Dokumentation von Team Hahner® – Engineers of (Word) Solutions
Lizenzierte Version für die Besucher der Internetseite von Team Hahner®

Word(-Bug): Dateien in „fremden“ Formaten aus Word heraus via VBA-Aufrufen

Dieses Dokument ist gemäß den Vorgaben des BFSG, BITV 2.0 und WCAG 2.1 barrierefrei.

Version **1.01**

Autor **Dipl.-Ing. (FH) Markus Hahner**
Team Hahner® – Engineers of (Word) Solutions
www.hahner.de | www.schauen-statt-lesen.de |
www.office-sicherheit.de

Diese Dokumentation enthält vertrauliche und/oder rechtlich geschützte Informationen und ist ausschließlich für den auf der Titelseite und den Fußzeilen dieser Dokumentation benannten Empfänger(kreis) bestimmt. Das unerlaubte Kopieren, die (teilweise) Entnahme von Inhalten sowie die unbefugte Weitergabe dieser Dokumentation oder von Ausschnitten in jeglicher Form ist nicht gestattet. Diese Dokumentation wurde vom Autor ohne KI erstellt.

In dieser Dokumentation wird darauf verzichtet, bei Personenbezeichnungen sowohl die weibliche, die männliche und die diverse Form zu nennen. Das generische Maskulinum umfasst hier generell alle unabhängig vom Geschlecht.

Hahner®
Engineers of Word Solutions

Inhaltsverzeichnis

1. Öffnen von Dateien in „fremden" Formaten via VBA aus Word heraus 4
2. Alternative zur ShellExecute-API 8
3. Verhalten anderer API-Deklarationen 9
4. Erweiterungen/Versionen 10
4.1 Geplante Erweiterungen dieser Dokumentation 10
4.2 Versionsverlauf der Dokumentation 10
Abbildungsverzeichnis 11
Über den Autor/Trainer | Kontakt 12
Impressum 13

Sämtliche Screenshots, Kommandos und Einstellungen dieser Dokumentation basieren auf Windows 11/10 und dem Microsoft 365-Paket, gelten aber im Wesentlichen auch für aktuelle Office-Versionen wie Office 2024.

Bei Office 365 dient die jeweils aktuelle Version mit dem Update-Kanal „Aktueller Kanal" als Grundlage. Die dort veröffentlichten Funktionen fließen zeitversetzt auch in die anderen Update-Kanäle wie „Monatlicher Enterprise-Kanal" von Office 365 ein. Die LTSC-Versionen von Office 2024 und Office 2021 enthalten im Vergleich zur Kaufversion kaum Programmaktualisierungen jedoch regelmäßige Sicherheitsupdates, sie basieren alle wie die Version Office 365 auf der C2R-Installationsart.

Das gleiche gilt auch für die von Microsoft nicht weiter unterstützte Volume-License-Version von Office 2019. Bei der ebenfalls nicht mehr unterstützten Version Office 2016 wird je nach Funktion zwischen der Kaufversion und der Volume License („Firmen-Lizenz") unterschieden. Die Volume-License-Version verfügt im Vergleich zur Kaufversion über eine andere Installationsart (MSI anstelle von C2R) und enthält weniger Programmaktualisierungen, jedoch wie die Kaufversion auch regelmäßige Sicherheitsupdates.

Die Office-Versionen 2021, 2019, 2016, 2013, 2010 und 2007 sind entsprechend nur noch teilweise Bestandteil dieser Dokumentation.

Supportzeitraum von Microsoft (siehe https://docs.microsoft.com/de-de/lifecycle/):

- Der erweiterte Support für Office 2007 endete am 10. Oktober 2017.
- Der erweiterte Support für Office 2010 endete am 13. Oktober 2020.
- Der erweiterte Support für Office 2013 endete am 11. April 2023.
- Der erweiterte Support für Office 2016 und Office 2019 endete am 14. Oktober 2025.
- Der erweiterte Support für Office 2021 endete am 12. Oktober 2026 (Kaufversion sowie LTSC-Version).

- Der erweiterte Support für Office 2024 endet am 9. Oktober 2029 (Kaufversion sowie LTSC-Version).
- Der erweiterte Support für Windows 10 endete am 14. Oktober 2025 – bei aktuellstem Release 22H2 (sowohl für Enterprise, Pro, Home und Education). Unter den von Microsoft festgelegten Bestimmungen werden ein weiteres Jahr lang Updates zur Verfügung gestellt. Bei entsprechender Registrierung (benötigt ein Microsoft-Konto) kann der Support bis zum 12. Oktober 2027 kostenfrei verlängert werden.
- Der erweiterte Support für Windows 10 Enterprise LTSC 2021 endet am 12. Januar 2027.
- Das endgültige Support-Ende von Windows 11 ist noch nicht festgelegt. Je nach installiertem Release gibt es für diese Releases bereits definierte Support-Endtermine (siehe https://learn.microsoft.com/de-de/lifecycle/products/windows-11-home-and-pro).

Alle Pfadangaben basieren auf der zum Zeitpunkt der Veröffentlichung dieses Dokumentes (20.10.2026) aktuellen Windows-11-Pro-Version.

Hahner



1. Öffnen von Dateien in „fremden“ Formaten via VBA aus Word heraus

Aus einem VBA-basierenden Add-In beispielsweise eine PDF-Datei öffnen? Dies war bislang mithilfe der API-Funktion **ShellExecute** problemlos möglich – sowohl in der 32- als auch der 64-Bit-Version von Word. Für den Aufruf musste lediglich die API wie folgt deklariert werden:

```
Public Declare PtrSafe Function ShellExecute Lib "shell32.dll" Alias "ShellExecuteA" ( _
               ByVal hwnd As LongPtr, _
               ByVal lpOperation As String, _
               ByVal lpFile As String, _
               ByVal lpParameters As String, _
               ByVal lpDirectory As String, _
               ByVal nShowCmd As LongPtr) As Long
```

Dank des Attributs **PtrSafe** arbeitet die API-Deklaration sowohl in der 32- als auch der 64-Bit-Version, es muss also nicht mithilfe einer bedingten Kompilierung im Stil von

```
#If Win64 Then
    <64-Bit-Deklaration>
#Else
    <32-Bit-Deklaration>
#End If
```

zwischen der Bitanzahl unterschieden werden.

Aufgerufen wird die API beispielsweise mit folgender Routine, der als Parameter der Pfad (**svFilepath**) und der Dateiname (**svFilename**) übergeben werden:

```
Sub pLoadExternalFile(ByVal svFilepath As String, ByVal svFilename As String)
    Dim lReturn as Long

    On Error Resume Next
    lReturn = ShellExecute(hwnd:=0, _
                           lpOperation:="Open", _
                           lpFile:=svFilename, _
                           lpParameters:="", _
                           lpDirectory:=svFilepath, _
                           nShowCmd:=3)
    On Error GoTo 0
End Sub
```

Über den Rückgabewert **lReturn** kann bei Bedarf geprüft werden, ob das Ausführen der API erfolgreich war. Ist der Wert kleiner oder gleich 32, ist ein Fehler aufgetreten. Ist der Wert größer 32, war die Ausführung erfolgreich.

Seit Anfang/Mitte Oktober (je nach Microsoft 365-Update-Kanal) liefert die API als Rückgabewert **0** und öffnet somit die übergebene Datei nicht mehr.

Wird die Routine **pLoadExternalFile** innerhalb des VBA-Codes – wie in der nachfolgenden Routine **pTest** zu sehen – aufgerufen, passiert entsprechend nichts. Die PDF-Datei wird nicht geöffnet.

```
Sub pTest()
    Dim sPath as String
    Dim sFilename as String

    sPath = "D:\Daten"
    sFilename = "Markus Hahner Testdatei.pdf"

    pLoadExternalFile svFilepath:=sPath, svFilename:=sFilename
End Sub
```

Andere API-Deklarationen – wie in Kapitel 3 „Verhalten anderer API-Deklarationen“ aufgeführt – arbeiten hingegen nach wie vor fehlerfrei.

Hinweis: Das in diesem Kapitel dokumentierte Verhalten ist ausschließlich in Word 365 und Word 2024 (Kaufversion) zu beobachten. Die API **`ShellExecute`** arbeitet bislang sowohl in Excel 365 und PowerPoint 365 mit den in diesem Kapitel beschriebenen Aufrufen fehlerfrei. Auch in Word 2024 LTSC ist das Verhalten nicht zu erkennen.

Die nachfolgende Übersicht zeigt die Konto-Informationen der betroffenen/nicht betroffenen Versionen:

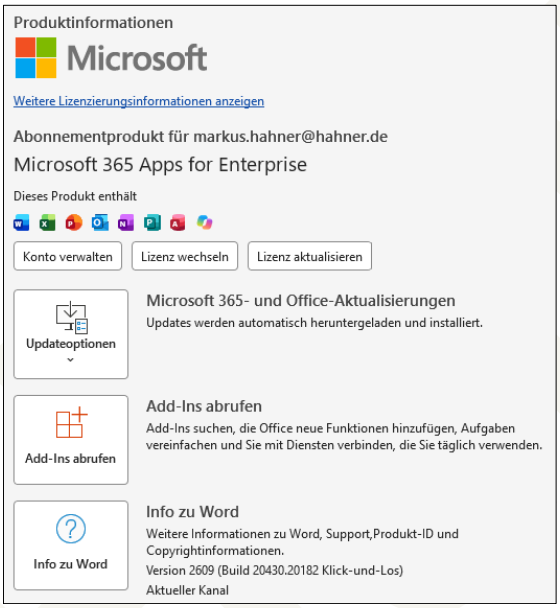


Abbildung 1: Konto-Informationen von Microsoft 365 Apps for Enterprise, Aktueller Kanal
Aktuell **betroffene** Version

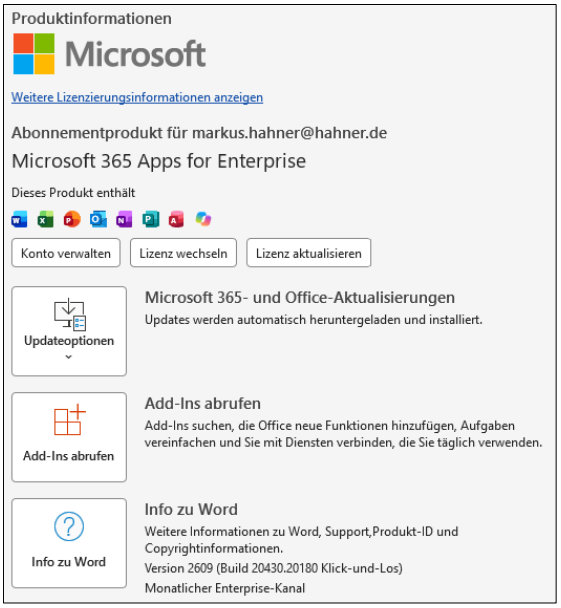


Abbildung 2: Konto-Informationen von Microsoft 365 Apps for Enterprise, Monatlicher Enterprise-Kanal
Aktuell **betroffene** Version

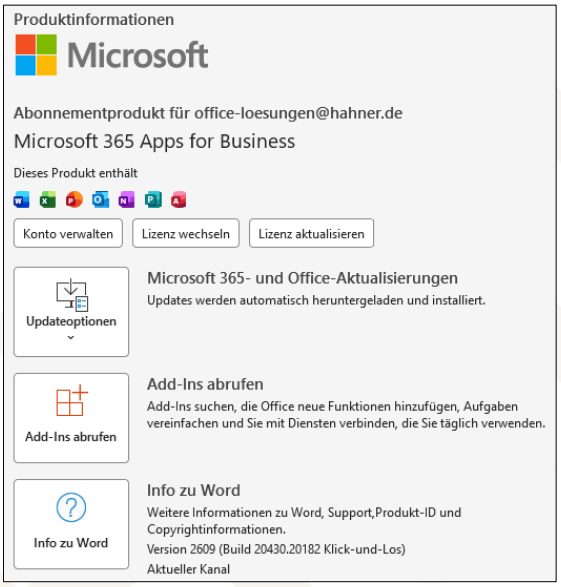


Abbildung 3: Konto-Informationen von Microsoft 365 Apps for Business, Aktueller Kanal
Aktuell **betroffene** Version



Abbildung 4: Konto-Informationen von Microsoft Office Home & Business 2024, Aktueller Kanal
Aktuell **betroffene** Version



Abbildung 5: Konto-Informationen von Microsoft Office LTSC Professional Plus 2024
Aktuell **nicht betroffene** Version

2. Alternative zur ShellExecute-API

Um Dateien in fremden Formaten dennoch via VBA zu öffnen, kann als Ersatz zur ShellExecute-API das WScript.Shell-Objekt zum Einsatz kommen. Das Objekt ist eine Windows-Script-Host-Komponente („WSH") und kann direkt in VBA deklariert und aufgerufen werden.

Die folgende Routine ersetzt die gleichnamige Routine aus Kapitel 1 „Öffnen von Dateien in „fremden" Formaten via VBA aus Word heraus". Ihr wird sowohl der Pfad (**svFilepath**) als auch der Dateiname (**svFilename**) übergeben, der dann innerhalb der Routine unter Berücksichtigung eines eventuell vorhandenen Backslash am Ende des Pfades zu einem String zusammengebaut wird.

```
Sub pLoadExternalFile(ByVal svFilepath As String, ByVal svFilename As String)
    Dim oWScriptShell As Object

    On Error Resume Next
    If Right(svFilepath, 1) <> "\" Then svFilepath = svFilepath & "\"
    Set oWScriptShell = CreateObject("WScript.Shell")
    oWScriptShell.Run Chr(34) & svFilepath & svFilename & Chr(34)
    Set oWScriptShell = Nothing
    On Error GoTo 0
End Sub
```

Da das Leerzeichen im Pfad oder Dateinamen zu einer Fehlfunktion führen würde, wird der komplette Pfad in doppelte Anführungszeichen (**Chr(34)**) gesetzt; dies ist beim Aufruf der ShellExecute-API nicht notwendig.

3. Verhalten anderer API-Deklarationen

Die in Kapitel 1 „Öffnen von Dateien in „fremden" Formaten via VBA aus Word heraus" dokumentierte Einschränkung (Bug?) konnte im Test nur bei API `ShellExecute` beobachtet werden. Andere API-Aufrufe wie die nachfolgenden zum Auslesen der Registry arbeiten auch mit den Anfang/Mitte Oktober veröffentlichten Word-365-Updates fehlerfrei.

```
Public Declare PtrSafe Function RegOpenKey Lib "advapi32.dll" Alias "RegOpenKeyA" ( _
                ByVal hKey As LongPtr, _
                ByVal lpSubKey As String, _
                phkResult As Long) As LongPtr

Public Declare PtrSafe Function RegCloseKey Lib "advapi32.dll" (ByVal hKey As LongPtr)
As LongPtr

Public Declare PtrSafe Function RegEnumValue Lib "advapi32.dll" Alias "RegEnumValueA" (
_
                ByVal hKey As LongPtr, _
                ByVal dwIndex As Long, _
                ByVal lpValueName As String, _
                lpcbValueName As Long, _
                ByVal lpReserved As Long, _
                lpType As Long, _
                lpData As Any, _
                lpcbData As Long) As LongPtr

Public Declare PtrSafe Function GetPrivateProfileString Lib "kernel32.dll" Alias
"GetPrivateProfileStringA" ( _
                ByVal lpApplicationName As String, _
                ByVal lpKeyName As Any, _
                ByVal lpDefault As String, _
                ByVal lpReturnedString As String, _
                ByVal nSize As LongPtr, _
                ByVal lpFileName As String) As LongPtr

Public Declare PtrSafe Function WritePrivateProfileString Lib "kernel32.dll" Alias
"WritePrivateProfileStringA" ( _
                ByVal lpApplicationName As String, _
                ByVal lpKeyName As Any, _
                ByVal lpString As Any, _
                ByVal lpFileName As String) As Long
```

4. Erweiterungen/Versionen

Nachfolgend sind geplante Erweiterungen sowie durchgeführte Änderungen dieser Dokumentation aufgeführt.

4.1 Geplante Erweiterungen dieser Dokumentation

Folgende Erweiterungen für diese Dokumentation sind bereits in Planung/Arbeit:

- -/-

4.2 Versionsverlauf der Dokumentation

Versions-nummer	Veröffentlichungs-datum	Informationen zur Version
v1.00	18.10.2026	■ Basisversion
v1.01	20.10.2026	■ Ergänzen der Routine **pLoadExternalFile** in Kapitel 2 „Alternative zur ShellExecute-API" mit dem Kommando **Set oWScriptShell = Nothing** ■ Zusätzliche englischsprachige Version

Hahner

Abbildungsverzeichnis

Abbildung 1: Konto-Informationen von Microsoft 365 Apps for Enterprise, Aktueller Kanal Aktuell **betroffene** Version 5
Abbildung 2: Konto-Informationen von Microsoft 365 Apps for Enterprise, Monatlicher Enterprise-Kanal Aktuell **betroffene** Version 6
Abbildung 3: Konto-Informationen von Microsoft 365 Apps for Business, Aktueller Kanal Aktuell **betroffene** Version 6
Abbildung 4: Konto-Informationen von Microsoft Office Home & Business 2024, Aktueller Kanal Aktuell **betroffene** Version 7
Abbildung 5: Konto-Informationen von Microsoft Office LTSC Professional Plus 2024 Aktuell **nicht betroffene** Version 7

Hahner

Über den Autor/Trainer | Kontakt



Markus Hahner

Diplom-Ingenieur (FH)

+49 7720 810046

markus.hahner@hahner.de

www.hahner.de | www.schauen-statt-lesen.de | www.office-sicherheit.de

- Projekterfahrener Ingenieur für Mittelständler, Großunternehmen, Behörden und Verbände beim Einsatz von Microsoft 365 und Office-Rollout
- Entwickler von Vorlagen-Lösungen/Vorlagen mit vielen zehntausend Installationen (siehe https://www.hahner.de/word-solutions/)
- Zertifizierter Trainer (Microsoft Certified Educator, Microsoft MIE, Microsoft Certified Trainer 2012 bis 2026) mit den Schwerpunkten Word, .NET, VBA, XML, Office-Sicherheit und Microsoft 365-Apps (Zertifizierungen siehe https://www.hahner.de/zertifizierungen/)
- Fachbuchautor mit weltweit über 50 Büchern u. a. bei Microsoft Press; LinkedIn-Learning/video2brain-Trainer zu Word (siehe DNB, Katalog der Deutschen Nationalbibliothek), Fachlektorat zahlreicher IT-Bücher
- Video-Blogger (schauen-statt-lesen.de) und Office-Blogger (hahner.de)



Impressum

© 2015 – 2026, Team Hahner® – Engineers of (Word) Solutions, Dipl.-Ing. (FH) Markus Hahner

Alle Daten urheberrechtlich geschützt. Jegliches Kopieren ist verboten. All copyrights for data reserved. Unauthorized downloading or other kinds of copying prohibited. Keine Weitergabe oder Veröffentlichung in gedruckter oder elektronischer Form ohne ausdrückliche schriftliche Genehmigung von Team Hahner® – Engineers of (Word) Solutions, Dipl.-Ing. (FH) Markus Hahner.

Team Hahner® – Engineers of (Word) Solutions
Dipl.-Ing. (FH) Markus Hahner, MCE/MCP/MIE/MOSM
Arndtstraße 28
78054 Villingen-Schwenningen
Germany

+49 7720 810046
Chatten mit Markus Hahner via Teams
info@hahner.de
www.hahner.de | www.schauen-statt-lesen.de | www.office-sicherheit.de

Sämtliche Screenshots in dieser Dokumentation wurden mit Snagit® erstellt (Details siehe www.techsmith.de/snagit.html).

CAMTASIA, SNAGIT UND TECHSMITH SIND MARKEN VON TECHSMITH CORPORATION UND IN DEN USA SOWIE IN ANDEREN LÄNDERN ALS SOLCHE EINGETRAGEN.

Maßgeschneiderte Trainings, Workshops & Online-Seminare
zu Snagit, Camtasia und Audacity | www.screencast-training.de

Version: 1.01 vom 20.10.2026