opentext™

ChangeMan ZDD

COM Programming Interface Guide

8.3

Table of Contents

1. About this Guide

This guide describes how to access ChangeMan ZDD functionality, using COM automation, from your own programs and scripts.

ChangeMan ZDD is a network file system that operates on a PC networked with a z/OS® operating system. From your PC, you can access data sets, job output, and ChangeMan® ZMF components that reside on a z/OS server.

Before You Begin

The COM programming interface documented in this manual is still supported in ChangeMan ZDD 8.2 Patch 4; however, a new .NET programming interface was introduced with the 6.1 release. It is recommended that you use the .NET programming interface for new programs and scripts, for the following reasons:

- The .NET interface contains more functionality than the COM interface.
- The .NET framework provides a more powerful programming environment.

Refer to the new *ChangeMan ZDD .NET Programming Interface Guide* for information on how to use the .NET interface.

Audience and scope:

This manual is intended for System Administrators or any other users who want to perform ChangeMan ZDD operations from their own programs and scripts that support COM (Component Object Model) objects.

By accessing the functionality of ChangeMan ZDD, you can simplify some common tasks, such as:

- Automating configuration tasks for setting up ChangeMan ZDD on multiple desktops.
- Logging on to a z/OS server from your program or script.
- Submitting JCL to a z/OS server from your program or script.

Organization

This guide is organized as follows:

This chapter…	Contains this information…
1	Information about this guide.
2	Overview of ChangeMan ZDD and the Automation interface.
3	Description of the ChangeMan ZDD object model and how to use the Automation interface to access ChangeMan ZDD functions.
4	Examples of how to use ChangeMan ZDD functions from VBScript and JScript®.

Guide to ChangeMan ZDD Documentation

The following sections provide basic information about ChangeMan ZDD documentation. These manuals are available through the Micro Focus website at `https://www.microfocus.com/documentation/changeman-zmf-client-pack/`.

Documentation Suite

The ChangeMan ZDD documentation set includes the following manuals in PDF format.

Manual	Description
ChangeMan ZDD User's Guide	Explains how to:
	Install and configure the client components on your PC
	Access and perform operations on mainframe data from your desktop
ChangeMan ZDD .NET Programming Interface Guide	Describes how to use the .NET programming interface to access ChangeMan ZDD functionality from your own programs and scripts.

Manual	Description
ChangeMan ZDD COM Programming Interface Guide	Describes how to access ChangeMan ZDD functionality, using COM Automation, from your own programs and scripts.
ChangeMan ZDD Tools Guide	Describes the following tools that you can use to assist in your development:
	ChangeMan Edit
	ChangeMan Diff
	These tools use the Template Manager to control how your code is displayed.
ChangeMan ZDD Server Installation Guide	Instructions for installing the server components of ChangeMan ZDD on the mainframe.
ChangeMan ZDD Edit Reference Card	Provides a summary of keyboard shortcuts that you can use with ZDD editing facilities.
SER10TY User's Guide	Instructions for applying licenses to enable ChangeMan ZDD servers on the mainframe.

Related Documents

The following documents provide additional information that may be useful to ChangeMan ZDD users.

Manual	Description
ChangeMan ZMF User's Guide	Provides instructions for using functions and facilities of ChangeMan ZMF to manage changes to application software. Many of these functions are available through ChangeMan ZDD.
ChangeMan ZMF Messages Guide	Provides explanations for informational, warning, and error messages for ChangeMan ZMF. These messages may be displayed when accessing ChangeMan ZMF through ChangeMan ZDD.
ChangeMan ZMF: XML Services User's Guide	Describes how to use XML Services, an XML programming interface to ChangeMan ZMF.

Using the Manuals

The ChangeMan ZDD manuals use the Adobe Portable Document Format (PDF). To view PDF files, use Adobe® Reader®, which is freely available from www.adobe.com.

> **Tip**
>
> Be sure to download the *full version* of Reader. The more basic version does not include the search feature.

This section highlights some of the main Reader features. For more detailed information, see the Adobe Reader online help system.

The PDF manuals include the following features:

- **Bookmarks.** All of the manuals contain predefined bookmarks that make it easy for you to quickly jump to a specific topic. By default, the bookmarks appear to the left of each online manual.
- **Links.** Cross-reference links within a manual enable you to jump to other sections within the manual and to other manuals with a single mouse click. These links appear in blue.
- **Printing.** While viewing a manual, you can print the current page, a range of pages, or the entire manual.
- **Advanced search.** Starting with version 6, Adobe Reader includes an advanced search feature that enables you to search across multiple PDF files in a specified directory. (This is in addition to using any search index created by Adobe Catalog—see step 3 below.)

To search within multiple PDF documents at once, perform the following steps (requires Adobe Reader version 6 or higher):

1. In Adobe Reader, select Edit | Search (or press CTRL+F).
2. In the text box, enter the word or phrase for which you want to search.
3. Select the **All PDF Documents in** option, and browse to select the folder in which you want to search.
4. Optionally, select one or more of the additional search options, such as **Whole words only** and **Case-Sensitive**.
5. Click the **Search** button.

> **Note**
>
> Optionally, you can click the Use Advanced Search Options link near the lower right corner of the application window to enable additional, more powerful search options. (If this link says Use Basic Search Options instead, the advanced options are already enabled.) For details, see Adobe Reader's online help.

Accessing Online Help

The online help is the primary source of information about ChangeMan ZDD. The online help includes:

- Overviews of key elements within the application
- Detailed procedures for completing tasks
- Context-sensitive descriptions of fields and buttons

Viewing Help Topics

You can view Help topics by clicking the Help button in the dialog box in which you are working. From there, you can do the following:

To	Do This
View a list of topics in the Contents	Click **Contents**.
Locate a topic in the Index	Click **Index**.
Locate an overview or procedure by searching on a word or words	Click **Search**.

Viewing Context-Sensitive Help

To view field-level help for an item in a dialog box:

- Click ? and then click the field or button for which you want a description. or
- Position the cursor in the field and press F1.

Accessing Help for the ChangeMan Utilities

When you are using the ChangeMan Edit and ChangeMan Diff utilities, you can open the online Help by:

- Pressing F1 from anywhere in the screen.
- Holding the left or right mouse button down on a toolbar icon or menu command and pressing F1.

Updates to this Guide

See the Readme file at `https://www.microfocus.com/documentation/changeman-zmf-client-pack/` for the latest updates and corrections for this manual.

2. Introduction

What is ChangeMan ZDD?

ChangeMan ZDD is a software infrastructure technology that simulates a network file system. It provides seamless access to data sets, jobs, and ChangeMan ZMF® components on a z/OS® system, from a Windows® platform. No special execution environment or programming interface is required. Data sets, job output, and ChangeMan ZMF components are accessed as though they were local files on your PC or files on a Windows network.

For a more detailed introduction to ChangeMan ZDD, see the *ChangeMan® ZDD User's Guide*.

What is Automation?

Automation is a technology that allows you to access the functionality of an application, such as ChangeMan ZDD, and use it in your own programs or scripts. Automation is based upon the *Component Object Model* (COM). COM is a standard software architecture that separates code into self-contained objects or components.

Using Automation, ChangeMan ZDD exposes its functionality as a set of programmable *objects*. Each object can be programmatically examined and controlled. Examples of ChangeMan ZDD objects are: *network*, *servers*, and *folders*.

Each object exposes a set of *properties* and *methods*. A *property* is an attribute of an object that can be set or retrieved. A *method* is a function that performs some action on an object. For a server object, examples of properties are *server name* or *IP address*; examples of methods are *logon* or *logoff*.

A special type of object is a *collection* object, which contains a set of other objects. A *servers* object is a collection of *server* objects, and a *folders* object is a collection of *folder* objects.

The ChangeMan ZDD Automation interface allows ChangeMan ZDD operations to be performed from Visual Basic®, C++, VBScript, JScript®, or any other language that supports COM objects. Following are some typical operations you can perform from a program or script:

- Configure ChangeMan ZDD for a new user See Configuring ChangeMan ZDD for a new user.
- Submit JCL to a server See Submitting JCL to a Server.
- Log on to a server See Logging on to a Server.

In this document, examples are shown in both VBScript and JScript. Other languages may be used as well, but examples are not given.

Security

ChangeMan ZDD is compatible with RACF®, CA-ACF2®, and CA-Top Secret®.

Access to mainframe objects and functions is granted through your mainframe security system. You are required to provide your user ID and password in ChangeMan ZDD to connect to the mainframe.

The operation of ChangeMan ZDD does not affect the existing operation of either mainframe-based applications or PC network operations.

Compatibility

PC Requirements

- Windows® operating system (refer to the Readme for the supported versions)
- 10 megabytes (MB) of available disk space
- CD-ROM drive or access to a CD-ROM over a network
- VGA or higher-resolution display adapter
- Microsoft® Mouse or compatible pointing device

Mainframe Server Requirements

- ChangeMan ZDD server installed on the mainframe LPARs to be accessed by ChangeMan ZDD on your PC.
- IBM® z/OS® operating system (any version supported by IBM).
- TCP/IP must be installed and running.

ChangeMan ZMF Requirements

One of the following releases are required for accessing ChangeMan ZMF functionality from your program or script:

- ChangeMan ZMF 8.1 - any release
- ChangeMan ZMF 7.1 - any release

- ChangeMan ZMF 6.1 - any release

> **Note**
>
> When using ChangeMan ZDD with earlier releases of ChangeMan ZMF, only the functionality supported within that ChangeMan ZMF release will be available.

Changes

A number of class names in ZDD's COM programming interface have been renamed so that they are now consistent with the .NET programming interface. This has been a source of confusion in the past.

These class names are primarily used to describe the ZDD COM interface in the documentation.

This has no impact on customer scripts or Visual Basic because these names are not used in scripting languages. The names would only need to be changed if you write a C++ program to invoke the COM interface.

The following class names have been renamed:

- ZosChangeManFolder --> ZosChangeManInstance
- ZosChangeManFolders --> ZosChangeManInstance
- ZosFileExtension --> ZosFileExtensionMapping
- ZosFileExtensions --> ZosFileExtensionMappings
- ZosDataType --> ZosFileFormatMapping
- ZosDataTypes --> ZosFileFormatMappings
- ZosFilters --> ZosNameFilters
- ZosLibType --> ZosLibypeMapping
- ZosLibTypes --> ZosLibTypeMappings
- ZosPrefix --> ZosPrefixMapping
- ZosPrefixes --> ZosPrefixMappings

In the event that someone is actually using C++ to invoke the COM interface, then they can either change the names above, or simply add the following definitions to the program:

```
#define ZosChangeManFolder ZosChangeManInstance
#define ZosChangeManFolders ZosChangeManInstance
#define ZosFileExtension ZosFileExtensionMapping
#define ZosFileExtensions ZosFileExtensionMappings
#define ZosDataType ZosFileFormatMapping
#define ZosDataTypes ZosFileFormatMappings
#define ZosFilters ZosNameFilters
#define ZosLibType ZosLibypeMapping
#define ZosLibTypes ZosLibTypeMappings
#define ZosPrefix ZosPrefixMapping
#define ZosPrefixes ZosPrefixMappings
```

3. Using the Programming Interface

This chapter describes the ChangeMan ZDD object model and how to access ChangeMan ZDD functionality, using the Automation interface, from your own programs and scripts. Examples are shown in both VBScript and JScript.

See What is Automation for more information.

Naming Conventions

The examples in this document use variable name prefixes to indicate the type of variable, as follows:

Prefix	Variable
obj	Object
str	String
var	Variant
n	Integer
b	Boolean

Connecting to ChangeMan ZDD

Connecting to ChangeMan ZDD is similar to connecting to any COM or ActiveX object:

For . . .	You would use . . .
Visual Basic or VBScript	CreateObject
JScript	new ActiveXObject
Visual C++	CoCreateInstance

For examples of connecting to ChangeMan ZDD, see ZosNetwork.

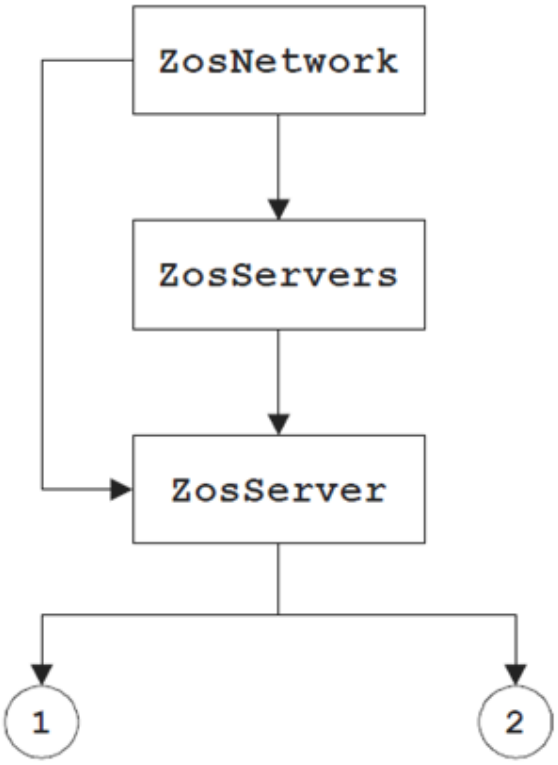
Object Model

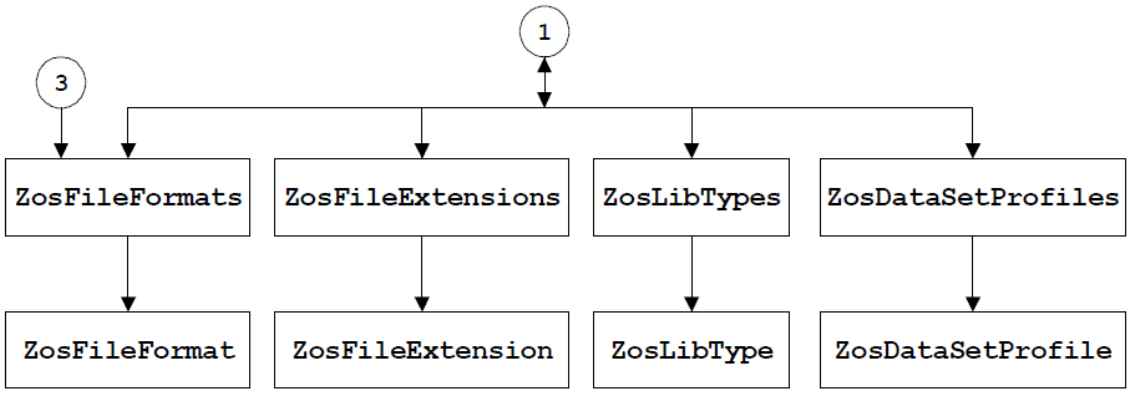
The following table summarizes the types of objects available in the ChangeMan ZDD object model:

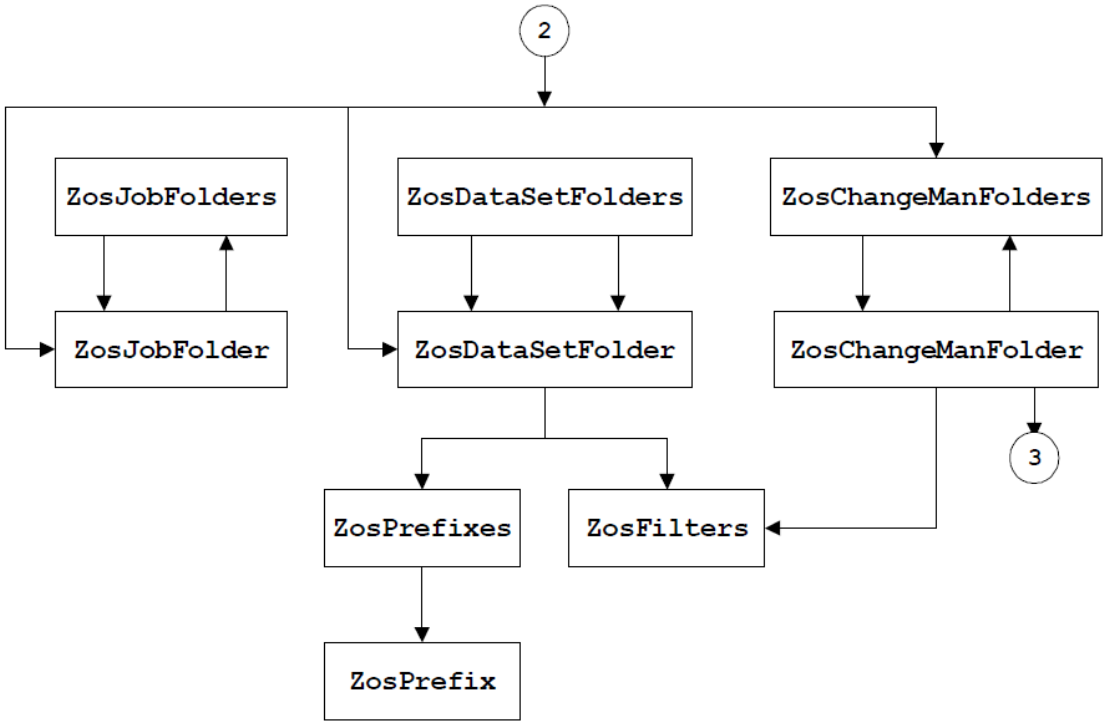
Object	Description
ZosChangeManInstance	A ChangeMan ZMF folder. See ZosChangeManInstance.
ZosChangeManInstances	Collection of all ChangeMan ZMF folders on the same server. See ZosChangeManInstances.
ZosDataSetFolder	A data set folder. See ZosDataSetFolder.
ZosDataSetFolders	Collection of all data set folders with the same parent folder. See ZosDataSetFolders.
ZosDataSetProfile	A data set profile. See ZosDataSetProfile.
ZosDataSetProfiles	Collection of all data set profiles for a server. See ZosDataSetProfiles.
ZosFileExtensionMapping	A file extension definition. See ZosFileExtensionMapping.
ZosFileExtensionMappings	Collection of all file extension definitions for a server. See ZosFileExtensionMappings.
ZosFileFormatMapping	A file format mapping. See ZosFileFormatMapping.
ZosFileFormatMappings	Collection of file format mappings for a server or ChangeMan instance. See ZosFileFormatMappings.
ZosJobFolder	A job folder. See ZosJobFolder.
ZosJobFolders	Collection of all job folders with the same parent folder. See ZosJobFolders.
ZosLibypeMapping	A library type definition. See ZosLibypeMapping.
ZosLibypeMappings	Collection of all library type definitions for a server. See ZosLibypeMappings.
ZosNameFilters	Collection of all filters for a folder. See ZosNameFilters.
ZosNetwork	Entire Serena™ Network. See ZosNetwork.
ZosPrefixMapping	A data set name prefix definition. See ZosPrefixMapping.
ZosPrefixMappings	Collection of all data set name prefix definitions for a folder. See ZosPrefixMappings.

Object	Description
ZosServer	A server. See ZosServer.
ZosServers	Collection of all servers in the network. See ZosServers.

The ChangeMan ZDD object model is illustrated in the following diagrams. The *ZosNetwork* object is always the starting point. All of the other objects are obtained as properties of another object. The arrows show how each object is obtained from another object







Collections

You can iterate through any of the collection objects in Visual Basic or VBScript using the "For Each…Next" statement, or in JScript using the "Enumerator" object:

Visual Basic or VBScript:

```
Dim objServers
    Dim objServer
    For Each objServer in objServers
    …
    Next
```

JScript:

```
var objServers;
    var objServer;
    var objEnum;
    objEnum = new Enumerator(objServers);
    for (; !objEnum.atEnd(); objEnum.moveNext())
    {
    objServer = objEnum.item();
    …
    }
```

4. Reference

Reference Information

The ZDD COM Programming Interface includes the following objects:

- ZosChangeManInstances
- ZosChangeManInstances
- ZosDataSetFolders
- ZosDataSetFolders
- ZosDataSetProfile
- ZosDataSetProfiles
- ZosFileExtensionMapping
- ZosFileExtensionMappings
- ZosFileFormatMapping
- ZosFileFormatMappings
- ZosJobFolder
- ZosJobFolders
- ZosLibypeMapping
- ZosLibypeMappings
- ZosNameFilters
- ZosNetwork
- ZosPrefixMapping
- ZosPrefixMappings
- ZosServer
- ZosServers

ZosChangeManInstance

The ZosChangeManInstance object represents a single ChangeMan ZMF folder. This object can be obtained using the ChangeManInstance property of ZosServer or the Item property of ZosChangeManInstances.

ZosChangeManInstance Properties

ZosChangeManInstance exposes the following properties:

Property	Type	R/W	Description
Name	String	R	Name of the folder.
Path	String	R	Full path name of the folder.
Port	Integer (long)	R/W	I/P port number for ChangeMan ZMF instance
Description	String	R/W	ChangeMan ZMF description
Filters	Object (IZosNameFilters)	R	Collection of all application name filters for folder. Default is all applications.
FileFormats	Object (IZosFileFormatMappings)	R	Collection of file format mappings for ChangeMan ZMF components
IsHidden	Boolean	R / W	Indicates that this ChangeMan is hidden in the File Explorer and the ZDD user interface. This is a user-specific setting.

ZosChangeManInstance Methods

ZosChangeManInstance exposes the following methods:

SubmitXml Method

Submits an XML request to the XML Services processor (for ChangeMan services). The bSuppressMessage parameter is optional. It is used to indicate whether message boxes should be suppressed. It is useful for running scripts in an automated tool where there is nobody present to press the OK button on a message box.

```
SubmitXml (
    strInputFileName,
    strOutputFileName,
    bSuppressMessage
    )
```

Parameters

`bSuppressMessage` - Specifies whether to suppress or display messages.

To suppress message boxes, specify:

`bSuppressMessage = true`

To display message boxes, specify:

`bSuppressMessage = false`

If this parameter is not specified, it defaults to false and message boxes will be displayed normally.

ZosChangeManInstances

The ZosChangeManInstances object is a collection of all ChangeMan ZMF folders for a server. This object is obtained using the ChangeManInstances property of the ZosServer object.

ZosChangeManInstances Properties

ZosChangeManInstances exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosChangeManInstance)	R	Folder with specified name or index.
Count	Integer (long)	R	Number of objects in collection.

ZosChangeManInstances Methods

ZosChangeManInstances exposes the following methods:

Add Method

Adds a new ChangeMan ZMF instance folder.

```
Add (
    strName,
    nPort,
    strDescription
    )
```

Returns

```
Object (IZosChangeManInstance)
```

Refresh Method

Refreshes collection.

```
Refresh ()
```

Remove Method

Deletes a ChangeMan ZMF instance folder.

```
Remove (
    strName
    )
```

ZosDataSetFolder

The ZosDataSetFolder object represents a single data set folder. This object can be obtained using the DataSetFolder property of ZosServer or the Item property of ZosDataSetFolders.

ZosDataSetFolder Properties

ZosDataSetFolder exposes the following properties:

Property	Type	R/W	Description
Name	String	R	Name of the folder.
Path	String	R	Full path name of the folder.
Subfolders	Object (IZosDataSetFolders)	R	Collection of all subfolders for this folder.
Filters	Object (IZosNameFilters)	R	Collection of all data set name filters for folder.
MemberFilters	Object (IZosNameFilters)	R	Collection of member name filters for libraries under this folder.
Prefixes	Object (IZosPrefixMappings)	R	Collection of all data set name prefixes for folder.

ZosDataSetFolders

The ZosDataSetFolders object is a collection of all data set folders with the same parent folder. This object is obtained using the Subfolders property of the ZosDataSetFolder object.

ZosDataSetFolders Properties

ZosDataSetFolders exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosDataSetFolder)	R	Folder with specified name or index.
Count	Integer (long)	R	Number of objects in collection.

ZosDataSetFolders Methods

ZosDataSetFolders exposes the following methods:

Add Method

Adds a new folder.

```
Add (
     strFolderName
     )
```

Returns

```
Object  (
     IZosDataSetFolder
     )
```

Refresh Method

Refreshes collection.

```
Refresh ()
```

Remove Method

Deletes a folder.

```
Remove (
     strFolderName
     )
```

ZosDataSetProfile

The ZosDataSetProfile object represents a single file extension definition. This object can be obtained using the Item property of ZosDataSetProfiles.

ZosDataSetProfile Properties

ZosDataSetProfile exposes the following properties:

Property	Type	R/W	Description
DataSetName	String	R	Data set name pattern.
DataSetType	String	R	Data set type: "SEQ", "PDS", or "PDSE".
RecordFormat	String	R	Record format.
RecordLength	Integer (short)	R	Record length.
BlockSize	Integer (short)	R	Block size.
DataClass	String	R	SMS data class.
StorageClass	String	R	SMS storage class.
ManagementClass	String	R	SMS management class.
SpaceUnit	String	R	Space unit: "CYL", "TRK", or "BLK".
PrimarySpace	Integer (long)	R	Primary space quantity.
SecondarySpace	Integer (long)	R	Secondary space quantity.
DirectoryBlocks	Integer (long)	R	PDS directory blocks.
UnitName	String	R	Unit name.
Volume	String	R	Volume serial number.
PdseVersion	Integer (short)	R	PDSE version number: 0 (default), 1, 2
MaxGens	Integer (long)	R	Maximum number of PDSE member generations

ZosDataSetProfiles

The ZosDataSetProfiles object is a collection of all data set profiles for a server. This object is obtained using the DataSetProfiles property of the ZosServer object.

ZosDataSetProfiles Properties

ZosDataSetProfiles exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosDataSetProfile)	R	Data set profile with specified index or data set name pattern.
Count	Integer (long)	R	Number of objects in collection.

ZosDataSetProfiles Methods

ZosDataSetProfiles exposes the following methods:

Add Method

Adds a new data set profile. Data set type can be "SEQ", "PDS", or "PDSE". Space unit can be "CYL", "TRK", or "BLK".

Index indicates position for new item. Specify –1 to insert at end.

```
Add (
    nIndex,
    strDataSetName,
    strDataSetType,
    strRecordFormat,
    nRecordLength,
    nBlockSize,
    strDataClass,
    strStorageClass,
    strManagementClass,
    strSpaceUnit,
    nPrimarySpace,
    nSecondarySpace,
    nDirectoryBlocks,
    strUnitName,
    strVolume,
    nPdseVersion,
    MaxGens
    )
```

Returns

`Object  (IZosDataSetProfile)`

Move Method

Changes the order of data set profiles.

```
Move (
    nIndexTo,
    nIndexFrom
    )
```

Refresh Method

Refreshes collection.

```
Refresh ()
```

Remove Method

Deletes a deta set profile, specified by index or data set name pattern.

```
Remove (
    varIndex
    )
```

ZosFileExtensionMapping

The ZosFileExtensionMapping object represents a single file extension definition. This object can be obtained using the Item property of ZosFileExtensionMappings.

ZosFileExtensionMapping Properties

ZosFileExtensionMapping exposes the following properties:

Property	Type	R/W	Description
DataSetName	String	R	Data set name pattern.
FileExtension	String	R	File extension.

ZosFileExtensionMappings

The ZosFileExtensionMappings object is a collection of all file extension definitions for a server. This object is obtained using the FileExtensions property of the ZosServer object.

ZosFileExtensionMappings Properties

ZosFileExtensionMappings exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosFileExtension Mapping)	R	File extension definition with specified index or data set name pattern.
Count	Integer (long)	R	Number of objects in collection.

ZosFileExtensionMappings Methods

ZosFileExtensionMappings exposes the following methods:

Add

Adds a new file extension definition. Index indicates position for new item. Specify –1 to insert at end.

```
Add (
    nIndex,
    strDataSetName,
    strFileExtension
    )
```

Returns

`Object (IZosFileExtensionMapping)`

Move

Changes the order of file extension definitions.

```
Move (
    nIndexTo,
    nIndexFrom
    )
```

Refresh

```
Refresh () | | Refreshes collection.|
```

Remove

Deletes a file extension, specified by index or data set name pattern.

```
Remove (
    varIndex
    )
```

ZosFileFormatMapping

The ZosFileFormatMapping object represents a single file format definition. This object can be obtained using the Item property of ZosFileFormatMappings.

ZosFileFormatMapping Properties

ZosFileFormatMapping exposes the following properties:

Property	Type	R/W	Description
Name	String	R	Name pattern.
FileFormat	String	R	File format:
			"AT" (ASCII Text)
			"AD" (ASCII Data)
			"ET" (EBCDIC Text)
			"ED" (EBCDIC Data)
			"UT" (UNICODE Text)
			"VT" (UTF-8 Text)
			"B" (Binary)
			"BT" (Binary CRLF)

Examples of using ZosFileFormatMapping:

Visual Basic or VBScript:

```
Dim objFileFormat
    Dim strName
    Dim strDataType
    strName = objFileFormat.DataSetName
    strDataType = objFileFormat.DataType
```

JScript:

```
var objFileFormat;
    var strName;
    var strDataType;
    strName = objFileFormat.DataSetName;
    strDataType = objFileFormat.DataType;
```

ZosFileFormatMappings

The ZosFileFormatMappings object is a collection of file format definitions for a server or

ChangeMan instance. This object is obtained using the DataSetFileFormats property or the UnixFileFormats property of the ZosServer object. For ChangeMan instances, it can be obtained using the FileFormats property of the ZosChangeManInstance object.

ZosFileFormatMappings Properties

ZosFileFormatMappings exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosFileFormatMapping)	R	File format definition with specified index or name pattern.
Count	Integer (long)	R	Number of objects in collection.

ZosFileFormatMappings Methods

ZosFileFormatMappings exposes the following methods:

Add Method

Adds a new file format definition.

```
Add (
    nIndex,
    strName,
    strFileFormat
    )
```

Parameters

`nIndex` - Index indicates position for new item. Specify –1 to insert at end.

`strFileFormat` - One of the following values that specifies the file format: - "AT" (ASCII Text) - "AD" (ASCII Data) "ET" (EBCDIC Text) - "ED" (EBCDIC Data) - "UT" (UNICODE Text) - "VT" (UTF-8 Text) - "B" (Binary) - "BT" (Binary CRLF)

Returns

```
Object (IZosFileFormatMapping)
```

Move Method

Changes the order of file format definitions.

```
Move (
    nIndexTo,
    nIndexFrom
    )
```

Refresh Method

Refreshes collection.

```
Refresh ()
```

Remove Method

Deletes a file format, specified by index or name pattern.

```
Remove (
    varIndex
    )
```

Examples of using ZosFileFormatMappings:

Visual Basic or VBScript:

```
Dim objFileFormats
    objFileFormats.Add -1, "**.BINARY", "B"
    objFileFormats.Remove "**.TRASH"
    objFileFormats.Move 4,2
```

JScript:

```
var objFileFormats;
    objFileFormats.Add(-1, "**.BINARY", "B");
    objFileFormats.Remove("**.TRASH");
    objFileFormats.Move(4,2);
```

ZosJobFolder

The ZosJobFolder object represents a single job folder. This object can be obtained using the JobFolder property of ZosServer or the Item property of ZosJobFolders.

ZosJobFolder Properties

ZosJobFolder exposes the following properties:

Property	Type	R/W	Description
Name	String	R	Name of the folder.
Path	String	R	Full path name of the folder.
Subfolders	Object (IZosJobFolders)	R	Collection of all subfolders for this folder.
QueryType	String	R/W	Type of job query. Can be "QN" (name), "QP" (prefix), "AP" (active prefix), "AU" (active userid), or "A" (all active).
QueryArgument	String	R/W	Query search argument is job name, prefix, or userid. Default argument is logged-on userid.

ZosJobFolders

The ZosJobFolders object is a collection of all job folders with the same parent folder. This object is obtained using the Subfolders property of the ZosJobFolder object.

ZosJobFolders Properties

ZosJobFolders exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosJobFolder)	R	Folder with specified name or index.
Count	Integer (long)	R	Number of objects in collection.

ZosJobFolders Methods

ZosJobFolders exposes the following methods:

Add Method

Adds a new folder. Query type can be "QN" (name), "QP" (prefix), "AP" (active prefix), "AU" (active userid), or "A" (all active). Search argument is job name, prefix, or userid. Default argument is logged-on userid.

```
Add (
    strFolderName,
    strQueryType,
    strQueryArg
    )
```

Returns

```
Object (
    IZosJobFolder
    )
```

Refresh Method

Refreshes collection.

```
Refresh()
```

Remove Method

Deletes a folder.

```
Remove (
    strFolderName
    )
```

ZosLibypeMapping

The ZosLibypeMapping object represents a single library type definition. This object can be obtained using the Item property of ZosLibypeMappings.

ZosLibypeMapping Properties

ZosLibypeMapping exposes the following properties:

Property	Type	R/W	Description
DataSetName	String	R	Data set name pattern.
LibType	String	R	Library type: "S" (Standard), "L" (Librarian), or "P" (Panvalet).

Examples of using ZosLibypeMapping.

Visual Basic or VBScript:

```
Dim objLibType
    Dim strDataSetName
    Dim strLibType
    strDataSetName = objLibType.DataSetName
    strLibType = objLibType.LibType
```

JScript:

```
var objLibType;
    var strDataSetName;
    var strLibType;
    strDataSetName = objLibType.DataSetName;
    strLibType = objLibType.LibType;
```

ZosLibypeMappings

The ZosLibypeMappings object is a collection of all library type definitions for a server. This object is obtained using the LibTypes property of the ZosServer object. Library types only need to be defined if you are using Librarian or Panvalet libraries.

ZosLibypeMappings Properties

ZosLibypeMappings exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosLibypeMapping)	R	Library type definition with specified index or data set name pattern.
Count	Integer (long)	R	Number of objects in collection.

ZosLibypeMappings Methods

ZosLibypeMappings exposes the following methods:

Add Method

Adds a new library type definition. Library type can be "S" (Standard), "L" (Librarian), or "P" (Panvalet).
Index indicates position for new item. Specify –1 to insert at end.

```
Add (
    nIndex,
    strDataSetName,
    strDataType
    )
```

Returns

```
Object (
    IZosLibypeMapping
    )
```

Refresh Method

Refreshes collection.

```
Refresh ()
```

Remove Method

Deletes a library type, specified by index or data set name pattern.

```
Remove (
    varIndex
    )
```

Move Method

Changes the order of library type definitions.

```
Move (
    nIndexTo,
    nIndexFrom
    )
```

Examples of using ZosLibypeMappings:

Visual Basic or VBScript:

```
Dim objLibTypes
    objLibTypes.Add -1, "**.PANVALET", "P"
    ObjLibTypes.Remove "**.LIBRARY"
    ObjLibTypes.Move 4,2
```

JScript:

```
var objLibTypes;
    objLibTypes.Add(-1, "**.PANVALET", "P");
    ObjLibTypes.Remove("**.LIBRARY");
    ObjLibTypes.Move(4,2);
```

ZosNameFilters

The ZosNameFilters object is a collection of all name filters for a folder. This object is obtained using the Filters property of the ZosDataSetFolder object or the Filters property of the ZosChangeManInstance object.

ZosNameFilters Properties

ZosNameFilters exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	String	R	Filter with specified index or data set name pattern.
Count	Integer (long)	R	Number of objects in collection.

ZosNameFilters Methods

ZosNameFilters exposes the following methods:

Refresh Method

Refreshes collection.

```
Refresh ()
```

Add Method

Adds a new file extension definition.

```
Add(
    strDataSetName
    )
```

Remove Method

Deletes a filter, specified by index or data set name pattern.

```
Remove (
    varIndex
    )
```

ZosNetwork

The ZosNetwork object represents the overall ZDD Network. ZosNetwork is always the starting point for ChangeMan ZDD Automation. It is created as follows:

Visual Basic or VBScript:

```
Dim objNetwork
    Set objNetwork = CreateObject("ZosCom.ZosNetwork")
```

JScript:

```
var objNetwork;
    objNetwork = new ActiveXObject("ZosCom.ZosNetwork");
```

The COM program ID used to access ChangeMan ZDD has changed to **ZosCom.ZosNetwork**. For backward compatibility, the old program ID of **ZosShell.Network** will still work. However, **ZosShell.ZosNetwork** has been deprecated, and you should begin using **ZosCom.ZosNetwork** instead.

ZosNetwork Properties

ZosNetwork exposes the following properties:

Property	Type	R/W	Description
Servers	Object (IZosServers)	R	Collection of all servers.
Server (strName)	Object (IZosServer)	R	Server with specified name.
CacheFolder	String	R/W	Name of folder used to store cached files.
CacheDays	Integer (long)	R/W	Number of days to keep cached files.

Property	Type	R/W	Description
SystemSettingsFolder	String	R/W	The system settings folder allows you to share system settings, server configuration, and security controls, between machines. This folder is optional. You can export the system settings from one machine into a shared folder. The system settings are automatically imported from this folder each time the system is booted.
UserSettingsFolder	String	R/W	The user settings folder is used to save user configuration such as folders and filters. You can share settings between machines by using a shared folder. This is a system-wide setting, but the path name can include environment variables, such as %USERNAME% to make the folder user-specific.
NotifyPort	Integer (long)	R/W	TCP/IP port number used to receive notification messages from the server. This port number should be unblocked on your local machine in the Windows firewall or other firewall software.
NotifyDelay	Integer (long)	R/W	Time delay, in seconds, before a message box is displayed. The time delay allows messages to accumulate so that several messages can be displayed in a single message box.
NotifyMessageBox	Boolean	R/W	Display message box for notify messages.
AsciiCodePage	Integer (long)	R/W	Windows code page number.

Property	Type	R/W	Description
TimeOut	Integer (long)	R/W	Network timeout interval, in minutes. Network operations are aborted if no response is received after this period of time. Must be in the range 3 - 30 minutes.
KeepAlive	Integer (long)	R/W	TCP/IP keep alive time interval, in minutes. TCP/IP keep alive packets are sent after this many minutes of inactivity to detect lost connections.

ZosNetwork Methods

ZosNetwork exposes the following methods:

StartNetwork Method

Start the ZDD Network service. This service provides all communication with z/OS servers.

```
StartNetwork()
```

StopNetwork Method

Stop the ZDD Network service. This service provides all communication with z/OS servers.

```
StopNetwork()
```

ExportUserSettings method

Exports user settings to a file in the specified folder. User settings are those settings that are userspecific, such as “DataSets”, “Jobs”, or “Unix” folder definitions, as well as filters, such as “Applications”, “Packages”, or “Releases” filters.

```
ExportUserSettings(
    strFolder
    )
```

ImportUserSettings Method

Imports user settings from a file in the specified folder. User settings are those settings that are userspecific, such as “DataSets”, “Jobs”, or “Unix” folder definitions, as well as filters, such as “Applications”, “Packages”, or “Releases” filters.

```
ImportUserSettings(
    strFolder
    )
```

Examples of getting or setting network properties:

Visual Basic or VBScript:

```
Dim objNetwork
    Dim objServers
    Dim objServer
    Set objServers = objNetwork.Servers
    Set objServer = objNetwork.Server("SYSA")
    objNetwork.CacheFolder = "C:\Temp"
    objNetwork.CacheDays = 3
    objNetwork.NotifyPort = 4000
    objNetwork.NotifyDelay = 60
    objNetwork.NotifyMessageBox = True
    objNetwork.AsciiCodePage = 1252
```

JScript:

```
var objNetwork;
    var objServers;
    var objServer;
    objServers = objNetwork.Servers;
    objServer = objNetwork.Server("SYSA")
    objNetwork.CacheFolder = "C:\\Temp";
    objNetwork.CacheDays = 3;
    objNetwork.NotifyPort = 4000;
    objNetwork.NotifyDelay = 60;
    objNetwork.NotifyMessageBox = true;
    objNetwork.AsciiCodePage = 1252
```

ZosPrefixMapping

The ZosPrefixMapping object represents a single prefix definition. This object can be obtained using the Item property of ZosPrefixMappings.

ZosPrefixMapping Properties

ZosPrefixMapping exposes the following properties:

Property	Type	R/W	Description
DataSetName	String	R	Data set name pattern.
Prefix	String	R	Data set name prefix.

ZosPrefixMappings

The ZosPrefixMappings object is a collection of all data set name prefix definitions for a folder. This object is obtained using the Prefixes property of the ZosDataSetFolder object.

ZosPrefixMappings Properties

ZosPrefixMappings exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosPrefixMapping)	R	Prefix definition with specified index or data set name pattern.
Count	Integer (long)	R	Number of objects in collection.

ZosPrefixMappings Methods

ZosPrefixMappings exposes the following methods:

Refresh Method

Refreshes collection.

```
Refresh()
```

Add Method

Adds a new prefix definition. Index indicates position for new item. Specify –1 to insert at end.

```
Add ( nIndex, strDataSetName,  strFileExtension  )
```

Returns

```
Object (
    IZosPrefixMapping
    )
```

Remove Method

Deletes a prefix definition, specified by index or data set name pattern.

```
Remove(
    varIndex
    )
```

Move Method

Changes the order of prefix definitions.

```
Move (
    nIndexTo,
    nIndexFrom
    )
```

ZosServer

The ZosServer object represents a single server. This object can be obtained using the Server property of ZosNetwork or the Item property of ZosServers.

ZosServer Properties

ZosServer exposes the following properties:

Property	Type	R/W	Description
Name	String	R	Name of the server.
Description	String	R/W	Server description (volume label).

Property	Type	R/W	Description
Address	String	R/W	I/P address or DNS name.
Port	Integer (long)	R/W	XCH port number of server.
Secure	Boolean	R/W	Enables TLS security for all ports on this server, including ChangeMan ports.
PasswordPhrase	Boolean	R/W	Indicates whether password phrases (long passwords) are allowed.
DisablePort	Boolean	R/W	Disables XCH port. If the port is disabled, the “DataSets”, “Jobs”, and “Unix” folders are hidden and unavailable.
MaxSessions	Integer (short)	R/W	Maximum number of concurrent sessions.
User	String	R	User ID of currently logged on user.
DataSetFolder (strPath)	Object (IZosDataSetFolder)	R	Data set folder with specified path name (relative to "DataSets" share name).

Property	Type	R/W	Description
JobFolder (strPath)	Object (IZosJobFolder)	R	Job folder with specified path name (relative to "Jobs" share name).
ChangeManInstance (strName)	Object (IZosChangeManInstance)	R	ChangeMan ZMF folder with specified name.
EbcdicCodePage	Integer (long)	R/W	Server code page number.
ChangeManInstances	Object (IZosChangeManInstances)	R	Collection of all ChangeMan ZMF folders for server.
LibTypes	Object (IZosLibypeMappings)	R	Collection of all library types for server.
FileExtensions	Object (IZosFileExtensionMappings)	R	Collection of file extensions for server.
DataSetFileFormats	Object (IZosFileFormatMappings)	R	Collection of file format mappings for data sets
UnixFileFormats	Object (IZosFileFormatMappings)	R	Collection of file format mappings for Unix files. This property is available on version 7.1+ servers only.
DataSetProfiles	Object (IZosDataSetProfiles)	R	Collection of all data set profiles for server.

Property	Type	R/W	Description
IsHidden	Boolean	R / W	Indicates that this server is hidden in the File Explorer and the ZDD user interface. This is a user-specific setting.

ZosServer Methods

ZosServer exposes the following methods:

Logon Method

Log on to server.

```
Logon (
       strUserId,
       strPassword,
       strNewPassword
       )
```

Logoff Method

Log off from server.

```
Logoff()
```

NotifyChange Method

Notifies file system driver that a data set or member has been created or deleted by an external process.

```
NotifyChange(
        strDataSetName,
        strMemberName
        )
```

SubmitJcl Method

Submit JCL to server. The bSuppressMessage parameter is optional. It is used to indicate whether message boxes should be suppressed. It is useful for running scripts in an automated tool where there is nobody present to press the OK button on a message box.

```
SubmitJcl (
        strFileName,
        bSuppressMessage,
        bNotify
        )
```

To suppress message boxes, specify:

`bSuppressMessage = true`

To display message boxes, specify:

`bSuppressMessage = false`

If this parameter is not specified, it defaults to false and message boxes will be displayed normally.

The bNotify parameter is optional. If bNotify is set to true, a notify job step will be added to the submitted job. The notify step will send you a message listing the return code for the job. |

Examples of using ZosServer:

Visual Basic or VBScript:

```
Dim objServer
        Dim strUser
        strUser = objServer.User
        objServer.Address = "199.90.90.9"
        objServer.Logon "USR001", "password"
        objserver.NotifyChange "USR001.NEW.DATA", "MEMBER1"
        objServer.SubmitJcl "C:\JCL\Print.jcl"
```

JScript:

```
var objServer;
        var strUser;
        strUser = objServer.User
        objServer.Address = "199.90.90.9";
        objServer.Logon("USR001", "password");
        objserver.NotifyChange("USR001.NEW.DATA", "MEMBER1");
        objServer.SubmitJcl("C:\JCL\Print.jcl");
```

ZosServers

The ZosServers object is a collection of all servers in the ZDD Network. This object is obtained using the Servers property of the ZosNetwork object.

ZosServers Properties

ZosServers exposes the following properties:

Property	Type	R/W	Description
Item (varIndex)	Object (IZosServer)	R	Server with specified name or index.
Count	Integer (long)	R	Number of objects in collection.

ZosServers Methods

ZosServers exposes the following methods:

Add Method

Adds a new server. If you specify port number 0, the port will be disabled. The “DataSets”, “Jobs”, and “Unix” folders are not available if the port is disabled. If you specify bSecure, TLS security will be enabled for all ports, including ChaangeMan ports

```
Add (
    strName,
    strAddress,
    nPort,
    strDescription,
    nMaxSessions,
    bPasswordPhrase,
    bSecure
    )
```

Refresh Method

Refreshes collection.

```
Refresh()
```

Returns

`Object (IZosServer)`

Remove Method

Deletes a server.

```
Remove (strName)
```

Examples of using ZosServers

Visual Basic or VBScript:

```
Dim objServers
    Dim objServer
    Dim nCount
    nCount = objServers.Count
    Set objServer = objServers.Item("SYSA")
    objServers.Add "Server1", "199.90.90.9", 5000,
    "Description1"
    objServers.Remove "Server1"
```

JScript:

```
var objServers;
    var objServer;
    var nCount;
    nCount = objServers.nCount;
    objServer = objServers. Item("SYSA")
    objServers.Add("Server1", "199.90.90.9", 5000,
    "Description1");
    objServers.Remove("Server1");
```

5. Examples

Example Scripts

Several sample scripts are included with ChangeMan ZDD that illustrate how to use the Automation interface to perform some common ChangeMan ZDD operations. They are in the Samples folder, under the folder where ChangeMan ZDD is installed on your PC; they are in both VBScript and JScript.

The following scripts are described in this chapter.

Script Description:

- Logging on to a Server
- Submitting JCL to a Server
- Configuring ChangeMan ZDD for a New User

This chapter also contains an example of how to use the Windows Task Scheduler to schedule your programs and scripts to run at specified times. See Using Windows Task Scheduler

Logging on to a Server

You can use the **Logon** method (function) from your program or script to log on to a z/OS server. An example of when you would use **Logon** is when your program or script is accessing a data set on a z/OS server.

The following scripts illustrate how to log on to a z/OS server.

VBScript Example

```
'**********************************************************************
'
' VBScript Example
'
' File Name: Logon.vbs
'
Description: Log on to server. If userid and password not specified,
'            user will be prompted.
'
Usage: Logon.vbs <server> [<userid>] [<password>] [<newpassword>]
'**********************************************************************
Dim strServerName
Dim strUserId
Dim strPassword
Dim strNewPassword

Dim objNetwork
Dim objServer

'----------------------------
' Get command line arguments
'----------------------------

If WScript.Arguments.Count < 1 Then
    WScript.Echo "Usage: Logon.vbs <server> [<userid>] [<password>] _
        [<newpassword>]"
    WScript.Quit(1)
End If

strServerName = WScript.Arguments(0)

If WScript.Arguments.Count > 1 Then
    strUserId = WScript.Arguments(1)
Else
    strUserId = ""
End If
```

```
If WScript.Arguments.Count > 2 Then
    strPassword = WScript.Arguments(2)
Else
    strPassword = ""
End If

If WScript.Arguments.Count > 3 Then
    strNewPassword = WScript.Arguments(3)
Else
    strNewPassword = ""
End If

'------------------
' Log on to server
'------------------

Set objNetwork = CreateObject("ZosCom.ZosNetwork")
Set objServer = objNetwork.Server(strServerName)

objServer.Logon strUserId, strPassword, strNewPassword

WScript.Echo "Logon: Server=" & strServerName, "UserId=" & strUserId
```

JScript Example

```
/*********************************************************************
* File Name: Logon.js
*
Description: Log on to server. If userid and password not specified,
*            user will be prompted.
*
Usage: Logon.js <server> [<userid>] [<password>] [<newpassword>]
*********************************************************************/
var strServerName;
var strUserId;
var strPassword;
var strNewPassword;
var objNetwork;
var objServer;
```

```
////////////////////////////
// Get command line arguments
////////////////////////////

if (WScript.Arguments.Count() < 1)
{
    WScript.Echo("Usage: Logon.js <server> [<userid>] [<password>] _
    [<newpassword>]");
    WScript.Quit(1);
}

strServerName = WScript.Arguments(0);

if (WScript.Arguments.Count() > 1)
{
    strUserId = WScript.Arguments(1);
} else
{
    strUserId = "";
}

if (WScript.Arguments.Count() > 2)
{
    strPassword = WScript.Arguments(2);
} else
{
    strPassword = "";
}

if (WScript.Arguments.Count() > 3)
{
    strNewPassword = WScript.Arguments(3);
} else
{
    strNewPassword = "";
}
////////////////////
// Log on to server
////////////////////

objNetwork = new ActiveXObject("ZosCom.ZosNetwork");
objServer = objNetwork.Server(strServerName);

objServer.Logon(strUserId, strPassword, strNewPassword);

WScript.Echo("Logon: Server=" + strServerName, "UserId=" + strUserId);
```

Submitting JCL to a Server

You can use the **SubmitJCL** method (function) from your program or script to submit JCL to a z/OS server. A situation where you might use **SubmitJCL** is when a program or script, that runs from Windows Task Scheduler, needs to submit a nightly batch job to a z/ OS server.

The following scripts illustrate how to submit JCL to a z/OS server.

VBScript Example

```
'**********************************************************************
' File Name: SubmitJcl.vbs
' Description: Submit a JCL file to a server.
' Usage: SubmitJcl.vbs <server> <file.name>
'**********************************************************************

Dim strServerName
Dim strFileName
Dim objNetwork
Dim objServer

Dim bSuppressMessage


'----------------------------
' Get command line arguments
'----------------------------

If WScript.Arguments.Count < 2 Then
    WScript.Echo "Usage: SubmitJcl.vbs <server> <file.name>"
    WScript.Quit(1)
 End If

strServerName = WScript.Arguments(0)
strFileName = WScript.Arguments(1)

'------------
' Submit JCL
'------------

Set objNetwork = CreateObject("ZosCom.ZosNetwork")
Set objServer = objNetwork.Server(strServerName)

bSuppressMessage = False

objServer.SubmitJcl strFileName, bSuppressMessage

WScript.Echo "Jcl submitted: Server=" & strServerName, _
    "FileName=" & strFileName
```

JScript Example

```
/*********************************************************************
*
* JScript Example
*
* File Name: SubmitJcl.js
*
* Description: Submit a JCL file to a server.
*
* Usage: SubmitJcl.js <server> <file.name>
*********************************************************************/
var strServerName;
var strFileName;

var objNetwork;
var objServer;

var bSuppressMessage;

//////////////////////////////
// Get command line arguments
//////////////////////////////

if (WScript.Arguments.Count() < 2)
{
    WScript.Echo("Usage: SubmitJcl.js <server> <file.name>");
    WScript.Quit(1);
}
strServerName =
WScript.Arguments(0);
strFileName = WScript.Arguments(1);

//////////////
// Submit JCL
//////////////

objNetwork = new ActiveXObject("ZosCom.ZosNetwork");
objServer = objNetwork.Server(strServerName);

bSuppressMessage = false;

objServer.SubmitJcl(strFileName bSuppressMessage);

WScript.Echo("Jcl submitted: Server=" + strServerName,
    "FileName=" + strFileName);
```

Configuring ChangeMan ZDD for a New User

To simplify the setup of ChangeMan ZDD for multiple desktops, you can write a script to automate many of the configuration tasks. Then, a new user can configure ChangeMan ZDD for their own desktop simply by executing the script.

The following scripts illustrate how the configuration tasks can be performed.

VBScript Example

```
'**********************************************************************
' VBScript Example
'
' File Name: NewConfig.vbs
'
' Description: Sample for creating a new configuration.
'
' Usage: NewConfig.vbs <userid>
'
' Copyright ©2003-2011, Serena Software. Licensed material. All rights reserved.
'**********************************************************************

Dim userID
Dim network
Dim servers
Dim server
Dim fileFormats
Dim libTypes
Dim fileExtensions
Dim dsProfiles
Dim folders
Dim folder
Dim subfolders
Dim subfolder
Dim filters
Dim members
Dim prefixes


'----------------------------
' Get command line arguments
'----------------------------

If WScript.Arguments.Count < 1 Then
    WScript.Echo "Usage: NewConfig.vbs <userid>"
    WScript.Quit(1)
End If

userID = WScript.Arguments(0)
```

```
'--------------------------
' Update network properties
'--------------------------

Set network = CreateObject("ZosCom.ZosNetwork")

network.CacheFolder = "C:\Temp"
network.CacheDays = 3
network.NotifyPort = 8000
network.NotifyMessageBox = True

'--------------------
' Add the new servers
'--------------------

Set servers = network.Servers

servers.Add "Server1", "172.20.20.1", 5000, 1140, "Description1"
servers.Add "Server2", "172.20.20.2", 5000, 1140, "Description2"
servers.Add "Server3", "172.20.20.3", 5000, 1140, "Description3"

'--------------------------------------
' Update the properties for each server
'--------------------------------------

For Each server In servers

    '--------------------------
    ' Add the data type entries
    '--------------------------

    Set fileFormats = server.DataSetFileFormats

    fileFormats.Add -1, "**.ASCII.TEXT",   "AT"
    fileFormats.Add -1, "**.ASCII.DATA",   "AD"
    fileFormats.Add -1, "**.UNICODE.TEXT", "UT"
    fileFormats.Add -1, "**.EBCDIC.TEXT",  "ET"
    fileFormats.Add -1, "**.EBCDIC.DATA",  "ED"
    fileFormats.Add -1, "**.BINARY",       "BT"

    '-----------------------------------------------------------
    ' Add the Unix file format entries
    ' This is supported on version 7.1+ servers only.
    ' The lines below should be removed for back level servers.
    '-----------------------------------------------------------

    Set fileFormats = server.UnixFileFormats

    fileFormats.Add -1, "*.TEXT",  "AT"
    fileFormats.Add -1, "*.UTEXT", "UT"
    fileFormats.Add -1, "*.BIN",   "B"

    '------------------------------
    ' Add the library type entries
    '------------------------------

    Set libTypes = server.LibTypes

    libTypes.Add -1, "**.LIBRARY",  "L"
    libTypes.Add -1, "**.PANVALET", "P"
```

```
'--------------------------------
' Add the file extension entries
'--------------------------------

Set fileExtensions = server.FileExtensions

fileExtensions.Add -1, "**.CNTL",  "jcl"
fileExtensions.Add -1, "**.COBOL", "cbl"
fileExtensions.Add -1, "**.LIST",  "txt"
fileExtensions.Add -1, "**.WORD",  "doc"
fileExtensions.Add -1, "**.EXCEL", "xls"

'-----------------------------------
' Add the profiles for new data sets
'-----------------------------------

Set dsProfiles = server.DataSetProfiles

dsProfiles.Add -1, "**.DATA", "SEQ", "FB", 80, 0, "DATACLS1",
"STORCLS1", "MGMTCLS1", "TRK", 2, 1, 5, "SYSDA", "VOL001"
dsProfiles.Add -1, "**.TEMP", "SEQ", "FB", 80, 0, "DATACLS2",
"STORCLS2", "MGMTCLS2", "CYL", 2, 1, 5, "SYSDA", "VOL002"
dsProfiles.Add -1, "**.LIST", "SEQ", "VB", 80, 0, "", "",
"", "BLK", 500, 50, 5, "SYSDA", ""

'----------------------
' Add data set folders
'----------------------

Set folder = server.DataSetFolder
Set subfolders = folder.Subfolders

'-----------------------------------------------
' "My DataSets" folder for all user's data sets
'-----------------------------------------------

Set subfolder = subfolders.Add("My DataSets")

Set filters = subfolder.Filters
Set prefixes = subfolder.Prefixes

filters.Add userID & ".**"

prefixes.Add -1, "**", userID

'-----------------------------------------------
' "My Source" folder for user's source libraries
'-----------------------------------------------

Set subfolder = subfolders.Add("My Source")

Set filters = subfolder.Filters
Set members = subfolder.MemberFilters
Set prefixes = subfolder.Prefixes

filters.Add userID & ".**.COBOL"
filters.Add userID & ".**.ASM"

members.Add "ABC*"
members.Add "X*"
```

```
    prefixes.Add -1, "**", userID

    '-----------------
    ' Add job folders
    '-----------------

    Set folder = server.JobFolder
    Set subfolders = folder.Subfolders

    '---------------------------------------
    ' "My Jobs" folder for jobs owned by user
    '---------------------------------------

    subfolders.Add "My Jobs", "QU", userID

    '-----------------------------------------------------
    ' "ChangeMan" folder for job names prefixed with "CMN"
    '-----------------------------------------------------

    subfolders.Add "ChangeMan", "QN", "CMN*"

    '-------------------------------------
    ' "Active" folder for all active jobs
    '-------------------------------------

    subfolders.Add "Active", "A"

    '-----------------------
    ' Add ChangeMan folders
    '-----------------------

    Set folders = server.ChangeManInstances

    folders.Add "ChangeMan-Prod", 3000, "Production ChangeMan"
    folders.Add "ChangeMan-Test", 3001, "Test ChangeMan"

    For Each folder In folders

        '-------------------------------------
        ' Add the ChangeMan file format entries
        '-------------------------------------

        Set fileFormats = folder.FileFormats

        fileFormats.Add -1, "SRC", "AT"
        fileFormats.Add -1, "DOC", "UT"
        fileFormats.Add -1, "BIN", "BT"

    Next

Next
```

JScript Example

```
/*****************************************************************
*
* File Name: NewConfig.js
**
Description: Sample for creating a new configuration.
**
Usage:       NewConfig.js <userid>
**
Copyright ©2003-2011, Serena Software. Licensed material. All rights
reserved.
*****************************************************************/

var userID;

var network;
var servers;
var server;
var fileFormats;
var libTypes;
var fileExtensions;
var dsProfiles;
var folders;
var folder;
var subfolders;
var subfolder;
var filters;
var members;
var prefixes;

var enumerator;

////////////////////////////
// Get command line arguments
////////////////////////////

if (WScript.Arguments.Count() < 1)
{
    WScript.Echo("Usage: NewConfig.js <userid>");
    WScript.Quit(1);
}
```

```
userID = WScript.Arguments(0);

////////////////////////////
// Update network properties
////////////////////////////

network = new ActiveXObject("ZosCom.ZosNetwork");

network.CacheFolder = "C:\\Temp";
network.CacheDays = 3;
network.NotifyPort = 8000;
network.NotifyMessageBox = true;

//////////////////////
// Add the new servers
//////////////////////

servers = network.Servers;

servers.Add("Server1", "172.20.20.1", 5000, 1140, "Description1");
servers.Add("Server2", "172.20.20.2", 5000, 1140, "Description2");
servers.Add("Server3", "172.20.20.3", 5000, 1140, "Description3");

////////////////////////////////////////
// Update the properties for each server
////////////////////////////////////////

serverEnum = new Enumerator(servers);

for (; !serverEnum.atEnd(); serverEnum.moveNext())
{
    server = serverEnum.item();

    ////////////////////////////////////
    // Add the data set file format entries
    ////////////////////////////////////

    fileFormats = server.DataSetFileFormats;
    fileFormats.Add(-1, "**.ASCII.TEXT",   "AT");
    fileFormats.Add(-1, "**.ASCII.DATA",   "AD");
    fileFormats.Add(-1, "**.UNICODE.TEXT", "UT");
    fileFormats.Add(-1, "**.EBCDIC.TEXT",  "ET");
    fileFormats.Add(-1, "**.EBCDIC.DATA",  "ED");
    fileFormats.Add(-1, "**.BINARY",       "BT");

    ////////////////////////////////////////////////////////////
    // Add the Unix file format entries
    // This is supported on version 7.1+ servers only.
    // The lines below should be removed for back level servers.
    ////////////////////////////////////////////////////////////

    fileFormats = server.UnixFileFormats;

    fileFormats.Add(-1, "*.TEXT",  "AT");
    fileFormats.Add(-1, "*.UTEXT", "UT");
    fileFormats.Add(-1, "*.BIN",   "B" );

    ////////////////////////////////
    // Add the library type entries
    ////////////////////////////////
```

```
libTypes = server.LibTypes;

libTypes.Add(-1, "**.LIBRARY",  "L");
libTypes.Add(-1, "**.PANVALET", "P");

////////////////////////////////
// Add the file extension entries
////////////////////////////////

fileExtensions = server.FileExtensions;

fileExtensions.Add(-1, "**.CNTL",  "jcl");
fileExtensions.Add(-1, "**.COBOL", "cbl");
fileExtensions.Add(-1, "**.LIST",  "txt");
fileExtensions.Add(-1, "**.WORD",  "doc");
fileExtensions.Add(-1, "**.EXCEL", "xls");

////////////////////////////////////
// Add the profiles for new data sets
////////////////////////////////////

dsProfiles = server.DataSetProfiles;

dsProfiles.Add(-1, "**.DATA", "SEQ", "FB", 80, 0, "DATACLS1",
"STORCLS1", "MGMTCLS1", "TRK", 2, 1, 5, "SYSDA", "VOL001");
dsProfiles.Add(-1, "**.TEMP", "SEQ", "FB", 80, 0, "DATACLS2",
"STORCLS2", "MGMTCLS2", "CYL", 2, 1, 5, "SYSDA", "VOL002");
dsProfiles.Add(-1, "**.LIST", "SEQ", "VB", 80, 0, "", "",
"", "BLK", 500, 50, 5, "SYSDA", "");

//////////////////////
// Add data set folders
//////////////////////

folder = server.DataSetFolder;
subfolders = folder.Subfolders;

////////////////////////////////////////////////
// "My DataSets" folder for all user's data sets
////////////////////////////////////////////////

subfolder = subfolders.Add("My DataSets");

filters = subfolder.Filters;
prefixes = subfolder.Prefixes;

filters.Add(userID + ".**");

prefixes.Add(-1, "**", userID);

////////////////////////////////////////////////
// "My Source" folder for user's source libraries
////////////////////////////////////////////////

subfolder = subfolders.Add("Source");

filters = subfolder.Filters;
members = subfolder.MemberFilters;
prefixes = subfolder.Prefixes;
```

```
    filters.Add(userID + ".**.COBOL");
    filters.Add(userID + ".**.ASM");
    members.Add("ABC*");
    members.Add("X*");

    prefixes.Add(-1, "**", userID);

    ////////////////////
    // Add job folders
    ////////////////////

    folder = server.JobFolder;
    subfolders = folder.Subfolders;

    ////////////////////////////////////////
    // "My Jobs" folder for jobs owned by user
    ////////////////////////////////////////

    subfolder = subfolders.Add("My Jobs", "QU", userID);

    ////////////////////////////////////////////////////////
    // "ChangeMan" folder for job names prefixed with "CMN"
    ////////////////////////////////////////////////////////

    subfolder = subfolders.Add("ChangeMan", "QN", "CMN*");

    ////////////////////////////////////
    // "Active" folder for all active jobs
    ////////////////////////////////////

    subfolder = subfolders.Add("Active", "A");

    //////////////////////////
    // Add ChangeMan folders
    //////////////////////////

    folders = server.ChangeManInstances;

    folders.Add("ChangeMan-Prod", 3000, "Production ChangeMan");
    folders.Add("ChangeMan-Test", 3001, "Test ChangeMan");

    folderEnum = new Enumerator(folders);

    for (; !folderEnum.atEnd(); folderEnum.moveNext())
    {
        folder = folderEnum.item();

        ////////////////////////////////////////
        // Add the ChangeMan file format entries
        ////////////////////////////////////////

        fileFormats = folder.FileFormats;

        fileFormats.Add(-1, "SRC", "AT");
        fileFormats.Add(-1, "DOC", "UT");
        fileFormats.Add(-1, "BIN", "B");
    }
}
```

Using Windows Task Scheduler

The Windows Task Scheduler allows you to schedule programs to run at specified times. For example, you can schedule nightly job cycles to run automatically.

The following example shows how to logon to a z/OS server and submit a job. This .bat file contains commands to execute scripts that use the **Logon** and **SubmitJCL** methods. For examples of these scripts, see Logging on to a Server and Submitting JCL to a Server.

```
Rem This is a batch file that logs on to the z/OS Host.

Rem After logging on, a JCL member on the Host is submitted.

C:\\MyJobs\\WSCRIPT Logon.vbs HOSTNAME USERID PASSWORD

C:\\MyJobs\\WSCRIPT SubmitJCL.vbs M:\\USER999.CNTL.JCL\\MYJOB

Say 'Your Job was Submitted'
Pause
```

You can schedule this .bat file to run automatically using the Windows Task Scheduler. To access the Windows Task Scheduler:

1. Choose **Programs>Accessories>System Tools>Scheduled Tasks** from the Windows **Start** Menu. The **Scheduled Tasks** dialog box appears.
2. Click **Add Scheduled Task** and a wizard will guide you through the process.

6. Legal Notice

For information about legal notices, trademarks, disclaimers, warranties, export and other use restrictions, U.S. Government rights, patent policy, and FIPS compliance, see https://www.microfocus.com/en-us/legal.

The only warranties for products and services of Micro Focus and its affiliates and licensors ("Micro Focus") are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Third-Party Notices

Additional third-party notices, including copyrights and software license texts, can be found in a 'thirdpartynotices' file in the root directory of the software.

Specific notices

In accordance with the GNU General Public License version 2 with Classpath Exception, you are entitled to the complete OpenJDK source code that went into the JRE used by this product which includes the source code for 3 subclasses of that standard OpenJDK; MultipleGradientPaint, MultipleGradientPaintContext and TypeResolver. Please contact product support if you wish to obtain the source code. This source code will be available for 3 years from the general availability date for version 17.0 SP1.