i>propertyName</i> with one of the following:</p> <ul style="list-style-type: none"> <li>• defaultpartnerprofileidsaml20: default Partner Profile for SAML 2.0 IdP Partners</li> <li>• defaultpartnerprofilespsaml20: default Partner Profile for SAML 2.0 SP Partners</li> <li>• defaultpartnerprofileidsaml11: default Partner Profile for SAML 1.1 IdP Partners</li> <li>• defaultpartnerprofilespsaml11: default Partner Profile for SAML 1.1 SP Partners</li> <li>• defaultpartnerprofileidpopenid20: default Partner Profile for OpenID 2.0 IdP Partners</li> <li>• defaultpartnerprofilespsopenid20: default Partner Profile for OpenID 2.0 SP Partners</li> <li>• If : <ul style="list-style-type: none"> <li>"defaultattributeprofileidp: default Attribute Profile for IdP Partners</li> <li>"defaultattributeprofilesps: default Attribute Profile SP Partners</li> </ul> </li> </ul>

**Example**

```
getStringProperty("/fedserverconfig/defaultpartnerprofileidpopenid20")
```

### 3.1.39 isFederationPartnerPresent

The `isFederationPartnerPresent` command is an online command that verifies if the partner is configured in Access Manager.

**Description**

Checks whether the specified federation partner is defined in Access Manager.

**Syntax**

```
isFederationPartnerPresent(partnerName, partnerType)
```

Argument	Definition
<i>partnerName</i>	The partner ID.
<i>partnerType</i>	Specifies whether the partner is a service provider or an identity provider. Valid values are sp, idp.

**Example**

```
isFederationPartnerPresent(partnerABC, SP)
```

### 3.1.40 listIdPPartnerAttributeProfileIDs

The listIdPPartnerAttributeProfileIDs command is an online command that lists the IdP partner attribute profiles.

#### Description

List the identifiers of the existing IdP Partner Attribute Profiles.

#### Syntax

```
listIdPPartnerAttributeProfileIDs()
```

#### Example

```
listIdPPartnerAttributeProfileIDs()
```

### 3.1.41 listSPPartnerAttributeProfileIDs

The listSPPartnerAttributeProfileIDs command is an online command that lists the SP partner attribute profiles.

#### Description

List the identifiers of the existing SP Partner Attribute Profiles.

#### Syntax

```
listSPPartnerAttributeProfileIDs()
```

#### Example

```
listSPPartnerAttributeProfileIDs()
```

### 3.1.42 putStringProperty

The putStringProperty command is an online command that puts a string value under a designated path in the OSTS configuration.

#### Description

Puts a string value under a designated path in the OSTS configuration.

#### Syntax

```
putStringProperty(path="/validationtemplates/username-wss-validation-template/  
StringNAME",value="TestString")
```

Argument	Definition
<i>path</i>	Path inside the configuration where the String property will be put.
<i>value</i>	The string.

**Example**

```
putStringProperty("/spglobal/defaultssoidp", "testpartner")
```

### 3.1.43 setDefaultSSOIdPPartner

The `setDefaultSSOIdPPartner` command is an online command that sets the IdP partner to serve as the default IdP for federated single sign-on (SSO).

**Description**

If not set by the federation authentication plugin at run time, sets the IdP partner to serve as the default IdP during federated SSO.

**Syntax**

```
setDefaultSSOIdPPartner (partnerName)
```

Argument	Definition
<i>partnerName</i>	ID of the partner which will serve as the default IdP for federated SSO.

**Example**

```
setDefaultSSOIdPPartner (partnerName="partner25")
```

### 3.1.44 setFederationPartnerEncryptionCert

The `setFederationPartnerEncryptionCert` command is an online command that sets the encryption certificate for a partner.

**Description**

Sets the encryption certificate for a federation partner.

**Syntax**

```
setFederationPartnerEncryptionCert (partnerName, partnerType, certFile)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated
<i>partnerType</i>	The partner type. Valid values are idp, sp.
<i>certFile</i>	The full path and name of file that stores the encryption certificate. Certificates can be in either PEM or DER format.

**Example**

```
setFederationPartnerEncryptionCert  
(partnerName="customPartner", partnerType="idp",  
certFile="/temp/encryption_cert")
```

### 3.1.45 setFederationPartnerSigningCert

The `setFederationPartnerSigningCert` command is an online command that sets the signing certificate for a federation partner.

#### Description

Sets the signing certificate for a federation partner.

#### Syntax

```
setFederationPartnerSigningCert (partnerName,partnerType,certFile)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>partnerType</i>	The partner type. Valid values are idp, sp.
<i>certFile</i>	Specifies the full path and name of file that stores the signing certificate. Certificates can be in either PEM or DER format.

#### Example

```
setFederationPartnerSigningCert
(partnerName="customPartner", partnerType="idp",
certFile="/temp/signing_cert")
```

### 3.1.46 setIdPPartnerAttributeProfile

The `setIdPPartnerAttributeProfile` command is an online command that sets the IdP partner attribute profile to use when performing a federation single sign-on with an IdP partner.

#### Description

Sets the IdP partner attribute profile to use when performing a federation single sign-on with an IdP partner.

#### Syntax

```
setIdPPartnerAttributeProfile (partnerName, attrProfileID)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>attrProfileID</i>	The IdP partner attribute profile ID to be set.

#### Example

```
setIdPPartnerAttributeProfile (partnerName="partnerID5", attrProfileID="idp-attribute-
profile")
```

### 3.1.47 setIdPDefaultScheme

The setIdPDefaultScheme command is an online command that sets the default OAM Authentication Scheme to be used to challenge a user.

#### Description

Sets the default OAM Authentication Scheme that will be used to challenge a user.

#### Syntax

```
setIdPDefaultScheme(authnScheme, appDomain, hostID,  
  authzPolicy="ProtectedResourcePolicy")
```

Argument	Definition
<i>authnScheme</i>	The OAM Authentication Scheme.
<i>appDomain</i>	Optional. The application domain in which the underlying policy components will be created.
<i>hostID</i>	Optional. The HostID to be used when creating the underlying resource policy object.
<i>authzPolicy</i>	Optional. The name of the Authorization Policy to be used to protect underlying resource policy object being created.

#### Example

```
setIdPDefaultScheme('LDAPScheme')
```

Prepend the command with "fed." if running on the WebSphere platform.

### 3.1.48 setSPPartnerAttributeProfile

The setSPPartnerAttributeProfile command is an online command that sets an SP partner attribute profile to an SP partner.

#### Description

Sets the SP partner attribute profile to use with an SP partner.

#### Syntax

```
setSPPartnerAttributeProfile(partnerName, attrProfileID)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>attrProfileID</i>	The ID of the SP partner attribute profile to be set.

**Example**

```
setSPPartnerAttributeProfile(partnerName="partnerID5", attrProfileID="sp-attribute-profile")
```

### 3.1.49 setIdPPartnerAttributeProfileEntry

The setIdPPartnerAttributeProfileEntry command is an online command that sets the IdP federation partner profile.

**Description**

Update an entry in the IdP Partner Attribute Profile.

**Syntax**

```
setIdPPartnerAttributeProfileEntry(attrProfileID, messageAttributeName, oamSessionAttributeName, requestFromIdP)
```

Argument	Definition
<i>attrProfileID</i>	The IdP partner attribute profile.
<i>messageAttributeName</i>	The name of the message attribute.
<i>oamSessionAttributeName</i>	The name of the attribute as it will appear in the Access Manager session.
<i>requestFromIdP</i>	Determines whether this attribute should be requested from the IdP partner. Valid values are true, false.

**Example**

```
setIdPPartnerAttributeProfileEntry(attrProfileID="idp-attribute-profile", messageAttributeName="first_name", oamSessionAttributeName="first_name", requestFromIdP="true")
```

### 3.1.50 setSPPartnerAttributeProfileEntry

The setSPPartnerAttributeProfileEntry command is an online command that sets the SP federation partner profile.

**Description**

Sets an entry in the SP Partner Attribute Profile.

**Syntax**

```
setSPPartnerAttributeProfileEntry(attrProfileID, messageAttributeName, value, alwaysSend)
```

Argument	Definition
<i>attrProfileID</i>	The identifier referencing the SP Partner Attribute Profile in which the entry will be set.



Argument	Definition
<i>messageAttributeName</i>	The name of the attribute as it will appear in the outgoing message.
<i>value</i>	Value of the attribute element. It can be a static string, user attribute, session attribute or a combination of those types.
<i>alwaysSend</i>	Signifies whether or not this attribute should always be sent to the SP Partner. Valid values are true, false. If false it will only be sent if the SP Partner requests it (OpenID supports this).

**Example**

```
setSPPartnerAttributeProfileEntry(attrProfileID="sp-attribute-profile",
  messageAttributeName="first_name", value="$user.attr.givenname",
  alwaysSend="true")
```

### 3.1.51 setSPPartnerAttributeValueMapping

The setSPPartnerAttributeValueMapping command is an online command that adds or updates an outgoing attribute value mappings in an SP profile.

**Description**

Adds or updates an outgoing attribute value mappings in an SP profile.

**Syntax**

```
setSPPartnerAttributeValueMapping(attrProfileID, messageAttributeName,
  sendUnmappedValues,
  localValue, externalValue, default, ignoreCase, localNull, externalNull)
```

Argument	Definition
<i>attrProfileID</i>	Id of the SP profile
<i>messageAttributeName</i>	Message attribute name as defined in the attribute name mappings.
<i>sendUnmappedValues</i>	Optional If set to true, values, for which a mapping is not defined is sent. This setting applies to the attribute itself and not to a particular mapping. Default is false for the first mapping. For the following mappings, the sendUnmappedValues value is not changed if not set.
<i>localValue</i>	Optional The local value of the attribute. Default is empty string.
<i>externalValue</i>	Optional The corresponding value to send in external messages. Default is empty string.
<i>default</i>	Optional If set to true, indicates that the external value is used if a local value can be mapped to different external values. Only one mapping can have default set to true. Default is false.

Argument	Definition
<i>ignoreCase</i>	Optional If set to true, indicates that the string comparison must be case-sensitive when matching the attribute values. Default is false.
<i>localNull</i>	Optional If set to true, indicates that the local value equals a null string (this is different from an empty string ""). Default is false.
<i>externalNull</i>	Optional If set to true, indicates that the external value equals a null string (this is different from an empty string ""). Default is false.

**Example**

```
setSPPartnerAttributeValueMapping(attrProfileID='idp-attribute-profile',
messageAttributeName='TITLE', sendUnmappedValues='true', localValue='Doctor', externalValue=
'Dr', ignoreCase="true")
```

### 3.1.52 deleteSPPartnerAttributeValueMapping

The `deleteSPPartnerAttributeValueMapping` command is an online command that deletes one or all the value mappings of an outgoing attribute configured in an SP profile.

**Description**

Deletes one or all the value mappings of an outgoing attribute configured in an SP profile

**Syntax**

```
deleteSPPartnerAttributeValueMapping(attrProfileID, messageAttributeName, localValue,
externalValue)
```

Argument	Definition
<i>attrProfileID</i>	Id of the SP profile
<i>messageAttributeName</i>	Message attribute name as defined in the attribute name mappings.
<i>localValue</i>	Optional The local value of the mapping that needs to be deleted. If both <code>localValue</code> and <code>externalValue</code> are not set, all the value mappings of the attribute are deleted.
<i>externalValue</i>	Optional The external value of the mapping that needs to be deleted. If both <code>localValue</code> and <code>externalValue</code> are not set, all the value mappings of the attribute are deleted.

**Example 1**

To delete all the value mappings of attribute `TITLE`:

```
deleteSPPartnerAttributeValueMapping(attrProfileID='sp-attribute-profile',
messageAttributeName='TITLE')
```

### Example 2

To delete a value mapping of attribute `TITLE` identified by its local and external values:

```
deleteSPPartnerAttributeValueMapping(attrProfileID='sp-attribute-profile',
messageAttributeName='TITLE',
localValue='Doctor', externalValue='Dr')
```

## 3.1.53 displaySPPartnerAttributeValueMapping

The `displaySPPartnerAttributeValueMapping` command is an online command that displays the value mappings of one or all outgoing attributes configured in an SP profile.

### Description

Displays the value mappings of one or all outgoing attributes configured in an SP profile.

### Syntax

```
displaySPPartnerAttributeValueMapping(attrProfileID, messageAttributeName)
```

Argument	Definition
<i>attrProfileID</i>	Id of the SP profile
<i>messageAttributeName</i>	Optional Message attribute name as defined in the attribute name mappings. If set, only the value mappings for the specified attribute are displayed. If not set, the value mappings for all attributes are displayed.

### Example 1

To display the value mappings of all attributes:

```
displaySPPartnerAttributeValueMapping(attrProfileID='sp-attribute-profile')
```

### Example 2

To display the value mappings of attribute `TITLE`:

```
displaySPPartnerAttributeValueMapping(attrProfileID='sp-attribute-profile',
messageAttributeName='TITLE')
```

## 3.1.54 setIdPPartnerAttributeValueMapping

The `setIdPPartnerAttributeValueMapping` command is an online command that adds or updates an incoming attribute value mappings in an IdP profile.

### Description

Adds or updates an incoming attribute value mappings in an IdP profile.

## Syntax

```
setIdPPartnerAttributeValueMapping(attrProfileID, oamSessionAttributeName,
receiveUnmappedValues,
receivedValue, externalValue, default, ignoreCase, receivedNull, externalNull)
```

Argument	Definition
<i>attrProfileID</i>	Id of the IdP profile
<i>oamSessionAttributeName</i>	OAM session attribute name, as defined in the attribute name mappings.
<i>receiveUnmappedValues</i>	Optional If set to true, values for which a mapping is not defined is received. This setting applies to the attribute itself and not to a particular mapping. Default is false for the first mapping. For the subsequent mappings, the <i>receiveUnmappedValues</i> value is not changed if not set.
<i>receivedValue</i>	Optional The value received in the message. Default is empty string.
<i>externalValue</i>	Optional The corresponding value to be saved in the session. Default is empty string.
<i>default</i>	Optional If set to true, indicates that the external value is used if a received value can be mapped to different external values. Only one mapping can have default set to true. Default is false.
<i>ignoreCase</i>	Optional If set to true, indicates that the string comparison must be case-sensitive when matching the attribute values. Default is false.
<i>receivedNull</i>	Optional If set to true, indicates that the received value equals a null string (this is different from an empty string ""). Default is false.
<i>externalNull</i>	Optional If set to true, indicates that the external value equals a null string (this is different from an empty string ""). Default is false.

## Example

```
setIdPPartnerAttributeValueMapping(attrProfileID='idp-attribute-profile',
oamSessionAttributeName='title',
receiveUnmappedValues='true', localValue='Doctor', externalValue='Dr',
ignoreCase="true")
```

### 3.1.55 deleteIdPPartnerAttributeValueMapping

The `deleteIdPPartnerAttributeValueMapping` command is an online command that deletes one or all the value mappings of an incoming attribute configured in an IdP profile.

#### Description

Deletes one or all the value mappings of an incoming attribute configured in an IdP profile.

#### Syntax

```
deleteSPPartnerAttributeValueMapping(attrProfileID, oamSessionAttributeName,
receivedValue, externalValue)
```

Argument	Definition
<i>attrProfileID</i>	Id of the IdP profile
<i>oamSessionAttributeName</i>	OAM session attribute name, as defined in the attribute name mappings.
<i>receivedValue</i>	Optional The received value of the mapping that needs to be deleted. If both <code>receivedValue</code> and <code>externalValue</code> are not set, all the value mappings of the attribute are deleted.
<i>externalValue</i>	Optional The external value of the mapping that needs to be deleted. If both <code>receivedValue</code> and <code>externalValue</code> are not set, all the value mappings of the attribute are deleted.

#### Example 1

To delete all the value mappings of attribute `TITLE`:

```
deleteIdPPartnerAttributeValueMapping(attrProfileID='idp-attribute-profile',
oamSessionAttributeName='TITLE')
```

#### Example 2

To delete a value mapping of attribute `TITLE` identified by its received and external values:

```
deleteIdPPartnerAttributeValueMapping(attrProfileID='idp-attribute-profile',
oamSessionAttributeName='TITLE',
receivedValue='Doctor', externalValue='Dr')
```

### 3.1.56 displayIdPPartnerAttributeValueMapping

The `displayIdPPartnerAttributeValueMapping` command is an online command that displays the value mappings of one or all incoming attributes configured in an IdP profile.

#### Description

Displays the value mappings of one or all incoming attributes configured in an IdP profile.

#### Syntax

```
displayIdPPartnerAttributeValueMapping(attrProfileID, oamSessionAttributeName)
```

Argument	Definition
<i>attrProfileID</i>	Id of the IdP profile
<i>oamSessionAttributeName</i>	Optional OAM session attribute name, as defined in the attribute name mappings. If set, only the value mappings for the specified attribute are displayed. If not set, the value mappings for all attributes are displayed.

**Example 1**

To display the value mappings of all attributes:

```
displayIdPPartnerAttributeValueMapping(attrProfileID='idp-attribute-profile')
```

**Example 2**

To display the value mappings of attribute `TITLE`:

```
displayIdPPartnerAttributeValueMapping(attrProfileID='idp-attribute-profile',
oamSessionAttributeName='TITLE')
```

### 3.1.57 setSPPartnerAttributeValueFilter

The `setSPPartnerAttributeValueFilter` command is an online command that adds or updates an attribute value filter in an SP profile.

**Description**

Adds or updates an attribute value filter in an SP profile.

**Syntax**

```
setSPPartnerAttributeValueFilter(attrProfileID, messageAttributeName, conditionOperator,
condition, expression, ignoreCase)
```

Argument	Definition
<i>attrProfileID</i>	Id of the SP profile
<i>messageAttributeName</i>	Message attribute name as defined in the attribute name mapping.
<i>conditionOperator</i>	Optional If set to <code>and</code> , indicates that all conditions need to be met for an attribute to be sent. If set to <code>or</code> , indicates that meeting one condition is sufficient to send an attribute. This setting applies to the attribute itself and not to a particular mapping. Default is <code>and</code> for the first filter. For the subsequent filters, the <code>conditionOperator</code> is not changed if not set

Argument	Definition
<i>condition</i>	<p><b>Mandatory</b></p> <p>The condition that is used to evaluate the attribute value. Following are the values that can be set:</p> <ul style="list-style-type: none"> <li>• equals, not-equals</li> <li>• startswith, endswith</li> <li>• contains, not-contains</li> <li>• equals-null, not-equals-null</li> <li>• regexp</li> </ul> <p>Default is empty string.</p>
<i>expression</i>	<p><b>Optional</b></p> <p>The value or regular expression that is used to evaluate the attribute value.</p> <p>Default is empty string.</p>
<i>ignoreCase</i>	<p><b>Optional</b></p> <p>If set to true, indicates that the string comparison must be case-sensitive when matching the attribute values.</p> <p>Default is false.</p>

### Example

```
setSPPartnerAttributeValueFilter(attrProfileID='sp-attribute-profile',
messageAttributeName='GROUP',
conditionOperator='and', condition='contains', expression='Sale', ignoreCase="true")
```

## 3.1.58 deleteSPPartnerAttributeValueFilter

The `deleteSPPartnerAttributeValueFilter` command is an online command that deletes one or all the value filters of an attribute configured in an SP profile.

### Description

Deletes one or all the value filters of an attribute configured in an SP profile

### Syntax

```
deleteSPPartnerAttributeValueFilter(attrProfileID, attributeName, condition, expression)
```

Argument	Definition
<i>attrProfileID</i>	Id of the SP profile
<i>messageAttributeName</i>	Message attribute name as defined in the attribute name mapping.
<i>condition</i>	<p><b>Optional</b></p> <p>The condition of the filter to be deleted.</p> <p>To delete a specific filter, the values of both the condition and expression parameters must be set. If both condition and expression parameters are not set, all value filters of the attribute are deleted.</p>

Argument	Definition
<i>expression</i>	Optional The expression of the filter to be deleted. To delete a specific filter, the values of both the condition and expression parameters must be set. If both condition and expression parameters are not set, all value filters of the attribute are deleted.

**Example 1**

To delete all the value filters of attribute `GROUP`:

```
deleteSPPartnerAttributeValueFilter(attrProfileID='idp-attribute-profile',
messageAttributeName='GROUP')
```

**Example 2**

To delete a value filter of attribute `GROUP` identified by its condition and expression values:

```
deleteSPPartnerAttributeValueFilter(attrProfileID='idp-attribute-profile',
messageAttributeName='GROUP', condition='contains', expression='Sale')
```

### 3.1.59 displaySPPartnerAttributeValueFilter

The `displaySPPartnerAttributeValueFilter` command is an online command that displays the value filters of one or all attributes configured in an SP profile.

**Description**

Displays the value filters of one or all attributes configured in an SP profile.

**Syntax**

```
displaySPPartnerAttributeValueFilter(attrProfileID, messageAttributeName)
```

Argument	Definition
<i>attrProfileID</i>	Id of the SP profile
<i>messageAttributeName</i>	Optional Message attribute name as defined in the attribute name mappings. If set, only the value filters for the specified attribute are displayed. If not set, the value filters for all attributes are displayed.

**Example 1**

To display the value filters of all attributes:

```
displaySPPartnerAttributeValueFilter(attrProfileID='idp-attribute-profile')
```

**Example 2**

To display the value filters of attribute `GROUP`:

```
displaySPPartnerAttributeValueFilter(attrProfileID='idp-attribute-profile',
messageAttributeName='GROUP')
```



### 3.1.60 setIdPPartnerBasicAuthCredential

The setIdPPartnerBasicAuthCredential command is an online command that sets a partner's basic authentication credentials.

#### Description

Sets or updates a federation partner's HTTP basic authentication credentials.

#### Syntax

```
setIdPPartnerBasicAuthCredential (partnerName, username, password)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>username</i>	The user ID of the user.
<i>password</i>	The password corresponding to the username.

#### Example

```
setIdPPartnerBasicAuthCredential (partnerName="partnerID4", username="user1")
```

### 3.1.61 setIdPPartnerMappingAttribute

The setIdPPartnerMappingAttribute command is an online command that sets a partner's assertion mapping attribute.

#### Description

Specify that an attribute from the OpenID assertion received from the IdP be mapped to a given data store attribute in order to identify the user.

#### Syntax

```
setIdPPartnerMappingAttribute (partnerName, assertionAttr, userstoreAttr)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>assertionAttr</i>	The attribute name in the assertion used to map the user to the identity store.
<i>userstoreAttr</i>	The name of the attribute in the identity store to which to map the assertion attribute value.

#### Example

```
setIdPPartnerMappingAttribute (partnerName="partnerID",  
assertionAttr="email", userstoreAttr="mail")
```

### 3.1.62 setIdPPartnerMappingAttributeQuery

The setIdPPartnerMappingAttributeQuery command is an online command that updates a partner for assertion mapping of user with attribute query.

#### Description

Sets or updates a partner to specify the attribute query to map an assertion to the user store.

#### Syntax

```
setIdPPartnerMappingAttributeQuery (partnerName, attrQuery)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated
<i>attrQuery</i>	The attribute query to be used. The LDAP query can contain placeholders referencing the attributes in the SAML Assertion, as well as the NameID. An attribute from the SAML Assertion will be referenced by its name and surrounded by the % character; for example, if the attribute name is Userlastname, the attribute will be referenced as %Userlastname%. The NameID Value is referenced as %fed.nameidvalue%.

#### Example

```
setIdPPartnerMappingAttributeQuery (partnerName="partnerID",  
attrQuery="(&(sn=%Userlastname%) (givenname=%Userfirstname%)) ")
```

### 3.1.63 setIdPPartnerMappingNameID

The setIdPPartnerMappingNameID command is an online command that sets the assertion mapping nameID value for an IdP federation partner.

#### Description

Sets the assertion mapping nameID value for an IdP federation partner.

#### Syntax

```
setIdPPartnerMappingNameID (partnerName, userstoreAttr)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>userstoreAttr</i>	The attribute name in the identity store to which the assertion nameID is to be mapped.

#### Example

```
setIdPPartnerMappingNameID  
(partnerName="partnerID", userstoreAttr="ldapattr")
```

### 3.1.64 setPartnerAlias

The setPartnerAlias command is an online command that sets a federation partner's alias.

#### Description

Sets or updates a federation partner's alias.

#### Syntax

```
setPartnerAlias (partnerName,partnerType,partnerAlias)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>partnerType</i>	Specifies the partner type. Valid values are sp or idp.
<i>partnerAlias</i>	The partner's alias.

#### Example

```
setPartnerAlias (partnerName="partnerID",  
partnerType="sp", partnerAlias="tenant1")
```

### 3.1.65 setPartnerIDStoreAndBaseDN

The setPartnerIDStoreAndBaseDN command is an online command that sets a partner's identity store and base DN of a federation partner.

#### Description

Sets or updates the identity store and base DN of a federation partner.

#### Syntax

```
setPartnerIDStoreAndBaseDN (partnerName,partnerType,storeName,searchBaseDN)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>partnerType</i>	The partner type. Valid values are sp or idp.
<i>storeName</i>	The name of the identity store.If left blank, the Default OAM Identity Store will be used. (Optional)
<i>searchBaseDN</i>	The search base DN for the LDAP. If left blank, the Search Base DN configured in the Identity Store will be used. (Optional)

### Example

```
setPartnerIDStoreAndBaseDN(partnerName="partnerID",
partnerType="sp/idp", storeName="testldap",
searchBaseDN="dc=company,dc=com")
```

## 3.1.66 setSPSAMLPartnerNameID

The setSPSAMLPartnerNameID command is an online command that updates a partner by setting the NameID during assertion issuance.

### Description

Sets the NameID for a SAML partner.

### Syntax

```
setSPSAMLPartnerNameID(<partnerName>, <nameIDFormat>, <nameIDValue>)
```

Argument	Definition
<i>partnerName</i>	The name of the partner to be configured.
<i>nameIDFormat</i>	The NameID format to be used. Possible values include: <ul style="list-style-type: none"> <li>orafed-emailaddress for urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</li> <li>orafed-x509 for urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName</li> <li>orafed-kerberos for urn:oasis:names:tc:SAML:2.0:nameid-format:Kerberos</li> <li>orafed-transient for urn:oasis:names:tc:SAML:2.0:nameid-format:transient</li> <li>orafed-windowsnamequalifier for urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName</li> <li>orafed-persistent for urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</li> <li>orafed-undefined for urn:oasis:names:tc:SAML:1.1:nameid-format:undefined</li> <li>orafed-none for no NameID</li> <li>If the format is set to any other value, the Assertion will be populated with that value.</li> </ul>
<i>nameIDValue</i>	Value of the NameID element. It can be a static string, user attribute, session attribute or a combination of those types.

### Example

```
setSPSAMLPartnerNameID(partnerName="partnerID", nameIDFormat="emailAddress",
nameIDValue="$user.attr.mail")
```

## 3.1.67 setSPPartnerAlternateScheme

The `setSPPartnerAlternateScheme` command is an online command that provides a way to authenticate clients with an alternate Authentication Scheme.

### Description

Identity Federation evaluates an HTTP Header to determine if the alternate Authentication Scheme should be used for this Partner.

### Syntax

```
setSPPartnerAlternateScheme(<partner>, <enabled="true">, <httpHeaderName="">,
<httpHeaderExpression="">, <authnScheme="">, <appDomain="IAM Suite">,
<hostID="IAMSuiteAgent">, <authzPolicy="Protected Resource Policy">,
<remove="false">)
```

Argument	Definition
<i>partner</i>	The ID of the partner.
<i>enabled</i>	Indicates whether or not Identity Federation should evaluate the HTTP Header sent by the client
<i>httpHeaderName</i>	Required if <i>enabled</i> is true, the HTTP Header to evaluate. IMPORTANT: This is a global setting and will affect all partners.
<i>httpHeaderExpression</i>	Required if <i>enabled</i> is true, this is the regular expression used to evaluate the value of the HTTP Header.
<i>authnScheme</i>	Required if <i>enabled</i> is true, the alternate Authentication Scheme to be used instead of the default.
<i>appDomain</i>	Optional. The application domain in which the underlying policy components will be created
<i>hostID</i>	Optional. The HostID used when creating the underlying resource policy object
<i>authzPolicy</i>	Optional. The Authorization Policy Name that will be used to protect underlying resource policy object being created.
<i>remove</i>	Optional. If set to true, removes the properties for the alternate scheme in the partner configuration.

### Note:

Since this operation creates policy objects, it is possible to specify the Application Domain (default: "IAM Suite"), the HostID (default "IAMSuiteAgent") and the Authorization Policy (default "Protected Resource Policy") to be used although the default values can be used.

### Example

In this example, Identity Federation is configured to enable the alternate Authentication Scheme at a partner level for the SP partner Acme because the user's browser sends the

HTTP Header "User-Agent" with the iPhone string in it. The string triggers the BasicScheme for authentication rather than the default Authentication Scheme.

```
setSPPartnerAlternateScheme("acmeSP", "true", httpHeaderName="User-Agent",
    httpHeaderExpression=".*iPhone.*", authnScheme="BasicScheme")
```

### 3.1.68 setSPPartnerDefaultScheme

The setSPPartnerDefaultScheme command is an online command that defines the default Authentication Scheme for the SP partner.

#### Description

Defines the default Authentication Scheme for the SP partner.

#### Syntax

```
setSPPartnerDefaultScheme(<partner>, <authnScheme="">, <appDomain="IAM Suite">,
    <hostID="IAMSuiteAgent">, <authzPolicy="Protected Resource Policy">)
```

Argument	Definition
<i>partner</i>	The ID of the partner.
<i>authnScheme</i>	The OAM Authentication Scheme to be used.
<i>appDomain</i>	Optional. The application domain in which the underlying policy components will be created
<i>hostID</i>	Optional. The HostID used when creating the underlying resource policy object
<i>authzPolicy</i>	Optional. The Authorization Policy Name that will be used to protect the underlying resource policy object being created.

#### Example

```
setSPPartnerDefaultScheme(partnerProfile="acmeSP",
    authnScheme="BasicScheme")
```

### 3.1.69 setSPPartnerProfileAlternateScheme

The setSPPartnerProfileAlternateScheme command is an online command that provides a way to authenticate clients with an alternate Authentication Scheme.

#### Description

Identity Federation evaluates an HTTP Header to determine if the alternate Authentication Scheme should be used for partners assigned to this Partner Profile.

#### Syntax

```
setSPPartnerProfileAlternateScheme(<partnerProfile>,
    <enabled="true">, <httpHeaderName="">, <httpHeaderExpression="">,
    <authnScheme="">, <appDomain="IAM Suite">, <hostID="IAMSuiteAgent">,
    <authzPolicy="Protected Resource Policy">, <remove="false">)
```

Argument	Definition
<i>partnerProfile</i>	The ID of the partner profile.
<i>enabled</i>	Indicates whether or not Identity Federation should evaluate the HTTP Header sent by the client
<i>httpHeaderName</i>	Required if enabled is true, the HTTP Header to evaluate. IMPORTANT: This is a global setting and will affect all partners.
<i>httpHeaderExpression</i>	Required if enabled is true, this is the regular expression used to evaluate the value of the HTTP Header.
<i>authnScheme</i>	Required if enabled is true, the alternate Authentication Scheme to be used instead of the default.
<i>appDomain</i>	Optional. The application domain in which the underlying policy components will be created
<i>hostID</i>	Optional. The HostID used when creating the underlying resource policy object
<i>authzPolicy</i>	Optional. The Authorization Policy Name that will be used to protect the underlying resource policy object being created.

 **Note:**

Since this operation creates policy objects, it is possible to specify the Application Domain (default: "IAM Suite"), the HostID (default "IAMSuiteAgent") and the Authorization Policy (default "Protected Resource Policy") to be used although the default values can be used.

**Example**

```
setSPPartnerProfileAlternateScheme("acmeSP", "true",
  httpHeaderName="User-Agent", httpHeaderExpression=".*iPhone.*",
  authnScheme="BasicScheme")
```

### 3.1.70 setSPPartnerProfileDefaultScheme

The setSPPartnerProfileDefaultScheme command is an online command that sets the default OAM authentication scheme to be used to challenge a user for a specific SP partner profile.

**Description**

Sets the default OAM Authentication Scheme to be used to challenge a user for a specific SP Partner Profile.

**Syntax**

```
setSPPartnerProfileDefaultScheme(<partnerProfile>,
  <authnScheme="">, <appDomain="IAM Suite">, <hostID="IAMSuiteAgent">,
  <authzPolicy="Protected Resource Policy">)
```

Argument	Definition
<i>partnerProfile</i>	The ID of the partner profile.
<i>authnScheme</i>	The OAM Authentication Scheme to be used.
<i>appDomain</i>	Optional. The application domain in which the underlying policy components will be created
<i>hostID</i>	Optional. The HostID used when creating the underlying resource policy object
<i>authzPolicy</i>	Optional. The Authorization Policy Name that will be used to protect the underlying resource policy object being created.

**Example**

```
setSPPartnerProfileDefaultScheme("saml20-sp-partner-profile",
  "LDAPScheme")
```

### 3.1.71 updatePartnerMetadata

The `updatePartnerMetadata` command is an online command that updates federation partner metadata.

**Description**

Updates the metadata for a federation partner.

**Syntax**

```
updatePartnerMetadata(partnerName,partnerType,metadataFile)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated
<i>partnerType</i>	Specifies the partner type. Valid values are sp or idp.
<i>metadataFile</i>	The location of the metadata file. Specify the complete path and name.

**Example**

```
updatePartnerMetadata(partnerName="partnerID",
  partnerType="sp", metadataFile="/common/idm/abc_metadata_file")
```

### 3.1.72 updatePartnerProperty

The `updatePartnerProperty` command is an online command that updates a partner property.

**Description**

Configures or updates the specified property for a federation partner.

See [Advanced Identity Federation Commands](#) for information regarding SAML 1.1.



## Syntax

```
updatePartnerProperty(partnerName,partnerType,propName,propValue,type)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated. By replacing the value of <partnerName> with the partner ID and including the <code>includecertinsignature</code> parameter, the certificate will be included with the signature. See <a href="#">Advanced Identity Federation Commands</a> for information regarding SAML 1.1.
<i>partnerType</i>	Specifies the partner type. Valid values are sp or idp.
<i>propName</i>	The name of the property to configure.
<i>propValue</i>	The property value to be set.
<i>type</i>	The data type of the property. Valid values are string, long, or boolean.

## Example

```
updatePartnerProperty(partnerName="partnerID", partnerType="idp",
propName="providertrusted",
propValue="true",type="boolean")
```

## 3.2 Advanced Identity Federation Commands

The Advanced Identity Federation WLST commands do not have applicable administrative fields for configuration in the Access Management console. Administration for Authentication mappings and partner profiles are available using WLST commands only. [Table 3-2](#) lists the Advanced Identity Federation commands documented in this section. The commands are organized as follows.

- Federation Service and Datastore
- Federation Access Configuration
- Attribute Sharing Configuration
- Authentication Method Mapping Management - All Authentication Method/Scheme/Level mappings are configured using WLST at the partner level or, if not defined at the partner level, at the partner profile level.
- Partner Profile Management - All Partner Profile management is done with WLST.
- Using WLST with SAML 1.1



### Note:

The Advanced Identity Federation command definitions begin with "`configureFederationService.`"

**Table 3-2 Advanced Identity Federation WLST Commands**

Use this command...	To...	Use with WLS T...
<b>Federation Service and Datastore</b>		
<code>configureFederationService</code>	Enable or disable Federation Service features.	
<code>setFederationStore</code>	Enables and configures the federation store.	
<b>Federation Access Configuration</b>		
<code>configureIdPAuthnRequest</code>	Configure an IdP partner or IdP partner profile for Force Authentication and/or IsPassive.	
<code>configureFedSSOAuthz</code>	Enables or disables Authorization for Federation SSO.	
<code>configureFedDigitalSignature</code>	Configure the Hashing algorithm used in digital signatures.	
<code>configureFedSignEncKey</code>	Configure the signing and/or encryption key alias to be used for digital signature and encryption operations.	
<b>Attribute Sharing Configuration</b>		
<code>configureAttributeSharingSPPartnerNameIDMapping</code>	Configures the NameID to user store attribute mapping to be used during Attribute Sharing.	
<code>configureAttributeSharingIdPPartner</code>	Configures the default attribute sharing nameid and nameid format for the IdP Partner.	
<code>configureAttributeSharingUserDNToldPPartnerMapping</code>	Configures Attribute Sharing DN to IdP Mappings.	
<code>configureAttributeSharing</code>	Configures the Attribute Sharing feature by setting a default attribute authority.	
<code>removeAttributeSharingFromAuthModule</code>	Removes the Attribute Sharing plug-in from the Authentication Module.	
<code>configureAttributeSharingPlugin</code>	Lists the Federated Authentication Method mappings for a specific Partner Profile.	
<code>insertAttributeSharingInToAuthModule</code>	Inserts the attribute sharing step into the Authentication Module flow.	
<b>Authentication Method Mapping Management</b>		
<code>setSPPartnerAlternateScheme</code>	Provides a way to authenticate clients with an alternate Authentication Scheme (Partner).	
<code>setSPPartnerDefaultScheme</code>	Defines the default Authentication Scheme for the SP partner.	
<code>setSPPartnerProfileAlternateScheme</code>	Provides a way to authenticate clients with an alternate Authentication Scheme (Partner Profile).	
<code>setSPPartnerProfileDefaultScheme</code>	Sets the default OAM Authentication Scheme to be used to challenge a user for a specific SP Partner Profile.	
<code>addSPPartnerAuthnMethod</code>	Defines a mapping between a Federated Authentication Method and an Access Manager Authentication Scheme for a specific SP Partner.	

**Table 3-2 (Cont.) Advanced Identity Federation WLST Commands**

Use this command...	To...	Use with WLS T...
<a href="#">addSPPartnerProfileAuthnMethod</a>	Defines a mapping between a Federated Authentication Method to an Access Manager Authentication Scheme for a specific SP Partner Profile.	
<a href="#">addIdPPartnerAuthnMethod</a>	Sets the Authentication Level to use when creating a session for a Federated Authentication Method for a specific IdP Partner.	
<a href="#">addIdPPartnerProfileAuthnMethod</a>	Sets the Authentication Level to use when creating a session for a Federated Authentication Method for a specific IdP Partner Profile.	
<a href="#">listPartnerAuthnMethods</a>	Lists the Federated Authentication Method mappings for a specific Partner.	
<a href="#">listPartnerProfileAuthnMethods</a>	Lists the Federated Authentication Method mappings for a specific Partner Profile.	
<a href="#">removePartnerAuthnMethod</a>	Removes the mapping between a Federated Authentication Method and Access Manager Authentication Scheme for a specific Partner.	
<a href="#">removePartnerProfileAuthnMethod</a>	Removes the mapping between a Federated Authentication Method and Access Manager Authentication Scheme for a specific Partner.	
<a href="#">setIdPPartnerRequestAuthnMethod</a>	Sets the Federated Authentication Method that will be requested during Federation SSO for a specific IdP Partner.	
<a href="#">setIdPPartnerProfileRequestAuthnMethod</a>	Sets the Federated Authentication Method that will be requested during Federation SSO for a specific IdP Partner Profile.	
<a href="#">useProxiedFedAuthnMethod</a>	Configure the Identity Provider to use the proxied Federation Authentication Method when performing Federation SSO.	
<b>Partner Profile Management</b>		
<a href="#">createFedPartnerProfileFrom</a>	Creates a Federation Partner Profile based on the specified existing one.	
<a href="#">deleteFedPartnerProfile</a>	Deletes the specified Federation Partner Profile.	
<a href="#">displayFedPartnerProfile</a>	Displays the properties defined in the specified Federation Partner Profile.	
<a href="#">listFedPartnerProfiles</a>	Lists all of the existing Federation Partner Profiles.	
<a href="#">listFedPartnersForProfile</a>	Lists the partners bound to the specified Federation Partner Profile.	
<a href="#">getFedPartnerProfile</a>	Gets the ID of the Partner Profile bound to the specified partner.	
<a href="#">setFedPartnerProfile</a>	Sets the Federation Partner Profile ID for the specified partner.	

**Using WLST with SAML 1.1**

When an IDP partner is configured for SAML 1.1, the following URL is used by the SP to start the SSO process.

```
http://idphost:port/ssourl?TARGET=targeturl&providerid=http://spproviderid
```

By using these WLST commands, the URL can be populated with the applicable information.

Use this command...	To...	Use with WLST...
<code>idpinitiatedssoprovideridparam</code>	Value is used by the peer provider to identify the provider ID of the SP.	
<code>idpinitiatedssotargetparam</code>	Sets the target URL for the specified SP partner.	

The following SAML 1.1 configuration parameters are not exposed through the Oracle Access Management Console. The values of these parameters can be modified using WLST.

Use this command...	To...	Use with WLST...
<code>"deletePartnerProperty"</code>	Delete a partner property.	
<code>"getPartnerProperty"</code>	Retrieve a partner property.	
<code>"updatePartnerProperty"</code>	Update a partner property.	
<b>Subject Confirmation Check</b>		
<code>subjectconfirmationcheck</code>	Enables or Disables the subject confirmation data check in SAML assertion.	

### 3.2.1 configureFederationService

The `configureFederationService` command enables or disables the Federation Service AttributeRequester or AttributeResponder.

#### Description

Enable or disable Federation Service features.

#### Syntax

```
configureFederationService(<serviceType>,<enabled>)
```

Argument	Definition
<i>serviceType</i>	Takes as a value IDP, SP, AttributeResponder or AttributeRequester.
<i>enabled</i>	Takes as a value either true or false.

#### Example

```
configureFederationService("idp", "true")
```

```
configureFederationService("AttributeResponder", "true")
```

## 3.2.2 setFederationStore

The setFederationStore command enables and configures for the use of the federation store.

### Description

This will set the jndiname of the datastore to be used to store federation records and will set the store as a RDBMS.

### Syntax

```
setFederationStore (<enable>, <jndiname>)
```

Argument	Definition
<i>enable</i>	Enable or disable the Federation data store.
<i>jndiname</i>	Indicates the JNDI name of the datastore.

### Example

```
setFederationStore(enable="true", jndiname="jdbc/oamds")
```

## 3.2.3 configureIdPAuthnRequest

The configureIdPAuthnRequest command configures an IdP partner or an IdP partner profile for Force Authentication and/or IsPassive.

### Description

Configure an IdP partner or IdP partner profile for Force Authentication and/or IsPassive.

### Syntax

```
configureIdPAuthnRequest (<partner="">, <partnerProfile="">, <partnerType="">, <isPassive="false">, <forceAuthn="false">, <displayOnly="false">, <delete="false">)
```

Argument	Definition
<i>partner</i>	Indicates the IdP partner to be configured. partner and partnerProfile are exclusive, with one of the two required.
<i>partnerProfile</i>	Indicates the IdP partner profile to be configured. partner and partnerProfile are exclusive, with one of the two required.
<i>partnerType</i>	The type of partner (sp or idp).
<i>isPassive</i>	Indicates if the IdP partner or IdP partner profile should be configured, so that the Authn Request message sent to the IdP will indicate that the IdP should not interact with the user during Federation SSO. True indicates that the IdP should not interact with the user. Optional.
<i>forceAuthn</i>	Indicates if the IdP partner or IdP partner profile should be configured, so that the Authn Request message sent to the IdP will indicate that the IdP should challenge the user even if a valid session exists. True indicates that the user will be challenged. Optional.

Argument	Definition
<i>displayOnly</i>	Indicates whether or not this command should display the Is Passive and Force Authn settings. Default is false. Optional.
<i>delete</i>	Indicates whether or not this command should delete the Is Passive and Force Authn settings from the specified partner or partner profile. Default is false. Optional.

**Example**

```
configureIdPAuthnRequest (partner="acme", isPassive="false", forceAuthn="true")
```

## 3.2.4 configureFedSSOAuthz

The `configureFedSSOAuthz` command enables or disables Authorization for Federation SSO.

**Description**

Enables or disables Authorization for Federation SSO. By default, the authorization feature for Federation SSO will be turned off.

**Syntax**

```
configureFedSSOAuthz (enabled)
```

Argument	Definition
<i>enabled</i>	Takes as a value true or false.

**Example**

```
configureFedSSOAuthz ("true")
```

## 3.2.5 configureFedDigitalSignature

The `configureFedDigitalSignature` command configures the Hashing algorithm used in digital signatures.

**Description**

If the `displayOnly` and `delete` parameters are false, this command will set the algorithm.

**Syntax**

```
configureFedDigitalSignature (<partner="">,
 <partnerProfile="">, <partnerType="">, <default="false">,
 <algorithm="SHA-256">, <displayOnly="false">, <delete="false">)
```

Argument	Definition
<i>partner</i>	The ID of the SP partner profile
<i>partnerProfile</i>	The Federation Authentication Method to which the Access Manager Authentication Scheme will be mapped

Argument	Definition
<i>partnerType</i>	The Access Manager Authentication Scheme to which the Federated Authentication Method will be mapped
<i>default</i>	Optional. Boolean indicating whether or not the specified Access Manager Authentication Scheme should be used to challenge the user when the SP requests the Federated Authentication Method
<i>algorithm</i>	Optional. Indicates the authentication level to be used in the mapping in cases when the session authentication level is different from the authentication scheme level
<i>displayOnly</i>	Optional. The application domain in which the underlying policy components will be created
<i>delete</i>	Optional. The HostID used when creating the underlying resource policy object

### Example

```
configureFedDigitalSignature (default="true",
algorithm="SHA-256")
```

## 3.2.6 configureFedSignEncKey

The `configureFedSignEncKey` command configures the signing and/or encryption key alias to be used for digital signature and encryption operations.

### Description

Configure the signing and/or encryption key alias to be used for digital signature and encryption operations.

### Syntax

```
configureFedSignEncKey(<partner="">, <partnerProfile="">, <partnerType="">,
<default="false">, <signAlias="">, <encAlias="">, <displayOnly="false">,
<delete="false">
```

Argument	Definition
<i>partner</i>	Indicates the partner for which the signing and/or encryption key alias is to be configured. <i>partner</i> , <i>partnerProfile</i> and <i>default</i> parameters are exclusive, with one of the three required
<i>partnerProfile</i>	Indicates the partner profile for which the signing and/or encryption key alias is configured for. <i>partner</i> , <i>partnerProfile</i> and <i>default</i> parameters are exclusive, with one of the three required.
<i>partnerType</i>	Indicates the partner type for which the signing and/or encryption key alias is to be configured. Required when specifying <i>partner</i> or <i>partnerProfile</i> . Valid values are <i>sp</i> or <i>idp</i> .
<i>default</i>	Indicates the global default signing and/or encryption key alias to be configured. <i>partner</i> , <i>partnerProfile</i> and <i>default</i> parameters are exclusive, with one of the three required.
<i>signAlias</i>	The signing key alias. Required when setting the value.
<i>encAlias</i>	The encryption key alias. Required when setting the value.
<i>displayOnly</i>	Indicates whether or not this command should display the signing and encryption key aliases. Default is <i>false</i> . Optional.

Argument	Definition
<i>delete</i>	Indicates whether or not this command should delete the signing and/or encryption key alias from the specified partner or partner profile. Default is false. Optional.

### Example

```
configureFedSignEncKey(default="true", signAlias="osts_signing")
```

## 3.2.7 configureAttributeSharingSPPartnerNameIDMapping

The `configureAttributeSharingSPPartnerNameIDMapping` command configures the NameID to user store attribute mapping to be used during Attribute Sharing.

### Description

If `displayOnly` is true the command displays the NameID to userstore attribute mapping. Else if `delete` is true the command deletes the specified mapping. Else it sets the enabled flag to the given value and the sets a nameid to userstore attribute mapping.

### Syntax

```
configureAttributeSharingSPPartnerNameIDMapping(<partner="">,  
<partnerProfile="">, <enabled="true">, <nameidformat="">,  
<userStoreAttribute="">, <displayOnly="false">, <delete="false">)
```

Argument	Definition
<i>partner</i>	ID of the partner being configured. Optional. <code>partner</code> and <code>partnerProfile</code> parameters are exclusive, with one of the two required.
<i>partnerProfile</i>	Indicates the partner profile for which the mapping is being configured. Optional. <code>partner</code> and <code>partnerProfile</code> parameters are exclusive, with one of the two required
<i>enabled</i>	Boolean indicating if the nameID to userstore attribute mapping is enabled/disabled. Optional. Default value is true.



Argument	Definition
<i>nameidformat</i>	<p>The NameID format that is mapped to a userStoreAttribute. Optional. Needs to be specified for delete and create/update operations. If not specified for a display operation all the mappings for the specified partner or partnerprofile are displayed. Allowed NameID formats are:</p> <ul style="list-style-type: none"> <li>• orafed-emailaddress for urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</li> <li>• orafed-x509 for urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName</li> <li>• orafed-windowsnamequalifier for urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName</li> <li>• orafed-kerberos for urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos</li> <li>• orafed-transient for urn:oasis:names:tc:SAML:2.0:nameid-format:transient</li> <li>• orafed-persistent for urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</li> <li>• orafed-unspecified for urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</li> <li>• &lt;customnameidformaturi&gt; for a custom nameid format</li> </ul> <p>If the format is set to any other value, the Assertion will be populated with that value.</p>
<i>userStoreAttribute</i>	<p>The userstore attribute to which the specified NameID Format is mapped. Optional. Needs to be specified only for a create or update operation.</p>
<i>displayOnly</i>	<p>Indicates whether or not this command should display the NameID to userstore attribute mapping. Default is false. Optional. If set to true the mapping is displayed. If no NameID parameter is specified all the mappings are displayed.</p>
<i>delete</i>	<p>Indicates whether or not this command should delete NameID to userstore attribute mapping. Default is false. Optional.</p>

## Examples

```
configureAttributeSharingSPPartnerNameIDMapping (partner="acme", nameidformat="orafed-emailaddress", userStoreAttribute="mail")
```

```
configureAttributeSharingSPPartnerNameIDMapping (partnerProfile="saml20-idp-partner-profile", nameidformat="orafed-emailaddress", userStoreAttribute="mail")
```

```
configureAttributeSharingSPPartnerNameIDMapping (partner="acme")
```

```
configureAttributeSharingSPPartnerNameIDMapping (partner="acme", enabled="false")
```

```
configureAttributeSharingSPPartnerNameIDMapping (partner="acme", displayOnly="true")
```

```
configureAttributeSharingSPPartnerNameIDMapping (partner="acme", nameidformat="orafed-emailaddress", delete="true")
```

```
configureAttributeSharingSPPartnerNameIDMapping (partner="acme", nameidformat="orafed-emailaddress", displayOnly="true")
```

## 3.2.8 configureAttributeSharingIdPPartner

The `configureAttributeSharingIdPPartner` command configures the default attribute sharing `nameid` and `nameid format` for the IdP Partner.

### Description

Configures the default attribute sharing `nameid` and `nameid format` for the IdP Partner.

### Syntax

```
configureAttributeSharingIdPPartner(<partner="">, <partnerProfile="">, <nameidformat="">,
<nameidattribute="">)
```

Argument	Definition
<i>partner</i>	ID of the partner being configured. Optional. <code>partner</code> and <code>partnerProfile</code> parameters are exclusive, with one of the two required.
<i>partnerProfile</i>	Indicates the partner profile for which the mapping is being configured. Optional. <code>partner</code> and <code>partnerProfile</code> parameters are exclusive, with one of the two required
<i>nameidformat</i>	The NameID format that is mapped to a <code>userStoreAttribute</code> . Optional. Needs to be specified for delete and create/update operations. If not specified for a display operation all the mappings for the specified partner or <code>partnerprofile</code> are displayed. Allowed NameID formats are: <ul style="list-style-type: none"> <li>• <code>orafed-emailaddress</code> for <code>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</code></li> <li>• <code>orafed-x509</code> for <code>urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName</code></li> <li>• <code>orafed-windowsnamequalifier</code> for <code>urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName</code></li> <li>• <code>orafed-kerberos</code> for <code>urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos</code></li> <li>• <code>orafed-transient</code> for <code>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</code></li> <li>• <code>orafed-persistent</code> for <code>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</code></li> <li>• <code>orafed-unspecified</code> for <code>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</code></li> <li>• <code>orafed-custom</code> for a custom <code>nameid</code></li> </ul>
<i>nameIDAttribute</i>	The attribute in the userstore that should be used as the <code>nameid</code> . Optional.
<i>displayOnly</i>	Indicates whether or not this command should display the NameID to userstore attribute mapping. Default is false. Optional. If set to true the mapping is displayed. If no NameID parameter is specified all the mappings are displayed.

### Example

```
configureAttributeSharingIdPPartner(partner="acme",
nameidformat="orafed-emailaddress", nameidattribute="mail")
```

## 3.2.9 configureAttributeSharingUserDNToIdPPartnerMapping

The `configureAttributeSharingUserDNToIdPPartnerMapping` command configures Attribute Sharing DN to IdP Mappings.

### Description

If `displayOnly` is set to true the configuration is displayed. If `delete` is set to true the command deletes a specified mapping; otherwise, a mapping is created or updated.

### Syntax

```
configureAttributeSharingUserDNToIdPPartnerMapping(<dn="">,
<idp="">, <displayOnly="false">, <delete="false">)
```

Argument	Definition
<i>dn</i>	The DN string to map to the given IdP. Optional. Needs to be specified to delete a mapping and set a mapping. If specified for a display operation the mapping for this DN only is displayed.
<i>idp</i>	The partner ID of the IdP to use as Attribute Authority for the given DN. Optional. Needs to be specified only when creating or updating a mapping.
<i>displayOnly</i>	Indicates whether or not this command should display the NameID to userstore attribute mapping. Default is false. Optional. If set to true the mapping is displayed. If no NameID parameter is specified all the mappings are displayed.
<i>delete</i>	Indicates whether or not this command should delete NameID to userstore attribute mapping. Default is false. Optional.

### Examples

```
configureAttributeSharingUserDNToIdPPartnerMapping
(dn="dc=us,dc=oracle, dc=com", displayOnly="true")
```

```
configureAttributeSharingUserDNToIdPPartnerMapping(displayOnly="true")
```

```
configureAttributeSharingUserDNToIdPPartnerMapping(dn="dc=us,dc=oracle,dc=com",
delete="true")
```

```
configureAttributeSharingUserDNToIdPPartnerMapping(dn="dc=us,dc=oracle,dc=com",
idp="acme")
```

## 3.2.10 configureAttributeSharing

The `configureAttributeSharing` command configures the Attribute Sharing feature by setting a default attribute authority.

### Description

Configures the Attribute Sharing feature by setting a default attribute authority.

### Syntax

```
configureAttributeSharing(<defaultAttributeAuthority="">)
```

Argument	Definition
<i>defaultAttributeAuthority</i>	ID of the partner to use as the default Attribute Authority. Only used when this server is functioning in the SP mode.

**Example**

```
configureAttributeSharing (defaultAttributeAuthority="acme")
configureAttributeSharing ("acme")
```

### 3.2.11 removeAttributeSharingFromAuthnModule

The `removeAttributeSharingFromAuthnModule` command removes the Attribute Sharing plugin from the Authentication Module.

**Description**

Lists the Federated Authentication Method mappings for the specified Partner.

**Syntax**

```
removeAttributeSharingFromAuthnModule (<authnModule>, <stepName="">)
```

Argument	Definition
<i>authnModule</i>	The name of the <code>authnModule</code> from which to delete Attribute Sharing plugin.
<i>stepName</i>	The <code>stepName</code> of the Attribute Sharing plugin step to remove. Only needed if there is more than one attribute sharing step. Optional.

**Example**

```
removeAttributeSharingFromAuthnModule (authnModule="LDAPPlugin")
removeAttributeSharingFromAuthnModule (authnModule="LDAPPlugin",
stepName="FedAttributeSharing")
```

### 3.2.12 configureAttributeSharingPlugin

The `configureAttributeSharingPlugin` command lists the Federated Authentication Method mappings for a specific Partner Profile.

**Description**

Configures the input parameters of the Attribute Sharing plugin.

**Syntax**

```
configureAttributeSharingPlugin (<authnModule>, <stepName=None>,
<nameIDVariable=None>, <idpVariable=None>, <defaultIdP=None>,
<nameIDFormatVariable=None>, <defaultNameIDFormat=None>,
<requestedAttributes=None>)
```

Argument	Definition
<i>authnModule</i>	The name of the authnModule from which to delete Attribute Sharing plugin.
<i>stepName</i>	The stepName of the Attribute Sharing plugin step to remove. Only needed if there is more than one attribute sharing step. Optional.
<i>nameIDVariable</i>	The name of the variable in the session or context that contains the nameID of the user.
<i>idpVariable</i>	The name of the variable in the session or context that contains the idp name to which to send the attribute request.
<i>defaultIdP</i>	The name of the default IdP to send the attribute request to if no IdP can be determined from the session or context.
<i>nameIDFormatVariable</i>	The name of the variable in the session or context that contains the nameID format to use in the attribute request.
<i>defaultNameIDFormat</i>	The default NameID format to use if no nameid format could be determined from the session or context. Allowed NameID formats are: <ul style="list-style-type: none"> <li>• orafed-emailaddress for urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</li> <li>• orafed-x509 for urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName</li> <li>• orafed-windowsnamequalifier for urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName</li> <li>• orafed-kerberos for urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos</li> <li>• orafed-transient for urn:oasis:names:tc:SAML:2.0:nameid-format:transient</li> <li>• orafed-persistent for urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</li> <li>• orafed-unspecified for urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</li> </ul> If the format is set to any other value, the Assertion will be populated with that value.
<i>requestedAttributes</i>	The attributes to request from the IdP. This string is in the URL query string format.

### Example

```
configureAttributeSharingPlugin(authnModule="LDAPPlugin",
nameIDVariable="dn", idpVariable="attr.idpname", defaultIdP="acme",
nameIDFormatVariable="attr.nameidformat", defaultNameIDFormat="orafed-x509",
requestedAttributes="mail&accessAllowed=allowed")
```

## 3.2.13 insertAttributeSharingInToAuthnModule

The insertAttributeSharingInToAuthnModule command inserts the attribute sharing step into the Authentication Module flow.

### Description

Can also be used to remove the attribute sharing step from the Authentication Module flow.

## Syntax

```
insertAttributeSharingInToAuthnModule (<authnModule>,
<fromStep=None>, <fromCond=None>, <toStep=None>, <toCond=None>, <stepName=None>)
```

Argument	Definition
<i>authnModule</i>	The name of the authnModule into which the Attribute Sharing plugin is inserted.
<i>fromStep</i>	The name of the step after which the Attribute Sharing Step (or the step of given name) should be inserted.
<i>fromCond</i>	The condition under which the Attribute Sharing (or step of given name) is called after the fromStep. It has to be one of OnSuccess, OnFailure or OnError.
<i>toStep</i>	The name of the step to go to after the attribute sharing step (or step of given name).
<i>toCond</i>	The condition under which the toStep is called after the Attribute Sharing step (or step of given name).
<i>stepName</i>	The name of the step being added to the flow.

## Example

```
insertAttributeSharingInToAuthnModule (authnModule="LDAPPlugin",
fromStep="stepUA", fromCond="OnSuccess")
```

```
insertAttributeSharingInToAuthnModule (authnModule="LDAPPlugin", fromStep="stepUA",
fromCond="OnSuccess", stepName="success")
```

## 3.2.14 addSPPartnerAuthnMethod

The addSPPartnerAuthnMethod command defines a mapping between a Federated Authentication Method and an Access Manager Authentication Scheme for a specific SP Partner.

### Description

Maps a Federated Authentication Method to an Access Manager Authentication Scheme for an SP Partner.

### Syntax

```
addSPPartnerAuthnMethod (partner, authnMethod, authnScheme,
isDefault="true", authnLevel="-1", appDomain="IAM Suite",
hostID="IAMSuiteAgent", <authzPolicy="Protected Resource Policy">)
```

Argument	Definition
<i>partner</i>	The ID of the SP partner.
<i>authnMethod</i>	The Federation Authentication Method to which the Access Manager Authentication Scheme will be mapped

Argument	Definition
<i>authnScheme</i>	The Access Manager Authentication Scheme to which the Federated Authentication Method will be mapped
<i>isDefault</i>	Optional. Boolean indicating whether or not the specified Access Manager Authentication Scheme should be used to challenge the user when the SP requests the Federated Authentication Method
<i>authnLevel</i>	Optional. Indicates the authentication level to be used in the mapping in cases when the session authentication level is different from the authentication scheme level
<i>appDomain</i>	Optional. The application domain in which the underlying policy components will be created
<i>hostID</i>	Optional. The HostID used when creating the underlying resource policy object
<i>authzPolicy</i>	Optional. The Authorization Policy Name that will be used to protect underlying resource policy object being created.

### Example

```
addSPPartnerAuthnMethod("acmeSP",
    "urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport",
    "LDAPScheme")
```

## 3.2.15 addSPPartnerProfileAuthnMethod

The `addSPPartnerProfileAuthnMethod` command defines a mapping between a Federated Authentication Method to an Access Manager Authentication Scheme for a specific SP Partner Profile.

### Description

Maps a Federated Authentication Method to an Access Manager Authentication Scheme for an SP Partner Profile.

### Syntax

```
addSPPartnerProfileAuthnMethod(partnerProfile, authnMethod,
    authnScheme, isDefault="true", authnLevel="-1", appDomain="IAM Suite",
    hostID="IAMSuiteAgent", <authzPolicy="Protected Resource Policy">)
```

Argument	Definition
<i>partnerProfile</i>	The ID of the SP partner profile
<i>authnMethod</i>	The Federation Authentication Method to which the Access Manager Authentication Scheme will be mapped
<i>authnScheme</i>	The Access Manager Authentication Scheme to which the Federated Authentication Method will be mapped
<i>isDefault</i>	Optional. Boolean indicating whether or not the specified Access Manager Authentication Scheme should be used to challenge the user when the SP requests the Federated Authentication Method

Argument	Definition
<i>authnLevel</i>	Optional. Indicates the authentication level to be used in the mapping in cases when the session authentication level is different from the authentication scheme level
<i>appDomain</i>	Optional. The application domain in which the underlying policy components will be created
<i>hostID</i>	Optional. The HostID used when creating the underlying resource policy object
<i>authzPolicy</i>	Optional. The Authorization Policy Name that will be used to protect underlying resource policy object being created.

**Example**

```
addSPPartnerProfileAuthnMethod("saml20-sp-partner-profile",
    "urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport",
    "LDAPScheme")
```

## 3.2.16 addIdPPartnerAuthnMethod

The `addIdPPartnerAuthnMethod` command sets the Authentication Level to use when creating a session for a Federated Authentication Method for a specific IdP Partner.

**Description**

Defines the level to which to which users from this IdP partner are authenticated.

**Syntax**

```
addIdPPartnerAuthnMethod(partner, authnMethod, authnLevel)
```

Argument	Definition
<i>partner</i>	The ID of the SP partner profile
<i>authnMethod</i>	The Federated Authentication Method
<i>authnLevel</i>	The level to use to create the Access Manager user session during a Federation SSO flow for the specified Federated Authentication Method

**Example**

```
addIdPPartnerAuthnMethod("acmeIdP",
    "urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport", "1")
```

## 3.2.17 addIdPPartnerProfileAuthnMethod

The `addIdPPartnerProfileAuthnMethod` command sets the Authentication Level to use when creating a session for a Federated Authentication Method for a specific IdP Partner Profile.

**Description**

Defines the level to which to which users from this IdP partner profile are authenticated.



**Syntax**

```
addIdPPartnerProfileAuthnMethod(partnerProfile, authnMethod,
    authnLevel)
```

Argument	Definition
<i>partnerProfile</i>	The ID of the SP partner profile
<i>authnMethod</i>	The Federated Authentication Method
<i>authnLevel</i>	The level to use to create the Access Manager user session during a Federation SSO flow for the specified Federated Authentication Method

**Example**

```
addIdPPartnerProfileAuthnMethod("saml20-idp-partner-profile",
    "urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport", "1")
```

### 3.2.18 listPartnerAuthnMethods

The `listPartnerAuthnMethods` command lists the Federated Authentication Method mappings for a specific Partner.

**Description**

Lists the Federated Authentication Method mappings for the specified Partner.

**Syntax**

```
listPartnerAuthnMethods(partner, partnerType)
```

Argument	Definition
<i>partner</i>	The ID of the partner
<i>partnerType</i>	The type of the partner (SP or IdP)

**Example**

```
listPartnerAuthnMethods("acmeSP", "SP")
```

### 3.2.19 listPartnerProfileAuthnMethods

The `listPartnerProfileAuthnMethods` command lists the Federated Authentication Method mappings for a specific Partner Profile.

**Description**

Lists the Federated Authentication Method mappings for the specified Partner Profile.

**Syntax**

```
listPartnerProfileAuthnMethods(partnerProfile, partnerType)
```

Argument	Definition
<i>partnerProfile</i>	The ID of the partner profile
<i>partnerType</i>	The type of the partner (SP or IdP)

**Example**

```
listPartnerProfileAuthnMethods("saml20-sp-partner-profile", "SP")
```

## 3.2.20 removePartnerAuthnMethod

The `removePartnerAuthnMethod` command removes the mapping between a Federated Authentication Method and Access Manager Authentication Scheme for a specific Partner.

**Description**

Removes the mapping between a Federated Authentication Method and Access Manager Authentication Scheme for the specified Partner.

**Syntax**

```
removePartnerAuthnMethod(<partner>, <partnerType>, <authnMethod>)
```

Argument	Definition
<i>partner</i>	The ID of the partner
<i>partnerType</i>	The type of the partner (SP or IdP)
<i>authnMethod</i>	The Access Manager Authentication Scheme

**Example**

```
removePartnerAuthnMethod("acmeSP", "SP",  
"urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport")
```

## 3.2.21 removePartnerProfileAuthnMethod

The `removePartnerProfileAuthnMethod` command removes the mapping between a Federated Authentication Method and Access Manager Authentication Scheme for a specific Partner.

**Description**

Removes the mapping between a Federated Authentication Method and Access Manager Authentication Scheme for the specified Partner.

**Syntax**

```
removePartnerProfileAuthnMethod(<partnerProfile>,  
<partnerType>, <authnMethod>)
```

Argument	Definition
<i>partnerProfile</i>	The ID of the partner profile
<i>partnerType</i>	The type of the partner (SP or IdP)
<i>authnMethod</i>	The Federated Authentication Method

**Example**

```
removePartnerProfileAuthnMethod("saml20-sp-partner-profile",
"SP", "urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport")
```

### 3.2.22 setIdPPartnerRequestAuthnMethod

The setIdPPartnerRequestAuthnMethod command sets the Federated Authentication Method that will be requested during Federation SSO for a specific IdP Partner.

**Description**

Sets the Federated Authentication Method that will be requested during Federation SSO for the specified IdP Partner.

**Syntax**

```
setIdPPartnerRequestAuthnMethod(<partner>, <authnMethod>)
```

Argument	Definition
<i>partner</i>	The ID of the IdP partner
<i>authnMethod</i>	The Federated Authentication Method

**Example**

```
setIdPPartnerRequestAuthnMethod("acmeIdP",
"urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport")
```

### 3.2.23 setIdPPartnerProfileRequestAuthnMethod

The setIdPPartnerProfileRequestAuthnMethod command sets the Federated Authentication Method that will be requested during Federation SSO for a specific IdP Partner Profile.

**Description**

Sets the Federated Authentication Method that will be requested during Federation SSO for the specified IdP Partner Profile.

**Syntax**

```
setIdPPartnerProfileRequestAuthnMethod(<partnerProfile>,
<authnMethod>)
```

Argument	Definition
<i>partnerProfile</i>	The ID of the IdP partner profile
<i>authnMethod</i>	The Federated Authentication Method

### Example

```
setIdPPartnerProfileRequestAuthnMethod("saml20-idp-partner-profile",
    "urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport")
```

## 3.2.24 useProxiedFedAuthnMethod

The `useProxiedFedAuthnMethod` command configures the Identity Provider to use the proxied Federation Authentication Method when performing Federation SSO.

### Description

If the server acts as an SP with a remote IdP to authenticate the user, when acting as an Identity Provider in a different Federation SSO operation, the server can use the Federation Authentication Method sent by the remote Identity Provider. The server will send the proxied Federation Authentication Method for the list of specified Federation Authentication Schemes. The server will only send the proxied Federation Authentication Method if the Federation protocol used between the server and the Service Provider is the same Federation protocol as the one used between the server and the Identity Provider.

### Syntax

```
useProxiedFedAuthnMethod(<enabled="false">,
    <displayOnly="false">, <authnSchemeToAdd="">, <authnSchemeToRemove="">,
    <appDomain="IAM Suite">, <hostID="IAMSuiteAgent">,
    <authzPolicy="Protected Resource Policy">)
```

Argument	Definition
<i>enabled</i>	Indicates whether or not the proxied Federation Authentication Method should be used. Default is to disable the feature. Optional.
<i>displayOnly</i>	Indicates whether or not this command should display the list of Federation Schemes for which the server should send the proxied Federation Authentication Method. Default is false. Optional.
<i>authnSchemeToAdd</i>	The OAM Federation Authentication Scheme to be added to the list of schemes for which the server should send the proxied Federation Authentication Method. <code>authnSchemeToAdd</code> and <code>authnSchemeToRemove</code> parameters are exclusive.
<i>authnSchemeToRemove</i>	The OAM Federation Authentication Scheme to be removed from the list of schemes for which the server should send the proxied Federation Authentication Method. <code>authnSchemeToAdd</code> and <code>authnSchemeToRemove</code> parameters are exclusive.
<i>appDomain</i>	The application domain in which the underlying policy components will be created. Optional.
<i>hostID</i>	The HostID that will be used when creating the underlying resource policy object. Optional.

Argument	Definition
<i>authzPolicy</i>	Optional. The Authorization Policy Name that will be used to protect the underlying resource policy object being created.

**Example**

```
useProxiedFedAuthnMethod(enabled="true",
  authnSchemeToAdd="FederationScheme")
```

## 3.2.25 createFedPartnerProfileFrom

The `createFedPartnerProfileFrom` command creates a Federation Partner Profile based on the specified existing one.

**Description**

Creates a new partner profile based on the specified existing partner profile.

**Syntax**

```
createFedPartnerProfileFrom(<newPartnerProfile>,
  <existingPartnerProfile>)
```

Argument	Definition
<i>newPartnerProfile</i>	The ID of the new partner profile.
<i>existingPartnerProfile</i>	The ID of the existing partner profile

**Example**

```
createFedPartnerProfileFrom("newAcmeSPPProfile", "acmeSPPProfile")
```

## 3.2.26 deleteFedPartnerProfile

The `deleteFedPartnerProfile` command deletes the specified Federation Partner Profile.

**Description**

Removes the specified partner profile.

**Syntax**

```
deleteFedPartnerProfile(<PartnerProfile>)
```

Argument	Definition
<i>PartnerProfile</i>	The ID of the partner profile being deleted.

**Example**

```
deleteFedPartnerProfile("acmeSPPProfile")
```

## 3.2.27 displayFedPartnerProfile

The `displayFedPartnerProfile` command displays the properties defined in the specified Federation Partner Profile.

### Description

Displays the properties in the specified Federation Partner Profile.

### Syntax

```
displayFedPartnerProfile(<PartnerProfile>)
```

Argument	Definition
<i>PartnerProfile</i>	The ID of the partner profile.

### Example

```
displayFedPartnerProfile("saml20-idp-partner-profile")
```

## 3.2.28 listFedPartnerProfiles

The `listFedPartnerProfiles` command lists all of the existing Federation Partner Profiles.

### Description

Lists the existing Federation Partner Profiles.

### Syntax

```
listFedPartnerProfiles()
```

This command has no arguments.

### Example

```
listFedPartnerProfiles()
```

## 3.2.29 listFedPartnersForProfile

The `listFedPartnersForProfile` command lists the partners bound to the specified Federation Partner Profile.

### Description

Lists all the partners bound to the specified Federation Partner Profile.

### Syntax

```
listFedPartnersForProfile(<PartnerProfile>)
```

Argument	Definition
<i>PartnerProfile</i>	The ID of the partner profile.

**Example**

```
listFedPartnersForProfile("acmeSPProfile")
```

### 3.2.30 getFedPartnerProfile

The `getFedPartnerProfile` command retrieves the ID of the Partner Profile bound to the specified partner.

**Description**

Retrieves the ID of the Partner Profile bound to the specified partner.

**Syntax**

```
getFedPartnerProfile(<partner>, <partnerType>)
```

Argument	Definition
<i>partner</i>	The ID of the partner.
<i>partnerType</i>	The type of the partner (sp or idp).

**Example**

```
getFedPartnerProfile("acmeIDP", "idp")
```

### 3.2.31 setFedPartnerProfile

The `setFedPartnerProfile` command sets the Federation Partner Profile ID for the specified partner.

**Description**

Sets the partner profile for the specified partner profile based on the specified partner profile ID.

**Syntax**

```
setFedPartnerProfile(<partner>, <partnerType>, <partnerProfile>)
```

Argument	Definition
<i>partner</i>	The ID of the partner.
<i>partnerType</i>	The type of the partner (sp or idp).
<i>partnerProfile</i>	The ID of the partner profile.

**Example**

```
setFedPartnerProfile("acmeIDP", "idp",
    "saml20-idp-partner-profile")
```

## 3.2.32 idpinitiatedssoprovideridparam

The `idpinitiatedssoprovideridparam` command sets the value to identify the provider ID for the SP.

**Description**

The value held by `idpinitiatedssoprovideridparam` is used by the peer provider to identify the provider ID of the SP.

**Syntax**

```
updatePartnerProperty(partnerName, partnerType,
    "idpinitiatedssoprovideridparam", "providerid", "string")
```

Argument	Definition
<code>partnerName</code>	The ID of the partner
<code>partnerType</code>	Takes as a value either <code>idp</code> or <code>sp</code>
<code>propName</code>	Name of the property being configured or modified
<code>propValue</code>	The value of the property being configured. For an OIF peer IDP, the parameter name must be "providerid". Changing this property will change the parameter name used in the above URL.
<code>type</code>	The data type of the property value. Valid values are <code>string</code> , <code>long</code> , or <code>boolean</code> .

**Example**

```
updatePartnerProperty(partnerName, "idp",
    "idpinitiatedssoprovideridparam", "providerid", "string")
```

## 3.2.33 idpinitiatedssotargetparam

The `idpinitiatedssotargetparam` command sets the target URL for the specified SP partner.

**Description**

Identifies the target resource. The value held by `idpinitiatedssotargetparam` is used by the peer provider to identify the desired resource; `TARGET` in the case of Oracle Identity Federation.

**Syntax**

```
updatePartnerProperty(partnerName, partnerType,
    "idpinitiatedssotargetparam", "TARGET", "string")
```

Argument	Definition
<code>partnerName</code>	The ID of the partner
<code>partnerType</code>	Takes as a value either <code>idp</code> or <code>sp</code>
<code>propName</code>	Name of the property being configured or modified



Argument	Definition
<code>propValue</code>	The location of the resource. The default value is <code>TARGET</code> .
<code>type</code>	The data type of the property value. Valid values are <code>string</code> , <code>long</code> , or <code>boolean</code> .

**Example**

```
updatePartnerProperty(partnerName, "idp",
    "idpinitiatedssotargetparam", "TARGET", "string")
```

 **Note:**

A certificate can be included in a SAML 1.1 signature. By replacing the value of `<partnerName>` with the partner ID and including the `includecertinsignature` parameter, the certificate will be included with the signature. For example:

```
updatePartnerProperty("<partnerName>", "sp",
    "includecertinsignature", "true", "boolean")

getPartnerProperty("<partnerName>", "sp", "includecertinsignature")

deletePartnerProperty("<partnerName>", "sp",
    "includecertinsignature")
```

## 3.2.34 subjectconfirmationcheck

The `subjectconfirmationcheck` command enables or disables the Subject Confirmation Data check.

**Description**

Enable or disable the Subject Confirmation Data check in SAML assertion.

**Syntax**

```
updatePartnerProperty(partnerName,partnerType,propName,propValue,type)
```

Argument	Definition
<i>partnerName</i>	The ID of the partner to be updated.
<i>partnerType</i>	Specifies the partner type. Valid values are <code>sp</code> or <code>idp</code> .
<i>propName</i>	Set the property name as 'subjectconfirmationcheck'.
<i>propValue</i>	Specify the property value. Valid values are <code>true</code> or <code>false</code> .
<i>type</i>	Data type of the property. It can only be <code>boolean</code> .

### Example

```
updatePartnerProperty(partnerName="testIDP", partnerType="IDP",  
propName="subjectconfirmationcheck",  
propValue="true", type="boolean")
```

# 4

## Identity Directory Service WLST Commands

Use these WebLogic Scripting Tool (WLST) commands to manage Identity Directory Service.

The [Identity Directory Service Commands](#) section lists the Identity Directory Service WLST commands and contains links to the command reference details.

### 4.1 Identity Directory Service WLST Commands

This chapter describes the WLST commands for Identity Directory Service. Use the WLST commands listed in [Table 4-1](#) to manage Identity Directory Service entity attributes, entity definitions, relationships and default operational configurations.

**Table 4-1 WLST Identity Directory Service Commands**

Use this command...	To...	Use with WLST...
<a href="#">activateIDSCfgChanges</a>	Reload the Identity Directory Service configuration.	Online
<a href="#">addAttributeInEntityConfig</a>	Add a new attribute to the entity configuration.	Online
<a href="#">addAttributePropsInEntityConfig</a>	Add new properties for an attribute in an entity configuration.	Online
<a href="#">addAttributeRefForEntity</a>	Add a new attribute to the specified entity.	Online
<a href="#">addAttrrefPropsInEntityConfig</a>	Add new properties for an attribute reference in an entity configuration.	Online
<a href="#">addCommonPropertyForOperationConfig</a>	Add a new property for a specified operation configuration.	Online
<a href="#">addEntity</a>	Add a new entity to the entity configuration.	Online
<a href="#">addEntityProps</a>	Add new properties for an entity in an entity configuration.	Online
<a href="#">addEntityRelation</a>	Add a new entity relation to the entity configuration.	Online
<a href="#">addIdentityDirectoryService</a>	Add a new Identity Directory Service to the configuration.	Online
<a href="#">addOperationConfig</a>	Add a new operation configuration to the entity configuration.	Online
<a href="#">addPropertyForOperationConfig</a>	Add a new property to a specified operation configuration.	Online
<a href="#">deleteAttributeInEntityConfig</a>	Delete an attribute from an entity configuration.	Online
<a href="#">deleteAttributePropsInEntityConfig</a>	Delete attribute properties in an entity configuration.	Online
<a href="#">deleteAttrrefPropsInEntityConfig</a>	Delete attribute reference properties in an entity configuration.	Online
<a href="#">deleteEntity</a>	Delete an entity from an entity configuration.	Online
<a href="#">deleteEntityProps</a>	Delete entity properties in an entity configuration.	Online
<a href="#">deleteEntityRelation</a>	Delete the specified entity relation.	Online

**Table 4-1 (Cont.) WLST Identity Directory Service Commands**

Use this command...	To...	Use with WLST...
<code>deleteIdentityDirectoryService</code>	Delete the specified Identity Directory Service in the configuration.	Online
<code>deleteOperationConfig</code>	Delete operation configuration in an entity configuration.	Online
<code>listAllAttributeInEntityConfig</code>	List all attributes in the entity configuration.	Online
<code>listAllEntityInEntityConfig</code>	List all entities defined in the specified entity configuration.	Online
<code>listAllIdentityDirectoryService</code>	List all Identity Directory Services in the configuration.	Online
<code>removeAttributeRefForEntity</code>	Remove an attribute from the specified entity.	Online
<code>removeCommonPropertyForOperationConfig</code>	Remove a property for the specified operation configuration.	Online
<code>removePropertyForOperationConfig</code>	Remove a property for the specified operation configuration.	Online
<code>updateAttributeInEntityConfig</code>	Update attributes in an entity configuration.	Online
<code>updateAttributePropsInEntityConfig</code>	Update attribute properties in an entity configuration.	Online
<code>updateAttrrefPropsInEntityConfig</code>	Update attribute reference properties in an entity configuration.	Online
<code>updateEntity</code>	Update an entity's properties in an entity configuration.	Online
<code>updateEntityAttrs</code>	Update an entity's properties in an entity configuration.	Online
<code>updateEntityProps</code>	Update the entity properties in an entity configuration.	Online
<code>deleteAttributePropsInEntityConfig</code>	Delete the attribute properties in an entity configuration.	Online
<code>dumpConnectionPoolStatsForInMemoryConfig</code>	Dumps the LDAP connection pool statistics for the associated in-memory IDS configuration for the current JVM into a specified file.	Online
<code>dumpConnectionPoolStatsForAllInMemoryConfig</code>	Dumps the LDAP connection pool statistics for all in-memory IDS configuration for the current JVM into a specified file.	Online
<code>dumpConnectionPoolStatsForAllFileBasedConfig</code>	Dumps the LDAP connection pool statistics for all file-based IDS configuration for the current JVM into a specified file.	Online
<code>dumpConnectionPoolStatsForAllFileBasedConfig</code>	Dumps the LDAP connection pool statistics for all file-based IDS configuration for the current JVM into a specified file.	Online

### 4.1.1 activateIDSConfigChanges

Online command that reloads the configuration for Identity Directory Service.

#### Description

Reloads the Identity Directory Service configuration.

**Syntax**

```
activateIDSConfigChanges()
```

This command has no arguments.

**Example**

The following command reloads the Identity Directory Service configuration:

```
activateIDSConfigChanges()
```

## 4.1.2 addAttributeInEntityConfig

Online command that adds an attribute to the entity configuration.

**Description**

Adds a new attribute to the entity configuration.

**Syntax**

```
addAttributeInEntityConfig(name, datatype, description, readOnly, pwdAttr, appName)
```

**Table 4-2 addAttributeInEntityConfig Arguments**

Argument	Definition
<i>name</i>	Name of the attribute to be added.
<i>datatype</i>	The attribute's type is defined as one of the following: <ul style="list-style-type: none"> <li>• binary</li> <li>• boolean</li> <li>• datetime</li> <li>• double</li> <li>• integer</li> <li>• rfc822name</li> <li>• string</li> <li>• x500name</li> </ul>
<i>description</i>	Description of the attribute to be added.
<i>readOnly</i>	Flag to specify whether the attribute is read only or can be modified.
<i>pwdAttr</i>	Flag to specify whether the attribute defines a password or not.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds an attribute `commonname` of `userrole` entity:

```
addAttributeInEntityConfig('commonname','string','common name',false,false,'userrole')
```

## 4.1.3 addAttributePropsInEntityConfig

Online command that adds properties for an attribute in an entity configuration.

**Description**

Adds new properties for an attribute in an entity configuration.

**Syntax**

```
addAttributePropsInEntityConfig(name, propNames, propVals, appName)
```

**Table 4-3 addAttributePropsInEntityConfig Arguments**

Argument	Definition
<i>name</i>	Name of the attribute to be added.
<i>propNames</i>	List of property names separated by " ". The properties ( <i>propNames</i> and <i>propVals</i> ) are free key/value pairs. Applications can store any required metadata at the attribute level in these properties. The Identity Directory Service does not perform any validation for these property names and does not interpret or use these properties internally. For configuration attributes, however, the Identity Directory Service performs a schema check and interprets the configuration names and their values.
<i>propVals</i>	List of corresponding property values separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds an attribute `orgunit` of entity `userrole`:

```
addAttributePropsInEntityConfig('orgunit','labelname|multivalued','common name|true','userrole')
```

## 4.1.4 addAttributeRefForEntity

Online command that adds attribute to an entity.

**Description**

Adds a new attribute to the specified entity.

**Syntax**

```
addAttributeRefForEntity(name, attrRefName, attrRefFilter, attrRefDefaultFetch, appName)
```

**Table 4-4 addAttributeRefForEntity Arguments**

Argument	Definition
<i>name</i>	Name of the entity to which the attribute will be added.
<i>attrRefName</i>	Name of the attribute to be added to the entity.

**Table 4-4 (Cont.) addAttributeRefForEntity Arguments**

Argument	Definition
<i>attrRefFilter</i>	Type of filter to be used with the attribute, defined as one of the following: <ul style="list-style-type: none"> <li>• beginswith</li> <li>• contains</li> <li>• doesnotcontain</li> <li>• dynamic</li> <li>• endswith</li> <li>• equals</li> <li>• greaterequal</li> <li>• greaterthan</li> <li>• lessequal</li> <li>• lessthan</li> <li>• none</li> <li>• notequals</li> </ul>
<i>attrRefDefaultFetch</i>	Flag to specify whether the attribute is fetched by default.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds an attribute `User` to `userrole` entity:

```
addAttributeRefForEntity('User','givenname','none','true','userrole')
```

## 4.1.5 addAttrrefPropsInEntityConfig

Online command that adds property for an attribute reference.

**Description**

Adds new properties for an attribute reference in an entity configuration.

**Syntax**

```
addAttrrefPropsInEntityConfig(entityName, attrName, propNames, propVals, appName)
```

**Table 4-5 addAttrrefPropsInEntityConfig Arguments**

Argument	Definition
<i>entityName</i>	Name of the entity.
<i>attrName</i>	Name of the attribute reference.
<i>propNames</i>	List of property names separated by " ". The properties ( <i>propNames</i> and <i>propVals</i> ) are free key/value pairs. Applications can store any required metadata at the attribute level in these properties. The Identity Directory Service does not perform any validation for these property names and does not interpret or use these properties internally. For configuration attributes, however, the Identity Directory Service performs a schema check and interprets the configuration names and their values.

**Table 4-5 (Cont.) addAttrrefPropsInEntityConfig Arguments**

Argument	Definition
<i>propVals</i>	List of corresponding property values separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds a multivalued property `labelname` for `org` entity:

```
addAttrrefPropsInEntityConfig('org', 'orgunit', 'labelname|multivalued', 'common name|true', 'userrole')
```

## 4.1.6 addCommonPropertyForOperationConfig

Online command that adds a property for an operation configuration.

**Description**

Adds a new property for a specified operation configuration.

**Syntax**

```
addCommonPropertyForOperationConfig(entityName, propName, propValue, appName)
```

**Table 4-6 addCommonPropertyForOperationConfig Arguments**

Argument	Definition
<i>entityName</i>	Name of the entity.
<i>propName</i>	Name of the property to be added for this operation configuration.
<i>propValue</i>	Value of the property to be added for this operation configuration.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds a new property `member`:

```
addCommonPropertyForOperationConfig('groupmember.attr', 'member', 'userrole')
```

## 4.1.7 addEntity

Online command that adds an entity to the configuration.

**Description**

Adds a new entity to the entity configuration.

**Syntax**

```
addEntity(name, type, idAttr, create, modify, delete, search, attrRefNames, attrRefFilters, attrRefDefaultFetches, appName)
```



**Table 4-7 addEntity Arguments**

Argument	Definition
<i>name</i>	Name of the entity to which the attribute will be added.
<i>type</i>	Name of the attribute to be added to the entity.
<i>idAttr</i>	Identity attribute of the entity to be added.
<i>create</i>	Flag to specify the create is allowed.
<i>modify</i>	Flag to specify the modify is allowed.
<i>delete</i>	Flag to specify the delete is allowed.
<i>search</i>	Flag to specify the search is allowed.
<i>attrRefNames</i>	Array of attribute names.
<i>attrRefFilters</i>	An array of filter type values, defined as one of the following: <ul style="list-style-type: none"> <li>• beginswith</li> <li>• contains</li> <li>• doesnotcontain</li> <li>• dynamic</li> <li>• endswith</li> <li>• equals</li> <li>• greaterequal</li> <li>• greaterthan</li> <li>• lessequal</li> <li>• lessthan</li> <li>• none</li> <li>• notequals</li> </ul>
<i>attrRefDefaultFetches</i>	Array of boolean strings (true, false).
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds an attribute `group` to the `Group` entity.

```
addEntity('Group', 'group', 'commonname', true, true, true, true, 'name|commonname', 'none|none', 'true|false', 'userrole')
```

## 4.1.8 addEntityProps

Adds property for an entity.

**Description**

Online command that adds new properties for an entity in an entity configuration.

**Syntax**

```
addEntityProps(name, propNames, propVals, appName)
```

**Table 4-8 addEntityProps Arguments**

Argument	Definition
<i>name</i>	Name of the entity.
<i>propNames</i>	List of property names separated by " ".
<i>propValues</i>	List of corresponding property values separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds `inclobjclasses` and `exclobjclasses` properties:

```
addEntityProps('User','inclobjclasses|exclobjclasses','inetorgperson|
orclidxperson','userrole')
```

## 4.1.9 addEntityRelation

Online command that adds entity relation to an entity.

**Description**

Add a new entity relation to the entity configuration for the specified attributes.

**Syntax**

```
addEntityRelation(name, type, fromEntity, fromAttr, toEntity, toAttr, recursive, appName)
```

**Table 4-9 addEntityRelation Arguments**

Argument	Definition
<i>name</i>	Name of the relation between the entities for the given attributes.
<i>type</i>	Type of the entity relation ("ManyToMany", "ManyToOne", "OneToMany", "OneToOne").
<i>fromEntity</i>	Name of the from entity.
<i>fromAttr</i>	Name of the from attribute.
<i>toEntity</i>	Name of the to entity.
<i>toAttr</i>	Name of the to attribute.
<i>recursive</i>	Flag to set the entity relationship as recursive.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds the `manager` relation between the `manager` and `User` entities:

```
addEntityRelation('manager', 'ManyToOne', 'User', 'manager', 'User', 'principal', false,
'userrole')
```

## 4.1.10 addIdentityDirectoryService

Online command that adds an Identity Store Service.

**Description**

Adds a new IdentityStoreService to the Identity Directory Service configuration.

**Syntax**

```
addIdentityDirectoryService(name, description, propNames, propValues)
```

**Table 4-10 addIdentityDirectoryService Arguments**

Argument	Definition
<i>name</i>	Name of the IdentityStoreService to be added.
<i>description</i>	Description of the IdentityStoreService.
<i>propNames</i>	An array of property names to be added to the IdentityStoreService configuration.
<i>propValues</i>	An array of values to be defined for the property names added to the IdentityStoreService configuration.

**Example**

The following command adds the `userrole` IdentityStoreService:

```
addIdentityDirectoryService('userrole', 'user role', 'ovd.context|entity.config',
'default|userrole')
```

## 4.1.11 addOperationConfig

Online command that adds operation configuration to an entity.

**Description**

Adds a new operation configuration to the entity configuration.

**Syntax**

```
addOperationConfig(entityName, propNames, propValues, appName)
```

**Table 4-11 addOperationConfig Arguments**

Argument	Definition
<i>entityName</i>	Name of the entity to which the operation configuration will be added.
<i>propNames</i>	An array of property names to be added to the operation configuration.
<i>propValues</i>	An array of property values for the properties added to the operation configuration.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command adds the `User` entity to which the operation configuration will be added:

```
addOperationConfig('User', 'entity.searchbase', 'cn=users,dc=oracle,dc=com', 'userrole')
```

## 4.1.12 addPropertyForOperationConfig

Online command that adds a property to an operation configuration.

### Description

Adds a new property to a specified operation configuration.

### Syntax

```
addPropertyForOperationConfig(entityName, propName, propValue, appName)
```

**Table 4-12 addPropertyForOperationConfig Arguments**

Argument	Definition
<i>entityName</i>	Name of the entity to which the operation configuration will be added.
<i>propName</i>	A property name to be added to the operation configuration.
<i>propValue</i>	A value for the property added to the operation configuration.
<i>appName</i>	Name of the Identity Directory Service.

### Example

The following command adds the property to the operation configuration:

```
addPropertyForOperationConfig('User','entity.searchbase', 'cn=users,dc=oracle,dc=com',
'userrole')
```

## 4.1.13 deleteAttributeInEntityConfig

Online command that deletes attribute from an entity.

### Description

Deletes an attribute from an entity configuration.

### Syntax

```
deleteAttributeInEntityConfig(name, appName)
```

**Table 4-13 deleteAttributeInEntityConfig Arguments**

Argument	Definition
<i>name</i>	Name of the attribute to be deleted.
<i>appName</i>	Name of the Identity Directory Service.

### Example

The following command deletes the `commonname` attribute.

```
deleteAttributeInEntityConfig('commonname', 'userrole')
```

## 4.1.14 deleteAttributePropsInEntityConfig

Online command that deletes the properties of an attribute.

### Description

Deletes attribute properties in an entity configuration.

### Syntax

```
deleteAttributePropsInEntityConfig(name, propNames, appName)
```

**Table 4-14** deleteAttributePropsInEntityConfig Arguments

Argument	Definition
<i>name</i>	Name of the attribute.
<i>propNames</i>	List of property names separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

### Example

The following example deletes the property `labelname` from the `userrole` entity:

```
deleteAttributePropsInEntityConfig('orgunit','labelname|multivalued','userrole')
```

## 4.1.15 deleteAttrrefPropsInEntityConfig

Online command that deletes attribute reference properties in an entity.

### Description

Deletes one or more attribute reference properties in an entity configuration.

### Syntax

```
deleteAttrrefPropsInEntityConfig(entityName, attrName, propNames, appName)
```

**Table 4-15** deleteAttrrefPropsInEntityConfig Arguments

Argument	Definition
<i>entityName</i>	Name of the entity.
<i>attrName</i>	Name of the attribute reference.
<i>propNames</i>	List of property names to be deleted. If multiple properties are to be deleted, they should be separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

### Example

The following command deletes two properties from attribute reference `orgunit` of entity `org`:

```
deleteAttrrefPropsInEntityConfig('org', 'orgunit','labelname|multivalued','userrole')
```

## 4.1.16 deleteEntity

Online command that deletes an entity.

### Description

Deletes an entity from an entity configuration.

### Syntax

```
deleteEntity(name, appName)
```

**Table 4-16 deleteEntity Arguments**

Argument	Definition
<i>name</i>	Name of the entity to be deleted.
<i>appName</i>	Name of the Identity Directory Service.

### Example

The following command deletes the `User` entity.

```
deleteEntity('User', 'userrole')
```

## 4.1.17 deleteEntityProps

Online command that deletes the properties of an entity.

### Description

Deletes entity properties in an entity configuration.

### Syntax

```
deleteEntityProps(name, propNames, appName)
```

**Table 4-17 deleteEntityProps Arguments**

Argument	Definition
<i>name</i>	Name of the entity.
<i>propNames</i>	List of property names separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

### Example

The following command deletes the two properties `inclobjclasses` and `exclobjclasses` of `User` entity:

```
deleteEntityProps('User', 'inclobjclasses|exclobjclasses', 'userrole')
```

## 4.1.18 deleteEntityRelation

Online command that deletes the relationship between entities.

**Description**

Deletes the specified entity relation between entities for the given attributes.

**Syntax**

```
deleteEntityRelation(name, appName)
```

**Table 4-18 deleteEntityRelation Arguments**

Argument	Definition
<i>name</i>	Name of the relation between the entities for the given attributes.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command deletes the `manager` relation specified between entities:

```
deleteEntityRelation('manager', 'userrole')
```

## 4.1.19 deleteIdentityDirectoryService

Online command that deletes the specified IdentityStoreService.

**Description**

Deletes the specified IdentityStoreService in the Identity Directory Service configuration.

**Syntax**

```
deleteIdentityDirectoryService(name)
```

where *name* is the name of the IdentityStoreService configuration to be deleted.

**Example**

The following example deletes `ids1` IdentityStoreService configuration.

```
deleteIdentityDirectoryService('ids1')
```

## 4.1.20 deleteOperationConfig

Online command that deletes an operation configuration.

**Description**

Deletes an operation configuration in an entity configuration.

**Syntax**

```
deleteOperationConfig(entityName, appName)
```

**Table 4-19 deleteOperationConfig Arguments**

Argument	Definition
<i>entityName</i>	Name of the entity from which the operation configuration will be removed.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command deletes the operation configuration associated with entity `User` and application `userrole`:

```
deleteOperationConfig('User','userrole')
```

### 4.1.21 listAllAttributeInEntityConfig

Online command that lists all attributes.

**Description**

Lists all attributes in the entity configuration.

**Syntax**

```
listAllAttributeInEntityConfig(appName)
```

where *appName* is the name of the Identity Directory Service that contains the entity configuration from which the list of attributes is retrieved.

**Example**

The following command obtains the list of attributes from `userrole` entity:

```
listAllAttributeInEntityConfig('userrole')
```

### 4.1.22 listAllEntityInEntityConfig

Online command that lists all entities for an entity configuration.

**Description**

Lists all entities defined in the specified entity configuration.

**Syntax**

```
listAllEntityInEntityConfig(appName)
```

where *appName* is the name of the Identity Directory Service that contains the entity configuration from which the list of entities is retrieved.

**Example**

The following command obtains the list of entities associated with `userrole` entity:

```
listAllEntityInEntityConfig('userrole')
```



## 4.1.23 listAllIdentityDirectoryService

Online command that lists all IdentityStoreService for an Identity Directory Service configuration.

### Description

Lists all IdentityStoreService in Identity Directory Service configuration.

### Syntax

```
listAllIdentityDirectoryService()
```

This command has no arguments.

### Example

The following command lists all the IdentityStoreService for an Identity Directory Service configuration:

```
listAllIdentityDirectoryService()
```

## 4.1.24 removeAttributeRefForEntity

Online command that deletes an attribute from an entity.

### Description

Removes an attribute from the specified entity.

### Syntax

```
removeAttributeRefForEntity(name, attrRefName, appName)
```

**Table 4-20 removeAttributeRefForEntity Arguments**

Argument	Definition
<i>name</i>	Name of the entity from which the attribute will be removed.
<i>attrRefName</i>	The name of the attribute to be removed.
<i>appName</i>	Name of the Identity Directory Service.

### Example

The following command deletes the `givenname` attribute associated with `User` entity:

```
removeAttributeRefForEntity('User', 'givenname', 'userrole')
```

## 4.1.25 removeCommonPropertyForOperationConfig

Online command that deletes a property for an operation configuration.

### Description

Removes a property for the specified operation configuration.

**Syntax**

```
removeCommonPropertyForOperationConfig(entityName, propName, appName)
```

**Table 4-21** removeCommonPropertyForOperationConfig Arguments

Argument	Definition
<i>entityName</i>	Name of the entity.
<i>propName</i>	Name of property to be removed for this operation configuration.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command removes `groupmember.attr` property associated with `User` entity:

```
removeCommonPropertyForOperationConfig('User', 'groupmember.attr', 'userrole')
```

## 4.1.26 removePropertyForOperationConfig

Online command that removes a property for an operation configuration.

**Description**

Removes a property for the specified operation configuration.

**Syntax**

```
removePropertyForOperationConfig(entityName, propName, appName)
```

**Table 4-22** removePropertyForOperationConfig Arguments

Argument	Definition
<i>entityName</i>	Name of the entity from which the operation configuration will be removed.
<i>propName</i>	A property name to be removed from the operation configuration.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command removes `entity.searchbase` property associated with `User` entity:

```
removePropertyForOperationConfig('User', 'entity.searchbase', 'userrole')
```

## 4.1.27 updateAttributeInEntityConfig

Online command that updates an attribute for an entity configuration.

**Description**

Updates attributes in an entity configuration.

**Syntax**

```
updateAttributeInEntityConfig(name, attrNames, attrVals, appName)
```

**Table 4-23** updateAttributeInEntityConfig Arguments

Argument	Definition
<i>name</i>	Name of the entity attribute to be updated.
<i>attrNames</i>	List of configuration attribute names separated by " ". Valid configuration attribute names are: <ul style="list-style-type: none"> <li>• dataType</li> <li>• description</li> <li>• readOnly</li> <li>• pwdAttr</li> <li>• attrInUse</li> </ul>
<i>attrVals</i>	List of corresponding attribute values separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command updates the `commonname` attribute:

```
updateAttributeInEntityConfig('commonname', 'readOnly|pwdAttr|attrInUse', 'true|false|false', 'userrole')
```

## 4.1.28 updateAttributePropsInEntityConfig

Online command that updates the properties of an attribute for an entity.

**Description**

Updates attribute properties in an entity configuration.

**Syntax**

```
updateAttributePropsInEntityConfig(name, propNames, propVals, appName)
```

**Table 4-24** updateAttributePropsInEntityConfig Arguments

Argument	Definition
<i>name</i>	Name of the attribute to be updated.
<i>propNames</i>	List of property names separated by " ".
<i>propVals</i>	List of corresponding property values separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command updates the properties for `orgunit` attribute associated with `userrole` application:

```
updateAttributePropsInEntityConfig('orgunit', 'multivalued', 'multivalued', 'userrole')
```

## 4.1.29 updateAttrrefPropsInEntityConfig

Online command that updates attribute reference properties for an entity.

**Description**

Updates attribute reference properties in an entity configuration.

**Syntax**

```
updateAttrrefPropsInEntityConfig(entityName, attrName, propNames, propVals, appName)
```

**Table 4-25 updateAttrrefPropsInEntityConfig Arguments**

Argument	Definition
<i>entityName</i>	Name of the entity.
<i>attrName</i>	Name of the attribute reference.
<i>propNames</i>	List of property names separated by " ".
<i>propVals</i>	List of corresponding property values separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command updates the attribute reference properties:

```
updateAttrrefPropsInEntityConfig('org',
'orgunit','entity.searchbase','multivalued','userrole')
```

## 4.1.30 updateEntity

Online command that updates properties of an entity.

**Description**

Updates an entity's properties in an entity configuration.

**Syntax**

```
updateEntity(name, type, idAttr, create, modify, delete, search, appName)
```

**Table 4-26 updateEntity Arguments**

Argument	Definition
<i>name</i>	Name of the entity to be updated.
<i>type</i>	Type of the entity.
<i>idAttr</i>	Identity attribute of the entity.
<i>create</i>	Flag to specify the create is allowed.
<i>modify</i>	Flag to specify the modify is allowed.
<i>delete</i>	Flag to specify the delete is allowed.
<i>search</i>	Flag to specify the search is allowed.
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command updates the properties associated with `Group` entity:

```
updateEntity('Group','group','commonname',true,true,true,true,'userrole')
```

## 4.1.31 updateEntityAttrs

Online command that updates the configuration attributes for an entity.

**Description**

Updates the configuration attributes for an entity attribute.

**Syntax**

```
updateEntityAttrs(name, attrNames, attrVals, appName)
```

**Table 4-27 updateEntityAttrs Arguments**

Argument	Definition
<i>name</i>	Name of the entity attribute. To update the properties of an entity attribute, see <a href="#">updateAttributePropsInEntityConfig</a> .
<i>attrNames</i>	List of configuration attribute names. If multiple configuration attributes are to be updated, they should be separated by " ". Valid configuration attribute names are: <ul style="list-style-type: none"> <li>• <code>idAttr</code></li> <li>• <code>pwdAttr</code></li> <li>• <code>firstnameAttr</code></li> <li>• <code>lastnameAttr</code></li> <li>• <code>mailAttr</code></li> <li>• <code>displaynameAttr</code></li> <li>• <code>descriptionAttr</code></li> <li>• <code>challengeQnAttr</code></li> <li>• <code>challengeAnsAttr</code></li> <li>• <code>commonIdAttr</code>.</li> </ul>
<i>attrVals</i>	List of corresponding configuration attribute values separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command updates configuration attributes associated with `User` entity:

```
updateEntityAttrs('User','idAttr|firstnameAttr','uid|givenname','userrole')
```

## 4.1.32 updateEntityProps

Online command that updates the properties of an entity.

**Description**

Updates the entity properties in an entity configuration.

**Syntax**

```
updateEntityProps(name, propNames, propVals, appName)
```

**Table 4-28** updateEntityProps Arguments

Argument	Definition
<i>name</i>	Name of the attribute to be updated.
<i>propNames</i>	List of property names separated by " ".
<i>propVals</i>	List of corresponding property values separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command updates the properties associated with `User` entity:

```
updateEntityProps('User', 'inclobjclasses|exclobjclasses', 'inetorgperson|
orclidxperson', 'userrole')
```

### 4.1.33 deleteAttributePropsInEntityConfig

Online command that deletes the attribute properties in an entity configuration.

**Description**

Deletes the attribute properties in an entity configuration.

**Syntax**

```
deleteAttributePropsInEntityConfig(name, propNames, appName)
```

**Table 4-29** deleteAttributePropsInEntityConfig

Argument	Definition
<i>name</i>	Name of the attribute to be deleted.
<i>propNames</i>	List of property names separated by " ".
<i>appName</i>	Name of the Identity Directory Service.

**Example**

The following command deletes the attribute property, `orgunit` from the `userrole`.

```
deleteAttributePropsInEntityConfig('orgunit', 'labelname|validvalues', 'userrole')
```

### 4.1.34 dumpConnectionPoolStatsForInMemoryConfig

Online command that dumps the LDAP connection pool statistics for the associated in-memory IDS configuration for the current JVM on which WLS is configured into a specified file.

**Description**

Dumps the LDAP connection pool statistics for the associated in-memory IDS configuration for the current JVM on which WLS is configured into a specified file.

**Syntax**

```
dumpConnectionPoolStatsForInMemoryConfig(name, fileName)
```

**Table 4-30 dumpConnectionPoolStatsForInMemoryConfig**

Argument	Definition
<i>name</i>	Name of the in-memory IDS configuration.
<i>fileName</i>	Refers to the full path of the file.

**Example**

The following example dumps the connection pool statistics for the in-memory IDS configuration `ids1` into the specified file:

```
dumpConnectionPoolStatsForInMemoryConfig('ids1', '/tmp/dump.txt')
```

## 4.1.35 dumpConnectionPoolStatsForAllInMemoryConfig

Online command that dumps the LDAP connection pool statistics for all in-memory IDS configuration for the current JVM on which WLS is configured into a specified file.

**Description**

Dumps the LDAP connection pool statistics for all in-memory IDS configuration for the current JVM on which WLS is configured into a specified file.

**Syntax**

```
dumpConnectionPoolStatsForAllInMemoryConfig(fileName)
```

**Table 4-31 dumpConnectionPoolStatsForAllInMemoryConfig**

Argument	Definition
<i>fileName</i>	Refers to the full path of the file.

**Example**

The following example dumps LDAP connection pool statistics for all in-memory IDS configuration into the specified file:

```
dumpConnectionPoolStatsForAllInMemoryConfig('/tmp/dump.txt')
```

## 4.1.36 dumpConnectionPoolStatsForAllFileBasedConfig

Online command that dumps the LDAP connection pool statistics for all file-based IDS configuration for the current JVM on which WLS is configured into a specified file.

**Description**

Dumps the LDAP connection pool statistics for all file-based IDS configuration for the current JVM on which WLS is configured into a specified file.

**Syntax**

```
dumpConnectionPoolStatsForAllFileBasedConfig(name, fileName)
```

**Table 4-32 dumpConnectionPoolStatsForAllFileBasedConfig**

Argument	Definition
<i>name</i>	Name of the file-based IDS configuration.
<i>fileName</i>	Refers to the full path of the file.

**Example**

The following example dumps the connection pool statistics for ids file-based configuration into the specified file:

```
dumpConnectionPoolStatsForFileBasedConfig('ids1', '/tmp/dump.txt')
```

## 4.1.37 dumpConnectionPoolStatsForAllFileBasedConfig

Online command that dumps the LDAP connection pool statistics for all file-based IDS configuration in the current JVM on which WLS is configured into a specified file.

**Description**

Dumps the LDAP connection pool statistics for all file-based IDS configuration for the current JVM on which WLS is configured into a specified file.

**Syntax**

```
dumpConnectionPoolStatsForAllFileBasedConfig(fileName)
```

**Table 4-33 dumpConnectionPoolStatsForAllFileBasedConfig**

Argument	Definition
<i>fileName</i>	Refers to the full path of the file.

**Example**

The following example dumps the connection pool statistics all file-based IDS configuration into the specified file:

```
dumpConnectionPoolStatsForFileBasedConfig('/tmp/dump.txt')
```



# 5

## Library Oracle Virtual Directory WLST commands

Use these custom WebLogic Scripting Tool (WLST) commands to manage Library Oracle Virtual Directory.

The [Library Oracle Virtual Directory commands](#) section lists the Library Oracle Virtual Directory WLST commands and contains links to the command reference details.

### 5.1 Library Oracle Virtual Directory WLST Commands

This chapter describes the Library Oracle Virtual Directory (libOVD) WLST commands. Use the WLST commands listed in [Table 5-1](#) to manage a libOVD configuration associated with a specific Oracle Platform Security Services (OPSS) context.

**Table 5-1 WLST libOVD Commands**

Use this command...	To...	Use with WLST...
<a href="#">addDNAttribute</a>	Add an attribute to the DN attributes list for an existing adapter.	Online
<a href="#">activateLibOVDConfigChanges</a>	Reload the libOVD configuration.	Online
<a href="#">addAttributeExclusionRule</a>	Add a attribute exclusion rule.	Online
<a href="#">addAttributeRule</a>	Add a new attribute mapping rule.	Online
<a href="#">addDomainExclusionRule</a>	Add a domain exclusion rule.	Online
<a href="#">addDomainRule</a>	Add a new domain mapping rule.	Online
<a href="#">addJoinRule</a>	Add a join rule to an existing Join Adapter for a libOVD configuration.	Online
<a href="#">addLDAPHost</a>	Add a new remote host to an existing LDAP adapter.	Online
<a href="#">addMappingContext</a>	Create a new mapping context.	Online
<a href="#">addPlugin</a>	Add a plug-in to an existing adapter or at the global level.	Online
<a href="#">addPluginParam</a>	Add new parameter values to the existing adapter level plug-in or global plug-in.	Online
<a href="#">addToRequestControlExcludeList</a>	Add a control to the Request Control Exclude List for an existing LDAP adapter configuration.	Online
<a href="#">addToRequestControlIncludeList</a>	Add a control to the Request Control Include List for an existing LDAP adapter configuration.	Online
<a href="#">assignViewToAdapter</a>	Assign the given view to an adapter.	Online
<a href="#">createJoinAdapter</a>	Create a new Join Adapter for a libOVD configuration.	Online
<a href="#">createLDAPAdapter</a>	Create a new LDAP adapter for a libOVD configuration.	Online

Table 5-1 (Cont.) WLST libOVD Commands

Use this command...	To...	Use with WLST...
<a href="#">createLDAPAdapterWithDefaultPlugins</a>	Create a new LDAP adapter with default plug-ins based on the specified directory type.	Online
<a href="#">createView</a>	Create a new view.	Online
<a href="#">deleteAdapter</a>	Delete an existing adapter for a libOVD configuration.	Online
<a href="#">deleteAttributeExclusionRule</a>	Delete a attribute exclusion rule.	Online
<a href="#">deleteAttributeRule</a>	Delete a attribute mapping rule.	Online
<a href="#">deleteDomainExclusionRule</a>	Delete a domain exclusion rule.	Online
<a href="#">deleteDomainRule</a>	Delete a domain mapping rule.	Online
<a href="#">deleteMappingContext</a>	Delete the specified mapping context.	Online
<a href="#">deleteView</a>	Delete the specified view.	Online
<a href="#">getAdapterDetails</a>	Display the details of an existing adapter for a libOVD configuration.	Online
<a href="#">listAdapters</a>	List the name and type of all adapters that are configured for a libOVD configuration.	Online
<a href="#">listAllMappingContextIds</a>	List all the mapping contexts.	Online
<a href="#">listAttributeRules</a>	List all the attribute rules.	Online
<a href="#">listDomainRules</a>	List all the domain rules.	Online
<a href="#">listViews</a>	List all views	Online
<a href="#">modifyLDAPAdapter</a>	Modify the existing LDAP adapter configuration.	Online
<a href="#">modifySocketOptions</a>	Modify the socket options for an existing LDAP adapter configuration.	Online
<a href="#">removeAllRequestControlExcludeList</a>	Remove all controls from the Request Control Exclude List for an existing LDAP adapter configuration.	Online
<a href="#">removeAllRequestControlIncludeList</a>	Remove all controls from a Request Control Include List for an existing LDAP adapter configuration.	Online
<a href="#">removeDNAttribute</a>	Remove an attribute from the DN attributes list for an existing LDAP adapter configuration.	Online
<a href="#">removeFromRequestControlExcludeList</a>	Remove a control from the Request Control Exclude List for an existing LDAP adapter configuration.	Online
<a href="#">removeFromRequestControlIncludeList</a>	Removes a control from the Request Control Include List for an existing LDAP adapter configuration.	Online
<a href="#">removeJoinRule</a>	Remove a join rule from a Join Adapter configured for a libOVD configuration.	Online
<a href="#">removeLDAPHost</a>	Remove a remote host from an existing LDAP adapter configuration.	Online
<a href="#">removePlugin</a>	Remove a plug-in from an existing adapter or at the global level.	Online
<a href="#">removePluginParam</a>	Remove an existing parameter from a configured adapter level plug-in or global plug-in.	Online

**Table 5-1 (Cont.) WLST libOVD Commands**

Use this command...	To...	Use with WLST...
<a href="#">replacePluginParam</a>	Replace existing parameter values for an adapter level plug-in or global plug-in.	Online
<a href="#">unassignViewFromAdapter</a>	Un-assign a view from an adapter.	Online
<a href="#">listSSLStoreType</a>	List the type of SSL store in use for libOVD.	Online
<a href="#">enableKSSFForSSL</a>	Enable KSS for libOVD.	Online
<a href="#">enableJKSForSSL</a>	Enable JKS for libOVD.	Online
<a href="#">createKeyStoreAndEnableJKSForSSL</a>	Enable JKS for libOVD.	Online
<a href="#">importTrustedCertificateIntoSSLStore</a>	Import given trusted certificate into SSL store.	Online
<a href="#">migrateAllTrustedCertificatesFromJKSToKSS</a>	Migrate all trusted certificates from JKS to KSS store.	Online
<a href="#">migrateTrustedCertificatesFromJKSToKSS</a>	Migrate given trusted certificates from JKS to KSS store.	Online
<a href="#">changeLDAPHostPort</a>	Change given LDAP host and port in an existing LDAP adapter configuration to the new host and port.	Online
<a href="#">removeLDAPHostPort</a>	Remove a remote host and a port from an existing LDAP adapter configuration.	Online
<a href="#">setReadOnlyForLDAPHost</a>	Set the given host and port to read-only/writable in an existing LDAP adapter configuration.	Online
<a href="#">dumpLdapConnectionPoolStats</a>	Dumps the current connection pool statistics for an adapter to a file for the given JVM.	Online
<a href="#">addCipherSuite</a>	Adds cipher suite to an existing LDAP adapter configuration.	Online
<a href="#">removeCipherSuite</a>	Removes cipher suite to an existing LDAP adapter configuration.	Online

## 5.1.1 addDNAttribute

Online command that adds an attribute to the DN Attributes List.

### Description

Adds an attribute to the DN Attributes List for an existing adapter configured for the libOVD configuration associated with an OPSS context.

### Syntax

```
addDNAttribute(adapterName, attributeName, [contextName])
```

**Table 5-2 addDNAttribute Arguments**

Argument	Definition
<i>adapterName</i>	Name of the adapter to be updated.
<i>attributeName</i>	Name of the new DN attribute to be added.

**Table 5-2 (Cont.) addDNAttribute Arguments**

Argument	Definition
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is default.

**Example**

The following example adds `memberof` attribute to `ldap1` adapter:

```
addDNAttribute(adapterName='ldap1', attributeName='memberof', contextName='default')
```

## 5.1.2 activateLibOVDConfigChanges

Online command that reloads the libOVD configuration.

**Description**

Reloads the libOVD configuration associated with a specific OPSS context.

**Syntax**

```
activateLibOVDConfigChanges([contextName])
```

**Table 5-3 activateLibOVDConfigChanges Arguments**

Argument	Definition
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is default.

**Example**

The following command reloads the default libOVD configuration for a specified OPSS context:

```
activateLibOVDConfigChanges('default')
```

## 5.1.3 addAttributeExclusionRule

Online command that adds an attribute exclusion rule.

**Description**

Adds an attribute exclusion rule to the exclusion list.

**Syntax**

```
addAttributeExclusionRule(attribute, mappingContextId, [contextName])
```

**Table 5-4 addAttributeExclusionRule Arguments**

Argument	Definition
<i>attribute</i>	Name of the attribute to be added to the exclusion list.
<i>mappingContextId</i>	Name of the mapping context.

**Table 5-4 (Cont.) addAttributeExclusionRule Arguments**

Argument	Definition
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is default.

**Example**

The following command add `objectsid` to the exclusion list:

```
addAttributeExclusionRule('objectsid','userrole')
```

## 5.1.4 addAttributeRule

Online command that adds a new attribute mapping rule.

**Description**

Adds a new attribute mapping rule to the libOVD configuration associated with a specific OPSS context..

**Syntax**

```
addAttributeRule(srcAttrs, srcObjectClass, srcAttrType, dstAttr, dstObjectClass,
dstAttrType, mappingExpression, direction, mappingContextId, [contextName])
```

**Table 5-5 addAttributeRule Arguments**

Argument	Definition
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is default.

**Example**

The following command creates a mapping rule for the libOVD configuration. Here, the `lastname` is mapped to the `cn`.

```
addAttributeRule('lastname','','','sn','','','','Inbound','userrole')
```

## 5.1.5 addDomainExclusionRule

Online command that adds a domain exclusion rule.

**Description**

Adds a domain exclusion rule to the exclusion list.

**Syntax**

```
addDomainExclusionRule(domain, mappingContextId, [contextName])
```

**Table 5-6 addDomainExclusionRule Arguments**

Argument	Definition
<i>domain</i>	Distinguished name (DN) of the attribute to be added to the exclusion list.
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command adds `cn=group,dc=oracle,dc=com` to the exclusion list:

```
addDomainExclusionRule('cn=group,dc=oracle,dc=com','userrole')
```

## 5.1.6 addDomainRule

Online command that adds a new domain mapping rule.

**Description**

Adds a new domain mapping rule.

**Syntax**

```
addDomainRule(srcDomain, destDomain, domainConstructRule, mappingContextId,
[contextName])
```

**Table 5-7 addDomainRule Arguments**

Argument	Definition
<i>srcDomain</i>	Source domain.
<i>destDomain</i>	Destination domain
<i>domainConstructRule</i>	Name of the attribute to be added to the exclusion list.
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is default.

**Example**

The following command creates a domain mapping rule:

```
addDomainRule('dc=oracle,dc=com', 'dc=oracle,dc=com', '', 'defaultContext', 'default')
```

## 5.1.7 addJoinRule

Online command that adds a join rule to a Join Adapter.

**Description**

Adds a join rule to an existing Join Adapter for the libOVD configuration associated with the specified OPSS context.

## Syntax

```
addJoinRule(adapterName, secondary, condition, [joinerType], [contextName])
```

**Table 5-8 addJoinRule Arguments**

Argument	Definition
<i>adapterName</i>	Name of the Join Adapter to be modified.
<i>secondary</i>	Name of the adapter to join to.
<i>condition</i>	The attribute(s) to join on.
<i>joinerType</i>	Optional. Defines the type of Join. Values can be Simple (default), Conditional, OneToMany, or Shadow.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is default.

## Examples

The following commands create different join rules for an existing Join adapter:

```
addJoinRule('join1','secondaryldap','cn=cn', 'Simple', 'default')

addJoinRule('join1','secondaryldap','cn=cn', 'Conditional', 'default')

addJoinRule(adapterName='join1', secondary='LDAP3', condition='uid=cn',
JoinerType='OneToMany')

addJoinRule(adapterName='join1', secondary='LDAP2', condition='uid=cn',
contextName='myContext')
```

## 5.1.8 addLDAPHost

Online command that adds a new remote host.

### Description

Adds a new remote host (host and port) to an existing LDAP adapter. By default, the new host is configured in Read-Write mode with percentage set to 100.

### Syntax

```
addLDAPHost(adapterName, host, port, [contextName])
```

**Table 5-9 addLDAPHost Arguments**

Argument	Definition
<i>adapterName</i>	Name of the Join Adapter to be modified.
<i>host</i>	Remote LDAP host to which the LDAP adapter will communicate.
<i>port</i>	Remote LDAP host port.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following commands add a host and a port to an existing LDAP adapter:

```
addLDAPHost(adapterName='ldap1', host='myhost.example.com', port=389)

addLDAPHost('ldap1', 'myhost.example.com', '389', 'myContext')
```

## 5.1.9 addMappingContext

Online command that creates a new mapping context.

**Description**

Creates a new mapping context for the libOVD configuration associated with the specified OPSS context.

**Syntax**

```
addMappingContext(mappingContextId, [contextName])
```

**Table 5-10 addMappingContext Arguments**

Argument	Definition
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command creates a mapping context for the libOVD configuration:

```
addMappingContext('defaultContext', 'context')
```

## 5.1.10 addPlugin

Online command that adds a plug-in to an existing adapter or at the global level.

**Description**

Adds a plug-in to an existing adapter or at the global level. The "i"th key corresponds to "i"th value. The plug-in is added to default chain.

**Syntax**

```
addPlugin(pluginName, pluginClass, paramKeys, paramValues, [adapterName], [contextName])
```

**Table 5-11 addPlugin Arguments**

Argument	Definition
<i>pluginName</i>	Name of the plug-in to be created.
<i>pluginClass</i>	Class of the plug-in.
<i>paramKeys</i>	Init Param Keys separated by " ".



**Table 5-11 (Cont.) addPlugin Arguments**

Argument	Definition
<i>paramValues</i>	Init Param Values separated by " ".
<i>adapterName</i>	Optional. Name of the adapter to be modified. If not specified, the plug-in is added at the global level.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Examples

The following commands add a plug-in to an existing adapter:

```
wls:/mydomain/serverConfig> addPlugin(adapterName='ldap1',
pluginName='VirtualAttr',pluginClass='oracle.ods.virtualization.engine.chain.plugins.virt
ualattr.VirtualAttributePlugin', paramKeys='AddAttribute | MatchFilter | ContainerDN',
paramValues='cn=%uid% | objectclass=person | dc=oracle,dc=com')
```

```
wls:/mydomain/serverConfig>
addPlugin(pluginName='VirtualAttr',pluginClass='oracle.ods.virtualization.engine.chain.pl
ugins.virtualattr.VirtualAttributePlugin', paramKeys='AddAttribute | MatchFilter |
ContainerDN', paramValues='cn=%uid% | objectclass=person | dc=oracle,dc=com')
```

```
wls:/mydomain/serverConfig>
addPlugin(pluginName='DMSMetrics',pluginClass='oracle.ods.virtualization.engine.chain.plu
gins.DMSMetrics.MonitorPerformance',
paramKeys='None',paramValues='None',adapterName='ldap1',contextName='default')
```

## 5.1.11 addPluginParam

Online command that adds new parameter values to the existing adapter level plug-in or global plug-in.

### Description

Adds new parameter values to the existing adapter level plug-in or the global plug-in. If the parameter already exists, the new value is added to the existing set of values. The "i"th key corresponds to "i"th value.

### Syntax

```
addPluginParam(pluginName, paramKeys, paramValues, [adapterName], [contextName])
```

**Table 5-12 addPluginParam Arguments**

Argument	Definition
<i>pluginName</i>	Name of the plug-in to be modified.
<i>paramKeys</i>	Init Param Keys separated by " ".
<i>paramValues</i>	Init Param Values separated by " ".
<i>adapterName</i>	Optional Name of the adapter to be modified. If not specified, the global plug-in is modified.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

## Examples

The following commands add a new plug-in parameter for an existing plug-in:

```
wls:/mydomain/serverConfig> addPluginParam(adapterName='ldap1',
pluginName='VirtualAttr', paramKeys='ReplaceAttribute | MatchFilter',
paramValues='cn=%uid% | objectclass=person')
```

```
wls:/mydomain/serverConfig> addPluginParam(pluginName='VirtualAttr',
paramKeys='ReplaceAttribute | MatchFilter', par)
```

## 5.1.12 addToRequestControlExcludeList

Online command that adds a control to the Request Control Exclude List.

### Description

Adds a control to the Request Control Exclude List for an existing LDAP adapter configuration.

### Syntax

```
addToRequestControlExcludeList(adapterName, control, [contextName])
```

**Table 5-13** addToRequestControlExcludeList Arguments

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be modified.
<i>control</i>	LDAP control object identifier (OID).
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command adds 2.16.840.1.113894.1.8.31 control to ldap1 adapter's Request Control Exclude List:

```
addToRequestControlExcludeList(adapterName='ldap1', control='2.16.840.1.113894.1.8.31',
contextName='default')
```

## 5.1.13 addToRequestControlIncludeList

Online command that adds a control to the Request Control Include List.

### Description

Adds a control to the Request Control Include List for an existing LDAP adapter configuration.

### Syntax

```
addToRequestControlIncludeList(adapterName, control, [contextName])
```

**Table 5-14** addToRequestControlIncludeList Arguments

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be modified.

**Table 5-14 (Cont.) addToRequestControlIncludeList Arguments**

Argument	Definition
<i>control</i>	LDAP control object identifier (OID).
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command adds 2.16.840.1.113894.1.8.31 control to ldap1 adapter's Request Control Include List:

```
addToRequestControlIncludeList(adapterName='ldap1', control='2.16.840.1.113894.1.8.31',
contextName='default')
```

## 5.1.14 assignViewToAdapter

Online command that assigns a view to an LDAP adapter.

**Description**

Assigns a view to an LDAP adapter in the libOVD configuration associated with an OPSS context.

**Syntax**

```
assignViewToAdapter(viewName, adapterName, [contextName])
```

**Table 5-15 assignViewToAdapter Arguments**

Argument	Definition
<i>viewName</i>	Name of the view.
<i>adapterName</i>	Name of the LDAP adapter.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command assigns userView to ldap1 adapter:

```
assignViewToAdapter('userView','ldap1', 'default')
```

## 5.1.15 createJoinAdapter

Online command that creates a new join adapter.

**Description**

Creates a new join adapter for the libOVD configuration associated with an OPSS context.

**Syntax**

```
createJoinAdapter(adapterName, root, primaryAdapter, [bindAdapter],[contextName])
```

**Table 5-16 createJoinAdapter Arguments**

Argument	Definition
<i>adapterName</i>	Name of the Join Adapter to be created.
<i>primaryAdapter</i>	Specifies the identifier of the primary adapter, which is the adapter searched first in the join operation.
<i>root</i>	root
<i>bindAdapter</i>	Specifies identifier of the bind adapter(s), which are the adapter(s) whose proxy account is used to bind in the LDAP operation. By default, <i>primaryAdapter</i> is set as <i>bindAdapter</i> .
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Examples**

The following commands create a Join adapter:

```
createJoinAdapter('join1','dc=join','primaryldap','myldap', 'myContext')
createJoinAdapter(adapterName='join1', root='dc=join', primaryAdapter='myldap')
```

## 5.1.16 createLDAPAdapter

Online command that creates a new LDAP adapter.

**Description**

Creates a new LDAP adapter for the libOVD configuration associated with an OPSS context.

**Syntax**

```
createLDAPAdapter(adapterName, root, host, port, remoteBase, [isSecure], [bindDN],
[bindPasswd], [passCred], [contextName])
```

**Table 5-17 createLDAPAdapter Arguments**

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be created.
<i>root</i>	Virtual Namespace of the LDAP adapter.
<i>host</i>	Remote LDAP host with which the LDAP adapter will communicate.
<i>port</i>	Remote LDAP host port number.
<i>remoteBase</i>	Location in the remote DIT to which root corresponds.
<i>isSecure</i>	Optional. Boolean value that enables secure SSL/TLS connections to the remote hosts when set to <code>true</code> . The default is <code>false</code> .
<i>bindDN</i>	Optional. Proxy BindDN used to communicate with remote host. Default is "".
<i>bindPasswd</i>	Optional. Proxy BindPasswd used to communicate with the remote host. Default is "".

**Table 5-17 (Cont.) createLDAPAdapter Arguments**

Argument	Definition
<i>passCred</i>	Optional. Controls the credentials, if any, the libOVD configuration will pass to the back-end (remote host) LDAP server. Values can be Always (default), None, or BindOnly.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Examples

The following commands create an LDAP adapter:

```
createLDAPAdapter("testLDAP", "dc=us,dc=oracle,dc=com", "myhost.example.com", 3060,
"dc=uk,dc=oid", false, "cn=testuser", "welcome1", "Always", "myContext"
```

```
createLDAPAdapter(adapterName='ldap1', root='dc=com', host='myhost.example.com',
port=5566, remoteBase='dc=oid')
```

## 5.1.17 createLDAPAdapterWithDefaultPlugins

Online command that creates a new LDAP adapter.

### Description

Creates a new LDAP adapter with default plug-ins based on the directory type for the libOVD configuration associated with an OPSS context.

### Syntax

```
createLDAPAdapterWithDefaultPlugins(adapterName, directoryType, root, host, port,
remoteBase, [isSecure], [bindDN], [bindPasswd], [contextName])
```

**Table 5-18 createLDAPAdapterWithDefaultPlugins Arguments**

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be created.
<i>directoryType</i>	Directory type. The value can be one of the following directories: <ul style="list-style-type: none"> <li>• OID - Oracle Internet Directory</li> <li>• OUD - Oracle Unified Directory</li> <li>• SUNONE- Sun Java System Directory Server</li> <li>• OVD - Oracle Virtual Directory</li> <li>• ACTIVE_DIRECTORY - Microsoft Active Directory</li> <li>• EDIRECTORY - Novell eDirectory</li> <li>• OPEN_LDAP - Open LDAP</li> <li>• WLS_OVD - Oracle WebLogic Server OVD</li> <li>• TIVOLI - IBM Tivoli Directory Server</li> </ul>
<i>root</i>	Virtual Namespace of the LDAP adapter.
<i>host</i>	Remote LDAP host to which LDAP adapter should communicate.
<i>port</i>	Remote host port.
<i>remoteBase</i>	Location in the remote DIT to which the root corresponds.

**Table 5-18 (Cont.) createLDAPAdapterWithDefaultPlugins Arguments**

Argument	Definition
<i>isSecure</i>	Optional. Boolean value that enables secure SSL/TLS connections to the remote hosts when set to <code>true</code> . The Default is <code>false</code> .
<i>bindDN</i>	Optional. Proxy <code>BindDN</code> used to communicate with remote host. Default is <code>""</code> .
<i>bindPasswd</i>	Optional. Proxy <code>BindPasswd</code> used to communicate with the remote host. Default is <code>""</code> .
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is <code>"default"</code> .

### Examples

The following commands create an LDAP adapter with default plug-ins based on the directory type:

```
wls:/mydomain/serverConfig> createLDAPAdapterWithDefaultPlugins("testLDAP", "OID",
"dc=us,dc=oracle,dc=com", "myhost.example.domain.com", 3060, "dc=uk,dc=oid", false,
"cn=testuser", "welcome1", "myContext")
```

```
wls:/mydomain/serverConfig> createLDAPAdapterWithDefaultPlugins(adapterName='ldap1',
directoryType="OID", root='dc=com', host='myhost.example.domain.com', port=5566,
remoteBase='dc=oid',bindDN="cn=testuser",bindPasswd="welcome1",contextName='default')
```

## 5.1.18 createView

Online command that creates a new view.

### Description

Creates a new view for the libOVD configuration associated with an OPSS context.

### Syntax

```
createView(viewName, [contextName])
```

**Table 5-19 createView Arguments**

Argument	Definition
<i>viewName</i>	Name of the new view.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is <code>"default"</code> .

### Example

The following command create a view named, `userView`:

```
createView('userView','default')
```

## 5.1.19 deleteAdapter

Online command that deletes an existing adapter.

**Description**

Deletes an existing adapter for the libOVD configuration associated with an OPSS context.

**Syntax**

```
deleteAdapter(adapterName, [contextName])
```

**Table 5-20 deleteAdapter Arguments**

Argument	Definition
<i>adapterName</i>	Name of the Join Adapter to be deleted.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command deletes `join1` adapter:

```
deleteAdapter(adapterName='join1') deleteAdapter('join1', 'default')
```

## 5.1.20 deleteAttributeExclusionRule

Online command that deletes an attribute exclusion rule.

**Description**

Deletes an attribute exclusion rule for the libOVD configuration associated with an OPSS context.

**Syntax**

```
deleteAttributeExclusionRule(attribute, mappingContextId, [contextName])
```

**Table 5-21 deleteAttributeExclusionRule Arguments**

Argument	Definition
<i>attribute</i>	Name of the attribute to be removed from the exclusion list.
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command deletes the `objectsid` attribute exclusion rule for the associated libOVD configuration:

```
deleteAttributeExclusionRule('objectsid', 'userrole')
```

## 5.1.21 deleteAttributeRule

Online command that delete an attribute mapping rule.

**Description**

Deletes an attribute mapping rule for the libOVD configuration associated with an OPSS context.

**Syntax**

```
deleteAttributeRule(srcAttrs, dstAttr, mappingContextId, [contextName])
```

**Table 5-22 deleteEntityRelation Arguments**

Argument	Definition
<i>srcAttrs</i>	Source attributes.
<i>dstAttr</i>	Destination attribute.
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command deletes the `lastname` attribute mapping rule from `cn`:

```
deleteAttributeRule('lastname','sn')
```

## 5.1.22 deleteDomainExclusionRule

Online command that deletes a domain exclusion rule.

**Description**

Deletes a domain exclusion rule for the libOVD configuration associated with an OPSS context.

**Syntax**

```
deleteDomainExclusionRule(domain, mappingContextId, [contextName])
```

**Table 5-23 deleteEntityRelation Arguments**

Argument	Definition
<i>domain</i>	Distinguished Name of the container to be removed from the exclusion list.
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command deletes `'cn=group,dc=oracle,dc=com'` domain exclusion rule:

```
deleteDomainExclusionRule('cn=group,dc=oracle,dc=com')
```



## 5.1.23 deleteDomainRule

Online command that deletes a domain mapping rule.

### Description

Deletes a domain mapping rule for the libOVD configuration associated with an OPSS context.

### Syntax

```
deleteDomainRule(srcDomain, destDomain, mappingContextId, [contextName])
```

**Table 5-24 deleteDomainRule Arguments**

Argument	Definition
<i>srcDomain</i>	Source domain.
<i>destDomain</i>	Destination domain.
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command deletes 'dc=oracle,dc=com' domain mapping rule:

```
deleteDomainRule('dc=oracle,dc=com','dc=oracle,dc=com','userrole')
```

## 5.1.24 deleteMappingContext

Online command that deletes a mapping context.

### Description

Deletes the specified mapping context for the libOVD configuration associated with an OPSS context.

### Syntax

```
deleteMappingContext(mappingContextId, [contextName])
```

**Table 5-25 deleteMappingContext Arguments**

Argument	Definition
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command deletes a mapping context for a libOVD configuration:

```
deleteMappingContext('defaultContext','context')
```

## 5.1.25 deleteView

Online command that deletes a view.

### Description

Deletes a view for the libOVD configuration associated with an OPSS context.

### Syntax

```
createView(viewName, [contextName])
```

**Table 5-26 createView Arguments**

Argument	Definition
<i>viewName</i>	Name of the view to delete.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command deletes `userView` view:

```
deleteView('userView','default')
```

## 5.1.26 getAdapterDetails

Online command that displays the details of an existing adapter.

### Description

Displays the details of an existing adapter configured for the libOVD configuration associated with an OPSS context.

### Syntax

```
getAdapterDetails(adapterName, [contextName])
```

**Table 5-27 getAdapterDetails Arguments**

Argument	Definition
<i>adapterName</i>	Name of the adapter that contains the details to be displayed.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Examples

The following commands display the details of `ldap1` and `join1` adapter respectively:

```
getAdapterDetails(adapterName='ldap1', contextName='default')
```

```
getAdapterDetails(adapterName='join1')
```

## 5.1.27 listAdapters

Online command that lists the name and type of all adapters.

### Description

Lists the name and type of all adapters that are configured for the libOVD configuration associated with an OPSS context.

### Syntax

```
listAdapters([contextName])
```

**Table 5-28 listAdapters Arguments**

Argument	Definition
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command displays the name and type of all adapters configured for a libOVD configuration:

```
listAdapters()  
  
listAdapters(contextName='myContext')
```

## 5.1.28 listAllMappingContextIds

Online command that lists all mapping contexts.

### Description

Lists the mapping contexts associated with the specified OPSS context.

### Syntax

```
listAllMappingContextIds([contextName])
```

**Table 5-29 listAllMappingContextIds Arguments**

Argument	Definition
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command lists all the mapping contexts:

```
listAllMappingContextIds('default')
```

## 5.1.29 listAttributeRules

Online command that lists all the attribute rules.

### Description

List all the attribute rules in the format *SOURCE\_ATTRIBUTE:DESTINATION\_ATTRIBUTE:DIRECTION*.

### Syntax

```
listAttributeRules(mappingContextId, [contextName])
```

**Table 5-30 listAttributeRules Arguments**

Argument	Definition
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command lists all the attribute rules:

```
listAttributeRules('defaultContext','default')
```

## 5.1.30 listDomainRules

Online command that lists all domain rules.

### Description

Lists all the domain rules in the format of *SOURCE\_DOMAIN:DESTINATION\_DOMAIN*.

### Syntax

```
listDomainRules(mappingContextId, [contextName])
```

**Table 5-31 listDomainRules Arguments**

Argument	Definition
<i>mappingContextId</i>	Name of the mapping context.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command lists all domain rules:

```
listDomainRules('defaultContext','default')
```

## 5.1.31 listViews

Online command that lists all views

### Description

Lists all views for a libOVD configuration associated with an OPSS context.

## Syntax

```
listViews ([contextName])
```

**Table 5-32 listViews Arguments**

Argument	Definition
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

## Example

The following command lists all views:

```
listViews ('default')
```

## 5.1.32 modifyLDAPAdapter

Modifies parameters in an LDAP adapter.

### Description

Modifies the following LDAP adapter parameters:

- Remote Base
- Root
- Secure
- BindDN
- BindPassword
- PassCredentials
- MaxPoolSize
- MaxPoolChecks
- MaxPoolWait
- InitialPoolSize
- PoolCleanupInterval
- MaxPoolConnectionIdleTime
- Active
- PingProtocol
- PingBindDN
- PingBindPassword
- PageSize
- HeartBeatInterval
- OperationTimeout
- SearchCountLimit
- Visible

- Critical
- InclusionFilter
- ExclusionFilter
- DNPattern
- RequestControlAllowServerSupported
- MaxPoolConnectionReuseTime
- ConnectTimeout
- PoolConnectionReclaimTime
- Protocols

### Syntax

```
modifyLDAPAdapter(adapterName, attribute, value, [contextName])
```

**Table 5-33 modifyLDAPAdapter Arguments**

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be modified.
<i>attribute</i>	Name of the attribute to be modified.
<i>value</i>	New value for the attribute.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Examples

The following examples illustrate how to set attributes in `ldap1`:

```
modifyLDAPAdapter(adapterName='ldap1', attribute='Root', value='dc=us, dc=oracle, dc=com', contextName='mydefault')
```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='RemoteBase', value='dc=org', contextName='mydefault')
```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='PassCredentials', value='BindOnly', contextName='mydefault')
```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='BindDN', value='cn=proxyuser,dc=com', contextName='mydefault')
```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='BindPassword', value='testwelcome123', contextName='mydefault')
```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='Secure', value=true, contextName='mydefault')
```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='MaxPoolSize', value=500, contextName='mydefault')
```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='MaxPoolChecks', value=10, contextName='mydefault')
```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='MaxPoolWait', value=120000, contextName='mydefault') [value is in milliseconds]
```

```

modifyLDAPAdapter(adapterName='ldap1', attribute='InitialPoolSize', value=10,
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PoolCleanupInterval', value=300,
contextName='mydefault') [value is in seconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='MaxPoolConnectionIdleTime', value=300,
contextName='mydefault') [value is in seconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='Active', value=false,
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PingProtocol', value='LDAP',
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PingBindDN', value='cn=proxyuser',
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PingBindPassword', value='welcome1',
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PageSize', value=500,
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='HeartBeatInterval', value=120,
contextName='mydefault') [value is in seconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='OperationTimeout', value=120000,
contextName='mydefault') [value is in milliseconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='SearchCountLimit', value=100,
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='Visible', value='Yes',
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='Critical', value='false',
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='InclusionFilter',
value='objectclass=inetorgperson#base', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='ExclusionFilter',
value='uniquemember=*#base', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='DNPattern', value='(.*cn=[a-z0-9]*$',
contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='RequestControlAllowServerSupported',
value=false, contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='MaxPoolConnectionReuseTime',
value=3600, contextName='mydefault') [value is in seconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='ConnectTimeout', value=10000,
contextName='mydefault') [value is in milli seconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='PoolConnectionReclaimTime', value=180,
contextName='mydefault')

```

```
modifyLDAPAdapter(adapterName='ldap1', attribute='Protocols', value='TLSv1.2',
contextName='mydefault')
```

## 5.1.33 modifySocketOptions

Online command that modifies socket options.

### Description

Modifies socket options for an existing LDAP adapter configuration.

### Syntax

```
modifySocketOptions(adapterName, reuseAddress, keepAlive, tcpNoDelay, readTimeout,
[contextName])
```

**Table 5-34 modifySocketOptions Arguments**

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be modified.
<i>reuseAddress</i>	Value of <i>reuseAddress</i> .
<i>keepAlive</i>	Value of <i>keepAlive</i> .
<i>tcpNoDelay</i>	Value of <i>tcpNoDelay</i> .
<i>readTimeout</i>	Value of <i>readTimeout</i> in seconds.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command modifies the socket option for `ldap1` adapter:

```
modifySocketOptions(adapterName='ldap1', reuseAddress=true, keepAlive=true,
tcpNoDelay=true, readTimeout=180000, contextName='default')
```

## 5.1.34 removeAllRequestControlExcludeList

Online command that removes all controls from the Request Control Exclude List.

### Description

Removes all controls from the Request Control Exclude List for an existing LDAP adapter configuration.

### Syntax

```
removeAllRequestControlExcludeList(adapterName, [contextName])
```

**Table 5-35 removeAllRequestControlExcludeList Arguments**

Argument	Definition
<i>adapterName</i>	Name of the adapter to be updated.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".



**Example**

The following command removes all controls from `ldap1` adapter's Request Control Exclude List:

```
removeAllRequestControlExcludeList(adapterName='ldap1', contextName='default')
```

## 5.1.35 removeAllRequestControlIncludeList

Online command that removes all controls from the Request Control Include List.

**Description**

Removes all controls from the Request Control Include List for an existing LDAP adapter configuration.

**Syntax**

```
removeAllRequestControlIncludeList(adapterName, [contextName])
```

**Table 5-36** removeAllRequestControlIncludeList Arguments

Argument	Definition
<i>adapterName</i>	Name of the adapter to be updated.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command removes all controls from `ldap1` adapter's Request Control Include List:

```
removeAllRequestControlIncludeList(adapterName='ldap1', contextName='default')
```

## 5.1.36 removeDNAttribute

Online command that removes a attribute from the DN Attributes List.

**Description**

Removes a attribute from the DN Attributes List for an existing adapter that is configured for the libOVD associated with an OPSS context.

**Syntax**

```
removeDNAttribute(adapterName attributeName, [contextName])
```

**Table 5-37** removeDNAttribute Arguments

Argument	Definition
<i>adapterName</i>	Name of the adapter to be updated.
<i>attributeName</i>	Name of the new DN attribute to be removed.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command removes `memberof` attribute from `ldap1` adapter's attribute list:

```
removeDNAttribute(adapterName='ldap1', attributeName='memberof', contextName='default')
```

## 5.1.37 removeFromRequestControlExcludeList

Online command that removes a control from the Request Control Exclude List.

**Description**

Removes a control from the Request Control Exclude List for an existing LDAP adapter configuration.

**Syntax**

```
removeFromRequestControlExcludeList(adapterName, control, [contextName])
```

**Table 5-38** `removeFromRequestControlExcludeList` Arguments

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be modified.
<i>control</i>	LDAP control object identifier (OID).
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command removes `2.16.840.1.113894.1.8.31` control from `ldap1` adapter's Request Control Exclude List:

```
removeFromRequestControlExcludeList(adapterName='ldap1',
control='2.16.840.1.113894.1.8.31', contextName='default')
```

## 5.1.38 removeFromRequestControlIncludeList

Online command that removes a control from the Request Control Include List.

**Description**

Removes a control from the Request Control Include List for an existing LDAP adapter configuration.

**Syntax**

```
removeFromRequestControlIncludeList(adapterName, control, [contextName])
```

**Table 5-39** `removeFromRequestControlIncludeList` Arguments

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be modified.
<i>control</i>	LDAP control object identifier (OID).

**Table 5-39 (Cont.) removeFromRequestControlIncludeList Arguments**

Argument	Definition
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command removes 2.16.840.1.113894.1.8.31 control from ldap1 adapter's Request Control Include List:

```
removeFromRequestControlIncludeList(adapterName='ldap1',
control='2.16.840.1.113894.1.8.31', contextName='default')
```

## 5.1.39 removeJoinRule

Online command that removes a join rule from a Join Adapter.

**Description**

Removes a join rule from a Join Adapter configured for the libOVD configuration associated with the specified OPSS context.

**Syntax**

```
removeJoinRule(adapterName, secondary, [contextName])
```

**Table 5-40 removeJoinRule Arguments**

Argument	Definition
<i>adapterName</i>	Name of the Join Adapter to be modified.
<i>secondary</i>	The join rules corresponding to this secondary adapter are removed from the Join Adapter.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Examples**

The following command removes 2.16.840.1.113894.1.8.31 control from ldap1 adapter's Request Control Include List:

```
removeJoinRule('join1','secondaryldap1', 'default')
removeJoinRule(adapterName='join1', secondary='LDAP3')
```

## 5.1.40 removeLDAPHost

Online command that removes a remote host from an existing LDAP adapter.

**Description**

Removes a remote host (host:port) from an existing LDAP adapter.

**Syntax**

```
removeLDAPHost(adapterName, host, [contextName])
```

**Table 5-41 removeLDAPHost Arguments**

Argument	Definition
<i>adapterName</i>	Name of the LDAP adapter to be modified.
<i>host</i>	Location of a remote LDAP host with which the LDAP adapter will communicate.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following command removes the host and port from `ldap1` adapter:

```
removeLDAPHost(adapterName='ldap1', host='myhost.example.com')
removeLDAPHost('ldap1', 'myhost.example.com', 'myContext')
```

## 5.1.41 removePlugin

Online command that removes a plug-in from an existing adapter.

**Description**

Removes a plug-in from an existing adapter or at the global level.

**Syntax**

```
removePlugin(pluginName, [adapterName], [contextName])
```

**Table 5-42 removePlugin Arguments**

Argument	Definition
<i>pluginName</i>	Name of the plug-in to be removed.
<i>adapterName</i>	Optional. Name of the adapter to be modified. If not specified, the global plug-in is removed.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

**Example**

The following commands remove `VirtualAttr` plug-in from `ldap1` adapter:

```
removePlugin(adapterName='ldap1', pluginName='VirtualAttr')
removePlugin(pluginName='VirtualAttr')
```

## 5.1.42 removePluginParam

Online command that removes an existing parameter from a configured adapter level plug-in.

### Description

Removes an existing parameter from a configured adapter level plug-in or a global plug-in. This command removes all values of a particular parameter from the plug-in.

### Syntax

```
removePluginParam(pluginName, paramKey, [adapterName], [contextName])
```

**Table 5-43 removePluginParam Arguments**

Argument	Definition
<i>pluginName</i>	Name of the plug-in to be modified.
<i>paramKey</i>	Parameter to be removed.
<i>adapterName</i>	Optional. Name of the adapter to be modified. If not specified, the global plug-in is modified.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following commands remove the plug-in parameter `ReplaceAttribute` from `VirtualAttr` plug-in:

```
removePluginParam(adapterName='ldap1', pluginName='VirtualAttr',  
paramKey='ReplaceAttribute')
```

```
removePluginParam(pluginName='VirtualAttr', paramKey='ReplaceAttribute')
```

## 5.1.43 replacePluginParam

Online command that replaces parameter values for a plug-in.

### Description

Replaces existing parameter values for the specified adapter level plug-in or global plug-in.

### Syntax

```
replacePluginParam(pluginName, paramName, paramValues, [adapterName],[contextName])
```

**Table 5-44 replacePluginParam Arguments**

Argument	Description
<i>pluginName</i>	Name of the plug-in to be modified.
<i>paramName</i>	Name of the parameter to be replaced.
<i>paramValues</i>	New values of the parameter. For more than one new value, separate each new parameter value are by a " ".
<i>adapterName</i>	Optional. Name of the adapter to be modified. If not specified, the global plug-in is modified.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

## Examples

The following commands replace the parameter values for the associated plug-in for an adapter:

```
replacePluginParam(adapterName='ldap1', pluginName='VirtualAttr',
paramName='ReplaceAttribute', paramValues='cn=%uid%')

replacePluginParam(adapterName='ldap1', pluginName='UserManagement',
paramName='mapAttribute', paramValues='orclguid=objectGuid | uniquemember=member')
```

## 5.1.44 unassignViewFromAdapter

Online command that unassigns a view from an adapter.

### Description

Unassigns a view from an LDAP adapter configuration.

### Syntax

```
unassignViewFromAdapter (viewName, adapterName, [contextName])
```

**Table 5-45 unassignViewFromAdapter Arguments**

Argument	Definition
<i>viewName</i>	Name of the view.
<i>adapterName</i>	Name of the LDAP adapter.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default is "default".

### Example

The following command unassigns `userView` associated with `ldap1` adapter:

```
unassignViewFromAdapter('userView','ldap1', 'default')
```

## 5.1.45 listSSLStoreType

Online command that lists the type of SSL store in use.

### Description

This command lists the type of SSL store in use for libOVD (`JKS` or `KSS`).

### Syntax

```
listSSLStoreType (contextName=[contextName])
```

**Table 5-46 listSSLStoreType Arguments**

Argument	Definition
<i>contextName</i>	Name of the OPSS context with which libOVD configuration is associated. The default value is <code>default</code> .

**Example**

This following command list the SSL store types in use:

```
listSSLStoreType(contextName='default')
```

## 5.1.46 enableKSSForSSL

Online command to enable `KSS` for libOVD.

**Description**

This command enables `KSS` for SSL, and disables `JKS` if it was enabled before. For more information about `KSS`, see *Oracle® Fusion Middleware Securing Applications with Oracle Platform Security Services*.

**Syntax**

```
enableKSSForSSL(contextName=[contextName])
```

**Table 5-47 enableKSSForSSL Arguments**

Argument	Definition
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is <code>default</code> .

**Example**

The following command enables `KSS` for SSL:

```
enableKSSForSSL(contextName='default')
```

## 5.1.47 enableJKSForSSL

Online command to enable `JKS` for libOVD.

**Description**

This command enables `JKS` for SSL, and disables `KSS` if it was enabled before. The command assumes that the libOVD adapters.jks file exists.

**Syntax**

```
enableJKSForSSL(contextName=[contextName])
```

**Table 5-48 enableJKSForSSL Arguments**

Argument	Definition
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is <code>default</code> .

**Example**

The following command enables `JKS` for SSL:

```
enableJKSForSSL(contextName='default')
```

## 5.1.48 createKeyStoreAndEnableJKSForSSL

Online command to enable JKS for SSL.

### Description

This command enables JKS for SSL, and disables KSS if it was enabled before. The command creates the libOVD adapters.jks file.

### Syntax

```
createKeyStoreAndEnableJKSForSSL(keystorePassword=[password], contextName=[contextName])
```

**Table 5-49 createKeyStoreAndEnableJKSForSSL Arguments**

Argument	Definition
keystorePassword	Password for libOVD adapters.jks file.
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.

### Example

The following command enable JKS for SSL:

```
createKeyStoreAndEnableJKSForSSL(keystorePassword='welcome1', contextName='default')
```

## 5.1.49 importTrustedCertificateIntoSSLStore

Online command to import trusted certificate into SSL store.

### Description

This command imports the provided trusted certificate into SSL store.

### Syntax

```
importTrustedCertificateIntoSSLStore(certificateFileName=[cert_file], aliasName=[aliasName], contextName=[contextName])
```

**Table 5-50 importTrustedCertificateIntoSSLStore Arguments**

Argument	Definition
certificateFileName	File name that contains the certificate.
aliasName	Alias name for the certificate.
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.

### Example

The following command imports the provided trusted certificate into SSL store:

```
importTrustedCertificateIntoSSLStore(certificateFileName='/tmp/cert.txt', aliasName='myCert1', contextName='default')
```



## 5.1.50 migrateAllTrustedCertificatesFromJKSToKSS

Online command to migrate all trusted certificates from JKS-based libOVD truststore to KSS store.

### Description

This command migrates all trusted certificates from JKS-based libOVD truststore to KSS store.

### Syntax

```
migrateAllTrustedCertificatesFromJKSToKSS (contextName=[contextName])
```

**Table 5-51 migrateAllTrustedCertificatesFromJKSToKSS Arguments**

Argument	Definition
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.

### Example

The following command migrates all trusted certificates from JKS-based libOVD truststore to KSS store:

```
migrateAllTrustedCertificatesFromJKSToKSS (contextName='default')
```

## 5.1.51 migrateTrustedCertificatesFromJKSToKSS

Online command to migrate given trusted certificates from JKS-based libOVD truststore to KSS store.

### Description

This command migrates the given trusted certificates from JKS-based libOVD truststore to KSS store.

### Syntax

```
migrateTrustedCertificatesFromJKSToKSS (aliasNames=[alias_names],
contextName=[contextName])
```

**Table 5-52 migrateTrustedCertificatesFromJKSToKSS Arguments**

Argument	Definition
aliasNames	List of alias names to migrate separated by a comma.
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.

### Example

The following command migrates the specified trusted certificates from JKS-based libOVD truststore to KSS store:

```
migrateTrustedCertificatesFromJKSToKSS (aliasNames='alias1,alias2',
contextName='default')
```

## 5.1.52 changeLDAPHostPort

Online command to change given LDAP host and port in an existing LDAP adapter configuration to a new host and port.

### Description

This command changes given LDAP host and port in an existing LDAP adapter configuration to a new host and port.

### Syntax

```
changeLDAPHostPort(adapterName=[adapterName], oldHost=[oldHost], oldPort=[oldPort],
newHost=[newHost], newPort=[newPort], contextName=[contextName])
```

**Table 5-53 changeLDAPHostPort Arguments**

Argument	Definition
adapterName	Name of the LDAP adapter to be modified.
oldHost	Old LDAP host.
oldPort	Old LDAP port.
newHost	New LDAP host.
newPort	New LDAP port.
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.

### Example

The following command changes given LDAP host and port in an existing LDAP adapter configuration to a new host and port:

```
changeLDAPHostPort(adapterName='ldap1', oldHost='oldhost.example.domain.com',
oldPort=389, newHost='newhost.example.domain.com', newPort=389)
```

## 5.1.53 removeLDAPHostPort

Online command to remove a remote host and a port from an existing LDAP adapter configuration.

### Description

This command removes a remote host and a port from an existing LDAP adapter configuration.

### Syntax

```
removeLDAPHostPort(adapterName=[adapterName], host=[host], port=[port],
contextName=[contextName])
```

**Table 5-54 removeLDAPHostPort Arguments**

Argument	Definition
adapterName	Name of the LDAP adapter to be modified.
host	Remote LDAP host.
port	Remote LDAP port.
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.

**Example**

The following command removes a remote host and a port from an existing LDAP adapter configuration:

```
removeLDAPHostPort(adapterName='ldap1', host='myhost.example.domain.com', port=389)
```

## 5.1.54 setReadOnlyForLDAPHost

Online command to set the given host and port to read-only/writable in an existing LDAP adapter configuration.

**Description**

This command sets the given host and port to read-only/writable in an existing LDAP adapter configuration.

**Syntax**

```
setReadOnlyForLDAPHost(adapterName=[adapterName], host=[host], port=[port],
readOnly=[true/false], contextName=[contextName])
```

**Table 5-55 setReadOnlyForLDAPHost Arguments**

Argument	Definition
adapterName	Name of the LDAP adapter to be modified.
host	LDAP host.
port	LDAP port.
readOnly	It has values: true or false.
contextName	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.

**Example**

The following command sets the given host and port to read-only in an existing LDAP adapter configuration:

```
setReadOnlyForLDAPHost(adapterName='ldap1', host='myhost.example.domain.com', port=389,
readOnly=true)
```

## 5.1.55 dumpLdapConnectionPoolStats

Online command that dumps the current connection pool statistics for an adapter to a file for the given JVM on which WLS is configured.

### Description

This command dumps the current connection pool statistics for an adapter to a file for the given JVM on which WLS is configured.

### Syntax

```
dumpLdapConnectionPoolStats(fileName=[fileName], adapterName=[adapterName],
contextName=[contextName])
```

**Table 5-56 dumpLdapConnectionPoolStats Arguments**

Argument	Definition
<code>fileName</code>	Refers to the full path of the file.
<code>adapterName</code>	Name of the LDAP adapter.
<code>contextName</code>	Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is <code>default</code> .

### Example

The following example dumps the connection pool statistics for `ldap1` adapter into the specified file:

```
dumpLdapConnectionPoolStats('/tmp/poolstats1.txt','ldap1','default')
```

## 5.1.56 addCipherSuite

Online command that adds cipher suites to an existing LDAP adapter configuration.

### Description

Adds cipher suites to an existing LDAP adapter configuration.

### Syntax

```
addCipherSuite(adapterName, cipherSuite, [contextName])
```

**Table 5-57 addCipherSuite Arguments**

Argument	Definition
<code>adapterName</code>	Name of the adapter to be updated.
<code>cipherSuite</code>	Name of the cipher suite to be added.
<code>contextName</code>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is <code>default</code> .

### Example

The following example adds a cipher suite `TLS_DHE_RSA_WITH_AES_128_CBC_SHA` to an existing adapter, `ldap1`:

```
addCipherSuite("ldap1", 'TLS_DHE_RSA_WITH_AES_128_CBC_SHA', "myContext")
```

## 5.1.57 removeCipherSuite

Online command that removes cipher suites from an existing LDAP adapter configuration.

### Description

Removes cipher suite from an existing LDAP adapter configuration

### Syntax

```
removeCipherSuite(adapterName, cipherSuite, [contextName])
```

**Table 5-58** removeCipherSuite Arguments

Argument	Definition
<i>adapterName</i>	Name of the adapter to be updated.
<i>cipherSuite</i>	Name of the cipher suite to be removed.
<i>contextName</i>	Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.

### Example

The following example deletes the cipher suite `TLS_DHE_RSA_WITH_AES_128_CBC_SHA` from `ldap1` configuration:

```
removeCipherSuite("ldap1", 'TLS_DHE_RSA_WITH_AES_128_CBC_SHA', "myContext")
```

# Part II

## WLST for Oracle Mobile Security Suite

Part II describes the WLST commands for Mobile Access Security Server.

# 6

## Mobile Security Access Server WLST Commands

Use these custom WebLogic Scripting Tool (WLST) commands to manage Mobile Access Security Server (MSAS) instances. You can also manage in the MSAS console. Mobile Access Security Server WLST commands are divided into the following categories to manage MSAS instances:

- [Using the WLST Commands](#)
- [MSAS Configuration Commands](#)
- [MSAS Identity Store Profile Commands](#)
- [Repository Commands](#)
- [Session Commands](#)
- [Token Issuer Trust Configuration Commands](#)
- [Diagnostic Commands](#)

### 6.1 Using the WLST Commands

Prerequisites to be followed before running the WLST commands.

- Ensure that the Administration Server for the MSM domain is running.
- Execute the MSAS WLST commands from the `IDM_HOME/common/bin` directory. For example, `/home/oracle/oms/ORACLE_IDM/common/bin`.
- Connect to a running instance of the server and enter `help('msasManage')` to display the help for the MSAS configuration, identity store, and repository management commands.
- Connect to a running instance of the server and enter `help('wsmManage')` to display the help for `resetWSMPolicyRepository`, session commands, and trusted issuer configuration commands

### 6.2 MSAS Configuration Commands

Use these MSAS Configuration Commands to manage MSAS configurations and application metadata.

**Table 6-1 MSAS Configuration Commands**

Use this command...	To...	Use with WLST...
<a href="#">displayMSASConfiguration</a>	Display the MSAS configuration properties, and their values and groups, for a MSAS instance.	Online
<a href="#">setMSASConfiguration</a>	Create or modify the configuration properties for a MSAS instance.	Online

**Table 6-1 (Cont.) MSAS Configuration Commands**

Use this command...	To...	Use with WLST...
<a href="#">setMSASLogLevel</a>	Set the logger configuration for a MSAS instance to a specified level.	Online
<a href="#">getMSASLogLevel</a>	Get logger configuration for a MSAS instance.	Online
<a href="#">listMSASLoggers</a>	List the logger configurations for a MSAS instance.	Online

## 6.2.1 displayMSASConfiguration

The `displayMSASConfiguration` command is an online command that displays the configuration properties, and their values and groups, for a MSAS instance.

### Description

Displays the configuration properties, and their values and groups, for a MSAS instance. If a property is not defined in the MSAS instance's configuration document, then the default value defined for the product is displayed. If a MSAS instance name is not specified, or the specified MSAS instance does not exist, then the default values defined for the product are displayed.

### Syntax

```
displayMSASConfiguration(instanceName=None)
```

Argument	Definition
<i>instanceName</i>	Optional. The name of the MSAS instance associated with the configuration document from which property values are displayed.

### Example

This example displays the MSAS configuration properties, including their values and groups, for an instance named `MSAS-123456`.

```
wls:/mydomain/serverConfig> displayMSASConfiguration('MSAS-123456')
```

This example displays the default MSAS configuration properties.

```
wls:/mydomain/serverConfig> displayMSASConfiguration()
```

## 6.2.2 setMSASConfiguration

The `setMSASConfiguration` command is an online command that sets the configuration properties in the configuration document associated with a MSAS instance

### Description

Sets the configuration properties in the configuration document associated with a MSAS instance. If the configuration document does not exist for the instance, it is created automatically. A new property with values and/or groups of values can be added inside the configuration document. The set of acceptable properties are determined by the default set of properties supported by the product. Specific property values or groups of values can also be



removed from the configuration document. The configuration document is removed if it does not have any properties.

### Syntax

```
setMSASConfiguration(instanceName,categoryName,propertyName,group=None,values=None)
```

Argument	Definition
<i>instanceName</i>	The name of the MSAS instance associated with the configuration document to be modified.
<i>categoryName</i>	The category of the configuration property. This is verified with the default set of properties for acceptability.
<i>propertyName</i>	The name of the configuration property. This is verified with the default set of properties for acceptability.
<i>groupName</i>	Optional. The group containing the set of values to add in the configuration document. If the group exists, and this value is set to <code>None</code> , then the group is removed.
<i>propertyValues</i>	Optional. The array of values to set for a property or group inside the configuration document. Default is <code>None</code> which refers to an empty array list

### Example

This example is setting the location of the SSL keystore in a MSAS instance named MSAS-123456.

```
wls:/mydomain/serverConfig>
setMSASConfiguration('MSAS-123456','ServerSettings','ssl.keystore.location',None,['kss://
mag1/sslkeystore'])
```

## 6.2.3 setMSASLogLevel

The `setMSASLogLevel` command is an online command that sets the logger configuration for a MSAS instance to a specified level.

### Description

Sets the logger configuration for a MSAS instance to a specified level (for example, `WARNING`).

### Syntax

```
setMSASLogLevel(instanceName,logger,level)
```

Argument	Definition
<i>instanceName</i>	The name of the MSAS instance logging configuration to be modified. This option is required; there is no default value.
<i>logger</i>	A logger name. The empty string denotes the root logger. This option is required; there is no default value.

Argument	Definition
<i>level</i>	<p>The level name. It has to be one of the following:</p> <ul style="list-style-type: none"> <li>• SEVERE</li> <li>• WARNING</li> <li>• INFO</li> <li>• CONFIG</li> <li>• FINE</li> <li>• FINER</li> <li>• FINEST</li> </ul> <p>An empty string can be used to set the level to null (inherited from parent). This option is required; there is no default value.</p>

### Example

This example is setting a logging level of `FINEST` for an MSAS logger named `oracle.idm.gateway.grs` in an instance named `MSAS-123456`.

```
wls:/mydomain/serverConfig>
setMSASLogLevel('MSAS-123456','oracle.idm.gateway.grs','FINEST'])
```

This example is setting a logging level to null for an MSAS logger named `oracle.idm.gateway.grs` in an instance named `MSAS-123456`.

```
wls:/mydomain/serverConfig> setMSASLogLevel('MSAS-123456','oracle.idm.gateway.grs',''])
```

This example is setting a logging level of `SEVERE` for the root logger in an instance named `MSAS-123456`.

```
wls:/mydomain/serverConfig> setMSASLogLevel('MSAS-123456','','SEVERE'])
```

## 6.2.4 getMSASLogLevel

The `getMSASLogLevel` command is an online command that retrieves the logger configuration for a MSAS instance.

### Description

Gets the logger configuration for a MSAS instance.

### Syntax

```
getMSASLogLevel(instanceName,logger)
```

Argument	Definition
<i>instanceName</i>	The MSAS instance name of the logging configuration. This option is required; there is no default value.
<i>logger</i>	A logger name. The empty string denotes the root logger. This option is required; there is no default value.

### Example

This example is getting the logging level for an MSAS logger named `oracle.idm.gateway.grs` in an instance named `MSAS-123456`.

```
wls:/mydomain/serverConfig> getMSASLogLevel('MSAS-123456','oracle.idm.gateway.grs')
```

This example is getting the logging level for the root MSAS logger in an instance named MSAS-123456.

```
wls:/mydomain/serverConfig> getMSASLogLevel('MSAS-123456','oracle.idm.gateway.grs')
```

## 6.2.5 listMSASLoggers

The listMSASLoggers command is an online command that lists the logger configurations for a MSAS instance.

### Description

Lists the logger configurations for a MSAS instance.

### Syntax

```
listMSASLoggers(instanceName)
```

Argument	Definition
<i>instanceName</i>	The MSAS instance name of the logging configuration. This option is required; there is no default value.

### Example

This example is listing the loggers for a MSAS instance named MSAS-123456.

```
wls:/mydomain/serverConfig> listMSASLoggers('MSAS-123456')
```

## 6.3 MSAS Identity Store Profile Commands

Use these commands to manage an identity store profile, which is a logical representation of a user repository. The identity store configuration is stored in an Identity Profile Document in the MSAS Repository. All identity store profile management commands must be performed in the context of a repository session. A repository session can only act on a single document.

**Table 6-2 Identity Profile Management Commands**

Use this command...	To...	Use with WLST...
<a href="#">createIdentityProfile</a>	Create a new identity store profile in a MSAS instance within a repository session.	Online
<a href="#">displayIdentityProfile</a>	Display the contents of a specified identity store profile, or list the names of all identity profiles, in a MSAS instance.	Online
<a href="#">selectIdentityProfile</a>	Select an identity store profile in a MSAS instance for modification within a repository session.	Online
<a href="#">deleteIdentityProfile</a>	Delete an identity store profile from a specified MSAS instance within a repository session.	Online
<a href="#">setIdentityProfileDirectory</a>	Set or update the directory information for an identity store profile in a MSAS instance within a repository session.	Online

**Table 6-2 (Cont.) Identity Profile Management Commands**

Use this command...	To...	Use with WLST...
<a href="#">setIdentityProfileUser</a>	Set or update the user information for an identity store profile in a MSAS instance within a repository session.	Online
<a href="#">setIdentityProfileGroup</a>	Set or update the group information for an identity store profile for a MSAS instance within a repository session.	Online

## 6.3.1 createIdentityProfile

The `createIdentityProfile` command is an online command that creates a new identity store profile in a MSAS instance within a repository session.

### Description

Create a new identity store profile in a MSAS instance within a repository session. The identity store profile must be associated with a specified MSAS instance. Issuing this command outside of a repository session will result in an error.

#### Note:

- This command only creates a new identity store profile for a MSAS instance. To use an identity store profile at runtime, the MSAS configuration must already be set, as described in [setMSASConfiguration](#). After running `setMSASConfiguration`, you must also restart the Oracle MSAS Management Server for the runtime to access the configured profile.  
  
You cannot create multiple identity store profiles in the same session. You must commit the session after creating a profile, and then start a new session to create another profile.
- If an error occurs during the creation of an identity store profile, the profile must first be committed before it can be deleted.

### Syntax

```
createIdentityProfile(instanceName, ProfileName, description=None)
```

Argument	Definition
<i>instanceName</i>	Name of the MSAS instance in which you want to create an identity store profile.
<i>ProfileName</i>	Name of the new identity store profile.
<i>description</i>	Optional. Description of the new identity store profile.

### Example

This example creates an identity store profile named `identity-profile` in a MSAS instance named `MSAS-123456`, with a description of `Identity profile`.

```
wls:/mydomain/serverConfig> createIdentityProfile('MSAS-123456','identity-profile','Identity profile')
```

## 6.3.2 displayIdentityProfile

The `displayIdentityProfile` command is an online command that displays the contents of a specified identity store profile in a MSAS instance, or list the names of all identity store profiles associated with a specified MSAS instance.

### Description

Display the contents of a specified identity store profile in a MSAS instance, or list the names of all identity store profiles associated with a specified MSAS instance. If this command is executed from an active session, session changes are also displayed. If this command is executed outside of active session, content from the repository is displayed.

### Syntax

```
displayIdentityProfile(instanceName, ProfileName=None)
```

Argument	Definition
<i>instance</i>	Name of the MSAS instance associated with the identity store profiles to be displayed.
<i>ProfileName</i>	Optional. The name of the identity store profile to be displayed. If a name is not specified, then the names of the all available identity profiles for the MSAS instance are displayed.

### Example

This example displays the contents of an identity store profile named `identity-profile` in an instance named `MSAS-123456`.

```
wls:/mydomain/serverConfig> displayIdentityProfile('MSAS-123456','identity-profile')
```

This example displays the names of the all available identity store profiles in the MSAS instance `MSAS-123456`.

```
wls:/mydomain/serverConfig> displayIdentityProfile('MSAS-123456')
```

## 6.3.3 selectIdentityProfile

The `selectIdentityProfile` command is an online command that selects an identity store profile for modification in a MSAS instance within a repository session.

### Description

Select an identity store profile for modification in a MSAS instance within a repository session.

### Syntax

```
selectIdentityProfile(instanceName, ProfileName, description=None)
```

Argument	Definition
<i>instanceName</i>	Name of the MSAS instance associated with the identity store profile to be modified.
<i>ProfileName</i>	Name of the identity store profile to be modified.

### Example

This example selects the identity profile `identity-profile` in MSAS instance `MSAS-123456` for modification:

```
wls:/mydomain/serverConfig> selectIdentityProfile('MSAS-123456','identity-profile')
```

## 6.3.4 deleteIdentityProfile

The `deleteIdentityProfile` command is an online command that deletes an identity store profile from a MSAS instance within a repository session.

### Description

Delete an identity store profile from a MSAS instance within a repository session. Issuing this command outside of a repository session will result in an error.



#### Note:

You cannot both create and delete the same identity store profile in a single repository session. You must commit the session after creating a profile, and then start a new session in order to delete that profile.

### Syntax

```
deleteIdentityProfile(instanceName,ProfileName)
```

Argument	Definition
<i>instanceName</i>	Name of the MSAS instance associated with the identity store profile to be deleted.
<i>ProfileName</i>	Name of the identity store profile to be deleted.

### Example

This example deletes the identity profile `identity-profile` from the MSAS instance `MSAS-123456`.

```
wls:/mydomain/serverConfig> deleteIdentityProfile('MSAS-123456','identity-profile')
```

## 6.3.5 setIdentityProfileDirectory

The `setIdentityProfileDirectory` command is an online command that sets or updates the directory information for an identity store profile in a MSAS instance within a repository session.

### Description

Set or update the directory information for an identity store profile in a MSAS instance within a repository session. The `secure` argument can be used to determine if the identity store connection should be made over SSL. Before running this command, an identity store profile must be selected for modification or must be created in the current session.

### Syntax

```
setIdentityProfileDirectory(DirectoryType,hosts,bindDN,bindPass,baseDN,isSecure)
```

Argument	Definition
<i>directoryType</i>	Type of identity store profile directory. Supported types are: <ul style="list-style-type: none"> <li>• OID (Oracle Internet Directory)</li> <li>• OUD (Oracle Unified Directory)</li> <li>• ACTIVE_DIRECTORY</li> <li>• ODSEE (Oracle Directory Server Enterprise Edition)</li> <li>• WLS_LDAP (Embedded LDAP in WebLogic Server)</li> </ul>
<i>hosts</i>	Host port information of the directory in the form <code>host:port</code> .
<i>bindDN</i>	The distinguished name of the user to be used for connecting to the directory. For example: <code>CN=Administrator,CN=Users,DC=mycompany,DC=com</code> .
<i>bindPass</i>	Bind distinguished name (DN) password for accessing the directory.
<i>baseDN</i>	Base distinguished name (DN) information of the directory. For example: <code>DC=mycompany,DC=com</code>
<i>isSecure</i>	Flag (boolean) to indicate if the connection to the directory should be made over SSL. When set to true the connection is configured over SSL.

### Example

This example creates an identity store profile directory with a type of `OID` on a host named `['host.example.com:1234']`, with bind distinguished name of `cn=host,dn=oracle,dn=com`, using a password of `welcome`, and a base distinguished name of `cn=host,dn=oracle,dn=com`, which does not require an SSL connection because the `secure` argument is `false`.

```
wls:/mydomain/serverConfig> setIdentityProfileDirectory('OID',
['host.example.com:1234'], 'cn=host,dn=oracle,dn=com', 'welcome', 'cn=us,dn=oracle,dn=com', f
alse)
```

This example creates an identity store profile directory that is identical to the one created in the previous example except that it requires an SSL connection to access the directory because the `secure` argument is `true`.

```
wls:/mydomain/serverConfig> setIdentityProfileDirectory('OID',
['host.example.com:1234'], 'cn=host,dn=oracle,dn=com', 'welcome', 'cn=us,dn=oracle,dn=com', t
rue)
```

## 6.3.6 setIdentityProfileUser

The `setIdentityProfileUser` command is an online command that sets or updates the user information for an identity store profile in a MSAS instance within a repository session.

### Description

Set or update the user information for an identity store profile in a MSAS instance within a repository session. Before running this command, an identity store profile must be selected for modification or must be created in the current session.

### Syntax

```
setIdentityProfileUser(baseDN,loginIDAttribute,objectClassNames)
```

Argument	Definition
<i>baseDN</i>	The base distinguished name (DN) used to create users.
<i>loginIDAttribute</i>	The login identity of the user.
<i>objectClassNames</i>	The fully-qualified names of one or more of LDAP object classes used to identify users.

### Example

This example creates an identity store profile user with base DN of `cn=host,dn=oracle,dn=com`, with login ID of `uid`, which is represented by a schema object class named `inteorgperson`.

```
wls:/mydomain/serverConfig> setIdentityProfileUser('cn=user,dn=oracle,dc=com','uid',
['inteorgperson'])
```

## 6.3.7 setIdentityProfileGroup

The `setIdentityProfileGroup` command is an online command that sets or updates the group information for an identity store profile in a MSAS instance within a repository session.

Command Category: MSAS Configuration Management

Use with WLST: Online

### Description

Set or update the group information for an identity store profile in a MSAS instance within a repository session. Before running this command, an identity store profile must be selected for modification or must be created in current session.

### Syntax

```
setIdentityProfileGroup(baseDN,groupNameAttribute,objectClassNames)
```

Argument	Definition
<i>baseDN</i>	The base DNs used to create groups or enterprise roles.



Argument	Definition
<i>groupNameAttribute</i>	The attribute that uniquely identifies the name of the enterprise role or group.
<i>objectClassNames</i>	The fully-qualified names of one or more LDAP object classes used to represent enterprise roles or groups.

**Example**

This example creates an identity store profile user group with a base DN of `cn=host,dn=oracle,dn=com`, with group name attribute of `cn`, which is represented by a schema object class named `groupofuniquenames`.

```
wls:/mydomain/serverConfig> setIdentityProfileGroup('cn=group,dn=oracle,dc=com','cn',
['groupofuniquenames'])
```

## 6.4 Repository Commands

Use these Repository WLST commands to manage the documents stored in the Oracle Repository.

**Table 6-3 Policy Repository Management Commands**

Use this command...	To...	Use with WLST...
<a href="#">exportMSASAppMetadata</a>	Export MSAS application metadata from the repository to a specified ZIP archive.	Online
<a href="#">importMSASAppMetadata</a>	Import MSAS application metadata into the repository from a specified ZIP archive.	Online
<a href="#">migrateMSASAppHostports</a>	Migrate physical host:port values for applications in a MSAS instance to a mapped host:port value.	Online
<a href="#">resetWSMPolicyRepository</a>	Delete the existing policies stored in the repository and refresh it with the latest set of predefined policies that are provided in the new installation of the Oracle MSAS software.	Online

### 6.4.1 exportMSASAppMetadata

The `exportMSASAppMetadata` command is an online command that exports MSAS application metadata from the repository into a specified ZIP archive.

**Description**

Exports MSAS application metadata from the repository into a specified ZIP archive. If the specified archive already exists, you can choose whether to merge the documents into the existing archive, overwrite the archive, or cancel the operation. By default, all metadata for MSAS applications in the current domain is exported to the archive. However, you can use a MSAS instance name and MSAS application name to export specific metadata for these MSAS applications in the repository.

## Syntax

```
exportMSASAppMetadata (archiveFileName, [instanceName=None], [applicationName=None],
[includeShared='false'])
```

Argument	Definition
<i>archiveFileName</i>	Name of the ZIP archive. If the specified archive already exists, you can choose whether to overwrite the archive, merge the documents into the existing archive, or cancel the operation. During override, the original archive is backed up and a message describes the location of the backup archive.
<i>instanceName</i>	Optional. The name of the MSAS instance from which you want to export the metadata of one or more MSAS applications. If no MSAS instance name is specified, but a valid MSAS application name is specified, then the metadata for all MSAS instances containing this MSAS application is exported. When neither the MSAS instance nor MSAS application name is specified, then the metadata for all the MSAS applications across all MSAS instances is exported.  A wildcard '%' is allowed. If this argument is set to None, empty "", or '%', then all MSAS instances are searched.
<i>applicationName</i>	Optional. The name of the MSAS application for the metadata to be exported.  A wildcard '%' is allowed. If this argument is set to None, empty "", or '%', then all applications in the specified instance are searched.
<i>includeShared</i>	Optional. Specifies whether a shared resource (such as a referenced policy) should be included with exported metadata.

## Example

This example exports the metadata for all the MSAS applications across all MSAS instances into an archive named `MSASApplications.zip`.

```
wls:/mydomain/serverConfig> exportMSASAppMetadata ('/tmp/MSASApplications.zip')
```

This example also exports the metadata for all MSAS applications across all MSAS instances into the `MSASApplications.zip` archive.

```
wls:/mydomain/serverConfig> exportMSASAppMetadata ('/tmp/MSASApplications.zip','', [''])
```

This example exports the metadata for all applications across all MSAS instances that begin with `MSAS` into the `MSASApplications.zip` archive.

```
wls:/mydomain/serverConfig> exportMSASAppMetadata ('/tmp/MSASApplications.zip','MSAS%')
```

This example exports the metadata for all applications that begin with `virtual` on an instance named `MSAS-123456` into the `MSASApplications.zip` archive.

```
wls:/mydomain/serverConfig> exportMSASAppMetadata ('/tmp/MSASApplications.zip','MSAS-1234561', ['virtual%'])
```

This example exports the metadata for an application named `Virtual_Foo` on the `MSAS-123456` instance into the `MSASApplications.zip` archive.

```
wls:/mydomain/serverConfig> exportMSASAppMetadata ('/tmp/MSASApplications.zip','MSAS-1234561', ['virtual_Foo'])
```

This example exports the metadata for all applications that begin with `virtual` across all MSAS instances into the `MSASApplications.zip` archive.

```
wls:/mydomain/serverConfig> exportMSASAppMetadata('/tmp/MSASApplications.zip','',
['virtual%'])
```

This example exports the metadata for all applications that begin with `virtual`, including application's shared resources, on the `MSAS-123456` instance into the `MSASApplications.zip` archive.

```
wls:/mydomain/serverConfig> exportMSASAppMetadata('/tmp/
MSASApplications.zip','MSAS-1234561',['virtual%'],true)
```

## 6.4.2 importMSASAppMetadata

The `importMSASAppMetadata` command is an online command that imports MSAS application metadata into the repository from a specified ZIP archive.

### Description

Import MSAS application metadata into the repository from a specified ZIP archive. You can use the `map` argument to provide the location of a file that describes how to map physical information from the source environment to the target environment. You can generate a new map file by setting the `generateMapFile` argument to `true`.

### Syntax

```
importMSASAppMetadata(archiveFileName,[mapFileName=None],[generateMapFlag='false'])
```

Argument	Definition
<i>archiveFileName</i>	Name of the ZIP archive to be imported.
<i>mapFileName</i>	Optional. Location of a sample map file that describes how to map physical information from the source environment to the target environment. You can generate a new map file by setting the <code>generateMapFlag</code> argument to <code>true</code> . If the map file already exists, it will be overwritten; however, no import takes place in the repository. If you specify a map file without setting the <code>generateMapFlag</code> , or setting it to <code>false</code> , and the map file does not exist, the operation fails and an error is displayed.
<i>generateMapFlag</i>	Optional. Specify whether to generate a map file at the location specified by the <code>mapFileName</code> argument. No metadata is imported when this argument is set to <code>true</code> , only a map file is generated. The default is <code>false</code> .

### Example

This example imports application metadata from the `MSASartifacts.zip` archive without using a map file.

```
wls:/mydomain/serverConfig> importMSASAppMetadata('/tmp/MSASartifacts.zip')
```

This example collects all application metadata from the `MSASartifacts.zip` archive and puts it into a map file named `MSASmapfile.txt`. If the specified map file already exists, it will be overwritten.

```
wls:/mydomain/serverConfig> importMSASAppMetadata('/tmp/MSASartifacts.zip','/tmp/
MSASmapfile.txt', true)
```

This example imports application metadata from the specified `MSASartifacts.zip` archive according to the `MSASmapfile.txt` map file.

```
wls:/mydomain/serverConfig> importMSASAppMetadata('/tmp/MSASartifacts.zip','/tmp/MSASmapfile.txt')
```

## 6.4.3 migrateMSASAppHostports

The `migrateMSASAppHostports` command is an online command that replaces the source `host:port` values with `host:port` values according to the source-to-target mapping in the specified `mapFileName` for applications in the repository that match the specified MSAS instance name and application name.

### Description

For applications in the repository that match the specified MSAS instance name and application name, replaces the source `host:port` values (for example, URLs to a back-end service) with `host:port` values, according to the source-to-target mapping in the specified `mapFileName`. You can generate a new map file by setting the `generateMapFlag` argument to `true`.

### Syntax

```
migrateMSASAppHostports(instanceName,applicationName,mapFileName,
[generateMapFlag='false'])
```

Argument	Definition
<i>instanceName</i>	Name of the MSAS instance. A wildcard is not accepted.
<i>applicationName</i>	Name of MSAS application whose referenced back-end service <code>host:port</code> information needs to be migrated or replaced. A wildcard <code>%</code> is allowed. If this argument is set to <code>None</code> , <code>"empty "</code> , or <code>'%'</code> , then all applications in the specified MSAS instance are searched.
<i>mapFileName</i>	Location of a input map file that describes how to map source MSAS <code>host:port</code> values to a target <code>host:port</code> . You can generate a new map file by setting the <code>generateMapFlag</code> argument to <code>true</code> . If the map file already exists, it will be overwritten. The generated map file provides destination <code>host:port</code> values for the source <code>host:port</code> values that need to be replaced. If you specify a map file without setting the <code>generateMapFlag</code> argument to <code>true</code> , the map file is treated as an input map file and it will be read in for source-to-target <code>host:port</code> replacement for all matched applications. If you specify a map file without setting the <code>generateMapFlag</code> , or by setting it to <code>false</code> , and the map file does not exist, the operation fails and an error is displayed.
<i>generateMapFlag</i>	Optional. Specify whether to generate a map file at the location specified by the <code>mapFileName</code> argument. No MSAS application metadata is modified when this argument is set to <code>true</code> , only a map file is generated. The default is <code>false</code> .

### Example

This example collects all `host:port` values for the `myApp` application on an instance named `MSAS-1234`, and puts it into a newly-generated map file named `generatedMapfile.txt`. (Note that if a specified map file already exists, it will be overwritten).

```
wls:/mydomain/serverConfig> migrateMSASAppHostports ('MSAS-1234','myApp','/tmp/generatedMapfile.txt',true)
```

This example migrates the host:port values for all applications that begin with `myApp` on the `MSAS-1234` instance according to the `myMapfile.txt` map file.

```
wls:/mydomain/serverConfig> migrateMSASAppHostports('MSAS-1234', 'myApp%', '/tmp/myMapfile.txt')
```

The following migrates the host:port values for all applications on the `MSAS-1234` instance according to the `myMapfile.txt` map file.

```
wls:/mydomain/serverConfig> migrateMSASAppHostports('MSAS-1234', 'None', '/tmp/myMapfile.txt')
```

## 6.4.4 resetWSMPolicyRepository

The `resetWSMPolicyRepository` command is an online command that deletes the existing policies stored in the repository and refresh it with the latest set of predefined policies that are provided in the new installation of the Oracle MSAS software.

### Caution:

This command will delete all MSAS artifacts, including registered MSAS application and configuration documents. Restarting the server would recover the seed MSAS documents in the repository but will not recover other user-created documents. Therefore, prior to running this command, Oracle recommends running the [exportMSASAppMetadata](#) command to back up all your MSAS metadata.

### Description

Delete the existing policies stored in the repository and refresh it with the latest set of predefined policies that are provided in the new installation of the Oracle MSAS software. You can use the `clearStore` argument to specify whether to delete all policies, including custom user policies, from the repository before loading the new predefined policies.

### Note:

In order to reseed the repository with all Oracle MSAS predefined policies, you must restart the Mobile Security Manager (MSM) server after running the `resetWSMPolicyRepository` command.

### Syntax

```
resetWSMPolicyRepository([clearStore='false'])
```

Argument	Definition
<code>clearStore='false'</code>	Policies to be deleted. Valid values are: <ul style="list-style-type: none"> <li><code>true</code>—All policies in the repository, including custom user policies, are deleted.</li> <li><code>false</code>—Only the predefined policies supplied by Oracle are deleted. The default is <code>false</code>.</li> </ul>

**Example**

The following example deletes all the policies in the repository, including user policies, and adds the predefined policies provided in the current product installation:

```
wls:/wls-domain/serverConfig>resetWSPolicyRepository(true)
```

## 6.5 Session Commands

Use these custom WLST commands to manage the session. Some MSAS WLST commands, which modify repository documents and trusted token issuers must be executed in the context of a session.

**Table 6-4 Session Management WLST Commands**

Use this command...	To...	Use with WLST...
<a href="#">abortRepositorySession</a>	Abort the current modification session, discarding any changes that were made during the session.	Online
<a href="#">beginRepositorySession</a>	Begin a session to modify the repository documents.	Online
<a href="#">commitRepositorySession</a>	Write the contents of the current session to the repository.	Online
<a href="#">describeRepositorySession</a>	Describe the contents of the current session. This will indicate either that the session is empty or list the name of the document that is being updated, along with the type of update (create, modify, or delete).	Online

### 6.5.1 abortRepositorySession

The `abortRepositorySession` command is an online command that aborts the current Oracle Repository modification session, discarding any changes that were made to the repository during the session.

**Description**

Abort the current Oracle Repository modification session, discarding any changes that were made to the repository during the session.

**Syntax**

```
abortRepositorySession()
```

**Examples**

The following example aborts the current session.

```
wls:/wls-domain/serverConfig>abortRepositorySession()
```

## 6.5.2 beginRepositorySession

The `beginRepositorySession` command is an online command that initiates a session to modify the Oracle Repository.

### Description

Begin a session to modify the Oracle Repository. A repository session can only act on a single document. An error will be displayed if there is already a current session.

### Syntax

```
beginRepositorySession()
```

### Example

The following example begins a session.

```
wls:/wls-domain/serverConfig>beginRepositorySession()
```

## 6.5.3 commitRepositorySession

The `commitRepositorySession` command is an online command that writes the contents of the current session to the Oracle Repository.

### Description

Write the contents of the current session to the Oracle Repository. Messages are displayed that describe what was committed. An error will be displayed if there is no current session.

### Syntax

```
commitRepositorySession()
```

### Example

The following example commits the current repository modification session.

```
wls:/wls-domain/serverConfig>commitRepositorySession()
```

## 6.5.4 describeRepositorySession

The `describeRepositorySession` command is an online command that describes the contents of the current session.

### Description

Describe the contents of the current session. This will either indicate that the session is empty or list the name of the document that is being updated, along with the type of update (create, modify, or delete). An error will be displayed if there is no current session.

### Syntax

```
describeRepositorySession()
```

### Examples

The following example describes the current session.

```
wls:/wls-domain/serverConfig>describeRepositorySession()
```

## 6.6 Token Issuer Trust Configuration Commands

Use these WLST commands to view and define trusted issuers, trusted distinguished name (DN) lists, and token attribute rule filters for trusted SAML or JWT signing certificates. When using WLST commands to create, modify, and delete token issuer trust documents, you must execute the commands in the context of a session. Each session applies to a single trust document only.



### Note:

Connect to a running instance of the server and enter `help('wsmManage')` to view the help for these commands.

**Table 6-5 Token Issuer Trust Commands**

Use this command...	To...	Use with WLST...
<a href="#">createWSMTokenIssuerTrustDocument</a>	Create a new token issuer trust document using the name provided.	Online
<a href="#">deleteWSMTokenIssuerTrust</a>	Delete the entry for the issuer, including the DN list in it.	Online
<a href="#">deleteWSMTokenIssuerTrustAttributeRule</a>	Delete a token attribute rule associated with a trusted DN.	Online
<a href="#">deleteWSMTokenIssuerTrustDocument</a>	Delete the token issuer trust document, specified by the name argument, from the repository.	Online
<a href="#">displayWSMTokenIssuerTrust</a>	Display the names of the DN lists associated with a specified issuer.	Online
<a href="#">exportWSMTokenIssuerTrustMetadata</a>	Export the trust configuration (issuers, DNs, and token attribute rules) for all trusted issuers.	Online
<a href="#">importWSMTokenIssuerTrustMetadata</a>	Import the trust configuration (Issuers, DNs, and token attribute rules) for all trusted issuers.	Online
<a href="#">listWSMTokenIssuerTrustDocuments</a>	List the token issuer trust documents in the repository.	Online
<a href="#">revokeWSMTokenIssuerTrust</a>	Remove trusted issuers and associated configurations (DNs and token attribute rules).	Online
<a href="#">selectWSMTokenIssuerTrustDocument</a>	Select the token issuer trust document, identified by the name argument, to be modified in the session.	Online
<a href="#">setWSMTokenIssuerTrust</a>	Specify a trusted issuer with a DN list.	Online
<a href="#">setWSMTokenIssuerTrustAttributeFilter</a>	Add, delete, or update token attribute rules for a given token signing certificate DN.	Online
<a href="#">setWSMTokenIssuerTrustAttributeMapping</a>	Specify the DN of a token signing certificate and a list of trusted users. The name ID and the attribute can be mapped to another user ID.	Online
<a href="#">setWSMTokenIssuerTrustDisplayName</a>	Sets or resets the display name of the Token Issuer Trust document currently selected in the session.	Online



## 6.6.1 createWSMTokenIssuerTrustDocument

The `createWSMTokenIssuerTrustDocument` command is an online command that creates a new token issuer trust document using the name provided within a session.

### Description

Within a session, create a new token issuer trust document using the name provided. A display name can also be provided as the second argument. You must start a session (`beginWSMSession`) before creating or modifying any token issuer trust documents. If there is no current session or there is already an existing modification process, an error is displayed.

### Syntax

```
createWSMTokenIssuerTrustDocument(name, displayName)
```

Arguments	Definition
<i>name</i>	Name of the document to be created. An error is thrown if a name is not provided if a document by this name already exists, then a new document will not be created.
<i>displayName</i>	Optional. Display name for the document.

### Examples

In the following example, the trust document named `tokenissuertrustWLSbase_domain` is created, with a display name of `wls_domain Trust Document`. In the second example, no display name is provided.

```
wls:/wls-domain/serverConfig>
createWSMTokenIssuerTrustDocument("tokenissuertrustWLSbase_domain","wls_domain Trust
Document")
wls:/wls-domain/serverConfig>
createWSMTokenIssuerTrustDocument("tokenissuertrustWLSbase_domain")
```

## 6.6.2 deleteWSMTokenIssuerTrust

The `deleteWSMTokenIssuerTrust` command is an online command that deletes a trusted token issuer and its associated trusted DN list within a session.

### Description

Within a session, delete a trusted token issuer and its associated trusted DN list. Supported values for a SAML assertion or JWT token type are `dns.sv`, `dns.hok`, or `dns.jwt`. This issuer must exist in the token issuer trust document selected in the session for modification. If no trusted key identifiers exist, then the issuer itself is deleted.

To delete a specified list of trusted key identifiers for an issuer, use [selectWSMTokenIssuerTrustDocument](#). You must start a session (`beginWSMSession`) and select a token issuer trust document for modification before executing this command. If there is no current session or there is already an existing modification process, an error is displayed. You cannot modify the default token issuer trust document.

### Syntax

```
deleteWSMTokenIssuerTrust(type, issuer)
```

Arguments	Definition
<i>type</i>	The type of SAML assertion or JWT tokens the trusted issuer issues: <ul style="list-style-type: none"> <li>• <code>dns.sv</code> – SAML sender vouches client list</li> <li>• <code>dns.hok</code> – SAML HOK or Bearer</li> <li>• <code>dns.jwt</code> – JWT token.</li> </ul>
<i>issuer</i>	The name of the issuer whose trusted DN list will be deleted (for example, SAML assertion or JWT token). The issuer will also be deleted.

### Examples

In the following example, the issuer `www.yourCompany.com` and the DN list in the `dns.sv` trusted SAML sender vouches client list for the issuer are deleted:

```
wls:/wls-domain/serverConfig> deleteWSMTokenIssuerTrust('dns.sv', 'www.yourCompany.com')
```

In the following example, the issuer `www.yourCompany.com` and the DN list in the `dns.jwt` trusted JWT token sender vouches client list for the issuer are deleted:

```
wls:/wls-domain/serverConfig> deleteWSMTokenIssuerTrust('dns.jwt ',  
'www.yourCompany.com')
```

## 6.6.3 deleteWSMTokenIssuerTrustAttributeRule

The `deleteWSMTokenIssuerTrustAttributeRule` command is an online command that deletes a token attribute rule associated with a trusted DN from the token issuer trust document.

### Description

Delete a token attribute rule associated with a trusted DN from the token issuer trust document. To delete only the list of filter values for an attribute, use the [setWSMTokenIssuerTrustAttributeFilter](#) command. You must start a session (`beginWSMSession`) and select a token issuer trust document for modification before executing this command. If there is no current session or there is already an existing modification process, an error is displayed.

### Syntax

```
deleteWSMTokenIssuerTrustAttributeRule(dn)
```

Arguments	Description
<i>dn</i>	The DN of the token signing certificate that identifies the rule to be deleted.

### Examples

In the following example, the token attribute rule associated with the `'CN=weblogic, OU=Oracle Test Encryption Purposes Only, O=Oracle, C=US'` trusted DN is deleted.

```
wls:/wls-domain/serverConfig> deleteWSMTokenIssuerTrustAttributeRule('CN=weblogic,  
OU=Oracle Test Encryption Purposes Only, O=Oracle, C=US')
```

## 6.6.4 deleteWSMTokenIssuerTrustDocument

The `deleteWSMTokenIssuerTrustDocument` command is an online command that deletes the specified token issuer trust document permanently from the repository.

### Description

Deletes the specified token issuer trust document permanently from the repository. The default token issuer trust document (`oracle-default`) cannot be deleted.

### Syntax

```
deleteWSMTokenIssuerTrustDocument (name)
```

Arguments	Definition
<i>name</i>	Name of the token issuer trust document to be deleted.

### Examples

In the following example, the token issuer trust document `tokenissuertrustWLSbase_domain` trust document is deleted:

```
wls:/wls-domain/serverConfig>
deleteWSMTokenIssuerTrustDocument('tokenissuertrustWLSbase_domain')
```

## 6.6.5 displayWSMTokenIssuerTrust

The `displayWSMTokenIssuerTrust` command is an online command that displays a trusted token issuer and its associated trusted DN list.

### Description

Display a trusted token issuer and its associated trusted DN list. Supported values for a SAML assertion or JWT token type are `dns.hok`, `dns.sv`, or `dns.jwt`. The `issuer` argument is optional. If the issuer and type is specified and exists in the trusted issuer list for the type, then the associated DN lists for the issuer is displayed. If `issuer` is not set, then all trusted issuers of the given type are listed.

### Syntax

```
displayWSMTokenIssuerTrust (type, issuer)
```

Arguments	Definition
<i>type</i>	The type of SAML assertion or JWT tokens the trusted issuer issues: <ul style="list-style-type: none"> <li><code>dns.sv</code> – SAML sender vouches client list</li> <li><code>dns.hok</code> – SAML HOK or Bearer</li> <li><code>dns.jwt</code> – JWT token.</li> </ul>
<i>issuer</i>	Optional. The issuer whose trusted DN list is displayed (for example, SAML assertion or JWT token). If not set, the list of all the trusted issuers is displayed.

## Examples

In the following example, the DN lists in the `dns.sv` trusted SAML sender vouches client list for the `www.oracle.com` trusted issuer are displayed:

```
wls:/wls-domain/serverConfig>displayWSMTokenIssuerTrust('dns.sv', 'www.oracle.com')
```

In the following example, the names of all trusted SAML issuers associated with the `dns.sv` trusted SAML sender vouches client list are displayed:

```
wls:/wls-domain/serverConfig>displayWSMTokenIssuerTrust('dns.sv', None)
```

## 6.6.6 exportWSMTokenIssuerTrustMetadata

The `exportWSMTokenIssuerTrustMetadata` command is an online command that exports all trust configurations (issuer, DNSs, and token attribute rules) for all trusted issuers.

### Description

Export all the trust configurations (issuer, DNSs, and token attribute rules) for all trusted issuers. The trust configuration will be exported to an XML file identified by the specified location. The trust configuration for the issuers specified in the `exclude` list will not be exported. If no argument is passed, the trust configuration for all trusted issuers will be exported.

### Syntax

```
exportWSMTokenIssuerTrustMetadata (trustFile,excludeIssuers=None)
```

Arguments	Definition
<i>trustFile</i>	The location of the file where the exported metadata will be stored.
<i>excludeIssuers</i>	Optional. The list of issuers for which trust configuration should not be exported.

### Examples

In the following example, all trusted issuer configurations are exported to the specified XML file except for `www.oracle.com` and `www.yourcompany.com`, which have been excluded:

```
wls:/wls-domain/serverConfig>exportWSMTokenIssuerTrustMetadata (trustFile='/tmp/  
trustData.xml', ['www.oracle.com', 'www.myissuer.com'])
```

```
Starting Operation exportWSMTokenIssuerTrustMetadata ...  
Configuration for trusted issuers successfully exported.
```

In the following example, all specified trusted issuer configurations are exported to the specified XML file:

```
wls:/wls-domain/serverConfig>exportWSMTokenIssuerTrustMetadata (trustFile='/tmp/  
trustData.xml')
```

```
Starting Operation exportWSMTokenIssuerTrustMetadata ...  
Configuration for trusted issuers successfully exported.
```

## 6.6.7 importWSMTokenIssuerTrustMetadata

The `importWSMTokenIssuerTrustMetadata` command is an online command that imports the trust configurations (issuers, DN, and token attribute rules) for all trusted issuers.

### Description

Import the trust configurations (issuers, DN, and token attribute rules) for all trusted issuers. The trust configuration will be imported from an XML file identified by the specified location.

### Syntax

```
importWSMTokenIssuerTrustMetadata (trustFile)
```

Arguments	Definition
<i>trustFile</i>	The location of the file where the imported metadata will be stored.

### Examples

In the following example, all trusted issuer configurations are imported from the specified XML file:

```
wls:/wls-domain/serverConfig>importWSMTokenIssuerTrustMetadata (trustFile='/tmp/trustData.xml')
```

```
Starting Operation importWSMTokenIssuerTrustMetadata ...
Configuration for trusted issuers successfully imported.
```

## 6.6.8 listWSMTokenIssuerTrustDocuments

The `listWSMTokenIssuerTrustDocuments` command is an online command that lists all token issuer trust documents in the repository.

### Description

When used without any arguments, this command lists all the token issuer trust documents in the repository. If the `detail` argument is set to `true`, the display name and the status of the document are also displayed. You can use the wildcard character (\*) in combination with other characters. If no wildcard character is specified in the `name` argument, the document that matches the `name` argument exactly is displayed. If the `detail` argument is set to `true`, the contents of the document are listed. This command can be executed inside and outside of a session.

### Syntax

```
listWSMTokenIssuerTrustDocuments (name=None, detail='false')
```

Arguments	Definition
<i>name</i>	Optional. Name of the token issuer trust document. You can use wildcards with this argument.
<i>detail</i>	Optional. List the details for the requested document. The default is <code>false</code> .

## Examples

In the following example, the token issuer trust document `tokenissuertrustWLSbase_domain` trust document is listed along with any details:

```
wls:/wls-domain/serverConfig>
listWSMTokenIssuerTrustDocuments (tokenissuertrustWLSbase_domain, 'true')
```

## 6.6.9 revokeWSMTokenIssuerTrust

The `revokeWSMTokenIssuerTrust` command is an online command that revokes trust by removing all trusted issuers and associated configurations (DNs and token attribute rules).

### Description

Revokes trust by removing all trusted issuers and associated configurations (DNs and token attribute rules). The issuers specified in the optional `exclude` list will not be removed. If no argument is passed, then all trusted issuers and the associated configuration are removed.

### Syntax

```
revokeWSMTokenIssuerTrust (excludeIssuers=None)
```

Arguments	Definition
<code>excludeIssuers</code>	Optional. The list of issuers for which trust configuration should not be removed.

### Examples

In the following example, all trusted issuer configurations are removed except for `www.oracle.com` and `www.yourcompany.com`, which have been excluded:

```
wls:/wls-domain/
serverConfig>revokeWSMTokenIssuerTrust(['www.oracle.com', 'www.yourcompany.com'])
```

```
Starting Operation revokeWSMTokenIssuerTrust ...
Configuration for trusted issuers successfully removed.
```

In the following example, all trusted issuer configurations are removed:

```
wls:/wls-domain/serverConfig>revokeWSMTokenIssuerTrust()
```

```
Starting Operation revokeWSMTokenIssuerTrust ...
Configuration for trusted issuers successfully removed.
```

## 6.6.10 selectWSMTokenIssuerTrustDocument

The `selectWSMTokenIssuerTrustDocument` command is an online command that selects the token issuer trust document, identified by the name argument, to be modified in the session.

### Description

Selects the token issuer trust document, identified by the name argument, to be modified in the session. The name must match the value of the name attribute in the document. You must start a session (`beginWSMSession`) before executing this command. If there is no current session or there is already an existing modification process, an error is displayed. You cannot modify the default token issuer trust document.

## Syntax

```
selectWSMTokenIssuerTrustDocument (name)
```

Argument	Definition
<i>name</i>	Name of the document to modified in the session. An error is thrown if a name is not provided.

## Examples

In the following example, the `tokenissuertrustWLSbase_domain` document is selected for modification:

```
wls:/wls-domain/serverConfig>
selectWSMTokenIssuerTrustDocument('tokenissuertrustWLSbase_domain')
```

## 6.6.11 setWSMTokenIssuerTrust

The `setWSMTokenIssuerTrust` command is an online command that configures a trusted token issuer and define trusted keys or a trusted DN list for the issuer.

### Description

Configure a trusted token issuer and define trusted keys or a trusted DN list for the issuer. Supported values for a SAML assertion or JWT token type are `dns.hok`, `dns.sv`, or `dns.jwt`. The `trustedKeyIDs` argument is optional. If you do not set this argument, only the trusted issuer will be set for the specified type. This command can be used to specify the DN list associated with a trusted token issuer, update the list, or delete the list.

### Syntax

```
setWSMTokenIssuerTrust(type, issuer, trustedKeyIDs)
```

Argument	Definition
<i>issuer</i>	The name of the trusted issuer, for example <code>www.oracle.com</code> .
<i>type</i>	The type of SAML assertion or JWT tokens the trusted issuer issues: <ul style="list-style-type: none"> <li><code>dns.sv</code> – SAML sender vouches client list</li> <li><code>dns.hok</code> – SAML HOK or Bearer</li> <li><code>dns.jwt</code> – JWT token.</li> </ul>
<i>trustedKeyIDs</i>	Optional. An array of DNs for token signing certificates associated with the issuer for the specified type. This is a comma-separated list with the format [ <code>'CN=name1', 'CN=name2', 'CN=name3', ...</code> ]. If you enter an empty set ( <code>[]</code> ) the list of DN values will be deleted for the issuer.

### Examples

In the following example, `CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US'` is set as a DN in the `dns.sv` DN list for the `www.oracle.com` trusted issuer:

```
wls:/wls-domain/serverConfig>setWSMTokenIssuerTrust('dns.sv', 'www.oracle.com',
['CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US'])
```

In the following example, the name `CN=orcladmin, OU=Doc, O=Oracle, C=US'` in added to the `dns.sv` DN list for the `www.oracle.com` trusted issuer:

```
wls:/wls-domain/serverConfig>setWSMTokenIssuerTrust('dns.sv','www.oracle.com',
['CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US', 'CN=orcladmin,
OU=Doc, O=Oracle, C=US'])
```

In the following example, the list of DN values in the `dns.sv` DN list is removed from the `www.oracle.com` trusted issuer:

```
wls:/wls-domain/serverConfig>setWSMTokenIssuerTrust('dns.sv', 'www.oracle.com', [])
```

## 6.6.12 setWSMTokenIssuerTrustAttributeFilter

The `setWSMTokenIssuerTrustAttributeFilter` command is an online command that specifies token attribute filtering rules for a trusted DN list.

### Description

Specify token attribute filtering rules for a trusted DN list. For each trusted DN configured for an issuer, a token attribute filtering rule can be configured and applied. Each rule has two parts: a name ID and an attributes part for attributes in a SAML assertion or a JWT token. The name ID and each attribute can contain a filter with multiple value patterns. To remove the list of filters for an attribute for the signing certificate, use an empty set (`[]`) for the value of `filters`.



#### Note:

You must first use the `setWSMTokenIssuerTrust` command to configure a list of trusted DN names for an issuer.

### Syntax

```
setWSMTokenIssuerTrustAttributeFilter(dn, attrName, filters)
```

Argument	Definition
<i>dn</i>	The DN of the token signing certificate. The DN as the identifier of the token attribute rule where modifications would be done.
<i>attrName</i>	The name of the user attribute for which the filtering will be applied. The value can be as follows: <ul style="list-style-type: none"> <li><code>name-id</code>—assert a subject name ID.</li> </ul>
<i>filters</i>	Optional. List of filters for the attribute. The list has the format <code>['value1', 'value2', 'value3, ...</code> . Each value can be an exact name or a name pattern with a wildcard character <code>"*"</code> .  When <code>name-id</code> is selected for the <code>attrName</code> argument, then the value of the subject name ID in the incoming SAML assertion must match one of the specified values to go through. If no values are specified, then any value for the subject name ID will go through.

### Examples

In the following example, the name ID `yourTrustedUser` is set as a trusted user for the `weblogic` trusted DN:

```
wls:/wls-domain/serverConfig> setWSMTokenIssuerTrustAttributeFilter('CN=weblogic,
OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US', 'name-id', ['yourTrustedUser'])
```



In the following example, the name IDs `jd` is added to the list of trusted users for the `weblogic` trusted DN:

```
wls:/wls-domain/serverConfig> setWSMTokenIssuerTrustAttributeFilter('CN=weblogic,
OU=Oracle Test Encryption Purposes Only, O=Oracle, C=US', 'name-id', ['yourTrustedUser',
'jd'])
```

In the following example, the list of trusted users for the `weblogic` trusted DN is removed:

```
wls:/wls-domain/serverConfig> setWSMTokenIssuerTrustAttributeFilter('CN=weblogic,
OU=Oracle Test Encryption Purposes Only, O=Oracle, C=US', 'name-id', [])
```

## 6.6.13 setWSMTokenIssuerTrustAttributeMapping

The `setWSMTokenIssuerTrustAttributeMapping` command is an online command that specifies token attribute mapping rules for a trusted DN list.

### Description

Specify token attribute mapping rules for a trusted DN list. For each trusted DN configured for a token issuer, a token attribute mapping rule can be configured and applied. Each rule has two parts: a name ID and an attributes part for attributes associated with a SAML assertion or a JWT token. For a trusted DN, a token attribute mapping rule sets the mapping for the value of an attribute as specified by the `attrName` argument. The `userAttribute` argument is optional and indicates the local user attribute it corresponds to. The `userMappingAttribute` argument is optional and indicates the user attribute to be used in the system to authenticate the users. If the attribute as identified by `attrName` exists for a token attribute rule for the DN, the mapping is overwritten by the new value. For example, in federated environments, where the user subject ID (for example, `mail`) in the token is different from the user attribute (for example, `uid`) for authenticating the same user, the name ID and each attribute can map the local user attribute for the subject name ID to the local user attribute to authenticate a trusted user.



### Note:

You must first use the `setWSMTokenIssuerTrust` command to configure a list of trusted DN names for an issuer.

### Syntax

```
setTokenIssuerTrustAttributeMapping(dn, attrName, userAttribute=None,
userMappingAttribute=None)
```

Arguments	Description
<code>dn</code>	The trusted DN of a token signing certificate.
<code>attrName</code>	The name of the use attribute for which the mapping will be applied The value can be as follows: <ul style="list-style-type: none"> <li><code>name-id</code></li> </ul>
<code>userAttribute</code>	Optional. The local name of the user attribute in the local identity store that the subject name ID corresponds to. The value can be as follows: <ul style="list-style-type: none"> <li><code>mail</code></li> </ul>

Arguments	Description
<i>userMappingAttribute</i>	Optional. The value of the local name of the user attribute in the local identity store that the subject name ID maps to for authentication. The value can be as follows: <ul style="list-style-type: none"> <li>uid</li> </ul>

### Examples

In the following example, the `mail` attribute for the Subject ID in the token is mapped to the `uid` attribute.

```
wls:/base_domain/serverConfig>setTokenIssuerTrustAttributeMapping('CN=weblogic,
OU=Oracle Test Encryption Purposes Only, O=Oracle, C=US', 'name-id', 'mail', 'uid')
```

```
Starting Operation setWSMTokenIssuerTrustAttributeMapping ...
The token attribute mapping are successfully set
```

In the following example, the local user attribute for the Subject ID in the token is mapped to the `uid` attribute.

```
wls:/base_domain/serverConfig>setTokenIssuerTrustAttributeMapping('CN=weblogic,
OU=Oracle Test Encryption Purposes Only, O=Oracle, C=US', 'name-id', '', 'uid')
```

## 6.6.14 setWSMTokenIssuerTrustDisplayName

The `setWSMTokenIssuerTrustDisplayName` command is an online command that sets or resets the display name of the Token Issuer Trust document currently selected in the session.

### Description

Sets or resets the display name of the Token Issuer Trust document currently selected in the session.

You must start a session (`beginWSMSession`) before creating or modifying any token issuer trust documents. If there is no current session or there is already an existing modification process, an error is displayed.

### Syntax

```
setWSMTokenIssuerTrustDisplayName("displayName")
```

Arguments	Definition
<i>displayName</i>	Name to be set as a display name for the document currently selected for modification in the session.

### Examples

In the following example, the display name for the trust document being modified is set to `Test Document`.

```
wls:/wls-domain/serverConfig> setWSMTokenIssuerTrustDisplayName("Test Document")
```

## 6.7 Diagnostic Commands

Use the WLST diagnostic command to check the status of the Oracle components that are required for proper functioning of the product.

### 6.7.1 checkWSMStatus

The checkWSMStatus command is an online command that verifies the status of the Oracle components that are required for proper functioning of the product.

#### Description

Check the status of the Oracle components that are required for proper functioning of the product. The Oracle components that are checked are the policy manager (`wsm-pm`), the agent (`agent`), and the credential store and keystore configuration. The status of the components can be checked together or individually.

#### Note:

The Policy Manager (`wsm-pm`) application must be deployed and running for the check status tool to function correctly.

#### Syntax

```
checkWSMStatus([component=None],[address=None],[verbose=true])
```

Arguments	Description
<i>component</i>	Optional. All checks will be performed if no value is specified. Valid options are: <ul style="list-style-type: none"> <li><code>wsm-pm</code>—Policy Manager. Checks the configuration state of the policy manager component.</li> <li><code>agent</code>—Enforcement Agent. Checks status of end-to-end service-side enforcement through the <code>wsm</code> agent component. The enforcement check is specific only to the environment from which the command is run.</li> <li><code>credstore</code>—Credential Store. Checks whether the credentials are configured for the keystore password, signing, and encryption certificates in the keystore.</li> </ul>
<i>address</i>	Optional. The HTTP URL of the host running the Policy Manager <code>wsm-pm</code> application. This value is required for checking enforcement through an agent component, for example, <pre>checkWSMStatus('agent', 'http://localhost:7001')</pre> <p>The address is not required in the WebLogic Server domain where auto-discovery is present.</p>
<i>verbose</i>	Optional. If the value of this flag is <code>true</code> , then the detailed messages (including stack trace, if any) are displayed. Default is <code>false</code> .

## Examples

In the following example, the `checkWSMStatus` command is run without arguments. The status of the credential store, policy manager, and enforcement agent is returned.

```
wls:/base_domain/serverConfig> checkWSMStatus ()
```

Credential Store Configuration:

PASSED.

Message(s):

keystore.pass.csf.key : Property is configured and its value is "keystore-csf-key".

Description: The "keystore.pass.csf.key" property points to the CSF alias that is mapped to the username and password of the keystore. Only the password is used; username is redundant in the case of the keystore.

keystore-csf-key : Credentials configured.

keystore.sig.csf.key : Property is configured and its value is "sign-csf-key".

Description: The "keystore.sig.csf.key" property points to the CSF alias that is mapped to the username and password of the private key that is used for signing.

sign-csf-key : Credentials configured.

Sign Key : Key configured.

Alias - orakey

Sign Certificate : Certificate configured.

Alias - CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle,

C=US

Expiry - June 28, 2020 11:17:12 AM PDT

keystore.enc.csf.key : Property is configured and its value is "enc-csf-key".

Description: The "keystore.enc.csf.key" property points to the CSF alias that is mapped to the username and password of the private key that is used for decryption.

enc-csf-key : Credentials configured.

Encrypt Key : Key configured.

Alias - orakey

Encrypt Certificate : Certificate configured.

Alias - CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle,

C=US

Expiry - June 28, 2020 11:17:12 AM PDT

Policy Manager:

PASSED.

Message(s):

OWSM Policy Manager connection state is OK.

OWSM Policy Manager connection URL is "host.example.com:1234".

Enforcement Agent:

PASSED.

Message(s):

Enforcement is successful.

Service URL: http://host.example.com:7001/Diagnostic/DiagnosticService?wsdl

In the following example, the credential store key `keystore-csf-key` is deleted and the `checkWSMStatus` command is rerun for the credential store `credstore`. The status check fails because the csf-key `keystore-csf-key` is not present in the credential store:

```
wls:/base_domain/serverConfig> deleteCred(map="oracle.wsm.security", key="keystore-csf-key")
wls:/base_domain/serverConfig> checkWSMStatus('credstore')
```

Credential Store Configuration:

FAILED.

Message(s):

keystore.pass.csf.key : Property is configured and its value is "keystore-csf-key".

Description: The "keystore.pass.csf.key" property points to the CSF alias that is mapped to the username and password of the keystore. Only the password is used; username is redundant in the case of the keystore.

keystore-csf-key : Credentials not configured.

Credential Store Diagnostic Messages:

Message(s):

The csf-key keystore-csf-key is not present in the credential store.

Perform the following steps to update the credential store (using WLST commands):-

1. connect()
2. createCred(map="oracle.wsm.security", key="keystore-csf-key", user="keystore-csf-key", password="<keystore-password>", desc="Keystore Password CSF Key")

NOTE:- All the above commands are based on the Domain level configurations. The actual csf key may be overridden at runtime due to config override. See Documentation for more details.

In the following example, the csf-key keystore-csf-key is configured and the checkWSMStatus command is rerun. The configuration check passes.

```
wls:/base_domain/serverConfig> createCred(map="oracle.wsm.security", key="keystore-csf-key", user="keystore-csf-key", password="welcome1", desc="Keystore Password CSF Key")
Already in Domain Runtime Tree
```

```
wls:/base_domain/serverConfig> checkWSMStatus('credstore')
```

Credential Store Configuration:

PASSED.

Message(s):

keystore.pass.csf.key : Property is configured and its value is "keystore-csf-key".

Description: The "keystore.pass.csf.key" property points to the CSF alias that is mapped to the username and password of the keystore. Only the password is used; username is redundant in the case of the keystore.

keystore-csf-key : Credentials configured.

keystore.sig.csf.key : Property is configured and its value is "sign-csf-key".

Description: The "keystore.sig.csf.key" property points to the CSF alias that is mapped to the username and password of the private key that is used for signing.

sign-csf-key : Credentials configured.

Sign Key : Key configured.

Alias - orakey

Sign Certificate : Certificate configured.

Alias - CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle,

C=US

Expiry - June 28, 2020 11:17:12 AM PDT

keystore.enc.csf.key : Property is configured and its value is "enc-csf-key".

Description: The "keystore.enc.csf.key" property points to the CSF

```
alias that is mapped to the username and password of the private key that is used for
decryption.
    enc-csf-key : Credentials configured.
    Encrypt Key : Key configured.
        Alias - orakey
    Encrypt Certificate : Certificate configured.
        Alias - CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle,
C=US
        Expiry - June 28, 2020 11:17:12 AM PDT
true
```

The following example checks the enforcement status of the agent component at the URL `http://localhost:7001`.

```
wls:/test_domain1/serverConfig> checkWSMStatus('agent','http://localhost:7001')
```

```
Enforcement Agent:
```

Note: Enforcement might succeed if OWSM Policy Manager is down due to policy caching. For such scenarios wsm-pm test must be run prior to this test.

```
PASSED.
```

```
Message(s):
```

```
Enforcement is successful.
```

```
Service URL: http://localhost:7001/Diagnostic/DiagnosticService?wsdl
```