

PDF Converter Services Installation & Administration Guide

Muhimbi Ltd

Version 10.3

Document Control

Draft	Author	Date	Comment
3.0 – 10.2	Muhimbi	12/11/2009 - 07/11/2021	Historical versions
10.2	Muhimbi	31/03/2022	Updated for version 10.2
10.2.1	Muhimbi	08/03/2023	Updated for version 10.2.1
10.3	Stephen Carter	14/06/2023	Updated for version 10.3

Purpose and audience of document

This document describes the installation steps as well as general administrative topics related to the Muhimbi PDF Converter Services.

The intended audience is anyone involved in the installation and administration of this solution.

Disclaimer

© Muhimbi. All rights reserved. No part of this document may be altered, reproduced or distributed in any form without the expressed written permission of Muhimbi.

This document was created strictly for information purposes. No guarantee, contractual specification or condition shall be derived from this document unless agreed to in writing. Muhimbi reserves the right to make changes in the products and services described in this document at any time without notice and this document does not represent a commitment on the part of Muhimbi in the future.

While Muhimbi uses reasonable efforts to ensure that the information and materials contained in this document are current and accurate, Muhimbi makes no representations or warranties as to the accuracy, reliability or completeness of the information, text, graphics, or other items contained in the document. Muhimbi expressly disclaims liability for any errors or omissions in the materials contained in the document and would welcome feedback as to any possible errors or inaccuracies contained herein.

Muhimbi shall not be liable for any special, indirect, incidental, or consequential damages, including without limitation, lost revenues or lost profits, which may result from the use of these materials. All offers are non-binding and without obligation unless agreed to in writing.

Contents

1	Introduction	5
1.1	High level solution architecture	5
1.2	Prerequisites	6
2	Deployment	7
2.1	Before you begin	7
2.2	Installing Prerequisites & Dependencies	8
2.2.1	.Net framework 4	8
2.2.3	Ghostscript	10
2.3	Installation steps	11
2.4	Installation Verification	18
2.5	Installing the License	19
2.6	Tuning the Document Conversion service	20
2.6.1	Authentication (Generic)	20
2.6.2	Authentication (from SharePoint)	21
2.6.3	Concurrency	21
2.6.4	Timeouts and File Size limitations	23
2.6.5	Logging	23
2.6.6	Adding custom converters / changing file extensions	24
2.6.7	Exception handling	25
2.6.8	Regional settings	25
2.6.9	InfoPath specific switches	25
2.6.10	HTML specific switches	27
2.6.11	Word processing (MS-Word) specific switches	28
2.6.12	Spreadsheets (Excel) specific switches	30
2.6.13	Presentations (PowerPoint) specific switches	30
2.6.14	AutoCAD specific switches	31
2.6.15	MSG & EML (email) specific switches	32
2.6.16	Switches used for overriding settings	34
2.6.17	PDF & Security Settings	36
2.7	Hardening the Conversion Service	37
2.8	Uninstalling	37
2.9	Upgrading from a previous version	38
3	Troubleshooting & Other common tasks	39
3.1	Windows Event Log	39
3.2	Trace Log	39
3.3	Common issues & Errors	39
3.3.1	Error messages related to printer drivers or the printer spooler are logged	39
3.3.2	Problems parsing the WSDL	39
3.3.3	Documents using non standard fonts (e.g. Japanese) are not converted properly / The fonts in the destination document are not correct	40
3.3.4	Problems converting InfoPath forms without a shared XSN file	40
3.3.5	InfoPath forms using Ink controls fail to convert	40
3.3.6	Error 403 (Forbidden) when converting InfoPath forms	41
3.3.7	InfoPath files are converted using an old version of the XSN template	41
	Appendix - Using InfoPath with External Data Sources	42
	Details for InfoPath 2007	42
	Details for InfoPath 2010 & 2013	44
	Digitally signing forms	44
	Using Muhimbi's 'AutoTrustForms' feature	44
	Appendix – Switching between InfoPath Converters	46

Enabling the high fidelity InfoPath Converter	46
Enabling the legacy InfoPath Converter	46
Appendix - Post processing PDF output to PDF/A	47
Appendix - Unattended (un)installation	49
Installation	49
Uninstallation	50
Upgrading	50
Appendix - Advanced Deployment Scenarios	52
Appendix - Creating Custom Converters	56
Appendix - Invoke 3 rd party Converters	62
Appendix - Relevant articles on the Muhimbi Blog	64
Appendix - Licensing	66

1 Introduction

This document describes the installation steps as well as general administrative topics related to the Muhimbi PDF Converter Services.

The intended audience is anyone involved in the installation and administration of this solution. It is assumed that the audience has some familiarity with installing services on the Windows platforms and have been given the privileges to install and deploy solutions.

For more details about this product please see:

1. Product Information:
<https://www.muhimbi.com/Products/PDF-Converter-Services>
2. Product Overview:
<https://www.muhimbi.com/guides/pdf-converter/pdf-converter-services>
3. Knowledge Base / Frequently Asked Questions:
<https://www.muhimbi.com/guides/pdf-converter-services/knowledge-base>
4. Release Notes:
<https://www.muhimbi.com/guides/pdf-converter-services/release-notes/>
5. User & Development Guide:
<https://www.muhimbi.com/Products/PDF-Converter-Services/Documentation>
6. PDF Converter Service-related content on the Muhimbi Blog:
<https://www.muhimbi.com/blog/tags/pdf-converter/>

To keep on top of the latest news and releases, please subscribe to our blog or twitter feed at <https://www.muhimbi.com/contact>.

1.1 High level solution architecture

The *Muhimbi PDF Converter Services* is a highly optimised solution to programmatically convert, merge, watermark, secure and OCR documents created in typical MS-Office applications, as well as other formats such as InfoPath, HTML, MSG (email), AutoCAD and images, to PDF or XPS format using any Web Services based environment including Java and .NET.

The converter runs as a Windows Service that can be deployed to either a separate system / virtual machine or to the server hosting your own application.

Although the actual converter must be installed on a Windows based environment, it can be invoked from any platform that supports Web Service calls including Windows, Linux, Solaris, AIX and Mac OS X.

Conversions can be *scaled up* by running multiple conversions in parallel and *scaled out* using standard HTTP Load balancers. For details see *Appendix - Advanced Deployment Scenarios*.

.

To achieve optimal conversion quality, for some file formats the Conversion Service uses MS-Office's own libraries in the background to carry out the actual conversion. Muhimbi's software stack ensures that this happens in a

robust, reliable, and scalable manner without taking up excessive system resources. Common and uncommon problems are detected, and corrective action is taken automatically without requiring any attention from system administrators.

1.2 Prerequisites

The solution has been designed to work on a wide number of platforms. The prerequisites are as follows:

Server O.S.	Windows Server 2008 R2 Windows Server 2012 (including R2) Windows Server 2016 Windows Server 2019 Windows Server 2022
Client O.S.	The product is OS agnostic
Supported Languages	Any language that supports Web Services including .NET, Java, PHP and Ruby.
Office Version	Office 2007 (SP2) / 2010 / 2013 / 2016 applications for the relevant converters ¹ .
.NET Framework	Version 4 (on the server, any version on clients)
System Memory	This depends on the size and complexity of the documents that are converted and the number of concurrent conversions taking place. We recommend a minimum of 1GB of total memory.
CPU	Any CPU that can comfortably run the selected Operating System will be suitable. We recommend one or more multi-core CPUs.
Disk Space	This product requires 400MB of space

¹ Please do not use Office 2019, it is not considered stable for server use. Office 2016 supports all Office 2019 documents just fine.

2 Deployment

Unless your solution will be installed in a single server environment, it is worth reading 'Appendix - Advanced Deployment Scenarios' first.

Please note that, unless specified otherwise, installation instructions are the same for all Windows versions. When upgrading from a previous version of the Muhimbi PDF Converter, please follow the instructions in section 0.

If you are experiencing any problems then please check out chapter 3 – 'Troubleshooting & Other common tasks' or contact support@muhimbi.com.

Please do not skim over the information and instructions in this chapter. Installation is generally very easy, but it is essential that you follow the correct steps.

2.1 Before you begin

Before starting the deployment process, please make sure you have access to the following:

1. A user account with the appropriate privileges to deploy Windows software.
2. A user account to run the Conversion Service under, with the following attributes:
 - a. Use a real account, **do not** use built-in Windows accounts such as *Local System* or *Local Service*.
 - b. Local administrator on the server the Conversion Service will be installed on.
3. For Production deployments, license file(s) for the Muhimbi PDF Converter. A license file is not needed to deploy the evaluation version.
4. If there is a requirement to convert MS-Office formats such as MS-Word, Excel and InfoPath then please make sure you have access to a full MS-Office installation set (**not the Office 365 / Click-to-run versions**).
5. If the server the deployment is carried out on does not have an active internet connection AND you require PDF/A output, or conversion of InfoPath files, then please download [Ghostscript 10.01.0](#) (**unless instructed to do so by our support desk, do not install any other version**). If your server is connected to the internet then it will be downloaded automatically.

2.2 Installing Prerequisites & Dependencies

The PDF Converter is a comprehensive solution that includes a large number of different features. Some functionality depends on third party software that must be installed on the server running the Muhimbi Conversion Service.

If your organisation is using FIPS, please make sure it is disabled on the server running the Muhimbi Conversion Service. This can be achieved using the “System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing” Local Security Policy. Microsoft is no longer recommending FIPS mode as it breaks perfectly valid and secure .net framework encryption as used by the Muhimbi PDF Converter.

2.2.1 .Net framework 4

To maximise compatibility with old as well as new systems, the Conversion Service has been built on version 4.0 of Microsoft’s .net framework. The installer will automatically detect, and if needed download, this framework on Windows Server 2008R2.

When deploying the Conversion Service on Windows Server 2012 or newer then deploy version 4.0 (or later) of the .net framework using Windows’ Server Manager, or using the following command.

```
Dism /online /Enable-Feature /FeatureName:NetFx4 /All
```

2.2.2 MS-Office

To ensure the quality of converted documents is 100% perfect, some converters have a dependency on external applications such as MS-Office. If it is a requirement to convert these file types then please install the relevant MS-Office applications, but only on the server that runs the Muhimbi Conversion Service.

The supported file formats, and their dependencies, are as follows, please install the dependencies for the file formats needed in your environment.

Converter	Supported file types	Dependency
HTML & Web pages	html, htm, mht and any url that returns HTML such as .aspx or .jsp.	-
Image formats	gif, png, jpg, bmp, tif, tiff	-
AutoCAD formats	dwg, dxf	-
InfoPath forms	xml, infopathxml	InfoPath
Word Processing	doc, docx, docm, dot, rtf, txt, wps, xml, odt, ott, wpd	MS-Word
Emails	msg, eml	MS-Word
Spreadsheets	xls, xlsx, xslm, xlsb, xml, csv, dif, ods, ots	Excel
Presentations	ppt, pptx, pptm, xml, odp, otp, pps, ppsx, ppsm	PowerPoint
Publisher	pub	Publisher
Vector formats	vsd, vdx, vdw, svg, svgz,	Visio
Postscript	ps, eps	Ghostscript

At the time of writing, it is recommended to use MS-Office 2016, taking the following into account:

1. Remove older MS-Office versions from the conversion server as environments with mixed versions on the same system are not supported, even though it may work.
2. The minimum supported MS-Office version is Office 2007 SP2.
3. **If it is a requirement to convert InfoPath forms, then you must install the 64-bit version of InfoPath when using 64 bit versions of Windows.**
4. Do not install the *click-to-run* or *Office 365* editions of MS-Office. Those versions are not compatible; the full version of MS-Office will need to be installed.
5. Do not install trial or non-activated versions of MS-Office, they are not compatible.
6. Although Office 2016 works fine in combination with the Muhimbi PDF Converter, please take the following into account:
 - a. This Office version no longer ships with InfoPath. If InfoPath support is important then Install InfoPath 2013 (64 bit) separately.
 - b. Most Office 2016 installers are 'click-to-run' based. In order to use this Office version on the server, please install the full 'Professional Plus' version (available from the *Microsoft Volume Licensing Center*).
7. Due to the frequent changes made by Microsoft, we do not recommend installing Office 2019 on the server. Office 2016 and earlier versions can still process documents created in Office 2019.

Once MS-Office has been installed, please carry out the following steps:

1. Log in to the desktop of the server running the conversion service using the account the conversion service runs under or will run under.
2. Launch the various MS-Office applications and carry out Microsoft's activation process for the current user.
3. Close the MS-Office applications again.

It is essential to execute these steps, even if Office has already been activated.

2.2.3 Ghostscript

The Muhimbi PDF Converter relies on Ghostscript for a small subset of functionality. When installing the Muhimbi Conversion Service on a computer that does not have an active internet connection, which is not uncommon in data centers, AND you expect to use any of the features listed below, then you will need to deploy Ghostscript manually. If the server does have an active internet connection, then the Muhimbi installer will download and install Ghostscript automatically.

1. PDF/A output, a subset of the PDF standard intended for long term archiving.
2. High fidelity InfoPath conversions.
3. Conversion of PostScript files to PDF

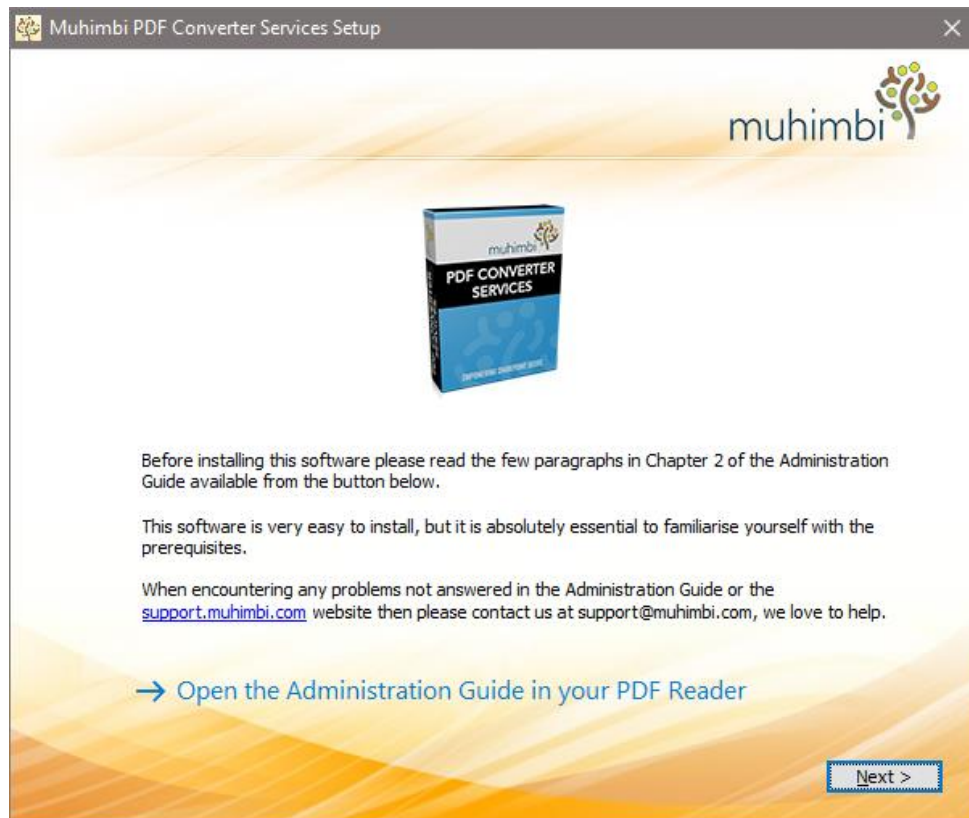
If you require any of these features, and your server does not have an active internet connection, then deploy Ghostscript manually, preferably before running the Muhimbi installer:

1. Download [Ghostscript 10.01.0](#). Unless instructed by the Muhimbi support desk, do not install any other version.
2. Run the Ghostscript installer and deploy it to the default path (select a different drive letter if needed, but leave the rest of the path unchanged)
3. Accept all default options.

2.3 Installation steps

The installation steps are as follows. For details about carrying a silent installation, see *Appendix - Unattended (un)installation*.

1. Log in to the desktop of the server that will be used to run the Conversion Service and launch setup.exe. This results in the following screen from where you can open the Administration Guide, *this document*.

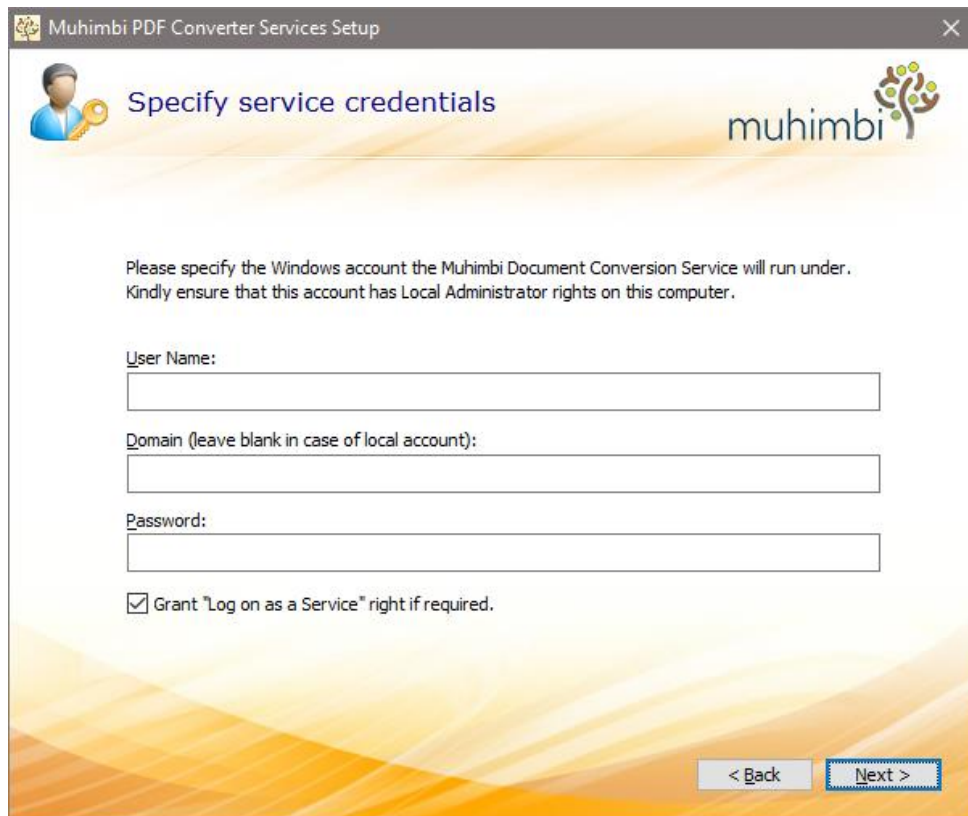


2. Click *Next*, read and accept the license agreement and click *Next* again.
3. Accept or change the default installation folder. Click *Next* to continue.

- Specify the details of the Windows account the conversion service will run under.

This may be a local machine account (in that case leave the domain name empty) or a domain account (please enter the domain in the separate field). Please make sure this account matches the exact requirements specified in section 2.1 *Before you begin*.

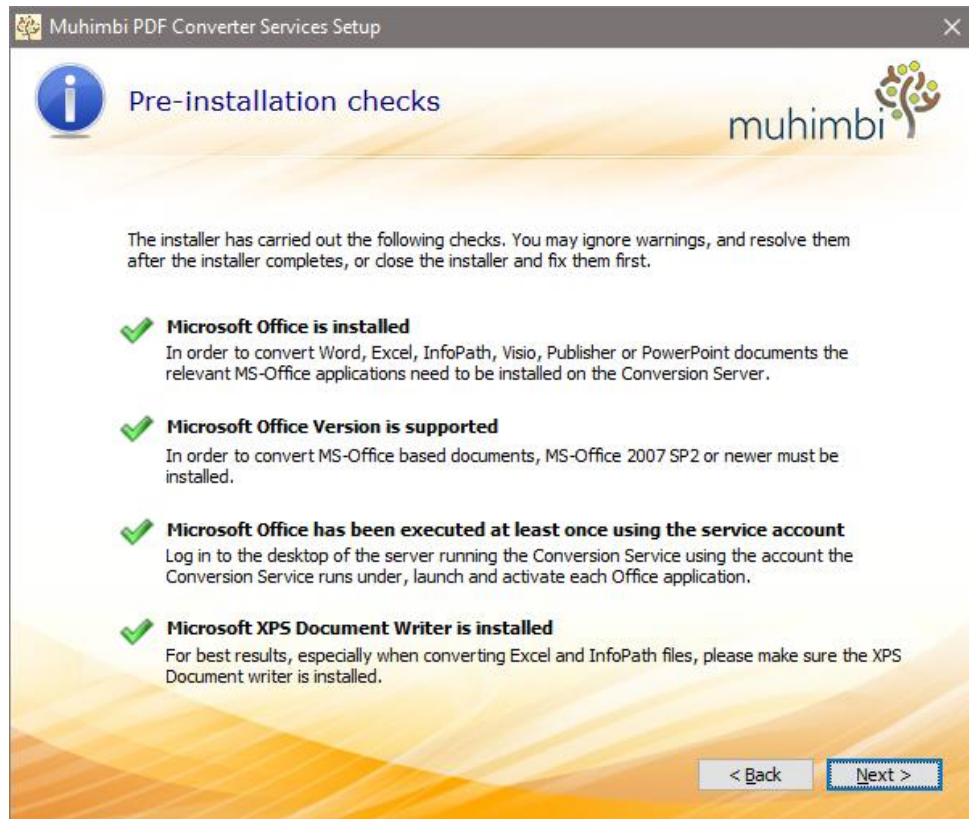
Unless specified otherwise, the account will be granted 'Log on as a Service' rights automatically.



The screenshot shows a Windows-style dialog box titled "Muhimbi PDF Converter Services Setup". The main heading is "Specify service credentials" with a user icon and a key icon. The Muhimbi logo is in the top right. The text reads: "Please specify the Windows account the Muhimbi Document Conversion Service will run under. Kindly ensure that this account has Local Administrator rights on this computer." There are three input fields: "User Name:", "Domain (leave blank in case of local account):", and "Password:". A checkbox is checked and labeled "Grant 'Log on as a Service' right if required." At the bottom right, there are "< Back" and "Next >" buttons.

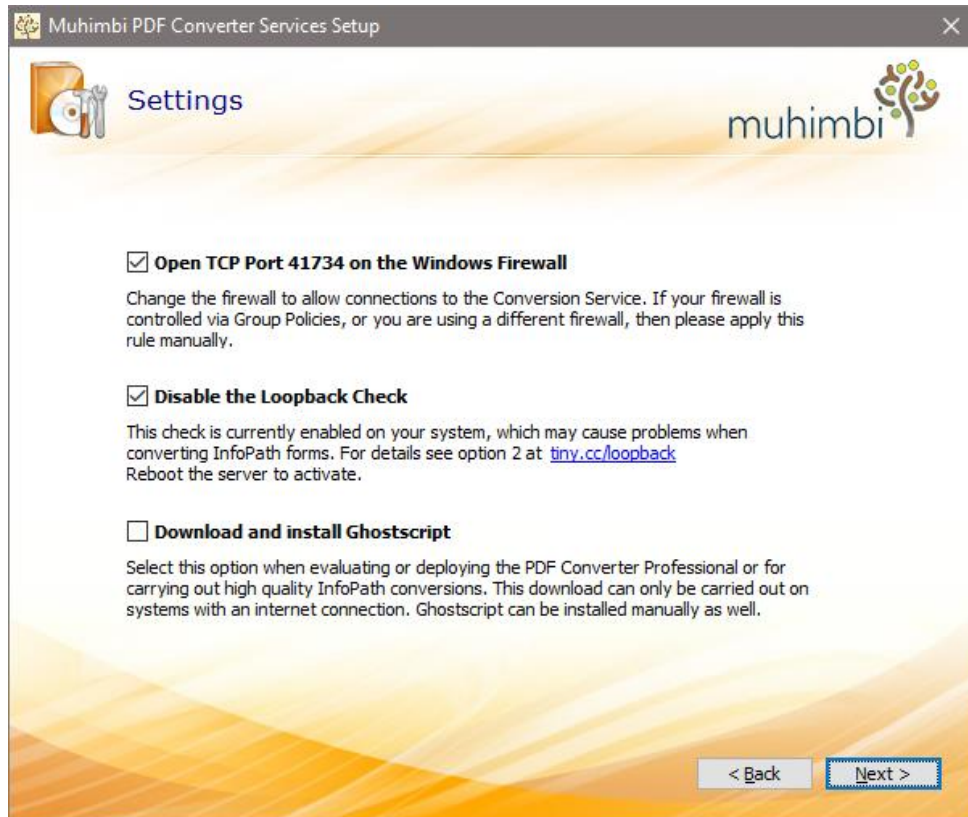
Click *Next* to validate the specified credentials and proceed to the next screen.

- The installer will carry out a number of checks to validate that the environment the software is installed on matches the prerequisites specified in sections 2.1 - 2.2, and present the results on the next screen.



If all checks pass, and show a green tick mark, then continue to the next screen. However, if any of the checks fail then you have the option to continue. This may be suitable for situations where there is no need to carry out the conversion of any MS-Office related file types, in which case you may ignore the MS-Office related validation failures.

6. The Settings screen is displayed next.



- a. **Open TCP Port 41734 on the Windows Firewall:** In a multi-server farm it is essential that the front-end servers can communicate with the conversion server. The installer can open the correct port automatically, but please take into account that this only works for a basic Windows Firewall installation. When deploying the software in an environment with a different firewall or when the firewall is automatically configured and locked down, you may need to open this port manually.²
- b. **Disable the Loopback Check:** In certain cases, a security feature in Windows makes it difficult to connect to a server by machine name, this is known as the *Loopback Check*. Providing this feature is enabled (the default in Windows) you can use the installer to disable it. If this feature is already disabled (perhaps by an administrator or another process) then you will not be able to change it using the Muhimbi installer. This prevents interoperability problems with other software. If you wish to change this setting by hand then see [this blog post](#).

² The port is only opened on the conversion server. If remote servers do not allow outgoing connections on port 41734 then the firewall rules on those servers will need to be adjusted manually.

- c. **Download and install Ghostscript:** If you require any of the features listed under 0 AND the server the installer is being executed on has an active internet connection, then enable this option to automatically download and install Ghostscript. If the server is not connected to the internet, then follow the steps in [Installing Prerequisites & Dependences - Ghostscript](#).

As of Ghostscript 10.1, there is no “silent install”, so the installation process requires manual intervention, please follow the on-screen instructions.

7. The InfoPath configuration screen is displayed next.

Unless the PDF Converter is being upgraded from a pre-8.0 version, and InfoPath conversions are already working perfectly, it is strongly recommended to accept the default values and enable the new *High quality InfoPath converter*³.

If the PDF Converter will be used to convert InfoPath forms, and the High Quality InfoPath converter is enabled, then please make sure Ghostscript is enabled as well. For details see point ‘c’ on the previous page.



Please keep in mind that on 64-bit systems the new InfoPath converter only works in combination with the 64-bit version of InfoPath.

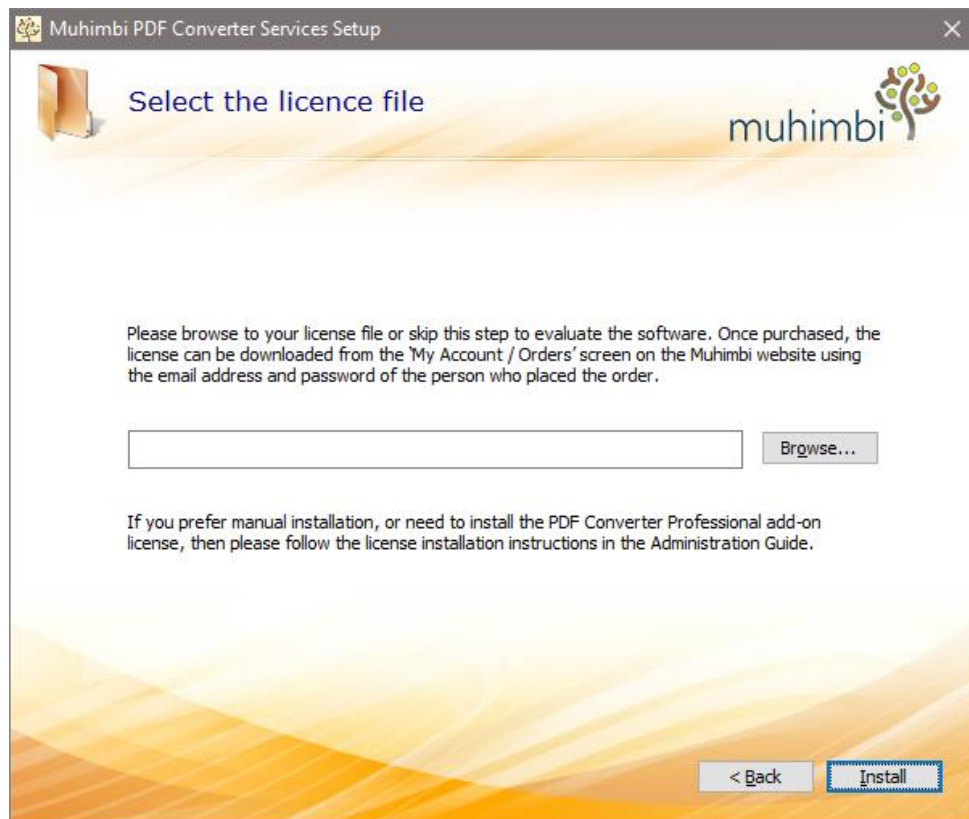
Click *Next* to continue.

³ For details about how to switch between InfoPath converters at any time, see *Appendix – Switching between InfoPath Converters*

8. When deploying the PDF Converter to a Production or Disaster Recovery environment, use this screen to specify the location of the license file. Doing so is optional, when the license key is not specified the software will automatically run in trial mode.

To install the license at a later time - e.g. to activate a previously installed trial version, or when installing the PDF Converter Professional - please follow the instructions in [Appendix - Licensing](#)

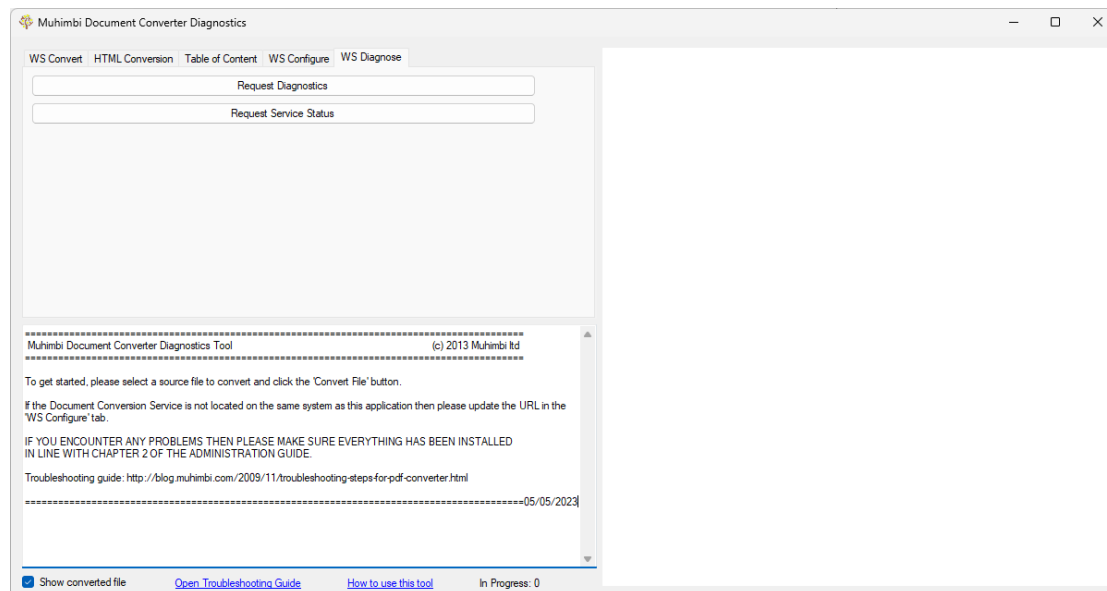
Installing the License.



Click *Next* to complete the installation process.

2.4 Installation Verification

To verify that the software and all prerequisites have been setup and configured correctly, execute the Diagnostic tool located in the Start Menu.



The Diagnostics Tool exposes the following functionality:

- **WS Convert:** Individual files as well as folders can be converted from this tab. Although the default settings are sufficient for most conversions, feel free to play around with the various settings.
- **HTML Conversion:** Convert a URL or HTML fragment to PDF format.
- **WS Configure:** Request configuration data detailing what converters are available and which file extensions they know how to process. The location of the Web Service, as used by the other tabs, can be configured on this tab as well.
- **WS Diagnose:** The button available on this screen invokes each of the individual converters using an internal test file to check if everything has been installed correctly.

Clicking the button on the *WS Diagnose* tab verifies if the installation has been carried out correctly.

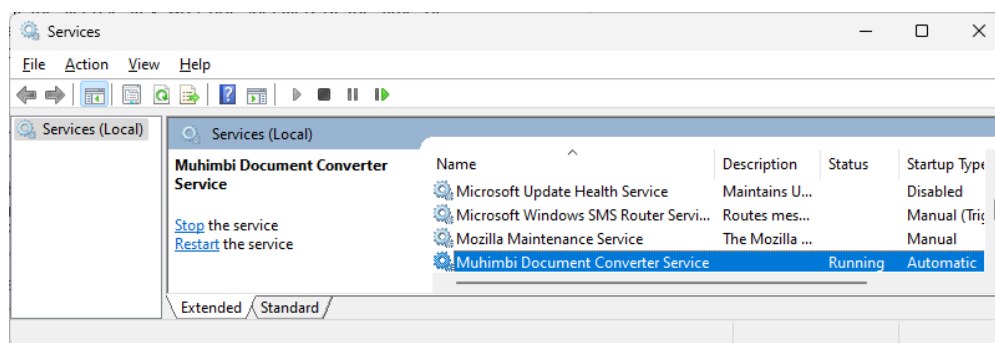
Note that the *Diagnostics Tool* is not used to configure the Conversion Service. It is merely a test harness that can be used to test if the system has been installed correctly. The full source code of this tool is included and can be accessed from the Start Menu.

For more details about the usage of this tool see this [Knowledge Base article](#).

2.5 Installing the License

The PDF Converter's installer provides the option to install the license key automatically. However, if the license key was not specified at the time of installation, or when installing the PDF Converter Professional add-on license, then please follow the procedure below.

- Copy the license file to the directory where the Conversion Service has been installed in. A shortcut to this folder (*Open Installation Folder*) can be found in the Windows Start Menu.
- If you have purchased a license for the *PDF Converter Professional* as well then please make sure that license file is added using the same steps. Please note that a *PDF Converter Professional* license cannot be used unless a license for the *PDF Converter for Services* is also in place.
- After copying the license file, restart the Conversion Service using the *Windows Services Management Console* (services.msc). Select the service, right click and select restart.



Alternatively restart it using the following command lines:

```
Net stop "Muhimbi Document Converter Service"  
Net start "Muhimbi Document Converter Service"
```

2.6 Tuning the Document Conversion service

The settings for the Document Conversion Service can be changed by editing the *Muhimbi.DocumentConverter.Service.exe.config* file located in the directory the Conversion Service has been installed in⁴.

The various settings that can be changed are described below. Note that the Service must be restarted after making changes to the configuration file. Use the *Windows Services* MMC or the command line to do this:

```
Net stop "Muhimbi Document Converter Service"  
Net start "Muhimbi Document Converter Service"
```

Please note that some additional settings related to the post processing of PDF/A files can be found in Appendix - Post processing PDF output to PDF/A.

2.6.1 Authentication (Generic)

To make the initial installation as simple as possible, particularly for environments that access the Conversion Service from non-Windows based platforms, anonymous access is enabled by default.

Although in general Production environments are shielded by a firewall, depending on your organisation you may want to enable an extra layer of authentication.

Authentication and Authorization are controlled by the following attributes and elements in the *Config* file:

- **ConversionClientsGroup:** The name of the Windows group that contains the accounts that are allowed to carry out conversions.
- **ConversionAdministratorsGroup:** The name of the Windows group that contains the accounts that can execute typical Administrative tasks such as running diagnostics.
- **Security mode:** Either use *TransportCredentialOnly* or *None*.
- **ClientCredentialType:** The type of credential used for client authentication. Either use *Windows* or *None*.

The Document Conversion Service uses Microsoft's Windows Communication Foundation (WCF) framework. For further details about configuring security have a look at Microsoft's MSDN site at <http://msdn.microsoft.com/en-us/library/ms731925.aspx>.

⁴ The Start menu contains a shortcut to this folder.

The following table contains a number of common scenarios:

	Conversion Clients Group	Conversion Administrators Group	Security mode	Client Credential Type
Anonymous	<leave empty>	<leave empty>	"None"	"None"
SharePoint	wss_wpg	wss_admin_wpg	TransportCredentialOnly	Windows

2.6.2 Authentication (from SharePoint)

If you intend to use the Document Conversion Service from a SharePoint environment, then it is recommended to configure security as per the table in section 2.6.1.

This will restrict use to members of the standard SharePoint *wss_wpg* and *wss_admin_wpg* groups. These groups, however, are local to the SharePoint machine, which may cause problems if the Document Conversion Service is installed on a separate system that does not have these local groups.

The solution is to either manually create these groups on the server hosting the Document Converter, and populate them with the same users as on the SharePoint servers, or to change the name of the groups in the config file.

The group names are defined in the following 2 configurations keys:

- **ConversionClientsGroup:** For SharePoint set this to *wss_wpg*
- **ConversionAdministratorsGroup:** For SharePoint set this to *wss_admin_wpg*

If there is no need to restrict access to the back end of the Document Converter Service then you may want to consider changing the group names to '*NT AUTHORITY\authenticated users*'.

The *authenticated users* group allows any user with a valid login account to connect. Note that this is not the same as anonymous access.

Keep in mind that SharePoint connects to the Document Conversion service using the Web Application's *application pool* account, not the user's account.

2.6.3 Concurrency

The Document Converter allows multiple operations to be processed simultaneously. The default settings are sufficient for most situations, but if you are running the service on a standalone server or if you expect the majority of your conversions to be for a single specific format then you may want to tune the concurrency settings.

The following settings can be changed in the config file.

- **serviceThrottling / maxConcurrentCalls:** This setting represents the maximum number of concurrent requests that can be executed across all operations before new requests are queued.

Please note that this number includes any requests for applying watermarks or Security on documents that are already in PDF format and don't require conversion.

- **MuhimbiDocumentConverters / WordProcessing / maxInstances:** The maximum number of concurrent MS-Word conversion requests before new requests are queued. This value defaults to 2.
- **MuhimbiDocumentConverters / SpreadSheets / maxInstances:** The maximum number of concurrent Excel conversion requests before new requests are queued. This value defaults to 2.
- **MuhimbiDocumentConverters / Vector / maxInstances:** The maximum number of concurrent Visio / Vector conversion requests before new requests are queued. This value defaults to 2.
- **MuhimbiDocumentConverters / CAD / maxInstances:** The maximum number of concurrent AutoCAD conversion requests before new requests are queued. This value defaults to 2.
- **MuhimbiDocumentConverters / TIFF / maxInstances:** This option is not available as it relies on the **MuhimbiDocumentConverters / Image / maxInstances** settings.
- **MuhimbiDocumentConverters / Image / maxInstances:** The maximum number of concurrent image conversion requests (including the separately configured TIFF converter) before new requests are queued. This value defaults to 2.
- **MuhimbiDocumentConverters / MSG / maxInstances:** The maximum number of concurrent MSG & EML (email) conversion requests before new requests are queued. This value defaults to 6.
- **MuhimbiDocumentConverters / Presentations / maxInstances:** Do **not** change this value as PowerPoint does not allow concurrent requests.
- **MuhimbiDocumentConverters / MicrosoftPublisher / maxInstances:** The maximum number of concurrent Microsoft Publisher conversion requests before new requests are queued. This value defaults to 2.
- **MuhimbiDocumentConverters / InfoPath / maxInstances:** Do **not** change this value as InfoPath does not allow concurrent requests.
- **MuhimbiDocumentConverters / HTML / maxInstances:** The maximum number of concurrent HTML conversion requests before new requests are queued. This value defaults to 2.
- **MuhimbiDocumentConverters / Postscript / maxInstances:** The maximum number of concurrent Postscript (.ps) conversion requests before new requests are queued. This value defaults to 2.
- **MuhimbiDocumentConverters / CommandLineConverter / maxInstances:** Maximum number of concurrent requests (of all combined command line converters, if any) before new requests are queued. This value defaults to 2.
- **MuhimbiOCRProcessors / Muhimbi / maxInstances:** The maximum number of concurrent OCR operations before new requests are queued.

2.6.4 Timeouts and File Size limitations

To prevent users from sending overly complex documents to the conversion service and blocking access for other users for a long amount of time, it is possible to restrict the maximum time a conversion process is allowed to run. The following settings can be changed to deal with long running conversions:

- **Maximum run duration:** By default individual conversion requests are not allowed to run for more than 10 minutes. This should be sufficient for even the most complex documents. However, this value can be changed as follows:
 - Change the global value of *MuhimbiOperationTypes / operationTypes / maxRunDuration*, or override this value for individual operation types in the same config section. This allows, for example, OCR to be set to a longer timeout (Defaults to 1 hour).
 - Change the value of *receiveTimeout* to the same amount.
- **Maximum file size:** By default, the maximum size of a source file is 100MB. This value can be changed using the following settings:
 - *maxBufferSize*: Specify the new maximum file size in bytes.
 - *maxReceivedMessageSize*: Enter the same value here.
 - *maxArrayLength*: Enter the same value here.
 - *maxStringContentLength*: Enter the same value here.

2.6.5 Logging

The Document Conversion Service uses the industry standard *log4net* framework to write logging and trace data to a log file. Out-of-the-box information is logged to the `Logs\DocumentConverter.log` file stored in the directory the Document Conversion service has been installed in. A new file is created for each day and the default logging level is set to 'INFO'.

Warnings and Errors are also written to the Windows Event Log.

You may want to consider changing the following settings:

- **Log file location:** change the path of the log file name in the *appender* element to a location of your preference.
- **Log Level:** By default only 'INFO' and critical events are logged. To get a better view of what the service is doing, e.g., during a troubleshooting session, you may want to consider switching the `<root>` log level to DEBUG mode.

More information can be found at <http://logging.apache.org/log4net/index.html>.

2.6.6 Adding custom converters / changing file extensions

The Conversion Service makes it possible to add new converters as well as change the file extensions managed by each converter. This makes it possible to add new converters that are not shipped with the product as well as control which file types each converter deals with.

This information is stored in the *MuhimbiDocumentConverters* element in the service's config file, for example:

```
<add key="WordProcessing"
      description="Word Processing (e.g. MS-Word, RTF, TXT)"
      fidelity="Full"
      maxInstances="2"
      supportedExtensions="doc,docx,docm,rtf,txt,wps,xml,eml,odt"
      supportedOutputFormats="xps,pdf,doc,docx,rtf,txt,html,mht,xml,odt"
      type="Muhimbi.DocumentConverter.WebService.WordProcessorConverterFull
          Fidelity, Muhimbi.DocumentConverter.WebService, Version=1.0.1.1,
          Culture=neutral, PublicKeyToken=c9db4759c9eaaad12" />
```

Each converter has the following attributes:

- **key:** The name of the converter. Do not change this for existing converters unless absolutely needed as client side applications may store information against this key.
- **description:** An optional, human readable, description of the converter. This value may be used by client-side application to display details about the converter.
- **fidelity:** The system has the notion of Full Fidelity and High-Fidelity converters. When programming against the Web Services interface it is possible to select which fidelity to use.

This makes it possible to have 2 different converters that deal with the same file type. For example, MS-Word can be used to convert complex documents (Full Fidelity) whereas a streamlined - high performance - third party component can be used to convert simple files at very high speed (High Fidelity).

- **maxInstances:** The maximum number of concurrent instances allowed. This value is optional
- **supportedExtensions:** A comma separated list of file extensions that are recognised by the converter.
- **supportedOutputFormats:** A comma separated list of file types that the converter can generate.
- **type:** The full .NET type of the converter.

For details about how to create custom converters see *Appendix - Creating Custom Converters*.

2.6.7 Exception handling

By default exceptions that occur in the Conversion Service are passed to the calling client application including a full stack trace. If this is not desired then set the *includeStackTraceInFaultReason* key in the config file to *false*.

As a result Exceptions will still be thrown, but the full stack trace will not be included.

2.6.8 Regional settings

Some converters display language specific information in the generated documents, e.g., the *From*, *To* and *Subject* labels in an email. By default, the Converter will detect the *display language* of the account the Conversion Service is running under, providing translations are available for that language. This *display language* can be overridden in the config file by setting the value of the *ConversionLocalization* value to the relevant language code, e.g., 'de' for German.

Language *Resource files* must be found in the 'Resources' sub folder of the installation directory, otherwise this setting will have no effect. This setting does not affect date or number formats. At the time of writing this setting only affects the conversion of MSG and EML files.

2.6.9 InfoPath specific switches

To keep InfoPath installation relatively straight forward, the Document Conversion Server makes certain modifications to the files before conversion takes place. In some cases, this is not desirable, for example if data from an external data source is used for display purposes, or if certain embedded code must run before conversion.

If additional control over the InfoPath conversion is required, then consider modifying the following settings.

- Should InfoPath forms marked as *requiring Full Trust* be processed based on the parameters below (e.g. *StripDotNETCode*) or not ? Generally, leave the default value of 'true' unless your XSN file is digitally signed
<add key="InfoPathConverterFullFidelity.ProcessFullTrustForms" value="true"/>
- Remove all .net code from the form before conversion. Defaults to *True*.
<add key="InfoPathConverterFullFidelity.StripDotNETCode" value="true"/>
- Remove all external data connections before conversion. Defaults to *True*.
<add key="InfoPathConverterFullFidelity.StripDataObjects" value="true"/>
- Process any rulesets that may be present. Defaults to *True*.
<add key="InfoPathConverterFullFidelity.ProcessRuleSets" value="true"/>
- Some forms with complex rules such as *get-SharePointServerRootUrl()* require the trust level to be restricted. When this value is set to true *StripDotNETCode* and *StripDataObjects* *MUST* be set to 'true' as well.
<add key="InfoPathConverterFullFidelity.RestrictTrustLevel" value="false"/>

Note that modifying these options makes InfoPath configuration more complex. Do not change these settings unless you have a good reason to do so. For details see *Appendix - Using InfoPath with External Data Sources*.

Although we are happy to assist, we cannot guarantee that the PDF Converter will operate correctly when the before mentioned settings have been modified. Having said that, many of our customers use the PDF Converter without any problems using custom settings.

By default, attachments found in InfoPath views are converted to PDF and merged into a single PDF file. This behaviour can be disabled using the following configuration key.

```
<add key="InfoPathConverterFullFidelity.ConvertAttachments" value="true"/>
```

To convert InfoPath forms to PDF the converter retrieves the XSN file associated with the InfoPath XML file. By default, it uses the Muhimbi Service's credentials when downloading this XSN file, but in some environments this may not be desirable. For those situations it is possible to override the credentials using the following configuration keys:

```
<add key="InfoPathConverterFullFidelity.XSNUserName" value=""/>
```

```
<add key="InfoPathConverterFullFidelity.XSNPassword" value=""/>
```

```
<add key="InfoPathConverterFullFidelity.XSNDomain" value=""/>
```

For details about the *InfoPathConverterFullFidelity.AutoTrustForms* setting see *Appendix - Using InfoPath with External Data Sources*

InfoPath internally caches files for each request. Over time this may use up a lot of space on the server's hard disk. By default, this cache is cleared by the PDF Converter once an hour, but this duration can be changed using the *InfoPathConverterFullFidelity.XSNCacheClearInterval* setting.

When Internet Explorer 10 was released, InfoPath's internal PDF Export capabilities were damaged (e.g., check boxes are no longer displayed correctly when *ticked*). To mitigate this problem the PDF Converter provides 2 options:

1. Switch to the *High Quality InfoPath Converter*, a completely re-architected InfoPath converter introduced in version 8.0 of the Muhimbi PDF Converter. The PDF Converter's installer provides an option to enable this converter, but it can also be enabled manually by setting the following configuration setting to *true*:

```
<add key="InfoPathConverterFullFidelity.UseNativePrintEngine" value="true"/>
```

This converter requires Ghostscript to be installed, see [Installing Prerequisites & Dependencies - Ghostscript](#). For more information see *Appendix – Switching between InfoPath Converters*.

2. If switching to the High Quality InfoPath Converter is not an option then either roll back to Internet Explorer 9 (on the server running the Conversion Service) or set the value of *InfoPathConverterFullFidelity.RenderingMode* in the PDF Converter's configuration file to 'Bitmap'. Although Bitmap based output looks better, the content of these PDFs cannot be indexed or searched.

Providing the High Quality InfoPath Converter is enabled, the following configuration settings can be changed as well:

- **InfoPathConverterFullFidelity.DefaultPaperSize:** The output paper size to use for those InfoPath views where the printer / paper size is not specified. This does not change the paper size for views where the printer / paper size IS specified.

Leave this value empty to take the value from the default printer or specify a [named format](#) such as 'A4' or 'Letter'. Please note that this value is case sensitive.
- **InfoPathConverterFullFidelity.ForcePaperSize:** Force the paper size regardless of the printer / paper size being present or not in the definition of the InfoPath view.
- **InfoPathConverterFullFidelity.DefaultPageOrientation:** The Page orientation for InfoPath views that don't explicitly specify a printer / paper size. Either 'Portrait' or 'Landscape'. Leave empty to let InfoPath decide.
- **InfoPathConverterFullFidelity.ForcePageOrientation:** Force the page orientation regardless of the printer / paper size being present or not in the definition of the InfoPath view.

2.6.10 HTML specific switches

The Muhimbi PDF Converter comes with 3 different HTML to PDF Conversion engines. Legacy ones, based on Internet Explorer and WebKit, and a fully up to date high-fidelity converter based on the Chromium framework. Please keep in mind that the Chromium converter is enabled by default, and that switching back to the legacy converters is possible, but generally discouraged.

HTML is not particularly well suited for printing or PDF Conversion, however our software generally generates good results, especially with guidance provided in the following Knowledge Base articles:

- [Converting HTML - Empty page / Authentication problems.](#)
- [Solving formatting issues when converting HTML to PDF.](#)

The conversion service's config file provides a high level of control over the HTML Converter. The key settings – including their default values - are listed below. Please consult the config file's in-line documentation for more details:

Generic

- HtmlRenderingEngine: Chromium
- ForceJavaScript: True
- PaperSize: Letter
- PageOrientation: Portrait
- PageMargin: 0.5,0.5,0.5,0.5
- ScaleMode: FitWidth

- URLUsername
- URLPassword
- ConversionDelay: 0
- AuthenticationMode: WebAuthentication
- ErrorReporting: Detail
- SanitizeHyperlinks: True

WebKit & Chromium

- EnableWebKitOfflineMode: false
- WebKitViewPortSize: Paper
- MediaType: Print
- WebKitUserAgent

Webkit specific

- WebKitProxyURL
- WebKitProxyUsername
- WebKitProxyPassword

WebKit & IE

- SplitTextLines: False
- SplitImages: False

Chromium specific

- CompressionThreshold: 2048

IE specific

- ClearBrowserCache: True
- IEAlternativeFormRendering: false

Please note that these settings can also be overridden on a request-by-request basis using the *ConverterSpecificSettings* property on the web services interface. For details see the Developer Guide.

2.6.11 Word processing (MS-Word) specific switches

The following settings are specific to the Word processing-based converter and can be specified either globally for all requests using the service's config file or on a *request-by-request* basis using the *ConverterSpecificSettings* property on the web services interface.

- **RevisionsAndCommentsDisplayMode:** Specify the default value for how to view the proposed changes to the document.

- **FinalShowingMarkup:** Convert the document with all proposed changes highlighted.
- **Final:** Convert the document with all proposed changes included.
- **OriginalShowingMarkup:** Convert the original document with all proposed changes highlighted.
- **Original:** Convert the document before any changes were made.

```
<add key="WordProcessorConverterFullFidelity.RevisionsAndComments  
DisplayMode" value="Final"/>
```
- **RevisionsAndCommentsMarkupMode:** Specify the default value for how to visualise revisions to the document. You can show revisions as balloons in the margins of the document or show them directly within the document itself.
 - **InLine:** Show all revisions Inline
 - **Balloon:** Show all revisions in balloons
 - **Mixed:** Show only comments and formatting in balloons

```
<add key="WordProcessorConverterFullFidelity.RevisionsAndComments  
MarkupMode" value="InLine"/>
```
- **ProcessDocumentTemplate:** Specify if the MS-Word template will need to be stripped out for DOCX files. Leave the default setting (true) unless formatting problems occur.
- **ProcessDOCFiles:** Pre-convert DOC files to DOCX to allow processing using the above mentioned *ProcessDocumentTemplate* setting. Only enable this option when experiencing problems with Document Information Panels. Note that there may be some side effects so talk to support@muhimbi.com before changing this setting.
- **RefreshFull:** When *OpenOptions.RefreshContent* (web service) or *OpenOptions.ForceRefreshContent* (config file) is set to *True*, the PDF Converter refreshes the Table of Content, references and other fields.

For certain field types the CPU usage is unreasonably high, which is why these are ignored by default. By setting *RefreshFull* to *True*, these 'difficult' fields are refreshed as well.

This setting only applies to files of type DOCX, DOTX, DOCM, and DOTM. The fields in question are *PrintDate* and *SaveDate*. This setting is set to *False* by default.
- **RefreshFullLegacyFormats:** This setting is similar to *RefreshFull*, but for legacy document formats (any type except DOCX, DOTX, DOCM, and DOTM). For these file types it is only possible to refresh content in a CPU intensive manner.

Due to the potential large number and important fields involved, not just *PrintDate* and *SaveDate*, this value is set to *True* by default.

2.6.12 Spreadsheets (Excel) specific switches

The following settings are specific to the Spreadsheet converter and can be specified either globally for all requests using the service's config file or on a *request-by-request* basis using the *ConverterSpecificSettings* property on the web services interface.

- **FitToPagesWide:** The default value for the number of pages tall the worksheet will be scaled to when it is converted.
- **FitToPagesTall:** The default value for the number of pages wide the worksheet will be scaled to when it is converted.
- **UnhideAllRows & UnhideAllColumns:** Attempt to include hidden rows and columns in the destination document. This may fail when the source document contains protected or locked content. Unless overridden programmatically, this defaults to *not configured*, meaning *False*.
- **AutoTrustDocuments:** Enable this setting to allow XLS files caught by Excel's 'Protected View' facility to be converted. The default is *False*.

2.6.13 Presentations (PowerPoint) specific switches

The following settings are specific to the Presentations based converter and can be specified either globally for all requests using the service's config file or on a *request-by-request* basis using the *ConverterSpecificSettings* property on the web services interface.

- **PrintOutputType:** Specify the default value for the part of the presentation to print. Supported values are: *Slides*, *NotesPages*, *Outline*, *OneSlideHandouts*, *TwoSlideHandouts*, *ThreeSlideHandouts*, *FourSlideHandouts*, *SixSlideHandouts*, *NineSlideHandouts*.

```
<add key="PresentationConverterFullFidelity.PrintOutputType" value="Slides"/>
```

- **FrameSlides:** Specify the default value for the border around the slide. Accepts *True* or *False*.

```
<add key="PresentationConverterFullFidelity.FrameSlides" value="True"/>
```

2.6.14 AutoCAD specific switches

The AutoCAD converter is highly configurable and allows the following settings to be specified either globally for all requests using the service's config file or on a *request-by-request* basis using the *ConverterSpecific Settings* property on the web services interface.

For full details search the service's config file for all keys starting with *CadConverterFullFidelity*.

- **PaperSize:** Specify the paper size of the generated PDF file, either using standardised paper size names such as *A4* or *Letter*, or using custom dimensions in inches or millimetres.
- **PageMargin:** Determine the margin around the page, either as a single uniform value or using a different value for *Left*, *Top*, *Right* and *Bottom*.
- **PaperOrientation:** Automatically detect the drawing's orientation or force it to *Portrait* or *Landscape*.
- **BackgroundColor:** Some AutoCAD drawings have been created using a non-white background colour, in which case you may want to force the background colour in the PDF file using this setting.
- **ForegroundColor:** The PDF Converter supports a number of automatic re-colouring options to make sure that foreground colours don't conflict with background colours or to convert all colours to grey scale. For details see the in-line documentation in the service's config file.
- **Content:** By default, all valid layouts found in a DXF or DWG file are converted to PDF. However, you may want to override this setting and only convert specific layouts. This can be particularly useful when converting 3D AutoCAD files.
- **EmptyLayoutDetectionMode:** Specifies how the conversion handles empty or nearly empty layouts.
- **LayoutSortOrder:** Specify the sort order for layout names.
- **ExternalReferences:** Optional path for resolving external references in drawings. Note that all subfolders are automatically searched as well.

2.6.15 MSG & EML (email) specific switches

The email converter allows the following settings to be specified either globally for all requests, using the service's config file, or on a *request-by-request* basis using the *ConverterSpecificSettings* property on the web services interface.

For full details search the service's config file for all keys starting with *MSGConverterFullFidelity*.

- **PaperSize:** Specify the paper size of the generated PDF file, either using standardised paper size names such as *A4* or *Letter*, or using custom dimensions in inches or millimetres.
- **PageMargin:** The margin / border around the generated PDF file. One or four {value}{dim} components separated by commas (,) where:
 - {value} is a numerical value
 - {dim} is the dimension which can be 'mm', 'in.' or 'inches'. (Defaults to inches when nothing is specified)

When multiple values are specified then the sequence is: left, top, right and bottom. For example: "12mm, 24mm, 12mm, 24mm"

- **PageOrientation:** Specifies the page orientation, either 'Portrait' or 'Landscape'.
- **ConvertAttachments:** This config value determines if email attachments will be converted (and merged to the main email), or not.
- **DisplayAttachmentSummary:** Specify whether the attachment filenames are displayed in the email header. This setting works independently to *ConvertAttachments*.
- **AttachmentMergeMode:** Control how to deal with attachments if the source email contains any. There are options to convert attachments to PDF or attach them in their original format.
- **UnsupportedAttachmentBehaviour:** Control how the system behaves when unsupported attachments are found. Ignore them or raise an error.
- **Include (and exclude) AttachmentTypes:** Control which attachments will be processed, or ignored, based on their file name.
- **MinimumImageAttachmentDimension:** Filter out small images, typically included in a sender's signature.
- **HTMLScaleMode:** Define how images inside HTML based emails are scaled. Unless you have a real good reason to change this value, leave it on the default *FitWidthScaleImagesOnly* setting.
- **PlainTextBodyLineBreaks:** Determine how return characters (new lines) in plain text MSG bodies are handled. By default, Outlook removes certain return characters in plain text emails and the PDF Converter emulates this behaviour. Accepted values are:
 - RetainAll - All carriage returns in the plain text message are retained. Lines may wrap before 80 characters in length.
 - RemoveExtra - Lines that appear to be wrapped due to their length have the return characters removed. Similar to Outlook's implementation.

- Legacy - A legacy implementation of *RemoveExtra* used prior to version 7.1 of the PDF Converter
- **BestBodyMode:** Determines which email body content (Text / HTML / RTF / RTFHTML) to extract when processing MSG files. Accepted values are:
 - Strict - Implementation of the MS-OXBBODY Best Body algorithm. When RTF content is in sync with the native email content (HTML / Text) the RTF content takes precedence.
 - Default - The default implementation uses the first content found in order of HTML, RTFHTML, RTFTEXT, RTFTEXT and lastly Text.
NOTE: Always used when converting EML emails.
- **EmailAddressDisplayMode:** Determines how email addresses (except the *From address or Calendar Organizer*) are displayed. Where multiple types of email addresses are found (SMTP, Exchange), the SMTP address is favoured. Accepted values are:
 - Name - Display the email name only, omitting any address details. If the name is not found the email address is displayed.
 - NameAndAddress - Display both the name and email address.
 - Address - Display the email address only, omitting any name details. If the email address is not found, the name is displayed.
 - NameAndSMTPAddress - Display the email recipient name as well as their email address, but only if an SMTP address is found.
- **FromEmailAddressDisplayMode:** As per *EmailAddressDisplayMode* except only applicable to the *From addresses and Calendar Organizer*.
- **BreakOnUnsupportedAttachment:** When an unsupported attachment is found, the conversion is halted, and an error message is returned.
- **BreakOnUnsupportedEmbeddedObject:** When an unsupported embedded object is found, e.g., an embedded OLE object where no file type identification is provided, the conversion is halted, and an error message is returned.
- **EmbeddedObjectDisplayMode:** Determines how embedded objects are displayed. NOTE: Where the embedded object is displayed as an icon, use *EmbeddedObjectIconDisplayMode*. Accepted values are:
 - InlineNoScale - The embedded object is displayed as it appears in Outlook. If the object is larger than the current page size, it will be cropped.
 - InlineFitWidth - The embedded object is scaled so that its (width) content fits on the page.
 - Disabled - The embedded object is hidden from any output.

- **EmbeddedObjectIconDisplayMode:** Determines how embedded objects are displayed where they are stored as an icon. Accepted values are:
 - IconOnly - The embedded object icon is displayed as it appears in Outlook.
 - Disabled - The embedded object icon is hidden from any output.
- **SentDateMissingDisplayMode:** Control what is displayed when the *sent date* of an email is not found (ie, the email was never sent). Accepted values are:
 - ModifiedDate - Displays the modified date of the MSG file itself (not supported in EML)
 - Any Text - Displays the text as entered.

2.6.16 Switches used for overriding settings

The following settings can be changed to allow any client side settings to be globally overridden. *Unless you have a specific reason to change them, leave these settings alone.*

- **Merging.ForceBreakOnError:** Override the *BreakOnError* value during merge operations. Leave empty to use the setting specified in the web service call (MergeSettings). Specify 'True' to fail the entire operation when an error occurs. Use 'False' to continue and skip any problematic files.
- **Merging.ForceOmitErrorPages:** Override the *OmitErrorPages* value during merge operations. Leave empty to use the setting specified in the web service call (MergeSettings). Specify 'False' to insert pages into the PDF indicating that an error has occurred while converting a specific document.
- **Merging.StripDigitalSignatures:** When merging PDF files (or InfoPath / MSG attachments) that contain digital signatures then please set this value to 'true' to remove all digital signatures. Signed documents cannot be merged.
- **Merging.DefaultDocumentStartPage:** Default value for the *DocumentStart Page* value when merging documents. This value is used when 'Default' is specified in the MergeSettings object. The available options are:
 - Next - When merging, start each document on the next page.
 - Odd - When merging, start each document on an odd page (usually the right hand page when printing double sided)
 - Even - When merging, start each document on the next even page
- **OpenOptions.ForceAllowMacros:** Override the *AllowMacros* value during conversion. Leave empty to use the setting specified in the web service call (in OpenOptions). Accepted values are members of the *Muhimbi.DocumentConverter.WebService.Data.MacroSecurityOption* enumeration or an empty string.
 - None - Don't allow any macros to run
 - SignedOnly - Only allow macros to run with a valid digital signature
 - All - Allow all macros, regardless of their origin and digital signature. This is not recommended as it may compromise the security of the server.

- **OpenOptions.ForceRefreshContent:** Override the *RefreshContent* value during conversion. Leave empty to use the setting specified in the web service call (in OpenOptions). Accepted values are 'true', 'false' or empty string.
- **ConversionSettings.ForceQuality:** Override the *Quality* value during conversion. Leave empty to use the setting specified in the web service call (in ConversionSettings). Accepted values are members of the *Muhimbi.DocumentConverter.WebService.Data.ConversionQuality* enumeration or an empty string.
 - OptimizeForPrint - Optimise the file size and resolution for print purposes.
 - OptimizeForOnScreen - Optimise the file size and resolution for use on a computer screen
- **ConversionSettings.ForceRange:** Override the *Range* value during conversion. Leave empty to use the setting specified in the web service call (in ConversionSettings). Accepted values are members of the *Muhimbi.DocumentConverter.WebService.Data.ConversionRange* enumeration or an empty string.
 - VisibleDocuments - Skips, in case of Excel and PowerPoint, any hidden tabs or slides.
 - AllDocuments - Export all tabs or slides in a workspace.
 - ActiveDocuments - Exports, in case of Excel, the selected tabs.
- **ConversionSettings.ForceGenerateBookmarks:** Override the *GenerateBookmarks* value during conversion. Leave empty to use the setting specified in the web service call (in ConversionSettings). Accepted values are members of the *Muhimbi.DocumentConverter.WebService.Data.BookmarkGenerationOption* enumeration or an empty string.
 - Disabled - Don't generate any bookmarks.
 - Automatic - Based on headings, if applicable.
 - Custom - Based on bookmarks defined in the document, e.g. MS-Word Bookmarks, if applicable.
- **ConversionSettings.ForcePDFProfile:** Override the *PDFProfile* value during conversion. Leave empty to use the setting specified in the web service call (in ConversionSettings). Accepted values are members of the *Muhimbi.DocumentConverter.WebService.Data.PDFProfile* enumeration or an empty string.
 - PDF_1_5 - Use PDF Version 1.5
 - PDF_A1B - Use the PDF/A standard for long term archiving

2.6.17 PDF & Security Settings

The default PDF Security and Encryption settings have been designed to be compatible with the widest possible range of PDF Readers. However, this means that security has been 'dumbed down' a bit. If all your users are on relatively recent versions of PDF Readers, then you may want to tighten up the security settings.

- **PDF.EncryptionKeySize:** Set the security key size for encrypting PDF files. Possible values are 40, 128 and 256. Please note that 256 Bit is not supported on versions of Acrobat older than version 9 and many other PDF readers.
- **PDF.EncryptionAlgorithm:** Set the security algorithm, either *RC4* or *AES*. Although cryptographically stronger, be careful when using *AES* as only Acrobat Reader 7 or newer can open *AES* encrypted files.
- **PostProcessSecuredFiles:** Define whether PDF files that are already secured will be post processed by the system. Post processing can include watermarking and applying security settings.
- **PostProcessPDF1.5:** For legacy reasons *PDFProfile.PDF_1_5* is treated the same as the *PDFProfile.Default* PDF output type. As a result, it is not guaranteed to be 100% 1.5 compliant. Enable this setting to make sure the file is post processed and forced to be 1.5 compliant. (Requires Pro license)
- **ConvertAttachments:** Convert, and merge, files that are attached to the PDF. This setting is disabled by default.
- **ConvertAttachmentMode:** What to do with attachments after conversion (Only used when *PDF.ConvertAttachments* is enabled):
 - **RemoveAll:** Convert and delete all files attached to the PDF, even file formats not supported by the converter. This is the default setting.
 - **RemoveSupported:** Convert and delete all files supported by the PDF converter, but leave unsupported files attached.
- **AttachmentMergeMode:** Control how to deal with attachments if the source PDF contains any. There are options to convert attachments to PDF, or attach them in their original format.
- **UnsupportedAttachmentBehaviour:** Control how the system behaves when unsupported attachments are found. Ignore them or raise an error.
- **Include (and exclude) AttachmentTypes:** Control which attachments will be processed, or ignored, based on their file name.
- **NamedDestinationProcessingMode:** How to deal with the automatic generation of 'Named Destinations' using the PDF's Bookmarks. The default value is 'None'.
 - **None:** Do not change the named destinations defined in the document.
 - **ClearAll:** Remove all named destinations. (All bookmarks pointing to existing named destinations will be fixed up automatically)
 - **Merge:** Keep existing named destinations and add new ones based on the PDF's bookmarks.
 - **Replace:** Remove all existing named destinations and add new ones based on the PDF's bookmarks.

- **IgnorePortfolioCoverSheet:** When merging multiple files embedded in a PDF Portfolio file, include the portfolio's cover sheet or not.
- **Watermark.DisableIncrementalUpdate:** When watermarking PDF documents with large number of pages, using incremental update can increase the file size considerably. Use this setting to disable it. Processing, generally, takes longer when incremental update is disabled.

2.7 Hardening the Conversion Service

Some environments have particularly strict security requirements and require the server running the conversion service to be 'hardened' from a security perspective. For detailed guidance on this topic see the Knowledge Base article at <https://www.muhimbi.com/knowledge-base/hardening-securing-the-server-that-runs-the-conversion-service/>

2.8 Uninstalling

The Conversion Service can be uninstalled using the standard Windows *Programs and Features* control panel. (also known as *Add / Remove Programs*, appwiz.cpl).

2.9 Upgrading from a previous version

Although the PDF Converter is a friendly and well tested application, its underpinnings are quite complex. It is rare for the Muhimbi support desk to receive any complaints about the upgrade process, yet we don't recommend upgrading unless there is a specific feature in a new version that you need. We recommend following industry best practices by deploying new versions in a Development and Test environments, and only deploy to Production after thorough testing.

Upgrading the software from a previous version is simple and straight forward. However, there are some things to consider:

1. **Breaking changes:** It is rare that the software breaks something between releases, but sometimes a fundamental change is made (E.g., a brand-new HTML converter) that may have side effects on certain customer-specific edge cases. Before upgrading, please consult <https://www.muhimbi.com/licensing/upgrading-the-muhimbi-pdf-converter/>.
2. **Service's config file:** The Conversion Service uses a standard `.config` file to control the default behaviour of the product (see see 2.6 *Tuning the Document Conversion service*). Please note that any changes to this file are not maintained during an upgrade. You will need to manually re-apply any changes that may have been made since the previous installation. We advise making a copy of this file before starting the unistall.

Aside from the points and actions described above, the upgrade process is as follows:

1. Uninstall the software are described in 2.8 *Uninstall*.
2. Install the new version as described in 2.3 *Installation steps*.
3. Make any of the post installation changes described in the introduction of this section.

3 Troubleshooting & Other common tasks

This section provides some guidance related to troubleshooting problems.

If you still have questions after reading this chapter, then please visit our support desk at support.muhimbi.com or contact our support team at support@muhimbi.com.

3.1 Windows Event Log

The following entries may be written to the event log:

1. **Warnings:** If you are running an evaluation copy of the software, or your license has expired then this is reported as a warning message in the Application Event Log.
2. **Errors:** All errors are written to the event log. The main cause of errors is corrupt documents.

Note that all event entries written by Muhimbi's products use Event ID 41734.

3.2 Trace Log

The Document Converter Service maintains a detailed trace log named *Logs\DocumentConverter.log* in the directory the service has been installed in.

For details on how to change the log settings see section 2.6.5 *Logging*.

3.3 Common issues & Errors

3.3.1 Error messages related to printer drivers or the printer spooler are logged

The Excel and InfoPath converters require the *Printer Spooler* service to be started. The installer used by the Document Conversion Service attempts to start the Printer Spooler service automatically, but if the service is subsequently disabled then you may receive an error.

These converters also require at least one printer driver to be installed. Although in general it doesn't matter which driver is installed, some drivers such as VMWare's *Virtual Printer* or the *OneNote printer* will cause problems. Windows Server 2003 R2 and Windows Server 2008 automatically install the XPS driver, it is recommended to make this the default printer.

3.3.2 Problems parsing the WSDL

By default, the Conversion Service uses the host name of the local system as the base address. Most web service client libraries deal with this correctly, however if the service is exposed using a different machine name, then you may need to update the *base address* to the system's IP-address.

To change this, modify the *baseAddress* attribute in the configuration file and restart the service. For details see [Generating Web Service proxies against a remote machine fails \(muhimbi.com\)](#).

3.3.3 Documents using non standard fonts (e.g. Japanese) are not converted properly / The fonts in the destination document are not correct

If the fonts in the converted document are not displayed properly then please make sure that the correct fonts are installed on the server hosting the Document Conversion Service, **restart the server after installing new fonts**.

For example, if Japanese characters are not displayed properly then make sure the Japanese language pack for Windows Server is installed.

If a font cannot be found then the system will attempt to find an alternative, but similar, font. If an alternative cannot be found, then the system will use the Times New Roman font.

When using the optional *PDF/A post processing facility* then all used fonts are automatically embedded in the PDF document. If fonts are deployed to the server after Ghostscript was installed, then please execute the following command from the Ghostscript bin directory.

```
gswin32c -q -dBATCH -sFONTDIR=c:/windows/fonts -  
sCIDFMAP=../lib/cidfmap ../lib/mkcidfm.ps
```

Please replace *gswin32c* with *gswin64c* when using the 64-bit version of Ghostscript. If the Windows directory is located elsewhere then change the *sFONTDIR*, please use forward slashes, not backslashes.

For more details about how the optional Ghostscript installation is used for PDF/A post processing see *Appendix - Post processing PDF output to PDF/A*.

AutoCAD fonts can be placed in the *SHXFonts* folder under the Conversion Service's installation folder.

3.3.4 Problems converting InfoPath forms without a shared XSN file

If you intend to convert InfoPath forms that don't rely on a shared XSN template stored on a file share or in SharePoint then the XSN file must be registered on the Document Conversion Server using RegForm.exe.

The RegForm application can be found in C:\Program Files\Microsoft Office\Office12. For details see the second half of the following article:

<http://msdn.microsoft.com/en-us/library/bb251025.aspx>

3.3.5 InfoPath forms using Ink controls fail to convert

When using Ink Controls, for example to capture signatures on InfoPath forms, then please make sure the Ink controls are installed on the machine(s) that run the Muhimbi Conversion Service. For example, in Windows Server 2008 these controls are installed as part of the "Ink & Handwriting features", part of the "Desktop Experience" Windows Feature.

3.3.6 Error 403 (Forbidden) when converting InfoPath forms

When InfoPath conversion fails and the error message in the Windows Application Event Log refers to “*The remote server returned an error: (403) Forbidden*” then this may be caused by the authentication type specified on the Web Application. If, for example, Forms Based Authentication is enabled, but NTLM authentication is not configured on the same Web Application then the Muhimbi Conversion Service will not be able to download the XSN file.

The problem can be solved by adding NTLM authentication to the Web Application.

3.3.7 InfoPath files are converted using an old version of the XSN template

Although rare, InfoPath sometimes re-uses an old XSN file when carrying out a conversion. Running the typical `InfoPath /cache ClearAll` command will not work as XSN files are stored in a different location when running as a service.

Please clear all *FormCache* folders manually from the following location:

64bit: %windir%\SysWOW64\config\systemprofile\AppData\Local\Microsoft\InfoPath

32bit: %windir%\System32\config\systemprofile\AppData\Local\Microsoft\InfoPath

Appendix - Using InfoPath with External Data Sources

Please note that this appendix only applies to environments that have 'external data sources' enabled. Enabling data sources is a manual action, default deployments of the Muhimbi PDF Converter never ship with this option enabled. For more details see 2.6.9 'InfoPath specific switches'

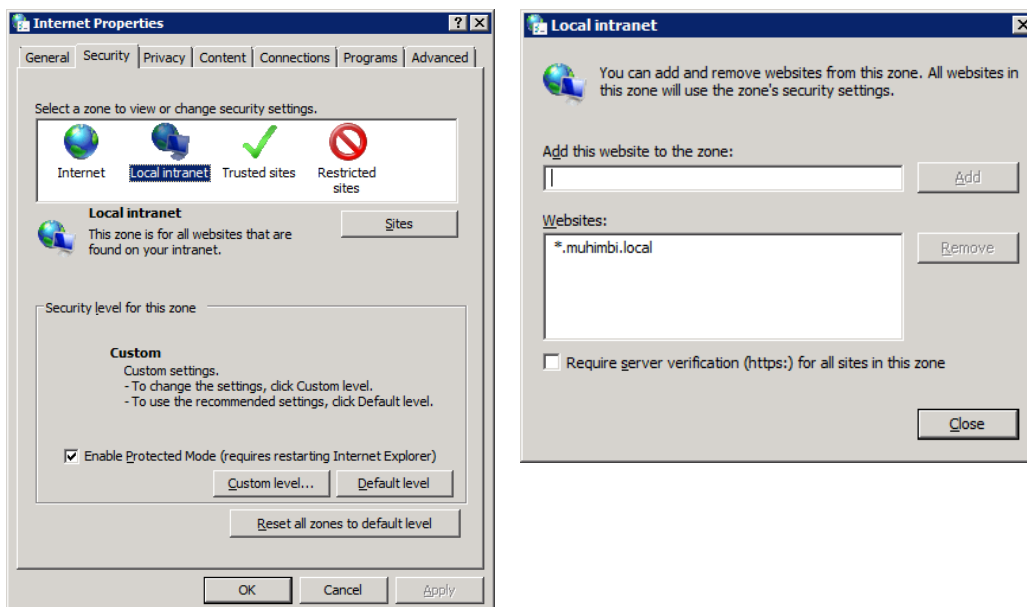
Although we are happy to assist, we do not officially support conversions that run with 'StripDotNETCode', 'StripDataObjects' or 'ProcessRuleSets' set to 'false'. Having said that, many of our customers use the PDF Converter without any problems using these custom settings.

Details for InfoPath 2007

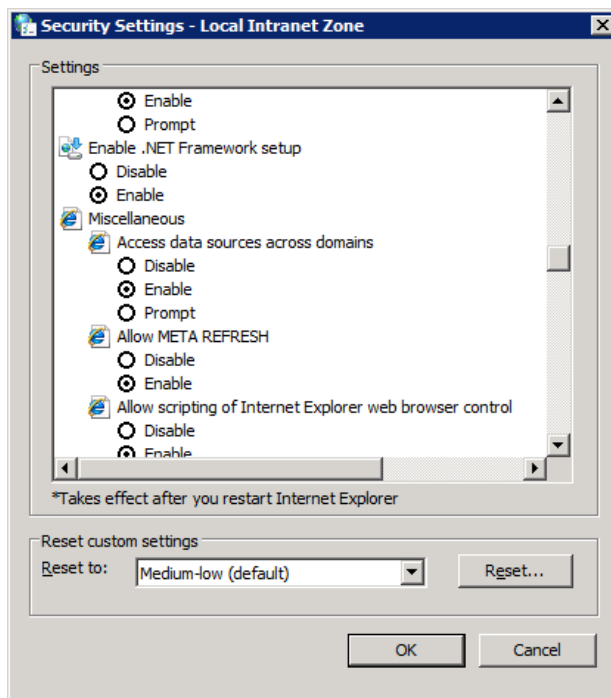
When an InfoPath document containing external connections, e.g. a dropdown list with the contents of a SharePoint list, fails to convert then this may be caused by the location of the XSN file not being trusted or the *access data sources across domains* setting not being enabled for the trusted site.

Ideally this configuration change should be made by a Domain Administrator using a group policy. However, the change can be made manually as well using the steps outlined below:

1. Log in using the account the *Muhimbi Document Converter Service* is running under.
2. Open *Internet Options* either from Internet Explorer or the Control Panel.
3. Verify the site that hosts the XSN file is recognised as a *Local Intranet* site by selecting *Local Intranet*, clicking the *Sites* button followed by the *Advanced* button. You may need to uninstall / disable *Internet Explorer Enhanced Security* on your server in order for this to work.

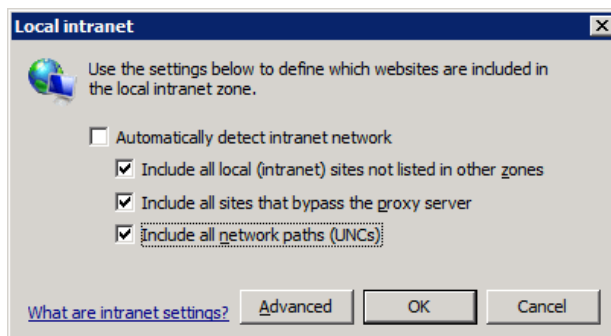


4. With *Local Intranet* selected, click the *Custom Level* button and verify that *Access data sources across domains* is set to *Enable*.



- In the same screen navigate to the *User Authentication* section and select *Automatic Logon with current user name and password*.

If the previous steps fail to resolve the problem then make sure automatic detection of the Intranet network is disabled and the individual options (as per the screenshot below are enabled).



This screen can be opened from *Internet Options / Security / Local Intranet / Sites*. Note that in Internet Explorer 6 the *Automatically detect intranet network* option is not present, but the individual options are.

If conversions of InfoPath documents don't work consistently then flush the local Form Template Cache by logging in as the service account and issuing the following command from Windows' Start / Run menu:

```
Infopath /cache clearall
```

Details for InfoPath 2010 & 2013

In InfoPath 2010 Microsoft has made changes to the way forms are trusted. As a result, the instructions for InfoPath 2010 are different from those for InfoPath 2007.

Digitally signing forms

The most reliable way to convert forms *as-is*, with full support for Data Connections, Rule Sets, and custom code, is to digitally sign the XSN file.

For details about how to do this see the following resources.

- <http://www.thorprojects.com/blog/archive/2007/06/30/domain-certificate-authority-signing-infopath-2007-forms.aspx>
- <http://msdn.microsoft.com/en-us/library/ee526348.aspx>
- <http://msdn.microsoft.com/en-us/library/ee526349.aspx>
- <http://msdn.microsoft.com/en-us/library/aa946782.aspx>

Please make sure to use a digital certificate that uses an *authority* that is trusted by the server that runs the Muhimbi Conversion Service. Temporary test certificates created on a development machine will not work when used on other machines.

When using digitally signed forms AND you wish to access external data connections and/or custom code during conversion then please set the *InfoPathConverterFullFidelity.ProcessFullTrustForms* key in the Muhimbi Service's config file to *false*.

Using Muhimbi's 'AutoTrustForms' feature

Starting with version 5.1 of the Muhimbi PDF Converter it is possible to convert InfoPath forms that use either the *Domain* or *Automatic* trust levels, with full support for External Data Sources, Rule Sets and running custom code.

To prevent compatibility problems with any previous versions of the software this feature is disabled by default. In order to enable it set the following value to *true* in the Muhimbi Service's config file and restart the service.

```
<add key="InfoPathConverterFullFidelity.AutoTrustForms" value="false"/>
```

Just enabling this feature by itself will achieve nothing, the idea is to disable at least one of the following options as well:

```
<add key="InfoPathConverterFullFidelity.StripDotNETCode" value="true"/>
```

```
<add key="InfoPathConverterFullFidelity.StripDataObjects" value="true"/>
```

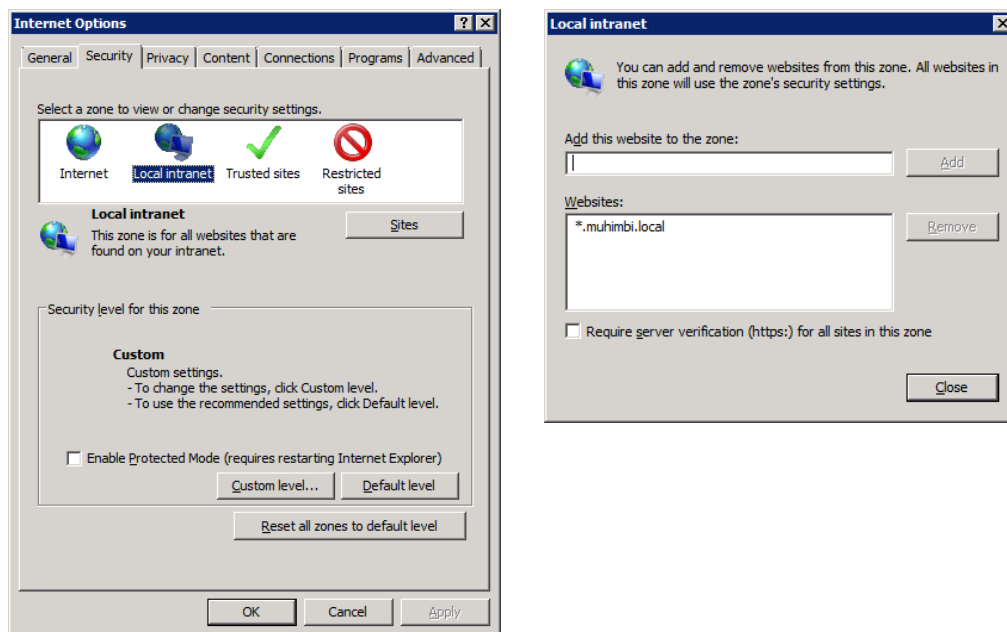
```
<add key="InfoPathConverterFullFidelity.ProcessRuleSets" value="true"/>
```

For more details about these settings as well as editing the service's config file see 2.6.9 *InfoPath specific switches*.

In order to use this functionality file sharing must be enabled on the server that runs the Muhimbi Service and the default *Administrative Drive Shares* (C\$, D\$ etc) must be available as well.

When the *AutoTrustForms* facility is used to connect to external systems, e.g. a web service on your SharePoint server, then please make sure the server name is recognised as a *local intranet* system by adding the server name / entire domain to the list of *local intranet* sites. This can be done either manually or, even better, using a group policy. The manual steps are as follows:

1. Log in to the desktop using the account the Muhimbi Conversion Service runs under.
2. Start Internet Explorer and navigate to Tools / Internet Options / Security.
3. Select 'Local Intranet' and click the 'Default Level' button.
4. Click the 'Sites' button followed by 'Advanced'.
5. Add the server that is being connected to to the list.



Similar to the *Details for InfoPath 2007* section above, click the *Custom Level* button and make sure that *User Authentication* is set to *Automatic Logon with current user name and password* and that *Access data sources across domains* is set to *Enable*.

Appendix – Switching between InfoPath Converters

As of version 8.0 the PDF Converter comes with two different InfoPath converters, a *legacy converter* as well as a brand new *high fidelity converter*. Both converters provide the same facilities for converting InfoPath forms, including *dynamic view selection* and conversion of InfoPath attachments; however the *high fidelity converter* generates much higher quality output.

This appendix describes how to switch between converters. For more background information as well as additional details on this subject see <https://www.muhimbi.com/blog/introducing-the-new-high-fidelity-infopath-converter/>. If you wish to enable the new InfoPath converter on Windows Server 2003 then please contact support@muhimbi.com for details.

Enabling the high fidelity InfoPath Converter

The high fidelity converter is enabled by default. If the decision was made to enable the legacy converter during installation, but there is a need to switch to the high fidelity version, then please follow the steps below:

1. If not already done so during the PDF Converter's original installation process, deploy Ghostscript to the server running the Conversion Service. For details see [Installing Prerequisites & Dependencies GhostScript](#).
2. Edit the conversion service's configuration file. For details about how to do this see [this article](#).
3. Set the value of the `InfoPathConverterFullFidelity.UseNativePrintEngine` setting to `true`.
4. Save the configuration file and restart the Conversion Service

If the Conversion Service is running on multiple systems then repeat these steps for each system.

Enabling the legacy InfoPath Converter

If the PDF Converter was installed using the default - *high fidelity* - option, but there is a need to switch back to the legacy converter, perhaps because the InfoPath forms were designed specifically for the legacy converter, then you can switch back as follows:

1. Edit the conversion service's configuration file. For details about how to do this see <https://www.muhimbi.com/knowledge-base/how-to-edit-the-conversion-service-s-configuration-file/>.
2. Set the value of the `InfoPathConverterFullFidelity.UseNativePrintEngine` setting to `false`.
3. Save the configuration file and restart the Conversion Service.

If the Conversion Service is running on multiple systems then repeat these steps for each system.

Appendix - Post processing PDF output to PDF/A

Muhimbi's range of PDF Conversion products supports output in PDF/A1b, A2b & A3b format using the PDF Converter Professional Add-on license. This appendix provides details about how to configure the system to output in PDF/A format.

For more background information as well as additional details on this subject see <https://www.muhimbi.com/blog/pdf-a-support-in-the-muhimbi-pdf-converter-services-sharepoint/>.

Configuring PDF/A

To assist with some of the underlying processing, the Muhimbi PDF Converter utilises Ghostscript in the background. If you wish to convert files to PDF/A then please make sure that Ghostscript is deployed either as part of the installation process or [installed manually](#).

Please note that a license for the [PDF Converter Professional](#) is required in addition to a valid *PDF Converter for SharePoint* or *PDF Converter Services* License in order to use this functionality.

Depending on your needs you may need to change the following settings in the Conversion Service's configuration file (see 2.6 for details about how to edit this file).

1. **Ghostscript.Path:** Leave this setting empty to auto-detect the location. If Ghostscript was deployed manually to a non-standard path then you will need to enter it here, including the name of the executable, e.g. "E:\Program Files\gs\gs9.04\bin\gswin64c.exe".
2. **PDFA.PostProcessing:** Some underlying converters can natively output files in PDF/A format, which in some rare cases may be preferential over using Muhimbi's PDF/A facility. Specify one of the following values to control this behaviour:
 - *All:* Post Process files generated by all converters, including the ones that are supposed to already support PDF/A.
 - *WhenNeeded:* Post process files for only those converters that do not support native PDF/A output.
 - *None:* Do not post process files generated by any converters. This is the default option).

Please note that these values will only be used if the output format is set to *PDF_A1B*, either in the web service call or via the global '*ConversionSettings.ForcePDFProfile*' config value. PDF_A2b and A3b output is always processed by the PDF Converter and ignores this setting.

Unless you have a good reason to change this, leave this setting alone.

3. **PDFA.RasterizeTransparentContent:** Define how transparent content is dealt with during conversion to PDF/A1b. The default setting (False) removes all transparency. If you wish to retain transparent objects then set this value to *True*, which will result in pages being rasterized resulting in considerably larger and slower PDF files. PDF/A2b and A3b natively supports transparent content and ignores this setting.

4. **ConversionSettings.ForcePDFProfile:** Override the *ConversionSettings.PDFProfile* value during conversion. Leave this setting empty to use the value specified in the web service call. Accepted values are members of the *Muhimbi.DocumentConverter.WebService.Data.PDFProfile* enum or an empty string. For example: 'PDF_1_5' (Use PDF Version 1.5) or 'PDF_A2B' (Use the PDF/A standard for long term archiving).

Don't forget to restart the *Muhimbi Conversion Service* after making changes to the configuration file.

For details about how to convert to PDF/A using the Web Services interface follow the instructions in <https://www.muhimbi.com/blog/converting-pdf-document-to-pdf-a1b-using-the-muhimbi-pdf-converter-web-service/>. Additional details can be found at <https://www.muhimbi.com/blog/set-pdf-version-enable-fast-web-views-embed-strip-fonts-using-pdf-converter-services/>.

Appendix - Unattended (un)installation

Many organisations carry out software deployments via a central software distribution service such as *Microsoft Systems Management Server*. This allows a product to be rolled out to one or more servers without user intervention.

This chapter describes how to automate deployment of the Muhimbi PDF Converter. For more details regarding the various settings, see chapter 2 *Deployment*.

Installation

The installer follows the standard convention for deploying services in unattended mode. A typical example is provided below:

```
setup.exe /quiet TARGETDIR="C:\Program Files\Muhimbi Document
Converter" USERNAME="<dom>\<user>" PASSWORD="<password>"
GRANT_LOGON_AS_SERVICE="true" DISABLE_LOOPBACK="true"
OPEN_INBOUND_PORT_ON_FIREWALL="true"
INSTALL_GHOSTSCRIPT="true" INFOPATH_CONVERTER="Latest"
INSTALL_PRINTER_DRIVER="true"
LICENSE_FILE="c:\licenses\muhimbi.txt" /! *v install.log
```

The parameters are as follows. Please note they are case sensitive.

Name	Description
TARGETDIR	The folder to deploy the Conversion Service to. Omit this parameter to use the default path, "%programfiles%\Muhimbi PDF Converter".
USERNAME	The Windows username – including domain – of the account the conversion service will run under. For local accounts use just the account name.
PASSWORD	The password associated with the account.
GRANT_LOGON_AS_SERVICE	Automatically grant <i>logon as a service</i> to the specified account.
OPEN_INBOUND_PORT_ON_FIREWALL	Configure the standard Windows Firewall to allow inbound traffic on port 41734.
DISABLE_LOOPBACK	Disable the Windows Loopback check.
INSTALL_GHOSTSCRIPT	Download and install Ghostscript.
INFOPATH_CONVERTER	The type of InfoPath converter to use, either <i>Latest</i> or <i>Legacy</i> .
INSTALL_PRINTER_DRIVER	Option to deploy the printer driver needed for the InfoPath converter.
LICENSE_FILE	The path of the Muhimbi License file to deploy. Omit this value to run in Trial mode.

Uninstallation

To carry out a silent uninstallation of the Muhimbi PDF Converter, retrieve the *UninstallPath* key from the Windows Registry, taking the version number of the installed product into account, e.g., for version 8.4.0.130:

```
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\Muhimbi PDF Converter Services 8.4.0.130
```

Please note that the product code is different for each release, but the same for all systems where that specific release has been deployed to. It is typically recommended to add the `/quiet /L*V "uninstall.log"` switches to the command. It then becomes something like the following:

```
msiexec /x {DE188642-5204-4399-B868-0DF596840121} /quiet /L*V "uninstall.log"
```

Uninstallations can also be carried out programmatically. A PowerShell Example is provided below:

```
$productname = "Muhimbi PDF Converter Services"  
$app = Get-WmiObject -Class Win32_Product -Filter "Name = '$productname'"  
$app.Uninstall()
```

Please make sure the user is 'elevated' when uninstalling programmatically.

Upgrading

The PDF Converter does not support in-place upgrades. When upgrading the software, schedule the *uninstall* command followed by the *install* command.

For more details about upgrading see 0

Upgrading from a previous version

Appendix - Advanced Deployment Scenarios

Muhimbi's range of server based PDF Conversion products have been developed with [performance, scalability and reliability](#) in mind. As a result, the software scales from the most humble '*everything on the same server*' environments to environments that deal with millions of conversions a day across a farm of servers. This appendix discusses the most common deployment scenarios.

Whenever this appendix mentions 'Server' it doesn't matter if this is a physical or virtualised server. The software does not differentiate between the two and will work fine on either type.

Introduction – Architecture

Both the PDF Converter for SharePoint and the PDF Converter Services ship with the same central conversion engine. This engine, *The Muhimbi Conversion Service*, is responsible for carrying out all the work including conversion of files, OCR, watermarking, merging, splitting and security activities. Although in case of the PDF Converter for SharePoint the front end is quite comprehensive, all it really does is prepare requests for the Conversion Service and receive responses containing new or modified documents.

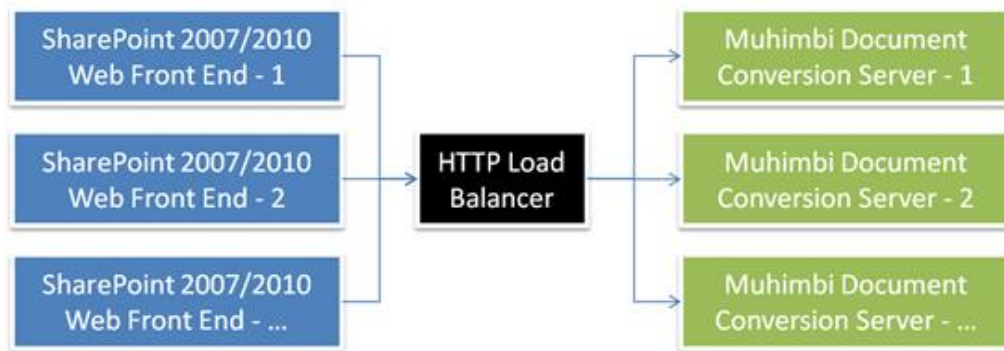
The Muhimbi Conversion Service is a standard Windows Service that starts automatically when Windows boots up and requires no user interaction or anyone to be logged in to the server console.

This Windows Service contains a WCF based Web Service that exposes functionality to any Web Services capable environment including *Java, C#, VB.NET, Documentum, SAP, Ruby, PHP* etc. Typically, when administrators think about web services, they assume that they need to host this inside a web server such as IIS or Apache. Although that may be true for many web services, Muhimbi's PDF Conversion software runs inside a *self-hosted* WCF service that does not have any external dependencies on third party web servers.

Basing the conversion service on WCF results in a lot of benefits, including:

1. No external dependencies on web servers and other 3rd party products.
2. A mature framework with support for different message and transport types, built-in security and advanced features such as [MTOM encoding](#) for large attachments.
3. And most importantly, all functionality is exposed via standard HTTP based Web Service requests.

This last point about requests being HTTP based is very important as it allows the Conversion Service to be scaled across multiple servers using standard hardware or software-based load balancers, including the free NLBS that ships with Windows. By utilising a load balanced environment, you can achieve linear scalability and automatic failover.



Example based on SharePoint Front ends, but Java and .NET deployments work the same.

For details about tuning the various options of the Muhimbi Conversion Service, and installation in general, see the *Resources* section at the end of this appendix.

Single Server Farm

The most basic configuration possible is to install everything on a single server. Just follow Chapter 2 in this Administration Guide and choose the default option to deploy both the Conversion Service and the SharePoint front-end. There is nothing else to configure and, if needed, the Web Service can be accessed on the following URL:

```
http://localhost:41734/Muhimbi.DocumentConverter.WebService/
```

Small farms with a single Conversion Server

For slightly larger deployments of 2 or more servers, but with a single conversion server, deployment is very simple as well. Let's take the following example:

- *Server 1*: A new or existing Application Server that will run the Muhimbi Conversion Service.
- *Server 2*: A server that will run either a SharePoint WFE or a custom solution.

In this particular case, it is a matter of executing the installer on *Server 1* as per the instructions in Chapter 2 of this Administration Guide.

If the PDF Converter is installed in a SharePoint environment, then the default option is to install the Conversion Service on the local machine (in this case *Server 1*) and also deploy the SharePoint front-end (WSP files) to all SharePoint servers (in this example *Server 2*).

If the decision is made to only install the Conversion Service, then the installer will need to be executed again on *Server 2* to deploy just the SharePoint front-end without deploying another instance of the Conversion Service. In that case you will be asked to enter the host name of the server that runs the conversion service, which – in this example – is the hostname of *Server 1*.

The web service URL is now as follows:

```
http://Server1:41734/Muhimbi.DocumentConverter.WebService/
```

Where *Server1* is the conversion server's host name.

Large farms with multiple conversion servers

More complex environments that require a high level of scalability and the ability to fail-over between servers usually utilise multiple front-end servers and multiple conversion servers. For example:

- *Server 1*: A new or existing Application Server that will run the Muhimbi Conversion Service.
- *Server 2*: A new or existing Application Server that will run the Muhimbi Conversion Service.
- *Server 3*: A server that will run either a SharePoint WFE or a custom solution.
- *Server 4*: A server that will run either a SharePoint WFE or a custom solution.
- Load Balancer

In this scenario the Conversion Service will need to be installed on *Server 1* and *Server 2* as per the instructions in Chapter 2 of this Administration Guide. Once installation is complete the web service can be reached on the following two URLs:

```
http://Server1:41734/Muhimbi.DocumentConverter.WebService/  
http://Server2:41734/Muhimbi.DocumentConverter.WebService/
```

Although in theory you could build functionality into your software to alternate requests between these two URLs, it is much easier and more robust to use an off-the-shelf HTTP load balancer or Windows NLBS. How this works in detail differs per load balancer, but it usually involves creating a *virtual host* and configure this virtual host to send requests to *Server1* and *Server2*. In this example we assume that this virtual host is named *LoadBalancer*, resulting in the following Web Service URL:

```
http://LoadBalancer:41734/Muhimbi.DocumentConverter.WebService/
```

In case of a SharePoint deployment execute the installer on one of the Web Front End servers (*Server 3* or *4*, it doesn't matter which) and choose the option to *Install the SharePoint front-end on the entire farm*.

You will be asked to enter the network name or ip-address of the server running the Conversion Service, which in this example is *LoadBalancer*. The installer will check that a Conversion Service is listening at the specified address, so make sure the Conversion Service and load balancer are configured before deploying the SharePoint front-end.

If you have multiple SharePoint Web Front End servers, it may be tempting to install a copy of the Conversion Service on each WFE. Although this is a supported scenario, we recommend deploying the Conversion Service on separate Application Servers to make sure that all the WFE's resources are available for running SharePoint, which can be quite resource hungry by itself.

Licensing

Please make sure that licenses are in place for the correct number of servers. Muhimbi's software is licensed on a 'per server' basis. This means that you need a license for every server that runs our software including all Web Front End servers in your SharePoint farm and all servers running the conversion service.

For more details see this [Knowledge Base Article](#).

Resources

For more details about deployment and configuration see the following resources:

- [Deployment related topics in the Knowledge base](#).
- [Hardening / Securing the server that runs the Conversion Service](#).

Appendix - Creating Custom Converters

The *Muhimbi PDF Conversion Service* allows custom converters to be added with relative ease. This is useful for converting file types specific to your organisation or for file types that have not (yet) been implemented by Muhimbi in the main product.

The following example describes how to replace the existing MS-Word converter, which relies on MS-Word being present, with a simple third party converter that works well enough for simple documents. Some programming knowledge is required.

It is also possible to use 3rd party executables as custom converters without any programming, see Appendix - Invoke 3rd party Converters.

The steps are as follows:

1. Create a new Visual Studio project, select *Class Library (C#, .net 4.0)* as the template and give the project an appropriate name, e.g. *CustomConverters*.
2. Add references to the following DLLs. They are located in the folder the Muhimbi Document Conversion Service has been installed in, usually *c:\Program Files\Muhimbi....*
 - Muhimbi.dll
 - Muhimbi.DocumentConverter.WebService.dll
 - Muhimbi.DocumentConverter.WebService.Data.dll
 - System.Runtime.Serialization (Add reference from the .NET tab)
3. Change your project's Assembly name and default namespace to something sensible, e.g. *Muhimbi.DocumentConverter.WebService.CustomConverters*. This can be done by *right-clicking* on your project and selecting *Properties*. The *Application* tab allows these settings to be changed.
4. Delete the automatically generated *class1.cs* file and add a new class named *WordConverter.cs*. **Make sure the class definition is *public*.**
5. Inherit *Muhimbi.DocumentConverter.WebService.AbstractDocumentConverter* in the *WordConverter* class and implement the members (*right-click* on the base class name and select *Implement Abstract Class*).
6. Add the following 2 constructors and make sure they call the base constructors.

```
public WordConverter() : base()
{

public WordConverter(Stream sourceFile, OpenOptions openOptions,
                    ConversionSettings conversionSettings)
    : base(sourceFile, openOptions, conversionSettings)
{
```


- Next up, we need to implement the *RunDiagnostics* method. This method is normally used to carry out an internal end-to-end conversion to verify that the converter and all prerequisites have been installed correctly. In this test we simply return a new *DiagnosticResultItem* with the *Valid* property set to *true*.

```
public override DiagnosticResultItem RunDiagnostics()
{
    DiagnosticResultItem dri = new DiagnosticResultItem();
    dri.Valid = true;
    return dri;
}
```

- If we need to look further than just the file extension to determine the file type then we can optionally override the *CanConvert* method and look inside the stream (available in the *_sourceFile* member variable). This is not necessary for this sample converter, but an example is provided below.

```
public override bool CanConvert(string[] fileExtensions)
{
    // ** Do we know anything about this extension
    if (base.CanConvert(fileExtensions) == false)
        return false;

    // ** Investigate in more detail
    ...implement your own...
}
```

- The next and final method to implement is the actual *Convert* method, which is where all the magic happens. As it is not feasible to develop an MS-Word to PDF converter from scratch, the sensible approach to take is to use a third party library such as [SyncFusion DocIO](#) or [Aspose.Words](#) (download the archive that contains just the DLLs). In this example we are going to use Aspose's library for processing MS-Word files. It is not perfect, but for some documents such as forms and simple text documents it works very well.

Copy *Aspose.Words.dll* into the project directory and add a reference to it. Copy the following code into the *WordConverter* class.

```
public override Stream Convert()
{
    try
    {
        // ** Validate as certain options are not supported by this converter
        if (_openOptions.AllowMacros != MacroSecurityOption.None)
            Logger.Warn("Macros are not supported by this converter.");

        // ** Set the licences for Aspose.Words.
        Aspose.Words.License wordLicence = new Aspose.Words.License();
    }
}
```

```
//wordLicence.SetLicence("Enter your license in here.");

Document asposeDocument = new Document(_sourceFile, null,
                                         LoadFormat.Auto, _openOptions.Password);

// ** Do we need to refresh the fields etc?
if (_openOptions.RefreshContent == true)
    asposeDocument.Range.UpdateFields();

// ** Convert the Document to PDF and save it as a memory stream.
if (_conversionSettings.Format == OutputFormat.PDF)
{
    MemoryStream convertedStream = new MemoryStream();
    PdfOptions options = new PdfOptions();
    // ** Specify the PDF Profile
    if (_conversionSettings.PDFProfile == PDFProfile.PDF_1_5)
        options.Compliance = PdfCompliance.Pdf15;
    else
        options.Compliance = PdfCompliance.PdfA1b;

    // ** How to deal with bookmarks
    if (_conversionSettings.GenerateBookmarks ==
        BookmarkGenerationOption.Automatic)
        options.HeadingsOutlineLevels = 9;
    else if (_conversionSettings.GenerateBookmarks ==
        BookmarkGenerationOption.Custom)
        options.BookmarksOutlineLevel = 9;

    // ** Correct the start and end pages if needed
    int startPage = _conversionSettings.StartPage != 0 ?
        _conversionSettings.StartPage - 1 : 0;
    int pageCount = asposeDocument.PageCount - startPage;
    if (_conversionSettings.EndPage != 0)
        pageCount = Math.Min(_conversionSettings.EndPage - startPage,
                             pageCount);

    // ** Carry out the actual conversion
    asposeDocument.SaveToPdf(startPage, pageCount, convertedStream,
                             options);
    return convertedStream;
}
else
{
    throw new NotSupportedException("Outputformat '" +
        _conversionSettings.Format +
        "' not supported by this Converter.");
}
}
catch (UnsupportedFileFormatException ex)
{
    throw new WebServiceInternalException(
        WebServiceExceptionType.FileFormatNotSupported, ex.Message);
}
catch (Exception ex)
{
    string message = "An error occurred while converting a file";
    if (_openOptions != null && _openOptions.OriginalFileName != null)
        message += " - " + _openOptions.OriginalFileName;
    Logger.Error(message, ex);
    throw new WebServiceInternalException(
        WebServiceExceptionType.InternalError, message);
}
}
```

10. Compile the project and copy the output DLL as well as *Aspose.Words.dll* to the directory that holds the *Muhimbi Document Conversion Service*.

11. Edit the service's config file and make the following changes:

- If the file extensions for the new converter are currently handled by a different converter then remove these extensions from the existing converter.
- Add the definition for the new converter to the config file as per the following example. For details see 0

- *Adding custom converters / changing file extensions.*

```
<add key="CustomWordConverter"  
description="Custom MS-Word Converter"  
fidelity="Full"  
supportedExtensions="doc,docx"  
type="Muhimbi.DocumentConverter.WebService.CustomConverters.WordConverter,  
Muhimbi.DocumentConverter.WebService.CustomConverters,  
Version=1.0.0.0, Culture=neutral, PublicKeyToken=null" />
```

12. Restart the service to activate the changes.

```
Net stop "Muhimbi Document Converter Service"  
Net start "Muhimbi Document Converter Service"
```

13. Finally test if everything is working correctly, either from:

- **SharePoint:** Open *Central Administration / Application Management / Muhimbi Document Converter Settings* (In *SharePoint 2010/2013* this screen is located in *Central Admin / General Application Settings / Muhimbi Document Converter Settings*), verify that the new converter is added to the list, check the tick box and click *Validate Settings*. If everything is working correctly then don't forget to save the changes using the *OK* button.
- **Winforms Diagnostics Tool:** Launch the *Diagnostics Tool* from the Windows Start Menu, navigate to the *WS Diagnose* Tab and click the *Request Diagnostics* button. Verify the new converter is listed and *Valid = True*.

Any errors are logged to the Windows Application Event Log.

Congratulations, you have created your first custom converter. Source code for *WordConverter.cs* and the latest version of this tutorial can be found on the [Muhimbi website](#).

Exception handling

Although you can let exceptions bubble up, we recommend catching any exceptions, inspecting the root cause of the problem and then throwing a specific *WebServiceInternalException* using one of the following exception types.

```
public enum WebServiceExceptionType
{
    /// <summary>
    /// Unknown error
    /// </summary>
    Unknown,
    /// <summary>
    /// File format not supported
    /// </summary>
    FileFormatNotSupported,
    /// <summary>
    /// File corrupt
    /// </summary>
    CorruptDocument,
    /// <summary>
    /// An error occurred while opening the file
    /// </summary>
    ErrorOpeningFile,
    /// <summary>
    /// Conversion process timeout
    /// </summary>
    ConversionTimeOut,
    /// <summary>
    /// Application hang. Can happen when document is password protected
    /// </summary>
    ConverterNotResponding,
    /// <summary>
    /// The underlying converter has not been installed or not correctly installed.
    /// </summary>
    ConverterNotInstalled,
    /// <summary>
    /// Internal Validation (Should only happen during development)
    /// </summary>
    InternalError,
    /// <summary>
    /// The specified output format is not supported (e.g. XPS output for HTML
    /// conversion)</summary>
    OutputFormatNotSupported,
    /// <summary>
    /// Configuration file is invalid, e.g. no steps defined in a multi step converter
    /// </summary>
    ConfigurationError,
    /// <summary>
    /// The trial has expired (e.g. when processing non-PDF files)
    /// </summary>
    TrialExpired,
    /// <summary>
    /// Problem in external dependency, e.g. Ghostscript not installed or
    /// wrong version.
    /// </summary>
    ExternalDependencyError
}
```

Appendix - Invoke 3rd party Converters

The PDF Converter has had the ability to add custom converters for a while (See *Appendix - Creating Custom Converters*). However, although these plug-ins work very well, if you are not a developer, or you are not familiar with .net based development, then implementing a custom converter may be less than trivial.

As of version 6.1 it is possible to use existing command line based 3rd party conversion engines using the new *Command Line Converter*.

The latest version, and further details, of this topic can be found [in this post on our Blog](#). A number of examples can be found in our [Knowledge Base](#).

The Command Line Converter is very simple to setup as you can see in the following config file fragment for Siemens Teamcenter:

```
<add key="CommandLineConverter"
description="Converts using 3rd party executables"
fidelity="Full"
supportedExtensions="hp,hpg,hpgl,hpgl2,plt,cgm,m1r"
supportedOutputFormats="pdf"
type="Muhimbi.DocumentConverter.WebService.CommandLineConverter,
Muhimbi.DocumentConverter.WebService, Version=1.0.1.1, Culture=neutral,
PublicKeyToken=c9db4759c9eaad12"
parameter="c:\splm\vis20072\VVCP\prepare.exe | -PDF {0} -combine -page all
-out {1} -overwrite -size {Parameter1}"/>
```

The parameters are as follows:

1. **key:** Give the converter a unique name in case multiple Command Line Converters are in use. Otherwise accept the default value.
2. **description:** The description of the converter exposed via the web services API. As, for example, displayed in our SharePoint Central Administration screen.
3. **supportedExtensions:** A list of input file formats supported by this converter.
4. **supportedOutputFormats:** The file formats the converter can generate. This is not limited to just PDF as our software fully supports cross-conversion.
5. **type:** Do not touch.
6. **parameter:** The location of the 3rd party converter and the arguments to send to it. Note that these 2 values are separated by a vertical pipe character '|'. The arguments section supports the following parameters:
 - **{0}:** This will automatically be replaced with the full path to the input file.
 - **{1}:** This will automatically be replaced with the full path and filename where the output file should be generated.

- **{Parameter1}, {Parameter2}...{Parameter10}**: Optional parameters that can be passed in via the web services interface using *ConverterSpecificSettings_CommandLineConverter*. Use this to pass proprietary information to the third party converter such as page size or special processing instructions.

Once everything has been configured, all mapped file formats will be picked up automatically and treated exactly the same as all other file formats supported by the Muhimbi PDF Converter.

Appendix - Relevant articles on the Muhimbi Blog

The [Muhimbi Blog](#) is updated frequently with new articles related to this product. The following posts are relevant to readers of this Administration Guide.

- [Troubleshooting steps for the PDF Converter.](#)
- [Troubleshooting InfoPath to PDF Conversion / Document Converter Architecture.](#)
- [Performance metrics for the Muhimbi PDF Converter.](#)
- [Enable SSL/HTTPS Communication in the Muhimbi Conversion Service.](#)
- [Convert Office files to PDF Format from .NET using a Web Service](#)
- [Invoking the PDF Converter Web Service from Visual Studio 2005 using vb.net](#)
- [Convert files to PDF Format from Java using Web Services](#) (WSImport)
- [Convert files to PDF Format from Java using Web Services](#) (Axis2)
- [Convert files to PDF Format from PHP using a Web Services based interface](#)
- [Convert files to PDF Format from Ruby using a Web Services based interface](#)
- [Set PDF Version, enable Fast Web Views, embed / strip fonts](#)
- [Specifying PDF Viewer Preferences](#)
- [Converting and merging multiple files using Web Services](#) (.NET)
- [Converting and merging multiple files using Web Services](#) (Java)
- [Splitting PDF Files using the PDF Converter Web Service and .NET / C#](#)
- [Adding custom Converters to Muhimbi's range of PDF Conversion products](#)
- [Using the awesome new watermarking features of the Muhimbi PDF Converter Services](#)
- [Using the PDF Watermarking features from Java based environments](#)
- [Converting InfoPath forms including all attachments to a single PDF file](#)
- [Controlling which views to export to PDF format in InfoPath](#)
- [Dealing with hyperlinks when converting InfoPath files to PDF format](#)
- [Convert InfoPath to MS-Word, Excel, XPS and PDF](#)
- [Convert between document types \(xls to xlsx, doc to docx, xls to doc\)](#)
- [Convert HTML pages to PDF format](#)
- [Convert AutoCAD \(DXF, DWG\) files to PDF](#)
- [Using Third Party AutoCAD Converters](#)
- [Convert MicroStation DGN files to PDF](#)
- [Convert TIFF files to PDF](#)
- [Convert any file format to TIFF](#)
- [Convert PCL files to PDF](#)
- [Convert legacy files formats \(Lotus Manuscript, DisplayWrite, Wordstar etc\)](#)
- [Convert Outlook MSG files to PDF including all attachments](#)
- [PDF/A Support in the Muhimbi PDF Converter Services & SharePoint](#)
- [OCR Facilities provided by Muhimbi's server based PDF Conversion products](#)
- [Converting PDF document to PDF/A1b using a Web Service](#)
- [Reduce PDF Converter Web Service message size using MTOM](#)

A number of articles written for SharePoint based environments are available as well.

- [Tuning SharePoint's workflow engine](#)
- [Using the PDF Converter from a SharePoint Designer workflow](#)
- [Convert and merge multiple PDF files using a SharePoint Designer workflow](#)
- [Converting multiple SharePoint files to PDF Format using Nintex workflow](#)
- [Watermark PDFs using Nintex Workflow](#)
- [Secure PDFs using Nintex Workflow](#)
- [Convert and Merge PDFs using Nintex Workflow](#)
- [Convert HTML to PDF using Nintex Workflow](#)
- [Inserting SharePoint List data into a PDF document using a workflow](#)
- [Configure PDF Security from a SharePoint Designer Workflow](#)
- [Watermarking features of the Muhimbi PDF Converter for SharePoint](#)
- [Applying user specific watermarks when a PDF document is opened](#)
- [Merging dynamic data into watermarks using the PDF Converter for SharePoint](#)
- [Use SharePoint Workflows to inject JavaScript into PDFs and print the 'open date'](#)
- [Automatically convert files to PDF using an e-mail enabled Document Library](#)
- [Batch print InfoPath Forms using the PDF Converter for SharePoint](#)
- [Using SharePoint Forms Services to convert InfoPath forms to PDF format](#)
- [Convert SharePoint HTML pages to PDF format](#)
- [Converting SharePoint Lists to PDF format using a SharePoint Designer Workflow](#)
- [Convert and merge files to PDF using the SharePoint User Interface](#)

Appendix - Licensing

All Muhimbi products are licensed in a way that allows maximum flexibility. Please familiarise yourself with the licensing agreement, particularly section 3 – *Grant of License*, before purchasing our software.

For details see:

1. [License Agreement](#)
2. [Details about pricing & licensing](#)

In summary we support the following license types.

1. **Free evaluation version:** If the software is installed without a license then you are using the evaluation version. The software is fully functional without any time limits, but a small evaluation watermark will be displayed in each processed document. It is not permitted to use the evaluation version in production environments. Support is provided using any of the means in the Support area on our website.
2. **Basic License:** Starter edition for environments consisting of a single conversion server.
3. **Small Farm License:** Covers up to 3 individual servers, either stand-alone or load balanced.
4. **Enterprise License:** Covers an unlimited number of Servers, Developers and Users in a single legal entity.
5. **OEM License:** If you wish to bundle our software with your own solution and redistribute it to 3rd parties then you require an OEM License. Please read the details in the Software License Agreement for more information.

Note that you are not allowed to use our Products to develop derived works that offer similar functionality as the Product or expose the features of the Product for use by an unlicensed third party unless agreed with Muhimbi. From a licensing perspective embedding our software in a SAAS based solution is considered redistributing our software and requires an OEM license.

Please note that some older license types have been discontinued. However, these are still valid for those customers that have purchased them in the past. Please see the License Agreement for details about these old style licenses.

PDF Converter license types

The PDF Converter for SharePoint license is limited to use in SharePoint environments only. If you wish to invoke the PDF Conversion Service from a non-SharePoint based environment, e.g. Java, .NET or any other Web Services capable system then you will need to purchase a license for the PDF Converter Services.

The PDF Converter Professional license is an add-on that adds additional functionality to either the PDF Converter for SharePoint or the PDF Converter Services.

This functionality, e.g. PDF/A post-processing and OCR, is usually associated with more complex environments and has therefore been separated from the main product.

Please note that the PDF Converter Professional is a license that must be applied alongside a valid license of the PDF Converter for SharePoint or PDF Converter services. A separate download of the Professional version of the software is not needed, the license unlocks all functionality.