Guide for administrators

officeatwork Output Method Instructions

This guide has been created using officeatwork Advanced.
24. September 2020

Table of Contents

About this guide

For whom is the guide intended

This guide is intended for Administrators who want to integrate or modify any printer-driver integration into their officeatwork solution.

What is covered in this guide

The guide illustrates the OOMI-files concept of officeatwork. It gives you an overview of the architecture and shows how to modify or extend the officeatwork output management with your own printer drivers through configuration of OOMI-files.

Knowledge required

To understand this guide, a good general knowledge of computing as well as basic knowledge regarding XML is needed.

Typographic conventions

Before reading this guide, you should be familiar with the typographic conventions used.

The following graphic descriptions highlight sections of text with particular significance.

Formatting Convention	Type of Information
Triangle ➢	Step-by-step procedure. You can follow these instructions to perform a specific task.
Bold Typeface	Objects needed for selection, such as menus, buttons, items in a list or table headers.
CAPITAL LETTERS	Key legends on the keyboard. For example `SHIFT`, `CTRL` or `ALT`.
KEY+KEY	Key combinations which must be pressed at the same time are marked with +. Examples: `CTRL+P` or `ALT+F4`.

CHAPTER 1

Concept

Introduction

The officeatwork output management architecture is an open architecture based on XML definitions. The part that is in charge of the definition on how to operate the required printer driver is stored in separate files with the extension *.oomi. OOMI stands for «Officeatwork Output Method Instruction». OOMI-Files are XML formatted text files and need to be registered in your officeatwork solution. Each output variant can have multiple OOMI-Files registered.

The following illustration shows the elements involved when creating an output using the officeatwork output-managment.

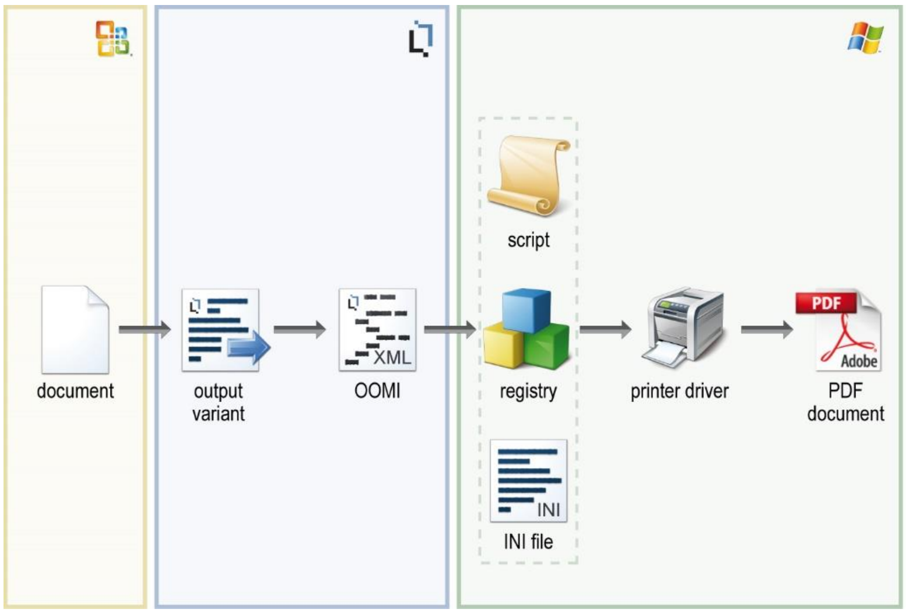

Figure 1: officeatwork OOMI-concept overview

It is for instance possible to have the output variation «Save Draft PDF» which uses an OOMI-File to create a low resolution PDF, whereas a «Save as final PDF» variation uses an OOMI-File that provides maximum resolution for all images in the PDF. You could also include the instruction in your OOMI-file to protect your PDF output with a password. It is even possible to have a variation that converts the text to a picture (only if supported by your driver) for maximum security (especially for documents like contracts).

Description of the steps in sequence

When an officeatwork output variation is executed the following steps occur in sequence.

➢ ***Steps for officatwork sending or saving variations***

- The user chooses an officeatwork sending or saving output variation.
- officeatwork changes the content of the document according to the variation.
- officeatwork checks the availability of the driver described in the oomi-file for this output variation.
- If the driver is not found, officeatwork checks the driver noted within the next OOMI-file registered for this output variation. officeatwork repeats this until it finds the desired driver on the user's PC. If no matching driver is found, the document is restored to its original state and the output process is stopped. In this case a message informs the user that the required driver could not be found for this output variation. If a driver is found, the output process is continued.
- If necessary, officeatwork switches to the currently required printer driver.

- officeatwork reads the «BeforeAction» section of the oomi-file and performs the defined instructions.
- officeatwork executes the instructions defined in the «Action» section of the oomi file.
- officeatwork executes the instructions defined in the «AfterAction» section of the oomi file.
- officeatwork restores the connected printer driver to its original state.
- officeatwork restores the document to its original state.

Basic XML Structure overview

An OOMI-file is split up into three main sections, the BeforeAction, the Action and AfterAction section.

BeforeAction	This section is used to prepare the driver before the driver is activated. This is where all driver settings are usually set with their appropriate values. At this stage the document that needs printing has already been updated and its values can be accessed and passed on to the driver. Most commonly the file name of the newly to create document is passed on here. It is also possible to set a flag so that the changes made in the BeforeAction revert to original automatically, after the BeforeAction section has been processed.
Action	In this section the driver is activated by printing the document within the office application.
AfterAction	This section allows you to restore the driver settings if necessary. It is also possible to receive updated information from the document.

CHAPTER 2

Definitions

XML Definitions

The following sample XML File represents the complete structure of an OOMI-File:

Sample XML File:

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <BeforeAction RestoreSettingsAfterAction="0 | -1">
      <Registry>
         <Keys>
            <Key HKEY="HKEY_CLASSES_ROOT | HKEY_CURRENT_USER | HKEY_LOCAL_MACHINE"
Path="Software\PDFDriver\Program" >
               <Values>
                  <Value Name="RunProgramAfterSaving" Value="0"/>
                  <Value Name="UseAutosave" Value="1"/>
                  <Value Name="AutosaveDirectory" Value="%Desktop%"/>
                  <Value Name="AutosaveFilename" Value=""/>
                  <Value Name="StandardTitle" Value=""/>
                  <Value Name="StandardSubject" Value=""/>
               </Values>
            </Key>
         </Keys>
      </Registry>
      <IniFiles>
         <IniFile Path="%AppData%\PDFDriver" Filename="settings.ini">
            <Sections>
               <Section Name="PDFDriver">
                  <Entries>
                     <Entry Name="Output" Value="%Desktop%\.pdf"/>
                     <Entry Name="ConfirmOverwrite" Value="no"/>
                     <Entry Name="ShowSettings" Value="never"/>
                     <Entry Name="ShowPDF" Value="yes"/>
                     <Entry Name="Title" Value=""/>
                     <Entry Name="Subject" Value=""/>
                  </Entries>
               </Section>
            </Sections>
         </IniFile>
      </IniFiles>
      <VBScripts>
         <RunVBScript ResultType="File | Printout" Location="%Scripts%\Script.vbs" AllowUI="0 | -1"
UseSafeSubset="0 | -1" Function="FunctionName">
            <ParameterDefinitions>
               <HostApplication/>
               <ActiveFileName/>
               <ActivePath/>
               <DocumentLanguage/>
               <Changed/>
               <MasterProperties>
                  <MasterProperty IDName="Signature1">
                     <Field Name="Name"/>
                     <Field Name="Function"/>
                  </MasterProperty>
                  <MasterProperty IDName="CustomField">
                     <Field Name="External"/>
                     <Field Name="No"/>
                     <Field Name="ShowTimes"/>
                  </MasterProperty>
               </MasterProperties>
               <WordCustomDocumentProperties>
                  <WordCustomDocumentProperty Name="TestProperty"/>
               </WordCustomDocumentProperties>
               <WordBuiltInDocumentProperties>
                  <WordBuiltInDocumentProperty Name="Author"/>
               </WordBuiltInDocumentProperties>
               <WordVariables>
                  <WordVariable Name="TestVariable"/>
               </WordVariables>
               <WordBookmarks>
                  <WordBookmark Name="TestBookmark"/>
               </WordBookmarks>
            </ParameterDefinitions>
         </RunVBScript>
      </VBScripts>
      <VBAMacros>
         <RunVBAMacro ResultType="File | Printout" Function="Module.MacroName">
            <ParameterDefinitions>
               <HostApplication/>
               <ActiveFileName/>
               <ActivePath/>
               <DocumentLanguage/>
               <Changed/>
               <MasterProperties>
```

```
                        <MasterProperty IDName="Signature1">
                            <Field Name="Name"/>
                            <Field Name="Function"/>
                        </MasterProperty>
                        <MasterProperty IDName="CustomField">
                            <Field Name="External"/>
                            <Field Name="No"/>
                            <Field Name="ShowTimes"/>
                        </MasterProperty>
                    </MasterProperties>
                    <WordCustomDocumentProperties>
                        <WordCustomDocumentProperty Name="TestProperty"/>
                    </WordCustomDocumentProperties>
                    <WordBuiltInDocumentProperties>
                        <WordBuiltInDocumentProperty Name="Author"/>
                    </WordBuiltInDocumentProperties>
                    <WordVariables>
                        <WordVariable Name="TestVariable"/>
                    </WordVariables>
                    <WordBookmarks>
                        <WordBookmark Name="TestBookmark"/>
                    </WordBookmarks>
                </ParameterDefinitions>
            </RunVBAMacro>
        </VBAMacros>
    </BeforeAction>
    <Action>
        <Print ResultFilename="%Desktop%\.pdf" PrinterName="Printer driver name as displayed in the Microsoft 
Office Print Dialog"/>
        <Save ResultFilename="%Desktop%\.pdf" Filetype="any file type available from Microsoft Office"/>
        <RunVBScript ResultType="File | Printout" Location="%Scripts%\Script.vbs" AllowUI="0 | -1" 
UseSafeSubset="0 | -1" Function="FunctionName">
            <ParameterDefinitions>
                <HostApplication/>
                <ActiveFileName/>
                <ActivePath/>
                <DocumentLanguage/>
                <Changed/>
                <MasterProperties>
                    <MasterProperty IDName="Signature1">
                        <Field Name="Name"/>
                        <Field Name="Function"/>
                    </MasterProperty>
                    <MasterProperty IDName="CustomField">
                        <Field Name="External"/>
                        <Field Name="No"/>
                        <Field Name="ShowTimes"/>
                    </MasterProperty>
                </MasterProperties>
                <WordCustomDocumentProperties>
                    <WordCustomDocumentProperty Name="TestProperty"/>
                </WordCustomDocumentProperties>
                <WordBuiltInDocumentProperties>
                    <WordBuiltInDocumentProperty Name="Author"/>
                </WordBuiltInDocumentProperties>
                <WordVariables>
                    <WordVariable Name="TestVariable"/>
                </WordVariables>
                <WordBookmarks>
                    <WordBookmark Name="TestBookmark"/>
                </WordBookmarks>
            </ParameterDefinitions>
        </RunVBScript>
        <RunVBAMacro ResultType="File | Printout" Function="Module.MacroName">
            <ParameterDefinitions>
                <HostApplication/>
                <ActiveFileName/>
                <ActivePath/>
                <DocumentLanguage/>
                <Changed/>
                <MasterProperties>
                    <MasterProperty IDName="Signature1">
                        <Field Name="Name"/>
                        <Field Name="Function"/>
                    </MasterProperty>
                    <MasterProperty IDName="CustomField">
                        <Field Name="External"/>
                        <Field Name="No"/>
                        <Field Name="ShowTimes"/>
                    </MasterProperty>
                </MasterProperties>
                <WordCustomDocumentProperties>
                    <WordCustomDocumentProperty Name="TestProperty"/>
                </WordCustomDocumentProperties>
                <WordBuiltInDocumentProperties>
                    <WordBuiltInDocumentProperty Name="Author"/>
                </WordBuiltInDocumentProperties>
                <WordVariables>
                    <WordVariable Name="TestVariable"/>
                </WordVariables>
                <WordBookmarks>
                    <WordBookmark Name="TestBookmark"/>
                </WordBookmarks>
            </ParameterDefinitions>
        </RunVBAMacro>
    </Action>
    <AfterAction>
        <Registry>
            <Keys>
                <Key HKEY="HKEY_CLASSES_ROOT | HKEY_CURRENT_USER | HKEY_LOCAL_MACHINE"
    Path="Software\PDFDriver\Program" >
                    <Values>
                        <Value Name="RunProgramAfterSaving" Value="0"/>
                        <Value Name="UseAutosave" Value="1"/>
                        <Value Name="AutosaveDirectory" Value="%Desktop%"/>
                        <Value Name="AutosaveFilename" Value=""/>
                        <Value Name="StandardTitle" Value=""/>
                        <Value Name="StandardSubject" Value=""/>
                    </Values>
                </Key>
            </Keys>
        </Registry>
        <IniFiles>
            <IniFile Path="%AppData%\PDFDriver" Filename="settings.ini">
                <Sections>
                    <Section Name="PDFDriver">
                        <Entries>
                            <Entry Name="Output" Value="%Desktop%\.pdf"/>
                            <Entry Name="ConfirmOverwrite" Value="no"/>
                            <Entry Name="ShowSettings" Value="never"/>
                            <Entry Name="ShowPDF" Value="yes"/>
                            <Entry Name="Title" Value=""/>
                            <Entry Name="Subject" Value=""/>
                        </Entries>
                    </Section>
                </Sections>
            </IniFile>
        </IniFiles>
        <VBScripts>
            <RunVBScript ResultType="File | Printout" Location="%Scripts%\Script.vbs" AllowUI="0 | -1"
    UseSafeSubset="0 | -1" Function="FunctionName">
                <ParameterDefinitions>
                    <HostApplication/>
                    <ActiveFileName/>
                    <ActivePath/>
                    <DocumentLanguage/>
                    <Changed/>
                    <MasterProperties>
                        <MasterProperty IDName="Signature1">
                            <Field Name="Name"/>
                            <Field Name="Function"/>
                        </MasterProperty>
                        <MasterProperty IDName="CustomField">
                            <Field Name="External"/>
                            <Field Name="No"/>
                            <Field Name="ShowTimes"/>
                        </MasterProperty>
                    </MasterProperties>
                    <WordCustomDocumentProperties>
                        <WordCustomDocumentProperty Name="TestProperty"/>
                    </WordCustomDocumentProperties>
                    <WordBuiltInDocumentProperties>
                        <WordBuiltInDocumentProperty Name="Author"/>
                    </WordBuiltInDocumentProperties>
                    <WordVariables>
                        <WordVariable Name="TestVariable"/>
                    </WordVariables>
                    <WordBookmarks>
                        <WordBookmark Name="TestBookmark"/>
                    </WordBookmarks>
                </ParameterDefinitions>
            </RunVBScript>
        </VBScripts>
        <VBAMacros>
            <RunVBAMacro ResultType="File | Printout" Function="Module.MacroName">
                <ParameterDefinitions>
                    <HostApplication/>
                    <ActiveFileName/>
                    <ActivePath/>
                    <DocumentLanguage/>
                    <Changed/>
                    <MasterProperties>
                        <MasterProperty IDName="Signature1">
                            <Field Name="Name"/>
                            <Field Name="Function"/>
                        </MasterProperty>
                        <MasterProperty IDName="CustomField">
                           <Field Name="External"/>
                           <Field Name="No"/>
                           <Field Name="ShowTimes"/>
                        </MasterProperty>
                     </MasterProperties>
                     <WordCustomDocumentProperties>
                        <WordCustomDocumentProperty Name="TestProperty"/>
                     </WordCustomDocumentProperties>
                     <WordBuiltInDocumentProperties>
                        <WordBuiltInDocumentProperty Name="Author"/>
                     </WordBuiltInDocumentProperties>
                     <WordVariables>
                        <WordVariable Name="TestVariable"/>
                     </WordVariables>
                     <WordBookmarks>
                        <WordBookmark Name="TestBookmark"/>
                     </WordBookmarks>
                  </ParameterDefinitions>
               </RunVBAMacro>
            </VBAMacros>
         </AfterAction>
    </OutputMethodInstruction>
```

Parameter definitons

The «ParameterDefinition» tag within the OOMI XML gives you access to document values including officeatwork specific variables. Most values can be read and written. To read a value you must define a request statement that will then be returned to you with the requested values. To write a value you must define a set statement.

ActiveFileName()

This parameter returns the filename of the active document. The file can have the extensions *.doc, *.ows or *.owt, depending on the host application.

This parameter is read-only.

Request Syntax

```
<ActiveFileName/>
```

Return Syntax

```
<ActiveFileName Value=""/>
```

Parameter

The parameter «ActiveFileName» has the following attributes:

Name	Description
Value	String. Filename of active file

Request Sample

```
<ParameterDefinitions>
   <ActiveFileName/>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <ActiveFileName Value="Exampledocument.ows"/>
</ParameterDefinitions>
```

ActivePath()

This parameter returns the path of the active document.

This parameter is read-only.

Request Syntax

```
<ActivePath/>
```

Return Syntax

```
<ActivePath Value=""/>
```

Parameter

The parameter «ActivePath» has the following attributes:

Name	**Description**
Value	String. Path of active file

Request Sample

```
<ParameterDefinitions>
   <ActivePath/>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <ActivePath Value="Q:\officeatwork\SmartTemplates"/>
</ParameterDefinitions>
```

Cancel()

This parameter indicates whether officeatwork shall continue the output process or not.

If the value of the parameter «cancel» is returned with the value «-1» from the script, then officeatwork discontinues the output process. This mechanism can be used whenever the processed document should not be printed, sent or saved. This is useful whenever a document based on some status or values is not valid for creating any output.

This parameter is read-only.

In case of a cancellation of an output-process based on this cancel parameter, officeatwork stops the output process without notifying the user. Should you want to inform the user about the cancellation, you must alert the user within your scripts or Macros.

Request Syntax

```
<Cancel/>
```

Return Syntax

```
<Cancel Value="-1"/>
```

Parameter

The parameter «Cancel» has the following attributes:

Name	Description
Value	Long. **0**, continue the output process **-1**, cancel the output process Default value is **0**

Return Sample

```
<ParameterDefinitions>
   <Cancel Value="-1"/>
</ParameterDefinitions>
```

Changed()

This parameter indicates whether the document has been modified since it was last saved.

In combination with for instance a DMS solution, it may be necessary to know if the document has changed since it was last saved. In this case the changed parameter can be used to forbid any output from documents that have been changed.

This parameter is read-only.

Request Syntax

```
<Changed/>
```

Return Syntax

```
<Changed Value="0"/>
```

Parameter

The parameter «Changed» has the following attributes:

Name	Description
Value	Long. **0**, the document has not been modified since it was opened **-1**, the document has been modified since it was opened

Request Sample

```
<ParameterDefinitions>
   <Changed/>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <Changed Value="-1"/>
</ParameterDefinitions>
```

DocumentLanguage ()

This parameter returns the document language in form of an LCID. A detailed list of available Language IDs (LCID) can be found in the appendix.

This parameter is read-only.

Request Syntax

```
<DocumentLanguage/>
```

Return Syntax

```
<DocumentLanguage Value=""/>
```

Parameter

The parameter «DocumentLanguage» has the following attributes:

Name	**Description**
Value	Long. The documents LCID

Request Sample

```
<ParameterDefinitions>
   <DocumentLanguage/>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <DocumentLanguage Value="2055"/>
</ParameterDefinitions>
```

HostApplication()

This parameter returns the application name of the host application.

This parameter is useful to make sure your output profiles are only executed in the office applications.

This parameter is read-only.

Request Syntax

```
<HostApplication/>
```

Return Syntax

```
<HostApplication Value=""/>
```

Parameter

The parameter «HostApplication» has the following attributes:

Name	Description
Value	String. Name of the host application Possible values are «DocumentWizard» or «SmartTemplateManager»

Request Sample

```
<ParameterDefinitions>
   <HostApplication/>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <HostApplication Value="DocumentWizard"/>
</ParameterDefinitions>
```

MasterProperties ()

With this parameter all fields of any officeatwork MasterProperty can be read and written.

Request Syntax

```
<MasterProperties>
     <MasterProperty IDName="">
        <Field Name=""/>
     </MasterProperty>
</MasterProperties>
```

Set Syntax

```
<MasterProperties>
     <MasterProperty IDName="" Where="" Is="">
        <Field Name="" Value=""/>
     </MasterProperty>
</MasterProperties>
```

Return Syntax

```
<MasterProperties>
     <MasterProperty IDName="">
        <Field Name="" Value=""/>
```

```
    </MasterProperty>
</MasterProperties>
```

Parameter

The element «MasterProperties» has the following element:

Name	Description
MasterProperty	Element. Element representing a specific MasterProperty. If you wish to work with two different MasterProperties, you need to add multiple MasterProperty elements to your MasterProperties element.

The element «MasterProperty» has the following attributes:

Name	Description
IDName	String. Value of the field IDName
Where	String. Name of filed to be used to match value of Is attribute. Only available in the Set Syntax.
Is	String. Value used in combination with the Where attributes to find a specific record within a MasterProperty. Only available in the Set Syntax.

The element «MasterProperty» has the following sub-elements:

Name	Description
Field	Element. Element representing a specific field on a MasterProperty.

The element «Field» has the following attributes:

Name	Description
Name	String. Name of the Field.
Value	String. Corresponding field value.

Request Sample

```
<ParameterDefinitions>
   <MasterProperties>
      <MasterProperty IDName="CustomFields">
         <Field Name="External"/>
      </MasterProperty>
      <MasterProperty IDName="Signature1">
         <Field Name="Name"/>
         <Field Name="Function"/>
      </MasterProperty>
   </MasterProperties>
</ParameterDefinitions>
```

Set Sample

```
<ParameterDefinitions>
   <MasterProperties>
      <MasterProperty IDName="Signature1" Where="EmployeeID" Is="AT874KK">
         <Field Name="Function" Value="CEO"/>
      </MasterProperty>
   </MasterProperties>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <MasterProperties>
      <MasterProperty IDName="CustomFields">
         <Field Name="External" Value="Wert A"/>
      </MasterProperty>
      <MasterProperty IDName="Signature1">
         <Field Name="Name" Value="Max von Arx"/>
         <Field Name="Function" Value="Angestellter"/>
      </MasterProperty>
   </MasterProperties>
</ParameterDefinitions>
```

OutputFileName()

This parameter returns the filename that shall be used within a send or save output variant.

The filename extension must match with the requirements of the involved driver.

Request Syntax

```
<OutputFileName/>
```

Return Syntax

```
<OutputFileName Value=""/>
```

Parameter

The element «OutputFileName» has the following attributes:

Name	**Description**
Value	String. Name of the file that will be used for the output variant. In most cases this will be the name of the PDF File that will be created in this output variant.

Request Sample

```
<ParameterDefinitions>
   <OutputFileName/>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <OutputFileName Value="MyPDFDocument.pdf"/>
</ParameterDefinitions>
```

OutputPath()

This parameter defines the path used for the output variant. It will be used in a send or save output variant for the file to be sent or saved.

Request Syntax

```
<OutputPath/>
```

Return Syntax

```
<OutputPath Value=""/>
```

Parameter

The element «OutputPath» has the following attributes:

Name	Description
Value	String. Path used to save the file. In most cases this will be the path of the PDF file.

Request Sample

```
<ParameterDefinitions>
   <OutputPath/>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <OutputPath Value="C:\myFiles"/>
</ParameterDefinitions>
```

WordBookmarks ()

This parameter allows you to read and write values of word bookmarks.

If bookmarks are linked to officeatwork values, whatever you set the bookmark to will be replaced with what officeatwork fills the bookmarks with. So make sure the bookmarks you are using are not used by officeatwork.

Request Syntax

```
<WordBookmarks>
     <WordBookmark Name=""/>
</WordBookmarks>
```

Set and Return Syntax

```
<WordBookmarks>
     <WordBookmark Name="" Value=""/>
</WordBookmarks>
```

Parameter

The WordBookmarks element has the following sub-elements:

Name	**Description**
WordBookmark	Element. Each WordBookmark element represents a Word bookmark.

The element «WordBookmark» has the following attributes:

Name	**Description**
Name	String. Name of bookmark
Value	String. Bookmark value

Request Sample

```
<ParameterDefinitions>
       <WordBookmarks>
          <WordBookmark Name="DMS_Datum"/>
          <WordBookmark Name="DMS_Status"/>
       </WordBookmarks>
</ParameterDefinitions>
```

Set and Return Sample

```
<ParameterDefinitions>
       <WordBookmarks>
          <WordBookmark Name="DMS_Datum" Value="05.04.2007"/>
          <WordBookmark Name="DMS_Status" Value="Draft"/>
       </WordBookmarks>
</ParameterDefinitions>
```

WordBuiltInDocumentProperties ()

This parameter allows you to read and write values of Word BuiltInDocumentProperties.

If Word BuiltInDocumentProperties are linked to officeatwork values, whatever you set the Word BuiltInDocumentProperties to will be replaced with what officeatwork fills the Word BuiltInDocumentProperties with. So make sure the Word BuiltInDocumentProperties you are using are not used by officeatwork.

Request Syntax

```
<WordBuiltInDocumentProperties>
     <WordBuiltInDocumentProperty Name=""/>
</WordBuiltInDocumentProperties>
```

Return Syntax

```
<WordBuiltInDocumentProperties>
     <WordBuiltInDocumentProperty Name="" Value=""/>
</WordBuiltInDocumentProperties>
```

Parameter

The element «WordBuiltInDocumentProperties» has the following sub-elements:

Name	Description
WordBuiltInDocumentProperty	Element. Each element represents a Word BuiltInDocumentProperty.

The element «WordBuiltInDocumentProperty» has the following attributes:

Name	Description
Name	String. Name of Word BuiltInDocumentProperty.
Value	String. Value of Word BuiltInDocumentProperty

Request Sample

```
<ParameterDefinitions>
       <WordBuiltInDocumentProperties>
          <WordBuiltInDocumentProperty Name="Author"/>
          <WordBuiltInDocumentProperty Name="Company"/>
       </WordBuiltInDocumentProperties>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
       <WordBuiltInDocumentProperties>
          <WordBuiltInDocumentProperty Name="Author" Value="Name des Autors"/>
          <WordBuiltInDocumentProperty Name="Company" Value="Firmenname"/>
       </WordBuiltInDocumentProperties>
</ParameterDefinitions>
```

WordCustomDocumentProperties ()

This parameter allows you to read and write values of Word CustomDocumentProperties.

If Word CustomDocumentProperties are linked to officeatwork values, whatever you set the Word CustomDocumentProperties to will be replaced with what officeatwork fills the Word CustomDocumentProperties with. So make sure the Word BuiltInDocumentProperties you are using are not used by officeatwork.

Request Syntax

```
<WordCustomDocumentProperties>
     <WordCustomDocumentProperty Name=""/>
</WordCustomDocumentProperties>
```

Return Syntax

```
<WordCustomDocumentProperties>
     <WordCustomDocumentProperty Name="" Value=""/>
</WordCustomDocumentProperties>
```

Parameter

The element «WordCustomDocumentProperties» has the following sub-elements:

Name	Description

WordCustomDocumentProperty	Element. Each element represents a Word CustomDocumentProperty

The element «WordCustomDocumentProperty» has the following attributes:

Name	**Description**
Name	String. Name of Word CustomDocumentProperty
Value	String. Value of Word CustomDocumentProperty

Request Sample

```
<ParameterDefinitions>
   <WordCustomDocumentProperties>
      <WordCustomDocumentProperty Name="DMS_Datum"/>
      <WordCustomDocumentProperty Name="DMS_Status"/>
   </WordCustomDocumentProperties>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <WordCustomDocumentProperties>
      <WordCustomDocumentProperty Name="DMS_Datum" Value="5. April 2007"/>
      <WordCustomDocumentProperty Name="DMS_Status" Value="Draft"/>
   </WordCustomDocumentProperties>
</ParameterDefinitions>
```

WordVariables ()

This parameter allows you to read and write values of Word Variables.

If Word variables are linked to officeatwork values, whatever you set the Word variables to will be replaced with what officeatwork fills the Word variables with. So make sure the Word variables you are using are not used by officeatwork.

Request Syntax

```
<WordVariables>
     <WordVariable Name=""/>
</WordVariables>
```

Return Syntax

```
<WordVariables>
     <WordVariable Name="" Value=""/>
</WordVariables>
```

Parameter

The element «WordVariables» has the following sub-elements:

Name	**Description**
WordVariable	Element. Each element represents a Word Variable.

The element «WordVariable» has the following attributes:

Name	**Description**

Name	String. Name of Word Variable.
Value	String. Value of Word Variable

Request Sample

```
<ParameterDefinitions>
   <WordVariables>
      <WordVariable Name="DMS_Datum"/>
      <WordVariable Name="DMS_Status"/>
   </WordVariables>
</ParameterDefinitions>
```

Return Sample

```
<ParameterDefinitions>
   <WordVariables>
      <WordVariable Name="DMS_Datum" Value="05.04.2007"/>
      <WordVariable Name="DMS_Status" Value="Draft"/>
   </WordVariables>
<ParameterDefinitions>
```

Variables

officeatwork provides numerous variables that can be used within an OOMI-file. The variables can be placed within your OOMI-file and will be replaced with their appropriate values during the output process. The following list explains all the values available through variables.

[[Path]]	provides the current path of the document
[[Filename]]	provides the current filename of the document
[[HostApplicationFullName]]	provides the name of the current office application that triggered the output. For example «Word» or «PowerPoint»
[[Title]]	provides the value of the document Title as defined in the Master Template
[[Subject]]	provides the value of the document Subject as defined in the Master Template
[[Author]]	provides the value of the document Author as defined in the Master Template
[[Manager]]	provides the value of the document Manager as defined in the Master Template
[[Company]]	provides the value of the document Company as defined in the Master Template
[[Category]]	provides the value of the document Category as defined in the Master Template
[[Keywords]]	provides the value of the document Keywords as defined in the Master Template
[[Comments]]	provides the value of the document Comments as defined in the Master Template
[[Hyperlinkbase]]	provides the value of the document Hyperlinkbase as defined in the Master Template

With the exception of the [[HostApplicationFullName]] variable, all variables are dependent on the document used in the output process. Many of the variable definitions are defined in the appropriate Master Template.

CHAPTER 3

Creating OOMI-Files

Creation

The OOMI-files are located in the OutputMethodInstructions Folder within your Solution folder.The XML-text of the OOMI-file can be created/edited using a simple text editor, such as the application «notepad.exe». Of course, you can also use your favorite XML editor application. The easiest way to create a new OOMI-file is to duplicate an existing file and make your changes within the copied file.

CHAPTER 4

Integrating OOMI-Files

Integration into an officeatwork solution

For an OOMI-file to be accessible by an output variant, it needs to be registered in the officeatwork solution. A detailed description on how to integrate OOMI-Files can be found in the «Solution-Manager Manual».

Integration in output variations

For an output variation to utilize an OOMI-file, the OOMI-file needs to be registered in the output variation. It is possible to register multiple OOMI-files in the same output variation. In this case the sequence of the registration is responsible for the order in which officeatwork picks the OOMI-file when executing the output variation. A detailed description on how to register OOMI-Files within an output variation can be found in the «Solution-Manager Manual».

C H A P T E R 5

Samples

Sample 1: Adobe Acrobat (Save PDF)

The following sample uses the «Adobe PDF» printer driver.

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <Action>
      <Print ResultFilename="[[Path]]\[[Filename]].pdf" PrinterName="Adobe PDF"/>
   </Action>
</OutputMethodInstruction>
```

Sample 2: Adobe Acrobat (Send PDF)

The following sample uses the «Adobe PDF» printer driver. In this case the path and filename are taken from the active document. The path and filename are additionally written into the registry. This will allow officeatwork to attach the pdf to an email later in the process

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <BeforeAction RestoreSettingsAfterAction="0">
      <Registry>
         <Keys>
            <Key HKEY="HKEY_CURRENT_USER" Path="Software\Adobe\Acrobat Distiller\PrinterJobControl">
               <Values>
                  <Value Name="[[HostApplicationFullName]]" Type="String" Value="[[Path]]\[[Filename]
].pdf"/>
               </Values>
            </Key>
         </Keys>
      </Registry>
   </BeforeAction>
   <Action>
      <Print ResultFilename="[[Path]]\[[Filename]].pdf" PrinterName="Adobe PDF"/>
   </Action>
</OutputMethodInstruction>
```

Sample 3: PDF Creator

The following sample uses the «PDF Creator» printer driver. In this case the filename is taken from the active document. The final PDF file is saved to the desktop. Various registry entries are set in the «BeforeAction» section. Those variables will be used by the PDF driver when creating the PDF file. After completion of the «BeforeAction» section, all changes will be reverted to their original state as «RestoreSettingsAfterAction» is set to true.

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <BeforeAction RestoreSettingsAfterAction="-1">
      <Registry>
         <Keys>
            <Key HKEY="HKEY_CURRENT_USER" Path="Software\PDFCreator\Program" >
               <Values>
                  <Value Name="RunProgramAfterSaving" Value="0"></Value>
                  <Value Name="RunProgramBeforeSaving" Value="0"></Value>
                  <Value Name="UseAutosave" Value="1"></Value>
                  <Value Name="UseAutosaveDirectory" Value="1"></Value>
                  <Value Name="AutosaveDirectory" Value="%Desktop%"></Value>
                  <Value Name="AutosaveFilename" Value="[[Filename]]"></Value>
                  <Value Name="AutosaveFormat" Value="0"></Value>
               </Values>
            </Key>
            <Key HKEY="HKEY_CURRENT_USER" Path="Software\PDFCreator\Printing" >
               <Values>
                  <Value Name="StandardAuthor" Value="[[Author]]"></Value>
                  <Value Name="UseStandardAuthor" Value="1"></Value>
                  <Value Name="StandardTitle" Value="[[Title]]"></Value>
                  <Value Name="StandardSubject" Value="[[Subject]]"></Value>
                  <Value Name="StandardKeywords" Value="[[Keywords]]"></Value>
               </Values>
            </Key>
         </Keys>
      </Registry>
   </BeforeAction>
   <Action>
      <Print ResultFilename="%Desktop%\[[Filename]].pdf" PrinterName="PDFCreator"/>
   </Action>
</OutputMethodInstruction>
```

Sample 4: BullZip PDF

The following sample uses the «BullZip PDF» printer driver. In this case the filename is taken from the active document. The final PDF file is saved to the desktop. Various registry entries are set in the «BeforeAction» section. Those variables will be used by the PDF driver when creating the PDF file. After completion of the «BeforeAction» section, all changes will be reverted to their original state as «RestoreSettingsAfterAction» is set to true.

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <BeforeAction RestoreSettingsAfterAction="-1">
      <IniFiles>
         <IniFile Path="%AppData%\Bullzip\PDF Printer" Filename="runonce.ini">
            <Sections>
               <Section Name="PDF Printer">
                  <Entries>
                     <Entry Name="Output" Value="%Desktop%\[ [Filename] ].pdf"/>
                     <Entry Name="ConfirmOverwrite" Value="no"/>
                     <Entry Name="ShowSettings" Value="never"/>
                     <Entry Name="ShowPDF" Value="yes"/>
                     <Entry Name="WatermarkText" Value="[ [Title] ], [ [Author] ]"/>
                     <Entry Name="WatermarkSize" Value="5"/>
                     <Entry Name="WatermarkRotation" Value="55"/>
                  </Entries>
               </Section>
            </Sections>
         </IniFile>
      </IniFiles>
   </BeforeAction>
   <Action>
      <Print ResultFilename="%Desktop%\[[Filename]].pdf" PrinterName="BullZip PDF Printer"/>
   </Action>
</OutputMethodInstruction>
```

Sample 5: PDF Factory Save

The following sample uses the «PDF Factory» printer driver. In this case the path and filename are taken from the active document. The registry entry «ShowDlg» is set in the «BeforeAction» section. Those variables will be used by the PDF driver when creating the PDF file. After completion of the «BeforeAction» section, all changes will be reverted to their original state as «RestoreSettingsAfterAction» is set to true.

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <BeforeAction RestoreSettingsAfterAction="-1">
      <Registry>
         <Keys>
            <Key HKEY="HKEY_CURRENT_USER" Path="Software\FinePrint
Software\pdfFactory3\FinePrinters\,,Server,pdfFactory">
               <Values>
                  <Value Name="ShowDlg" Value="1"></Value>
               </Values>
            </Key>
         </Keys>
      </Registry>
   </BeforeAction>
   <Action>
      <Print ResultFilename="[[Path]]\[[Filename]].pdf" PrinterName="\\Server\pdfFactory"/>
   </Action>
</OutputMethodInstruction>
```

Note: PDF Factory will replace the «,» in the registry entries with «\» before using the entry.

Sample 6: PDF Factory Send

The following sample uses the «PDF Factory» printer driver. In this case the path and filename are taken from the active document. This will allow officeatwork to pick up the finished PDF document and attach it to an email. Various registry entries are set in the «BeforeAction» section. Those variables will be used by the PDF driver when creating the PDF file. After completion of the «BeforeAction» section, all changes will be reverted to their original state as «RestoreSettingsAfterAction» is set to true.

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <BeforeAction RestoreSettingsAfterAction="-1">
      <Registry>
         <Keys>
            <Key HKEY="HKEY_CURRENT_USER" Path="Software\FinePrint
Software\pdfFactory3\FinePrinters\,,Server,pdfFactory">
               <Values>
                  <Value Name="ShowDlg" Value="2"></Value>
                  <Value Name="PdfAction" Value="0"></Value>
               </Values>
            </Key>
            <Key HKEY="HKEY_CURRENT_USER" Path="Software\FinePrint Software\pdfFactory3">
               <Values>
                  <Value Name="OutputFile" Value="[[Path]]\[[Filename]].pdf"></Value>
               </Values>
            </Key>
         </Keys>
      </Registry>
   </BeforeAction>
   <Action>
      <Print ResultFilename="[[Path]]\[[Filename]].pdf" PrinterName="\\Server\pdfFactory"/>
   </Action>
</OutputMethodInstruction>
```

Note: PDF Factory will replace the «,» in the registry entries with «\» before using the entry.

Sample 7: FreePDF XP Save

The following sample uses the «FreePDF XP» printer driver. In this case the filename is taken from the active document. The final PDF file is saved to the desktop. Various registry entries are set in the «BeforeAction» section. Those variables will be used by the PDF driver when creating the PDF file. After completion of the «BeforeAction» section, all changes will be reverted to their original state as «RestoreSettingsAfterAction» is set to true.

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <BeforeAction RestoreSettingsAfterAction="-1">
      <Registry>
         <Keys>
            <Key HKEY="HKEY_CURRENT_USER" Path="Software\shbox\FreePdfXp" >
               <Values>
                  <Value Name="ADefault" Value="desktop"></Value>
                  <Value Name="Desktop" Value="%desktop%"></Value>
                  <Value Name="OpenPDF" Value="0"></Value>
                  <Value Name="psFor" Value="[[Author]]"></Value>
               </Values>
            </Key>
         </Keys>
      </Registry>
   </BeforeAction>
   <Action>
      <Print ResultFilename="%Desktop%\[[Filename]].pdf" PrinterName="FreePDF XP"/>
   </Action>
</OutputMethodInstruction>
```

Sample 8: FreePDF XP Send

The following sample uses the «FreePDF XP» printer driver. In this case the filename is taken from the active document. Various registry entries are set in the «BeforeAction» section. Those variables will be used by the PDF driver when creating the PDF file. After completion of the «BeforeAction» section, all changes will be reverted to their original state as «RestoreSettingsAfterAction» is set to true.

```
<?xml version="1.0" encoding="UTF-8"?>
<OutputMethodInstruction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <BeforeAction RestoreSettingsAfterAction="-1">
      <Registry>
         <Keys>
            <Key HKEY="HKEY_CURRENT_USER" Path="Software\shbox\FreePdfXp" >
               <Values>
                  <Value Name="ADefault" Value="email"></Value>
                  <Value Name="OpenPDF" Value="0"></Value>
                  <Value Name="psFor" Value="[[Author]]"></Value>
               </Values>
            </Key>
         </Keys>
      </Registry>
   </BeforeAction>
   <Action>
      <Print ResultFilename="[[Path]]\[[Filename]].pdf" PrinterName="FreePDF XP"/>
   </Action>
</OutputMethodInstruction>
```

Note: The «FreePDF XP» driver will not accept a predefined filename. That is why we cannot use officeatwork to generate the email message and attach the pdf document as it does not know the name of the pdf document to attach. Moreover, we use the email function built into the «FreePDF XP» driver to create the email itself. This way the subject and body of the mail message cannot be set by officeatwork. This is, of course, subject to change depending on what the developers of «FreePDF XP» have in mind for future versions of their driver.

Appendix

LCID's

Language	ID
Afrikaans - South Africa	1078
Albanian - Albania	1052
Amharic - Ethiopia	1118
Arabic - Saudi Arabia	1025
Arabic - Algeria	5121
Arabic - Bahrain	15361
Arabic - Egypt	3073
Arabic - Iraq	2049
Arabic - Jordan	11265
Arabic - Kuwait	13313
Arabic - Lebanon	12289
Arabic - Libya	4097
Arabic - Morocco	6145
Arabic - Oman	8193
Arabic - Qatar	16385
Arabic - Syria	10241
Arabic - Tunisia	7169
Arabic - U.A.E.	14337
Arabic - Yemen	9217
Armenian - Armenia	1067
Assamese	1101
Azeri (Cyrillic)	2092
Azeri (Latin)	1068
Basque	1069
Belarusian	1059
Bengali	1093
Bengali (Bangladesh)	2117
Bosnian (Bosnia/Herzegovina)	5146
Bulgarian	1026
Burmese	1109
Catalan	1027
Cherokee - United States	1116
Chinese - People's Republic of China	2052
Chinese - Singapore	4100
Chinese - Taiwan	1028
Chinese - Hong Kong SAR	3076

Language	ID
Chinese - Macao SAR	5124
Croatian	1050
Croatian (Bosnia/Herzegovina)	4122
Czech	1029
Danish	1030
Divehi	1125
Dutch - Netherlands	1043
Dutch - Belgium	2067
Edo	1126
English - United States	1033
English - United Kingdom	2057
English - Australia	3081
English - Belize	10249
English - Canada	4105
English - Caribbean	9225
English - Hong Kong SAR	15369
English - India	16393
English - Indonesia	14345
English - Ireland	6153
English - Jamaica	8201
English - Malaysia	17417
English - New Zealand	5129
English - Philippines	13321
English - Singapore	18441
English - South Africa	7177
English - Trinidad	11273
English - Zimbabwe	12297
Estonian	1061
Faroese	1080
Farsi	1065
Filipino	1124
Finnish	1035
French - France	1036
French - Belgium	2060
French - Cameroon	11276
French - Canada	3084

Language	ID
French - Democratic Rep. of Congo	9228
French - Cote d'Ivoire	12300
French - Haiti	15372
French - Luxembourg	5132
French - Mali	13324
French - Monaco	6156
French - Morocco	14348
French - North Africa	58380
French - Reunion	8204
French - Senegal	10252
French - Switzerland	4108
French - West Indies	7180
Frisian - Netherlands	1122
Fulfulde - Nigeria	1127
FYRO Macedonian	1071
Gaelic (Ireland)	2108
Gaelic (Scotland)	1084
Galician	1110
Georgian	1079
German - Germany	1031
German - Austria	3079
German - Liechtenstein	5127
German - Luxembourg	4103
German - Switzerland	2055
Greek	1032
Guarani - Paraguay	1140
Gujarati	1095
Hausa - Nigeria	1128
Hawaiian - United States	1141
Hebrew	1037
Hindi	1081
Hungarian	1038
Ibibio - Nigeria	1129
Icelandic	1039
Igbo - Nigeria	1136
Indonesian	1057
Inuktitut	1117
Italian - Italy	1040

Language	ID
Italian - Switzerland	2064
Japanese	1041
Kannada	1099
Kanuri - Nigeria	1137
Kashmiri	2144
Kashmiri (Arabic)	1120
Kazakh	1087
Khmer	1107
Konkani	1111
Korean	1042
Kyrgyz (Cyrillic)	1088
Lao	1108
Latin	1142
Latvian	1062
Lithuanian	1063
Malay - Malaysia	1086
Malay - Brunei Darussalam	2110
Malayalam	1100
Maltese	1082
Manipuri	1112
Maori - New Zealand	1153
Marathi	1102
Mongolian (Cyrillic)	1104
Mongolian (Mongolian)	2128
Nepali	1121
Nepali - India	2145
Norwegian (Bokmål)	1044
Norwegian (Nynorsk)	2068
Oriya	1096
Oromo	1138
Papiamentu	1145
Pashto	1123
Polish	1045
Portuguese - Brazil	1046
Portuguese - Portugal	2070
Punjabi	1094
Punjabi (Pakistan)	2118
Quecha - Bolivia	1131

Language	ID
Quecha - Ecuador	2155
Quecha - Peru	3179
Rhaeto-Romanic	1047
Romanian	1048
Romanian - Moldava	2072
Russian	1049
Russian - Moldava	2073
Sami (Lappish)	1083
Sanskrit	1103
Sepedi	1132
Serbian (Cyrillic)	3098
Serbian (Latin)	2074
Sindhi - India	1113
Sindhi - Pakistan	2137
Singhalese - Sri Lanka	1115
Slovak	1051
Slovenian	1060
Somali	1143
Sorbian	1070
Spanish - Spain (Modern Sort)	3082
Spanish - Spain (Traditional Sort)	1034
Spanish - Argentina	11274
Spanish - Bolivia	16394
Spanish - Chile	13322
Spanish - Colombia	9226
Spanish - Costa Rica	5130
Spanish - Dominican Republic	7178
Spanish - Ecuador	12298
Spanish - El Salvador	17418
Spanish - Guatemala	4106
Spanish - Honduras	18442
Spanish - Latin America	58378
Spanish - Mexico	2058
Spanish - Nicaragua	19466
Spanish - Panama	6154
Spanish - Paraguay	15370
Spanish - Peru	10250
Spanish - Puerto Rico	20490

Language	ID
Spanish - United States	21514
Spanish - Uruguay	14346
Spanish - Venezuela	8202
Sutu	1072
Swahili	1089
Swedish	1053
Swedish - Finland	2077
Syriac	1114
Tajik	1064
Tamazight (Arabic)	414
Tamazight (Latin)	1119
Tamil	1097
Tatar	1092
Telugu	1098
Thai	1054
Tibetan - Bhutan	2129
Tibetan - People's Republic of China	1105
Tigrigna - Eritrea	2163
Tigrigna - Ethiopia	1139
Tsonga	1073
Tswana	1074
Turkish	1055
Turkmen	1090
Uighur - China	1152
Ukrainian	1058
Urdu	1056
Urdu - India	2080
Uzbek (Cyrillic)	2115
Uzbek (Latin)	1091
Venda	1075
Vietnamese	1066
Welsh	1106
Xhosa	1076
Yi	1144
Yiddish	1085
Yoruba	1130
Zulu	1077

OutputMethodInstruction schema

CHAPTER 6

Support

Get access to a wide range of support resources on officeatwork Connect (connect.officeatwork.com) such as:

- Knowledge Base
- Q & A
- Download Center
- Installers
- Manuals
- Video guides
- Forum
- Glossary
- etc.

To access officeatwork Connect you need to register your Microsoft-Account at www.officeatwork.com → Connect

---

All support options and resources can be found on the website www.officeatwork.com → Support

More services offered by officeatwork such as Education and Consulting can be found on the website www.officeatwork.com → Services

Index

officeatwork AG
Bundesplatz 12
6300 Zug, Switzerland

T +41 41 544 7100

www.officeatwork.com
mail@officeatwork.com