



Microsoft® Deployment Toolkit 2013

Toolkit Reference

Published: September 2013

For the latest information and to leave feedback, please visit Microsoft Connect at <http://connect.microsoft.com>.



The information in this document and any document referenced herein is provided for informational purposes only, is provided AS IS AND WITH ALL FAULTS and cannot be understood as substituting for customized service and information that might be developed by Microsoft Corporation for a particular user based upon that user's particular environment. RELIANCE UPON THIS DOCUMENT AND ANY DOCUMENT REFERENCED HEREIN IS AT THE USER'S OWN RISK.

© 2013 Microsoft Corporation. All rights reserved.

If the user of this work is using the work SOLELY FOR NON-COMMERCIAL PURPOSES INTERNALLY WITHIN A COMPANY OR ORGANIZATION, then this work is licensed under the Creative Commons Attribution-NonCommercial License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/2.5> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.

MICROSOFT CORPORATION PROVIDES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION CONTAINED IN THIS DOCUMENT AND ANY DOCUMENT REFERENCED HEREIN. Microsoft Corporation provides no warranty and makes no representation that the information provided in this document or any document referenced herein is suitable or appropriate for any situation, and Microsoft Corporation cannot be held liable for any claim or damage of any kind that users of this document or any document referenced herein may suffer. Your retention of and/or use of this document and/or any document referenced herein constitutes your acceptance of these terms and conditions. If you do not accept these terms and conditions, Microsoft Corporation does not provide you with any right to use any part of this document or any document referenced herein.

Complying with the applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights or other intellectual property rights covering subject matter within this document. Except as provided in any separate written license agreement from Microsoft, the furnishing of this document does not give you, the user, any license to these patents, trademarks, copyrights or other intellectual property.

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred.

Microsoft, Active Directory, BitLocker, Internet Explorer, SQL Server, Visual Basic, Visual Studio, Windows, Windows Live, Windows Media, and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Contents

Introduction to Toolkit Reference	1
Task Sequence Steps.....	2
Common Properties and Options for Task Sequence Step Types	2
Common Options	3
Specific Properties and Settings for Task Sequence Step Types.....	5
Apply Network Settings	5
Authorize DHCP	6
Capture Network Settings.....	7
Configure ADDS.....	7
Configure DHCP.....	9
Configure DNS	11
Enable BitLocker	13
Execute Runbook	13
Format and Partition Disk.....	16
Gather	17
Inject Drivers	18
Install Application	18
Install Operating System	19
Install Roles and Features.....	20
Install Language Packs Offline	20
Install Language Packs Online	21
Install Updates Offline	21
Recover from Domain Join Failure	22
Restart computer.....	22
Run Command Line	23
Run PowerShell Script.....	23
Set Task Sequence Variable	24
Uninstall Roles and Features	25
Validate	25
Out-of-Box Task Sequence Steps	26
Apply Network Settings	26
Apply Patches	27
Apply Windows PE	28
Backup	29
Capture Groups.....	30
Capture User State.....	30
Check BIOS.....	31
Configure.....	32
Copy Scripts.....	33
Copy Sysprep Files	34
Create BitLocker Partition.....	34
Create WIM.....	35
Disable BDE Protectors.....	36
Enable BitLocker	37
Enable OEM Disk Configuration.....	38

End Phase.....	38
Execute Sysprep	39
Force Diskpart Action.....	40
Format and Partition Disk.....	41
Gather local only	42
Generate Application Migration File	43
Inject Drivers	43
Install Applications.....	44
Install Operating System	45
Next Phase.....	46
Post-Apply Cleanup.....	47
Recover from Domain.....	48
Restart computer.....	48
Restore Groups	49
Restore User State.....	50
Set Image Build	51
Set Image Flags	51
Tattoo	52
Validate	53
Windows Update (Pre-Application Installation).....	54
Windows Update (Post-Application Installation).....	55
Wipe Disk	55
Properties	57
Property Definition.....	57
_SMSTSOrgName	57
_ADDSLogPath	58
_ADDSPassword.....	58
_ADDSUserDomain	59
_ADDSUserName	60
Administrators	60
AdminPassword	61
Applications.....	61
ApplicationSuccessCodes	62
ApplyGPOPack	63
Architecture	63
AreaCode	64
AssetTag	64
AutoConfigDNS	65
BackupDir.....	66
BackupDrive.....	67
BackupFile.....	67
BackupShare.....	68
BDEAllowAlphaNumericPin.....	69
BDEDriveLetter	69
BDEDriveSize.....	70
BDEInstall.....	71
BDEInstallSuppress	72

BDEKeyLocation	72
BDEPin.....	73
BDERecoveryKey.....	73
BDEWaitForEncryption.....	74
BitsPerPel.....	75
BuildID.....	76
CapableArchitecture.....	76
CaptureGroups.....	77
ChildName.....	78
ComputerBackupLocation	78
ComputerName	79
ConfigFileName.....	80
ConfigFilePackage	80
ConfirmGC	80
CountryCode	81
CriticalReplicationOnly	82
CustomDriverSelectionProfile.....	82
CustomPackageSelectionProfile	83
CustomWizardSelectionProfile	83
Database	84
DatabasePath.....	84
DBID	85
DBPwd	86
Debug.....	87
DefaultGateway.....	88
DeployDrive.....	89
DeploymentMethod	89
DeploymentType	90
DeployRoot.....	91
DestinationDisk	92
DestinationLogicalDrive.....	92
DestinationPartition	93
DHCPScopes	94
DHCPScopesxDescription.....	94
DHCPScopesxEndIP	95
DHCPScopesxExcludeEndIP	95
DHCPScopesxExcludeStartIP	96
DHCPScopesxIP	96
DHCPScopesxName	97
DHCPScopesxOptionDNSDomainName.....	97
DHCPScopesxOptionDNSServer	98
DHCPScopesxOptionLease	98
DHCPScopesxOptionNBTNodeType	99
DHCPScopesxOptionPXEClient.....	100
DHCPScopesxOptionRouter	100
DHCPScopesxOptionWINSServer	101
DHCPScopesxStartIP.....	101

DHCPScopesxSubnetMask.....	102
DHCPServerOptionDNSDomainName.....	102
DHCPServerOptionDNSServer	103
DHCPServerOptionNBTNodeType	103
DHCPServerOptionPXEClient.....	104
DHCPServerOptionRouter	104
DHCPServerOptionWINSServer	105
Dialing	105
DisableTaskMgr	106
DNSServerOptionBINDSecondaries	107
DNSServerOptionDisableRecursion.....	108
DNSServerOptionEnableNetmaskOrdering.....	108
DNSServerOptionEnableRoundRobin.....	109
DNSServerOptionEnableSecureCache.....	109
DNSServerOptionFailOnLoad	110
DNSServerOptionNameCheckFlag	111
DNSZones.....	111
DNSZonesxDirectoryPartition.....	112
DNSZonesxFileName.....	113
DNSZonesxMasterIP.....	113
DNSZonesxName	114
DNSZonesxScavenge	114
DNSZonesxType	115
DNSZonesxUpdate	116
DoCapture	116
DomainAdmin.....	117
DomainAdminDomain	118
DomainAdminPassword	118
DomainLevel	119
DomainNetBiosName.....	120
DomainOUs.....	120
DoNotCreateExtraPartition	121
DoNotFormatAndPartition	122
DriverGroup.....	122
DriverInjectionMode	123
DriverPaths.....	124
DriverSelectionProfile	125
EventService	125
EventShare.....	126
FinishAction.....	126
ForceApplyFallback.....	127
ForestLevel.....	128
FullName	129
GPOPackPath	130
Groups.....	131
HideShell.....	131
OSHome_Page	132

HostName	132
ImagePackageID	133
InputLocale	133
InstallPackageID	134
Instance	135
IPAddress	135
IsDesktop	136
IsHypervisorRunning	136
IsLaptop	137
IsServer	138
IsServerCoreOS	138
IsServerOS	139
IsUEFI	139
IsVMM	140
JoinDomain	140
JoinWorkgroup	141
KeyboardLocale	142
KeyboardLocalePE	143
LanguagePacks	143
LoadStateArgs	144
Location	145
LongDistanceAccess	147
MACAddress	147
MachineObjectOU	148
Make	149
MandatoryApplications	149
Memory	150
Model	151
NetLib	151
NewDomain	152
NewDomainDNSName	153
Order	153
OrgName	154
OSArchitecture	155
OSCurrentBuild	156
OSCurrentVersion	156
OSDAdapterxDescription	156
OSDAdapterxDNSDomain	157
OSDAdapterxDNSServerList	158
OSDAdapterxDNSSuffix	158
OSDAdapterxEnableDHCP	159
OSDAdapterxEnableDNSRegistration	159
OSDAdapterxEnableFullDNSRegistration	160
OSDAdapterxEnableLMHosts	160
OSDAdapterxEnableIPProtocolFiltering	161
OSDAdapterxEnableTCPFiltering	162
OSDAdapterxEnableTCPIPFiltering	162

OSDAdapterxEnableWINS.....	163
OSDAdapterxGatewayCostMetric.....	163
OSDAdapterxGateways	164
OSDAdapterxIPAddressList	164
OSDAdapterxIPProtocolFilterList	165
OSDAdapterxMacAddress	166
OSDAdapterxName.....	166
OSDAdapterxSubnetMask	167
OSDAdapterxTCPFilterPortList.....	167
OSDAdapterxTCPIPNetBiosOptions.....	168
OSDAdapterxUDPFILTERPortList.....	168
OSDAdapterxWINSServerList.....	169
OSDAdapterCount	169
OSDAnswerFilePath	170
OSDBitLockerCreateRecoveryPassword	171
OSDBitLockerMode.....	171
OSDBitLockerRecoveryPassword.....	172
OSDBitLockerStartupKey	173
OSDBitLockerStartupKeyDrive.....	174
OSDBitLockerTargetDrive	174
OSDBitLockerWaitForEncryption	175
OSDComputerName	176
OSDDiskAlign.....	176
OSDDiskIndex.....	177
OSDDiskOffset.....	178
OSDDiskPartBiosCompatibilityMode.....	178
OSDImageCreator.....	179
OSDImageIndex.....	179
OSDImagePackageID	180
OSDInstallEditionIndex.....	180
OSDInstallType	181
OSDisk	181
OSDPartitions.....	181
OSDPartitionsxBootable.....	182
OSDPartitionsxFileSystem	183
OSDPartitionsxQuickFormat	183
OSDPartitionsxSize	184
OSDPartitionsxSizeUnits.....	184
OSDPartitionsxType	185
OSDPartitionsxVolumeLetterVariable	186
OSDPartitionsxVolumeName	186
OSDPreserveDriveLetter.....	187
OSDStateStorePath	187
OSDTARGETSystemDrive.....	188
OSDTARGETSystemRoot.....	189
OSFeatures	189
OSInstall.....	190

OSRoles	191
OSRoleServices	191
OSSKU	192
OSVersion	192
OSVersionNumber	193
OverrideProductKey	193
PackageGroup	194
Packages	195
PackageSelectionProfile	195
Parameters	196
ParameterCondition	197
ParentDomainDNSName	198
Password	198
Phase	199
Port	200
PowerUsers	200
PrepareWinRE	201
Priority	202
ProcessorSpeed	202
Product	203
ProductKey	204
Properties	204
ReplicaDomainDNSName	205
ReplicaOrNewDomain	205
ReplicationSourceDC	206
ResourceDrive	207
ResourceRoot	207
Role	208
SafeModeAdminPassword	209
ScanStateArgs	210
SerialNumber	211
SiteName	211
SkipAdminAccounts	212
SkipAdminPassword	213
SkipApplications	214
SkipBDDWelcome	215
SkipBitLocker	216
SkipBuild	217
SkipCapture	218
SkipComputerBackup	218
SkipComputerName	219
SkipDomainMembership	220
SkipFinalSummary	221
SkipGroupSubFolders	222
SkipLocaleSelection	223
SkipPackageDisplay	223
SkipProductKey	224

SkipRearm.....	225
SkipRoles	226
SkipSummary	227
SkipTaskSequence	228
SkipTimeZone	229
SkipUserData	230
SkipWizard	231
SLShare	232
SLShareDynamicLogging.....	232
SMSTSAssignUserMode.....	233
SMSTSRunCommandLineUserName	234
SMSTSRunCommandLineUserPassword.....	234
SMSTSUdaUsers.....	235
SQLServer.....	235
SQLShare.....	236
StatePath.....	237
StoredProcedure	238
SupportsHyperVRole.....	239
SysVolPath.....	242
Table	243
TaskSequenceID.....	244
TaskSequenceName.....	244
TaskSequenceVersion	245
TimeZoneName.....	245
TPMOwnerPassword	246
UDDir.....	247
UDProfiles	248
UDShare.....	249
UILanguage.....	249
UserDataLocation.....	250
UserDomain	251
UserID	252
UserLocale	252
UserPassword	253
USMTConfigFile	254
USMTLocal.....	255
USMTMigFiles.....	256
USMTOfflineMigration	257
UUID.....	258
ValidateDomainCredentialsUNC	258
VHDCreateDiffVHD	259
VHDCreateFileName.....	261
VHDCreateSizeMax	262
VHDCreateSource.....	263
VHDCreateType	264
VHDDisks.....	265
VHDInputVariable.....	266

VHDOutputVariable	267
VHDTargetDisk	268
VMHost.....	269
VMName	270
VMPlatform.....	271
VRefresh	271
VSSMaxSize	272
WDSServer	273
WindowsSource	273
WipeDisk	274
WizardSelectionProfile	275
WSUSServer	275
WUMU_ExcludeKB	276
WUMU_ExcludeID	277
XResolution.....	277
YResolution.....	278
Providing Properties for Skipped Deployment Wizard Pages.....	279
Scripts.....	282
BDD_Autorun.wsf.....	282
BDD_Welcome_ENU.xml.....	283
Credentials_ENU.xml	284
Credentials_scripts.vbs	285
DeployWiz_Definition_ENU.xml	285
DeployWiz_Initialization.vbs	288
DeployWiz_Validation.vbs	290
LiteTouch.vbs	292
LiteTouch.wsf	292
LTIAppl.wsf	296
LTICleanup.wsf.....	297
LTICopyScripts.wsf.....	299
LTIGetFolder.wsf	299
LTIOEM.wsf.....	300
LTISuspend.wsf.....	302
LTISysprep.wsf.....	303
NICSettings_Definition_ENU.xml	304
Summary_Definition_ENU.xml	305
Summary_scripts.vbs	306
Wizard.hta	306
WizUtility.vbs	308
ZTIAppllications.wsf	309
ZTIAppXmlGen.wsf	310
ZTIAuthorizeDHCP.wsf	311
ZTIBackup.wsf.....	313
ZTIBCDUtility.vbs	314
ZTIBde.wsf	315
ZTIBIOSCheck.wsf	317
ZTICoalesce.wsf.....	318

ZTIConfigFile.vbs	320
ZTIConfigure.wsf	320
ZTIConfigureADDSS.wsf	322
ZTIConfigureDHCP.wsf	323
ZTIConfigureDNS.wsf	325
ZTIConnect.wsf	327
ZTICopyLogs.wsf	328
ZTIDataAccess.vbs	328
ZTIDisableBDEProtectors.wsf	329
ZTIDiskpart.wsf	330
ZTIDiskUtility.vbs	332
ZTIDomainJoin.wsf	333
ZTIDrivers.wsf	335
ZTIEncryptRunbook.wsf	336
ZTIGather.wsf	338
ZTIGroups.wsf	339
ZTILangPacksOnline.wsf	341
ZTIModifyVol.wsf	342
ZTIMoveStateStore.wsf	343
ZTINextPhase.wsf	344
ZTINICConfig.wsf	345
ZTINICUtility.vbs	346
ZTIOSRole.wsf	347
ZTIPatches.wsf	348
ZTIPowerShell.wsf	350
ZTIPreq.vbs	351
ZTISCCM.wsf	351
ZTISetVariable.wsf	353
ZTITatoo.wsf	354
ZTIUserState.wsf	357
ZTIUtility.vbs	359
ZTIVValidate.wsf	361
ZTIVHDCreate.wsf	363
ZTIWindowsUpdate.wsf	365
ZTIWipeDisk.wsf	369
Support Files	370
ApplicationGroups.xml	370
Applications.xml	370
BootStrap.ini	370
CustomSettings.ini	371
Deploy.xml	371
DriverGroups.xml	371
Drivers.xml	371
LinkedDeploymentShares.xml	371
ListOfLanguages.xml	371
MediaGroups.xml	372
Medias.xml	372

OperatingSystemGroups.xml.....	372
OperatingSystems.xml	372
PackageGroups.xml	372
Packages.xml	373
SelectionProfileGroups.xml	373
SelectionProfiles.xml	373
ServerManager.xml	373
Settings.xml	373
TaskSequenceGroups.xml	373
TaskSequences.xml	374
TS.xml	374
Wimscript.ini	374
ZTIBIOSCheck.xml.....	374
ZTIConfigure.xml.....	375
ZTIGather.xml.....	375
ZTIUserState_config.xml	375
ZTITatoo.mof	375
Utilities	377
BCDBoot.exe.....	377
BDDRun.exe.....	378
Bootsect.exe.....	378
Compact.exe	379
Diskpart.exe.....	380
Expand.exe.....	380
ImageX.exe	381
Microsoft.BDD.PnpEnum.exe	381
Mofcomp.exe	382
Netsh.exe	382
Reg.exe	382
Regsvr32.exe	383
Wpeutil.exe.....	383
MDT Windows PowerShell Cmdlets	384
Add-MDTPersistentDrive	386
Disable-MDTMonitorService.....	388
Enable-MDTMonitorService.....	389
Get-MDTDeploymentShareStatistics.....	391
Get-MDTMonitorData	393
Get-MDTOperatingSystemCatalog.....	396
Get-MDTPersistentDrive	397
Import-MDTApplication.....	398
Import-MDTDriver	403
Import-MDTOperatingSystem.....	406
Import-MDTPackage	411
Import-MDTTaskSequence.....	413
New-MDTDatabase	421
Remove-MDTMonitorData.....	425
Remove-MDTPersistentDrive	428

Restore-MDTPersistentDrive.....	430
Set-MDTMonitorData.....	432
Test-MDTDeploymentShare.....	435
Test-MDTMonitorData	435
Update-MDTDatabaseSchema.....	437
Update-MDTDeploymentShare	440
Update-MDTLinkedDS	443
Update-MDTMedia	445
Tables and Views in the MDT DB	447
Tables in the MDT DB	447
Views in the MDT DB	449
Windows 7 Feature Dependency Reference	452
UDI Reference.....	454
UDI Concepts	454
Display Name	454
Flow.....	454
Page Library	454
Page Name	455
Prestaged Media Deployments	455
Stage Group.....	455
Stage.....	455
Task	456
UDI Task Sequence	456
UDI Wizard	457
UDI Wizard Application Configuration File.....	458
UDI Wizard Configuration File	458
UDI Wizard Designer.....	458
Validator	459
Wizard Page.....	459
Wizard Page Editor	459
OSDResults Reference	460
User-Centric App Installer Reference.....	462
UDI Stage Reference	466
REFRESH Stage.....	470
REPLACE and REPLACE.WinPE Stages.....	472
UDI Task Reference	472
UDI Task Overview	473
UDI Task Configuration Settings	473
Built-in UDI Tasks.....	474
UDI Validator Reference.....	478
UDI Validator Overview	479
Built-in UDI Validators	479
UDI Wizard Page Reference	480
UDI Wizard Page Overview	480
Built-in UDI Wizard Pages	481
UDI Build Your Own Page Toolbox Control Reference	505
Checkbox Control.....	507

Combobox Control	509
Line Control.....	510
Label Control.....	511
Radio Control	512
Bitmap Control.....	513
Textbox Control	514
UDI Task Sequence Variables.....	516
OSDAddAdmin.....	516
OSDApplicationList	517
OSDArchitecture	517
OSDBitlockerStatus.....	517
OSDDiskPart.....	518
OSDDomainName.....	518
OSDDomainOUName	518
OSDImageIndex.....	519
OSDImageName	519
OSDJoinAccount	520
OSDJoinPassword	520
OSDLocalAdminPassword	520
OSDNetworkJoinType	520
OSDSetupWizCancelled	521
OSDTargetDrive.....	521
OSDWinPEWinDir.....	522
OSDWorkgroupName.....	522
OSDResults.exe.config File Element Values.....	522
backgroundOpacity	522
backgroundWallpaper	523
completedText.....	523
headerImagePath.....	523
timeoutMinutes	524
welcomeText	524

Introduction to Toolkit Reference

This reference is part of Microsoft® Deployment Toolkit (MDT) 2013 and provides configuration settings that you can use in the deployment process. Review the MDT 2013 documents *Microsoft Deployment Toolkit Samples Guide* and *Using the Microsoft Deployment Toolkit* for help in customizing configuration settings for the deployment environment.

Note In this document, *Windows* applies to the Windows 8.1, Windows 8, Windows 7, Windows Server® 2012 R2, Windows Server 2012, and Windows Server 2008 R2 operating systems unless otherwise noted. MDT does not support ARM processor-based versions of Windows. Similarly, *MDT* refers to MDT 2013 unless otherwise stated.

Task Sequence Steps

Task sequences are created by the Task Sequence Editor and consist of a combined series of steps that are designed to complete an action. Task sequences can operate across a computer restart and can be configured to automate tasks on a computer without requiring user intervention. In addition, you can add task sequence steps to a task sequence group, which helps keep similar task sequence steps together for better organization and error control.

Each task sequence step performs a specific task, such as validating that the target computer is capable of receiving the deployment image, storing user data in a safe location, deploying an image to a target computer, and restoring saved user data. These task sequence steps accomplish their tasks by using utilities and scripts provided with MDT or by the deployment team. Use this reference to help determine the correct task sequence groups and task sequence steps to configure the deployment process and the valid properties and options to use.

The following information is provided for each task sequence group and step:

- **Name.** The name of the task sequence group or step
- **Description.** A description of the purpose of the task sequence group or step and any pertinent information regarding its customization
- **Properties.** Indicates the valid configuration properties that you can specify for the task sequence group or step that define how the task is performed
- **Options.** Indicates the valid configuration options that you can specify for the task sequence group or step that define if and when the task is performed and what is considered a successful exit code from the task

For more information about the Task Sequence Editor, see [Operating System Deployment: Task Sequence Editor](#).

Common Properties and Options for Task Sequence Step Types

Each task sequence group and step has configurable settings on the **Properties** and **Options** tabs that are common to all task sequence groups and steps. These common settings are briefly described in the following sections.

Common Properties

Table 1 shows the settings that are available on the **Properties** tab of each task sequence step. For more information about the **Properties** tab for a particular task sequence step, see the topic that corresponds to the step later in this reference.

Note The task sequence step types listed here are those that are available in the Deployment Workbench. Additional task sequence step types might be available when configuring task sequences using Microsoft System Center 2012 R2 Configuration Manager.

Table 1. Settings Available on the Properties Tab

Name	Description	Group	Step
Type	A read-only value that indicates the task sequence group or step type. The type will be set to one of these values: <ul style="list-style-type: none">• Apply Network Settings• Authorize DHCP• Capture Network Settings• Configure ADDS• Configure DHCP• Configure DNS• Enable BitLocker• Format and Partition Disk• Gather• Group• Inject Drivers• Install Application• Install Operating System• Install Roles and Features• Install Updates Offline• Recover From Domain Join Failure• Restart computer• Run Command Line• Set Task Sequence Variable• Validate	●	●
Name	A user-defined name that should allow easy identification and differentiation from other task sequence steps.	●	●
Description	A user-defined description that should make the task sequence step requirements and tasks easily understandable.	●	●

Common Options

Table 2 shows the settings that are available on the **Options** tab of a task sequence step. For more information about the **Options** tab, see [Task Sequence Options Tab](#).

Table 2. Settings Available on the Options Tab

Name	Description	Group	Step
Disable this step	Select this option to disable this task sequence step.	●	●
Success codes	Exit codes of the utility associated with this task sequence step that indicate that the step has finished successfully.		●
Continue on error	Select this option to allow the Task Sequencer to process additional task sequence steps if a failure occurs.	●	●
Conditional statements	<p>One or more conditions that limit the running of this task sequence group or step. These conditional are based on the following:</p> <ul style="list-style-type: none"> • File properties • Folder properties • Operating system version: <ul style="list-style-type: none"> • Is a certain architecture • Is a certain version • Query Windows Management Instrumentation (WMI) • Registry setting: <ul style="list-style-type: none"> • Exists • Does not exist • Equals • Does not equal • Greater than • Greater than or equals • Less than • Less than or equals • Installed software • Task sequence variable: <ul style="list-style-type: none"> • Exists • Equals • Does not equal • Greater than • Greater than or equals • Less than • Less than or equals <p>These conditions can be grouped using IF</p>	●	●

Name	Description	Group	Step
	statements that test all conditions, any condition, or no condition that evaluates as True.		

Note Additional conditional statements might be available when using Configuration Manager to configure task sequence steps.

Specific Properties and Settings for Task Sequence Step Types

Some properties and parameters of each task sequence step type are unique to that type. Each type with unique properties and settings is shown in the following sections with its unique task sequence step properties and settings.

Apply Network Settings

This task sequence step configures the network adapter on the target computer. For more information about what script accomplishes this task and which properties are used, see [ZTINICConfig.wsf](#).

The unique properties and settings for the **Apply Network Settings** task sequence step type are:

Properties

Name	Value
Type	Apply Network Settings

Settings

Name	Value
Name	The name to be assigned to the network connection.
Obtain an IP address automatically	When selected, Dynamic Host Configuration Protocol (DHCP) is used to obtain the required Internet Protocol (IP) configuration settings for the network connection. This is the default selection.
Use the following IP address	When selected, you can provide one or more IP address and subnet mask combinations in addition to gateways that will be assigned to the network connection.
Obtain a Domain Name System (DNS) server automatically	When selected, DHCP is used to obtain the required IP configuration settings for the network connection. This is the default selection.

Name	Value
Use the following DNS servers	When selected, you can provide one or more DNS server IP addresses that will be assigned to the network connection.
DNS Suffix	The DNS suffix that will be applied to all network connections that use TCP/IP.
Register this connection's address in DNS	Specifies that the computer will attempt dynamic registration of the IP addresses (through DNS) of this connection with the full computer name of this computer.
Use this connection's DNS suffix in DNS registration	Specifies whether DNS dynamic update is used to register the IP addresses and the connection-specific domain name of this connection.
WINS server addresses	You can provide one or more Windows Internet Naming Service (WINS) server IP addresses that will be assigned to the network connection.
Enable LMHOSTS lookup	Specifies whether a local area network (LAN) Manager Hosts (LMHOSTS) file for network basic input/output system (NetBIOS) name resolution is used.
Default	Specifies whether this network connection obtains the setting to enable or disable NetBIOS over TCP/IP (NetBT) from a DHCP server. This is the default selection.
Enable NetBIOS over TCP/IP	Specifies that this network connection uses NetBT and WINS.
Disable NetBIOS over TCP/IP	Specifies that this network connection does not use NetBT and WINS.

Authorize DHCP

This task sequence step authorizes the target computer as a DHCP server. For more information about which script accomplishes this task and which properties you use, see [ZTIAuthorizeDHCP.wsf](#).

The unique properties and settings for the **Authorize DHCP** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Authorize DHCP Server .

Settings

Name	Description
Account	A user account that is a member of the Enterprise Admins group, to be used when authorizing DHCP for the target computer.

Capture Network Settings

This task sequence step gathers the network adapter settings from the target computer. For more information about which script accomplishes this task and which properties you use, see [ZTINICConfig.wsf](#).

The unique properties and settings for the **Capture Network Settings** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Capture Network Settings .

Settings

Name	Description
None	None

Configure ADDS

This task sequence step configures the target computer as an Active Directory® Domain Services (AD DS) domain controller. For more information about the settings listed in the following tables and which this task sequence step can configure, see the Microsoft Help and Support article, [Unattended promotion and demotion of Windows 2000 and Windows Server 2003 domain controllers](#).

The unique properties and settings for the **Configure ADDS** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Configure ADDS .

Settings

Name	Description
Create	Specifies the configuration set that will be used to configure the target computer. The configuration sets are:

Name	Description
	<ul style="list-style-type: none"> New domain controller replica. Creates an additional domain controller in an existing AD DS domain New read-only domain controller (RODC) replica. Creates an RODC New domain in existing forest. Creates a domain in an existing AD DS forest New domain tree in existing forest. Creates a new tree in an existing AD DS forest New forest. Creates a new AD DS forest
Domain DNS name	The DNS name of the new or existing domain.
Domain NetBIOS name	The NetBIOS name of the new child domain, child domain tree, or forest that pre-AD DS clients use to access the domain. This name must be unique on the network.
DNS name	The DNS name of the child domain or domain tree.
Replication source domain controller	The name of the domain controller from which to source AD DS on new replica or backup domain controller upgrade installations. If no value is supplied, the closest domain controller from the domain being replicated will be selected by default.
Account	The account to be used to perform the configuration.
Recovery (safe mode) password	The password for the offline Administrator account that is used in AD DS Repair mode.
Install DNS if not already present	When selected, DNS will be installed if it has not already been installed.
Make this domain controller a global catalog (GC) server	Specifies whether the replica will also be a GC server. When selected, the target computer will be configured as a GC server if the replication source domain controller is a GC server.
Wait for critical replication only	When selected, this setting specifies that only critical replication is sourced during the replication phase of Dcpromo. Noncritical replication resumes when the computer restarts as a domain controller.
Forest functional level	<p>Specifies the functional level for a new forest. Available options are:</p> <ul style="list-style-type: none"> Windows Server 2003 Windows Server 2008 Windows Server 2008 R2

Name	Description
Domain functional level	Specifies the functional level for a new domain. Available options are: <ul style="list-style-type: none"> • Windows Server 2003 • Windows Server 2008 • Windows Server 2008 R2
Database	Fully qualified, non-Universal Naming Convention (UNC) directory on a hard disk of the local computer that will host the AD DS database (NTDS.dit). If the directory exists, it must be empty. If it does not exist, it will be created. Free disk space on the logical drive selected must be 200 megabytes (MB) and possibly larger when rounding errors are encountered and to accommodate all objects in the domain. For best performance, the directory should be located on a dedicated hard disk.
Log files	Fully qualified, non-UNC directory on a hard disk on the local computer to host the AD DS log files. If the directory exists, it must be empty. If it does not exist, it will be created.
SYSVOL	Fully qualified, non-UNC directory on a hard disk of the local computer that will host the AD DS System Volume (SYSVOL) files. If the directory exists, it must be empty. If it does not exist, it will be created. The directory must be located on a partition that is formatted with the NTFS version 5.0 file system. For best performance, the directory should be located on a different physical hard disk than the operating system.
Site name	The value of an existing AD DS site on which to locate the new domain controller. If not specified, an appropriate site will be selected. This option only applies to the new tree in a new forest scenario. For all other scenarios, a site will be selected using the current site and subnet configuration of the forest.

Configure DHCP

This task sequence step configures the DHCP server service on the target computer. For more information about which script accomplishes this task and which properties you use, see [ZTIConfigureDHCP.wsf](#).

The unique properties and settings for the **Configure DHCP** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Configure DHCP Server .

Settings

Name	Description
Name	Configure DHCP
Scope Details	<p>These options apply to any client computers that obtain a lease within that particular scope. Configured scope option values always apply to all computers obtaining a lease in a given scope unless they are overridden by options assigned to class or client reservation.</p> <p>Within the Scope Details setting, the following sub-settings are configurable:</p> <ul style="list-style-type: none"> • Scope Name. A user-definable name • Start IP address. The starting IP address for the scope • End IP address. The ending IP address for the scope • Subnet mask. The subnet mask of the client subnet • Lease duration for DHCP clients. The duration that the DHCP lease is valid for the client • Description. A description of the scope • Exclude IP address range, Start IP address. The starting IP address for the range of IP addresses that are to be excluded from the scope • Exclude IP address range, End IP address. The ending IP address for the range of IP addresses that are to be excluded from the scope • 003 Router. A list of IP addresses for routers on the client subnet • 006 DNS Servers. A list of IP addresses for DNS name servers available to the client • 015 DNS Domain Name. The domain name that the DHCP client should use when resolving unqualified domain names with DNS • 044 WINS/NBNS Servers. Lists the IP addresses for NetBIOS name servers (NBNSes) on the network • 046 WINS/NBT Node Type. Configures the client node type for NetBT clients • 060 PXE Client. The address used for Pre-Boot

Name	Description
	Execution Environment (PXE) client bootstrap code
Server Options	<p>These options apply globally for all scopes and classes defined at each DHCP server and for any clients that a DHCP server services. Configured server option values always apply unless they are overridden by options assigned to other scope, class, or client reservation.</p> <p>Within the Server Options setting, the following sub-settings are configurable:</p> <ul style="list-style-type: none"> • 003 Router. A list of IP addresses for routers on the client subnet • 006 DNS Servers. A list of IP addresses for DNS name servers available to the client • 015 DNS Domain Name. The domain name that the DHCP client should use when resolving unqualified domain names with the DNS • 044 WINS/NBNS Servers. Lists the IP addresses for NBNSes on the network • 046 WINS/NBT Node Type. Configures the client node type for NetBT clients • 060 PXE Client. The address used for PXE client bootstrap code

Configure DNS

This task sequence step configures DNS on the target computer. For more information about which script accomplishes this task and which properties you use, see [ZTIConfigureDNS.wsf](#).

The unique properties and settings for the **Configure DNS** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Configure DNS Server .

Settings

Name	Description
Name	Configure DNS
Zones	Within the Scope Details setting, the following sub-settings are configurable:

Name	Description
	<ul style="list-style-type: none"> • DNS zone name. A user-definable name • Type. The type of DNS zone to be created • Replication. Specifies the replication scheme used to share information among DNS servers • Zone file name. The zone's DNS database file • Dynamic updates. Enables DNS client computers to register and dynamically update their resource records with a DNS server whenever changes occur • Scavenge stale resource records. Removes stale resource records
Server Properties	<p>Within the Server Properties setting, the following sub-settings are configurable:</p> <ul style="list-style-type: none"> • Disable recursion. Specifies that the DNS server will not perform recursion on any query • BIND secondaries. Specifies whether to use fast transfer format to transfer a zone to DNS servers running legacy Berkeley Internet Name Domain (BIND) implementations • Fail on load if bad data. Specifies the DNS server should parse files strictly • Enable round robin. Specifies the DNS server should use the round robin mechanism to rotate and reorder a list of resource records if multiple resource records exist of the same type exist for a query answer • Enable netmask ordering. Specifies whether the DNS server should reorder resource records within the same resource record set in its response to a query based on the IP address of the source of the query • Secure cache against pollution. Specifies whether the DNS server will attempt to clean up responses to avoid cache pollution • Name checking. Configures the name-checking method to be used

Note The **Configure DNS** task sequence step uses the DnsCmd tool, which is included in Windows Support Tools, to configure DNS. Be sure that Windows Support Tools is installed before running the **Configure DNS** task sequence step.

Note For more information about these server properties, see [DnsCmd](#).

Enable BitLocker

This task sequence step configures BitLocker® Drive Encryption on the target computer. For more information about this step type, see [Enable BitLocker](#).

The unique properties and settings for the **Enable BitLocker** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Enable BitLocker .

Settings

Name	Description
Current operating system drive	When selected, the operating system drive will be configured. This is the default selection.
Specific drive	When selected, the specified drive will be configured.
TPM only	When selected, the Trusted Platform Module (TPM) is required. This is the default selection.
Startup key on USB only	When selected, a startup key is required on the specified USB drive.
TPM and startup key on USB	When selected, the TPM is required in addition to a startup key on the specified USB drive.
In Active Directory	When selected, the recovery key is stored in AD DS. This is the default selection.
Do not create a recovery key	When selected, the recovery key is not created. Using this option is not recommended.
Wait for BitLocker to complete	When selected, this step will not finish until after BitLocker has finished processing all drives.

Execute Runbook

This task sequence step runs Microsoft System Center 2012 Orchestrator runbooks on the target computer. An Orchestrator *runbook* is the sequence of activities that orchestrate actions on computers and networks. You can initiate Orchestrator runbooks in MDT using this task sequence step type.

Note This task sequence step is not included any MDT task sequence templates. You must add this task sequence step to any task sequences you create.

The unique properties and settings for the **Execute Runbook** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Execute Runbook .
Name	The name of the task sequence step, which should reflect the name of the runbook being run.
Description	Informative text that provides additional information about the task sequence step

Settings

Name	Description
Orchestrator Server	<p>Type the URL for the Orchestrator web service, which includes the server name. The Orchestrator web service can use either Hypertext Transfer Protocol (HTTP) or HTTP over Secure Sockets Layer (HTTPS). The Orchestrator web service defaults to port 81.</p> <p>The Orchestrator web service supports multiple runbook servers. By default, a runbook can run on any runbook server. A runbook can be configured to specify which runbook servers should be used to run the runbook.</p> <p>Note The Orchestrator web service supports the ability to run a runbook on a specific runbook server. This feature is not supported in MDT.</p> <p>Specify the URL in any of the following formats:</p> <ul style="list-style-type: none"> • servername. When using this format, the URL defaults to: <code>http://<servername>: 81/Orchestrator2012/Orchestrator.svc</code> • servername:port. When using this format, the URL defaults to: <code>http://<servername: port>/Orchestrator2012/Orchestrator.svc.</code> • http://servername:port. When using this format, the URL defaults to: <code>http://<servername: port>/Orchestrator2012/Orchestrator.svc.</code> • https://servername:port. When using this format, the URL defaults to: <code>https://<servername: port>/Orchestrator2012/Orchestrator.svc.</code> • http://servername:port/Orchestrator2012/Orchestrator.svc. When using this format, MDT assumes that you are providing the fully qualified URL,

Name	Description
	<p>because the value ends with .svc.</p> <ul style="list-style-type: none"> • https://servername:port/Orchestrator2012/Orchestrator.svc. When using this format, MDT assumes that you are providing the fully qualified URL, because the value ends with .svc.
Runbook	<p>Click Browse, and then select the name of the Orchestrator runbook that this task sequence should run.</p> <p>Note To successfully browse for Orchestrator runbooks, install the ADO.NET Data Services Update for .NET Framework 3.5 SP1 for Windows 7 and Windows Server 2008 R2.</p>
Automatically provide runbook parameters	<p>Select this option to automatically provide the Orchestrator runbook input parameter values(which assumes that the runbook parameter values are task sequence variables). For example, if a runbook has an input parameter named OSDComputerName, then the OSDComputerName task sequence variable value is passed to the runbook.</p> <p>Note This option works only for input parameters that are valid task sequence variable names and do not contain spaces or other special characters. Although spaces and other special characters are supported as Orchestrator parameter names, they are not valid task sequence variable names. If you need to pass values to parameters with spaces or other special characters, use the Specify explicit runbook parameters option.</p> <p>The other option is Specify explicit runbook parameters.</p> <p>Note The values provided for the runbook input parameters to the Orchestrator web service are formatted as XML. Passing values that contain data that is or resembles XML-formatted data may cause errors.</p>
Specify explicit runbook parameters	<p>Select this option to explicitly provide the Orchestrator runbook input parameters.</p> <p>You must configure the following settings for each input parameter that the Orchestrator runbook requires:</p> <ul style="list-style-type: none"> • Name. This is the name of the input runbook parameter. <p>Note If you change the parameters for an existing Orchestrator runbook, you need to browse (reselect) for the runbook again, because MDT only retrieves the parameter list when initially adding the Orchestrator runbook.</p> <ul style="list-style-type: none"> • Value. This can be a constant or a variable, such as a task sequence variable or an environment

Name	Description
Wait for the runbook to finish before continuing	<p>variable. For example, you can specify a value of %OSDComputerName%, which will pass the value of the OSDComputerName task sequence variable to the runbook input parameter.</p> <p>This check box controls whether the task sequence step will wait for the runbook to finish before proceeding to the next task sequence step.</p> <p>If this check box is:</p> <ul style="list-style-type: none"> Selected, then the task sequence step will wait for the runbook to finish before proceeding on to the next task sequence step. <p>When this check box is selected, the task sequence step will poll the Orchestrator web service for the runbook to finish. The amount of time between polls starts at 1 second, then increases to 2, 4, 8, 16, 32, and 64 seconds between each poll. Once the amount of time reaches 64 seconds, the task sequence step continues to poll every 64 seconds.</p> <ul style="list-style-type: none"> Cleared, then the task sequence step will not wait for the runbook to finish before proceeding to the next task sequence step. <p>Note This check box must be selected if the runbook returns output parameters.</p>

Format and Partition Disk

This task sequence step partitions and formats disks on the target computer. For more information about this step type, see [Format and Partition Disk](#).

The unique properties and settings for the **Format and Partition Disk** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Format and Partition Disk .

Settings

Name	Description
Disk number	The physical number of the disk to be configured.
Disk type	The type of drive to be created. Values are:

Name	Description
	<ul style="list-style-type: none"> • Standard (MBR) (Master Boot Record) • GPT (GUID [globally unique identifier] Partition Table). <p>The default selection is Standard (MBR).</p>
Volume	<p>Within the Volume setting, the following sub-settings are configurable:</p> <ul style="list-style-type: none"> • Partition Name. A user-definable name. • Partition Type. Values vary by disk type: <ul style="list-style-type: none"> • MBR: Primary only • GPT: Primary, EFI, or MSR • Use a percentage of remaining space. • Use specific drive size. Values are in increments of 1 MB or 1 gigabyte (GB). • Make this a boot partition. • File System. Values are NTFS or FAT32. • Quick Format. When selected, a quick format is performed. • Variable. The drive letter that was assigned to this newly configured partition.

Note When using the CustomSettings.ini file to specify the hard disk and partition configurations, only the first hard disk and first two partitions will be configured. Edit ZTIGather.xml to configure additional hard disks or partitions.

Gather

This task sequence step gathers data and processing rules for the target computer. The unique properties and settings for the **Gather** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Gather .

Settings

Name	Description
Gather only local data	When selected, this step processes only the properties contained in the ZTIGather.xml file.
Gather local data and process rules	When selected, this step processes the properties contained in the ZTIGather.xml file and the properties

Name	Description
	contained in the file that the Rules file specifies. This is the default selection.
Rules file	The name of the Rules file to process. If left blank, the task sequence step attempts to locate and process the CustomSettings.ini file.

Note This task sequence step is natively available in System Center 2012 R2 Configuration Manager as **Set Dynamic Variables** in the General group.

Inject Drivers

This task sequence step injects drivers that have been configured for deployment to the target computer. The unique properties and settings for the **Inject Drivers** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Inject Drivers .

Settings

Name	Description
Install only matching drivers	Injects only the drivers that the target computer requires and that match what is available in Out-of-Box Drivers
Install all drivers	Installs all drivers
Selection profile	Installs all drivers in the selected profile

Install Application

This task sequence step installs applications on the target computer. For more information about this step type, see [Install Software](#).

The unique properties and settings for the **Install Application** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Install Application .

Settings

Name	Description
Install multiple	Install mandatory applications that the

Name	Description
applications	MandatoryApplications property has specified and optional applications that the Applications property has specified. These properties are configured by rules or are specified during the Deployment Wizard interview process. This is the default selection.
Install a single application	The specific application to install. You select the application from a drop-down list that consists of applications that have been configured in the Applications node of the Deployment Workbench.
Success codes	A space-delimited list of application installation exit codes that should be used when determining the successful installation of applications.

Install Operating System

This task sequence step installs an operating system on the target computer. MDT can deploy Windows 8.1, Windows 8, Windows 7, Windows Server 2012 R2, Windows Server 2012, and Windows Server 2008 R2 using:

- **setup.exe.** This method is the traditional method used, initiated by running setup.exe from the installation media. MDT uses setup.exe by default.
- **imagex.exe.** This method installs the operating system image using imagex.exe with the **/apply** option. MDT uses this method when the setup.exe method cannot be used (i.e., it falls back to using imagex.exe).

You can control which of these methods is used by using the **ForceApplyFallback** property, which also affects which operating system task sequences are listed in the Deployment Wizard for a specific processor architecture boot image. For more information, see the [ForceApplyFallback](#) property.

The unique properties and settings for the **Install Operating System** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Install Operating System .

Settings

Name	Description
Operating system to install	The name of the operating system to be installed on the target computer. You select the operating system from a drop-down list compiled from operating systems that have been configured in the Operating Systems node of

Name	Description
	the Deployment Workbench.
Disk	The disk on which to install the operating system.
Partition	The partition on which to install the operating system.

Install Roles and Features

This task sequence step installs the selected roles and features on the target computer. For more information about which script accomplishes this task and the properties used, see [ZTOSRole.wsf](#).

The unique properties and settings for the **Install Roles and Features** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Install Roles and Features .
Description	Informative text that describes the purpose of the task sequence step.

Settings

Name	Description
Select the operating system for which the roles are to be installed	Select the operating system to be deployed to the target computer.
Select the roles and features that should be installed	Select one or more roles and features for installation on the target computer.

Install Language Packs Offline

This task sequence step installs updates to the image on the target computer after the operating system has been deployed but before the target computer has been restarted. These updates include language packs. For more information about which script accomplishes this task and which properties you use, see [ZTIPatches.wsf](#).

The unique properties and settings for the **Install Language Packs Offline** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Install Updates Offline .

Settings

Name	Description
Package Name	The name of the language pack package that should be applied to the target computer

Note This task sequence step is valid only when using MDT with Configuration Manager.

Install Language Packs Online

This task sequence step installs language packs to the image on the target computer after the operating system has been deployed and after the target computer has been restarted. For more information about which script accomplishes this task and which properties you use, see [ZTILangPacksOnline.wst](#).

The unique properties and settings for the **Install Language Packs Online** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Install Language Packs Online .

Settings

Name	Description
Package Name	The name of the language pack package that should be applied to the target computer

Note This task sequence step is valid only when using MDT with Configuration Manager.

Install Updates Offline

This task sequence step installs updates to the image on the target computer after the operating system has been deployed but before the target computer has been restarted. These updates include language packs. For more information about which script accomplishes this task and which properties you use, see [ZTIPatches.wst](#).

The unique properties and settings for the **Install Updates Offline** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Install Updates Offline .

Settings

Name	Description
Selection Profile	<p>The name of the selection profile that should be applied to the target computer</p> <p>Note When using MDT with Configuration Manager, specify the name of the update package that should be applied.</p>

Recover from Domain Join Failure

This task sequence step verifies that the target computer has joined a domain. The unique properties and settings for the **Recover from Domain Join Failure** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Recover from Domain Join Failure .

Settings

Name	Description
Auto recover	The task sequence step attempts to join the target computer to a domain.
Manual recover	If the target computer fails to join a domain, the task sequence step causes the Task Sequencer to pause, allowing you to attempt to join the target computer to a domain.
No recover	If the target computer is not able to join a domain, the task sequence fails, stopping the task sequence.

Restart computer

This task sequence step restarts the target computer. The unique properties and settings for the **Restart computer** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Restart computer .

Settings

Name	Description
None	None

Run Command Line

This task sequence step runs the specified commands on the target computer. For more information about this step type, see [Run Command Line](#).

The unique properties and settings for the **Run Command Line** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Run Command Line .

Settings

Name	Description
Command Line	The commands to be run when this task sequence step is processed
Start in	The starting folder for the application (The path must be a valid path on the target computer.)
Run this step as the following account	Allows specification of user credentials that will be used to run the specified command
Account	The user credentials that will be used to run the specified command
Load the user's profile	When selected, loads the user profile for the specified account

Run PowerShell Script

This task sequence step runs the specified Windows PowerShell™ script on the target computer. For more information about what script accomplishes this task and which properties are used, see [ZTIPowerShell.wsf](#).

The unique properties and settings for the **Run PowerShell Script** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Run PowerShell Script .

Settings

Name	Description
PowerShell script	The Windows PowerShell script to be run when this task sequence step is processed

Parameters	<p>The parameters to be passed to the Windows PowerShell script. These parameters should be specified the same as if you were adding them to the Windows PowerShell script from a command line.</p> <p>The parameters provided should be only those parameters the script consumes, not for the Windows PowerShell command line.</p> <p>The following example would be a valid value for this setting:</p> <pre>-MyParameter1 MyValue1 -MyParameter2 MyValue2</pre> <p>The following example would be an invalid value for this setting (bold items are incorrect):</p> <pre>-nologo -executinpolicy unrestricted -File MyScript.ps1 -MyParameter1 MyValue1 -MyParameter2 MyValue2</pre> <p>The previous example is invalid, because the value includes Windows PowerShell command-line parameters (-nologo and -executionpolicy unrestricted).</p>
-------------------	--

Note This task sequence step is natively available in System Center 2012 R2 Configuration Manager as **Run PowerShell Script** in the General group.

Set Task Sequence Variable

This task sequence step sets the specified task sequence variable to the specified value. For more information about this step type, see [Set Task Sequence Variable](#).

The unique properties and settings for the **Set Task Sequence Variable** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Set Task Sequence Variable .

Settings

Name	Description
Task Sequence Variable	The name of the variable to modify
Value	The value to assign to the specified variable

Uninstall Roles and Features

This task sequence step uninstalls the selected roles and features from the target computer. For more information about which script accomplishes this task and the properties used, see [ZTOSRole.wsf](#).

The unique properties and settings for the **Uninstall Roles and Features** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Uninstall Roles and Features .
Description	Informative text that describes the purpose of the task sequence step.

Settings

Name	Description
Select the operating system for which the roles are to be installed	Select the operating system to be deployed to the target computer.
Select the roles and features that should be installed	Select one or more roles and features for uninstallation from the target computer.

Validate

This task sequence step verifies that the target computer meets the specified deployment prerequisite conditions. The unique properties and settings for the **Validate** task sequence step type are:

Properties

Name	Description
Type	Set this read-only type to Validate .

Settings

Name	Description
Ensure minimum memory	When selected, this step verifies that the amount of memory, in megabytes, installed on the target computer meets or exceeds the amount specified. This is a default selection.
Ensure minimum processor speed	When selected, this step verifies that the speed of the processor, in megahertz (MHz), installed in the target computer meets or exceeds the amount specified. This is a default selection.
Ensure specified image size will fit	When selected, this step verifies that the amount of free disk space, in megabytes, on the target computer meets or exceeds the amount specified.
Ensure current operating system to be refreshed	When selected, this step verifies that the operating system installed on the target computer meets the requirement specified. This is a default selection.

Note This task sequence step is natively available in System Center 2012 R2 Configuration Manager as **Check Readiness** in the General group.

Out-of-Box Task Sequence Steps

The following task sequence steps are referenced by one or more of the available task sequence templates included with MDT. Each of the following examples lists the preconfigured properties, parameters, and options and can be used as a basis for building custom task sequences.

Only the task sequence step properties, parameters, and options, and their corresponding values are listed in the examples.

Note For more information about each task sequence step, see the corresponding topics in [Common Properties and Options for Task Sequence Step Types](#) and [Specific Properties and Settings for Task Sequence Step Types](#).

Apply Network Settings

This task sequence step configures the network adapter on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties are used, see [ZTINICConfig.wsf](#).

The default configuration of the **Apply Network Settings** task sequence step is:

Properties

Name	Value
Type	Apply Network Settings
Name	Apply Network Settings
Description	Not specified

Settings

Name	Value
	No parameters are preconfigured for this step. This causes this step, by default, to configure the network adapter to use DHCP.

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Note When using the CustomSettings.ini file to specify the network adapter configurations, only the first network adapter will be configured. Edit ZTIGather.xml to configure additional network adapters.

Apply Patches

This task sequence step installs updates to the image on the target computer after the operating system has been deployed but before the target computer has been restarted. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties you use, see [ZTIPatches.wsf](#).

The default configuration of the **Install Updates Offline** task sequence step is:

Properties

Name	Value
Type	Install Updates Offline
Name	Apply Patches
Description	Not specified

Settings

Name	Value
Selection profile	The name of the profile used when selecting the patches to install on the target computer

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Apply Windows PE

This task sequence step prepares the target computer to start in Windows Preinstallation Environment (Windows PE). Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties you use, see [LTIApply.wsf](#).

The default configuration of the **Apply Windows PE** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Apply Windows PE
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCIRIPTROOT%\LTIApply.wsf" /PE
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Backup

This task sequence step backs up the target computer before starting the operating system deployment. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties you use, see [ZTIBackup.wsf](#).

The default configuration of the **Backup** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Backup
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\ZTI Backup.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Capture Groups

This task sequence step captures group membership of local groups that exist on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties you use, see [ZTIGroups.wsf](#).

The default configuration of the **Capture Groups** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Capture Groups
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\ZTI Groups.wsf" /capture
Start in	Not specified.
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Capture User State

This task sequence step captures the user state for user profiles that exist on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about what script accomplishes this task and what properties are used, see [ZTIUserState.wsf](#). For more information about this step type, see [Capture User State](#).

The default configuration of the **Capture User State** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Capture User State
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCRI PTROOT%\ZTI UserState.wsf" /capture
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Check BIOS

This task sequence step checks the basic input/output system (BIOS) of the target computer to ensure that it is compatible with the operating system you are deploying. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties are used, see [ZTIBIOSCheck.wsf](#).

The default configuration of the **Check BIOS** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Check BIOS
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCRIptroot%\ZTI BI OSCheck.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Configure

This task sequence step configures the Unattend.xml file with the required property values that are applicable to the operating system you are deploying to the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties you use, see [ZTIConfigure.wsf](#).

The default configuration of the **Configure** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Configure
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCRIptroot%\ZTI Configure.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Copy Scripts

This task sequence step copies the deployment scripts used during the deployment processes to a local hard disk on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties you use, see [LTIcopyScripts.wsf](#).

The default configuration of the **Copy Scripts** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Copy Scripts
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCIRIPTROOT%\LTI CopyScripts.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Copy Sysprep Files

This task sequence step copies the Sysprep files to the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and which properties you use, see [LTISysprep.wsf](#).

The default configuration of the **Copy Sysprep Files** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Copy Sysprep Files
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\LTI Sysprep.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Create BitLocker Partition

This task sequence step sets the **BDEInstall** property to True, indicating that BitLocker should be installed on the target computer. The unique properties and settings for the **Create BitLocker Partition** task sequence step type are:

Properties

Name	Value
Type	Set Task Sequence Variable

Name	Value
Name	Create BitLocker Partition
Description	None

Settings

Name	Value
Task Sequence Variable	BDE Install
Value	True

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Create WIM

This task sequence step creates a backup of the target computer. The unique properties and settings for the **Create WIM** task sequence step type are:

Properties

Name	Value
Type	Run Command Line
Name	Create WIM
Description	None

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\ZTI Backup.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Disable BDE Protectors

If BitLocker is installed on the target computer, this task sequence step disables the BitLocker protectors.

The unique properties and settings for the **Disable BDE Protectors** task sequence step type are:

Properties

Name	Value
Type	Run Command Line
Name	Disable BDE Protectors
Description	None

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\ZTIDisableBDEProtectors.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Enable BitLocker

This task sequence step enables BitLocker on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about which script accomplishes this task and what properties are used, see [ZTIBde.wsf](#).

The default configuration of the **Enable BitLocker** task sequence step is:

Properties

Name	Value
Type	Enable BitLocker
Name	Enable BitLocker
Description	None

Settings

Name	Value
Current operating system drive	Selected
TPM only	Selected
Startup key on USB only	Not selected
TPM and startup key on USB	Not selected
Specific drive	Not selected
In Active Directory	Selected
Do not create a recovery key	Not selected
Wait for BitLocker to complete	Not selected

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	BdeInstallSuppress does not equal YES

Enable OEM Disk Configuration

This task sequence step sets the **DeploymentType** property to **NEWCOMPUTER**, which allows the target computer's disk to be partitioned and formatted.

The unique properties and settings for the **Enable OEM Disk Configuration** task sequence step type are:

Properties

Name	Value
Type	Set Task Sequence Variable
Name	Enable OEM Disk Configuration
Description	None

Settings

Name	Value
Task Sequence Variable	DeploymentType
Value	NEWCOMPUTER

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

End Phase

This task sequence step ends the current deployment phase and restarts the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

The default configuration of the **End Phase** task sequence step is:

Properties

Name	Value
Type	Restart computer
Name	End Phase

Name	Value
Description	Not specified

Settings

Name	Value
None	None

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Execute Sysprep

This task sequence step starts Sysprep on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates. For more information about what script accomplishes this task and what properties are used, see [LTISysprep.wsf](#).

The default configuration of the **Execute Sysprep** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Execute Sysprep
Description	None

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\LTISysprep.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Force Diskpart Action

If the C:\oem.wsf file exists, this task sequence step deletes the C:\oem.wsf file, which will allow the **Format and Partition Disk** task sequence step to run. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

The default configuration of the **Force Diskpart Action** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Force Diskpart Action
Description	Not specified

Settings

Name	Value
Command line	cmd.exe /c if exist c:\oem.wsf del /q c:\oem.wsf
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0.1
Continue on error	Selected
Conditional qualifier	None

Format and Partition Disk

This task sequence step configures and formats disk partitions on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIDiskpart.wsf](#).

The default configuration of the **Format and Partition Disk** task sequence step is:

Properties

Name	Value
Type	Format and Partition Disk
Name	Format and Partition Disk
Description	Not specified

Settings

Name	Value
Disk number	0
Disk type	Standard (MBR)
Volume	Within the Volume setting, the following sub-settings are configured: <ul style="list-style-type: none"> • Partition Name. OSDisk • Partition Type. Primary • Use a percentage of remaining space. Selected • Size(%). 100 • Use specific drive size. Not selected • Make this a boot partition. Selected • File System. NTFS • Quick Format. Selected • Variable. Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010

Name	Value
Continue on error	Not selected
Conditional qualifier	Not specified

Note When using the CustomSettings.ini file to specify the hard disk and partition configurations, only the first hard disk and first two partitions will be configured. Edit ZTIGather.xml to configure additional hard disks or partitions.

Gather local only

This task sequence step gathers deployment configurations settings from local sources that apply to the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIGather.wsf](#).

The default configuration of the **Gather local only** task sequence step is:

Properties

Name	Value
Type	Gather
Name	Gather local only
Description	Not specified

Settings

Name	Value
Gather only local data	Selected
Gather local data and process rules	Not selected
Rules file	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	None

Generate Application Migration File

This task sequence step generates the ZTIAppXmlGen.xml file, which contains a list of file associations that are installed on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIAppXmlGen.wsf](#).

The default configuration of the **Generate Application Migration File** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Generate Application Migration File
Description	Not specified

Settings

Name	Value
Command Line	cscript.exe "%SCIRIPTROOT%\ZTI AppXml Gen.wsf" /capture
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	None

Inject Drivers

This task sequence step injects drivers that have been configured for deployment to the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIDrivers.wsf](#).

The default configuration of the **Inject Drivers** task sequence step is:

Properties

Name	Value
Type	Inject Drivers
Name	Inject Drivers
Description	Not specified

Settings

Name	Value
Install only matching drivers	Injects only the drivers which are required by the target computer and match with what is available in Out-of-Box Drivers
Install all drivers	Injects all drivers
Selection profile	Injects drivers which are associated with the selected profile

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Install Applications

This task sequence step installs applications on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIAplications.wsf](#).

The default configuration of the **Install Applications** task sequence step is:

Properties

Name	Value
Type	Install Applications
Name	Install Applications

Name	Value
Description	Not specified

Settings

Name	Value
Install multiple applications	Selected
Install a single application	Not selected

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Install Operating System

This task sequence step installs an operating system on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

The default configuration of the **Install Operating System** task sequence step is:

Properties

Name	Value
Type	Install Operating System
Name	Install Operating System
Description	Not specified

Settings

Name	Value
Operating system to install	This value corresponds to the operating system that was selected when the task sequence was created.
Disk	The disk where the operating system is to be installed.
Partition	The partition where the operating system is to be

Name	Value
	installed.

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Next Phase

This task sequence step updates the **Phase** property to the next phase in the deployment process. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTINextPhase.wsf](#).

The default configuration of the **Next Phase** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Next Phase
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\ZTI Next Phase.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010

Name	Value
Continue on error	Not selected
Conditional qualifier	Not specified

Post-Apply Cleanup

This task sequence step cleans up unnecessary files after the installation of an image on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [LTIAppl.wsf](#).

The default configuration of the **Post-Apply Cleanup** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Post-Apply Cleanup
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCIRIPTROOT%\LTIAppl.wsf" /post
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Recover from Domain

This task sequence step will verify the target computer has joined a domain. For more information about which script accomplishes this task and which properties are used, see [ZTIDomainJoin.wsf](#).

The unique properties and settings for the **Recover from Domain** task sequence step type are:

Properties

Name	Description
Type	This read-only type is set to Recover from Domain Join Failure .

Settings

Name	Description
Auto recover	The task sequence step will attempt to join the target computer to a domain.
Manual recover	If the target computer fails to join a domain, the task sequence step will cause the task sequencer to pause, allowing the user attempts to join the target computer to a domain.
No recover	If the target computer is not able to join a domain, the task sequence fails, stopping the task sequence.

Restart computer

This task sequence step restarts the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

The default configuration of the **Restart computer** task sequence step is:

Properties

Name	Value
Type	Restart computer
Name	Restart computer
Description	Not specified

Settings

Name	Value

Name	Value
None	None

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Restore Groups

This task sequence step restores the previously captured group membership of local groups on the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIGroups.wsf](#).

The default configuration of the **Restore Groups** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Restore Groups
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCIRIPTROOT%\ZTI Groups.wsf" /restore
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
None	None

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	If all conditions are true: <ul style="list-style-type: none">• DoCapture does not equal YES• DoCapture does not equal PREPARE

Restore User State

This task sequence step restores previously captured user state to the target computer. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIUserState.wsf](#).

For more information about this step type, see [Restore User State](#).

The default configuration of the **Restore User State** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Restore User State
Description	Not specified

Settings

Name	Value
Command Line	cscript.exe "%SCIRIPTROOT%\ZTIUserState.wsf" /restore
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010

Name	Value
Continue on error	Not selected
Conditional qualifier	If all conditions are true: <ul style="list-style-type: none"> • If DoCapture does not equal YES • If DoCapture does not equal PREPARE

Set Image Build

This task sequence step sets the **ImageBuild** property to the value contained in **OSCurrentVersion**. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

The default configuration of the **Set Image Build** task sequence step is:

Properties

Name	Value
Type	Set Task Sequence Variable
Name	Set Image Build
Description	Not specified

Settings

Name	Value
Task Sequence Variable	ImageBuild
Value	%OSCurrentVersion%

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Set Image Flags

This task sequence step sets the **ImageFlags** property to the value contained in **OSSKU**. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

The default configuration of the **Set Image Flags** task sequence step is:

Properties

Name	Value
Type	Set Task Sequence Variable
Name	Set Image Flags
Description	Not specified

Settings

Name	Value
Task Sequence Variable	ImageFlags
Value	%OSSKU%

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Tattoo

This task sequence step tattoos the target computer with identification and version information. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTITatoo.wsf](#).

The default configuration of the **Tattoo** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Tattoo
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCIRPTROOT%\ZTI Tatoo.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Validate

This task sequence step validates that the target computer meets the specified deployment prerequisite conditions. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIValidate.wsf](#).

The default configuration of the **Validate** task sequence step is:

Properties

Name	Value
Type	Validate
Name	Validate
Description	Not specified

Settings

Name	Value
Ensure minimum memory (MB)	Selected. The value selector is set to 768 .
Ensure minimum processor speed (MHz)	Selected. The value selector is set to 800 .

Name	Value
Ensure specified image size will fit (MB)	Not selected.
Ensure current operating system to be refreshed	Selected. The value selector is set to Server or Client , depending on the template used to create the task sequence.

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Windows Update (Pre-Application Installation)

This task sequence step installs updates to the target computer prior to the installation of applications. Following is a brief listing of the settings that show how this step was originally configured in one of the MDT task sequence templates.

For more information about what script accomplishes this task and what properties are used, see [ZTIWindowsUpdate.wsf](#).

The default configuration of the **Windows Update (Pre-Application Installation)** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Windows Update (Pre-Application Installation)
Description	Not specified

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\ZTI WindowsUpdate.wsf"
Start in	Not specified
Run this step as the	Not specified

Name	Value
following account	

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Windows Update (Post-Application Installation)

This task sequence step is the same as the **Windows Update (Pre-Application Installation)** task sequence step.

Wipe Disk

This task sequence step wipes all information from the disk using the **Format** command.

For more information about what script accomplishes this task and what properties are used, see [ZTIWipeDisk.wsf](#).

The default configuration of the **Wipe Disk** task sequence step is:

Properties

Name	Value
Type	Run Command Line
Name	Wipe Disk
Description	This will only run if WipeDisk=TRUE in CustomSettings.ini

Settings

Name	Value
Command line	cscript.exe "%SCRIPTROOT%\ZTI WipeDisk.wsf"
Start in	Not specified
Run this step as the following account	Not specified

Options

Name	Value
Disable this step	Not selected
Success codes	0 3010
Continue on error	Not selected
Conditional qualifier	Not specified

Properties

The scripts used in Lite Touch Installation (LTI) and ZTI reference properties to determine the process steps and configuration settings used during the deployment process. The scripts create some of these properties automatically. Other properties must be configured in the CustomSettings.ini file. Some of these properties are:

- Specific to ZTI only
- Specific to LTI only
- For use in both ZTI and LTI

Use this reference to help determine the correct properties to configure and the valid values to include for each property.

For each property the following information is provided:

- **Description.** Provides a description of the purpose of the property and any pertinent information regarding the customization of the property.
Note Unless explicitly specified for ZTI or LTI only, a property is valid for both ZTI and LTI.
- **Value and Description.** Indicates the valid values to be specified for the property and a brief description of what each value means. (Values in italics indicate that a value is substituted—for example the value *user1*, *user2* indicates that *user1* and *user2* would be replaced with the actual name of user accounts.)
- **Example.** Provides an example of a property use as it might appear in the .ini files.

For more information about these and other task sequence properties that might be referenced while performing a ZTI deployment, see [Operating System Deployment Task Sequence Variables](#).

The deployment scripts generally require values to be specified in upper case so that they are properly read. Therefore, when specifying property values, use uppercase letters.

Property Definition

The following sections describe the properties that are available for LTI and ZTI deployments in MDT.

Tip The properties are sorted in alphabetical order.

_SMSTSOrgName

Customizes the Task Sequencer engine's display banner.

Property configured by	
------------------------	--

Property applies to	
---------------------	--

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>name</i>	The name that will be used in the Task Sequencer engine's display banner

Example
[Settings] Priority=Default
[Default] _SMSTS0rgName=Woodgrove Bank

ADDSLogPath

Fully qualified, non-UNC directory on a hard disk on the local computer to host the AD DS log files. If the directory exists it must be empty. If it does not exist, it will be created.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>log_path</i>	Fully qualified, non-UNC directory on a hard disk on the local computer to host the AD DS log files

Example
[Settings] Priority=Default
[Default] ADDSLogPath=%DestinationLogicalDrive%\Windows\NTDS

ADDSPassword

Account credentials that can be used when promoting the server to a domain controller.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI	●		
Value	Description				
<i>password</i>	Account credentials that can be used for the promotion operation				
Example					
<pre>[Settings] Priority=Default [Default] ADDSUserName=Administrator ADDSUserDomain=WoodGroveBank ADDSPassword=<complex_password></pre>					

ADDSUserDomain

This is the domain the account specified by **ADDSUserName** should be taken from. If the operation is to create a new forest or to become a member server from a backup domain controller upgrade there is no default. If the operation is to create a new tree, the default is the DNS name of the forest the computer is currently joined to. If the operation is to create a new child domain or a replica then the default is the DNS name of the domain the computer is joined to. If the operation is to demote the computer and the computer is a domain controller in a child domain, the default is the DNS name of the parent domains. If the operation is to demote the computer, and the computer is a domain controller of a tree root domain, the default is the DNS name of the forest.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI	●		
Value	Description				
<i>domain</i>	Domain the UserName account should be taken from				
Example					
<pre>[Settings] Priority=Default</pre>					

Example

```
[Default]
ADDSUserName=Administrator
ADDSUserDomain=WoodGroveBank
ADDSPassword=<complex_password>
```

ADDSUserName

Account credentials that will be used when promoting the server to a domain controller.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>user_name</i>	Account credentials that will be used for the promotion operation

Example

```
[Settings]
Priority=Default

[Default]
ADDSUserName=Administrator
ADDSUserDomain=WoodGroveBank
ADDSPassword=complex_password
```

Administrators

A list of user accounts and domain groups that will be added to the local Administrator group on the target computer. The **Administrators** property is a list of text values that can be any non-blank value. The **Administrators** property has a numeric suffix (for example, **Administrators001** or **Administrators002**).

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>name</i>	The name of a user or group that is to be added to the local Administrator group

Example
[Settings]
Priority=Default
[Default]
Administrators001=WOODGROVEBANK\NYC Help Desk Staff
Administrators002=WOODGROVEBANK\North America East Help Desk Staff
PowerUsers001=WOODGROVEBANK\User01
PowerUsers002=WOODGROVEBANK\User02

AdminPassword

Defines the password that will be assigned to the local Administrator user account on the target computer. If not specified, the pre-deployment password of the Administrator user account will be used.

Property configured by	Property applies to
Bootstrap.ini	●
CustomSettings.ini	●
MDT DB	●

Value	Description
<i>admin_password</i>	The password that is to be assigned to the Administrator user account on the target computer

Example
[Settings]
Priority=Default
[Default]
Administrators001=WOODGROVEBANK\NYC Help Desk Staff
AdminPassword=<admin_password>

Applications

A list of application GUIDs that should be installed on the target computer. These applications are specified on the Applications node in Deployment Workbench. These GUIDs are stored in the Applications.xml file. The **Applications** property

is a list of text values that can be any non-blank value. The **Applications** property has a numeric suffix (for example, **Applications001** or **Applications002**).

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>application_guid</i>	The GUID is specified by Deployment Workbench for the application to be deployed to the target computer. The GUID corresponds to the application GUID stored in the Applications.xml file.

Example
[Settings]
Priority=Default
[Default]
Applications001={ 1D7DF331-47B7-472C-87B3-442597EC2F7D}
Applications002={ 9d2b8999-5e4d-4f3d-bb05-edaaf4fe5628}

ApplicationSuccessCodes

A space-delimited list of error codes used by the ZTIApplications script that determine the successful installation of applications.

Note This property is only applicable to the **Install Application** task sequence step type and when **Install multiple applications** is selected.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>error_codes</i>	The error codes that determine when applications have been successfully installed. Default values are 0 and 3010 .

Example
[Settings]

Example

```
Pri ority=Default
```

```
[Default]
```

```
Appl i cati onSuccessCodes=0 3010
```

ApplyGPOPack

This property is used to determine whether the **Apply Local GPO Package** task sequence step is performed.

Note The default value for this property always performs the **Apply Local GPO Package** task sequence step. You must explicitly provide a value of "NO" to override this behavior..

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The Apply Local GPO Package task sequence step is performed. This is the default value.
NO	The Apply Local GPO Package task sequence step is not performed.

Example

```
[Settings]
```

```
Pri ority=Default
```

```
[Default]
```

```
Appl yGPOPack=NO
```

Architecture

The processor architecture of the processor that is currently running, which is not necessarily the processor architecture supported by the target computer. For example, when running a 32-bit-compatible operating system on a 64-bit processor, **Architecture** will indicate that the processor architecture is 32 bit.

Use the **CapableArchitecture** property to identify the actual processor architecture that the target computer supports.

Note This property is dynamically set by MDT scripts and is not configured in CustomSettings.ini. Treat this property as read only. However, you can use this property within CustomSettings.ini, as shown in the following examples, to aid in defining the configuration of the target computer.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini					
MDT DB		ZTI	●		
Value	Description				
x86	Processor architecture is 32 bit.				
x64	Processor architecture is 64 bit.				
Example					
None					

AreaCode

The area code to be configured for the operating system on the target computer. This property allows only numeric characters. This value is inserted into the appropriate configuration settings in Unattend.xml.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI	●		
Value	Description				
area_code	The area code where the target computer is to be deployed				
Example					
[Settings] Priority=Default					
[Default] AreaCode=206 CountryCode=001 Dialing=TONE LongDistanceAccess=9					

AssetTag

The asset tag number associated with the target computer. The format for asset tag numbers is undefined. Use this property to create a subsection that contains settings targeted to a specific computer.

Note This property is dynamically set by MDT scripts and cannot have its value set in CustomSettings.ini or the MDT DB. Treat this property as read only. However, you can use this property within CustomSettings.ini or the MDT DB, as shown in the following examples, to aid in defining the configuration of the target computer.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
asset_tag	The format of the asset tag is undefined and is determined by the asset tag standard of each organization.

Example 1

```
[Settings]
Priority=Default

[Default]
OSDComputerName=HP-%AssetTag%
```

Example 2

```
[Settings]
Priority=AssetTag, Default

[Default]
OSInstall=YES

[0034034931]
OSDComputerName=HPD530-1

[0034003233]
OSDNEWMACHINENAME=BVMXP
```

AutoConfigDNS

Specifies whether the Active Directory Installation Wizard configures DNS for the new domain if it detects that the DNS dynamic update protocol is not available.

Caution This property value must be specified in uppercase so that the deployment scripts can properly read it.

Property configured by	
------------------------	--

Property applies to	
---------------------	--

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
YES	Configures DNS for the new domain if the DNS dynamic update protocol is not available
NO	Does not configure DNS for the domain

Example
[Settings] Priority=Default
[Default] AutoConfigureDNS=YES

BackupDir

The folder in which backups of the target computer are stored. This folder exists beneath the UNC path specified in the **BackupShare** property. If the folder does not already exist, it will be created automatically.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
Folder	The name of the folder that exists beneath the shared folder specified in the BackupShare property

Example
[Settings] Priority=Default
[Default] DoCapture=YES BackupShare=\NYC- AM- FIL- 01\Backup\$ BackupDir=%OSDComputerName% BackupDrive=C:

BackupDrive

The drive to include in the backup of the target computer. This property defaults to the drive that contains disk 0 partition 1. It can be also set to **ALL**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>backup_drive</i>	The drive letter of the drive to back up
ALL	Back up all drives on the target computer

Example

```
[Settings]
Priority=Default

[Default]
DoCapture=YES
BackupShare=\NYC-AM-FIL-01\Backup$ 
BackupDir=%OSDComputerName%
BackupDrive=C:
```

BackupFile

Specifies the WIM file that will be used by the ZTIBackup.wsf script. For more information about what script uses this property, see [ZTIBackup.wsf](#).

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>backup_file</i>	The name of the Windows Imaging Format (WIM) file to be used during back up.

Example

```
[Settings]
Priority=Default
```

Example

```
[Default]
DoCapture=YES
BackupShare=\\NYC-AM-FIL-01\Backup$
BackupDir=%OSDComputerName%
BackupFile=%OSDComputerName%.wim
```

BackupShare

The shared folder in which backups of the target computer are stored.

The credentials used to access this shared folder for:

- LTI are the credentials entered in the Deployment Wizard.
- ZTI are the credentials used by the Configuration Manager Advanced Client Network Access account.

The permissions required on this share are as follows:

- **Domain Computers.** Allow the Create Folders/Append Data permission.
- **Domain Users.** Allow the Create Folders/Append Data permission.
- **Creator Owner.** Allow the Full Control permission.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
UNC_path	The UNC path of the shared folder Note The UNC path specified in this property must exist before deploying the target operating system.

Example

```
[Settings]
Priority=Default

[Default]
DoCapture=YES
BackupShare=\\NYC-AM-FIL-01\Backup$
BackupDir=%OSDComputerName%
BackupDrive=C:
```

BDEAllowAlphaNumericPin

This property configures whether BitLocker PINs contain alphanumeric values.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	Alphanumeric characters are allowed in the PIN. Note In addition to setting this property to YES , the Allow enhanced PINs for startup group policy setting must be enabled.
NO	Only numeric characters are allowed in the PIN.

Example

```
[Settings]
Priority=Default

[Default]
BDEInstallSuppress=NO
BDEAllowAlphaNumericPin=YES
BDEDriveLetter=S:
BDEDriveSize=2000
BDEInstall=TPMKey
BDERecoveryKey=AD
BDEKeyLocation=C:
```

BDEDriveLetter

The drive letter for the partition that is not encrypted by BitLocker, also known as the *System Volume*. SYSVOL is the directory that contains the hardware-specific files needed to load Windows computers after the BIOS has booted the platform.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>drive_letter</i>	The letter designation for the logical drive for the System Volume (such as S or T). The default value is

Value	Description
	S.

Example

```
[Settings]
Priority=Default

[Default]
BDEInstall Suppress=No
BDEDriveLetter=S:
BDEDriveSize=2000
BDEInstall=TPMKey
BDERecoveryKey=AD
BDEKeyLocation=C:
```

BDEDriveSize

The size of the BitLocker system partition. The value is specified in megabytes. In the example, the size of the BitLocker partition to create is almost 2 GB (2,000 MB).

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
drive_size	The size of the partition in megabytes; the default sizes are: <ul style="list-style-type: none"> Windows 7 and Windows Server 2008 R2: 300 MB

Example

```
[Settings]
Priority=Default

[Default]
BDEInstall Suppress=No
BDEDriveLetter=S:
BDEDriveSize=2000
BDEInstall=TPMKey
BDERecoveryKey=AD
BDEKeyLocation=C:
```

BDEInstall

The type of BitLocker installation to be performed. Protect the target computer using one of the following methods:

- A TPM microcontroller
- A TPM and an external startup key (using a key that is typically stored on a USB flash drive [UFD])
- A TPM and PIN
- An external startup key

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
TPM	Protect the computer with TPM only. The TPM is a microcontroller that stores keys, passwords, and digital certificates. The microcontroller is typically an integral part of the computer motherboard.
TPMKey	Protect the computer with TPM and a startup key. Use this option to create a startup key and to save it on a UFD. The startup key must be present in the port each time the computer starts.
TPMPin	Protect the computer with TPM and a pin. Use this option in conjunction with the BDEPin property.
Key	Protect the computer with an external key (the recovery key) that can be stored in a folder, in AD DS, or printed.

Example
<pre>[Settings] Priority=Default [Default] BDEInstall Suppress=NO BDEDriveLetter=S: BDEDriveSize=2000 BDEInstall=TPMKey BDERecoveryKey=AD BDEKeyLocation=C:</pre>

BDEInstallSuppress

Indicates whether the deployment process should skip the BitLocker installation.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	Do not attempt to install BitLocker.
NO	Attempt to install BitLocker.

Example
[Settings] Priority=Default
[Default] BDEInstallSuppress=YES

BDEKeyLocation

The location for storing the BitLocker recovery key and startup key.

Note If this property is configured using the Deployment Wizard, the property must be the drive letter of a removable disk. If the **SkipBitLocker** property is set to **TRUE** so that the **Specify the BitLocker configuration** wizard page is skipped, this property can be set to a UNC path in CustomSettings.ini or in the MDT database (MDT DB).

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>Location</i>	Specifies where the recovery key will be stored; must be a UNC path or the drive letter of a removable disk. If not set, the first available removable drive will be used.

Example

```
[Settings]
Priority=Default

[Default]
BDEInstall Suppress=N0
BDEDriveLetter=S:
BDEDriveSize=2000
BDEInstall=TPMKey
BDERecoveryKey=AD
BDEKeyLocation=C:
```

BDEPin

The PIN to be assigned to the target computer when configuring BitLocker and the **BDEInstall** or **OSDBitLockerMode** properties are set to **TPMPin**.

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●
Value	
<i>Pin</i>	The PIN to be used for BitLocker. The PIN can be between 4 and 20 digits long.

Example

```
[Settings]
Priority=Default

[Default]
BDEInstall Suppress=N0
BDEDriveLetter=S:
BDEDriveSize=2000
BDEInstall=TPMPin
BDEPin=123456789
```

BDERecoveryKey

A Boolean value that indicates whether the process creates a recovery key for BitLocker. The key is used for recovering data encrypted on a BitLocker volume. This key is cryptographically equivalent to a startup key. If available, the recovery

key decrypts the volume master key (VMK), which, in turn, decrypts the full volume encryption key (FVEK).

Note The recovery key is stored in the location specified in the **BDEKeyLocation** property.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI			
Value		Description			
AD		A recovery key is created.			
Not specified		A recovery key is not created.			
Example					
<pre>[Settings] Priority=Default [Default] BDEInstallSuppress=NO BDEDriveLetter=S: BDEDriveSize=2000 BDEInstall=TPMKey BDERecoveryKey=AD BDEKeyLocation=C:</pre>					

BDEWaitForEncryption

Specifies that the deployment process should not proceed until BitLocker has completed the encryption process for all specified drives. Specifying TRUE could dramatically increase the time required to complete the deployment process.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	
Value		Description	
TRUE		Specifies that the deployment process should wait for drive encryption to complete.	

Value	Description
FALSE	Specifies that the deployment process should not wait for drive encryption to complete.

Example

```
[Settings]
Priority=Default

[Default]
BDEInstallSuppress=NO
BDEDriveLetter=S:
BDEDriveSize=2000
OSDBitLockerMode=TPMKey
OSDBitLockerStartupKeyDrive=C:
OSDBitLockerCreateRecoveryPassword=AD
BDEWaitForEncryption=TRUE
```

BitsPerPel

A setting for displaying colors on the target computer. The property can contain numeric digits and corresponds to the color quality setting. In the example, **32** indicates 32 bits per pixel for color quality. This value is inserted into the appropriate configuration settings in Unattend.xml.

Note The default values (in the Unattend.xml template file) are 1,024 pixels horizontal resolution, 768 pixels vertical resolution, 32-bit color depth, and 60 Hertz (Hz) vertical refresh rate.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>bits_per_pixel</i>	The number of bits per pixel to use for color. The default value is the default for the operating system being deployed.

Example

```
[Settings]
Priority=Default

[Default]
BitsPerPel=32
```

Example

```
VRefresh=60
XResolution=1024
YResolution=768
```

BuildID

Identifies the operating system task sequence to be deployed to the target computer. You create the task sequence ID on the Task Sequences node in the Deployment Workbench. The **BuildID** property allows alphanumeric characters, hyphens (-), and underscores (_). The **BuildID** property cannot be blank or contain spaces.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>build_id</i>	Identifier of the operating system task sequence as defined in the Deployment Workbench for the target operating system being deployed Note Make certain to use the TaskSequenceID specified in the Deployment Workbench user interface (UI) and not the GUID of the TaskSequenceID .

Example

```
[Settings]
Priority=Default

[Default]
BuildID=BareMetal
```

CapableArchitecture

The processor architecture of the processor supported by the target computer, not the current processor architecture that is running. For example, when running a 32-bit-compatible operating system on a 64-bit processor, **CapableArchitecture** will indicate that the processor architecture is 64 bit.

Use the **Architecture** property to see the processor architecture that is currently running.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●
Value	Description
x86	Processor architecture is 32 bit.
x64	Processor architecture is 64 bit.
Example	
None	

CaptureGroups

Controls whether the group membership of local groups on the target computer is captured. This group membership is captured during the State Capture Phase and is restored during the State Restore Phase.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
NO	Captures no group membership information.
ALL	Captures the membership of all local groups on the target computer.
YES	Captures the membership of the Administrator and Power Users built-in groups and the groups listed in the groups' properties. This is the default value if some other value is specified. (YES is the typical value.)
Example	
<pre>[Settings] Priority=Default</pre>	
<pre>[Default] DeployRoot=\NYC-AM-FIL-01\Distribution\$</pre>	

Example

```
ResourceRoot=\NYC- AM- FIL- 01\Resource$  
UDShare=\NYC- AM- FIL- 01\MyData$  
CaptureGroups=YES  
Groups1=NYC Application Management  
Groups2=NYC Help Desk Users
```

ChildName

Specifies whether to append the DNS label at the beginning of the name of an existing directory service domain when installing a child domain.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>name</i>	The name of the child domain

Example

```
[Settings]  
Priority=Default  
  
[Default]  
ChildName=childdom.parentdom.WoodGroveBank.com
```

ComputerBackupLocation

The network shared folder where the computer backup is stored. If the target folder does not already exist, it is automatically created.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>blank</i>	Same as AUTO .

Value	Description
<i>UNC_path</i>	The UNC path to the network shared folder where the backup is stored.
AUTO	Creates a backup on a local hard disk if space is available. Otherwise, the backup is saved to a network location specified in the BackupShare and BackupDir properties.
NETWORK	Creates a backup on a network location specified in BackupShare and BackupDir .
NONE	No backup will be performed.

Example

```
[Settings]
Priority=Default

[Default]
DeployRoot=\NYC-AM-FIL-01\Distribution$ 
ResourceRoot=\NYC-AM-FIL-01\Resource$ 
UDShare=\NYC-AM-FIL-01\Media$ 
ComputerBackupLocation=NETWORK 
BackupShare=\NYC-AM-FIL-01\Backup$ 
BackupDir=%OSDComputerName% 
UDDir=%OSDComputerName% 
SLShare=\NYC-AM-FIL-01\Logs$ 
UDProfiles=Administrator, User-01, ExtranetUser 
UserDataLocation=NONE
```

ComputerName

This property has been deprecated. Use **OSDComputerName** instead.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	
MDT DB	ZTI
Value	Description
None	None

Value	Description
None	None

Example
None

ConfigFileName

Specifies the name of the configuration file used during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	
MDT DB	ZTI
Value	Description
<i>file_name</i>	Specifies the name of the configuration file used during OEM deployments
Example	
None	

ConfigFilePackage

Specifies the package ID for the configuration package used during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	
MDT DB	ZTI
Value	Description
<i>package</i>	Specifies the package ID for the configuration package used during OEM deployments
Example	
None	

ConfirmGC

Specifies whether the replica is also a global catalog.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●
Value	Description		
YES	Makes the replica a global catalog if the backup was a global catalog.		
NO	Does not make the replica a global catalog.		
Example			
<pre>[Settings] Priority=Default</pre> <pre>[Default] ConfirmGC=YES</pre>			

CountryCode

The country code to be configured for the operating system on the target computer. This property allows only numeric characters. This value is inserted into the appropriate configuration settings in Unattend.xml.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●
Value	Description		
country_code	The country code where the target computer is to be deployed		
Example			
<pre>[Settings] Priority=Default</pre> <pre>[Default] AreaCode=206 CountryCode=001 Dialing=TONE LongDistanceAccess=9</pre>			

CriticalReplicationOnly

Specifies whether the promotion operation performs only critical replication and then continues, skipping the noncritical (and potentially lengthy) portion of replication.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
YES	Skips noncritical replication
NO	Does not skip noncritical replication

Example
[Settings] Priority=Default
[Default] CriticalReplicationOnly=YES

CustomDriverSelectionProfile

Specifies the custom selection profile used during driver installation.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	

Value	Description
profile	Custom selection profile used during driver installation

Example
[Settings] Priority=Default
[Default]

Example

CustomDriverSelectionProfile=CustomDrivers

CustomPackageSelectionProfile

Specifies the custom selection profile used during package installation.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>profile</i>	Custom selection profile used during package installation

Example

[Settings]
Priority=Default

[Default]
CustomPackageSelectionProfile=CustomPackages

CustomWizardSelectionProfile

Specifies the custom selection profile used by the wizard for filtering the display of various items.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>profile</i>	Custom selection profile by the wizard for filtering the display of various items

Example

[Settings]
Priority=Default

[Default]

Example

CustomWizardSelectionProfile=CustomWizard

Database

The property that specifies the database to be used for querying property values from columns in the table specified in the **Table** property. The database resides on the computer specified in the **SQLServer** property. The instance of Microsoft SQL Server® on the computer is specified in the **Instance** property.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
database	The name of the database to be used for querying property values

Example

```
[Settings]
Priority=Computers, Default
```

```
[Default]
OSInstall=YES
```

```
[Computers]
SQLServer=NYC-SQL-01
SQLShare=SQL$
Database=MDTDB
Instance=SQLEnterprise2005
Table=Computers
Parameters=Serial Number, AssetTag
ParameterCondition=OR
```

DatabasePath

Specifies the fully qualified, non-UNC path to a directory on a fixed disk of the target computer that contains the domain database.

Property configured by	
BootStrap.ini	

Property applies to	
LTI	●

Property configured by		Property applies to	
CustomSettings.ini	●		
MDT DB	●	ZTI	●
Value	Description		
<i>path</i>	Specifies the fully qualified, non-UNC path to a directory on a fixed disk of the local computer that contains the domain database		
Example			
<pre>[Settings] Priority=Default</pre> <pre>[Default] DatabasePath=%DestinationLogicalDrive%\Windows\NTSD</pre>			

DBID

Specifies the user account used to connect to the computer running SQL Server (specified by the **SQLServer** property) using SQL Server authentication. The **DBPwd** property provides the password for the user account in the **DBID** property.

Note SQL Server authentication is not as secure as Integrated Windows authentication. Integrated Windows authentication is the recommended authentication method. Using the **DBID** and **DBPwd** properties stores the credentials in clear text in the CustomSettings.ini file and therefore is not secure. For more information about using Integrated Windows authentication, see the [SQLShare](#) property.

Note This property is configurable only by manually editing the CustomSettings.ini and BootStrap.ini files.

Property configured by		Property applies to	
BootStrap.ini	●	LTI	●
CustomSettings.ini	●		
MDT DB		ZTI	●
Value	Description		
<i>user_id</i>	The name of the user account credentials used to access the computer running SQL Server using SQL Server authentication		
Example			
<pre>[Settings] Priority=Computers, Default</pre>			

Example

```
[Default]
OSInstall=YES

[Computers]
SQLServer=NYC-SQL-01
DBID=SQL_User-01
DBPwd=<complex_password>
NetList=DBNMPNTW
Database=MDTDB
Instance=SQLEnterprise2005
Table=Computers
Parameters=Serial Number, AssetTag
ParameterCondition=OR
```

DBPwd

Specifies the password for the user account specified in the **DBID** property. The **DBID** and **DBPwd** properties provide the credentials for performing SQL Server authentication to the computer running SQL Server (specified by the **SQLServer** property).

Note SQL Server authentication is not as secure as Integrated Windows authentication. Integrated Windows authentication is the recommended authentication method. Using the **DBID** and **DBPwd** properties stores the credentials in clear text in the CustomSettings.ini file and therefore is not secure. For more information about using Integrated Windows authentication, see the [SQLShare](#) property.

Note This property is configurable only by manually editing the CustomSettings.ini and BootStrap.ini files.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>user_password</i>	The password for the user account credentials specified in the DBID property for using SQL Server authentication

Example

```
[Settings]
Priority=Computers, Default
```

Example

```
[Default]
OSInstall=YES

[Computers]
SQLServer=NYC-SQL-01
DBID=SQL_User-01
DBPwd=<complex_password>
NetLab=DBNMPNTW
Database=MDTDB
Instance=SQLEnterprise2005
Table=Computers
Parameters=Serial Number, AssetTag
ParameterCondition=OR
```

Debug

Controls the verbosity of messages written to the MDT log files. This property can be configured to help assist in troubleshooting deployments by providing extended information about the MDT deployment process.

You can set this property by starting the LiteTouch.vbs script with the **/debug:true** command-line parameter as follows:

```
cscript.exe LiteTouch.vbs /debug:true
```

After the LiteTouch.vbs script is started, the **Debug** property's value is set to **TRUE**, and all other scripts are automatically read the value of this property and provide verbose information.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or in the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	
ZTI	●

Value	Description
TRUE	Debug logging is enabled, which includes the following: <ul style="list-style-type: none"> Verbose messages are logged. Deprecated messages are logged as errors.

Value	Description
FALSE	Debug logging is not enabled. This is the default value.

Example
None

DefaultGateway

The IP address of the default gateway being used by the target computer. The format of the IP address returned by the property is standard dotted-decimal notation; for example, 192.168.1.1. Use this property to create a subsection that contains settings targeted to a group of computers based on the IP subnets on which they are located.

Note This property is dynamically set by MDT scripts and cannot have its value set in CustomSettings.ini or the MDT DB. Treat this property as read only. However, you can use this property within CustomSettings.ini or the MDT DB, as shown in the following examples, to aid in defining the configuration of the target computer.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>default_gateway</i>	The IP address of the default gateway in standard dotted-decimal notation

Example
<pre>[Settings] Priority=DefaultGateway, Default [Default] OSInstall=YES [DefaultGateway] 192.168.0.1=HOUSTON 11.1.1.11=REDMOND 172.28.20.1=REDMOND [REDMOND] Packages001=XXX00004: Program4 Packages002=XXX00005: Program5</pre>

Example

```
[HOUSTON]
Packages001=XXX00006: Program6
Packages002=XXX00007: Program7
Packages003=XXX00008: Program8
```

DeployDrive

The value used by the scripts to access files and run programs in the deployment share that the Deployment Workbench creates. The property returns the drive letter mapped to the **DeployRoot** property. ZTIAplications.wsf uses the **DeployDrive** property when running any command-line programs with a .cmd or .bat extension.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>drive_letter</i>	The letter designation for the logical drive where the target operating system is to be installed (such as C or D)

Example

None

DeploymentMethod

The method being used for the deployment (UNC, media, or Configuration Manager).

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	

Property applies to	
LTI	●
ZTI	

Property configured by		Property applies to	
MDT DB		ZTI	●
Value	Description		
UNC	The deployment is made to the target computer over the network.		
Media	The deployment is made from local media (such as DVD or hard disk) at the target computer.		
SCCM	ZTI uses this method for Configuration Manager.		
Example			
None			

DeploymentType

The type of deployment being performed based on the deployment scenario. For ZTI, this property is set dynamically by MDT scripts and is not configured in CustomSettings.ini. For LTI, you can bypass the page in the Deployment Wizard on which the deployment type is selected. In addition, you can specify the deployment type by passing one of the values listed below to the LiteTouch.wsf script as a command-line option.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB		ZTI	●
Value	Description		
NEWCOMPUTER	The target computer is a new computer that has never been a member of the network.		
REFRESH	The target computer is an existing computer on the network that needs the desktop environment standard to be redeployed.		
REPLACE	An existing computer on the network is being replaced with a new computer. The user state migration data is transferred from the existing computer to a new computer.		
Example			
[Settings]			

Example

```
Priority=Default
[Default]
DeploymentType=NEWCOMPUTER
```

DeployRoot

Specifies the UNC or local path to the folder that is the root of the folder structure that MDT uses. This folder structure contains configuration files, scripts, and other folders and files that MDT uses. The value of this property is set based on the following MDT deployment technologies:

- **LTI.** This property is the UNC path to the deployment share that the Deployment Workbench creates. Use this property to select a specific deployment share. The most common use of this property is in the BootStrap.ini file to identify a deployment share before the connection to the deployment share is established. All other deployment share folders are relative to this property (such as device drivers, language packs, or operating systems).
- **ZTI.** This property is the local path to the folder to which the MDT files package is copied. The **Use Toolkit Package** task sequence step copies the MDT files package to a local folder on the target computer, and then automatically sets this property to the local folder.

Note For ZTI, this property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or in the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
path	The UNC or local path to the .

Example

```
[Settings]
Priority=Default
[Default]
DeployRoot=\NYC-AM-FIL-01\Distribution$ 
UserDataLocation=NONE
```

DestinationDisk

Disk number that the image will be deployed to.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
<i>disk_number</i>	The number of the disk to which the image will be deployed

Example
[Settings] Priority=Default
[Default] DestinationDisk=0

DestinationLogicalDrive

The logical drive to which the image will be deployed.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
<i>logical_drive_number</i>	The logical drive to which the image will be deployed

Example 1
[Settings] Priority=Default
[Default] DestinationLogicalDrive=0

Example 2
[Settings] Priority=Default

Example 2

```
[Default]
InstallDNS=YES
DomainNetBIOSName=WoodGroveBank
NewDomain=Child
DomainLevel=3
ForestLevel=3
NewDomainDNSName=newdom.WoodGroveBank.com
ParentDomainDNSName=WoodGroveBank.com
AutoConfigDNS=YES
ConfigRmGC=YES
CriticalRepliCatIonOnly=NO
ADDSUserName=Administrator
ADDSUserDomain=WoodGroveBank
ADDSPassword=<complex_password>
DatabasePath=%DestinationLogicalDrive%\Windows\NTDS
ADDSLogPath=%DestinationLogicalDrive%\Windows\NTDS
SysVol Path=%DestinationLogicalDrive%\Windows\SYSVOL
SafeModeAdminPassword=<complex_password>
```

DestinationPartition

Disk partition to which the image will be deployed.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
<i>partition_number</i>	The number of the partition to which the image will be deployed

Example

```
[Settings]
Priority=Default

[Default]
DestinationPartition=1
```

DHCPScopes

Specifies the number of DHCP scopes to configure.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
scopes	Specifies the number of DHCP scopes to configure

Example

```
[Settings]
Priority=Default

[Default]
DHCPScopes=1
```

DHCPScopesxDescription

The description of the DHCP scope.

Note The x in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
description	The description of the DHCP scope

Example

```
[Settings]
Priority=Default

[Default]
DHCPScopes0Description=DHCPScope0
```

DHCPScopesxEndIP

Specifies the ending IP address for the DHCP scope.

Note The x in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
end_IP	Specifies the ending IP address for the DHCP scope

Example
[Settings]
Priority=Default
[Default]
DHCPScopes0EndIP=192.168.0.30

DHCPScopesxExcludeEndIP

Specifies the ending IP address for the DHCP scope exclusion. IP addresses that are excluded from the scope are not offered by the DHCP server to clients obtaining leases from this scope.

Note The x in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
exclude_end_IP	Specifies the ending IP address for the DHCP scope exclusion

Example
[Settings]
Priority=Default

Example

[Default]

DHCPScopes0ExcludeEndIP=192.168.0.15

DHCPScopesxExcludeStartIP

Specifies the starting IP address for the DHCP scope exclusion. IP addresses that are excluded from the scope are not offered by the DHCP server to clients obtaining leases from this scope.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>exclude_start_IP</i>	Specifies the starting IP address for the DHCP scope exclusion

Example

[Settings]

Priority=Default

[Default]

DHCPScopes0ExcludeStartIP=192.168.0.10

DHCPScopesxIP

Specifies the IP subnet of the scope.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>IP</i>	Specifies the IP subnet of the scope

Example

[Settings]
Priority=Default
[Default]
DHCPScopes0IP=192.168.0.0

DHCPScopesxName

A user-definable name to be assigned to the scope.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>name</i>	A user-definable name to be assigned to the scope

Example

[Settings]
Priority=Default
[Default]
DHCPScopes0Name=DHCPScope0

DHCPScopesxOptionDNSDomainName

Specifies the domain name that the DHCP client should use when resolving unqualified domain names with the DNS.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description

Value	Description
DNS_domain_name	Specifies the domain name that the DHCP client should use when resolving unqualified domain names with the DNS

Example
[Settings] Priority=Default
[Default] DHCPScopes00ptionDNSDomainName=WoodGroveBank.com

DHCPScopesxOptionDNSServer

Specifies a list of IP addresses for DNS name servers available to the client. When more than one server is assigned, the client interprets and uses the addresses in the specified order.

Note The x in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	●

Value	Description
DNS_server	Specifies a list of IP addresses for DNS name servers available to the client

Example
[Settings] Priority=Default
[Default] DHCPScopes00ptionDNSServer=192.168.0.2

DHCPScopesxOptionLease

The duration that the DHCP lease is valid for the client.

Note The x in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	Property applies to

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
lease	The duration that the DHCP lease is valid for the client

Example

```
[Settings]
Priority=Default

[Default]
DHCPScopes00ptionLease=7
```

DHCPScopesxOptionNBTNodeType

Specifies the client node type for NetBT clients.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
1	Configures the node type as b-node
2	Configures the node type as p-node
4	Configures the node type as m-node
8	Configures the node type as h-node

Example

```
[Settings]
Priority=Default

[Default]
DHCPScopes00ptionNBTNodeType=4
```

DHCPScopesxOptionPXEClient

Specifies the IP address used for PXE client bootstrap code.

Note The x in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
PXE_client	Specifies the IP address used for PXE client bootstrap code

Example
[Settings] Priority=Default
[Default] DHCPScopes00ptionPXEClient=192.168.0.252

DHCPScopesxOptionRouter

Specifies a list of IP addresses for routers on the client subnet. When more than one router is assigned, the client interprets and uses the addresses in the specified order. This option is normally used to assign a default gateway to DHCP clients on a subnet.

Note The x in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
router	Specifies a list of IP addresses for routers on the client subnet

Example
[Settings] Priority=Default

Example

[Default]

DHCPScopes0ptionRouter=192. 168. 0. 253

DHCPScopesxOptionWINSServer

Specifies the IP addresses to be used for NBNSes on the network.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
WINS_server	Specifies the IP addresses to be used for NBNSes on the network

Example

[Settings]

Pri ority=Default

[Default]

DHCPScopes0ptionWINSServer=192. 168. 0. 2

DHCPScopesxStartIP

The starting IP address for the range of IP addresses that are to be included in the scope.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
start_IP	The starting IP address for the range of IP addresses

Value	Description
	that are to be excluded from the scope

Example
[Settings] Priority=Default
[Default] DHCPSscopes0StartIP=192. 168. 0. 20

DHCPSscopesxSubnetMask

Specifies the subnet mask of the client subnet.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DHCP configurations.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Value	Description
<i>subnet_mask</i>	Specifies the subnet mask of the client IP subnet

Example
[Settings] Priority=Default
[Default] DHCPSscopes0SubnetMask=255. 255. 255. 0

DHCPServerOptionDNSDomainName

Specifies the connection-specific DNS domain suffix of client computers.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Value	Description

Value	Description
DNS_domain_name	Specifies the connection-specific DNS domain suffix of client computers

Example
[Settings] Priority=Default
[Default] DHCPServerOptionDNSDomainName=Fabrikam.com

DHCPServerOptionDNSServer

Specifies a list of IP addresses to be used as DNS name servers that are available to the client.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	● ZTI

Value	Description
DNS_server	Specifies a list of IP addresses to be used as DNS name servers that are available to the client

Example
[Settings] Priority=Default
[Default] DHCPServerOptionDNSServer=192.168.0.1, 192.168.0.2

DHCPServerOptionNBNodeType

Specifies the client node type for NetBT clients.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	● ZTI

Value	Description

Value	Description
1	Configures the node type as b-node
2	Configures the node type as p-node
4	Configures the node type as m-node
8	Configures the node type as h-node

Example
[Settings] Priority=Default
[Default] DHCPServerOptionNBTNodeType=4

DHCPServerOptionPXEClient

Specifies the IP address used for PXE client bootstrap code.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Value	Description
<i>PXE_client</i>	Specifies the IP address used for PXE client bootstrap code

Example
[Settings] Priority=Default
[Default] DHCPServerOptionPXEClient=192.168.0.252

DHCPServerOptionRouter

Specifies a list of IP addresses for routers on the client subnet. When more than one router is assigned, the client interprets and uses the addresses in the specified order. This option is normally used to assign a default gateway to DHCP clients on a subnet.

Property configured by	Property applies to

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>router</i>	Specifies a list of IP addresses for routers on the client subnet

Example
[Settings] Priority=Default
[Default] DHCPServerOptionRouter=192.168.0.253

DHCPServerOptionWINSServer

Specifies the IP addresses to be used for NBNSes on the network.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>WINS_server</i>	Specifies the IP addresses to be used for NBNSes on the network

Example
[Settings] Priority=Default
[Default] DHCPServerOptionWINSServer=192.168.0.2

Dialing

The type of dialing supported by the telephony infrastructure where the target computer is located. This value is inserted into the appropriate configuration settings in Unattend.xml.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
PULSE	The telephony infrastructure supports pulse dialing.
TONE	The telephony infrastructure supports touch-tone dialing.

Example
[Settings]
Priority=Default
[Default]
AreaCode=206
CountryCode=001
Dialing=TONE
LongDistanceAccess=9

DisableTaskMgr

This property controls a user's ability to start Task Manager by pressing CTRL+ALT+DEL. After the user starts Task Manager, he or she could interrupt the LTI task sequence while running in the new operating system on the target computer. This property is used in conjunction with the **HideShell** property and is only valid when the **HideShell** property is set to YES.

Note This property and the **HideShell** property must both be set to YES to prevent the user pressing CTRL+ALT+DEL and interrupting the LTI task sequence.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	Prevent the user from being able to start Task Manager by pressing CTRL+ALT+DEL and subsequently interrupting the LTI task sequence.

Value	Description
NO	Allow the user to start Task Manager by pressing CTRL+ALT+DEL and subsequently interrupt the LTI task sequence. This is the default value.

Example
[Settings] Priority=Default
[Default] DisableTaskMgr=YES HideShell=YES

DNSServerOptionBINDSecondaries

Determines whether to use fast transfer format for transfer of a zone to DNS servers running legacy BIND implementations.

By default, all Windows-based DNS servers use a fast zone transfer format. This format uses compression, and it can include multiple records per TCP message during a connected transfer. This format is also compatible with more recent BIND-based DNS servers that run version 4.9.4 and later.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Value	Description
TRUE	Allows BIND secondaries
FALSE	Does not allow to BIND secondaries

Example
[Settings] Priority=Default
[Default] DNSServerOptionBINDSecondaries=TRUE

DNSServerOptionDisableRecursion

Determines whether or not the DNS server uses recursion. By default, the DNS Server service is enabled to use recursion.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
TRUE	Disables recursion on the DNS server
FALSE	Enables recursion on the DNS server
Example	
<pre>[Settings] Priority=Default</pre> <pre>[Default] DNSServerOptionDisableRecursion=TRUE</pre>	

DNSServerOptionEnableNetmaskOrdering

Determines whether the DNS server reorders address (A) resource records within the same resource record that is set in the server's response to a query based on the IP address of the source of the query.

By default, the DNS Server service uses local subnet priority to reorder A resource records.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
TRUE	Enables netmask ordering
FALSE	Disables netmask ordering

Example

[Settings]
Priority=Default
[Default]
DNSServerOptionEnableNetmaskOrdering=TRUE

DNSServerOptionEnableRoundRobin

Determines whether the DNS server uses the round robin mechanism to rotate and reorder a list of resource records if multiple resource records exist of the same type that exist for a query answer.

By default, the DNS Server service uses round robin.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	Enables round robin
FALSE	Disables round robin

Example

[Settings]
Priority=Default
[Default]
DNSServerOptionEnableRoundRobin=TRUE

DNSServerOptionEnableSecureCache

Determines whether the DNS server attempts to clean up responses to avoid cache pollution. This setting is enabled by default. By default, DNS servers use a secure response option that eliminates adding unrelated resource records that are included in a referral answer to their cache. In most cases, any names that are added in referral answers are typically cached, and they help expedite the resolution of subsequent DNS queries.

With this feature, however, the server can determine that referred names are potentially polluting or insecure and then discard them. The server determines

whether to cache the name that is offered in a referral on the basis of whether it is part of the exact, related, DNS domain name tree for which the original queried name was made.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	Enables cache security
FALSE	Disables cache security

Example
[Settings] Priority=Default
[Default] DNSServerOptionEnableSecureCache=TRUE

DNSServerOptionFailOnLoad

Specifies that loading of a zone should fail when bad data is found.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	Enable fail on load
FALSE	Disable fail on load

Example
[Settings] Priority=Default

Example

[Default]

DNSServerOpti onFailOnLoad=TRUE

DNSServerOptionNameCheckFlag

Specifies which character standard is used when checking DNS names.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
0	Uses ANSI characters that comply with Internet Engineering Task Force (IETF) Request for Comments (RFCs). This value corresponds to the Strict RFC (ANSI) selection when configuring DNS in the Deployment Workbench.
1	Uses ANSI characters that do not necessarily comply with IETF RFCs. This value corresponds to the Non RFC (ANSI) selection when configuring DNS in the Deployment Workbench.
2	Uses multibyte UCS Transformation Format 8 (UTF-8) characters. This is the default setting. This value corresponds to the Multibyte (UTF-8) selection when configuring DNS in the Deployment Workbench.
3	Uses all characters. This value corresponds to the All names selection when configuring DNS in the Deployment Workbench.

Example

[Settings]

Priority=Default

[Default]

DNSServerOpti onNameCheckFl ag=2

DNSZones

Specifies the number of DNS zones to configure.

Property configured by	
------------------------	--

Property applies to	
---------------------	--

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
zones	Specifies the number of DNS zones to configure

Example

```
[Settings]
Priority=Default

[Default]
DNSZones=1
DNSZones0Name=MyNewZone
DNSZones0DirectoryPartition=Forest
DNSZones0FileName=MyNewZone.dns
DNSZones0MasterIP=192.168.0.1,192.168.0.2
DNSZones0Type=Secondary
```

DNSZonesxDirectoryPartition

Specifies the directory partition on which to store the zone when configuring secondary or stub zones.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DNS configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
Domain	Replicates zone data to all DNS server in the AD DS domain
Forest	Replicates zone data to all DNS server in the AD DS forest
Legacy	Replicates zone data to all domain controllers in the AD DS domain

Example

Example

[Settings]
Priority=Default
[Default]
DNSZones0DirectoryPartition=Forest

DNSZonesxFileName

Specifies the name of the file that will store the zone information.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DNS configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>file_name</i>	Specifies the name of the file that will store the zone information

Example

[Settings]
Priority=Default
[Default]
DNSZones0FileName=MyNewZone.dns

DNSZonesxMasterIP

A comma delimited list of IP addresses of the master servers to be used by the DNS server when updating the specified secondary zones. This property must be specified when configuring a secondary or stub DNS zone.

Note The *x* in this properties name is a placeholder for a zero-based array that contains DNS configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>IP1,IP2</i>	A comma-delimited list of IP addresses of the master servers

Example
[Settings] Priority=Default
[Default] DNSZones0MasterIP=192.168.0.1,192.168.0.2

DNSZonesxName

Specifies the name of the zone.

Note The x in this properties name is a placeholder for a zero-based array that contains DNS configurations.

Property configured by
BootStrap.ini
CustomSettings.ini
MDT DB

Property applies to
LTI
ZTI
ZTI

Value	Description
<i>name</i>	Specifies the name of the zone

Example
[Settings] Priority=Default
[Default] DNSZones0Name=MyNewZone

DNSZonesxScavenge

Configures the Primary DNS server to "scavenge" stale records—that is, to search the database for records that have aged and delete them.

Note The x in this properties name is a placeholder for a zero-based array that contains DNS configurations.

Property configured by
BootStrap.ini
CustomSettings.ini

Property applies to
LTI
ZTI

Property configured by		Property applies to	
MDT DB	●	ZTI	●

Value	Description
TRUE	Allow stale DNS records to be scavenged.
FALSE	Do not allow stale DNS records to be scavenged.

Example
[Settings] Priority=Default
[Default] DNSZones0Scavenge=TRUE

DNSZonesxType

Specifies the type of zone to create.

Note The x in this properties name is a placeholder for a zero-based array that contains DNS configurations.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
DSPrimary	Creates a primary zone and specifying that it should be stored in AD DS on a DNS server configured as a domain controller
DSSub	Creates a stub zone and specifying that it should be stored in AD DS on a DNS server configured as a domain controller
Primary	Creates a primary zone
Secondary	Creates a secondary zone
Stub	Creates a stub zone

Example
[Settings] Priority=Default
[Default]

Example

DNSZones0Type=Secondary

DNSZonesxUpdate

Configures the Primary DNS server to perform dynamic updates.

Note The x in this properties name is a placeholder for a zero-based array that contains DNS configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
0	Does not allow dynamic updates
1	Allows dynamic updates
2	Allows secure dynamic updates

Example

```
[Settings]
Priority=Default

[Default]
DNSZones0Update=1
```

DoCapture

Indicator of whether an image of the target computer is to be captured. If it is, Sysprep is run on the target computer to prepare for image creation. After Sysprep has run, a new WIM image is created and stored in the folder within the shared folder designated for target computer backups (**BackupDir** and **BackupShare**, respectively).

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
YES	Copy the necessary files to run Sysprep on the target computer, run Sysprep on the target computer, and capture a WIM image.
NO	Do not run Sysprep on the target computer, and do not capture a WIM image.
PREPARE	Copy the necessary files to run Sysprep on the target computer, but do not run Sysprep or other image-capture processes.
SYSPREP	Copy the necessary files to run Sysprep on the target computer, run Sysprep on the target computer, but do not capture a WIM image. Note The primary purpose of this value is to allow the creation of a VHD that contains an operating system after Sysprep has been run and no image capture is necessary.

Example
<pre>[Settings] Priority=Default [Default] DoCapture=YES DeployRoot=\NYC-AM-FIL-01\Distribution\\$ ResourceRoot=\NYC-AM-FIL-01\Resourse\\$ UDShare=\NYC-AM-FIL-01\MigData\\$ UDDir=%0SDComputerName%</pre>

DomainAdmin

The user account credentials used to join the target computer to the domain specified in **JoinDomain**. Specify as *UserName*.

Note For ZTI, the credentials that Configuration Manager specifies typically are used. If the **DomainAdmin** property is specified, the credentials in the **DomainAdmin** property override the credentials that Configuration Manager specifies.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
<i>domain_admin</i>	The name of the user account credentials

Example

```
[Settings]
Priority=Default

[Default]
DomainAdmin=NYCAdmin
DomainAdminDomain=WOODGROVEBANK
DomainAdminPassword=<complex_password>
```

DomainAdminDomain

The domain in which the user's credentials specified in **DomainAdmin** reside.

Note For ZTI, the credentials that Configuration Manager specifies typically are used. If the **DomainAdmin** property is specified, the credentials in the **DomainAdmin** property override the credentials that Configuration Manager specifies.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>domain_admin_domain</i>	The name of the domain where the user account credentials reside

Example

```
[Settings]
Priority=Default

[Default]
DomainAdmin=NYCAdmin
DomainAdminDomain=WOODGROVEBANK
DomainAdminPassword=<complex_password>
```

DomainAdminPassword

The password used for the domain Administrator account specified in the **DomainAdmin** property to join the computer to the domain.

Note For ZTI, the credentials that Configuration Manager specifies typically are used. If the **DomainAdmin** property is specified, the credentials in the **DomainAdmin** property override the credentials that Configuration Manager specifies.

Property configured by	
------------------------	--

Property applies to	
---------------------	--

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>domain_admin_password</i>	The password for the domain Administrator account on the target computer

Example
[Settings]
Priority=Default
[Default]
DomainAdmin=NYCAdmin
DomainAdminDomain=WOODGROVEBANK
DomainAdminPassword=<complex_password>

DomainLevel

This entry specifies the domain functional level. This entry is based on the levels that exist in the forest when a new domain is created in an existing forest.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>Level</i>	Sets the domain functional level to one of the following: <ul style="list-style-type: none"> • 2, Windows Server 2003 • 3, Windows Server 2008 • 4, Windows Server 2008 R2 • 5, Windows Server 2012

Example
[Settings]
Priority=Default
[Default]

Example

DomainLevel =3

DomainNetBiosName

Assigns a NetBIOS name to the new domain.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
Name	Assigns a NetBIOS name to the new domain

Example

[Settings] Priority=Default

[Default] DomainNetBiosName=NewDom
--

DomainOUs

A list of AD DS organizational units (OUs) where the target computer account can be created. The **DomainOUs** property lists text values that can be any non-blank value. The **DomainOUs** property has a numeric suffix (for example, **DomainOUs1** or **DomainOUs2**). The values specified by **DomainOUs** will be displayed in the Deployment Wizard and selectable by the user. The **MachineObjectOU** property will then be set to the OU selected.

In addition, the same functionality can be provided by configuring the DomainOUList.xml file. The format of the DomainOUList.xml file is as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<DomainOUs>
    <DomainOU>
        OU=Computers, OU=Teleilers, OU=NYC, DC=WOODGROVEBANK, DC=Com
    </DomainOU>
    <DomainOU>
        OU=Computers, OU=Managers, OU=NYC, DC=WOODGROVEBANK, DC=Com
    </DomainOU>
</DomainOUs>
```

Property configured by	
------------------------	--

Property applies to	
---------------------	--

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
OU	The OU in which the target computer account can be created

Example
[Settings]
Priority=Default
[Default]
OSInstall=Y
Domain0Us1=OU=Computers, OU=Tellers, OU=NYC, DC=WOODGROVEBANK, DC=Com
Domain0Us2=OU=Computers, OU=Managers, OU=NYC, DC=WOODGROVEBANK, DC=Com

DoNotCreateExtraPartition

Specifies that deployments of Windows 7 and Windows Server 2008 R2 will not create the 300 MB system partition.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The additional system partition will not be created.
NO	The additional system partition will be created.

Example
[Settings]
Priority=Default
[Default]

Example

```
OSInstall=YES
DoNotCreateExtraPartition=YES
```

Note Do not use this property in conjunction with properties to configure BitLocker settings.

DoNotFormatAndPartition

This property is used to configure whether MDT performs any of the partitioning and formatting task sequence steps in task sequences created using the MDT task sequence templates.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The partitioning and formatting task sequence steps in an MDT task sequence will be performed.
<i>Any other value</i>	The partitioning and formatting task sequence steps in an MDT task sequence will not be performed. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
OSInstall=YES
SkipUserData=YES
USMTOfflineMigration=TRUE
DoNotFormatAndPartition=YES
OSDStateStorePath=\\WDG-MDT-01\StateStores
```

DriverGroup

A list of text values that associates out-of-box drivers created in the Deployment Workbench with each other (typically based on the make and model of a computer). A driver can be associated with one or more driver groups. The **DriverGroup** property allows the drivers within one or more groups to be deployed to a target computer.

The text values in the list can be any non-blank value. The **DriverGroup** property value has a numeric suffix (for example, **DriverGroup001** or **DriverGroup002**). After it is defined, a driver group is associated with a computer. A computer can be associated with more than one driver group.

For example, there are two sections for each of the computer manufacturers [Mfgr01] and [Mfgr02]. Two driver groups are defined for the manufacturer Mfgr01: Mfgr01 Video Drivers and Mfgr01 Network Drivers. For the manufacturer Mfgr02, one driver group is defined, Mfgr02 Drivers. One driver group, Shared Drivers, is applied to all computers found in the [Default] section.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
<i>driver_group_name</i>	The name of the driver group defined in the Deployment Workbench
Example	
<pre>[Settings] Priority=Make, Default [Default] DriverGroup001=Shared Drivers :: [Mfgr01] DriverGroup001=Mfgr01 Video Drivers DriverGroup002=Mfgr01 Network Drivers [Mfgr02] DriverGroup001=Mfgr02 Drivers</pre>	

DriverInjectionMode

This property is used to control the device drivers that are injected by the [Inject Drivers](#) task sequence step.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
<i>DriverInjectionMode</i>	The mode in which drivers are injected. Valid values are On , Off , and Ask .

Value	Description
Auto	Inject only matching drivers from the selection profile or folder. This is the same behavior as MDT 2008, which injects all drivers that matched one of the plug and play (PnP) identifiers (IDs) on the target computer.
All	Inject all drivers in the selection profile or folder.

Example
<pre>[Settings] Priority=Default [Default] DriverInjectionMode=ALL DriverSelectionProfile=Nothing DriverPaths001=\NYC-AM-FIL-01\Drivers\$ DriverPaths002=\NYC-AM-FIL-03\WinDrv\$</pre>

DriverPaths

A list of UNC paths to shared folders where additional device drivers are located. These device drivers are installed with the target operating system on the target computer. The MDT scripts copy the contents of these folders to the C:\Drivers folder on the target computer. The **DriverPaths** property is a list of text values that can be any non-blank value. The **DriverPaths** property has a numeric suffix (for example, **DriverPaths001** or **DriverPaths002**).

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	
Value	Description
UNC_path	UNC path to the shared folder in which the additional drivers reside

Example
<pre>[Settings] Priority=Default [Default] DriverPaths001=\NYC-AM-FIL-01\Drivers\$</pre>

Example

DriverPaths002=\NYC- AM- FIL- 03\Win8Drv\
--

DriverSelectionProfile

Profile name used during driver installation.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>profile_name</i>	None

Example

[Settings]	
Priority=Default	
[Default]	
DriverSelectionProfile=MonitorDrivers	

EventService

The **EventService** property specifies the URL where the MDT monitoring service is running. By default, the service uses TCP port 9800 to communicate. The MDT monitoring service collects deployment information on the deployment process that can be viewed in the Deployment Workbench and using the [Get-MDTMonitorData](#) cmdlet.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>url_path</i>	The URL to the MDT monitoring service.

Example

[Settings]	
Priority=Default	

Example

```
[Default]
EventService=http://WDG-MDT-01:9800
DeployRoot=\\NYC-AM-FIL-01\Distribution\$
```

ResourceRoot=\\NYC-AM-FIL-01\Resource\\$

EventShare

The **EventShare** property points to a shared folder in which the MDT scripts record events.

By default, the shared folder is created in C:\Events.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
UNC_path	The UNC path to the shared folder in which the MDT scripts record events. The default share name is Events.

Example

```
[Settings]
Priority=Default

[Default]
EventShare=\\NYC-AM-FIL-01\Events
DeployRoot=\\NYC-AM-FIL-01\Distribution\$
```

ResourceRoot=\\NYC-AM-FIL-01\Resource\\$

FinishAction

Specifies the action to be taken when an LTI task sequence finishes, which is after the **Summary** wizard page in the Deployment Wizard.

Tip Use this property in conjunction with the **SkipFinalSummary** property to skip the **Summary** wizard page in the Deployment Wizard and automatically perform the action.

Caution This property value must be specified in uppercase so that the deployment scripts can read it properly.

Property configured by	
BootStrap.ini	

Property applies to	
LTI	●

Property configured by	Property applies to
CustomSettings.ini	<input checked="" type="radio"/>
MDT DB	<input checked="" type="radio"/>
Value	Description
<i>action</i>	<p>Where <i>action</i> is one of the following:</p> <ul style="list-style-type: none"> • SHUTDOWN. Shuts down the target computer. • REBOOT. Restarts the target computer. • RESTART. Same as REBOOT. • LOGOFF. Log off the current user. If the target computer is currently running Windows PE, then the target computer will be restarted. • blank. Exit the Deployment Wizard without performing any additional actions. This is the default setting.
Example	
<pre>[Settings] Priority=Default [Default] FinishAction=REBOOT</pre>	

ForceApplyFallback

Controls the method used for installed Windows:

- **setup.exe.** This method is the traditional method, initiated by running setup.exe from the installation media. MDT uses this method by default.
- **imagex.exe.** This method installs the operating system image using imagex.exe with the **/apply** option. MDT uses this method when the setup.exe method cannot be used (i.e., MDT falls back to using imagex.exe).

Besides controlling the method used to install these operating systems, this property affects which operating system task sequences are listed in the Deployment Wizard for a specific processor architecture boot image. When the value of this property is set to **NEVER**, only operating system task sequences that match the processor architecture of the boot image are displayed. If the value of this property is set to any other value or is blank, all task sequences that can use the imagex.exe installation method are shown, regardless of the processor architecture.

Property configured by	Property applies to
Bootstrap.ini	<input checked="" type="radio"/>

Property configured by		Property applies to	
CustomSettings.ini	●		
MDT DB	●	ZTI	

Value	Description
NEVER	MDT always uses the imagex.exe method if necessary. Only task sequences that deploy an operating system that matches the boot image are displayed in the Deployment Wizard.
Any other value, including blank	MDT attempts to install the operating system using the setup.exe method but falls back to the imagex.exe method, if necessary. Any task sequence that supports the imagex.exe method is displayed in the Deployment Wizard.

Example
<pre>[Settings] Priority=Default [Default] OSInstall=YES ForceApplyFallback=NEVER</pre>

ForestLevel

This entry specifies the forest functional level when a new domain is created in a new forest.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
<i>level</i>	Sets the domain functional level to one of the following: <ul style="list-style-type: none"> • 2, Windows Server 2003 • 3, Windows Server 2008 • 4, Windows Server 2008 R2 • 5, Windows Server 2012

Example

Example

```
[Settings]
Priority=Default

[Default]
ForestLevel=3
```

FullName

The full name of the user of the target computer provided during the installation of the operating system. This value is inserted into the appropriate configuration settings in Unattend.xml.

Note This value is different from the user credentials created after the operating system is deployed. The **FullName** property is provided as information to systems administrators about the user running applications on the target computer.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
full_name	The full name of the user of the target computer

Example

```
[Settings]
Priority=MACAddress, Default
Properties=CustomProperty, ApplicationInstall
```

```
[Default]
CustomProperty=TRUE
OrgName=Woodgrove Bank
```

```
[00:0F:20:35:DE:AC]
OSDNEWMACHINENAME=HPD530-1
ApplicationInstall=Custom
FullName=Woodgrove Bank User
```

```
[00:03:FF:FE:FF:FF]
OSDNEWMACHINENAME=BVMXP
ApplicationInstall=Minim
FullName=Woodgrove Bank Manager
```

GPOPackPath

This property is used to override the default path to the folder in which the GPO packs reside. The path specified in this property is relative to the `Templates\GPOPacks` folder in a distribution share. MDT automatically scans a specific subfolder of this folder based on the operating system being deployed to the target computer, such as `Templates\GPOPacks\operating_system` (where `operating_system` is the operating system being deployed). Table 3 lists the supported operating systems and the subfolders that correspond to each operating system.

Table 3. Windows Operating Systems and Corresponding GPO Pack Subfolder

Operating system	GPO pack subfolder
Windows 7 with SP1	Win7SP1-MDTGPOPack
Windows Server 2008 R2	WS2008R2SP1-MDTGPOPack

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	

Value	Description
<code>path</code>	The path relative to the <code>distribution_share\Templates\GPOPacks</code> folder (where <code>distribution_share</code> is the root folder of the distribution share). The default value is the <code>distribution_share\Templates\GPOPacks\operating_system</code> folder (where <code>operating_system</code> is a subfolder based on the operating system version). In the example below, setting the GPOPackPath property to a value of "Win7-HighSecurity" configures MDT to use the <code>distribution_share\Templates\GPOPacks\Win7-HighSecurity</code> folder as the folder where the GPO packs are stored.

Example
[Settings]
Priority=Default
[Default]
GPOPackPath=Win7-HighSecurity

Groups

The list of local groups on the target computer whose membership will be captured. This group membership is captured during the State Capture Phase and is restored during the State Restore Phase. (The default groups are Administrators and Power Users.) The **Groups** property is a list of text values that can be any non-blank value. The **Groups** property has a numeric suffix (for example, **Groups001** or **Groups002**).

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	● ZTI
Value	Description
<i>group_name</i>	The name of the local group on the target computer for which group membership will be captured
Example	
<pre>[Settings] Priority=Default [Default] DeployRoot=\NYC-AM-FIL-01\Distribution\$ ResourceRoot=\NYC-AM-FIL-01\Resource\$ UDShare=\NYC-AM-FIL-01\Media\$ CaptureGroups=YES Groups001=NYC Application Management Groups002=NYC Help Desk Users</pre>	

HideShell

This property controls the display of Windows Explorer while the LTI task sequence is running in the new operating system on the target computer. This property can be used in conjunction with the **DisableTaskMgr** property.

Note This property can be used with the **DisableTaskMgr** property to help prevent users from interrupting the LTI task sequence. For more information, see the [DisableTaskMgr](#) property.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	● ZTI
Value	Description

Value	Description
YES	Windows Explorer is hidden until the task sequence is complete.
NO	Windows Explorer is visible while the task sequence is running. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
DisableTaskMgr=YES
HideShell=YES
```

OSHome_Page

The URL to be used as the Windows Internet Explorer® home page after the target operating system is deployed.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	●

Value	Description
URL	The URL of the web page to be used as the home page for Internet Explorer on the target computer

Example

```
[Settings]
Priority=Default

[Default]
Home_Page=http://portal.woodgrovebank.com
```

HostName

The IP host name of the target computer (the name assigned to the target computer).

Note This is the computer name of the target computer, not the NetBIOS computer name of the target computer. The NetBIOS computer name can be shorter than the computer name. Also, this property is dynamically set by MDT scripts and cannot have its value set in CustomSettings.ini or the MDT DB. Treat this property as read only. However, you can use this property within

CustomSettings.ini or the MDT DB, as shown in the following examples, to aid in defining the configuration of the target computer.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	
MDT DB	ZTI
Value	Description
<code>host_name</code>	The IP host name assigned to the target computer
Example	
None	

ImagePackageID

The package ID used for the operating system to install during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	
MDT DB	ZTI
Value	Description
None	The package ID used for the operating system to install during OEM deployments
Example	
None	

InputLocale

A list of input locales to be used with the target operating system. More than one input locale can be specified for the target operating system. Each locale must be separated by a semicolon (;). If not specified, the Deployment Wizard uses the input locale configured in the image being deployed.

Exclude this setting in the Windows User State Migration Tool (USMT) when backing up and restoring user state information. Otherwise, the settings in the

user state information will override the values specified in the **InputLocale** property.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	● ZTI
Value	Description
<i>input_locale1;</i> <i>input_locale2</i>	The locale for the keyboard attached to the target computer
Example	
<pre>[Settings] Priority=Default [Default] UserLocale=en-us InputLocale=0409:00000409;0413:00020409;0413:00000409;0409:00020409</pre>	

InstallPackageID

The package ID used for the operating system to install during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	● ZTI
Value	Description
None	The package ID used for the operating system to install during OEM deployments
Example	
None	

Instance

The instance of SQL Server used for querying property values from columns in the table specified in the **Table** property. The database resides on the computer specified in the **SQLServer** property. The instance of SQL Server on the computer is specified in the **Instance** property.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>instance</i>	The name of the instance of SQL Server to be used for querying property values

Example

```
[Settings]
Priority=Computers, Default

[Default]
OSInstall=YES

[Computers]
SQLServer=NYC-SQL-01
Database=MDTDB
Instance=SQLEnterprise2005
Table=Computers
Parameters=Serial Number, AssetTag
ParameterCondition=OR
```

IPAddress

The IP address of the target computer. The format of the IP address returned by the property is standard dotted-decimal notation; for example, 192.168.1.1. Use this property to create a subsection that contains settings targeted to a specific target computer based on the IP address.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	

Property applies to	
LTI	●

Property configured by		Property applies to	
MDT DB		ZTI	●
Value	Description		
<i>ip_address</i>	The IP address of the target computer in standard dotted-decimal notation		
Example			
None			

IsDesktop

Indicator of whether the computer is a desktop, because the **Win32_SystemEnclosure ChassisType** property value is **3, 4, 5, 6, 7, or 15**.

Note Only one of the following properties will be true at a time: **IsDesktop**, **IsLaptop**, **IsServer**.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini					
MDT DB		ZTI	●		
Value		Description			
TRUE		The target computer is a desktop computer.			
FALSE		The target computer is not a desktop computer.			
Example					
None					

IsHypervisorRunning

Specifies whether a hypervisor is present on the target computer. This property is set using information from the **CPUID** interface.

For further information collected about VMs and information returned from the **CPUID** interface, see the following properties:

- **IsVM**
- **SupportsHyperVRole**
- **SupportsNX**
- **SupportsVT**

- **Supports64Bit**
- **VMPlatform**

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Note The IsVM property should be used to determine whether the target computer is a virtual or physical machine.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●

Value	Description
TRUE	A hypervisor is detected.
FALSE	A hypervisor is not detected.

Example
None

IsLaptop

Indicator of whether the computer is a portable computer, because the **Win32_SystemEnclosure ChassisType** property value is **8, 10, 12, 14, 18, or 21**.

Note Only one of the following properties will be true at a time: **IsDesktop, IsLaptop, IsServer**.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●

Value	Description
TRUE	The target computer is a portable computer.
FALSE	The target computer is not a portable computer.

Example
None

IsServer

Indicator of whether the computer is a server, because the **Win32_SystemEnclosure ChassisType** property value is **23**.

Note Only one of the following properties will be true at a time: **IsDesktop**, **IsLaptop**, **IsServer**.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	The target computer is a server.
FALSE	The target computer is not a server.

Example	
None	

IsServerCoreOS

Indicator of whether the current operating system running on the target computer is the Server Core installation option of the Windows Server operating system.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	The operating system on the target computer is the Server Core installation option of Windows Server.
FALSE	The operating system on the target computer is not the Server Core installation option of Windows Server.

Example	
None	

IsServerOS

Indicator of whether the current operating system running on the target computer is a server operating system.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●
Value	Description
TRUE	The operating system on the target computer is a server operating system.
FALSE	The operating system on the target computer is not a server operating system.
Example	
None	

IsUEFI

Specifies whether the target computer is currently running with Unified Extensible Firmware Interface (UEFI). The UEFI is a specification that defines a software interface between an operating system and platform firmware. UEFI is a more secure replacement for the older BIOS firmware interface present in some personal computers. For more information on UEFI, go to <http://www.uefi.org>.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●
Value	Description
TRUE	The target computer is currently running with UEFI.
FALSE	The target computer is not currently running with UEFI. Note It is possible that the target computer may support UEFI, but is running in a compatibility mode that emulates the older BIOS firmware interface. In this situation this value of this property will

Value	Description
	set to FALSE even though the target computer supports UEFI.
Example	
None	

IsVM

Specifies whether the target computer is a VM based on information gathered from the **CPUID** interface. You can determine the specific VM environment using the **VMPlatform** property.

For further information collected about VMs and information returned from the **CPUID** interface, see the following properties:

- **IsHypervisorRunning**
- **SupportsHyperVRole**
- **SupportsNX**
- **SupportsVT**
- **Supports64Bit**
- **VMPlatform**

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	LTI <input checked="" type="checkbox"/>
CustomSettings.ini	ZTI <input checked="" type="checkbox"/>
MDT DB	
Value	Description
TRUE	The target computer is a VM.
FALSE	The target computer is not a VM.
Example	
None	

JoinDomain

The domain that the target computer joins after the target operating system is deployed. This is the domain where the computer account for the target computer is created. The **JoinDomain** property can contain alphanumeric characters, hyphens (-), and underscores (_). The **JoinDomain** property cannot be blank or contain spaces.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
<i>domain_name</i>	The name of the domain that the target computer joins

Example	
[Settings]	
Priority=Default	
[Default]	
JoinDomain=WOODGROVEBANK	
MachineObjectOU=OU=Reception, OU=NYC, DC=Woodgrovebank, DC=com	

JoinWorkgroup

The workgroup that the target computer joins after the target operating system is deployed. The **JoinWorkgroup** property can contain alphanumeric characters, hyphens (-), and underscores (_). The **JoinWorkgroup** property cannot be blank or contain spaces.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
<i>workgroup_name</i>	The name of the workgroup that the target computer joins

Example	
[Settings]	
Priority=Default	
[Default]	
JoinWorkgroup=WDGV_WORKGROUP	

KeyboardLocale

A list of keyboard locales to be used with the target operating system. More than one keyboard locale can be specified for the target operating system. Each locale must be separated by a semicolon (;). If not specified, the Deployment Wizard uses the keyboard locale configured in the image being deployed.

Exclude this setting in USMT when backing up and restoring user state information. Otherwise, the settings in the user state information will override the values specified in the **KeyboardLocale** property.

Note For this property to function properly, it must be configured in both CustomSettings.ini and BootStrap.ini. BootStrap.ini is processed before a deployment share (which contains CustomSettings.ini) has been selected.

Property configured by		Property applies to	
BootStrap.ini	●	LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
<code>keyboard_locale1; keyboard_locale2</code>	The locale of the keyboard attached to the target computer. The value can be specified in the following formats: <ul style="list-style-type: none">• Text (en-us)• Hexadecimal (0409:00000409)

Example 1

```
[Settings]
Priority=Default

[Default]
UserLocale=en-us
KeyboardLocale=en-us
```

Example 2

```
[Settings]
Priority=Default

[Default]
UserLocale=en-us
KeyboardLocale=0409:00000409; 1809:00001809; 041A:0000041A; 083b:000
1083b
```

KeyboardLocalePE

The name of the keyboard locale to be used while in Windows PE only.

Note For this property to function properly, it must be configured in both CustomSettings.ini and BootStrap.ini. BootStrap.ini is processed before a deployment share (which contains CustomSettings.ini) has been selected.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<code>keyboard_locale</code>	<p>The locale of the keyboard attached to the target computer.</p> <p>The value can be specified in the following formats:</p> <ul style="list-style-type: none"> Text (en-us) Hexadecimal (0409:00000409)

Example 1

```
[Settings]
Priority=Default

[Default]
KeyboardLocalePE=en-us
```

Example 2

```
[Settings]
Priority=Default

[Default]
KeyboardLocalePE=0409: 00000409
```

LanguagePacks

A list of the GUIDs for the language packs to be deployed on the target computer. Deployment Workbench specifies these language packs on the OS Packages node. These GUIDs are stored in the Packages.xml file. The **LanguagePacks** property has a numeric suffix (for example, **LanguagePacks001** or **LanguagePacks002**).

Property configured by	
BootStrap.ini	

Property applies to	
LTI	●

Property configured by	Property applies to
CustomSettings.ini	●
MDT DB	
ZTI	
Value	Description
<i>language_pack_guid</i>	The GUID that the Deployment Workbench specifies for the language packs to install on the target computer. The GUID corresponds to the language pack GUID stored in Packages.xml.
Example	
<pre>[Settings] Priority=Default [Default] LanguagePacks001={a1923f8d-b07b-44c7-ac1e-353b7cc4c1ad}</pre>	

LoadStateArgs

The arguments passed to the USMT Loadstate process. The ZTI script inserts the appropriate logging, progress, and state store parameters. If this value is not included in the settings file, the user state restore process is skipped.

If the Loadstate process finishes successfully, the user state information is deleted. In the event of a Loadstate failure (or non-zero return code), the local state store is moved to %WINDIR%\StateStore to prevent deletion and to ensure that no user state information is lost.

Note Do not add any of the following command-line arguments when configuring this property: **/hardlink**, **/nocompress**, **/decrypt**, **/key**, or **/keyfile**. The MDT scripts will add these command-line arguments if applicable to the current deployment scenario.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
Arguments	<p>The command-line arguments passed to Loadstate.exe.</p> <p>The default arguments specified by Deployment Workbench are as follows:</p> <ul style="list-style-type: none"> • /v. Enables verbose output in the Loadstate log. The default is 0. Specify any number from 0 to 15.

Value	Description
	<p>The value 5 enables verbose and status output.</p> <ul style="list-style-type: none"> • /c. When specified, Loadstate will continue to run even if there are nonfatal errors. Without the /c option, Loadstate exits on the first error. • /lac. Specifies that if the account being migrated is a local (non-domain) account, and it does not exist on the destination computer, then USMT will create the account but it will be disabled. <p>For more information about these and other arguments, see the USMT Help files.</p>

Example
<pre>[Settings] Priority=Default [Default] OSInstall=YES ScanStateArgs=/v: 5 /o /c LoadStateArgs=/v: 5 /c /lac DeployRoot=\\NYC-AM-FIL-01\Distribution\$ ResourceRoot=\\NYC-AM-FIL-01\Resourse\$ UDShare=\\NYC-AM-FIL-01\ImgData\$ UDDir=%0SDComputerName%</pre>

Location

The geographic location of the target computers. A list of IP addresses that correspond to the default gateways defined for the computers within that location defines the **Location** property. An IP address for a default gateway can be associated with more than one location.

Typically, the value for the **Location** property is set by performing a database query on the database managed using Deployment Workbench. Deployment Workbench can assist in creating the locations, defining property settings associated with the locations, and then in configuring CustomSettings.ini to perform the database query for the Location property and the property settings associated with the locations.

For example, a **LocationSettings** section in CustomSettings.ini can query the **LocationSettings** view in the database for a list of locations that contain the value specified in the **DefaultGateway** property listed in the **Parameters** property. The query returns all settings associated with each default gateway.

Then the scripts parse each section that corresponds to the locations returned in the query. For example, the value **[Springfield]** and the section

[Springfield-123 Oak Street-4th Floor] in CustomSettings.ini can represent the corresponding locations. This is an example of how one computer can belong to two locations. The [Springfield] section is for all computers in a larger geographic area (an entire city), and the [Springfield-123 Oak Street-4th Floor] section is for all computers on the fourth floor at 123 Oak Street, in Springfield.

Property configured by		Property applies to	
BootStrap.ini	●	LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
<i>location1</i> , <i>location2</i>	The list of locations to be assigned to an individual computer or a group of computers

Example
<pre>[Settings] Priority=LSettings, Default [Default] UserDataLocation=AUTO DeployRoot=\\W2K3-SP1\Distribution\\$ OSInstall=YES ScanStateArgs=/v: 15 /o /c LoadStateArgs=/v: 7 /c [LSettings] SQLServer=w2k3-sp1 Instance=MDT2010 Database=MDTDB Netlib=DBNMPNTW SQLShare=SQLS Table=LocationSettings Parameters=DefaultGateway [Springfield] UDDir=%OSDComputerName% UDShare=\\Springfield-FIL-01\UserData [Springfield-123 Oak Street-4th Floor] DeployRoot=\\Springfield-BDD-01\Distribution\\$</pre>

LongDistanceAccess

The dialing digits to gain access to an outside line to dial long distance. The property can contain only numeric digits. This value is inserted into the appropriate configuration settings in Unattend.xml.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
<i>language_pack_guid</i>	The GUID that the Deployment Workbench specifies for the language packs to install on the target computer. The GUID corresponds to the language pack GUID stored in Packages.xml.
Example	
<pre>[Settings] Priority=Default [Default] AreaCode=206 CountryCode=001 Dialing=TONE LongDistanceAccess=9</pre>	

MACAddress

The media access control (MAC) layer address of the primary network adapter of the target computer. The **MACAddress** property is included on the **Priority** line so that property values specific to a target computer can be provided. Create a section for each MAC address for each of the target computers (such as [00:0F:20:35:DE:AC] or [00:03:FF:FE:FF:FF]) that contain target computer-specific settings.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	
Value	Description

Value	Description
<i>mac_address</i>	The MAC address of the target computer

Example
[Settings] Priority=MACAddress, Default
[Default] CaptureGroups=YES Groups1=NYC Application Management Groups2=NYC Help Desk Users
[00:0F:20:35:DE:AC] OSDNEWMACHINENAME=HPD530-1
[00:03:FF:FE:FF:FF] OSDNEWMACHINENAME=BVMXP

MachineObjectOU

The AD DS OU in the target domain where the computer account for the target computer is created.

Note The OU specified in this property must exist before deploying the target operating system.

Note If a computer object already exists in AD DS, specifying **MachineObjectOU** will not cause the computer object to be moved to the specified OU.

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Value	Description
<i>OU_name</i>	The name of the OU where the computer account for the target computer will be created

Example
[Settings] Priority=Default
[Default] JoinDomain=WOODGROVEBANK

Example

```
Machine0bjectOU=OU=Reception, OU=NYC, DC=Woodgrovebank, DC=com
```

Make

The manufacturer of the target computer. The format for **Make** is undefined. Use this property to create a subsection that contains settings targeted to a specific computer manufacturer (most commonly in conjunction with the **Model** and **Product** properties).

Note This property is dynamically set by MDT scripts and cannot have its value set in CustomSettings.ini or the MDT DB. Treat this property as read only. However, you can use this property within CustomSettings.ini or the MDT DB, as shown in the following examples, to aid in defining the configuration of the target computer.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
make	The manufacturer of the target computer

Example

```
[Settings]
Priority=Make, Default

[Default]

[Dell Computer Corporation]
Subsection=Dell - %Model %

[Dell - Latitude D600]
Packages001=XXX00009: Program9
Packages002=XXX0000A: Program10
```

MandatoryApplications

A list of application GUIDs that will be installed on the target computer. These applications are specified on the Applications node in the Deployment Workbench. The GUIDs are stored in the Applications.xml file. The **MandatoryApplications** property is a list of text values that can be any non-blank value. The **MandatoryApplications** property has a numeric suffix (for example, **MandatoryApplications001** or **MandatoryApplications002**).

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>application_guid</i>	The GUID specified by the Deployment Workbench for the application to be deployed to the target computer. The GUID corresponds to the application GUID stored in the Applications.xml file.

Example
<pre>[Settings] Priority=Default [Default] MandatoryApplications001={ 1D7DF331-47B7-472C-87B3-442597EC2F7D} MandatoryApplications002={ 9d2b8999-5e4d-4f3d-bb05-edaaaf4fe5628} Administrators001=WOODGROVEBANK\NYC Help Desk Staff</pre>

Memory

The amount of memory installed on the target computer in megabytes. For example, the value **2038** indicates 2,038 MB (or 2 GB) of memory is installed on the target computer.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>memory</i>	The amount of memory installed on the target computer in megabytes

Example
None

Model

The model of the target computer. The format for **Model** is undefined. Use this property to create a subsection that contains settings targeted to a specific computer model number for a specific computer manufacturer (most commonly in conjunction with the **Make** and **Product** properties).

Note This property is dynamically set by MDT scripts and cannot have its value set in CustomSettings.ini or the MDT DB. Treat this property as read only. However, you can use this property within CustomSettings.ini or the MDT DB, as shown in the following examples, to aid in defining the configuration of the target computer.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>model</i>	The model of the target computer

Example
[Settings]
Priority=Make, Default
[Default]
[Dell Computer Corporation]
Subsection=Dell - %Model%
[Dell - Latitude D600]
Packages001=XXX00009: Program9
Packages002=XXX0000A: Program10

NetLib

The protocol to be used to communicate with the computer running SQL Server specified in the **SQLServer** property.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description

Value	Description
DBNMPNTW	Use the named pipes protocol to communicate.
DBMSSOCN	Use TCP/IP sockets to communicate.

Example

```
[Settings]
Priority=Computers, Default
```

```
[Default]
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
```

```
[Computers]
SQLServer=NYC-SQL-01
SQLShare=SQLS
NetLib=DBNMPNTW
Database=MDTDB
Instance=SQLEnterprise2005
Table=Computers
Parameters=Serial Number, AssetTag
ParameterCondition=OR
```

NewDomain

Indicates the type of a new domain: whether a new domain in a new forest, the root of a new tree in an existing forest, or a child of an existing domain.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	ZTI

Value	Description
Child	The new domain is a child of an existing domain.
Forest	The new domain is the first domain in a new forest of domain trees.
Tree	The new domain is the root of a new tree in an existing forest.

Example

```
[Settings]
```

Example	
Priority=Default	
[Default]	
NewDomain=Tree	

NewDomainDNSName

Specifies the required name of a new tree in an existing domain or when Setup installs a new forest of domains.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Value	Description
<i>name</i>	Specifies the required name of a new tree in an existing domain or when Setup installs a new forest of domains

Example	
[Settings]	
Priority=Default	
[Default]	
NewDomainDNSName=newdom.WoodGroveBank.com	

Order

The sorting order for the result set on a database query. The result set is based on the configuration settings of the **Database**, **Table**, **SQLServer**, **Parameters**, and **ParameterCondition** properties. More than one property can be provided to sort the results by more than one property.

For example, if **Order=Sequence** is specified in the CustomSettings.ini file, then an **ORDER BY** sequence clause is added to the query. **Specifying Order=Make, Model** adds an **ORDER BY Make, Model** clause to the query.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Value	Description
<i>property1, property2, ...</i>	Properties to define the sort order for the result set (where <i>propertyn</i> represents the properties in the sort criteria)

Example

```
[Settings]
Priority=Computers, Default
```

```
[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
```

```
[Computers]
SQLServer=NYC-SQL-01
SQLShare=SQLS
NetLabel=DBNMPNTW
Database=MDTDB
Instance=SQLEnterprise2005
Table=MakeModel Settings
Parameters=Serial Number, AssetTag
ParameterCondition=OR
Order=Make, Model
```

OrgName

The name of the organization that owns the target computer. This value is inserted into the appropriate configuration settings in Unattend.xml.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>org_name</i>	The name of the organization that owns the target computer

Example

```
[Settings]
Priority=MACAddress, Default
```

Example

```
Properties=CustomProperty, ApplicationInstall
```

```
[Default]
```

```
OSInstall=YES
```

```
ScanStateArgs=/v: 5 /o /c
```

```
LoadStateArgs=/v: 5 /c /lac
```

```
UserDataLocation=NONE
```

```
CustomProperty=TRUE
```

```
OrgName=Woodgrove Bank
```

```
[00: 0F: 20: 35: DE: AC]
```

```
OSDNEWMACHINENAME=HPD530-1
```

```
ApplicationInstall=Custom
```

```
FullName=Woodgrove Bank User
```

```
[00: 03: FF: FE: FF: FF]
```

```
OSDNEWMACHINENAME=BVMXP
```

```
ApplicationInstall=Minimum
```

```
FullName=Woodgrove Bank Manager
```

OSArchitecture

The processor architecture type for the target operating system. This property is referenced during OEM deployments. Valid values are **x86** and **x64**.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
x86	The processor architecture type for the operating system is 32 bit.
x64	The processor architecture type for the operating system is 64 bit.

Example

```
None
```

OSCurrentBuild

The build number of the currently running operating system.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI
Value	Description
7600	Windows 7
9600	Windows 8.1

Example
None

OSCurrentVersion

The version number of the currently running operating system.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI
Value	Description
<i>version_number</i>	The operating system major version, minor version, and build numbers (<i>major.minor.build</i>). For example, 6.3.9600 would represent Windows 8.1.

Example
None

OSDAdapterxDescription

Specifies the name of the network connection as it appears in the Control Panel Network Connections item. The name can be between 0 and 255 characters in length.

This property is for LTI only. For the equivalent property for ZTI, see [OSDAdapterxName](#).

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0Description** or **OSDAdapter1Description**.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	ZTI
Value	Description
Description	The name of the network connection as it appears in the Control Panel Network Connections item
Example	
None	

OSDAdapterxDNSDomain

Specifies the DNS domain name (DNS suffix) that will be assigned to the network connection. This property is for ZTI only. For LTI, see the [OSDAdapterxDNSSuffix](#) property.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0DNSDomain** or **OSDAdapter1DNSDomain**.

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●
Value	Description
DNS_domain_name	A DNS domain name (DNS suffix) that will be assigned to the network connection
Example	
[Settings] Priority=Default	
[Default] OSDAdapter0DNSDomain=WoodGroveBank.com	

OSDAdapterxDNSServerList

This is a comma-delimited list of DNS server IP addresses that will be assigned to the network connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0DNSServerList** or **OSDAdapter1DNSServerList**.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
DNS_servers	A comma-delimited list of DNS server IP addresses that will be assigned to the network connection

Example
[Settings] Priority=Default
[Default] OSDAdapter0DNSServerList=192. 168. 0. 254, 192. 168. 100. 254

OSDAdapterxDNSSuffix

A DNS suffix that will be assigned to the network connection. This property is for LTI only. For ZTI, see the [OSDAdapterxDNSDomain](#) property.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0DNSSuffix** or **OSDAdapter1DNSSuffix**.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
DNS_suffix	A DNS suffix that will be assigned to the network connection

Example
[Settings] Priority=Default

Example

[Default]
OSDAdapter0DNSSuffix= WoodGroveBank. com

OSDAdapterxEnableDHCP

Specifies whether the network connection will be configured via DHCP.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0EnableDHCP** or **OSDAdapter1EnableDHCP**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	The network connection will be configured via DHCP.
FALSE	The network connection will be configured with static configuration.

Example

[Settings]
Priority=Default

[Default]
OSDAdapter0EnableDHCP=TRUE

OSDAdapterxEnableDNSRegistration

Specifies whether DNS registration is enabled on the network connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0EnableDNSRegistration** or **OSDAdapter1EnableDNSRegistration**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description

Value	Description
TRUE	Enables DNS registration
FALSE	Disables DNS registration

Example

```
[Settings]
Priority=Default

[Default]
OSDAdapter0EnableDNSRegistration=TRUE
```

OSDAdapterxEnableFullDNSRegistration

Specifies whether full DNS registration is enabled on the network connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0EnableFullDNSRegistration** or **OSDAdapter1EnableFullDNSRegistration**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	Enables full DNS registration
FALSE	Disables full DNS registration

Example

```
[Settings]
Priority=Default

[Default]
OSDAdapter0EnableFullDNSRegistration=TRUE
```

OSDAdapterxEnableLMHosts

Specifies whether LMHOSTS lookup is enabled on the network connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0EnableLMHosts** or **OSDAdapter1EnableLMHosts**.

Property configured by	
------------------------	--

Property applies to	
---------------------	--

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	Enables LMHOSTS lookup
FALSE	Disables LMHOSTS lookup

Example
[Settings] Priority=Default
[Default] OSDAdapter0EnableLMHosts=TRUE

OSDAdapterxEnableIPProtocolFiltering

This property specifies whether IP protocol filtering should be enabled on the network connection.

Note The *x* in this property's name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0EnableIPProtocolFiltering** or **OSDAdapter1EnableIPProtocolFiltering**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	Enables IP protocol filtering
FALSE	Disables IP protocol filtering

Example
[Settings] Priority=Default
[Default] OSDAdapter0EnableIPProtocolFiltering =TRUE

OSDAdapterxEnableTCPFiltering

Specifies whether TCP/IP filtering should be enabled on the network connection. This property is for ZTI only. For LTI, see the [OSDAdapterxEnableTCPIPFiltering](#) property.

Note The *x* in this property's name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0EnableTCPFiltering** or **OSDAdapter1EnableTCPFiltering**.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
TRUE	Enables TCP/IP filtering
FALSE	Disables TCP/IP filtering

Example
[Settings] Priority=Default
[Default] OSDAdapter0EnableTCPFiltering=TRUE

OSDAdapterxEnableTCPIPFiltering

Specifies whether TCP/IP filtering should be enabled on the network connection. This property is for LTI only. For ZTI, see the [OSDAdapterxEnableTCPFiltering](#) property.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0EnableTCPIPFiltering** or **OSDAdapter1EnableTCPIPFiltering**.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	

Value	Description
TRUE	Enables TCP/IP filtering
FALSE	Disables TCP/IP filtering

Example

[Settings]
Priority=Default
[Default]
OSDAdapter0EnableTCPIPFILTERING=TRUE

OSDAdapterxEnableWINS

Specifies whether WINS will be enabled on the network connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0EnableWINS** or **OSDAdapter1EnableWINS**.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	Enables WINS
FALSE	Disables WINS

Example

[Settings]
Priority=Default
[Default]
OSDAdapter0EnableWINS=TRUE
OSDAdapter0WINSERVERLIST=192.168.0.1, 192.168.100.1

OSDAdapterxGatewayCostMetric

A comma-delimited list of Gateway Cost Metrics specified as either integers or the string "Automatic" (if empty, uses "Automatic") that will be configured on the connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0GatewayCostMetric** or **OSDAdapter1GatewayCostMetric**.

Property configured by	
------------------------	--

Property applies to	
---------------------	--

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
cost_metrics	A comma-delimited list of Gateway Cost Metrics

Example	
[Settings]	
Priority=Default	
[Default]	
OSDAdapter0GatewayCostMetrics=Automatic	

OSDAdapterxGateways

A comma-delimited list of gateways to be assigned to the network connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0Gateways** or **OSDAdapter1Gateways**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
gateways	A comma-delimited list of gateways

Example	
[Settings]	
Priority=Default	
[Default]	
OSDAdapter0Gateways=192. 168. 0. 1, 192. 168. 100. 1	

OSDAdapterxIPAddressList

A comma-delimited list of IP addresses to be assigned to the network connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0IPAddressList** or **OSDAdapter1IPAddressList**.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI			
Value	Description				
<i>IP_addresses</i>	A comma delimited list of IP addresses				
Example					
<pre>[Settings] Priority=Default [Default] OSDAdapter0IPAddressList=192.168.0.40,192.168.100.40 OSDAdapter0SubnetMask=255.255.255.0,255.255.255.0</pre>					

OSDAdapterxIPProtocolFilterList

A comma-delimited list of IP protocol filters to be assigned to the network connection. This property can be configured using the CustomSettings.ini file or the MDT DB but not the Deployment Workbench. If using Configuration Manager it is also configurable using an **Apply Network Settings** task sequence step.

Note The x in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0IPProtocolFilterList** or **OSDAdapter1IPProtocolFilterList**.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI	●		
Value	Description				
<i>protocol_filter_list</i>	A comma-delimited list of IP protocol filters				
Example					
<pre>[Settings] Priority=Default [Default] OSDAdapter0IPProtocolFilterList=a list of approved IP protocols</pre>					

OSDAdapterxMacAddress

Assign the specified configuration settings to the network interface card that matches the specified MAC address.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0MacAddress** or **OSDAdapter1MacAddress**.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
MAC_address	Network adapter MAC address
Example	
<pre>[Settings] Priority=Default [Default] OSDAdapter0MacAddress=00:0C:29:67:A3:6B</pre>	

OSDAdapterxName

Assign the specified configuration settings to the network adapter that matches the specified name. This property is for ZTI only. For the equivalent property for LTI, see [OSDAdapterxDescription](#).

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0Name** or **OSDAdapter1Name**.

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●
Value	Description
name	Network adapter name
Example	
<pre>[Settings] Priority=Default</pre>	

Example
[Default]
OSDAdapter0Name=3Com 3C920 Integrated Fast Ethernet Controller

OSDAdapterxSubnetMask

A comma-delimited list of IP subnet masks to be assigned to the network connection.

Note The x in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0SubnetMask** or **OSDAdapter1SubnetMask**.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	ZTI
Value	Description
<i>subnet_masks</i>	A comma-delimited list of IP subnet masks

Example
[Settings]
Priority=Default
[Default]
OSDAdapter0IPAddressList=192.168.0.40,192.168.100.40
OSDAdapter0SubnetMask=255.255.255.0,255.255.255.0

OSDAdapterxTCPFilterPortList

A comma-delimited list of TCP filter ports to be assigned to the network connection. This property can be configured using the CustomSettings.ini file or the MDT DB but not the Deployment Workbench. If using Configuration Manager it is also configurable using an **Apply Network Settings** task sequence step.

Note The x in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0TCPFilterPortList** or **OSDAdapter1TCPFilterPortList**.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	ZTI

Value	Description
<code>port_list</code>	A comma-delimited list of TCP/IP filter ports

Example
[Settings] Priority=Default
[Default] OSDAdapter0TCPFilterPortList=a list of approved TCP ports

OSDAdapterxTCPIPNetBiosOptions

Specifies the TCP/IP NetBIOS options to be assigned to the network connection.

Note The `x` in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0TCPIPNetBiosOptions** or **OSDAdapter1TCPIPNetBiosOptions**.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	ZTI

Value	Description
0	Disable IP forwarding.
1	Enable IP forwarding.

Example
[Settings] Priority=Default
[Default] OSDAdapter0TCPIPNetBiosOptions=0

OSDAdapterxUDPFILTERPortList

A comma-delimited list of User Datagram Protocol (UDP) filter ports to be assigned to the network connection. This property can be configured using the CustomSettings.ini file and the MDT DB but not the Deployment Workbench. If using Configuration Manager it is also configurable using an **Apply Network Settings** task sequence step.

Note The `x` in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0UDPFILTERPortList** or **OSDAdapter1UDPFILTERPortList**.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●
Value	Description		
<code>port_list</code>	A comma-delimited list of UDP filter ports		
Example			
<pre>[Settings] Priority=Default</pre> <pre>[Default] OSDAdapter0UDPFILTERPortList=a list of approved UDP ports</pre>			

OSDAdapterxWINSServerList

A two-element, comma-delimited list of WINS server IP addresses to be assigned to the network connection.

Note The *x* in this properties name is a placeholder for a zero-based array that contains network adapter information, such as **OSDAdapter0WINSServerList** or **OSDAdapter1WINSServerList**.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●
Value	Description		
<code>WINS_server_list</code>	A comma-delimited list of WINS server IP addresses		
Example			
<pre>[Settings] Priority=Default</pre> <pre>[Default] OSDAdapter0EnableWINS=TRUE OSDAdapter0WINSServerList=192.168.0.1,192.168.100.1</pre>			

OSDAdapterCount

Specifies the number of network connections that are to be configured.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI	●		
Value	Description				
count	The number of network adapters				
Example					
<pre>[Settings] Priority=Default [Default] OSDAdapterCount=1 OSDAdapter0EnableDHCP=False OSDAdapter0IPAddressList=192.168.0.40,192.168.100.40 OSDAdapter0SubnetMask=255.255.255.0,255.255.255.0 OSDAdapter0Gateways=192.168.0.1,192.168.100.1 OSDAdapter0EnableWINS=True OSDAdapter0WINServerList=192.168.0.1,192.168.100.1 OSDAdapter0TCPIPNetBIOSOptions=0 OSDAdapter0MacAddress=00:0C:29:67:A3:6B OSDAdapter0GatewayCostMetrics=Automatic OSDAdapter0EnableTCPPIPFFiltering=True OSDAdapter0EnableLMHosts=True OSDAdapter0EnableFullDNSRegistration=True OSDAdapter0EnableEDNSRegistration=True OSDAdapter0DNSSuffix=WoodGroveBank.com</pre>					

OSDAnswerFilePath

Specifies the path to the answer file to be used during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini			
MDT DB		ZTI	
Value	Description		

Value	Description
<i>file_path</i>	Specifies the path to the answer file to be used during OEM deployments
Example	
None	

OSDBitLockerCreateRecoveryPassword

A Boolean value that indicates whether the process creates a recovery key for BitLocker. The key is used for recovering data encrypted on a BitLocker volume. This key is cryptographically equivalent to a startup key. If available, the recovery key decrypts the VMK, which, in turn, decrypts the FVEK.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	ZTI
Value	Description
AD	A recovery key is created.
Not specified	A recovery key is not created.

Example
<pre>[Settings] Priority=Default [Default] BDEInstallSuppress=N0 BDEDriveLetter=S: BDEDriveSize=2000 OSDBitLockerMode=TPMKey OSDBitLockerCreateRecoveryPassword=AD OSDBitLockerStartupKeyDrive=C:</pre>

OSDBitLockerMode

The type of BitLocker installation to be performed. Protect the target computer using one of the following methods:

- A TPM microcontroller

- A TPM and an external startup key (using a key that is typically stored on a UFD)
- A TPM and PIN
- An external startup key

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
TPM	Protect the computer with TPM only. The TPM is a microcontroller that stores keys, passwords, and digital certificates. The microcontroller is typically an integral part of the computer motherboard.
TPMKey	Protect the computer with TPM and a startup key. Use this option to create a startup key and to save it on a UFD. The startup key must be present in the port each time the computer starts.
TPMPin	Protect the computer with TPM and a pin. Use this option in conjunction with the BDEPin property. Note This value is not valid when using ZTI.
Key	Protect the computer with an external key (the recovery key) that can be stored in a folder, in AD DS, or printed.

Example
<pre>[Settings] Priority=Default [Default] BDEInstallSuppress=N0 BDEDriveLetter=S: BDEDriveSize=2000 OSDBitLockerMode=TPM OSDBitLockerCreateRecoveryPassword=AD</pre>

OSDBitLockerRecoveryPassword

Instead of generating a random recovery password, the **Enable BitLocker** task sequence action uses the specified value as the recovery password. The value must be a valid numerical BitLocker recovery password.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
password	A valid 48-digit password

Example

```
[Settings]
Priority=Default

[Default]
BDEInstallSuppress=NO
BDEDriveLetter=S:
BDEDriveSize=2000
OSDBitLockerMode=TPMKey
OSDBitLockerCreateRecoveryPassword=AD
OSDBitLockerRecoveryPassword=621280128854709621167486709731081433
315062587367
OSDBitLockerStartupKeyDrive=C:
```

OSDBitLockerStartupKey

Instead of generating a random startup key for the key management option **Startup Key on USB only**, the **Enable BitLocker** task sequence action uses the value as the startup key. The value must be a valid, Base64-encoded BitLocker startup key.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
startupkey	Base64-encoded BitLocker startup key

Example

```
[Settings]
Priority=Default
```

Example	
[Default]	
BDEInstallSuppress=NO	
BDEDriveLetter=S:	
BDEDriveSize=2000	
BDEInstall=KEY	
OSDBitLockerCreateRecoveryPassword=AD	
OSDBitLockerStartupKey=8F4922B8-2D8D-479E-B776-12629A361049	

OSDBitLockerStartupKeyDrive

The location for storing the BitLocker recovery key and startup key.

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●
Value	Description
<i>location</i>	The storage location for the recovery key and startup key (either local to the target computer or to a UNC that points to a shared network folder)

Example	
[Settings]	
Priority=Default	
[Default]	
BDEInstallSuppress=NO	
BDEDriveLetter=S:	
BDEDriveSize=2000	
OSDBitLockerMode=TPMKey	
OSDBitLocker CreateRecoveryPassword=AD	
OSDBitLockerStartupKeyDrive=C:	

OSDBitLockerTargetDrive

Specifies the drive to be encrypted. The default drive is the drive that contains the operating system.

Property configured by	Property applies to
BootStrap.ini	●

Property configured by		Property applies to	
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
drive	The drive that is to be encrypted

Example
[Settings]
Priority=Default
[Default]
BDEInstallSuppress=N0
BDEDriveLetter=S:
BDEDriveSize=2000
BDERecoveryPassword=TRUE
OSDBitLockerMode=TPMKey
OSDBitLockerCreateRecoveryPassword=AD
OSDBitLockerTargetDrive=C:

OSDBitLockerWaitForEncryption

Specifies that the deployment process should not proceed until BitLocker has completed the encryption process for all specified drives. Specifying TRUE could dramatically increase the time required to complete the deployment process.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
TRUE	Specifies that the deployment process should wait for drive encryption to finish
FALSE	Specifies that the deployment process should not wait for drive encryption to finish

Example
[Settings]
Priority=Default

Example

```
[Default]
BDEInstallSuppress=N0
BDEDriveLetter=S:
BDEDriveSize=2000
OSDBi tLockerMode=TPMKey
OSDBi tLockerStartupKeyDrive=C:
OSDBi tLockerCreateRecoveryPassword=AD
OSDBi tLockerWaitForEncryption=TRUE
```

OSDComputerName

The new computer name to assign to the target computer.

Note This property can also be set within a task sequence using a customized **Set Task Sequence Variable** task sequence step.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	ZTI
Value	Description
<i>computer_name</i>	The new computer name to assign to the target computer

Example
[Default] OSDComputerName=%_SMSTSMachineName%

OSDDiskAlign

This property is used to pass a value to the **align** parameter of the **create partition primary** command in the **DiskPart** command. The **align** parameter is typically used with hardware RAID Logical Unit Number (LUN) arrays to improve performance when the logical units (LUs) are not cylinder aligned. The **align** parameter aligns a primary partition that is not cylinder aligned at the beginning of a disk and rounds the offset to the closest alignment boundary. For more information on the **align** parameter, see [Create partition primary](#).

Note This property can be used in conjunction with the **OSDDiskOffset** property to set the **offset** parameter for the **create partition primary** command in the **DiskPart** command. For more information, see the [OSDDiskOffset](#) property.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>alignment_value</i>	Specifies the number of kilobytes (KB) from the beginning of the disk to the closest alignment boundary.

Example
[Settings] Priority=Default
[Default] OSDDiskAlign=1024 OSDDiskOffset=2048

OSDDiskIndex

Specifies the disk index that will be configured.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>disk_index</i>	Specifies the disk index that will be configured (The default value is 0 .)

Example
[Settings] Priority=Default
[Default] OSDDiskIndex=0

OSDDiskOffset

This property is used to pass a value to the **offset** parameter of the **create partition primary** command in the **DiskPart** command. For more information on the **offset** parameter, see [Create partition primary](#).

Note This property can be used in conjunction with the **OSDDiskAlign** property to set the **align** parameter for the **create partition primary** command in the **DiskPart** command. For more information, see the [OSDDiskAlign](#) property.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
<code>offset_value</code>	Specifies the byte offset at which to create the partition. For master boot record (MBR) disks, the offset rounds to the closest cylinder boundary.
Example	
<pre>[Settings] Priority=Default</pre> <pre>[Default] OSDDiskAlign=1024 OSDDiskOffset=2048</pre>	

OSDDiskPartBiosCompatibilityMode

This property specifies whether to disable cache alignment optimizations when partitioning the hard disk for compatibility with certain types of BIOS.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
TRUE	Enables cache alignment optimizations when partitioning the hard disk for compatibility with certain types of BIOS
FALSE	Disables cache alignment optimizations when partitioning the hard disk for compatibility with certain

Value	Description
	types of BIOS (This is the default value.)

Example
[Settings] Priority=Default
[Default] OSDDiskPartBIOSCompatibilityMode=TRUE

OSDImageCreator

Specifies the name of the installation account that will be used during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI

Value	Description
<i>image_creator</i>	Specifies the name of the installation account that will be used during OEM deployments

Example
None

OSDImageIndex

Specifies the index of the image in the .wim file. This property is referenced during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI

Value	Description

Value	Description
<i>index</i>	Specifies the index of the image in the WIM file
Example	
None	

OSDImagePackageID

Specifies the package ID for the image to install during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●
Value	Description
<i>package_ID</i>	Specifies the package ID for the image to install during OEM deployments
Example	
None	

OSDInstallEditionIndex

Specifies the index of the image in the WIM file. This property is referenced during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●
Value	Description
<i>index</i>	Specifies the index of the image in the WIM file
Example	
None	

OSDInstallType

Specifies the installation type used for OEM deployments. The default is **Sysprep**.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>install_type</i>	Specifies the installation type used for OEM deployments

Example	
None	

OSDisk

Specifies the drive used to install the operating system during OEM deployments. The default value is **C:**.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>disk</i>	Specifies the drive used to install the operating system during OEM deployments

Example	
None	

OSDPartitions

Specifies the number of defined partitions configurations. The maximum number of partitions that can be configured is two. The default is **None**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>partitions</i>	Specifies the number of defined partitions configurations

Example
<pre>[Settings] Priority=Default [Default] OSDPartitions=1 OSDPartitions0Bootable=TRUE OSDPartitions0FileSystem=NTFS OSDPartitions0QuickFormat=TRUE OSDPartitions0Size=60 OSDPartitions0SizeUnits=GB OSDPartitions0Type=Primary OSDPartitions0VolumeName=OSDisk OSDPartitions0VolumeLetterVariable=NewDrive1</pre>

OSDPartitionsxBootable

The partition at the specified index should be set bootable. The default first partition is set bootable.

Note The *x* in this properties name is a placeholder for a zero-based array that contains partition configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
TRUE	The partition should be set to bootable.
FALSE	Do not set the partition to bootable.

Example

Example

```
[Settings]
Priority=Default

[Default]
OSDPartitions0Bootable=TRUE
```

OSDPartitionsxFileSystem

The type of file system for the partition at the specified index. Valid values are **NTFS** or **FAT32**.

Note The *x* in this properties name is a placeholder for a zero-based array that contains partition configurations.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
file_system	The type of file system for the partition

Example

```
[Settings]
Priority=Default

[Default]
OSDPartitions0FileSystem=NTFS
```

OSDPartitionsxQuickFormat

The partition at the specified index should be quick formatted. The default is **TRUE**.

Note The *x* in this properties name is a placeholder for a zero-based array that contains partition configurations.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	

Property applies to	
LTI	●

Property configured by		Property applies to	
CustomSettings.ini	●		
MDT DB	●	ZTI	

Value	Description
TRUE	Quick-format the partition.
FALSE	Do not quick-format the partition.

Example
[Settings] Priority=Default
[Default] OSDPartitions0QuickFormat=TRUE

OSDPartitionsxSize

The size of the partition at the specified index.

Note The *x* in this properties name is a placeholder for a zero-based array that contains partition configurations.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	

Value	Description
Size	Partition size

Example
[Settings] Priority=Default
[Default] OSDPartitions0Size=60 OSDPartitions0SizeUnits=GB

OSDPartitionsxSizeUnits

The units of measure used when specifying the size of the partition. Valid values are **MB**, **GB**, or **%**. The default value is **MB**.

Note The x in this properties name is a placeholder for a zero-based array that contains partition configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<code>size_units</code>	The units of measure used when specifying the size of the partition

Example
<code>[Settings] Priority=Default</code>
<code>[Default] OSDPartitions0Size=60 OSDPartitions0SizeUnits=GB</code>

OSDPartitionsxType

The type of partition to be created at the specified index.

Note The x in this properties name is a placeholder for a zero-based array that contains partition configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<code>Primary</code>	Create a primary partition. This is the default value.
<code>Logical</code>	Create a logical partition.
<code>Extended</code>	Create an extended partition.

Example
<code>[Settings] Priority=Default</code>
<code>[Default]</code>

Example**OSDPartitions0Type=Primary****OSDPartitionsxVolumeLetterVariable**

The property that receives the drive letter that is assigned to the partition being managed.

Note The *x* in this properties name is a placeholder for a zero-based array that contains partition configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>volume_letter_variable</i>	The name of the variable that will be assigned the drive letter of the partition being managed

Example

```
[Settings]
Priority=Default
```

```
[Default]
OSDPartitions0VolumeLetterVariable=NewDrive1
```

OSDPartitionsxVolumeName

The volume name that will be assigned to the partition at the specified index.

Note The *x* in this properties name is a placeholder for a zero-based array that contains partition configurations.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>volume_name</i>	The volume name that will be assigned to the partition

Example

```
[Settings]
```

Example

```
Priority=Default
[Default]
OSDPartitions0VolumeName=OSDisk
```

OSDPreserveDriveLetter

This property is used to determine whether the **Apply OS** task sequence step should preserve the drive letter in the operating system image file (.wim file) being deployed to the target computer.

Note This property should only be set in a task sequence step, not in the CustomSettings.ini file or in the MDT DB.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	
MDT DB	ZTI

Value	Description
TRUE	The drive letters in the operating system image file (.wim file) and the operating system drives letters after deployment are identical to the drive letters in the .wim file.
FALSE	The drive letters in the operating system image file (.wim file) are ignored, which allows the task sequence to override the driver letters in the .wim file. Note For MDT, this value should always be selected.

Example

```
None
```

OSDStateStorePath

LTI and ZTI use this property to set the path where the user state migration data will be stored, which can be a UNC path, a local path, or a relative path.

Note The **OSDStateStorePath** property takes precedence over the [StatePath](#) or [UserDataLocation](#) property when those properties are also specified.

In a Replace Computer deployment scenario in ZTI, the [Restore User State](#) task sequence step is skipped if the **OSDStateStorePath** property is set to a valid local or UNC path. The workaround is to set the [USMTLocal](#) property to TRUE. Doing so forces ZTI UserState.wsf to recognize the path in the

OSDStateStorePath property. This is caused by the **Request State Store** task sequence step being skipped and the previous value in the **OSDStateStorePath** property being retained.

In a Replace Computer deployment scenario in ZTI, where user state migration data and the entire computer are being backed up, the Backup.wim file is stored in the folder specified in the **OSDStateStorePath** property. This may be caused by specifying the wrong value for the [ComputerBackupLocation](#) property.

For example, the following CustomSettings.ini file will cause the Backup.wim file to be stored in the same folder specified in the **OSDStateStorePath** property:

```
USMTLocal =True
OSDStateStorePath=\fs1\Share\Replace
```

```
ComputerBackupLocation=NETWORK
BackupShare=\fs1\Share\ComputerBackup
BackupDir=Client01
```

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
<i>Path</i>	The path where the user state migration data will be stored, which can be a UNC path, a local path, or a relative path

Example
<pre>[Settings] Priority=Default [Default] USMTLocal =True OSDStateStorePath=\fs1\Share\Replace ComputerBackupLocation=\fs1\Share\ComputerBackup\Client01</pre>

OSDTarg etSystemDrive

Specifies the drive where the operating system will be installed during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read-only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	ZTI
Value	Description
<code>system_drive</code>	Specifies the drive where the operating system will be installed during OEM deployments
Example	
None	

OSDTargetSystemRoot

Specifies the install path where the operating system will be installed during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	ZTI
Value	Description
<code>system_root</code>	Specifies the install path where the operating system will be installed during OEM deployments
Example	
None	

OSFeatures

A comma-delimited list of server feature IDs that will be installed on the target computer.

Note Not all features listed in the ServerManager.xml file are compatible with all server operating systems.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	Property applies to
BootStrap.ini	●

Property configured by		Property applies to	
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
ID1, ID2	The server features that are to be installed on the target computer. Valid values are located in the <i>program_files\Microsoft Deployment Toolkit\Bin\ServerManager.xml</i> file on the MDT server.

Example
[Settings] Priority=Default
[Default] OSFeatures=CMAK, MSMQ-Multicasting, RSAT

OSInstall

Indicates whether the target computer is authorized to have the target operating system installed. If the **OSInstall** property is not listed, the default is to allow deployment of operating systems to any target computer.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
YES	Deployment of an operating system to the target computer is authorized. This is the default value.
NO	Deployment of an operating system to the target computer is not authorized.

Example
[Settings] Priority=Default
[Default] OSInstall=YES

OSRoles

A comma-delimited list of server role IDs that will be installed on the target computer.

Note Not all roles are compatible with all server operating systems.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	● ZTI
Value	Description
<i>ID1, ID2</i>	The server role that is to be installed on the target computer.

See “C:\Program Files\Microsoft Deployment Toolkit\Bin\ServerManager.xml” for valid ID values.

Example
[Settings] Priority=Default
[Default] OSRoles=ADDS

OSRoleServices

A comma-delimited list of server role service IDs that will be installed on the target computer.

Note Not all server role service IDs are compatible with all server operating systems.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	● ZTI
Value	Description
<i>ID</i>	The server role service that will be installed on the target computer. The valid value is:

Value	Description
	<ul style="list-style-type: none"> • ADDS-Domain-Controller

Example
[Settings] Priority=Default
[Default] OSRoleServices=ADDS-Domain-Controller

OSSKU

The edition of the currently running operating system. The operating system edition is determined by using the **OperatingSystemSKU** property of the **Win32_OperatingSystem WMI** class. For a list of the editions the **OperatingSystemSKU** property returns, see the section, "OperatingSystemSKU," at [Win32_OperatingSystem class](#).

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●

Value	Description
edition	The operating system edition. For example, "BUSINESS" for a Business edition of an operating system or "ENTERPRISE" for an Enterprise edition of an operating system.

Example
None

OSVersion

The version of the currently running operating system. This property should only be used to detect if the currently running operating system is Windows PE. Use the [OSVersionNumber](#) property to detect other operating systems.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI

Value	Description
WinPE	Windows PE
2008R2	Windows Server 2008 R2
Win7Client	Windows 7
Other	Operating systems other than those listed, including Windows 8 and Windows Server 2012

Example
None

OSVersionNumber

The operating system major and minor version number. This property is referenced during OEM deployments.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI

Value	Description
<i>version_number</i>	The operating system major and minor version number

Example
None

OverrideProductKey

The Multiple Activation Key (MAK) string to be applied after the target operating system is deployed to the target computer. The value specified in this property is used by the ZTILicensing.wsf script during the State Restore Phase to apply the MAK to the target operating system. The script also configures the volume licensing image to use MAK activation instead of Key Management Service (KMS). The

operating system needs to be activated with Microsoft after the MAK is applied. This is used when the target computer is unable to access a server that is running KMS.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI	●		
Value	Description				
MAK	The MAK string to be provided to the target operating system				
Example					
<pre>[Settings] Priority=Default [Default] ProductKey=AAAAAA-BBBBB-CCCCC-DDDD-EEEEEE-FFFFF OverideProductKey=AAAAAA-BBBBB-CCCCC-DDDD-EEEEEE-FFFFF</pre>					

PackegeGroup

A list of text values that associates operating system packages with each other (typically based on the type of operating system package). An operating system package can be associated with one or more package groups. The **PackageGroup** property allows the operating system packages within one or more groups to be deployed to a target computer.

The text values in the list can be any non-blank value. The **PackageGroup** property value has a numeric suffix (for example, **PackageGroup001** or **PackageGroup002**). After it is defined, a package group is associated with a computer. A computer can be associated with more than one package group.

Note Operating system packages are created on the OS Packages node in the Deployment Workbench.

Note The **PackageGroup** property can be specified in the format *PackageGroup1=Updates* or *PackageGroup001=Updates*.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB		ZTI	
Value	Description		

Value	Description
<code>package_group_name</code>	Name of the package group to be deployed to the target computer

Example
[Settings] Priority=Default
[Default] PackageGroup001=Updates

Packages

The list of Configuration Manager packages to be deployed to the target computer. The **Packages** property has a numeric suffix (for example, Packages001 or Packages002).

Note The **PackageGroup** property can be specified in the format *PackageGroup1=Updates* or *PackageGroup001=Updates*.

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Value	Description
<code>package_id:program_name</code>	Name of the package to be deployed to the target computer

Example
[Settings] Priority=Default
[Default] Packages001=NYC00010: Install Packages002=NYC00011: Install

PackageSelectionProfile

Profile name used during package installation.

Property configured by	Property applies to
BootStrap.ini	●

Property configured by		Property applies to			
CustomSettings.ini	●				
MDT DB	●	ZTI			
Value		Description			
<i>profile_name</i>		Profile name used during package installation			
Example					
<pre>[Settings] Priority=Default</pre> <pre>[Default] PackageSelectionProfile=CoreApplications</pre>					

Parameters

The parameters to be passed to a database query that returns property values from columns in the table specified in the **Table** property. The table is located in the database specified in the **Database** property on the computer specified in the **SQLServer** property. The instance of SQL Server on the computer is specified in the **Instance** property.

Property configured by		Property applies to			
BootStrap.ini	●	LTI	●		
CustomSettings.ini	●				
MDT DB		ZTI	●		
Value		Description			
<i>parameter1,</i> <i>parameter2</i>		The list of parameters to pass to the database query			
Example					
<pre>[Settings] Priority=Computers, Default</pre> <pre>[Default] OSInstall=YES</pre> <pre>[Computers] SQLServer=NYC-SQL-01 SQLShare=SQLS Database=MDTDB</pre>					

Example

```
Instance=SQLEnterprise2005
Table=Computers
Parameters=Serial Number, AssetTag
ParameterCondition=OR
```

ParameterCondition

Indicator of whether a Boolean AND or OR operation is performed on the properties listed in the **Parameters** property.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
AND	A Boolean AND operation is performed on the properties listed in the Parameters property. Only results that match all properties specified in the Parameters property are returned. This is the default value.
OR	A Boolean OR operation is performed on the properties listed in the Parameters property. Results that match any property specified in the Parameters property are returned.

Example

```
[Settings]
Priority=Computers, Default

[Default]
OSInstall=YES

[Computers]
SQLServer=NYC-SQL-01
SQLShare=SQLS
Database=MDTDB
Instance=SQLEnterprise2005
Table=Computers
```

Example

Parameters=Serial Number, AssetTag
ParameterCondition=OR

ParentDomainDNSName

Specifies the DNS domain name of an existing directory service domain when installing a child domain.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>name</i>	Specifies the DNS domain name of an existing directory service domain when installing a child domain

Example

[Settings]
Priority=Default

[Default]
ParentDomainDNSName=WoodGroveBank.com

Password

Specifies the password for the user name (account credentials) to use for promoting the member server to a domain controller.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>password</i>	Specifies the password for the user name (account credentials) to use for promoting the member server to a domain controller

Example

Example
[Settings] Priority=Default
[Default] Password=<complex_password>

Phase

The current phase of the deployment process. The Task Sequencer uses these phases to determine which tasks must be completed.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●

Value	Description
VALIDATION	Identifies that the target computer is capable of running the scripts necessary to complete the deployment process.
STATECAPTURE	Saves any user state migration data before deploying the new target operating system.
PREINSTALL	Completes any tasks that need to be done (such as creating new partitions) before the target operating system is deployed.
INSTALL	Installs the target operating system on the target computer.
POSTINSTALL	Completes any tasks that need to be done before restoring the user state migration data. These tasks customize the target operating system before starting the target computer the first time (such as installing updates or adding drivers).
STATERESTORE	Restores the user state migration data saved during the State Capture Phase.

Example

Example
None

Port

The number of the port that should be used when connecting to the SQL Server database instance that is used for querying property values from columns in the table specified in the **Table** property. The database resides on the computer specified in the **SQLServer** property. The instance of SQL Server on the computer is specified in the **Instance** property. The port used during connection is specified in the **Port** property.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	
Value	Description
<i>port</i>	The number of the port used when connecting to SQL Server

Example
<pre>[Settings] Priority=Computers, Default [Default] OSInstall=YES [Computers] SQLServer=NYC-SQL-01 Database=MDTDB Instance=MDT2010 Port=1433 Table=Computers Parameters=Serial Number, AssetTag ParameterCondition=OR</pre>

PowerUsers

A list of user accounts and domain groups to be added to the local Power Users group on the target computer. The **PowerUsers** property is a list of text values that can be any non-blank value. The **PowerUsers** property has a numeric suffix (for example, **PowerUsers1** or **PowerUsers2**).

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI	●		
Value	Description				
<i>name</i>	Name of the user or group to be added to the local Power Users group				
Example					
<pre>[Settings] Priority=Default [Default] Administrators001=WOODGROVEBANK\NYC Help Desk Staff PowerUsers001=WOODGROVEBANK\User01 PowerUsers002=WOODGROVEBANK\User02</pre>					

PrepareWinRE

This property specifies if the LiteTouchPE.wim file, which includes Windows RE and optionally DaRT, is applied to the system drive as the recovery partition. This allows the target computer to use the LiteTouchPE.wim image to perform recovery tasks. DaRT may optionally be included in the image, which makes DaRT recovery features available on the target computer.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB		ZTI			
Value	Description				
YES	The LiteTouchPE.wim file, which includes Windows RE and optionally DaRT, is applied to the system drive as the recovery partition.				
<i>any other value</i>	The LiteTouchPE.wim file, which includes Windows RE and optionally DaRT, is not applied to the system drive as the recovery partition. This is the default value.				
Example					
<pre>[Settings]</pre>					

Example
Priority=Default
[Default]
PrepareWinRE=YES

Priority

The reserved property that determines the sequence for finding configuration values. The **Priority** reserved property lists each section to be searched and the order in which the sections are searched. When a property value is found, the ZTIGather.wsf script quits searching for the property, and the remaining sections are not scanned for that property.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	● ZTI
MDT DB	●

Value	Description
section1, section2	The sections to be searched in the order they are to be searched

Example
[Settings]
Priority=MACAddress, Default
[Default]
UserDataLocation=NONE
CustomProperty=TRUE
[00:0F:20:35:DE:AC]
OSDNEWMACHINENAME=HPD530-1
[00:03:FF:FE:FF:FF]
OSDNEWMACHINENAME=BVMXP

ProcessorSpeed

The speed of the processor installed on the target computer in MHz. For example, the value **1995** indicates the processor on the target computer is running at 1,995 MHz or 2 gigahertz.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	
Value	Description
<i>processor_speed</i>	The speed of the processor on the target computer in megahertz
Example	
None	

Product

The product name of the target computer. With some computer vendors, the make and model might not be sufficiently unique to identify the characteristics of a particular configuration (for example, hyperthreaded or non-hyperthreaded chipsets). The **Product** property can help to differentiate.

The format for **Product** is undefined. Use this property to create a subsection that contains settings targeted to a specific product name for a specific computer model number for a specific computer manufacturer (most commonly in conjunction with the **Make** and **Model** properties).

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	
Value	Description
<i>product</i>	The product name of the target computer
Example	
None	

ProductKey

The product key string to be configured for the target computer. Before the target operating system is deployed, the product key specified is automatically inserted into the appropriate location in Unattend.xml.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>product_key</i>	The product key to be assigned to the target computer

Example
[Settings] Priority=Default
[Default]
ProductKey=AAAAA-BBBBB-CCCCC-DDDD-EEEEEE-FFFFF

Properties

A reserved property that defines any custom, user-defined properties. These user-defined properties are located by the ZTIGather.wsf script in the CustomSettings.ini file, BootStrap.ini file, or the MDT DB. These properties are additions to the predefined properties in MDT.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>custom_property1</i> , <i>custom_property2</i>	Custom, user-defined properties to be resolved

Example
[Settings] Priority=MACAddress, Default
Properties=CustomProperty, ApplicationInstall

Example

```
[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
UserDataLocation=NONE
CustomProperty=TRUE

[00:0F:20:35:DE:AC]
OSDNEWMACHINENAME=HPD530-1
ApplicationInstall=Custom

[00:03:FF:FE:FF:FF]
OSDNEWMACHINENAME=BVMXP
ApplicationInstall=Minimum
```

ReplicaDomainDNSName

Specifies the DNS domain name of the domain to replicate.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>name</i>	Specifies the DNS domain name of the domain to replicate

Example

```
[Settings]
Priority=Default

[Default]
ReplicaDomainDNSName=WoodGroveBank. com
```

ReplicaOrNewDomain

Specifies whether to install a new domain controller as the first domain controller in a new directory service domain or to install it as a replica directory service domain controller.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
Replica	Installs the new domain controller as a replica directory service domain controller.
Domain	Installs the new domain controller as the first domain controller in a new directory service domain. You must specify the TreeOrChild entry with a valid value.

Example
[Settings] Priority=Default
[Default] ReplicaOrNewDomain=Domain

ReplicationSourceDC

Indicates the full DNS name of the domain controller from which you replicate the domain information.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>name</i>	Indicates the full DNS name of the domain controller from which you replicate the domain information

Example
[Settings] Priority=Default
[Default] ReplicationSourceDC=dc01.WoodGroveBank.com

ResourceDrive

The drive letter mapped to the **ResourceRoot** property for the ZTIDrivers.wsf and ZTIPatches.wsf scripts to use to install drivers and patches to the target computer.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●
Value	Description
<i>drive_letter</i>	The letter designation for the logical drive that contains the resources
Example	
None	

ResourceRoot

The value of this property is used by the ZTIDrivers.wsf and ZTIPatches.wsf scripts to install drivers and patches to the target computer.

Note For LTI, the scripts automatically set the **ResourceRoot** property to be the same as the **DeployRoot** property. For ZTI, the values in the **DeployRoot** and **ResourceRoot** properties can be unique.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●
Value	Description
<i>UNC_path</i>	The UNC path to the shared folder that contains the resources
Example	
[Settings] Priority=Default	
[Default] DeployRoot=\NYC-AM-FIL-01\Distribution\$	

Example

```
ResourceDrive=R:  

ResourceRoot=\\NYC-AM-FIL-01\Resourc$  

UserDataLocation=NONE
```

Role

The purpose of a computer based on the tasks performed by the user on the target computer. The **Role** property lists text values that can be any non-blank value. The **Role** property value has a numeric suffix (for example, **Role1** or **Role2**). When defined, a role is associated with a computer. A computer can perform more than one role.

Typically, the value for the **Role** property is set by performing a database query in the MDT DB. The Deployment Workbench can assist in creating the role and property settings associated with the role, and then the Deployment Workbench can configure CustomSettings.ini to perform the database query for the **Role** property and the property settings associated with the role.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>Role</i>	The roles to be assigned to an individual computer or a group of computers

Example 1

```
[Settings]
Priority=RoleSettings, Default

[Default]
SkipCapture=NO
UserDataLocation=AUTO
DeployRoot=\\W2K3-SP1\Distribution$
OSInstall=YES
ScanStateArgs=/v: 15 /o /c
LoadStateArgs=/v: 7 /c

[RoleSettings]
SQLServer=w2k3-sp1
Instance=MDT2010
```

Example 1

```
Database=MDTDB
Netlib=DBNMPNTW
SQLShare=SQL_Share
Table=RoleSettings
Parameters=Role
```

Example 2

```
[Settings]
Priority=RoleSettings, Default

[Default]
SkipCapture=NO
UserDataLocation=AUTO
DeployRoot=\W2K3-SP1\Distribution\$

Install=YES
Role1=Teller
Role2=Woodgrove User

[RoleSettings]
SQLServer=w2k3-sp1
Instance=MDT2010
Database=MDTDB
Netlib=DBNMPNTW
SQLShare=SQL_Share
Table=RoleSettings
Parameters=Role
```

SafeModeAdminPassword

Supplies the password for the administrator account when starting the computer in Safe mode or a variant of Safe mode, such as Directory Services Restore mode.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
password	Supplies the password for the administrator account when starting the computer in Safe mode or a variant

Value	Description
	of Safe mode, such as Directory Services Restore mode
Example	
[Settings]	
Priority=Default	
[Default]	
SafeModeAdminPassword=<complex_password>	

ScanStateArgs

Arguments passed to the **USMT Scanstate** process. The scripts call Scanstate.exe, and then insert the appropriate logging, progress, and state store parameters. If this value is not included in the settings file, the user state backup process is skipped.

Note Use the **USMTMigFiles** property to specify the .xml files to be used by Scanstate.exe instead of using the **/I** parameter in the **ScanStateArgs** property. This prevents the ZTIUserState.wsf script from potentially duplicating the same list of .xml files.

Note Do not add any of the following command line arguments when configuring this property: **/hardlink**, **/nocompress**, **/encrypt**, **/key**, or **/keyfile**. The MDT scripts will add these command-line arguments if applicable to the current deployment scenario.

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Value	Description
arguments	<p>The command-line arguments passed to Scanstate.exe.</p> <p>The default arguments specified by the Deployment Workbench are as follows:</p> <ul style="list-style-type: none"> • /v. Enables verbose output in the Scanstate log. The default is 0. Specify any number from 0 to 15. The value 5 enables verbose and status output. • /o. Overwrites any existing data in the store. If not specified, Scanstate will fail if the store already contains data. This option cannot be specified more than once in a Command Prompt window. • /c. When specified, Scanstate will continue to run even if there are nonfatal errors. Without the /c

Value	Description
	option, Scanstate exits on the first error. For more information about these and other arguments, see the USMT Help files.

Example
<pre>[Settings] Priority=Default [Default] ScanStateArgs=/v: 5 /o /c LoadStateArgs=/v: 5 /c /lac DeployRoot=\NYC-AM-FIL-01\Distribution\\$ ResourceRoot=\NYC-AM-FIL-01\Resource\\$ UDShare=\NYC-AM-FIL-01\Media\\$ UDDir=%OSDComputerName%</pre>

SerialNumber

The serial number of the target computer. The format for serial numbers is undefined. Use this property to create a subsection that contains settings targeted to a specific computer.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	
MDT DB	●

Value	Description
serial_number	The format of the serial number is undefined and is determined by the serial number standard of each computer manufacturer.

Example
None

SiteName

Specifies the name of an existing site where you can place the new domain controller.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>name</i>	Specifies the name of an existing site where you can place the new domain controller

Example
[Settings] Priority=Default
[Default] SiteName=FirstSite

SkipAdminAccounts

Indicates whether the **Local Administrators** wizard page is skipped.

Note This default value for this property is **YES**, which means that the **Local Administrators** wizard page will be skipped by default. To display this wizard page, you must specifically set the value of this property to **NO** in CustomSettings.ini or in the MDT DB.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	Wizard page is not displayed, and the information on that page is not collected. This is the default value.
NO	Wizard page is displayed, and the information on that page is collected.

Example
[Settings] Priority=Default

Example

```
[Default]
SkipWizard=NO
SkipCapture=NO
SkipAdministratorAccounts=NO
SkipAdministratorPassword=NO
SkipApplications=NO
SkipComputerBackup=NO
SkipDomainMembership=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=YES
SkipProductKey=YES
```

SkipAdministratorPassword

Indicates whether the **Administrator Password** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
YES	Wizard page is not displayed, and the information on that page is not collected.
NO	Wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
```

Example

```
SkipAdminPassword=YES
SkipApplications=NO
SkipComputerBackup=NO
SkipDomainMembership=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
SkipProductKey=YES
```

SkipApplications

Indicates whether the **Select one or more applications to install** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	Wizard page is not displayed, and the information on that page is not collected.
NO	Wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipAdminPassword=NO
SkipApplications=YES
SkipComputerBackup=NO
SkipDomainMembership=NO
```

Example

```
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=YES
SkipProductKey=YES
```

SkipBDDWelcome

Indicates whether the **Welcome to Windows Deployment** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Note For this property to function properly it must be configured in both CustomSettings.ini and BootStrap.ini. BootStrap.ini is processed before a deployment share (which contains CustomSettings.ini) has been selected.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	Wizard page is not displayed, and the information on that page is not collected.
NO	Wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipAdminPassword=YES
SkipApplications=NO
SkipBDDWelcome=YES
SkipComputerBackup=NO
SkipDomainMembership=NO
SkipUserData=NO
```

Example

```
SkipPackageDisplay=NO
SkipLocalSelection=NO
SkipProductKey=YES
```

SkipBitLocker

Indicates whether the **Specify the BitLocker configuration** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
YES	Wizard page is not displayed, and the information on that page is not collected.
NO	Wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipApplications=NO
SkipBDDWelcome=YES
SkipBitLocker=YES
SkipComputerBackup=NO
SkipDomainMembership=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
```

SkipBuild

Indicates whether the **Select a task sequence to execute on this computer** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	Wizard page is not displayed, and the information on that page is not collected.
NO	Wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipAdminPassword=YES
SkipApplications=NO
SkipBDDWelcome=YES
SkipBuild=YES
SkipComputerBackup=NO
SkipComputerName=NO
SkipDomainMembership=NO
SkipFinalSummary=NO
SkipSummary=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
```

SkipCapture

Indicates whether the **Specify whether to capture an image** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=YES
SkipApplications=NO
SkipComputerBackup=NO
SkipDomainMembership=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
```

SkipComputerBackup

Indicates whether the **Specify where to save a complete computer backup** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB	●	ZTI			
Value	Description				
YES	The wizard page is not displayed, and the information on that page is not collected.				
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.				
Example					
<pre>[Settings] Priority=Default [Default] SkipWizard=NO SkipCapture=NO SkipAdminPassword=NO SkipApplications=NO SkipComputerBackup=YES SkipDomainMembership=NO SkipUserData=NO SkipPackageDisplay=NO SkipLocalSelection=NO</pre>					

SkipComputerName

Indicates whether the **Configure the computer name** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	
Value	Description		

Value	Description
YES	Wizard page is not displayed, and the information on that page is not collected.
NO	Wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipAdminPassword=NO
SkipApplications=NO
SkipComputerBackup=NO
SkipComputerName=YES
SkipDomainMembership=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
```

SkipDomainMembership

Indicates whether the **Join the computer to a domain or workgroup** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipAdminPassword=NO
SkipApplications=NO
SkipComputerBackup=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
SkipDomainMembership=NO
```

SkipFinalSummary

Indicates whether the **Operating system deployment completed successfully** wizard page is skipped.

For other properties that must be configured when this property is set to YES, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
```

Example

```
SkipCapture=NO
SkipApplications=NO
SkipBDDWelcome=YES
SkipComputerBackup=NO
SkipComputerName=NO
SkipDomainMembership=NO
SkipFinalSummary=YES
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
SkipProductKey=YES
```

SkipGroupSubFolders

By default, when specifying folders to be included when injecting drivers, patches (packages), and so on, values are specified something like:

```
DriverGroup001=TopFolder\SecondFolder
PackageGroup001=TopFolder\SecondFolder
```

This would, by default, also include all sub-folders located under the "SecondFolder." If **SkipGroupSubFolders** is set to **YES** in CustomSettings.ini, this behavior will change so that the subfolders will be excluded and only the contents of "SecondFolder" will be added.

To exclude subfolders when matching against groups such as DriverGroup001, PackageGroup001, and so on, set **SkipGroupSubFolders** to **YES**.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	Do not include subfolders when matching against groups.
NO	Include subfolders when matching against groups. This is the default behavior.

Example

```
[Settings]
```

Example

```
Pri ority=Default
```

```
[Default]
```

```
SkipGroupSubFolders=NO
```

SkipLocaleSelection

Indicates whether the **Locale Selection** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
```

```
Pri ority=Default
```

```
[Default]
```

```
SkipWizard=NO
```

```
SkipCapture=NO
```

```
SkipApplications=NO
```

```
SkipComputerBackup=NO
```

```
SkipDomainMembership=NO
```

```
SkipUserData=NO
```

```
SkipPackageDisplay=NO
```

```
SkipLocalSelection=NO
```

SkipPackageDisplay

Indicates whether the **Packages** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●
Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.
Example	
<pre>[Settings] Priority=Default [Default] SkipWizard=NO SkipCapture=NO SkipApplications=NO SkipComputerBackup=NO SkipDomainMembership=NO SkipUserData=NO SkipPackageDisplay=YES SkipLocalSelection=NO</pre>	

SkipProductKey

Indicates whether the **Specify the product key needed to install this operating system** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	Property applies to
BootStrap.ini	●

Property configured by		Property applies to	
CustomSettings.ini	●		
MDT DB	●	ZTI	
Value	Description		
YES	The wizard page is not displayed, and the information on that page is not collected.		
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.		
Example			
<pre>[Settings] Priority=Default</pre>			
<pre>[Default] SkipWizard=NO SkipCapture=NO SkipAdminPassword=YES SkipApplications=NO SkipComputerBackup=NO SkipDomainMembership=NO SkipUserData=NO SkipPackageDisplay=NO SkipLocalSelection=NO SkipProductKey=YES</pre>			

SkipRearm

This property is used to configure whether MDT rearms the Microsoft Office 2010 25-day activation grace period. If Microsoft Office 2010 is captured in a custom image, the user sees activation notification dialog boxes immediately after the image is deployed instead of 25-days after deployment.

By default, MDT rearms the Microsoft Office 2010 25-day activation grace period when running the LTISSysprep.wsf script. You can set the value of this property to **YES** so that MDT skips the rearming of the Microsoft Office 2010 25-day activation grace period.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by		Property applies to	
Bootstrap.ini		LTI	●
CustomSettings.ini	●		

Property configured by		Property applies to	
MDT DB		ZTI	
Value	Description		
YES	MDT does not rearm the Microsoft Office 2010 25-day activation grace period.		
NO	MDT rearms the Microsoft Office 2010 25-day activation grace period. This is the default value.		
Example	<pre>[Settings] Priority=Default [Default] OSInstall=Y SkipCapture=YES SkipAdminPassword=NO SkipProductKey=YES SkipRearm=YES DoCapture=YES</pre>		

SkipRoles

Indicates whether the **Roles and Features** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	
Value	Description		
YES	The wizard page is not displayed, and the information on that page is not collected.		
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.		
Example			

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipAdministratorPassword=YES
SkipApplications=YES
SkipBDDWelcome=YES
SkipTaskSequence=Yes
SkipComputerBackup=NO
SkipComputerName=NO
SkipDomainMembership=NO
SkipFinalSummary=NO
SkipRoles=YES
SkipSummary=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
```

SkipSummary

Indicates whether the **Ready to begin** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.

Example

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=No
SkipCapture=No
SkipAdminPassword=YES
SkipApplications=No
SkipBDDWelcome=YES
SkipTaskSequence=Yes
SkipComputerBackup=No
SkipComputerName=No
SkipDomainMembership=No
SkipFinalSummary=No
SkipSummary=No
SkipUserData=No
SkipPackageDisplay=No
SkipLocalSelection=No
```

SkipTaskSequence

Indicates whether the **Select a task sequence to execute on this computer** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Note Specify the **SkipBuild** property when using the Deployment Workbench to configure the Deployment Wizard to skip the **Select a task sequence to execute on this computer** wizard page.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on

Value	Description
	that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipApplications=NO
SkipBDDWelcome=YES
SkipTaskSequence=NO
SkipComputerBackup=NO
SkipComputerName=NO
SkipDomainMembership=NO
SkipFinalSummary=NO
SkipSummary=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
```

SkipTimeZone

Indicates whether the **Set the Time Zone** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=No
SkipCapture=No
SkipAdminPassword=YES
SkipApplications=No
SkipBDDWelcome=YES
SkipTaskSequence=YES
SkipComputerBackup=No
SkipComputerName=No
SkipDomainMembership=No
SkipFinalSummary=No
SkipSummary=No
SkipTimezone=No
SkipUserData=No
SkipPackageDisplay=No
SkipLocalSelection=No
```

SkipUserData

Indicates whether the **Specify whether to restore user data** and **Specify where to save your data and settings** wizard page is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The wizard page is not displayed, and the information on that page is not collected.
NO	The wizard page is displayed, and the information on that page is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
SkipWizard=NO
SkipCapture=NO
SkipAdministratorPassword=YES
SkipApplications=NO
SkipComputerBackup=NO
SkipDomainMembership=NO
SkipUserData=NO
SkipPackageDisplay=NO
SkipLocalSelection=NO
SkipProductKey=YES
```

SkipWizard

Indicates whether the entire **Deployment Wizard** is skipped.

For other properties that must be configured when this property is set to **YES**, see [Providing Properties for Skipped Deployment Wizard Pages](#).

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
YES	The entire wizard is not displayed, and none of the information on the wizard pages is collected.
NO	The wizard is displayed, and the information on the enabled wizard pages is collected. This is the default value.

Example

```
[Settings]
Priority=Default

[Default]
```

Example

`SkipWizard=YES`

SLShare

The network shared folder in which the deployment logs are stored at the end of the deployment process.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<code>shared_folder</code>	The name of the network shared folder in which script logs are stored

Example

```
[Settings]
Priority=Default

[Default]
DeployRoot=\NYC-AM-FIL-01\Distribution$
ResourceRoot=\NYC-AM-FIL-01\Resourse$ 
UDShare=\NYC-AM-FIL-01\Media$ 
UDDir=%OSDComputerName% 
SLShare=\NYC-AM-FIL-01\Logs$ 
UDProfiles=Administrator, User-01, ExtranetUser 
UserDataLocation=NONE 
SkipCapture=NO 
SkipAdminPassword=YES 
SkipProductKey=YES
```

SLShareDynamicLogging

The network shared folder in which all MDT logs should be written during deployment. This is used for advanced real-time debugging only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●

Property applies to	
LTI	●

Property configured by		Property applies to				
MDT DB	●	ZTI	●			
Value	Description					
shared_folder	The name of the network shared folder in which script logs are stored					
Example						
<pre>[Settings] Priority=Default [Default] DeployRoot=\NYC-AM-FIL-01\Distribution ResourceRoot=\NYC-AM-FIL-01\Resources UDShare=\NYC-AM-FIL-01\Media UDDir=%OSDComputerName% SLShare=\NYC-AM-FIL-01\Logs SLShareDynamicLogging=\NYC-AM-FIL-01\Logs UDProfiles=Administrator, User-01, ExtranetUser UserDataLocation=NONE SkipCapture=NO SkipAdminPassword=YES SkipProductKey=YES</pre>						

SMSTSAssignUserMode

Specifies whether user device affinity (UDA) should be enabled and whether approval is required. This property only works with the UDA feature in Configuration Manager.

Property configured by		Property applies to	
BootStrap.ini		LTI	
Value	Description		
CustomSettings.ini			
MDT DB	●	ZTI	●
Value	Description		
Auto	The affinity between a user and the target device is established, and approval is automatically performed.		
Pending	The affinity between a user and the target device is established, and approval is submitted for Configuration Manager administrator approval.		
Disable	The affinity between a user and the target device is		

Value	Description
	not established.

Example

```
[Settings]
Priority=Default
```

```
[Default]
SMSTSAssignUserMode=Auto
SMSTSUserUsers=Fabrikam\Ken, Fabrikam\Pilar
```

SMSTSRunCommandLineUserName

Specifies the user name in *Domain\User_Name* format that should be used with a **Run Command Line** step that is configured to run as a user.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>user_name</i>	Specifies the user name in that should be used with a Run Command Line step

Example

```
[Settings]
Priority=Default
```

```
[Default]
SMSTSRunCommandLineUserName=Fabrikam\Ken
SMSTSRunCommandLineUserPassword=<complex_password>
```

SMSTSRunCommandLineUserPassword

Specifies the password that should be used with a **Run Command Line** step that is configured to run as a user.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<code>user_password</code>	Specifies the password that should be used with a Run Command Line step

Example
<pre>[Settings] Priority=Default [Default] SMSTSRunCommandLineUserName=Fabrikam\Ken SMSTSRunCommandLineUserPassword=<complex_password></pre>

SMSTSUdaUsers

Specifies the users who will be assigned affinity with a specific device using the UDA feature, which is available only in Configuration Manager.

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Value	Description
<code>user1, user2, ...</code>	<p>The comma-separated list of users in <i>Domain\User_Name</i> format that will be assigned affinity with the target device.</p> <p>Note You can only use the NetBIOS domain name in this value, such as <i>Fabrikam\Ken</i>. You cannot use the fully qualified domain name (<i>fabrikam.com\Ken</i>) or the UPN notation (<i>ken@fabrikam.com</i>).</p>

Example
<pre>[Settings] Priority=Default [Default] SMSTSAssignUserMode=Auto SMSTSUdaUsers=Fabrikam\Ken, Fabrikam\Pilar</pre>

SQLServer

The identity of the computer running SQL Server that performs a database query that returns property values from columns in the table specified in the **Table**

property. The query is based on parameters specified in the **Parameters** and **ParameterCondition** properties. The instance of SQL Server on the computer is specified in the **Instance** property.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
SQL_server	The name of the computer running SQL Server

Example

```
[Settings]
Priority=Computers, Default

[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac

[Computers]
SQLServer=NYC-SQL-01
SQLShare=SQLS
Database=MDTDB
Instance=SQLEnterprise2005
Table=Computers
Parameters=Serial Number, AssetTag
ParameterCondition=OR
```

SQLShare

The name of a shared folder on the computer running SQL Server (specified by the **SQLServer** property). The credentials used for authentication are provided by the **UserDomain**, **UserID**, and **UserPassword** properties (for LTI and ZTI) or by the Configuration Manager Advanced Client account credentials (ZTI only).

Note This property must be specified to perform Integrated Windows authentication. This is the recommended authentication method, rather than using the **DBID** and **DBPwd** properties (which support the SQL Server authentication method).

Property configured by	
BootStrap.ini	●

Property applies to	
LTI	●

Property configured by		Property applies to			
CustomSettings.ini	●				
MDT DB		ZTI	●		
Value	Description				
<code>shared_folder</code>	The name of a shared folder on the computer running SQL Server				
Example					
<pre>[Settings] Priority=Computers, Default Properties=MyCustomProperty [Default] OSInstall=YES ScanStateArgs=/v: 5 /o /c LoadStateArgs=/v: 5 /c /lac [Computers] SQLServer=NYC-SQL-01 SQLShare=SQL\$</pre>					
<pre>Database=MDTDB Instance=MDT2010 Table=Computers Parameters=Serial Number, AssetTag ParameterCondition=OR</pre>					

StatePath

This property is used to set the path where the user state migration data will be stored, which can be a UNC path, a local path, or a relative path. The [OSDStateStorePath](#) property takes precedence over the **StatePath** or [UserDataLocation](#) property when those properties are also specified.

Note This property is provided for backward compatibility with previous versions of MDT. Use the [OSDStateStorePath](#) property instead.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB		ZTI	
Value	Description		

Value	Description
<i>Path</i>	The path where the user state migration data will be stored, which can be a UNC path, a local path, or a relative path

Example
[Settings]
Priority=Default
[Default]
SitePath=\fs1\Share\Replace
ComputerBackupLocation=\fs1\Share\ComputerBackup\Client01

StoredProcedure

The name of the stored procedure used when performing a database query that returns property values from columns in the table or view. The stored procedure is located in the database specified in the **Database** property. The computer running SQL Server is specified in the **SQLServer** property. The instance of SQL Server on the computer is specified in the **Instance** property. The name of the stored procedure is specified in the **StoredProcedure** property.

For more information about using a stored procedure to query a SQL Server database, see the section, "Deploying Applications Based on Earlier Application Versions", in the MDT document *Microsoft Deployment Toolkit Samples Guide*.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>stored_procedure</i>	The name of the stored procedure used to query the SQL Server database

Example
[Settings]
Priority=DynamicPackages, Default
[Default]
OSInstall=YES
[DynamicPackages]
SQLDefault=DB_DynamicPackages

Example

```
[DB_DynamicPackages]
SQLServer=SERVER1
Database=MDTDB
StoredProcedure=RetrievePackages
Parameters=MacAddress
SQLShare=Logs
Instance=MDT2013
Port=1433
Netlib=DBNMPNTW
```

SupportsHyperVRole

Specifies whether the processor resources on the target computer can support the Hyper-V server role in Windows Server. This property is True if the value for the following properties is set to **TRUE**:

- **SupportsNX**
- **SupportsVT**
- **Supports64Bit**

Each of the previous properties is set using information from the **CPUID** interface. For further information collected about VMs and information returned from the **CPUID** interface, see the following properties:

- **IsHypervisorRunning**
- **IsVM**
- **SupportsNX**
- **SupportsVT**
- **Supports64Bit**
- **VMPlatform**

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description

Value	Description
TRUE	The processor resources of the target computer can support the Hyper-V server role in Windows Server.
FALSE	The processor resources of the target computer cannot support the Hyper-V server role in Windows Server.

Example
None

SupportsNX

Specifies whether the processor resources on the target computer support the No Execute (NX) technology. The NX technology is used in processors to segregate areas of memory for use by either storage of processor instructions (code) or for storage of data. This property is set using information from the **CPUID** interface.

For further information collected about VMs and information returned from the **CPUID** interface, see the following properties:

- **IsHypervisorRunning**
- **IsVM**
- **SupportsHyperVRole**
- **SupportsVT**
- **Supports64Bit**
- **VMPlatform**

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by
BootStrap.ini
CustomSettings.ini
MDT DB

Property applies to
LTI
ZTI

Value	Description
TRUE	The processor resources of the target computer support NX technology.
FALSE	The processor resources of the target computer do not support NX technology.

Example

Example
None

SupportsVT

Specifies whether the processor resources on the target computer support the Virtualization Technology (VT) feature. VT is used to support current virtualized environments, such as Hyper-V. This property is set using information from the **CPUID** interface.

For further information collected about VMs and information returned from the **CPUID** interface, see the following properties:

- **IsHypervisorRunning**
- **IsVM**
- **SupportsHyperVRole**
- **SupportsNX**
- **Supports64Bit**
- **VMPlatform**

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
TRUE	The processor resources of the target computer support VT technology.
FALSE	The processor resources of the target computer do not support VT technology.

Example
None

Supports64Bit

Specifies whether the processor resources on the target computer support Windows 64-bit operating systems. Most modern virtualization environments require 64-bit processor architecture. This property is set using information from the **CPUID** interface.

For further information collected about VMs and information returned from the **CPUID** interface, see the following properties:

- **IsHypervisorRunning**
- **IsVM**
- **SupportsHyperVRole**
- **SupportsNX**
- **SupportsVT**
- **VMPlatform**

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI

Value	Description
TRUE	The processor resources of the target computer support a Windows 64-bit operating system.
FALSE	The processor resources of the target computer do not support a Windows 64-bit operating system.

Example
None

SysVolPath

Specifies the fully qualified, non-UNC path to a directory on a fixed disk of the local computer.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI

Value	Description
path	Specifies the fully qualified, non-UNC path to a directory on a fixed disk of the local computer

Example

Example

```
[Settings]
Priority=Default

[Default]
SysVol Path=%DestinationLogicalDrive%\Windows\Sysvol
```

Table

The name of the table or view to be used in performing a database query that returns property values from columns in the table or view. The query is based on parameters specified in the **Parameters** and **ParameterCondition** properties. The table or view is located in the database specified in the **Database** property. The computer running SQL Server is specified in the **SQLServer** property. The instance of SQL Server on the computer is specified in the **Instance** property.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
table_name	The name of the table or view to be queried for property values

Example

```
[Settings]
Priority=Computers, Default

[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac

[Computers]
SQLServer=NYC-SQL-01
SQLShare=SQL$ 
Database=MDTDB
Instance=MDT2010
Table=Computers
Parameters=Serial Number, AssetTag
ParameterCondition=OR
```

TaskSequenceID

Identifies the operating system task sequence to be deployed to the target computer. The task sequence ID is created on the Task Sequences node in the Deployment Workbench. The **TaskSequenceID** property allows alphanumeric characters, hyphens (-), and underscores (_). The **TaskSequenceID** property cannot be blank or contain spaces.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	

Value	Description
<code>task_sequence_id</code>	Identifier of the operating system task sequence defined in the Deployment Workbench for the target operating system being deployed Note Be sure to use the TaskSequenceID specified in the Deployment Workbench UI, not the GUID of the TaskSequenceID .

Example
[Settings] Priority=Default [Default] TaskSequenceID=BareMetal

TaskSequenceName

Specifies the name of the task sequence being run.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini			
MDT DB		ZTI	
Value	Description		
<code>task_sequence_name</code>	Name of the task sequence being run, such as Deploy Windows 8.1 to Reference Computer		

Example
None

TaskSequenceVersion

Specifies the version of the task sequence being run.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
<i>task_sequence_version</i>	Version of the task sequence being run, such as 1.00

Example
None

TimeZoneName

The time zone in which the target computer is located. This value is inserted into the appropriate configuration settings in Unattend.xml.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>time_zone_name</i>	The text value that indicates the time zone where the target computer is located

Example

```
[Settings]
Priority=Default

[Default]
TimeZoneName=Pacific Standard Time
DeployRoot=\NYC-AM-FIL-01\Distribution\$ 
ResourceRoot=\NYC-AM-FIL-01\Resource\$
```

Example

```
UDShare=\\NYC- AM- FIL- 01\Mi gData$  
UDDir=%OSComputerName%  
SLShare=\\NYC- AM- FIL- 01\Logs$  
UDProfiles=Administrator, User- 01, ExtranetUser  
UserDataLocation=NONE
```

ToolRoot

Specifies the UNC path to the Tools\proc_arch folder (where *proc_arch* is the processor architecture of the currently running operating system and can have a value of **x86** or **x64**), which is immediately beneath the root of the folder structure specified in the **DeployRoot** property. The Tools\proc_arch folder contains utilities that MDT uses during the deployment process.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	● LTI
CustomSettings.ini	
MDT DB	● ZTI

Value	Description
<i>path</i>	The UNC or local path to the Tools\proc_arch folder (where <i>proc_arch</i> is the processor architecture of the currently running operating system and can have a value of x86 or x64) immediately beneath the root of the folder structure specified by the DeployRoot property

Example

None

TPMOwnerPassword

The TPM password (also known as the *TPM administration password*) for the owner of the target computer. The password can be saved to a file or stored in AD DS.

Note If the TPM ownership is already set or TPM ownership is not allowed, then the **TPMOwnerPassword** property is ignored. If the TPM password is needed and the **TPMOwnerPassword** property is not provided, the TPM password is set to the local Administrator password.

Property configured by	Property applies to
------------------------	---------------------

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>password</i>	The TPM password for the owner of the target computer

Example
<pre>[Settings] Priority=Default [Default] BDEDriveLetter=S: BDEDriveSize=2000 BDEInstall=TPMKey BDERecoveryKey=TRUE BDEKeyLocation=C: TPMOwnerPassword=<complex_password> BackupShare=\\NYC-AM-FIL-01\Backup\$</pre> <p>BackupDir=%OSDComputerName%</p> <p>DeployRoot=\\NYC-AM-FIL-01\Distribution\$</p> <p>ResourceRoot=\\NYC-AM-FIL-01\Resource\$</p> <p>UDShare=\\NYC-AM-FIL-01\MigData\$</p> <p>UDDir=%OSDComputerName%</p>

UDDir

The folder in which the user state migration data is stored. This folder exists beneath the network shared folder specified in **UDShare**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>folder</i>	The name of the folder that exists beneath the network shared folder

Example

```
[Settings]
Priority=Default

[Default]
DeployRoot=\NYC-AM-FIL-01\Distribution$
ResourceRoot=\NYC-AM-FIL-01\Resourse$ 
UDShare=\NYC-AM-FIL-01\ImgData$ 
UDDir=%OSDComputerName% 
SLShare=\NYC-AM-FIL-01\Logs$ 
UDProfiles=Administrator, User-01, ExtranetUser 
UserDataLocation=NONE 
SkipCapture=NO
```

UDProfiles

A comma-delimited list of user profiles that need to be saved by Scanstate.exe during the State Capture Phase.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
user_profiles	The list of user profiles to be saved, separated by commas

Example

```
[Settings]
Priority=Default

[Default]
DeployRoot=\NYC-AM-FIL-01\Distribution$ 
ResourceRoot=\NYC-AM-FIL-01\Resourse$ 
UDShare=\NYC-AM-FIL-01\ImgData$ 
UDDir=%OSDComputerName% 
SLShare=\NYC-AM-FIL-01\Logs$ 
UDProfiles=Administrator, User-01, ExtranetUser 
UserDataLocation=NONE 
SkipCapture=NO
```

UDShare

The network share where user state migration data is stored.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>UNC_path</i>	The UNC path to the network share where user state migration data is stored

Example
<pre>[Settings] Priority=Default [Default] DeployRoot=\NYC-AM-FIL-01\Distribution ResourceRoot=\NYC-AM-FIL-01\Resources UDShare=\NYC-AM-FIL-01\MigData UDDir=%OSDComputerName% SLShare=\NYC-AM-FIL-01\Logs UDProfiles=Administrator, User-01, ExtranetUser UserDataLocation=NONE SkipCapture=No</pre>

UILanguage

The default language to be used with the target operating system. If not specified, the **Deployment Wizard** uses the language configured in the image being deployed.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>UI_language</i>	The default language for the operating system on the target computer

Example

```
[Settings]
Priority=Default

[Default]
UserLocal=en-us
UILanguage=en-us
KeyboardLocal=0409:00000409
```

UserDataLocation

The location in which USMT stores user state migration data.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
blank	If UserDataLocation is not specified or is left blank, the Deployment Wizard will default to using the AUTO behavior.
<i>UNC_path</i>	The UNC path to the network shared folder where the user state migration data is stored.
AUTO	The deployment scripts store the user state migration data on a local hard disk if space is available. Otherwise, the user state migration data is saved to a network location, which is specified in the UDShare and UDDir properties.
NETWORK	The user state migration data is stored in the location designated by the UDShare and UDDir properties.
NONE	The user state migration data is not saved.

Example

```
[Settings]
Priority=Default

[Default]
OSInstall=YES
```

Example

```

ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
DoCapture=YES
BackupShare=\\NYC- AM- FIL- 01\Backup$ 
BackupDir=%OSDComputerName%
UserDataLocation=NETWORK
DeployRoot=\\NYC- AM- FIL- 01\Distribution$ 
ResourceRoot=\\NYC- AM- FIL- 01\Resourc$ 
UDShare=\\NYC- AM- FIL- 01\MigData$ 
UDDir=%OSDComputerName%

```

UserDomain

The domain in which a user's credentials (specified in the **UserID** property) reside.

Note For a completely automated LTI deployment, provide this property in both CustomSettings.ini and BootStrap.ini. However, note that storing the user credentials in these files stores the credentials in clear text and therefore is not secure.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>domain</i>	The name of the domain where the user account credentials reside

Example

```

[Settings]
Priority=Default

[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
DeployRoot=\\NYC- AM- FIL- 01\Distribution$ 
ResourceRoot=\\NYC- AM- FIL- 01\Resourc$ 
UserDataLocation=NONE
UserDomain=WOODGROVEBANK
UserID=NYC Help Desk Staff

```

Example

```
UserPassword=<compl ex_password>
```

UserID

The user credentials for accessing network resources.

Note For a completely automated LTI deployment, provide this property in both CustomSettings.ini and BootStrap.ini. However, note that storing the user credentials in these files stores the credentials in clear text and therefore is not secure.

Property configured by	
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
user_id	The name of the user account credentials used to access the network resources

Example

```
[Settings]
Priority=Default

[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
DeployRoot=\\NYC-AM-FIL-01\Distribution$
ResourceRoot=\\NYC-AM-FIL-01\Resourc$"
UserDataLocation=NONE
UserDomain=WOODGROVEBANK
UserID=NYC-HelpDesk
UserPassword=<compl ex_password>
```

UserLocale

The user locale to be used with the target operating system. If not specified, the **Deployment Wizard** uses the user locale configured in the image being deployed.

Property configured by	
BootStrap.ini	

Property applies to	
LTI	●

Property configured by		Property applies to	
CustomSettings.ini	●		
MDT DB	●	ZTI	●

Value	Description
<i>user_locale</i>	The locale for the user on the target computer. The value is specified as a text value (en-us).

Example 1
[Settings]
Priority=Default
[Default]
UserLocal e=en-us
KeyboardLocal e=0409: 00000409

Example 2
[Settings]
Priority=Default
[Default]
UserLocal e=en-us
KeyboardLocal e=en-us

UserPassword

The password for user credentials specified in the **UserID** property.

Note For a completely automated LTI deployment, provide this property in both CustomSettings.ini and BootStrap.ini. However, note that storing the user credentials in these files stores the credentials in clear text and therefore is not secure.

Property configured by		Property applies to	
BootStrap.ini	●	LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	

Value	Description
<i>user_password</i>	The password for the user account credentials

Example
[Settings]
Priority=Default

Example

```
[Default]
UserDataLocation=NONE
UserDomain=WOODGROVEBANK
UserID=NYC-HelpDesk
UserPassword=<complex_password>
```

USMTConfigFile

The USMT configuration XML file that should be used when running **Scanstate** and **Loadstate**.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
USMTConfigFile	The name of the XML configuration file that should be used when running Scanstate.exe and Loadstate.exe

Example

```
[Settings]
Priority=Default

[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
DeployRoot=\NYC-AM-FIL-01\Distribution$
ResourceRoot=\NYC-AM-FIL-01\Resourse$ 
UDShare=\NYC-AM-FIL-01\ImgData$ 
UDDir=%OSDComputerName% 
SLShare=\NYC-AM-FIL-01\Logs$ 
USMTImgFiles1=ImgApp.xml 
USMTImgFiles2=ImgUser.xml 
USMTImgFiles3=ImgSys.xml 
USMTImgFiles4=ImgCustom.xml 
USMTConfigFile=USMTConfig.xml 
UserDataLocation=NONE
```

USMTLocal

This property specifies whether the USMT user state information is stored locally on the target computer. This property is primarily used by the [ZTIUserState.wsf](#) and [ZTIBackup.wsf](#) scripts to indicate that the **Request State Store** and **Release State Store** task sequence steps for Configuration Manager deployments are skipped. For more information, see the [OSDStateStorePath](#) property.

Note This property should only be used in the circumstance described in the [OSDStateStorePath](#) property).

Property configured by		Property applies to	
BootStrap.ini		LTI	
CustomSettings.ini			
MDT DB		ZTI	●

Value	Description
TRUE	The USMT user state information is stored locally on the target computer, and the Request State Store and Release State Store task sequence steps are skipped.
FALSE	The USMT user state information is not stored locally on the target computer, and the Request State Store and Release State Store task sequence steps are performed.

Example

```
[Settings]
Priority=Default

[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
DeployRoot=\\NYC-AM-FIL-01\Distribution$
ResourceRoot=\\NYC-AM-FIL-01\Resource$
UDShare=\\NYC-AM-FIL-01\Media$ 
UDDir=%OSDComputerName%
SLShare=\\NYC-AM-FIL-01\Logs$ 
USMTLocal =TRUE
USMTMigFiles001=MigApp.xml
USMTMigFiles002=MigUser.xml
USMTMigFiles003=MigSys.xml
USMTMigFiles004=MigCustom.xml
```

Example

UserDataLocation=NONE

USMTMigFiles

A list of files in XML format that are used by USMT (Scanstate.exe) to identify user state migration information to be saved. When this property is not specified, the ZTIUserState.wsf script uses MigApp.xml, MigUser.xml, and MigSys.xml. Otherwise, ZTIUserState.wsf uses the files explicitly referenced in this property. The **USMTMigFiles** property has a numeric suffix (for example, **USMTMigFiles001** or **USMTMigFiles002**).

Note Use this property to specify the XML files to be used by Scanstate.exe instead of using the **/I** parameter in the **ScanStateArgs** property. This prevents the ZTIUserState.wsf script from potentially duplicating the same list of XML files.

Note This property name can be specified using single-digit nomenclature (**USMTMigFiles1**) or triple-digit nomenclature (**USMTMigFiles001**).

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	
Value	Description
USMTMigFile	<p>The name of the .xml file to be used as input for Scanstate.exe, on separate lines. If not specified, the default is MigApp.xml, MigUser.xml, and MigSys.xml.</p> <p>Note If this value is specified, the default files (MigApp.xml, MigUser.xml, and MigSys.xml) must also be added to the list if these files are to be included.</p>

Example

```
[Settings]
Priority=Default

[Default]
OSInstall=YES
ScanStateArgs=/v: 5 /o /c
LoadStateArgs=/v: 5 /c /lac
DeployRoot=\NYC-AM-FIL-01\Distribution$
ResourceRoot=\NYC-AM-FIL-01\Resource$
UDShare=\NYC-AM-FIL-01\MigData$
UDDir=%OSDComputerName%
SLShare=\NYC-AM-FIL-01\Logs$
```

Example

```
USMTMi gFi les001=Mi gApp.xml
USMTMi gFi les002=Mi gUser.xml
USMTMi gFi les003=Mi gSys.xml
USMTMi gFi les004=Mi gCustom.xml
UserDataLocation=NONE
```

USMTOfflineMigration

This property determines whether MDT uses USMT to perform an offline user state migration. In an offline migration, the capture is performed in Windows PE instead of the existing operating system.

Offline migration is using USMT is performed for:

- UDI always, regardless of the setting of the **USMTOfflineMigration** property
 - ZTI only for the MDT Refresh Computer deployment scenario and only when the **USMTOfflineMigration** property is set to "TRUE"
- Note** You cannot perform USMT offline user state migration in the MDT New Computer deployment scenario using ZTI.
- LTI for the:
 - MDT New Computer deployment scenario using the **Move Data and Settings** wizard page in the Deployment Wizard
 - MDT Refresh Computer deployment scenario and only when the **USMTOfflineMigration** property is set to "TRUE"

For more information about using MDT and USMT to perform an offline user state migration, see "Configure USMT Offline User State Migration".

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	
ZTI	●

Value	Description
TRUE	MDT uses USMT to perform an offline user state migration.
<i>Any other value</i>	MDT does not perform an offline user state migration. Instead, user state migration is captured in the existing operating system. This is the default value.

Example

[Settings]

Example

```
Priority=Default
[Default]
OSInstall=YES
SkipUserData=YES
USMTOfflineMigration=TRUE
DoNotFormatAndPartition=YES
OSDStateStorePath=\WDG-MDT-01\StateStore$
```

UUID

The Universal Unique Identifier (UUID) stored in the System Management BIOS of the target computer.

The format for UUID is a 16-byte value using hexadecimal digits in the following format: 12345678-1234-1234-1234-123456789ABC. Use this property to create a subsection that contains settings targeted to a specific computer.

Note This property is dynamically set by MDT scripts and cannot have its value set in CustomSettings.ini or the MDT DB. Treat this property as read only. However, you can use this property within CustomSettings.ini or the MDT DB, as shown in the following examples, to aid in defining the configuration of the target computer.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini			
MDT DB		ZTI	●
Value		Description	
UUID		The UUID of the target computer	
Example			
None			

ValidateDomainCredentialsUNC

This property is used to specify a UNC path to a network shared folder that is used to validate the credentials provided for joining the target computer to a domain. The credentials being validated are specified in the **DomainAdmin**, **DomainAdminDomain**, and **DomainAdminPassword** properties.

Note Ensure that no other properties in MDT use the server sharing the folder in this property. Using a server that is already referenced by other MDT properties could result in improper validation of the credentials.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	

Value	Description
<i>unc_path</i>	Specifies the fully qualified UNC path to a network shared folder

Example
[Settings] Priority=Default
[Default] Val i dateDomai nCredenti al sUNC=\wdg-fs-01\Source\$

MDT DB		ZTI	
--------	--	-----	--

Value	Description
<i>filename</i>	Specifies the name of the differencing VHD file, which is located in the same folder as the parent VHD file Note The differencing VHD file cannot have the same name as the parent VHD file.
RANDOM	Automatically generates a random name for the differencing VHD file, which is located in the same folder as the parent VHD file

Example
[Settings] Priority=Default
[Default] VHDCreateDiffVHD=Win7Diff_C.vhd VHDI nputVari abl e=VHDTar getDi sk

VHDCreateDiffVHD

This property is used to specify the name of a differencing VHD (also known as a *child VHD*) file. A differencing VHD is similar to a dynamically expanding VHD but contains only the modified disk blocks of the associated parent VHD. The parent VHD is read only, so you must modify the differencing VHD. The differencing VHD file is created in the same folder as the parent VHD file, so only the file

name is specified in this property. This property is only valid for the MDT New Computer deployment scenario.

Note All parent VHD files created by MDT are stored in the VHD folder in the root of the parent drive.

This property is commonly set using a task sequence step created using the **Create Virtual Hard Disk (VHD)** task sequence type. You can override the value the **Create Virtual Hard Disk (VHD)** task sequence step sets by configuring this property in CustomSettings.ini.

Note To configure this property in CustomSettings.ini, you must add this property to the **Properties** line in CustomSettings.ini.

For related properties that are used with VHD files, see:

- **VHDCREATEFileName**
- **VHDCREATESizeMax**
- **VHDCREATESource**
- **VHDCREATEType**
- **VHDDisks**
- **VHDIInputVariable**
- **VHDOOutputVariable**
- **VHDTTargetDisk**

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Value	Description
<i>filename</i>	Specifies the name of the differencing VHD file, which is located in the same folder as the parent VHD file Note The differencing VHD file cannot have the same name as the parent VHD file.
RANDOM	Automatically generates a random name for the differencing VHD file, which is located in the same folder as the parent VHD file

Example
[Settings]
Priority=Default
[Default]

Example

```
VHDCreateDiffVHD=Win7Diff_C.vhd
VHDInputVariable=VHDTar
```

VHDCreateFileName

This property is used to specify the name of a VHD file. The type of VHD file is based on the value of the **VHDCreateType** property. The property only includes the file name, not the path to the file name, and is valid only for the MDT New Computer deployment scenario.

Note The VHD files created by MDT are stored in the VHD folder in the root of the parent drive.

This property is commonly set using a task sequence step created using the **Create Virtual Hard Disk (VHD)** task sequence type. You can override the value the **Create Virtual Hard Disk (VHD)** task sequence step sets by configuring this property in CustomSettings.ini.

Note To configure this property in CustomSettings.ini, you must add this property to the **Properties** line in CustomSettings.ini.

For related properties that are used with VHD files, see:

- **VHDCreateDiffVHD**
- **VHDCreateSizeMax**
- **VHDCreateSource**
- **VHDCreateType**
- **VHDDisks**
- **VHDInputVariable**
- **VHDOOutputVariable**
- **VHDTar**

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
<i>file_name</i>	Specifies the name of the VHD file
RANDOM	Automatically generates a random name for the VHD file, which is located in the VHD folder in the root of the parent drive
Blank	Same as RANDOM

Example

```
[Settings]
Priority=Default

[Default]
VHDCreateSizeMax=130048
VHDCreateType=EXPANDABLE
VHDCreateFileName=Win7_C.vhd
VHDInputVariable=VHDTargetDisk
```

VHDCreateSizeMax

This property is used to specify the maximum size of a VHD file in megabytes (MB). The size of the VHD file at creation time is based on the type of VHD file being created. For more information, see the [VHDCreateType](#) property. This property is valid only for the MDT New Computer deployment scenario.

Note If this property is not specified, the default value for the maximum size of a VHD file is 90% of the available disk space on the parent disk.

This property is commonly set using a task sequence step created using the **Create Virtual Hard Disk (VHD)** task sequence type. You can override the value that the **Create Virtual Hard Disk (VHD)** task sequence step sets by configuring this property in CustomSettings.ini.

Note To configure this property in CustomSettings.ini, you must add this property to the **Properties** line in CustomSettings.ini.

For related properties that are used with VHD files, see:

- **VHDCreateDiffVHD**
- **VHDCreateFileName**
- **VHDCreateSource**
- **VHDCreateType**
- **VHDDisks**
- **VHDInputVariable**
- **VHDOOutputVariable**
- **VHDTTargetDisk**

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
size	The maximum size of the VHD file specified in MB. For example, 130,048 MB equals 127 GB. The default value is 90% of the available disk space on the parent disk.

Example

```
[Settings]
Priority=Default

[Default]
VHDCreateSizeMax=130048
VHDCreateType=FIXED
VHDCreateFileName=Win7_C.vhd
VHDInputVariable=VHDTargetDisk
```

VHDCreateSource

This property is used to specify the name of a VHD file that is used as a template (source) for creating a new VHD file. You can specify the file name using a UNC path, local path, relative path, or just the file name. If just the file name is specified, then MDT attempts to find the VHD file on the target computer. This property is valid only for the MDT New Computer deployment scenario.

This property is commonly set using a task sequence step created using the **Create Virtual Hard Disk (VHD)** task sequence type. You can override the value that the **Create Virtual Hard Disk (VHD)** task sequence step sets by configuring this property in CustomSettings.ini.

Note To configure this property in CustomSettings.ini, you must add this property to the **Properties** line in CustomSettings.ini.

For related properties that are used with VHD files, see:

- **VHDCreateDiffVHD**
- **VHDCreateFileName**
- **VHDCreateSizeMax**
- **VHDCreateType**
- **VHDDisks**
- **VHDInputVariable**
- **VHDOOutputVariable**
- **VHDTargetDisk**

Property configured by	
------------------------	--

Property applies to	
---------------------	--

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●	ZTI			
MDT DB					
Value	Description				
<i>name</i>	The file name, which can be specified using a UNC path, local path, relative path, or just the file name. If just the file name is specified, then MDT attempts to find the VHD file on the target computer.				
Example					
<pre>[Settings] Priority=Default [Default] VHDCreateSizeMax=130048 VHDCreateSource=\\wdg-mdt-01\vhds\win7_template.vhd VHDCreateType=FIXED VHDCreateFileName=Win7_C.vhd VHDInputVariable=VHDTargetDisk</pre>					

VHDCreateType

This property is used to specify the type of VHD file that is specified in the **VHDCreateFileName** property and can be one of the following VHD file types:

- **Fixed VHD file.** For this VHD type, the size of the VHD specified at creation is allocated and does not change automatically after creation. For example, if you create a 24-gigabyte (GB) fixed VHD file, the file will be approximately 24 GB in size (with some space used for the internal VHD structure) regardless of how much information is stored in the VHD file.
- **Dynamically expanding VHD file.** For this VHD type, only a small percentage of the size of the VHD specified at creation time is allocated. Then, the VHD file continues to grow as more and more information is stored in it. However, the VHD file cannot grow beyond the size specified at creation. For example, if you create a 24 GB dynamically expanding VHD, it will be small at creation. However, as information is stored in the VHD file, the file will continue to grow but never exceed the maximum size of 24 GB.

This property is only valid for the MDT New Computer deployment scenario.

Note The maximum size of the VHD file is specified in the **VHDCreateSizeMax** property.

This property is commonly set using a task sequence step created using the **Create Virtual Hard Disk (VHD)** task sequence type. You can override the value

that the **Create Virtual Hard Disk (VHD)** task sequence step sets by configuring this property in CustomSettings.ini.

Note To configure this property in CustomSettings.ini, you must add this property to the **Properties** line in CustomSettings.ini.

For related properties that are used in creating VHD files, see:

- **VHDCreateDiffVHD**
- **VHDCreateFileName**
- **VHDCreateSizeMax**
- **VHDCreateSource**
- **VHDDisks**
- **VHDIInputVariable**
- **VHDOOutputVariable**
- **VHDTargetDisk**

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
EXPANDABLE	Creates a fixed VHD file
FIXED	Creates a dynamically expanding VHD file

Example	
[Settings]	
Priority=Default	
[Default]	
VHDCreateSizeMax=130048	
VHDCreateType=EXPANDABLE	
VHDCreateFileName=Win7_C.vhd	
VHDIInputVariable=VHDTargetDisk	

VHDDisks

This property contains a list of the physical drive numbers assigned to VHD files separated by spaces. Each time a VHD file is created, MDT adds the disk index of the newly created disk to this property using the **Index** property of the **Win32_DiskDrive** WMI class.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

This property is commonly set using a task sequence step created using the **Create Virtual Hard Disk (VHD)** task sequence type. You can override the value that the **Create Virtual Hard Disk (VHD)** task sequence step sets by configuring this property in CustomSettings.ini.

Note To configure this property in CustomSettings.ini, you must add this property to the **Properties** line in CustomSettings.ini.

For related properties that are used with VHD files, see:

- **VHDCreateDiffVHD**
- **VHDCreateFileName**
- **VHDCreateSizeMax**
- **VHDCreateSource**
- **VHDCreateType**
- **VHDInputVariable**
- **VHDOOutputVariable**
- **VHDTTargetDisk**

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
<i>index1 index2 index3</i>	A list of the physical drive numbers assigned to the VHD files separated by spaces—for example, 1 2 5.

Example
None

VHDInputVariable

This property contains a variable that contains the drive on the target computer where the VHD files will be created. MDT creates the VHD files in the VHD folder in the root of this drive.

Note If this property is omitted, MDT attempts to create the VHD files in the VHD folder in the root of the first system drive.

This property is commonly set using a task sequence step created using the **Create Virtual Hard Disk (VHD)** task sequence type. You can override the value

that the **Create Virtual Hard Disk (VHD)** task sequence step sets by configuring this property in CustomSettings.ini.

Note To configure this property in CustomSettings.ini, you must add this property to the **Properties** line in CustomSettings.ini.

For related properties that are used with VHD files, see:

- **VHDCreateDiffVHD**
- **VHDCreateFileName**
- **VHDCreateSizeMax**
- **VHDCreateSource**
- **VHDCreateType**
- **VHDDrives**
- **VHDOOutputVariable**
- **VHDTarDisk**

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	

Value	Description
<i>variable</i>	Variable that contains the drive letter on the target computer where the VHD files will be created. MDT creates the VHD files in the VHD folder in the root of this drive. For example, if this property has a value of VHDTarDisk , the VHDTarDisk property contains the drive letter (such as <i>H</i>).

Example
VHDCreateSi zeMax=130048
VHDCreateType=EXPANDABLE
VHDCreateFi leName=Wi n7_C.vhd
VHDOOutputVariabl e=VHDTarDisk

VHDOOutputVariable

This property contains a variable that contains the physical drive number that was assigned to the newly created VHD file. Each time a VHD file is created, MDT sets this property to the disk index of the newly created disk using the **Index** property of the **Win32_DiskDrive** WMI class.

This property is commonly set using a task sequence step created using the **Create Virtual Hard Disk (VHD)** task sequence type. You can override the value that the **Create Virtual Hard Disk (VHD)** task sequence step sets by configuring this property in CustomSettings.ini.

Note To configure this property in CustomSettings.ini, you must add this property to the **Properties** line in CustomSettings.ini.

For related properties that are used with VHD files, see:

- **VHDCreateDiffVHD**
- **VHDCreateFileName**
- **VHDCreateSizeMax**
- **VHDCreateSource**
- **VHDCreateType**
- **VHDDisks**
- **VHDInputVariable**
- **VHDTargetDisk**

Property configured by	Property applies to
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Value	Description
Variable	Variable will contain the physical drive number assigned to the newly created VHD file. For example, if this property has a value of OSDDiskIndex , the OSDDiskIndex property will contain the physical drive number assigned to the newly created VHD file (such as 4).

Example
None

VHDTargetDisk

Specifies the drive on the target computer where the VHD is to be created. This property is later referenced in the [VHDInputVariable](#) property.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

For related properties that are used with VHD files, see:

- **VHDCreateDiffVHD**

- **VHDCreateFileName**
- **VHDCreateSizeMax**
- **VHDCreateSource**
- **VHDCreateType**
- **VHDDisks**
- **VHDIInputVariable**
- **VHDOOutputVariable**

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
Disk	Specifies the drive where the VHD is to be created

Example	
None	

VMHost

Specifies the name of the Hyper-V host running the VM where MDT is running. This property is available only when the Hyper-V Integration Components are installed and running.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Table 4 lists the Windows operating systems that MDT supports and their corresponding Hyper-V Integration Components support.

Table 4. Windows Operating Systems and Hyper-V Integration Components Support

Operating system	Hyper-V Integration Components
Windows PE	Integration Components are unavailable.
Windows 7	Available by default in Enterprise, Ultimate, and Professional editions.
Windows Server 2008 R2	Available by default in all editions.

Property configured by	
BootStrap.ini	

Property applies to	
LTI	●

Property configured by		Property applies to	
CustomSettings.ini			
MDT DB		ZTI	●
Value	Description		
Name	The name of the Hyper-V host running the VM where MDT is running		
Example			
None			

VMName

Specifies the name of the VM where MDT is running. This property is only available when the Hyper-V Integration Components are installed and running.

Table 5 lists the Windows operating systems supported by MDT and their corresponding Hyper-V Integration Components support.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Table 5. Windows Operating Systems and Hyper-V Integration Components Support

Operating system	Hyper-V Integration Components		
Windows PE	Integration Components are unavailable.		
Windows 7	Available by default in Enterprise, Ultimate, and Professional editions.		
Windows Server 2008 R2	Available by default in all editions.		
Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini			
MDT DB		ZTI	●
Value	Description		
name	The name of the VM where MDT is running		
Example			
None			

VMPlatform

Specifies specific information about the virtualization environment for the target computer when the target computer is a VM. The VM platform is determined by using WMI.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	
BootStrap.ini	
CustomSettings.ini	
MDT DB	

Property applies to	
LTI	●
ZTI	●

Value	Description
Hyper-V	Hyper-V
VirtualBox	Virtual Box
VMware	VMware virtualization platform
Xen	Citrix Xen Server

Example	
None	

VRefresh

The vertical refresh rate for the monitor on the target computer. The vertical refresh rate is specified in Hertz. In the example, the value **60** indicates that the vertical refresh rate of the monitor is 60 Hz. This value is inserted into the appropriate configuration settings in Unattend.xml.

Note The default values (in the Unattend.xml template file) are 1,024 pixels horizontal resolution, 768 pixels vertical resolution, 32-bit color depth, and 60 Hz vertical refresh rate.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
refresh_rate	The vertical refresh rate for the monitor on the target computer in Hertz

Example	
[Settings]	

Example
<pre>Pri ority=Default [Default] BitsPerPel=32 VRefresh=60 XResolution=1024 YResolution=768</pre>

VSSMaxSize

This property is used to pass a value to the **maxsize** parameter of the **vssadmin resize shadowstorage** command in the **Vssadmin** command. The **maxsize** parameter is used to specify the maximum amount of space on the target volume that can be used for storing shadow copies. For more information on the **maxsize** parameter, see [Vssadmin resize shadowstorage](#).

Property configured by	Property applies to
BootStrap.ini	●
CustomSettings.ini	●
MDT DB	●

Value	Description
<i>maxsize_value</i>	<p>Specifies the maximum amount of space that can be used for storing shadow copies. The value can be specified in bytes or as a percentage of the target volume.</p> <p>To specify the value:</p> <ul style="list-style-type: none"> In bytes, the value must be 300 MB or greater and accept the following suffixes: KB, MB, GB, TB, PB and EB. You can also use B, K, M, G, T, P, and E as suffixes—for example: VSSMaxSize=60GB As a percentage, use the % character as the suffix to the numeric value—for example: VSSMaxSize=20% <p>Note If a suffix is not supplied, the default suffix is bytes. For example, VSSMaxSize=1024 indicates that the VSSMaxSize will be set to 1,024 bytes.</p> <p>If the value is set to UNBOUNDED, then there is no limit placed on the amount of storage space that can be used—for example: VSSMaxSize=UNBOUNDED</p>

Example
[Settings] Priority=Default
[Default] VSSMaxSize=25%

WDSServer

The computer running Windows Deployment Services that is used for installing Windows Deployment Services images. The default value is the server running Windows Deployment Services from which the image was initiated.

Note This property is dynamically set by the MDT scripts and is not configured in CustomSettings.ini or the MDT DB. Treat this property as read only.

Property configured by	Property applies to
BootStrap.ini	LTI
CustomSettings.ini	ZTI
MDT DB	
Value	Description
WDS_server	The name of the computer running Windows Deployment Services

Example
None

WindowsSource

MDT uses this property to set the location of the sources\sxs folder in a network shared folder that contains the operating system source files. This property is used when:

- MDT is running a custom task sequence or deploying a custom image
- MDT is installing roles or features in Windows 8 and Windows Server 2012
- The computer does not have access to the Internet

When the situation described in the bulleted list above occurs, MDT may be unable to find the operating system source files locally, and the installation will attempt to download the files from the Internet. Because the computer does not have Internet access, the process will fail. Setting this property to the appropriate value helps prevent this problem from occurring.

Property configured by		Property applies to			
BootStrap.ini		LTI	●		
CustomSettings.ini	●				
MDT DB		ZTI	●		
Value	Description				
folder_unc	A UNC path to the Sources\sxs folder for the operating system being deployed. Note The UNC path must include the Sources\sxs folder.				
Example					
<pre>[Settings] Priority=Default [Default] WindowsSource=%DeployRoot%\Operating Systems\Windows 8\Sources\sxs</pre>					

WipeDisk

Specifies whether the disk should be wiped. If **WipeDisk** is TRUE, the ZTIWipeDisk.wsf script will clean the disk using the **Format** command. The **Format** command is not the most "secure" way of wiping the disk.

Securely wiping the disk should be done so in a manner that follows the U.S. Department of Defense standard 5220.22-M, which states, "To clear magnetic disks, overwrite all locations three times (first time with a character, second time with its complement, and the third time with a random character)."

When MDT wipes the disk, it uses the **Format** command with the **/P:3** switch, which instructs **Format** to zero every sector on the volume and to perform the operation three times. There is no way to tell the **Format** command to use a particular character or a random character.

Note If the disk must be securely wiped, a non-Microsoft secure disk wipe tool should be added to the task sequence using the **Run Command Line** task sequence step.

Caution This property value must be specified in uppercase letters so that the deployment scripts can properly read it.

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB	●	ZTI	●
Value	Description		

Value	Description
TRUE	If WipeDisk is set to TRUE , the Win32_DiskPartition at DiskIndex 0 and Index 0 will be formatted.
FALSE	The disk will not be formatted.

Example
[Settings] Priority=Default
[Default] WipeDisk=TRUE

WizardSelectionProfile

Profile name used by the wizard for filtering the display of various items.

Property configured by	Property applies to
Bootstrap.ini	
CustomSettings.ini	●
MDT DB	●

Value	Description
<i>profile_name</i>	Profile name used by the wizard for filtering the display of various items

Example
[Settings] Priority=Default
[Default] WizardSelectionProfile>SelectTaskSequenceOnly

WSUSServer

This is the name of the Windows Server Update Services (WSUS) server that the target computer should use when scanning for, downloading, and installing updates.

For more information about what script uses this property, see [ZTIWindowsUpdate.wsf](#).

Property configured by	Property applies to
Bootstrap.ini	●

Property configured by		Property applies to	
CustomSettings.ini	●		
MDT DB	●	ZTI	●
Value	Description		
server_name	The name of the WSUS server, specified in HTTP format		
Example			
[Settings]			
Priority=Default			
[Default]			
WSUSServer=http://WSUSServerName			

WUMU_ExcludeKB

The list of Windows Update/Microsoft Update software updates to ignore (by associated Knowledge Base articles).

Deployment project team members will want to periodically review the list of updates being installed by the ZTIWindowsUpdate.wsf script to verify that each update meets the project's needs and expectations. All updates are logged and recorded in the ZTIWindowsUpdate.log file, which is generated during deployment. Each update will indicate its status as INSTALL or SKIP and lists the UpdateID, the update name, and the QNumber associated with each update. If an update needs to be excluded, that update should be added to the CustomSettings.ini file (for LTI deployments).

Property configured by		Property applies to	
BootStrap.ini		LTI	●
CustomSettings.ini	●		
MDT DB		ZTI	
Value	Description		
WUMU_ExcludeKB	The list of Windows Update/Microsoft Update software updates to ignore by QNumber		
Example			
[Settings]			
Priority=Default			
[Default]			

Example**WUMU_ExcludeID=925471****WUMU_ExcludeID**

The list of Windows Update/Microsoft Update software updates to ignore (by associated update ID).

Deployment project team members will want to periodically review the list of updates being installed by the ZTIWindowsUpdate.wsf script to verify that each update meets the project's needs and expectations. All updates are logged and recorded in the ZTIWindowsUpdate.log file, which is generated during deployment. Each update will indicate its status as INSTALL or SKIP and lists the UpdateID, the update name, and the QNumber associated with each update. If an update should be excluded, that update should be added to the CustomSettings.ini file (for LTI deployments).

For example, if the installation of the Windows Malicious Software Removal Tool should be excluded, look up the line in the ZTIWindowsUpdate.log that shows where the update was identified and installed, and then select the UpdateID number. For example, the UpdateID number for the Windows Malicious Software Removal Tool is adbe6425-6560-4d40-9478-1e35b3cdab4f.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	

Property applies to	
LTI	●
ZTI	

Value	Description
WUMU_ExcludeID	The list of Windows Update/Microsoft Update software updates to ignore, by UpdateID number

Example

```
[Settings]
Priority=Default
```

```
[Default]
WUMU_ExcludeID={adbe6425-6560-4d40-9478-1e35b3cdab4f}
```

XResolution

The horizontal resolution of the monitor on the target computer, specified in pixels. In the example, the value **1024** indicates the horizontal resolution of the monitor is 1,024 pixels. This value is inserted into the appropriate configuration settings in Unattend.xml.

Note The default values (in the Unattend.xml template file) are 1,024 pixels horizontal resolution, 768 pixels vertical resolution, 32-bit color depth, and 60 Hz vertical refresh rate.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>horizontal_resolution</i>	The horizontal resolution of the monitor on the target computer in pixels

Example

```
[Settings]
Priority=Default

[Default]
BitsPerPel=32
VRefresh=60
XResolution=1024
YResolution=768
```

YResolution

The vertical resolution of the monitor on the target computer, specified in pixels. In the example, the value **768** indicates the vertical resolution of the monitor is 768 pixels. This value gets inserted into the appropriate configuration settings in Unattend.xml.

Note The default values (in the Unattend.xml template file) are 1,024 pixels horizontal resolution, 768 pixels vertical resolution, 32-bit color depth, and 60 Hz vertical refresh rate.

Property configured by	
BootStrap.ini	
CustomSettings.ini	●
MDT DB	●

Property applies to	
LTI	●
ZTI	●

Value	Description
<i>vertical_resolution</i>	The vertical resolution of the monitor on the target computer in pixels

Example

```
[Settings]
```

Example
<pre> Priority=Default [Default] BitsPerPixel=32 VRefresh=60 XResolution=1024 YResolution=768 </pre>

Providing Properties for Skipped Deployment Wizard Pages

Table 6 lists the individual Deployment Wizard pages, the property to skip the corresponding wizard page, and the properties that must be configured when skipping the wizard page.

If the **SkipWizard** property is used to skip all the Deployment Wizard pages, provide all the properties in the **Configure these properties** column. For examples of various deployment scenarios that skip Deployment Wizard pages, see the section, "Fully Automated LTI Deployment Scenario", in the MDT document *Microsoft Deployment Toolkit Samples Guide*.

Note In instances where the **Configure These Properties** column is blank, no properties need to be configured when skipping the corresponding wizard page.

Table 6. Deployment Wizard Pages

Skip this wizard page	Using this property	Configure these properties
Welcome	SkipBDDWelcome	
Specify credentials for connecting to network shares	Skipped by providing properties in next column	<ul style="list-style-type: none"> • UserID • UserDomain • UserPassword
Task Sequence	SkipTaskSequence	<ul style="list-style-type: none"> • TaskSequenceID
Computer Details	SkipComputerName, SkipDomainMembership	<ul style="list-style-type: none"> • OSDComputerName • JoinWorkgroup —or— • JoinDomain • DomainAdmin
User Data	SkipUserData	<ul style="list-style-type: none"> • UDDir • UDShare • UserDataLocation

Skip this wizard page	Using this property	Configure these properties
Move Data and Settings	SkipUserData	<ul style="list-style-type: none"> • UDDir • UDShare • UserDataLocation
User Data (Restore)	SkipUserData	<ul style="list-style-type: none"> • UDDir • UDShare • UserDataLocation
Computer Backup	SkipComputerBackup	<ul style="list-style-type: none"> • BackupDir • BackupShare • ComputerBackupLocation
Product Key	SkipProductKey	<ul style="list-style-type: none"> • ProductKey —or— • OverrideProductKey
Language Packs	SkipPackageDisplay	<ul style="list-style-type: none"> • LanguagePacks
Locale and Time	SkipLocaleSelection, SkipTimeZone	<ul style="list-style-type: none"> • KeyboardLocale • UserLocale • UILanguage • TimeZoneName
Roles and Features	SkipRoles	<ul style="list-style-type: none"> • OSRoles • OSRoleServices • OSFeatures
Applications	SkipApplications	<ul style="list-style-type: none"> • Applications
Administrator Password	SkipAdminPassword	<ul style="list-style-type: none"> • AdminPassword
Local Administrators	SkipAdminAccounts	<ul style="list-style-type: none"> • Administrators
Capture Image	SkipCapture	<ul style="list-style-type: none"> • ComputerBackupLocation
Bitlocker	SkipBitLocker	<ul style="list-style-type: none"> • BDEDriveLetter • BDEDriveSize • BDEInstall • BDEInstallSuppress • BDERecoveryKey • TPMOwnerPassword • OSDBitLockerStartupKeyDri

Skip this wizard page	Using this property	Configure these properties
		<ul style="list-style-type: none">• <code>OSDBitLockerWaitForEncryption</code>
Ready to begin	<code>SkipSummary</code>	—
Operating system deployment completed successfully	<code>SkipFinalSummary</code>	—
Operating system deployment did not complete successfully	<code>SkipFinalSummary</code>	—

Scripts

The scripts used in LTI and ZTI deployments reference properties that determine the process steps and configuration settings used during the deployment process. Use this reference section to help it determine the correct scripts to include in actions and the valid arguments to provide when running each script. The following information is provided for each script:

- **Name.** Specifies the name of the script.
- **Description.** Provides a description of the purpose of the script and any pertinent information regarding script customization.
- **Input.** Indicates the files used for input to the script.
- **Output.** Indicates the files created or modified by the script.
- **References.** Indicates other scripts or configuration files that are referenced by the script.
- **Location.** Indicates the folder where the script can be found. In the information for the location, the following variables are used:
 - **program_files.** This variable points to the location of the Program Files folder on the computer where MDT is installed.
 - **distribution.** This variable points to the location of the Distribution folder for the deployment share.
 - **platform.** This variable is a placeholder for the operating system platform (x86 or x64).
- **Use.** Provides the commands and options that you can specify.
- **Arguments and description.** Indicate the valid arguments to be specified for the script and a brief description of what each argument means.
- **Properties.** The properties referenced by the script.

BDD_Autorun.wsf

This script displays a dialog box that indicates the user inserted deployment media created by the MDT process (such as a bootable DVD or a removable hard disk). The message is displayed for 15 seconds. If no action is taken, the script starts LiteTouch.vbs.

For more information about LiteTouch.vbs, see the corresponding topic in [Scripts](#).

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information required by the

Value	Description
	scripts to complete the deployment process
Output	None
References	LiteTouch.vbs. Initiates LTI
Location	<i>distribution\Scripts</i>
Use	None

Arguments

Value	Description
None	None

Properties

Name	Read	Write
None		

BDD_Welcome_ENU.xml

This XML file contains the script code and HTML layout for the **Welcome to Windows Deployment** page that is displayed at the start of the Deployment Wizard. This XML file is read by Wizard.hta, which runs the wizard pages embedded in this XML file.

Value	Description
Input	None
Output	None
References	<ul style="list-style-type: none"> NICSettings_Definition_ENU.xml. Allows the user to provide configuration settings for network adapters Wizard.hta. Displays the Deployment Wizard pages WPEUtil.exe. Initializes Windows PE and network connections; initiates LTI
Location	<i>distribution\Tools\platform</i>
Use	<code>mshta. exe Wizard. hta BDD_Welcome_ENU. xml</code>

Arguments

Value	Description

Value	Description
None	None

Properties

Name	Read	Write
KeyboardLocalePE	●	
WelcomeWizardCommand		●
WizardComplete		●

Credentials_ENU.xml

This XML file contains the script code and HTML layout for the **Specify credentials for connecting to network shares** wizard page in the Deployment Wizard. This XML file is read by Wizard.hta, which runs the wizard pages embedded in this XML file.

Note This wizard page is only displayed if there is a failure while validating the predefined user credentials.

Value	Description
Input	None
Output	None
References	Credentials_scripts.vbs. Contains user credential support functions
Location	<i>distribution\Scripts</i>
Use	<code>mshta.exe Wizard.hta /NotWizard /definition:Credentials_ENU.xml [/ValidateAgainstDomain: domain /ValidateAgainstUNCPath: uncpath] </DoNotSave> </LeaveShareOpen></code>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
None		

Credentials_scripts.vbs

This script parses the arguments that were provided when loading the Credentials_ENU.xml file into the Deployment Wizard. It also performs user credential validation. This script is read by the Credentials_ENU.xml file.

For more information about Credentials_ENU.xml, see the corresponding topic in [Scripts](#).

Value	Description
Input	None
Output	Event message are written to these log files: <ul style="list-style-type: none"> Credentials_scripts.log. Log file that contains events generated by this script BDD.log. Log file that contains events generated by all MDT scripts
References	None
Location	<i>distribution\Scripts</i>
Use	<script language="VBScript" src="Credentials_scripts.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
UserCredentials		●
UserDomain	●	

DeployWiz_Definition_ENU.xml

This XML file contains the script code and HTML layout for each wizard page in the Deployment Wizard. This file is read by Wizard.hta, which runs the wizard pages embedded in this XML file. This .xml file contains the following wizard pages:

- **Welcome**
- **Specify credentials for connecting to network shares**
- **Task Sequence**

- Computer Details
- User Data
- Move Data and Settings
- User Data (Restore)
- Computer Backup
- Product Key
- Language Packs
- Locale and Time
- Roles and Features
- Applications
- Administrator Password
- Local Administrators
- Capture Image
- BitLocker
- Ready to Begin

Value	Description
Input	None
Output	None
References	<ul style="list-style-type: none"> • DeployWiz_Initialization.vbs. Includes support functions and subroutines used by the script • DeployWiz_Validation.vbs. Includes support functions and subroutines used by the script • ZTIBackup.wsf. Creates a backup of the target computer • ZTIPatches.wsf. Installs updates (language packs, security updates, and so on) • ZTIUserState.wsf. Initializes user state migration to capture and restore user state on the target computer
Location	<i>distribution\Scripts</i>
Use	None

Arguments

Value	Description

Value	Description
None	None

Properties

Name	Read	Write
DeploymentMethod	●	
DeploymentType	●	
DoCapture	●	
ImageBuild	●	
ImageFlags	●	
IsBDE	●	
IsServerOS	●	
JoinDomain	●	
OSDComputerName	●	
OSVersion	●	
SkipAdminAccounts	●	
SkipAdminPassword	●	
SkipApplications	●	
SkipBitLocker	●	
SkipCapture	●	
SkipComputerBackup	●	
SkipComputerName	●	
SkipDomainMembership	●	
SkipLocaleSelection	●	
SkipPackageDisplay	●	
SkipProductKey	●	
SkipRoles	●	
SkipSummary	●	
SkipTaskSequence	●	
SkipTimeZone	●	
SkipUserData	●	
TaskSequenceTemplate	●	

Name	Read	Write
UserDomain	●	
UserID	●	
UserPassword	●	
USMTOfflineMigration	●	

DeployWiz_Initialization.vbs

This script initializes the pages in the **Deployment Wizard** (stored in [DeployWiz_Definition_ENU.xml](#)). It also contains functions and subroutines that the Deployment Wizard calls during an LTI deployment.

Value	Description
Input	<ul style="list-style-type: none"> • DomainOUList.xml. Contains a list of domain OUs • ListOfLanguages.xml • LocationServer.xml. Contains a list of available deployment shares • Environment variables. Contains the list of property values, custom properties, database connections, deployment rules, and other information that the scripts require to complete the deployment process; the environment variables are populated by ZTIGather.wsf
Output	<p>Event message are written to these log files:</p> <ul style="list-style-type: none"> • DeployWiz_Initialization.log. Log file that contains events generated by this script • BDD.log. Log file that contains events generated by all MDT scripts
References	ZTIApplications.wsf . Initiates application installation
Location	<i>distribution\Scripts</i>
Use	<script language="VBScript" src="DeployWiz_Initialization.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
Architecture	●	
Applications	●	
BackupDir	●	
BackupFile	●	
BackupShare	●	
BDEInstall	●	
BDEKeyLocation	●	
BDERecoveryKey	●	
BDEWaitForEncryption	●	
CapableArchitecture	●	
ComputerBackupLocation	●	
CustomWizardSelectionProfile	●	
DeploymentType	●	
DeployRoot	●	
DomainAdmin	●	
DomainAdminDomain	●	
DomainAdminPassword	●	
DomainOUs	●	
ImageBuild	●	
ImageFlags	●	
ImageLanguage	●	
ImageLanguage001	●	
ImageProcessor	●	
IsServerOS	●	
KeyboardLocale	●	
KeyboardLocale_Edit	●	
LanguagePacks	●	
LanguagePacks001	●	
LocalDeployRoot	●	
MandatoryApplications	●	
OSDComputerName	●	

Name	Read	Write
OSCurrentBuild	●	
OSDBitLockerCreateRecoveryPassword	●	
OSDBitLockerMode	●	
OSDBitLockerStartupKeyDrive	●	
OSDBitLockerWaitForEncryption	●	
OSSKU	●	
OSVersion	●	
OverrideProductKey	●	
ProductKey	●	
SkipCapture	●	
SkipDomainMembership	●	
TaskSequenceID	●	
TimeZoneName	●	
TSGUID	●	
UDDir	●	
UDShare	●	
UILanguage	●	
UserDataLocation	●	
UserDomain	●	
UserID	●	
UserLocale	●	
UserPassword	●	
WizardSelectionProfile	●	

DeployWiz_Validation.vbs

This script initializes and validates the information typed in the pages of the Deployment Wizard (stored in [DeployWiz_Definition_ENU.xml](#)). This script contains functions and subroutines that the Deployment Wizard calls during an LTI deployment.

Value	Description
Input	<ul style="list-style-type: none"> ● OperatingSystems.xml. Contains the list of operating systems available for deployment

Value	Description
	<ul style="list-style-type: none"> Environment variables. Contains the list of property values, custom properties, database connections, deployment rules, and other information required by the scripts to complete the deployment process; the environment variables are populated by ZTIGather.wsf
Output	None
References	<ul style="list-style-type: none"> Credentials_ENU.xml. Prompts the user for credentials that will be used when connecting to network resources ZTIGather.wsf. Gathers properties and processing rules
Location	<i>distribution\Scripts</i>
Use	<script language="VBScript" src="DeployWiz_Val idation.vbs" />

Arguments

Value	Description
None	None

Properties

Name	Read	Write
Architecture	●	
DeploymentType	●	●
DeployTemplate		●
ImageBuild	●	
ImageProcessor	●	●
OSVersion	●	
TaskSequenceID		●
TSGUID	●	
UserCredentials	●	
UserDomain		●
UserID		●
UserPassword		●

LiteTouch.vbs

This script is called by the Deployment Wizard to initiate LTI. The script:

- Removes the C:\MININT folder (if it exists)
- Checks that the target computer meets the requirements for running the Deployment Wizard by calling [ZTIPrereq.vbs](#)
- Starts the Deployment Wizard by running [LiteTouch.wsf](#)

Value	Description
Input	None
Output	None
References	<ul style="list-style-type: none"> • BDDRun.exe • ZTIPrereq.vbs. Used to determine whether the target computer meets the prerequisites for deploying a new operating system • LiteTouch.wsf. The script responsible for controlling the LTI deployment process
Location	<i>distribution\Scripts</i>
Use	<code>cscript LiteTouch.vbs </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
None		

LiteTouch.wsf

This script is called by [LiteTouch.vbs](#) and is responsible for controlling the LTI deployment process. This includes:

- Running the Deployment Wizard
- Running the LTI deployment process by using the appropriate task sequence file

Value	Description
Input	<ul style="list-style-type: none">• task_sequence_file.xml. Contains the tasks and sequence of tasks for the LTI deployment process• Environment variables. Contains the list of property values, custom properties, database connections, deployment rules, and other information required by the scripts to complete the deployment process; the environment variables are populated by ZTIGather.wsf
Output	<ul style="list-style-type: none">• LiteTouch.log. Log file that contains events that this script generates• BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none">• BDD_Welcome_ENU.xml. Displays the Deployment Wizard Welcome page for LTI deployment• DeployWiz_Definition_ENU.xml. Displays the Deployment Wizard pages for LTI deployment• Diskpart.exe. Utility that allows the automated management of disks, partitions, and volumes• LТИCleanup.wsf. Performs cleanup tasks after deployment finishes• LТИCopyScripts.wsf. Copies the deployment scripts to a local hard drive on the target computer• MSHTA.exe. HTML application host• RecEnv.exe. If this utility exists, the user is prompted to determine whether to launch Windows Recovery Environment.• Regsvr32.exe. Registers files (.dll, .exe, .ocx, and so on) with the operating system• Summary_Definition_ENU.xml. Displays the summary results for the LTI deployment• TsmBootBootstrap.exe. Task sequence Bootstrap utility• Wizard.hta. Displays the Deployment Wizard pages• WPEUtil.exe. Initializes Windows PE and network connections; initiates LTI• ZTIGather.wsf. Gathers properties and processing rules

Value	Description
	<ul style="list-style-type: none"> • ZTIPrereq.vbs. Checks that the target computer meets the requirements for running the Deployment Wizard • ZTINICConfig.wsf. Configures activated network adapters • ZTIUtility.vbs. Includes support functions and subroutines the script uses
Location	<i>distribution\Scripts</i>
Use	<code>BDDRun. exe "wscript. exe <ScriptDirectory>\LiteTouch. wsf </debug: value>"</code>

Arguments

Value	Description
<code>/debug:value</code>	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)
<code>/Start</code>	Creates a shortcut in the new operating system that runs once the shell starts

Properties

Name	Read	Write
<code>_DoNotCleanLiteTouch</code>	●	
<code>_SMSTSPackageName</code>		●
<code>AdminPassword</code>	●	
<code>Architecture</code>	●	●
<code>BootPE</code>	●	●
<code>ComputerBackupLocation</code>		●
<code>ComputerName</code>	●	
<code>DeployDrive</code>	●	●
<code>DeploymentMethod</code>	●	●
<code>DeploymentType</code>	●	●

Name	Read	Write
DeployRoot	●	●
DestinationLogicalDrive		●
DomainAdmin		●
DomainAdminDomain		●
DomainAdminPassword		●
FinishAction	●	
HostName	●	
IsServerCoreOS	●	
JoinDomain	●	
JoinWorkgroup	●	●
KeyboardLocalePE	●	
LTI_suspend	●	
OSDAdapterCount	●	
OSDComputerName	●	●
Phase	●	●
ResourceDrive	●	●
ResourceRoot	●	●
RetVal		●
SkipBDDWelcome	●	
SkipFinalSummary	●	●
SkipWizard	●	
SMSTSLocalDataDrive		●
TaskSequenceID	●	
TimeZoneName		●
UserDataLocation	●	●
UserDomain	●	
UserID	●	
UserPassword	●	
WelcomeWizardCommand	●	
WizardComplete	●	

LTIApply.wsf

This script is responsible for installing a Windows PE image to the target computer. The Windows PE image is used to collect information about the target computer and to run the deployment tasks on the target computer.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> LTIApply.log. Log file that contains events that this script generates LTIApply_wdsmdcast.log. Log file that contains events that the Wdsmdcast utility generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> CMD.exe. Allows the running of command-line tools Bootsect.exe. Applies a boot sector to the hard disk ImageX.exe. A utility used to create and manage WIM files ZTIBCDUtility.vbs. Includes utility functions used when performing Boot Manager tasks ZTIConfigFile.vbs. Includes routines for processing XML files ZTIDiskUtility.vbs. Includes support functions and subroutines the script uses ZTIUtility.vbs. Includes support functions and subroutines the script uses Wdsmdcast.exe. A utility that target computers use to join a multicast transmission
Location	<i>distribution\Scripts</i>
Use	<code>cscript LTIApply.wsf </pe> </post> </debug: value></code>

Arguments

Value	Description
/pe	Uses the process for installing the Windows PE image on the target computer
/post	Cleans up unnecessary files after the installation of an

Value	Description
	image
/debug:value	Outputs the event messages to the console and to the .log files; if the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
Architecture	●	
BootPE		●
DeployRoot	●	
DestinationLogicalDrive	●	●
OSGUID	●	
OSCurrentVersion	●	
OSVersion	●	
ImageBuild	●	
ImageFlags	●	
ImageProcessor	●	
ISBDE	●	
SourcePath		●
TaskSequenceID	●	
UserDomain	●	
UserID	●	
UserPassword	●	
WDSServer	●	

LTCleanup.wsf

This script removes any files or configuration settings (such as scripts, folders, registry entries, or automatic logon configuration settings) from the target computer after the deployment process finishes.

Value	Description
Input	Environment variables. Contains the list of property values, custom properties, database connections, deployment rules, and other information that the scripts require to complete the deployment process. The environment variables are populated by ZTIGather.wsf.
Output	<ul style="list-style-type: none"> LTIcleanup.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> Bootsect.exe. Applies a boot sector to the hard disk Net.exe. Performs network management tasks RegSvr32.exe. Registers files (.dll, .exe, .ocx, and so on) with the operating system ZTIBCDUtility.vbs. Includes utility functions used when performing Boot Manager tasks ZTUtility.vbs. Includes support functions and subroutines the script uses
Location	<i>distribution\Scripts</i>
Use	<code>script LTIcleanup.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
_DoNotCleanLiteTouch	●	
DeployRoot	●	
DestinationLogicalDrive	●	
OSVersion	●	

LTIcopyScripts.wsf

This script copies the deployment scripts for the LTI and ZTI deployment processes to a local hard drive on the target computer.

Value	Description
Input	<ul style="list-style-type: none"> Summary_Definition_ENU.xml. Displays the summary results for the LTI deployment Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> LTIcopyScripts.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	ZTUtility.vbs. Includes support functions and subroutines the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript LTIcopyScripts.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
None		

LTIGetFolder.wsf

This script displays a dialog box that allows the user to browse to a folder. The selected folder path is stored in the FOLDERPATH environment variable.

Value	Description
Input	Environment variables. Contains the list of property values, custom properties, database connections, deployment rules, and other information that the scripts require to complete the deployment process. The environment variables are populated by ZTIGather.wsf.
Output	None
References	<ul style="list-style-type: none"> • ZTIUtility.vbs. Includes support functions and subroutines that the script uses • WizUtility.vbs. Includes support functions and subroutines that the UI uses (such as wizard pages)
Location	<ul style="list-style-type: none"> • <i>distribution\Scripts</i> • <i>program_files\Microsoft Deployment Toolkit\Scripts</i>
Use	<code>cscript LTI GetFolder.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
DefaultFolderPath	●	
FolderPath		●

LTIOEM.wsf

This script is used by an OEM during an LTI OEM scenario to copy the contents of a media deployment share to the target computer's hard disk to prepare it for duplication.

Value	Description
Input	Environment variables. Contains the list of property

Value	Description
	values, custom properties, database connections, deployment rules, and other information that the scripts require to complete the deployment process. The environment variables are populated by ZTIGather.wsf.
Output	<ul style="list-style-type: none"> LTI OEM.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> RoboCopy.exe. File and folder copy tool ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript LTI OEM.wsf </BITLOCKER /BDE> </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)
/BITLOCKER	Enables BitLocker
/BDE	Enables BitLocker

Properties

Name	Read	Write
_DoNotCleanLiteTouch		●
DeployDrive	●	
DeployRoot	●	
TSGUID	●	

LTI_suspend.wsf

This script suspends a task sequence to allow manual tasks to be performed. When this script runs, it creates a **Resume Task Sequence** shortcut on the user's desktop that allows the user to restart the task sequence after all manual tasks are completed.

Note This script is only supported while in the full operating system.

Value	Description
Input	Environment variables. Contains the list of property values, custom properties, database connections, deployment rules, and other information the scripts require to complete the deployment process. The environment variables are populated by ZTIGather.wsf.
Output	<ul style="list-style-type: none"> LTI_suspend.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> LiteTouch.wsf. Controls the LTI deployment process LTIcopyScripts.wsf. Copies the deployment scripts to a local hard drive on the target computer ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript LTI_Suspend.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)
/Resume	–

Properties

Name	Read	Write
------	------	-------

Name	Read	Write
LTI_suspend		●
SMSTSRebootRequested		●

LTI_Sysprep.wsf

This script prepares the target computer for running Sysprep, runs Sysprep on the target computer, and then verifies that Sysprep ran successfully.

Value	Description
Input	Environment variables. Contains the list of property values, custom properties, database connections, deployment rules, and other information that the scripts require to complete the deployment process. The environment variables are populated by ZTIGather.wsf.
Output	<ul style="list-style-type: none"> • LTI_Sysprep.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • Expand.exe. Expands compressed files • Sysprep.exe. Prepares computers for duplication • ZTIConfigFile.vbs. Contains routines for processing XML files • ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript LTI_Sysprep.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
Architecture	●	
DeployRoot	●	
DestinationLogicalDrive	●	
DoCapture	●	
OSCurrentBuild	●	
OSDAnswerFilePath	●	
OSGUID	●	
SourcePath	●	●
TaskSequenceID	●	

NICSettings_Definition_ENU.xml

This XML file contains the script code and HTML layout for the **Configure Static IP Network Settings** wizard page in the Deployment Wizard. During an LTI deployment, Wizard.hta reads this file and runs the embedded wizard page that prompts for the required network addressing configuration. If no static IP addressing configuration is supplied, the deployment scripts will default to using DHCP to obtain the required network configuration.

Value	Description
Input	None
Output	None
References	ZTINICUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	None

Arguments

Value	Description
None	None

Properties

Name	Read	Write
OSDAdapterxDNSServerList		●

Name	Read	Write
OSDAdapterxDNSSuffix		●
OSDAdapterxGateways		●
OSDAdapterxIPAddressList		●
OSDAdapterxMacAddress		●
OSDAdapterxSubnetMask		●
OSDAdapterxWINSServerList		●
OSDAdapterCount		●

Note The *x* in the property names listed above is a placeholder for a zero-based array that contains network adapter information.

Summary_Definition_ENU.xml

This XML file contains the script code and HTML layout for the **Deployment Summary** wizard page in the Deployment Wizard. During an LTI deployment, Wizard.hta reads this file and runs the embedded wizard page that displays the summary results for the LTI deployment. This XML file contains the following wizard pages:

- **Success.** Notification regarding the successful completion of the deployment tasks
- **Failure.** Notification regarding the failure to successfully complete the deployment tasks

Value	Description
Input	None
Output	None
References	Summary_Scripts.vbs. Includes support functions and subroutines that the wizard pages embedded in this XML file use
Location	<i>distribution\Scripts</i>
Use	None

Arguments

Value	Description
None	None

Properties

Name	Read	Write
SkipFinalSummary	●	
RetVal	●	

Summary_scripts.vbs

This script is called by the **Summary** wizard page of the Deployment Wizard. It contains functions and subroutines used for initialization and validation.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	Event message are written to these log files: <ul style="list-style-type: none"> • Summary_scripts.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	None
Location	<i>distribution\Scripts</i>
Use	<script language="VBScript" src="Summary_Scripts.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
DeploymentType	●	
RetVal	●	

Wizard.hta

This Hypertext Application displays the Deployment Wizard pages.

Value	Description
Input	Environment variables. Contains the list of property values, custom properties, database connections, deployment rules, and other information that the scripts require to complete the deployment process. The environment variables are populated by ZTIGather.wsf.
Output	<ul style="list-style-type: none"> • Wizard.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • LTIGetFolder.wsf. Script file that initiates a BrowseForFolder dialog box • ZTIConfigFile.vbs. Includes routines for processing XML files • ZTIUtility.vbs. Includes support functions and subroutines that the script uses • WizUtility.vbs. Includes support functions and subroutines that the script uses
Location	<ul style="list-style-type: none"> • <i>distribution\Scripts</i> • <i>program_files\Microsoft Deployment Toolkit\Scripts</i>
Use	<pre>mshta.exe Wizard.htm </definition:filename> </NotWizard> </debug:value></pre>

Arguments

Value	Description
/debug:value	<p>Outputs the event messages to the console and to the .log files. If the value specified in value is:</p> <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/NotWizard	Used to bypass wizard page prompts
/Definition:filename	Specifies the XML file that is to be loaded into the wizard

Properties

Name	Read	Write
Definition	●	

Name	Read	Write
DefaultFolderPath		●
FolderPath	●	
WizardComplete		●

WizUtility.vbs

This script contains functions and subroutines that the various Deployment Wizard scripts reference.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • WizUtility.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	LTIGetFolder.wsf. Script file that initiates a BrowseForFolder dialog box
Location	<ul style="list-style-type: none"> • <i>distribution\Scripts</i> • <i>program_files\Microsoft Deployment Toolkit\Scripts</i>
Use	<script language="VBScript" src="WizUtility.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
DefaultFolderPath		●
DefaultDestinationDisk	●	
DefaultDestinationIsDirty	●	
DefaultDestinationPartition	●	
DeploymentType	●	

Name	Read	Write
DestinationDisk	●	
FolderPath	●	
OSVersion	●	
UserDomain	●	
UserCredentials		●

ZTIApplications.wsf

This script initiates an installation of applications that have been configured in the Applications node in Deployment Workbench. This script will not attempt to install any application that:

- Does not support the target computer's platform type
- Does not support the target computer's processor type
- Has an uninstall entry in the registry under
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall

Note If the listed application has any dependent applications defined, this script attempts to install those dependent applications before installing the listed application.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIApplications.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • ZTIConfigFile.vbs. Includes routines for processing XML files • ZTIUtility.vbs. Includes support functions and subroutines that the script uses • BDDRun.exe. Runs a command that requires user interaction
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI Applications.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
ApplicationGUID	●	
ApplicationSuccessCodes	●	
DependentApplications	●	
DeploymentMethod	●	
InstalledApplications	●	●
ResourceDrive	●	
ResourceRoot	●	●
SMSTSRebootRequested		●
SMSTSRetryRequested		●

ZTIAppXmlGen.wsf

This script generates an XML file—ZTIAppXmlGen.xml—to use when automatically capturing user data (documents) associated with installed applications. It does so through the **HKEY_CLASSES_ROOT\Software\Classes** registry key and captures any applications that:

- Are not associated with one of these file extensions: .mp3, .mov, .wma, .wmv, .chm, .evt, .evtx, .exe, .com, or .fon
- Are not associated with Microsoft Office, such as the 2007 Office system or Microsoft Office 2003.
- Have a valid open handler listed at
HKEY_CLASSES_ROOT\application\shell\open\command

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections,

Value	Description
	deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIAppXmlGen.xml. Contains a list of applications installed on the target computer ZTIAppXmlGen.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI AppXml Gen.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
DeploymentMethod	●	
DeploymentType	●	
ImageBuild	●	
OSCurrentVersion	●	
USMTMigFiles	●	●

ZTIAuthorizeDHCP.wsf

This script uses the Netsh tool to configure the target computer so that it is an authorized DHCP server in AD DS.

For more information about authorizing DHCP servers, see [How to Use Netsh.exe to Authorize, Unauthorized and List DHCP Servers in Active Directory](#).

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIAuthorizeDHCP.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> Netsh.exe. A utility used to automate the configuration of networking components ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI AuthorizeDHCP.wsf </debug: value></code>

Note The security context that this script runs under must be a member of the Enterprise Admins group.

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
IPAddress	●	

ZTIBackup.wsf

This script performs a backup of the target computer using the ImageX utility. The backup is stored in the location specified in the [BackupDir](#) and [BackupShare](#) properties.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIBackup.log. Log file that contains events that this script generates ZTIBackup_imagem.log. Log file that contains events that ImageX generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> ImageX.exe. A utility used to create and manage WIM files ZTIBCDUtility.vbs. Includes utility functions used when performing Boot Manager tasks ZTIDiskUtility.vbs. Includes support functions and subroutines that the script uses ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI Backup.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
------	------	-------

Name	Read	Write
BackupDir	●	
BackupDisk	●	
BackupDrive	●	
BackupFile	●	
BackupPartition	●	
BackupScriptComplete		●
BackupShare	●	
ComputerBackupLocation	●	
DeploymentMethod	●	
DeploymentType	●	
DestinationLogicalDrive	●	●
DoCapture	●	
ImageBuild	●	
ImageFlags	●	
OSDStateStorePath	●	
Phase	●	
TaskSequenceID	●	
USMTLocal	●	

ZTIBCDUtility.vbs

This script contains utility functions that some MDT scripts use when performing Boot Manager tasks.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	None
References	BCDEdit.exe. A tool for editing the Windows boot configuration
Location	<ul style="list-style-type: none"> • <i>distribution\Scripts</i> • <i>program_files\Microsoft Deployment Toolkit\Scripts</i>

Value	Description
Use	<script language="VBScript" src="ZTIBCDUtility.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
None		

ZTIBde.wsf

This script installs and configures BitLocker on the target computer. BitLocker configuration is limited to New Computer scenarios that have hard disks configured with a single partition.

Note For ZTI and UDI deployments, the **UILanguage** property must be set in CustomSettings.ini or in the MDT DB, because ZTIBde.wsf tries to read the locale from the **UILanguage** property.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIBde.log. Log file that contains events that this script generates ZTIBdeFix_diskpart.log. Log file that contains events that the Diskpart tool generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> CMD.exe. Allows running of command-line tools Defrag.exe. Defragments the hard disk Diskpart.exe. Utility that allows for the automated management of disks, partitions, and volumes ServerManagerCmd.exe ZTIDiskUtility.vbs. Includes support functions and subroutines that the script uses ZTIOSRole.wsf. Installs server roles

Value	Description
	<ul style="list-style-type: none"> • ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI Bde.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	<p>Outputs the event messages to the console and to the .log files. If the value specified in value is:</p> <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
AdminPassword	●	
BDEDriveLetter	●	●
BDEDriveSize	●	
BDEInstall	●	
BDEInstallSuppress	●	
BDEKeyLocation	●	
BDEPin	●	
BDERecoveryKey	●	
BDESecondPass	●	●
BdeWaitForEncryption	●	
BitlockerInstalled	●	●
DeploymentMethod	●	
ISBDE	●	
OSDBitLockerCreateRecoveryPassword	●	
OSDBitLockerMode	●	
OSDBitLockerStartupKey	●	

Name	Read	Write
OSDBitLockerStartupKeyDrive	●	
OSDBitLockerTargetDrive	●	
OSDBitLockerWaitForEncryption	●	
OSCurrentBuild	●	
OSCurrentVersion	●	
OSFeatures	●	●
OSRoles	●	●
OSRoleServices	●	●
OSVersion	●	
SMSTSRebootRequested	●	●
SMSTSRetryRequested		●
TPMOwnerPassword	●	

ZTIBIOSCheck.wsf

This script checks the BIOS on the target computer, and then looks at a list of BIOSes that are incompatible with Windows. The list of incompatible BIOSes is stored in the [ZTIBIOSCheck.xml](#) file.

If the BIOS on the target computer is listed in the [ZTIBIOSCheck.xml](#) file, then the script returns a status that indicates the BIOS is incompatible with Windows and the deployment process should be terminated. For information on populating the list of incompatible BIOSes, see [ZTIBIOSCheck.xml](#).

Value	Description
Input	<ul style="list-style-type: none"> ZTIBIOSCheck.xml. Contains a list of BIOSes that are known to be incompatible with Windows Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIBIOSCheck.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	ZTIUtility.vbs. Includes support functions and subroutines that the script uses

Value	Description
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIBI0SCheck.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (this is the behavior when the argument is not provided)

Properties

Name	Read	Write
None		

ZTICoalesce.wsf

Configuration Manager requires packages to be numbered sequentially starting with **PACKAGES001**, with no gaps in the number sequence. Otherwise, installation will fail.

This script allows you to define and name variables using identifying information about the program to run—for example, **ComputerPackages100**, **ComputerPackages110**, or **CollectionPackages150**. Then, when this script is run, Configuration Manager finds all variables that match a pattern (for example, all variable names that contain the string **Packages**) and builds a sequential list, without gaps, using the base name **PACKAGES**.

For example, if the following variables were defined (using computer variables, collection variables, or in CustomSettings.ini or the MDT DB, for example):

- **ComputerPackages100=XXX00001:Program**
- **ComputerPackages110=XXX00002:Program**
- **CollectionPackages150=XXX00003:Program**
- **Packages001=XXX00004:Program**

After the script runs, the list would be:

- **PACKAGES001=XXX00004:Program**

- **PACKAGES002=XXX00001:Program**
- **PACKAGES003=XXX00002:Program**
- **PACKAGES004=XXX00003:Program**

Configuration Manager would then be able to run all four programs.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTICoalesce.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTICoalesce.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/CoalesceDigits:value	Specifies the number of digits that need to be provided when creating the numbering sequence. For example, a value of: <ul style="list-style-type: none"> • 2 would create PACKAGE03 • 3 would create PACKAGE003 The default value if this argument is not provided is 3 .

Properties

Name	Read	Write
CoalescePattern	●	

Name	Read	Write
CoalesceTarget	●	

ZTIConfigFile.vbs

This script contains common routines for processing MDT XML files.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIConfigFile.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	Net.exe
Location	<i>distribution\Scripts</i>
Use	<script language="VBScript" src="ZTIConfigFile.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
IsSafeForWizardHTML	●	
MandatoryApplications	●	
SkipGroupSubFolders	●	

ZTIConfigure.wsf

This script configures the Unattend.xml file with the property values specified earlier in the MDT deployment process. The script configures the appropriate file based on the operating system being deployed.

This script reads the ZTIConfigure.xml file to determine how to update the Unattend.xml file with the appropriate values specified in the deployment

properties. The ZTIConfigure.xml file contains the information to translate properties to settings in the Unattend.xml file.

Value	Description
Input	<ul style="list-style-type: none"> ZTIConfigure.xml. Contains a list of property values (specified earlier in the deployment process) and their corresponding configuration settings Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIConfigure.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	ZTICUtility.vbs . Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIConfigure.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
ComputerName	●	●
DeploymentType	●	
DeploymentMethod	●	
DeployRoot	●	
DestinationLogicalDrive	●	

Name	Read	Write
DomainAdminDomain	●	
ImageBuild	●	
OSDAnswerFilePath	●	
OSDAnswerFilePathSysprep	●	
OSDComputerName	●	
Phase	●	
TaskSequenceID	●	

ZTIConfigureADDS.wsf

This script starts Dcpromo to configure the target computer as an AD DS domain controller. For more information about Dcpromo.exe, see [Dcpromo](#).

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIConfigureADDS.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> Dcpromo.exe. Installs and removes AD DS Net.exe. Performs network management tasks ZTICUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIConfigureADDS.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log

Value	Description
	files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
ADDSLogPath	●	
ADDSPassword	●	
ADDSUserDomain	●	
ADDSUserName	●	
AutoConfigDNS	●	
ChildName	●	
ConfirmGC	●	
DatabasePath	●	
DomainLevel	●	
DomainNetBiosName	●	
ForestLevel	●	
NewDomain	●	
NewDomainDNSName	●	
OSVersion	●	
ParentDomainDNSName	●	
ReplicaOrNewDomain	●	●
ReplicaDomainDNSName	●	
ReplicationSourceDC	●	
SafeModeAdminPassword	●	
SiteName	●	
SysVolPath	●	

ZTIConfigureDHCP.wsf

This script configures DHCP on the target computer.

Note DHCP should already be installed on the target computer before running this script.

Value	Description

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIConfigureDHCP.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • Netsh.exe. A utility that permits automating the configuration of networking components • ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI ConfigureDHCP.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
DHCPSscopesxDescription	●	
DHCPSscopesxEndIP	●	
DHCPSscopesxExcludeStartIP	●	
DHCPSscopesxExcludeEndIP	●	
DHCPSscopesxIP	●	
DHCPSscopesxName	●	
DHCPSscopesxOptionRouter	●	
DHCPSscopesxOptionDNSDomainName	●	

Name	Read	Write
DHCPScopesxOptionDNSServer	●	
DHCPScopesxOptionLease	●	
DHCPScopesxOptionNBTNodeType	●	
DHCPScopesxOptionPXEClient	●	
DHCPScopesxOptionWINSServer	●	
DHCPScopesxStartIP	●	
DHCPScopesxSubnetmask	●	
DHCPServerOptionDNSDomainName	●	
DHCPServerOptionDNSServer	●	
DHCPServerOptionNBTNodeType	●	
DHCPServerOptionPXEClient	●	
DHCPServerOptionRouter	●	
DHCPServerOptionWINSServer	●	

Note The *x* in the properties listed here is a placeholder for a zero-based array that contains DHCP configuration information.

ZTIConfigureDNS.wsf

This script configures DNS on the target computer. To perform the actual configuration tasks, the script uses the Dnscmd utility.

For more information about Dnscmd.exe, see [Dnscmd Overview](#).

Note DNS should already be installed on the target computer before running this script.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIConfigureDNS.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> Dnscmd.exe. Assists administrators with DNS management ZTUtility.vbs. Includes support functions and subroutines that the script uses

Value	Description
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI ConfigureDNS.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
DNSServerOptionDisableRecursion	●	
DNSServerOptionBINDSecondaries	●	
DNSServerOptionFailOnLoad	●	
DNSServerOptionEnableRoundRobin	●	
DNSServerOptionEnableNetmaskOrdering	●	
DNSServerOptionEnableSecureCache	●	
DNSServerOptionNameCheckFlag	●	
DNSZonesxName	●	
DNSZonesxType	●	
DNSZonesxMasterIP	●	
DNSZonesxDirectoryPartition	●	
DNSZonesxFileName	●	
DNSZonesxScavenge	●	
DNSZonesxUpdate	●	

Note The x in the properties listed here is a placeholder for a zero-based array that contains DNS configuration information.

ZTIConnect.wsf

The MDT deployment process uses this script to authenticate with a server computer (such as a computer running SQL Server or another server that has a shared network folder). When this script is run, it validates that a connection can be created to the network shared folder specified in the **/uncpath** argument.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIConnect.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI Connect.wsf /UNCPath: <uncpath> </debug: value></code>

Arguments

Value	Description
/UNCPath:uncpath	Specifies a fully qualified UNC path to a network shared folder
/debug:value	Outputs the event messages to the console and to the .log files; if the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
None		

ZTICopyLogs.wsf

Copy the Smsts.log and BDD.log files to a subfolder beneath the share that the **SLShare** property specifies. The subfolder takes the name that **OSDComputerName**, **_SMSTSMachineName**, or **HostName** specifies.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTICopyLogs.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	ZTICopyLogs.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTICopyLogs.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
None		

ZTIDataAccess.vbs

This script contains common routines for database access.

Value	Description

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIDataAccess.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	None
Location	<i>distribution\Scripts</i>
Use	<script language="VBScript" src="ZTI DataAccess.vbs" />

Arguments

Value	Description
None	None

Properties

Name	Read	Write
_SMSTSReserved1	●	
_SMSTSReserved2	●	
RulesFile	●	
UserDomain	●	●
UserID	●	●
UserPassword	●	●

ZTIDisableBDEProtectors.wsf

If BitLocker is enabled, this script suspends the BitLocker protectors configured on the system.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process

Value	Description
Output	<ul style="list-style-type: none"> ZTIDisableBDEProtectors.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI DisableBDEProtectors.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	<p>Outputs the event messages to the console and to the .log files. If the value specified in value is:</p> <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
ImageBuild	●	
ISBDE		●
OSCurrentBuild	●	
OSCurrentVersion	●	
OSVersion	●	

ZTIDiskpart.wsf

This script creates the disk partitions on the target computer by calling the Diskpart utility. The parameters used to configure the disk are specified by the Task Sequencer or in CustomSettings.ini. ZTIDiskpart.wsf is primarily run in New Computer scenarios. The process works like this:

1. The MDT deployment process runs the ZTIDiskpart.wsf script based on the steps and sequence of steps in the Task Sequencer.

2. ZTIDiskpart.wsf starts the Diskpart utility and sends it the required configuration commands.
3. ZTIDiskpart.wsf runs Diskpart.exe and provides a .txt file as a command-line parameter.
4. The disk is initially cleaned by sending Diskpart the **CLEAN** command.
5. If this is the first disk and no disk configuration has been specified by the Task Sequencer or in CustomSettings.ini, a single partition is created to store the operating system. However, if a disk configuration has been specified, the disk will be configured according to the specified configuration.
6. If BitLocker is to be enabled, space is reserved at the end of the first disk.
7. All format commands are queued until after Diskpart has finished. If not explicitly specified by the Task Sequencer or in CustomSettings.ini, ZTIDiskpart.wsf performs a quick format of drive C using the following command: **FORMAT C: /FS: NTFS /V: OSDisk /Q /Y**.
8. ZTIDiskpart.wsf copies the ZTIDiskpart_diskpart.log and BDD.log files from the RAM disk back to the hard drive.

Customize the disk configuration of the target computer by providing the required information in the Task Sequencer or in CustomSettings.ini.

For more information about configuring disks, see the MDT document *Using the Microsoft Deployment Toolkit*.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIDiskpart.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • Diskpart.exe. Utility that allows for the automated management of disks, partitions, and volumes • Format.com. Formats the hard disk • ZTIDiskUtility.vbs. Includes support functions and subroutines that the script uses • ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIDiskpart.wsf </debug: value></code>

Arguments

Value	Description
/debug: <i>value</i>	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
BDEDriveLetter	●	
BDEDriveSize	●	
BDEInstall	●	
DeployDrive	●	
DeploymentType	●	
DestinationDisk	●	
DestinationLogicalDrive		●
DoNotCreateExtraPartition	●	
ImageBuild	●	
OSDDiskIndex	●	
OSDDiskpartBiosCompatibilityMode	●	●
OSDDiskType	●	
OSDPartitions	●	
OSDPartitionStyle	●	
SMSTSLocalDataDrive		●
VolumeLetterVariable	●	

ZTIDiskUtility.vbs

This script contains disk-related functions and subroutines that the various scripts in the MDT deployment process call.

Value	Description

Value	Description
Input	None
Output	<ul style="list-style-type: none"> ZTIDiskUtility.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> BcdBoot.exe. Configures the system partition DiskPart.exe. Utility that allows for the automated management of disks, partitions, and volumes
Location	<i>distribution\Scripts</i>
Use	<script language="VBScript" src="ZTI DiskUtility.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
DestinationLogicalDrive	●	
UILanguage	●	●

ZTIDomainJoin.wsf

During the State Restore deployment phase, this script verifies that the computer is joined to a domain and recovers from failed attempts to join a domain.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIDomainJoin.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> LTSuspend.wsf ZTIUtility.vbs. Includes support functions and

Value	Description
	subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI DomainJoin.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/DomainErrorRecovery: value	Attempts to join the computer to the domain. If the value specified in value is: <ul style="list-style-type: none"> AUTO. Retry the domain join process. Restart and retry. This is the default script behavior. FAIL. Stops all processing. All task sequence processing stops. MANUAL. Stop processing; allows the user to manually join the computer to the domain.

Properties

Name	Read	Write
DomainAdmin	●	
DomainAdminDomain	●	
DomainAdminPassword	●	
DomainErrorRecovery	●	
DomainJoinAttempts	●	●
JoinDomain	●	
JoinWorkgroup	●	
LTI_suspend		●
MachineObjectOU	●	
SMSTSRebootRequested		●
SMSTSRetryRequested		●

ZTIDrivers.wsf

This script installs additional device drivers onto the target computer before initiating the configuration of the operating system. This script reads the Drivers.xml file and copies the list of device driver files in the Drivers.xml file (created by and managed in the Drivers node in the Deployment Workbench) to the target computer.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> PnpEnum.xml. Contains a list of all devices installed on the target computer ZTIDrivers.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> Attrib.exe. Sets file and folder attributes CMD.exe. Allows running of command-line tools Microsoft.BDD.PnpEnum.exe. Utility that enumerates Plug and Play devices Reg.exe. The console registry tool for reading and modifying registry data ZTIConfigFile.vbs. Includes routines for processing XML files ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIDrivers.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log

Value	Description
	files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
Architecture	●	
CustomDriverSelectionProfile	●	
DeploymentMethod	●	
DeploymentType	●	
DestinationLogicalDrive	●	●
DoCapture	●	
DriverPaths	●	
DriverSelectionProfile	●	
ImageBuild	●	
InstallFromPath	●	
OSDAnswerFilePath	●	
OSDAnswerFilePathSysPrep	●	
OSDPlatformArch	●	
Phase	●	
ResourceRoot	●	

ZTIEexecuteRunbook.wsf

This script runs Orchestrator runbooks on the target computer. An Orchestrator *runbook* is the sequence of activities that orchestrate actions on computers and networks. You can initiate Orchestrator runbooks in MDT using the [Execute Runbook](#) task sequence step type, which in turn runs this script.

Value	Description
Input	Environment variables contain the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process.
Output	<ul style="list-style-type: none"> BDD.log contains events that all MDT scripts generate. Return status of the runbook completion.

Value	Description
	<ul style="list-style-type: none"> Return parameters from the runbook output.
References	<ul style="list-style-type: none"> ZTIUtility.vbs includes support functions and subroutines that the script uses.
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI ExecuteRunbook.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	<p>Outputs the event messages to the console and to the .log files. If the value specified in value is:</p> <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
OrchestratorServer	●	
RunbookName	●	
RunbookID	●	
RunbookParameterMode	●	
RunbookParametersxParameterID	●	
RunbookParametersxParameterName	●	
RunbookParametersxParameterValue	●	
RunbookOutputParameters		●
Note If a runbook returns output parameters, a task sequence variable is created for each parameter and the return value of the parameter is assigned to the task sequence variable.		

This script creates the task sequence variables listed in the following table for internal script use. Do not set these task sequence variables in CustomSettings.ini or in the MDT DB.

Name	Description
OrchestratorServer	Name of the server running Orchestrator specified in

Name	Description
	Orchestrator Server in the Execute Runbook task sequence step
RunbookName	Name of the runbook specified in Runbook in the Execute Runbook task sequence step
RunbookID	Identifier assigned to the runbook on the Orchestrator server
RunbookParametersxParameterID	Identifier assigned to a specific runbook parameter on the Orchestrator server
RunbookParametersxParameterName	Name assigned to a specific runbook parameter on the Orchestrator server
RunbookParametersxParameterValue	Value assigned to a specific runbook parameter on the Orchestrator server

ZTIGather.wsf

This script gathers the properties and processing rules that control the deployment process. The properties and rules (also known as *local properties*) are explicitly defined in this script and contained in the ZTIGather.xml file, in the CustomSettings.ini file, and in the MDT DB (created in the Database node in the Deployment Workbench).

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIGather.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> Wpeutil.exe. Initializes Windows PE and network connections; initiates LTI ZTIDataAccess.vbs. Contains routines for database access ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIGather.wsf </debug: value></code>

Value	Description
	</Localonly> </ini file: ini_file_name>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/localonly	Returns only information about the target computer and the current operating system installed on the target computer; does not parse the input .ini file (specified in the /infile argument); returns properties and rules specified in the .ini file If not specified, the script returns information about the target computer and the currently installed operating system; parses the .ini file
/infile:ini_file_name	Name and path of the input .ini file that contains the properties and rules used in the deployment process If not specified, the script uses the default value in CustomSettings.ini

Properties

Name	Read	Write
All	●	●

ZTIGroups.wsf

This script captures and restores the local group membership on the target computer. This script is called with the **/capture** argument to back up the group membership from the target computer before deploying the operating system. The **CaptureGroups** property contains the list of groups that script backs up. The script is called with the **/restore** argument to restore the group membership after the operating system is deployed. When performing a restore operation, it restores the membership of all groups that were backed up when the script was run using the **/capture** argument.

Note When restoring group membership, the script does not create any destination groups that do not already exist on the target computer. Therefore, be sure to include all required groups in the reference computer when building the image file.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIGroups.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generates
References	ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIGroups.wsf </debug: value> </backup> </restore></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/capture	Backs up the group membership of the local groups on the target computer as specified in the CaptureGroups property
/restore	Restores the group membership to the local groups backed up earlier in the deployment process

Properties

Name	Read	Write
CaptureGroups	●	
Groups	●	●
HostName	●	

ZTILangPacksOnline.wsf

This script installs language packs for Windows operating systems.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTILangPacksOnline.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> CMD.exe. Allows running of command-line tools Lpksetup.exe. The Language Pack Setup tool used to add or remove language packs ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>ecscript ZTILangPacksOnline.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
Architecture	●	
OSVersion	●	

ZTIModifyVol.wsf

This script modifies a volume to set the GPT ID and attributes for utility volumes, which is necessary for creating Windows RE partitions on computers with UEFI. This script needs to be called when deploying to computers with UEFI for these situations:

- LTI deployments where custom partition (volume) structures are being created, such as creating five partition instead of the standard four partitions that are typically created for use with UEFI
- All ZTI and UDI deployments

Note This script is intended to be called only when creating partitions structures for use with UEFI. This script should not be called when creating partition structures to be used in deployments without UEFI.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	BDD.log contains events that all MDT scripts generate.
References	ZTUtility.vbs includes support functions and subroutines that the script uses.
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIModifyVol.wsf /UtilityVol:value </debug:value></code>

Arguments

Value	Description
/UtilityVol:value	Provides the drive letter of the volume that needs to be configured for a Windows RE Tools partition for use with computers with UEFI (for example, "E:")
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
UtilityVol	●	

ZTIMoveStateStore.wsf

This script moves the captured user state and backup files to C:\Windows\Temp\StateStore.

Note This script is run only when deploying images using Configuration Manager.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIMoveStateStore.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	ZTILUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTILMoveStateStore.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
None		

ZTINextPhase.wsf

This script updates the **Phase** property to the next phase in the deployment process. The Task Sequencer uses these phases to determine the sequence in which each task must be completed. The **Phase** property includes the following values:

- **VALIDATION.** Identify that the target computer is capable of running the scripts necessary to complete the deployment process.
- **STATECAPTURE.** Save any user state migration data before deploying the new target operating system.
- **PREINSTALL.** Complete any tasks that need to be done (such as creating new partitions) before the target operating system is deployed.
- **INSTALL.** Install the target operating system on the target computer.
- **POSTINSTALL.** Complete any tasks that need to be done before restoring the user state migration data. These tasks customize the target operating system before starting the target computer the first time after deployment (such as installing updates or adding drivers).
- **STATERESTORE.** Restore the user state migration data saved during the State Capture Phase.

For more information about the **Phase** property, see the corresponding topic in [Properties](#).

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTINextPhase.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTINextPhase.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is:

Value	Description
	<ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
DeploymentMethod	●	
Phase	●	●

ZTINICConfig.wsf

This script configures activated network adapters with values that ZTIGather.wsf captured based on the properties listed in the CustomSettings.ini file or the MDT DB (created in the Database node in the Deployment Workbench).

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTINICConfig.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • ZTUtility.vbs. Includes support functions and subroutines that the script uses • ZTNicUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTINICConfig.wsf </debug: value> </ForceCapture> </RestoreWithWinPE></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is:

Value	Description
	<ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/ForceCapture	If there are any local networking adapters with static IP addresses saved, this script captures those settings and saves them to the local environment—for example, C:\MININT\SMSOSD\OSDLogs\Variables.dat. This script can be useful in capturing static IP settings for a large number of computers for automation.
/RestoreWithinWinPE	When specified, applies any saved static IP network settings to the local computer, when appropriate; used for internal processing only.

Properties

Name	Read	Write
DeployDrive	●	●
DeploymentMethod	●	
DeploymentType	●	
DeployRoot	●	
OSDAdapterCount	●	●
OSGuid	●	
OSDMigrateAdapterSettings	●	
Phase	●	

ZTINICUtility.vbs

This script contains network adapter-related functions and subroutines that the various scripts in the MDT deployment process call.

Value	Description
Input	None
Output	None
References	<ul style="list-style-type: none"> • CMD.exe. Allows running of command-line tools • Netsh.exe. A utility used to automate the configuration of networking components

Value	Description
Location	<i>distribution\Scripts</i>
Use	<script language="VBScript" src="ZTINicUtility.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
OSDAdapterAdapterIndexAdapterName	●	●

Note *AdapterIndex* in this property is a placeholder for a zero-based array that contains network adapter information.

ZTIOSRole.wsf

This script installs server roles for target computers that are running Windows operating systems. The script reads the **OSRoles**, **OSRoleServices**, and **OSFeatures** properties to determine what should be installed.

Note This script is intended to be called only by the **Install Roles and Features** and **Uninstall Roles and Features** task sequence steps. Calling this script directly is not supported.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIOSRole.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> CMD.exe. Allows running of command-line tools OCSetup.exe. Adds or removes Windows optional components ServerManagerCmd.exe. Installs, configures, and manages Windows Server roles and features Sysocmgr.exe. Adds or removes Windows components

Value	Description
	<ul style="list-style-type: none"> • ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI0SRole.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/Uninstall	If provided, this argument indicates that the roles and features will be uninstalled. If not provided, the script assumes the roles and features will be installed.

Properties

Name	Read	Write
IsServerCoreOS	●	
OSFeatures	●	
OSRoles	●	
OSRoleServices	●	
OSVersion	●	
SMSTSRebootRequested		●

ZTPatches.wsf

This script installs updates (language packs, security updates, and so on) that are listed in the Packages.xml file. The script self-terminates if the deployment is not in one of the following states:

- **Phase** equals **PREINSTALL**
- **DeploymentMethod** equals **SCCM**

The script starts Pkgmgr if **DeploymentMethod** equals **SCCM**.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIPatches.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • Expand.exe. Expands compressed files • Pkgmgr.exe. Installs or updates Windows Vista offline • ZTIConfigFile.vbs. Includes routines for processing XML files • ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIPatches.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	<p>Outputs the event messages to the console and to the .log files. If the value specified in value is:</p> <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
Architecture	●	
CustomPackageSelectionProfile	●	
DeployRoot	●	
DeploymentMethod	●	
DeploymentType	●	
DestinationLogicalDrive	●	

Name	Read	Write
LanguagePacks	●	
OSDAnswerFilePath	●	
OSDPlatformArch	●	
PackageSelectionProfile	●	
Phase	●	
ResourceRoot	●	

ZTIPowerShell.wsf

This script runs a Windows PowerShell script using a custom Windows PowerShell host.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIPowerShell.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate Return code. The numeric value returned by the Windows PowerShell script after completion, which indicates the completion status of the script.
References	<ul style="list-style-type: none"> Microsoft.BDD.TaskSequencePSHost.exe. Custom Windows PowerShell host used to run the Windows PowerShell script.
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIPowerShell.wsf</code>

Arguments

Value	Description
None	

Properties

Name	Read	Write

Name	Read	Write
None		

ZTIPrereq.vbs

This script verifies that the target computer has the prerequisite software installed and that it is functional. The checks the script performs are:

- Determine whether the Windows Script version is equal to or greater than version 5.6.
- Verify that errors do not occur when object references are instantiated to Wscript.Shell, Wscript.Network, Scripting.FileSystemObject MSXML2.DOMDocument, and the Process environment.

If any one of the checks fails, an error is raised and the script exits the **ValidatePrereq** procedure.

Value	Description
Input	None
Output	None
References	None
Location	<i>distribution\Scripts</i>
Use	None

Arguments

Value	Description
None	None

Properties

Name	Read	Write
None		

ZTISCCM.wsf

This script initializes ZTI when deploying using Configuration Manager. The script performs the following procedure:

1. If debugging is activated, the script creates the OSD.Debug file.
2. The script configures these properties:

- **ScriptRoot** is set to the parent folder of the currently running script.
 - **DeployRoot** is set to the parent folder of **ScriptRoot**.
 - **ResourceRoot** is set to **DeployRoot**.
 - **DeploySystemDrive** is set to **C:**.
 - **DeploymentMethod** is set to **SCCM**.
3. When **DeployRoot** contains ****:
- The **DeployRoot** folder is copied to **_SMSTSMDATAPath\WDPackage**
 - **ScriptRoot** is set to **_SMSTSMDATAPath\WDPackage\Scripts**
 - **DeployRoot** is set to the parent folder of **ScriptRoot**
 - **ResourceRoot** is set to **DeployRoot**
4. When **Phase** is **NULL**:
- If the **%SystemDrive%** environment variable is **X:**, then **DeploymentType** is set to **NEWCOMPUTER** and **Phase** is set to **PREINSTALL**. Otherwise, **DeploymentType** is set to **REPLACE** and **Phase** is set to **VALIDATION**.
 - If the **OldComputer.tag** file exists in the parent folder of the current running script, **DeploymentType** is set to **REPLACE** and **Phase** is set to **VALIDATION**. Otherwise, **DeploymentType** is set to **REFRESH** and **Phase** is set to **VALIDATION**.

For more information about these properties, see the corresponding topics in [Properties](#).

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTISCCM.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTISCCM.wsf </debug: value></code>

Arguments

Value	Description
-------	-------------

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
_SMSTSMDDataPath	●	
Architecture	●	
BDDPackageID	●	●
DeploymentMethod	●	●
DeploymentType	●	●
DeployRoot	●	●
Phase	●	●
ResourceRoot	●	●
ScriptRoot	●	●
ToolRoot	●	●

ZTISetVariable.wsf

This script sets the specified global task sequence variable that corresponds to the name contained in **VariableName** to the value contained in **VariableValue**.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTISetVariable.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	ZTIUtility.vbs. Includes support functions and

Value	Description
	subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI SetVariable.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
VariableName	●	
VariableValue	●	

ZTITatoo.wsf

This script tattoos the target computer with identification and version information. The script performs the following procedure:

1. Locate and copy the ZTITatoo.mof file to the %SystemRoot%\System32\Wbem folder. Any preexisting ZTITatoo.mof that exists at the destination will be deleted before starting the copy operation.
2. Mofcomp.exe will be run using the following command:
`%SystemRoot%\System32\Wbem\Mofcomp.exe - autorecover
%SystemRoot%\System32\Wbem\ZTITatoo.mof.`
3. For all deployment methods (LTI, ZTI, and UDI), these deployment details are written for all deployment methods to the registry at **HKEY_LOCAL_MACHINE\Software\Microsoft\Deployment 4:**
 - **Deployment Method** is set to the deployment method being used and can be set to **LTI**, **ZTI**, or **UDI**, depending on the deployment method being performed.
 - **Deployment Source** is set to the source for the deployment and can be set to **OEM**, **MEDIA**, or the value in the **DeploymentMethod** property.

- **Deployment Type** is set to the **DeploymentType** property.
 - **Deployment Timestamp** is set to the current date in WMI date format.
 - **Deployment Toolkit Version** is set to the **Version** property.
4. For LTI deployments, these deployment details are written to the registry at **HKEY_LOCAL_MACHINE\Software\Microsoft\Deployment 4:**
- **Task Sequence ID** is set to the **TaskSequenceID** property.
 - **Task Sequence Name** is set to the **TaskSequenceName** property.
 - **Task Sequence Version** is set to the **TaskSequenceVersion** property.
5. For all Configuration Manager deployments (ZTI and UDI for Configuration Manager), these deployment details are written to the registry at **HKEY_LOCAL_MACHINE\Software\Microsoft\Deployment 4:**
- **OSD Package ID** is set to the **_SMSTSPackageID** task sequence variable.
 - **OSD Program Name** is always set to **"*"**.
 - **OSD Advertisement ID** is set to the **_SMSTSAdvertID** task sequence variable.
6. For LTI deployments where an image is being captured, these deployment details are written to the registry at **HKEY_LOCAL_MACHINE\Software\Microsoft\Deployment 4:**
- **Capture Method** is set to the deployment method being used and can be set to **LTI**, **ZTI**, or **UDI**, depending on the deployment method being performed.
 - **Capture Timestamp** is set to the current date in WMI date format.
 - **Capture Toolkit Version** is set to the **Version** property.
 - **Capture Task Sequence ID** is set to the **TaskSequenceID** property.
 - **Capture Task Sequence Name** is set to the **TaskSequenceName** property.
 - **Capture Task Sequence Version** is set to the **TaskSequenceVersion** property.
7. For all Configuration Manager deployments (ZTI and UDI for Configuration Manager) in which an image is being captured, these deployment details are written to the registry at **HKEY_LOCAL_MACHINE\Software\Microsoft\Deployment 4:**
- **Capture OSD Package ID** is set to the **_SMSTSPackageID** task sequence variable.
 - **Capture OSD Program Name** is always set to **"*"**.
 - **Capture OSD Advertisement ID** is set to the **_SMSTSAdvertID** task sequence variable.

Note This script is not designed to run on Windows PE.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTITatoo.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • Mofcomp.exe. Command-line .mof file compiler • ZTUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTITatoo.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
_SMSTSAdvertiserID	●	
_SMSTSPackageID	●	
_SMSTSSiteCode	●	
DeploymentMethod	●	
DeploymentType	●	
Version	●	
TaskSequenceID	●	
TaskSequenceName	●	
TaskSequenceVersion	●	

ZTIUserState.wsf

This script initializes USMT to capture and restore user state on the target computer.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> ZTIUserState.log. Log file that contains events that this script generates BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> CMD.exe. Allows running of command-line tools Loadstate.exe. Deposits user state data on a target computer Msiexec.exe. Manages the installation of .msi-based applications Scanstate.exe. Collects user data and settings USMT Application Files ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIUserState.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> TRUE, event messages are sent to the console and the .log files FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/Capture	–
/Estimate	–

Value	Description
/Restore	–

Properties

Name	Read	Write
Architecture	●	
DeploymentMethod	●	
DeploymentType	●	
DestinationLogicalDrive	●	
ImageBuild	●	
ImageSize	●	
ImageSizeMultiplier	●	
InstallFromPath	●	
IsServerOS	●	
LoadStateArgs	●	
OSCurrentVersion	●	
OSDMigrateAdditionalCaptureOptions	●	●
OSDMigrateAdditionalRestoreOptions	●	●
OSDPackagePath	●	
OSDStateStorePath		●
OSVersion	●	
ScanStateArgs	●	
StatePath	●	●
UDDir	●	
UDProfiles	●	
UDShare	●	
UserDataLocation	●	●
USMTConfigFile	●	
USMTEstimate	●	●
USMTLocal		●
USMTMigFiles	●	

ZTIUtility.vbs

This script contains utility functions that most of the MDT scripts use.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	None
References	<ul style="list-style-type: none"> Credentials_ENU.xml. Prompts the user for credentials that will be used when connecting to network resources IPConfig.exe. Displays all current TCP/IP network configuration values and refreshes DHCP and DNS settings MSHTA.exe. HTML application host Regsvr32.exe. Registers files (.dll, .exe, .ocx, and so on) with the operating system Xcopy.exe. Copies files and directories, including subdirectories
Location	<ul style="list-style-type: none"> <i>distribution\Scripts</i> <i>program_files\Microsoft Deployment Toolkit\Scripts</i>
Use	<script language="VBScript" src="ZTIUtility.vbs"/>

Arguments

Value	Description
None	None

Properties

Name	Read	Write
_SMSTSAdvertiserID	●	
_SMSTSCurrentActionName	●	
_SMSTSCustomProgressDialogMessage	●	
_SMSTSInstructionTableSize	●	
_SMSTSLogPath	●	
_SMSTSMachineName	●	

Name	Read	Write
_SMSTSNextInstructionPointer	●	
_SMSTSOrgName	●	
_SMSTSPackageID	●	
_SMSTSPackageName	●	
_SMSTSPackagePath	●	
_SMSTSReserved1	●	
_SMSTSReserved2	●	
Architecture	●	
AssetTag	●	
ComputerName	●	
Debug	●	●
DeploymentMethod	●	
DeployRoot	●	
DestinationDisk	●	●
DestinationLogicalDrive	●	●
DestinationPartition	●	●
EventShare	●	
HostName	●	
ImageBuild	●	●
ImageFlags		●
ImageIndex		●
ImageLanguage		●
ImageProcessor		●
ImageSize		●
InstallFromPath		●
JoinDomain	●	
LogPath	●	●
MacAddress	●	
OSCurrentVersion	●	
OSDAdvertID	●	
OSDAnswerFilePath	●	●

Name	Read	Write
OSDAnswerFilePathSysprep	●	●
OSDComputerName	●	●
OSDPackageID	●	
OSDPackagePath	●	
OSDTargetSystemDrive	●	
OSGUID		●
OSSKU	●	
OSVersion	●	
Phase	●	
Processor_Architecture	●	
ResourceRoot	●	
SLShare	●	
SLShareDynamicLogging	●	
TaskSequenceID	●	
TaskSequenceName		●
TaskSequenceVersion		●
UDDir	●	
UDShare	●	
UserDomain	●	●
UserID	●	●
UserPassword	●	●
UUID	●	
Version	●	●
Note This variable is an internal variable that represents the version of MDT.		
WDSServer	●	

ZTIValidate.wsf

This script ensures that it is safe for the deployment to continue by validating the condition of the target computer. The script processes are:

- If **DeploymentType** equals REFRESH and the target computer is a server, the script exits.

- If **OSInstall** exists and is not equal to YES, the script exits.
- Verify that the minimum amount of RAM exists on the target computer; if not, the script exits.
- Verify that the processor meets the minimum required speed; if not, the script exits.
- Verify that the hard disk size meets the minimum size requirements; if not, the script exits.
- Verify that the target computer's operating system is installed on drive C; if not, the script exits.
- If **DeploymentType = REFRESH**, verify that drive C is not compressed by running **Compact /u C:\.**

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIValidate.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • Compact.exe. Displays or alters the compression of files on NTFS file system partitions • ZTIVUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIValidate.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
DeploymentType	●	
DestinationLogicalDrive	●	●
ImageBuild	●	
ImageMemory	●	
ImageProcessorSpeed	●	
ImageSize	●	
ImageSizeMultiplier	●	
IsServerOS	●	
Memory	●	
OSDPackagePath	●	
OSInstall	●	
ProcessorSpeed	●	
SMSTSLocalDataDrive		●
VerifyOS	●	

ZTIVHDCreate.wsf

This script is used to create a virtual hard disk (.vhd or .avhd) file on the target computer and mount the .vhd file as a disk. Then, other portions of the LTI deployment process deploy the Windows operating system and applications to the newly created virtual hard disk. The script processes are as follows:

- The **Class_Initialize** method is used to initialize the **VHDInputVariable** variable.
- Validate that **VHDCreateSource** is defined and locates the source .vhd file (if specified).
- Generate a random .vhd file name if **VHDCreateFilename** equals RANDOM or "" (null).
- Verify that the folder exists where the .vhd file (specified in **VHDCreateFileName**) is to be created.
- Create the .vhd file using the values in **VHDCreateSizePercent**, **VHDCreateSizeMax**, and **VHDCreateType**.
- Create a differencing disk (if specified) using the value in **VHDCreateDiffVHD**.
- The newly created .vhd file and the optional differencing disk are mounted.
- The disk number of the mounted virtual hard disk is returned.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIVHDCreate.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • ZTIDiskUtility.vbs. Includes support functions and subroutines the script uses • ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTIVHDCreate.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
VHDCreateDiffVHD	●	
VHDCreateFileName	●	
VHDCreateSizeMax	●	
VHDCreateSource	●	
VHDCreateType	●	
VHDDisks		●
VHDIInputVariable	●	
VHDOOutputVariable	●	

ZTIWindowsUpdate.wsf

This script downloads and installs updates from computers on a corporate network that are running WSUS, Windows Update, or Microsoft Update using the [Windows Update Agent \(WUA\)](#) application programming interface (API). By default, this feature is disabled in each task sequence and must be manually activated to run.

Most enterprises will already have teams and infrastructures in place to update newly deployed computers over the corporate network. This process involves tracking the latest set of patches, drivers, and updates available for each desktop configuration and determining which updates should be downloaded and installed for each configuration. If the organization already has an established process, this script might not be necessary. This script was designed to fill a need for deployment teams that might not have established processes, yet want to ensure that target computers are updated when deployed.

This script automatically scans the target computer and downloads a wide range of updates that are found to be applicable. Among these are:

- Windows service packs
- Non-Microsoft drivers that were placed on Windows Update
- The latest hotfix updates
- Microsoft Office updates
- Microsoft Exchange Server and SQL Server updates
- Microsoft Visual Studio® updates
- Some non-Microsoft application updates

Tip Many hardware manufacturers have placed their drivers on Windows Update. These drivers no longer need to be maintained in the Out-of-Box Drivers directory. Experiment by removing drivers from the distribution share to see which ones are available on Windows Update. Note that if the drivers are not included with Windows by default, do not remove networking or storage drivers, because the operating system will require user input.

MDT supports the ability to deploy an updated version of WUA as part of the operating system deployment. This helps ensure that target computers are running the correct version of WUA when they are deployed. It also helps eliminate the need to connect to the Internet and download the latest version of WUA after deployment.

MDT can also configure WUA to collect updates from computers on the corporate network that are running WSUS instead of connecting to Microsoft Updates over the Internet. MDT can optionally configure WUA to use a specific computer running WSUS using the **WSUSServer** property.

For additional information and for WUA deployment instructions, see [How to Install the Windows Update Agent on Client Computers](#).

Obtain the latest version of the WUA stand-alone installer for:

- x86 versions (WindowsUpdateAgent30-x86.exe) at
<http://go.microsoft.com/fwlink/?LinkId=100334>
- x64 version (WindowsUpdateAgent30-x64.exe) at
<http://go.microsoft.com/fwlink/?LinkId=100335>

Windows 7 and later include the most recent version of WUA, so no upgrade is necessary.

For more information, see [Updating Windows Update Agent](#).

When enabled in the Task Sequencer, this script runs multiple times while in the State Restore Phase of operating system deployment. It is first run after the operating system has started for the first time. Ensure that the latest updates and service packs are installed before the installation of any applications that might depend on specific updates or service packs being installed on the target computer. For example, an application might be dependent on the latest version of the Microsoft .NET Framework being installed.

This script also runs after the installation of applications, which ensures that the latest application service packs and updates have been applied. For example, use this script to ensure that the latest updates are applied to Microsoft Office 2010 or the 2007 Office system.

It is possible, during the installation of one or more updates, the target computer will need to be restarted to allow an update installation to finish fully. To ensure that updates are properly installed, if the script detects that the installation of an update requires the target computer to be restarted, the script automatically restarts the target computer and resumes if additional updates have been detected and are pending installation. The script exits if it determines that the target computer is fully up to date. An error will be logged if, while updating the target computer, the script has seven unsuccessful attempts to install the updates and the target computer still requires a restart.

During run time, the script performs the following tasks:

- Configure the target computer to use a WSUS server, if the **WSUSServer** property was specified.
- Verify that the latest version of the WUA is installed on the target computer.
- Search the target computer for applicable updates that are not already installed and that might be typically hidden.
- Each update has an associated **UpdateID** and **QNumber** property:
 - The **UpdateID** property is in GUID form, such as *67da2176-5c57-4614-a514-33abbd51f67*.
 - The **QNumber** property is a numerical value, such as *987654*.
- The script compares the **UpdateID** and **KBArticle** property values against the list of exclusions specified in the following MDT properties:

- **WUMU_ExcludeID.** A list of UpdateIDs to exclude; any update with an **UpdateID** found in this list will not be installed.
- **WUMU_ExcludeKB.** A list of **QNumbers** to exclude; any update with a **QNumber** found in this list will not be installed.
- In addition, any update that requires user input will be excluded and not installed.
- All updates that require approval of an End User License Agreement (EULA) will automatically be approved by the script. Be sure to manually read and check each EULA before running this script in a production environment.
- The activity for each update is written to the **ZTIWindowsUpdate.log** file, with the string **INSTALL** or **SKIP** if the update has been approved for installation, along with the **UpdateID**, a short description of the update, and the **QNumber**.
- Each update to be installed is downloaded and installed in batches.
- The target computer might require more than one restart during the update installation.

Note Windows Internet Explorer 7 requires user interaction, so it is not installed using this script.

Note By default, include **QNumber 925471** in the **WUMU_ExcludeKB** list to prevent Windows Vista Ultimate from installing extra language packs.

Note If intranet sources are not available, this script downloads files from two Microsoft sites:
<http://update.microsoft.com/redist/wuredist.cab> and
<http://download.windowsupdate.com/v6/windowsupdate/redist/standalone/muauth.cab>.

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIWindowsUpdate.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • Expand.exe. Expands compressed files • Net.exe. Performs network management tasks • WindowsUpdateAgent30-x86.exe. Installs WUA • WindowsUpdateAgent30-x64.exe. Installs WUA • ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI WindowsUpdate.wsf </debug: value> </UpdateCommand: "<IsInstalled=0 1><IsHidden=0 1>"> </Query: true false></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)
/UpdateCommand:param	<ul style="list-style-type: none"> • IsInstalled. Set to 0 to query for updates that are not installed. • IsHidden. Set to 0 to query for updates that are hidden.
/Query:value	<ul style="list-style-type: none"> • True. Query only for required updates. Do not download and install any binaries. • False. Query for and install required updates. Download and install binaries.

Note When specified, **UpdateCommand** requires at least one option.

Note If specifying both options for **UpdateCommand**, they must be separated by *and*.

Note The default value for **UpdateCommand** is **IsInstalled=0** and **IsHidden=0**.

Note For more information about **UpdateCommand**, see [IUpdateSearcher::Search Method](#).

Properties

Name	Read	Write
Architecture	●	
DoCapture	●	
InstalledUpdates		●
MSIT_WU_Count	●	●
NoAutoUpdate_Prevous	●	●
SMSTSRebootRequested	●	●
SMSTSRetryRequested	●	●
WSUSServer	●	
WUMU_ExcludeID	●	
WUMU_ExcludeKB	●	

ZTIWipeDisk.wsf

This script formats the target computer's hard disk. The script:

- Exits if **WipeDisk** is not equal to **TRUE**
- Determines the appropriate drive to format
- Formats the drive by calling `cmd /c format <Drive> /fs: ntfs /p: 3 /Y` (where `<Drive>` is the drive letter of the hard disk drive to be formatted)

Value	Description
Input	Environment variables. Contains the property values, custom property values, database connections, deployment rules, and other information that the scripts require to complete the deployment process
Output	<ul style="list-style-type: none"> • ZTIWipeDisk.log. Log file that contains events that this script generates • BDD.log. Log file that contains events that all MDT scripts generate
References	<ul style="list-style-type: none"> • CMD.exe. Allows running of command-line tools • Format.com. Formats the hard disk • ZTIUtility.vbs. Includes support functions and subroutines that the script uses
Location	<i>distribution\Scripts</i>
Use	<code>cscript ZTI WipeDisk.wsf </debug: value></code>

Arguments

Value	Description
/debug:value	Outputs the event messages to the console and to the .log files. If the value specified in value is: <ul style="list-style-type: none"> • TRUE, event messages are sent to the console and the .log files • FALSE, event messages are sent only to the .log files (This is the behavior when the argument is not provided.)

Properties

Name	Read	Write
WipeDisk	●	

Support Files

The utilities and scripts used in LTI and ZTI deployments reference external configuration files to determine the process steps and configuration settings used during the deployment process.

The following information is provided for each utility:

- **Name.** Specifies the name of the file
- **Description.** Provides a description of the purpose of the file
- **Location.** Indicates the folder where the file can be found; in the information for the location, the following variables are used:
 - **program_files.** This variable points to the location of the Program Files folder on the computer where MDT is installed.
 - **distribution.** This variable points to the location of the Distribution folder for the deployment share.
 - **platform.** This variable is a placeholder for the operating system platform (x86 or x64).

ApplicationGroups.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

Applications.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

BootStrap.ini

The configuration file used when the target computer is not able to connect to the appropriate deployment share. This situation occurs in the New Computer and the Replace Computer scenarios.

Value	Description
Location	<i>distribution\Control</i>

CustomSettings.ini

The primary configuration file for the MDT processing rules used in all scenarios.

Value	Description
Location	<i>distribution\Control</i>

Deploy.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>program_files\Microsoft Deployment Toolkit\Control</i>

DriverGroups.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

Drivers.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

LinkedDeploymentShares.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Scripts</i>

ListOfLanguages.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description

Value	Description
Location	<i>distribution\Scripts</i>

MediaGroups.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Scripts</i>

Medias.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Scripts</i>

OperatingSystemGroups.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

OperatingSystems.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

PackageGroups.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

Packages.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

SelectionProfileGroups.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

SelectionProfiles.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

ServerManager.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>program_files\Microsoft Deployment Toolkit\Bin</i>

Settings.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

TaskSequenceGroups.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description

Value	Description
Location	<i>distribution\Control</i>

TaskSequences.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control</i>

TS.xml

Note This XML file is managed by MDT and should not require modification.

Value	Description
Location	<i>distribution\Control\task_sequence_id</i>

Note *Task_sequence_id* is a placeholder for the task sequence ID that was assigned to each task sequence when it was created in the Task Sequences node in the Deployment Workbench.

Wimscript.ini

This .ini file is an ImageX configuration file that contains the list of folders and files that will be excluded from an image. It is referenced by ImageX during the LTI Capture Phase.

For assistance with customizing this file, see the section, "Create an ImageX Configuration File," in the *Windows Preinstallation Environment (Windows PE) User's Guide*.

Value	Description
Location	<i>distribution\Tools\platform</i>

ZTIBIOSCheck.xml

This XML file contains metadata about BIOSes for target computers. This file is edited manually and is read by [ZTIBIOSCheck.wsf](#). Extract the necessary information from a target computer to create an entry in this XML file using the Microsoft Visual Basic® Scripting Edition (VBScript) program (ZTIBIOS_Extract.Utility.vbs) that is embedded in this XML file.

Value	Description

Value	Description
Location	<i>distribution\Scripts</i>

ZTIConfigure.xml

This XML file is used by the [ZTIConfigure.wsf](#) script to translate property values (specified earlier in the deployment process) to configure settings in the Unattend.xml file. This file is already customized to make the appropriate translations and should not require further modification.

Value	Description
Location	<i>distribution\Scripts</i>

ZTIGather.xml

Note This XML file is preconfigured and should not require modification. Define custom properties in the CustomSettings.ini file or the MDT DB.

Value	Description
Location	<i>distribution\Scripts</i>

ZTIUserState_config.xml

This XML file is used by the [ZTIUserState.wsf](#) script as a default USMT configuration file. This file is used by default if no custom configuration file is specified by the [USMTConfigFile](#) property. See the [Config.xml File](#) topic in the USMT documentation for more information on syntax and use.

Value	Description
Location	<i>distribution\Scripts</i>

ZTITatoo.mof

This .mof file, when imported into the WMI repository of the target computer using Mofcomp.exe, creates the **Microsoft_BDD_Info** WMI class. This class contains deployment-related information, such as:

- DeploymentMethod
- DeploymentType
- DeploymentTimestamp
- BuildID

- BuildName
- BuildVersion
- OSDPackageID
- OSDProgramName
- OSDAdvertisementID
- TaskSequenceID
- TaskSequenceName
- TaskSequenceVersion

Value	Description
Location	<i>distribution\Scripts</i>

Utilities

The scripts used in LTI and ZTI reference utilities that perform specialized tasks supporting the steps used during the deployment process. Use the following information to help determine the correct utilities to include in actions and the valid arguments to provide when running each utility.

The following information is provided for each utility:

- **Name.** Specifies the name of the utility
- **Description.** Provides a description of the purpose of the utility
- **Location.** Indicates the folder where the utility can be found; in the information for the location, the following variables are used:
 - **program_files.** This variable points to the location of the Program Files folder on the computer where MDT is installed.
 - **distribution.** This variable points to the location of the Distribution folder for the deployment share.
 - **platform.** This variable is a placeholder for the operating system platform (x86 or x64).
- **Use.** Provides the commands and options that can be specified
- **Arguments and description.** Indicates the valid arguments to be specified for the utility and a brief description of what each argument means

BCDBoot.exe

BCDBoot is a tool used to quickly set up a system partition or repair the boot environment located on the system partition. The system partition is set up by copying a small set of boot environment files from an installed Windows image. BCDBoot also creates a Boot Configuration Data (BCD) store on the system partition, with a new boot entry that enables Windows to boot to the installed Windows image.

Value	Description
Location	Included in the Windows source files

Arguments

Value	Description
	See the command-line help provided by this utility.

BDDRun.exe

This utility is run as an action by the Task Sequencer for executables (such as a script or other code) that require user interaction. By default, the task sequence cannot run an executable that requires user interaction. However, this utility allows the Task Sequencer to run an executable that requires user interaction.

The executable that requires user interaction is provided as an argument to this utility. This utility runs the executable in a separate command environment.

Note This utility can only be used in LTI deployments. ZTI deployments prohibit any user interaction.

Value	Description
Location	<i>distribution\Tools\platform</i>
Use	<code>BDDRun. exe commandline</code>

Arguments

Value	Description
<i>commandline</i>	The command to be run that requires user interaction

Note Put double quotation marks around any part of the *command-line* portion of the argument that contains blanks. For example: `BDDRun. exe MyAppInstall. exe /destinationDir: "%ProgramFiles%\AppName"`.

Bootsect.exe

Bootsect.exe updates the master boot code for hard disk partitions to switch between BOOTMGR and NTLDR. Use this utility to restore the boot sector on the computer.

For more information on Bootsect.exe, see the section, "Bootsect Command-Line Options," in the *Windows Preinstallation Environment (Windows PE) User's Guide*.

Value	Description
Location	<i>distribution\Tools\platform</i>
Use	<code>bootsect. exe /nt52 C:</code>

Arguments

Value	Description
/Help	Displays the use instructions listed here.
/nt52	Applies the master boot code compatible with NTLDR to SYS , ALL , or <i>DriveLetter</i> . The operating system

Value	Description
	installed on SYS , ALL , or <i>DriveLetter</i> must be an earlier version of Windows Vista.
/nt60	Applies the master boot code compatible with BOOTMGR to SYS , ALL , or <i>DriveLetter</i> . The operating system installed on SYS , ALL , or <i>DriveLetter</i> must be Windows Vista.
SYS	Updates the master boot code on the system partition used to boot Windows.
All	Updates the master boot code on all partitions. ALL does not necessarily update the boot code for each volume. Instead, this option updates the boot code on volumes that can be used as Windows boot volumes, which excludes any dynamic volumes not connected with an underlying disk partition. This restriction is present, because the boot code must be located at the beginning of a disk partition.
<i>DriveLetter</i>	Updates the master boot code on the volume associated with this drive letter. The boot code will not be updated if either (1) DriveLetter is not associated with a volume or (2) DriveLetter is associated with a volume not connected to an underlying disk partition.
/Force	Forcibly dismounts the volumes during the boot code update. Use this option with caution.

Compact.exe

Displays or alters the compression of files on NTFS file system partitions.

Value	Description
Location	Included in the Windows source files

Arguments

Value	Description
/C	Compresses the specified files. Directories will be marked so that files added afterward will be compressed.
/V	Decompresses the specified files. Directories will be marked so that files added afterward will not be compressed.
/S	Performs the specified operation on files in the given directory and in all subdirectories. Default dir is the

Value	Description
	current directory.
/A	Displays files with the hidden or system attributes. These files are omitted by default.
/I	Continues performing the specified operation even after errors have occurred. By default, Compact.exe stops when an error is encountered.
/F	Forces the compress operation on all specified files, even those which are already compressed. Already-compressed files are skipped by default.
/Q	Reports only the most essential information.
<i>filename</i>	Specifies a pattern, file, or directory.

Diskpart.exe

Diskpart is a text-mode command interpreter that allows management of objects (disks, partitions, or volumes) using scripts or direct input in a Command Prompt window.

For more information on Diskpart.exe, see the section, "Diskpart Command-Line Options," in the *Windows Preinstallation Environment (Windows PE) User's Guide*.

Value	Description
Location	Included in the Windows PE source files

Arguments

Value	Description
	See the guide referenced in the utility description.

Expand.exe

This utility is run to expand (extract) files from compressed files.

Value	Description
Location	Included in the Windows source files
Use	Expand. exe - r wuredi st. cab - F: wuRedi st. xml %temp%

Arguments

Value	Description
-r	Renames expanded files
-D	Displays the list of files in the source directory
Source	Source file specification (Wildcards can be used.)
-F:Files	Name of files to expand from a .cab file
Destination	Destination file path specification (Destination can be a directory. If Source is multiple files and -r is not specified, Destination must be a directory.)

ImageX.exe

ImageX is a command-line utility that enables OEMs and corporations to capture, modify, and apply file-based disk images for rapid deployment. ImageX works with WIM files for copying to a network, or it can work with other technologies that use WIM images, such as Windows Setup and Windows Deployment Services.

For more information about ImageX, see the section, "What is ImageX," in the *Windows Preinstallation Environment (Windows PE) User's Guide*.

Value	Description
Location	<i>distribution\Tools\platform</i>

Arguments

Value	Description
	See the guide referenced in the utility description.

Microsoft.BDD.PnpEnum.exe

This utility is run to enumerate Plug and Play devices installed on the target computer.

Value	Description
Location	<i>distribution\Tools\platform</i>

Arguments

Value	Description
None	–

Mofcomp.exe

Mofcomp.exe is the Managed Object Format compiler that parses a file that contains Managed Object Format statements and adds the classes and class instances defined in the file to the WMI repository. Mofcomp.exe provides command-line help on the switch use options.

Value	Description
Location	Included in the Windows source files

Arguments

Value	Description
	See the command-line help that this utility provides.

Netsh.exe

Netsh.exe is a command-line and scripting utility used to automate the configuration of networking components. For more information about Netsh.exe, see [The Netsh Command-Line Utility](#).

Value	Description
Location	Included in the Windows source files

Arguments

Value	Description
	See the command-line help that this utility provides or the information found at the URL listed in the utility description.

Reg.exe

The Console Registry Tool is used to read and modify registry data.

Value	Description
Location	Included in the Windows source files

Arguments

Value	Description
	See the command-line help that this utility provides.

Regsvr32.exe

This utility is used to register files (.dll, .exe, .ocx, and so on) with the operating system.

Value	Description
Location	Included in the Windows source files

Arguments

Value	Description
file	The name of the file to register or unregister
/s	Runs the utility in silent mode
/u	Unregisters the file

Wpeutil.exe

The Windows PE utility (Wpeutil) is a command-line utility with which various commands can be run in a Windows PE session. For example, an administrator can shut down or reboot Windows PE, activate or deactivate a firewall, configure language settings, and initialize a network. MDT uses the utility to initialize Windows PE and network connections, and start LTI deployments.

For more information on Wpeutil.exe, see the section, "Wpeutil Command-Line Options," in the *Windows Preinstallation Environment (Windows PE) User's Guide*.

Value	Description
Location	Included in the Windows PE source files

Arguments

Value	Description
	See the guide referenced in the utility description.

MDT Windows PowerShell Cmdlets

In addition to the Deployment Workbench, MDT deployment shares can be managed using Windows PowerShell cmdlets. The MDT Windows PowerShell cmdlets are included in a Windows PowerShell snap-in—`Microsoft.BDD.PSSnapIn`—which is included with the installation of MDT.

The MDT cmdlets must be run from a Windows PowerShell console that has the MDT Windows PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT Windows PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Table 7 lists the MDT Windows PowerShell cmdlets and provides a brief description of each cmdlet. Each cmdlet is discussed in further detail in a subsequent section.

Table 7. MDT Windows PowerShell Cmdlets

Cmdlet	Description
Add-MDTPersistentDrive	Adds a deployment share to the list of MDT persisted drives that can be restored using the Restore-MDTPersistentDrive cmdlet.
Disable-MDTMonitorService	Disables the MDT monitoring services.
Enable-MDTMonitorService	Enables the MDT monitoring services.
Get-MDTDeploymentShareStatistics	Displays the statistics of a deployment share, including the number of entities per major folder in the deployment share.
Get-MDTMonitorData	Displays the MDT monitoring information collected for one or more monitored MTD deployments.
Get-MDTOperatingSystemCatalog	Returns the operating system catalog for a specific operating system. If the operating system catalog does not exist or is out of date, then the operating system catalog is regenerated.
Get-MDTPersistentDrive	Displays the list of deployment shares that can be restored using the Restore-MDTPersistentDrive cmdlet.
Import-MDTApplication	Imports an application into a deployment share.
Import-MDTDriver	Imports one or more device drivers

Cmdlet	Description
	into a deployment share.
Import-MDTOperatingSystem	Imports one or more operating systems into a deployment share.
Import-MDTPackage	Imports one or more operating system packages into a deployment share.
Import-MDTTaskSequence	Imports a task sequence into a deployment share.
New-MDTDatabase	Creates or upgrades an MDT DB database that is associated with a deployment share.
Remove-MDTMonitorData	Removes one or more MDT monitoring data items from the collected MDT monitoring data in a deployment share.
Remove-MDTPersistentDrive	Removes a deployment share from the list of MDT persisted Windows PowerShell drives that can be restored using the Restore-MDTPersistentDrive cmdlet.
Restore-MDTPersistentDrive	Creates a Windows PowerShell drive for each deployment share in the list of MDT persisted Windows PowerShell drives.
Set-MDTMonitorData	Creates a new or updates an existing MDT monitoring data item in the collected MDT monitoring data in a deployment share.
Test-MDTDeploymentShare	Verifies the integrity of a deployment share.
Test-MDTMonitorData	Verifies that the MDT monitoring services is configured correctly and running.
Update-MDTDatabaseSchema	Updates the MDT DB database schema.
Update-MDTDeploymentShare	Updates a deployment share.
Update-MDTLinkedDS	Replicates content from a deployment share to a linked deployment share.
Update-MDTMedia	Replicates content from a deployment share to a deployment media folder.

Add-MDTPersistentDrive

This section describes the **Add-MDTPersistentDriveWindows PowerShell** cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Add-MDTPersistentDrive [-Name] <String> [-InputObject] <PSObject> [<CommonParameters>]
```

Description

This cmdlet adds an existing Windows PowerShell drive created using the **MDTProvider** to a list of drives that are persisted in the Deployment Workbench or in a Windows PowerShell session using the [Restore-MDTPersistentDrive](#) cmdlet. This cmdlet is called when you create or open a deployment share in the Deployment Workbench.

Note The list of persisted **MDTProvider** drives is maintained on a per-user based in the user profile.

The list of persisted **MDTProvider** drives can be displayed using the [Get-MDTPersistentDrive](#) cmdlet.

Parameters

This subsection provides information about the various parameters that can be used with the **Add-MDTPersistentDriveWindows** cmdlet.

-Name <String>

Specifies the name of a Windows PowerShell drive created using the MDT provider and corresponds to an existing deployment share. The name was created using the [New-PSDrive](#) cmdlet and specifying the **MDTProvider** in the *PSPrinter* parameter.

For more information on how to create a new Windows PowerShell drive using the **MDTProvider** and how to create a deployment share using Windows PowerShell, see the section "Creating a Deployment Share Using Windows PowerShell" in the MDT document, *Microsoft Deployment Toolkit Samples Guide*.

Parameter	Value
Required?	True
Position?	2 and Named
Default value	None

Parameter	Value
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

-InputObject <PSObject>

This parameter specifies a Windows PowerShell drive object that was created earlier in the process. Enter a PSObject object, such as one generated by the [New-PSDrive](#) cmdlet.

Parameter	Value
Required?	False
Position?	3 and Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSObject** type object for the Windows PowerShell drive object was added to the list of persisted drives.

This cmdlet also outputs a **String** type object if the *Verbose* common parameter is included.

Example 1

```
Add-MDTPersistentDrive -Name DS001
```

Description

This example adds the deployment share with the Windows PowerShell drive name of *DS001* to the list of persisted drives.

Example 2

```
$MDTPSDrive = New-PSDrive -Name "DS001" -PSProvider "MDTProvider" -Root "C:\DeploymentShare\$" -Description "MDT Deployment Share" -NetworkPath \\WDG-MDT-01\DeploymentShare\$ -Verbose  
Add-MDTPersistentDrive -InputObject $MDTPSDrive
```

Description

This example adds the Windows PowerShell drive name *DS001*, created by the [New-PSDrive](#) cmdlet, to the list of persisted MDT drives using the `$MDTPSDrive` variable.

Example 3

```
New-PSDrive -Name "DS001" -PSProvider "MDTProvider" -Root "C:\DeploymentShare\$" -Description "MDT Deployment Share" -NetworkPath \\WDG-MDT-01\DeploymentShare\$ -Verbose | Add-MDTPersistentDrive -Verbose
```

Description

This example adds the Windows PowerShell drive name *DS001*, created by the [New-PSDrive](#) cmdlet, to the list of persisted MDT drives by piping the newly created Windows PowerShell drive object to the **Add-MDTPersistentDrive** cmdlet.

Disable-MDTMonitorService

This section describes the **Disable-MDTMonitorService** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Disable-MDTMonitorService [<CommonParameters>]
```

Description

This cmdlet disables the MDT monitoring service, which runs on the computer where MDT is installed. The MDT monitoring service collects monitoring information that can be displayed:

- In the Monitoring node in a deployment share in the Deployment Workbench
- Using the [Get-MDTMonitorData](#) cmdlet

The MDT monitoring service can subsequently be enabled using the [Enable-MDTMonitorService](#).

For more information on the MDT monitoring service, see the section "Monitoring MDT Deployments" in the MDT document, *Using the Microsoft Deployment Toolkit*.

Parameters

This subsection provides information about the various parameters that can be used with the **Disable-MDTMonitorService** cmdlet.

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, "about_CommonParameters," which you can accessed by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **String** type object if the *Verbose* common parameter is included; otherwise, no output is generated.

Example 1

Disable-MDTMonitorService

Description

This example disables the MDT monitoring service.

Enable-MDTMonitorService

This section describes the **Enable-MDTMonitorService** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Enable-MDTMonitorService [-EventPort] <Int32> [-DataPort] <Int32>
[<CommonParameters>]
```

Description

This cmdlet enables the MDT monitoring service, which runs on the computer where MDT is installed. The MDT monitoring service collects monitoring information that can be displayed:

- In the Monitoring node in a deployment share in the Deployment Workbench.
- Using the [Get-MDTMonitorData](#) cmdlet

The MDT monitoring service can be disabled using the [Disable-MDTMonitorService](#).

For more information on the MDT monitoring service, see the section "Monitoring MDT Deployments" in the MDT document, *Using the Microsoft Deployment Toolkit*.

Parameters

This subsection provides information about the various parameters that can be used with the **Enable-MDTMonitorService** cmdlet.

-EventPort <Int32>

This parameter specifies the TCP port used as the event port for the MDT monitoring service.

Parameter	Value
Required?	False
Position?	2 and Named
Default value	9800
Accept pipeline input?	False
Accept wildcard characters?	False

-DataPort <Int32>

This parameter specifies the TCP port used as the data port for the MDT monitoring service.

Parameter	Value
Required?	False
Position?	3 and Named
Default value	9801
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, "about_CommonParameters," which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **String** type object if the **Verbose** common parameter is included; otherwise, no output is generated.

Example 1

```
Enable-MDTMonitorService
```

Description

This example enables the MDT monitoring service on the local computer using the default value of **9800** for the event port and the value of **9801** for the data port on the MDT monitoring service.

Example 2

```
Enable-MDTMonitorService -EventPort 7000 -DataPort 7001
```

Description

This example enables the MDT monitoring service on the local computer using the value of **7000** for the event port and the value of **7001** for the data port on the MDT monitoring service.

Get-MDTDeploymentShareStatistics

This section describes the **Get-MDTDeploymentShareStatistics** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Get-MDTDeploymentShareStatistics [-Path <String>] [<CommonParameters>]
```

Description

This cmdlet displays the statistics of a deployment share based on the MDTProvider drive that is specified in the *Path* parameter. The statistics include the number of items in the specified deployment share:

- Applications
- Drivers
- Operating Systems
- Packages
- Task Sequences

- Selection Profiles
- Linked Deployment Shares
- MDT Media
- Computers in the MDT DB
- Make and Models in the MDT DB
- Locations in the MDT DB
- Roles in the MDT DB

Note The values for the statistics that relate to the MDT DB are not populated and always return a value of zero.

Parameters

This subsection provides information about the various parameters that can be used with the **Get-MDTDestinationShareStatistics** cmdlet.

-Path <String>

This parameter specifies the MDTProvider Windows PowerShell drive for the desired deployment share.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to a location within the desired MDTProvider Windows PowerShell drive.

Parameter	Value
Required?	False
Position?	2 and Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSObject** type object that contains the statistics for the deployment share.

Example 1

```
Get-MDTDeploymentShareStatistics -Path DS001:
```

Description

This example returns the deployment share statistics for the deployment share that is specified in the DS001: MDTProvider Windows PowerShell drive.

Example 2

```
cd DS001:
```

```
Get-MDTDeploymentShareStatistics
```

Description

This example returns the deployment share statistics for the deployment share that is specified in the DS001: MDTProvider Windows PowerShell drive. Use the **cd** command to set the working directory for Windows PowerShell to the DS001: MDTProvider Windows PowerShell drive.

Get-MDTMonitorData

This section describes the **Get-MDTMonitorData** Windows PowerShell cmdlet.

Run this cmdlet from a Windows PowerShell console that has the MDT

PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Get-MDTMonitorData [-Path <String>] [-ID <Nullable>]<br/>[<CommonParameters>]
```

Description

This cmdlet displays the MDT monitoring data that is being reported to the deployment share that is specified in the *Path* parameter. The following is example output from this cmdlet:

Name	:	WDG- REF- 01
PercentComplete	:	100
Settings	:	
Warnings	:	0
Errors	:	0
DeploymentStatus	:	3
StartTime	:	5/23/2012 6: 45: 39 PM
EndTime	:	5/23/2012 8: 46: 32 PM
ID	:	1
UniqueId	:	94a0830e- f2bb- 421c- b1e0- 6f86f9eb9fa1

```

CurrentStep      : 88
Total Steps     : 88
StepName        :
Last Time       : 5/23/2012 8:46:32 PM
DartIP          :
DartPort        :
DartTicket      :
VMHost          : WDG-HOST-01
VMName          : WDG-REF-01
ComputerIdentities : {}

```

Note The MDTProvider Windows PowerShell drive that this cmdlet references must exist prior to running this cmdlet.

Parameters

This subsection provides information about the various parameters that you can use with the **Get-MDTMonitorData** cmdlet.

-Path <String>

This parameter specifies the MDTProvider Windows PowerShell drive for the desired deployment share.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to a location within the desired MDTProvider Windows PowerShell drive.

Parameter	Value
Required?	False
Position?	2 and Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-ID <Nullable>

This parameter specifies the specific identifier for the deployment of a specific computer. If this parameter is not specified, then all monitoring data for deployments in the deployment share are displayed.

Parameter	Value
Required?	False
Position?	3 and Named

Parameter	Value
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSObject** type object for each monitored computer, which contains the monitoring data for the computer.

Example 1

```
Get-MDTMonitorData -Path DS001:
```

Description

This example returns the monitoring data for all deployments in the deployment share that is specified in the DS001: MDTProvider Windows PowerShell drive.

Example 2

```
cd DS001:  
Get-MDTMonitorData
```

Description

This example returns the monitoring data for all deployments in the deployment share that is specified in the DS001: MDTProvider Windows PowerShell drive. Use the **cd** command to set the working directory for Windows PowerShell to the DS001: MDTProvider Windows PowerShell drive.

Example 3

```
Get-MDTMonitorData -Path DS001: -ID 22
```

Description

This example returns the monitoring data for the deployment with an ID of 22 in the deployment share that is specified in the DS001: MDTProvider Windows PowerShell drive.

Get-MDTOperatingSystemCatalog

This section describes the **Get-MDTOperatingSystemCatalog** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Get-MDTOperatingSystemCatalog [-ImageFile] <String> [-Index]
<Int32> [<CommonParameters>]
```

Description

This cmdlet retrieves or creates an operating system catalog for a custom operating system image so that you can modify the corresponding unattend.xml file using Windows System Image Manager (WSIM). If no operating system catalog is available or if the existing operating system catalog is invalid or out of date, this cmdlet will generate a new operating system catalog.

Note The process of generating a new operating system catalog may take a long time as the custom operating system image must be mounted, inspected, and unmounted before the operating system catalog creation completes.

Parameters

This subsection provides information about the various parameters that can be used with the **Get-MDTOperatingSystemCatalog** cmdlet.

-ImageFile <String>

This parameter specifies the fully qualified path to the custom operating system image file (.wim file), including the name of the custom operating system image file.

Parameter	Value
Required?	True
Position?	2 and Named
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

-Index <Int32>

This parameter specifies the index of the desired operating system image within the operating system image file (.wim file).

Parameter	Value
Required?	True
Position?	3 and Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSObject** type object that contains the path to the operating system catalog.

Example 1

```
Get-MDTOperatingSystemCatalog -ImageFile "DS001:\OperatingSystems\Windows 8\sources\install.wim" -Index 2
```

Description

This example returns the operating system catalog for the operating system image at the index of 2 in the operating system image file DS001:\OperatingSystems\Windows 8\sources\install.wim.

Get-MDTPersistentDrive

This section describes the **Get-MDTPersistentDrive** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see “Loading the MDT Windows PowerShell Snap-In”.

Syntax

```
Get-MDTPersistentDrive [<CommonParameters>]
```

Description

This cmdlet displays the list of persisted MDT Windows PowerShell drives. The list of persisted MDT Windows PowerShell drives is managed using the [Add-MDTPersistentDrive](#) and [Remove-MDTPersistentDrive](#) cmdlets or the Deployment Workbench.

The output from this cmdlet contains the following information:

- Windows PowerShell drive name, such as DS001
- Directory path, such as \\WDG-MDT-01\DeploymentShare\$

Persisted MDT Windows PowerShell drives are similar to persisted network drive mappings.

Note This list of persisted MDT Windows PowerShell drives is maintained on a per user basis and are stored in the user profile.

Parameters

This subsection provides information about the various parameters that can be used with the **Get- MDTPersistentDrive** cmdlet.

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, "about_CommonParameters," which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSObject** type object for each MDT persisted drive that is identical to the **PSObject** type object that the [New-PSDrive](#) cmdlet returns.

Example 1

```
Get-MDTPersistentDrive
```

Description

This example displays a list of the MDT persisted drives.

Import-MDTApplication

This section describes the **Import-MDTApplication** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Import-MDTApplication [-Path <String>] -Name  
<String> -ApplicationSourcePath <String> -DestinationFolder  
<String> [-Move] [<CommonParameters>]
```

-or-

```
Import-MDTApplication [-Path <String>] -Name <String> -NoSource  
[<CommonParameters>]
```

-or-

```
Import-MDTApplication [-Path <String>] -Name <String> -Bundle  
[<CommonParameters>]
```

Description

This cmdlet imports an application into a deployment share. The following application types can be imported using this cmdlet:

- Applications that have source files, using the *ApplicationSourcePath*, *DestinationFolder*, and *Move* parameters. The first syntax example illustrates the use of this cmdlet for this type of application.
- Applications without source files or with source files located on other network shared folders using the *NoSource* parameter. The second syntax example illustrates the use of this cmdlet for this type of application.
- Application bundles, which are used to group a set of related applications, using the *Bundle* parameter. The last syntax example illustrates the use of this cmdlet for this type of application.

Parameters

This subsection provides information about the various parameters that can be used with the **Import-MDTApplication** cmdlet.

-Path <String>

This parameter specifies the fully qualified path to an existing folder where the application being imported will be placed within the deployment share. If the *DestinationFolder* parameter is used, then the folder specified in the *DestinationFolder* parameter is created beneath the folder specified in this parameter. This parameter is used in all syntax usages for this cmdlet.

Note If this parameter is not provided, the Windows PowerShell working directory must default to the desired location within the deployment share.

Parameter	Value
Required?	False
Position?	Named
Default value	–

Parameter	Value
Accept pipeline input?	False
Accept wildcard characters?	False

-Name <String>

This parameter specifies the name of the application to be added to the deployments share and must be unique within the deployment share. This parameter is used in all syntax usages for this cmdlet.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-ApplicationSourcePath <String>

This parameter specifies the fully qualified path to the application source files for the application that will be imported into the deployment share. This parameter is only valid for use in the first syntax example.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-DestinationFolder <String>

This parameter specifies the folder in the deployment share where the application source files are to be imported. This folder is created beneath the folder specified in the *Path* parameter. This parameter is only valid for use in the first syntax example.

Parameter	Value

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Move [*<SwitchParameter>*]

This parameter specifies whether the application's source files should be moved (instead of copied) from the folder where the application's source files are located, which is specified in the *ApplicationSourcePath* parameter.

If this parameter is:

- Specified, then the files are moved and the files in the folder specified in the *ApplicationSourcePath* parameter are deleted
- Not specified, then the files are copied and the files in the folder specified in the *ApplicationSourcePath* parameter are retained

This parameter is only valid for use in the first syntax example.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-NoSource [*<SwitchParameter>*]

This parameter specifies that the application being imported is an application that has no source files to be copied. When using this parameter, the application source files are:

- On a network shared folder, which is specified in the application installation command line or working directory configuration settings
- Already present in the operating system image

This parameter is only valid for use in the second syntax example.

Parameter	Value

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

-Bundle [<SwitchParameter>]

This parameter specifies that the application being imported is an application that is a bundle of two or more applications. This parameter is only valid for use in the last syntax example.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSSObject** type object that references the application just imported.

Example 1

```
Import-MDTApplication -Path "DS001:\Applications" -Name "Office 2010 Professional Plus 32-bit" -ApplicationSourcePath "\\\WDG-MDT-01\Source$\Office2010ProPlus\x86" -DestinationFolder "Office2010ProPlusx86"
```

Description

This example imports an application with source files from the network shared folder at \\WDG-MDT-01\Source\$\Office2010ProPlus\x86 and copies the source files to DS001:\Applications\Office2010ProPlusx86 within the deployment share. The source files are retained.

Example 2

```
Import-MDTApplication -Path "DS001:\Applications" -Name "Office 2010 Professional Plus 32-bit" -ApplicationSourcePath "\\WDG-MDT-01\Source$\Office2010ProPlus\x86" -DestinationFolder "Office2010ProPlusx86" -Move
```

Description

This example imports an application with source files from the network shared folder at \\WDG-MDT-01\Source\$\Office2010ProPlus\x86 and moves the source files to DS001:\Applications\Office2010ProPlusx86 within the deployment share. The source files are removed from the network shared folder at \\WDG-MDT-01\Source\$\Office2010ProPlus\x86. The application is named *Office 2012 Professional Plus 32-bit*.

Example 3

```
Import-MDTApplication -Path "DS001:\Applications" -Name "Office 2010 Professional Plus 32-bit" -NoSource
```

Description

This example imports an application named *Office 2012 Professional Plus 32-bit* with no source files.

Example 4

```
Import-MDTApplication -Path "DS001:\Applications" -Name "Woodgrove Bank Core Applications" -Bundle
```

Description

This example imports an application bundle named *Woodgrove Bank Core Applications*.

Import-MDTDriver

This section describes the **Import-MDTDriver** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Import-MDTDriver [-Path <String>] -SourcePath <String[]> [-ImportDuplicates] [<CommonParameters>]
```

Description

This cmdlet imports one or more device drivers into a deployment share. This cmdlet searches for device drivers starting at the folder specified in the *SourcePath* parameter. This cmdlet will locate multiple device drivers found in that folder structure.

Parameters

This subsection provides information about the various parameters that can be used with the **Import-MDTDriver** cmdlet.

-Path <String>

This parameter specifies the fully qualified path to an existing folder where the device driver being imported will be placed within the deployment share.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the desired location within the deployment share. This parameter must be provided if the *SourcePath* parameter is not provided.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-SourcePath <String[]>

This parameter specifies one or more fully qualified paths in a string array for the source folders where the device driver files are located. Each folder structure, starting with the folder specified in this parameter, is searched for device drivers, including all subfolders and the contents of .cab files in the folder structure.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the folder where the device driver files are located. This parameter must be provided if the *Path* parameter is not provided.

Parameter	Value
Required?	True
Position?	1 and Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-ImportDuplicates [<SwitchParameter>]

This parameter specifies whether this cmdlet should import duplicate device drivers. By default, duplicate device drivers are not imported. Duplicate device drivers are detected by calculating a hash values for all the files in a device driver folder. If the calculated hash value matches another device driver, the device driver to be imported is considered a duplicate.

If a duplicate driver is detected and this parameter is not provided, the device driver will be added and linked to the original, existing device driver.

If this parameter is:

- Specified, then the duplicate device drivers are imported
- Not specified, then the device drivers will be added and linked to the original, existing device drivers

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs one or more **PSSObject** type objects (one for each device driver imported).

Example 1

```
Import-MTDriver -Path
"DS001:\Out-of-Box Drivers" -SourcePath "\\\WDG-MDT-
01\Source$\Drivers"
```

Description

This example imports all device drivers in the folders structure with the root of the folder structure at \\WDG-MDT-01\Source\$\Drivers. The device drivers are stored in the Out-of-Box Drivers folder in the deployment share that is mapped to the DS001: MDTProvider Windows PowerShell drive. If any duplicate device drivers are detected, the device drivers will be added and linked to the original, existing device drivers in the deployment share.

Example 2

```
$DriverSourcePath="\\WDG-MDT-01\Source$\VendorADrivers", "\\WDG-MDT-01\Source$\VendorBDrivers"  
Import-MDTDriver -Path "DS001:\Out-of-Box Drivers" -SourcePath  
$DriverSourcePath -ImportDuplicates
```

Description

This example imports all device drivers in the folders structure specified in the string array \$DriverSourcePath. The device drivers are stored in the Out-of-Box Drivers folder in the deployment share that is mapped to the DS001: MDTProvider Windows PowerShell drive. If any duplicate device drivers are detected, the duplicate device drivers are imported.

Import-MDTOperatingSystem

This section describes the **Import-MDTOperatingSystem** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Import-MDTOperatingSystem [-Path <String>] -SourcePath <String>  
[-DestinationFolder <String>] [-Move] [<CommonParameters>]
```

—or—

```
Import-MDTOperatingSystem [-Path <String>] [-DestinationFolder  
<String>] -SourceFile <String> [-SetupPath <String>] [-Move]  
[<CommonParameters>]
```

—or—

```
Import-MDTOperatingSystem [-Path <String>] -WDSServer <String>  
[<CommonParameters>]
```

Description

This cmdlet imports an operating system into a deployment share. The following operating system types can be imported using this cmdlet:

- Operating systems from the original source files, using the *SourcePath* parameters. The first syntax example illustrates the use of this cmdlet for this type of operating system import.
- Custom operating systems image files, such as capture images from reference computers, using the *SourceFile* parameter. The second syntax example illustrates the use of this cmdlet for this type of operating system import.
- Operating system images that are present in Windows Deployment Services using the *WDSServer* parameter. The last syntax example illustrates the use of this cmdlet for this type of operating system import.

Parameters

This subsection provides information about the various parameters that can be used with the **Import-MDTOperatingSystem** cmdlet.

-Path <String>

This parameter specifies the fully qualified path to an existing folder within the deployment share where the operating system being imported will be placed. If the *DestinationFolder* parameter is used, then the folder specified in the *DestinationFolder* parameter is created beneath the folder specified in this parameter. This parameter is used in all syntax usages for this cmdlet.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the desired location within the deployment share.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-SourcePath <String>

This parameter specifies the fully qualified path to the operating system source files for the operating system that will be imported into the deployment share. This parameter is only valid for use in the first syntax example.

Parameter	Value
Required?	True
Position?	Named

Parameter	Value
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-DestinationFolder <String>

This parameter specifies the folder in the deployment share where the operating system source files are to be imported. This folder is created beneath the folder specified in the *Path* parameter. This parameter is only valid for use in the first and second syntax examples.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Move [<SwitchParameter>]

This parameter specifies if the operating system source files should be moved (instead of copied) from the folder where the operating system source files are located, which is specified in the *DestinationFolder* parameter.

If this parameter is:

- Specified, then the files are moved and the files in the folder specified in the *DestinationFolder* parameter are deleted
- Not specified, then the files are copied and the files in the folder specified in the *DestinationFolder* parameter are retained

This parameter is only valid for use in the first and second syntax examples.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-SourceFile <String>

This parameter specifies the fully qualified path to the operating system source .wim file for the operating system that will be imported into the deployment share. This parameter is only valid for use in the second syntax example.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-SetupPath <String>

This parameter specifies the fully qualified path to the operating system setup files that need to be imported along with the .wim file specified in the *SourceFile* parameter. This parameter is only valid for use in the second syntax example.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-WDSServer <String>

This parameter specifies the name of the Windows Deployment Services server on which the operating system image files to be imported are located. All operating image files on the Windows Deployment Services server will be imported into the deployment share. The actual operating system image files are not copied to the deployment share. Instead, the deployment share contains a link to each operating system file on the Windows Deployment Services server.

This parameter is only valid for use in the last syntax example.

Parameter	Value
Required?	False

Parameter	Value
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs one or more **PSSObject** type objects (one for each operating system that was imported).

Example 1

```
Import-MDTOperatingSystem -Path "DS001:\Operating Systems" -SourcePath "\\\WDG-MDT-01\Source$\Windows8" -DestinationFolder "Windows8x64"
```

Description

This example imports an operating system from the network shared folder at \\WDG-MDT-01\Source\$\Windows8 and copies the source files to DS001:\Operating Systems\Windows8x64 within the deployment share. The source files are retained.

Example 2

```
Import-MDTOperatingSystem -Path "DS001:\Operating Systems" -SourcePath "\\\WDG-MDT-01\Source$\Windows8" -DestinationFolder "Windows8x64" -Move
```

Description

This example imports an operating system from the network shared folder at \\WDG-MDT-01\Source\$\Windows8 and copies the source files to DS001:\Operating Systems\Windows8x64 within the deployment share. The source files are removed from the network shared folder at \\WDG-MDT-01\Source\$\Windows8.

Example 3

```
Import-MDTOperatingSystem -Path "DS001:\Operating Systems" -DestinationFolder "Windows8x64-Reference" -SourceFile "\\\WDG-MDT-01\Capture$\WDG-REF-01_Capture.wim"
```

Description

This example imports an operating system captured, custom image file (.wim file) from \\WDG-MDT-01\ Capture\$\WDG-REF-01_Capture.wim and copies the image file to DS001:\Operating Systems\Windows8x64-Reference within the deployment share. The source .wim file is retained on the network shared folder.

Example 4

```
Import-MDTOperatingSystem -Path "DS001:\Operating Systems" -WDServer "WDG-WDS-01"
```

Description

This example imports all the operating system images from the Windows Deployment Services server named WDG-WDS-01 and creates a link to each operating system image in DS001:\Operating Systems within the deployment share. The source operating system image files on the Windows Deployment Services server are retained on the Windows Deployment Services server.

Import-MDTPackage

This section describes the **Import-MDTPackage** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Import-MDTPackage [-Path <String>] [[-SourcePath] <String[]>] [<CommonParameters>]
```

Description

This cmdlet imports one or more operating system packages into a deployment share. The types of operating system packages that can be imported include security updates, language packs, or new components. Service packs should not be imported as operating system packages as they cannot be installed offline.

Parameters

This subsection provides information about the various parameters that can be used with the **Import-MDTPackage** cmdlet.

-Path <String>

This parameter specifies the fully qualified path to an existing folder within the deployment share where the operating system packages being imported will be placed.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the desired location within the deployment share.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-SourcePath <String>

This parameter specifies the fully qualified path to a folder structure to be scanned for operating system packages to import. The specified folder structure will be scanned for .cab and .msu files. For .msu files, the .cab files inside the .msu files are automatically extracted.

Parameter	Value
Required?	True
Position?	1 and Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSSObject** type object that references the package just imported.

Example 1

```
Import-MDTOperatingSystem -Path "DS001:\Packages" -SourcePath  
"\WDG-MDT-01\Source$\OSPackages"
```

Description

This example scans network shared folder at \\WDG-MDT-01\Source\$\OSPackages for operating system packages and copies the source files to DS001:\Packages folder within the deployment share. The source files are removed from the network shared folder at \\WDG-MDT-01\Source\$\OSPackages.

Import-MDTTaskSequence

This section describes the **Import-MDTTaskSequence** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Import-MDTTaskSequence [-Path <String>] -Template <String> -Name  
<String> -ID <String> [[-Comments] <String>] [[-Version]  
<String>] [-OperatingSystemPath <String>] [-OperatingSystem  
<PSObject>] [-FullName <String>] [-OrgName <String>] [-HomePage  
<String>] [-ProductKey <String>] [-OverrideProductKey <String>]  
[-AdminPassword <String>] [<CommonParameters>]
```

Description

This cmdlet imports a task sequence into a deployment share. The newly imported task sequence will be based on an existing task sequence template specified in the *Template* property.

Parameters

This subsection provides information about the various parameters that can be used with the **Import-MDTPackage** cmdlet.

-Path <String>

This parameter specifies the fully qualified path to an existing folder within the deployment share where the task sequence being imported will be placed. By default, the path should point to the Control folder and or a subfolder of the Control folder in the deployment share. The value of the *ID* parameter will be used to create a subfolder within the path specified in this parameter.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the desired location within the deployment share.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Template <String>

This parameter specifies the task sequence template to be used for importing the new task sequence. Task sequence templates are .xml files that contain the task sequence steps for a particular type of task sequence. If the task sequence template is located in:

- The *installation_folder\Templates* folder (where *installation_folder* is the folder in which MDT is installed), then only the .xml file name is required.
- Another folder, then the fully qualified path, including the name of the task sequence template .xml, is required.

For more information on the task sequence templates that are included with MDT for LTI deployments, see the section "Create a New Task Sequence in the Deployment Workbench" in the MDT document, *Using the Microsoft Deployment Toolkit*.

Parameter	Value
Required?	True
Position?	1 and Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Name <String>

This parameter specifies the name of the task sequence to be imported. The value of this parameter must be unique within the deployment share.

Parameter	Value
Required?	True
Position?	2 and Named

Parameter	Value
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

-ID <String>

This parameter specifies the identifier of the task sequence to be imported. The value of this parameter must be unique within the deployment share. The value assigned to this parameter should be in uppercase and not have any spaces or special characters. This value is used to create a subfolder in the folder specified in the *Path* parameter, which should be under the Control folder in the deployment share.

Parameter	Value
Required?	True
Position?	3 and Named
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

-Comments <String>

This parameter specifies the text that provides additional, descriptive information about the task sequence to be imported. This descriptive information is visible in the Deployment Workbench.

Parameter	Value
Required?	False
Position?	4 and Named
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

-Version <String>

This parameter specifies the version number of the task sequence to be imported. The value of this parameter is informational only and is not used by MDT for version-related processing.

Parameter	Value
Required?	False
Position?	4 and Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-OperatingSystemPath <String>

This parameter specifies the fully qualified Windows PowerShell path to the folder in the deployment share that contains the operating system to be used with this task sequence, such as DS001:\Operating Systems\Windows 8. The operating system must already exist in the deployment share where the task sequence is being imported.

Note If you do not provide this parameter and the task sequence needs to reference an operating system, then you must provide the *OperatingSystem* parameter.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-OperatingSystem <PSObject>

This parameter specifies the operating system object to be used with this task sequence. The operating system must already exist in the deployment share where the task sequence is being imported.

You can retrieve the Windows PowerShell object for an operating system using the **Get-Item** cmdlet, such as the following example:

```
$OS=Get-Item "DS001:\Operating Systems\Windows 8"
```

For more information on the **Get-Item** cmdlet, see [Using the Get-Item Cmdlet](#).

Note If you do not provide this parameter and the task sequence needs to reference an operating system, then you must provide the *OperatingSystemPath* parameter.

Parameter	Value

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-FullName <String>

This parameter specifies the name of the registered owner of the operating system to be used with this task sequence. This name is saved in the **RegisteredOwner** registry key at **HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion**. The value of this parameter is injected into the Unattend.xml file to be associated with this task sequences.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-OrgName <String>

This parameter specifies the name of the organization for the registered owner of the operating system to be used with this task sequence. This name is saved in the **RegisteredOrganization** registry key at **HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion**. The value of this parameter is injected into the Unattend.xml file to be associated with this task sequences.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-HomePage <String>

This parameter specifies the URL to be used as the home page in Internet Explorer. The value of this parameter is injected into the Unattend.xml file to be associated with this task sequences.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-ProductKey <String>

This parameter specifies the product key to be used for the operating system to be used with this task sequence. This product key is valid only for retail versions of Windows operating systems. The value of this parameter is injected into the Unattend.xml file to be associated with this task sequences.

Note If this parameter is not provided, then the product key must be provided when deploying this task sequence in the Deployment Wizard, in the CustomSettings.ini file, or in the MDT DB.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-OverrideProductKey <String>

This parameter specifies the MAK key to be used for the operating system to be used with this task sequence. This product key is valid only for volume license versions of Windows. The value of this parameter is injected into the Unattend.xml file to be associated with this task sequences.

Note If this parameter is not provided, then the MAK key must be provided when deploying this task sequence in the Deployment Wizard, in the CustomSettings.ini file, or in the MDT DB.

Parameter	Value
Required?	False

Parameter	Value
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-AdminPassword <String>

This parameter specifies the password to be assigned to the built-in, local Administrator account on the target computer. The value of this parameter is injected into the Unattend.xml file to be associated with this task sequences.

Note If this parameter is not provided, then the password to be assigned to the built-in, local Administrator account on the target computer must be provided when deploying this task sequence in the Deployment Wizard, in the CustomSettings.ini file, or in the MDT DB.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSSObject** type object that references the task sequence just imported.

Example 1

```
Import-MDTTaskSequence -Path "DS001:\Control" -Template
"Client.xml" -Name "Deploy Windows 8 to Reference Computer" -ID
"WIN8REFERENCE" -Comments "Task sequence for deploying Windows 8
to the reference computer (WDG-REF-01)" -Version "1.00" -
OperatingSystemPath "DS001:\Operating Systems\Windows 8_x64" -
```

```
FullName "Woodgrove Bank Employee" -OrgName "Woodgrove
Bank" -HomePage
"http://www.woodgrovebank.com" -OverrideProductKey
"12345-12345-12345-12345-12345" -AdministratorPassword "P@ssw0rd"
```

Description

This example imports a task sequence named *Deploy Windows 8 to Reference Computer* and creates the task sequence in the DS001:\Control\WIN8REFERENCE folder in the deployment share. The comment, “Task sequence for deploying Windows 8 to the reference computer (WDG-REF-01),” is assigned to the task sequence. The version number of the task sequence is set to **1.00**.

The operating system associated with the task sequence is located at DS001:\Operating Systems\Windows 8_x64 in the deployment share. The registered owner of the operating system will be set to **Woodgrove Bank Employee**. The registered organization of the operating system will be set to **Woodgrove Bank**. The Internet Explorer home page will default to <http://www.woodgrovebank.com>. The password for the local, built-in Administrator account will be set to a value of **P@ssw0rd**. The product key for the operating system will be set to **12345-12345-12345-12345-12345**.

Example 2

```
$OSObject = Get-Item "DS001:\Operating Systems\Windows 8_x64"
Import-MDTTaskSequence -Path "DS001:\Control" -Template
"Client.xml" -Name "Deploy Windows 8 to Reference Computer" -ID
"WIN8REFERENCE" -Comments "Task sequence for deploying Windows 8
to the reference computer (WDG-REF-01)" -Version "1.00" -
OperatingSystem $OSObject -FullName "Woodgrove Bank Employee" -
OrgName "Woodgrove Bank" -HomePage
"http://www.woodgrovebank.com" -AdministratorPassword "P@ssw0rd"
```

Description

This example imports a task sequence named *Deploy Windows 8 to Reference Computer* and creates the task sequence in the DS001:\Control\WIN8REFERENCE folder in the deployment share. The comment, “Task sequence for deploying Windows 8 to the reference computer (WDG-REF-01),” is assigned to the task sequence. The version number of the task sequence is set to **1.00**.

The operating system associated with the task sequence is located at DS001:\Operating Systems\Windows 8_x64 in the deployment share, which is passed to the cmdlet using the `$OSObject` variable. The `$OSObject` variable is set to an existing operating system object using the **Get-Item** cmdlet.

The registered owner of the operating system will be set to **Woodgrove Bank Employee**. The registered organization of the operating system will be set to **Woodgrove Bank**. The Internet Explorer home page will default to <http://www.woodgrovebank.com>. The password for the local, built-in

Administrator account will be set to a value of **P@ssw0rd**. The product key for the operating system will need to be provided when deploying this task sequence in the Deployment Wizard, in the CustomSettings.ini file, or in the MDT DB.

New-MDТDatabase

This section describes the **New-MDТDatabase** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
New-MDТDatabase [-Path <String>] [-Force] -SQLServer <String> [-Instance <String>] [-Port <String>] [-Netlib <String>] -Database <String> [-SQLShare <String>] [<CommonParameters>]
```

Description

This cmdlet creates a new MDT DB database that is associated with a deployment share. Each deployment share can be associated with only one MDT DB database.

Parameters

This subsection provides information about the various parameters that can be used with the **New-MDТDatabase** cmdlet.

-Path <String>

This parameter specifies the fully qualified Windows PowerShell path to the deployment share to which the new MDT DB database will be associated placed.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the desired location within the deployment share.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Force [<SwitchParameter>]

This parameter specifies that tables within the MDT DB should be recreated if the database specified in the *Database* parameter already exist. If this parameter is:

- Provided, then the tables within an existing MDT DB will be re-created
- Omitted, then the tables within an existing MDT DB will not be re-created

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

-SQLServer <String>

This parameter specifies the name of the computer running SQL Server where the new MDT DB database will be created.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Instance <String>

This parameter specifies the SQL Server instance in which the new MDT DB database will be created. If this parameter is omitted, the MDT DB database is created in the default SQL Server instance.

Note The SQL Browser service must be running on the computer running SQL Server for the cmdlet to locate the instance specified in this parameter.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Port <String>

This parameter specifies the TCP port to be used in communication with the SQL Server instance specified in the *SQLServer* parameter. The default port that SQL Server uses is 1433. Specify this parameter when SQL Server is configured to use a port other than the default value. The value of this parameter must match the port configured for SQL Server.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Netlib <String>

This parameter specifies the SQL network library used in communication with the SQL Server instance specified in the *SQLServer* parameter. The parameter can be set to one of the following values:

- **DBNMPNTW**, which is used to specify named pipes communication
- **DBSMSOCN**, which is used to specify TCP/IP sockets communication

If this parameter is not provided, the named pipes SQL network library (DBNMPNTW) is used.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Database <String>

This parameter specifies the name of the database to be created in the SQL Server instance specified in the *Instance* parameter on the SQL Server specified in the *SQLServer* parameter. The default location and naming convention will be used for the database and log files when creating the database.

If the database specified in this parameter already exists, the database will not be recreated. The tables within the database can be recreated based on the **Force** parameter.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-SQLShare <String>

This parameter specifies the name of a network shared folder on the computer where SQL Server is running. This connection is used to establish Windows Integrated Security connections using the Named Pipes protocol.

Note If this parameter is not included, then a secured IPC\$ connection is not established. As a result, named pipes communication with SQL Server may fail.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSObject** type object for the new MDT DB that was created.

Example 1

```
New-MDTDatabase -Path "DS001:" -SQLServer "WDG-SQL-01" -Database "MDTDB" -SQLShare "\\\WDG-SQL-01\MDTShare$"
```

Description

This example creates an MDT DB named *MDTDB* in the default SQL Server instance on a computer named *WDG-SQL-01*. If the database already exists, the tables in the existing database will not be recreated. The connection will be made using the default SQL Server TCP port and the Named Pipes protocol.

Example 2

```
New-MDTDatabase -Path "DS001:" -Force -SQLServer "WDG-SQL-01" -Instance "MDTInstance" -Database "MDTDB" -SQLShare "\\\WDG-SQL-01\MDTShare$"
```

Description

This example creates an MDT DB named *MDTDB* in the SQL Server instance named *MDTInstance* on a computer named *WDG-SQL-01*. If the database already exists, the tables in the existing database will be recreated. The connection will be made using the default SQL Server TCP port and the Named Pipes protocol.

Remove-MDTMonitorData

This section describes the **Get-MDTPersistentDrive** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Remove-MDTMonitorData [-Path <String>] [-ID <Int32>]  
[<CommonParameters>]
```

—or—

```
Remove-MDTMonitorData [-Path <String>] [-ComputerObject  
<PSObject>] [<CommonParameters>]
```

Description

This cmdlet removes collected monitoring data from the existing collected monitoring data in a deployment share. You can identify the monitoring data to remove by specifying the:

- Identifier (ID) of the monitoring item for a specific deployment share. The monitoring item IDs are automatically generated and assigned to the item when the item is created for the deployment share. The first syntax example illustrates this usage.

- Computer object for the monitoring item in the deployment share. The computer object can be obtained using the **Get-MDTMonitorData** cmdlet. The last syntax example illustrates this usage.

Note Once the monitoring data has been removed, there is no method for recovering the information.

Parameters

This subsection provides information about the various parameters that can be used with the **Get-MDTMonitorData** cmdlet.

-Path <String>

This parameter specifies the **MDTProvider** Windows PowerShell drive for the desired deployment share.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to a location within the desired MDTProvider Windows PowerShell drive.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-ID <Nullable>

This parameter specifies the monitoring data item to be removed using the identifier of the monitoring data item. If this parameter is not specified, then the *ComputerObject* parameter must be specified to identify a particular monitoring data item.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

-ComputerObject <PSObject>

This parameter specifies the monitoring data item to be removed using a computer object. If this parameter is not specified, then the *ID* parameter must be specified to identify a particular monitoring data item.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet may output a **String** type object if the *Verbose* common parameter is included; otherwise, no output is generated.

Example 1

```
Remove-MDTMonitorData -Path "DS001:" -ID 3
```

Description

This example removes the monitoring data item with an ID that has a value of **3** from the deployment share at the Windows PowerShell path DS001:.

Example 2

```
Remove-MDTMonitorData -ID 3
```

Description

This example removes the monitoring data item with an ID that has a value of **3** from the deployment share at the default Windows PowerShell path.

Example 3

```
$MonitorObject=Get-MDTMonitorData | Where-Object {$_.Name -eq 'WDG-REF-01'}
```

```
Remove-MDTMonitorData -ComputerObject $MonitorObject
```

Description

This example removes any monitoring data item where the name of the computer is WDG-REF-01. The object is found using the **Get-MDTMonitorData** cmdlet and the **Where-Object** cmdlet. For more information on the **Where-Object** cmdlet, see [Using the Where-Object Cmdlet](#).

Remove-MDTPersistentDrive

This section describes the **Remove-MDTPersistentDriveWindows** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Remove-MDTPersistentDrive [-Name] <String> [[-InputObject] <PSObject>] [<CommonParameters>]
```

Description

This cmdlet removes an existing Windows PowerShell drive created using the **MDTProvider** from the list of drives that are persisted in the Deployment Workbench or in a Windows PowerShell session using the [Restore-MDTPersistentDrive](#) cmdlet. This cmdlet is called when a deployment share is closed in (removed from) the Deployment Workbench.

Note The list of persisted **MDTProvider** drives is maintained on a per-user basis in the user profile.

The list of persisted **MDTProvider** drives can be displayed using the **Get-MDTPersistentDrive** cmdlet. An MDTProvider drive can be added to the list of persisted drives using the **Add-MDTPersistentDrive** cmdlet.

Parameters

This subsection provides information about the various parameters that can be used with the **Add-MDTPersistentDriveWindows** cmdlet.

-Name <String>

Specifies the name of a Windows PowerShell drive created using the MDT provider and corresponds to an existing deployment share. The name was created using the [New-PSDrive](#) cmdlet and specifying the **MDTProvider** in the **PSProvider** parameter.

For more information on how to create a new Windows PowerShell drive using the **MDTProvider** and how to create a deployment share using Windows PowerShell, see the section "Creating a Deployment Share Using Windows

PowerShell" in the MDT document, *Microsoft Deployment Toolkit Samples Guide*.

Parameter	Value
Required?	True
Position?	1 and Named
Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

-InputObject <PSObject>

This parameter specifies a Windows PowerShell drive object that was created earlier in the process. Enter a **PSObject** object, such as one generated by the [New-PSDrive](#) cmdlet.

Parameter	Value
Required?	False
Position?	2 and Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, "about_CommonParameters," which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet provides no outputs.

Example 1

```
Remove-MDTPersistentDrive -Name "DS001"
```

Description

This example removes the deployment share with the Windows PowerShell drive name of *DS001* from the list of persisted drives.

Example 2

```
$MDTPSDrive = Get-PSDrive | Where-Object {$_.Root -eq "C:\DeploymentShare" -and $_.Provider -like "*MDTProvider"}  
Remove-MDTPersistentDrive -InputObject $MDTPSDrive
```

Description

This example removes the deployment share at C:\DeploymentShare\$ from the list of persisted drives. The **Get-PSDrive** and **Where-Object** cmdlets are used to return the MDT persisted Windows PowerShell drive to the **Remove-MDTPersistentDrive** cmdlet using the \$MDTPSDrive variable. For more information on the **Where-Object** cmdlet, see [Using the Where-Object Cmdlet](#). For more information on the **Get-PSDrive** cmdlet, see [Using the Get-PSDrive Cmdlet](#).

Restore-MDTPersistentDrive

This section describes the **Restore-MDTPersistentDrive** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Restore-MDTPersistentDrive [-Force] [<CommonParameters>]
```

Description

This cmdlet restores a persisted MDT Windows PowerShell drive to the list of active Windows PowerShell drive for each deployment share that was added to the list of persisted MDT Windows PowerShell drives. The list of persisted MDT Windows PowerShell drives is managed using the [Add-MDTPersistentDrive](#) and [Remove-MDTPersistentDrive](#) cmdlets or the Deployment Workbench.

This cmdlet calls the **New-PSDrive** cmdlet to create a Windows PowerShell drive for each drive in the MDT persisted list. Persisted MDT Windows PowerShell drives are similar to persisted network drive mappings.

Note This list of persisted MDT Windows PowerShell drives is maintained on a per-user basis and are stored in the user profile.

Parameters

This subsection provides information about the various parameters that can be used with the **Restore-MDTPersistentDrive** cmdlet.

-Force [<SwitchParameter>]

This parameter specifies that the deployment share should be upgraded when restored (if required). If this parameter is:

- Provided, then the deployment share will be upgraded when restored (if required)
- Omitted, then deployment share will not be upgraded when restored

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSSObject** type object for each MDT Provider Windows PowerShell drive that is restored.

Example 1

```
Get-MDTPersistentDrive
```

Description

This example restores the list of MDT persisted drives, by creating a Windows PowerShell drive using the **MDTProvider** type. The deployment share will not be upgraded when restored.

Example 2

```
Get-MDTPersistentDrive -Force
```

Description

This example restores the list of MDT persisted drives, by creating a Windows PowerShell drive using the **MDTProvider** type. The deployment share will be upgraded when restored (if required).

Set-MDTMonitorData

This section describes the **Get-MDTPersistentDrive** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Set-MDTMonitorData [-Path <String>] [-ComputerObject <PSObject>]  
[-Settings <Hashtable>] [<CommonParameters>]
```

–or–

```
Set-MDTMonitorData [-Path <String>] [-MacAddress <String>]  
[-Settings <Hashtable>] [<CommonParameters>]
```

Description

This cmdlet creates a new monitoring data item, or updates an existing monitoring data item, in a deployment share. You can identify the monitoring data to remove by specifying the:

- Computer object for the monitoring item in the deployment share. The computer object can be obtained using the **Get-MDTMonitorData** cmdlet. The first syntax example illustrates this usage.
- MAC address of the primary network adapter of the monitoring item for a specific deployment share. The MAC address is automatically assigned to the monitoring data item when the item is created for the deployment share. The last syntax example illustrates this usage.

Note Once the monitoring data has been removed, there is no method for recovering the information.

Parameters

This subsection provides information about the various parameters that can be used with the **Get-MDTMonitorData** cmdlet.

-Path <String>

This parameter specifies the MDTProvider Windows PowerShell drive for the desired deployment share.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to a location within the desired MDTProvider Windows PowerShell drive.

Parameter	Value
Required?	False
Position?	Named

Parameter	Value
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-ComputerObject <PSObject>

This parameter specifies the monitoring data item to be created or updated using a computer object. If this parameter is not specified, then the *MACAddress* parameter must be specified to identify a particular monitoring data item.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

-MACAddress <String>

This parameter specifies the monitoring data item to be created or updated using the MAC address of the primary network adapter of the computer being monitored. The format of the MACAddress is *xx:xx:xx:xx:xx:xx*, where *x* is a hexadecimal character specified in uppercase (as required). If this parameter is not specified, then the *ComputerObject* parameter must be specified to identify a particular monitoring data item.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

-Settings <Hashtable>

This parameter specifies the monitoring data settings for the monitoring data item to be created or updated. The format of the hashtable provided with this parameter is @{ "Setting"="Value"; "Setting1"="Value1";

"Setting2" = "Value2} . If this parameter is not specified, then the monitoring data item is created, but no monitoring information is stored.

"Setting" can be any property listed in the ZTIGather.xml file. Value can be any valid value for the property specified in "Setting".

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, "about_CommonParameters," which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet does not generate any output.

Example 1

```
$MonitorObject=Get-MDTMonitorData | Where-Object {$_.Name -eq 'WDG-REF-01'}
Set-MDTMonitorData -ComputerObject $MonitorObject -Setting @{
    "OSDComputerName" = "WDG-MDT-01";
    "SkipWizard" = "YES"
}
```

Description

This example removes any monitoring data item where the name of the computer is **WDG-REF-01**. The object is found using the **Get-MDTMonitorData** cmdlet and the **Where-Object** cmdlet. For more information on the **Where-Object** cmdlet, see [Using the Where-Object Cmdlet](#). The [OSDComputerName](#) property is recorded as having a value of **WDG-MDT-01**, and the [SkipWizard](#) property is recorded as having a value of **YES**.

Example 2

```
Set-MDTMonitorData -MACAddress "00:11:22:33:44:55"
MonitorObject -Setting @{
    "OSDComputerName" = "WDG-MDT-01";
    "SkipWizard" = "YES"
}
```

Description

This example creates or updates a monitoring data item with a **MACAddress** that has a value of **00:11:22:33:44:55**. The [OSDComputerName](#) property is recorded as having a value of **WDG-MDT-01**, and the [SkipWizard](#) property is recorded as having a value of **YES**.

Test-MDTDeploymentShare

Although this cmdlet is returned using the **Get-Command** cmdlet as being in the Microsoft.BDD.PSSnapIn snap-in, it is not implemented.

Test-MDTMonitorData

This section describes the **Test-MDTMonitorData** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Test-MDTMonitorData -ServerName <String> -EventPort <Int32> -  
DataPort <Int32> [<CommonParameters>]
```

Description

This cmdlet validates if the MDT monitoring service, which runs on the computer on which MDT is installed, is enabled and running properly. The MDT monitoring service collects monitoring information that can be displayed:

- In the Monitoring node in a deployment share in the Deployment Workbench
- Using the [Get-MDTMonitorData](#) cmdlet

The MDT monitoring service can be disabled using the [Disable-MDTMonitorService](#). Monitoring information can be written to the MDT monitoring service using the [Set-MDTMonitorData](#) cmdlet.

Note For this cmdlet to function properly there must be at least one MDT monitoring data item in the deployment share. If no MDT monitoring information has been recorded, the deployment share will fail the test.

For more information on the MDT monitoring service, see the section "Monitoring MDT Deployments" in the MDT document, *Using the Microsoft Deployment Toolkit*.

Parameters

This subsection provides information about the various parameters that can be used with the **Test-MDTMonitorData** cmdlet.

-Server <String>

Specifies the name of the computer on which MDT is installed and the MDT monitoring service is running.

Parameter	Value
Required?	True
Position?	Named
Default value	None
Accept pipeline input?	False
Accept wildcard characters?	False

-DataPort <Int32>

This parameter specifies the TCP port used as the data port for the MDT monitoring service.

Parameter	Value
Required?	True
Position?	Named
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

-EventPort <Int32>

This parameter specifies the TCP port used as the event port for the MDT monitoring service.

Parameter	Value
Required?	True
Position?	Named
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and

WarningVariable. For more information, see the topic, "about_CommonParameters," which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a Boolean value that represents the success (true) or failure (false) of the text.

Example 1

```
Test-MDTMonitorData -Server "WDG-MDT-01" -DataPort  
"9801" -EventPort "9800"
```

Description

This example verifies if the MDT monitoring service on WDG-MDT-01 is installed and running. The cmdlet will verify using a data port of 9801 and an event port of 9800.

Update-MDTDatabaseSchema

This section describes the **Update-MDTDatabaseSchema** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Update-MDTDatabaseSchema -SQLServer <String> [-Instance <String>]  
[-Port <String>] [-Netlib <String>] -Database <String> [-SQLShare  
<String>] [<CommonParameters>]
```

Description

This cmdlet updates an existing MDT DB database to the latest version of the MDT DB database schema. Each deployment share can be associated with only one MDT DB database.

This cmdlet is automatically called when a deployment share is being upgraded, such as when running the [Restore-MDTPersistentDrive](#) cmdlet with the *Force* parameter and the [Update-MDTDeploymentShare](#) cmdlet.

Parameters

This subsection provides information about the various parameters that can be used with the **Upgrade-MDTDatabaseSchema** cmdlet.

-SQLServer <String>

This parameter specifies the name of the computer running SQL Server where the MDT DB database will be upgraded.

Parameter	Value
Required?	True
Position?	Named
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

-Instance <String>

This parameter specifies the SQL Server instance on which the MDT DB database to be upgraded exists. If this parameter is omitted, then the MDT DB database is assumed to be in the default SQL Server instance.

Note The SQL Browser service must be running on the computer running SQL Server for the cmdlet to locate the instance specified in this parameter.

Parameter	Value
Required?	False
Position?	Named
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

-Port <String>

This parameter specifies the TCP port to be used in communication with the SQL Server instance specified in the *SQLServer* parameter. The default port that SQL Server uses is 1433. Specify this parameter when SQL Server is configured to use a port other than the default value. The value of this parameter must match the port configured for SQL Server.

Parameter	Value
Required?	False
Position?	Named
Default value	–
Accept pipeline input?	False

Parameter	Value
Accept wildcard characters?	False

-Netlib <String>

This parameter specifies the SQL network library that is used in communication with the SQL Server instance specified in the *SQLServer* parameter. The parameter can be set to one of the following values:

- **DBNMPNTW**, which is used to specify named pipes communication
- **DBSMSOCN**, which is used to specify TCP/IP sockets communication

If this parameter is not provided, the named pipes SQL network library (**DBNMPNTW**) is used.

Note The Deployment Workbench does not provide the option for configuring the SQL network library. The Deployment Workbench always uses named pipes communication. However, the SQL network library can be configured in the *CustomSettings.ini* file.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Database <String>

This parameter specifies the name of the database to be upgraded in the SQL Server instance specified in the *Instance* parameter on the SQL Server instance specified in the *SQLServer* parameter.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, "about_CommonParameters," which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **PSObject** type object for the MDT database that was upgraded. This cmdlet also outputs a **String** type data if the *Verbose* common parameter is included.

Example 1

```
Update-MTDatabaseSchema -SQLServer "WDG-SQL-01" -Database "MDTDB"
```

Description

This example updates the schema for an MDT database named *MDTDB* in the default SQL Server instance on a computer named *WDG-SQL-01*. The connection will be made to the SQL Server instance using the default TCP port and the Named Pipes protocol.

Example 2

```
Update-MTDatabaseSchema -SQLServer "WDG-SQL-01" -Instance "MDTInstance" -Port "6333" -Database "MDTDB"
```

Description

This example updates the schema for an MDT database named *MDTDB* in the SQL Server instance named *MDTInstance* on a computer named *WDG-SQL-01*. The connection will be made to the SQL Server using TCP port 6333 and the Named Pipes protocol.

Update-MTDeploymentShare

This section describes the **Update-MTDeploymentShare** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Update-MTDeploymentShare [-Path <String>] [-Force] [-Compress] [<CommonParameters>]
```

Description

This cmdlet updates an existing deployment share with the latest files from the Windows ADK. This cmdlet also updates or regenerates the required Windows PE boot images in both WIM and ISO file formats.

Parameters

This subsection provides information about the various parameters that can be used with the **Update-MDTDeploymentShare** cmdlet.

-Path <String>

This parameter specifies the fully qualified path to an existing folder in the deployment share that is being updated.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the desired location within the deployment share.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

-Force [<SwitchParameter>]

This parameter specifies whether Windows PE boot images (.iso and .wim files) for the deployment share should be completely regenerated. If this parameter is:

- Provided, then the cmdlet creates new versions of the Windows PE boot images. This process takes more time than optimizing the existing Windows PE boot images.
- Omitted, then the cmdlet optimizes the existing Windows PE boot images. This process takes less time than generating new versions of the Windows PE boot images. If this parameter is omitted, the *Compress* parameter can be used to reduce the size of the boot images as a part of the Windows PE boot image optimization process.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)

Parameter	Value
Accept wildcard characters?	False

-Compress [<SwitchParameter>]

This parameter specifies whether Windows PE boot images (.iso and .wim files) for the deployment share should be compressed when they are optimized (without the *Force* parameter). If this parameter is:

- Provided, then the cmdlet compresses the Windows PE boot images as they are being optimized
- Omitted, then the cmdlet does not compress the Windows PE boot images as they are being optimized

Note This parameter should only be provided if the *Force* parameter is not provided. If the *Force* parameter is included, new Windows PE boot images are generated and are compressed to the minimal size.

Parameter	Value
Required?	False
Position?	Named
Default value	—
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **String** type data and produces additional **String** type data if the *Verbose* common parameter is included.

Example 1

Update-MDTDeploymentShare

Description

This example updates the deployment share at the Windows PowerShell working directory. The Windows PE boot images will be optimized. The Windows PE boot images will not be compressed.

Example 2

```
Update-MDTDeploymentShare -Path "DS001:"
```

Description

This example updates the deployment share at the MDT Windows PowerShell drive named *DS001*:. The Windows PE boot images will be optimized. The Windows PE boot images will not be compressed.

Example 3

```
Update-MDTDeploymentShare -Path "DS001:" -Compress
```

Description

This example updates the deployment share at the MDT Windows PowerShell drive named *DS001*:. The Windows PE boot images will be optimized. The Windows PE boot images will be compressed.

Example 4

```
Update-MDTDeploymentShare -Path "DS001:" -Force
```

Description

This example updates the deployment share at the MDT Windows PowerShell drive named *DS001*:. New versions of the Windows PE boot images will be generated.

Update-MDTLinkedDS

This section describes the **Update-MDTLinkedDS** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Update-MDTLinkedDS -Path <String> [<CommonParameters>]
```

Description

This cmdlet replicates content from a deployment share to a linked deployment share using the selection profile used to define the linked deployment share. The replication behavior is determined based on the configuration settings for the linked deployment share.

Parameters

This subsection provides information about the various parameters that can be used with the **Update-MDTLinkedDS** cmdlet.

-Path <String>

This parameter specifies the fully qualified path to the linked deployment share that is being updated.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the desired location within the deployment share.

Parameter	Value
Required?	True
Position?	Named
Default value	—
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **String** type data and produces additional **String** type data if the *Verbose* common parameter is included.

Example 1

```
Update-MDTLinkedDS -Path "DS001:\Linked Deployment Shares\LINKED001"
```

Description

This example replicates content from the deployment share to the linked deployment share at the Windows PowerShell path DS001:\Linked Deployment Shares\LINKED001 folder.

Update-MDTMedia

This section describes the **Update-MDTMedia** Windows PowerShell cmdlet. Run this cmdlet from a Windows PowerShell console that has the MDT PowerShell snap-in loaded. For more information on how to start a Windows PowerShell console that has the MDT PowerShell snap-in loaded, see "Loading the MDT Windows PowerShell Snap-In".

Syntax

```
Update-MDTMedia -Path <String> [<CommonParameters>]
```

Description

This cmdlet replicates content from a deployment share to a folder that contains deployment media using the selection profile used to define the deployment media. The replication behavior is determined based on the configuration settings for the deployment media.

Media in LTI allows you to perform LTI deployments solely from local media without connecting to a deployment share. You can store the media on a DVD, USB hard disk, or other portable device. After you create the media, generate bootable WIM images that allow the deployment to be performed from portable media devices locally available on the target computer.

Parameters

This subsection provides information about the various parameters that can be used with the **Update-MDTMedia** cmdlet.

-Path <String>

This parameter specifies the fully qualified path to the folder that contains the deployment media that is being updated.

Note If this parameter is not provided, then the Windows PowerShell working directory must default to the desired location within the deployment share.

Parameter	Value
Required?	True
Position?	Named
Default value	–
Accept pipeline input?	False
Accept wildcard characters?	False

<CommonParameters>

This cmdlet supports the following common parameters: *Verbose*, *Debug*, *ErrorAction*, *ErrorVariable*, *OutBuffer*, *OutVariable*, *WarningAction*, and *WarningVariable*. For more information, see the topic, “about_CommonParameters,” which you can access by typing the following command, and then pressing ENTER:

```
Get-Help about_CommonParameters
```

Outputs

This cmdlet outputs a **String** type data and produces additional **String** type data if the *Verbose* common parameter is included.

Example 1

```
Update-MDTMedia -Path "DS001:\Media\MEDIA001"
```

Description

This example replicates content from the deployment share to the folder containing the deployment media at the Windows PowerShell path DS001:\Media\MEDIA001 folder.

Tables and Views in the MDT DB

In MDT, many property settings can be stored (typically configured in the CustomSettings.ini file) in a database. Configuring the properties in a database helps create a generic CustomSettings.ini file that requires fewer modifications and allows one CustomSettings.ini file to be used in more images (because the file is more generic).

Customize the database in the Database node in the Deployment Workbench. Using the Deployment Workbench, the deployment settings can be configured and saved in tables.

However, queries about the information in the tables are done using views. Views help simplify the queries by joining results from multiple tables.

ZTIGather.wsf queries the views to return the result set that the **Parameters** and **ParameterCondition** properties specify.

Tables in the MDT DB

The following table lists the database tables that Deployment Workbench creates and manages.

Table	Description
ComputerIdentity	Used to identify a specific computer using any combination of the AssetTag , UUID , SerialNumber , and MACAddress properties. The table includes a Description column to provide a user-friendly method of describing the computer (usually the computer name).
Descriptions	Contains descriptions of all properties configurable via the database.
LocationIdentity	Used to identify geographic locations using the Location property. The values for this property are stored in a corresponding column in the table.
LocationIdentity_DefaultGateway	Relates the default gateway values with a location identified in the LocationIdentity table. There is a one-to-many relationship between this table and the LocationIdentity table.
MakeModelIdentity	Used to identify a specific make and model of a computer using the Make and Model properties. The values for these properties are stored in corresponding columns in the table.

Table	Description
PackageMapping	Used to associate the name presented in the Add or Remove Programs Control Panel item with a Configuration Manager package and program to be deployed in place of the application in Add or Remove Programs. For more information on this table, see the section, "Deploying Applications Based on Earlier Application Versions", in the MDT document <i>Microsoft Deployment Toolkit Samples Guide</i> .
RoleIdentity	Used to identify the purpose of a computer or the users of a computer using the Role property. The values for this property are stored in a corresponding column in the table.
Settings	Identifies the settings that are applied to an individual computer or a group of computers based on the settings in the Computers, Roles, Locations, and Make and Model nodes in the Database node in the Deployment Workbench.
Settings_Administrators	Identifies the user accounts to be added to the local Administrator group on the target computer based on the settings in the Computers, Roles, Locations, and Make and Model nodes in the Database node in the Deployment Workbench.
Settings_Applications	Identifies the applications to be deployed to the target computer based on the settings in the Computers, Roles, Locations, and Make and Model nodes in the Database node in the Deployment Workbench.
Settings_Packages	Identifies the packages to be deployed to the target computer based on the settings in the Computers, Roles, Locations, and Make and Model nodes in the Database node in the Deployment Workbench.
Settings_Roles	Identifies the roles to be associated with the target computer based on the settings in the Computers, Locations, and Make and Model nodes in the Database node in the Deployment Workbench.

Views in the MDT DB

The following table lists and describes the database views that are used when querying configuration information in the MDT DB.

View	Description
ComputerAdministrators	Used to find all accounts to be made members of the local Administrators group on the target computer. The view is a join of the ComputerIdentity and Settings_Administrators tables.
ComputerApplications	Used to find all applications to be deployed to the target computer. The view is a join of the ComputerIdentity and Settings_Applications tables.
ComputerPackages	Used to find all packages to be deployed to the target computer. The view is a join of the ComputerIdentity and Settings_Packages tables.
ComputerRoles	Used to find all roles to be associated with the target computer. The view is a join of the ComputerIdentity and Settings_Roles tables.
ComputerSettings	Used to find all property settings to be configured for the target computer. The view is a join of the ComputerIdentity and Settings tables.
LocationAdministrators	Used to find all the accounts to be made a member of the local Administrators group on the target computers within a location. The view is a join of the LocationIdentity, LocationIdentity_DefaultGateway, and Settings_Administrators tables.
LocationApplications	Used to find all the applications to be deployed to the target computers within a location. The view is a join of the LocationIdentity, LocationIdentity_DefaultGateway, and Settings_Applications tables.
LocationPackages	Used to find all the packages to be deployed to the target computers within a location. The view is a join of the LocationIdentity, LocationIdentity_DefaultGateway, and Settings_Packages tables.
LocationRoles	Used to find all the roles to be associated with the target computers within a location. The

View	Description
	view is a join of the LocationIdentity, LocationIdentity_DefaultGateway, and Settings_Roles tables.
Locations	Used to find the IP addresses for the default gateways within a location or for all the locations that contain a specified IP address for a default gateway. The view is a join of the LocationIdentity and LocationIdentity_DefaultGateway tables.
LocationSettings	Used to find all the property settings to be configured for the target computers within a location. The view is a join of the LocationIdentity, LocationIdentity_DefaultGateway, and Settings tables.
MakeModelAdministrators	Used to find all accounts to be made members of the local Administrators group on the target computers with a given make and model. The view is a join of the MakeModelIdentity and Settings_Administrators tables.
MakeModelApplications	Used to find all applications to be deployed to the target computers with a given make and model. The view is a join of the MakeModelIdentity and Settings_Applications tables.
MakeModelPackages	Used to find all packages to be deployed to the target computers with a given make and model. The view is a join of the MakeModelIdentity and Settings_Applications tables.
MakeModelRoles	Used to find all roles associated with the target computers with a given make and model. The view is a join of the MakeModelIdentity and Settings_Roles tables.
MakeModelSettings	Used to find all property settings to be configured for the target computers with a given make and model. The view is a join of the MakeModelIdentity and Settings tables.
RoleAdministrators	Used to find all accounts to be made members of the local Administrators group on the target computers with a given role. The view is a join of the RoleIdentity and Settings_Administrators tables.

View	Description
RoleApplications	Used to find all applications to be deployed to the target computers with a given role. The view is a join of the RoleIdentity and Settings_Applications tables.
RolePackages	Used to find all packages to be deployed to the target computers with a given role. The view is a join of the RoleIdentity and Settings_Packages tables.
RoleSettings	Used to find all property settings to be configured for the target computers with a given role. The view is a join of the RoleIdentity and Settings tables.

Windows 7 Feature Dependency Reference

Table 8 lists the Windows 7 features, the parent feature, and any dependent features. You can use this information to determine which features and roles need to be installed to support a specific feature using the [Install Roles and Features](#) and [Uninstall Roles and Features](#) task sequence steps.

Table 8. Windows 7 Feature Dependency Reference

Feature	Parent Feature	Dependent Features
Windows Media® Center	Media Features	Might affect other Windows features
Windows DVD Maker	Media Features	Might affect other Windows features
Windows Media Player	Media Features	Might affect other Windows features
Windows Search	N/A	Might affect other Windows features
Internet Explorer (amd64)	N/A	Might affect other Windows features
World Wide Web services	Microsoft Internet Information Services (IIS)	<ul style="list-style-type: none">• Microsoft Message Queuing (MSMQ) HTTP support• Windows Communication Foundation (WCF) HTTP activation
IIS 6 WMI compatibility	IIS, Web management tools, IIS 6 management compatibility	IIS 6 scripting tooling
Microsoft .NET extensibility	IIS, World Wide Web services, application development features	<ul style="list-style-type: none">• Microsoft ASP.NET• MSMQ HTTP support• WCF HTTP activation
Default document	IIS, World Wide Web services, common HTTP features	MSMQ HTTP support
Directory browsing	IIS, World Wide Web services, common HTTP features	MSMQ HTTP support
HTTP redirection	IIS, World Wide Web services, common	MSMQ HTTP support

Feature	Parent Feature	Dependent Features
	HTTP features	
Static content	IIS, World Wide Web services, common HTTP features	<ul style="list-style-type: none"> • Web-based Distributed Authoring and Versioning (WebDAV) publishing • MSMQ HTTP support
Custom logging	IIS, World Wide Web services, health and diagnostics	MSMQ HTTP support
HTTP logging	IIS, World Wide Web services, health and diagnostics	MSMQ HTTP support
ODBC logging	IIS, World Wide Web services, health and diagnostics	MSMQ HTTP support
Request Monitor	IIS, World Wide Web services, health and diagnostics	MSMQ HTTP support
Tracing	IIS, World Wide Web services, health and diagnostics	MSMQ HTTP support
Static content compression	IIS, World Wide Web services, performance features	MSMQ HTTP support
Security	IIS, World Wide Web services	<ul style="list-style-type: none"> • Microsoft .NET extensibility • MSMQ HTTP support • WCF HTTP activation
Request Filtering	IIS, World Wide Web services, security	<ul style="list-style-type: none"> • Microsoft .NET extensibility • MSMQ HTTP support • WCF HTTP activation
XPS Viewer	N/A	Might affect other Windows features

UDI Reference

This reference provides further information about UDI and includes topics on:

- UDI concepts as described in [UDI Concepts](#)
- OSDResults as described in [OSDResults Reference](#)
- User Centric App Installer as described in [User-Centric App Installer Reference](#)
- UDI stages as described in [UDI Stage Reference](#)
- UDI tasks as described in [UDI Task Reference](#)
- UDI validators as described in [UDI Validator Reference](#)
- UDI Wizard Pages as described in [UDI Wizard Page Reference](#)

Each of these reference topics are discussed in subsequent sections.

UDI Concepts

This section contains concepts that help describe UDI, the UDI Wizard, and the UDI Wizard Designer.

Display Name

The display name is used to provide a user-friendly, descriptive name for a wizard page within the Page Library in the UDI Wizard Designer. The display name is displayed in blue text for each wizard page in the Page Library and on the **Flow** tab in the UDI Wizard Designer.

When you add a page to the Page Library, you must provide the display name. After the wizard page is added to the Page Library, you cannot change the display name.

Flow

The **Flow** tab displays the list of wizard pages within a UDI stage in the UDI Wizard Designer. You can use the **Flow** tab to perform the following tasks:

- Add a wizard page from the Page Library to a UDI stage by dragging the page from the Page Library to the UDI stage.
- Remove a wizard page from a UDI stage.
- Change the sequence of wizard pages within a UDI stage.

Page Library

The Page Library contains all the pages currently loaded in the UDI Wizard Designer. When loading a UDI Wizard configuration file, all of the wizard pages defined in the configuration file are displayed to the Page Library. The Page

Library shows the wizard pages in alphabetical order by page types. Each instance of a specific page type is listed under the page type.

For example, you may need two different **WelcomePage** wizard pages for different stages. The two **WelcomePage** wizard pages will be listed under the **WelcomePage** wizard page type in the Page Library in the UDI Wizard Designer.

In addition, each wizard page instance in the Page Library indicates how many times the wizard page is used in the stage flows. When you hover over a wizard page in the Page Library, a thumbnail of the wizard page is displayed along with the stages that include that page.

Page Name

The page name is used to uniquely identify a wizard page within the Page Library in the UDI Wizard Designer. The *page name* is the name a UDI stage references so that the UDI Wizard knows which wizard page to display within a specific UDI stage. When you add a page to the Page Library, you must provide the page name. After the wizard page is added to the Page Library, you cannot change the page name. In the UDI Wizard Designer, the page name is shown at the bottom of each wizard page in the Page Library in smaller, non-bold text.

Prestaged Media Deployments

Prestaged media support is an operating system deployment feature in Configuration Manager that allows an administrator to copy and apply prestage bootable media and an operating system image to a hard disk prior to the provisioning process. This work can reduce network traffic and the time needed for the provisioning process. Prestaged media can be deployed as part of the manufacturing process or at an enterprise staging center that is not connected to the Configuration Manager environment.

For more information about prestaged media deployments, see the following resources:

- [Planning for Media Operating System Deployments in Configuration Manager](#)
- [About Prestaged Media for Operating System Deployment](#)

Stage Group

Use a stage group to group one or more stages in the UDI Wizard Designer. UDI stage groups are loosely related to MDT deployment scenarios, but there is no one-to-one correlation between the two.

Stage

A *stage* is a subset of all the pages in the UDI Wizard configuration file that an MDT deployment scenario uses. When you start the UDI Wizard using the **UDI Wizard** task sequence step, the **/stage** parameter specifies the stage to run, which in turn specifies the set of pages to use. You can preview how wizard pages will appear in a stage by clicking **Preview** in the **Preview Wizard** group on

the Ribbon. You can use a UDI stage in more than one MDT deployment scenario, even though the UDI stage is defined only once in the UDI Wizard Designer. For example, the NewComputer stage can be used in the MDT New Computer and Replace Computer deployment scenarios.

Task

UDI tasks are software that is run on a wizard page to perform specific functions. In some instances, these tasks are used to verify that the target computer is ready for deployment. Other tasks can be used to perform deployment steps, such as copying configuration or result files.

Note The **Next** button on the wizard page where the tasks are run will be disabled if any of the tasks finish with warning or error completion status.

UDI includes several built-in tasks that allow you to perform most of the tasks necessary for deployment. For more information about the UDI built-in tasks, see [Built-in UDI Tasks](#).

The **Shell Execute** built-in UDI task allows you to run any software (scripts) that can be initiated from a command line, such as Visual Basic or Windows PowerShell scripts. This functionality allows you create tasks using familiar scripting languages. For more information, see [Shell Execute Task](#).

If your requirements go beyond scripting, you can write custom UDI tasks. *UDI tasks* are DLLs written in C++ and implement the **ITask** interface. You register the DLL with the UDI Wizard Designer task library by creating a UDI Wizard Designer configuration (.config) file and placing it in the *installation_folder\Bin\Config* folder (where *installation_folder* is the folder in which you installed MDT). For more information on developing custom UDI tasks, see the section, "Creating Custom UDI Tasks", in the *User-Driven Installation Developers Guide*.

UDI Task Sequence

You create a UDI task sequence using one of the following UDI-specific MDT task sequence templates, which run the UDI Wizard at the appropriate step in the task sequence:

- **User-Driven Installation Task Sequence.** This task sequence template is used for the New Computer, Refresh Computer, and Replace Computer MDT deployment scenarios.
- **User-Driven Installation Replace Task Sequence.** This task sequence template is the first step in a two-step process in the Replace Computer deployment scenario and is used to capture user state migration data. The second step in the two-step process is the User-Driven Installation Task Sequence task sequence template, which you use to deploy the target applications and operating system and restore the user state migration data saved during the first step of the process.

For more information about UDI task sequence templates, see the section, "Identify the UDI Task Sequence Templates in MDT", in the MDT document

Using the Microsoft Deployment Toolkit. For more information about these components, see the section, "Identify UDI Deployment Process Components", in the MDT document *Using the Microsoft Deployment Toolkit*, which is included with MDT.

UDI Wizard

The UDI Wizard provides the UI for collecting deployment settings that the UDI task sequences consume. The UDI Wizard is initiated as a part of a UDI task sequence and collects the necessary configuration information for customizing the deployment of the Windows client operating systems and applications. The wizard pages read their configuration settings from the UDI Wizard configuration file, which is customized using the UDI Wizard Designer.

The UDI Wizard is initiated by the **UDI Wizard** task sequence step in task sequences created using the UDI task sequence templates. The **UDI Wizard** task sequence step runs the UDIWizard.wsf script, which in turn initiates the UDI Wizard (OSDSetupWizard.exe). Table 9 lists the UDI Wizard command-line parameters and provides a brief description of each.

Table 9. UDI Wizard Command-Line Parameters

Parameter	Description
/preview	Allows you to preview the current configuration of the wizard by enabling the Next button, which allows you to move from page to page without requiring valid input.
/xml	Specifies the name of the UDI Wizard configuration file. The UDIWizard.wsf script automatically sets this parameter to the OSDSetupWizard.xml file, which is stored in the folder in which the task sequence stores log files. This parameter defaults to the config.xml file. The syntax for this parameter is as follows (where <full_path> is the fully qualified path to the .xml file, including the file name and extension): <code>/xml : <full_path></code>
/stage	Specifies the name of the UDI stage to run. The UDIWizard.wsf script automatically sets this parameter to the appropriate stage, as described in UDI Stage Reference . This parameter defaults to the first stage in the UDI Wizard configuration file. The syntax for this parameter is as follows (where <stage_name> is the name of the stage to be run): <code>/stage: <stage_name></code> Note The value for <stage_name> is case sensitive.
/locale	Specifies the language to use in the UDI Wizard in the form of a locale identifier (LCID), which is represented by a numeric value. For a list of the available LCIDs, see Locale IDs Assigned

Parameter	Description
	<p>by Microsoft.</p> <p>You would use this list to identify the language you want to use, and then provide the corresponding LCID.</p> <p>The syntax for this parameter is as follows (where <local_e_id> is the numeric value of the LCID to be used):</p> <p>/local_e:<local_e_id></p>

UDI Wizard Application Configuration File

The **ApplicationPage** wizard page configures the UDI Wizard application configuration file, which maintains the list of software to be installed. This file contains an entry for each Configuration Manager application or program and package that was added using the UDI Wizard Designer.

This file has the same name as the UDI Wizard configuration file but with a .app extension. For example, if the UDI Wizard configuration file is named *Config.xml*, then the corresponding UDI Wizard application configuration file would be *Config.xml.app*. This file is the companion to the UDI Wizard configuration file.

UDI Wizard Configuration File

The UDI Wizard reads the UDI Wizard configuration file to determine the wizard pages to be displayed, the sequence of the wizard pages, any default for controls on the wizard pages, and whether the controls are enabled or disabled. This file contains all the configuration settings that are displayed in the UDI Wizard and are configured using the UDI Wizard Designer.

A separate configuration file—the UDI Wizard application configuration file—is used to configure applications to be installed on the target computer.

UDI Wizard Designer

The UDI Wizard Designer is the primary tool for customizing wizard pages for the different deployment scenarios that UDI supports. Changes made in the UDI Wizard Designer are saved in the UDI Wizard configuration file and ultimately reflected in the user experience in the UDI Wizard. The user performing the deployment will see only the wizard pages in the UDI Wizard that you have selected and configured using the UDI Wizard Designer.

Although the UDI Wizard would run with the default UDI Wizard configuration file, the wizard pages would not be configured correctly. It is recommended that you use the UDI Wizard Designer to configure the UDI Wizard user experience.

Note To run the UDI Wizard Designer, you must have the appropriate rights in Configuration Manager to access objects such as packages, applications, or images.

Validator

You use UDI validators to help ensure that the correct information is entered into text fields on wizard pages in the UDI Wizard. UDI includes several built-in validators that help you perform typical validations of fields used for entering text, such as preventing users from entering invalid characters and ensuring that the field is not empty. When a validator detects an invalid entry in a text box, a message is displayed on the wizard page, and the **Next** button is disabled until all invalid entries are resolved.

UDI includes built-in validators that allow you to perform most of the validation necessary for deployment. For more information about the UDI built-in validators, see [Built-in UDI Validators](#).

If your requirements go beyond the built-in UDI validators, you can write custom UDI validators. *UDI validators* are DLLs written in C++ that implement the **IValidator** interface. Register the DLL with the UDI Wizard Designer validator library by creating a UDI Wizard Designer configuration (.config) file and placing it in the *installation_folder\Bin\Config* folder (where *installation_folder* is the folder in which you installed MDT). For more information on developing custom UDI tasks, see the section, "Creating Custom UDI Validators", in the MDT document *User-Driven Installation Developers Guide*.

Wizard Page

You use a wizard page to collect configuration information in the UDI Wizard. Configure UDI wizard pages using the UDI Wizard Designer. The configuration settings are stored in the UDI Wizard configuration file and are read by the wizard page when the page is initialized in the UDI Wizard.

Wizard pages are stored in the wizard Page Library, and they can be used in one or more UDI stages. This design allows you to configure a wizard page that is shared between stages once for all stages, dramatically reducing the amount of effort required and the complexity of updating wizard page configuration.

UDI includes built-in wizard pages and wizard page editors that are typically sufficient for most deployments. For more information about the built-in wizard pages, see [Built-in UDI Wizard Pages](#).

If your requirements go beyond the built-in UDI wizard pages and corresponding wizard page editors, you can write custom UDI wizard pages and wizard page editors. UDI wizard pages are implemented as DLLs that the UDI Wizard reads. Wizard page editors are created using C++ in Visual Studio.

For more information on developing custom UDI wizard pages, see the section, "Creating Custom UDI Wizard Pages", in the MDT document *User-Driven Installation Developers Guide*.

Wizard Page Editor

You use a wizard page editor to configure a wizard page in the UDI Wizard Designer. A wizard page editor updates the wizard page configuration settings in

the UDI Wizard configuration file; UDI includes a built-in wizard page editor for each built-in wizard page. For more information about the built-in wizard pages and wizard page editors, see [Built-in UDI Wizard Pages](#).

If your requirements go beyond the built-in UDI wizard pages and corresponding wizard page editors, you can write custom UDI wizard pages and wizard page editors. UDI wizard page editors are implemented as DLLs that the UDI Wizard Designer reads. Create wizard page editors using:

- [Windows Presentation Foundation](#) version 4.0
- [Microsoft Prism](#) version 4.0
- [Microsoft Unity Application Block](#) (Unity) version 2.1

For more information on developing custom UDI wizard page editors, see the section, "Creating Custom Wizard Page Editors", in the MDT document *User-Driven Installation Developers Guide*.

OSDResults Reference

OSDResults is a part of UDI that displays the results of a deployment performed using UDI. **OSDResults** displays the **Deployment Complete** dialog box.

OSDResults is displayed prior to Windows logon the first time the target computer is started. The user can use **OSDResults** and the information in the **Deployment Complete** dialog box to determine the completion status of the deployment process and the configuration of the computer prior to logging on for the first time. In addition, the information in **OSDResults** can be used for troubleshooting any problems encountered during the deployment process.

You can configure some of the user interface elements for **OSDResults** using the OSDResults.exe.config file, which resides in Tools\OSDResults in the MDT files Configuration Manager package. Table 10 lists the configuration settings in the OSDResults.exe.config file.

Table 10. Configuration Settings in the OSDResults.exe.config File

Setting	Description
headerImagePath	This setting allows you to specify the fully qualified or relative path to a .bmp file that is displayed in the header of the OSDResults dialog box. The default value for this setting is as follows: <code>images\UDI_Wizard_Banner.bmp</code>
backgroundWallpaper	This setting allows you to specify the fully qualified or relative path to a .jpg file that is displayed as the wallpaper in the OSDResults dialog box. The default value for this setting is as follows: <code>images\Wallpaper.jpg</code>
welcomeText	This setting allows you to specify the text that

Setting	Description
	welcomes the user and provides information about the deployment process. It is displayed in the OSDResults dialog box.
completedText	This setting allows you to specify the text that indicates whether the deployment process is complete. It is displayed in the OSDResults dialog box.
timeoutMinutes	This setting allows you to specify the length of time the OSDResults dialog box is displayed prior to automatically displaying the Windows logon screen. The value for this setting is specified in minutes. The default value for this setting is zero (0), which indicates that the OSDResults dialog box will be displayed indefinitely until manually closed.

The following is the high-level process for how the **OSDResults** feature works in UDI:

1. A task sequence runs on the target computer.

The task sequence is based on one of the following UDI task sequence templates:

- **User Driven Installation Task Sequence.** This task sequence template is used for the MDT New Computer, Refresh Computer, and Replace Computer MDT deployment scenarios.
- **User-Driven Installation Replace Task Sequence.** This task sequence template is the first step in a two-step process in the MDT Replace Computer deployment scenario and is used to capture user state migration data. The second step in the two step process is the MDT New Computer deployment scenario using the **User Driven Installation Task Sequence** task sequence template, which is used to deploy the target applications and operating system and restore the user state migration data saved during the first step of the process

For more information about the:

- UDI task sequence templates, see the section, "Identify the UDI Task Sequence Templates in MDT", in the MDT document *Using the Microsoft Deployment Toolkit*
 - Relationship between MDT deployment scenarios and UDI stages, see [UDI Stage Reference](#)
2. During the task sequence, the configuration settings provided by task sequence variables and from user input in the UDI Wizard are saved in the %DEPLOYROOT%\Tools\OSDResults folder on the target computer (where %DEPLOYROOT% is the root of the folder in which the MDT files are locally cached on the target computer).

3. In the **OSD Results and Branding** group in the task sequence, the following task sequence steps are run that affect **OSDResults**:
 - **Cache OSD Results.** This task sequence step copies the contents of the %DEPLOYROOT%Tools\OSDResults folder to the %WINDIR%\UDI folder on the target computer. This ensures that the contents of the OSDResults folder will be persisted after the task sequence finishes.
 - **Run OSD Results.** This tasks sequence step configures the target computer to run **OSDResults** the first time the computer starts.
4. The target computer starts for the first time, and OSDResults.exe is run prior to the Windows logon screen.

The **Welcome** tab in the **Deployment Complete** dialog box is displayed. The **Welcome** tab provides helpful information about the deployment and contact information in the event that issues with the deployment are discovered.

Review the information on the **Deployment Summary** and **Applications Installed** tabs to verify that the operating system and applications were installed correctly. When you have completed reviewing these tables, click **Start Windows** to log on to Windows 7 for the first time.

Note Configuration Manager applications are not displayed on the **Applications Installed** tab. The Configuration Manager applications are detected after the user logs on to the target computer the first time.

5. The Windows logon screen is displayed, and the logon process continues normally.

ApplInstall.exe is run the first time a user logs on to the target computer. For more information on this process, see [User-Centric App Installer Reference](#).

User-Centric App Installer Reference

The User-Centric App Installer feature in UDI is used to report any applications installed during the UDI deployment process to the Application Catalog feature in Configuration Manager. The User-Centric App Installer feature provides the link between the applications selected on the **ApplicatonPage** wizard page in the UDI Wizard and any optional Configuration Manager applications advertised to the users.

For more information on the Application Catalog feature in Configuration Manager, see [Application Management in Configuration Manager](#).

The following is the high-level process for how the App Install feature works in UDI:

1. Configuration Manager applications are created in Configuration Manager.

For more information about creating and managing Configuration Manager applications, see the following resources:

- [How to Create Applications in Configuration Manager](#)

- [Operations and Maintenance for Application Management in Configuration Manager](#)
2. The Configuration Manager user collections are created, and users are added to the collection.

For more information about creating and managing user collections and adding users to collections, see the following resources:

 - [Collections in Configuration Manager](#)
 - [How to Create Collections in Configuration Manager](#)
3. The Configuration Manager applications are deployed to the user collections.

For more information about how to deploy the applications to user collections, see [How to Deploy Applications in Configuration Manager](#).
4. The Configuration Manager applications are made available on the **ApplicatonPage** wizard page using the UDI Wizard Designer.

For more information about how to make Configuration Manager applications available on the **ApplicatonPage** wizard page, see the section, "Step 5-11: Customize the UDI Wizard Configuration File for the Target Computer", in the MDT document *Quick Start Guide for User-Driven Installation*.
5. UDA is configured using one of the following methods:
 - In the Configuration Manger console (For more information about configuring UDA in the Configuration Manager console, see [How to Manage User Device Affinity in Configuration Manager](#).)
 - On the **UDAPage** wizard page in the UDI Wizard (For more information about the **UDAPage** wizard page, see [UDAPage](#).)

After UDA is configured, the specified user account will be the primary user for the target computer.

Note UDA can only be configured by UDI in the New Computer deployment scenario. It cannot be configured in the Refresh Computer or Replace Computer deployment scenarios.
6. The task sequence is run, and the user selects the Configuration Manager applications on the **ApplicatonPage** wizard page in the UDI Wizard.

The UDI Wizard is run in the **UDI Wizard** task sequence step in the **Preinstall** group of the task sequence. When the user selects Configuration Manager applications on the **ApplicatonPage** wizard page, the wizard page creates a separate task sequence variable for each application selected.

For more information on selecting the Configuration Manager applications on the **ApplicatonPage** wizard page in the UDI Wizard, see the section, "Step 6-4: Start the Target Computer with the Task Sequence Bootable Media", in the MDT document *Quick Start Guide for User-Driven Installation*.
7. The task sequence installs the Configuration Manager applications that were selected in the previous step.

The Configuration Manager applications are installed using the following task sequence steps in the **Install Applications** group in the task sequence:

- **Convert list to two digits**
 - **Install Application**
8. The task sequence performs the following tasks in the **OSD Results and Branding** group prior to starting the target operating system for the first time:
- Copies the information used for OSDResults.exe to the %WINDIR%\UDI folder on the target computer in the **Cache OSD Results** task sequence step
 - Records the task sequence variables created in step 6 for the Configuration Manager applications in the registry on the target computer in the **Branding to Reg** and **Branding to Reg x64** task sequence steps
- The tasks sequence variables are saved in the following location in the registry:
- HKEY_LOCAL_MACHINE\Software\Microsoft\MPSD\OSD**
- Configures the target operating system to automatically run OSDResults.exe when the computer starts prior to the Windows logon screen in the **Run OSD Results** task sequence step
 - Configures the target operating system to automatically run AppInstall.exe when a user logs on to the computer for the first time in the **Run OSD Results** task sequence step
 - Configures a task on the target operating system to remove the %WINDIR%\UDI folder one month from the date of the deployment
9. The target computer is started, and OSDResults.exe is run.

For more information about OSDResults.exe, see [OSDResults Reference](#).

10. A user logs on to the target computer, and AppInstall.exe starts automatically.
11. AppInstall checks whether the currently logged-on user is a primary user who was configured in UDA.

A *primary user* is a user who uses the device on a regular basis and is considered the owner, or one of the owners, of the device.

If the currently logged-on user is:

- Not a primary user, then AppInstall.exe stops
 - A primary user, then AppInstall.exe reads the registry entries saved in step 8 to determine which applications were installed
12. AppInstaller connects to Configuration Manager and reads the Application Catalog using the following steps:
- a. AppInstall will wait 5 minutes after it starts to allow the Configuration Manager policies to be available.

- b. After 5 minutes, AppInstall attempts to connect to the Application Catalog.
- c. If AppInstall is unable to connect, then it will wait for a period of time before attempting to connect again.
- d. AppInstall attempts to connect up to five times before exiting.

You can configure the connection time-out delay and the number of retries for AppInstall using the AppInstall.exe.config file, which resides in the Tools\OSDResults folder in the MDT files Configuration Manager package. Table 11 lists the configuration settings in the AppInstall.exe.config file.

Table 11. Configuration Settings in the AppInstall.exe.config File

Setting	Description
timeoutMinutes	This setting allows you to specify the length of time for AppInstall to wait for a response from the Configuration Manager Application Catalog before timing out. The value is specified in minutes. The default value for this setting is 5 .
delayTimer	This setting allows you to specify the length of time for AppInstall to wait prior to attempting the connection to the Configuration Manager Application Catalog. The value is specified in minutes. The default value for this setting is 5 .

13. AppInstall compares the list of applications discovered in the registry with the list of applications available from the Configuration Manager Application Catalog for the user currently logged on.

If the application discovered in the registry:

- Is available in the Application Catalog, then AppInstall.exe maps the applications and identifies the applications as existing both in the registry and in the Application Catalog. These applications will be used in the following step.
- Is not available in the Application Catalog, then AppInstall.exe does not create a mapping. These applications will not be used in the following step.

14. AppInstall uses Configuration Manager APIs to initiate the installation of the mapped applications.

The applications used in this step were mapped in the previous step. That is to say, they were both listed in the registry and found in the Application Catalog.

15. As a part of the installation process, Configuration Manager detects whether the application is already installed.

Because the application has already been installed, Configuration Manager records that the application has been successfully deployed to that user, and the application will be listed in Software Center for that user. Configuration Manager begins management and monitoring of the application for that user.

16. After 1 month, the task created on the target computer in step 8 runs and removes the %WINDIR%\UDI folder.

The folder is retained for 1 month so that the primary users have an opportunity to log on and run ApplInstall.exe.

UDI Stage Reference

The MDT deployment scenarios use one or more UDI [stage](#). Each UDI stage used in the MDT deployment scenarios is discussed in a subsequent section in the context of the MDT deployment scenario. In some MDT deployment scenarios, only one stage is used. In other MDT deployment scenarios, multiple stages are used within the scenario. For more information on the MDT deployment scenarios, see the section, "Identifying Deployment Scenarios", in the MDT document *Using the Microsoft Deployment Toolkit*.

Table 12 lists the MDT deployment scenarios and provides a brief description of each, how each scenario is selected, and which UDI stages are used in each deployment scenario. MDT automatically determines which MDT deployment scenario to use based on the MDT task sequence template you use to create your task sequence and on how the task sequence is initiated.

Each UDI stage used in the MDT deployment scenarios is discussed in a subsequent section in the context of the MDT deployment scenario. In some MDT deployment scenarios, only one stage is used. In other MDT deployment scenarios, multiple stages are used within the scenario. For more information on the MDT deployment scenarios, see the section, "Identifying Deployment Scenarios", in the MDT document *Using the Microsoft Deployment Toolkit*.

Table 12. MDT Deployment Scenarios and UDI Stages

Scenario	Description
New Computer	<p>MDT for UDI automatically selects this scenario when you:</p> <ul style="list-style-type: none"> • Create the advertised task sequence using the User-Driven Installation Task Sequence task sequence template • Start the task sequence in Windows PE using PXE boot, task sequence boot media, or prestaged media for the NEWCOMPUTER.Prestaged stage <p>This scenario can be used with traditional deployments or with prestaged media deployments as supported in Configuration Manager. Run the UDI Wizard with the following UDI stages to support each type of deployment:</p>

Scenario	Description
	<ul style="list-style-type: none"> • NEWCOMPUTER stage. The UDI Wizard is run with this stage in the User-Driven Installation Task Sequence task sequence when the operating system image is stored on distribution points. For more information, see NEWCOMPUTER Stage. • NEWCOMPUTER.Prestage stage. The UDI Wizard is run with this stage in the User-Driven Installation Task Sequence task sequence when the operating system image is stored on a local disk on the target computer (prestaged). For more information, see NEWCOMPUTER.Prestaged Stage.
Refresh Computer	<p>MDT for UDI automatically selects this scenario when you:</p> <ul style="list-style-type: none"> • Create the advertised task sequence using the User-Driven Installation Task Sequence task sequence template • Start the task sequence in the existing Windows operating system on the target computer (not in Windows PE) <p>The UDI Wizard is run with the REFRESH stage to support this deployment scenario. For more information, see REFRESH Stage.</p>
Replace Computer	<p>This scenario includes an existing computer and a replacement computer. A separate task sequence is created and run on each computer as described in the following process:</p> <ul style="list-style-type: none"> • On the existing computer. MDT for UDI automatically selects this portion of the scenario when you: <ul style="list-style-type: none"> • Create the advertised task sequence using the User-Driven Installation Replace Task Sequence task sequence template • Start the task sequence in the existing Windows operating system on the target computer (not in Windows PE) <p>The UDI Wizard is run with the following UDI stages to support this deployment scenario:</p> <ul style="list-style-type: none"> • REPLACE stage. This stage is run in the existing Windows operating system and captures configuration information from within Windows. • REPLACE.WinPE stage. This stage is run in Windows PE and completes the capturing of configuration information from the existing computer—for example, running USMT and capturing the user state migration data. <p>The user state is captured to a network shared folder or to</p>

Scenario	Description
	<p>a local USB drive.</p> <p>For more information on the REPLACE and REPLACE.WinPE stages, see REPLACE and REPLACE.WinPE Stages.</p> <ul style="list-style-type: none">• On the replacement computer. This portion of the scenario is identical to the New Computer scenario, except that the user state captured in the previous step is restored. MDT for UDI automatically selects this portion of the scenario when you:<ul style="list-style-type: none">• Create the advertised task sequence using the User-Driven Installation Task Sequence task sequence template• Start the task sequence in Windows PE using PXE boot, task sequence boot media, or prestaged media for the NEWCOMPUTER.Prestaged stage. <p>This portion of the scenario can be used with traditional deployments or with prestaged media deployments as supported in Configuration Manager. As a part of this portion of the scenario, the user state migration data is restored. The UDI Wizard is run with the following UDI stages to support each type of deployment:</p> <ul style="list-style-type: none">• NEWCOMPUTER stage. The UDI Wizard is run with this stage in the User-Driven Installation Task Sequence task sequence when the operating system image is stored on distribution points. For more information, see NEWCOMPUTER Stage.• NEWCOMPUTER.Prestage stage. The UDI Wizard is run with this stage in the User-Driven Installation Task Sequence task sequence when the operating system image is stored on a local disk on the target computer (prestaged). For more information, see NEWCOMPUTER.Prestaged Stage.

NEWCOMPUTER Stage

Figure 1 illustrates the use of the NEWCOMPUTER stage in a task sequence created using the User-Driven Installation Task Sequence task sequence template. The primary difference between the task sequences calling the NEWCOMPUTER stage and the NEWCOMPUTER.Prestaged stage is that the task sequence calling the NEWCOMPUTER.Prestaged stage does not run the **Apply Operating System Image** task sequence step, because the operating system image is already located on the target computer.

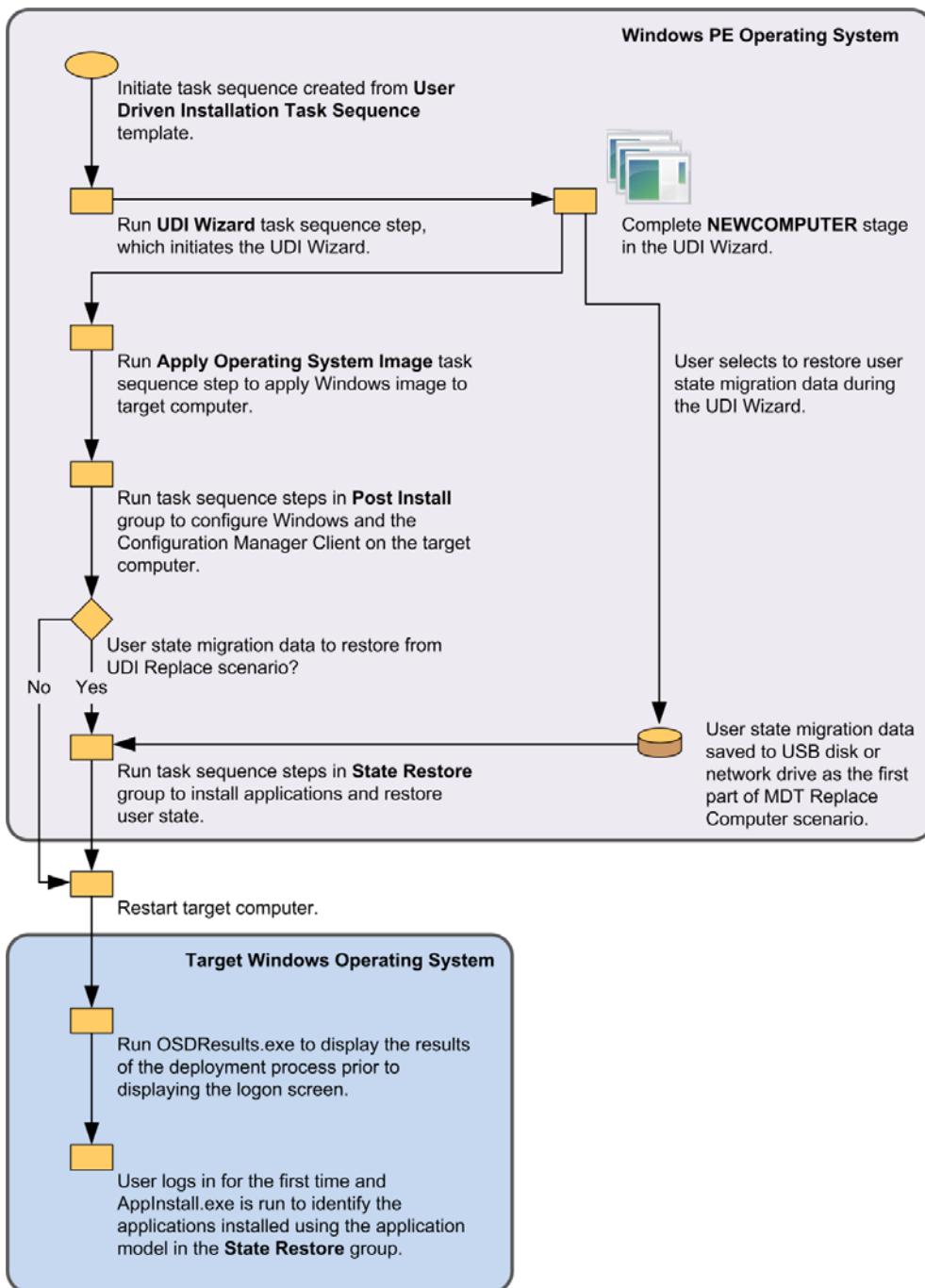


Figure 1. Process flow for the NEWCOMPUTER stage

NEWCOMPUTER.Prestaged Stage

Figure 2 illustrates the high-level process flow for the NEWCOMPUTER.Prestaged stage in a task sequence created using the User-Driven Installation Task Sequence task sequence template. The primary difference between the task sequences calling the NEWCOMPUTER stage and the NEWCOMPUTER.Prestaged stage is that the task sequence calling the

NEWCOMPUTER.Prestaged stage does not run the **Apply Operating System Image** task sequence step, because the operating system image is already located on the target computer.

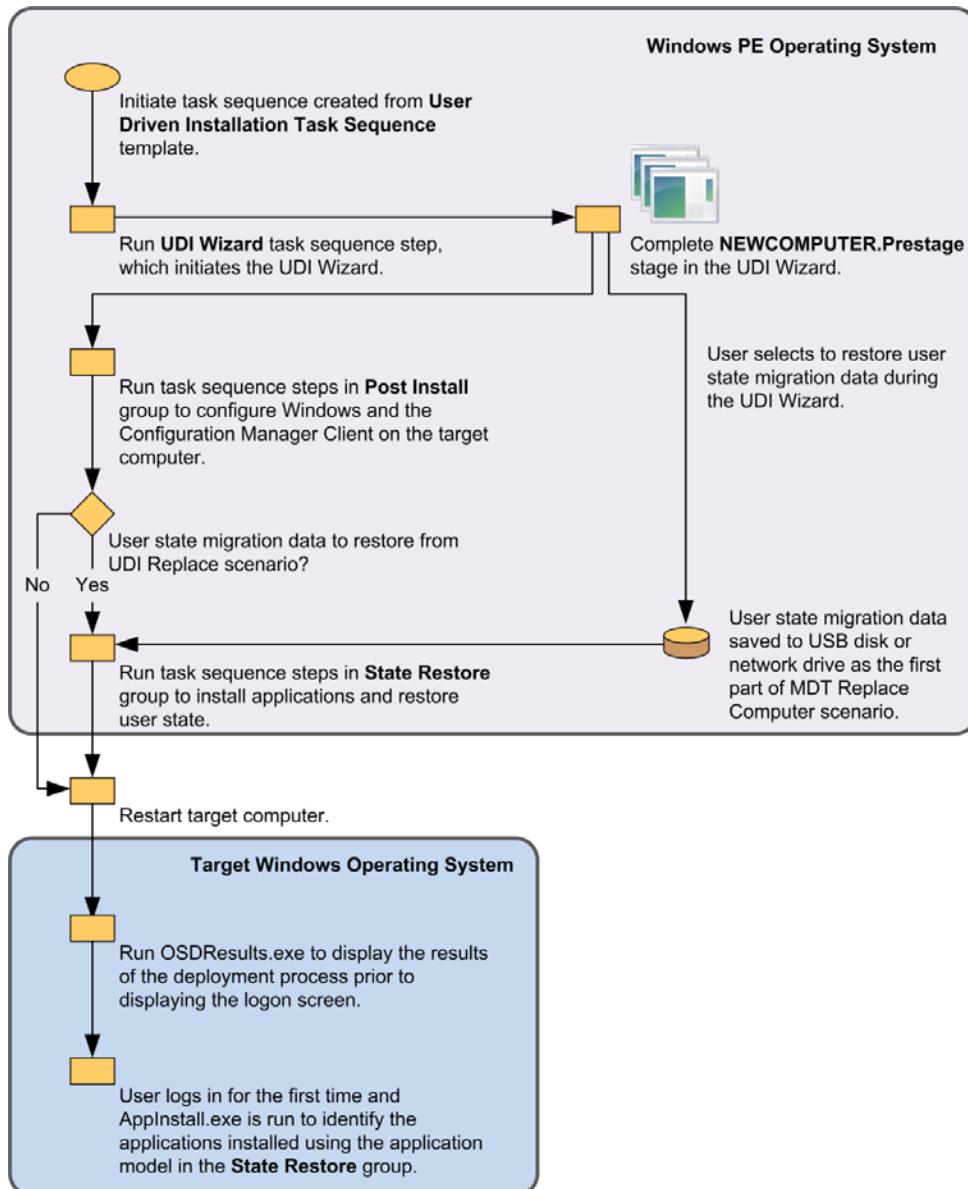


Figure 2. Process flow for the NEWCOMPUTER.Prestaged stage

REFRESH Stage

Figure 3 illustrates the high-level process flow for the REFRESH stage in a task sequence created using the User-Driven Installation Task Sequence task sequence template.

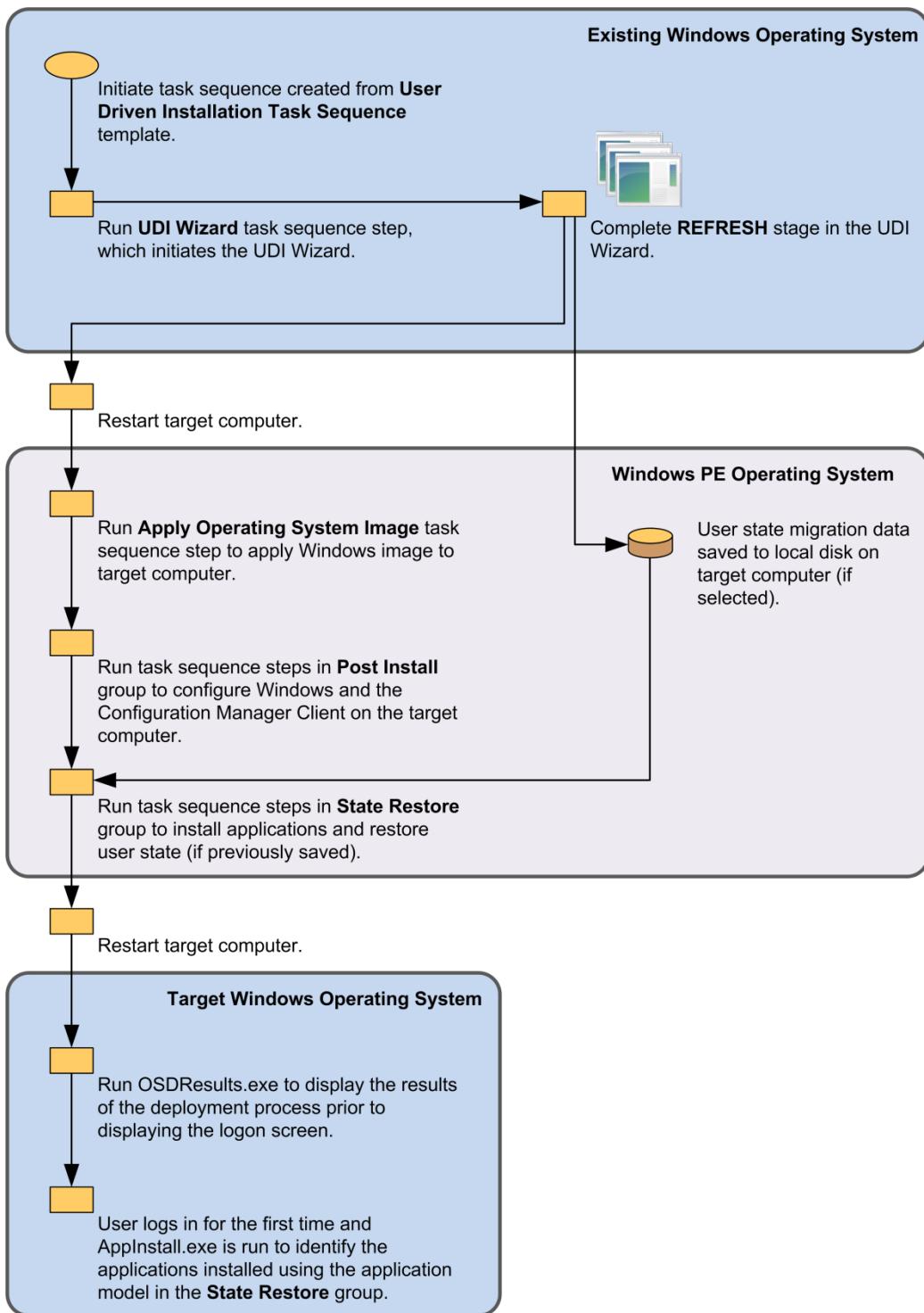


Figure 3. Process flow for the REFRESH stage

REPLACE and REPLACE.WinPE Stages

Figure 4 illustrates the high-level process flow for the REPLACE and REPLACE.WinPE stages in a task sequence created using the User-Driven Installation Replace Task Sequence task sequence template.

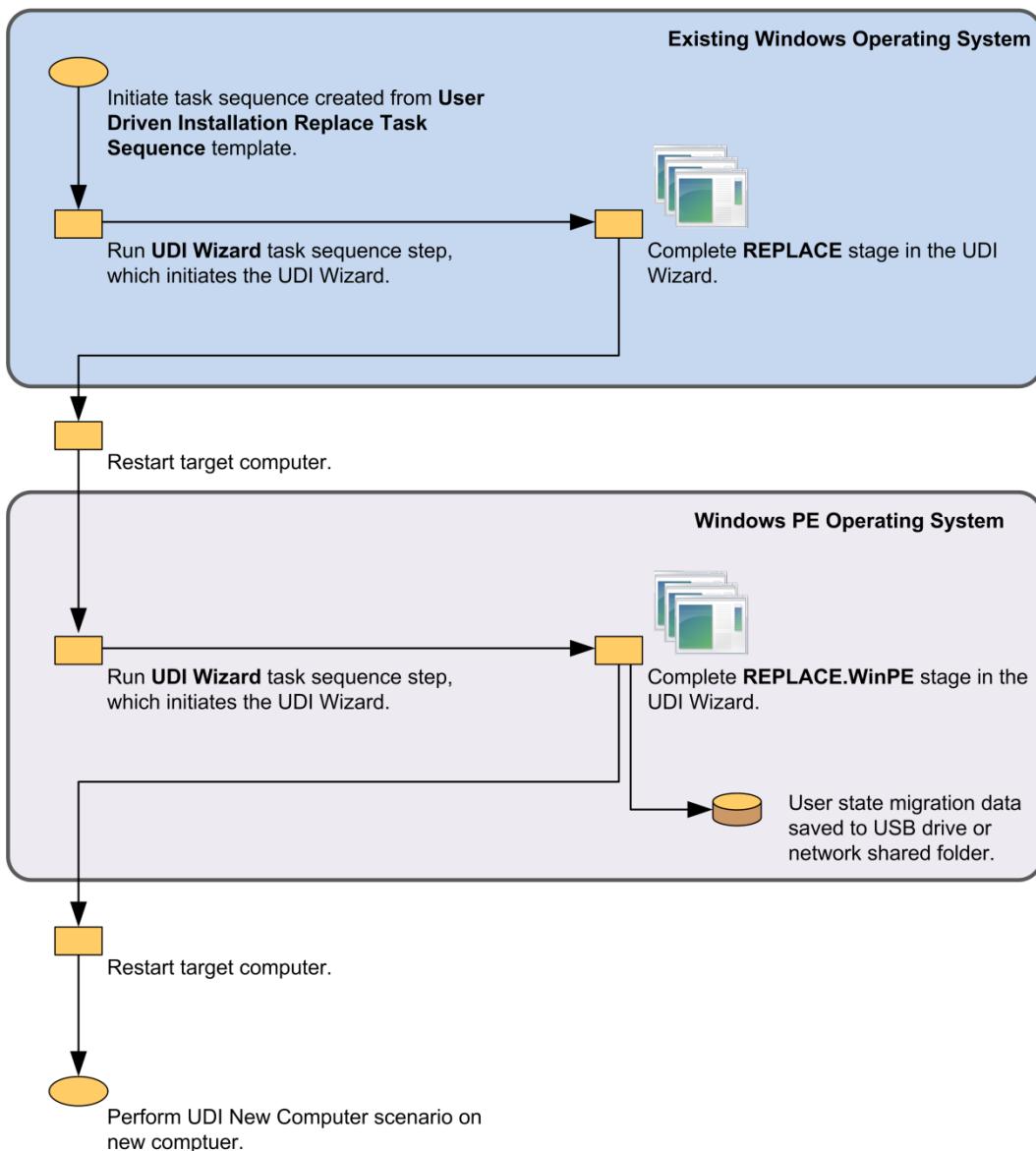


Figure 4. Process flow for the REPLACE and REPLACE.WinPE stages

UDI Task Reference

UDI tasks are software that is run on a wizard page that perform specific functions. In some instances, these tasks are used to verify that the target computer is ready for deployment. Other tasks can be used to perform deployment steps, such as copying configuration or result files.

Note The **Next** button on the wizard page where the tasks are run will be disabled if any of the tasks finish with warning or error completion status.

This reference includes:

- An overview of UDI tasks, as described in [UDI Task Overview](#)
- A description of the configuration settings for UDI tasks, as described in [UDI Task Configuration Settings](#)
- A description of the built-in UDI validators that are provided with MDT, as described in [Built-in UDI Tasks](#)

UDI Task Overview

UDI tasks allow you to run software on the target computer that helps with the deployment process. UDI includes several built-in tasks that help you perform common tasks, such as ensuring that the target computer is not running on a battery and is connected to a wired network connection.

In addition to the built-in UDI tasks, you can create custom UDI tasks using the UDI software development kit (SDK). For more information about creating custom UDI tasks using the UDI SDK, see *User-Driven Installation Developers Guide*.

UDI Task Configuration Settings

You manage tasks using the UDI Wizard Designer. You can add tasks, remove tasks, and edit the configuration of a task in the UDI Wizard Designer. The configuration settings for a task are stored in the UDI Wizard configuration file and are read by the UDI Wizard when the wizard page that contains the task is displayed.

UTI tasks have some configuration settings that are common to all UDI tasks, as listed in Table 13. For the configuration settings that are specific to each UDI task, see the corresponding section in [Built-in UDI Tasks](#).

Table 13. Configuration Settings Common to All UDI Tasks

Task	Description
Bitmap Filename	This parameter specifies the graphic used to indicate the task type.
Display Name	This specifies the name of the task, which is displayed on the wizard page when the task is run.
Exit Code Values	This specifies a list of possible return codes for the task. An item exists in the list for each possible return code.
Error Code Values	This specifies a list of possible unexpected exceptions that may be encountered (thrown) by the task. An item exists in the list for each possible exception.

Built-in UDI Tasks

Table 14 lists the built-in UDI tasks. Each built-in UDI task is discussed in a subsequent section.

Table 14. Built-in UDI Tasks

Task	Description
AC Power Check	This UDI task is used to identify whether the target computer is connected to AC power, not solely on battery.
Application Discovery	This UDI task is used to discover applications that are installed on the target computer.
CheckSMSFolderOnUSB	This UDI task is used to determine whether the _SMSTaskSequence folder is located on a USB drive on the target computer.
Copy Files Task	This UDI task is used to copy files while the UDI Wizard is running on the target computer.
Shell Execute Task	This UDI task is used to run software that can be initiated from a command line.
Wired Network Check	This UDI task is used to identify whether the target computer is connected to a wired network, not connected using a wireless network connection.

AC Power Check

Use this UDI task to identify whether the target computer is connected to AC power. This task uses only those parameters common to all UDI tasks. For more information about these parameters, see [UDI Task Configuration Settings](#).

Table 15 lists the error and exit codes that the **AC Power Check** task generates.

Table 15. Error and Exit Codes for the AC Power Check Task

Exit or error code	Value	Status
Exit	0	Success , which indicates that the target computer is plugged into AC power
Exit	*	Error , which indicates that the target computer is not plugged into AC power

Application Discovery

Use this UDI task to discover applications that are installed on the target computer.

Table 16 lists the parameters that the **Application Discovery** task uses.

Table 16. Parameters Used by the Application Discovery Task

Task	Description
Readcfg	This parameter specifies the fully qualified or relative path to the location of the .app file that has a list of applications for the task to discover. The .app file contains the list of available software items from which the user can select. The Application Discovery task reads the .app file and determines whether any of these software items is installed. If a software item is installed, the item is added to the file specified in the Writecfg parameter. Ensure that this parameter uses the same location and file name as the ApplicationPage wizard page.
Writecfg	This parameter specifies the fully qualified or relative path to the location of the .xml file that contains a list of the applications discovered by the task.
Log	This parameter specifies the fully qualified or relative path to the location of the log file generated by this task. The file name of the log file is AppDiscovery.log.

In addition to the parameters in Table 16, this task uses the parameters common to all UDI tasks. For more information about these common parameters, see [UDI Task Configuration Settings](#).

Table 17 lists the error and exit codes that the **Application Discovery** task generates.

Table 17. Error and Exit Codes for the Application Discovery Task

Exit or error code	Value	Status and description
Exit	0	Success , which indicates that the task successfully scanned for applications
Exit	*	Warning , which indicates that the application discovery engine could not be run for some unknown reason
Exit	1	Warning , which indicates that the application discovery engine encountered one or more warnings
Exit	16777216	Warning , which indicates that critical problems were encountered while initializing the application discovery engine
Exit	33554432	Warning , which indicates that critical problems were encountered while processing the application master list

CheckSMSFolderOnUSB

Use this UDI task to identify whether the _SMSTaskSequence folder is located on a USB drive on the target computer. By default, the Configuration Manager task sequencer places the _SMSTaskSequence folder on the drive with the most available free disk space. This can cause problems later in the deployment process if the USB drive is removed.

This task checks to see whether the folder is located on a USB drive and prevents the deployment from proceeding if it is. This task uses only those parameters common to all UDI tasks. For more information about these parameters, see [UDI Task Configuration Settings](#).

If the _SMSTaskSequence folder is located on a USB drive, this task fails and prevents the deployment from continuing. To resolve this issue and perform the deployment, complete the following steps:

1. Disconnect the USB drive from the target computer before starting the task sequence.
2. Start the task sequence.
3. Wait until the UDI Wizard starts.
4. Connect the USB drive.
5. Complete the UDI Wizard.

Table 18 lists the error and exit codes that the **CheckSMSFolderOnUSB** task generates.

Table 18. Error and Exit Codes for the CheckSMSFolderOnUSB Task

Exit or error code	Value	Status
Exit	0	Success , which indicates that the _SMSTaskSequence folder is not located on a USB drive and the deployment can continue.
Exit	*	Error , which indicates that the _SMSTaskSequence folder is located on a USB drive and the deployment cannot continue.

Copy Files Task

Use this UDI task to copy files while the UDI Wizard is running on the target computer.

Table 19 lists the parameters that the **Copy Files** task uses.

Table 19. Parameters Used by the Copy Files Task

Task	Description
Source	This parameter specifies the fully qualified or relative path to the source file, which can contain wildcards to

Task	Description
	copy multiple files using a single task.
Destination	This parameter specifies the fully qualified or relative path to the destination file without a file name.

In addition to the parameters in Table 19, this task uses parameters common to all UDI tasks. For more information about these parameters, see [UDI Task Configuration Settings](#).

Table 20 lists the error and exit codes that the **Copy Files** task generates.

Table 20. Error and Exit Codes for the Copy Files Task

Exit or error code	Value	Status and description
Exit	0	Success , which indicates that the copy process succeed
Exit	*	Error , which indicates that the copy process failed
Error	-1	Error , which indicates that the copy process failed

Shell Execute Task

Use this UDI task to run software that can be initiated from a command line.

Table 21 lists the parameters that the **Shell Execute** task uses.

Table 21. Parameters Used by the Shell Execute Task

Task	Description
Filename	This parameter specifies the fully qualified or relative path to the command for the task to run.
Parameters	This parameter specifies the command-line parameters that are to be provided when running the command.

In addition to the parameters in Table 21, this task uses parameters common to all UDI tasks. For more information about these parameters, see [UDI Task Configuration Settings](#).

You can also run custom Visual Basic scripts designed to run in cscript.exe using the **Shell Execute** task. To run Visual Basic scripts, perform the following steps:

1. Type the following text in the **Filename** parameter:

%windir%\system32\cscript.exe

2. Type name of the Visual Basic script file (.vbs file) in the **Parameters** parameter, including any command-line parameters for the script.

For example, to run a Visual Basic script named *SelfTest.vbs* with a parameter value of **Debug**, type the following (where *script_path* is the fully qualified path to the *SelfTest.vbs* file):

```
<script_path>\Sel fTest. vbs Debug
```

Table 22 lists the common error and exit codes that the **Shell Execute** task generates.

Note Each specific task based on the **Shell Execute** task has a unique set of error and exit codes. Please check the return codes for the software you are running using this task.

Table 22. Common Error and Exit Codes for the Shell Execute Task

Exit or error code	Value	Status and description
Exit	0	Success , which indicates that the task finished successfully
Exit	*	Error , which indicates that the task failed

Wired Network Check

Use this UDI task to determine whether the target computer is connected to a wired network, not using a wireless network connection. This task only uses parameters common to all UDI tasks. For more information about these parameters, see [UDI Task Configuration Settings](#).

Table 23 lists the common error and exit codes that the **Wired Network Check** task generates.

Table 23. Error and Exit Codes for the Wired Network Check Task

Exit or error code	Value	Status and description
Exit	0	Success , which indicates that the target computer is connected to a wired network
Exit	*	Error , which indicates that the target computer is not connected to a wired network

UDI Validator Reference

UDI validators are used to validate values entered in text fields on wizard pages. When a UDI validator detects an invalid entry, a message is displayed for the first error encountered at the bottom of the wizard page. The next validation error message, if any, is displayed after you resolve the first validation error. This process continues until all validation errors are resolved. The **Next** button is disabled until all validation errors on the wizard page are resolved.

This reference includes:

- An overview of UDI validators, as described in [UDI Validator Overview](#)
- A description of the built-in UDI validators provided with MDT, as described in [Built-in UDI Validators](#)

UDI Validator Overview

UDI validators are used to help ensure that users provide the correct information in the text fields on wizard pages in the UDI Wizard. UDI includes several built-in validators that help you perform typical validations of fields used for entering text, such as preventing users from entering invalid characters or ensuring that the field is not empty.

In addition to the built-in UDI validators, you can create custom UDI validators using the UDI SDK. For more information about creating custom UDI validators using the UDI SDK, see the MDT document *User-Driven Installation Developers Guide*.

Built-in UDI Validators

Table 24 lists the built-in UDI validators. Each built-in validator is discussed in a subsequent section. When a validator detects an invalid entry in a text box, a message is displayed on the wizard page, and the **Next** button is disabled until all invalid entries are resolved.

Table 24. Built-in UDI Validators

Validator	Description
InvalidChars	This validator identifies any invalid characters that have been entered from a list that you configure.
NamedPattern	This validator helps ensure that the text follows a predefined pattern.
NonEmpty	This validator is used to require text in a field.
RegEx	This validator allows you ensure that the text matches a regular expression that you specify as a part of the validator.

InvalidChars

This validator prevents users from entering specific characters. The **Message** box allows you to enter a message that is displayed if the text field contains any of the invalid characters. The **Invalid Characters** box allows you to enter the characters that are considered invalid. The characters are entered without spaces between them.

NamedPattern

This validator helps ensure that the text follows a predefined pattern. The **Message** box allows you to enter a message that is displayed if the text field does not match the named pattern. The **Named Pattern** box allows you to enter

the name of the predefined pattern and must be **Username**, **ComputerName**, or **Workgroup**. The names are case insensitive.

NonEmpty

Use this validator to require text in a field. The **Message** box allows you to enter a message that is displayed if the text field is empty.

RegEx

This validator allows you ensure that the text matches a regular expression that you specify as a part of the validator. The **Message** box allows you to enter a message that is displayed if the text field does not match the regular expression. The **Regular Expression** box allows you to enter the regular expression used for the validation. For more information about how to build regular expressions for this validator, see [TR1 Regular Expressions](#).

UDI Wizard Page Reference

You add a UDI [wizard page](#) to stages from the [Page Library](#) in the [UDI Wizard Designer](#). UDI wizard pages are displayed in the [UDI Wizard](#).

This reference includes:

- An overview of UDI wizard pages, as described in [UDI Wizard Page Overview](#)
- A description of the built-in UDI wizard pages that are provided with MDT, as described in [Built-in UDI Wizard Pages](#)

UDI Wizard Page Overview

Wizard pages are displayed in the [UDI Wizard](#) and collect the information required to complete the deployment process. You create wizard pages using C++ in Visual Studio. The custom wizard pages are implemented as DLLs that the UDI Wizard reads.

Each built-in UDI wizard page has a corresponding UDI [wizard page editor](#), which you use to configure the wizard page in the [UDI Wizard Designer](#).

In addition to the built-in UDI wizard pages, you can create custom UDI wizard pages using the UDI SDK. For more information about creating custom UDI wizard pages using the UDI SDK, see the MDT document *User-Driven Installation Developers Guide*.

Each wizard page can reference the following types of variables:

- Task sequence variables
- Memory variables
- Environment variables

You can reference task sequence and environment variables by bracketing the variable using percent signs (%), such as `%OSDImageIndex%`. You can reference memory variables by bracketing the variable using dollar signs (\$), such as `$VolumeArchitecture$`.

Note If a task sequence variable and an environment variable both have the same name, then the task sequence variable takes precedence over the environment variable.

Table 25 lists the memory variables that are set when the UDI Wizard starts, the description of the variables, and whether the UDI Wizard reads or writes the variables during startup.

Table 25. Memory Variables Set by the UDI Wizard at Startup and Their Descriptions

Variable	Read	Write
LogPath Specifies the fully qualified path to the log files for the UDI Wizard. You can set this variable to one of the following values: <ul style="list-style-type: none">• The value in the <code>_SMSTSLogPath</code> task sequence variable• The value of the <code>%TEMP%</code> environment variable if the <code>_SMSTSLogPath</code> task sequence variable is not set	No	Yes
WizardConfigFilename Specifies the name of the UDI Wizard configuration file currently in use. The ApplicationPage wizard page reads the value of this variable to find the corresponding .app file, which contains the list of applications. For example, if the UDI Wizard configuration file is named <code>config.xml</code> , then the wizard page will look for the corresponding .app file (<code>config.xml.app</code>).	No	Yes

Built-in UDI Wizard Pages

Table 26 lists the built-in UDI wizard pages. Each built-in UDI wizard page is discussed in a subsequent section.

Table 26. Built-in Wizard Pages and Their Descriptions

Wizard page	Description
<u>AdminAccounts</u>	Use this wizard page to set the password for the local administrator account and add other users to the local Administrators group on the target computer.

Wizard page	Description
ApplicationPage	Use this wizard page to configure the list of applications that can be installed during the setup process. These applications can include applications or packages and programs from Configuration Manager.
BitLockerPage	Use this wizard page to configure BitLocker settings for the target computer.
ComputerPage	Use this wizard page to configure the computer name of the target computer, the domain or workgroup to join, and the credential to be used when joining a domain.
ConfigScanPage	Use this wizard page to run UDI tasks that scan the configuration of the target computer to determine whether the target computer is ready for the deployment of the operating system image. This readiness includes having sufficient system resources and ensuring that any prerequisite software is installed and configured properly.
LanguagePage	Use this wizard page to determine which language pack should be installed, the default language for the target operating system, the keyboard locale, and the time zone in which the computer will be physically located.
ProgressPage	Use this wizard page to run UDI tasks that capture the user state migration data from the target computer.
RebootPage	Use this wizard page to notify the user that the target computer is going to be restarted. You can configure the notification message using the UDI Wizard Designer.
SummaryPage	Use this wizard page to notify the user about the configuration options that were selected while running the UDI Wizard. The configuration information displayed on this wizard page is automatically collected from other wizard pages. Some fields on other wizard pages allow you to configure the caption (label) displayed on this wizard page using the UDI Wizard Designer.
UDAPage	Use this wizard page to configure the UDA between the target computer and a specified user. Defining affinity between a computer and a user allows automatic installation of software that is deployed to a user. The UDA feature is only available in Configuration Manager and in the UDI New Computer scenario.
UserStatePage	Use this wizard page to configure the settings for capturing or restoring user state migration data. This wizard page allows the user to select the location to capture user state migration to or restore user state migration data from.

Wizard page	Description
VolumePage	Use this wizard page to configure the settings for the disk volume on target computer where the operating system will be deployed. These settings include selecting the target operating system, selecting the target drive, selecting any Windows installation, and determining whether the target drive should be formatted as a part of the deployment process.
WelcomePage	Use this wizard page to provide information to the user about UDI Wizard and the deployment process. You can configure the notification message using the UDI Wizard Designer.

AdminAccounts

Use this wizard page to set the password for the local administrator account and to add other user to the local Administrators group on the target computer.

Task Sequence Variables

Table 27 lists the **AdminAccounts** task sequence variables with the description and determines whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 27. AdminAccounts Task Sequence Variables

Variable	Read	Write	Config
OSDAddAdmin Specifies a list of additional user names to be added to the local Administrators group on the target computer.	Yes	Yes	Yes
OSDLocalAdminPassword Specifies the passwords for the local built-in Administrator account on the target computer.	Yes	Yes	Yes

ApplicationPage

Use this wizard page to configure the list of application software that can be installed during the setup process. These applications can include applications or packages and programs from Configuration Manager.

Note If applications appear to be disabled, the application may require administrator approval but has not yet been approved. If the **Require administrator approval if users request this application** check box is selected for the application, verify that the application has been approved. For more information, see [How to Deploy Applications in Configuration Manager](#).

Task Sequence Variables

Table 28 lists the **ApplicationPage** task sequence variables with the description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 28. ApplicationPage Task Sequence Variables

Variable	Read	Write	Config
ApplicationBaseVariable Specifies the name used as the base for the task sequence variable names created for each Configuration Manager application selected on the ApplicationPage wizard page. This variable is configured using the Edit Software Settings button in the Edit Settings group on the Ribbon in the UDI Wizard Designer. A separate task sequence variable is created for each application selected on this page. The default value for this variable is APPLICATIONS . So, for example, the default names of the task sequence variables created for each application selected on this page will be APPLICATIONS001 , APPLICATIONS002 , APPLICATIONS003 , and so forth.	No	Yes	Yes
OSDApplicationList Specifies the list of application identifiers that should be initially selected. The variable contains a list of numeric values separated by semicolons (;). The application identifiers are found in the Id attribute of the Application element in the UDI Wizard application configuration file (UDIWizard_Config.xml.app). There is a separate Application element for each application displayed in this wizard page.	Yes	No	No
OSDArchitecture Specifies the processor architecture of the target computer. The ApplicationPage wizard page uses this variable to filter the available applications when the VolumeArchitecture memory variable has not been set. However, if the VolumeArchitecture memory variable has been set, it always takes precedence over this task sequence variable for filtering the available applications. The value for this variable can be:	Yes	No	No

Variable	Read	Write	Config
<ul style="list-style-type: none"> • x86, which indicates a 32-bit processor architecture • amd64, which indicates a 64-bit processor architecture 			
OSDBaseVariableName <p>Specifies the name used as the base for the task sequence variable names created for each Configuration Manager package and program selected on the ApplicationPage wizard page. This variable is configured using the Edit Software Settings button in the Page Behavior group on the Ribbon in the UDI Wizard Designer.</p> <p>A separate task sequence variable is created for each application selected on this page. The default value for this variable is PACKAGES. So, for example, the default names of the task sequence variables created for each application selected on this page will be <i>PACKAGES001</i>, <i>PACKAGES002</i>, <i>PACKAGES003</i>, and so forth.</p>	No	Yes	Yes

Memory Variables

Table 29 lists the **ApplicationPage** memory variables with the description and whether the variable is read or written by the wizard page.

Table 29. ApplicationPage Memory Variables

Variable	Read	Write
VolumeArchitecture <p>Specifies the processor architecture of the target operating system image to be deployed (whether the image contains a 32-bit or 64-bit operating system). When this page is displayed, it checks to see if this variable has changed. If the variable has changed since the last time the wizard page was displayed, the wizard page filters the programs available for selection based on architecture of the target operating system. For example, if a 32-bit operating system is to be deployed, then the wizard page removes (filters) any 64-bit applications from the list of available applications on the wizard page.</p>	Yes	No
WizardConfigFilename <p>Specifies the name of the UDI Wizard configuration file currently in use. If the value of the Link.Uri setter property is empty, the ApplicationPage wizard page reads the</p>	Yes	No

Variable	Read	Write
<p>value of this variable to find the corresponding .app file, which contains the list of applications. For example, if the UDI Wizard configuration file is named <i>config.xml</i>, then the wizard page will look for the corresponding .app file (<i>config.xml.app</i>). This variable is set when the UDI Wizard starts.</p> <p>The Link.Uri setter property is set on the Software Settings dialog box, which can be opened using the Edit Software Settings button in the Page Behavior group on the Ribbon in the UDI Wizard Designer.</p>		

BitLockerPage

This wizard page is used to configure BitLocker settings for the target computer.

Task Sequence Variables

Table 30 lists the BitLockerPage task sequence variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 30. BitLockerPage Task Sequence Variables

Variable	Read	Write	Config
BDEInstallSuppress Specifies whether BitLocker installation should be suppressed. If the variable is set to: <ul style="list-style-type: none"> • YES, then the Enable BitLocker check box is selected and the installation is performed • NO, then the Enable BitLocker check box is cleared and the installation is not performed 	Yes	Yes	Yes
BDEKeyLocation Specifies the fully qualified path to the location where the BitLocker encryption keys are stored, which can be a local or UNC path. This variable is set to the value of the KeyLocation setter value in the UDI Wizard configuration file for the BitLockerPage . This variable is only considered valid when the OSDBitLockerMode is set to TPMKEY or KEY .	No	Yes	No
BDEPin Specifies the BitLocker PIN value if the Enable BitLocker using TPM and Pin option is selected.	Yes	Yes	Yes

Variable	Read	Write	Config
OSDBitLockerCreateRecoveryPassword Specifies whether a BitLocker recovery password should be stored in AD DS. If the variable is set to: <ul style="list-style-type: none">• AD, then the In Active Directory option is selected and the recovery keys will be stored in AD DS (recommended)• NONE, then the Do not create a recovery key option is selected and the recovery keys will not be stored in AD DS (not recommended)	No	Yes	No
OSDBitLockerMode Specifies the mode to be used when enabling BitLocker on the target computer. Valid values include: <ul style="list-style-type: none">• TPM. This value indicates that the Enable BitLocker using TPM only option is selected and that only TPM will be used when enabling BitLocker on the target computer.• TPMPIN. This value indicates that the Enable BitLocker using TPM and Pin option is selected and that TPM and a user-specified PIN will be used when enabling BitLocker on the target computer.• TPMKEY. This value indicates that the Enable BitLocker using TPM and Startup Key option is selected and that TPM and a startup key will be used when enabling BitLocker on the target computer.• KEY. This value indicates that the Enable BitLocker using only an External Startup Key option is selected and that only an external startup key will be used when enabling BitLocker on the target computer.	No	Yes	No
OSDBitLockerStartupKeyDrive Specifies the drive letter where the BitLocker external startup key will be stored on the target computer. This variable is only considered valid when OSDBitLockerMode is set to TPMKEY or KEY .	No	Yes	No
OSDBitLockerWaitForEncryption Specifies whether the task sequence should	Yes	Yes	Yes

Variable	Read	Write	Config
<p>wait until BitLocker encryption finishes. If the variable is set to:</p> <ul style="list-style-type: none"> • YES, then the Wait for BitLocker Encryption to complete on all drives before continuing check box is selected and the task sequence will wait until the installation is complete • NO, then the Wait for BitLocker Encryption to complete on all drives before continuing check box is cleared and the task sequence will not wait until the installation is complete 			

Configuration Variables

Table 31 lists the **BitLockerPage** configuration variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 31. BitLockerPage Configuration Variables

Variable	Read	Write	Config
<p>KeyLocation</p> <p>Specifies the fully qualified path to the location where the BitLocker encryption keys are stored, which can be a local or UNC path. This configuration value is used to set the value of the BDEKeyLocation task sequence variable for the BitLockerPage. This variable is only considered valid when OSDBitLockerMode is set to TPMKEY or KEY.</p>	Yes	No	Yes

ComputerPage

Use this wizard page to configure the computer name of the target computer, the domain or workgroup to join, and the credentials to be used when joining a domain. When you configure this page to join the target computer to a domain, this wizard page will validate the credentials you provide for joining the domain in AD DS by default. Then, this wizard page attempts to modify a computer object in AD DS to verify that the user credentials provided on this page have permissions to create or modify the computer object. You can disable either of these behaviors. If you disable the validation of the credentials, then the verification of permissions for creating or modifying computer objects is also disabled. Both of these validations occur when the **Next** button is clicked. If either of the validations encounters an error, an error message will be displayed and this page will continue to be displayed.

The following is the order of precedence for determining the default computer name:

1. If the **UserExistingComputerName** value in the UDI Wizard configuration file is set to **TRUE**, then the existing computer name is used (if present).
2. If the **OSDComputerName** task sequence variable is set, then the computer name in that variable is used.
3. If a default value is specified for the computer name in the UDI Wizard configuration file, then that value is used.

Task Sequence Variables

Table 32 lists the **ComputerPage** task sequence variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 32. ComputerPage Task Sequence Variables

Variable	Read	Write	Config
OSDComputerName Specifies the name of the target computer. The value of this variable is set in the Computer name box.	Yes	Yes	Yes
OSDDomainName Specifies the name of the domain to which the target computer is to be joined. The value of this variable is set in the Domain box.	Yes	Yes	Yes
OSDDomainOUName Specifies the name of the OU within the domain to which the target computer object is to be placed. The value of this variable is set in the Organizational Unit box.	Yes	Yes	Yes
OSDJoinAccount Specifies the user account used to join the target computer to the domain. The value for this variable is set in the User name box.	Yes	Yes	Yes
OSDJoinPassword Specifies the password for the user account used to join the target computer to the domain. The value for this variable is set in the Password and Confirm password boxes.	Yes	Yes	Yes
OSDNetworkJoinType Specifies if the target computer is to be joined to a workgroup or a domain. If the value is set	No	Yes	No

Variable	Read	Write	Config
<p>to:</p> <ul style="list-style-type: none"> • 0, then the Domain option is selected and the target computer will be joined to a domain • 1, then the Workgroup option is selected and the target computer will be joined to a workgroup 			
<p>SMSTSAssignUsersMode</p> <p>Specifies the mode for configuring user affinity in Configuration Manager. Use this variable to configure the behavior of creating affinity between the target computer and user accounts in the SMSTSUdaUsers task sequence variable. If this variable is not specified prior to displaying this page, the value of this variable is set to Pending.</p> <p>Possible values for this variable include:</p> <ul style="list-style-type: none"> • Auto. The affinity processing is automatically approved by Configuration Manager. • Pending. The affinity processing rules will require approval by a Configuration Manager administrator. • Disabled. No affinity processing will occur. 	No	Yes	No

Configuration Variables

Table 33 lists the **ComputerPage** configuration variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 33. ComputerPage Configuration Variables

Variable	Read	Write	Config
<p>ADComputerObjectCheck</p> <p>Specifies whether the ComputerPage wizard page will validate that the credentials provided have the appropriate permissions to modify a computer object in AD DS prior to continuing to the next wizard page.</p> <p>Note This configuration setting is ignored if ADCredentialCheck is set to FALSE.</p> <p>If the value is set to:</p>	Yes	No	Yes

Variable	Read	Write	Config
<ul style="list-style-type: none"> • TRUE, then the Active Directory Computer Object Check check box is selected in the wizard page editor in the Domain Join Credentials section in the UDI Wizard Designer, and permissions to modify a computer object for the credentials are validated • FALSE, then the Active Directory Computer Object Check check box is cleared in the wizard page editor in the Domain Join Credentials section in the UDI Wizard Designer, and permissions to modify a computer object for the credentials are not validated 			
<p>ADCredentialCheck</p> <p>Specifies whether the ComputerPage wizard page will validate the credentials provided for joining a domain prior to continuing to the next wizard page. If the value is set to:</p> <ul style="list-style-type: none"> • TRUE, then the Active Directory Credential Check check box is selected in the wizard page editor in the Domain Join Credentials section in the UDI Wizard Designer, and credentials are validated If this configuration setting is set to TRUE, then the credentials are validated even if the credential fields are disabled (locked). • FALSE, then the Active Directory Credential Check check box is cleared in the wizard page editor in the Domain Join Credentials section in the UDI Wizard Designer, and credentials are not validated If this configuration setting is set to FALSE, then the ADComputerObjectCheck configuration setting is ignored and the validation that the provided credentials can modify a computer object in AD DS is not performed. 	Yes	No	Yes
<p>UseExistingComputerName</p> <p>Specifies whether the ComputerPage wizard page will use the existing computer name on the target computer as the default for the computer name.</p> <p>Note This check box is only relevant for the Refresh</p>	Yes	No	Yes

Variable	Read	Write	Config
<p>Computer deployment scenario.</p> <p>If the value is set to:</p> <ul style="list-style-type: none"> • TRUE, then the Use Existing Computer Name check box is selected in the wizard page editor in the Computer Name section in the UDI Wizard Designer, and the existing computer name will be used as the default computer name for the target computer after the new operating system is deployed • FALSE, then the Use Existing Computer Name check box is cleared in the wizard page editor in the Computer Name section in the UDI Wizard Designer, and the existing computer name will not be used as the default computer name for the target computer after the new operating system is deployed 			

ConfigScanPage

Use this wizard page to run UDI tasks that scan the configuration of the target computer to determine whether the target computer is ready for the deployment of the operating system image. This readiness includes having sufficient system resources and any prerequisite software being installed and configured properly. In addition, other UDI tasks are run that collect configuration information about the target computer, such as identifying:

- Whether the computer is connected to power (as opposed to running on a battery)
- Whether the computer is connected to a wired network connection (as opposed to using a wireless network connection)
- Any installed applications
- Any installed printers

LanguagePage

Use this wizard page to determine which language packs should be installed, the default language for the target operating system, the keyboard locale, and the time zone in which the computer will be located.

Task Sequence Variables

Table 34 lists the **LanguagePage** task sequence variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 34. LanguagePage Task Sequence Variables

Variable	Read	Write	Config
InputLocale Specifies the input locale of the target operating system. You set the value of this variable in the Time and currency format box. If not specified, the input locale configured in the image is used.	Yes	Yes	Yes
KeyboardLocale Specifies the keyboard locale of the target operating system. Set the value of this variable in the Keyboard layout box. If not specified, the keyboard locale configured in the image is used.	Yes	Yes	Yes
OSDTimeZone Specifies the time zone where the target computer will be physically located. Set the value of this variable in the Time zone box. If not specified, the time zone configured in the image is used.	Yes	Yes	Yes
UILanguage Specifies the default language to be used for the target operating system. Set the value of this variable in the Language to install box. If not specified, the language configured in the image is used.	Yes	Yes	Yes

ProgressPage

Use this wizard page to run UDI tasks that capture the user state migration data from the target computer. These tasks include:

- Copying the application discovery file to the location selected on the [UserStatePage](#) wizard page
- Copying the printer configuration file to the location selected on the [UserStatePage](#) wizard page
- Copying the list of installed products to the location selected on the [UserStatePage](#) wizard page
- Running the USMT and saving the user state migration data to the location selected on the [UserStatePage](#) wizard page

RebootPage

Use this wizard page to notify the user that the target computer is going to be restarted. You can configure the notification message using the UDI Wizard Designer.

SummaryPage

Use this wizard page to notify the user about the configuration options that were selected while running the UDI Wizard. The configuration information displayed on this wizard page is automatically collected from other wizard pages. Some fields on other wizard pages allow you to configure the caption (label) displayed on this wizard page using the UDI Wizard Designer.

UDAPage

Use this wizard page to configure the UDA between the target computer and a specified user. Assigning a user as the primary user of a computer allows automatic installation of software that is deployed to that user. The UDA feature is only available in Configuration Manager and only in the New Computer deployment scenario.

Task Sequence Variables

Table 35 lists the **UDAPage** task sequence variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 35. UDAPage Task Sequence Variables

Variable	Read	Write	Config
SMSTSAssignUsersMode Specifies the mode for configuring user affinity in Configuration Manager. Use this variable to configure the behavior of creating affinity between the target computer and user accounts in the SMSTSUdaUsers task sequence variable. To set this variable, select the Use User Device Affinity check box. If the variable is set to: <ul style="list-style-type: none">• Auto, then the affinity processing is automatically approved by Configuration Manager• Pending, then the affinity processing rules will require approval by a Configuration Manager administrator (This is the value used when the Use User Device Affinity check box is selected.)• Disabled, then no affinity processing will	No	Yes	No

Variable	Read	Write	Config
occur			
SMSTSUdaUsers Specifies the users to be associated with the target computer. The User Device Affinity Account sets this variable. This variable can have one or many users specified and is in the format Domain\User1, Domain\User2.	Yes	Yes	Yes

UserStatePage

Use this wizard page to configure the settings for capturing or restoring user state migration data. This wizard page is used for both user state migration data capture and restore.

The **UserStatePage** can capture or restore user state migration data from a disk locally attached to the target computer, a USB drive attached to the target computer, or a network shared folder. In addition, you can select to not restore any user data. The code logic behind the wizard page enables, disables, or automatically selects each of the following options based on the deployment scenario and whether the disk is being formatted:

- **No Data to Restore.** This option indicates that there is no user state migration data to restore and sets the **OSDUserStateMode** task sequence variable and **UserStateMode** variable to **NoData**.
- **Local.** This option indicates that the user state migration data should be stored on a disk locally attached to the target computer and sets the **OSDUserStateMode** task sequence variable and **UserStateMode** variable to **Local**.
- **USB.** This option indicates that the user state migration data should be stored on a USB disk locally attached to the target computer and sets the **OSDUserStateMode** task sequence variable and **UserStateMode** variable to **USB**.
- **Network.** This option indicates that the user state migration data should be stored on a network shared folder and sets the **OSDUserStateMode** task sequence variable and **UserStateMode** variable to **Network**.

NEWCOMPUTER Stage Behavior

The NEWCOMPUTER stage is used for computers on which no user state migration data exists. The New Computer deployment scenario can be used as the second part of the Replace Computer deployment scenario. If the user selects to:

- Format the disk on the target computer, then the **UserStatePage** assumes that no user state migration data is located on the local hard disk, so the **Local** option is disabled and all other options are enabled

- Not format the disk on the target computer, then the **UserStatePage** assumes that there is user state migration data to be restored, and all options are disabled other than the **Local** option (Using the **Local** option provides a faster method for restoring the user state migration data than the USB or network shared folder methods.)

Table 36 lists the behavior of the options on the wizard page for the NEWCOMPUTER stage. The **Format** column indicates whether the target hard disk is to be formatted as a part of the deployment. The other columns indicate the configuration of the options when the **UserStatePage** is loaded.

Table 36. Behavior of Options for the NEWCOMPUTER Stage

Format	NoData	Local	USB	Network
Yes	Enabled	Disabled	Enabled	Enabled
No	Disabled	Selected	Disabled	Disabled

NewComputer.Prestaged Stage Behavior

The NEWCOMPUTER.Prestaged stage is based on the prestaged media feature in Configuration Manager. Because the local hard disk is new, there is no user state migration data to be restored from the local hard disk, so the **Local** option is disabled. All other options are valid for this deployment scenario and are enabled. No default option is selected.

Table 37 lists the behavior of the options on the wizard page for the NewComputer.Prestaged stage. The **Format** column indicates whether the target hard disk is to be formatted as a part of the deployment. The other columns indicate the configuration of the options when the **UserStatePage** is loaded.

Table 37. Behavior of Options for the NewComputer.Prestaged Stage

Format	NoData	Local	USB	Network
N/A	Enabled	Disabled	Enabled	Enabled

REFRESH Stage Behavior

The REFRESH stage is initiated in a full Windows operating system, instead of Windows PE. If the user selects to:

- Format the disk on the target computer, then the **UserStatePage** assumes that no user state migration data is to be restored, and all options are disabled other than the **NoData** option
- Not format the disk on the target computer, then the **UserStatePage** assumes that there is user state migration data to be restored, and all options are disabled other than the **Local** option (Using the **Local** option provides a faster method for restoring the user state migration data than the USB or network shared folder methods.)

Table 38 lists the behavior of the options on the wizard page for the REFRESH stage. The **Format** column indicates whether the target hard disk is to be formatted as a part of the deployment. The other columns indicate the configuration of the options when the **UserStatePage** is loaded.

Table 38. Behavior of Options for the REFRESH Stage

Format	NoData	Local	USB	Network
Yes	Selected	Disabled	Disabled	Disabled
No	Disabled	Selected	Disabled	Disabled

REPLACE.WinPE Stage Behavior

The REPLACE.WinPE stage captures the user state migration data from the existing (old) computer, and then restores the user state migration data later using one of the New Computer deployment scenarios. Because two different computers are involved in the deployment, the user state migration data must be saved to a USB drive or to a network shared folder. Saving user state migration data to a local disk is unavailable.

Table 39 lists the behavior of the options on the wizard page for the REPLACE.WinPE stage. The **Format** column indicates whether the target hard disk is to be formatted as a part of the deployment. The other columns indicate the configuration of the options when the **UserStatePage** is loaded.

Table 39. Behavior of Options for the REPLACE.WinPE Stage

Format	NoData	Local	USB	Network
N/A	Disabled	Disabled	Enabled	Enabled

Task Sequence Variables

Table 40 lists the **UserStatePage** task sequence variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 40. UserStatePage Task Sequence Variables

Variable	Read	Write	Config
_SMSTsInWinPE Specifies whether the UDI Wizard is running in Windows PE. If the variable is set to: <ul style="list-style-type: none"> • TRUE, then the UDI Wizard is running in Windows PE • FALSE, then the UDI Wizard is not running in Windows PE, but rather in a full Windows operating system 	Yes	No	No

Variable	Read	Write	Config
OSDDataSourceDirectory Specifies the directory in which the user state migration data is stored.	No	Yes	No
OSDDataSourceDrive Specifies the USB drive used for capturing and restoring user state migration data, which you select from the USB Target Drive box. If the variable is set prior to showing the wizard page, the value of the variable is used as the default value.	Yes	Yes	No
OSDDiskPart Specifies whether the drive selected for the installation of the target operating system should be formatted and partitioned. You set this variable on the VolumePage wizard page, and the code on this wizard page uses it to determine which options are selected and enabled by default. For more information, see UserStatePage .	Yes	No	Yes
OSDHardLinks Specifies whether the user state migration data is to be captured to or restored from a local drive. If the variable is set to: <ul style="list-style-type: none"> • TRUE, then the Local option was selected, and user state migration data will be captured or restored from a local drive that is attached to the target computer • FALSE, then the Local option was not selected, and no user state migration data will be captured or restored from a local drive that is attached to the target computer 	No	Yes	No
OSDRestoreData Specifies whether there is data to be restored. If the variable is set to: <ul style="list-style-type: none"> • TRUE, then the Local, USB Target Drive, or Network option was selected, and user state migration data will be captured or restored from the target computer • FALSE, then the No Data to Restore option was selected, and no user state migration data will be captured or restored from the target computer 	No	Yes	No

Variable	Read	Write	Config
OSDUserStateKey Specifies the user name used to secure the user state migration data. The user name is provided when the user state migration data is captured. The same user name and password must be provided when the user state migration data is restored. You set the value of this variable in the User name box.	Yes	Yes	Yes
OSDUserStateKeyPassword Specifies the password for the user name used to secure the user state migration data. Set the value of this variable in the Password and Confirm password boxes.	Yes	Yes	Yes
OSDUserStateMode Specifies the mode (method) for capturing or restoring the user state migration data. The value of this variable is set by the options selected. If the variable is set to: <ul style="list-style-type: none">• NoData, then the No Data to Restore option was selected, and no user state migration data will be captured or restored• Local, then the Local option was selected, and the user state migration data will be captured or restored from a local hard disk on the target computer• Network, then the Network option was selected, and the user state migration data will be captured to or restored from a network shared folder When used in capture mode, this option creates a folder based on a hash of the user name and password so that the identity of the user state migration data is protected. The exact same user name and password must be used when restoring the user state migration data so that the wizard page can accurately locate the folder.• USB, then the USB Target Drive option was selected, and the user state migration data will be captured to or restored from a USB drive that is physically attached to the target computer The wizard page behavior for USB drives is also affected by the Format,	No	Yes	No

Variable	Read	Write	Config
FormatPrompt , and MinimumDriveSize variables.			
SMSConnectNetworkFolderPath Specifies the network shared folder used for capturing and restoring user state migration data, which is selected from the Network box. The Network box displays a user-friendly name for the network shared folder that is configured in the Network Shares box in the Network Combo Box section on the wizard page editor in the UDI Wizard Designer. If the variable is set prior to showing the wizard page, the value of the variable is used as the default value.	Yes	Yes	Yes

Memory Variables

Table 41 lists the **UserStatePage** memory variables with a description and whether the variable is read or written by the wizard page.

Table 41. UserStatePage Memory Variables

Variable	Read	Write
DriveLetter Specifies the drive letter for the USB drive selected in the USB Target Drive box on the wizard page. The value of this variable will be the drive letter, including the colon (:) suffix, such as <i>M</i> :	No	Yes
TargetDrive Specifies the caption displayed in the USB Target Drive box on the wizard page for the USB drive selected on the target computer. The value of this variable will be similar to the following example: <i>M: VendorA Ultra TD v1.0 USB Device (74.5 GB)</i>	No	Yes
UserStateMode Specifies the option selected with the options on the wizard page and is set to the same value as the OSDUserStateMode variable. Valid values for this variable include: <ul style="list-style-type: none"> • NoData, which indicates that the No Data to Restore option was selected • Local, which indicates that the Local option was selected • USB, which indicates that the USB Target Drive 	No	Yes

Variable	Read	Write
<p>option was selected</p> <ul style="list-style-type: none"> • Network, which indicates that the Network option was selected 		

Configuration Variables

Table 42 lists the **UserStatePage** configuration variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 42. UserStatePage Configuration Variables

Variable	Read	Write	Config
DataSourceText <p>Specifies an informational message that instructs the user performing the user state capture or restore about how to use the wizard page. You set the value of this variable in the Instruction Text box in the Message section on the wizard paged editor in the UDI Wizard Designer.</p>	Yes	No	Yes
Format <p>Specifies whether the USB drive selected for capturing user state on the target computer should be partitioned and formatted prior to capturing user state migration data. Set the value of this variable by selecting the Format the USB drive before capture check box in the USB Combo Box section on the wizard paged editor in the UDI Wizard Designer.</p> <p>If the variable is set to:</p> <ul style="list-style-type: none"> • TRUE, then the drive is formatted prior to capturing user state migration data • FALSE, then the drive is not formatted prior to capturing user state migration data 	Yes	No	Yes
FormatPrompt <p>Specifies whether the user must confirm that the USB drive used for capturing user state migration data is to be formatted prior to performing the capture. Set the value of this variable by selecting the Prompt the user before formatting the target drive check box in the USB Combo Box section on the wizard paged editor in the UDI Wizard Designer.</p>	Yes	No	Yes

Variable	Read	Write	Config
Note This variable is only valid if the OSDUserStateMode task sequence variable is set to USB .			
MinimumDriveSize Specifies the minimum available free disk space in gigabytes required for a drive to be available for storing user state migration data. The value of this variable acts as a filter, and you set it in the Minimum Drive Size text box in the USB Combo Box section on the wizard paged editor in the UDI Wizard Designer.	Yes	No	Yes
NetworkDrive Specifies the drive letter that this wizard page uses to map to the network shared folder in the SMSConnectNetworkFolderPath task sequence variable. The network shared folder mapping is used for capturing or restoring the user state migration data. Set the value of this variable in the Mapped Drive Letter box in the Network Combo Box section on the wizard paged editor in the UDI Wizard Designer. The drive letter specified must include the colon (:) after the drive letter and must not be in use on the target computer. For example, if the target computer has drives C: and D:, then C: and D: could not be used for this variable. Note This variable is only valid if the OSDUserStateMode task sequence variable is set to Network .	Yes	No	Yes
State Specifies whether the wizard page is being used for capturing or restoring the user state migration data. Set the value of this variable in the Capture or Restore box in the Capture/Restore Location section on the wizard paged editor in the UDI Wizard Designer. If the variable is set to: <ul style="list-style-type: none"> • Capture, then the wizard page is used to capture user state migration data • Restore, then the wizard page is used to restore user state migration data 	Yes	No	Yes

VolumePage

Use this wizard page to configure the settings for the disk volume on the target computer on which the operating system will be deployed. These settings include selecting the target operating system, selecting the target drive, selecting any Windows installation, and determining whether the target drive should be formatted as a part of the deployment process.

Task Sequence Variables

Table 43 lists the **VolumePage** task sequence variables with a description and whether the variable is read by the wizard page, written by the wizard page, or can be configured in the UDI Wizard configuration file.

Table 43. VolumePage Task Sequence Variables

Variable	Read	Write	Config
OSDDiskPart Specifies whether the drive selected for deploying the target operating system on the target computer should be partitioned and formatted prior to capturing user state migration data. The value of this variable is set by the one of the following check boxes on the wizard page: <ul style="list-style-type: none"> • Clean the selected volume. This check box appears when the UDI Wizard is running in a full Windows operating system. You can configure the text message using the FormatFullOS setter property for the wizard page in the UDI Wizard configuration file. • Partition and format disk 0. This check box appears when the UDI Wizard is running in Windows PE. You can configure the text message using the FormatWinPE setter property for the wizard page in the UDI Wizard configuration file. The code logic behind the UserStatePage wizard page uses this variable to determine which options are selected and enabled by default. If the variable is set to: <ul style="list-style-type: none"> • TRUE, then the drive is partitioned and formatted prior to deploying the target operating system • FALSE, then the drive is not partitioned and formatted prior to deploying the target 	Yes	Yes	Yes

Variable	Read	Write	Config
operating system			
OSDImageIndex Specifies a numeric index of the operating system image in the .wim file, which is selected in the Image Selection combo box. You configure the list of possible operating system images in the Image Selection box in the Image Combo Box Values list in the Image Combo Box section on the VolumePage wizard page editor. The image index is configured as a part of each image in the Image Combo Box Values list.	Yes	Yes	Yes
OSDImageName Specifies the name of the operating system image in the .wim file, which is selected in the Image Selection box. The list of possible operating system images in the Image Selection combo box is configured in the Image Combo Box Values list in the Image Combo Box section on the VolumePage wizard page editor. The image name is configured as a part of each image in the Image Combo Box Values list.	No	Yes	No
OSDTargetDrive Specifies the drive letter for the volume selected in the Volume box on the wizard page. The value of this variable will be the drive letter, including the colon (:) suffix, such as C:.	No	Yes	No
OSDWinPEWindir Specifies the location of an existing installation of Windows on the target computer. Set the value of this variable in the Windows Directory box on the wizard page.	No	Yes	No

Memory Variables

Table 44 lists the **VolumePage** memory variables with a description and whether the variable is read or written by the wizard page.

Table 44. VolumePage Memory Variables

Variable	Read	Write
VolumeArchitecture Specifies the processor architecture of the operating	No	Yes

Variable	Read	Write
<p>system to be deployed, which is selected in the Image Selection box. The VolumeArchitecture wizard page consumes this variable to filter the architecture of applications displayed on that page. For example, if a 32-bit operating system is to be deployed, then the VolumeArchitecture wizard page removes (filters) any 64-bit applications from the list of available applications.</p> <p>If the variable is set to:</p> <ul style="list-style-type: none"> • x86, then a 32-bit operating system was selected • amd64, then 64-bit operating system was selected 		

WelcomePage

Use this wizard page to provide information to the user about the UDI Wizard and the deployment process. You can configure the notification message using the UDI Wizard Designer.

UDI Build Your Own Page Toolbox Control Reference

The Build Your Own Page feature in UDI allows you to create custom wizard pages that you can use to collect additional deployment information for use in UDI. You can create custom wizard pages using the:

- **Build Your Own Page feature.** This feature allows you to create a custom wizard page for collecting deployment information without requiring you to write code or have developer skills. Use this feature if you need to collect basic information without advanced user interaction. For example, you cannot add any code or customize UI fonts using this feature.
- **UDI SDK and Visual Studio.** Use this SDK if you want to create an advanced, fully customized wizard page in Visual Studio for collecting deployment information. Although the UDI SDK allows you to create customized wizard pages, such as adding custom code or changing fonts, this method requires developer skills.

For more information on using the UDI SDK to create custom wizard pages, see "Creating Custom UDI Wizard Pages" in the *User-Drive Installation Developers Guide*.

The Build Your Own Page feature includes a toolbox of controls that you can add to your custom wizard page from the Build Your Own Page toolbox, which is displayed when you view the custom wizard page on the **Configure** tab in the UDI Wizard Designer.

Table 45 lists the types of controls to your custom wizard page, which is illustrated in Figure 5. Each of these controls is discussed in further detail in a subordinate section.

Table 45. Types of Controls in the UDI Build Your Own Page Toolbox

Control type	Description
Checkbox control	This control allows you select or clear a configuration option and behaves as a traditional UI check box.
Combobox control	This control allows you to select an item from a list of items and behaves as a traditional UI drop-down list.
Line control	This control allows you to add a horizontal line to divide one portion of the custom wizard page from another.
Label control	This control allows you to add descriptive, read-only text to the wizard page.
Radio control	This control allows you to select one configuration option from a group of two or more options.
Bitmap control	This control allows you to add a bitmap graphic (.bmp file) to the custom wizard page.
Textbox control	This control allows you to enter text on the custom wizard page.

You can add any combination of these controls to your custom wizard page based on the information you want to collect. In addition, you can use the **Show Gridlines** check box to show or hide gridlines that can be used to assist in visually designing the custom wizard page.

Figure 5 provides an example of a custom wizard page and the Build Your Own Page toolbox.

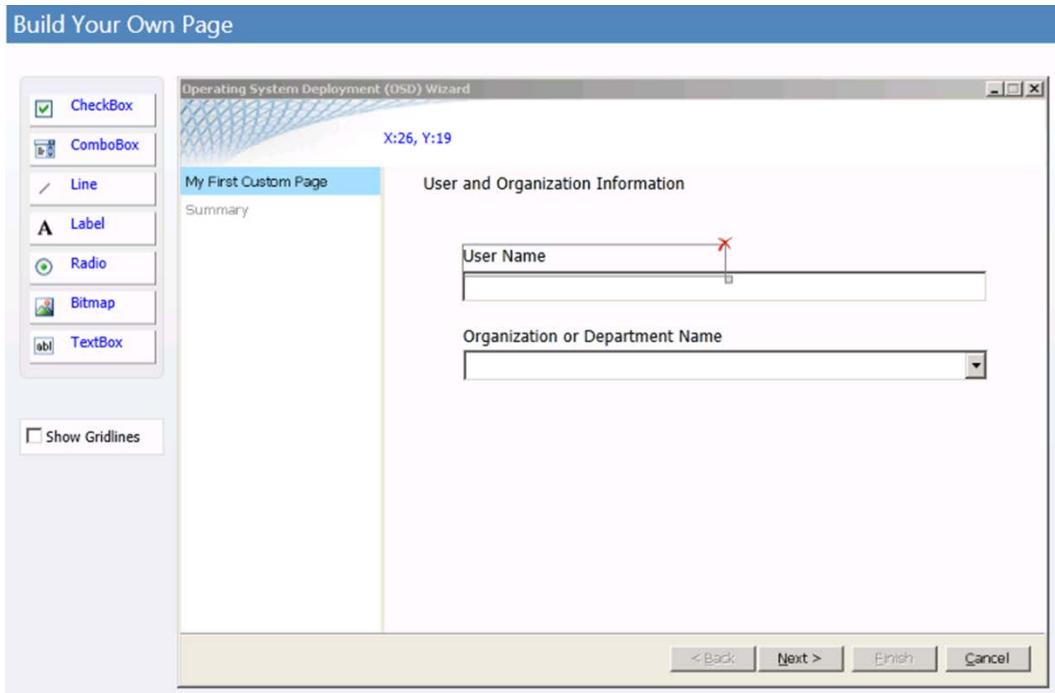


Figure 5. Example custom wizard page

Checkbox Control

This control allows you select or clear a configuration option and behaves as a traditional UI check box. This control has a corresponding label that you can use to describe the purpose of the check box. The state of this control is True when the check box is selected and False when the check box is cleared. The state of the check box is stored in the task sequence variable configured for this control.

Layout Properties

Layout properties are used to configure the UI characteristics of the control and are configured on the **Layout** tab in the UDI Wizard Designer. Table 46 lists the layout properties for the **Checkbox** control and provides a brief description of each property

Table 46. Checkbox Control Layout Properties

Property	Description
X	Use this property to configure the horizontal position of the control.
Y	Use this property to configure the vertical position of the control.
Label	Use this property to configure the descriptive text associated with the check box.
Width	Use this property to configure the width of the control.

Property	Description
	<p>Note If the text entered in the Label property is longer than the width of the control, the text is clipped and not displayed.</p>
Height	<p>Use this property to configure the height of the control.</p> <p>Note If the text entered in the Label property is taller than the height of the control, the text is clipped.</p>

Settings Properties

Settings properties are used to configure the data initially shown in a control (the default value) and where the information collected from the user is saved. Table 47 lists the settings properties for the **Checkbox** control and provides a brief description of each property.

Table 47. Checkbox Control Settings Properties

Property	Description
Default value	Use this property to configure the default value for the control. For a check box, the default value is False .
Task sequence variable name	<p>Use this property to configure the task sequence variable where the information collected from the user is stored. If the task sequence variable:</p> <ul style="list-style-type: none"> Does not already exist, the task sequence variable is created and set to the value the user provides Already exists, the existing value of the task sequence variable is overwritten with the value the user provides
Friendly display name visible in the summary page	Use this property to configure the descriptive name that appears on the Summary wizard page. This name is used to describe the value that was saved in the Task sequence variable name property for this control.
Unlocked	<p>Use this property to configure whether the user is able to interact with the control. By default, the control is enabled. This button displays the following status:</p> <ul style="list-style-type: none"> Unlocked. The control is enabled, and users can enter information using it. Locked. The control is disabled, and users are unable to enter information using it. <p>Note If you disable (lock) a control, you must provide the information the control collected by configuring MDT properties in CustomSettings.ini or in the MDT DB. Otherwise, the UDI Wizard will not collect the necessary information, and the UDI deployment will fail.</p>

Combobox Control

This control allows you to select an item from a list of items and behaves as a traditional UI drop-down list. This control allows you to add or remove items from the list and provide a corresponding value that will be set in the task sequence variable configured for this control.

Layout Properties

Layout properties are used to configure the UI characteristics of the control and are configured on the **Layout** tab in the UDI Wizard Designer. Table 48 lists the layout properties for the **Combobox** control and provides a brief description of each property.

Table 48. Combobox Control Layout Properties

Property	Description
X	Use this property to configure the horizontal position of the control.
Y	Use this property to configure the vertical position of the control.
Width	Use this property to configure the width of the control. Note If the text entered in the control is longer than the width of the control, the text is not displayed.
Height	Use this property to configure the height of the control. Note If the text entered in the control is taller than the height of the control, the text is clipped.
Data Items	Use this property to configure the list of data items displayed in the control. Each data item has the following properties: <ul style="list-style-type: none"> • Value. The value stored in the task sequence variable when the data item is selected • DisplayValue. The value displayed to the user in the control You can: <ul style="list-style-type: none"> • Add data items to the list using the blue plus sign button immediately to the right of the list of data items • Remove data items from the list using the red X button immediately to the right of the list of data items Note You cannot change the sequence of the data item in the list after an item is added to the list. Ensure that you enter the data items in the order you wish them to appear in the control.

Settings Properties

Settings properties are used to configure the data that is initially shown in a control (the default value) and where the information collected from the user is saved. Table 49 lists the settings properties for the **Combobox** control and provides a brief description of each property.

Table 49. Combobox Control Settings Properties

Property	Description
Task sequence variable name	<p>Use this property to configure the task sequence variable where the information collected from the user is stored. If the task sequence variable:</p> <ul style="list-style-type: none"> • Does not already exist, the task sequence variable is created and set to the value the user provides • Already exists, the existing value of the task sequence variable is overwritten with the value the user provides
Friendly display name visible in the summary page	<p>Use this property to configure the descriptive name that appears on the Summary wizard page. This name is used to describe the value that was saved in the Task sequence variable name property for this control.</p>
Unlocked	<p>Use this property to configure whether the user is able to interact with the control. By default, the control is enabled. This button displays the following status:</p> <ul style="list-style-type: none"> • Unlocked. The control is enabled, and users can enter information using it. • Locked. The control is disabled, and users are unable to enter information using it. <p>Note If you disable (lock) a control, you must provide the information the control collected by configuring MDT properties in CustomSettings.ini or in the MDT DB. Otherwise, the UDI Wizard will not collect the necessary information, and the UDI deployment will fail.</p>

Line Control

This control allows you to add a horizontal line to divide one portion of the custom wizard page from another. This control does not collect any configuration values but rather is used to visually enhance the UI.

Layout Properties

Layout properties are used to configure the UI characteristics of the control and are configured on the **Layout** tab in the UDI Wizard Designer. Table 50 lists the layout properties for the **Line** control and provides a brief description of each property.

Table 50. Line Control Layout Properties

Property	Description
----------	-------------

Property	Description
X	Use this property to configure the horizontal position of the control.
Y	Use this property to configure the vertical position of the control.
Width	Use this property to configure the width of the control.
Height	Use this property to configure the height of the control. Note Increasing this property does not increase the height or width of the line.

Settings Properties

The **Line** control has no settings properties.

Label Control

This control allows you to add descriptive, read-only text to the wizard page. This control does not collect any configuration values but rather is used to visually enhance the UI.

Layout Properties

Layout properties are used to configure the UI characteristics of the control and are configured on the **Layout** tab in the UDI Wizard Designer. Table 51 lists the layout properties for the **Label** control and provides a brief description of each property.

Table 51. Label Control Layout Properties

Property	Description
X	Use this property to configure the horizontal position of the control.
Y	Use this property to configure the vertical position of the control.
Label	Use this property to configure the descriptive text associated with this control.
Width	Use this property to configure the width of the control. Note If the text entered in the Label property is longer than the width of the control, the text is clipped and not displayed.
Height	Use this property to configure the height of the control. Note If the text entered in the Label property is taller than the height of the control, the text is clipped.

Settings Properties

The **Label** control has no settings properties.

Radio Control

This control allows you to select one option from a group of two or more options. As with traditional radio buttons, you can group two or more of these controls; then, the user can select one of the options in the group.

A unique value is assigned to each radio button. The value assigned to the selected radio button control is saved in the task sequence variable configured for this control.

Layout Properties

Layout properties are used to configure the UI characteristics of the control and are configured on the **Layout** tab in the UDI Wizard Designer. Table 52 lists the layout properties for the **Radio** control and provides a brief description of each property.

Table 52. Radio Control Layout Properties

Property	Description
X	Use this property to configure the horizontal position of the control.
Y	Use this property to configure the vertical position of the control.
Label	Use this property to configure the descriptive text associated with the radio button.
Width	Use this property to configure the width of the control. Note If the text entered in the Label property is longer than the width of the control, the text is clipped and not displayed.
Height	Use this property to configure the height of the control. Note If the text entered in the Label property is taller than the height of the control, the text is clipped.
RadioGroup	Use this property to group two or more radio buttons. When radio buttons belong to the same group, only one of the radio buttons within a group can be selected. If you need multiple groups of radio buttons, configure this property for each respective group of radio buttons.
Value	Use this property to configure the value stored in the task sequence variable when the radio button is selected.

Settings Properties

Settings properties are used to configure the data initially shown in a control (the default value) and where the information collected from the user is saved.

Table 53 lists the settings properties for the **Radio** control and provides a brief description of each property.

Table 53. Radio Control Settings Properties

Property	Description
Default value	Use this property to configure the default value for the control. By default, the value is set to the control ID.
Task sequence variable name	Use this property to configure the task sequence variable where the information collected from the user is stored. If the task sequence variable: <ul style="list-style-type: none"> Does not already exist, the task sequence variable is created and set to the value the user provides Already exists, the existing value of the task sequence variable is overwritten with the value the user provides
Friendly display name visible in the summary page	Use this property to configure the descriptive name that appears on the Summary wizard page. This name is used to describe the value that was saved in the Task sequence variable name property for this control.
Unlocked	Use this property to configure whether the user is able to interact with the control. By default, the control is enabled. This button displays the following status: <ul style="list-style-type: none"> Unlocked. The control is enabled, and users can enter information using it. Locked. The control is disabled, and users are unable to enter information using it. <p>Note If you disable (lock) a control, you must provide the information the control collected by configuring MDT properties in CustomSettings.ini or in the MDT DB. Otherwise, the UDI Wizard will not collect the necessary information, and the UDI deployment will fail.</p>

Bitmap Control

This control allows you to add a bitmap graphic (.bmp file) to the custom wizard page. This control does not collect any configuration values but rather is used to visually enhance the UI.

Layout Properties

Layout properties are used to configure the UI characteristics of the control and are configured on the **Layout** tab in the UDI Wizard Designer. Table 54 lists the layout properties for the **Bitmap** control and provides a brief description of each property.

Table 54. Bitmap Control Layout Properties

Property	Description
X	Use this property to configure the horizontal position of the control.
Y	Use this property to configure the vertical position of the control.
Width	Use this property to configure the width of the control. Note If the graphic selected in the Source property is longer than the width of the control, the graphic is clipped.
Height	Use this property to configure the height of the control. Note If the graphic selected in the Source property is taller than the height of the control, the graphic is clipped.
Source	Use this property to configure the fully qualified path to the .bmp file, including the file name. The path to the .bmp file is relative to the location of the UDI Wizard (OSDSetupWizard.exe), which is on one of the following folders (where <i>mdt_toolkit_package</i> is the location of the MDT toolkit package in Configuration Manager): <ul style="list-style-type: none"> • <i>mdt_toolkit_package\Tools\x86</i> • <i>mdt_toolkit_package\Tools\x64</i> To view the graphic when previewing the custom wizard page, the .bmp file must also be located in the following folders (where <i>mdt_install_folder</i> is the folder where you installed MDT): <ul style="list-style-type: none"> • <i>mdt_install_folder\Template\Distribution\Tools\x86</i> • <i>mdt_install_folder\Template\Distribution\Tools\x64</i>

Settings Properties

The **Bitmap** control has no settings properties.

Textbox Control

This control allows you to enter text on the custom wizard page. The text typed into this control is saved in the task sequence variable configured for this control.

Layout Properties

Layout properties are used to configure the UI characteristics of the control and are configured on the **Layout** tab in the UDI Wizard Designer. Table 55 lists the layout properties for the **Textbox** control and provides a brief description of each property.

Table 55. Textbox Control Layout Properties

Property	Description
X	Use this property to configure the horizontal position of the control.
Y	Use this property to configure the vertical position of the control.
Width	Use this property to configure the width of the control. Note If the text entered in the control is longer than the width of the control, the text is clipped and not displayed.
Height	Use this property to configure the height of the control. Note If the text entered in the control is taller than the height of the control, the text is clipped.

Settings Properties

Settings properties are used to configure the data that is initially shown in a control (the default value) and where the information collected from the user is saved. Table 56 lists the settings properties for the **Textbox** control and provides a brief description of each property

Table 56. Textbox Control Settings Properties

Property	Description
Default value	Use this property to configure the default value for the control.
Task sequence variable name	Use this property to configure the task sequence variable where the information collected from the user is stored. If the task sequence variable: <ul style="list-style-type: none"> Does not already exist, the task sequence variable is created and set to the value the user provides Already exists, the existing value of the task sequence variable is overwritten with the value the user provides
Friendly display name visible in the summary page	Use this property to configure the descriptive name that appears on the Summary wizard page. This name is used to describe the value that was saved in the Task sequence variable name property for this control.
List of validators assigned to this control	This property contains a list of validators used to verify that the content entered in the text box. You can: <ul style="list-style-type: none"> Add validators to the list using the blue plus sign button immediately to the right of the list of validators Edit validators in the list using the pencil button immediately to the right of the list of validators

Property	Description
	<ul style="list-style-type: none"> Remove validators from the list using the red X button immediately to the right of the list of validators
Unlocked	<p>Use this property to configure whether the user is able to interact with the control. By default, the control is enabled. This button displays the following status:</p> <ul style="list-style-type: none"> Unlocked. The control is enabled, and users can enter information using it. Locked. The control is disabled, and users are unable to enter information using it. <p>Note If you disable (lock) a control, you must provide the information the control collected by configuring MDT properties in CustomSettings.ini or in the MDT DB. Otherwise, the UDI Wizard will not collect the necessary information, and the UDI deployment will fail.</p>

UDI Task Sequence Variables

The task sequence variables in this section are used only in User-Driven Installation (UDI) deployments. In addition to these task sequence variables, the following ZTI task sequence variables are also used by UDI and are documented in their respective sections earlier in this guide:

- [KeyboardLocale](#)
- [OSDComputerName](#)
- [UILanguage](#)
- [UserLocale](#)

OSDAddAdmin

This task sequence variable specifies a list of domain-based accounts or local accounts to be added to the Administrators local built-in group on the target computer.

Value	Description
<i>domain\account_name1; computer\account_name2</i>	The format of the accounts to be made members of the Administrators group on the target computer in the format of <i>domain\account</i> and separated by semicolons, where <i>domain</i> can be the name of an Active Directory domain or the target computer name.

Example
OSDAddAdmin=domain\user01;Win7-01\Local User01

OSDApplicationList

This task sequence variable specifies which applications should be selected by default on the **Install Software** page of the Operating System Deployment (OSD) Setup Wizard.

Value	Description
<code>app_id1;app_id2</code>	A semicolon-delimited list of application to be selected by default on the Install Software page of the Operating System Deployment (OSD) Setup Wizard; each application is represented by an application ID and separated by a semicolon. The application ID is derived from the Id attribute of each application in the UDIWizard_Config.xml file. In the following excerpt from a UDIWizard_Config.xml file, the 2007 Microsoft Office system with SP2 application has an Id attribute of 1: <code><Application DisplayName="Office 2007 SP2" State="Disabled" Id="1"></code>

Example

<code>OSDApplicationList=2;3</code>

OSDArchitecture

This task sequence variable specifies the processor architecture of the target operating system to be deployed.

Value	Description
<code>x86</code>	The target operating system is a 32-bit operating system.
<code>amd64</code>	The target operating system is a 64-bit operating system.

Example

<code>OSDArchitecture=amd64</code>

OSDBitlockerStatus

This task sequence variable specifies if BitLocker is enabled on the target computer by the BitLocker preflight check.

Value	Description
<code>PROTECTED</code>	The target computer has BitLocker enabled.
<code>Does not exist</code>	If the target computer does not have BitLocker enabled, then the task sequence variable does not

Value	Description
	exist.

Example
None

OSDDiskPart

This task sequence variable specifies whether the target disk partition should be formatted.

Value	Description
TRUE	The target disk partition will be formatted.
FALSE	The target disk partition will not be formatted.

Example
OSDDiskPart=TRUE

OSDDomainName

This task sequence variable specifies the name of the domain to which the target computer will be joined if the computer is configured to be a domain member.

Value	Description
<i>domain_name</i>	<p>The name of the domain to which the target computer will be joined. If you have configured the Computer wizard page in the Operating System Deployment (OSD) Setup Wizard to be Silent, the value in this task sequence variable must match the values specified in the UDI Wizard Designer. Otherwise, the wizard page will be displayed.</p> <p>Note This task sequence variable is only necessary when you are creating a new computer account in the OU. If the computer account already exists, this variable is not needed.</p>

Example
OSDDomainName=domain01

OSDDomainOUName

This task sequence variable specifies the name of the OU in the domain to which the target computer account will be created when the computer joins a domain.

Value	Description
<i>ou_name</i>	The name of the OU in the domain in which the

Value	Description
	<p>computer account will be created</p> <p>Note This task sequence variable is only necessary when you are creating a new computer account in the OU. If the computer account already exists, this variable is not needed.</p>

Example
OSDDomai nOUName=NewDeployOU

OSDImageIndex

This task sequence variable specifies the index number of the target operating system in a WIM file.

Value	Description
<i>index_number</i>	The index number of the target, which starts with an index number of 1 for the first operating system in the WIM file

Example
OSDImageIndex=1

OSDImageName

This task sequence variable specifies the name of the operating system image in the .wim file selected in the **Image Selection** box on the [VolumePage](#) wizard page. The list of possible operating system images in the **Image Selection** box is configured in the **Image Combo Box Values** list in the **Image Combo Box** section on the [VolumePage](#) wizard page editor. The image name is configured as a part of each image in the **Image Combo Box Values** list.

Note This tasks sequence variable is set by the [VolumePage](#) wizard and should not be configured in the CustomSettings.ini file or in the MDT DB. However, this tasks sequence variable can be used to set conditions for task sequence steps, as described in the section, "Configure UDI Task Sequences to Deploy Different Operating Systems", in the MDT document *Using the Microsoft Deployment Toolkit*.

Value	Description
<i>image_name</i>	The name of the operating system image in the .wim file selected in the Image Selection box on the VolumePage wizard page

Example
None

OSDJoinAccount

This task sequence variable specifies the domain-based account used to join the target computer to the domain specified in the **OSDDomainName** task sequence variable. This task sequence variable is necessary if the target computer will be joined to a domain.

Value	Description
<i>account_name</i>	The name of the account used to join the target computer to the domain in the format of <i>domain\account</i>

Example
<code>OSDJoi nAccount=domai n\admi n01</code>

OSDJoinPassword

This task sequence variable specifies the password for the domain-based account used to join the target computer to the domain specified in the **OSDJoinAccount** task sequence variable. This task sequence variable is necessary if the target computer will be joined to a domain.

Value	Description
<i>password</i>	The password of the account used to join the domain

Example
<code>OSDJoi nPassword=P@ssw0rd10</code>

OSDLocalAdminPassword

This task sequence variable specifies the password for the Administrator local built-in account on the target computer.

Value	Description
<i>password</i>	The password of the Administrator local built-in account on the target computer

Example
<code>OSDLocal Admi nPassword=P@ssw0rd10</code>

OSDNetworkJoinType

This task sequence variable specifies whether the target computer joins a domain or a workgroup.

Value	Description
0	The target computer will join a domain.

Value	Description
	If you select this option and configure the corresponding Operating System Deployment (OSD) Setup Wizard page to be Silent , you must also provide values for the OSDJoinAccount , OSDJoinPassword , OSDDomainName , and OSDDomainOUName task sequence variables accordingly. In addition, you must select Domain in Default Selection in the Workspace pane on the Computer Page in the UDI Wizard Designer.
1	The target computer will join a workgroup. If you select this option and configure the corresponding Operating System Deployment (OSD) Setup Wizard page to be Silent , you must also provide a value for the OSDWorkgroupName task sequence variable. In addition, you must select Workgroup in Default Selection in the Workspace pane on the Computer Page in the UDI Wizard Designer.

Example

```
OSDNetworkJoinType=0
```

OSDSetupWizCancelled

This task sequence variable specifies if the user cancelled the Operating System Deployment (OSD) Setup Wizard.

Value	Description
TRUE	The user cancelled the Operating System Deployment (OSD) Setup Wizard.
Does not exist	If the wizard is not cancelled, then the task sequence variable does not exist.

Example

```
None
```

OSDTargtDrive

This task sequence variable specifies the disk volume where the target operating system will be deployed.

Value	Description
<i>disk_volume</i>	The disk volume designation

Example

OSDTargetDrive=C:

OSDWinPEWinDir

This task sequence variable specifies the folder in which the Windows operating system is currently installed on the target computer.

Value	Description
<i>windows_directory</i>	The directory in which the Windows operating system is currently installed

Example

OSDWinPEWinDir=C:\Windows

OSDWorkgroupName

This task sequence variable specifies the name of the workgroup to which the target computer will be joined if the computer is configured to be a workgroup member.

Value	Description
<i>workgroup_name</i>	The name of the workgroup to which the target computer will be joined

Example

OSDWorkgroupName=WORKGROUP01

OSDResults.exe.config File Element Values

The OSD Results program, OSDResults.exe, is run at the end of a UDI deployment and displays the results of the deployment process. The behavior of the OSD Results program can be customized by modifying the OSDResults.exe.config file element values. The OSDResults.exe.config file is stored in Tools\OSDResults in the MDT Package in the User Drive Installation Task Sequence.

backgroundOpacity

This XML element configures the opaqueness of the background wallpaper image specified as a decimal-formatted percentage in the **backgroundWallpaper** element.

Value	Description
<i>opacity_percent</i>	The percentage of opaqueness of the backgroundWallpaper element specified in a decimal formatted percentage—for example, a value

Value	Description
	of 0.8 designates 80% opaqueness.

Example

```
<add key="backgroundOpacity" value="0.8" />
```

backgroundWallpaper

This XML element provides the file name and relative path to the image that is displayed as the background in the **OSD Results** dialog box. The path is relative to the Tools\OSDResults folder in the MDT Package.

Value	Description
<i>path\file_name</i>	Includes the relative path and file name of the background image; the path is delimited with double forward slashes (//).

Example

```
<add key="backgroundWallpaper" value="images\\Wallpaper.jpg" />
```

completedText

This XML element provides the text that is displayed in the **OSD Results** dialog box when the deployment is complete.

Value	Description
<i>text</i>	The text to be displayed in the OSD Results dialog box in quotation marks when deployment is complete

Example

```
<add key="completedText" value="Deployment Complete" />
```

headerImagePath

This XML element provides the file name and relative path to the image that is displayed in the header of the **OSD Results** dialog box. The path is relative to the Tools\OSDResults folder in the MDT Package.

Value	Description
<i>path\file_name</i>	Includes the relative path and file name of the header image; the path is delimited with double backslashes (\\\).

Example

```
<add key="headerImagePath" value="images\\Windows7_h_rgb.png" />
```

timeoutMinutes

This XML element configures how many minutes the **OSD Result** dialog box is displayed before the dialog box is automatically closed and the computer is restarted.

Value	Description
<i>Non-numeric value</i>	The dialog box remains open until Start Windows is clicked.
<i>Negative value</i>	The dialog box remains open until Start Windows is clicked.
0	The dialog box remains open until Start Windows is clicked.
Include decimal point	The dialog box remains open until Start Windows is clicked.
1 - 10080	The number of minutes the dialog box will be displayed, with a minimum value of 1 minute and a maximum value of 10080 minutes (1 week).

Example

```
<add key="timeoutMinutes" value="30"/>
```

welcomeText

This XML element provides the welcome text that is displayed in the **OSD Results** dialog box.

Value	Description
<i>welcome_text</i>	The welcome text to be displayed in the OSD Results dialog box in quotation marks

Example

```
<add key="welcomeText" value="Congratulations, Windows 7 has been successfully deployed to your computer." />
```