



---

# dox42 Server

## V3.5.1-4.6

---

Documentation

---

---

# Summary

This document explains the service interface as well as the configuration for the dox42 server.

## Content

Summary.....	2
Content.....	2
Document details.....	3
Service interface.....	4
SOAP Interface.....	4
REST Interface.....	5
Service-Header.....	6
Azure Active Directory Header.....	6
OutputActions.....	7
SaveAction.....	8
SharepointAction.....	9
Set special SharePoint and Managed Metadata Fields.....	11
EmailAction.....	12
ReturnAction.....	17
dox42LIVEAction.....	18
AzureBlobStorageAction.....	19
Configuration parameters.....	21
CORS Configuration.....	24
Azure Active Directory Access.....	24
Custom Error Messages Sample.....	24
Hyphenation Dictionaries Sample.....	25
Maximum size of request or time-out.....	25
License Activation.....	25
Installation.....	27
Upgrade.....	29
Support.....	30

## Document details

Version: dox42 Server V3.5.1-4.6

Authors: Christian Bauer, Lisa Pulsinger

Date: 15 February 2024

## Service interface

The dox42 server is accessible via a SOAP and a REST Webservice interface.

### Operation: GenerateDocument/GenerateSlides

The operation can be operated with the **following parameters**:

- › **DocTemplate** (String)  
path to the document template (e.g. C:\doctemplates\solar\_subsidy.docx, C:\doctemplates\salesreport.pptx)
- › **InputParams** (Key-Value String/String)  
list of the input parameters and values:
  - › **ParamName:** name of the parameter (not case-sensitive)
  - › **Value:** value of the parameter
- › **PostGenActions**  
list of PostGenerateActions to process after generating the document:
  - › **ActionName:** name of the action (e.g. SaveAction), not case-sensitive
  - › **Params:** parameter of the action (like InputParams)

### Operation: GenerateSpreadSheet

The operation can be operated with the **following parameters**:

- › **DataMap** (String)  
path to the data map (e.g. C:\doctemplates\salesreport.dm)
- › **InputParams** (Key-Value String/String)  
list of the input parameters and values:
  - › **ParamName:** name of the parameter (not case-sensitive)
  - › **Value:** value of the parameter
- › **PostGenActions**  
list of PostGenerateActions to process after generating the document:
  - › **ActionName:** name of the action (e.g. SaveAction), not case-sensitive
  - › **Params:** parameter of the action (like InputParams)

The paramameters of all individual OutputActions are described in this document.

## SOAP Interface

URL: <http://www.YourDomain.com/dox42service.asmx>

WSDL: <http://www.YourDomain.com/dox42service.asmx?WSDL>

## REST Interface

URL: <http://www.YourDomain.com/dox42RestService.ashx>

The parameters and output actions for the REST and the SOAP interface are identical. The REST Interface uses the following URL-parameter syntax. Since version 3.6.3 you can also pass all parameter via POST:

Parametername	Erfordernis	Auswirkung
<b>Operation</b>	mandatory  GenerateDocument   GenerateSpreadSheet   GenerateSlides	The operation to call
<b>DocTemplate</b>	mandatory if Operation=GenerateDocument or GenerateSlides	Path of the document template
<b>DataMap</b>	mandatory if Operation= GenerateSpreadSheet	Path of the data map
<b>InputParam.</b> <ParameterName>	For each input parameter	Values for input parameters, e.g.:  InputParam.CustomerID=42
<OutputAction>. <ParameterName>	For each parameter of each OutputAction	Values for OutputAction- Parameters, z.B.: ReturnAction.Format=pdf
<b>ReturnAction.disp</b>	optional  Inline   attachment	Controls the how the web browser handles the returned document
<b>ReturnAction.Mode</b>	optional  File (Standard)   Text	Use Text to render the content of the generated document directly in the browser as text. This text may be HTML.
<b>ReturnAction.fileName</b>	optional, only if ReturnAction.disp = attachment	Name of the returned document
<b>RedirectURL</b>	Optional, only without ReturnAction	Redirect to this page after generation

<b>ReturnSavedFileInfo</b>	Optional (since 4.5)	If true, REST Calls return information about saved files as JSON.
----------------------------	----------------------	-------------------------------------------------------------------

**Caution:** Use URL-Encoding! ([http://www.w3schools.com/tags/ref\\_urlencode.asp](http://www.w3schools.com/tags/ref_urlencode.asp)):

„Customer.Name=Müller“ => „ Customer.Name=M%C3%BCller “

Use the syntax <%datasource.datafield%> as a place holder for dynamic values in your calls.

#### Sample for a REST call:

[http://www.YourDomain.com/dox42RestService.ashx?Operation=GenerateDocument&DocTemplate=%7e%5cTemplates%5cSales\\_Report%5cSales\\_Report.doc &InputParam.SelectEmployee=5&InputParam.SelectYear=2012&ReturnAction.format=pdf&ReturnAction.fileName=SalesReport <%Employees.Name%>.pdf&ReturnAction.disp=attachment](http://www.YourDomain.com/dox42RestService.ashx?Operation=GenerateDocument&DocTemplate=%7e%5cTemplates%5cSales_Report%5cSales_Report.doc &InputParam.SelectEmployee=5&InputParam.SelectYear=2012&ReturnAction.format=pdf&ReturnAction.fileName=SalesReport <%Employees.Name%>.pdf&ReturnAction.disp=attachment)

Your dox42 calls can easily be configured with our dox42 Server Designer application, which you can [download](#) from our website.

#### HTTP Request-Body

In case you need to transfer lots of data to the dox42 Server you better do that as POST parameters or in the Request-Body (since version 4.0). To extract the Request-Body you need to define an InputParameter of the name „RequestBody“ in your DataMap. The data from the HTTP Request-Body will be stored to this InputParameter and can be processed using a XML/JSON data source for example.

## Service-Header

A service header secures the dox42 SOAP service. This header contains the user name and the password. Use web.config to configure the utilisation of the service header.

## Azure Active Directory Header

The dox42 Service can process an Authorization Header, containing an Azure Active Directory Token (since version 4.0).

Javascript Example:

```
xhttp.setRequestHeader("Authorization", 'Bearer ' + token);
```

## HTTP Metadata Header and Base64 Metadata Header

Since V 4.6. you can send a custom string containing datafields in a dedicated dox42 HTTP Header “dox42MetaData”. Here is an example of such a string, the header will be returned with datafields replaced:

```
{"sender": "<%User.mail%>"
```

```
"receiver": "<%Customer.mail%>"
```

```
"subject": "Here is your document. <%quote.number%>"}
```

Since Version 4.6.4. you can also use “dox42MetaDataBeaderBase64” to send base64 encoded metadata in your header.

## OutputActions

The product dox42 server contains the following OutputActions:

- › **SaveAction:** Saves the generated document to a file folder
- › **SharepointAction:** Stores the generated document to a Sharepoint document library
- › **EmailAction:** Sends the generated document via email
- › **ReturnAction:** Returns the generated document via the WebServices interface
- › **PrintAction:** Print generated document
- › **dox42LiveAction:** Used for calls with the product dox42 Live
- › **AzureBlobStorageAction:** Stores the generated document on an Azure Blob Storage
- › **DropBoxAction:** Stores the generated document on Dropbox. This dox42 Output Action is described in the dox42 Dropbox Action documentation.
- › **DocuSignAction:** Sends the generated document to DocuSign for electronic signature. This dox42 Output action is described in the dox42 DocuSign documentation
- › **SQLAction:** Stores the generated document in a SQL database. This output action is described in the dox42 SQL Output Action.
- › **CustomOutputAction:** You can implement and use your own dox42 OutputAction. Custom Output Actions have to be registered in the web.config (analog Custom Data Sources) and use the following interface:  
dox42CustomOutputActionKit. [ICustomOutputAction](#)

All additional documentations mentioned above can be found here:

[https://www.dox42.com/Resources?filter\[\]=dokumentation&](https://www.dox42.com/Resources?filter[]=dokumentation&)

Since version 4.6. the optional parameter “Condition” is available for your dox42 Output Actions. If false, the selected output action will not be performed.

In the following, the parameters of the OutputActions are described. None of the parameter names are case-sensitive. All parameter names may contain data fields (for exceptions see table). Example:  
FileName = “Application\_<%Citizen.Name%>.pdf”

## SaveAction

Parameter name	Requirement	Effect
<b>FileName</b>	mandatory	<p>Defines the name of the document, which it will be saved as. e.g. C:\Documents\SalesReport.pdf</p> <p>The extension will define the file format (.docx, .xlsx, .pptx, .pdf, .html, ...). See all output formats below.</p>
<b>SpecialFormat</b> (dox42ForDocuments only)	optional	<p>When generating a PDF, the format may be specified: PDFA1b and PDFForm. For PDFForm Form fields are preserved in the PDF.</p> <p>For HTML, you may use these options:</p> <p><b>HTMLCode</b> which will cause the text of the dox42 template to be interpreted in HTML code.</p> <p><b>HTMLZIP</b> The HTML -file along with the contained images will be packaged in a ZIP-file. (since version 3.6.3).</p> <p>For text files (*.txt) you can set an alternate extension. That is useful for generating meta data files for archiving-systems. Example: FileName = Order_&lt;%Cust.OderNo%&gt;.txt SpecialFormat = jpl Result: Order_42.jpl (since version 3.5)</p>
<b>metadatafilepath</b>	optional	<p>Generates an additional file with meta data information. E.g.: C:/dox42Server/generateddocuments/metadata.xml</p>
<b>metadatafilecontent</b>	optional	<p>Defines the content that is written to the metadata file.</p>
<b>Sign</b>	optional	<p>Value: true</p> <p>The format PDF may bear a digital signature (dox42ForDocuments only). Certificate and corresponding password have to be defined in the web.config.</p>
<b>Condition</b>	Optional (since V. 4.6)	<p>If "false", the SaveAction will not be performed</p>

### Possible Output Formats:

Word: doc, docx, docm, dot, dotx, pdf, html, txt, jpg, png, tiff, bmp, rtf, odt  
 Excel: xls, xlsx, xlsxm, xlt, xltx, xltm, xlsb, txt, csv, html, pdf, xml, tiff, ods  
 PowerPoint: pptx, pps, pdf, tiff



## SharepointAction

Parameter name	Requirement	Effect
<b>Site</b>	mandatory	The Sharepoint site e.g. <a href="http://mysite.com">http://mysite.com</a>
<b>Library</b>	mandatory	The Sharepoint document library root folder name
<b>FileName</b>	mandatory	Defines the name of the document, which it will be saved as. e.g. SalesReport.pdf  The extension will define the file format (.docx, .xlsx, .pptx, .pdf, .html, ...).
<b>Folder</b>	optional	Saves the document to the specified folder.
<b>Rootfolder</b>	Optional (since Version 4.4.1.7)	Define the internal name of your SharePoint library here.
<b>SpecialFormat</b> (dox42ForDocuments only)	optional	When generating a PDF, the format may be specified: PDFA1b and PDFForm. For PDFForm Form fields are preserved in the PDF.  For HTML, you may use the option: HTMLCode which will cause the text of the generated document to be interpreted in HTML code.
<b>Sign</b>	optional	Value: true  The format PDF may bear a digital signature (dox42ForDocuments only). Certificate and corresponding password have to be defined in the web.config.
<b>Field1.Name</b> <b>Field1.Value</b>	optional	Fields allow you to write values to columns in the Sharepoint document library. You can create multiple Fields: Field1, Field2, ... Use the SharePoint internal field name to identify the field. Tipp: You may find out the internal field name by opening the list in the dox42 SharePoint data source and check "Use Internal Field Names".
<b>ContentType</b>	optional	Set this Content Type for the item

<b>DocumentDataField</b>	optional	<p>If this field is assigned, the generated document will not be saved in the document library, but the document in the stated dox42 Datafield will be loaded (e.g. Data.MyDocument) and saved in the Sharepoint document library.</p> <p>The DocumentDataField can contain a Path, an URL or a binary Document.</p>
<b>CheckInMessage</b>	optional	<p>If this Field is assigned, the server will check in the document after saving with the stated message. If a document of the same title is already existing this document will be checked out automatically</p>
<b>Username</b>	optional	<p>If not provided, the service account of the dox42 Server will be used for SharePoint authentication.</p>
<b>Password</b>	optional	
<b>SharePoint_online</b>	optional	<p>true/false Set this flag for SharePoint online (Office 365). Alternative to <b>Azure_AD</b></p>
<b>SharePoint_FBA</b>	optional	<p>true/false Set this flag for SharePoint sites with forms based authentication.</p>
<b>Azure_AD</b>	optional	<p>true/false The Request must include an Authorization Header containing Azure Active Directory Token - alternative to <b>SharePoint_online</b>. (Since version 4.0).</p>
<b>retry_count</b>	optional	<p>A number <math>\geq 0</math> Defines that the SharePointAction should retry to save the document in case of error (e.g. timeout) and specifies the number of retries. Default = 0 (no retry). (Since version 4.1.3).</p>
<b>Condition</b>	Optional (since V. 4.6)	<p>If "false", the SharePointAction will not be performed</p>

## Set special SharePoint and Managed Metadata Fields

For a choice field simply set the value.

To set a multi value choice field use “;#” to separate the values, e.g.: “red;#green;#yellow”.

For a Lookup/Person/Group-field set the ID.

To set a multi value Lookup/Person/Group-field, you need to set the ID and the value like „ID;#Value“, e.g.: „3;#Hawaii;#9;#Vienna;#13;Melbourne“. Actually only the IDs are used to set the field, the values are ignored (but may not be omitted).

## EmailAction

Parameter name	Requirement	Effect
<b>Mode</b>	optional  <b>Possible Values:</b>  TxtEMail HtmlEMail AttachmentTxtBody Attachment (default)	Allows to configure whether the generated document should be used as the body of a txt- or html-mail or attached to the email.  Use TxtEMail and HtmlEMail only with GenerateDocument.
<b>FileName</b>	mandatory if mode = Attachment, otherwise ignored	Name of the attachment  The extension will define the file format (.docx, .xlsx, .pptx, .pdf, .html, ...).
<b>SpecialFormat</b> (dox42ForDocuments only)	optional	When generating a PDF, the format may be specified: PDFA1b and PDFForm. For PDFForm Form fields are preserved in the PDF.  For HTML, you may use the option: HTMLCode which will cause the text of the generated document to be interpreted in HTML code.
<b>Sign</b>	optional	Value: true  The format PDF may bear a digital signature (dox42ForDocuments only). Certificate and corresponding password have to be defined in the web.config.
<b>Receiver</b>	mandatory	Receivers email addresses  Separate multiple addresses with a comma or semicolon.
<b>ReceiverCC</b>	optional	Carbon copy receivers
<b>ReceiverBCC</b>	optional	Blind carbon copy receivers
<b>Sender</b>	optional	Sender's email address  If no email address is specified, the address stated in the web.config will be used.

Parameter name	Requirement	Effect
<b>MailServerUsername</b>	optional	May be specified along with Sender if a different login for the mail server is required. Since Version 3.6.2.4
<b>SenderPassWord</b>	mandatory if Sender or MailServerUsername is specified, else: omit	Sender's password for his email account or MailServerUsername on the email server, configured in the web.config.
<b>Header</b>	mandatory	Email subject
<b>Body</b>	optional	Email text (mode = Attachment only)
<b>HtmlBody</b>	optional	Email text as HTML (mode = Attachment only)
<b>BodyTemplate</b>	optional	A dox42 Word Template to generate the email body. The template must use the same Datamap as the attachment. (mode = Attachment only) Since V 3.5.1 Mode = AttachmentTxtBody generates the body template in Txt format. (Since V. 4.4.1.5)
<b>StaticAttachment1.Path</b>  <b>StaticAttachment1.Base64</b>  <b>StaticAttachment1.Name</b>	optional (or use Base64) e.g. C:\Images\MyImage.jpg  pg  optional (or use Path) Base64 encoded attachment  mandatory if Base64 is used e.g. MyImage.jpg	Allows sending for static attachments. It is possible to send any number of attachments, like: StaticAttachment2... StaticAttachment3... You may specify either the path of a static file, or pass the attachment base64 encoded.
<b>MailServer</b>	optional	Use this parameter if you want to use a different mail server than configured in your web.config.
<b>MailServerPort</b>	Optional Since version 3.4.3.10	Change the SMTP Mail Server Port. You may want to use the standard ports 25 or 587.

Parameter name	Requirement	Effect
<b>SSL</b>	Optional Since version 3.5	true or false
<b>SaveMSGFile</b>	Optional Since version 3.5.0.19	File path to save the sent mail as msg-file, e.g. C:\Windows\Temp\Order_<%Customer.ID%>.msg
<b>SendMode</b>	Optional Since Version 3.6	send or save <b>send:</b> Default, send mail immediately <b>save:</b> Don't send mail, but save the mail as draft to the path defined in SaveMSGFile. The parameter SaveMSGFile must be specified. <b>return:</b> Don't send mail, but return the mail as draft in the ReturnMessage. Since Version 3.6.3.
<b>MailMethod</b>	Optional Since Version 4.3.3	Use this parameter to choose a specific method to send your dox42 emails. Possible values: exchangeonline, aspose, mailkit, onmailkiterrortryaspose, onasposeerrortrymailkit  <b>You need to use exchangeonline if you send emails via Microsoft Exchange Online.</b>  <b>onasposeerrortrymailkit:</b> When the Aspose Mail Method throws an error, the mailkit method is used to send dox42 emails.  When adding the MailMethod parameter you may also want to add the MailServerPort parameter, if you have not done so yet in your web.config.
<b>AzureADAppId</b>	Optional (Since Version 4.4.1.7)	Application ID of the app registration that has the necessary API permissions configured to send mails via Exchange Online (can be configured in the web.config/MAUI as well)
<b>AzureADTenantId</b>	Optional (Since Version 4.4.1.7)	Tenant ID in which the above app registration is configured to send mails via Exchange Online (can be configured in the web.config/MAUI as well)
<b>AzureADClientSecret</b>	Optional (Since Version 4.4.1.7)	Clientsecret that is associated with the App ID to send mails via Exchange Online (can be configured in the web.config/MAUI as well)

Parameter name	Requirement	Effect
<b>ExchangeOnlineUrl</b>	Optional (Since Version 4.4.1.7)	Comes with the default value <a href="https://outlook.office365.com/EWS/Exchange.asmx">https://outlook.office365.com/EWS/Exchange.asmx</a> and can be left empty
<b>Condition</b>	Optional (since V. 4.6)	If "false", the EmailAction will not be performed

To use an alias with an email address please use this syntax (since 3.6.2.5):

MyCompany Sales [sales@mycompany.com](mailto:sales@mycompany.com)

MyCompany Sales <[sales@mycompany.com](mailto:sales@mycompany.com)> , would work as well, but might cause an Error if used with the REST Service.

### Exchange Online

To send emails via Microsoft Exchange Online, you need to register an application in Azure Active Directory. Further information on the Azure AD App registration can be found in our SharePoint documentation: [https://www.dox42.com/Resources?filter\[\]=dokumentation&](https://www.dox42.com/Resources?filter[]=dokumentation&)

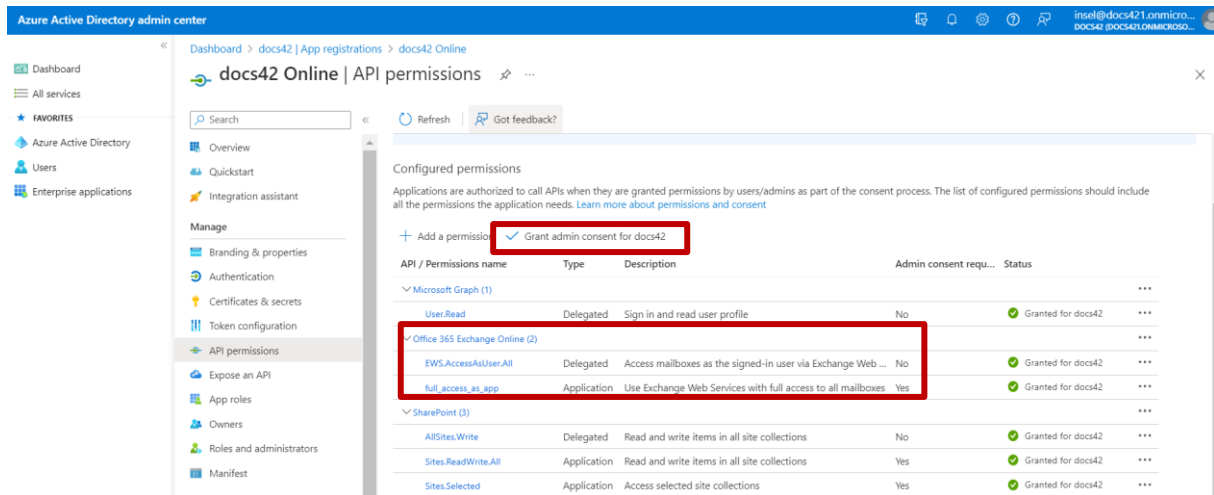
Furthermore, you need to add API permissions to this app registration that contain permissions to send Mails via Exchange online. This needs to be added via the Manifest:

**Under the „requiredResourceAccess“ Section in the Manifest of the App Registration**, you need to add a new JSON object with the following values:

You can copy and paste it from here:

```
{
  "resourceAppId": "00000002-0000-0ff1-ce00-000000000000",
  "resourceAccess": [
    {
      "id": "3b5f3d61-589b-4a3c-a359-5dd4b5ee5bd5",
      "type": "Scope"
    },
    {
      "id": "dc890d15-9560-4a4c-9b7f-a736ec74ec40",
      "type": "Role"
    }
  ]
}
```

Afterwards go to the API permissions section. The following additional permissions should appear: Click on “Grant admin Consent.”



Now you need to add a new Site to the AzureAD Section of your web.config or your dox42 Online MAUI. It needs to contain the App Id, Tenant ID and the following URL: <https://outlook.office365.com/>

When setting up email calls with Exchange online, do not forget to add the parameter mailmethod and the value “exchangeonline”.



## ReturnAction

Parameter name	Requirement	Effect
<b>Format</b>	mandatory	File format (.docx, .xlsx, .pptx, .pdf, .html, emf (since v.4.6.2.)...).
<b>Filename</b>	Optional	Defines the file name of a document. E. g. Contract.pdf. To generate a dynamic file name, use the following syntax: Contract_<%employee.fullname%>.pdf
<b>Mode</b>	optional  Possible Values: File (Standard) Text	Defines whether the generated Document should be returned as File (Mode=File) in ServiceResult.GeneratedDocs or as text (Mode=Text) in ServiceResult.ResultMessage.
<b>SpecialFormat</b> (dox42ForDocuments only)	optional	When generating a PDF, the format may be specified: PDFA1b and PDFForm. For PDFForm Form fields are preserved in the PDF.  For HTML, you may use the option: HTMLCode which will cause the text of the generated document to be interpreted in HTML code.  EMFZIP (since v. 4.6.2.)
<b>Sign</b>	optional	Value: true  The format PDF may bear a digital signature (dox42ForDocuments only). Certificate and corresponding password have to be defined in the web.config.
<b>Condition</b>	optional (since V. 4.6)	If "false", the ReturnAction will not be performed
<b>Numberofcopies</b>	optional (since v. 4.6.2.)	e.g. 1,2,3. Specifies how many copies should be generated within one file

## PrintAction

Parameter name	Requirement	Effect
<b>PrinterName</b>	optional	The used printer
<b>Collate</b>	optional (true/false)	Sort?
<b>Copies</b>	optional	Amount of copies
<b>Duplex</b>	Optional (Vertical, Horizontal, Simplex)	Duplexprint
<b>FromPage</b>	optional	Print from pagenumber ...
<b>ToPage</b>	optional	... to pagenumber
<b>Condition</b>	Optional (since V. 4.6)	If "false", the PrintAction will not be performed

## dox42LIVEAction

The dox42LIVEAction uses an output configuration from a dox42LIVE.config file to execute output actions. Please see the dox42 LIVE documentation to learn how to define your dox42LIVE.config. To use a dox42LIVEAction from the dox42 SOAP or REST Service the dox42LIVE.config file needs to be stored in the same directory as dox42RestService.ashx and dox42Service.asmx , typically:

C:\dox42Server.

Since version 4.1.3.

Parameter name	Requirement	Effect
<b>LiveActionName</b>	mandatory	The LiveActionName as defined in the dox42LIVE.config file.
<b>Condition</b>	Optional (since V. 4.6)	If "false", the dox42LIVEAction will not be performed

## AzureBlobStorageAction

With the Azure BlobStorageAction you can store your generated documents in an Azure Blob storage container. (Since 4.3.1.)

You need to configure the following settings in Azure to use this action:

In your Azure AD App registration, add **delegated Azure Storage user impersonation** rights and grant admin consent. Further information about dox42 Azure AD App registration can be found in the dox42 SharePoint and D365 CE Documentation:

[https://www.dox42.com/Resources?filter\[\]=.dokumentation.SharePoint&filter\[\]=.dokumentation.CE-CRM&](https://www.dox42.com/Resources?filter[]=.dokumentation.SharePoint&filter[]=.dokumentation.CE-CRM&)

Next, add your app registration as an Owner and as a Contributor to the Access Control Section of your Azure Storage account:

The screenshot shows the Azure portal interface for the 'dox42blobtest' storage account. The 'Access Control (IAM)' section is open, and the 'Role assignments' tab is selected. The interface displays a table of role assignments for the subscription. The table has columns for Name, Type, Role, Scope, and Condition. Two roles are highlighted with red boxes: 'Storage Blob Data Contributor' and 'Storage Blob Data Owner', both assigned to the 'AzureBlobStorageTest' app. The 'Contributor' role is also assigned to the app with a scope of 'Subscription (inherited)'.

Name	Type	Role	Scope	Condition
52a732f6981343e3865900da	App	Contributor	Subscription (inherited)	None
Storage Blob Data Contributor	App	Storage Blob Data Contributor	This resource	Add
AzureBlobStorageTest	App	Storage Blob Data Contributor	This resource	Add
dox42 Azuro Server	App	Storage Blob Data Contributor	This resource	Add
Storage Blob Data Owner	App	Storage Blob Data Owner	This resource	Add
AzureBlobStorageTest	App	Storage Blob Data Owner	This resource	Add
dox42 Azuro Server	App	Storage Blob Data Owner	This resource	Add

Finally, add <https://storage.azure.com> with the corresponding App Id, TenantID and Client Secret to the Azure AD Section in your web.config or your dox42 Online Maui. In your dox42 calls, use the following parameters:

Parameter name	Requirement	Effect
<b>AzureBlobStorageAccountname</b>	mandatory	The name of the Azure storage account.
<b>AzureBlobStorageContainername</b>	Mandatory	The name of the Azure storage container.
<b>UploadToAzurePath</b>	Mandatory	The file name (which can include directories) and extension of the uploaded file: Example: directoryA/document.docx
<b>Condition</b>	Optional (since V. 4.6)	If "false", the AzureBlobStorageAction will not be performed

## Configuration parameters

The following parameters can be configured for the **dox42** server in the AppSettings of the web.config.

- › **LicenseFilePath:** Indicates the path to the location of the license file. In case no license file exists, the **dox42** server will contact the **dox42** license server in order to verify the license key. If successful, the server will write the license file to the location under the path indicated in the configuration.
- › **LicenseKey:** license key of the **dox42** server
- › **MailServer:** email server for sending emails (optional)
- › **MailServerPort:** email server port (optional)
- › **dox42ServerMail:** sender's default address of the **dox42** service (optional)
- › **MailServerPassWord:** password for the **dox42** email server (optional)
- › **SendErrorsToMail:** standard email address for error messages (optional)
- › **Certificate:** certificate, used for signing (optional)
- › **CertificatePassWord:** certificate password (optional)
- › **CheckServiceHeader (true/false):** Indicates, whether the service header is to be analysed or not. If "false", no service header needs to be included.
- › **UserPermissionODBCConnectionString:** ODBC connection string to a data base, which contains user permissions  
(mandatory if CheckServiceHeader = true)
- › **CheckUserPermissionSQL:** The select statement, which is executed on the UserPermissionODBCConnectionString data base. The statement has to return 1 or "true" in the case of a successfully authorised user. Within the statement, the %username% and %password% tags will be replaced with the values, included in the header.  
(mandatory if CheckServiceHeader = true)
- › **ThrowExceptionOnError (true/false):**  
true: The service returns an exception on error.  
false (or not present): An error will be returned as ResultMessage.  
This configuration parameter is available since build 3.3.0.8.
- › **CustomErrorMessageFile:**  
Path of a XML Files containing Custom Error Messages (optional). Custom Error Messages allow for replacing Error Messages on the basis of a text search. Please find a sample later in the chapter.
- › **HyphenationDictionariesFile:**  
Path of a XML Files containing containing Hyphenation Dictionaries (optional). Please find a sample later in the chapter. (Since version 3.4.3.10)
- › **CheckTrustedTemplateLocations (true/false):**  
**true:** The path/link of the template or data map must start with one of the paths/links listed in the section „trustedTemplateLocations“. If not, the call is cancelled.  
**false** (or not present): No check is performed.  
**CAUTION: if you switch off this check, everybody who is allowed to call your dox42 service can generate templates/datamaps from any location including unknown URLs on your dox42 server. We strongly recommend to make sure this check is always switched on for security reasons!**

This configuration parameter is available since build 3.3.4.

List the trusted template locations in the section „trustedTemplateLocations“.

- › **TempFolder:**  
Folder to be used by the dox42 Server to store temporary files. If not configured the Windows standard Temp Folder is used.  
This configuration parameter is available since version 3.5
- › **RESTInterfaceReturnsFullErrorInfoTempFolder:**  
If true more detailed error information is returned for the dox42 REST call.  
This configuration parameter is available since version 3.6.1.
- › **RESTInterfaceReturnsHTTPError500 (true/false):**  
if true the http Statuscode 500 will be set on error for the dox42 REST call.  
This configuration parameter is available since version 4.0.1
- › **SOAPServiceReturnsFullErrorInfo (true/false):**  
If true more detailed error information is returned for the dox42 SOAP call.  
This configuration parameter is available since version 4.1.1.0
- › **SaveActionEnabled (true/false):** (This parameter is available since build 4.3.2)  
true: you can use the dox42 SaveAction to store generated documents on your server  
false: you cannot use the dox42 Save Action and an error will be returned.
- › **RedirectURLEnabled (true/false):** (This parameter is available since build 4.3.2)  
true: You can add the parameter &RedirectURL=https://www.yourURL.com to your calls to redirect to a certain URL after document generation  
false: The RedirectURL parameter will throw an exception, you will not be redirected
- › **MailMethod:** (This parameter is available since build 4.3.3)  
ExchangeOnline: Needs to be used, if you send mails via Microsoft Exchange Online. Read the EMailAction section for more information.  
onaspouseerrortrymailkit: uses Aspose to send mails, if Aspose method fails, MailKit is used to send dox42 emails  
aspose: uses Aspose to send mails  
onmailkiterrortryaspose: uses MailKit to send mails, if MailKit method fails, uses Aspose to send mails  
mailkit: uses MailKit to send mails
- › **LoadLocalFilesFromTrustedTemplateLocationOnly (true/false):** (This parameter is available since build 4.4.1)  
If true, you can only integrate files from trusted template locations in your data sources and dynamic fields
- › **AADLibrary (MSAL/ADAL):** (This parameter is available since build 4.6.1)  
Determines which Microsoft Authentication library is used. MSAL is the default value.

## Example

```
<appSettings>
```

```

<add key="LicenseFilePath" value="c:\dox42\dox42WebService\License" />
<add key="LicenseKey" value="..." />
<add key="MailServer" value="smtp.1und1.de"/>
<add key="dox42ServerMail" value="testserver@dox42.com"/>
<add key="MailServerPassWord" value="..." />
<add key="SendErrorsToMail" value="monitoring@dox42.com"/>
<add key="MailServerPort" value="587"/>
<add key="MailMethod" value="onaspouseerrortrymailkit"/>
<add key="Certificate" value="c:\dox42\dox42WebService\Certs\Certificate.pfx"/>
<add key="CertificatePassWord" value="..." />
<add key="CheckServiceHeader" value="true"/>
<add key="UserPermissionODBCConnectionString"
    value="Driver={SQL Server Native Client 10.0};Server=...;Database=...;Uid=...;Pwd=...;" />
<add key="CheckUserPermissionSQL"
    value="SELECT COUNT(*) FROM Users where Username =
    '%username%' and Password = '%password%'" />
<add key="ThrowExceptionOnError" value="false"/>
<add key="CustomErrorMessageFile" value="C:\dox42Server\CustomErrors.xml"/>
<add key="CheckTrustedTemplateLocations" value="true"/>
<add key="TempFolder" value="C:\dox42Server\WorkingDir"/>
<add key="RESTInterfaceReturnsFullErrorInfo" value="false"/>
<add key="RESTInterfaceReturnsHTTPError500" value="false" />
<add key="SOAPServiceReturnsFullErrorInfo" value="false" />
<add key="SaveActionEnabled" value="true" />
<add key="RedirectURLEnabled" value="true" />
<add key="LoadLocalFilesFromTrustedTemplateLocationOnly" value="false" />

```

```
</appSettings>
```

Custom data sources will be registered in a separate section under the name “customDataSources”.

```
<customDataSources>
```

```

<add key="CSVDataEngine.CSVDataSourceParser"
    value="CSV Datenquelle;c:\dox42\dox42WebService\bin\CSVDataEngine.dll;" />

```

```
</customDataSources>
```

Custom Output Actions Sources are registered in a specified section called “customOutputActions”.

```
<customOutputActions>
```

```

<add key="MyCustomOutputActions.MyOutputAction"
    value=" MyOutputAction; c:\dox42\dox42WebService\bin\MyCustomOutputActions.dll;" />

```

```
</customOutputActions>
```

Trusted template locations are registered in a specified section called “trustedTemplateLocations”.

```
<trustedTemplateLocations>
```

```

<add key="local" value="C:\dox42Server\templates"/>
<add key="sharepoint" value="http://www.mysharepoint.mycompany.com/dox42Templates"/>

```

```
</trustedTemplateLocations>
```

Configure log settings in the file NLog.config. For a complete documentation of the logging framework NLog see [www.nlog-project.org](http://www.nlog-project.org).

## CORS Configuration

If a web-application calls the dox42 Server e.g. via Javascript XMLHttpRequest under a different URL than the application itself, you might get a Cross-Origin Resource Sharing (CORS) error.

### Example:

Calling web-application: <https://yourcompany.sharepoint.com/yoursite>

dox42 Server: <https://dox42server.yourcompany.com>

To enable this scenario you could add a customHeaders section to your web.config, example:

```
<system.webServer>
  <httpProtocol>
    <customHeaders>
      <clear />
      <add name="Access-Control-Allow-Origin" value="*" />
      <add name="Access-Control-Allow-Headers"
        value="Content-Type, Accept, Authorization" />
      <add name="Access-Control-Allow-Methods" value="GET,POST,OPTIONS" />
      <add name="Access-Control-Expose-Headers"
        value="Content-Type, Content-Disposition, Accept" />
    </customHeaders>
  </httpProtocol>
</system.webServer>
```

### See also:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>

<http://www.w3.org/TR/cors/>

## Azure Active Directory Access

This is necessary to store dox42 Templates and Data-Maps on SharePoint Online.

### ConfigSection:

```
<configSections>
  . . .
  <section name ="azureAD" type ="dox42.Core.Utils.AzureADSectionHandler,dox42Core"/>
</configSections>
```

### Azure AD Section:

```
<azureAD>
  <add resource365="https://mytenant.sharepoint.com"
    appID="..."
    tenant="..."
    clientKey="*****" />
</azureAD>
```

## Custom Error Messages Sample

```
<?xml version="1.0" encoding="utf-8"?>
<CustomErrorMessagesDS xmlns="http://dox42.com/CustomErrorMessagesDS.xsd">
  <CustomErrorMessage>
    <Test2SearchInExc>Error uploading file http://mysharepoint</Test2SearchInExc>
```



```
<NewMessage>
SharePoint-farm not responding, please contact your administrator</NewMessage>
</CustomErrorMessage>
</CustomErrorMessagesDS>
```

In case you edit this file using a text editor, please be sure to save in UTF-8 encoding.

## Hyphenation Dictionaries Sample

```
<?xml version="1.0" standalone="yes"?>
<HyphenationDictionary xmlns="http://www.dox42.com/HyphenationDictionary.xsd">
  <Dictionary>
    <Culture>de-CH</Culture>
    <DictionaryPath>C:\dox42Server\hyph_de_CH.dic</DictionaryPath>
  </Dictionary>
</HyphenationDictionary>
```

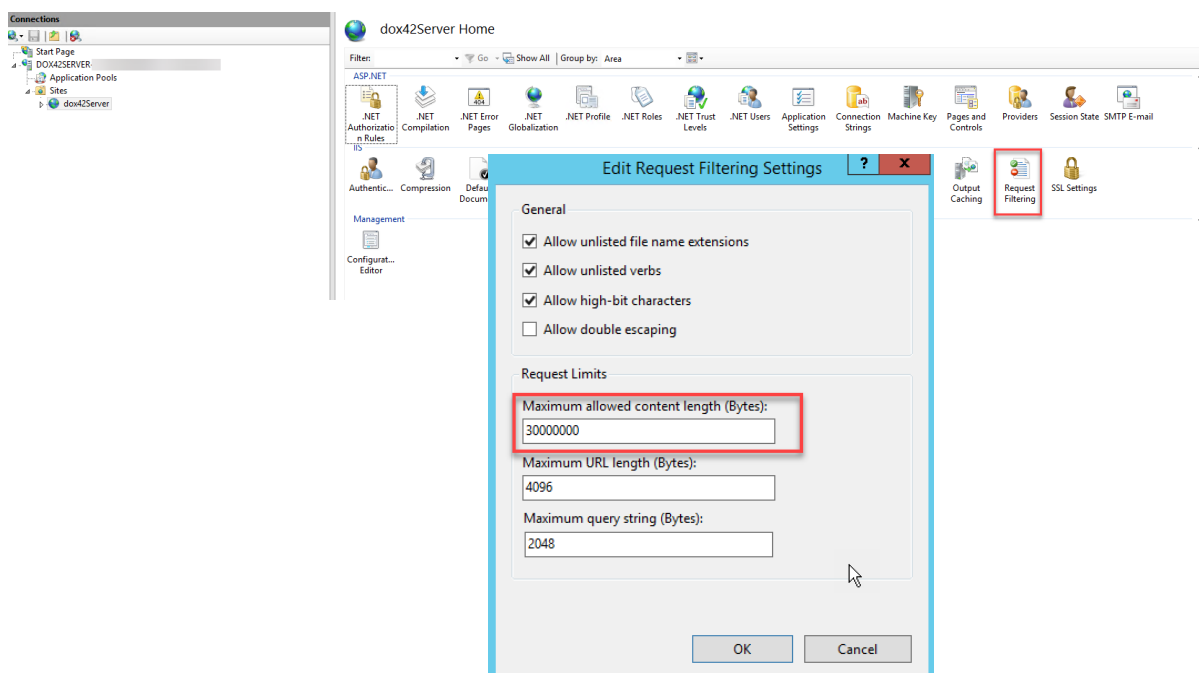
To use automatic hyphenation with the dox42 Word Add-In save such a file named HyphenationDictionary.config to C:\Users\<<USER>>\AppData\Local\dox42V1

## Maximum size of request or time-out

Should you be sending large of requests to dox42, return large files or start a bulk letter, it might become necessary to increase request size or time-out. (The default ASP.NET time-out is 90 sec. After that, a longer call will be aborted by IIS.) Configure this in the web.config as well.

```
<system.web>
  <httpRuntime maxRequestLength="10000000" executionTimeout="1000"/>
</system.web>
```

When you work with the https method POST (base64) (applicable to dox42 D365 FO scenarios) and you are getting the error message – “413 Request entity too large”, you can change the “Maximum allowed content length (bytes)” parameter on the IIS server. In ISS, navigate to Request Filtering, click on “Edit Feature Settings” and change the maximum allowed content length to 2147483647 from 30000000.



## License Activation

Please enter the license key in the Web.config as „LicenseKey“.

On the first call the dox42 Server will contact the dox42 license server to activate your key and write a license file (\*.lic) in the folder indicated under “LicenseFilePath”.

If the Server is only allowed to access the internet via Proxy, maybe the following entry in the web.config will help.



```
Web.config - Editor
Datei Bearbeiten Format Ansicht ?
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <configSections>
    <section name="customDataSources" type="System.Configuration.NameValueFileSectionHandler,System, Version=1.0.3300.0, Culture=neutral
    <section name="trustedTemplateLocations" type="System.Configuration.NameValueFileSectionHandler,System, Version=1.0.3300.0, Culture=n
  </configSections>
  <system.net>
    <defaultProxy useDefaultCredentials="false">
      <proxy usesystemdefault="True" proxyaddress="http://172.30.20.2:8080" bypassonlocal="True" />
    </defaultProxy>
  </system.net>
  <appSettings>
    <add key="LicenseFilePath" value="C:\dox42Server\license" />
  </appSettings>
</configuration>
```

In case you cannot establish an internet connection from your server, please contact [support@dox42.com](mailto:support@dox42.com), we will provide you with your license file.

# Installation

The following section describes the installation on a Windows 2012 Server with IIS 7.5. You can of course use newer versions or a Windows Azure Server to install dox42.

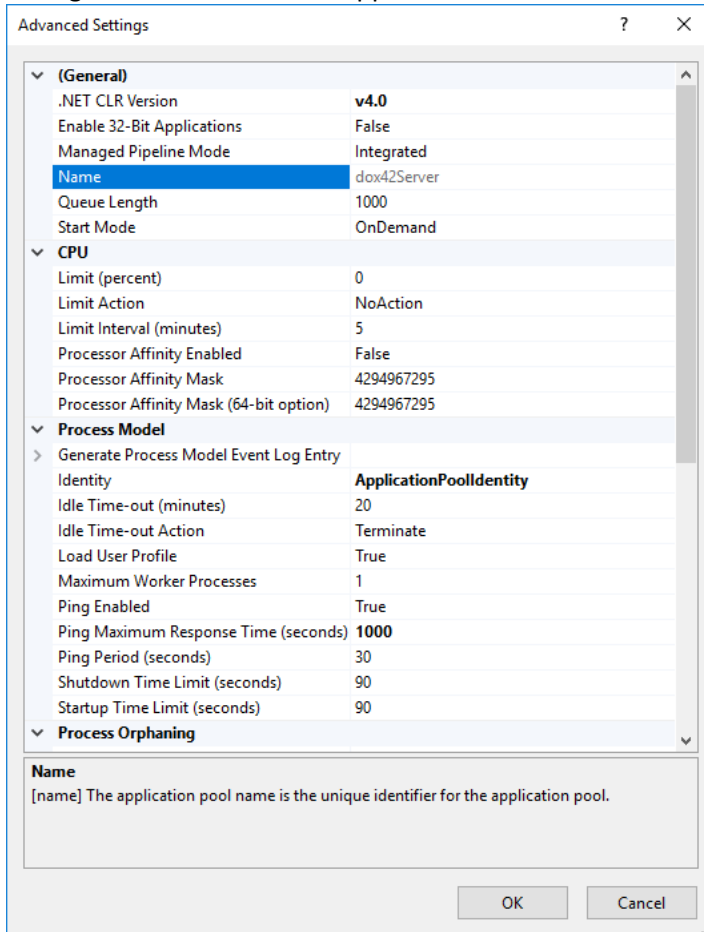
1. Copy the directory  
**dox42ServerPack\dox42ServerVXXXX\dox42Server**  
to  
**C:\dox42Server**  
(You can also choose a different virtual directory.)
2. Set up a site in the inetmgr:

The screenshot shows the 'Add Website' dialog box in IIS Manager. The 'Site name' is 'dox42Server' and the 'Application pool' is 'dox42Server'. The 'Physical path' is 'C:\dox42Server'. The 'Binding' is set to 'http' on 'All Unassigned' IP address and port '4242'. The 'Start Website immediately' checkbox is checked.

Please mind the port! You may use any other port, but bear in mind that the standard port (80) is often already in use by other applications e.g. Sharepoint. Of course dox42 server supports https.

3. Copy your Aspose.Total.Lic file to the BIN Folder (C:\dox42Server\Bin) - not applicable for Trial Installations
4. Install the NET 4.5.2 framework.
5. User Server Manager to activate ASP.NET (inclusive HTTP Activation)

6. If want to enable access to SharePoint Online (Office 365) for this dox42 Server, please install the Microsoft SharePoint Server 2013 Client Components SDK.  
Download: <http://www.microsoft.com/en-us/download/details.aspx?id=35585>
7. Run iisreset.
8. Configure the dox42 server-AppPool as follows:

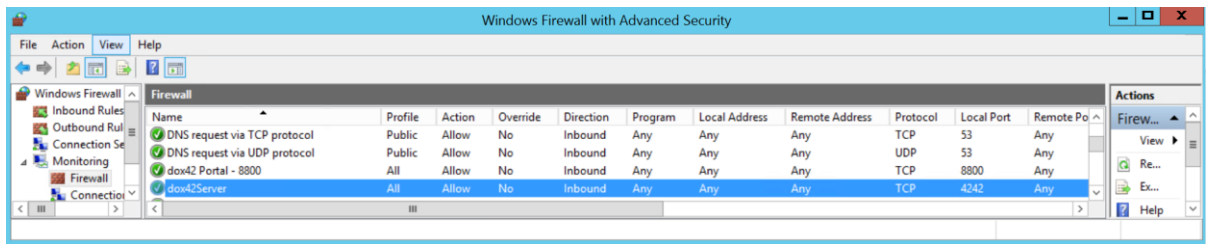


Instead of ApplicationPoolIdentity you can also use a Service-User.

In case you want to generate dox42 sample template which occasionally use MS Access (32-Bit needed) you might need to activate 32 bit applications.

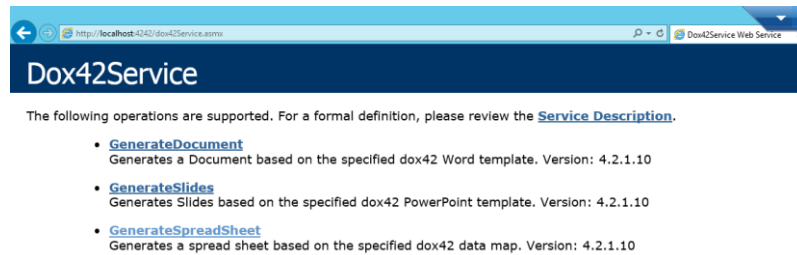
9. In inetmgr, change back to site > manage permissions  
Here, look for the user "IIS AppPool\dox42Server" (Attention, use the local computer as search path!) or a dox42 service account and give this user read **and** write permissions to **C:\dox42Server**
10. Configure web.config (key, license-dir, mail server) and NLog.config (log-dir).

11. Do not forget to activate the port on your firewall:



12. Do a test to reach your dox42 web service via your Browser:

"<http://localhost:4242/dox42Service.aspx>"



13. Afterwards test generating a document with the dox42 Server Designer. Install the Server Designer application and open it. You can find it at C:\dox42ServerPackVXXX\dox42ServerDesigner\_\*. Within the Server Designer, open the HelloWorld\_ServerDesigner.config from C:\dox42Server\templates\HelloWorldTest and click on "Call Service" to test your dox42 server document generation.

The dox42 Server Designer can be used for the easy configuration of your dox42 webservice calls. Please watch the [#dox42class Video Tutorial](#) for more information on how to use the Server Designer.

14. Be sure to use adequate Access Control to protect your dox42 Service against unauthorized use.

15. In order to use AzureAD you need an HTTPS certificate installed. In case you need help how to do this, watch this video: <https://youtu.be/HHU4616G2oc>

## Upgrade

To Upgrade a dox42 Server please perform the following steps:

1. Make a backup of the dox42 Server folder, so you can roll back in case anything goes wrong.
2. Delete all files in the bin folder EXCEPT Aspose.Total.lic and all dll-files, that belong to your custom data sources and output actions. Custom data sources and output actions are registered in the web.config, e.g. dox42SAP, dox42AX, dox42DynamicCRM, etc. Please see the documentation of each module to get a list of the necessary libraries.
3. Copy all files from the bin folder of the dox42 Server Pack to the bin folder on your server.
4. Copy all files EXCEPT dox42LIVE.config from the dox42LIVE folder of the dox42 Server Pack to the dox42LIVE folder on your server.

5. Check if Dox42Service.asmx and dox42RestService.ashx are present in your dox42 Server folder.
6. Check your web.config for new configuration options.
7. Restart your dox42 Server in IIS.
8. Test with dox42 ServerDesigner
9. That's all, you are done, go for a beer 😊

## Support

Should you have any questions, please do not hesitate to contact [support@dox42.com](mailto:support@dox42.com)

**Good luck with your dox42 server!**