

MS-SITESS Test Suite Specification

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

Contents

[1 Configuring the test suite 4](#_Toc387925448)

[1.1 Configuring the test suite client 4](#_Toc387925449)

[1.1.1 Configuring the test suite client manually 4](#_Toc387925450)

[1.1.2 Configuring the test suite client by scripts 5](#_Toc387925451)

[1.2 Configuring the system under test (SUT) 5](#_Toc387925452)

[1.2.1 Configuring the SUT manually 5](#_Toc387925453)

[1.2.2 Configuring the SUT by scripts 6](#_Toc387925454)

[1.3 Configuring the SHOULD/MAY requirements 6](#_Toc387925455)

[2 Test suite design 7](#_Toc387925456)

[2.1 Assumptions, scope and constraints 7](#_Toc387925457)

[Assumptions 7](#_Toc387925458)

[Scope 7](#_Toc387925459)

[In scope 7](#_Toc387925460)

[Out of scope 7](#_Toc387925461)

[Constraints 7](#_Toc387925462)

[2.2 Test suite architecture 7](#_Toc387925463)

[2.3 Technical dependencies and encryption considerations 9](#_Toc387925464)

[Dependencies 9](#_Toc387925465)

[Encryption consideration 9](#_Toc387925466)

[2.4 Adapter design 9](#_Toc387925467)

[2.4.1 Adapter overview 9](#_Toc387925468)

[Protocol adapter 9](#_Toc387925469)

[SUT control adapter 10](#_Toc387925470)

[2.4.2 Technical feasibility of adapter approach 10](#_Toc387925471)

[Message generation 10](#_Toc387925472)

[Message consumption 10](#_Toc387925473)

[SUT control adapter 10](#_Toc387925474)

[2.4.3 Adapter abstract layer 10](#_Toc387925475)

[Protocol adapters 10](#_Toc387925476)

[MS-SITESS adapter interface 10](#_Toc387925477)

[SUT control adapter 11](#_Toc387925478)

[2.4.4 Adapter details 12](#_Toc387925479)

[*2.4.4.1* Protocol adapter 12](#_Toc387925480)

[2.4.4.1.1 MS-SITESS protocol adapter 12](#_Toc387925481)

[Adapter interface 13](#_Toc387925482)

[Adapter implementation 14](#_Toc387925483)

[Other class: 14](#_Toc387925484)

[Enumeration 14](#_Toc387925485)

[*2.4.4.2* SUT control adapter 14](#_Toc387925486)

[2.4.4.2.1 SUT Control adapter interface class diagram 14](#_Toc387925487)

[2.5 Test scenarios 16](#_Toc387925488)

[2.5.1 S01\_MigrateSite 16](#_Toc387925496)

[Description 16](#_Toc387925497)

[Operations 16](#_Toc387925498)

[Prerequisites 16](#_Toc387925499)

[Cleanup 16](#_Toc387925500)

[2.5.2 S02\_ManageSubSite 16](#_Toc387925501)

[Description 16](#_Toc387925502)

[Operations 17](#_Toc387925503)

[Prerequisites 17](#_Toc387925504)

[Cleanup 17](#_Toc387925505)

[2.5.3 S03\_GetUpdatedFormDigest 17](#_Toc387925506)

[Description 17](#_Toc387925507)

[Operations 17](#_Toc387925508)

[Prerequisites 17](#_Toc387925509)

[Cleanup 17](#_Toc387925510)

[2.5.4 S04\_ExportSolution 17](#_Toc387925511)

[Description 17](#_Toc387925512)

[Operations 17](#_Toc387925513)

[Prerequisites 17](#_Toc387925514)

[Cleanup 17](#_Toc387925515)

[2.5.5 S05\_ExportWorkflowTemplate 17](#_Toc387925516)

[Description 17](#_Toc387925517)

[Operations 18](#_Toc387925518)

[Prerequisites 18](#_Toc387925519)

[Cleanup 18](#_Toc387925520)

[2.5.6 S06\_GetSite 18](#_Toc387925521)

[Description 18](#_Toc387925522)

[Operations 18](#_Toc387925523)

[Prerequisites 18](#_Toc387925524)

[Cleanup 18](#_Toc387925525)

[2.5.7 S07\_HTTPStatusCode 18](#_Toc387925526)

[Description 18](#_Toc387925527)

[Operations 18](#_Toc387925528)

[Prerequisites 18](#_Toc387925529)

[Cleanup 18](#_Toc387925530)

[2.6 Test case design 19](#_Toc387925531)

[2.6.1 Traditional test case design 19](#_Toc387925532)

[2.6.2 Test case description 20](#_Toc387925533)

# 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-SITESS\_TestSuite.deployment.ptfconfig file. The MS-SITESS \_TestSuite.deployment.ptfconfig file can also be configured by running the client setup script.

1. Open MS-SITESS\TestSuite\MS-SITESS\_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 property values to match SUT settings and configuration.

* Property name="SiteCollectionName" value="MSSITESS\_SiteCollection"
* Property name="SiteCollectionUrl" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]"
* Property name="SiteCollectionPath" value="sites/[SiteCollectionName]"
* Property name="SiteName" value="MSSITESS\_Site"
* Property name="SiteUrl" value= "[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/[SiteName]"
* Property name="ServiceUrl" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/\_vti\_bin/sites.asmx"
* Property name="SpecialSubsiteName" value="MSSITESS\_SpecialSubSite"
* Property name="WebPageUrl" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/[SiteName]/[SpecialSubsiteName]/MSSITESS\_SubSite\_DocumentLibrary/MSSITESS\_CustomPage.aspx"
* Property name="NormalSubsiteUrl" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/[SiteName]/MSSITESS\_NormalSubSite"
* Property name="SpecialSubsiteUrl" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/[SiteName]/[SpecialSubsiteName]"
* Property name="ValidLCID" value="1033"
* Property name="DefaultLCID" value="1033"
* Property name="NotInstalledLCID" value="2052"
* Property name="ValidLibraryName" value="MSSITESS\_DocumentLibrary"
* Property name="DataPath" value="[TransportType]://[SutComputerName]/sites/[SiteCollectionName]/[ValidLibraryName]"
* Property name="SolutionGalleryName" value="Solution Gallery"
* Property name="WorkflowTemplateName" value="Approval - SharePoint 2010"
* Property name="RegularExpression" value="&lt;input .\*name=\&quot;(?&lt;name&gt;.+?)\&quot; .\*?value=\&quot;(?&lt;value&gt;.+?)\&quot;"
* Property name="SubSitePropertyLanguage" value="language"
* Property name="SubSitePropertyLocale" value="locale"
* Property name="SubSitePropertyCurrentUser" value="currentUser"
* Property name="SubSitePropertyUserNameInPermissions" value="permissions"
* Property name="SubSitePropertyDefaultLanguage" value="defaultLanguage"
* Property name="SubSitePropertyAnonymous" value="anonymous"
* Property name="ExpireTimePeriodBySecond" value="60"
* Property name="DefaultExpireTimePeriod" value="1800"
* Property name="SoapServiceTimeOut" value="600"
* Property name="ExportWaitTime" value="15"
* Property name="ImportWebWaitTime" value="30"
* Property name="ExportRepeatTime" value="12"
* Property name="ImportWebRepeatTime" value="12"
* Property name="InvalidLibraryName" value="invalidlibraryname"
* Property name="InvalidWorkflowTemplateName" value="InvalidWorkflowTemplateName"
* Property name="NonExistentImportUrl" value="[TransportType]://nonExistentComputerName"
* Property name="NonExistentSiteName" value="nonExistentsitename"
* Property name="UnauthorizedUserDomain" value="UnauthorizedUserDomain"
* Property name="UnauthorizedUserName" value="UnauthorizedUserName"
* Property name="UnauthorizedUserPassword" value="UnauthorizedUserPassword!"

### 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

Implementation of the SHOULD/MAY and endnote-related requirements are pre-configured in the format "<Property name="RXXXEnabled" value="XXXX"/>" for three product versions in the following config files:

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

If the RXXXEnabled property 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 choosing SharePoint Foundation 2010, open **MS-SITESS\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig** and update the RXXXEnabled property accordingly.

# Test suite design

## Assumptions, scope and constraints

Assumptions

MS-SITESS Test Suite assumes that the SUT control adapter is able to configure the SUT as required by the test cases.

Scope

In scope

* MS-SITESS 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 ten operations.
* This test suite will verify the server-side and testable requirements.

Out of scope

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

Constraints

This test suite is constrained to single connection. The test suite only deals with a single connection between client and server.

## Test suite architecture

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



The architecture of the test suite

The details of the MS-SITESS Test Suite architecture are as follows:

* The SUT hosts the Versions Web Service which test suite runs against.
* From a third-party’s point of view, the SUT is a server implementation which supports the features of the Sites Web Service Protocol.
* The SUTs in Test Suite are the following six products running on Windows platform.
* Windows SharePoint Services 3.0 SP3
* Microsoft SharePoint Foundation 2010 SP2
* Microsoft SharePoint Foundation 2013 SP1
* Microsoft Office SharePoint Server 2007 SP3
* Microsoft SharePoint Server 2010 SP2
* Microsoft SharePoint Server 2013 SP1
* The test cases use the MS-SITESS protocol adapter to call the MS-SITESS operations and retrieve the results returned by the SUT. The test cases also use the SUT control adapter to configure the SUT to the test case specific situations.
* Protocol adapter
* The protocol adapter interface is used by the test cases for communicating with the protocol adapter. When the test cases call the methods in the protocol adapter interface, the protocol adapter will invoke the corresponding methods in the protocol adapter implementation to send operation requests to the server.
* The protocol adapter implementation implements the methods declared in the protocol adapter interface. It directly uses the methods and the data structures in the MS-SITESS proxy class.
* SUT control adapter
* The SUT control adapter interface declares the interface of the SUT control adapter. The Test Cases call the methods in the SUT Control Adapter Interface when it needs to configure the SUT or get some SUT status which will not be returned by the Open Specification operations.
* The SUT control adapter implementation implements the SUT control adapter interface. Test suite uses PowerShell implementation to configure the SUT implementation automatically.
* .NET Framework web services support library
* The .NET Framework web services support library is a library provided by .NET Framework. The proxy can call the library to send or receive SOAP messages over HTTP/HTTPS to communicate with the SUT.
* The protocol adapter parses the response messages and verifies the requirements related to the concrete value types, such as XML elements. The requirements related with logics are verified by the test cases. The test case can compare the actual return data with the expected returned data defined in the test case.

## Technical dependencies and encryption 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-SITESS proxy class.
* This test suite depends on the Protocol Test Framework (PTF) to derive managed adapters.
* This test suite depends on the description on MS-CSOMSPT section 2.2.1 to realize the scenario about how to use the GetUpdatedFormDigest and GetUpdatedFormDigestInformation operations when using form digest validation, which is described in MS-SITESS section 3.1.4.6.

Encryption consideration

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

## Adapter design

### Adapter overview

One protocol adapter and one SUT control adapter will be designed for test suite.

Protocol adapter

* The MS-SITESS adapter is a managed adapter derived from the ManagedAdapterBase in the Protocol Test Framework (PTF).
* The MS-SITESS adapter has the following functionalities:
* Be responsible for constructing messages;
* Choose HTTP or HTTPS and SOAP 1.1 or 1.2 for transport.
* Communicate with the SUT by sending requests to the SUT and receiving corresponding responses from the SUT;
* Parses the response messages and validates the messages according to the WSDL schema;
* Capture requirements and generate the testing log.
* The MS-SITESS adapter uses the C# proxy class, which is generated by running the wsdl.exe tool against the full WSDL of this Open Specification to send request messages and receive response messages. The wsdl.exe can be found in Microsoft .NET Framework SDK tools.
* The MS-SITESS adapter will use the proxy class to choose SOAP1.1 or SOAP1.2 to generate or receive the corresponding SOAP messages by using .Net class library. All the protocol adapter functionalities should base on the .Net framework Web Service infrastructure.

SUT control adapter

* The SUT control adapter will use PowerShell script implementation.
* The SUT control adapter has the following functionalities:
* Set user code to be enabled or disabled for a site.
* Set a time period for the form digest to be expired.
* Get some properties of a site.
* Get the exported packages and log file on the SUT
* Remove the exported packages and log file on the SUT.
* Remove created or imported site on the SUT.
* The SUT control adapter is invoked by the test cases.

### Technical feasibility of adapter approach

Message generation

The parameters constructed by MS-SITESS Adapter will be passed to the MS-SITESS proxy class. Then the MS-SITESS proxy class will serialize the parameters and generate the SOAP request message. The SOAP request message will be sent to the SUT by calling the .NET Framework web services support Library.

Message consumption

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

SUT control adapter

The SUT Control Adapter configures the SUT to the Test Case specific situations or Get SUT status.

### Adapter abstract layer

Protocol adapters

MS-SITESS adapter interface

* 11 methods declared in the MS-SITESS adapter interface IMS\_SITESSAdapter.
* Ten of the methods correspond to the ten MS-SITESS operations. The operators of the ten methods are abstracted the same as the operations specified in the MS-SITESS.
* The method InitializeWebService is used to initialize the sites service proxy with authentication information.

The following table shows the details of these ten interface methods.

|  |  |
| --- | --- |
| Interface methods | Description |
| CreateWeb | This interface method is designed to create a new subsite on the current site. |
| DeleteWeb | This interface method is designed to delete an existing subsite of the current site. |
| ImportWeb | This interface method is designed to import a site from content migration package files. |
| ExportWeb | This interface method is designed to export a site to content migration package files. |
| ExportSolution | This interface method is designed to export a site as a solution to the solution gallery. |
| ExportWorkflowTemplate | This interface method is designed to export a workflow template as a solution to the specified document library. |
| GetSite | This interface method is designed to retrieve information about the site collection. |
| GetSiteTemplates | This interface method is designed to retrieve information about available site templates. |
| GetUpdatedFormDigest | This interface method is designed to retrieve a new form digest validation. |
| GetUpdatedFormDigestInformation | This interface method is designed to retrieve a new form digest validation and its expiration time. |
| InitializeWebService | This interface method is designed to initialize the sites service proxy with authentication information. |

MS-SITESS Protocol adapter interface

SUT control adapter

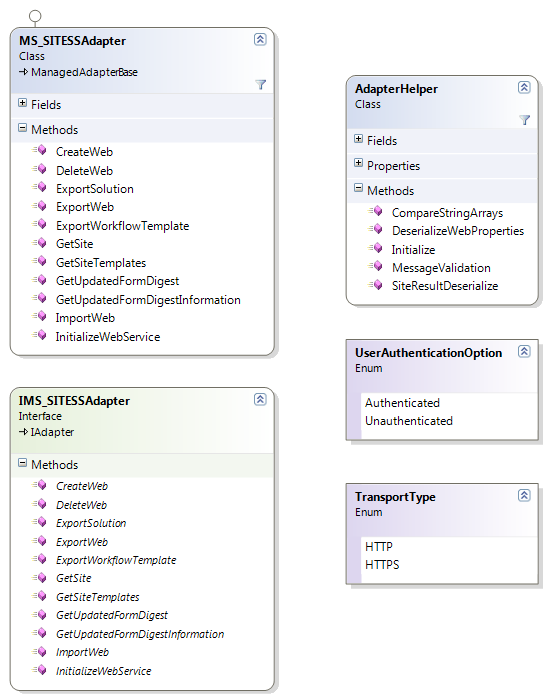
* SUT control adapter interface
* There are ten methods declared in the SUT control adapter interface ISUTControlAdapter.
* Remove all files in a document library.
* Remove all files in solution gallery.
* Remove a web with the specified web name.
* Get content file names including extension name of a list on a web, which is chosen to store the exported files.
* Get properties of the specified web which is created by the CreateWeb operation.
* Get the site collection identifier, a Globally unique identifier (GUID) that identifies the site collection.
* Set user code to be enabled or disabled on the site collection.
* Set the time period in seconds after which the security validation (also named form digest) will expire, which is used to help prevent security attacks where a user unknowingly posts data to a server.
* Get a form on a web page and Post the form with the digest value.
* Fetch status code from the log file generated by the server for the ExportWeb and ImportWeb operations, which extension name is specified as SNT for SharePoint server.

### Adapter details

#### Protocol adapter

##### MS-SITESS protocol adapter

The following figure shows the class diagram of the MS-SITESS Protocol Adapter.



MS-SITESS Protocol adapter class diagram

The following outlines details of the class diagram:

Adapter interface

* IMS\_SITESSAdapter is interface of the protocol adapter.
* IMS\_SITESSAdapter defines the ten methods which are specified in Open Specification invoked by test cases.

Adapter implementation

* MS\_SITESSAdapter is the protocol adapter class of test suite. It is used to implement IMS\_SITESSAdapter.
* The MS\_SITESSAdapter class implements the functions of the WSDL operations defined in the MS-SITESS Open Specification. The MS\_SITESSAdapter class receives the method calls from the protocol adapter interface IMS\_SITESSAdapter and invokes the corresponding MS-SITESS methods in the proxy class generated from the WSDL files.
* The Initialize method is used to initialize the MS-SITESS Test Suite.

Other class:

* SchemaValidation is generated by XMLDOM tool plus the full WSDL in the MS-SITESS, this class is used to validate whether a piece of Xml fragment in the response message complies with the XML Schema, this class is made to be a partial class of the MS-SITESS proxy class.
* The AdapterHelper class provides the methods of getting the property in configuration file, verifying the format of SOAP Fault error code, GUID and URL.

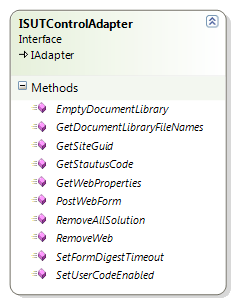
Enumeration

* TransportProtocol specification indicates the type of transport to be used by Test Suite. Possible values are HTTP and HTTPS.
* UserAuthenticationOption indicates whether Test Suite uses an authentication account, an account which is not authorized by the server.

#### SUT control adapter

##### SUT Control adapter interface class diagram

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



MS-SITESS SUT control adapter class diagram

The following outlines details of the class diagram:

* The ISUTControlAdapter 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.
* Ten methods are defined in the ISUTControlAdapter adapter interface.

## Test scenarios

There are seven scenarios designed in the MS-SITESS test suite to verify the server behavior. These scenarios are used to generate test case using the Traditional testing approach. The following table lists the scenarios designed for the MS-SITESS test suite.

|  |  |
| --- | --- |
| Scenario | Description |
| [S01\_MigrateSite](#_S1_MigrateSite) | This scenario is designed to migrate the content of a site. |
| [S02\_ManageSubSite](#_S2_ManageSubsite) | This scenario is designed to get available site template information and create/delete a subsite. |
| [S03\_GetUpdatedFormDigest](#_S3_GetUpdatedFormDigest) | This scenario is designed to get a new form digest validation and its expiration time. |
| [S04\_ExportSolution](#_S4_ExportSolution) | This scenario is designed to export the content related to a site to the solution gallery. |
| [S05\_ExportWorkflowTemplate](#_S5_ExportWorkflowTemplate) | This scenario is designed to export a workflow template as a site solution to the document library. |
| [S06\_GetSite](#_S6_GetSite) | This scenario is designed to get information about site collection. |
| [S07\_HTTPStatusCode](#_S7_UnauthorizedRequest) | This scenario is designed to send request with an unauthenticated account in order to trigger an HTTP Status Code fault. |

MS-SITESS scenarios



### S01\_MigrateSite

Description

Test suite migrates the content of a site, i.e. it exports the content related to a site into the content migration package files and then imports the site from the files.

Operations

* ExportWeb
* ImportWeb

Prerequisites

N/A

Cleanup

* Remove all files in a document Library, which is used as the store location for the files exported.
* Remove an imported site if it is imported successfully.

### S02\_ManageSubSite

Description

The client retrieves available site template information, then creates a subsite and deletes the subsite.

Operations

* GetSiteTemplates
* CreateWeb
* DeleteWeb

Prerequisites

N/A

Cleanup

Remove the created subsite.

### S03\_GetUpdatedFormDigest

Description

The client requests for a form digest validation and its expiration time.

Operations

* GetUpdatedFormDigest
* GetUpdatedFormDigestInformation

Prerequisites

N/A

Cleanup

Set the form digest timeout value to default value on the server.

### S04\_ExportSolution

Description

The client exports the content of a site as a solution.

Operations

ExportSolution

Prerequisites

N/A

Cleanup

Delete the exported solutions.

### S05\_ExportWorkflowTemplate

Description

The client exports a workflow template as a site solution.

Operations

ExportWorkflowTemplate

Prerequisites

N/A

Cleanup

Remove all the solution files if they are generated successfully.

### S06\_GetSite

Description

The client retrieves information about the site collection.

Operations

GetSite

Prerequisites

Create one site in an existing MS-SITESS site collection on the server.

Cleanup

N/A

### S07\_HTTPStatusCode

Description

The client sends a request with an unauthenticated account in order to trigger an HTTP Status Code fault.

Operations

GetSiteTemplates

Prerequisites

N/A

Cleanup

N/A

## Test case design

### Traditional test case design

Traditional testing is adopted as the test approach for the MS-SITESS Test Suite. 28 test cases are designed to cover the seven scenarios designed in [section 2.5](#_Test_Scenarios). And both negative and positive testing techniques are used in these test cases. The scenarios and the test cases are listed in the following table:

|  |  |
| --- | --- |
| Scenario ID | Test case name |
| S01\_MigrateSite | [MSSITESS\_S01\_TC01\_MigratingSuccessfully](#MSSITESS_S01_TC01_MigratingSuccessfully) |
| [MSSITESS\_S01\_TC02\_ExportingMutiplePackages](#MSSITESS_S01_TC02_ExportingMultiplePacka) |
| [MSSITESS\_S01\_TC03\_ExportingEqualto0x18CabSize](#MSSITESS_S01_TC03_ExportingEqualto0x18Ca) |
| [MSSITESS\_S01\_TC04\_ExportingFailureInvalidExportUrl](#MSSITESS_S01_TC04_ExportingFailureInvali) |
| [MSSITESS\_S01\_TC05\_ExportingFailureExportFileNoAccess](#MSSITESS_S01_TC05_ExportingFailureExport) |
| [MSSITESS\_S01\_TC06\_ExportingFailureOverwriteFailure](#MSSITESS_S01_TC06_ExportingFailureOverwr) |
| [MSSITESS\_S01\_TC07\_ImportingFailureOverwriteFailure](#MSSITESS_S01_TC07_ImportingFailureOverwr) |
| [MSSITESS\_S01\_TC08\_ImportingFailureInvalidImportUrl](#MSSITESS_S01_TC08_ImportingFailureInvali) |
| [MSSITESS\_S01\_TC09\_ImportingFailureImportFileNoAccess](#MSSITESS_S01_TC09_ImportingFailureImport) |
| [MSSITESS\_S01\_TC10\_ImportingFailureLogFileNoAccess](#MSSITESS_S01_TC10_ImportingFailureLogFil) |
| [MSSITESS\_S01\_TC11\_ImportingFailureImportWebNotEmpty](#MSSITESS_S01_TC11_ImportingFailureImport) |
| [MSSITESS\_S01\_TC12\_ImportingFailureLogPathEmpty](#MSSITESS_S01_TC12_ImportingFailureLogPat) |
| S02\_ManageSubSite | [MSSITESS\_S02\_TC01\_ManagingSubsiteSuccessfully](#MSSITESS_S02_TC01_ManagingSubsiteSuccess) |
| [MSSITESS\_S02\_TC02\_ManagingSubsiteWithoutOptionalParameters](#MSSITESS_S02_TC02_ManagingSubsiteWithout) |
| [MSSITESS\_S02\_TC03\_CreateWebFailureUrlAlreadyInUse](#MSSITESS_S02_TC03_CreateWebFailureUrlAlr) |
| [MSSITESS\_S02\_TC04\_CreateWebFailureTemplateNotExist](#MSSITESS_S02_TC04_CreateWebFailureTempla) |
| [MSSITESS\_S02\_TC05\_DeleteWebFailureNonExistentUrl](#MSSITESS_S02_TC05_DeleteWebFailureNonExi) |
| [MSSITESS\_S02\_TC06\_CreateWebWithZeroFalse](#MSSITESS_S02_TC06_CeateWebWithZeroFalse) |
| [MSSITESS\_S02\_TC07\_GetSiteTemplateNotInstalledLCID](#MSSITESS_S02_TC07_GetSiteTemplateNotInst) |
| [MSSITESS\_S02\_TC08\_GetSiteTemplatesSuccessfully](#MSSITESS_S02_TC08_GetSiteTemplatesSucces) |
| S03\_GetUpdatedFormDigest | [MSSITESS\_S03\_TC01\_GetUpdatedFormDigest](#MSSITESS_S03_TC01_GetUpdatedFormDigest) |
| [MSSITESS\_S03\_TC02\_GetUpdatedFormDigestInformation](#MSSITESS_S03_TC02_GetUpdatedFormDigestIn) |
| S04\_ExportSolution | [MSSITESS\_S04\_TC01\_ExportSolutionSucceed](#MSSITESS_S04_TC01_ExportSolutionSucceed) |
| S05\_ExportWorkflowTemplate | [MSSITESS\_S05\_TC01\_ExportWorkflowTemplateSucceed](#MSSITESS_S05_TC01_ExportWorkflowTemplate) |
| [MSSITESS\_S05\_TC02\_ExportWorkflowTemplateInvalidLibrary](#MSSITESS_S05_TC02_ExportWorkflowTemplate) |
| [MSSITESS\_S05\_TC03\_ExportWorkflowTemplateInvalidTemplate](#MSSITESS_S05_TC03_ExportWorkflowTemplate) |
| S06\_GetSite | [MSSITESS\_S06\_TC01\_GetSiteSucceed](#MSSITESS_S06_TC01_GetSiteSucceed) |
| S07\_HTTPStatusCode | [MSSITESS\_S07\_TC01\_HttpStatusCodesFault](#MSSITESS_S07_TC01_HttpStatusCodesFault) |

Test case scenario distribution

### Test case description

There are 28 test cases designed for the MS-SITESS test suite.

* The steps in the following test case definitions are using methods and parameter names in the adapter interfaces directly.
* Except otherwise specified, all test cases use HTTP as transport protocol.

The following table describes common prerequisites, common cleanup and common steps for all the test cases:

|  |  |
| --- | --- |
| Common prerequisites | The client uses HTTP/HTTPS protocol and an authenticated user account. |
| Common steps | N/A |
| Common cleanups | 1. Delete all files in the “MSSITESS\_DocumentLibrary” folder of the site “MSSITESS\_Site”; 2. Remove the subsite created by the ImportWeb and CreateWeb operations. |

Common prerequisites, steps, and cleanups for all MS-SITESS test cases

The followings tables show describe the details of 28 traditional test cases.

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| Test case ID | MSSITESS\_S01\_TC01\_MigratingSuccessfully |
| Description | This test case is designed to verify the ExportWeb and ImportWeb operations when migrating a site successfully. |
| Prerequisites | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| Test execution steps | * 1. The client calls the InitializeWebService operation to initialize the web service.   2. The client calls the ExportWeb operation with valid parameters.   3. The client calls the GetDocumentLibraryFileNames method to get tall the file names in the document library. |
| Cleanup | 1. <Common Cleanups> 2. Delete the site created upon the ImportWeb request (in step 3), if there is any. |

MSSITESS\_S01\_TC01\_MigratingSuccessfully

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| Test case ID | MSSITESS\_S01\_TC02\_ExportingMutiplePackages |
| Description | This test case is designed to verify the ExportWeb operation when exporting a site whose content exceeds 0x0018 megabytes, with cabSize parameter set in different values that smaller than 0x18. In this case, multiple content migration package files are expected to be exported. |
| Prerequisites | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. * “MSSITESS\_Site” has a subsite (“MSSITESS\_SpecialSubSite”) that contains a file which size exceeds 0x0018MB in the “MSSITESS\_SubsiteDocumentLibrary” folder of “MSSITESS\_SpecialSubSite”. |
| Test execution steps | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client calls the ExportWeb operation with webUrl in lowercase . 3. The client calls the ExportWeb operation with cabSize set to 1. 4. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library. |
| Cleanup | <Common Cleanups> |

MSSITESS\_S01\_TC02\_ExportingMultiplePackages

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC03\_ExportingEqualto0x18CabSize |
| **Description** | This test case is designed to verify the ExportWeb operation when exporting a site whose content equals 0x0018 megabytes, with cabSize parameter set out of the valid range (0 and 0x400 respectively). In this case, only one content migration package file is expected to be exported. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server.   “MSSITESS\_Site” has a subsite (“MSSITESS\_SpecialSubSite”) that contains a file which size exceeds 0x0018MB in the “MSSITESS\_SubSite\_DocumentLibrary” folder of “MSSITESS\_SpecialSubSite”. |
| **Test execution steps** | * 1. The client calls the InitializeWebService operation to initialize the web service.   2. The client calls the ExportWeb operation with cabSize set to 0.   3. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library.   4. The client calls the ExportWeb operation to invoke the ExportWeb operation with cabSize set to 0x0400.   5. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library.   6. The client calls the ExportWeb operation to invoke the ExportWeb operation with cabSize set to 0x0018.   7. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library. |
| **Cleanup** | <Common Cleanups> |

MSSITESS\_S01\_TC03\_ExportingEqualto0x18CabSize

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC04\_ExportingFailureInvalidExportUrl |
| **Description** | This test case is designed to verify the ExportWeb operation when the ExportWebResult equals to InvalidExportUrl. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | * + - 1. The client calls the InitializeWebService to initialize the web service.       2. The client calls the ExportWeb operation with invalid webUrl. |
| **Cleanup** | <Common Cleanups> |

MSSITESS\_S01\_TC04\_ExportingFailureInvalidExportUrl

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC05\_ExportingFailureExportFileNoAccess |
| **Description** | This test case is designed to verify the ExportWeb operation when the ExportWebResult equals to ExportFileNoAccess. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | * + - * 1. The client calls the InitializeWebService to initialize the web service.         2. The client calls the ExportWeb operation with invalid dataPath. |
| **Cleanup** | <Common Cleanups> |

MSSITESS\_S01\_TC05\_ExportingFailureExportFileNoAccess

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC06\_ExportingFailureOverwriteFailure |
| **Description** | This test case is designed to verify the ExportWeb operation when the ExportWebResult equals to ExportFileNoAccess. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService to initialize the web service. 2. The client calls the ExportWeb operation with valid dataPath. 3. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library. 4. The client calls the ExportWeb operation with overwrite set to false and exportJobName set to an existing job name. |
| **Cleanup** | <Common Cleanups> |

MSSITESS\_S01\_TC06\_ExportingFailureOverwriteFailure

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC07\_ImportingFailureOverwriteFailure |
| **Description** | This test case is designed to verify the ImportWeb operation when overwriting the log file to fail. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | * 1. The client calls the InitializeWebService to initialize the web service.   2. The client calls the ExportWeb operation with valid dataPath.   3. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library.   4. The client calls the ImportWeb operation with valid parameters.   5. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library.   6. The client calls the ImportWeb operation with overwrite set to false. |
| **Cleanup** | <Common Cleanups> |

MSSITESS\_S01\_TC07\_ImportingFailureOverwriteFailure

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC08\_ImportingFailureInvalidImportUrl |
| **Description** | This test case is designed to verify the ImportWeb operation when the ImportWebResult equals to InvalidImportUrl. |
| **Prerequisites** | * Common Prerequisites * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService to initialize the web service. 2. The client calls the ExportWeb operation with valid parameters. 3. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library. 4. The client calls the ImportWeb operation with invalid webUrl, and 4 is expected to be returned. |
| **Cleanup** | 1. <Common Cleanups> 2. Delete the site created upon the ImportWeb request (in step 4), if there is any. |

MSSITESS\_S01\_TC08\_ImportingFailureInvalidImportUrl

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC09\_ImportingFailureImportFileNoAccess |
| **Description** | This test case is designed to verify the ImportWeb operation when ImportWebResult equals to ImportFileNoAccess. |
| **Prerequisites** | * Common Prerequisites * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | * 1. The client calls the InitializeWebService to initialize the web service.   2. The client calls the ImportWeb operation with invalid dataFiles, and 5 is expected to be returned. |
| **Cleanup** | * 1. <Common Cleanups>   2. Delete the site created upon the ImportWeb request (in step 2), if there is any. |

MSSITESS\_S01\_TC09\_ImportingFailureImportFileNoAccess

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC10\_ImportingFailureLogFileNoAccess |
| **Description** | This test case is designed to verify the ImportWeb operation when the ImportWebResult equals to LogFileNoAccess. |
| **Prerequisites** | * Common Prerequisites * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService to initialize the web service. 2. The client calls the ExportWeb operation with valid parameters. 3. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library. 4. The client calls the ImportWeb operation with invalid logPath, and 11 is expected to be returned. |
| **Cleanup** | * + 1. <Common Cleanups>     2. Delete the site created upon the ImportWeb request (in step 4), if there is any. |

MSSITESS\_S01\_TC10\_ImportingFailureLogFileNoAccess

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC11\_ImportingFailureImportWebNotEmpty |
| **Description** | This test case is designed to verify the ImportWeb operation when the ImportWebResult equals to ImportWebNotEmpty. |
| **Prerequisites** | * Common prerequisites * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | * + 1. The client calls the InitializeWebService to initialize the web service.     2. The client calls the ExportWeb operation with valid parameters.     3. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library.     4. The client calls the ImportWeb operation with webUrl set to the site just exported, and 8 is expected to be returned. |
| **Cleanup** | * + - 1. <Common Cleanups>       2. Delete the site created upon the ImportWeb request (in step 4), if there is any. |

MSSITESS\_S01\_TC11\_ImportingFailureImportWebNotEmpty

|  |  |
| --- | --- |
| S01\_MigrateSite | |
| **Test case ID** | MSSITESS\_S01\_TC12\_ImportingFailureLogPathEmpty |
| **Description** | This test case is used to verify the ImportWeb operation when the logPath is omitted or empty. |
| **Prerequisites** | * Common prerequisites * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService to initialize the web service. 2. The client calls the ExportWeb operation with valid parameters. 3. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library. 4. The client calls the ImportWeb operation with logPath set to null, just exported, 1 is expected to be returned. 5. The client calls the ImportWeb operation with logPath set to empty string, just exported, a soup exception is expected to be returned. |
| **Cleanup** | 1. <Common Cleanups> 2. Delete the site created upon the ImportWeb request (in step 4), if there is any. |

MSSITESS\_S01\_TC12\_ImportingFailureLogPathEmpty

|  |  |
| --- | --- |
| S02\_ManageSubSite | |
| **Test case ID** | MSSITESS\_S02\_TC01\_ManagingSubsiteSuccessfully |
| **Description** | This test case is designed to verify GetSiteTemplates, CreateWeb and DeleteWeb operations when managing a subsite successfully. |
| **Prerequisites** | * <Common prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService to initialize the web service. 2. The client calls the GetSiteTemplates operation with valid parameters. 3. The client calls the CreateWeb operation with valid parameters. 4. The client calls GetWebProperties method to get properties of the create web. 5. The client calls the DeleteWeb operation. |
| **Cleanup** | Delete the subsite created upon the CreateWeb request (in step 3), if there is any. |

MSSITESS\_S02\_TC01\_ManagingSubsiteSuccessfully

|  |  |
| --- | --- |
| S02\_ManageSubSite | |
| **Test case ID** | MSSITESS\_S02\_TC02\_ManagingSubsiteWithoutOptionalParameters |
| **Description** | This test case is designed to verify GetSiteTemplates, CreateWeb and DeleteWeb operations when managing a subsite without optional parameters. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService to initialize the web service. 2. The client calls the GetSiteTemplates operation without optional parameters. 3. The client calls the CreateWeb operation without optional parameters. 4. The client calls GetWebProperties method to get properties of the create web. 5. The client calls GetWebProperties method to get properties of the parent web. 6. The client calls the DeleteWeb operation. |
| **Cleanup** | Delete the subsite created upon the CreateWeb request (in step 3), if there is any. |

MSSITESS\_S02\_TC02\_ManagingSubsiteWithoutOptionalParameters

|  |  |
| --- | --- |
| S02\_ManageSubSite | |
| **Test case ID** | MSSITESS\_S02\_TC03\_CreateWebFailureUrlAlreadyInUse |
| **Description** | This test case is designed to verify GetSiteTemplates and CreateWeb operations when the requested URL is already in use. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService to initialize the web service. 2. The client calls the GetSiteTemplates operation with valid parameters. 3. The client tries to invoke the CreateWeb operation with url parameter set to the web which is already used (i.e. Subsite1). |
| **Cleanup** | Delete the subsite created upon the CreateWeb request (in step 3), if there is any. |

MSSITESS\_S02\_TC03\_CreateWebFailureUrlAlreadyInUse

|  |  |
| --- | --- |
| S02\_ManageSubSite | |
| **Test case ID** | MSSITESS\_S02\_TC04\_CreateWebFailureTemplateNotExist |
| **Description** | This test case is designed to verify GetSiteTemplates and CreateWeb operations when the requested template does not exist. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | * + 1. The client calls the InitializeWebService operation to initialize the web service.     2. The client calls the GetSiteTemplates operation with valid parameters. |
| **Cleanup** | Delete the subsite created upon the GetSiteTemplates request (in step 2), if there is any. |

MSSITESS\_S02\_TC04\_CreateWebFailureTemplateNotExist

|  |  |
| --- | --- |
| S02\_ManageSubSite | |
| **Test case ID** | MSSITESS\_S02\_TC05\_DeleteWebFailureNonExistentUrl |
| **Description** | This test case is designed to verify DeleteWeb operation when webUrl is non-existent. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | * + - 1. The client calls the InitializeWebService operation to initialize the web service.       2. The client calls the DeleteWeb operation to delete a non-existent subsite of the current site. |
| **Cleanup** | N/A |

MSSITESS\_S02\_TC05\_DeleteWebFailureInvalidUrl

|  |  |
| --- | --- |
| S02\_ManageSubSite | |
| **Test case ID** | MSSITESS\_S02\_TC06\_CreateWebWithZeroFalse |
| **Description** | This test case is designed to verify the properties of CreateWeb operation. |
| **Prerequisites** | N/A |
| **Test execution steps** | * + - * 1. The client calls InitializeWebService operation to initialize the web service.         2. The client calls the GetSiteTemplates operation with valid parameters.         3. The client calls the CreateWeb operation to create web with language, locale, collationLocale zero and uniquePermissions, anonymous, presence false.         4. The client calls GetWebProperties method to get properties of the create web.         5. The client calls GetWebProperties method to get properties of the parent web.         6. The client calls DeleteWeb operation. |
| **Cleanup** | N/A |

MSSITESS\_S02\_TC06\_CeateWebWithZeroFalse

|  |  |
| --- | --- |
| S02\_ManageSubSite | |
| **Test case ID** | MSSITESS\_S02\_TC07\_GetSiteTemplateNotInstalledLCID |
| **Description** | This test case is designed to verify GetSiteTemplates operation. |
| **Prerequisites** | N/A |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client calls the GetSiteTempaltes operation without optional parameters. |
| **Cleanup** | N/A |

MSSITESS\_S02\_TC07\_GetSiteTemplateNotInstalledLCID

|  |  |
| --- | --- |
| S02\_ManageSubSite | |
| **Test case ID** | MSSITESS\_S02\_TC08\_GetSiteTemplatesSuccessfully |
| **Description** | This test case is designed to verify the successful status of GetSiteTemplates operation. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | * + - * 1. The client calls the InitializeWebService to initialize the web service.         2. The client calls the GetSiteTemplates operation with valid parameters. |
| **Cleanup** | Delete the subsite created upon the GetSiteTemplates request (in step 2), if there is any. |

MSSITESS\_S02\_TC08\_GetSiteTemplatesSuccessfully

|  |  |
| --- | --- |
| S03\_GetUpdatedFormDigest | |
| **Test case ID** | MSSITESS\_S03\_TC01\_GetUpdatedFormDigest |
| **Description** | This test case is designed to verify the GetUpdatedFormDigest operation. |
| **Prerequisites** | Common Prerequisites |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client calls the GetUpdatedFormDigest operation. 3. The client calls the GetUpdatedFormDigest operation again, and the returned security validation is expected to be different with the last one. |
| **Cleanup** | N/A |

MSSITESS\_S03\_TC01\_GetUpdatedFormDigest

|  |  |
| --- | --- |
| S03\_GetUpdatedFormDigest | |
| **Test case ID** | MSSITESS\_S03\_TC02\_GetUpdatedFormDigestInformation |
| **Description** | This test case is designed to verify the GetUpdatedFormDigestInformation operation. |
| **Prerequisites** | Common Prerequisites |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client calls the GetUpdatedFormDigestInformation operation. 3. The client calls the GetUpdatedFormDigestInformation operation again, and the returned security validation is expected to be different with the last one. |
| **Cleanup** | N/A |

MSSITESS\_S03\_TC02\_GetUpdatedFormDigestInformation

|  |  |
| --- | --- |
| S04\_ExportSolution | |
| **Test case ID** | MSSITESS\_S04\_TC01\_ExportSolutionSucceed |
| **Description** | This test case is designed to verify the successful status of ExportSolution. |
| **Prerequisites** | Common Prerequisites |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client calls the ExportSolution operation. 3. The client calls the GetDocumentLibraryFileNames operation to get all the file names in the solution library. 4. The client calls the ExportSolution operation again to export the content related to a site to the solution gallery. 5. The client calls the GetDocumentLibraryFileNames method to get all the file names in the solution library. |
| **Cleanup** | N/A |

MSSITESS\_S04\_TC01\_ExportSolutionSucceed

|  |  |
| --- | --- |
| S05\_ExportWorkflowTemplate | |
| **Test case ID** | MSSITESS\_S05\_TC01\_ExportWorkflowTemplateSucceed |
| **Description** | This test case is designed to verify the successful status of ExportWorkflowTemplate operation. |
| **Prerequisites** | Common Prerequisites |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client calls the ExportWorkflowTemplate operation 3. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library. 4. The client calls the ExportWorkflowTemplate operation to export a workflow template as a site solution to the specified document library. 5. The client calls the GetDocumentLibraryFileNames method to get all the file names in the document library. |
| **Cleanup** | N/A |

MSSITESS\_S05\_TC01\_ExportWorkflowTemplateSucceed

|  |  |
| --- | --- |
| S05\_ExportWorkflowTemplate | |
| **Test case ID** | MSSITESS\_S05\_TC02\_ExportWorkflowTemplateInvalidLibrary |
| **Description** | This test case is designed to verify ExportWorkflowTemplate when the library name is invalid. |
| **Prerequisites** | Common Prerequisites |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client tries to call the ExportWorkflowTemplate operation with valid template name and invalid document library path. |
| **Cleanup** | N/A |

MSSITESS\_S05\_TC02\_ExportWorkflowTemplateInvalidLibrary

|  |  |
| --- | --- |
| S05\_ExportWorkflowTemplate | |
| **Test case ID** | MSSITESS\_S05\_TC03\_ExportWorkflowTemplateInvalidTemplate |
| **Description** | This test case is used to verify the ExportWorkflowTemplate when Template name is invalid. |
| **Prerequisites** | Common Prerequisites |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client tries to call the ExportWorkflowTemplate operation with invalid Workflow Template. |
| **Cleanup** | N/A |

MSSITESS\_S05\_TC03\_ExportWorkflowTemplateInvalidTemplate

|  |  |
| --- | --- |
| S06\_GetSite | |
| **Test case ID** | MSSITESS\_S06\_TC01\_GetSiteSucceed |
| **Description** | This test case is designed to verify the successful status of GetSite. |
| **Prerequisites** | * <Common Prerequisites> * There is a team site (“MSSITESS\_Site”) in the site collection on the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client calls the GetSiteGuid operation to get the site collection identifier of the site collection. 3. The client calls the SetUserCodeEnabled operation to set whether user code is enabled for the site collection. 4. The client calls the GetSite operation with valid SiteUrl. 5. The client calls the SetUserCodeEnabled operation to set whether user code is enabled for the site collection. 6. The client calls the GetSite operation with valid SiteUrl. |
| **Cleanup** | N/A |

MSSITESS\_S06\_TC01\_GetSiteSucceed

|  |  |
| --- | --- |
| S07\_HTTPStatusCode | |
| **Test case ID** | MSSITESS\_S07\_TC01\_HttpStatusCodesFault |
| **Description** | This test case is designed to verify the protocol server fault by using HTTP Status Codes. A HTTP status code "Unauthorized" is expected to throw. |
| **Prerequisites** | The client uses an unauthenticated user account to login the server. |
| **Test execution steps** | 1. The client calls the InitializeWebService operation to initialize the web service. 2. The client tries to call the GetSiteTemplates operation. |
| **Cleanup** | N/A |

MSSITESS\_S07\_TC01\_HttpStatusCodesFault