

MS-LISTSWS Test Suite Specification

**Abstract:** This document provides information about how to configure the test suite and how the MS-LISTSWS test suite is designed to test the MS-LISTSWS Open Specification usability and accuracy. It describes test assumptions, scope and constraints of the test suite. It also specifies test scenarios, detail test cases, test suite architecture and adapter design.

Contents

[1 Configuring the test suite 4](#_Toc356305936)

[1.1 Configuring the test suite client 4](#_Toc356305937)

[1.1.1 Configuring the test suite client manually 4](#_Toc356305938)

[1.1.2 Configuring the test suite client by scripts 4](#_Toc356305939)

[1.2 Configuring the system under test (SUT) 5](#_Toc356305940)

[1.2.1 Configuring the SUT manually 5](#_Toc356305941)

[1.2.2 Configuring the SUT by scripts 5](#_Toc356305942)

[1.3 Configuring the SHOULD/MAY requirements 5](#_Toc356305943)

[2 Test suite design 5](#_Toc356305944)

[2.1 Assumptions, scope and constraints 5](#_Toc356305945)

[Assumptions 5](#_Toc356305946)

[Scope 6](#_Toc356305947)

[In scope: 6](#_Toc356305948)

[Out of scope: 6](#_Toc356305949)

[Constrains 6](#_Toc356305950)

[2.2 Test suite architecture 6](#_Toc356305951)

[2.3 Technical dependencies and considerations 8](#_Toc356305952)

[Dependencies 8](#_Toc356305953)

[Encryption consideration 8](#_Toc356305954)

[2.4 Adapter design 8](#_Toc356305955)

[2.4.1 Adapter overview 8](#_Toc356305956)

[Protocol adapter 8](#_Toc356305957)

[SUT control adapters 8](#_Toc356305958)

[2.4.2 Technical feasibility of adapter approach 10](#_Toc356305959)

[Message generation 10](#_Toc356305960)

[Message consumption 10](#_Toc356305961)

[SUT control adapter 10](#_Toc356305962)

[2.4.3 Adapter abstract layer 10](#_Toc356305963)

[Protocol adapters 10](#_Toc356305964)

[MS-LISTSWS Adapter Interface 10](#_Toc356305965)

[SUT control adapters 10](#_Toc356305966)

[SUT control adapter Interface 10](#_Toc356305967)

[2.4.4 Adapter details 10](#_Toc356305968)

[*2.4.4.1* Protocol adapter 10](#_Toc356305969)

[2.4.4.1.1 MS-LISTSWS protocol adapter 10](#_Toc356305970)

[Adapter interface 11](#_Toc356305971)

[Adapter implementation 12](#_Toc356305972)

[Other class 12](#_Toc356305973)

[Enumeration 13](#_Toc356305974)

[*2.4.4.2* SUT control adapter 14](#_Toc356305975)

[2.4.4.2.1 MS-LISTSWS SUT control adapter 14](#_Toc356305976)

[2.5 Test scenarios 14](#_Toc356305977)

[2.5.1 S01\_OperationOnList 15](#_Toc356305978)

[Description 15](#_Toc356305979)

[Operations 15](#_Toc356305980)

[Prerequisites 15](#_Toc356305981)

[Cleanup 15](#_Toc356305982)

[2.5.2 S02\_OperationOnContentType 15](#_Toc356305983)

[Description 15](#_Toc356305984)

[Operations 15](#_Toc356305985)

[Prerequisites 15](#_Toc356305986)

[Cleanup 16](#_Toc356305987)

[2.5.3 S03\_OperationOnListItem 16](#_Toc356305988)

[Description 16](#_Toc356305989)

[Operations 16](#_Toc356305990)

[Prerequisites 16](#_Toc356305991)

[Cleanup 16](#_Toc356305992)

[2.5.4 S04\_OperationOnAttachment 16](#_Toc356305993)

[Description 16](#_Toc356305994)

[Operations 16](#_Toc356305995)

[Prerequisites 16](#_Toc356305996)

[Cleanup 16](#_Toc356305997)

[2.5.5 S05\_OperationOnFiles 16](#_Toc356305998)

[Description 16](#_Toc356305999)

[Operations 17](#_Toc356306000)

[Prerequisites 17](#_Toc356306001)

[Cleanup 17](#_Toc356306002)

[2.6 Test case design 17](#_Toc356306003)

[2.6.1 Traditional test case design 17](#_Toc356306004)

[2.6.2 Test case description 27](#_Toc356306005)

# Configuring the test suite

## Configuring the test suite client

### Configuring the test suite client manually

Before you run the test suite, update the values in the MS-LISTSWS\_TestSuite.deployment.ptfconfig file. The MS-LISTSWS\_TestSuite.deployment.ptfconfig file can also be configured by running the client setup script.

1. Open MS-LISTSWS\TestSuite\MS-LISTSWS\_TestSuite.deployment.ptfconfig file.
2. Update the following value to specify the common configuration file.

Property name="CommonConfigurationFileName" value="SharePointCommonConfiguration.deployment.ptfconfig"

**Note**   This property can be removed or set to empty if the required properties are copied to the test suite specific configuration file. Any other changes to this property will cause all test cases in the test suite to fail during execution. The test suite first search through its specific configuration file and use the properties from there if they are defined, before looking for them from the common configuration file (if specified).

1. Update the following properties' values to match SUT settings and configuration.

* Property name="TargetServiceUrl" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/\_vti\_bin/lists.asmx"
* Property name="SiteCollectionName" value="MSLISTSWS\_SiteCollection"

1. The following properties are not associated with SUT settings and can normally retain with default values.

* Property name="ServiceTimeOut" value="10"
* Property name="MessageDataFileName" value="Message.txt"
* Property name="ListFieldText" value="EmployeeName"
* Property name="ListFieldCounter" value="Age"
* Property name="ListFeatureId" value="{00BFEA71-DE22-43B2-A848-C05709900100}"
* Property name="PropertyPrefix" value="vti\_"
* Property name="ContentTypeXmlQualifiedName" value="XmlElement1"
* Property name="ContentTypeXmlNamespaceUri" value="http://schemas.microsoft.com/sharepoint/soap/namespaceUri1/"
* Property name="ValidContentTypeXmlDocument" value="<DemoNode1 xmlns='http://schemas.microsoft.com/sharepoint/soap/namespaceUri1/'></DemoNode1>"
* Property name="AttachmentName" value="Attachment.txt"
* Property name="UserInfoListName" value="User Information List"

### Configuring the test suite client by scripts

To configure the test suite client using scripts, see section 5.2.4 of the [SharePointTestSuiteDeploymentGuide.docx.](../SharePointTestSuiteDeploymentGuide.docx)

## Configuring the system under test (SUT)

### Configuring the SUT manually

To manually configure the SUT, see section 5.1.3 of the [SharePointTestSuiteDeploymentGuide.docx](../SharePointTestSuiteDeploymentGuide.docx).

### Configuring the SUT by scripts

To configure the SUT using scripts, see section 5.1.2 of the [SharePointTestSuiteDeploymentGuide.docx.](../SharePointTestSuiteDeploymentGuide.docx)

## Configuring the SHOULD/MAY requirements

All the implementation of the SHOULD/MAY and endnote related requirements are pre-configured in the format "<Property name="RXXXEnabled" value="XXXX"/>" for six Microsoft product versions in six SHOULD/MAY PTFConfig files:

* MS-LISTSWS\_WindowsSharePointServices3\_SHOULDMAY.deployment.ptfconfig
* MS-LISTSWS\_SharePointServer2007\_SHOULDMAY.deployment.ptfconfig
* MS-LISTSWS\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig
* MS-LISTSWS\_SharePointServer2010\_SHOULDMAY.deployment.ptfconfig
* MS-LISTSWS\_SharePointFoundation2013\_SHOULDMAY.deployment.ptfconfig
* MS-LISTSWS\_SharePointServer2013\_SHOULDMAY.deployment.ptfconfig

If RXXXEnabled is set to true, the requirement must be checked. If false, the requirement must not be checked. For Microsoft product versions, all values should not be changed. For third-party products, the closest Microsoft product version should be chosen, and the value of RXXXEnabled should be updated according to the real product behavior. For example, if SharePoint Foundation 2010 is chosen,user can open **MS-LISTSWS\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig** and update the RXXXEnabled accordingly.

# Test suite design

## Assumptions, scope and constraints

Assumptions

* MS-LISTSWS test suite assumes that the SOAP messages generated by the proxy class completely follow all the definitions of elements in the Open Specifications.
* The proxy class is the main part of the adapter in the test suite and it is generated from MS-LISTSWS.wsdl which is derived from the Full WSDL in the Open Specifications by WSDL tool in Visual Studio.

Scope

In scope:

* This test suite will verify the accuracy and integrity of the technical content in the Open Specification against the results returned from the protocol server by using MS-LISTSWS operations.
* This test suite will verify the Full WSDL which is provided in the Open Specification.
* This test suite will verify the server-side and testable requirements by running all the test cases on both HTTP and HTTPS.
* This test suite will verify operations over SOAP 1.1 and SOAP 1.2.

Out of scope:

* This test suite will not verify the requirements related to client behaviors.
* This test suite will not verify the requirements related to server internal behaviors.
* This test suite will not verify the internal implementations of its transport protocol stack.

Constrains

None.

## Test suite architecture

This test suite verifies the server-side and testable requirements obtained from the Open Specification. The following figure shows the architecture of this test suite.

 **MS-LISTSWS test suite architecture**

The details of the MS-LISTSWS test suite architecture

* SUT hosts the Versions Web Service which this test suite runs against.
* From third-party user’s point of view, SUT is the protocol server implementation.
* The following products have been tested with the MS-LISTSWS test suite on Windows platform.
* Windows SharePoint Services 3.0 Service Pack 3 (SP3)
* Microsoft SharePoint Foundation 2010 Service Pack 2 (SP2)
* Microsoft SharePoint Foundation 2013 SP1
* Microsoft Office SharePoint Server 2007 Service Pack 3 (SP3)
* Microsoft SharePoint Server 2010 Service Pack 2 (SP2)
* Microsoft SharePoint Server 2013 SP1
* Test suite acts as the client to communicate with the SUT and validates the requirements gathered from MS-LISTSWS Open Specification.
* Test cases use the MS-LISTSWS adapter to call and get the results of the MS-LISTSWS operations. Test cases also use the SUT control adapter and the LISTSWS SUT control adapter to set the SUT to the test case specific situation.
* MS-LISTSWS adapter is used in the test cases. The test cases call the methods in the interfaces to invoke the MS-LISTSWS operations.
* SUT control adapter is used in the test cases. The test cases call the methods in the interfaces to configure the SUT.

## Technical dependencies and considerations

Dependencies

* This test suite depends on the SOAP messaging protocol for exchanging structured data and type information.
* This test suite depends on HTTP protocol or HTTPS protocol to transmit the messages.
* This test suite depends on the wsdl.exe tool in the .NET Framework SDK to generate the MS-LISTSWS proxy class.
* This test suite depends on the Protocol Test Framework (PTF) to derive managed adapters.

Encryption consideration

Transportation of MS-LISTSWS includes HTTP and HTTPS, and encryption will be handled by HTTPS.

## Adapter design

### Adapter overview

One protocol adapter and two SUT control adapters will be designed for this test suite.

Protocol adapter

* MS-LISTSWS adapter
* The MS-LISTSWS Adapter is a managed adapter, which is derived from the ManagedAdapterBase class in the Protocol Test Framework (PTF).
* MS-LISTSWS Adapter Interface declares the interface of the MS-LISTSWS Adapter.
* The MS-LISTSWS Adapter Implementation implements the interface mentioned above. It directly uses the methods and the data structures in the MS-LISTSWS proxy class.
* The MS-LISTSWS adapter has the following functionalities
* Choose HTTP or HTTPS and SOAP 1.1 or 1.2 for transport;
* Construct requests of 32 MS-LISTSWS operations;
* Communicate with the SUT by sending requests to the SUT and receive the corresponding responses from the SUT;
* Parse the response messages and validate the messages according to the WSDL schema;
* Generate the result log.
* The MS-LISTSWS Adapter uses the C# proxy class which is generated by running the wsdl.exe tool against the full WSDL of this protocol to send SOAP request messages and receive SOAP response messages. The wsdl.exe can be found in Microsoft .NET Framework SDK tools.

SUT control adapters

* SUT Control Adapter
* The SUT control adapter will be a PowerShell scripted adapter.
* The SUT control adapter has the following functionalities
* Upload files into a document library on the site
* Move file from source location to destination location on the site
* Set Custom Send To Destination Name and Url for document library list.
* Get the value of Presence setting of a Web application.
* Get the value of Recycle Bin setting of a Web application.
* Set the value of Presence setting of a Web application.
* Set the value of Recycle Bin setting of a Web application.
* Set MajorVersionLimit and MajorWithMinorVersionsLimit value when versioning is enabled in a list.
* Get the RootFolder value of a list.
* The SUT control adapter is invoked by the test cases.

The following figure shows the inner working among Test Case, protocol adapter, proxy class and SUT.

 **Sequence diagram**

* The test case invokes protocol adapter to initialize MS-LISTSWS service.
* The test case invokes SUT control adapter method.
* The SUT control adapter configures the SUT.
* The test case sends the abstract parameters to drive the protocol adapter to communicate with the web service.
* The Protocol adapter transfers the real parameters to the proxy class to invoke the web service method.
* The proxy class sends the SOAP messages over a network to the SUT.
* The proxy class receives the SOAP messages from the SUT.
* The proxy class sends the elements contained in the messages to the protocol adapter.
* The Protocol adapter transfers the returned values to the test case.
* The test case compares the returned values with its expected values to capture related requirements.

### Technical feasibility of adapter approach

Message generation

The MS-LISTSWS adapter gets the parameter values of the WSDL operations and calls the corresponding operations in MS-LISTSWS proxy class, the MS-LISTSWS proxy class serializes the parameter values to XML elements to format the SOAP request messages, then the SOAP request messages are sent out by the MS-LISTSWS Proxy class.

Message consumption

The messages received from the SUT will be parsed in the MS-LISTSWS proxy class and be passed upon to the MS-LISTSWS adapter. Then these messages are consumed in the MS-LISTSWS adapter to validate the message format and to validate the logic-related requirements in the test cases.

SUT control adapter

The SUT control adapter is designed to remotely configure the SUT and upload files into a document library on the site.

### Adapter abstract layer

Protocol adapters

MS-LISTSWS Adapter Interface

There are 32 methods declared in the MS-LISTSWS adapter interface IMS\_LISTSWSAdapter.

32 of the methods correspond to the 32 MS-LISTSWS operations. The operators of the 32 methods are abstracted the same as the operations specified in the MS-LISTSWS.

SUT control adapters

SUT control adapter Interface

There are nine methods declared in the SUT control adapter interface ISUTControlAdapter.

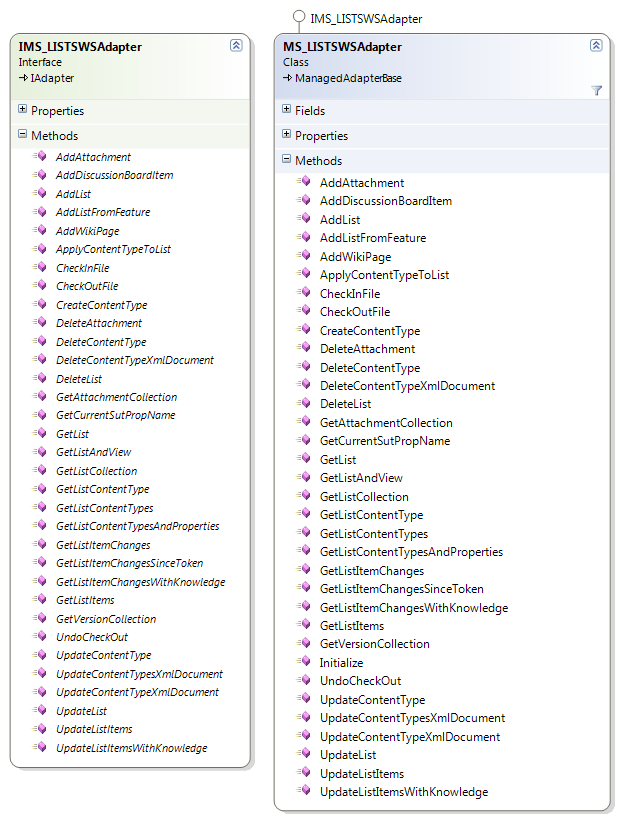
* Upload files into a document library on the site
* Move file from source location to destination location on the site
* Set Custom Send To Destination Name and Url for document library list.
* Get the value of Presence setting of a Web application.
* Get the value of Recycle Bin setting of a Web application.
* Set the value of Presence setting of a Web application.
* Set the value of Recycle Bin setting of a Web application.
* Set MajorVersionLimit and MajorWithMinorVersionsLimit value when versioning is enabled in a list.

### Adapter details

#### Protocol adapter

##### MS-LISTSWS protocol adapter

The following figure shows the class diagram of the MS-LISTSWS protocol adapter.



**Protocol adapter class diagram**

The following outlines details of the class diagram

Adapter interface

IMS\_LISTSWSAdapter is the interface of the protocol adapter.

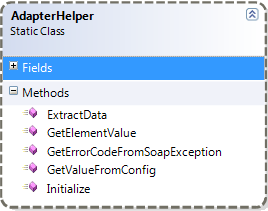
IMS\_LISTSWSAdapter defines the 32 methods which are specified in Open Specification invoked by test cases.

Adapter implementation

* MS\_LISTSWSAdapter is the protocol adapter class of the test suite. It is used to implement IMS\_LISTSWSAdapter.
* The 32 methods defined in IMS\_LISTSWSAdapter are implemented by generating SOAP requests, and then invoking these methods provided by the MS-LISTSWS proxy class to send SOAP requests; getting corresponding de-serialized response and verifying related adapter requirements.
* The Initialize method is used to configure HTTP or HTTPS and SOAP 1.1 or SOAP 1.2 for transport.
* The Reset method is used to reset the MS\_LISTSWSAdapter.

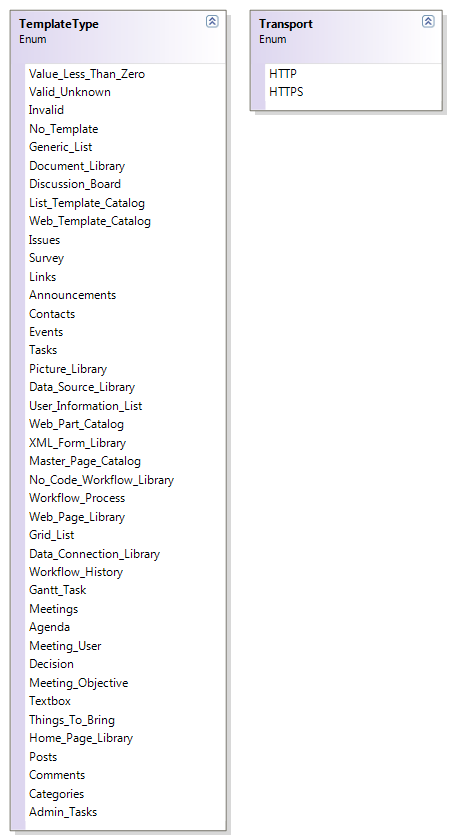
Other class

* ListsSoap is a schema validation class, which is a part of the proxy generated from the MS-LISTSWS WSDL. It overrides the Invoke method to generate a schema validation reader, and then use the kind of mechanism to validate whether the result xml is consistent with the XSD.
* The AdapterHelper class provides the methods of getting the property in configuration file, verifying the format of SOAP Fault error code, GUID and URL.



**AdapterHelper class diagram**

Enumeration



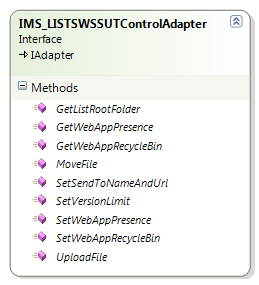
**Enumeration diagram**

* Enum templatetype indicates the server’s list template types.
* Enum transport indicates the possible types of transports used by this protocol: HTTP or HTTPS.

#### SUT control adapter

##### MS-LISTSWS SUT control adapter

The following figure shows the class diagram of the SUT control adapter.



**SUT control adapter class diagram**

The following outlines details of the class diagram

The IMS-LISTSWSSUTControlAdapter is the interface of the SUT control adapter which is implemented by Microsoft PowerShell script. The implementation can be substituted by other implementation for the third party’s need.

## Test scenarios

Five scenarios are designed to cover the testable requirements in MS-LISTSWS test suites. The details of five scenarios are as follows:

|  |  |
| --- | --- |
| Scenario | Description |
| [S01\_OperationOnList](#S1OL) | Implement the operations on lists and list collection. |
| [S02\_OperationOnContentType](#S2I) | Implement the operations on content types and content type XML documents. |
| [S03\_OperationOnListItem](#S3) | Implement the operations on list items. |
| [S04\_OperationOnAttachment](#S4) | Implement the operations on attachments. |
| [S05\_OperationOnFiles](#S05) | Implement the operations on files. |

MS-LISTSWS traditional testing covered scenarios

### S01\_OperationOnList

Description

This scenario is designed to test add/delete/get/update operations on lists and list collection with valid or invalid input parameters.

Operations

* AddList
* AddListFromFeature
* DeleteList
* GetList
* GetListAndView
* GetListCollection
* UpdateList

Prerequisites

N/A

Cleanup

N/A

### S02\_OperationOnContentType

Description

This scenario is designed to test apply/create/delete/get/update operations on content types and content type XML with valid or invalid input parameters.

Operations

* ApplyContentTypeToList
* CreateContentType
* DeleteContentType
* DeleteContentTypeXmlDocument
* GetListContentType
* GetListContentTypes
* GetListContentTypesAndProperties
* UpdateContentType
* UpdateContentTypeXMLDocument
* UpdateContentTypesXmlDocument

Prerequisites

N/A

Cleanup

N/A

### S03\_OperationOnListItem

Description

This scenario is designed to test get/update operations on list items.

Operations

* GetVersionCollection
* UpdateListItem
* GetListItemChanges
* GetListItmes
* UpdateListItemsWithKnowledge
* GetListItemChangesSinceToken
* GetListItemChangesWithKnowledge

Prerequisites

N/A

Cleanup

N/A

### S04\_OperationOnAttachment

Description

This scenario is designed to test add/get/delete operations on attachments.

Operations

* AddAttachment
* GetAttachmentCollection
* DeleteAttachment

Prerequisites

N/A

Cleanup

N/A

### S05\_OperationOnFiles

Description

This scenario is designed to test check in/checkout/undo checkout operations on files.

Operations

* CheckInFile
* CheckOutFile
* UndoCheckOut

Prerequisites

N/A

Cleanup

N/A

## Test case design

### Traditional test case design

Traditional Testing approach is selected as the test approach for this test suite. 224 test cases are designed to cover the server-side and testable requirements.

224 traditional test cases are designed to cover five scenarios mentioned in section [2.5 Test scenarios](#Testscenarios). Details of test case are specified in section [2.6.2 Test case description](#SEC262). Test case distributed among the scenario is listed in the following table:

|  |  |
| --- | --- |
| Scenario ID | Test case name |
| S01\_OperationOnList | [MSLISTSWS\_S01\_TC01\_AddListFromFeature\_Succeed](#S1_TC01) |
| [MSLISTSWS\_S01\_TC02\_AddListFromFeature\_EmptyFeatureID](#S1_TC02) |
| [MSLISTSWS\_S01\_TC03\_AddListFromFeature\_InvalidTemplateId](#S1_TC03) |
| [MSLISTSWS\_S01\_TC04\_AddListFromFeature\_UnknownTemplateId](#S1_TC04) |
| [MSLISTSWS\_S01\_TC05\_AddList\_Succeed](#S1_TC05) |
| [MSLISTSWS\_S01\_TC06\_AddList\_AlreadyUsedListName\_UnknownTemplateID](#S1_TC06) |
| [MSLISTSWS\_S01\_TC07\_AddList\_Negative](#S1_TC07) |
| [MSLISTSWS\_S01\_TC08\_AddList\_UniqueTemplate](#S1_TC08) |
| [MSLISTSWS\_S01\_TC09\_AuthorizationFaultsTest](#S1_TC09) |
| [MSLISTSWS\_S01\_TC10\_DeleteList\_Succeed](#S1_TC10) |
| [MSLISTSWS\_S01\_TC11\_DeleteList\_NonExistentListName](#S1_TC11) |
| [MSLISTSWS\_S01\_TC12\_DeleteNonExistentList\_WSS3](#S1_TC12) |
| [MSLISTSWS\_S01\_TC13\_GetListAndView\_Succeed](#S1_TC13) |
| [MSLISTSWS\_S01\_TC14\_GetListAndView\_InvalidGUID](#S1_TC14) |
| [MSLISTSWS\_S01\_TC15\_GetListAndView\_InvalidListName](#S1_TC15) |
| [MSLISTSWS\_S01\_TC16\_GetListAndView\_InvalidViewName](#S1_TC16) |
| [MSLISTSWS\_S01\_TC17\_GetListAndView\_WSS3](#S1_TC17) |
| [MSLISTSWS\_S01\_TC18\_GetListCollection\_Succeed](#S1_TC18) |
| [MSLISTSWS\_S01\_TC19\_GetList\_Succeed](#S1_TC19) |
| [MSLISTSWS\_S01\_TC20\_GetList\_NonExistentListName](#S1_TC20) |
| [MSLISTSWS\_S01\_TC21\_GetList\_WSS3](#S1_TC21) |
| [MSLISTSWS\_S01\_TC22\_UpdateList\_FailureErrorCodeInUpdateListFieldResults](#S1_TC22) |
| [MSLISTSWS\_S01\_TC23\_UpdateList\_InvalidListName](#S1_TC23) |
| [MSLISTSWS\_S01\_TC24\_UpdateList\_InvalidListVersion\_CanNotBeConvertedToInteger](#S1_TC24) |
| [MSLISTSWS\_S01\_TC25\_UpdateList\_InvalidListVersion\_MismatchedNumeric](#S1_TC25) |
| [MSLISTSWS\_S01\_TC26\_UpdateList\_InvalidListVersion\_OutOfUInt32](#S1_TC26) |
| [MSLISTSWS\_S01\_TC27\_UpdateList\_InvalidViewNameInUpdateListFieldResults](#S1_TC27) |
| [MSLISTSWS\_S01\_TC28\_UpdateList\_ListDefinitionCT](#S1_TC28) |
| [MSLISTSWS\_S01\_TC29\_UpdateList\_Succeed](#S1_TC29) |
| [MSLISTSWS\_S01\_TC30\_UpdateList\_WSS3](#S1_TC30) |
| S02\_OperationOnContentType | [MSLISTSWS\_S02\_TC01\_ApplyContentTypeToList\_IncorrectContentTypeID](#S2_TC01) |
| [MSLISTSWS\_S02\_TC02\_ApplyContentTypeToList\_InvalidListName](#S2_TC02) |
| [MSLISTSWS\_S02\_TC03\_ApplyContentTypeToList\_InvalidListName\_WSS3](#S2_TC03) |
| [MSLISTSWS\_S02\_TC04\_ApplyContentTypeToList\_WithListGuid\_Succeed](#S2_TC04) |
| [MSLISTSWS\_S02\_TC05\_ApplyContentTypeToList\_WithListTitle\_Succeed](#S2_TC05) |
| [MSLISTSWS\_S02\_TC06\_CreateContentType\_ContentTypeOrderNotReturned\_WSS3](#S2_TC06) |
| [MSLISTSWS\_S02\_TC07\_CreateContentType\_ErrorFieldName](#S2_TC07) |
| [MSLISTSWS\_S02\_TC08\_CreateContentType\_ErrorPropertyName](#S2_TC08) |
| [MSLISTSWS\_S02\_TC09\_CreateContentType\_IncorrectDisplayName](#S2_TC09) |
| [MSLISTSWS\_S02\_TC10\_CreateContentType\_InvalidListName](#S2_TC10) |
| [MSLISTSWS\_S02\_TC11\_CreateContentType\_InvalidListName\_WSS3](#S2_TC11) |
| [MSLISTSWS\_S02\_TC12\_CreateContentType\_InvalidParentType](#S2_TC12) |
| [MSLISTSWS\_S02\_TC13\_CreateContentType\_UseListGuid\_Succeed](#S2_TC13) |
| [MSLISTSWS\_S02\_TC14\_CreateContentType\_UseListTitle\_Succeed](#S2_TC14) |
| [MSLISTSWS\_S02\_TC15\_DeleteContentTypeXmlDocument\_ContentTypeCannotBeFound](#S2_TC15) |
| [MSLISTSWS\_S02\_TC16\_DeleteContentTypeXmlDocument\_InvalidListName](#S2_TC16) |
| [MSLISTSWS\_S02\_TC17\_DeleteContentTypeXmlDocument\_InvalidListName\_WSS3](#S2_TC17) |
| [MSLISTSWS\_S02\_TC18\_DeleteContentTypeXmlDocument\_NonExistentListName](#S2_TC18) |
| [MSLISTSWS\_S02\_TC19\_DeleteContentTypeXmlDocument\_Succeed](#S2_TC19) |
| [MSLISTSWS\_S02\_TC20\_DeleteContentTypeXmlDocument\_Success](#S2_TC20) |
| [MSLISTSWS\_S02\_TC21\_DeleteContentType\_ContentTypeCannotBeFound](#S2_TC21) |
| [MSLISTSWS\_S02\_TC22\_DeleteContentType\_IncorrectListName](#S2_TC22) |
| [MSLISTSWS\_S02\_TC23\_DeleteContentType\_InvalidListName\_WSS3](#S2_TC23) |
| [MSLISTSWS\_S02\_TC24\_DeleteContentType\_NonExistentContentType](#S2_TC24) |
| [MSLISTSWS\_S02\_TC25\_DeleteContentType\_UseListTitle\_Succeed](#S2_TC25) |
| [MSLISTSWS\_S02\_TC26\_DeleteContentType\_WithListGuid\_Succeed](#S2_TC26) |
| [MSLISTSWS\_S02\_TC27\_GetListContentType\_IncorrectContentType](#S2_TC27) |
| [MSLISTSWS\_S02\_TC28\_GetListContentType\_IncorrectListName](#S2_TC28) |
| [MSLISTSWS\_S02\_TC29\_GetListContentType\_InvalidListName\_WSS3](#S2_TC29) |
| [MSLISTSWS\_S02\_TC30\_GetListContentType\_UseListTitleAsListName](#S2_TC30) |
| [MSLISTSWS\_S02\_TC31\_GetListContentType\_UseValidGuidAsListName](#S2_TC31) |
| [MSLISTSWS\_S02\_TC32\_GetListContentTypesAndProperties\_Fault](#S2_TC32) |
| [MSLISTSWS\_S02\_TC33\_GetListContentTypesAndProperties\_ContentTypeVersion](#S2_TC33) |
| [MSLISTSWS\_S02\_TC34\_GetListContentTypesAndProperties\_IncorrectListName](#S2_TC34) |
| [MSLISTSWS\_S02\_TC35\_GetListContentTypesAndProperties\_OnUserInfoList](#S2_TC35) |
| [MSLISTSWS\_S02\_TC36\_GetListContentTypesAndProperties\_Succeed\_BestMatchAndPrefix](#S2_TC36) |
| [MSLISTSWS\_S02\_TC37\_GetListContentTypesAndProperties\_Succeed\_WithGuid](#S2_TC37) |
| [MSLISTSWS\_S02\_TC38\_GetListContentTypesAndProperties\_Succeed\_WithListName](#S2_TC38) |
| [MSLISTSWS\_S02\_TC39\_GetListContentTypes\_VersionTest](#S2_TC39) |
| [MSLISTSWS\_S02\_TC40\_GetListContentTypes\_BestMatch](#S2_TC40) |
| [MSLISTSWS\_S02\_TC41\_GetListContentTypes\_GetBestMatch](#S2_TC41) |
| [MSLISTSWS\_S02\_TC42\_GetListContentTypes\_IncorrectListName](#S2_TC42) |
| [MSLISTSWS\_S02\_TC43\_GetListContentTypes\_InvalidListName\_WSS3](#S2_TC43) |
| [MSLISTSWS\_S02\_TC44\_GetListContentTypes\_NoBestMatch](#S2_TC44) |
| [MSLISTSWS\_S02\_TC45\_GetListContentTypes\_OnUserInfoList](#S2_TC45) |
| [MSLISTSWS\_S02\_TC46\_GetListContentTypes\_Succeed\_BestMatch](#S2_TC46) |
| [MSLISTSWS\_S02\_TC47\_GetListContentTypes\_Succeed\_WithGuid](#S2_TC47) |
| [MSLISTSWS\_S02\_TC48\_GetListContentTypes\_Succeed\_WithListName](#S2_TC48) |
| [MSLISTSWS\_S02\_TC49\_UpdateContentTypeXMLDocument\_IncorrectContentTypeId](#S2_TC49) |
| [MSLISTSWS\_S02\_TC50\_UpdateContentTypeXMLDocument\_IncorrectListName](#S2_TC50) |
| [MSLISTSWS\_S02\_TC51\_UpdateContentTypeXMLDocument\_WithListGuid\_Succeed](#S2_TC51) |
| [MSLISTSWS\_S02\_TC52\_UpdateContentTypeXMLDocument\_WithListTitle\_Succeed](#S2_TC52) |
| [MSLISTSWS\_S02\_TC53\_UpdateContentTypeXmlDocument\_InvalidListName\_WSS3](#S2_TC53) |
| [MSLISTSWS\_S02\_TC54\_UpdateContentTypeXmlDocument\_Success](#S2_TC54) |
| [MSLISTSWS\_S02\_TC55\_UpdateContentType\_AddFieldsError](#S2_TC55) |
| [MSLISTSWS\_S02\_TC56\_UpdateContentType\_Combination](#S2_TC56) |
| [MSLISTSWS\_S02\_TC57\_UpdateContentType\_DeleteFieldsError](#S2_TC57) |
| [MSLISTSWS\_S02\_TC58\_UpdateContentType\_FieldExists](#S2_TC58) |
| [MSLISTSWS\_S02\_TC59\_UpdateContentType\_FieldNotExists](#S2_TC59) |
| [MSLISTSWS\_S02\_TC60\_UpdateContentType\_IncorrectContentTypeId](#S2_TC60) |
| [MSLISTSWS\_S02\_TC61\_UpdateContentType\_IncorrectListName](#S2_TC61) |
| [MSLISTSWS\_S02\_TC62\_UpdateContentType\_InvalidListName\_WSS3](#S2_TC62) |
| [MSLISTSWS\_S02\_TC63\_UpdateContentType\_UpdateFieldsError](#S2_TC63) |
| [MSLISTSWS\_S02\_TC64\_UpdateContentType\_UpdatePropertiesError](#S2_TC64) |
| [MSLISTSWS\_S02\_TC65\_UpdateContentType\_WithGuid\_Succeed](#S2_TC65) |
| [MSLISTSWS\_S02\_TC66\_UpdateContentType\_WithListTitle\_Succeed](#S2_TC66) |
| [MSLISTSWS\_S02\_TC67\_UpdateContentTypesXmlDocument\_IncorrectContentTypeId](#S2_TC67) |
| [MSLISTSWS\_S02\_TC68\_UpdateContentTypesXmlDocument\_IncorrectListName](#S2_TC68) |
| [MSLISTSWS\_S02\_TC69\_UpdateContentTypesXmlDocument\_WithGuid\_Succeed](#S2_TC69) |
| [MSLISTSWS\_S02\_TC70\_UpdateContentTypesXmlDocument\_WithListTitle\_Succeed](#S2_TC70) |
| S03\_OperationOnListItem | [MSLISTSWS\_S03\_TC01\_AddDiscussionBoardItem\_Fail\_ListNameNotExists](#S3_TC01) |
| [MSLISTSWS\_S03\_TC02\_AddDiscussionBoardItem\_Fail\_NoDiscussionBoard](#S3_TC02) |
| [MSLISTSWS\_S03\_TC03\_AddDiscussionBoardItem\_InvalidGUIDAndNotCorrespond\_SP3WSS3](#S3_TC03) |
| [MSLISTSWS\_S03\_TC04\_AddDiscussionBoardItem\_Success\_ListNameIsInvalidGuid](#S3_TC04) |
| [MSLISTSWS\_S03\_TC05\_AddDiscussionBoardItem\_Success\_ListNameIsValidGuid](#S3_TC05) |
| [MSLISTSWS\_S03\_TC06\_GetListItemChangesSinceToken\_EmptyChangeToken](#S3_TC06) |
| [MSLISTSWS\_S03\_TC07\_GetListItemChangesSinceToken\_MetaInfo](#S3_TC07) |
| [MSLISTSWS\_S03\_TC08\_GetListItemChangesSinceToken\_MoreThan100Changes](#S3_TC08) |
| [MSLISTSWS\_S03\_TC09\_GetListItemChangesSinceToken\_NoChangeToken](#S3_TC09) |
| [MSLISTSWS\_S03\_TC10\_GetListItemChangesSinceToken\_NotMatchChangeToken](#S3_TC10) |
| [MSLISTSWS\_S03\_TC11\_GetListItemChangesSinceToken\_RowLimit](#S3_TC11) |
| [MSLISTSWS\_S03\_TC12\_GetListItemChangesSinceToken\_Succeed](#S3_TC12) |
| [MSLISTSWS\_S03\_TC13\_GetListItemChangesSinceToken\_MoveAway](#S3_TC13) |
| [MSLISTSWS\_S03\_TC14\_GetListItemChangesSinceToken\_ValidListGUID](#S3_TC14) |
| [MSLISTSWS\_S03\_TC15\_GetListItemChangesSinceToken\_ValidListTitle](#S3_TC15) |
| [MSLISTSWS\_S03\_TC16\_GetListItemChangesSinceToken\_WSS3](#S3_TC16) |
| [MSLISTSWS\_S03\_TC17\_GetListItemChangesSinceToken\_changeTokenParameterSpecified](#S3_TC17) |
| [MSLISTSWS\_S03\_TC18\_GetListItemChangesSinceToken\_ListNameNotCorrespond](#S3_TC18) |
| [MSLISTSWS\_S03\_TC19\_GetListItemChangesWithKnowledge\_EmptyRowLimit](#S3_TC19) |
| [MSLISTSWS\_S03\_TC20\_GetListItemChangesWithKnowledge\_InvalidListGuid](#S3_TC20) |
| [MSLISTSWS\_S03\_TC21\_GetListItemChangesWithKnowledge\_ListNameNotCorrespond](#S3_TC21) |
| [MSLISTSWS\_S03\_TC22\_GetListItemChangesWithKnowledge\_MetaInfo](#S3_TC22) |
| [MSLISTSWS\_S03\_TC23\_GetListItemChangesWithKnowledge\_NoExistViewNameGuid](#S3_TC23) |
| [MSLISTSWS\_S03\_TC24\_GetListItemChangesWithKnowledge\_RowLimit](#S3_TC24) |
| [MSLISTSWS\_S03\_TC25\_GetListItemChangesWithKnowledge\_SpecifiedKnowledge](#S3_TC25) |
| [MSLISTSWS\_S03\_TC26\_GetListItemChangesWithKnowledge\_Succeed](#S3_TC26) |
| [MSLISTSWS\_S03\_TC27\_GetListItemChangesWithKnowledge\_VerifyPrefixOfZrow](#S3_TC27) |
| [MSLISTSWS\_S03\_TC28\_GetListItemChangesWithKnowledge\_WithoutItemChanges](#S3_TC28) |
| [MSLISTSWS\_S03\_TC29\_GetListItemChangesWithKnowledge\_ZeroRowLimit](#S3_TC29) |
| [MSLISTSWS\_S03\_TC30\_GetListItemChangesWithKnowledge\_ViewFields](#S3_TC30) |
| [MSLISTSWS\_S03\_TC31\_GetListItemChanges\_Fail\_ListNameNotExists](#S3_TC31) |
| [MSLISTSWS\_S03\_TC32\_GetListItemChanges\_Fail\_SinceIsEmpty](#S3_TC32) |
| [MSLISTSWS\_S03\_TC33\_GetListItemChanges\_FieldRefNameNotMatch](#S3_TC33) |
| [MSLISTSWS\_S03\_TC34\_GetListItemChanges\_MetaInfo](#S3_TC34) |
| [MSLISTSWS\_S03\_TC35\_GetListItemChanges\_SpecifiedTimespan](#S3_TC35) |
| [MSLISTSWS\_S03\_TC36\_GetListItemChanges\_Success\_ListNameIsInvalidGuid](#S3_TC36) |
| [MSLISTSWS\_S03\_TC37\_GetListItemChanges\_Success\_ListNameIsValidGuid](#S3_TC37) |
| [MSLISTSWS\_S03\_TC38\_GetListItemChanges\_WSS3](#S3_TC38) |
| [MSLISTSWS\_S03\_TC39\_GetListItemChanges\_WithContain](#S3_TC39) |
| [MSLISTSWS\_S03\_TC40\_GetListItemChanges\_Fail\_EmptyListName](#S3_TC40) |
| [MSLISTSWS\_S03\_TC41\_GetListItemChanges\_NullOrEmptyViewFields](#S3_TC41) |
| [MSLISTSWS\_S03\_TC42\_GetListItems\_EmptyListNameAndViewName](#S3_TC42) |
| [MSLISTSWS\_S03\_TC43\_GetListItems\_FilesOnly](#S3_TC43) |
| [MSLISTSWS\_S03\_TC44\_GetListItems\_InvalidViewNameWithNotGUID](#S3_TC44) |
| [MSLISTSWS\_S03\_TC45\_GetListItems\_ListNameNotCorrespond](#S3_TC45) |
| [MSLISTSWS\_S03\_TC46\_GetListItems\_NonExistentViewName](#S3_TC46) |
| [MSLISTSWS\_S03\_TC47\_GetListItems\_Prefix](#S3_TC47) |
| [MSLISTSWS\_S03\_TC48\_GetListItems\_PrefixForMetaInfoProperty](#S3_TC48) |
| [MSLISTSWS\_S03\_TC49\_GetListItems\_RowLimit](#S3_TC49) |
| [MSLISTSWS\_S03\_TC50\_GetListItems\_SucceedWithNonGuidFormatListName](#S3_TC50) |
| [MSLISTSWS\_S03\_TC51\_GetListItems\_SucceedWithValidGuidListName](#S3_TC51) |
| [MSLISTSWS\_S03\_TC52\_GetListItems\_SucceedWithValidGuidViewName](#S3_TC52) |
| [MSLISTSWS\_S03\_TC53\_GetListItems\_ViewFields](#S3_TC53) |
| [MSLISTSWS\_S03\_TC54\_GetListItems\_WSS3](#S3_TC54) |
| [MSLISTSWS\_S03\_TC55\_GetListItems\_WithPaging](#S3_TC55) |
| [MSLISTSWS\_S03\_TC56\_GetListItems\_WithoutPaging](#S3_TC56) |
| [MSLISTSWS\_S03\_TC57\_GetVersionCollection\_ExcludesParameter](#S3_TC57) |
| [MSLISTSWS\_S03\_TC58\_GetVersionCollection\_FieldNameNotCorrespond](#S3_TC58) |
| [MSLISTSWS\_S03\_TC59\_GetVersionCollection\_InvalidGUIDAndNotCorrespond\_SP3WSS3](#S3_TC59) |
| [MSLISTSWS\_S03\_TC60\_GetVersionCollection\_InvalidListGUID](#S3_TC60) |
| [MSLISTSWS\_S03\_TC61\_GetVersionCollection\_InvalidParameterWithEmptyListItemID](#S3_TC61) |
| [MSLISTSWS\_S03\_TC62\_GetVersionCollection\_InvalidParameterWithNullListName](#S3_TC62) |
| [MSLISTSWS\_S03\_TC63\_GetVersionCollection\_SpecifiedFieldName](#S3_TC63) |
| [MSLISTSWS\_S03\_TC64\_GetVersionCollection\_SucceedWithNotGUIDListName](#S3_TC64) |
| [MSLISTSWS\_S03\_TC65\_GetVersionCollection\_SucceedWithValidGUIDListName](#S3_TC65) |
| [MSLISTSWS\_S03\_TC66\_IncludeMandatoryColumns](#S3_TC66) |
| [MSLISTSWS\_S03\_TC67\_UpdateListItemWithKnowledge\_FieldNotExist](#S3_TC67) |
| [MSLISTSWS\_S03\_TC68\_UpdateListItemWithKnowledge\_InvalidListVersionWithSchemaLock](#S3_TC68) |
| [MSLISTSWS\_S03\_TC69\_UpdateListItemWithKnowledge\_OnErrorReturn](#S3_TC69) |
| [MSLISTSWS\_S03\_TC70\_UpdateListItemWithKnowledge\_OwsHiddenVersionConflict](#S3_TC70) |
| [MSLISTSWS\_S03\_TC71\_UpdateListItem\_FieldNotExist](#S3_TC71) |
| [MSLISTSWS\_S03\_TC72\_UpdateListItem\_IgnoreVersion](#S3_TC72) |
| [MSLISTSWS\_S03\_TC73\_UpdateListItem\_InvalidListVersionWithSchemaLock](#S3_TC73) |
| [MSLISTSWS\_S03\_TC74\_UpdateListItem\_OnErrorReturn](#S3_TC74) |
| [MSLISTSWS\_S03\_TC75\_UpdateListItem\_OwsHiddenVersionConflict](#S3_TC75) |
| [MSLISTSWS\_S03\_TC76\_UpdateListItemsWithKnowledge\_DateInUtc](#S3_TC76) |
| [MSLISTSWS\_S03\_TC77\_UpdateListItemsWithKnowledge\_Fail\_ListNameIsEmpty](#S3_TC77) |
| [MSLISTSWS\_S03\_TC78\_UpdateListItemsWithKnowledge\_Fail\_ListNameNotExists](#S3_TC78) |
| [MSLISTSWS\_S03\_TC79\_UpdateListItemsWithKnowledge\_LockSchema](#S3_TC79) |
| [MSLISTSWS\_S03\_TC80\_UpdateListItemsWithKnowledge\_Methods](#S3_TC80) |
| [MSLISTSWS\_S03\_TC81\_UpdateListItemsWithKnowledge\_OnErrorContinue](#S3_TC81) |
| [MSLISTSWS\_S03\_TC82\_UpdateListItemsWithKnowledge\_OnErrorReturn](#S3_TC82) |
| [MSLISTSWS\_S03\_TC83\_UpdateListItemsWithKnowledge\_PreCalcIgnore](#S3_TC83) |
| [MSLISTSWS\_S03\_TC84\_UpdateListItemsWithKnowledge\_Success](#S3_TC84) |
| [MSLISTSWS\_S03\_TC85\_UpdateListItemsWithKnowledge\_ValidListTitle](#S3_TC85) |
| [MSLISTSWS\_S03\_TC86\_UpdateListItems\_DateInUtc](#S3_TC86) |
| [MSLISTSWS\_S03\_TC87\_UpdateListItems\_InvalidGUIDAndNotCorrespond\_SP3WSS3](#S3_TC87) |
| [MSLISTSWS\_S03\_TC88\_UpdateListItems\_InvalidListTitle](#S3_TC88) |
| [MSLISTSWS\_S03\_TC89\_UpdateListItems\_LockSchema](#S3_TC89) |
| [MSLISTSWS\_S03\_TC90\_UpdateListItems\_Methods](#S3_TC90) |
| [MSLISTSWS\_S03\_TC91\_UpdateListItems\_OnErrorContinue](#S3_TC91) |
| [MSLISTSWS\_S03\_TC92\_UpdateListItems\_OnErrorReturn](#S3_TC92) |
| [MSLISTSWS\_S03\_TC93\_UpdateListItems\_OwsHiddenVersion](#S3_TC93) |
| [MSLISTSWS\_S03\_TC94\_UpdateListItems\_PreCalcIgnore](#S3_TC94) |
| [MSLISTSWS\_S03\_TC95\_UpdateListItems\_ValidListTitle](#S3_TC95) |
| [MSLISTSWS\_S03\_TC96\_UpdateListItems\_EmptyListName](#S3_TC96) |
| S04\_OperationOnAttachment | [MSLISTSWS\_S04\_TC01\_AddAttachment\_ContentLengthZero](#S4_TC01) |
| [MSLISTSWS\_S04\_TC02\_AddAttachment\_InvalidParameters](#S4_TC02) |
| [MSLISTSWS\_S04\_TC03\_AddAttachment\_NullFileName](#S4_TC03) |
| [MSLISTSWS\_S04\_TC04\_AddAttachment\_SP3](#S4_TC04) |
| [MSLISTSWS\_S04\_TC05\_AddAttachment\_Succeed](#S4_TC05) |
| [MSLISTSWS\_S04\_TC06\_DeleteAttachment\_InvalidParameters](#S4_TC06) |
| [MSLISTSWS\_S04\_TC07\_DeleteAttachment\_ListNameOrUrlExclude](#S4_TC07) |
| [MSLISTSWS\_S04\_TC08\_DeleteAttachment\_ListItemIDNotCorrespond](#S4_TC08) |
| [MSLISTSWS\_S04\_TC09\_DeleteAttachment\_NonExistentListName\_WSS3](#S4_TC09) |
| [MSLISTSWS\_S04\_TC10\_DeleteAttachment\_Succeed](#S4_TC10) |
| [MSLISTSWS\_S04\_TC11\_GetAttachmentCollection\_ZeroAttachmentTest](#S4_TC11) |
| [MSLISTSWS\_S04\_TC12\_GetAttachmentCollection\_InvalidParameters](#S4_TC12) |
| [MSLISTSWS\_S04\_TC13\_GetAttachmentCollection\_NonExistentListName\_WSS3](#S4_TC13) |
| [MSLISTSWS\_S04\_TC14\_GetAttachmentCollection\_NullItemId](#S4_TC14) |
| [MSLISTSWS\_S04\_TC15\_GetAttachmentCollection\_NullListName](#S4_TC15) |
| [MSLISTSWS\_S04\_TC16\_GetAttachmentCollection\_Succeed](#S4_TC16) |
| [MSLISTSWS\_S04\_TC17\_GetAttachmentCollection\_NegativeListItemID](#S4_TC17) |
| S05\_OperationOnFiles | [MSLISTSWS\_S05\_TC01\_CheckInFile\_WithoutDocument](#S5_TC01) |
| [MSLISTSWS\_S05\_TC02\_CheckInFile\_EmptyCheckInType](#S5_TC02) |
| [MSLISTSWS\_S05\_TC03\_CheckInFile\_InvalidParameter](#S5_TC03) |
| [MSLISTSWS\_S05\_TC04\_CheckInFile\_Succeed](#S5_TC04) |
| [MSLISTSWS\_S05\_TC05\_CheckOutFile\_WithInvalidDocument](#S5_TC05) |
| [MSLISTSWS\_S05\_TC06\_CheckOutFile\_InvalidParameter](#S5_TC06) |
| [MSLISTSWS\_S05\_TC07\_CheckOutFile\_Succeed](#S5_TC07) |
| [MSLISTSWS\_S05\_TC08\_UndoCheckOut\_WithNoDocument](#S5_TC08) |
| [MSLISTSWS\_S05\_TC09\_UndoCheckOut\_InvalidParameter](#S5_TC09) |
| [MSLISTSWS\_S05\_TC10\_UndoCheckOut\_Succeed](#S5_TC10) |

Test case scenario distribution

The test case is designed to verify the MS-LISTSWS response messages and the core operations of this protocol. For example, the request message sent to server is actually verified by the server and the response is sent back to the client with correct result.

### Test case description

Common Prerequisites and Common Cleanup for all the test cases are:

|  |  |  |
| --- | --- | --- |
| Common prerequisites | Common HTTP Prerequisites | 1. The client uses HTTP protocol and authenticated user account. |
| Common HTTPS Prerequisites | 1. The client uses HTTPS protocol and authenticated user account. |
| Common cleanups | N/A | |

Test case common steps

The steps in the following test case definitions use methods and parameters in the adapter interfaces directly.

The following tables describe the traditional test cases.

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC01\_AddListFromFeature\_Succeed |
| Description | This test case is used to verify the successful status of AddListFromFeature operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddListFromFeature with valid parameters. 2. Call method DeleteList to delete the added list from server. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC01\_AddListFromFeature\_Succeed

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC02\_AddListFromFeature\_EmptyFeatureID |
| Description | This test case is used to verify the negative status of AddListFromFeature operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | * 1. Call method AddListFromFeature to add a list with empty featureID.   2. Call method AddListFromFeature to add a list with unmatched featureID and templateID.   3. Call method AddList to add a list with correct value.   4. Call method AddList to add a list with the existed listName.   5. Delete the list.   6. Call method DeleteList to delete the added list from server |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC02\_AddListFromFeature\_EmptyFeatureID

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC03\_AddListFromFeature\_InvalidTemplateId |
| Description | This test case is used to validate the AddListFromFeature operation with invalid templateID whose value is less than 0. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method AddListFromFeature to add a list with a less than 0 templateID.  The server returns a SOAP fault without error code. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC03\_AddListFromFeature\_InvalidTemplateId

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC04\_AddListFromFeature\_UnknownTemplateId |
| Description | This test case is used to validate the AddListFromFeature operation with the templateID which is not one of the known list template identifiers. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method AddListFromFeature to add a list with an unknown template id. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC04\_AddListFromFeature\_UnknownTemplateId

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC05\_AddList\_Succeed |
| Description | This test case is used to verify the successful status of AddList operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList with valid input parameters.  2. Call method DeleteList to delete the added list from server. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC05\_AddList\_Succeed

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC06\_AddList\_AlreadyUsedListName\_UnknownTemplateID |
| Description | This test case is used to verify the negative status of AddList operation with existing list name and unknown templateID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | * 1. Call method AddList to add a list with correct value.   2. Call method AddList to add a list with listName used by step1.   3. Call method AddList to add a list with a templateID which is not one of the known list template identifiers.   4. Call method DeleteList to delete the added list from server. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC06\_AddList\_AlreadyUsedListName\_UnknownTemplateID

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC07\_AddList\_Negative |
| Description | The test case is used to verify the AddList operation when the templateID is less than 0. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method AddList to add a list using less than 0 template id. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC07\_AddList\_Negative

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC08\_AddList\_UniqueTemplate |
| Description | This test case is used to verify adding a list on the protocol server with a list based on the template specified by templateID that already exists, and the template is marked as unique in AddList operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list with unique template “User\_Information\_List” on server.  2. The client verifies whether the error code returned by server is “0x8102003c”. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC08\_AddList\_UniqueTemplate

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC09\_AuthorizationFaultsTest |
| Description | This test case is used to add a list on the protocol server with invalid password. The server will notify the client authorization faults using HTTP status codes. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Establish a NetworkCredential using an invalid password. 2. Call method AddList to add a temporary list on server. 3. The server will throw an authorization error. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC09\_AuthorizationFaultsTest

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC10\_DeleteList\_Succeed |
| Description | This test case is used to verify the successful status of DeleteList operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list on server.  2. Call method DeleteList to delete the added list from server.  3. Query and verify the deleted list. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC10\_DeleteList\_Succeed

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC11\_DeleteList\_NonExistentListName |
| Description | This test case is used to verify the negative status of DeleteList operation with a listName which does not correspond to any lists. |
| Prerequisites | Common Prerequisites |
| Test execution steps | * 1. Call method DeleteList to delete a non-existent list. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC11\_DeleteList\_NonExistentListName

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC12\_DeleteNonExistentList\_WSS3 |
| Description | This test case is used to verify the DeleteList operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method DeleteList to deleted the added list in above step.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to list title of the list that was added |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC12\_DeleteNonExistentList\_WSS3

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC13\_GetListAndView\_Succeed |
| Description | This test case is used to verify the successful status of GetListAndView operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list with correct value.  2. Call method GetListAndView with correct list name and set the viewName parameter to null.  3. Call method GetListAndView with correct list title which is an invalid GUID.  4. Call method GetListAndView by specifying a viewName with an empty string.  5. Call method DeleteList to delete the added list from server. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC13\_GetListAndView\_Succeed

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC14\_GetListAndView\_InvalidGUID |
| Description | This test case is used to verify the negative status of GetListAndView operation with invalid viewName. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list on server.  2. Call method GetListAndView with an invalid GUID. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC14\_GetListAndView\_InvalidGUID

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC15\_GetListAndView\_InvalidListName |
| Description | This test case is used to validate the GetListAndView operations with invalid listName. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method GetListAndView with invalid listName to get a list view.  **Input parameters:**  • listName: a listName that is not exist on server |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC15\_GetListAndView\_InvalidListName

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC16\_GetListAndView\_InvalidViewName |
| Description | This test case is used to validate the GetListAndView operations with invalid viewName but valid list name. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method GetListAndView with invalid viewNameto get a list view.  **Input parameters:**  • viewName: a viewName that is not exist on server |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC16\_GetListAndView\_InvalidViewName

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC17\_GetListAndView\_WSS3 |
| Description | This test case is used to verify the GetListAndView operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Initialize a list on server.  2. Call method GetListAndView to get a list view.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC17\_GetListAndView\_WSS3

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC18\_GetListCollection\_Succeed |
| Description | This test case is used to verify the successful status of GetListCollection operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list on server.  2. Call method GetListCollection with valid parameters. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC18\_GetListCollection\_Succeed

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC19\_GetList\_Succeed |
| Description | This test case is used to verify the successful status of GetList operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list on server. 2. Call method GetListItems to get the User Information List from server. 3. Set both Presence and RecycleBinEnable value to True. 4. Call method GetList to get the list from server. 5. Set both Presence and RecycleBinEnable value to False. 6. Call method GetList to get the list from server. 7. Restore Presence and RecycleBinEnable settings. 8. Call method GetList to get the list from server. 9. Call method GetList with ListName which does not have a valid GUID. 10. Call method DeleteList to delete the added list from server. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC19\_GetList\_Succeed

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC20\_GetList\_NonExistentListName |
| Description | This test case is used to verify the negative status of GetList operation with a listName which does not correspond to any lists. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method GetList with a non-exist listName. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC20\_GetList\_NonExistentListName

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC21\_GetList\_WSS3 |
| Description | This test case is used to verify the GetList operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Initialize a list on server.  2. Call method GetList to get a list from server.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC21\_GetList\_WSS3

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC22\_UpdateList\_FailureErrorCodeInUpdateListFieldResults |
| Description | This test case is used to verify the ErrorCode element in the response of UpdateList operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list on server.  2. Call method UpdateList with Delete method but the field which is to be deleted does not exist in the server.  3. The client verifies whether the value of the ErrorCode element in the response is not equal to “0x00000000. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC22\_UpdateList\_FailureErrorCodeInUpdateListFieldResults

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC23\_UpdateList\_InvalidListName |
| Description | This test case is used to test operation UpdateList when listName element is invalid value (Not valid List GUID, Not valid List Title). |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Ignore the test case in product server of Windows SharePoint Services 3.0  2. Invoke AddList operation to create a new generic list.  3. Invoke GetList operation to get the current list version of the generic list.  4. Invoke UpdateList with correct value of list version, and set the value of listName element to invalid value (Not valid List GUID, Not valid List Title). |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC23\_UpdateList\_InvalidListName

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC24\_UpdateList\_InvalidListVersion\_CanNotBeConvertedToInteger |
| Description | This test case is used to test operation UpdateList when the value of element listVersion cannot be converted to an integer by product server. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke UpdateList with the value of element listVersion that cannot be converted to an integer.  3. Delete the generic list. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC24\_UpdateList\_InvalidListVersion\_CanNotBeConvertedToInteger

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC25\_UpdateList\_InvalidListVersion\_MismatchedNumeric |
| Description | This test case is used to test operation UpdateList when the value of element listVersion string is numeric but does not match the version of the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke GetList operation to get the current list version of the generic list.  3. Invoke UpdateList with the value of element listVersion that is numeric, but does not match the version of the list.  4. Delete the generic list. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC25\_UpdateList\_InvalidListVersion\_MismatchedNumeric

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC26\_UpdateList\_InvalidListVersion\_OutOfUInt32 |
| Description | This test case is used to test operation UpdateList when the value of element listVersion string is numeric but not within the range of an unsigned 32-bit integer. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke UpdateList with the value of element listVersion that is numeric, but not within the range of an unsigned 32-bit integer.  3. Delete the generic list. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC26\_UpdateList\_InvalidListVersion\_OutOfUInt32

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC27\_UpdateList\_InvalidViewNameInUpdateListFieldResults |
| Description | This test case is used to verify the UpdateList operation with the Method.AddToView element in the UpdateListFieldsRequest. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Create a generic list on server. 2. Call method UpdateList to add three fields to the list with Method.AddToView element. The first element value is not a GUID, the second is an empty string, and the third is not presented. 3. Call method GetListAndView to get the fields in the list. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC27\_UpdateList\_InvalidViewNameInUpdateListFieldResults

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC28\_UpdateList\_ListDefinitionCT |
| Description | This test case is used to verify the ListDefinitionCT complex type in UpdateList operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | * + 1. Create a generic list on server.     2. Call method GetList to get the list from server.     3. Call method UpdateList to update the list properties.     4. Create a document library list on server.     5. Set Custom Send To Destination Name and Url for document library list.     6. Call method UpdateList to update the list properties.     7. Create an issues list on server.     8. Call method UpdateList to update the list properties.     9. Create a survey list on server.     10. Call method UpdateList to update the list properties.     11. Create a discussion borad list on server.     12. Call method UpdateList to update the list properties. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC28\_UpdateList\_ListDefinitionCT

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC29\_UpdateList\_Succeed |
| Description | This test case is used to validate the successful status of UpdateList operation when the parameters are separately set to different values. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method AddList to add a generic list on server. 2. Call method UpdateList to add a field to sever. 3. Call method UpdateList with correct listname. 4. Call method UpdateList with invalid GUID. 5. Call method UpdateList with listVersion which is set to 'null'. 6. Call method DeleteList to delete the added list from server. |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC29\_UpdateList\_Succeed

|  |  |
| --- | --- |
| S01\_OperationOnList | |
| Test case ID | MSLISTSWS\_S01\_TC30\_UpdateList\_WSS3 |
| Description | This test case is used to verify the UpdateList operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | Common Prerequisites. |
| Test execution steps | 1. Initialize a list on server.  2. Call method UpdateList to update the list on server.  **Input parameters:**  listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S01\_TC30\_UpdateList\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC01\_ApplyContentTypeToList\_IncorrectContentTypeID |
| Description | This test case is used to verify that the server returns soap fault without error code in ApplyContentTypeToList operation with not findable contentTypeID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method ApplyContentTypeToList to apply another content type to the list with incorrect contentTypeID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC01\_ApplyContentTypeToList\_IncorrectContentTypeID

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC02\_ApplyContentTypeToList\_InvalidListName |
| Description | This test case is used to verify that the server returns soap fault with error code in ApplyContentTypeToList operation with incorrect list name which does not correspond to any lists. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method ApplyContentTypeToList to apply the content type to the list with incorrect listName. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC02\_ApplyContentTypeToList\_InvalidListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC03\_ApplyContentTypeToList\_InvalidListName\_WSS3 |
| Description | This test case is used to verify the ApplyContentTypeToList operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product is Windows SharePoint Services 3.0 |
| Test execution steps | 1. Call method AddList to add a list on server.  **Input parameters:**  • listName: New a list name for example  2. Call method CreateContentType to create the content type for the list.  3. Call method ApplyContentTypeToList to apply the content type for the list.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC03\_ApplyContentTypeToList\_InvalidListName\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC04\_ApplyContentTypeToList\_WithListGuid\_Succeed |
| Description | This test case is used to test the ApplyContentTypeToList operation with correct contentTypeID and listName. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list.  2. Create a content type.  3. Call method ApplyContentTypeToList to apply another content type to the list with correct contentTypeID and listName title. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC04\_ApplyContentTypeToList\_WithListGuid\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC05\_ApplyContentTypeToList\_WithListTitle\_Succeed |
| Description | This test case is used to test the ApplyContentTypeToList operation with correct contentTypeID and listName GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Call method ApplyContentTypeToList to apply another content type to the list with correct contentTypeID and listName GUID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC05\_ApplyContentTypeToList\_WithListTitle\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC06\_CreateContentType\_ContentTypeOrderNotReturned\_WSS3 |
| Description | This test case is used to test the result doesn't contain GetListContentTypesResult.ContentTypes.ContentTypeOrder in GetListContentTypes operation under Windows SharePoint Services 3.0. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method AddList to add a list on server.  **Input parameters:**  • templateType: it's not a user information list  2. Call method CreateContentType to create the content type.    3. Call method GetListContentTypes to get content types for the list. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC06\_CreateContentType\_ContentTypeOrderNotReturned\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC07\_CreateContentType\_ErrorFieldName |
| Description | This test case is used to verify that the server returns SOAP fault if the fields element contains invalid field name in CreateContentType operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create a content type on the list.  **Input parameters:**  • fields: two field names, one valid and one invalid  3. Call method GetListContentType to get the content type created in step 2.  **Input parameters:**  • contentTypeId: the id returned by step2  4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC07\_CreateContentType\_ErrorFieldName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC08\_CreateContentType\_ErrorPropertyName |
| Description | This test case is used to verify that the server returns a SOAP fault when there is property naming error with the content type still added in CreateContentType operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create a content type on the list.  **Input parameters:**  • fields: one valid field name  • contentTypeProperties: set Title to a invalid name  3. Call method GetListContentTypes to get existing content types on the list.  **Input parameters:**  • contentTypeId: null  4. Call method GetListContentType to get detailed data of the content type.  **Input parameters:**  • contentTypeId: the id of the content type found in step 3  5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC08\_CreateContentType\_ErrorPropertyName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC09\_CreateContentType\_IncorrectDisplayName |
| Description | This test case is used to verify that the server returns soap fault in CreateContentType operation with incorrect display name. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type with invalid display name. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC09\_CreateContentType\_IncorrectDisplayName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC10\_CreateContentType\_InvalidListName |
| Description | This test case is used to verify that the server returns soap fault with error code in CreateContentType operation with invalid ListName which does not correspond to any lists. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method CreateContentType to create a content type with invalid ListName. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC10\_CreateContentType\_InvalidListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC11\_CreateContentType\_InvalidListName\_WSS3 |
| Description | This test case is used to verify the CreateContentType operation in Windows SharePoint Services 3.0 when ListName parameter can't be found in current existed lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method CreateContentType to create the content type.  **Input parameters:**  • listName: it is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC11\_CreateContentType\_InvalidListName\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC12\_CreateContentType\_InvalidParentType |
| Description | This test case is used to verify that the server returns soap fault in CreateContentType operation with incorrect parentType. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type with invalid parentType. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC12\_CreateContentType\_InvalidParentType

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC13\_CreateContentType\_UseListGuid\_Succeed |
| Description | This test case is used to test the CreateContentType operation with correct list name GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type with correct listname GUID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC13\_CreateContentType\_UseListGuid\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC14\_CreateContentType\_UseListTitle\_Succeed |
| Description | This test case is used to test the CreateContentType operation with correct list name title. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type with correct listname title. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC14\_CreateContentType\_UseListTitle\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC15\_DeleteContentTypeXmlDocument\_ContentTypeCannotBeFound |
| Description | This test case is used to verify the DeleteContentTypeXmlDocument operation returning the SOAP Fault if the content type is specified by the TypeId. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Create two generic lists: list1 and list2. 2. Invoke CreateContentType operation to create new content type for each generic list. 3. Invoke UpdateContentTypeXmlDocument operation to append a new XML document for the contentType1 in List1. 4. Invoke DeleteContentTypeXmlDocument operation in the list1 with the new XML Document, but set the value of "contentTypeId" to contentType2. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC15\_DeleteContentTypeXmlDocument\_ContentTypeCannotBeFound

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC16\_DeleteContentTypeXmlDocument\_InvalidListName |
| Description | This test case is used to test the successful status of DeleteContentTypeXmlDocument operation with an invalid GUID listName which corresponds to the list title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type. 3. Update content type XML document with correct listName with title. 4. Call method DeleteContentTypeXmlDocument to delete the content type with XML document. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC16\_DeleteContentTypeXmlDocument\_InvalidListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC17\_DeleteContentTypeXmlDocument\_InvalidListName\_WSS3 |
| Description | This test case is used to verify the DeleteContentTypeXmlDocument operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method DeleteContentTypeXmlDocument to delete the content type xml document.  **Input parameters:**  • listName: it is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC17\_DeleteContentTypeXmlDocument\_InvalidListName\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC18\_DeleteContentTypeXmlDocument\_NonExistentListName |
| Description | This test case is used to verify that the server returns soap fault with error code in DeleteContentTypeXmlDocument operation with non-existing listName. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type. 3. Update content type XML document with incorrect listName. 4. Call method DeleteContentTypeXmlDocument to delete the content type with XML document. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC18\_DeleteContentTypeXmlDocument\_NonExistentListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC19\_DeleteContentTypeXmlDocument\_Succeed |
| Description | This test case is used to test the DeleteContentTypeXmlDocument operation with correct listName GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type. 3. Update content type XML document with correct listName with GUID. 4. Call method DeleteContentTypeXmlDocument to delete the content type with XML document. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC19\_DeleteContentTypeXmlDocument\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC20\_DeleteContentTypeXmlDocument\_Success |
| Description | This test case is used to verify the successful status of DeleteContentTypeXmlDocument operation with deleting an xml document from content type. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create an content type.  **Input parameters:**  • fields: with valid fields names  3. Call method UpdateContentTypesXmlDocument to update the content type.  **Input parameters:**  • newDocument: a valid XML document  4. Call method DeleteContentTypeXmlDocument to delete the created content type.  5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC20\_DeleteContentTypeXmlDocument\_Success

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC21\_DeleteContentType\_ContentTypeCannotBeFound |
| Description | This test case is used to test operation DeleteContentType when the value of element "contentTypeId" cannot be found in the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create two generic list (one is List\_1, another is List\_2).  2. Invoke CreateContentType operation to create new content type for each generic list. (contentType1 for List\_1, contentType2 is List\_2).  3. Invoke DeleteContentType operation in the generic list List1, but set the value of "contentTypeId" to contentType2. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC21\_DeleteContentType\_ContentTypeCannotBeFound

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC22\_DeleteContentType\_IncorrectListName |
| Description | This test case is used to verify that the server returns soap fault with error code in DeleteContentType operation with incorrect listName. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type. 3. Call method DeleteContentType to delete the content type with incorrect listName. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC22\_DeleteContentType\_IncorrectListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC23\_DeleteContentType\_InvalidListName\_WSS3 |
| Description | This test case is used to verify DeleteContentType operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method DeleteContentType to delete the created content type.  **Input parameters:**  • listName: it is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC23\_DeleteContentType\_InvalidListName\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC24\_DeleteContentType\_NonExistentContentType |
| Description | This test case is used to verify that server returns soap fault without error code in DeleteContentType operation with non-existing. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type. 3. Call method DeleteContentType to delete the content type with incorrect contentTypeID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC24\_DeleteContentType\_NonExistentContentType

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC25\_DeleteContentType\_UseListTitle\_Succeed |
| Description | This test case is used to test the DeleteContentType operation with an invalid GUID listName but corresponds to the title of list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type. 3. Call method DeleteContentType to delete the content type with correct listName title. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC25\_DeleteContentType\_UseListTitle\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC26\_DeleteContentType\_WithListGuid\_Succeed |
| Description | This test case is used to test the DeleteContentType operation with correct listName with GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method CreateContentType to create a content type. 3. Call method DeleteContentType to delete the content type with correct listName GUID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC26\_DeleteContentType\_WithListGuid\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC27\_GetListContentType\_IncorrectContentType |
| Description | This test case is used to test GetListContentType with incorrect contentTypeID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content type with incorrect contentTypeID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC27\_GetListContentType\_IncorrectContentType

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC28\_GetListContentType\_IncorrectListName |
| Description | This test case is used to test GetListContentType with incorrect list name title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content type with incorrect listName title. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC28\_GetListContentType\_IncorrectListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC29\_GetListContentType\_InvalidListName\_WSS3 |
| Description | This test case is used to verify GetListContentType operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method GetListContentType to get content type of the list.  **Input parameters:**  • listName: it is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC29\_GetListContentType\_InvalidListName\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC30\_GetListContentType\_UseListTitleAsListName |
| Description | This test case is used to test operation GetListContentType when the value of element listName is title of the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList to create a generic list.  2. Invoke CreateContentType operation to create new content type for the new generic list.  3. Invoke GetList operation to get the title of the new generic list.  4. Invoke GetListContentType operation with the new content type in the generic list, set the value of listName as the title of the list. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC30\_GetListContentType\_UseListTitleAsListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC31\_GetListContentType\_UseValidGuidAsListName |
| Description | This test case is used to test operation GetListContentType when the value of element listName is the GUID of the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList to create a generic list.  2. Invoke CreateContentType operation to create new content type for the new generic list.  3. Invoke GetListContentType operation with the new content type in the generic list, set the value of listName as the GUID of the list. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC31\_GetListContentType\_UseValidGuidAsListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC32\_GetListContentTypesAndProperties\_Fault |
| Description | This test case is used to test server behavior when at least one of the input parameter is invalid in GetListContentTypesAndProperties operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method the GetListContentTypesAndProperties operation to retrieve all content types from a list and specified properties from the list and site property bags.   **Input parameter:**   * listName: Invalid * contentTypeId: empty * propertyPrefix: empty * includeWebProperties: false |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC32\_GetListContentTypesAndProperties\_Fault

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC33\_GetListContentTypesAndProperties\_ContentTypeVersion |
| Description | This test case is used to test the Version attribute in the response in GetListContentTypesAndProperties operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList operation to create a generic list on the server. 2. Call method GetListContentTypesAndProperties operation with valid input parameters. Record the value of the Version attribute in the response. 3. Call method UpdateContentType operation with valid input parameters. 4. Call method GetListContentTypesAndProperties operation with valid input parameters again. Record the value of the Version attribute in the response. 5. Check whether the Value recorded in step 4 is equal to the value recorded in step 2 plus 1. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC33\_GetListContentTypesAndProperties\_ContentTypeVersion

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC34\_GetListContentTypesAndProperties\_IncorrectListName |
| Description | This test case is used to test the GetListContentTypesAndProperties operation with incorrect list name. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content types and properties with incorrect listName. 4. Call method GetList with incorrect listName. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC34\_GetListContentTypesAndProperties\_IncorrectListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC35\_GetListContentTypesAndProperties\_OnUserInfoList |
| Description | This test case is used to test GetListContentTypesAndProperties operation on a UserInfoList. The ContentTypeOrder element will not be returned in Microsoft SharePoint Foundation 2010. |
| Prerequisites | The product should be Microsoft SharePoint Foundation 2010. |
| Test execution steps | 1. Call method GetListCollection to get a user info list.  2. Call method GetListContentTypesAndProperties to get the content type and properties for the list.  **Input parameters:**  • listName: a user information list’s listName |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC35\_GetListContentTypesAndProperties\_OnUserInfoList

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC36\_GetListContentTypesAndProperties\_Succeed\_BestMatchAndPrefix |
| Description | This test case is used to test server behavior when all input parameters are valid in GetListContentTypesAndProperties operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Add 5 items to the list. 3. Invoke GetListContentTypesAndProperties operation to retrieve all content types from a list and specified properties from the list and site property bags.   **Input parameter:**   * listName: Valid GUID and corresponds to the identification of a list on the site * contentTypeId: Specified and exist on the list * propertyPrefix: The prefix of the requested property keys, not empty * includeWebProperties: true, the protocol server must return properties and files from the site property bag |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC36\_GetListContentTypesAndProperties\_Succeed\_BestMatchAndPrefix

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC37\_GetListContentTypesAndProperties\_Succeed\_WithGuid |
| Description | This test case is used to test GetListContentTypesAndProperties with correct list name GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content types and properties with correct listname GUID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC37\_GetListContentTypesAndProperties\_Succeed\_WithGuid

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC38\_GetListContentTypesAndProperties\_Succeed\_WithListName |
| Description | This test case is used to test GetListContentTypesAndProperties with correct list name title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content types and properties with correct listname title. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC38\_GetListContentTypesAndProperties\_Succeed\_WithListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC39\_GetListContentTypes\_VersionTest |
| Description | This test case is used to test the content type's version on the protocol server in UpdateContentType operation and GetListContentTypes operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList operation to create a temporary list on the server. 2. Call method GetListContentTypes operation to receive GetListContentTypesResult element. 3. Call method UpdateContentType operation to update the first content type of the list. 4. Call method GetListContentTypes operation to get the version value of the first content type. 5. The client compares the version value which is get from the server with the version value saved in step 3. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC39\_GetListContentTypes\_VersionTest

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC40\_GetListContentTypes\_BestMatch |
| Description | This test case is used to test a BestMatch content type is returned in the response in GetListContentTypes operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method GetListContentTypes using the list's title as the list name.   **Input parameter:**   * contentTypeId: Set as ContentTypeID.Document |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC40\_GetListContentTypes\_BestMatch

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC41\_GetListContentTypes\_GetBestMatch |
| Description | This test case is used to test the input contentTypeId is an identifier of a content type in GetListContentTypes operation whether the BestMatch attribute of the content type is specified in the response. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method GetListContentTypes with empty contentTypeId to receive all content types in the list. 3. Call method GetListContentTypes with contentTypeId set to one of the identifier of the content types returned in step 1. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC41\_GetListContentTypes\_GetBestMatch

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC42\_GetListContentTypes\_IncorrectListName |
| Description | This test case is used to test GetListContentTypes with incorrect list name. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content types with incorrect listName. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC42\_GetListContentTypes\_IncorrectListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC43\_GetListContentTypes\_InvalidListName\_WSS3 |
| Description | This test case is used to verify GetListContentTypes operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method GetListContentTypes to get the content type for the list.  **Input parameters:**  • listName: it is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC43\_GetListContentTypes\_InvalidListName\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC44\_GetListContentTypes\_NoBestMatch |
| Description | This test case is used to test none BestMatch content type is returned in the response in GetListContentTypes operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Call method GetListContentTypes using the list's GUID as the list name.   **Input parameter:**   * contentTypeId: Set as an invalid content type id that the server cannot find |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC44\_GetListContentTypes\_NoBestMatch

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC45\_GetListContentTypes\_OnUserInfoList |
| Description | This test case is used to validate that the server does not return ContentTypeOrder element when GetListContentTypes operation functions on a UserInfoList. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method GetListCollection to get a user information list.  2. Call method GetListContentTypes to get the list content type.  **Input parameters:**  • listName: a user information list’s listName |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC45\_GetListContentTypes\_OnUserInfoList

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC46\_GetListContentTypes\_Succeed\_BestMatch |
| Description | This test case is used to test GetListContentTypes about finding best match. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content types with correct listName title and verify best match results. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC46\_GetListContentTypes\_Succeed\_BestMatch

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC47\_GetListContentTypes\_Succeed\_WithGuid |
| Description | This test case is used to test GetListContentTypes with correct list name GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content types with correct listName GUID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC47\_GetListContentTypes\_Succeed\_WithGuid

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC48\_GetListContentTypes\_Succeed\_WithListName |
| Description | This test case is used to test GetListContentTypes with correct list name title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Get the list content types with incorrect contentTypeID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC48\_GetListContentTypes\_Succeed\_WithListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC49\_UpdateContentTypeXMLDocument\_IncorrectContentTypeId |
| Description | This test case is used to test the UpdateContentType operation with incorrect contentTypeID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content type XML document with incorrect contentTypeID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC49\_UpdateContentTypeXMLDocument\_IncorrectContentTypeId

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC50\_UpdateContentTypeXMLDocument\_IncorrectListName |
| Description | This test case is used to test the UpdateContentType operation with incorrect listName. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content type XML document with incorrect listName. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC50\_UpdateContentTypeXMLDocument\_IncorrectListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC51\_UpdateContentTypeXMLDocument\_WithListGuid\_Succeed |
| Description | This test case is used to test the UpdateContentType operation with correct listName GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content type XML document with correct listName GUID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC51\_UpdateContentTypeXMLDocument\_WithListGuid\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC52\_UpdateContentTypeXMLDocument\_WithListTitle\_Succeed |
| Description | This test case is used to test the UpdateContentType operation with correct listName title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content type XML document with correct listName title. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC52\_UpdateContentTypeXMLDocument\_WithListTitle\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC53\_UpdateContentTypeXmlDocument\_InvalidListName\_WSS3 |
| Description | This test case is used to verify UpdateContentTypeXmlDocument operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method UpdateContentTypeXmlDocument to update the XML Document properties of the content type collection on a list. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC53\_UpdateContentTypeXmlDocument\_InvalidListName\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC54\_UpdateContentTypeXmlDocument\_Success |
| Description | This test case is used to verify that an xml document is successfully added into content type in UpdateContentTypeXmlDocument. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create a content type.  3. Call method UpdateContentTypesXmlDocument to update the content type xml document.  **Input parameters:**  • newDocument: an valid XML document with the existing NamespaceURI. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC54\_UpdateContentTypeXmlDocument\_Success

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC55\_UpdateContentType\_AddFieldsError |
| Description | This test case is used to verify the negative status of adding a new field but success of adding, updating and deleting of existing fields in UpdateContentType operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentTypeto create a content type for the list.  3. Call method GetListContentType to get the content type created in step2.  4. Call method UpdateContentType to add a invalid field, all other parameters are set to valid value.  **Input parameters:**  • addFields: a field with invalid field name  5. Call method GetListContentType method to get list content type again.  6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC55\_UpdateContentType\_AddFieldsError

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC56\_UpdateContentType\_Combination |
| Description | This test case is used to test successfully adding and deleting fields in UpdateContentType operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add a field into the list. 3. Call method CreateContentType to create a content type. 4. Add a field into the content type. 5. Call method UpdateContentType to update the content type. 6. Delete an existing field on a content type. 7. Call method UpdateContentType to update the content type. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC56\_UpdateContentType\_Combination

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC57\_UpdateContentType\_DeleteFieldsError |
| Description | This test case is used to verify the negative status of deleting a field but success of adding, updating and deleting of other fields in UpdateContentType operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create a content type for the list.  3. Call method GetListContentType to get the content type created in step2.  4. Call method UpdateContentType to delete a invalid field, all other parameters are set to valid value.  **Input parameters:**  • deleteFields: a invalid field  5. Call method GetListContentType method to get list content type again.  6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC57\_UpdateContentType\_DeleteFieldsError

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC58\_UpdateContentType\_FieldExists |
| Description | This test case is used to validate the UpdateContentType operation when the field name is existent. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create a content type for the list.  3. Call method UpdateContentType to update the content type.  **Input parameters:**   * listName: the listName initialized in step1 * contentTypeId: the id returned by step2 * contentTypeProperties: new a contentTypeProperty with description “UpdateContentType.Description” got from the ptfconfig file * newFields: null * updateFields: update operation * deleteFields: null * addToView: true   4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC58\_UpdateContentType\_FieldExists

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC59\_UpdateContentType\_FieldNotExists |
| Description | This test case is used to test that the error code will be non-zero when using UpdateContentType method to delete a field that does not exist in the list. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create a content type on the list.  3. Call method UpdateContentType  to update the previous step created content type with not exist field name.  **Input parameters:**  • updateFields: a field not existing in the list  4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC59\_UpdateContentType\_FieldNotExists

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC60\_UpdateContentType\_IncorrectContentTypeId |
| Description | This test case is used to test the UpdateContentType operation with incorrect contentTypeId. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create the content type of the list. 3. Update the content type with incorrect contentTypeId. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC60\_UpdateContentType\_IncorrectContentTypeId

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC61\_UpdateContentType\_IncorrectListName |
| Description | This test case is used to test the UpdateContentType operation with incorrect listName. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create the content type of the list. 3. Update the content type with incorrect listName. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC61\_UpdateContentType\_IncorrectListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC62\_UpdateContentType\_InvalidListName\_WSS3 |
| Description | This test case is used to verify UpdateContentType operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method UpdateContentType to update the content type.  **Input parameters:**  • listName: it is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC62\_UpdateContentType\_InvalidListName\_WSS3

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC63\_UpdateContentType\_UpdateFieldsError |
| Description | This test case is used to verify the negative status of updating a new field but success of adding, updating and deleting of other existing fields in UpdateContentType operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create a content type for the list.  3. Call method GetListContentType to get the content type created in step2.  4. Call method UpdateContentType to update a non-existed field, all other parameters are set to valid value.  **Input parameters:**  • updateFields: a non-existed field  5. Call method GetListContentType method to get list content type again.  6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC63\_UpdateContentType\_UpdateFieldsError

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC64\_UpdateContentType\_UpdatePropertiesError |
| Description | The test case is used to verify that the protocol server returns a SOAP fault when there is an error on updating a property in UpdateContentType operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method CreateContentType to create a content type for the list.  3. Call method GetListContentType to get the content type created in step2.  4. Call method UpdateContentType with a invalid contentTypeProperties, all other parameters are set to valid value.  **Input parameters:**  • contentTypeProperties: a invalid contentTypeProperties  5. Call method GetListContentType method to get list content type again.  6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC64\_UpdateContentType\_UpdatePropertiesError

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC65\_UpdateContentType\_WithGuid\_Succeed |
| Description | This test case is used to test UpdateContentType with correct listName with GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content type with correct listName with GUID. 4. Call method GetListContentType to get the updated content type in step3. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC65\_UpdateContentType\_WithGuid\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC66\_UpdateContentType\_WithListTitle\_Succeed |
| Description | This test case is used to test the UpdateContentType operation with correct listName with title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create the content type of the list. 3. Update the content type with correct listName with title. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC66\_UpdateContentType\_WithListTitle\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC67\_UpdateContentTypesXmlDocument\_IncorrectContentTypeId |
| Description | This test case is used to test UpdateContentTypesXmlDocument with incorrect contentTypeID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content types XML document with incorrect contentTypeID. 4. Update the content types XML document correct contentTypeID but the content type specified by the ContentType.ID attribute does not contain any child elements. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC67\_UpdateContentTypesXmlDocument\_IncorrectContentTypeId

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC68\_UpdateContentTypesXmlDocument\_IncorrectListName |
| Description | This test case is used to test UpdateContentTypesXmlDocument with incorrect listName. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content types XML document with incorrect listName. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC68\_UpdateContentTypesXmlDocument\_IncorrectListName

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC69\_UpdateContentTypesXmlDocument\_WithGuid\_Succeed |
| Description | This test case is used to test the UpdateContentTypesXmlDocument operation with correct listName GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content types XML document with correct listName GUID. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC69\_UpdateContentTypesXmlDocument\_WithGuid\_Succeed

|  |  |
| --- | --- |
| S02\_OperationOnContentType | |
| Test case ID | MSLISTSWS\_S02\_TC70\_UpdateContentTypesXmlDocument\_WithListTitle\_Succeed |
| Description | This test case is used to test the UpdateContentTypesXmlDocument operation with listName title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Create a content type. 3. Update the content types XML document with listName title. |
| Cleanup | N/A |

MSLISTSWS\_S02\_TC70\_UpdateContentTypesXmlDocument\_WithListTitle\_Succeed

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC01\_AddDiscussionBoardItem\_Fail\_ListNameNotExists |
| Description | This test case is used to test when the listName does not correspond to a list in AddDiscussionBoardItem operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call AddDiscussionBoardItem operation with the listName is not exists on the server. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC01\_AddDiscussionBoardItem\_Fail\_ListNameNotExists

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC02\_AddDiscussionBoardItem\_Fail\_NoDiscussionBoard |
| Description | This test case is used to test when the list is not a discussion board in AddDiscussionBoardItem operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | * 1. Initialize the list on server.   2. Call method AddDiscussionBoardItem with the List is not a discussion board item. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC02\_AddDiscussionBoardItem\_Fail\_NoDiscussionBoard

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC03\_AddDiscussionBoardItem\_InvalidGUIDAndNotCorrespond\_SP3WSS3 |
| Description | This test case is used to verify AddDiscussionBoardItem operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in the current existing lists. |
| Prerequisites | The server should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method AddDiscussionBoardItemto add a discussion board item.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC03\_AddDiscussionBoardItem\_InvalidGUIDAndNotCorrespond\_SP3WSS3

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC04\_AddDiscussionBoardItem\_Success\_ListNameIsInvalidGuid |
| Description | This test case is used to test when listName is not a valid GUID but corresponds to the list title of a list on the server in AddDiscussionBoardItem operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a discussion board. 2. Call method AddDiscussionBoardItem to add new discussion items to a specified Discussion Board with the ListName is an invalid GUID but exists on the server. 3. Call method GetListItems to retrieve the details about list items in a list that satisfies specified criteria. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC04\_AddDiscussionBoardItem\_Success\_ListNameIsInvalidGuid

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC05\_AddDiscussionBoardItem\_Success\_ListNameIsValidGuid |
| Description | This test case is used to test the AddDiscussionBoardItem operation when listName is valid GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method AddDiscussionBoardItem to add new discussion items to a specified Discussion Board with the ListName is a valid GUID. 3. Call method GetListItems to retrieve the details about list items in a list that satisfies specified criteria. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC05\_AddDiscussionBoardItem\_Success\_ListNameIsValidGuid

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC06\_GetListItemChangesSinceToken\_EmptyChangeToken |
| Description | This test case is used to verify the GetListItemChangesSinceToken operation when the change token is empty. |
| Prerequisites | The list item should exist on server. |
| Test execution steps | 1. Initialize the list on server. 2. Add items to the initialized list in step1. 3. Call method GetListItemChangesSinceToken with empty row limit when the SUT is not WSS3.0. 4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC06\_GetListItemChangesSinceToken\_EmptyChangeToken

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC07\_GetListItemChangesSinceToken\_MetaInfo |
| Description | The test case is used to verify GetListItemChangesSinceToken Operation when querying the MetaInfo field and Properties attribute equals to true. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method GetListItemChangesSinceToken.  **Input parameters:**  viewFields: Set the FieldRef with MetaInfo |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC07\_GetListItemChangesSinceToken\_MetaInfo

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC08\_GetListItemChangesSinceToken\_MoreThan100Changes |
| Description | This test case is used to verify the GetListItemChangesSinceToken operation when there are more than 100 changes after a valid change token. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server. 2. Add more than 100 items (such like 120 items) to the initialized list in step1. 3. Call method GetListItemChangesSinceToken with valid change token.   4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC08\_GetListItemChangesSinceToken\_MoreThan100Changes

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC09\_GetListItemChangesSinceToken\_NoChangeToken |
| Description | The test case is used to verify GetListItemChangesSinceToken Operation when the change since token is not specified. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Add 20 list items to the list created in step1.  3. Call method GetListItemChangesSinceTokento get list item changed message.  **Input parameters:**  • changeToken: null  4. Call method GetListItemChangesSinceTokento get list item changed message.  **Input parameters:**  • rowLimit: 10 |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC09\_GetListItemChangesSinceToken\_NoChangeToken

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC10\_GetListItemChangesSinceToken\_NotMatchChangeToken |
| Description | This test case is used to verify the GetListItemChangesSinceToken operation when the change token is not empty but invalid. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server. 2. Add items to the initialized list in step1. 3. Call method GetListItemChangesSinceToken with invalid change token.   4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC10\_GetListItemChangesSinceToken\_NotMatchChangeToken

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC11\_GetListItemChangesSinceToken\_RowLimit |
| Description | This method is used to test GetListItemChangesSinceToken whether the row limit is specified or not. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Insert more than expect row limit number rows, but no more than 40. 3. Call method GetListItemChangesSinceToken with the specified row limit number. 4. Call method GetListItemChangesSinceToken without row limit number. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC11\_GetListItemChangesSinceToken\_RowLimit

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC12\_GetListItemChangesSinceToken\_Succeed |
| Description | This test case is used to test the GetListItemChangesSinceToken operation when all input parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list. 2. Call method UpdateListItems to update the list items. 3. Call method GetListItemChangesSinceToken with all valid parameters. 4. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC12\_GetListItemChangesSinceToken\_Succeed

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC13\_GetListItemChangesSinceToken\_MoveAway |
| Description | This test case is used to test the GetListItemChangesSinceToken operation while the change type is MoveAway. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add 2 document libraries, one as the source library, and the other as the destination library. 2. Upload a file to the source document library. 3. Move the file from source library to destination library. 4. Call method GetListItemChangesSinceToken to get the changes in source document library. 5. Call method GetListItems to get information of the moved file in destination library. 6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC13\_GetListItemChangesSinceToken\_MoveAway

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC14\_GetListItemChangesSinceToken\_ValidListGUID |
| Description | This test case is used to verify the GetListItemChangesSinceToken operation when the list name is a valid GUID. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server. 2. Add items to the initialized list in step1. 3. Call method GetListItemChangesSinceToken to get the changes for the specified list.   **Input parameters:**   * listId: a valid GUID  1. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC14\_GetListItemChangesSinceToken\_ValidListGUID

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC15\_GetListItemChangesSinceToken\_ValidListTitle |
| Description | This test case is used to verify the GetListItemChangesSinceToken operation when the list name is list title. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server. 2. Add items to the initialized list in step1. 3. Call method GetListItemChangesSinceToken to get the changes of the list item.   **Input parameters:**   * listName: list title  1. Call method GetList to get the list definition. 2. Call method GetListItemChangesSinceToken using specified change token. 3. Call method GetList to get the list definition. 4. Call method GetListItemChangesSinceToken to get the changes of the list item.   **Input parameters:**   * viewFields: user specified * viewName: null  1. Call method GetListItemChangesSinceToken to get the changes of the list item.   **Input parameters:**   * viewFields: user specified * viewName: invalid viewName  1. Call method GetListItemChangesSinceToken with query. 2. Call method GetListItemChangesSinceToken with contains. 3. Call method GetListItemChangesSinceToken without QueryOptions. 4. Call method GetListItemChangesSinceToken with CamlQueryOptions and the default value specified by the document. 5. Call method GetListItemChangesSinceToken with the specified QueryOptions. 6. Call method GetListItemChangesSinceToken to get the changes of the list item.   **Input parameters:**   * viewFields: null * viewName: a valid GUID but refers to a view that does not exist  1. Call method GetListItemChangesSinceToken to get the changes of the list item.   **Input parameters:**   * viewFields: null * viewName: an invalid GUID  1. Call method GetListItemChangesSinceToken to get the changes of the list item.   **Input parameters:**   * viewFields: null * viewName: an invalid GUID  1. Call method GetListItemChangesSinceToken to get the changes of the list item.   **Input parameters:**   * viewFields: null * viewName: an invalid GUID  1. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC15\_GetListItemChangesSinceToken\_ValidListTitle

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC16\_GetListItemChangesSinceToken\_WSS3 |
| Description | This test case is used to verify GetListItemChangesSinceToken operation when the parameter changeToken is empty. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method AddList to add a new list to server.  2. Call method GetListItemChangesSinceToken.  **Input parameters:**  • changeToken: Keep this parameter empty |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC16\_GetListItemChangesSinceToken\_WSS3

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC17\_GetListItemChangesSinceToken\_changeTokenParameterSpecified |
| Description | This test case is used to verify that only the first 40 rows will be returned when changes of inserting or updating are more than 40 and the changeToken parameter is specified in GetListItemChangesSinceToken operation. |
| Prerequisites | The product should be SharePoint Foundation 2010. |
| Test execution steps | 1. Initialize the list on server.  2. Call method GetListItemChangesSinceToken**.**  3. Call method UpdateListItems to insert 45 items to the list added in step1.  4. Call method GetListItemChangesSinceToken to get added list items.  **Input parameters:**  • rowLimit: empty  • changeToken: set it with the changeToken returned in step2  5. Call method GetListItemChangesSinceToken to get the left one items.  **Input parameters:**  • rowLimit: empty  6. Call method UpdateListItems to update 45 items.  7. Call method GetListItemChangesSinceToken to get updated list items.  **Input parameters:**  • rowLimit: empty |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC17\_GetListItemChangesSinceToken\_changeTokenParameterSpecified

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC18\_GetListItemChangesSinceToken\_ListNameNotCorrespond |
| Description | This test case is used to verify the GetListItemChangesSinceToken operation when listName parameter does not correspond to a list. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Add a item to the list added in step1.  3. Call method GetListItemChangesSinceToken to get added list items.  **Input parameters:**  • listName: a list name does not correspond to a list |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC18\_GetListItemChangesSinceToken\_changeTokenParameterSpecified

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC19\_GetListItemChangesWithKnowledge\_EmptyRowLimit |
| Description | This test case is used to validate the GetListItemChangesWithKnowledge operation with rowLimit parameter empty. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Add 10 list items to the list created in step1.  3. Call method GetListItemChangesWithKnowledgeto get list items.  **Input parameters:**  • rowLimit: empty |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC19\_GetListItemChangesWithKnowledge\_EmptyRowLimit

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC20\_GetListItemChangesWithKnowledge\_InvalidListGuid |
| Description | This test case is used to test the GetListItemChangesWithKnowledge operation when the list name is not valid GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method GetListItemChangesWithKnowledge with random GUID value. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC20\_GetListItemChangesWithKnowledge\_InvalidListGuid

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC21\_GetListItemChangesWithKnowledge\_ListNameNotCorrespond |
| Description | This test case is used to test the GetListItemChangesWithKnowledge operation when at least one of its input parameters is invalid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list. 2. Call method UpdateListItems to update the list items. 3. Call method GetListItemChangesWithKnowledge when at least one of its input parameters is invalid. 4. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC21\_GetListItemChangesWithKnowledge\_ListNameNotCorrespond

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC22\_GetListItemChangesWithKnowledge\_MetaInfo |
| Description | The test case is used to verify GetListItemChangesWithKnowledge Operation when querying the MetaInfo field and Properties attribute equals to true. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method UpdateListItemsWithKnowledge to update the list items.  3. Call method GetListItemChangesWithKnowledge to get list items.  **Input parameters:**   * viewFields: Set the FieldRef with MetaInfo |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC22\_GetListItemChangesWithKnowledge\_MetaInfo

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC23\_GetListItemChangesWithKnowledge\_NoExistViewNameGuid |
| Description | The test case is used to verify GetListItemChangesWithKnowledge Operation when the viewFields parameter is not specified but contains a correct formatted GUID and refers to a view that does not exist. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Inser 1 list item the added list in step1.  3. Call method GetListItemChangesWithKnowledge to get list item.  **Input parameters:**  • viewName: specify a non-exist view name |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC23\_GetListItemChangesWithKnowledge\_NoExistViewNameGuid

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC24\_GetListItemChangesWithKnowledge\_RowLimit |
| Description | This test case is used to test the GetListItemChangesWithKnowledge operation whether the row limit is specified or not. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Insert 101 items to the initialized list in step1. 3. Call method GetListItemChangesWithKnowledge when the knowledge is not specified. 4. Call method GetListItemChangesWithKnowledge with list title. 5. Call method GetListItemChangesWithKnowledge with specified list id and row limit. 6. Delete 1 item from the list then call method GetListItemChangesWithKnowledge. 7. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC24\_GetListItemChangesWithKnowledge\_RowLimit

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC25\_GetListItemChangesWithKnowledge\_SpecifiedKnowledge |
| Description | This test case is used to verify GetListItemChangesWithKnowledge operation with specified knowledge. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Insert 1 item to the list added in step1.  3. Call method GetListItemChangesWithKnowledge to Get the syncKnowledge.  4. Insert 20 items to the list added in step1.  5. Call method GetListItemChangesWithKnowledge to get all the 20 items which are created after the knowledge is retrieved.  6. Insert another 20 items again.  7. Call method GetListItemChangesWithKnowledge with previous knowledge and specified RowLimit.  8. Add one field to make schema changed.  9. Call method GetListItemChangesWithKnowledge to retrieve the knowledge again. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC25\_GetListItemChangesWithKnowledge\_SpecifiedKnowledge

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC26\_GetListItemChangesWithKnowledge\_Succeed |
| Description | This test case is used to test the GetListItemChangesWithKnowledge operation when all input parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list. 2. Call method UpdateListItems to update the list items. 3. Call method GetListItemChangesWithKnowledge with all valid parameters. 4. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC26\_GetListItemChangesWithKnowledge\_Succeed

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC27\_GetListItemChangesWithKnowledge\_VerifyPrefixOfZrow |
| Description | This test case is used to verify the names of the attributes containing the list item data in inner z:row elements prefixed by "ows\_" in the response of GetListItemChangesWithKnowledgeResponse operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method AddList to add a specified list on server.  **Input parameters:**  • listName:TemporaryListName  2. Call method UpdateListItemsto add a list item.  **Input parameters:**  • listName: TemporaryListName  • updates: add one list item  • owshiddenVersion: null  • onError: null  3. Call method GetListItemChangesWithKnowledgeto get the list item changes.  **Input parameters:**  • listName: TemporaryListName  • viewName: null  • query: null  • viewFields: null  • rowLimit: null  • queryOptions: null  • syncScope: null  • knowledge: null  • contains: null |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC27\_GetListItemChangesWithKnowledge\_VerifyPrefixOfZrow

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC28\_GetListItemChangesWithKnowledge\_WithoutItemChanges |
| Description | This test case is used to test that the response does not contain ServerTime attribute when there is not item changes in GetListItemChangesWithKnowledge. |
| Prerequisites | Common Prerequisites. |
| Test execution steps | 1. Initialize the list on server. 2. Call method GetListItemChangesWithKnowledge without knowledge. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC28\_GetListItemChangesWithKnowledge\_WithoutItemChanges

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC29\_GetListItemChangesWithKnowledge\_ZeroRowLimit |
| Description | This test case is used to validate GetListItemChangesWithKnowledge operation when RowLimit parameter is set as 0. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Add 10 list items to the list created in step1.  3. Call method GetListItemChangesWithKnowledge to get the changed list item.  **Input parameters:**      •    rowLimit: 0 |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC29\_GetListItemChangesWithKnowledge\_ZeroRowLimit

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC30\_GetListItemChangesWithKnowledge\_ViewFields |
| Description | This test case is used to test the GetListItemChangesWithKnowledge operation with viewFields parameter. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call GetListItemChangesWithKnowledge method twice with null viewFields and null/invalid viewName parameter. 3. Call GetListItemChangesWithKnowledge method twice with valid viewFields parameter and valid/null viewName parameter. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC30\_GetListItemChangesWithKnowledge\_ViewFields

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC31\_GetListItemChanges\_Fail\_ListNameNotExists |
| Description | This test case is used to test that the server returns an error when the listName does not correspond to a list in GetListItemChanges operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method GetListItemChanges to retrieve the list item changes of a list not existing. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC31\_GetListItemChanges\_Fail\_ListNameNotExists

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC32\_GetListItemChanges\_Fail\_SinceIsEmpty |
| Description | This test case is used to test the GetListItemChanges operation when the date time is an empty string. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method GetListItemChanges when the date time is an empty string. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC32\_GetListItemChanges\_Fail\_SinceIsEmpty

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC33\_GetListItemChanges\_FieldRefNameNotMatch |
| Description | This test case is used to test the server behavior when the Name attribute in the input FieldRef element matches no field in AddWikiPage operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add 10 items to the initialized list in step1. 3. Call method GetListItemChanges with the first invalid field name. 4. Call method GetListItemChanges with the second invalid field name. 5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC33\_GetListItemChanges\_FieldRefNameNotMatch

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC34\_GetListItemChanges\_MetaInfo |
| Description | The test case is used to verify GetListItemChanges Operation when querying the MetaInfo field and Properties attribute equals to true. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method GetListItems to get the list items.  3. Call method GetListItemChanges to get list item changes.  **Input parameters:**  viewFields:Set the fieledRef with MetaInfo |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC34\_GetListItemChanges\_MetaInfo

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC35\_GetListItemChanges\_SpecifiedTimespan |
| Description | This test case is used to test the GetListItemChanges operation with a specified time span. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add items to the initialized list in step1. 3. Add items to the initialized list in step1 again. 4. Call method GetListItemChanges to get the changes of list item. 5. Call method GetList to get all lists field. 6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC35\_GetListItemChanges\_SpecifiedTimespan

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC36\_GetListItemChanges\_Success\_ListNameIsInvalidGuid |
| Description | This test case is used to test that the server returns success when ListName is not valid GUID but exists on the server in GetListItemChanges operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method GetListItemChanges to retrieve the list item changes of the specific list added in the step 1. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC36\_GetListItemChanges\_Success\_ListNameIsInvalidGuid

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC37\_GetListItemChanges\_Success\_ListNameIsValidGuid |
| Description | This test case is used to test that the server returns success when ListName is valid GUID in GetListItemChanges operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method GetListItemChanges to retrieve the list item changes of the specific list added in the step 1. 3. Call method GetListItemChanges to verify the contains and since element which are not match with any list items. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC37\_GetListItemChanges\_Success\_ListNameIsValidGuid

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC38\_GetListItemChanges\_WSS3 |
| Description | This test case is used to validate GetListItemChanges operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method GetListItemChangesto get the changes of the list item.  **Input parameters:**  • listName: listName is empty or not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC38\_GetListItemChanges\_WSS3

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC39\_GetListItemChanges\_WithContain |
| Description | This test case is used to test GetListItemChanges operation when the contained element is not empty. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add 10 items to the initialized list in step1. 3. Call method GetListItemChanges with the first added item id. 4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC39\_GetListItemChanges\_WithContain

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC40\_GetListItemChanges\_Fail\_EmptyListName |
| Description | This test case is used to test GetListItemChanges operation when the listName is an empty string. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method GetListItemChanges using an empty string.   3. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC40\_GetListItemChanges\_Fail\_EmptyListName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC41\_GetListItemChanges\_NullOrEmptyViewFields |
| Description | This test case is used to test GetListItemChanges operation when the viewFields is not specified by excluding the element or including an empty element. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Add a list. 2. Add a field to the list and default view. 3. Create a list item. 4. Call method GetListItemChanges to check if all fields in default view are return when 'viewFields' is not specified by excluding the element.   **Input parameter:**  listName: Valid GUID of list.  viewFields: null  since: null  contains: null   1. Call method GetListItemChanges again to check if all fields in default view are return when 'viewFields' is not specified by including an empty element.   **Input parameter:**  listName: Valid GUID of list.  viewFields: viewFields with empty element.  since: null  contains: null   1. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC41\_GetListItemChanges\_NullOrEmptyViewFields

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC42\_GetListItems\_EmptyListNameAndViewName |
| Description | This test case is used to verify the GetListItemChanges operation when both listName and viewName are empty string. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method GetListItemChanges using empty string for listName and viewName. 3. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC42\_GetListItems\_EmptyListNameAndViewName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC43\_GetListItems\_FilesOnly |
| Description | This test case is used to test the GetListItems operation when the value of element listName does not correspond to the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Ignore the test case in product server of Windows SharePoint Services 3.0.  2. Invoke AddList operation to create a new generic list.  3. Invoke GetListItems operation when the value of element "listName" does not correspond to the list.  4. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC43\_GetListItems\_FilesOnly

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC44\_GetListItems\_InvalidViewNameWithNotGUID |
| Description | This test case is used to test the GetListItems operation when the "viewName" element is not a GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke GetListItems operation when the value of element "viewName" is not a GUID.  3. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC44\_GetListItems\_InvalidViewNameWithNotGUID

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC45\_GetListItems\_ListNameNotCorrespond |
| Description | This test case is used to test GetListItems operation when the value of element listName does not correspond to the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke GetListItems operation when the value of element "viewName" is not a GUID.  3. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC45\_GetListItems\_ListNameNotCorrespond

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC46\_GetListItems\_NonExistentViewName |
| Description | This test case is used to test the GetListItems operation when the value of "viewName" element is a GUID that does not correspond to an existing view in the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke GetListItems operation when the value of element "viewName" is a GUID, but does not correspond to an exist view in the list.  3. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC46\_GetListItems\_NonExistentViewName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC47\_GetListItems\_Prefix |
| Description | This test case is used to verify all field names start with the prefix \_ows in GetListItems operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server. 2. Add more than 100 items to the initialized list in step1. 3. Call method GetListItems with the view name which is not specified but row limit is specified. 4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC47\_GetListItems\_Prefix

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC48\_GetListItems\_PrefixForMetaInfoProperty |
| Description | The test case is used to verify GetListItems Operation when querying the MetaInfo field and Properties attribute equals to true. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server.  2. Call method GetListItems to get the list items.  **Input parameters:**   * viewFields: Set the FieldRef with MetaInfo |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC48\_GetListItems\_PrefixForMetaInfoProperty

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC49\_GetListItems\_RowLimit |
| Description | This test case is used to test the GetListItems operation with "rowLimit" parameter. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call GetListAndView to get the default view. 3. Call GetListItems operation with null rowLimit element. 4. Call GetListItems operation with empty rowLimit element. 5. Call GetListItems operation with valid rowLimit element. 6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC49\_GetListItems\_RowLimit

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC50\_GetListItems\_SucceedWithNonGuidFormatListName |
| Description | This test case is used to test the GetListItems operation when the "listName" element is not a valid GUID but its value corresponds to the title of the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.      1. Invoke GetList operation to get the title of the new generic list. 2. Invoke UpdateListItems operation to add three new list items in the new generic list. 3. Invoke GetListItems operation with the title got in step 2 of the generic list as the value of element "listName". 4. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC50\_GetListItems\_SucceedWithNonGuidFormatListName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC51\_GetListItems\_SucceedWithValidGuidListName |
| Description | This test case is used to test the GetListItems operation when the "listName" element is a valid GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | * 1. Invoke AddList operation to create a new generic list and get the valid GUID of the generic list.   2. Invoke UpdateListItems operation to add three new list items in the new generic list.   3. Invoke GetListItems operation with the valid GUID of the generic list as the value of element "listName".   4. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC51\_GetListItems\_SucceedWithValidGuidListName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC52\_GetListItems\_SucceedWithValidGuidViewName |
| Description | This test case is used to test the GetListItems operation when the "viewName" element is a valid GUID which corresponds to an existing view. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke UpdateListItems operation to add three new list items in the new generic list.  3. Invoke GetListAndView operation to get valid GUID value for default view.  4. Invoke GetListItems operation with the GUID of existed view as the value of element "viewName".  5. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC52\_GetListItems\_SucceedWithValidGuidViewName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC53\_GetListItems\_ViewFields |
| Description | This test case is used to test the GetListItems operation with the "ViewFields" element. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call UpdateListItems to insert a new item. 3. Call GetListItems operation with empty ViewFields element. 4. Call GetListItems operation twice with valid query, viewFields, rowLimit, queryOptions parameters and valid/null viewName parameter. 5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC53\_GetListItems\_ViewFields

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC54\_GetListItems\_WSS3 |
| Description | This test case is used to verify GetListItems operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0 |
| Test execution steps | 1. Call method GetListItemsto get the list items.  **Input parameters:**  listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC54\_GetListItems\_WSS3

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC55\_GetListItems\_WithPaging |
| Description | This test case is used to test GetListItems operation when Paging element is not empty. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Insert 10 items to the initialized list in step1. 3. Call method GetListItems with excluding the Paging. 4. Call method GetListItems with ListItemCollectionPositionNext that is got by step3. 5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC55\_GetListItems\_WithPaging

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC56\_GetListItems\_WithoutPaging |
| Description | This test case is used to test GetListItems in case that the Paging element is empty or excluded. |
| Prerequisites | Common Prerequisites. |
| Test execution steps | 1. Initialize the list on server. 2. Insert 10 items to the initialized list in step1. 3. Call method GetListItems to get the list items with specified rowLimit (such as: 5). 4. Call method GetListItems with excluding the paging. 5. Call method GetListItems with Paging element is empty. 6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC56\_GetListItems\_WithoutPaging

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC57\_GetVersionCollection\_ExcludesParameter |
| Description | This test case is used to test GetVersionCollection operation when the client excludes one or more parameters. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list. 2. Call method UpdateListItems to update the list items. 3. Call method GetVersionCollection when the client excludes one or more parameters. 4. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC57\_GetVersionCollection\_ExcludesParameter

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC58\_GetVersionCollection\_FieldNameNotCorrespond |
| Description | This test case is used to test GetVersionCollection operation when strFieldName does not correspond to any field in the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a generic list. 2. Call method UpdateListItems to update the list items. 3. Call method GetVersionCollection when the client excludes one or more parameters. 4. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC58\_GetVersionCollection\_FieldNameNotCorrespond

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC59\_GetVersionCollection\_InvalidGUIDAndNotCorrespond\_SP3WSS3 |
| Description | This test case is used to verify GetVersionCollection operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method GetVersionCollectionto get the version of collection.  **Input parameters:**  • listName: listname is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC59\_GetVersionCollection\_InvalidGUIDAndNotCorrespond\_SP3WSS3

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC60\_GetVersionCollection\_InvalidListGUID |
| Description | This test case is used to test GetVersionCollection operation when the list ID is not a valid GUID and also not a valid title. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method CreateList to add a new list on server.  2. Call method AddListItems to set list item count from 1.  3. Call method GetVersionCollection with invalid list GUID. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC60\_GetVersionCollection\_InvalidListGUID

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC61\_GetVersionCollection\_InvalidParameterWithEmptyListItemID |
| Description | This test case is used to test GetVersionCollection operation when the value of element strListItemID is null or an empty string, or when the value of element strListItemID does not correspond to a list item in a specified list on the site. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke UpdateList operation to add a new "Required" field in the generic list.  3. Invoke UpdateListItems to add a new list item in the generic list with a valid value for the "Required" field.  4. Invoke GetVersionCollection operation; set the value of strListItemID to null or empty string.  5. Invoke GetVersionCollection operation, set strListItemID to a negative value.  6. Invoke GetVersionCollection operation, set strListItemID to zero.  7. Invoke GetVersionCollection operation, set strListItemID to an invalid positive value.    8. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC61\_GetVersionCollection\_InvalidParameterWithEmptyListItemID

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC62\_GetVersionCollection\_InvalidParameterWithNullListName |
| Description | This test case is used to test the negative status of GetVersionCollection operation when the value of element strListID is null. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke UpdateList operation to add a new "Required" field in the generic list.  3. Invoke UpdateListItems operation to add a new list item in the generic list with a valid value for the "Required" field.  4. Get the list item ID in the response of UpdateListItems.  5. Invoke GetVersionCollection operation set the value of element strListID to null.  6. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC62\_GetVersionCollection\_InvalidParameterWithNullListName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC63\_GetVersionCollection\_SpecifiedFieldName |
| Description | This test case is used to test the GetVersionCollection when the query field name is specified. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Add list item to the added list in step1, and count is 1.  3. Call method UpdateListItemsto update the item.  4. Call method GetVersionCollectionto get the list version.  **Input parameters:**  • FieldName: specified |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC63\_GetVersionCollection\_SpecifiedFieldName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC64\_GetVersionCollection\_SucceedWithNotGUIDListName |
| Description | This test case is used to test GetVersionCollection operation when the value of element strListID is not a valid GUID but its value corresponds to the title of the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke GetList operation to get the title of the new generic list.  3. Invoke UpdateList operation to add a new "Required" field in the generic list.  4. Invoke UpdateListItems operation to add a new list item in the generic list with a valid value for the "Required" field.  5. Get the list item ID in the response of the UpdateListItems.  6. Invoke GetVersionCollection operation use the list title as the value of element listName.  7. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC64\_GetVersionCollection\_SucceedWithNotGUIDListName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC65\_GetVersionCollection\_SucceedWithValidGUIDListName |
| Description | Test the GetVersionCollection operation when the value of element strListID is the valid GUID of the list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Invoke AddList operation to create a new generic list.  2. Invoke UpdateList operation to add a new "Required" field in the generic list.  3. Invoke UpdateListItems operation to add a new list item in the generic list with a valid value for the "Required" field.  4. Get the list item ID in the response of UpdateListItems.  5. Invoke GetVersionCollection operation use the list title as the value of element listName.  6. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC65\_GetVersionCollection\_SucceedWithValidGUIDListName

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC66\_IncludeMandatoryColumns |
| Description | This test case is used to confirm the following behavior of element "IncludeMandatoryColumns" in the complex type "CamlQueryOptions". The element "IncludeMandatoryColumns" specifies that required fields and fields used by specified calculated fields be returned in addition to the fields specified by the viewFields parameter, if the element "IncludeMandatoryColumns" is set to TRUE. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method AddList to create a new generic list.  2. Call method UpdateListto add a new "Required" field in the generic list.  3. Call method UpdateListItems to add a new list item in the generic list with a valid value for the "Required" field.  4. Call method GetListItems to get the list items.  **Input parameters:**  • IncludeMandatoryColumns: TRUE  • viewfield : Use “ID” field as one viewfield  5. Check if the "Required" field appears as one attribute of "z:row" element in the response of "GetListItems" in step 4.  6. Call method GetListItems to get the list items.  **Input parameters:**  • IncludeMandatoryColumns: FALSE  • viewfield : Use “ID” field as one viewfield  7. Check if the "Required" field does not appear as one attribute of "z:row" element in the response of "GetListItems" in step 6. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC66\_IncludeMandatoryColumns

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC67\_UpdateListItemWithKnowledge\_FieldNotExist |
| Description | This test case is used to test UpdateListItemWithKnowledge operation when inserting one item using non-existing field name. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method UpdateListItemsWithKnowledge.  **Input parameters:**  • updates: Batch.Method.Field value isn’t specified |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC67\_UpdateListItemWithKnowledge\_FieldNotExist

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC68\_UpdateListItemWithKnowledge\_InvalidListVersionWithSchemaLock |
| Description | The test case is used to verify UpdateListItemWithKnowledge operation when TRUE and ListVersion is specified with a number. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method GetListItemChangesSinceToken.  3. Call method UpdateListItemsWithKnowledge.  **Input parameters:**  • updates: set the updates.Batch.ListVersion to a version which is different with the version retrived in the operation GetList. Set the updates.Batch.LockSchema to true |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC68\_UpdateListItemWithKnowledge\_InvalidListVersionWithSchemaLock

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC69\_UpdateListItemWithKnowledge\_OnErrorReturn |
| Description | This test case is used to test UpdateListItemWithKnowledge operation in the case that the value of OnError attribute is OnErrorReturn. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method UpdateListItemsWithKnowledge.  **Input parameters:**   * updates: set updates.Batch.OnError to Return. Construct three items, the second one is invalid, the first and the last one is valid |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC69\_UpdateListItemWithKnowledge\_OnErrorReturn

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC70\_UpdateListItemWithKnowledge\_OwsHiddenVersionConflict |
| Description | The test case will verify UpdateListItemWithKnowledge operation when owsHiddenversion conflicts. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Add 1 list item into the added list in step1.  3. Call method GetListItemChanges to get the changes of the llist item.  4. Call method UpdateListItemsWithKnowledge.  **Input parameters:**  • updates: Batch.Method.Field value doesn’t equal to current wshiddenversion field's value   1. Call method UpdateListItemsWithKnowledge with the owshiddenversion which equals to the owshiddenversion on server. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC70\_UpdateListItemWithKnowledge\_OwsHiddenVersionConflict

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC71\_UpdateListItem\_FieldNotExist |
| Description | This test case is used to test UpdateListItem operation when inserting item using non-existing field name. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method UpdateListItems to update the list items.  **Input parameters:**  • updates: Batch.Method.Field value isn’t specified |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC71\_UpdateListItem\_FieldNotExist

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC72\_UpdateListItem\_IgnoreVersion |
| Description | This test case is used to test UpdateListItem operation in the case that the value of Version attribute will be ignored by the server. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method UpdateListItems to update list items.  **Input parameters:**  • updates: updates.Batch.Version = "X.0.0.XXXX"  3. Call method UpdateListItem.  4. Initialize the list on server again.  5. Call method UpdateListItems to update the list items again.  **Input parameters:**  • updates: updates.Batch.Version = "X.0.0.XXXX" |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC72\_UpdateListItem\_IgnoreVersion

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC73\_UpdateListItem\_InvalidListVersionWithSchemaLock |
| Description | This test case is used to test UpdateListItem operation when the SchemaLock attribute is set to TRUE and the ListVersion is invalid. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method GetListItemChangesSinceToken to get list item changes.  3. Call method UpdateListItems to update list items.  **Input parameters:**  • ListVersion: a version which is different with the version retrived in the operation GetList  • LockSchema: true |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC73\_UpdateListItem\_InvalidListVersionWithSchemaLock

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC74\_UpdateListItem\_OnErrorReturn |
| Description | This test case is used to test UpdateListItem operation in the case that the value of OnError attribute is OnErrorReturn. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method UpdateListItems to update the list items.  **Input parameters:**  • updates: set updates.Batch.OnError to Return. Construct three items, the second one is invalid, the first and the last one is valid |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC74\_UpdateListItem\_OnErrorReturn

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC75\_UpdateListItem\_OwsHiddenVersionConflict |
| Description | This test case is used to test UpdateListItem operation when owsHiddenversion conflicts. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method UpdateListItems to update the list items.  3. Call method UpdateListItems to updte the list items.  **Input parameters:**  • updates: Batch.Method.Field value doesn’t equal to current wshiddenversion field's value |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC75\_UpdateListItem\_OwsHiddenVersionConflict

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC76\_UpdateListItemsWithKnowledge\_DateInUtc |
| Description | This test case is used to test "DateInUtc" attribute in UpdateListItemsWithKnowledge operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server.      1. Call method UpdateListItems with setting "DateInUtc" attribute to "TRUE" in "Batch" element to add a new item into the new List. 2. Call method UpdateListItems with setting "DateInUtc" attribute to "FALSE" in "Batch" element to add a new item into the new List. 3. Call method UpdateListItems with setting "DateInUtc" attribute to "null" in "Batch" element to add a new item into the new List. 4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC76\_UpdateListItemsWithKnowledge\_DateInUtc

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC77\_UpdateListItemsWithKnowledge\_Fail\_ListNameIsEmpty |
| Description | This test case is used to test UpdateListItemsWithKnowledge operation when the specified listName is empty. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method UpdateListItemsWithKnowledge when the specified listName is empty. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC77\_UpdateListItemsWithKnowledge\_Fail\_ListNameIsEmpty

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC78\_UpdateListItemsWithKnowledge\_Fail\_ListNameNotExists |
| Description | This test case is used to test UpdateListItemsWithKnowledge operation when listName does not correspond to a list. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method UpdateListItemsWithKnowledge to delete one list item of a list not existing. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC78\_UpdateListItemsWithKnowledge\_Fail\_ListNameNotExists

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC79\_UpdateListItemsWithKnowledge\_LockSchema |
| Description | This test case is used to test LockSchema" attribute's default false value in UpdateListItemsWithKnowledge operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method GetList to get the current list version. 3. Call method UpdateListItemsWithKnowledge with setting "Batch" element "ListVersion" attribute to an invalid value and "LockSchema" attribute to "null" 4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC79\_UpdateListItemsWithKnowledge\_LockSchema

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC80\_UpdateListItemsWithKnowledge\_Methods |
| Description | This method is used to test the UpdateListItemsWithKnowledge operation when the Method element's Cmd attribute uses "New" "Update" "Delete" "Move" value. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method UpdateListItemsWithKnowledge with specified ID to insert the item to the list. 3. Call method UpdateListItemsWithKnowledge without ID to insert the item to the list. 4. Call method UpdateListItemsWithKnowledge to update 1 list item and delete another item. 5. Create 2 document library lists and upload a file to source list. 6. Call method UpdateListItemsWithKnowledge to move the file to destination list. 7. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC80\_UpdateListItemsWithKnowledge\_Methods

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC81\_UpdateListItemsWithKnowledge\_OnErrorContinue |
| Description | This test case is used to test the attribute "OnError" with value "Continue" or default value in UpdateListItemsWithKnowledge operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Insert three items to the list, item1 and item3 is valid, item2 is invalid. 3. Call method UpdateListItemsWithKnowledge to verify the response. 4. Call method GetListItems to get the current items. 5. Remove all the list items in the list. 6. Call method UpdateListItemsWithKnowledge using the default OnError value. 7. Call method GetListItems to get the current list items. 8. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC81\_UpdateListItemsWithKnowledge\_OnErrorContinue

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC82\_UpdateListItemsWithKnowledge\_OnErrorReturn |
| Description | This test case is used to test the server behavior when the value of OnError attribute in the input parameters is “Return” in UpdateListItemsWithKnowledge operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Insert three items to the list, item1 and item3 is valid, item2 is invalid. 3. Call method UpdateListItemsWithKnowledge to verify the response.      1. Call method GetListItems to get the current items. 2. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC82\_UpdateListItemsWithKnowledge\_OnErrorReturn

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC83\_UpdateListItemsWithKnowledge\_PreCalcIgnore |
| Description | This test case is used to test UpdateListItemsWithKnowledge operation in the case that the attribute "PreCalc" will be ignored by the server. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Insert one item into the new List. 3. Call method UpdateListItemsWithKnowledge to update the new item with attribute PreCalc is "TRUE" and Get the server response 1. 4. Call method UpdateListItemsWithKnowledge to update the new item with attribute PreCalc is “NotValidBoolean" and Get the server response 2. 5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC83\_UpdateListItemsWithKnowledge\_PreCalcIgnore

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC84\_UpdateListItemsWithKnowledge\_Success |
| Description | This test case is used to test UpdateListItemsWithKnowledge operation when the listName is a valid GUID. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method UpdateListItemsWithKnowledge to add one list item of the specific list added in the step 1 when listName is a valid GUID. 3. Call method GetListItemChangesWithKnowledge to verify whether the item exists in the specified list by using the List GUID. 4. Call method CreateUpdateListWithKnowledgeItems with valid GUID, key is “Age”; the value is “1”, and return the UpdateListItemsWithKnowledge instance. 5. Call method UpdateListItemsWithKnowledge to update the list create by step 4. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC84\_UpdateListItemsWithKnowledge\_Success

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC85\_UpdateListItemsWithKnowledge\_ValidListTitle |
| Description | This test case is used to test the UpdateListItemsWithKnowledge operation when listName is not a GUID but a valid list title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method UpdateListItemsWithKnowledge to add two list items, one is valid and another is invalid. 3. Call method GetListItems by using the List GUID to verify the operation of inserting item just happens on the specified list. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC85\_UpdateListItemsWithKnowledge\_ValidListTitle

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC86\_UpdateListItems\_DateInUtc |
| Description | This test case is used to test "DateInUtc" attribute in UpdateListItems operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add items to the initialized list in step1. 3. Call method UpdateListItems with setting "DateInUtc" attribute to "TRUE" in "Batch" element. 4. Call method UpdateListItems with setting "DateInUtc" attribute to "FALSE" in "Batch" element. 5. Call method UpdateListItems without setting "DateInUtc" attribute in "Batch" element. 6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC86\_UpdateListItems\_DateInUtc

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC87\_UpdateListItems\_InvalidGUIDAndNotCorrespond\_SP3WSS3 |
| Description | This test case is used to verify UpdateListItem operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Call method UpdateListItemsto update the list items.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC87\_UpdateListItems\_InvalidGUIDAndNotCorrespond\_SP3WSS3

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC88\_UpdateListItems\_InvalidListTitle |
| Description | This test case is used to test UpdateListItems operation when listName is in a not valid GUID and the list title does not exist. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method UpdateListItems using an invalid GUID. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC88\_UpdateListItems\_InvalidListTitle

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC89\_UpdateListItems\_LockSchema |
| Description | This test case is used to test "LockSchema" attribute's default false value in UpdateListItems operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add items to the initialized list in step1. 3. Call method UpdateListItems with an incorrect list version value. 4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC89\_UpdateListItems\_LockSchema

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC90\_UpdateListItems\_Methods |
| Description | This method is used to test the UpdateListItems operation when the Method element's Cmd attribute uses "New" "Update" "Delete" "Move" value. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method UpdateListItems with specified ID to insert the item to the list. 3. Call method UpdateListItems without ID to insert the item to the list. 4. Call method UpdateListItems to update 1 list item and delete another item. 5. Create 2 document library lists and upload a file to source list. 6. Call method UpdateListItems to move the file to destination list. 7. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC90\_UpdateListItems\_Methods

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC91\_UpdateListItems\_OnErrorContinue |
| Description | This test case is used to verify UpdateListItems operation in the case that the value of the attribute "OnError" is "Continue" or default. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add 3 items by sequence to the initialized list in step1. 3. Call method UpdateListItems to test "OnError" attribute with value "Continue". 4. Call method GetListItems to get the current list items. 5. Call method UpdateListItems using the default “OnError” value. 6. Call method GetListItems to get the current list items. 7. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC91\_UpdateListItems\_OnErrorContinue

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC92\_UpdateListItems\_OnErrorReturn |
| Description | This test case is used to verify UpdateListItems operation in the case that the value of attribute "OnError" is "Return. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add 3 items by sequence to the initialized list in step1. 3. Call method UpdateListItems to test the "OnError" attribute with value "Return". 4. Call method GetListItems to get the items of list. 5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC92\_UpdateListItems\_OnErrorReturn

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC93\_UpdateListItems\_OwsHiddenVersion |
| Description | This Test case is used to test UpdateListItems operation when listName is a valid GUID and whether OwsHiddenVersion is specified or not. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method UpdateListItems to insert two items using List GUID, one is valid and the other one is invalid. 3. Call method GetListItems by using the List GUID to verify the operation of inserting item just happen on the specified list. 4. Call method UpdateListItems to insert the item using List GUID. 5. Call method UpdateListItems to insert the item using List GUID. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC93\_UpdateListItems\_OwsHiddenVersion

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC94\_UpdateListItems\_PreCalcIgnore |
| Description | This Test case is used to test UpdateListItems operation in case that the attribute "PreCalc" will be ignored by the server. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Add items to the initialized list in step1. 3. Call method UpdateListItems to test "PreCalc" attribute that will be ignored by server. 4. Call method UpdateListItems with resetting the “PreCalc” to some invalid value.   5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC94\_UpdateListItems\_PreCalcIgnore

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC95\_UpdateListItems\_ValidListTitle |
| Description | This test case is used to test UpdateListItems operation when the listName is not a GUID but a valid list title. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method UpdateListItems using specified list name but no list GUID to insert two items, one is valid and the other one is invalid.      1. Call method GetListItems by using the List GUID to verify the operation of inserting item just happens on the specified list. 2. Call method CreateUpdateListItems with valid GUID, key is “Age”; the value is “1”, and return the UpdateListItemsUpdates instance. 3. Call method UpdateListItems to update the list create by step 4.   6. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC95\_UpdateListItems\_ValidListTitle

|  |  |
| --- | --- |
| S03\_OperationOnListItem | |
| Test case ID | MSLISTSWS\_S03\_TC96\_UpdateListItems\_EmptyListName |
| Description | This Test case is used to test UpdateListItems operation when listName is an empty string. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server. 2. Call method UpdateListItems using empty list name.   3. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S03\_TC96\_UpdateListItems\_EmptyListName

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC01\_AddAttachment\_ContentLengthZero |
| Description | This test case is used to verify that the server returns error code "0x8007000d" when adding an attachment to a list item that content length equals to zero in AddAttachment operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.   2. Call method AddListItems to add a list item.  3. Call method AddAttachment to add an attachment.  **Input parameters:**  • attachment: a length 0 attachment  4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC01\_AddAttachment\_ContentLengthZero

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC02\_AddAttachment\_InvalidParameters |
| Description | This test case is used to test AddAttachment operation when at least one of its input parameters is invalid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list on server. 2. Add a list item. 3. Add an attachment with invalid listName parameter. 4. Add an attachment with invalid listItemID parameter. 5. Add an attachment. 6. Add an attachment with the same fileName. 7. Delete the added attachment. 8. Add an attachment without listItemID parameter. 9. Add an attachment without attachment parameter. 10. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC02\_AddAttachment\_InvalidParameters

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC03\_AddAttachment\_NullFileName |
| Description | This test case is used to verify that the server returns error code "0x81020073" with FileName parameter is Null in AddAttachment operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method AddListItems to add a list item.  3. Call method AddAttachment to add an attachment.  **Input parameters:**  • listItemID: 0  4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC03\_AddAttachment\_NullFileName

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC04\_AddAttachment\_SP3 |
| Description | This test case is used to validate AddAttachment operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in the current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Initialize a list on server.  2. Call method AddAttachment to add the attattachment to the list.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC04\_AddAttachment\_SP3

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC05\_AddAttachment\_Succeed |
| Description | This test case is used to test AddAttachment operation when all its input parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list on server. 2. Add a list item. 3. Add an attachment using valid GUID as listName parameter. 4. Delete the added attachment. 5. Add an attachment using valid list title as listName parameter. 6. Delete the added attachment. 7. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC05\_AddAttachment\_Succeed

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC06\_DeleteAttachment\_InvalidParameters |
| Description | This test case is used to test DeleteAttachment operation when at least one of its input parameters is invalid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list on server. 2. Add a list item. 3. Add an attachment. 4. Delete the attachment with invalid listName parameter. 5. Delete the attachment without listItemID parameter. 6. Delete the attachment with invalid URL parameter. 7. Delete the attachment. 8. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC06\_DeleteAttachment\_InvalidParameters

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC07\_DeleteAttachment\_ListNameOrUrlExclude |
| Description | This test case is used to verify DeleteAttachment operation when the list ItemId is specified but the list name or URL is excluded. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method AddListItems to add a list item.  3. Call method AddAttachment to add an attachement.  4. Call method DeleteAttachment to delete the attachment when listName set to null.  **Input parameters:**   * listName: null * listItemID: 0 * urlFormat: valid value   5. Call method DeleteAttachment to delete the attachment when URL is null.  **Input parameters:**  • listName: valid value  • listItemID: 0  • urlFormat: null  6. Call method DeleteAttachment to delete the attachment when listName and URL are null.  **Input parameters:**  • listName: null  • listItemID: 0  • urlFormat: null  7. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC07\_DeleteAttachment\_ListNameOrUrlExclude

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC08\_DeleteAttachment\_ListItemIDNotCorrespond |
| Description | This test case is used to test the DeleteAttachment operation when ListItemID parameter does not correspond to a list item. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method AddListItems to add a list item.  3. Call method AddAttachment to add an attachement.  4. Call method DeleteAttachment to delete the attachment when listName set to null.  **Input parameters:**   * listName: valid value * listItemID: an item ID does not correspond to a list item * urlFormat: valid value   5. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC08\_DeleteAttachment\_ListItemIDNotCorrespond

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC09\_DeleteAttachment\_NonExistentListName\_WSS3 |
| Description | This test case is used to verify DeleteAttachment operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in the current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Initialize a list on server.  2. Call method AddAttachment to add a attachment.  3. Call method DeleteAttachment to delete the added attachment.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC09\_DeleteAttachment\_NonExistentListName\_WSS3

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC10\_DeleteAttachment\_Succeed |
| Description | This test case is used to test DeleteAttachment operation when all its input parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list on server. 2. Add a list item. 3. Add an attachment. 4. Delete the attachment with valid GUID as listName parameter. 5. Add an attachment. 6. Delete the attachment with valid list title as listName parameter. 7. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC10\_DeleteAttachment\_Succeed

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC11\_GetAttachmentCollection\_ZeroAttachmentTest |
| Description | This test case is used to test the server behavior when the input parameters are valid but no attachment exists in GetAttachmentCollection operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method DeleteList to delete the temporary list if the temporary list exists. 2. Call method AddList to create a temporary list on the server. 3. Call method UpdateListItems to add a list item on the list that previously added. This list item has no attachment by default. 4. Call method GetAttachmentCollection to receive the attachment information on the list item added in the step 3. |
| Cleanup | Call method DeleteList operation to delete the temporary |

MSLISTSWS\_S04\_TC11\_GetAttachmentCollection\_ZeroAttachmentTest

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC12\_GetAttachmentCollection\_InvalidParameters |
| Description | This test case is used to test GetAttachmentCollection operation when at least one of parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list on server. 2. Add a list item. 3. Add an attachment. 4. Get the attachment collection with invalid listName parameter. 5. Get the attachment collection with invalid listItemID parameter. 6. Delete the attachment. 7. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC12\_GetAttachmentCollection\_InvalidParameters

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC13\_GetAttachmentCollection\_NonExistentListName\_WSS3 |
| Description | This test case is used to verify GetAttachmentCollection operation in Windows SharePoint Services 3.0 when ListName parameter cannot be found in the current existing lists. |
| Prerequisites | The product should be Windows SharePoint Services 3.0. |
| Test execution steps | 1. Initialize a list on server.  2. Call method AddAttachment to add a attachment.  3. Call method GetAttachmentCollection to get the attachment collection.  **Input parameters:**  • listName: listName is not a valid GUID and does not correspond to the listname that was added |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC13\_GetAttachmentCollection\_NonExistentListName\_WSS3

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC14\_GetAttachmentCollection\_NullItemId |
| Description | The test case is used to verify GetAttachmentCollection operation when list item id is null. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method GetAttachmentCollection to get the attachment collection.  **Input parameters:**   * listItemID: null * listName: a valid list name   3. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC14\_GetAttachmentCollection\_NullItemId

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC15\_GetAttachmentCollection\_NullListName |
| Description | This test case is used to verify GetAttachmentCollection operation when the list name is null. |
| Prerequisites | N/A |
| Test execution steps | 1. Initialize the list on server.  2. Call method AddListItems to add a list item.  3. Call method GetAttachmentCollection to get attachment collection.  **Input parameters:**   * listItemID: a valid listItemID returned from step2 * listName: null   4. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC15\_GetAttachmentCollection\_NullListName

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC16\_GetAttachmentCollection\_Succeed |
| Description | This test case is used to test GetAttachmentCollection operation when all its input parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list on server. 2. Add a list item. 3. Add an attachment. 4. Get the attachment collection with valid GUID as listName parameter. 5. Get the attachment collection with valid list title as listName parameter. 6. Delete the attachment. 7. Get the attachment collection with valid GUID as listName parameter. 8. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC16\_GetAttachmentCollection\_Succeed

|  |  |
| --- | --- |
| S04\_OperationOnAttachment | |
| Test case ID | MSLISTSWS\_S04\_TC17\_GetAttachmentCollection\_NegativeListItemID |
| Description | The test case will verify GetAttachmentCollection operation when list item ID is negative. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Initialize the list on server.  2. Call method GetAttachmentCollection to get attachment collection.  **Input parameters:**   * listItemID: a negative value * listName: a valid listName   3. Clean up all the lists created in above operations. |
| Cleanup | N/A |

MSLISTSWS\_S04\_TC17\_GetAttachmentCollection\_NegativeListItemID

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC01\_CheckInFile\_WithoutDocument |
| Description | This test case is used to test that SOAP fault returns when the input pageUrl parameter does not refer to a document library in CheckInFile operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method CheckInFile with pageUrl not referring to a document library.   **Input parameter:**   * strPageUrl: Not refer to a document library |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC01\_CheckInFile\_WithoutDocument

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC02\_CheckInFile\_EmptyCheckInType |
| Description | This test case is used to verify that the server returns a SOAP fault without error code when the checkInType parameter is an empty string in CheckInFile operation. |
| Prerequisites | N/A |
| Test execution steps | 1. Call method AddList to add a list on server.  2. Call method UploadFile to upload a file.  3. Call method CheckOutFile to check out the added file.  4. Call method CheckInFile to check in the file which has been checked out.  **Input parameters:**  • checkInType: empty |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC02\_CheckInFile\_EmptyCheckInType

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC03\_CheckInFile\_InvalidParameter |
| Description | This test case is used to test CheckInFile operation when at least one of its input parameters is invalid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list whose template is document library. 2. Call method UploadFile to upload a file. 3. Call method CheckOutFile to check out the added file. 4. Call method CheckInFile to check in the file which has been checked out with invalid pageUrl parameter. 5. Call method CheckInFile to check in the file with empty pageUrl parameter 6. Call method CheckInFile to check in the file which has been checked out with pageUrl parameter set to Invalid URL. 7. Call method CheckInFile to check in the file with valid pageUrl parameter 8. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC03\_CheckInFile\_InvalidParameter

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC04\_CheckInFile\_Succeed |
| Description | This test case is used to test the CheckInFile operation when all its input parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a specific list whose template is document library. 2. Call method UploadFile to upload a file. 3. Call method CheckOutFile to check out the added file. 4. Call method CheckInFile to check in the file which has been checked out using the valid parameters. 5. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC04\_CheckInFile\_Succeed

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC05\_CheckOutFile\_WithInvalidDocument |
| Description | This test case is used to test that the SOAP fault returns when the input pageUrl parameter does not refer to a document library in CheckOutFile operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method CheckOutFile with pageUrl is not a valid document library.   **Input parameter:**   * strPageUrl: Not a valid document library |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC05\_CheckOutFile\_WithInvalidDocument

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC06\_CheckOutFile\_InvalidParameter |
| Description | This test case is used to test the CheckOutFile operation when at least one of its input parameters is invalid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list whose template is document library. 2. Call method UploadFile to upload a file. 3. Call method CheckOutFile to check out the added file with invalid pageUrl parameter. 4. Call method CheckOutFile to check out the added file with pageUrl is empty string. 5. Call method CheckOutFile to check out the added file with pageUrl is Invalid URL. 6. Call method CheckOutFile to check out the added file with the checkoutToLocal parameter which does not resolve to a valid Boolean string. 7. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC06\_CheckOutFile\_InvalidParameter

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC07\_CheckOutFile\_Succeed |
| Description | This test case is used to test the CheckOutFile operation when all its input parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list whose template is document library. 2. Call method CheckOutFile to check out the added file with all valid parameters.      1. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC07\_CheckOutFile\_Succeed

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC08\_UndoCheckOut\_WithNoDocument |
| Description | This test case is used to test that the SOAP fault returns when the input pageUrl parameter does not refer to a document library in UndoCheckOut operation. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method UndoCheckOut with pageUrl is not a valid document library.   **Input parameter:**   * strPageUrl: Not refer to a document library |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC08\_UndoCheckOut\_WithNoDocument

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC09\_UndoCheckOut\_InvalidParameter |
| Description | This test case is used to test the UndoCheckOut operation when at least one of its input parameters is invalid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list whose template is document library. 2. Call method UploadFile to upload a file. 3. Call method CheckOutFile to check out the added file. 4. Call method UndoCheckOut to undo the checkout with null pageUrl parameter. 5. Call method UndoCheckOut with pageUrl parameter setting to an invalid URL. 6. Call method UndoCheckOut to undo the checkout with the pageUrl is an empty string. 7. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC09\_UndoCheckOut\_InvalidParameter

|  |  |
| --- | --- |
| S05\_OperationOnFiles | |
| Test case ID | MSLISTSWS\_S05\_TC10\_UndoCheckOut\_Succeed |
| Description | This test case is used to test the UndoCheckOut operation when all its input parameters are valid. |
| Prerequisites | Common Prerequisites |
| Test execution steps | 1. Call method AddList to add a list whose template is document library. 2. Call method UploadFile to upload a file. 3. Call method CheckOutFile to check out the added file. 4. Call method UndoCheckOut to undo the checkout in step3. 5. Delete the list. |
| Cleanup | N/A |

MSLISTSWS\_S05\_TC10\_UndoCheckOut\_Succeed