XLS PADLOCK

XLS Padlock Benutzerhandbuch

Schutz, Lizenzierung und Verteilung von Excel-Arbeitsmappen

Version 2026.0 G.D.G. Software www.xlspadlock.com

Inhaltsverzeichnis

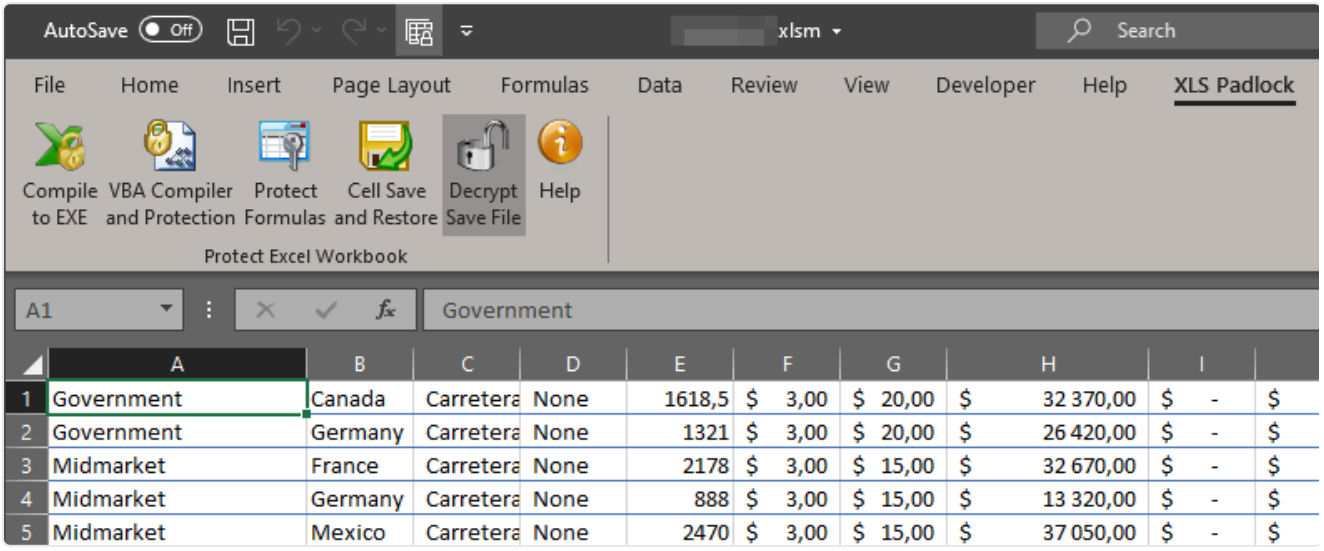
Über dieses Handbuch

Erste Ausgabe: Januar 2013. Aktualisiert: Juni 2026.

Version 2026.0, Changelog lesen

Einführung in XLS Padlock

XLS Padlock ist **eine leistungsstarke Schutz- und Lizenzierungslösung für Microsoft Excel**, die Ihre Arbeitsmappen vor dem Kopieren schützt. Es funktioniert als Compiler und ermöglicht es Ihnen, Ihre Excel-Arbeitsmappen in sichere, eigenständige Anwendungen umzuwandeln.



	A	B	C	D	E	F	G	H	I	
1	Government	Canada	Carretera	None	1618,5	$ 3,00	$ 20,00	$ 32 370,00	$ -	$
2	Government	Germany	Carretera	None	1321	$ 3,00	$ 20,00	$ 26 420,00	$ -	$
3	Midmarket	France	Carretera	None	2178	$ 3,00	$ 15,00	$ 32 670,00	$ -	$
4	Midmarket	Germany	Carretera	None	888	$ 3,00	$ 15,00	$ 13 320,00	$ -	$
5	Midmarket	Mexico	Carretera	None	2470	$ 3,00	$ 15,00	$ 37 050,00	$ -	$

Dies ermöglicht es Ihnen, Ihre Excel-Dateien sicher zu verteilen, den Benutzerzugriff zu steuern und das Kopieren von Tabellenblättern zu verhindern. Wichtige Excel-Funktionen können eingeschränkt werden, darunter:

- Das Kontextmenü des Tabellenblatts (Rechtsklick)
- Die Kopieren-/Einfügen-Funktion
- Die Bearbeitungsleiste
- Speichern und Drucken
- Der Zugriff auf das VBA-Projekt

XLS Padlock bietet mehrere Möglichkeiten, Ihre Dateien zu sichern, etwa **die Bindung an einen bestimmten USB-Stick**, die Verwendung eines **starken Dongle-Schutzes** oder die Ausgabe von **hardwaregebundenen Activation Keys**. Sie erhalten außerdem eine granulare Kontrolle über den Formelschutz, sodass Benutzer Ihre Formeln verwenden können, ohne sie anzeigen oder kopieren zu können.

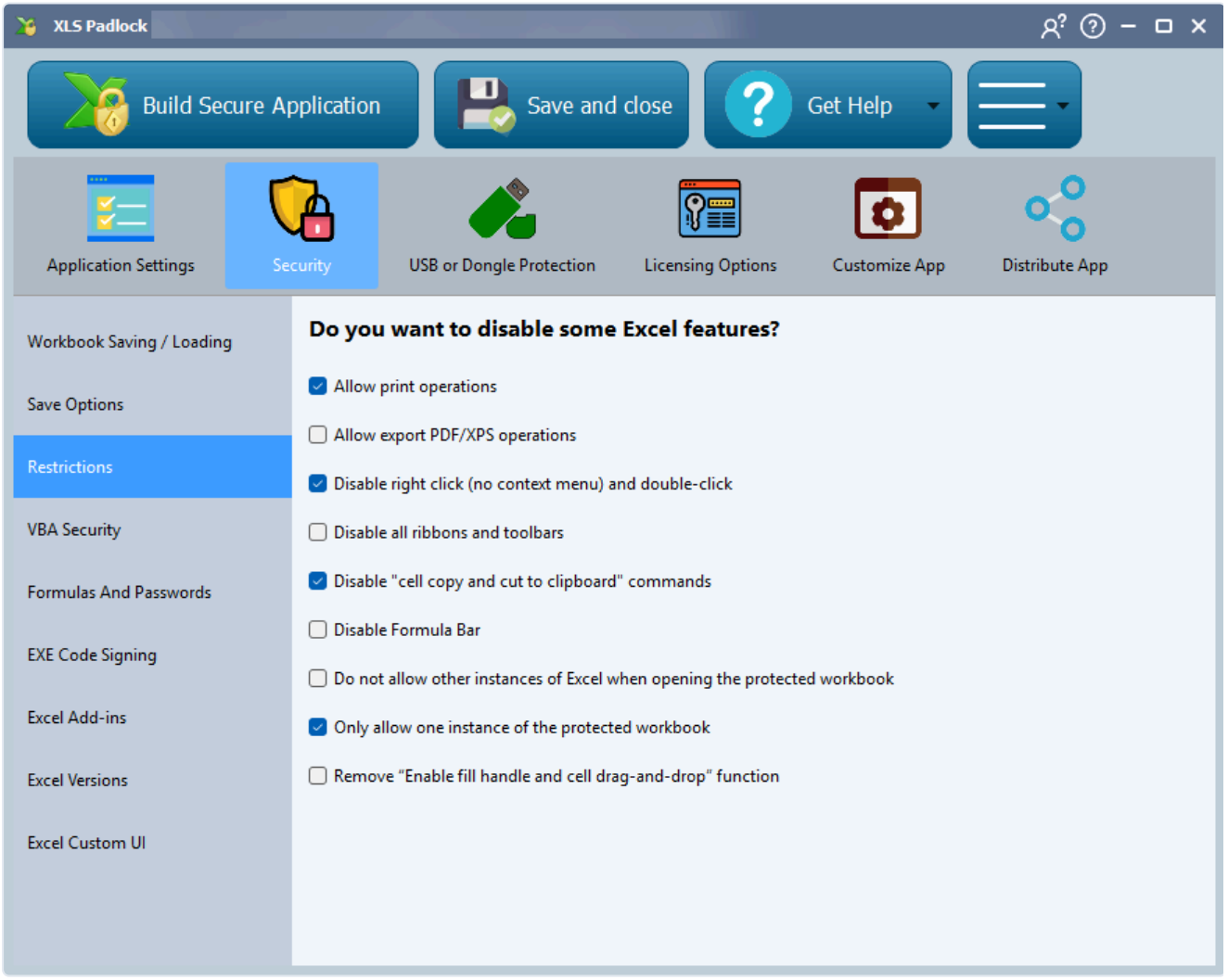
Erweiterter Schutz von VBA-Code

Mit seinem integrierten VBA Compiler ermöglicht es Ihnen XLS Padlock, **sensiblen VBA-Code in sicheren Byte-Code zu kompilieren und ihn so für Endbenutzer unzugänglich zu machen**. Ihre VBA-Makros sind wirklich geschützt, da der ursprüngliche Quellcode entfernt wird.

Der Compiler ist **kein einfacher Obfuskator**. Er wandelt VBA-Code in Binärcode um und speichert ihn sicher innerhalb der EXE der Anwendung. Um den Schutz abzuschließen, können Sie Ihr VBA-Projekt

sperren und so **Werkzeuge zum Knacken von Passwörtern wirkungslos machen**. Erfahren Sie mehr über den Schutz von VBA-Code.

Umfassende Sicherheitsoptionen



Anwendungen können so konfiguriert werden, dass sie **nach einer festgelegten Anzahl von Tagen ablaufen**, nach einer bestimmten Anzahl von Verwendungen oder an einem festen Datum. Für Testversionen kann ein anpassbarer Erinnerungsbildschirm angezeigt werden. **Die Online-Aktivierung** automatisiert den Lizenzierungsprozess und gibt Ihnen die **Fernsteuerung** darüber, wer auf Ihre Arbeitsmappen zugreifen kann.

Geschützte Arbeitsmappen-Anwendungen sind eigenständig und benötigen nur Microsoft Excel, um zu funktionieren. Alle Funktionen von Excel werden vollständig unterstützt.

👉 Sie können Ihre Anwendungen außerdem anpassen und mit Ihrer Marke versehen:

- Fügen Sie ein benutzerdefiniertes Symbol und Copyright-Informationen hinzu.
- Übersetzen Sie den gesamten für den Benutzer sichtbaren Text in jede beliebige Sprache.
- Zeigen Sie beim Start eine Lizenzvereinbarung (EULA) oder einen Begrüßungsbildschirm an.
- Signieren Sie Ihre kompilierten Arbeitsmappen digital mit Authenticode.

- Erstellen Sie ein professionelles Installationsprogramm für die Verteilung.
- Benachrichtigen Sie Benutzer über neue Versionen mit der integrierten Web-Update-Funktion.

Kompatibel mit Office 365 und Excel 2024, 2021, 2019, 2016, 2013, 2010, 2007 (SP3), 2002, sowohl 32-Bit- als auch 64-Bit-Versionen.

> **Warnung**
>
> Für die Erstellung von 64-Bit-Anwendungen ist eine 64-Bit-Version von Windows erforderlich.

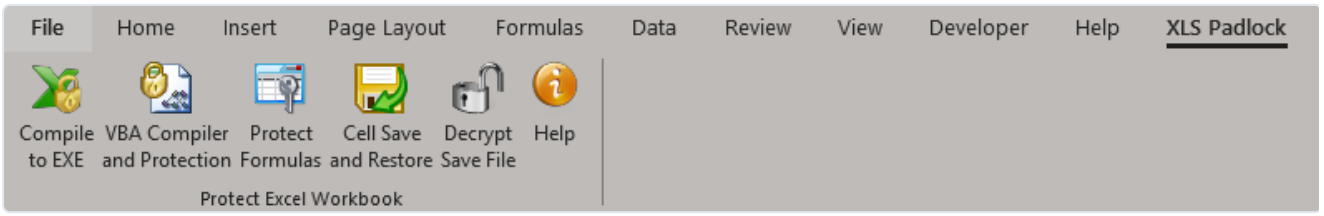
👉 Jetzt entdecken:

- XLS Padlock herunterladen und installieren
- So schützen Sie eine Excel-Arbeitsmappe
- XLS Padlock Website
- Neuigkeiten

Herunterladen und installieren

Auf unserer Website unter https://www.xlspadlock.com/download müssen Sie das Installationsprogramm auswählen, das zu Ihrer Version von Microsoft Excel passt (32-Bit oder 64-Bit). Das Installationsprogramm überprüft, ob Sie die korrekte Version installieren.

Nach einer erfolgreichen Installation **ist XLS Padlock direkt in Excel integriert** und erscheint als neue Registerkarte im Menüband. Um XLS Padlock zu verwenden, müssen Sie zunächst Ihre Excel-Arbeitsmappe öffnen.

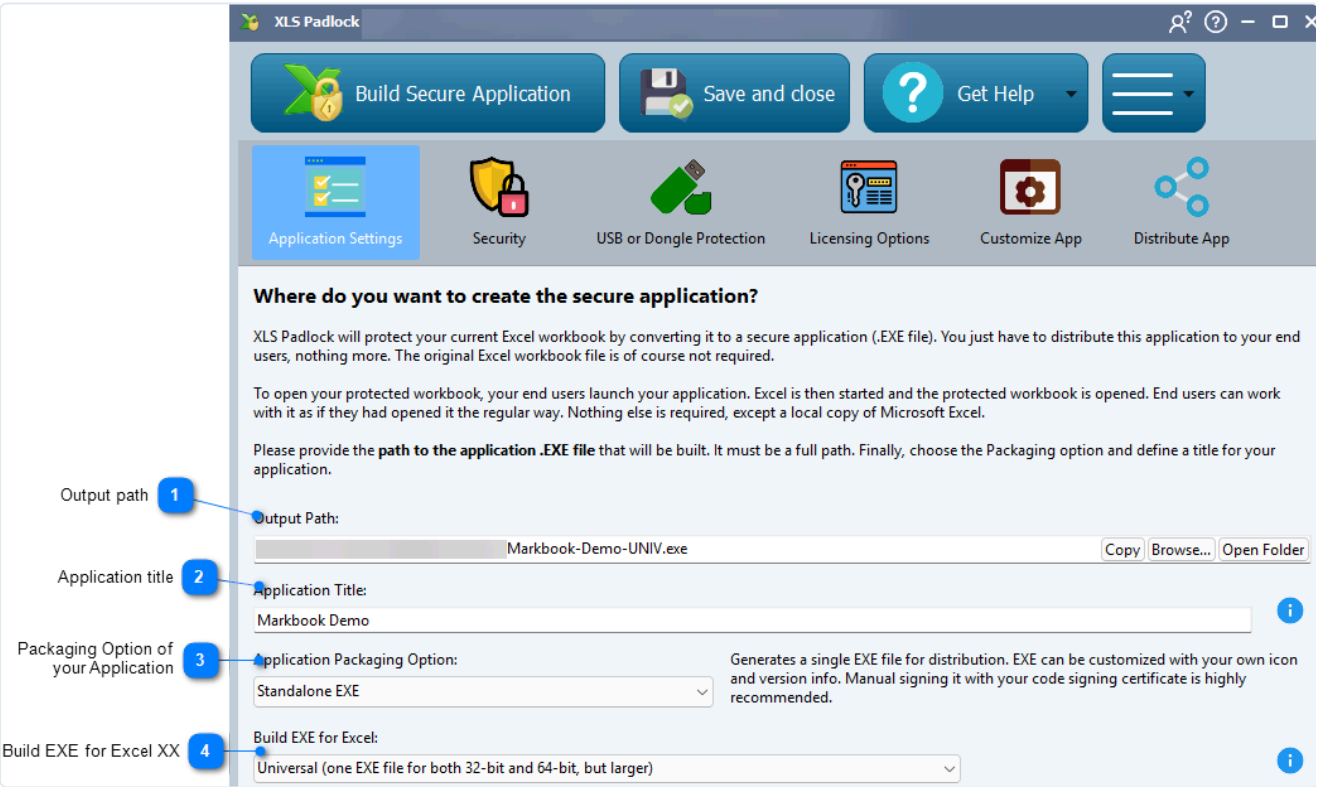


Außerdem werden zwei Verknüpfungen auf Ihrem Desktop erstellt: eine für dieses Benutzerhandbuch und eine weitere für den XLS Padlock Manager.

- XLS Padlock Manager
- So schützen Sie eine Excel-Arbeitsmappe

Anwendungseinstellungen

Die Seite Application Settings enthält die obligatorischen Einstellungen für Ihr Projekt.



Grundeinstellungen

Output Path

👉 Legt fest, wo die endgültige EXE-Datei erstellt wird.

XLS Padlock kompiliert Ihre Arbeitsmappe in eine einzige ausführbare Datei (EXE). Im Feld "Output Path" müssen Sie den vollständigen Pfad und den Dateinamen angeben, unter dem Ihre Anwendung erstellt werden soll.

> **TIPP**
>
> **Vergessen Sie nicht, die Erweiterung `.exe` im Dateinamen anzugeben.**

Relative Pfade werden ebenfalls akzeptiert; sie beziehen sich auf den Ordner, der Ihre Quell-Arbeitsmappendatei enthält. Standardmäßig schlägt XLS Padlock einen Pfad im selben Ordner wie Ihre Quell-Arbeitsmappe vor, Sie können ihn jedoch durch einen beliebigen Speicherort ersetzen.

Application Title

👉 Legt den Titel des Hauptfensters Ihrer Anwendung fest.

Sie können einen benutzerdefinierten Titel für Ihre Anwendung festlegen. Dieser Titel wird in der Titelleiste des Fensters angezeigt und ersetzt den Standardtext "Microsoft Excel".

Wenn Sie den Dateinamen einer geladenen Speicherdatei dynamisch anzeigen möchten, können Sie die folgenden Platzhalter im Titel verwenden:

- `%SAVEFILENAME%`: Wird nur durch den Dateinamen ersetzt (z. B. `MyData.xlsc`).
- `%SAVEFULLNAME%`: Wird durch den vollständigen Pfad zur Speicherdatei ersetzt (z. B. `C:\Users\Me\Documents\MyData.xlsc`).

Ein auf `My Application, %SAVEFILENAME%` gesetzter Titel würde beispielsweise als "My Application, MyData.xlsc" angezeigt, wenn der Benutzer diese Datei geöffnet hat.

> **HINWEIS**
>
> Wenn die ursprüngliche, nicht gespeicherte Arbeitsmappe geladen wird, sind die Platzhalter leer.

Application Packaging Option

Siehe Packaging Option: Wählen Sie zwischen einer eigenständigen EXE oder einem Bundle für die Verteilung.

Build EXE for Excel

Siehe Build EXE for Excel: Geben Sie an, ob für 32-Bit-, 64-Bit- oder universelle Kompatibilität kompiliert werden soll.

Zusätzliche Konfiguration

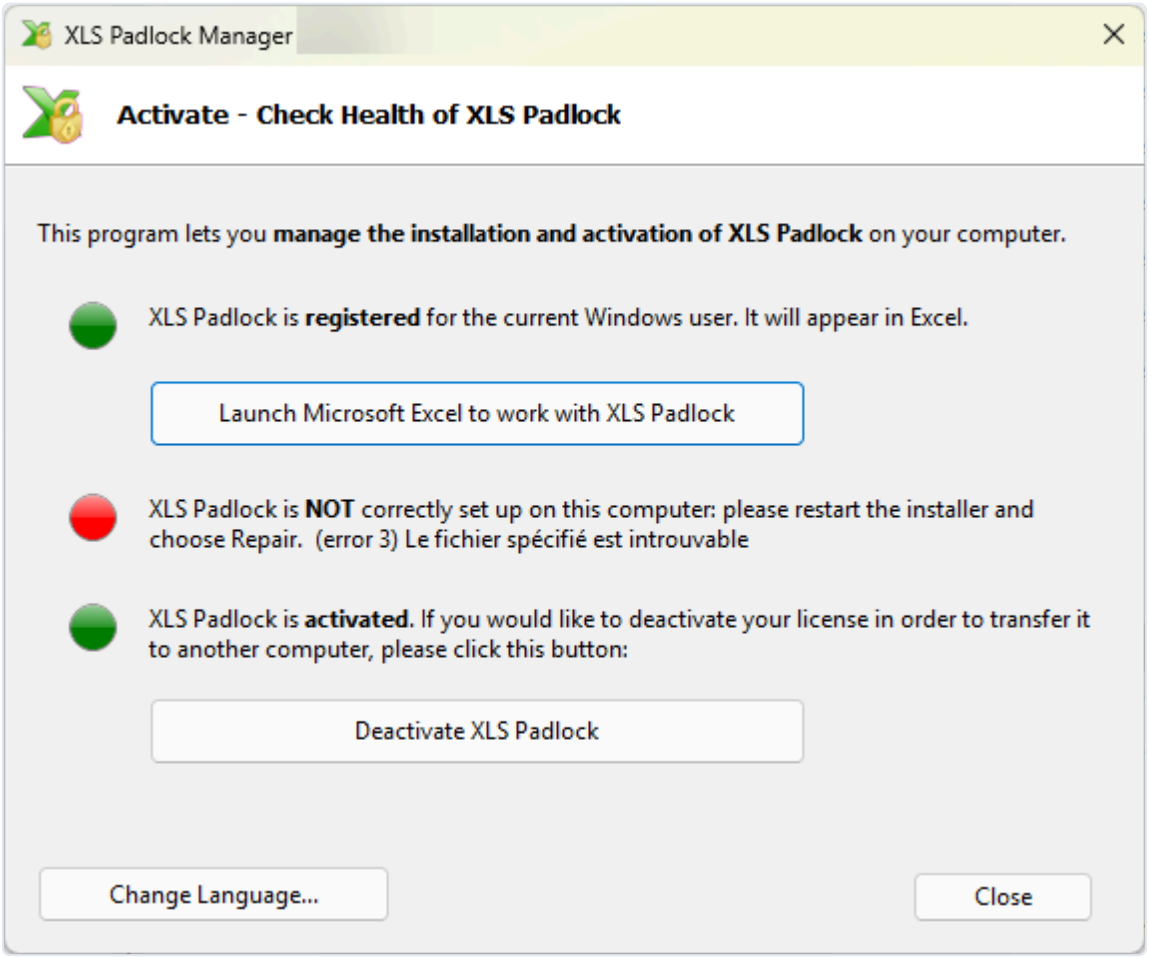
- Erweiterte Optionen konfigurieren
- Begleitdateien hinzufügen

XLS Padlock Manager

XLS Padlock Manager ist eine eigenständige Anwendung, mit der Sie **prüfen können, ob XLS Padlock korrekt eingerichtet und für den aktuellen Windows-Benutzer registriert ist**.

XLS Padlock **muss für jeden Windows-Benutzer registriert werden, der es verwenden möchte**. Wenn Sie XLS Padlock auf einem Computer installieren, wird es in der Regel nur für das Administratorkonto registriert, mit dem die Installation durchgeführt wurde. Andere Benutzerkonten müssen den XLS Padlock Manager verwenden, um die Software zu registrieren.

- Um XLS Padlock für Ihr Windows-Benutzerkonto zu registrieren, starten Sie "XLS Padlock for Excel, Manager" über Ihren Windows-Desktop. Das folgende Fenster wird angezeigt:



- Wenn alle Anzeigen grün sind, ist Ihre Installation korrekt.
- Wenn die erste Anzeige rot ist, klicken Sie auf "**Enable XLS Padlock for the current Windows user**" (XLS Padlock für den aktuellen Windows-Benutzer aktivieren). XLS Padlock wird dann registriert, und Sie können es verwenden. Für diesen Vorgang sind keine Administratorrechte erforderlich.

HINWEIS

Wenn Sie über eine XLS Padlock Lizenz verfügen, können Sie diese auch über den Manager aktivieren.

👉 So schützen Sie eine Excel-Arbeitsmappe

So schützen Sie eine Excel-Arbeitsmappe

Video-Tutorial verfügbar

Sehen Sie sich diese Video-Tutorials zu XLS Padlock von einem Excel-Experten an: https://excelvbaisfun.com/?ref=5

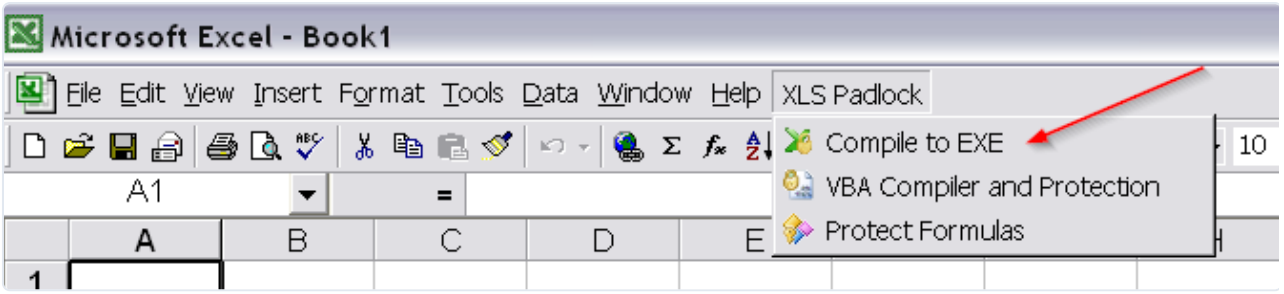
1\. Öffnen und speichern Sie Ihre Arbeitsmappe

Beginnen Sie damit, die Arbeitsmappe, die Sie schützen möchten, in Excel zu öffnen. Stellen Sie sicher, dass alle Änderungen gespeichert sind, bevor Sie fortfahren.

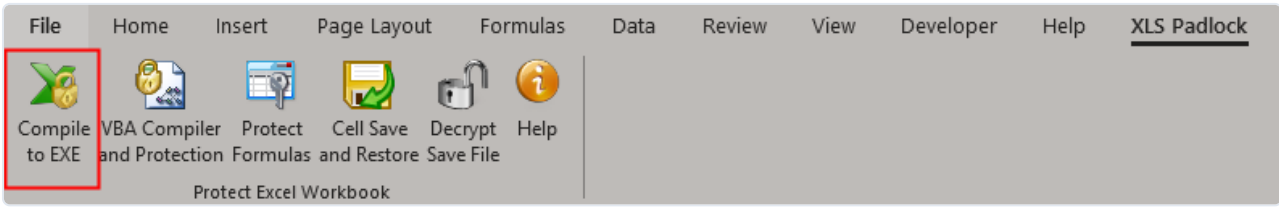
2. Konfigurieren Sie die Schutzoptionen

Öffnen Sie zunächst die XLS Padlock-Oberfläche über das Excel-Menüband:

- **Vor Excel 2007**: Verwenden Sie das Menü „XLS Padlock“.



- **Excel 2007 und höher**: Klicken Sie im Menüband-Register „XLS Padlock“ auf „Secure Compile to Exe“.



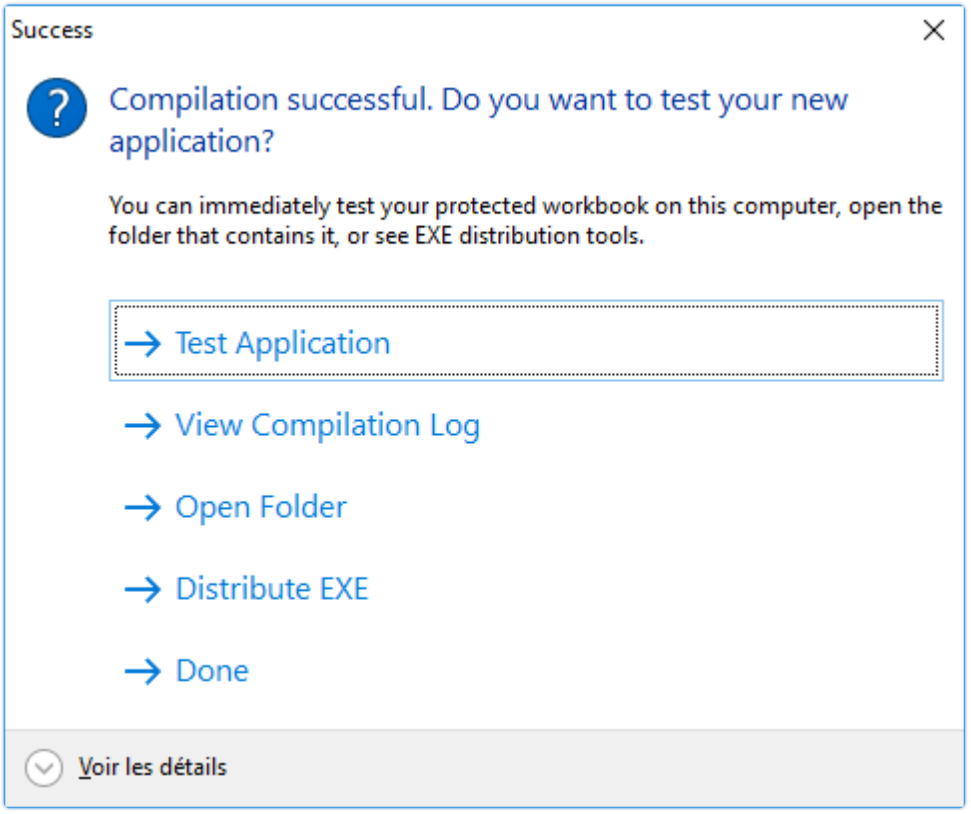
HINWEIS

Speichern Sie Ihre Excel-Arbeitsmappe immer, bevor Sie die XLS Padlock-Oberfläche öffnen, damit alle aktuellen Änderungen in die Kompilierung einbezogen werden.

3. Erstellen Sie Ihre Anwendung

Nachdem Sie die Einstellungen Ihrer Anwendung konfiguriert haben, klicken Sie auf die Schaltfläche „Build Secure Application“ oder drücken Sie **F5**. XLS Padlock kompiliert Ihre Arbeitsmappe und erstellt die EXE-Datei der Anwendung.

Nach Abschluss erscheint eine Bestätigungsmeldung:



In diesem Dialogfeld können Sie die Anwendung sofort ausführen, das Kompilierungsprotokoll einsehen (besonders nützlich zur Fehlerbehebung) oder den Zielordner öffnen.

TIPP

XLS Padlock verändert Ihre ursprüngliche Excel-Arbeitsmappendatei während der Kompilierung nicht. Bewahren Sie stets eine Sicherungskopie Ihrer ursprünglichen Arbeitsmappe auf.

Behebung von Formatierungsproblemen

Wenn Ihre geschützte Arbeitsmappe seltsame Ergebnisse oder Formatierungsprobleme anzeigt, versuchen Sie, die Option „Use Excel automation for formula protection“ auf der Seite „Formulas and Passwords“ zu aktivieren.

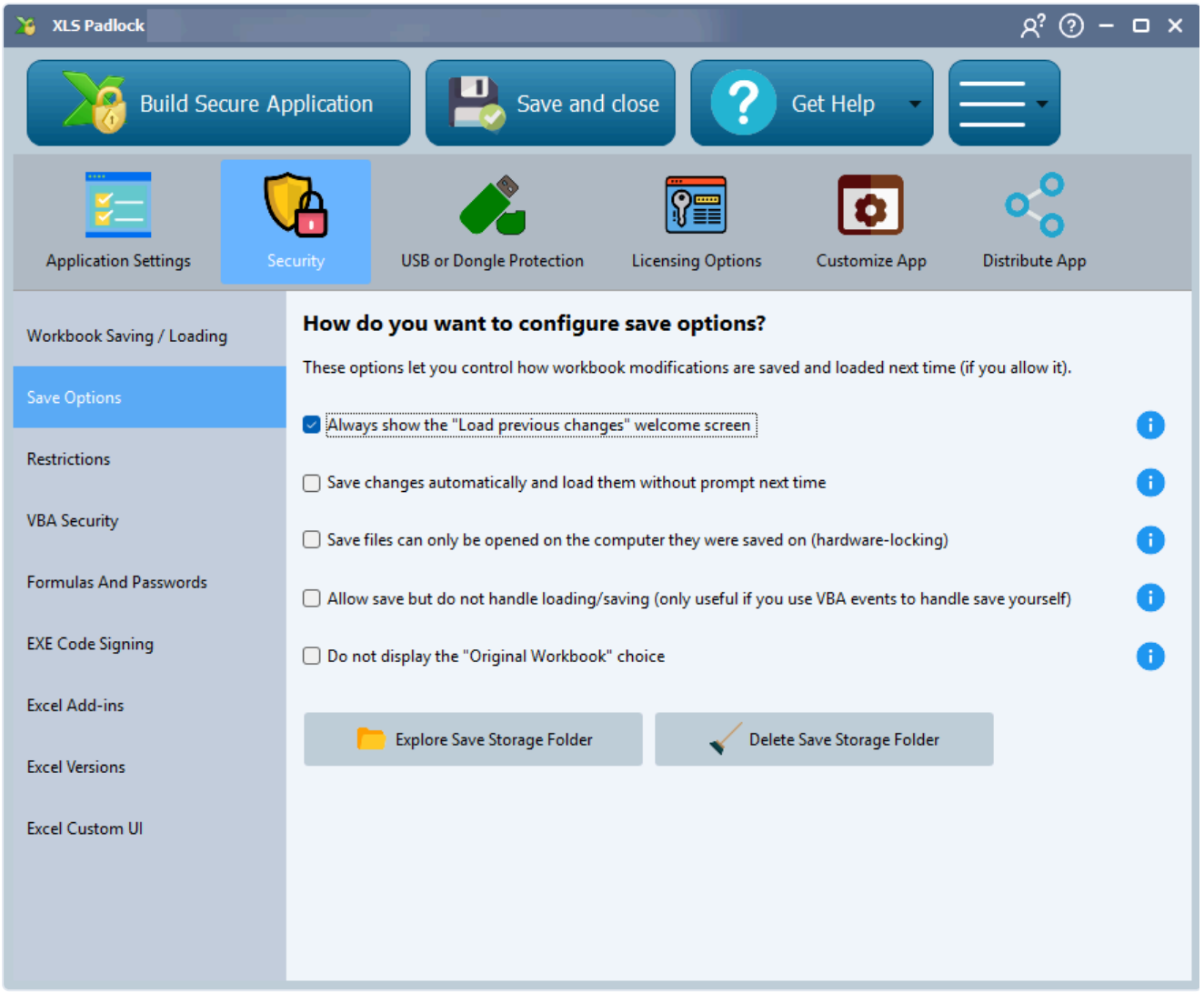
👉 Siehe auch:

So verteilen Sie eine geschützte Arbeitsmappe

Sicherheit

XLS Padlock bietet zahlreiche Möglichkeiten, Ihre Excel-Arbeitsmappe zu schützen, darunter Speicheroptionen, Funktionseinschränkungen und Bedingungen zum Öffnen Ihrer Anwendung, wie etwa das Erfordernis eines Aktivierungsschlüssels oder das Vorhandensein eines Dongles oder USB-Sticks.

Dieses Thema gibt einen Überblick über die verfügbaren Sicherheitsoptionen unter "**Security**" in XLS Padlock.



Workbook Saving / Loading

Legen Sie fest, ob Endbenutzer Änderungen an der geschützten Arbeitsmappe speichern können. Wenn das Speichern aktiviert ist, erstellt XLS Padlock sichere `.XLSC`- oder `.XLSCE`-Dateien, die nur von Ihrer Anwendung geöffnet werden können, wodurch die Daten des Benutzers geschützt werden.

➡️ Mehr über das Speichern und Laden der Arbeitsmappe erfahren

Save Options

Verfeinern Sie das Speicherverhalten mit Optionen wie dem automatischen Speichern von Änderungen beim Schließen, der Hardwarebindung von Speicherdateien an einen bestimmten Computer oder der ständigen Anzeige des Begrüßungsbildschirms, damit Benutzer zwischen einem Neuanfang und dem Laden einer früheren Arbeit wählen können.

➡️ Erweiterte Speicheroptionen entdecken

Restrictions

Schränken Sie zentrale Excel-Funktionen ein, um die Sicherheit zu erhöhen. Sie können das Kontextmenü der rechten Maustaste, die Bearbeitungsleiste, die Kopier-/Einfügebefehle, das Drucken und vieles mehr deaktivieren, um zu verhindern, dass Benutzer auf vertrauliche Teile Ihrer Arbeitsmappe zugreifen oder diese ändern.

➡️ Die Einschränkungsfunktionen ansehen

VBA Security

Schützen Sie Ihren VBA-Code vor dem Anzeigen oder Ändern. Sie können das VBA-Projekt mit einem Passwort sperren, das vor dem Benutzer verborgen bleibt, oder den Zugriff auf den VBA-Editor (VBE) vollständig blockieren, wodurch Werkzeuge zum Knacken von Passwörtern nutzlos werden. Für maximalen Schutz verwenden Sie den VBA Compiler.

➡️ Die verschiedenen VBA-Schutzmittel ansehen

Formulas and Passwords

Sichern Sie Ihre Formeln und die Struktur Ihrer Tabellenblätter. Verwenden Sie den Formelschutz von XLS Padlock, damit Benutzer mit Formeln arbeiten können, ohne sie sehen oder kopieren zu können. Sie können außerdem die Passwörter für Arbeitsmappe und Tabellenblätter direkt über die Oberfläche verwalten.

➡️ Mehr über den Formelschutz erfahren

EXE Code Signing

Signieren Sie Ihre kompilierte `.EXE`-Datei digital, um das Vertrauen Ihrer Benutzer zu gewinnen und Warnungen von Windows SmartScreen und Antivirensoftware zu vermeiden. XLS Padlock kann den Signiervorgang automatisieren, wenn Sie über ein Codesignaturzertifikat verfügen.

➡️ So signieren Sie Ihre Anwendung digital

Excel Add-ins

Standardmäßig deaktiviert XLS Padlock aus Sicherheitsgründen die meisten Add-Ins. In diesem Abschnitt können Sie gezielt bestimmte COM-Add-Ins oder gängige Excel-Add-Ins (wie das Analysis ToolPak oder den Solver) wieder aktivieren, die für die korrekte Funktion Ihrer Arbeitsmappe erforderlich sind.

➡️ So verwalten Sie Excel-Add-Ins

Excel Versions

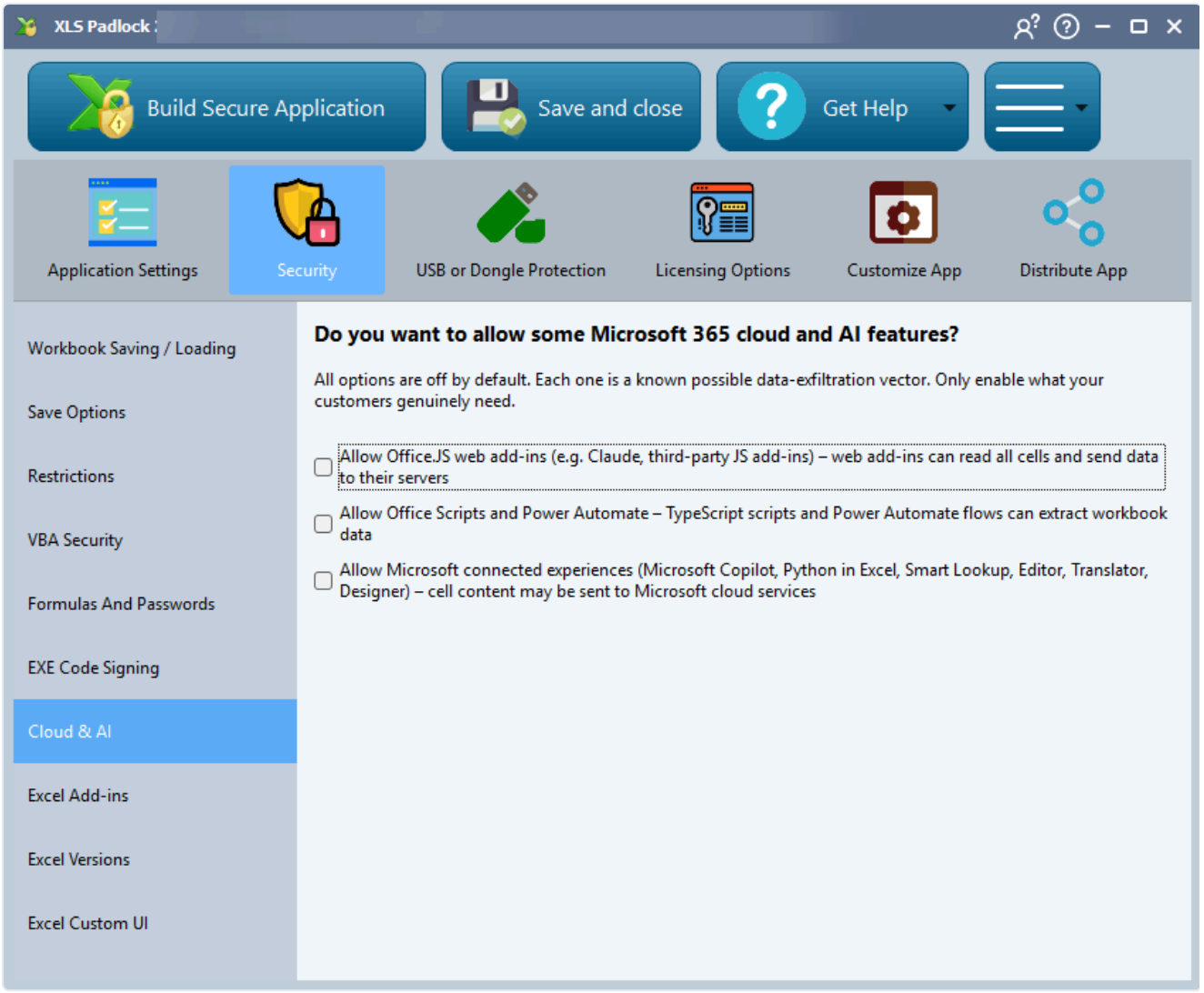
Geben Sie an, welche Versionen von Microsoft Excel mit Ihrer geschützten Anwendung kompatibel sind. Wenn ein Benutzer versucht, die EXE auf einer nicht unterstützten Version auszuführen, wird eine anpassbare Fehlermeldung angezeigt.

➡️ So fordern Sie bestimmte Excel-Versionen an

Cloud- und KI-Optionen

Modernes Excel bringt mehrere Cloud- und KI-Funktionen mit, die Ihre Arbeitsmappe auslesen und deren Inhalt an externe Server senden können. Damit Ihre geschützte Anwendung versiegelt bleibt, blockiert XLS Padlock alle diese Funktionen standardmäßig. Sie entscheiden einzeln, welche davon Sie zulassen.

Diese Einstellungen finden Sie im XLS Padlock Designer auf der Registerkarte **Security** (Sicherheit) auf der Seite **Cloud & AI** (Cloud und KI).



Warum diese Funktionen standardmäßig blockiert sind

Jede Option auf dieser Seite ist eine bekannte Möglichkeit, über die Daten der Arbeitsmappe den Rechner des Endbenutzers verlassen können. Eine geschützte Arbeitsmappe soll Ihre Formeln, Daten und Ihren VBA-Code unter Ihrer Kontrolle halten. Daher geht XLS Padlock von der sichersten Position aus: alles ist deaktiviert. Aktivieren Sie nur das, was Ihre Kunden wirklich benötigen.

Verfügbare Optionen

Alle drei Optionen sind standardmäßig deaktiviert (blockiert).

Allow Office JS web add-ins

Office JS web add-ins (auch web add-ins genannt) werden innerhalb von Excel ausgeführt und können jede Zelle der geöffneten Arbeitsmappe auslesen und diese Daten an ihre eigenen Server senden. Dazu gehören Add-ins von Drittanbietern wie das Plugin **Claude for Excel** sowie alle anderen JS-Add-ins, die aus dem Office store installiert oder per Sideloading geladen wurden.

Wenn diese Option deaktiviert ist, können sich web add-ins nicht in Ihre geschützte Arbeitsmappe laden. Aktivieren Sie sie nur, wenn Ihre Anwendung von einem bestimmten web add-in abhängt, dem Sie vertrauen.

Allow Office Scripts and Power Automate

Office Scripts (TypeScript-Automatisierungsskripte) und Power Automate-Flows können den Inhalt der Arbeitsmappe auslesen und aus der Anwendung herausführen. Lassen Sie diese Option deaktiviert, es sei denn, Ihr Workflow ist darauf angewiesen.

Allow Microsoft connected experiences

Microsoft connected experiences (verbundene Microsoft-Erlebnisse) senden Zellinhalte an die Microsoft-Clouddienste. Dies betrifft Microsoft Copilot, Python in Excel, Smart Lookup (Intelligente Suche), Editor, Translator (Übersetzer) und Designer. Aktivieren Sie diese Option nur, wenn Ihre Kunden diese Funktionen benötigen und Sie akzeptieren, dass die betreffenden Zellinhalte möglicherweise in der Microsoft-Cloud verarbeitet werden.

Hinweise

- Diese Optionen gelten für die kompilierte Anwendung, nicht für den XLS Padlock Designer.
- Das Blockieren dieser Funktionen blendet auch deren Einstiegspunkte im Excel-Menüband aus (zum Beispiel das Add-ins-Flyout, die Registerkarte Automate und die Schaltfläche Copilot), sodass Endbenutzer nicht zur Nutzung einer deaktivierten Funktion aufgefordert werden.
- COM-Add-ins und die in Excel integrierten Add-ins werden separat verwaltet. Siehe Excel-Add-ins verwalten.

Schutz verbessern

Warum das Kompilieren zu einer EXE nicht ausreicht

Wenn Sie eine Arbeitsmappe in eine geschützte Anwendung kompilieren, wird die ursprüngliche Arbeitsmappendatei (XLSX, XLSM usw.) verschlüsselt und in die resultierende .EXE-Datei eingebettet. Dies verhindert, dass Benutzer direkt auf die Originaldatei zugreifen.

> **Grundlegender Schutz reicht nicht aus**
>
> Das bloße Kompilieren Ihrer Arbeitsmappe bietet jedoch nur eine grundlegende Sicherheitsebene. Stellen Sie es sich vor wie das Schließen einer Tür, ohne sie abzuschließen. Um Ihr geistiges Eigentum wirklich zu sichern, müssen Sie die erweiterten Schutzfunktionen aktivieren, die XLS Padlock bietet.

Damit XLS Padlock funktioniert, muss es Ihre Arbeitsmappe in den Speicher von Excel laden. **Ohne die zusätzlichen Sicherheitsebenen von XLS Padlock zu aktivieren**, könnte ein versierter Angreifer die ursprüngliche Arbeitsmappe möglicherweise aus dem Speicher extrahieren, während die Anwendung läuft. Auch wenn dies keine triviale Aufgabe ist, verdeutlicht es die Notwendigkeit eines stärkeren Schutzes.

> **Eine extrahierte Arbeitsmappe unbrauchbar machen**
>
> Wenn Sie die Sicherheitsfunktionen von XLS Padlock korrekt anwenden, wird die extrahierte Arbeitsmappe unbrauchbar. Selbst wenn jemand die Datei extrahiert, funktioniert sie nicht korrekt. Der Schutz ist besonders stark für Arbeitsmappen, die Formeln und/oder VBA-Code enthalten, da diese kritischen Komponenten entweder verschlüsselt oder kompiliert werden, was eine unbefugte Nutzung oder Manipulation wirksam verhindert.

Die wirksamsten Sicherheitsfunktionen

Wir empfehlen, die folgenden Funktionen für den stärksten Schutz zu aktivieren:

- Schützen Sie Ihre Formeln mit dem Formelschutz
- Sichern Sie Ihr VBA mit dem VBA Compiler
- Verhindern Sie gängige VBA- und OLE-Hacks
- Verbieten Sie den Zugriff auf den VBA-Editor (VBE)
- Schützen Sie Ihre Arbeitsmappe mit einem Passwort
- Deaktivieren Sie unnötige Excel-Add-Ins

Zugriffssteuerung der Arbeitsmappe

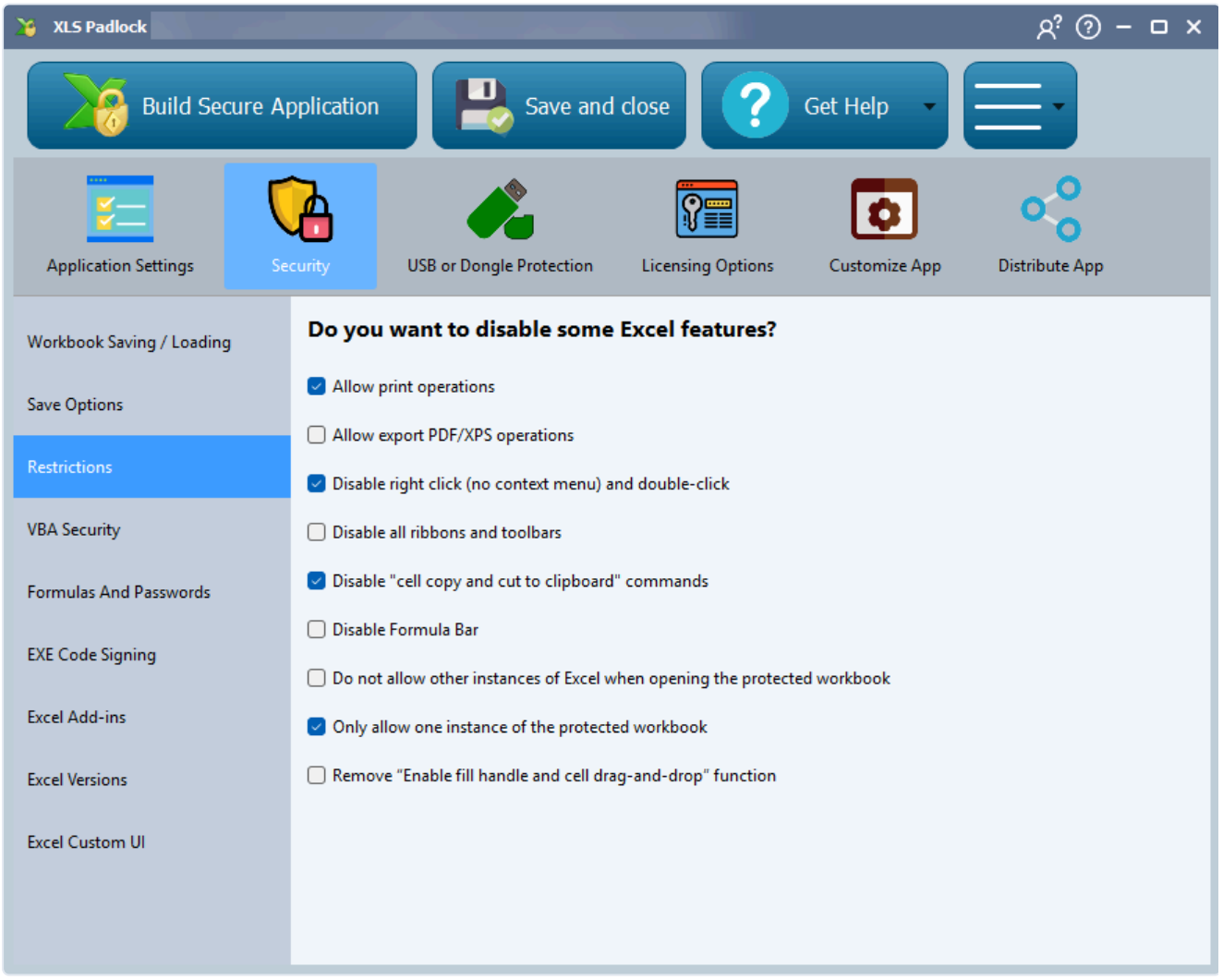
XLS Padlock bietet zahlreiche Sicherheitsfunktionen, um **den Zugriff auf Ihre Excel-Arbeitsmappen auf autorisierte Benutzer zu beschränken**. Sie können Aktivierungsschlüssel generieren, zeitlich begrenzte Testversionen einrichten und vieles mehr. Ein vorrangiges Ziel von XLS Padlock besteht darin, Ihnen den **Verkauf von Lizenzen für Ihre Arbeitsmappen** zu ermöglichen und Funktionen bereitzustellen, die Piraterie und unbefugte Weitergabe reduzieren.

Dieses Thema gibt einen Überblick über die Sicherheitsfunktionen, die der Zugriffskontrolle für Arbeitsmappen gewidmet sind:

- **Steuern Sie, wer Ihre Arbeitsmappe verwenden darf:** Aktivierungsschlüssel einrichten.
- **Verhindern Sie die Weitergabe, indem Sie eine Lizenz an einen einzigen Computer binden:** Verwenden Sie hardwaregebundene Aktivierungsschlüssel.
- **Aktivierungen aus der Ferne verwalten und Zugriff deaktivieren:** Verwenden Sie Online-Aktivierung, Deaktivierung und Validierung.
- **Erstellen Sie Testversionen Ihrer Arbeitsmappe:** Erfahren Sie, wie Sie Test-Arbeitsmappen erstellen.
- **Benutzeraktionen einschränken:** Drucken, Exportieren, Rechtsklick und mehr verbieten.
- **Ein physisches Gerät für den Zugriff erforderlich machen:** Binden Sie die Anwendung an einen Dongle oder USB-Stick.
- **Lizenzen und Abonnements online verkaufen:** Integrieren Sie E-Commerce-Plattformen wie WooCommerce oder FastSpring.

Einschränkungen der Excel-Arbeitsmappe

Dieses Thema behandelt die Optionen im Bereich 'Security > Restrictions' von XLS Padlock, mit denen Sie bestimmte Benutzeraktionen in Ihrer geschützten Arbeitsmappe einschränken können.



Allow Print Operations

Um zu verhindern, dass Benutzer Ihre Arbeitsmappe drucken, deaktivieren Sie die Option **Allow print operations** (Druckvorgänge zulassen) in den Security-Einstellungen.

Wenn diese Option deaktiviert ist, wird jeder Druckversuch blockiert, und der Benutzer erhält die Fehlermeldung "Printing is not allowed".

Allow Export to PDF/XPS

Um zu verhindern, dass Benutzer Arbeitsmappeninhalte in PDF- oder XPS-Dateien exportieren, deaktivieren Sie die Option "Allow export PDF/XPS operations" in den Security-Einstellungen.

Disable Right-Click (and optionally Double-Click)

Um das Kontextmenü beim Rechtsklick für Zellen in Ihrer Arbeitsmappe zu deaktivieren, aktivieren Sie die Option "Disable right-click" in den Security-Einstellungen.

Dies wird häufig verwendet, um zu verhindern, dass Benutzer Zelldaten und -formatierungen kopieren, einfügen oder ändern.

Disable All Ribbons and Toolbars

Die Option **Disable all ribbons and toolbars** (Alle Menübänder und Symbolleisten deaktivieren) blendet alle integrierten Menüband-Registerkarten und die meisten Befehle des Microsoft Office-Menüs aus und schafft so eine vereinfachte, kioskähnliche Oberfläche für Ihre Anwendung.

Diese Funktion arbeitet mit **Excel 2007 und höher**. Sie können dies erzwingen, indem Sie die mindestens erforderliche Excel-Version für Ihre Arbeitsmappe festlegen.

Disable "Cell Copy and Cut to Clipboard" Commands

Aktivieren Sie diese Option, um die Befehle **Copy** und **Cut** für Zellen zu deaktivieren. Dadurch wird verhindert, dass Endbenutzer Daten aus Ihrer Arbeitsmappe ohne Weiteres in die Zwischenablage kopieren, um sie in anderen Anwendungen weiterzuverwenden.

Disable Formula Bar

Die Option **Disable Formula Bar** (Bearbeitungsleiste deaktivieren) blendet die Bearbeitungsleiste von Excel aus. Zusätzlich wird die entsprechende Option im Ansichtsmenü von Excel deaktiviert, sodass Endbenutzer sie nicht wieder aktivieren können.

Für mehr Sicherheit empfehlen wir außerdem, den Zugriff auf den VBA-Editor zu deaktivieren.

Do Not Allow Other Instances of Excel

Wenn diese Option aktiviert ist, prüft sie, ob beim Start Ihrer kompilierten Arbeitsmappe durch einen Benutzer bereits eine andere Instanz von Excel ausgeführt wird. Wird ein bestehender Excel-Prozess gefunden, fordert eine Meldung den Benutzer auf, diesen zu schließen, bevor er fortfahren kann.

Dadurch wird sichergestellt, dass die Excel-Instanz, die Ihre geschützte Arbeitsmappe ausführt, die erste und einzige ist, was eine besser kontrollierte Umgebung bietet.

Diese Option finden Sie in den Security-Einstellungen.

Only Allow One Instance of the Protected Workbook

Die Option "Only allow one instance of the protected workbook" in den Security-Einstellungen verhindert, dass Endbenutzer mehr als eine Instanz Ihrer geschützten Anwendung gleichzeitig ausführen. Wenn ein Benutzer versucht, die Anwendung erneut zu öffnen, während sie bereits läuft, wird das vorhandene Fenster in den Vordergrund geholt. Diese Funktion verhindert wirksam, dass Benutzer mehrere Instanzen derselben geschützten Arbeitsmappe öffnen.

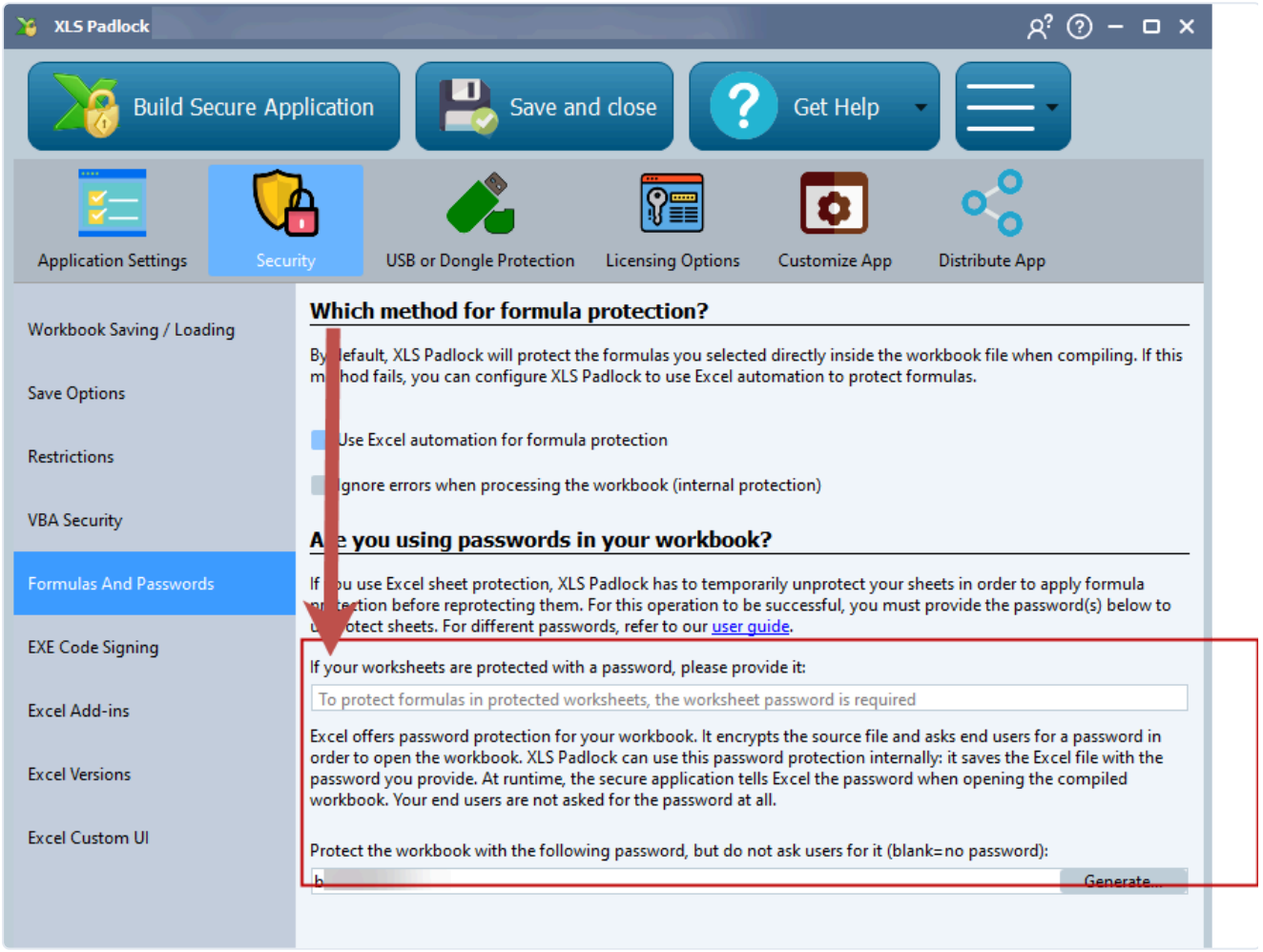
> **Kompatibilitätshinweis**
>
> Auf Excel-Versionen vor 2013 wird anstelle des Fokussierens des vorhandenen Fensters ein Meldungsfeld mit dem Hinweis "An instance of this application is already running" angezeigt.

Remove “Enable fill handle and cell drag-and-drop” function

Die Option "Remove “Enable fill handle and cell drag-and-drop” function" verhindert, dass Endbenutzer das Ausfüllkästchen und das Ziehen und Ablegen von Zellen in der geschützten Arbeitsmappe verwenden. Sie entspricht der Einstellung in den allgemeinen Optionen von Excel (Review”, “Check accessibility”, “Options accessibility”, “Advanced”, “Enable fill handle and cell drag-and-drop”), erlaubt es Ihnen jedoch, die Auswahl des lokalen Benutzers zu überschreiben.

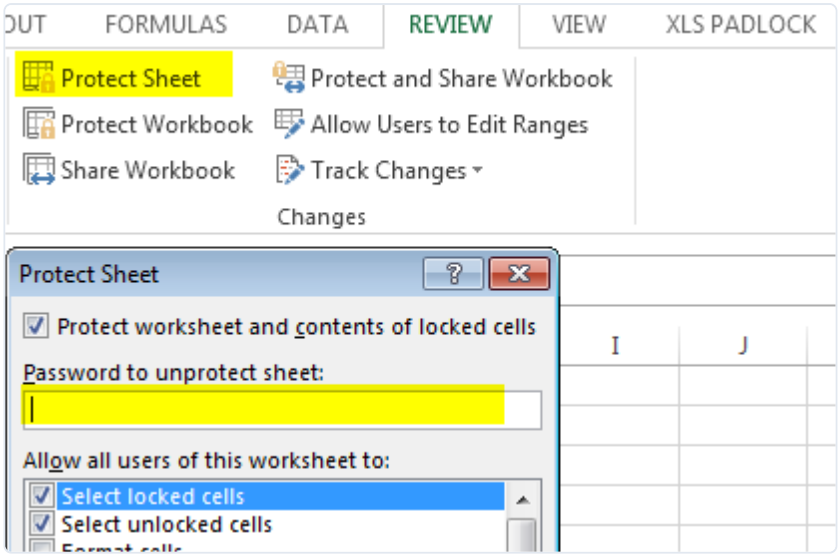
Kennwortschutz der Arbeitsblätter

Mit XLS Padlock können Sie sowohl Kennwörter für Arbeitsmappen als auch für Tabellenblätter verwalten, um die Sicherheit und Funktionalität Ihrer geschützten Anwendung zu verbessern. Diese Anleitung erläutert, wie Sie diese Kennworteinstellungen konfigurieren.



Kennwortschutz für Tabellenblätter

Wenn Sie den in Excel integrierten Blattschutz verwenden, um den Zugriff auf Zellen einzuschränken, muss XLS Padlock Ihr Kennwort kennen, um seine eigenen Sicherheitsebenen verwalten zu können.



Wenn der Formelschutz von XLS Padlock aktiv ist, muss er Ihre Tabellenblätter vorübergehend entsperren, seinen Schutz anwenden und sie anschließend erneut schützen. Damit dieser Vorgang erfolgreich ausgeführt werden kann, müssen Sie das in Excel für den Blattschutz verwendete Kennwort angeben.

> **HINWEIS**
>
> XLS Padlock wendet automatisch dieselben Blattschutzoptionen erneut an, die Sie in der ursprünglichen Arbeitsmappe definiert haben.

Wenn Sie ein einziges Kennwort für alle geschützten Tabellenblätter verwenden, geben Sie es in das Feld "If your worksheets are protected with a password, please provide it" (Wenn Ihre Tabellenblätter mit einem Kennwort geschützt sind, geben Sie es bitte an) ein.

Wenn Sie unterschiedliche Kennwörter für unterschiedliche Tabellenblätter verwenden, müssen Sie diese im folgenden JSON-Format angeben:

```
{"worksheet 1 name": "Password1", "worksheet 2 name": "Password2", ...}
```

Zum Beispiel:

```
{"Sheet1": "Hello World", "Sheet2": "Password2"}
```

Stiller Kennwortschutz der Arbeitsmappe

Excel bietet eine Funktion, mit der die gesamte Arbeitsmappe mit einem Kennwort verschlüsselt werden kann. XLS Padlock kann dieses Kennwort für Sie verwalten und bietet so eine zusätzliche Sicherheitsebene, ohne Ihre Endbenutzer dazu aufzufordern.

Wenn Sie im Feld "Protect the workbook with the following password" (Arbeitsmappe mit folgendem Kennwort schützen) ein Kennwort festlegen, speichert XLS Padlock die Arbeitsmappe mit diesem Kennwort. Zur Laufzeit gibt die sichere Anwendung das Kennwort beim Öffnen der Arbeitsmappe automatisch an Excel weiter. **Ihre Endbenutzer werden niemals nach dem Kennwort gefragt.**

Diese Kombination erhöht die Sicherheit Ihrer Anwendung, indem der Schutz von XLS Padlock mit der nativen Verschlüsselung von Excel kombiniert wird.

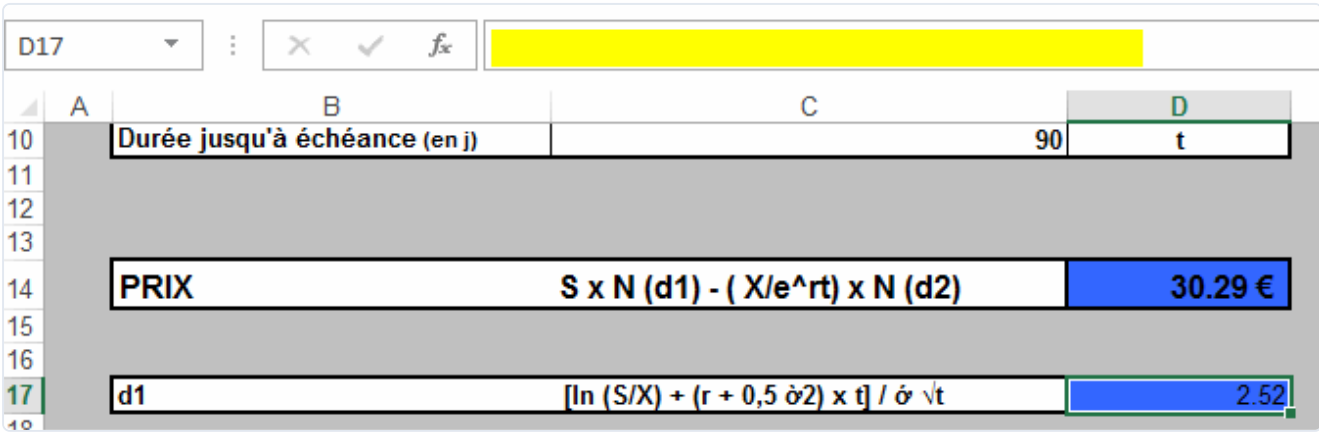
Mit der Schaltfläche **Generate** (Generieren) können Sie ein starkes, zufälliges Kennwort erstellen.

> **Ändern Sie das Kennwort nicht nach der Verteilung**
>
> Sobald Sie eine mit einem bestimmten Arbeitsmappenkennwort geschützte Anwendung verteilt haben, ändern Sie es nicht mehr. Andernfalls können Endbenutzer ihre zuvor gespeicherten Daten nicht mehr öffnen, da die Anwendung diese nicht entschlüsseln kann.

Wenn Sie das Kennwortfeld leer lassen, wird diese Funktion deaktiviert.

Excel- und XLS Padlock-Schutz kombinieren

Der Formelschutz von XLS Padlock ist vollständig mit den integrierten Blattschutzfunktionen von Excel kompatibel, etwa gesperrten oder ausgeblendeten Zellen. Wenn Sie beide zusammen verwenden, zeigen geschützte Zellen nichts in der Bearbeitungsleiste an, und die zugrunde liegenden `PLEvalForm`-Funktionsaufrufe bleiben verborgen.



Hinweis zu Abhängigkeiten

Geschützte Formeln, die von anderen Zellen abhängen, schlagen fehl, wenn diese Zellen über das Zellenformat-Attribut "Hidden" (Ausgeblendet) von Excel ausgeblendet werden. Vermeiden Sie das Ausgeblendet-Attribut bei Zellen, die Abhängigkeiten Ihrer geschützten Formeln sind.

Siehe auch:

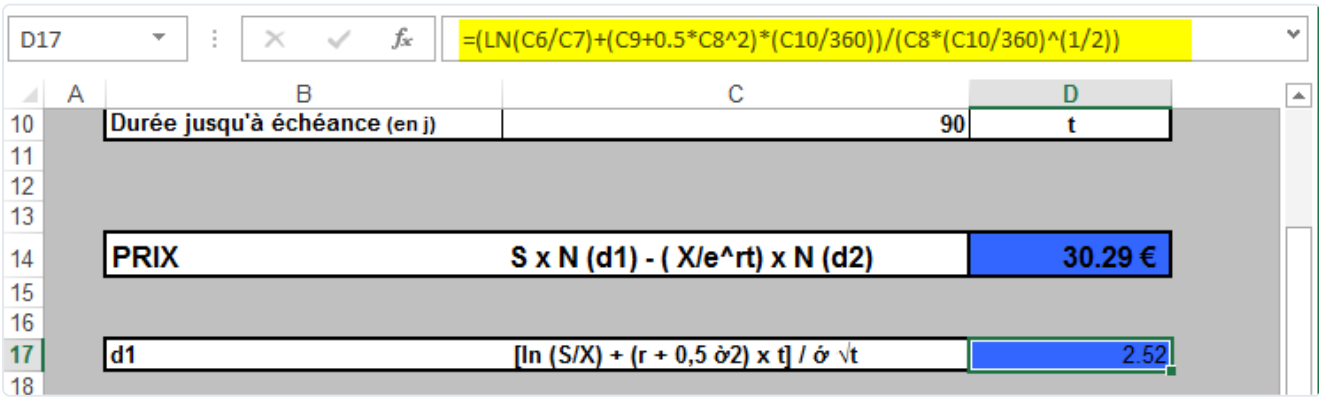
- Formelschutz deaktivieren

Formelschutz: Excel oder XLS Padlock

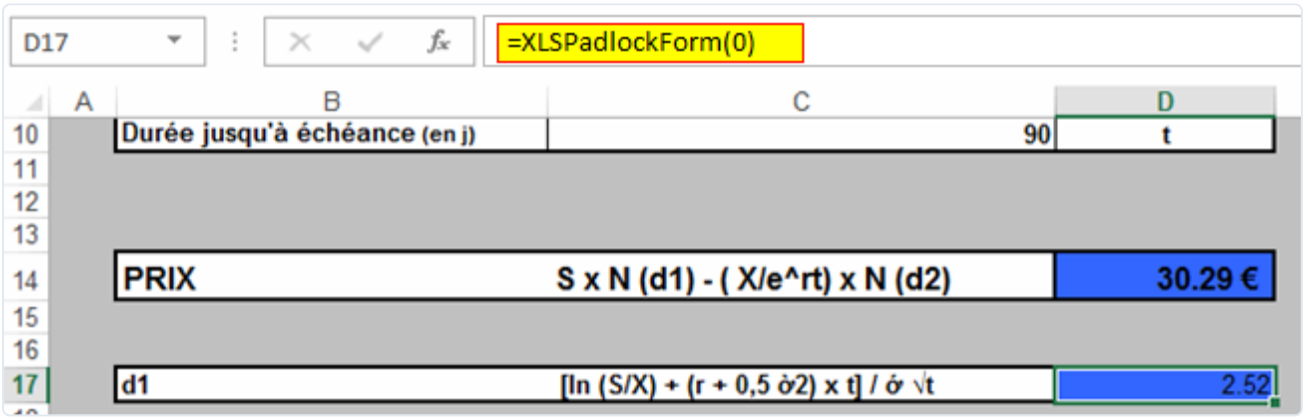
Excel erlaubt zwar das Ausblenden von Formeln auf einem geschützten Blatt, doch dieser Schutz ist begrenzt. XLS Padlock geht deutlich weiter, indem es Ihre Formeln durch eine sichere Funktion ersetzt, die nur verfügbar ist, während Ihre Anwendung ausgeführt wird.

In der kompilierten Arbeitsmappe werden die Formeln in der Bearbeitungsleiste durch anonyme Funktionen wie `=PLEvalFormD(0;0)` oder `=PLEvalFormD(1;COUNT($F7))` ersetzt. Die Berechnungen funktionieren jedoch weiterhin wie erwartet.

Vor dem Schutz kann jeder Ihre Formel sehen:

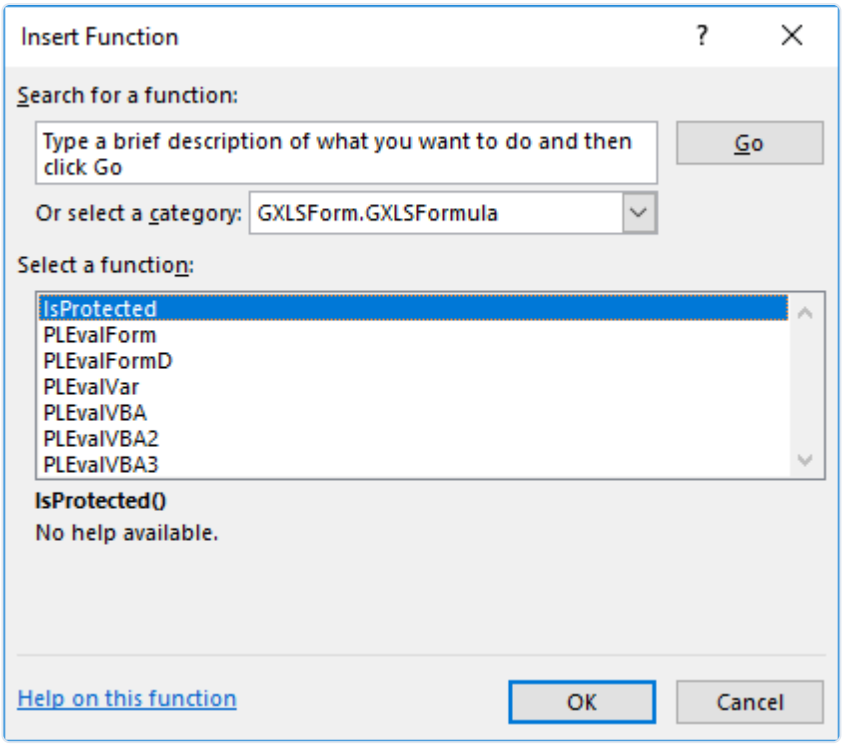


Nach dem Schutz mit XLS Padlock wird die Formel automatisch ersetzt und bleibt dabei funktionsfähig:



Dies erhöht die Sicherheit Ihrer Arbeitsmappe erheblich. Da die ursprünglichen Formeln aus der Arbeitsmappendatei entfernt werden, funktionieren die geschützten Zellen nur dann ordnungsgemäß, wenn die Arbeitsmappe innerhalb der von XLS Padlock erstellten sicheren Anwendung geöffnet wird.

Bereits zur Entwurfszeit können Sie feststellen, dass die Sicherheitsfunktionen von XLS Padlock wie `PLEvalForm` und `PLEvalFormD` von Excel erkannt werden:

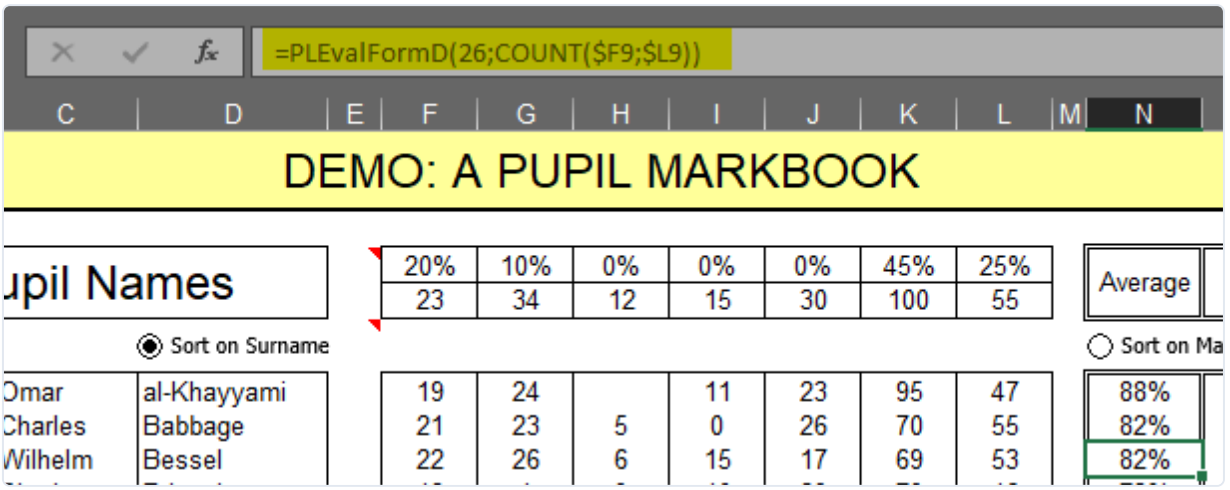


👉 Erfahren Sie mehr über das Schützen von Zellen mit XLS Padlock.

Formeln mit XLS Padlock schützen

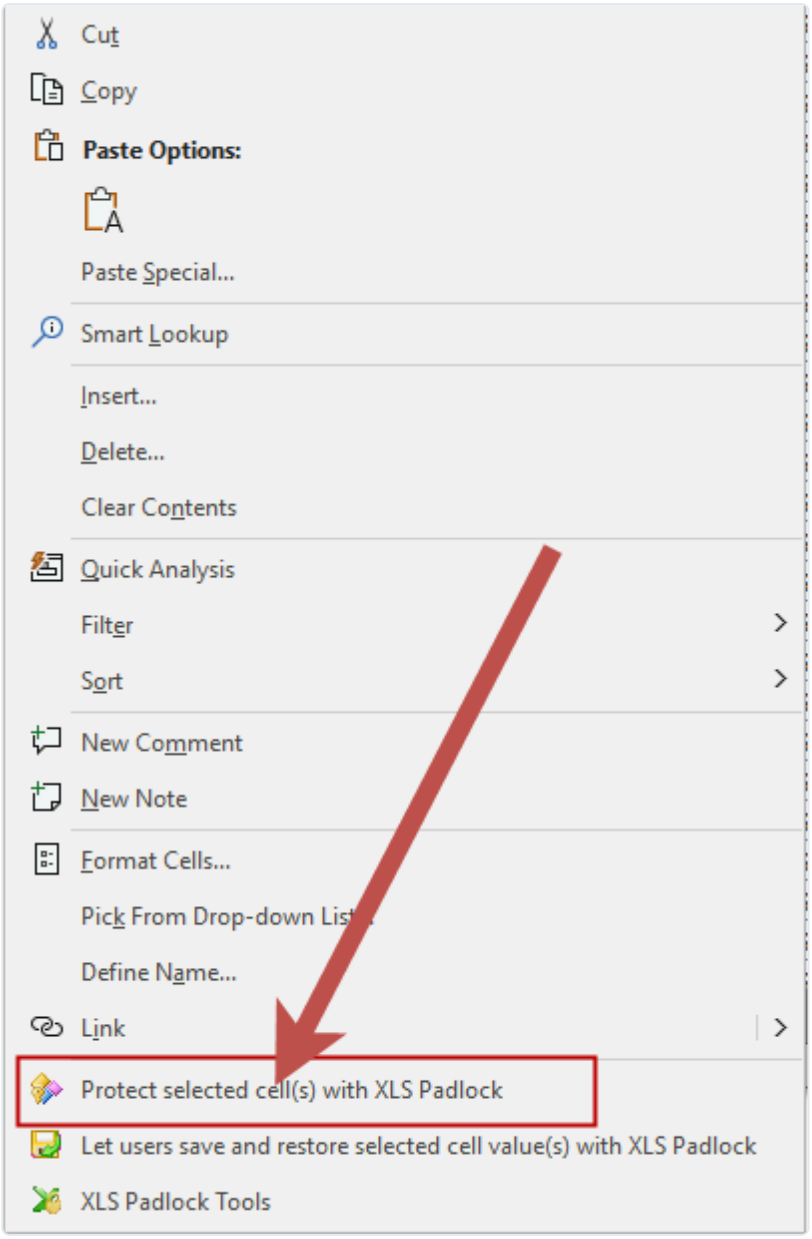
Zusätzlich zum Zellschutz von Excel **empfehlen wir Ihnen dringend, wichtige Formeln mit dem eigenen Formelschutz von XLS Padlock abzusichern**.

Nach dem Schutz mit XLS Padlock erscheinen Ihre Formeln nicht mehr in der Formelleiste, **bleiben aber voll funktionsfähig**. Stattdessen wird ein kryptischer Funktionsaufruf wie `PLEvalFormD` angezeigt:



👉 Ziel ist es, die Arbeitsmappe unbrauchbar zu machen, falls es jemandem gelingt, die Arbeitsmappendatei aus der kompilierten EXE zu extrahieren. Da sich die **von XLS Padlock geschützten Formeln nicht mehr in der Arbeitsmappendatei befinden**, funktioniert die extrahierte Arbeitsmappe nicht korrekt.

Mit XLS Padlock entscheiden Sie, welche Zellen geschützt werden, und Sie können **mehrere Zellen gleichzeitig auswählen und schützen**.



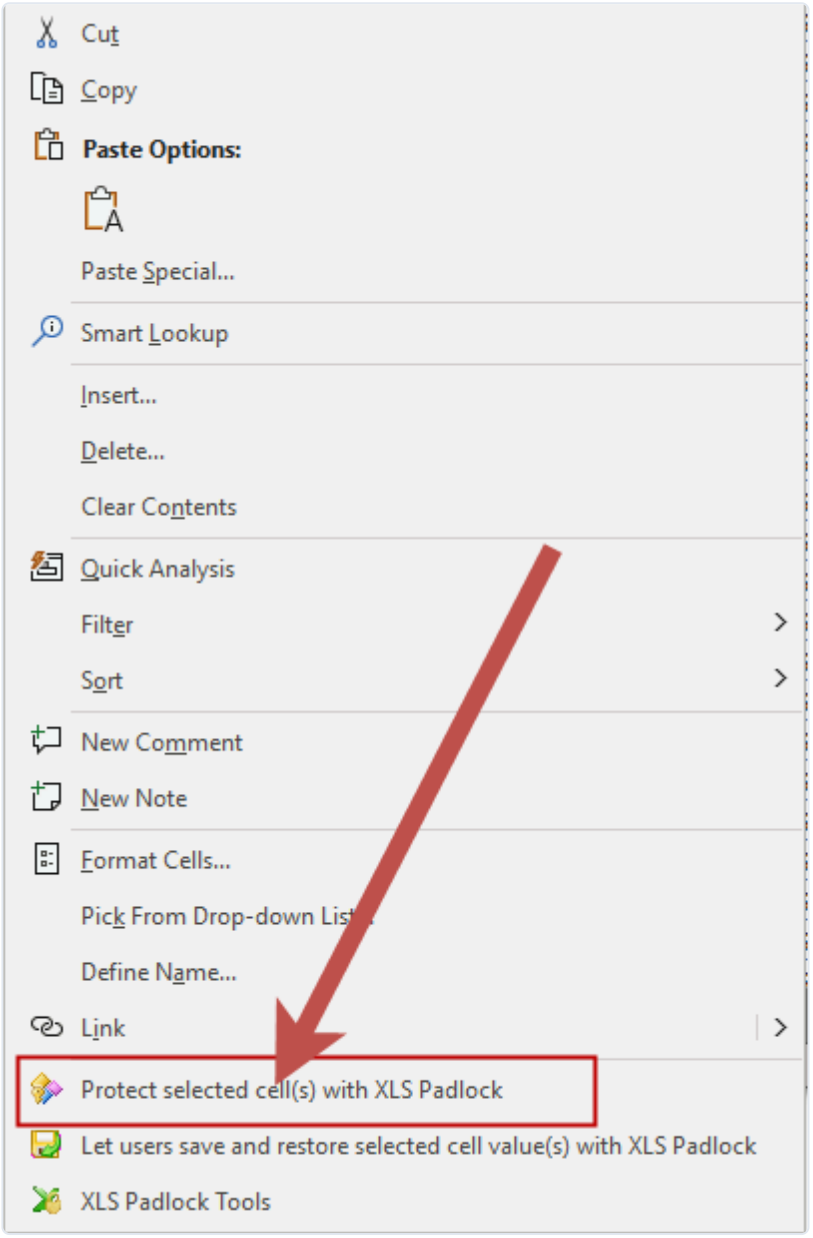
👉 Siehe wie Sie die zu schützenden Zellen auswählen

Siehe auch:

- Mehr darüber erfahren, wie Sie Ihre Formeln effizient schützen
- Gängige Excel-Add-Ins deaktivieren
- Den VBA Compiler für noch stärkeren Schutz verwenden

Zellen für den Formelschutz markieren

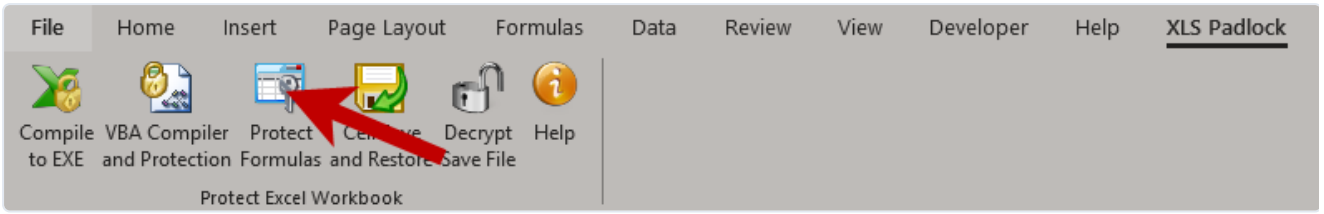
Um eine Formel zu schützen, klicken Sie einfach mit der **rechten Maustaste auf eine oder mehrere Zellen** mit den Formeln, die Sie ausblenden möchten, und wählen Sie „**Protect selected cell(s) with XLS Padlock**“ aus dem Kontextmenü.



XLS Padlock bestätigt daraufhin, dass die Zellen für den Schutz markiert sind.

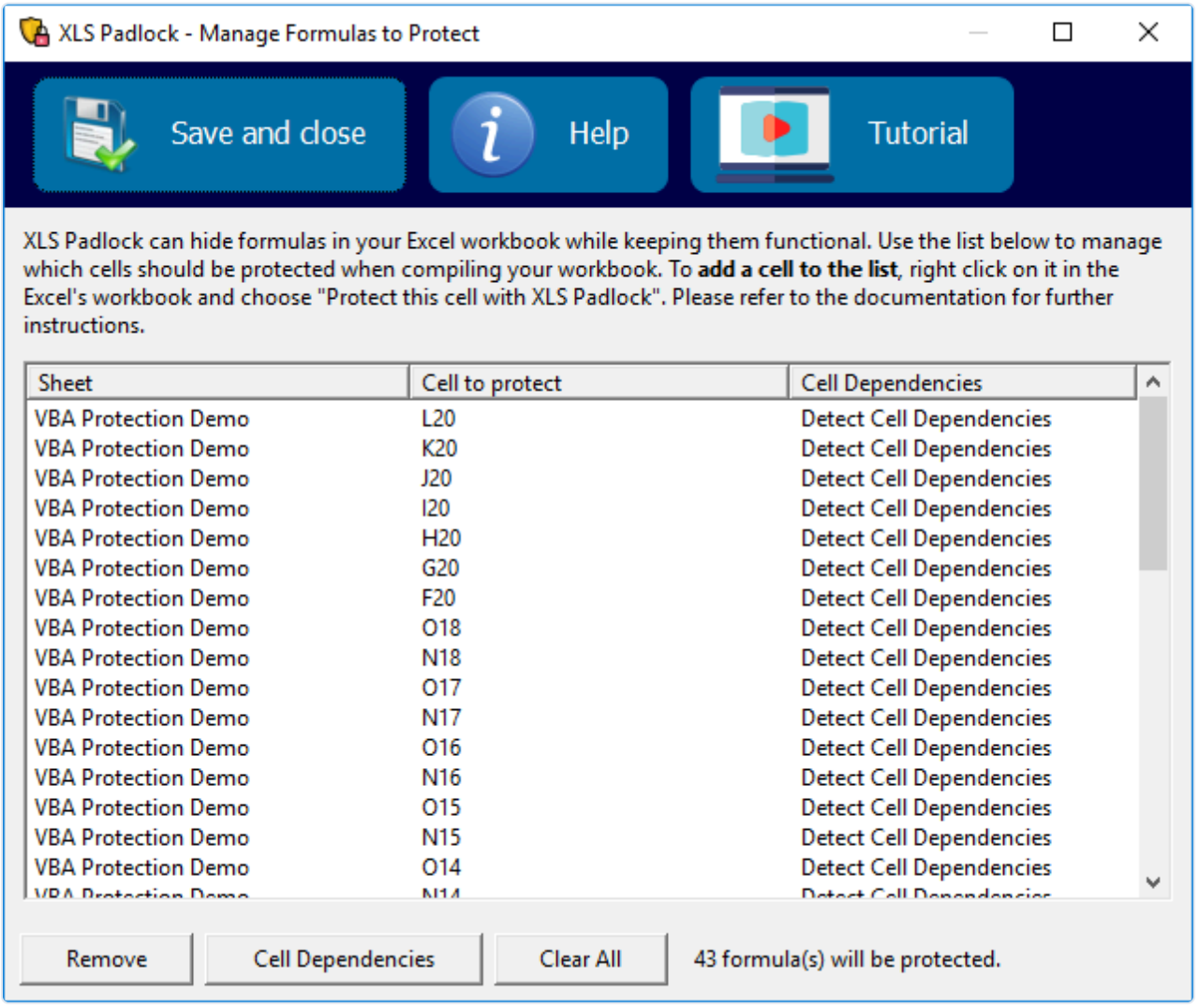
Übersicht der geschützten Zellen

Um eine Übersicht aller geschützten Zellen anzuzeigen, klicken Sie im XLS Padlock-Tab oder -Menü auf „**Protect Formulas**“:



Dadurch wird eine Liste aller für den Schutz konfigurierten Zellen geöffnet. In diesem Fenster können Sie das Schutzverhalten anpassen, den Schutz für bestimmte Zellen entfernen oder die gesamte Liste leeren.

Wenn Sie Ihre Arbeitsmappe kompilieren, ersetzt XLS Padlock alle aufgeführten Formeln durch generische Funktionsaufrufe wie `PLEvalForm(N)` und `PLEvalFormD(N, ….)`. **Ihre Zellen bleiben funktionsfähig, aber Endbenutzer können die zugrunde liegenden Formeln nicht ermitteln.** Die ursprünglichen Formeln sind in der kompilierten Arbeitsmappe nicht mehr vorhanden; sie werden von der EXE selbst verwaltet.



Zellabhängigkeiten

Über die Schaltfläche **Cell Dependencies** (Zellabhängigkeiten) steuern Sie, wie der Schutz angewendet wird. Es stehen zwei Optionen zur Verfügung: „Detect Cell Dependencies“ und „No“.

Standardmäßig erkennt XLS Padlock alle Zellbezüge und Bereichsnamen in Ihren Formeln (Zellabhängigkeiten) und erzeugt eine anonyme Funktion, die diese Bezüge enthält. Dadurch kann Excel die geschützten Zellen korrekt neu berechnen. Lautet die zu schützende Formel beispielsweise `=A3^2`, erzeugt XLS Padlock eine Funktion wie: `PLEvalFormD(1, COUNT(A3))`.

Wenn XLS Padlock eine Zelle nicht schützen kann, können Sie für deren Einstellung „Cell Dependencies“ die Option „**No**“ wählen. In diesem Fall wird eine einfache generische Funktion `PLEvalForm(N)` verwendet.

Einschränkungen des Schutzes

Einige komplexe Formeln werden vom Schutz nicht unterstützt und schlagen fehl, wobei in der Zelle ein `#ERROR!` oder `#VALUE!` angezeigt wird. Bitte testen Sie Ihre Formeln gründlich, bevor Sie Ihre geschützte Arbeitsmappe verteilen. Achten Sie besonders auf die folgenden Einschränkungen:

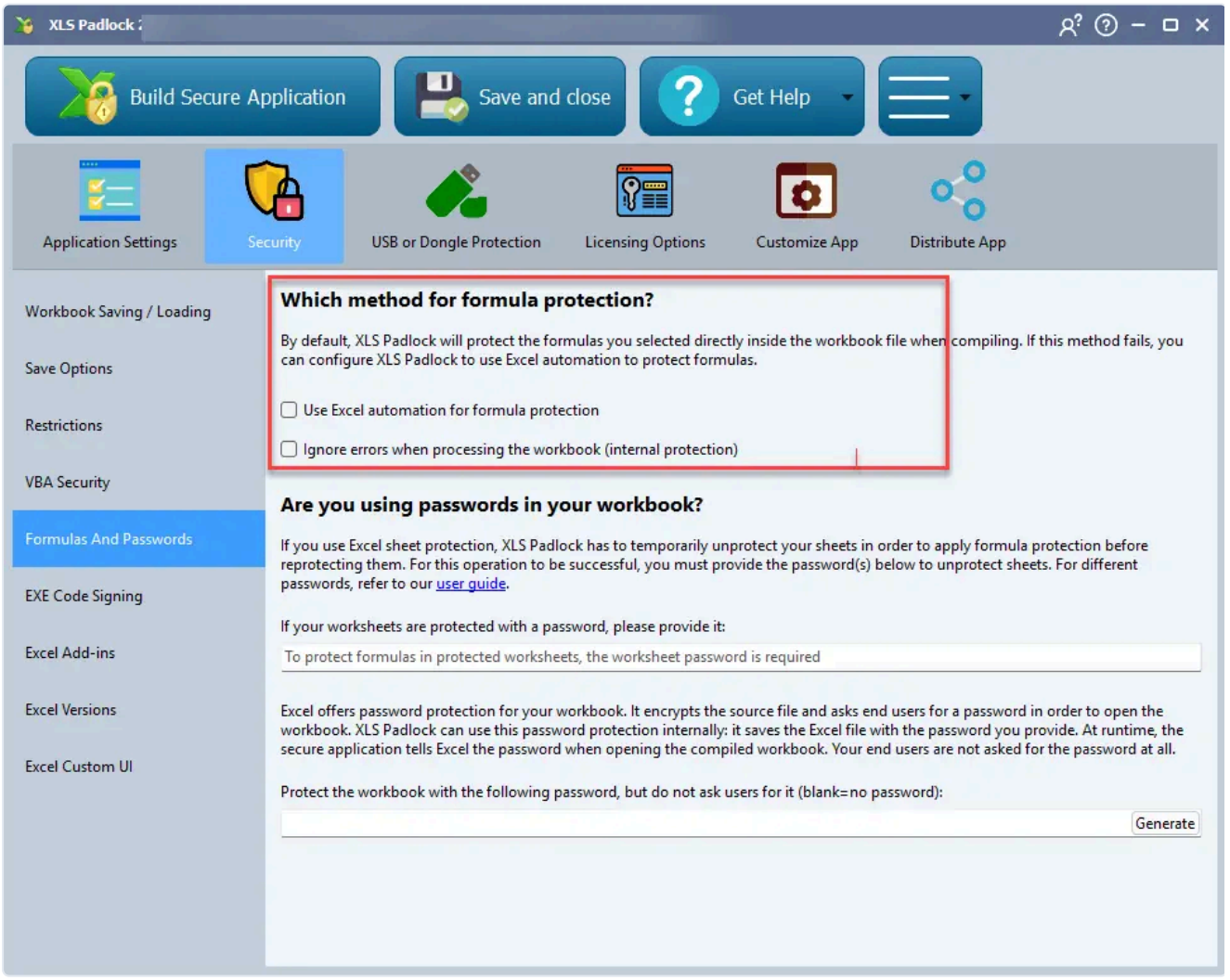
- Die Länge der Formel muss weniger als 256 Zeichen betragen.
- Die Funktion `INDIRECT` wird nicht unterstützt.
- Formeln dürfen keine benutzerdefinierten VBA-Funktionen enthalten, sondern nur reguläre Excel-Funktionen.
- Der Formelschutz sollte sparsam eingesetzt werden. Schützen Sie nur die wichtigsten Formeln, von denen Ihre Arbeitsmappe abhängt. Das Schützen mehrerer tausend Formeln wird nicht empfohlen, da dies die Größe Ihrer EXE-Datei erhöht und Excel auf älteren Computern verlangsamen kann.
- **Die Datengültigkeitsprüfung von Zellen und Regeln für die bedingte Formatierung werden nicht unterstützt**.

👉 Siehe auch

- Excel-Blattschutz und XLS Padlock-Schutz kombinieren

Methode des Formelschutzes

Wenn Sie den Formelschutz aktivieren, kann XLS Padlock eine von zwei Methoden verwenden:



1. **Direct Modification (Recommended)** (direkte Änderung, empfohlen): Diese Methode ändert die Formeln direkt in der Quelldatei der Arbeitsmappe. Sie ist die Standardmethode und funktioniert in den meisten Fällen zuverlässig.
2. **Excel Automation** (Excel-Automatisierung): Diese Methode verwendet Automatisierung, um Excel zu steuern und den Schutz anzuwenden.

Falls die empfohlene Methode fehlschlägt, können Sie XLS Padlock zur Verwendung der Automatisierungsmethode zwingen, indem Sie die Option "**Use Excel automation for formula protection**" aktivieren. Diese Methode wird auch automatisch für binäre Arbeitsmappen (XLSB) und sehr große Dateien verwendet.

Als letztes Mittel können Sie "Ignore errors when processing the workbook" aktivieren, achten Sie aber darauf, Ihre kompilierte Anwendung gründlich zu testen, wenn Sie diese Option aktivieren.

Formelschutz deaktivieren

Da der Formelschutz von XLS Padlock mit Microsoft Excel 2000 nicht kompatibel ist, müssen Sie die Option **Disable formula and VBA protection** auf der Seite Security aktivieren, wenn Sie Anwendungen erstellen müssen, die mit dieser Version kompatibel sind.

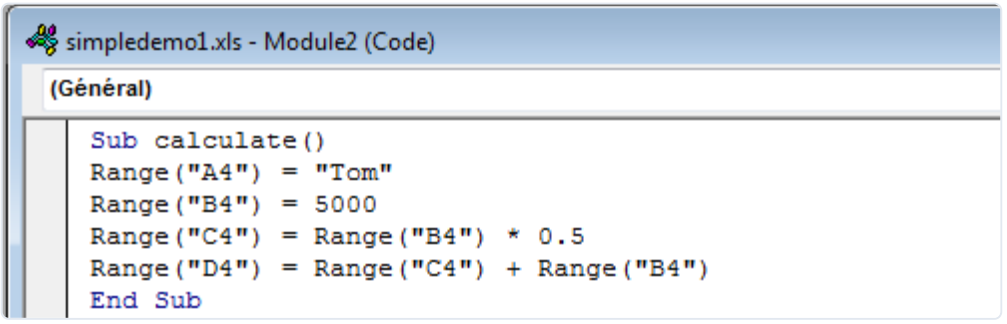
> **Der VBA Compiler wird ebenfalls deaktiviert**
>
> Das Aktivieren dieser Option deaktiviert auch den VBA-Code-Compiler.

Über den integrierten VBA Compiler

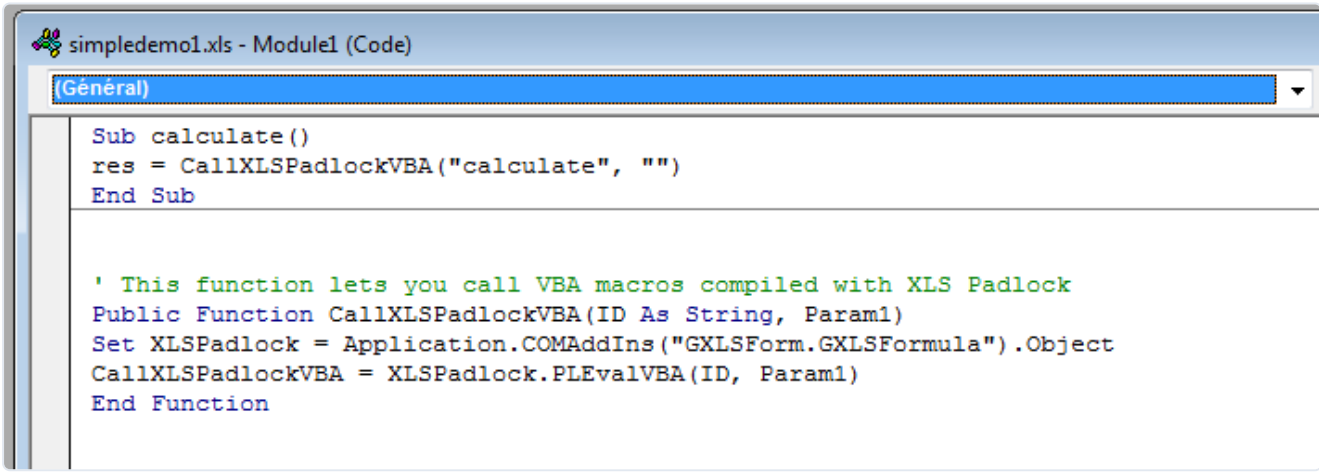
XLS Padlock verfügt über einen **integrierten VBA Compiler**, der Ihre Basic-Skripte in sicheren Bytecode umwandelt und sie so für Endbenutzer unzugänglich macht. Durch das Kompilieren von Teilen Ihrer VBA-Makros wird der ursprüngliche Quellcode entfernt, sodass er nicht kopiert werden kann.

> **Video-Tutorial**
>
> Ein Tutorial zum Kompilieren einer VBA-Funktion finden Sie in diesem Video.

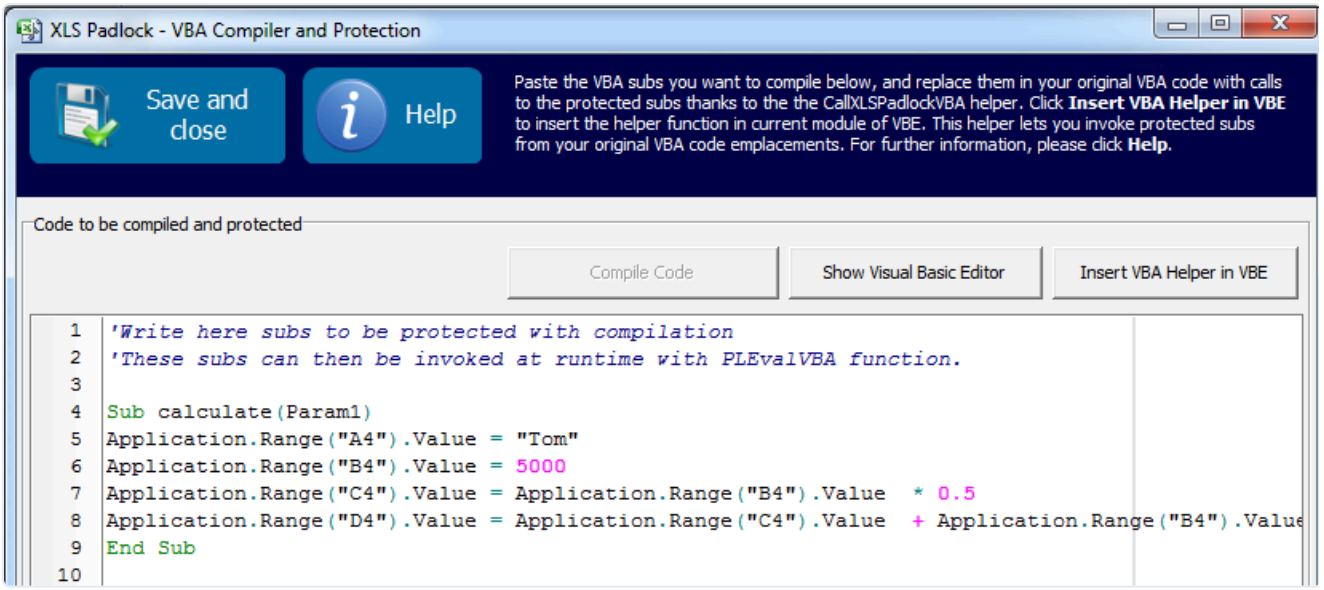
Betrachten Sie zum Beispiel diesen ursprünglichen Code:



Nach der Schützung wird der ursprüngliche `calculate()`-Sub durch einen Aufruf einer internen XLS Padlock-Funktion ersetzt, die den kompilierten Bytecode ausführt:



Der ursprüngliche Code wird in den XLS Padlock VBA Editor verschoben und von dort kompiliert:



Beachten Sie, dass einige Änderungen erforderlich sein können, da der Compiler die explizite Verwendung des `Application`-Objekts erfordert, um auf Excel-Objekte zuzugreifen.

Der Compiler ist kein einfacher Obfuskator; er wandelt Ihren VBA-Code vollständig in Binärcode um und speichert ihn sicher innerhalb der Anwendung. In Kombination mit dem Kennwortschutz Ihres VBA-Projekts macht dies Tools zum Knacken von Kennwörtern nutzlos, da sie nicht auf die ursprüngliche XLS-Datei zugreifen können.

Einschränkungen des Compilers

- Möglicherweise müssen Sie Ihren ursprünglichen VBA-Code ändern, damit er kompiliert werden kann, da der VBA-Compiler von XLS Padlock nicht so fortschrittlich ist wie der VB-Interpreter von Microsoft.
- Ganze Makros können in der Regel nicht kompiliert werden, sondern nur Teile davon.
- Einige Objekte und Standardeigenschaften sind nicht zugänglich.

👉 Siehe auch: [Erfahren Sie, wie Sie sicheren VBA-Code schreiben und kompilieren](#chapter-writing-and-compiling-secure-vba-code)

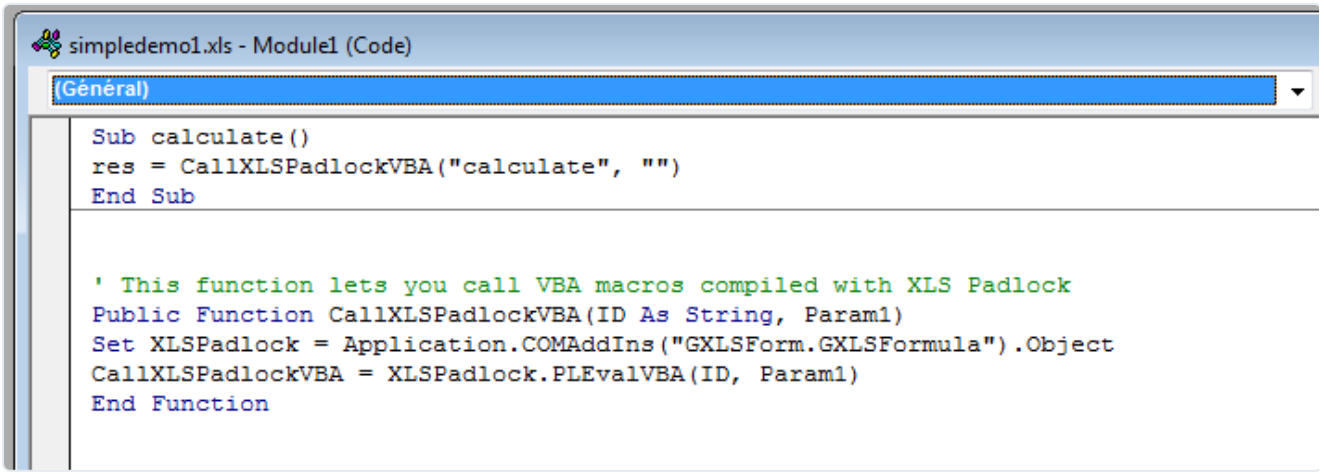
Echter VBA-Codeschutz

Manche Werkzeuge bieten eine VBA-Verschleierung an, machen Ihren Code damit aber nur schwer lesbar. Sie können die zugrunde liegende Logik nicht verbergen und auch nicht verhindern, dass diese kopiert wird.

👉 XLS Padlock geht weiter und ermöglicht es Ihnen, **den VBA-Code vollständig aus Ihrer Arbeitsmappe zu entfernen und dabei deren Funktionsfähigkeit zu erhalten**. Möglich wird dies durch den integrierten VBA Compiler.

Der **VBA Compiler von XLS Padlock wandelt Ihren VBA-Code in Bytecode um, der nur innerhalb der sicheren Anwendung ausgeführt werden kann**. Sollte es jemandem gelingen, auf die Datei der Arbeitsmappe zuzugreifen, wird er dort den ursprünglichen Code nicht finden, da er schlicht nicht vorhanden ist. Stattdessen wird der **kompilierte Bytecode sicher innerhalb der EXE gespeichert**.

Das Makro `calculate()` unten enthielt beispielsweise ursprünglich proprietären VBA-Code. Nach der Kompilierung enthält das Makro in der Arbeitsmappe nur noch einen Aufruf des kompilierten Codes. Die ursprüngliche Logik ist verschwunden.



Dies bietet einen **robusten Schutz**, da der ursprüngliche VBA-Code in der Arbeitsmappe nicht mehr existiert. Er wurde durch Bytecode ersetzt, der weder kopiert noch in einer anderen Excel-Arbeitsmappe verwendet werden kann.

Der Kompromiss besteht darin, dass Sie den zu schützenden VBA-Code **manuell in den VBA-Editor von XLS Padlock übertragen müssen**. Bewährte Praxis ist es, die kritischsten Teile Ihres Codes zu ermitteln, also jene, deren Fehlen Ihre Arbeitsmappe funktionsunfähig machen würde, und nur diese Abschnitte zu kompilieren.

👉 Siehe auch: Mehr über die Arbeit mit unserem VBA Compiler erfahren

Syntaxreferenz

Dieses Dokument bietet eine Referenz für die vom VBA Compiler von XLS Padlock unterstützte VBA-Syntax.

Skriptstruktur

Die Skriptstruktur besteht aus Deklarationen von Funktionen und Subroutinen (Sub).

```
SUB DoSomething
   CallSomething
END SUB

FUNCTION MyFunction
   MyFunction = "Ok!"
END FUNCTION
```

Anweisungen in einer einzelnen Zeile können durch das Zeichen `:` getrennt werden.

Kommentare

Kommentare können in ein Skript eingefügt werden. Sie können das Zeichen `'` oder `REM` verwenden. Der Kommentar erstreckt sich bis zum Ende der Zeile.

```
' This is a comment before ShowMessage
ShowMessage("Ok")

REM This is another comment
ShowMessage("More ok!")
```

Bezeichner

Bezeichnernamen (Variablen, Funktionen, Prozeduren usw.) müssen mit einem Buchstaben (a-z, A-Z) oder einem Unterstrich `_` beginnen. Darauf können alphanumerische Zeichen oder Unterstriche folgen. Namen dürfen keine anderen Zeichen oder Leerzeichen enthalten.

Variablen

Es ist nicht erforderlich, Variablen in einem Skript zu deklarieren, aber wenn Sie möchten, können Sie eine Variable mit der Direktive `DIM` deklarieren.

```
SUB Msg(Param1)
   DIM S
   S = "Hello world!"
   ShowMessage(S)
END SUB
```

Sie können globale Variablen auch als `PRIVATE` oder `PUBLIC` deklarieren.

```
PRIVATE A
PUBLIC B
B = 0
A = B + 1
ShowMessage(A)
```

Mit `DIM` deklarierte Variablen sind standardmäßig öffentlich. Private Variablen sind aus anderen Skripten nicht zugänglich. Variablen können auch bei der Deklaration initialisiert werden:

```
DIM A = "Hello world"
DIM B As Integer = 5
```

Zuweisungsanweisungen

Zuweisungsanweisungen verwenden den Operator `=`, um einer Variablen oder einer Objekteigenschaft einen Wert oder ein Ausdrucksergebnis zuzuweisen.

```
MyVar = 2
Application.Range("C4").Value = "This " + "is ok."
```

Arrays

Der Compiler bietet eine grundlegende Unterstützung für Array-Konstruktoren und Variant-Arrays. Um ein Array zu konstruieren, verwenden Sie die Zeichen `[` und `]`. Sie können ein mehrdimensionales Array konstruieren, indem Sie Array-Konstruktoren verschachteln.

Arrays im Compiler sind 0-basiert indiziert.

```
NewArray = [ 2,4,6,8 ]
Num = NewArray[1] 'Num receives 4

MultiArray = [ ["green","red","blue"] , ["apple","orange","lemon"] ]
Str = MultiArray[0,2] 'Str receives "blue"
MultiArray[1,1] = "new orange"
```

Dynamische Arrays:

```
' Create a dynamic array
DIM PTIM = VarArrayCreate([0,3000,0,5], 12)
' Assign a value:
PTIM[1,2] = 1530
```

Indizes

Zeichenketten, Arrays und Array-Eigenschaften können mit den Zeichen `[` und `]` indiziert werden. Wenn `Str` beispielsweise eine Zeichenkettenvariable ist, gibt der Ausdruck `Str[3]` das dritte Zeichen der Zeichenkette zurück.

```
MyChar = MyStr[2]
MyStr[1] = "A"
MyArray[1,2] = 1530
```

Schlüsselwörter und Operatoren

Die Basic-Syntax unterstützt:

- **Deklarationen:** `SUB...END SUB`, `FUNCTION...END FUNCTION`
- **Direktiven:** `BYREF`, `DIM`
- **Bedingungen:** `IF...THEN...ELSE...ELSEIF...END IF`, `SELECT CASE...END SELECT`
- **Schleifen:** `FOR...TO...STEP...NEXT`, `DO...WHILE...LOOP`, `DO...LOOP...WHILE`, `DO...UNTIL...LOOP`, `DO...LOOP...UNTIL`
- **Operatoren:** `^`, `*`, `/`, `AND`, `+`, `-`, `OR`, `<>`, `⩾`, `⩽`, `=`, `>`, `<`, `DIV`, `MOD`, `XOR`, `SHL`, `SHR`
- **Fehlerbehandlung:** `TRY...EXCEPT`, `TRY...FINALLY`
- **Sonstiges:** `EXIT`, Array-Konstruktoren `[1, 2, 3]`, Objektzugriff `ObjectName.Property`

`If`-Anweisungen

Es gibt zwei Formen: `IF...THEN...END IF` und `IF...THEN...ELSE...END IF`. Wenn der Ausdruck wahr ist, werden die `THEN`-Anweisungen ausgeführt. Wenn er falsch ist, werden die `ELSE`-Anweisungen ausgeführt (sofern vorhanden).

```
IF J <> 0 THEN
   Result = I/J
END IF

IF J = 0 THEN
   Exit
ELSE
   Result = I/J
END IF
```

Wenn die Anweisung in einer einzelnen Zeile steht, benötigen Sie kein `END IF`:

```
IF J <> 0 THEN Result = I/J
```

`Select Case`-Anweisungen

Wenn die `selectorExpression` mit einem der `caseexpr`-Ausdrücke übereinstimmt, werden die jeweiligen Anweisungen ausgeführt. Andernfalls wird die `CASE ELSE`-Anweisung ausgeführt.

```
SELECT CASE uppercase(Fruit)
  CASE "lime"
    ShowMessage("green")
  CASE "orange"
    ShowMessage("orange")
  CASE ELSE
    ShowMessage("black")
END SELECT
```

`For`-Anweisungen

Die `FOR`-Anweisung wiederholt die Ausführung von Anweisungen, bis ein Zähler einen Endwert erreicht.

```
FOR counter = initialValue TO finalValue STEP stepValue
  Statements
NEXT
```

Der `STEP`-Teil ist optional; wird er weggelassen, beträgt der Schrittwert 1.

```
FOR c = 1 TO 10 STEP 2
 a = a + c
NEXT
```

`While`-Anweisungen

Eine `WHILE`-Anweisung wiederholt Anweisungen, solange eine Steuerbedingung wahr ist. Die Bedingung wird vor der Ausführung der Anweisungen ausgewertet.

```
WHILE (Data[I] <> X)
 I = I + 1
END WHILE
```

`Do...Loop`-Anweisungen

Anweisungen werden ausgeführt, `WHILE` (solange) ein Ausdruck wahr ist, oder `UNTIL` (bis) ein Ausdruck wahr ist. Die Bedingung kann vor oder nach der Iteration geprüft werden.

```
' Condition tested after
DO
   K = I mod J
   I = J
   J = K
LOOP UNTIL J = 0

' Condition tested before
DO WHILE I < 0
   ...
LOOP
```

Deklarationen von Funktionen und Subroutinen

Die Deklarationen ähneln dem Standard-Basic. Um einen Wert aus einer Funktion zurückzugeben, weisen Sie ihn der impliziten Variablen mit demselben Namen wie die Funktion zu oder verwenden Sie die `Return`-Anweisung. Parameter können mit der Direktive `BYREF` per Referenz übergeben werden.

```
SUB HelloWord
   ShowMessage("Hello world!")
END SUB

FUNCTION Max(A,B)
   IF A > B THEN
      MAX = A
   ELSE
      MAX = B
   END IF
END FUNCTION

SUB SwapValues(BYREF A, B)
  DIM TEMP
  TEMP = A
  A = B
  B = TEMP
END SUB
```

Subroutinen (Sub) und Funktionen sind standardmäßig öffentlich, können aber als `PRIVATE` deklariert werden.

VBA-API-Kochbuch

Diese Rubrik bietet eine umfassende Sammlung von VBA-Code-Ausschnitten („Rezepten“) für die erweiterte Steuerung Ihrer geschützten Anwendung zur Laufzeit. Mit der VBA-API von XLS Padlock können Sie Systeminformationen abrufen, das Speichern und Laden verwalten, die Benutzeroberfläche anpassen und mit Online-Funktionen interagieren.

Die beiden wichtigsten API-Objekte sind: `Application.COMAddIns("GXLSForm.GXLSFormula").Object` für die meisten Laufzeitinformationen und `Application.COMAddIns("GXLS.GXLSPLock").Object` für speicherbezogene Operationen.

Jedes Rezept unten stellt eine einsatzbereite VBA-Funktion oder -Subroutine bereit.

Anwendungs- und Systeminformationen

Diese Rezepte helfen Ihnen, Informationen über die Umgebung und den Status der Anwendung abzurufen.

Prüfen, ob die Arbeitsmappe geschützt ist

Verwenden Sie dies, um zu bestätigen, dass Ihr Code innerhalb einer von XLS Padlock erstellten geschützten Anwendung ausgeführt wird.

```
Public Function XLSPadlockAvailable() As Boolean
    Dim XLSPadlock As Object
    On Error Resume Next
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    If Not XLSPadlock Is Nothing Then
        XLSPadlockAvailable = XLSPadlock.IsProtected()
    Else
        XLSPadlockAvailable = False
    End If
End Function
```

Den EXE-Dateinamen der Anwendung abrufen

Ruft den Dateinamen der laufenden `.exe`-Datei ab.

```
Public Function GetEXEFilename() As String
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    GetEXEFilename = XLSPadlock.PLEvalVar("EXEFilename")
    Exit Function
Err:
    GetEXEFilename = ""
End Function
```

EXE-Datei- und Produktversion abrufen

Ruft die Versionszeichenketten ab, die Sie im Abschnitt "EXE Version Info" (EXE-Versionsinformationen) von XLS Padlock definiert haben.

```
Public Function ReadVersionInfo(infoType As String) As String
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    ReadVersionInfo = XLSPadlock.PLEvalVar(infoType)
    Exit Function
Err:
    ReadVersionInfo = ""
End Function
```

Verwendung: `ReadVersionInfo("ProductVersion")` gibt die Product Version zurück, und `ReadVersionInfo("FileVersion")` gibt die File Version zurück.

Die System ID des Benutzers abrufen

Ruft die eindeutige System ID ab, die für hardwaregebundene Activation Keys erforderlich ist. Dies ist die ID, die Ihnen Ihre Kunden zusenden werden.

```
Public Function ReadSystemID() As String
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    ReadSystemID = XLSPadlock.PLEvalVar("SystemID")
    Exit Function
Err:
    ReadSystemID = ""
End Function
```

Befehlszeilenparameter abrufen

Ruft die Befehlszeilenparameter ab, die an die `.exe`-Datei Ihrer Anwendung übergeben wurden.

```
Public Function ReadParamStr(index As Integer) As String
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    ReadParamStr = XLSPadlock.PLEvalVar("ParamStr" & index)
    Exit Function
Err:
    ReadParamStr = ""
End Function
```

Verwendung: `ReadParamStr(1)` gibt den ersten Parameter zurück, `ReadParamStr(2)` den zweiten und so weiter.

Arbeitsmappen speichern und laden

Verwalten Sie sichere Speicherdateien (`.xlsc` oder `.xlsce`) programmgesteuert.

Pfad der aktuellen Speicherdatei abrufen

Ruft den vollständigen Pfad zur aktuell geladenen sicheren Speicherdatei ab.

```
Public Function GetSecureWorkbookFilename() As String
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLS.GXLSPLock").Object
    GetSecureWorkbookFilename = XLSPadlock.GetSaveFilename()
    Exit Function
Err:
    GetSecureWorkbookFilename = ""
End Function
```

Pfad des lokalen Speicherordners abrufen

Ruft den Pfad zum lokalen Ordner ab, in dem XLS Padlock die Einstellungen Ihrer Anwendung und die sicheren Speicherdateien ablegt.

```
Public Function GetStoragePath() As String
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    GetStoragePath = XLSPadlock.PLEvalVar("SPath")
    Exit Function
Err:
    GetStoragePath = ""
End Function
```

VBA-Code nach dem Speichern ausführen

XLS Padlock kann die Subroutine `XLSPadlock_OnAfterSave` automatisch aufrufen, nachdem ein Benutzer seine Arbeit gespeichert hat. Dies ist nützlich für die Protokollierung, das Anzeigen einer Bestätigungsmeldung oder andere Aktionen nach dem Speichern.

```
Sub XLSPadlock_OnAfterSave(SaveFilename As String)
    MsgBox "The workbook has been successfully saved as: " & SaveFilename
End Sub
```

Eine vorhandene Speicherdatei mit VBA öffnen

Öffnen Sie eine sichere Speicherdatei programmgesteuert. Beachten Sie, dass die Datei dabei in einer **neuen Instanz von Excel** geladen wird.

```
Public Sub LoadXLSPadlockSaveFile(FilePath As String)
    Dim XLSPadlock As Object
    On Error Resume Next
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    If Not XLSPadlock Is Nothing Then
        XLSPadlock.PLOpenSaveFile FilePath
    End If
End Sub

' Example of a caller function that shows a file browser:
Sub Load_Old_Save()
    Dim strFileToOpen As String
    strFileToOpen = Application.GetOpenFilename( _
        Title:="Please choose a save file to open", _
        FileFilter:="Save Files (*.xlsc),*.xlsc")

    If strFileToOpen <> "False" Then
        LoadXLSPadlockSaveFile strFileToOpen
    End If
End Sub
```

Einen Dateinamen für den Speicherdialog vorschlagen

Legt den Standarddateinamen fest, der im Dialogfeld "Save As" (Speichern unter) angezeigt wird.

```
Public Sub SetSecureWorkbookFilename(Filename As String)
    Dim XLSPadlock As Object
    On Error Resume Next
    Set XLSPadlock = Application.COMAddIns("GXLS.GXLSPLock").Object
    If Not XLSPadlock Is Nothing Then
        XLSPadlock.SetDefaultSaveFilename Filename
    End If
End Sub

' Example: Set the default name to "my save.xlsc"
' Call SetSecureWorkbookFilename("D:\My Documents\my save.xlsc")
```

Eine sichere Kopie ohne Abfrage speichern

Speichern Sie eine sichere Kopie der Arbeitsmappe direkt unter einem angegebenen Dateipfad, ohne das Dialogfeld "Save As" (Speichern unter) anzuzeigen.

```
Public Function SaveSecureWorkbookToFile(Filename As String) As Boolean
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLS.GXLSPLock").Object
    SaveSecureWorkbookToFile = XLSPadlock.SaveWorkbook(Filename)
    Exit Function
Err:
    SaveSecureWorkbookToFile = False
End Function

' Example: Save the workbook directly
' If SaveSecureWorkbookToFile("D:\My Documents\my save.xlsc") Then
'   MsgBox "Workbook saved successfully."
' End If
```

Benutzerdaten aus einer früheren Version migrieren

Wenn Sie eine neue Version Ihrer Arbeitsmappe veröffentlichen, müssen Sie möglicherweise Daten aus einer mit einer früheren Version erstellten Speicherdatei importieren.

➞ Erfahren Sie, wie Sie Benutzerdaten mit VBA migrieren

Benutzeroberfläche

Passen Sie UI-Elemente wie Dialogfelder an.

Warte-/Ladedialoge programmgesteuert ausblenden

Sie können das Dialogfeld "Loading workbook, please wait…" (Arbeitsmappe wird geladen, bitte warten…) mit einem VBA-Aufruf ausblenden, anstatt auf das automatische Schließen zu warten.

```
Public Sub HideLoadingDialog()
    Dim XLSPadlock As Object
    On Error Resume Next
    Set XLSPadlock = Application.COMAddIns("GXLS.GXLSPLock").Object
    If Not XLSPadlock Is Nothing Then
        ' A call with Option "3" and Value "0" hides the dialog.
        XLSPadlock.SetOption Option:="3", Value:="0"
    End If
End Sub
```

Online-Aktivierung und -Validierung

Interagieren Sie mit den Online-Funktionen von XLS Padlock.

Auf eine Internetverbindung prüfen

Testet, ob auf dem Computer des Benutzers eine aktive Internetverbindung verfügbar ist.

```
Public Function IsInternetAvailable() As Boolean
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    IsInternetAvailable = (XLSPadlock.PLEvalVar("InternetConnected") = "1")
    Exit Function
Err:
    IsInternetAvailable = False
End Function
```

Das Online-Aktivierungstoken abrufen

Ruft einen Hash des Aktivierungstokens ab, das der Webserver nach einer erfolgreichen Online-Aktivierung zurückgegeben hat.

```
Public Function GetValidationToken() As String
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    GetValidationToken = XLSPadlock.PLEvalVar("ValidationToken")
    Exit Function
Err:
    GetValidationToken = ""
End Function
```

Prüfen, ob die Online-Validierung erfolgreich war

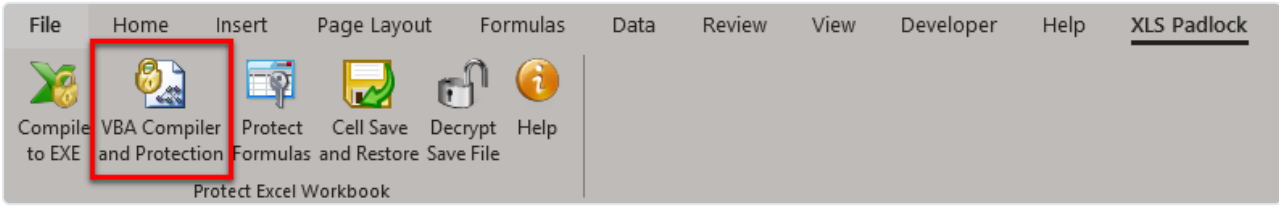
Ruft das Ergebnis des letzten Online-Validierungsversuchs ab.

```
Public Function IsValidationOK() As Boolean
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    IsValidationOK = (XLSPadlock.PLEvalVar("ValidationSuccess") = "1")
    Exit Function
Err:
    IsValidationOK = False
End Function
```

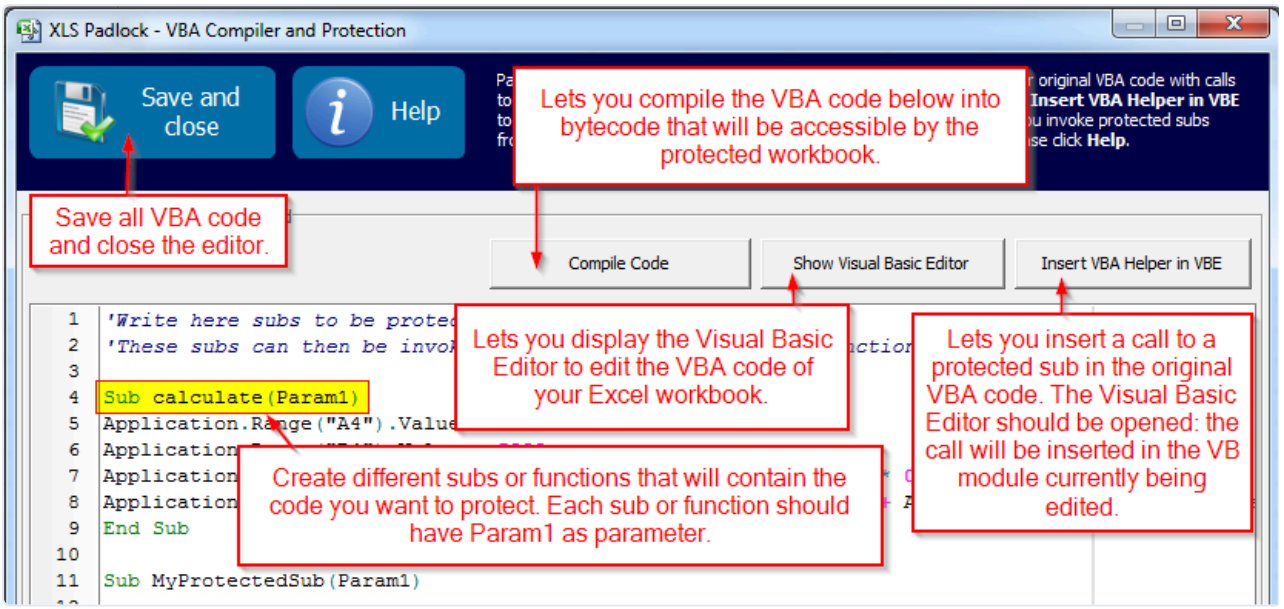
Sicheren VBA-Code schreiben und kompilieren

Sie können **sensible Teile Ihres VBA-Codes schützen, indem Sie sie mit XLS Padlock in Bytecode kompilieren**. Wir empfehlen, kompilierten Bytecode mit Ihrem vorhandenen VBA-Code zu kombinieren, um ein ausgewogenes Verhältnis zwischen Sicherheit und Entwicklung zu erreichen.

1. Klicken Sie in Excel auf „VBA Compiler and Protection", um den VBA-Editor und -Compiler von XLS Padlock zu öffnen:



   Der VBA-Editor von XLS Padlock wird geöffnet:



2. Geben Sie im Textbereich den VBA-Code ein, den Sie kompilieren möchten. Sie können Ihren Code in `subs` und `functions` gruppieren. Um diesen Vorgang zu vereinfachen, können Sie den Visual Basic Editor öffnen, indem Sie auf „Show Visual Basic Editor" klicken und Ihren Code kopieren und einfügen.

   Standardmäßig haben alle subs und functions einen Parameter namens `Param1`. Sie können jedoch auch mehr Parameter verwenden.

3. Wenn Sie fertig sind, klicken Sie auf „**Compile Code**". Der VBA-Code wird sofort in Bytecode kompiliert, den XLS Padlock speichert, bis Sie die EXE-Datei Ihrer Anwendung kompilieren.

> **TIPP**
>
> Wenn ein Fehler erkannt wird, wird er angezeigt und der entsprechende Code rot unterstrichen.

1. Klicken Sie auf „Save and close", um den Editor zu schließen.

👉 Nach dem Kompilieren Ihres Codes müssen Sie **Ihre ursprünglichen VBA-Makros so ändern, dass sie die kompilierte Version aufrufen**. Wie das geht, erfahren Sie im Leitfaden Aufrufen von kompiliertem VBA-Code zur Laufzeit.

Kompilierten VBA-Code zur Laufzeit aufrufen

Um Ihren kompilierten VBA-Code auszuführen, müssen Sie ihn aus dem regulären VBA-Code Ihrer Arbeitsmappe über eine Hilfsfunktion aufrufen.

Öffnen Sie zunächst den Visual Basic Editor und fügen Sie die folgende Hilfsfunktion in ein Modul Ihrer Arbeitsmappe ein:

```
' This function lets you call VBA macros compiled with XLS Padlock
Public Function CallXLSPadlockVBA(ID As String, Param1 As Variant) As Variant
    Dim XLSPadlock As Object
    On Error Resume Next
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    CallXLSPadlockVBA = XLSPadlock.PLEvalVBA(ID, Param1)
End Function
```

> **Automatisches Einfügen**
>
> Diese Funktion kann automatisch in Ihr aktives VB-Modul eingefügt werden, indem Sie im VBA-Editor von XLS Padlock auf die Schaltfläche "Insert VBA Helper in VBE" (VBA-Hilfsfunktion in den VBE einfügen) klicken.

Die Hilfsfunktion `CallXLSPadlockVBA` nimmt zwei Parameter entgegen:

- **ID**: Der Name des kompilierten `Sub` oder der kompilierten `Function`, die Sie aufrufen möchten.
- **Param1**: Ein optionaler Parameter, den Sie an den kompilierten Code übergeben möchten.

Beispiel

Stellen Sie sich vor, Ihr ursprüngliches VBA-Modul enthält den folgenden Code:

```
Sub Calculate()
    Range("A4") = "Tom"
    Range("B4") = 5000
    Range("C4") = Range("B4") * 0.5
    Range("D4") = Range("C4") + Range("B4")
End Sub
```

Nachdem Sie diesen Code in den VBA Compiler verschoben haben, würden Sie das ursprüngliche Sub durch einen Aufruf der Hilfsfunktion ersetzen:

```
Sub Calculate()
    Dim res As Variant
    res = CallXLSPadlockVBA("Calculate", "")
End Sub
```

Hier geben wir den Namen des geschützten Subs, `Calculate`, an und übergeben für den zweiten Parameter eine leere Zeichenfolge, da er in diesem Fall nicht verwendet wird. Die Variable `res` enthält das Ergebnis des Aufrufs, was vor allem dann nützlich ist, wenn der aufgerufene Code eine `Function` ist, die einen Wert zurückgibt.

> **Ausführung zur Entwurfszeit**
>
> Eine der leistungsstarken Funktionen von XLS Padlock besteht darin, dass diese Einrichtung selbst zur Entwurfszeit funktioniert. Wenn Sie das Sub `Calculate()` aus dem VBE ausführen, bevor Sie die Arbeitsmappe kompilieren, führt XLS Padlock den geschützten Code aus, sodass Sie Ihre Logik testen können, ohne die Anwendung zuvor kompilieren zu müssen.

👉 Siehe auch: Erfahren Sie, wie Sie [mehr als einen Parameter an Ihren kompilierten Code übergeben](#chapter-passing-more-parameters-to-the-compiled-vba-code).

Zugriff auf Excel-Objekte

Wenn Sie Code für den VBA Compiler schreiben, müssen Sie das `Application`-Objekt ausdrücklich verwenden, um auf das Objektmodell von Excel zuzugreifen.

Eine ausführliche Referenz der verfügbaren Eigenschaften und Methoden finden Sie in der Dokumentation des Microsoft Office Dev Center.

Beispiele

Eine Range referenzieren

Um eine Zelle oder einen Zellbereich auf Ihrem Tabellenblatt zu referenzieren, müssen Sie dem `Range`-Objekt `Application` voranstellen.

- **Standard-VBA:** `Range("A1:A4").Value = 2`
- **Kompiliertes VBA:** `Application.Range("A1:A4").Value = 2`

Tabellenfunktionen verwenden

Um Excel-Tabellenfunktionen zu verwenden, müssen Sie das Objekt `Application.WorksheetFunction` verwenden.

`Str1 = Application.WorksheetFunction.VLookup(Param1.ComboBox1, Application.Range("A2:C8"), 2, False)`

> **Funktionen erfordern alle Parameter**
>
> Wenn Sie integrierte Excel-Funktionen wie `InputBox` aus kompiliertem Code aufrufen, **müssen Sie alle Parameter angeben**, da Standardparameter nicht unterstützt werden.

Vollständiges Beispiel: Umwandlung eines Makros

Hier ist ein Standard-VBA-Makro:

```
Sub test()
    Dim qty As Integer
    Dim price As Single, amount As Single
    Range("A5").Value = "Item"
    ActiveCell.Offset(1, 0).Select
    ActiveCell.Value = InputBox("Enter the name of item")
    ActiveCell.Offset(0, 1).Select
    price = InputBox("Enter the price")
    ActiveCell.Value = price
    ' ... and so on
End Sub
```

Hier ist derselbe Code, angepasst für den VBA Compiler von XLS Padlock. Beachten Sie die Hinzufügung des `Application`-Objekts und der expliziten Parameter für `InputBox`.

```
Sub test(Param1)
    Dim qty As Integer
    Dim price As Single, amount As Single
    Application.Range("A5").Value = "Item"
    Application.ActiveCell.Offset(1, 0).Select
    Application.ActiveCell.Value = Application.InputBox("Enter the name of item", "Name", "")
    Application.ActiveCell.Offset(0, 1).Select
    price = Application.InputBox("Enter the price", "Price", "")
    Application.ActiveCell.Value = price
    ' ... and so on
End Sub
```

Ihr ursprüngliches VBA-Modul würde dann so angepasst, dass es die kompilierte Version aufruft:

```
Sub test()
    res = CallXLSPadlockVBA("test", "")
End Sub
```

Arrays übergeben

Sie können verschiedene Variablentypen an den kompilierten VBA-Code übergeben, einschließlich statischer Arrays.

Angenommen, Sie haben die folgende Funktion im VBA Compiler:

```
Function TestMultipleParams(Param1, Param2, Param3)
  MsgBox(Param2(1))
  TestMultipleParams = Param3 ^ 2
End Function
```

In Ihrem normalen Excel-VBA-Modul können Sie diese Funktion aufrufen und ein Array übergeben.

> **Warnung**
>
> Das Array muss als **Variant** definiert werden.

```
Sub MySubSample4()
    Dim XLSPadlock As Object
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object

    Dim NomTableau(2) As Variant
    NomTableau(0) = "a"
    NomTableau(1) = "b"
    NomTableau(2) = "c"

    MsgBox XLSPadlock.PLEvalVBA3("TestMultipleParams", "Param1", NomTableau, 3)

    Set XLSPadlock = Nothing
End Sub
```

Weitere Parameter übergeben

Standardmäßig unterstützt die Hilfsfunktion zum Ausführen von kompiliertem VBA-Code nur einen Parameter.

```
Public Function CallXLSPadlockVBA(ID As String, Param1)
    Dim XLSPadlock As Object
    On Error Resume Next
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    CallXLSPadlockVBA = XLSPadlock.PLEvalVBA(ID, Param1)
End Function
```

Die Methode `PLEvalVBA` des Objekts `XLSPadlock` nimmt zwei Parameter entgegen: die **ID** der kompilierten Sub/Funktion und einen einzelnen Parameter **Param1**.

Um mehr Parameter zu übergeben, stellt das Objekt `XLSPadlock` zusätzliche Methoden bereit:

- `XLSPadlock.PLEvalVBA2(ID, Param1, Param2)` für zwei Parameter.
- `XLSPadlock.PLEvalVBA3(ID, Param1, Param2, Param3)` für drei Parameter.

Es ist auch möglich, mehr Parameter mithilfe von Arrays zu übergeben.

Eine Hilfsfunktion zur Übergabe von zwei Parametern würde beispielsweise so aussehen:

```
Public Function CallXLSPadlockVBA2(ID As String, Param1, Param2)
    Dim XLSPadlock As Object
    On Error Resume Next
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    CallXLSPadlockVBA2 = XLSPadlock.PLEvalVBA2(ID, Param1, Param2)
End Function
```

Fehler behandeln

Bei der Ausführung Ihres kompilierten VBA-Codes können Laufzeitfehler auftreten. Während Standard-VBA die Anweisung `On Error` zur Fehlerbehandlung verwendet, nutzt der VBA Compiler von XLS Padlock einen `Try...Except`-Block.

Wenn innerhalb des `Try`-Blocks (oder in einer von ihm aufgerufenen Prozedur) ein Fehler auftritt, springt der Compiler zur Behandlung sofort zum `Except`-Block.

Syntax

```
Try
    ' ... Code to execute ...
Except
    ' ... Code to run if an exception occurs ...
End
```

> **TIPP**
>
> Ein nicht behandelter Fehler wird vom VBA Compiler zur Laufzeit angezeigt, sofern Sie diese Option nicht deaktivieren.

Codebeispiel

```
NumberStr = ""
if InputQuery("Input", "Type an integer from 1 to 7", NumberStr) then
    try
        Number = StrToFloat(NumberStr)
    except
        raise("Not a valid number")
    end

    select case Number
        case 1
            ShowMessage("One")
        case 1 + 1
            ShowMessage("Two")
        case 4.5 / 1.5
            ShowMessage("Three")
        case 2 * 2
            ShowMessage("Four")
        case Length("xxxxx")
            ShowMessage("Five")
        case 3 + 3, 3 + 4
            ShowMessage("Six or Seven")
        case else
            ShowMessage("You did not type an integer from 1 to 7")
    end select
end if
```

OLE error 800A03EC

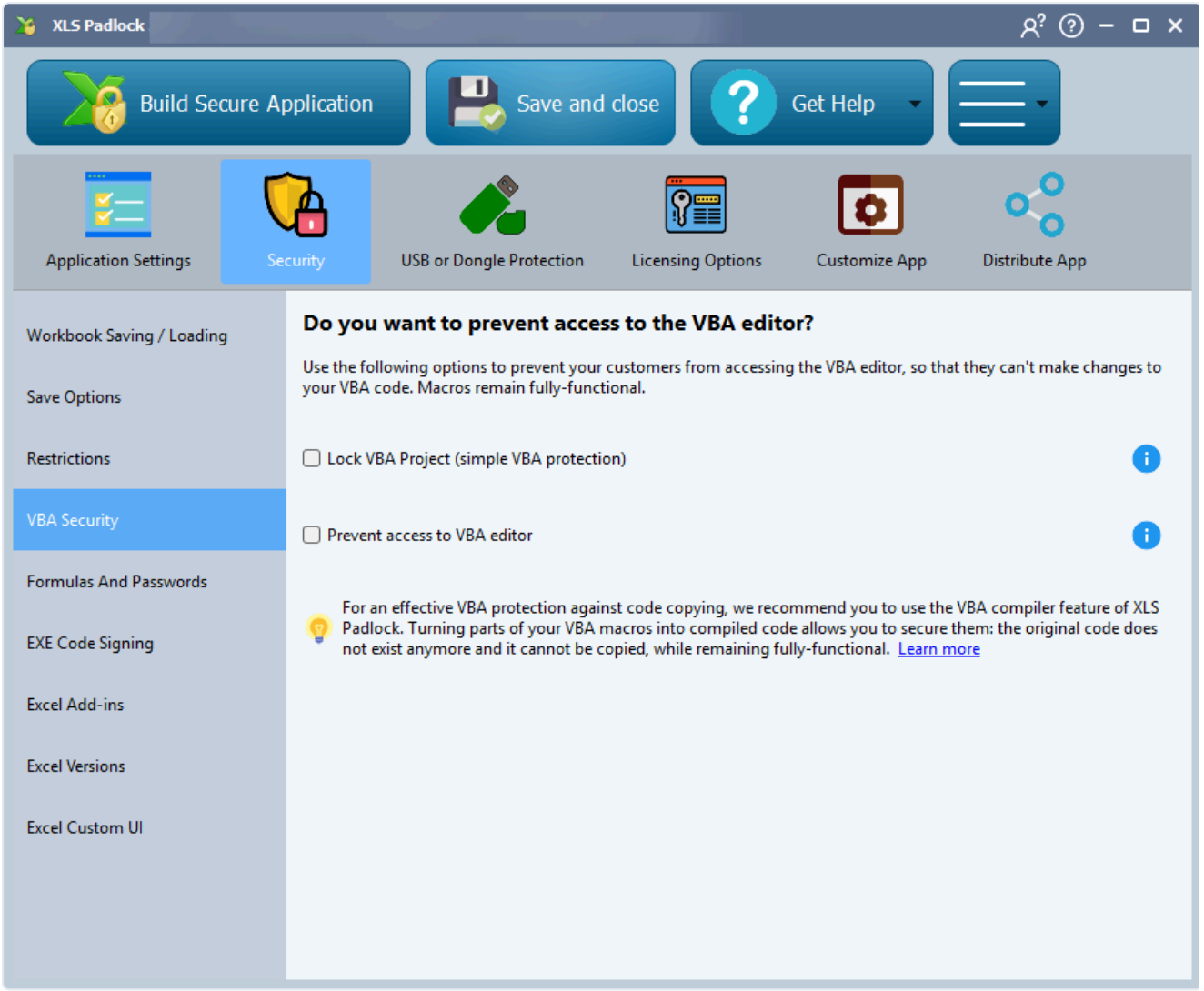
Stellen Sie zur Behebung dieses Fehlers sicher, dass die Option „**Trust access to the VBA project object model**“ in den Einstellungen des Trust Center von Excel aktiviert ist.

Eine Anleitung, wie Sie diese Option finden, finden Sie auf der offiziellen Supportseite von Microsoft Office:

Aktivieren oder Deaktivieren von Makros in Office-Dateien

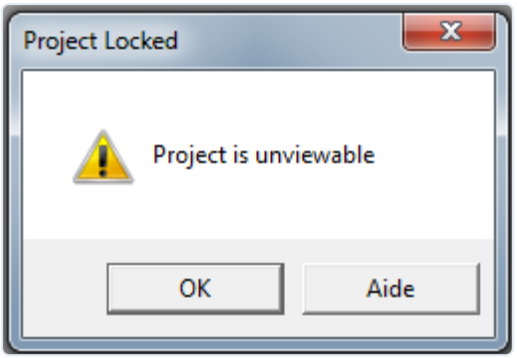
VBA-Codeschutz

Ihr VBA-Code ist ein wertvolles Gut. XLS Padlock bietet mehrere Schutzebenen, um ihn vor Manipulation oder Diebstahl zu sichern. Nachfolgend finden Sie zwei grundlegende Sicherheitsoptionen, gefolgt von unserer fortschrittlichsten Lösung für maximalen Schutz.



Option 1: Sperren Sie Ihr VBA-Projekt

Der einfachste Schritt besteht darin, Ihr VBA-Projekt zu sperren. Dadurch ist Ihr Code im Visual Basic Editor nicht mehr einsehbar. Wenn ein Benutzer versucht, auf das Projekt zuzugreifen, erhält er eine Fehlermeldung, was die beiläufige Einsicht in Ihren Code wirksam blockiert.



Option 2: Verhindern Sie den Zugriff auf den VBA-Editor

Für eine zusätzliche Sicherheitsebene können Sie das Öffnen des VBA-Editors vollständig verhindern. XLS Padlock überwacht kontinuierlich jeden Versuch, den VBE zu öffnen, und schließt ihn automatisch, sodass selbst versierte Benutzer blockiert werden.

Ultimativer Schutz: VBA-Kompilierung

Für das höchste Sicherheitsniveau ermöglicht Ihnen XLS Padlock, **Ihren VBA-Code in nativen Binärcode zu kompilieren**. Anders als die einfache Verschleierung (Obfuscation), die den Code lediglich schwer lesbar macht, wandelt die Kompilierung Ihre Logik in ein für Menschen nicht lesbares Format um, das nahezu unmöglich zurückzuentwickeln ist. Ihr ursprünglicher VBA-Code wird aus der Arbeitsmappe entfernt und durch Aufrufe des sicheren, kompilierten Codes innerhalb der EXE ersetzt. Dies ist der ultimative Schutz für Ihr geistiges Eigentum.

➞ Erfahren Sie mehr über unseren leistungsstarken VBA Compiler

VBA-Code ausblenden und sperren

XLS Padlock bietet eine dedizierte Funktion Lock VBA Project (VBA-Projekt sperren). Falls Sie sich jedoch dagegen entscheiden, diese zu verwenden, können Sie weiterhin auf den nativen Schutz von Excel zurückgreifen. XLS Padlock macht diese Methode sicherer.

So sperren Sie Ihr VBA-Projekt in Excel

1. Öffnen Sie in Excel den Visual Basic Editor (VBE), indem Sie `Alt+F11` drücken oder ihn über die Registerkarte `Developer` auswählen.
2. Gehen Sie im VBE in das Menü **Tools** und wählen Sie **VBAProject Properties…**.
3. Wechseln Sie im Dialogfeld zur Registerkarte **Protection**.
4. Aktivieren Sie das Kontrollkästchen **Lock project for viewing** und geben Sie ein starkes Passwort ein.

Sobald die Arbeitsmappe gespeichert und erneut geöffnet wurde, können Benutzer den VBA-Code ohne das Passwort weder anzeigen noch bearbeiten.

Warum dies mit XLS Padlock sicherer ist

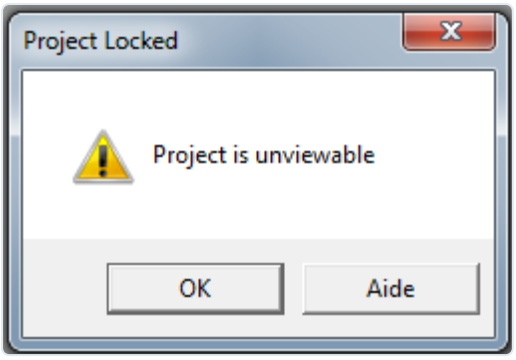
Normalerweise gilt der VBA-Passwortschutz von Excel als schwach, da zahlreiche Tools existieren, mit denen sich das Passwort entfernen lässt. Diese Tools funktionieren, indem sie direkt auf die Excel-Datei zugreifen.

Wenn Ihre Arbeitsmappe jedoch mit XLS Padlock zu einer EXE kompiliert wird, wird die ursprüngliche `.xls`- oder `.xlsm`-Datei verschlüsselt und in die Anwendung eingebettet. Tools zum Knacken von Passwörtern können nicht auf diese gesicherte Datei zugreifen, wodurch sie nutzlos werden. Diese Kombination bietet einen wesentlich stärkeren Schutz für Ihren VBA-Code.

VBA-Projekt sperren

Die Option **Lock VBA Project** bietet eine einfache Möglichkeit, Endbenutzer am Zugriff auf Ihr VBA-Projekt zu hindern. Diese Funktion markiert das VBA-Projekt als gesperrt, sodass es nicht angezeigt, aufgerufen oder geändert werden kann. Sie verwendet keinen Passwortschutz.

Wenn der Endbenutzer versucht, auf ein gesperrtes VBA-Projekt zuzugreifen, wird die folgende Fehlermeldung angezeigt (Project Locked, Project is unviewable):



Vorhandene VBA-Passwörter entfernen

Wenn Sie diese Option verwenden, empfehlen wir Ihnen, jedes vorhandene Passwort aus Ihrem VBA-Projekt zu entfernen. Ein Passwort wird überflüssig, wenn das Projekt gesperrt ist. Darüber hinaus kann die Verwendung eines langen Passworts mit dieser Option zu Abstürzen in Excel 2007 führen.

Wenn Sie ein Passwort beibehalten möchten, sollten Sie stattdessen die Option [Zugriff auf den VBA-Editor verhindern](#chapter-forbid-access-to-the-vba-editor-vbe) in Betracht ziehen.

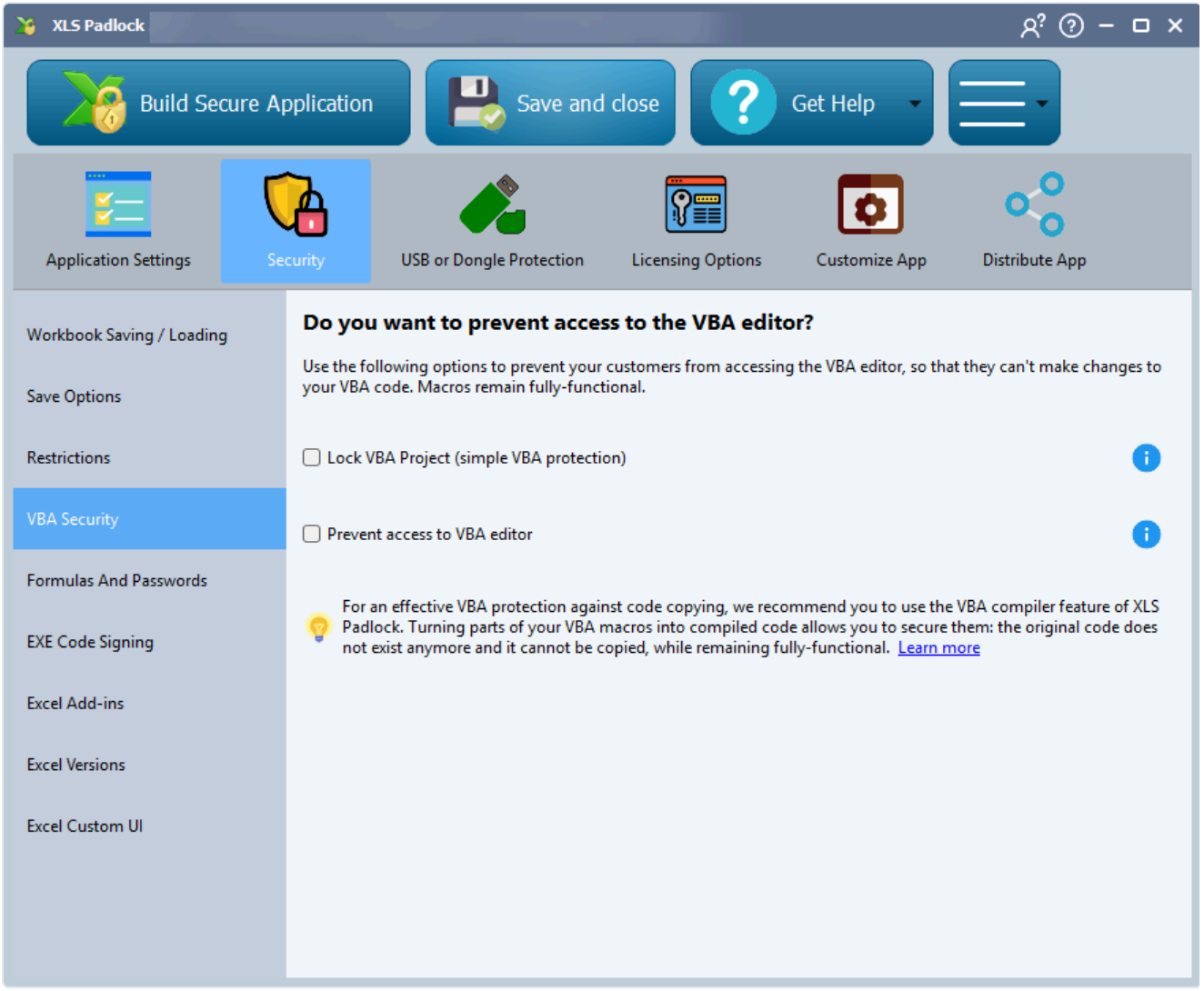
Beachten Sie, dass Makros funktionsfähig bleiben.

Diese Option ist mit unserem VBA Compiler kompatibel, der ein höheres Maß an Sicherheit für Ihren VBA-Code bietet.

VBE-Zugriff verbieten

XLS Padlock bietet Ihnen Optionen, um **zu verhindern, dass Ihre Kunden auf den VBA-Editor** (VBE) zugreifen, sodass sie Ihren VBA-Code weder einsehen noch verändern können, während Ihre Makros vollständig funktionsfähig bleiben.

Diese Optionen sind auf der Registerkarte "Security" von XLS Padlock verfügbar.



Lock VBA Project (einfacher VBA-Schutz)

Diese Option sperrt Ihr VBA-Projekt programmatisch, um den Zugriff zu erschweren. Sie bietet zwar ein grundlegendes Schutzniveau, kann aber von einigen Tools umgangen werden.

Prevent Access to VBA Editor

Dies ist eine stärkere Sicherheitsmaßnahme. XLS Padlock führt eine regelmäßige Prüfung durch, und wenn ein Benutzer versucht, den Visual Basic Editor (VBE) zu öffnen, schließt XLS Padlock das VBE-Fenster sofort. Dies verhindert, dass sie überhaupt versuchen, den Code einzusehen.

Diese Option ist mit der Option "Lock VBA Project" und unserem leistungsstarken VBA Compiler kompatibel.

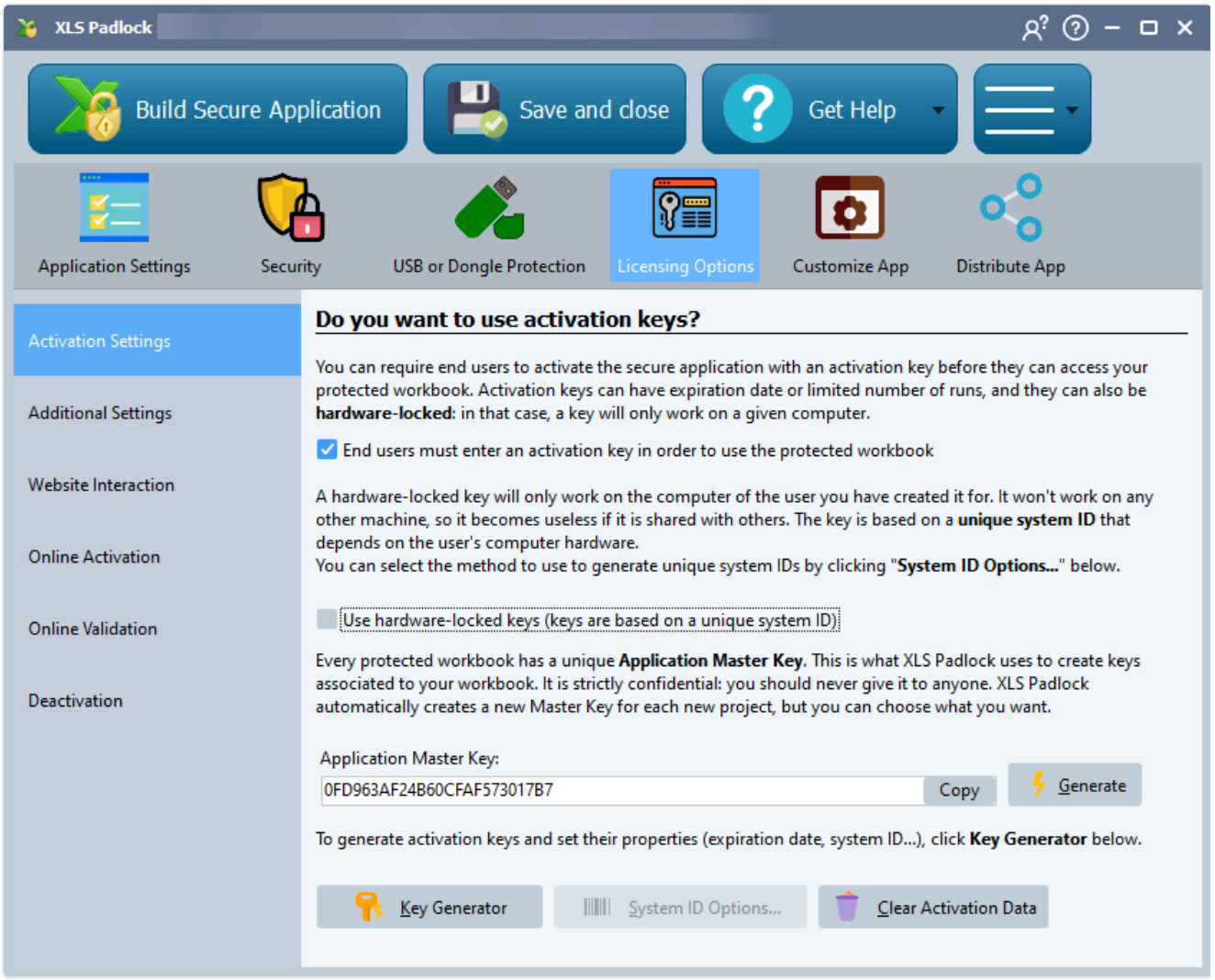
Für ultimativen Schutz des VBA-Codes

👉 Für robusten Schutz vor Codediebstahl und Reverse Engineering empfehlen wir dringend die Verwendung der **VBA Compiler-Funktion**.

Durch das Kompilieren von Teilen Ihrer VBA-Makros wird der ursprüngliche Quellcode entfernt und durch Binärcode ersetzt. Dieser kompilierte Code ist extrem schwer durch Reverse Engineering nachzuvollziehen, wodurch Ihre Algorithmen und Ihre Geschäftslogik geschützt sind. Es ist die wirksamste Methode, um Ihr geistiges Eigentum in VBA zu schützen.

Leitfaden zu Aktivierung und Lizenzierung

Dieser Leitfaden behandelt die verschiedenen Optionen im Zusammenhang mit **Aktivierungsschlüsseln, Lizenzierung, Online-Validierung und Deaktivierung** für Ihre geschützten Anwendungen.



Grundkonzepte

Aktivierungsschlüssel aktivieren

Um zu verlangen, dass Benutzer einen Schlüssel zur Nutzung Ihrer Anwendung besitzen, aktivieren Sie die Option: **"End users must enter an activation key in order to use the protected workbook"** (Endbenutzer müssen einen Aktivierungsschlüssel eingeben, um die geschützte Arbeitsmappe zu nutzen). Dies ist der zentrale Schalter, um die Lizenzierungsfunktionen einzuschalten.

Application Master Key

Jede geschützte Arbeitsmappe besitzt einen eindeutigen **Application Master Key**. XLS Padlock verwendet diesen Schlüssel, um die zu Ihrer Arbeitsmappe gehörenden Aktivierungsschlüssel zu generieren. Er ist **streng vertraulich** und darf niemals weitergegeben werden. XLS Padlock erstellt für jedes Projekt automatisch einen neuen Master Key.

Application GUID und Secret Key

- Die **Application GUID** ist ein eindeutiger Bezeichner, mit dem die Anwendung ihre Einstellungen speichert und Speicherdateien auf dem Computer eines Benutzers verwaltet.
- Der **Application Secret Key** wird verwendet, um die Speicherdateien des Benutzers (`.XLSC` oder `.XLSCE`) zu verschlüsseln und zu sichern. Er stellt sicher, dass mit Ihrer Anwendung erstellte Speicherdateien nur von Ihrer Anwendung geöffnet werden können.

WICHTIG

Wenn Sie die Application GUID oder den Secret Key ändern, können Ihre Benutzer nicht mehr auf zuvor erstellte Speicherdateien zugreifen.

Eine Aktualisierung veröffentlichen, die alte Benutzer-Speicherdateien ungültig machen soll

Um zu erzwingen, dass eine neue Version alle mit früheren Versionen erstellten Speicherdateien ignoriert, beispielsweise wenn sich die Struktur der Arbeitsmappe so stark geändert hat, dass alte Speicherdateien nicht erneut geladen werden sollen, generieren Sie vor dem erneuten Kompilieren einen **neuen Application Secret Key**, lassen aber die **Application GUID unverändert**, damit vorhandene Aktivierungsschlüssel gültig bleiben. Beim ersten Start durch den Benutzer wird die alte Speicherdatei als inkompatibel erkannt, und er wird aufgefordert, stattdessen die ursprünglich eingebettete Arbeitsmappe zu laden. Den vollständigen Aktualisierungsablauf finden Sie unter Arbeitsmappen-Aktualisierungen.

TIPP

Der Aktivierungsstatus (Testversion / registriert) kann über VBA-Code abgerufen werden.

Endbenutzer das Ändern des Aktivierungsschlüssels erlauben

Ein Benutzer kann seinen Aktivierungsschlüssel ändern, indem er die Anwendung mit dem Befehlszeilenschalter `-enterkey` ausführt (z. B. `MYAPP.EXE -enterkey`). Dies ist nützlich, um einen Schlüssel zu ersetzen, der bald abläuft.

Aktivierungsdaten löschen

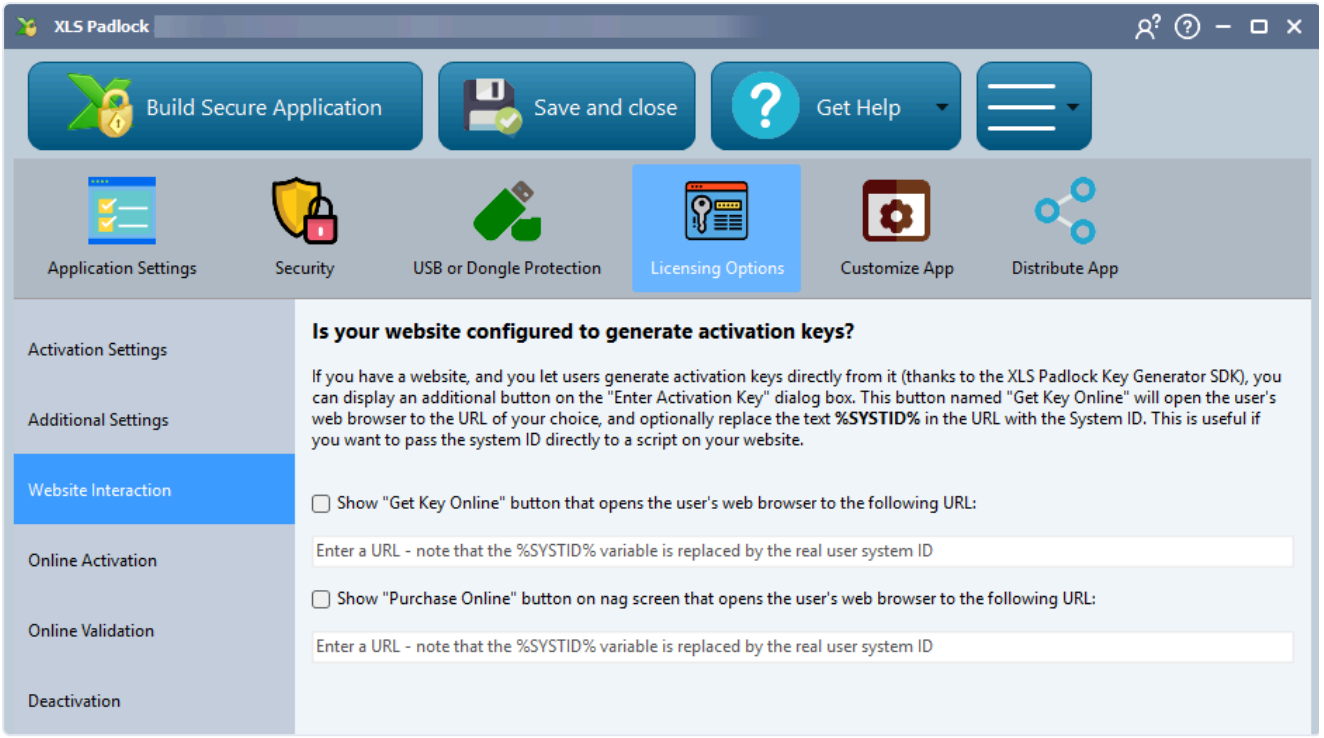
Zu Testzwecken auf Ihrem eigenen Computer setzt die Schaltfläche "**Clear Activation Data**" (Aktivierungsdaten löschen) im Key Generator alle lokal gespeicherten Aktivierungsinformationen zurück.

Hardwaregebundene Schlüssel

Ein hardwaregebundener Schlüssel funktioniert **nur auf dem bestimmten Computer**, für den er generiert wurde, wodurch er bei Weitergabe nutzlos wird. Der Schlüssel basiert auf einer **eindeutigen System ID**, die aus der Hardware des Computers des Benutzers abgeleitet wird.

- **Optionen für die System ID:** Sie können auswählen, welche Hardwarekomponenten (CPU, MAC-Adresse usw.) zur Generierung der System ID verwendet werden. Halten Sie diese Optionen nach der Bereitstellung Ihrer Anwendung konsistent.
- **System ID erhalten:** Wenn ein Benutzer die Anwendung ausführt, wird ihm seine System ID angezeigt, die er Ihnen senden muss. Sie verwenden diese ID dann im Key Generator, um seinen Schlüssel zu erstellen.
- **Online-Automatisierung:** Um den manuellen Austausch von System IDs zu vermeiden, können Sie die Funktionen der Online-Aktivierung verwenden.

Interaktion mit der Website



Schaltfläche "Get Key Online"

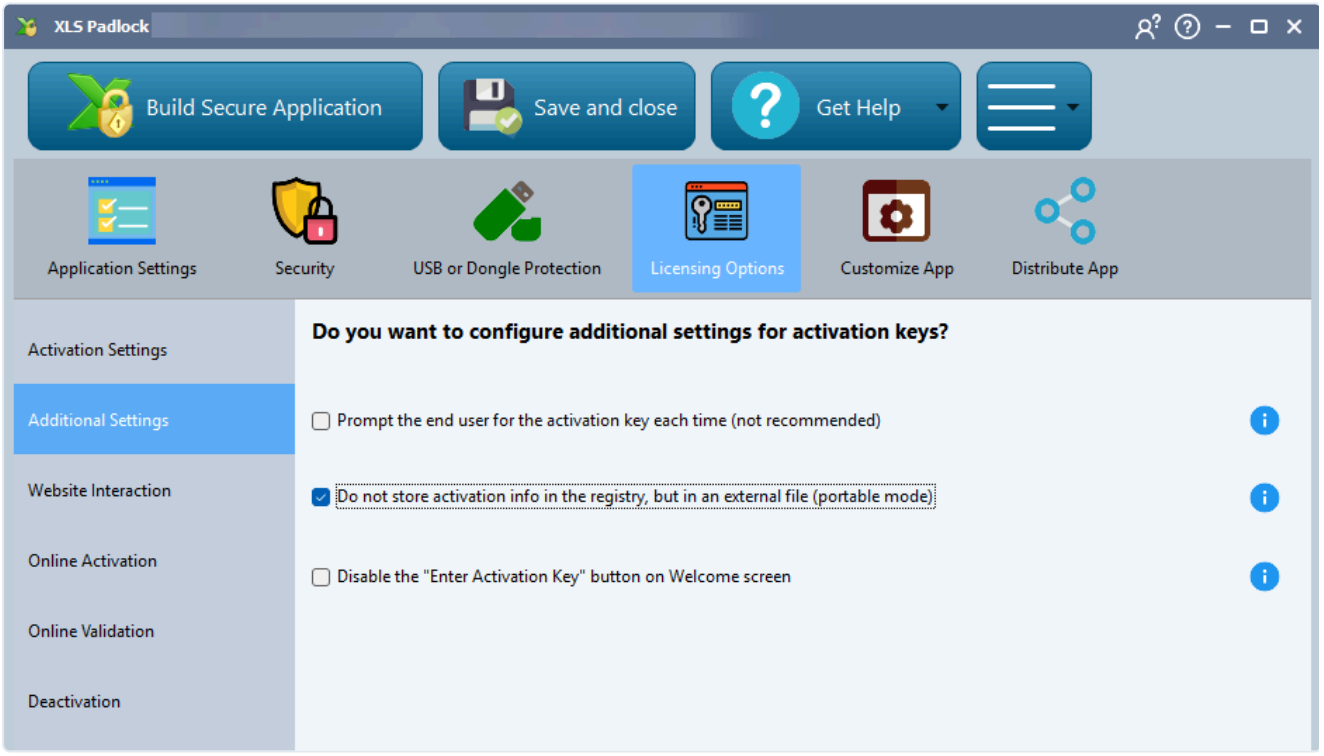
Sie können auf dem Aktivierungsfenster eine Schaltfläche "Get Key Online" (Schlüssel online abrufen) anzeigen. Diese Schaltfläche öffnet den Webbrowser des Benutzers an einer von Ihnen angegebenen URL. Sie können den Platzhalter `%SYSTID%` in der URL verwenden, um die System ID des Benutzers automatisch an das Skript zur Schlüsselgenerierung Ihrer Website zu übergeben.

Beispiel-URL: `https://www.yourwebsite.com/getkey.php?systid=%SYSTID%`

Schaltfläche "Purchase Online" anzeigen

Für Testversionen mit einem Erinnerungsbildschirm (Nag Screen) können Sie eine Schaltfläche "Purchase Online" (Online kaufen) hinzufügen, die den Benutzer zu Ihrem Shop oder Ihrer Kaufseite leitet.

Zusätzliche Einstellungen



Portabler Modus

Aktivieren Sie die Option "**Do not store activation info in the registry, but in an external file (portable mode)**" (Aktivierungsinformationen nicht in der Registrierung, sondern in einer externen Datei speichern, portabler Modus). Dadurch wird eine versteckte `.lic`-Datei im selben Ordner wie Ihre EXE erstellt, sodass die Anwendung von einem USB-Stick ausgeführt werden kann, ohne in die Windows-Registrierung zu schreiben.

> **WICHTIG**
>
> Dieser Modus wird für Testversionen nicht empfohlen, da das Löschen der `.lic`-Datei den Testzeitraum zurücksetzt.

Den Endbenutzer jedes Mal nach dem Aktivierungsschlüssel fragen

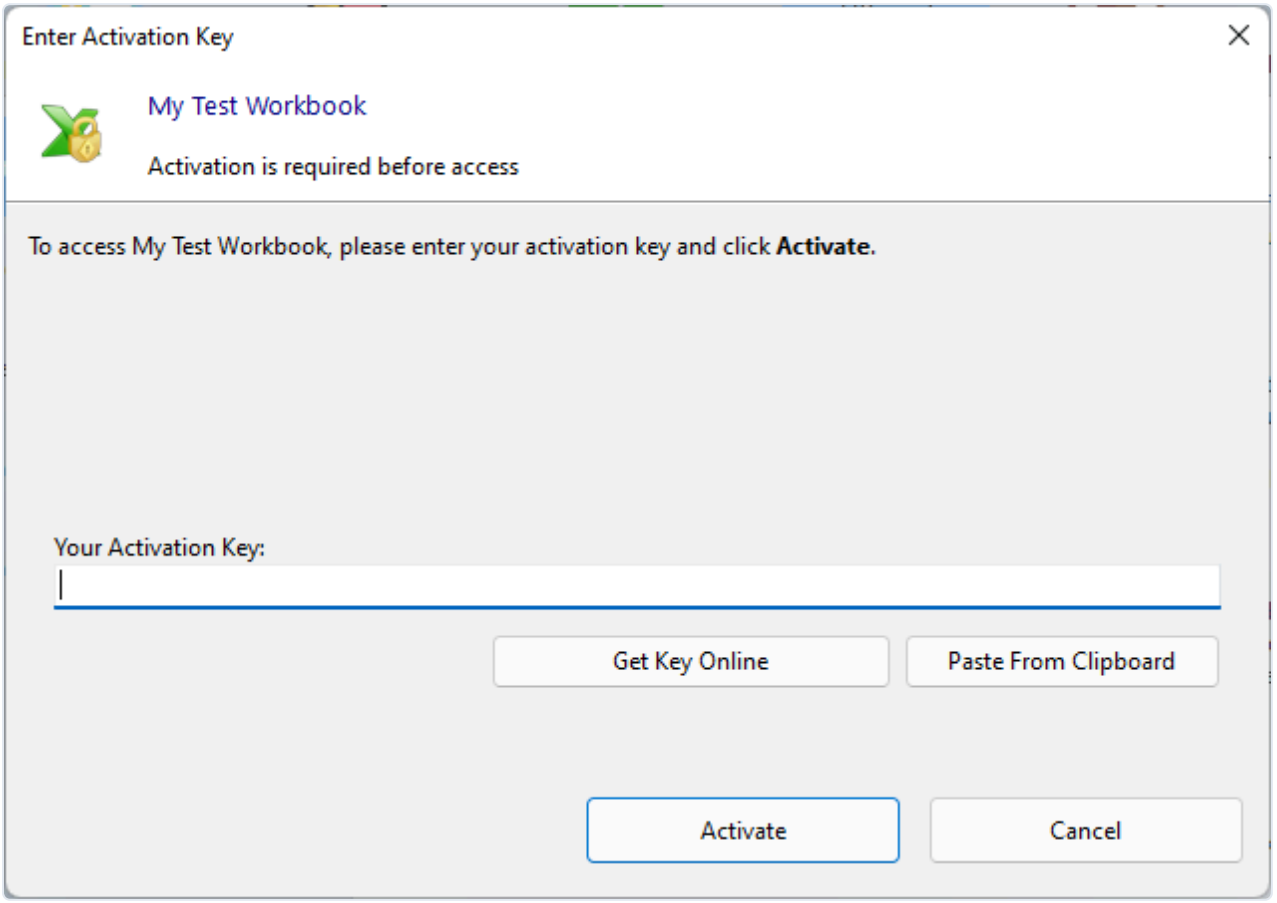
Wenn Sie "Prompt the end user for the activation key each time" (den Endbenutzer jedes Mal nach dem Aktivierungsschlüssel fragen) aktivieren, müssen Ihre Benutzer den Aktivierungsschlüssel bei jedem Öffnen Ihrer Anwendung eingeben. Dies wird nicht empfohlen, da es für Ihre Kunden mühsam ist und Sie keine Ablauffunktionen für den Schlüssel festlegen können.

Die Schaltfläche "Enter Activation Key" auf dem Willkommensbildschirm deaktivieren

Dadurch wird Ihren Endbenutzern die Möglichkeit genommen, beim Öffnen Ihrer Excel-Arbeitsmappen-Anwendung einen Aktivierungsschlüssel einzugeben. Es wird jedoch nicht empfohlen, diese Option zu verwenden.

Aktivierungsschlüssel

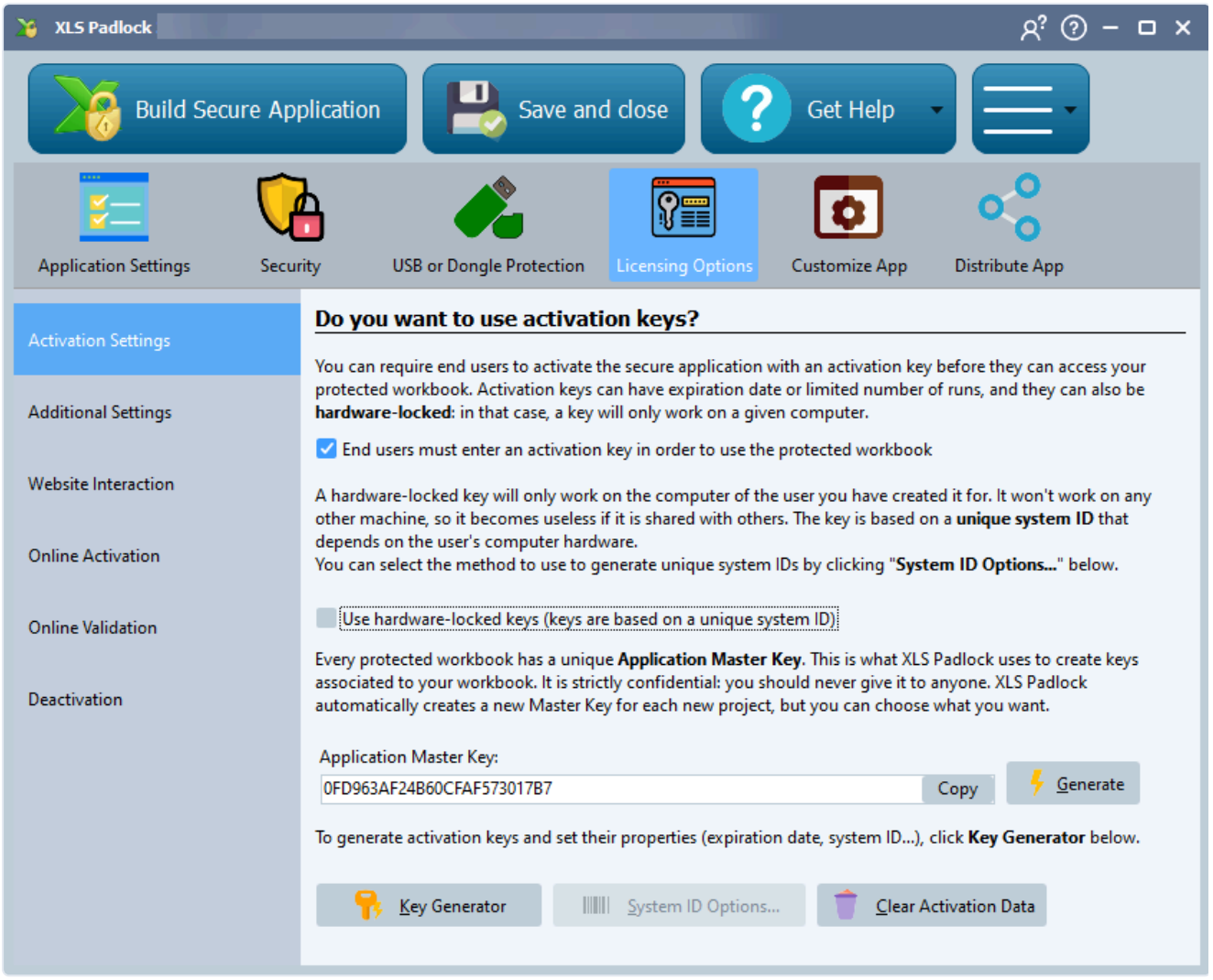
Aktivierungsschlüssel sind eine zentrale Funktion von XLS Padlock und bieten ein **leistungsstarkes Lizenzsystem für Ihre Excel-Arbeitsmappen**. Mit diesem System können Sie verlangen, dass Endbenutzer die Anwendung mit einem Schlüssel aktivieren, bevor sie auf Ihre geschützte Arbeitsmappe zugreifen können:



👉 Aktivierungsschlüssel können mit einem Ablaufdatum oder einer begrenzten Anzahl von Ausführungen konfiguriert werden. Sie können auch hardwaregebunden werden, was bedeutet, dass ein Schlüssel nur auf einem bestimmten Computer funktioniert.

> **TIPP**
>
> Zusätzlich zum integrierten Key Generator bieten wir einen eigenständigen Schlüsselgenerator an, der Tausende von Schlüsseln auf einmal erstellen kann, sowie ein Key-Generator-SDK zur Integration mit Ihrem Server oder Ihrer Website.

Schließlich können Sie Funktionen für die Online-Aktivierung und Validierung implementieren, um aus der Ferne zu steuern, wie Benutzer auf Ihre Arbeitsmappe zugreifen, und die Bereitstellung von Aktivierungsschlüsseln zu automatisieren.



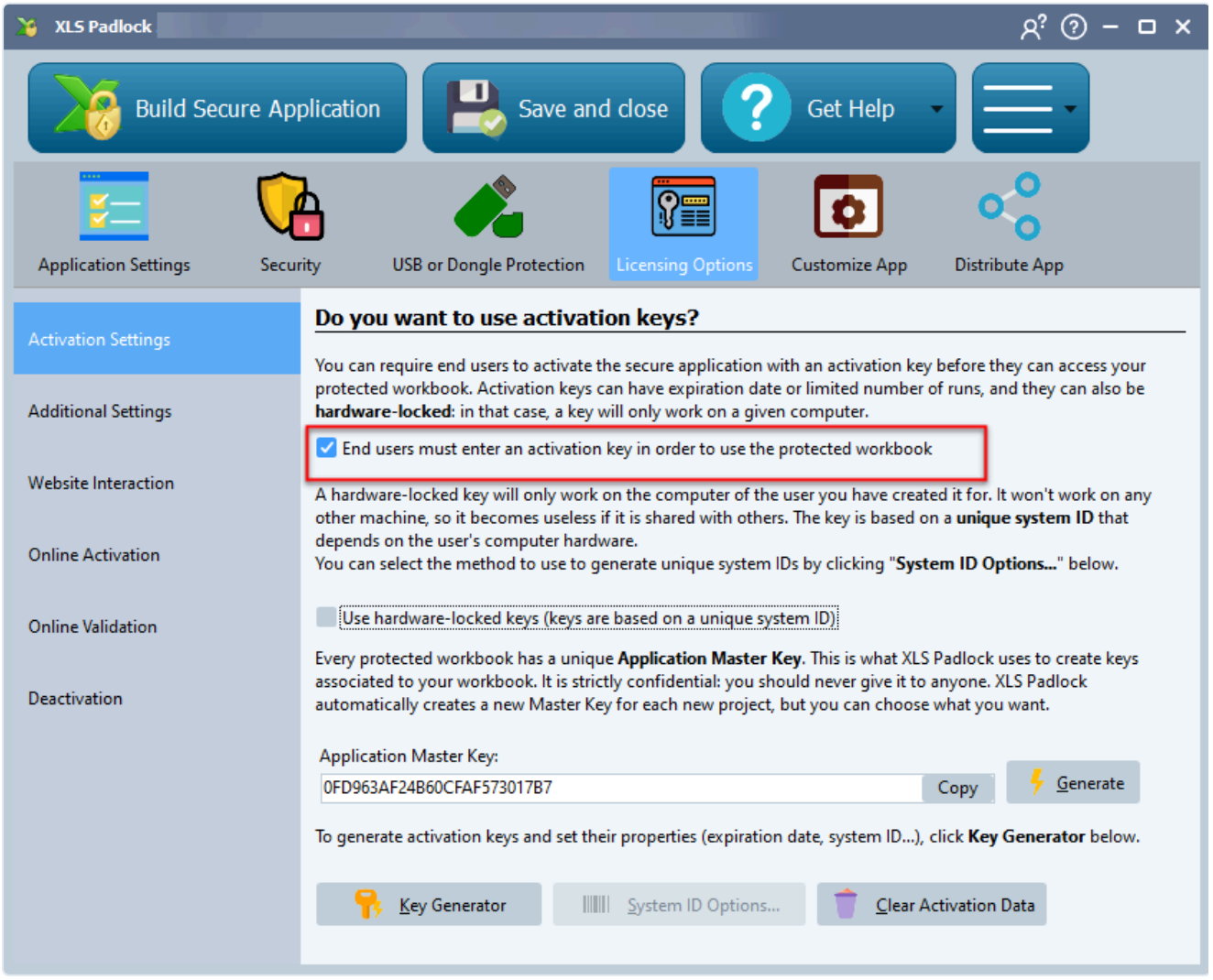
👉 Um zu beginnen, lesen Sie bitte unser Tutorial dazu, wie Sie Aktivierungsschlüssel einrichten.

Aktivierungsschlüssel einrichten

Aktivierungsschlüssel sind ein leistungsfähiges Lizenzsystem, mit dem Sie steuern können, wer auf Ihre geschützten Arbeitsmappen zugreifen darf. Diese Anleitung zeigt Ihnen, wie Sie damit beginnen.

1. Aktivierungsschlüssel einschalten

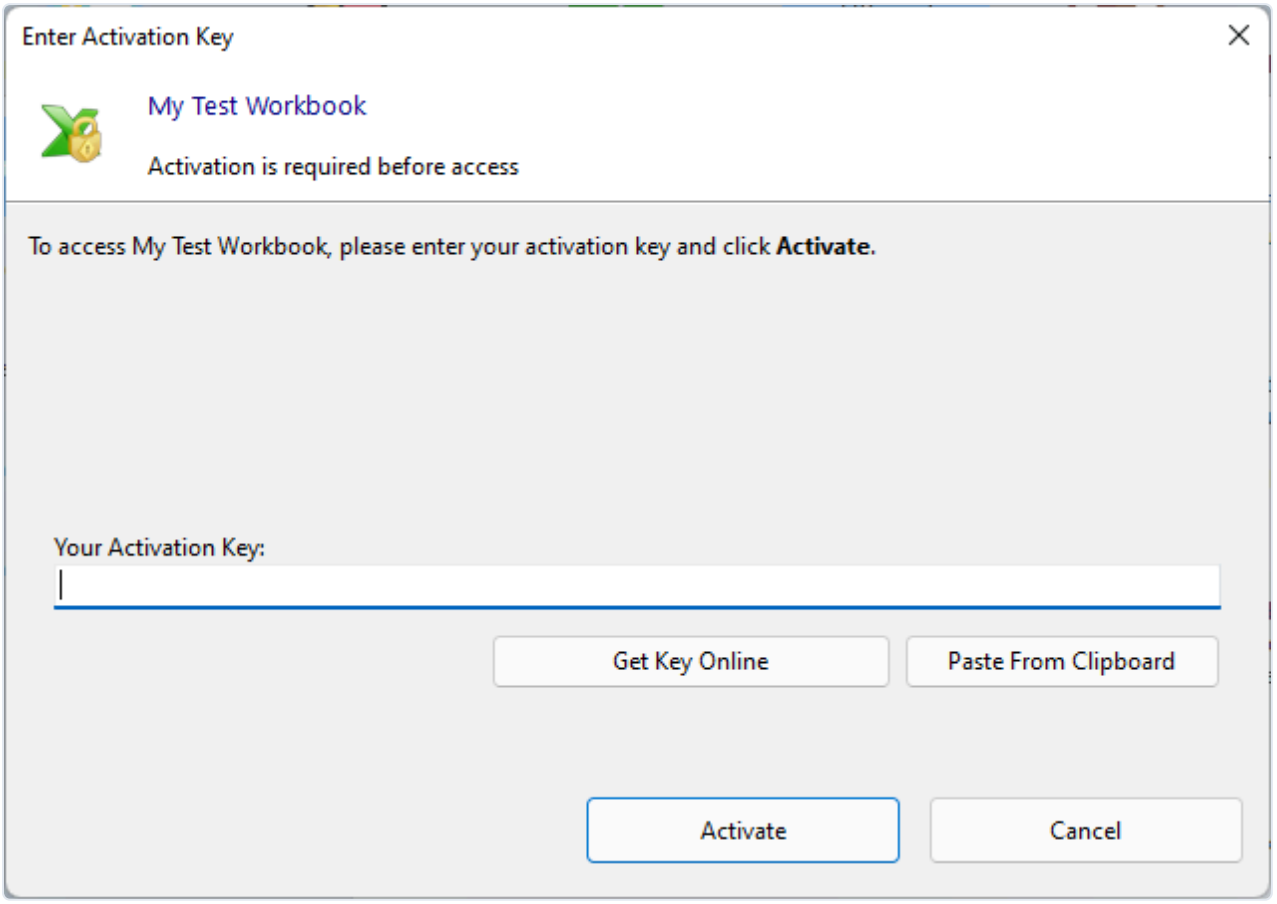
Aktivieren Sie in den XLS Padlock Sicherheitseinstellungen das Kontrollkästchen für **"End users must enter an activation key in order to use the protected workbook"** (Endanwender müssen einen Aktivierungsschlüssel eingeben, um die geschützte Arbeitsmappe verwenden zu können).



Sobald dies erledigt ist, erstellen Sie Ihre Anwendung neu.

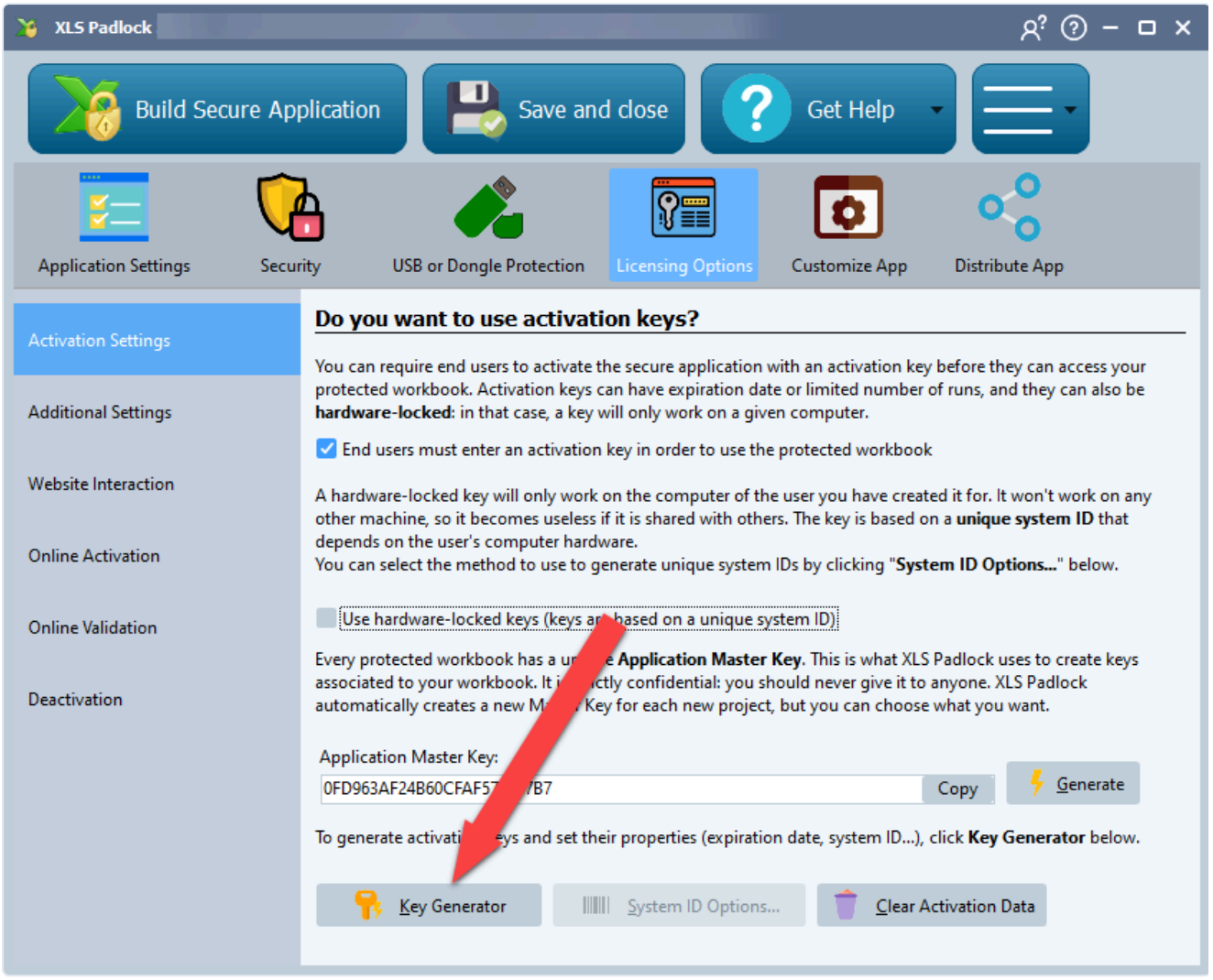
2. Ihre Anwendung testen

Wenn Sie die neu kompilierte Anwendung ausführen, werden Sie nun aufgefordert, einen Aktivierungsschlüssel einzugeben, bevor Sie auf die Arbeitsmappe zugreifen können:



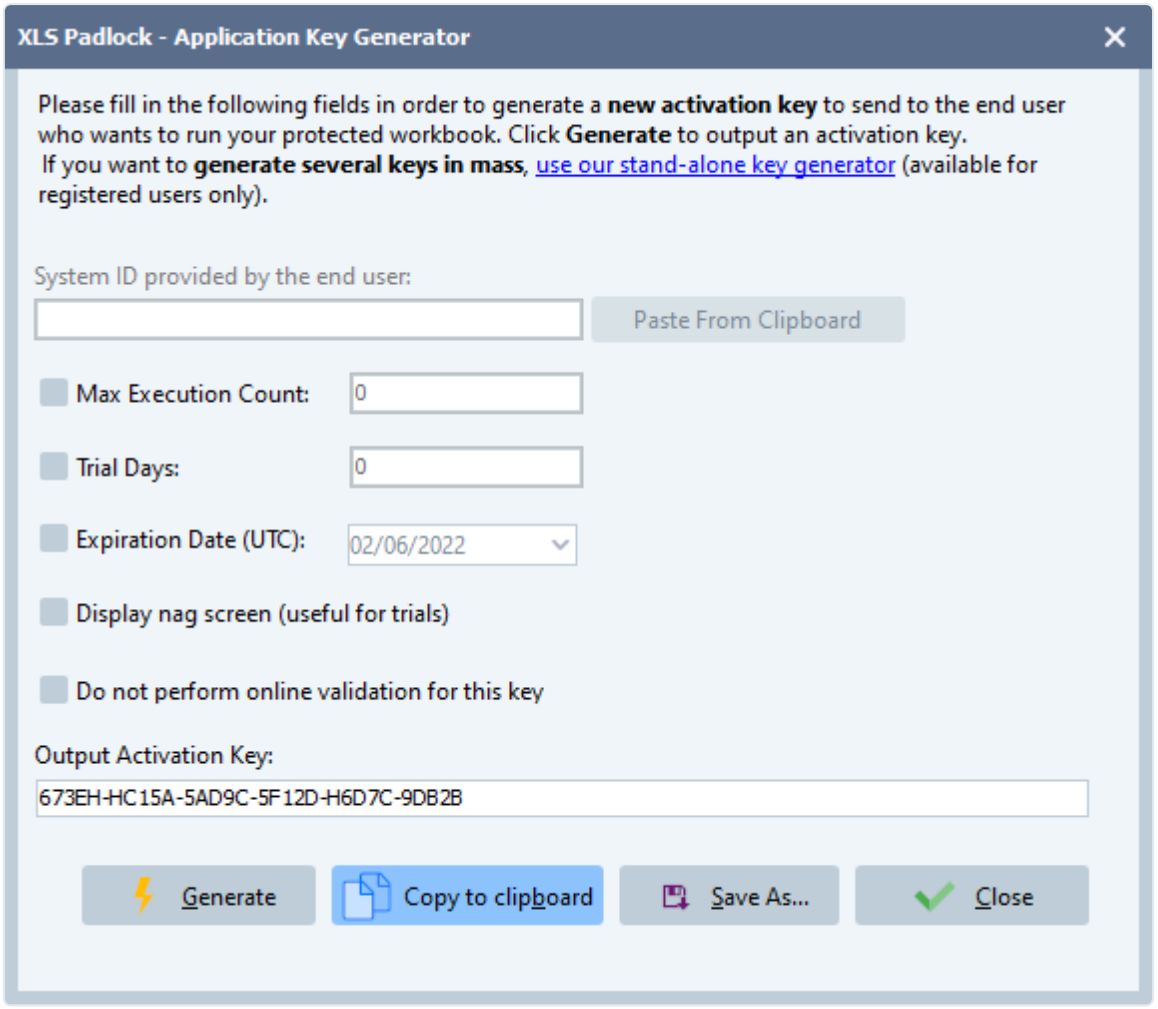
3. Aktivierungsschlüssel erzeugen

Um Aktivierungsschlüssel zu erstellen, öffnen Sie den Key Generator in XLS Padlock.

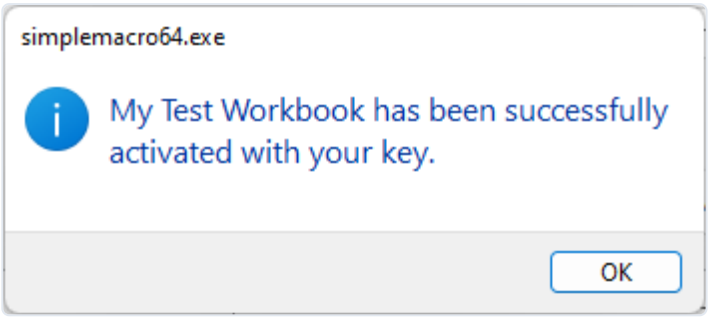


Falls Sie möchten, können Sie Einschränkungen für den Schlüssel festlegen, etwa die Anzahl der Ausführungen begrenzen oder ein Ablaufdatum setzen.

Klicken Sie auf die Schaltfläche **Generate** (Erzeugen), um sofort einen Schlüssel zu erstellen. Sie können diesen Schlüssel anschließend in die Zwischenablage kopieren, um ihn an Ihren Kunden zu senden.



Wenn Ihr Kunde den Schlüssel eingibt und auf **Activate** (Aktivieren) klickt, erhält er Zugriff auf die Arbeitsmappe. Die Anwendung fragt den Schlüssel nicht erneut ab, es sei denn, er läuft ab.



Schlüsselerzeugung automatisieren

Für eine einfachere Verteilung der Schlüssel steht registrierten Kunden eine eigenständige Anwendung zur Schlüsselerzeugung zur Verfügung. Wir stellen außerdem kostenlose Webanwendungen bereit, mit denen Sie Schlüssel automatisch über Ihre Website erzeugen können.

Hardwaregebundene Aktivierungsschlüssel

Hardwaregebundene Aktivierungsschlüssel sind ein wirksames Mittel, um das beiläufige Weitergeben Ihrer geschützten Arbeitsmappe zu verhindern, indem jede Lizenz an einen bestimmten Computer gebunden wird. Diese Anleitung zeigt Ihnen, wie Sie sie einrichten.

Video-Tutorial verfügbar

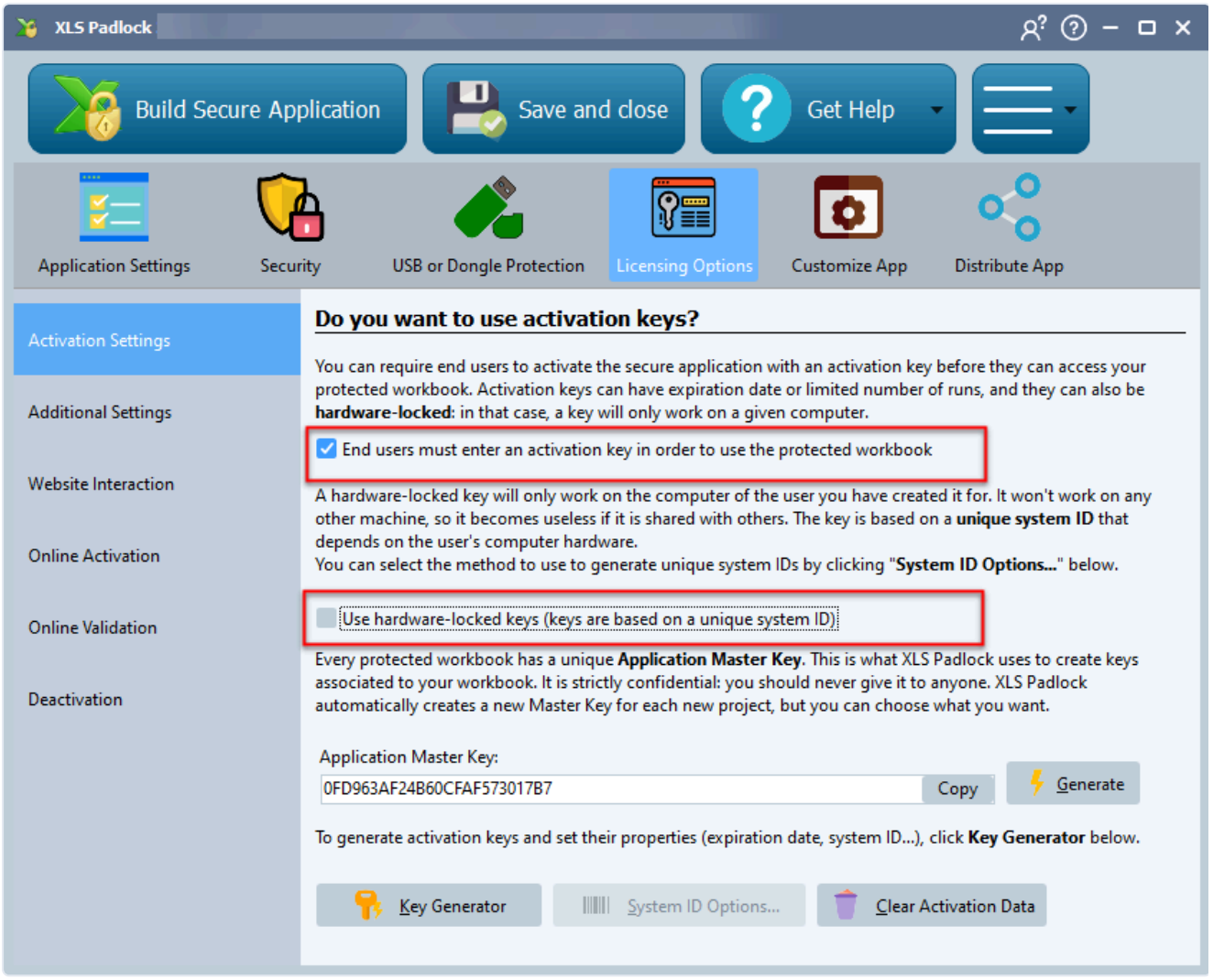
- Sehen Sie sich unser Video-Tutorial zur Hardwarebindung Ihrer Excel-Tabellenkalkulationen an
- Sehen Sie eine Live-Demonstration dieser Funktion

Über hardwaregebundene Schlüssel

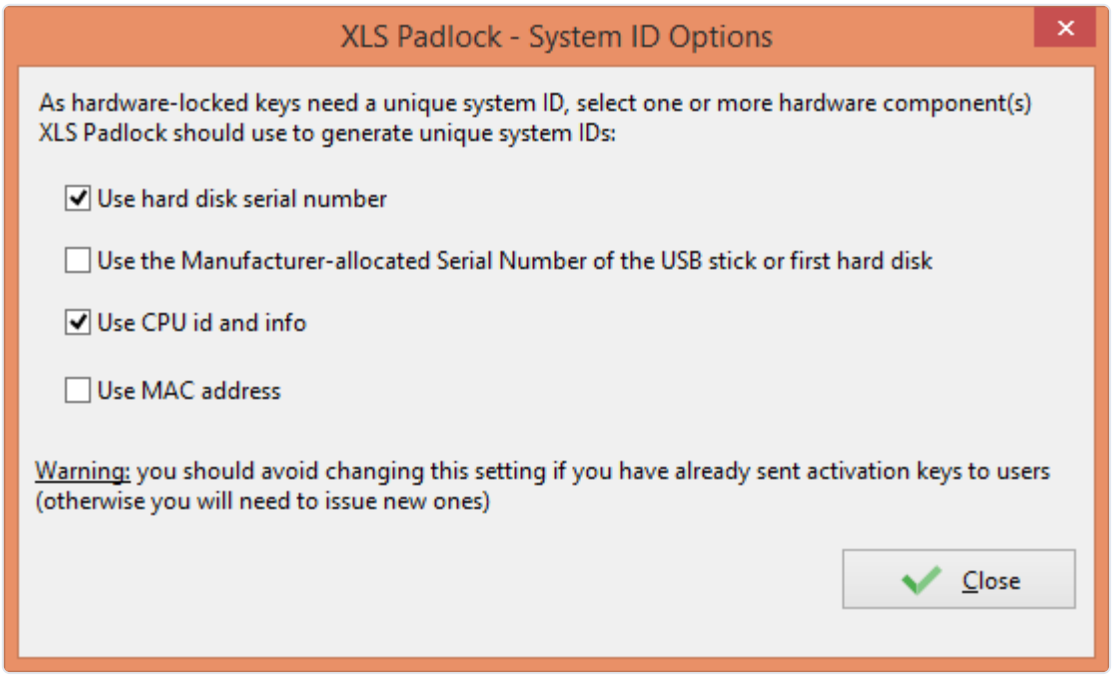
Ein hardwaregebundener Schlüssel funktioniert **nur auf dem Computer, für den er erzeugt wurde**. Er basiert auf einer eindeutigen **System ID**, die aus der Computerhardware des Anwenders abgeleitet wird, sodass der Schlüssel nutzlos ist, wenn er an andere weitergegeben wird.

1. Hardwaregebundene Schlüssel einschalten

Aktivieren Sie in den XLS Padlock Einstellungen sowohl "End users must enter an activation key..." als auch "**Use hardware-locked keys**" (hardwaregebundene Schlüssel verwenden):



Sie können optional festlegen, welche Hardwarekomponenten zur Erzeugung der System ID verwendet werden, indem Sie auf "**System ID Options...**" klicken. Es ist wichtig, diese Optionen nicht mehr zu ändern, nachdem Sie mit der Verteilung Ihrer Anwendung begonnen haben.



> **Hinweise zu den System-ID-Komponenten**
>
> - **CPU:** Wenn die CPU keine integrierte Serien-ID besitzt, werden ihre allgemeinen Informationen verwendet. Bei identischer Hardware könnte dies zu derselben ID führen. Es ist besser, mehrere Hardwareoptionen zu kombinieren.
> - **MAC Address:** Wenn Anwender mehrere Möglichkeiten haben, sich mit dem Internet zu verbinden (Wi-Fi, 4G usw.), kann sich ihre MAC-Adresse ändern, was ihren Schlüssel ungültig machen würde.

Erweiterten Hardware-Fingerabdruck verwenden

Ab XLS Padlock 2026.0 zeigt der System ID Options Dialog außerdem ein zusätzliches Kontrollkästchen an: **"Use enhanced hardware fingerprint"** (erweiterten Hardware-Fingerabdruck verwenden).

Wenn diese Option aktiviert ist, verwendet die geschützte EXE eine verstärkte Berechnung der System ID:

- Die Windows-Bindungsquelle liest die eindeutige Windows-Installationskennung direkt aus der Systemregistrierung anstelle des Datenträger-Volume-Scans, den der frühere Algorithmus verwendet. Diese Kennung bleibt über Volume-Neuformatierungen und physische Datenträgerwechsel hinweg stabil. Nur eine vollständige Windows-Neuinstallation erzeugt sie neu.
- Alle Hardwarequellen werden normalisiert (von Leerzeichen befreit, Groß-/Kleinschreibung vereinheitlicht) und innerhalb der Fingerabdruck-Berechnung klar abgegrenzt, wodurch eine Klasse struktureller Kollisionen beseitigt wird, bei der zwei unterschiedliche Maschinen gelegentlich auf derselben System ID landen konnten. Die neue Logik schützt außerdem vor Falschmeldungen, wenn eine Quelle vorübergehend nicht verfügbar ist (zum Beispiel, wenn ein Antivirenprogramm WMI-Aufrufe blockiert), indem die System ID stabil gehalten wird, statt sie zu verändern.
- Der zugrunde liegende Hash wird von dem älteren MD5 auf SHA-256 angehoben, wodurch ein veralteter Algorithmus aus dem kryptografischen Profil der geschützten EXE entfernt wird.

Das angezeigte Format der System ID bleibt unverändert, weiterhin 14 hexadezimale Zeichen in der gewohnten Form `XXXX-XXXX-XXXX`, sodass Ihre Endanwender und Ihr Ablauf zur Schlüsselerzeugung keinen Unterschied bemerken.

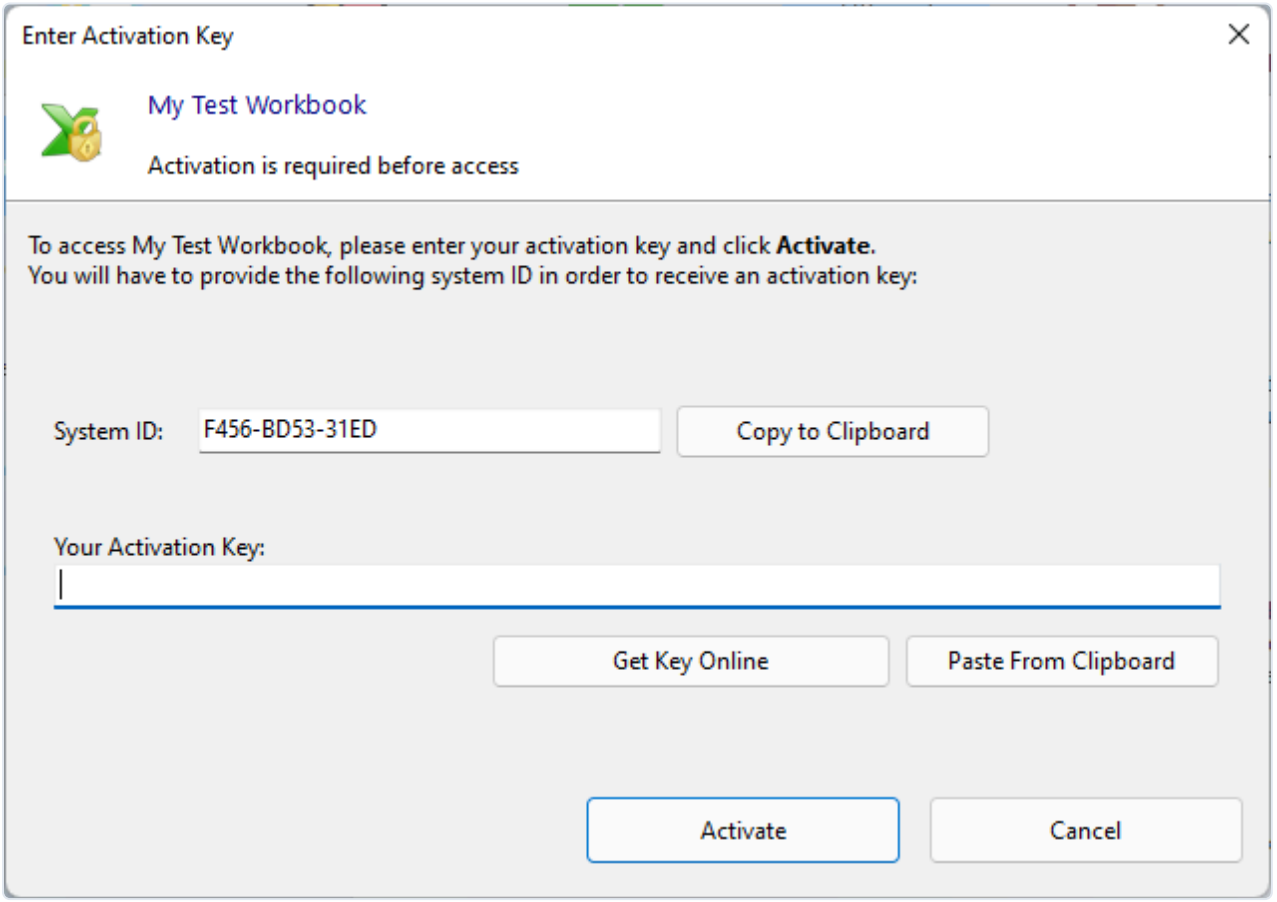
Standardverhalten:

- **Bestehende Projekte** (mit XLS Padlock 2025.3 oder früher gespeichert): Die Option ist standardmäßig **OFF** (aus). Dadurch bleibt jeder bereits im Umlauf befindliche Aktivierungsschlüssel erhalten: Die System IDs, die Ihre Kunden aus älteren Builds erhalten haben, passen weiterhin zu den von Ihnen ausgestellten Schlüsseln.
- **Neue Projekte** (mit XLS Padlock 2026.0 und später erstellt): Die Option ist standardmäßig **ON** (ein).

> ⚠️ **Wichtig:** Das Ändern dieser Option bei einem bereits verteilten Projekt rotiert die System ID für jeden Endanwender und macht alle bisher von Ihnen ausgestellten Aktivierungsschlüssel ungültig. Aktivieren Sie sie nur bei einem neuen Projekt oder stimmen Sie die Änderung mit einer Neuausstellung des Schlüssels jedes Kunden ab.

2. Wie Anwender ihre System ID erhalten

Wenn ein Anwender Ihre Anwendung zum ersten Mal ausführt, wird er aufgefordert, einen Aktivierungsschlüssel einzugeben. Seine eindeutige System ID wird in diesem Fenster angezeigt. Der Anwender muss diese ID kopieren und Ihnen zusenden.

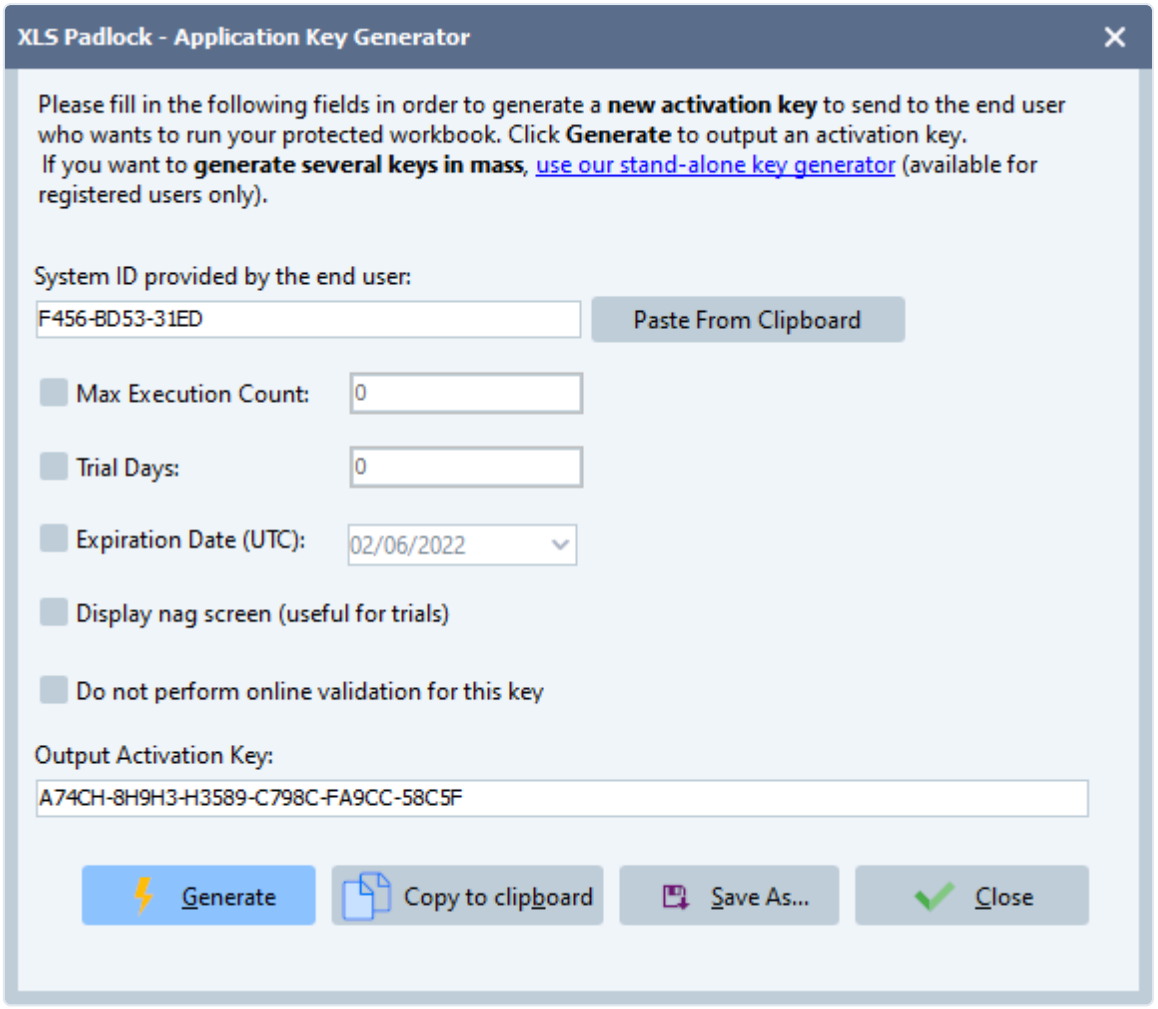


Schlüsselzustellung automatisieren

Um das manuelle Austauschen von System IDs zu vermeiden, können Sie eine Schaltfläche "Get Key Online" konfigurieren, die den Anwender mit seiner System ID auf Ihre Website weiterleitet, oder die vollständig automatisierte Funktion Online-Aktivierung verwenden.

3\. Den Aktivierungsschlüssel erzeugen

Öffnen Sie den Key Generator in XLS Padlock. Fügen Sie die System ID des Kunden in das dafür vorgesehene Feld ein und klicken Sie auf **Generate** (Erzeugen). Sie können den Schlüssel anschließend kopieren und an Ihren Kunden senden.



Wenn der Kunde den Schlüssel eingibt, wird die Anwendung aktiviert und fragt den Schlüssel nicht erneut ab (es sei denn, er hat ein Ablaufdatum). Da der Schlüssel an seine System ID gebunden ist, funktioniert er auf keinem anderen Computer.

👉 Möchten Sie die Erstellung hardwaregebundener Aktivierungsschlüssel automatisieren? Mehr über die Online-Aktivierung erfahren.

Online-Aktivierung

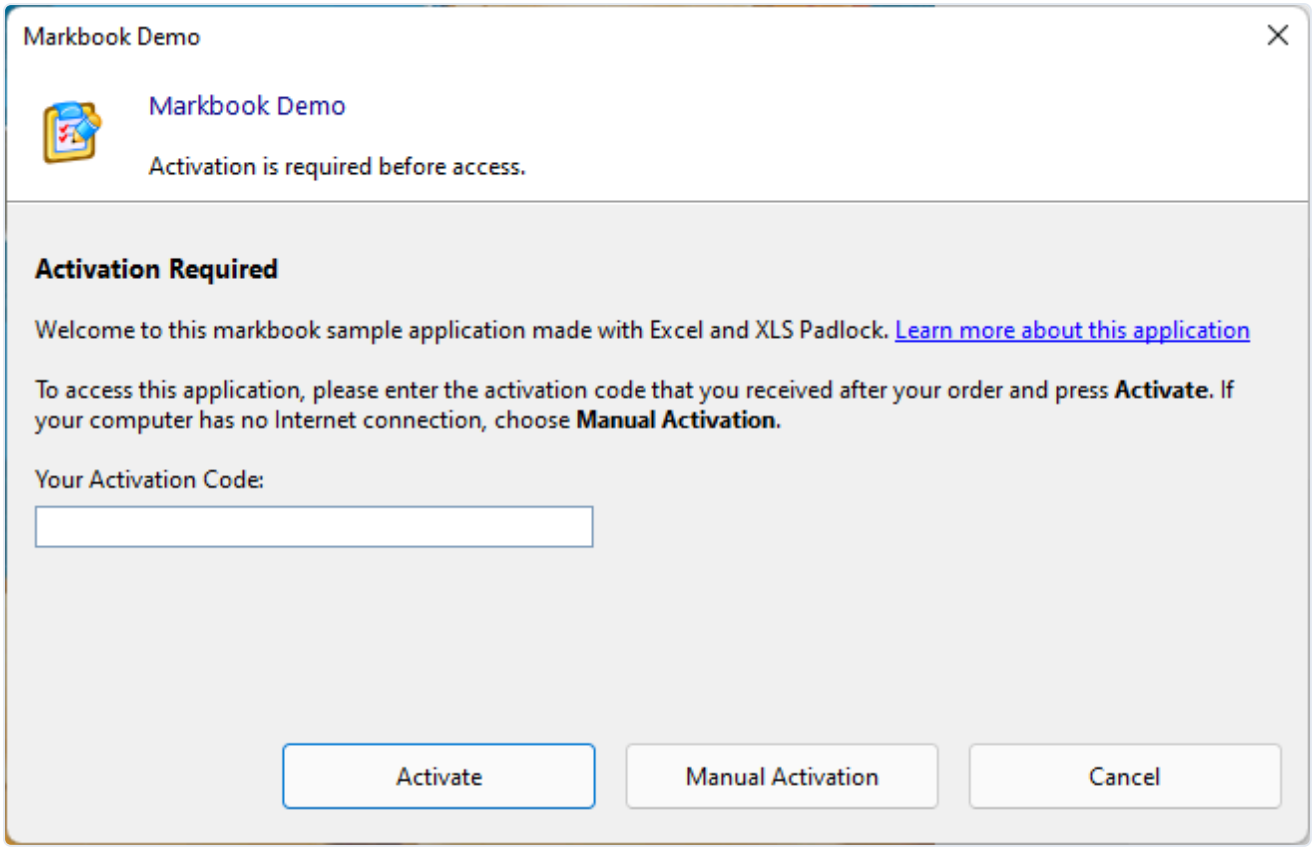
So funktioniert es

Die Online-Aktivierung automatisiert den Abruf von Aktivierungsschlüsseln über das Internet. Die geschützte Anwendung kommuniziert mit Ihrem Webserver, um einen Aktivierungsschlüssel direkt herunterzuladen, sodass Benutzer ihn nicht mehr manuell eingeben müssen.

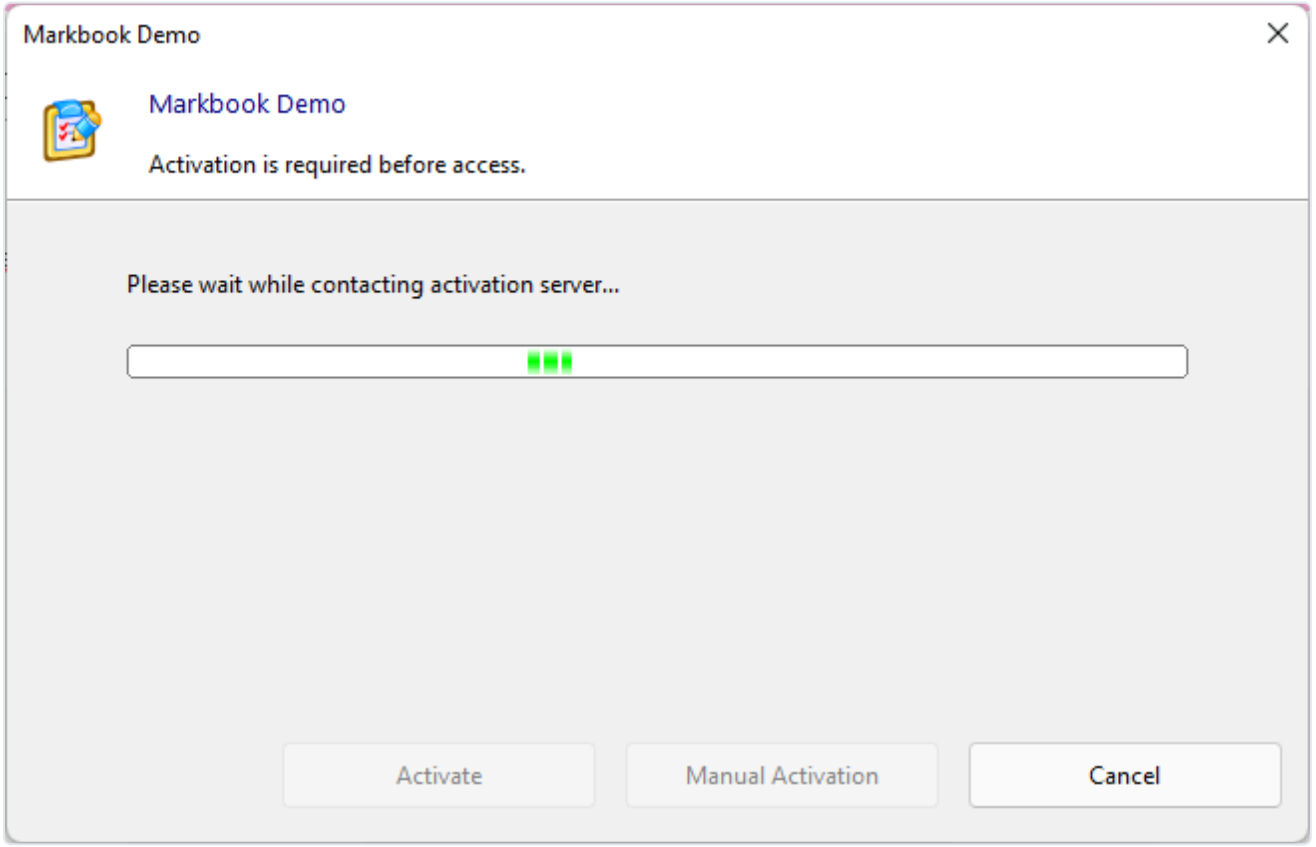
> **Serverseitiges Kit erforderlich**
>
> Um die Online-Aktivierung zu nutzen, müssen Sie eines unserer serverseitigen Kits auf Ihrem Webserver installieren, etwa das XLS Padlock Activation Kit, das WooCommerce Integration Kit oder das FastSpring Subscription kit. Sie können diese Kits von Ihrer Kontoseite herunterladen.

Wenn Sie sich für das Aktivierungsprotokoll 2026 entscheiden (Ed25519-Signatur der Antworten, siehe unten), stellen Sie sicher, dass Ihr Kit die Version 2026.0 oder höher ist. Das Kit erkennt das Protokoll bei jeder Anfrage automatisch, sodass ältere Arbeitsmappen mit derselben Kit-Installation weiterhin funktionieren.

👉 Wenn ein Endbenutzer die Anwendung startet, erscheint ein Dialogfeld, das ihn zur Aktivierung auffordert. Dieses Dialogfeld ersetzt das standardmäßige Dialogfeld "Enter Activation Key" und ist vollständig anpassbar. Sie können eigene Felder hinzufügen, um Daten vom Benutzer zu erfassen (etwa eine Bestellnummer oder eine E-Mail-Adresse), die an Ihren Webserver gesendet werden. Ihr Server überprüft diese Informationen anschließend und sendet bei Erfolg einen Aktivierungsschlüssel an die Anwendung zurück.



Wenn der Benutzer auf Activate klickt, werden die Daten an den Aktivierungsserver gesendet:



Nach einer erfolgreichen Aktivierung wird eine Bestätigungsmeldung angezeigt und die Anwendung startet neu. Tritt ein Fehler auf, erscheint ein Meldungsfeld, das es dem Benutzer ermöglicht, es erneut zu

versuchen.

Konfiguration

👉 Um die Online-Aktivierung zu aktivieren, müssen Sie die folgenden Optionen konfigurieren:

Base Activation URL

Geben Sie die vollständige URL zum auf Ihrem Server installierten Aktivierungs-Kit an. Wenn Sie das Kit beispielsweise in einem Unterordner namens „activation“ installiert haben, lautet die URL `https://www.yourdomain.com/activation/getactivation/`.

> **HTTPS verwenden**
>
> Sichere Verbindungen über TLS/SSL werden unterstützt. Sie sollten immer URLs verwenden, die mit `https://` beginnen.

⚠️ Lassen Sie das Feld leer, wenn Sie die Online-Aktivierung **nicht** verwenden möchten.

Kundenidentifikation: Sicherheitsschlüssel oder Ed25519-Schlüsselpaar

Ab XLS Padlock 2026.0 können Sie über die Option **Compatibility mode for pre-2026 activation kits** (Kompatibilitätsmodus für Aktivierungs-Kits vor 2026) zwischen zwei Aktivierungsprotokoll-Verfahren wählen:

- **Compatibility mode aktiviert** (Standard, empfohlen, wenn Ihr Aktivierungsserver älter als die Version 2026 ist): Das ältere GUID-Feld **Security Private Key** identifiziert Ihre Anwendung gegenüber dem Aktivierungs-Kit. Das Protokoll ist zeilenbasiert mit nicht signierten Antworten. Dies ist das Verhalten aller XLS Padlock-Versionen vor 2026.0.
- **Compatibility mode deaktiviert** (erfordert die Version 2026 des XLS Padlock Activation Kit, des WooCommerce Integration Kit oder des FastSpring Subscription Kit): Das ältere Feld wird ausgeblendet und ein projektspezifisches **Ed25519 keypair** (Ed25519-Schlüsselpaar) übernimmt. Klicken Sie auf die Schaltfläche **Generate keypair** (Schlüsselpaar erzeugen) auf der Seite Online Activation, um ein neues Schlüsselpaar zu erstellen. Der öffentliche Schlüssel wird in die geschützte Arbeitsmappe eingebettet; der private Schlüssel wird Ihnen in einem einmalig angezeigten Dialogfeld präsentiert, sodass Sie ihn in die Konfiguration Ihres Aktivierungs-Kits einfügen können (`xlspadlocksignkey` in `config.ini`).

Wenn Sie den privaten Schlüssel verlieren, klicken Sie auf derselben Seite auf **Show config.ini snippet**, um ihn erneut anzuzeigen. Der Schlüssel wird in Ihrer `.xplp`-Projektdatei gespeichert. Behandeln Sie die `.xplp`-Datei als vertraulich: Stellen Sie sie nicht unter Versionskontrolle und senden Sie sie nicht per E-Mail an den Support.

Das Protokoll 2026 verwendet einen JSON-Anfrage-Umschlag und überprüft eine abgetrennte Ed25519-Signatur bei jeder Antwort des Aktivierungsservers. Dies schützt Ihre Kunden vor gefälschten „aktiviert“-Antworten oder Phishing-Fehlermeldungen, die ein Angreifer auf Netzwerkebene andernfalls durch Umgehung von TLS einschleusen könnte (Unternehmens-Proxy mit CA-Injektion, schädliches Antivirenprogramm, kompromittiertes Stammzertifikat).

Allow Manual Activation if No Internet Connection

Einige Benutzer verfügen möglicherweise nicht über eine aktive Internetverbindung. Um ihnen die manuelle Aktivierung zu ermöglichen, aktivieren Sie die Option "**Allow Manual Activation if No Internet Connection**".

Diese manuelle Methode funktioniert genauso wie standardmäßige Registrierungsschlüssel. In diesem Fall müssen Sie darauf vorbereitet sein, Aktivierungsanfragen von Benutzern zu bearbeiten, die offline sind.

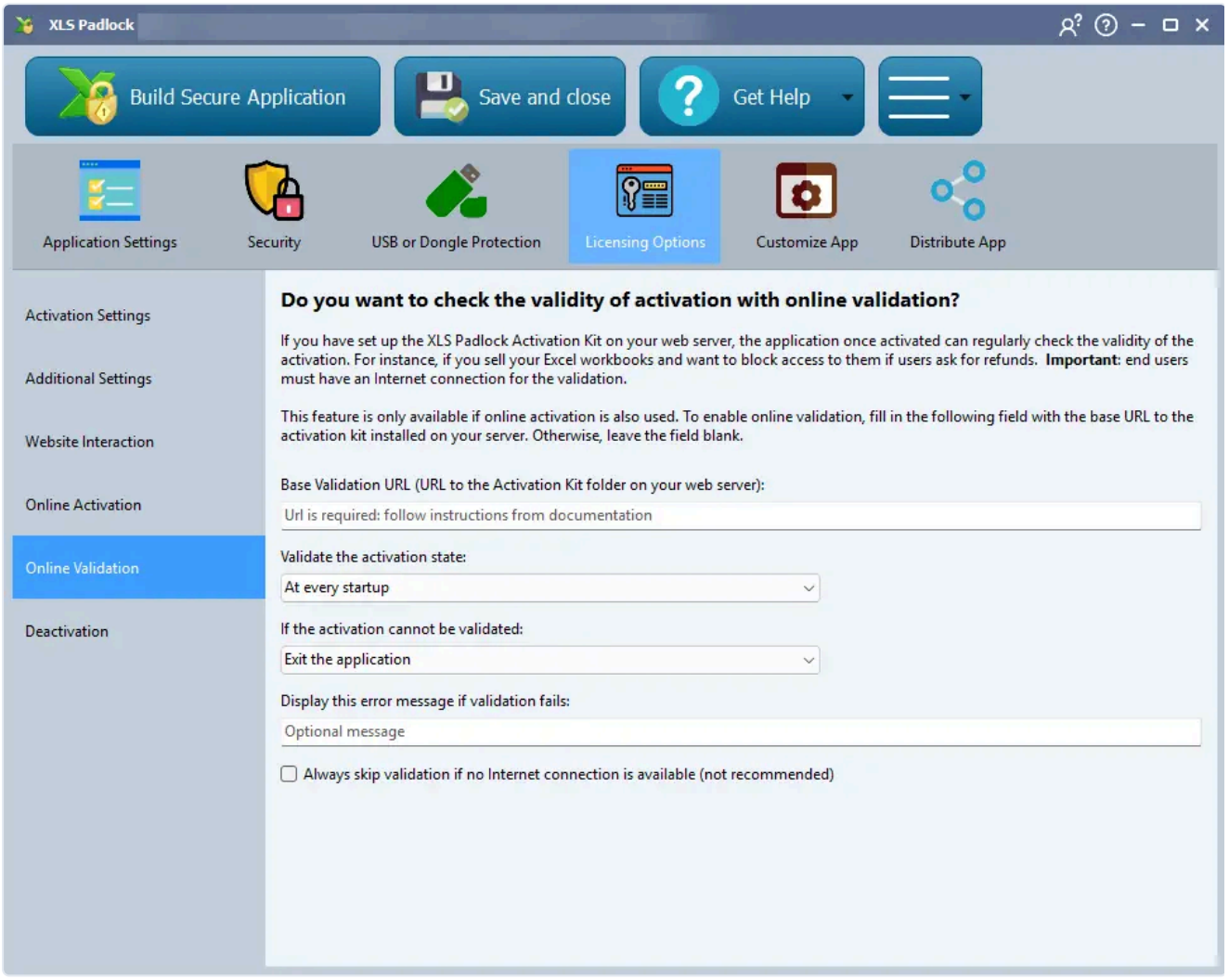
> **Warnung**
>
> Wenn Sie manuelle Aktivierungen akzeptieren, achten Sie darauf, Aktivierungsschlüssel zu erzeugen, die **keine Online-Validierung durchführen** (Option im Schlüsselgenerator). Andernfalls schlägt die Validierung fehl!

Siehe auch

- So verkaufen Sie Excel-Arbeitsmappen mit Abonnements
- Eine Schaltfläche "Purchase Online" auf dem Hinweisbildschirm anzeigen

Online-Validierung

Die Online-Validierung bietet eine **Fernsteuerung** über Ihre aktivierten Anwendungen. In Verbindung mit einem unserer serverseitigen Aktivierungs-Kits (etwa dem XLS Padlock Activation Kit, dem WooCommerce Kit oder dem FastSpring subscription kit) kann Ihre Anwendung in regelmäßigen Abständen bei Ihrem Server prüfen, ob ihre Aktivierung gültig bleibt.



> **Warnung**
>
> Diese Funktion setzt voraus, dass die Online-Aktivierung zuvor aktiviert und konfiguriert wurde.

Dies ist in mehreren Szenarien nützlich:

- Sie können eine Anwendung aus der Ferne deaktivieren, wenn ein Kunde eine Rückerstattung verlangt.
- Sie können einen abonnementbasierten Zugriff durchsetzen, bei dem die Anwendung aufhört zu funktionieren, wenn ein Abonnement abläuft.

So funktioniert es

- Die Online-Validierung verwendet denselben eindeutigen Token, der bei der ursprünglichen Online-Aktivierung erzeugt wurde, um das Gerät des Kunden zu identifizieren.
- Der Aktivierungsschlüssel selbst wird während der Validierung niemals über das Internet übertragen.
- Für die Durchführung der Validierungsprüfung ist eine aktive Internetverbindung erforderlich.

Konfiguration

👉 Sie müssen die **Base Validation URL** so konfigurieren, dass sie auf Ihr serverseitiges Validierungsskript verweist. Dies ist in der Regel die URL des XLS Padlock Activation Kit, des XLS Padlock WooCommerce Integration Kit oder des FastSpring subscription kit auf Ihrem Webserver. Wenn Sie das Aktivierungs-Kit beispielsweise in einem Unterordner namens „activation“ installiert haben, lautet die URL `https://www.yourdomain.com/activation/dovalidation/` .

> **HTTPS verwenden**
>
> Sichere Verbindungen über TLS/SSL werden unterstützt. Sie sollten immer URLs verwenden, die mit `https://` beginnen.

⚠️ Lassen Sie das Feld leer, wenn Sie die Online-Validierung **nicht** verwenden möchten.

Validierungshäufigkeit

Wählen Sie, wann die Anwendung eine Validierung durchführen soll: bei jedem Start, zufällig, alle X Tage oder alle X Ausführungen. Sie müssen den Wert für X angeben, wo dies erforderlich ist.

> **Warnung**
>
> Nach der Aktivierung ist beim nächsten Start eine erste Validierung erforderlich. Nach dieser ersten Prüfung hält sich die Anwendung an die ausgewählte Validierungshäufigkeit.

Aktion bei fehlgeschlagener Validierung

Legen Sie fest, was die Anwendung tun soll, wenn die Online-Validierung fehlschlägt: * **Exit the application:** Die Anwendung wird sofort geschlossen. * **Blacklist activation key:** Der aktuelle Schlüssel wird ungültig gemacht und der Benutzer wird aufgefordert, einen neuen einzugeben. Gibt der Benutzer denselben Schlüssel erneut ein, wird die Validierung erneut versucht. * **Do nothing:** Die Anwendung läuft weiter. Sie können die VBA-Erweiterungen von XLS Padlock verwenden, um den Validierungsstatus zu prüfen und eine eigene Logik umzusetzen.

Wenn die Aktivierung nicht validiert werden kann

Sie können dem Benutzer eine benutzerdefinierte Fehlermeldung anzeigen, wenn die Validierung fehlschlägt, und ihm so erklären, was als Nächstes zu tun ist.

Validierung überspringen, wenn offline

Sie können der Anwendung erlauben, die Validierungsprüfung zu überspringen, wenn keine Internetverbindung erkannt wird. Aus Sicherheitsgründen wird dies im Allgemeinen nicht empfohlen.

> **Warnung**
>
> Wenn diese Option deaktiviert ist, müssen die Benutzer für den Validierungsvorgang über eine aktive Internetverbindung verfügen. Andernfalls schlägt die Validierung fehl.

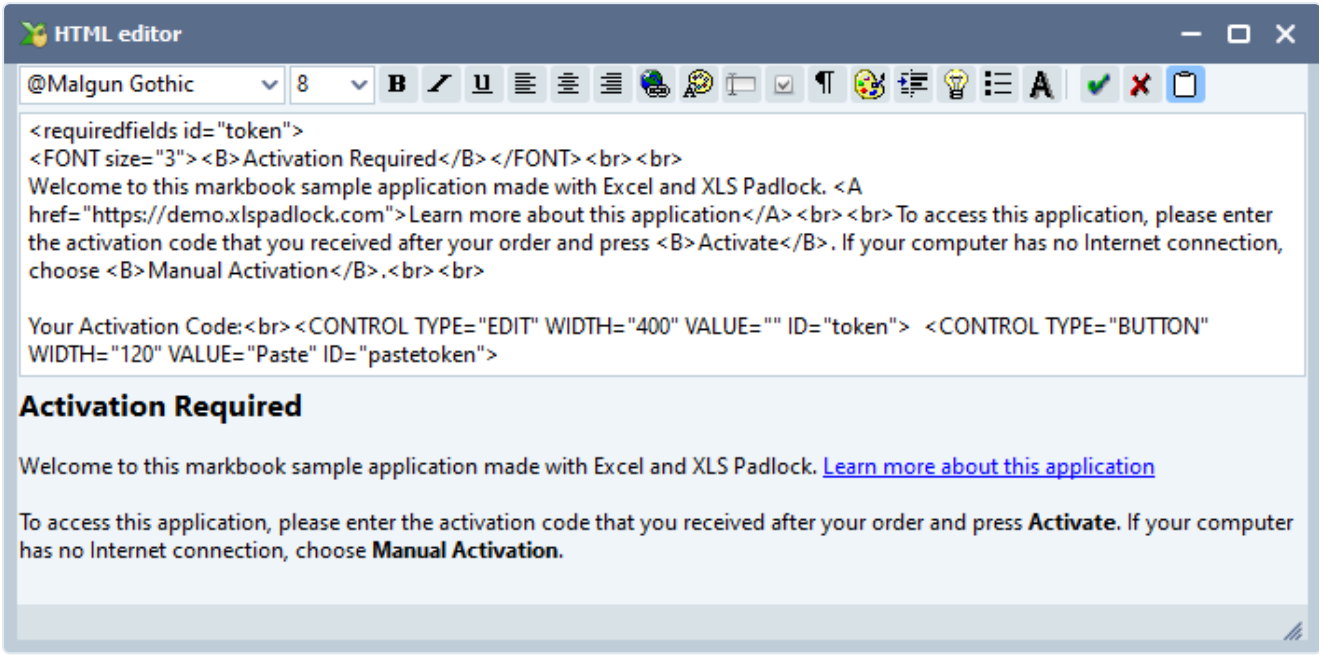
VBA: Abonnement-/Lizenzinformationen abrufen

Nach einer erfolgreichen Online-Validierung können Sie benutzerdefinierte Daten, die von Ihrem Server gesendet wurden, mit dem VBA-Aufruf `XLSPadlock.PLEvalVar("ValidationAddServerData")` abrufen. Dies ist nützlich, um abonnementspezifische Informationen an die Arbeitsmappe weiterzugeben.

Editor für das Registrierungsformular

Mit dem Registration Form Editor (Editor für das Registrierungsformular) passen Sie den Text an, der im Dialogfeld der Online-Aktivierung angezeigt wird. Der Editor unterstützt einfaches HTML, sodass Sie Tags wie `<b>`, `<br>`, `<font>` usw. verwenden können.

XLS Padlock stellt einen einfachen HTML-Editor mit einer Live-Vorschau des Dialogtextes bereit:



Benutzerdefinierte Steuerelemente

Sie können benutzerdefinierte Felder hinzufügen, um zusätzliche Informationen von den Nutzern anzufordern, die an Ihren Webserver gesendet werden. Sie können außerdem eine Schaltfläche hinzufügen, um Text in das Token-Feld einzufügen.

- **Paste Button (Einfügen-Schaltfläche):** `<CONTROL TYPE="BUTTON" WIDTH="120" VALUE="Paste" ID="pastetoken">`
- **Text Input (Texteingabe):** `<CONTROL TYPE="EDIT" WIDTH="400" VALUE="" ID="token">`

Pflichtfelder

Sie können Felder als Pflichtfelder festlegen, indem Sie deren IDs in einem `<requiredfields>`-Tag auflisten. Um beispielsweise die Felder "token" und "name" verpflichtend zu machen:

```
<requiredfields id="token,name">
```

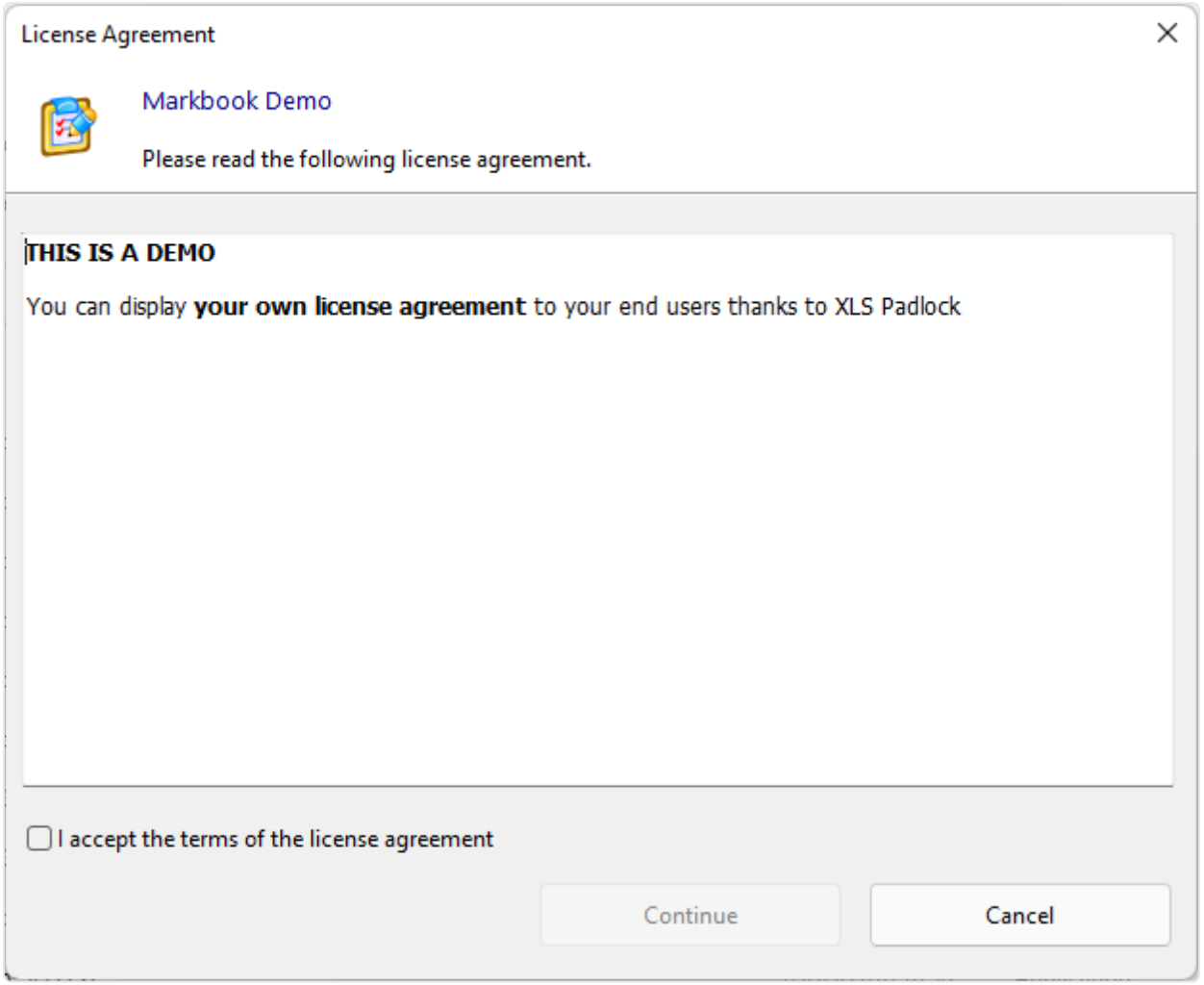
Vollständiges Beispiel

```
<requiredfields id="token">
<FONT size="3"><B>Activation Required</B></FONT><br><br>
Welcome to this application.<br><br>
To access this application, please enter the activation code that you received after your order
Your Activation Code:<br>
<CONTROL TYPE="EDIT" WIDTH="400" VALUE="" ID="token">  <CONTROL TYPE="BUTTON" WIDTH="120"
```

Lizenzvertrag anzeigen

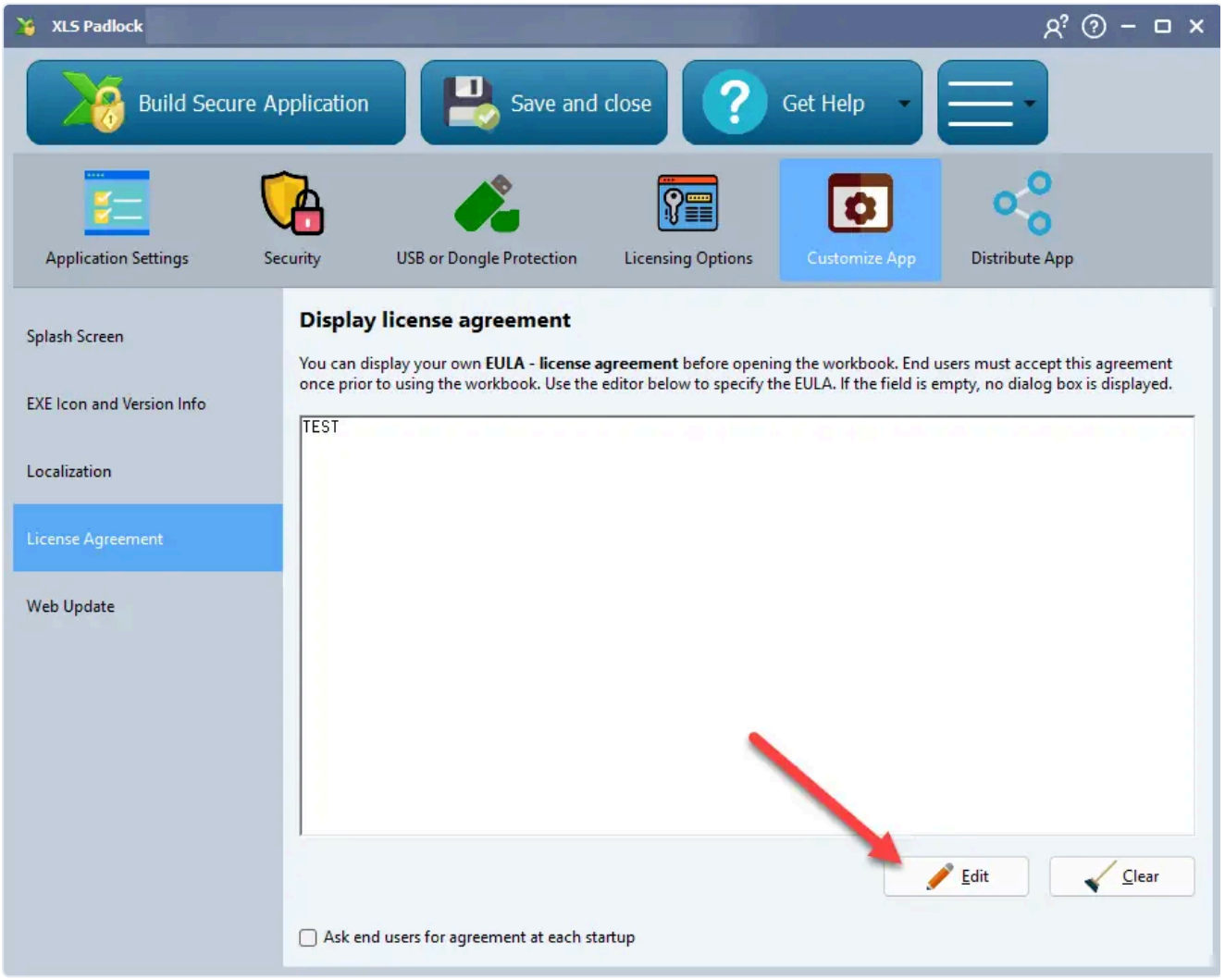
Sie können von Benutzern verlangen, dass sie einen **End User License Agreement (EULA)** (Endbenutzer-Lizenzvertrag) akzeptieren, bevor sie Ihre geschützte Arbeitsmappe öffnen können.

Die kompilierte Arbeitsmappe zeigt ein Dialogfeld an, wie unten dargestellt:



Wenn der Benutzer "I accept the terms of the license agreement" aktiviert, wird die Schaltfläche "Continue" aktiv, sodass er fortfahren kann.

👉 Um Ihre EULA hinzuzufügen, klicken Sie auf **Edit**, um auf dieser Seite einen Rich-Text-Editor zu öffnen:



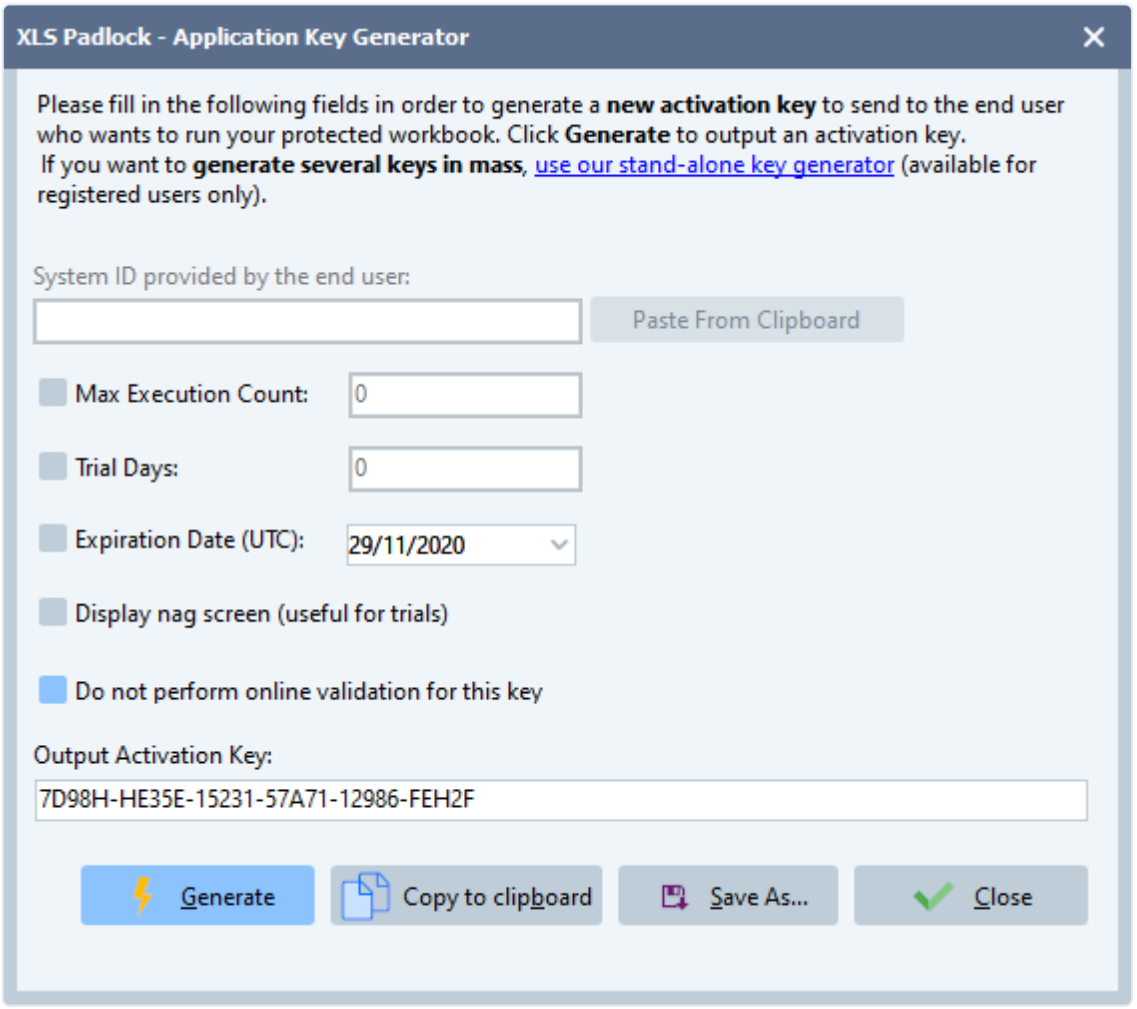
Sie können Inhalte aus einer RTF-Datei laden oder Text direkt aus einer Anwendung wie Microsoft Word einfügen.

> **HINWEIS**
>
> Wenn der EULA-Inhalt leer gelassen wird, wird kein Dialogfeld angezeigt.

ℹ️ Um die EULA bei jedem Start der Anwendung anzuzeigen, aktivieren Sie die Option **Ask end users for agreement at each startup**. Standardmäßig müssen Benutzer sie nur einmal akzeptieren.

Schlüsselgenerator (portabel und Server)

XLS Padlock enthält einen integrierten **Key Generator** (Schlüsselgenerator) zum Erstellen von Aktivierungsschlüsseln für Ihre Arbeitsmappen-Anwendungen.



- **Für einfache Aktivierungsschlüssel** klicken Sie einfach auf die Schaltfläche **Generate** (Generieren), um sofort einen Schlüssel zu erstellen.
- **Für hardwaregebundene Schlüssel** geben Sie die von Ihrem Benutzer bereitgestellte System ID ein und klicken anschließend auf **Generate**. Der daraus resultierende Schlüssel ist an diesen bestimmten Computer gebunden.

Sie können den generierten Schlüssel in die Zwischenablage kopieren, um ihn an Ihren Kunden zu senden, oder ihn als `.txt`-Datei für Ihre Unterlagen speichern.

> **Schlüssel sind arbeitsmappenspezifisch**
>
> Der Schlüsselgenerator erstellt Aktivierungsschlüssel nur für das aktuell geöffnete Arbeitsmappen-Projekt. Für eine Arbeitsmappe generierte Schlüssel funktionieren nicht mit einer anderen, es sei denn, sie verwenden denselben Application Master Key.

Schlüsselgenerator-SDK

Wir bieten ein kostenloses Key Generator SDK (eine PHP-basierte Webanwendung) an, das Sie auf Ihrer eigenen Website installieren können. Mit diesem SDK können Sie die Generierung und Auslieferung von Aktivierungsschlüsseln für Ihre Arbeitsmappen automatisieren.

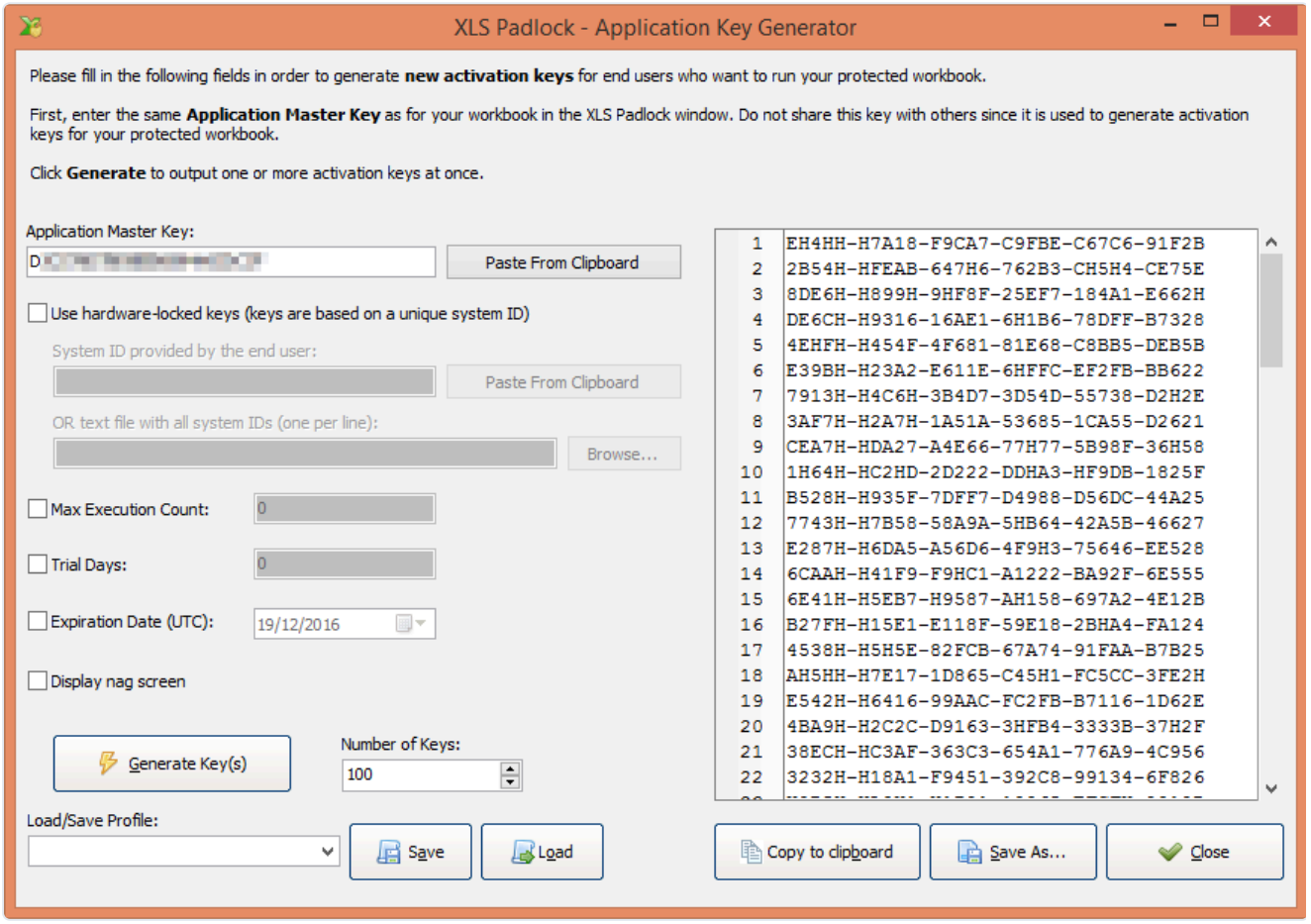
Im Gegensatz zum integrierten Schlüsselgenerator, der Schlüssel manuell innerhalb von XLS Padlock erstellt, wird dieses SDK auf Ihrem eigenen Server ausgeführt, um Schlüssel automatisch auszustellen.

Für eine vollständig automatisierte Lizenzierungslösung sollten Sie das SDK mit der Funktion zur Online-Aktivierung kombinieren.

👉 Sie können das Key Generator SDK von Ihrer Kontoseite herunterladen.

Eigenständiger Schlüsselgenerator

XLS Padlock bietet registrierten Kunden einen eigenständigen Schlüsselgenerator. Mit diesem Werkzeug können Sie Aktivierungsschlüssel erstellen für Ihre geschützten Arbeitsmappen, ohne Excel oder das XLS Padlock Add-In öffnen zu müssen.

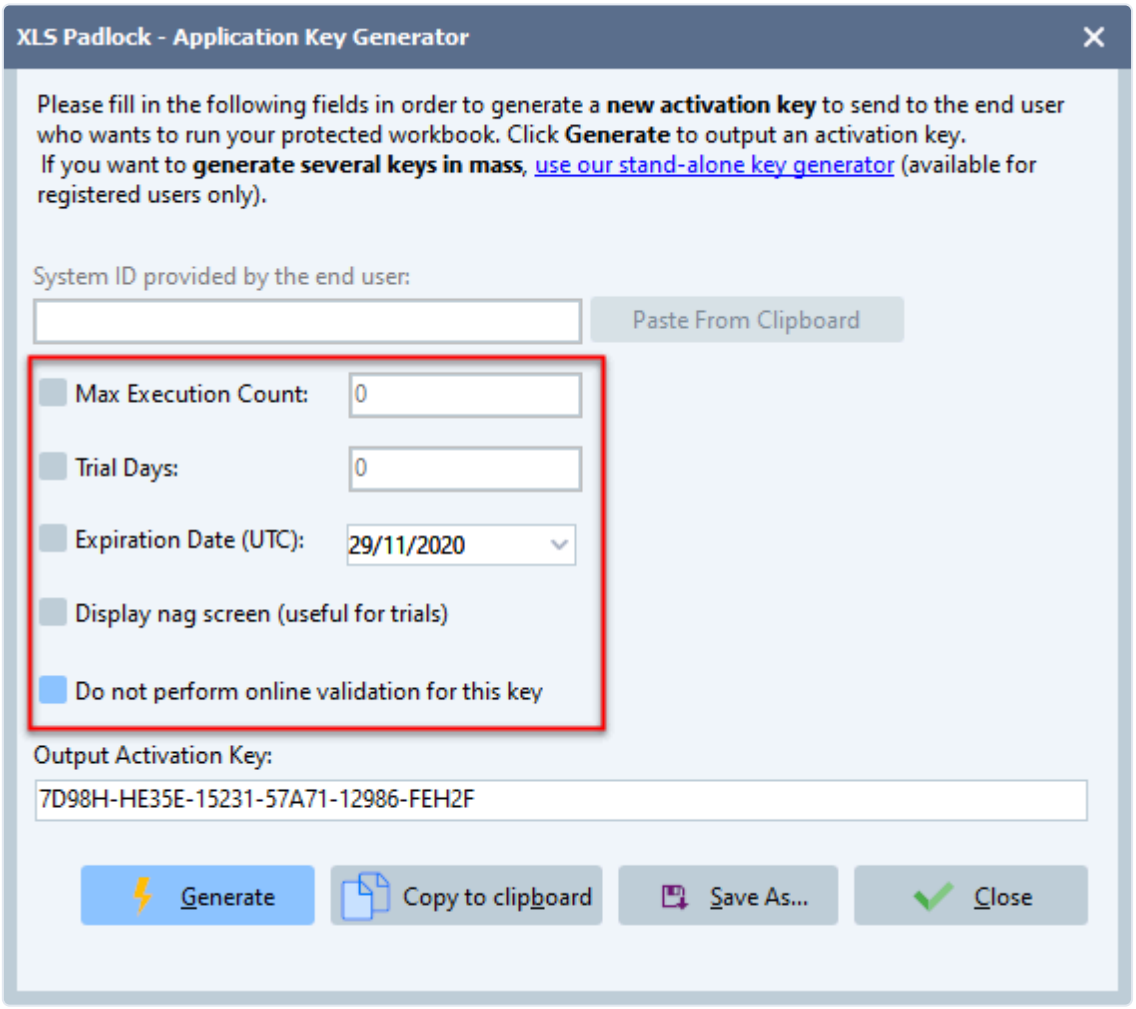


Für eine einfachere Verwaltung ermöglicht Ihnen der eigenständige Schlüsselgenerator außerdem, Einstellungen als **profiles** (Profile) zu speichern.

👉 Sie können den eigenständigen Schlüsselgenerator von Ihrer Kontoseite herunterladen.

Einschränkungen für Schlüssel

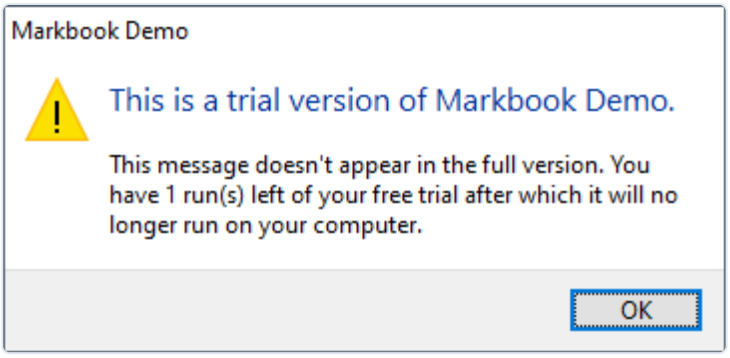
Wenn Sie mit XLS Padlock Activation Keys generieren, können Sie über die folgenden Optionen **verschiedene Einschränkungen anwenden**:



👉 Begrenzen Sie, wie oft Ihre Anwendung ausgeführt werden kann, indem Sie "**Max Execution Count**" aktivieren und die gewünschte Anzahl eingeben.

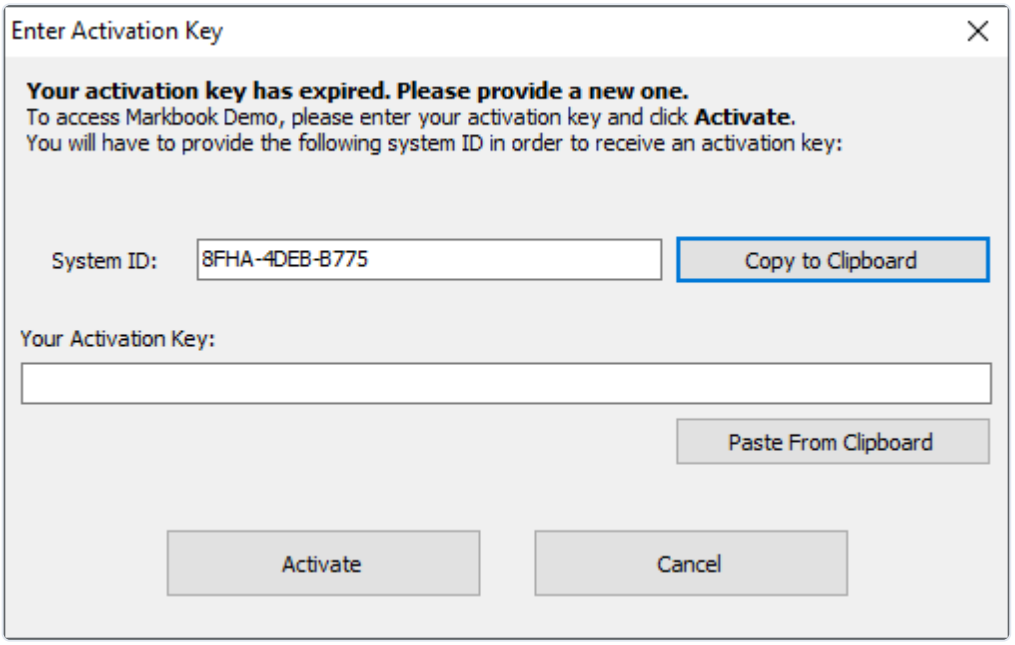
👉 Lassen Sie den Schlüssel nach einer bestimmten Anzahl von Tagen ("**Trial Days**" aktivieren) oder an einem bestimmten Datum ("**Expiration Date**" aktivieren) ablaufen.

👉 Zeigen Sie beim Start einen Erinnerungsdialog (oder "Nag Screen") an, indem Sie "**Display Nag Screen**" aktivieren. Dies ist nützlich für Testversionen, da der Benutzer dadurch über die verbleibenden Tage oder Ausführungen informiert wird.



Wenn Sie die Online-Validierung verwenden und einen bestimmten Schlüssel von dieser Prüfung ausnehmen müssen (zum Beispiel für einen Benutzer in einer Offline-Umgebung), können Sie schließlich "**Do not perform online validation for this key**" aktivieren. Dies wird nur empfohlen, wenn es in Kombination mit hardwaregebundenen Schlüsseln verwendet wird.

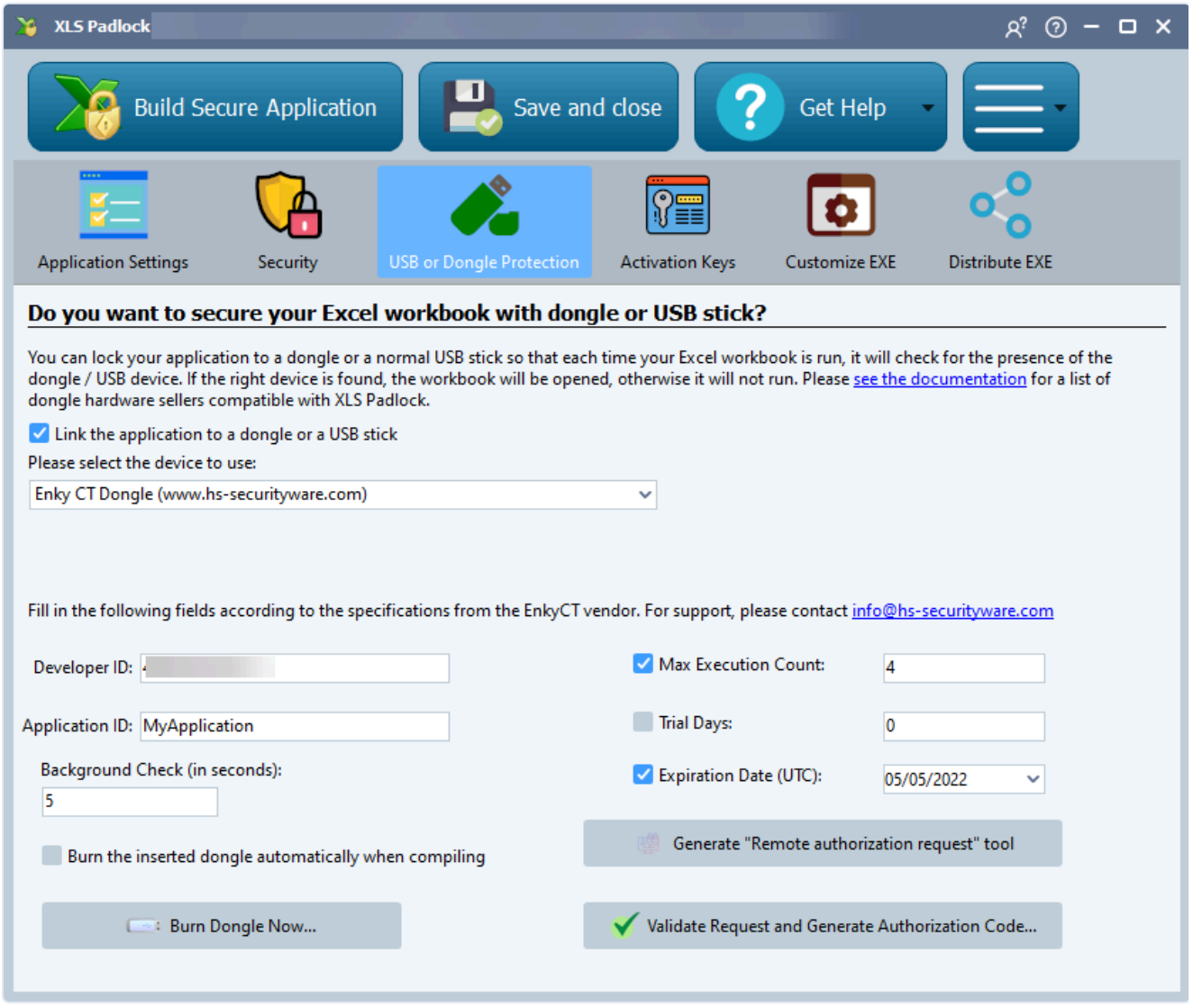
Wenn das Limit eines Schlüssels (Datum, Ausführungsanzahl oder Testtage) erreicht ist, wird der Benutzer aufgefordert, einen neuen Schlüssel einzugeben, um die Anwendung weiter nutzen zu können:



USB- oder Dongle-Schutz

Mit XLS Padlock können Sie Ihre Excel-Arbeitsmappen an einen USB-Stick oder einen speziellen Sicherheitsdongle binden. Das bedeutet, dass **die kompilierte Arbeitsmappen-Anwendung nicht funktioniert, wenn das richtige USB-Gerät nicht eingesteckt ist**. Die Anwendung prüft beim Start, ob das Gerät vorhanden ist, und kann dies auch während der Ausführung regelmäßig überprüfen.

Um diesen Schutz zu nutzen, aktivieren Sie auf der Seite "USB Protection" die Option "**Link the application to a dongle or a USB stick**" (Die Anwendung an einen Dongle oder USB-Stick binden) und wählen Sie das Gerätemodell aus, das Sie besitzen.



XLS Padlock unterstützt mehrere Arten von hardwarebasiertem Schutz, um Ihre Excel-Anwendung an ein physisches Gerät zu binden.

Enky CT Dongle

Laut Anbieter ist Enky CT ein einfach zu bedienender, flexibler 32-Bit-Smartcard-basierter Zeituhr-Dongle, der hauptsächlich für Softwareschutz und Zeitbegrenzung verwendet wird.

Für Unterstützung wenden Sie sich bitte an info@hs-securityware.com oder besuchen Sie https://hs-securityware.com.

So verwenden Sie den Enky CT Dongle

Geben Sie die von HS Securityware bereitgestellte **Developer ID** (Entwickler-ID) ein (Groß- und Kleinschreibung wird unterschieden).

Die **Application ID** (Anwendungs-ID) kann ein beliebiger Wert sein, mit dem Sie den Dongle identifizieren möchten, und sie sollte mit der Anwendung in Zusammenhang stehen, die Sie erstellen.

Background Check (Hintergrundprüfung)

Die Anwendung prüft beim Start, ob der Dongle vorhanden ist, und kann dies auch während der Laufzeit regelmäßig überprüfen. Sie können die Häufigkeit dieser Prüfungen festlegen; ein Wert von 20 bedeutet beispielsweise, dass die Anwendung alle 20 Sekunden prüft, ob der Dongle vorhanden ist. Wird der Dongle nicht gefunden, erscheint ein Hinweisfenster (Nag Screen), das den Benutzer auffordert, den Dongle innerhalb von 15 Sekunden einzustecken, andernfalls wird die Anwendung beendet.

Set restrictions on the application (Beschränkungen für die Anwendung festlegen)

Sie können die Anzahl der Ausführungen Ihrer Anwendung begrenzen, indem Sie einen **Max Execution Count** (maximale Ausführungsanzahl) festlegen. Sie können Ihre Anwendung außerdem nach einer bestimmten Anzahl von Tagen (**Trial Days**, Testtage) oder an einem bestimmten **Expiration Date** (Ablaufdatum) ablaufen lassen.

Funktionen zur Remote-Aktualisierung des Dongles

Mit diesem Dongle-Modell können Sie Dongles, die sich bereits bei Ihren Kunden befinden, aus der Ferne aktualisieren. Sie können beispielsweise ein Ablaufdatum verlängern oder weitere Ausführungen hinzufügen. Diese Remote-Aktualisierungsfunktion basiert auf einem System aus Anforderungs- und Autorisierungscode.

So aktualisieren Sie den Dongle eines Kunden aus der Ferne:

1. Erstellen Sie zunächst das "remote authorization request tool" (Werkzeug zur Anforderung der Remote-Autorisierung) aus XLS Padlock und senden Sie die daraus entstehenden EXE- und DAT-Dateien an Ihren Kunden.

   Warnung

   Diese beiden Dateien (`.EXE` und `.DAT`) müssen im selben Ordner verbleiben, damit das Werkzeug funktioniert.

2. Der Kunde führt das Werkzeug aus, um einen **request code** (Anforderungscode) zu erzeugen, den er Ihnen sendet.
3. Wählen Sie in XLS Padlock "Validate Request and Generate Authorization Code" (Anforderung validieren und Autorisierungscode erzeugen), fügen Sie den request code ein und klicken Sie auf

"Validate Request Code". Sie sehen dann den aktuellen Status des Dongles und können neue Beschränkungen festlegen (Nutzungsanzahl oder ein neues Ablaufdatum).

4. Erzeugen Sie den **authorization code** (Autorisierungscode) und senden Sie ihn an den Kunden zurück. Dieser verwendet dasselbe Werkzeug, um den Code einzugeben und die Aktualisierung auf seinen Dongle anzuwenden.

Enky LC Dongle

Für Unterstützung wenden Sie sich bitte an info@hs-securityware.com oder besuchen Sie https://hs-securityware.com.

Der Enky LC ist ein kostengünstiger, treiberloser HID-Dongle für den Softwareschutz.

So verwenden Sie den Enky LC2 Dongle

1. Geben Sie die **Developer ID** (Entwickler-ID) ein, die Sie von HS-Security Ware erhalten haben.
2. Geben Sie eine eindeutige **Product ID** (Produkt-ID) für Ihre Arbeitsmappe ein. Dadurch wird sichergestellt, dass nur Dongles mit der richtigen Product ID akzeptiert werden.

Schritte zur Konfiguration eines Enky LC2 Dongles

Um einen Dongle an Ihre Anwendung zu binden, müssen Sie ihn zunächst mit der Schaltfläche "**Burn Dongle Now**" (Dongle jetzt brennen) in XLS Padlock "brennen". Dadurch wird der Dongle mit Ihren IDs konfiguriert. Dieser Vorgang kann auch automatisch ablaufen, wenn Sie Ihre Anwendung kompilieren und ein kompatibler Dongle eingesteckt ist.

Background Check (Hintergrundprüfung)

Sie können eine Hintergrundprüfung (Background Check) aktivieren, um sicherzustellen, dass der Dongle während der Ausführung der Anwendung eingesteckt bleibt. Sie können das Intervall (in Sekunden) festlegen, in dem die Prüfung durchgeführt wird.

Schutz durch generische USB-Sticks

Sie können Ihre Anwendung anhand ihrer eindeutigen Hersteller-ID an einen oder mehrere generische USB-Sticks binden.

- Zunächst müssen Sie eine **Application Secret ID** (geheime Anwendungs-ID) eingeben. Diese wird mit der ID des USB-Sticks kombiniert, um einen eindeutigen Hash zu erzeugen.
- Wenn die Anwendung gestartet wird, durchsucht sie alle USB-Laufwerke. Wird ein Laufwerk mit einem autorisierten Hash gefunden, öffnet sich die Arbeitsmappe.

Um einen USB-Stick zu autorisieren, stecken Sie ihn ein, wählen Sie ihn in der Laufwerksliste in XLS Padlock aus und klicken Sie auf "Allow this USB disk" (Diesen USB-Datenträger zulassen). Sie können

mehrere USB-Sticks autorisieren.

> **HINWEIS**
>
> Der Schutz durch USB-Sticks bietet weniger Flexibilität und Sicherheit als der Schutz durch einen speziellen Dongle.

Deaktivierung

Einführung in die Deaktivierung

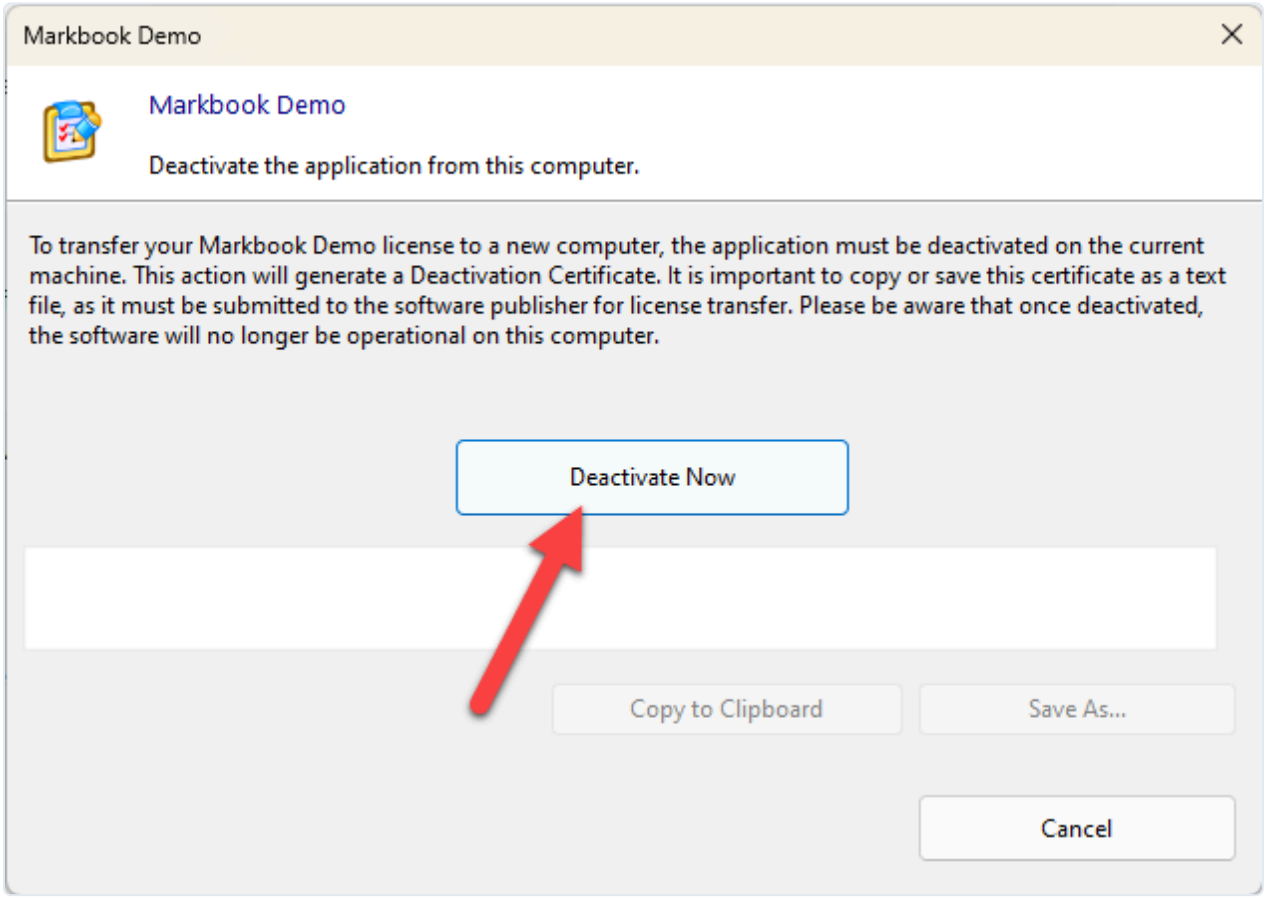
XLS Padlock bietet ein robustes Deaktivierungssystem, mit dem Endbenutzer die Registrierung ihrer Anwendung auf einem Computer aufheben können. Dies ist besonders nützlich, wenn Kunden eine Lizenz auf eine neue Maschine übertragen oder ihre Software nach erheblichen Systemänderungen erneut aktivieren müssen.

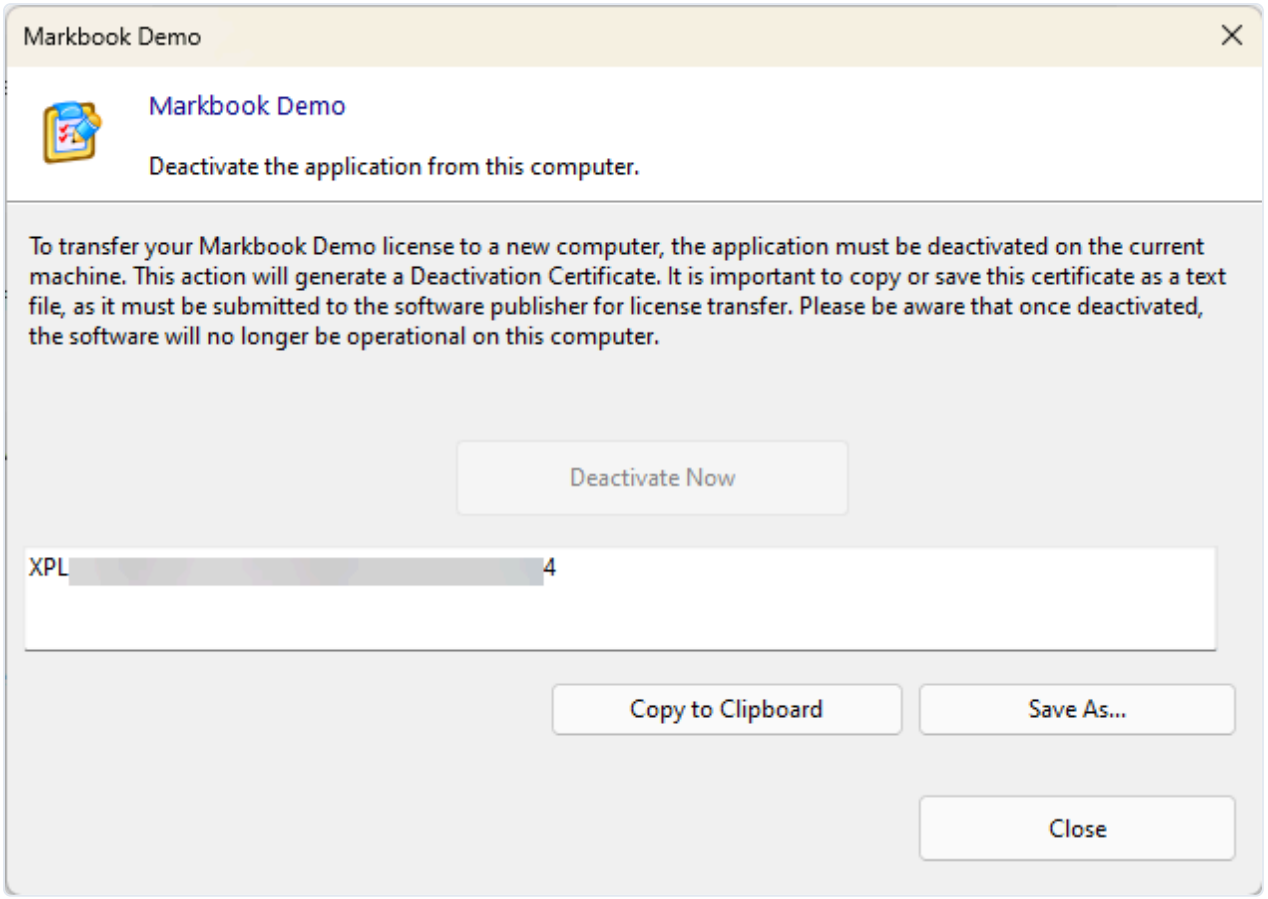
Wie die Deaktivierung funktioniert

Bei der Deaktivierung wird ein Zertifikat erzeugt, das der Endbenutzer Ihnen zusenden muss. Mit der Funktion **Test Deactivation Certificate** (Deaktivierungszertifikat prüfen) in XLS Padlock können Sie die Echtheit des Zertifikats überprüfen. Es ist wichtig zu beachten, dass die Aktivierungsschlüssel (Activation Key) einer Anwendung ungültig wird, sobald diese deaktiviert wurde. Sie müssen dem Benutzer einen neuen Schlüssel für jede künftige Aktivierung ausstellen.

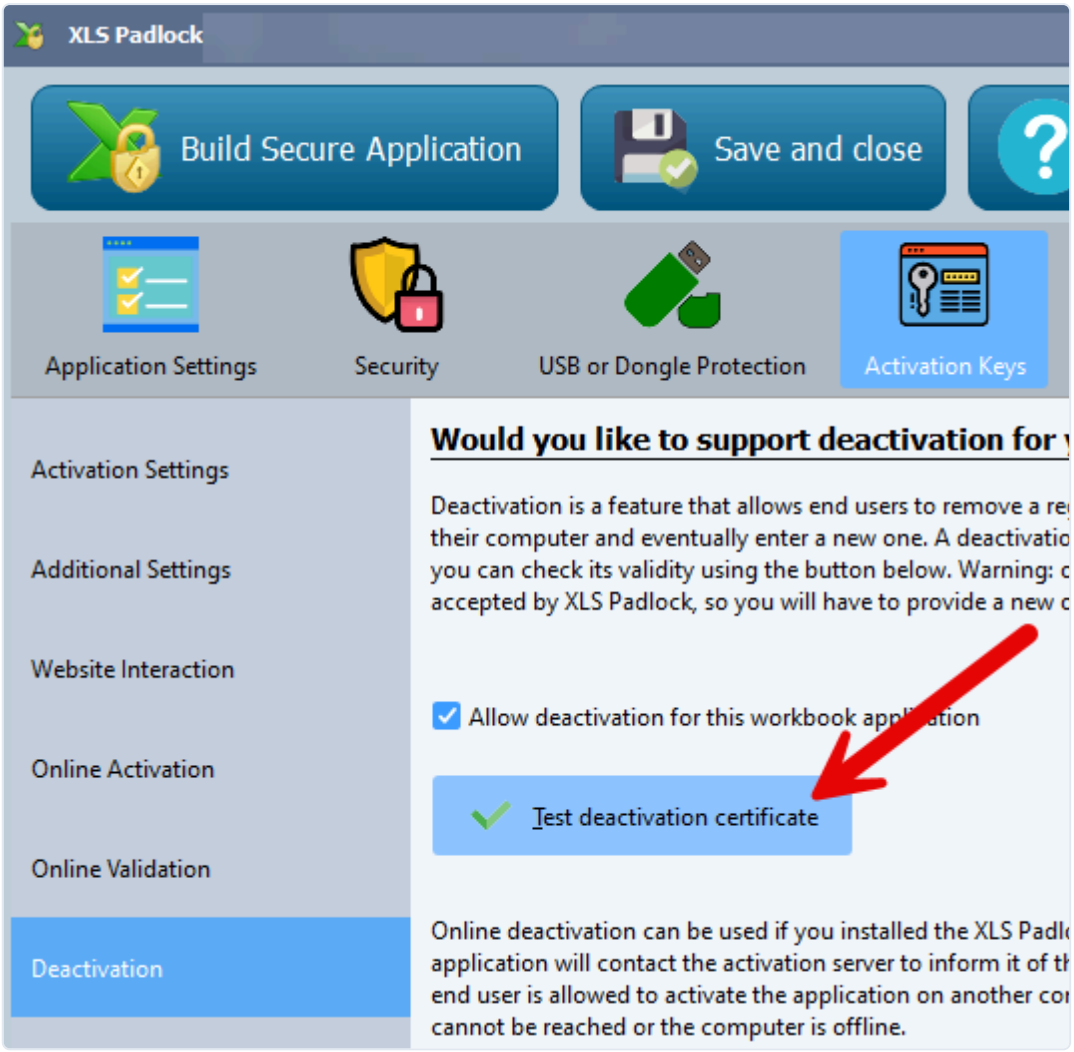
Manuelle Deaktivierung

Wenn ein Benutzer seine Anwendung deaktiviert, muss er ein Deaktivierungszertifikat erzeugen und es Ihnen zusenden:





Nach Erhalt können Sie die Schaltfläche 'Test deactivation certificate' in XLS Padlock verwenden, um das Zertifikat zu überprüfen. Wenn die Deaktivierung erfolgreich ist, zeigt XLS Padlock das Deaktivierungsdatum und die eindeutige Kennung des Computers des Benutzers an:

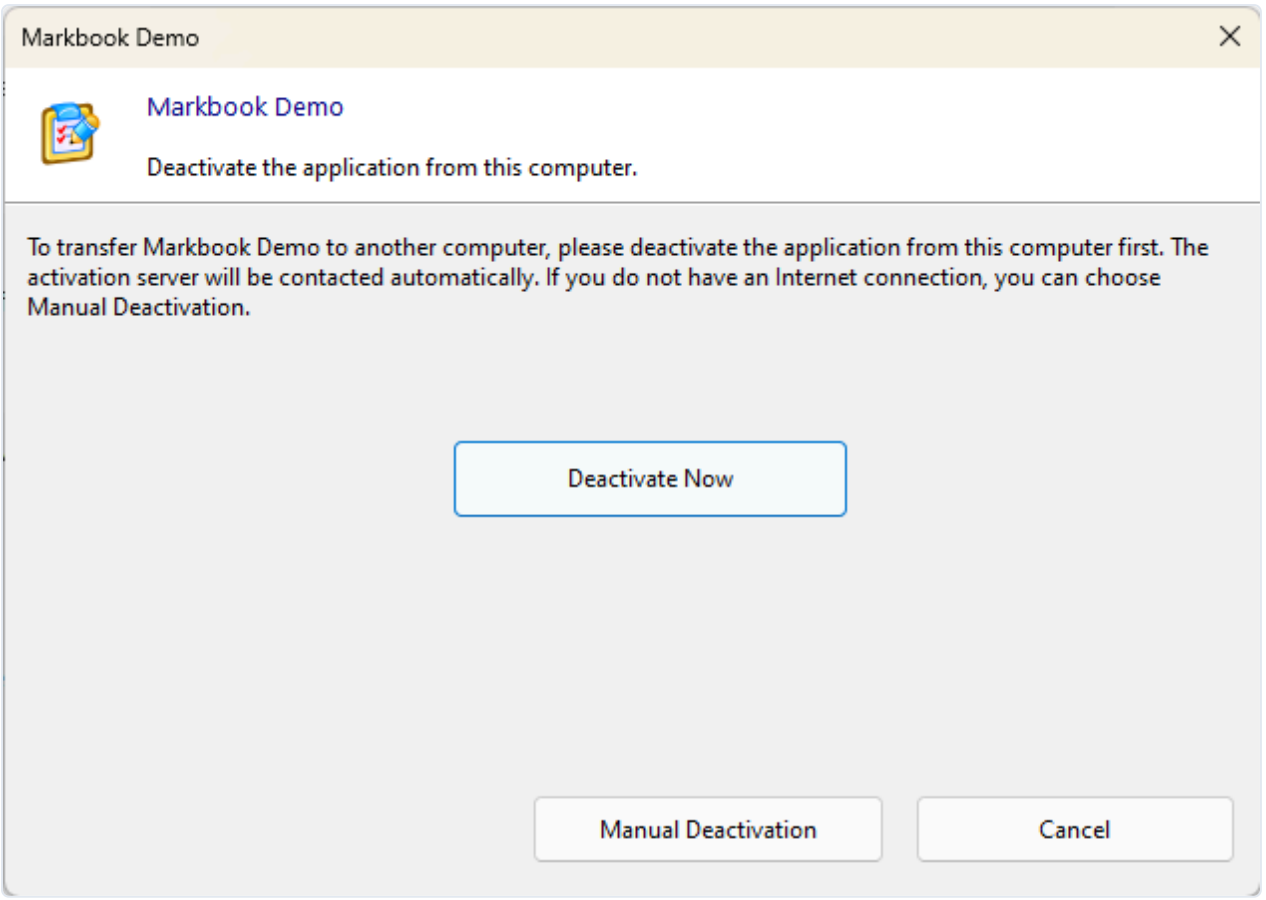

XLS Padlock
Build Secure Application
Save and close
Application Settings
Security
USB or Dongle Protection
Activation Keys
Activation Settings
Additional Settings
Website Interaction
Online Activation
Online Validation
Deactivation
Would you like to support deactivation for
Deactivation is a feature that allows end users to remove a re
their computer and eventually enter a new one. A deactivatio
you can check its validity using the button below. Warning: c
accepted by XLS Padlock, so you will have to provide a new c
Allow deactivation for this workbook appl ation
Test deactivation certificate
Online deactivation can be used if you installed the XLS Padl
application will contact the activation server to inform it of th
end user is allowed to activate the application on another co
cannot be reached or the computer is offline.


Test deactivation certificate
What certificate to test?
Please provide the deactivation certificate sent to you by the end user:
XPL
4
OK
Cancel

Diese Informationen ermöglichen es Ihnen, neue Aktivierungen für Ihren Kunden zu verwalten.

Online-Deaktivierung

Für einen reibungsloseren Ablauf unterstützt XLS Padlock die Online-Deaktivierung, sofern Sie das XLS Padlock Activation Kit auf Ihrem Webserver installiert haben. Die Anwendung kommuniziert mit Ihrem Server, um die Aktivierung automatisch aufzuheben. Wenn der Server nicht erreichbar ist oder der Computer offline ist, wird als Ausweichlösung ein Zertifikat für die manuelle Deaktivierung erstellt.



Die Online-Deaktivierung automatisiert den gesamten Vorgang und vereinfacht ihn sowohl für Sie als auch für Ihre Kunden.

Konfiguration der Online-Deaktivierung

Base Deactivation URL

Um die Online-Deaktivierung zu nutzen, müssen Sie die URL zum XLS Padlock Activation Kit oder zum XLS Padlock WooCommerce Integration Kit auf Ihrem Webserver angeben. Wenn Sie das Activation Kit beispielsweise in einem Unterordner namens „activation“ installiert haben, würde die URL `https://www.yourdomain.com/activation/dodeactivation` lauten.

> **Verwenden Sie HTTPS**
>
> Sichere Verbindungen mit TLS/SSL werden unterstützt. Sie sollten immer URLs verwenden, die mit `https://` beginnen.

⚠️ Wenn Sie die Deaktivierung lieber manuell verwalten möchten, lassen Sie dieses Feld leer.

Hide Manual Deactivation Button (in diesem Fall wird sie angezeigt, wenn die automatisierte Deaktivierung fehlschlägt)

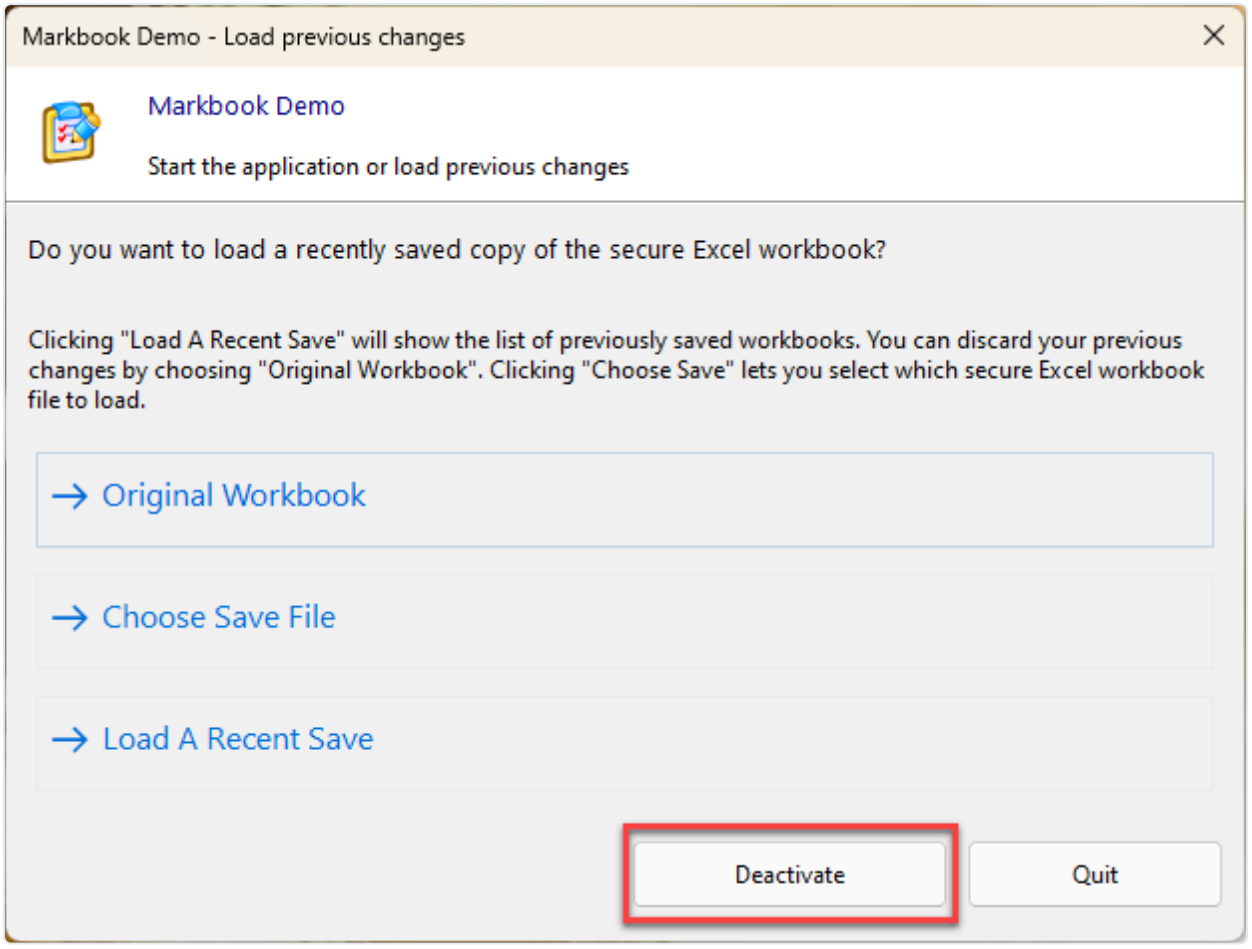
Standardmäßig ist die manuelle Deaktivierung immer als Ausweichlösung verfügbar. Wenn Sie Benutzer dazu ermutigen möchten, die Online-Deaktivierung zu verwenden, können Sie die Schaltfläche für die manuelle Deaktivierung ausblenden.

> **HINWEIS**
>
> Auch wenn die Schaltfläche ausgeblendet ist, bietet die Anwendung automatisch die manuelle Deaktivierung an, falls der Online-Vorgang fehlschlägt (z. B. wenn der Deaktivierungsserver nicht erreichbar ist oder der Benutzer offline ist).

So starten Sie die Deaktivierung

Ein Benutzer kann den Deaktivierungsvorgang auf drei Arten einleiten:

- Die erste Option führt über den integrierten Welcome-Dialog der Anwendung. Beim Starten der sicheren Anwendung erscheint der Begrüßungsdialog mit verschiedenen Optionen, darunter eine Schaltfläche "Deactivate". Durch Klicken auf diese Schaltfläche können Benutzer den Deaktivierungsvorgang auf benutzerfreundliche und geführte Weise beginnen:



- Die zweite Methode nutzt die Befehlszeile. Fortgeschrittene Benutzer bevorzugen diese Methode möglicherweise für die Systemautomatisierung oder Integration. Dazu muss der Benutzer die Windows-Eingabeaufforderung öffnen, in das Verzeichnis wechseln, in dem sich die ausführbare Datei der sicheren Anwendung befindet, und dann den Namen der ausführbaren Datei gefolgt vom Parameter `**-deact**` eingeben. Wenn die ausführbare Datei beispielsweise `Application.exe` heißt, würde der vollständige Befehl lauten:

```
C:\Path\to\application\Application.exe -deact
```

Durch Drücken der Eingabetaste wird der Parameter an die ausführbare Datei übergeben, wodurch der Deaktivierungsvorgang ausgelöst wird.

- Die dritte Methode verwendet die von XLS Padlock bereitgestellte VBA-API.

Die Deaktivierung ist auf einer Maschine endgültig

Bitte beachten Sie, dass ein Schlüssel, sobald er auf einem Computer deaktiviert wurde, auf dieser Maschine dauerhaft auf eine Sperrliste gesetzt wird und nicht wiederverwendet werden kann. Die Deaktivierung ist eine endgültige Aktion für dieses spezifische System. Um die Anwendung auf demselben Computer erneut zu verwenden, ist ein neuer Aktivierungsschlüssel (Activation Key) erforderlich. Beachten Sie, dass sich dies auf Activation Keys bezieht, nicht auf Aktivierungs-Token (wie im WooCommerce Integration Kit definiert).

👉 Siehe auch:, [Online-Aktivierung](#chapter-online-activation), Der integrierte Welcome-Dialog

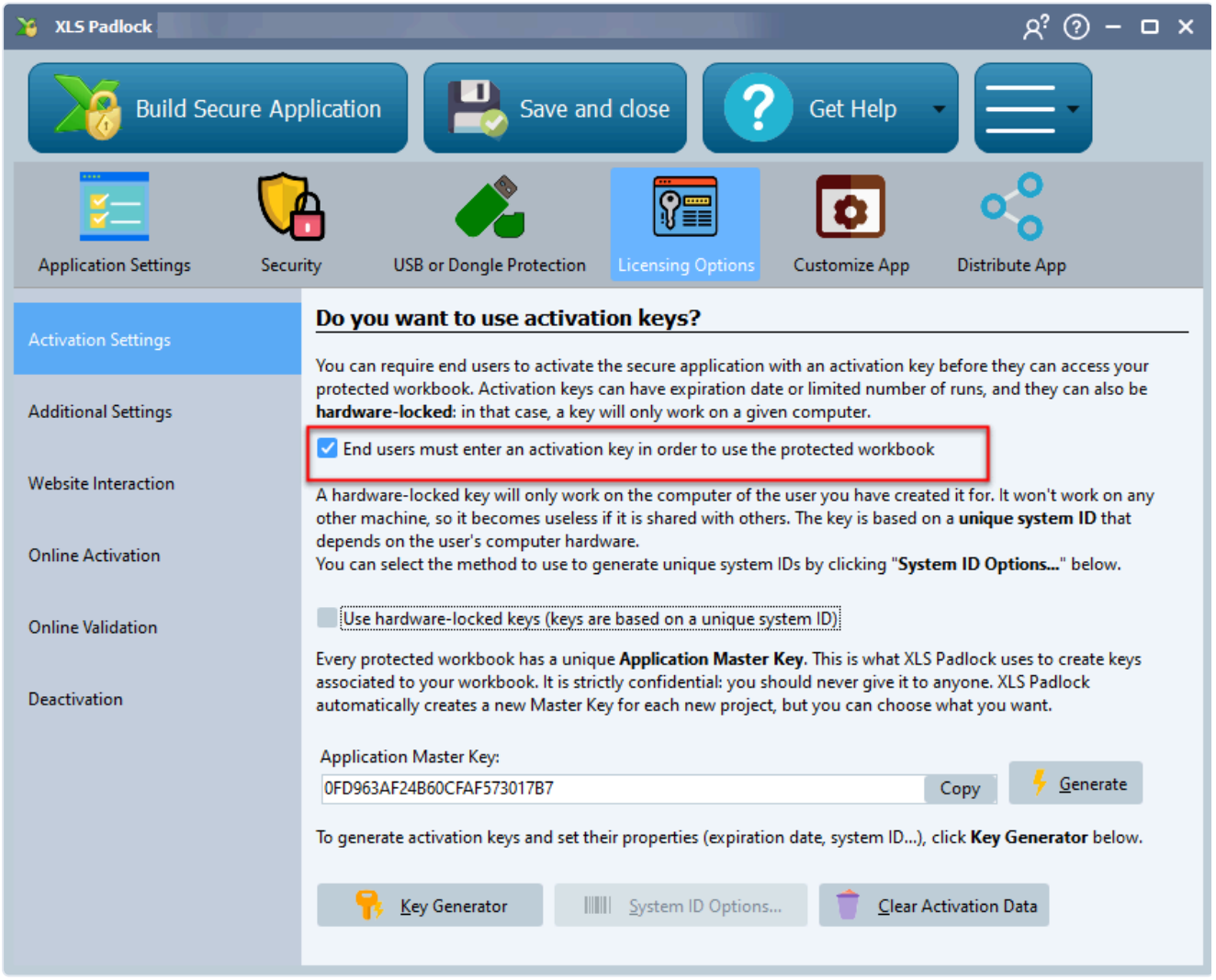
Test-Arbeitsmappen erstellen

Sie können potenziellen Kunden eine Vorschau Ihrer Excel-Arbeitsmappen anbieten, indem Sie eine Testversion erstellen, die eine Lizenz benötigt, um den vollen Funktionsumfang freizuschalten. Dies ist mit den Aktivierungsschlüsseln von XLS Padlock möglich.

Diese Anleitung zeigt Ihnen, wie Sie einen Testzeitraum für Ihre Arbeitsmappen-Anwendung einrichten.

1. Aktivierungsschlüssel aktivieren

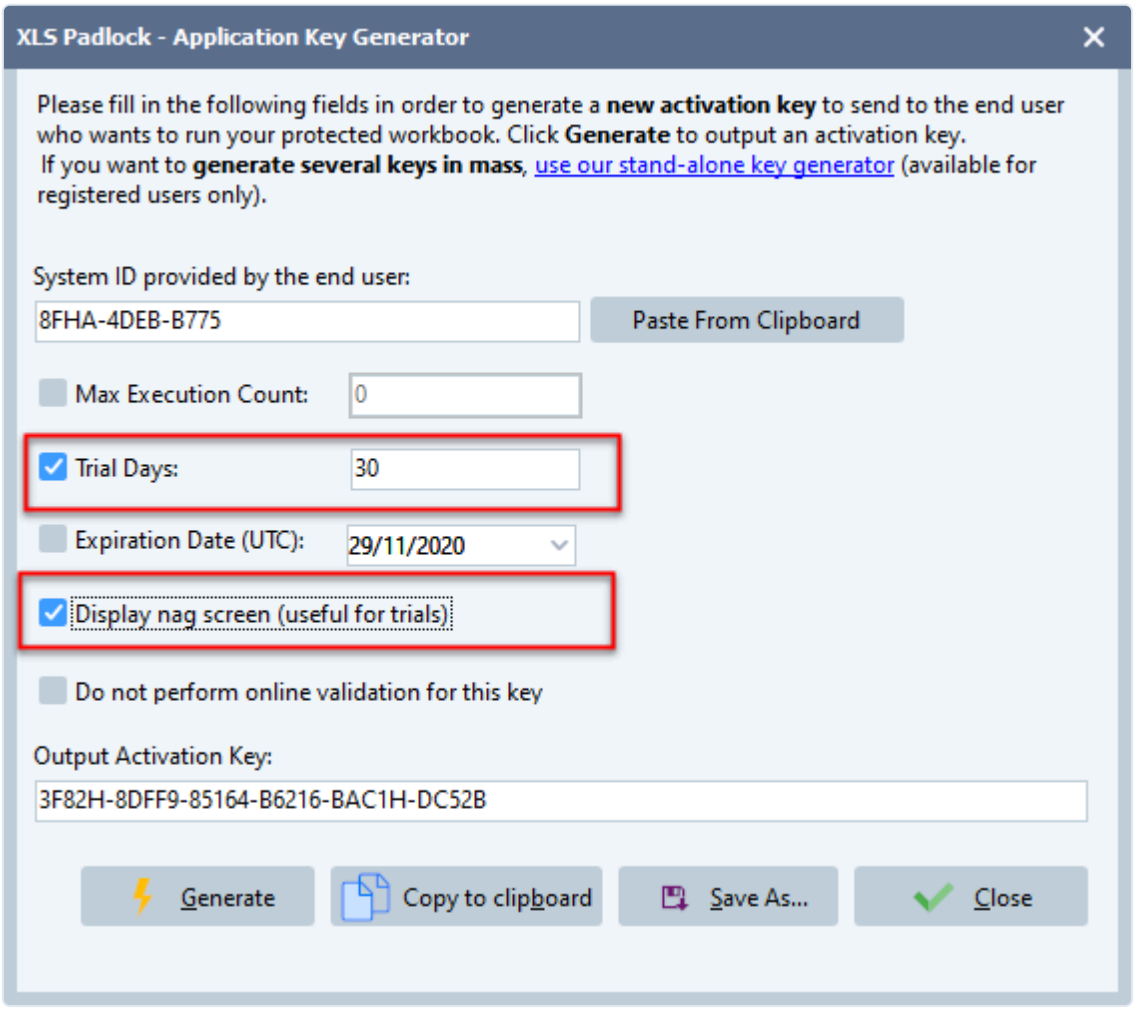
Aktivieren Sie zunächst die Option **"End users must enter an activation key in order to use the protected workbook"** (Endbenutzer müssen einen Aktivierungsschlüssel eingeben, um die geschützte Arbeitsmappe zu verwenden) und erstellen Sie Ihre Anwendung neu.



2. Einen Test-Aktivierungsschlüssel generieren

Öffnen Sie den Key Generator in XLS Padlock.

Um einen Testzeitraum festzulegen, können Sie eine Begrenzung der **Trial days** (Testtage, zum Beispiel 15 Tage), ein festes **Expiration Date** (Ablaufdatum) oder eine **Max Execution Count** (maximale Anzahl an Ausführungen) festlegen.

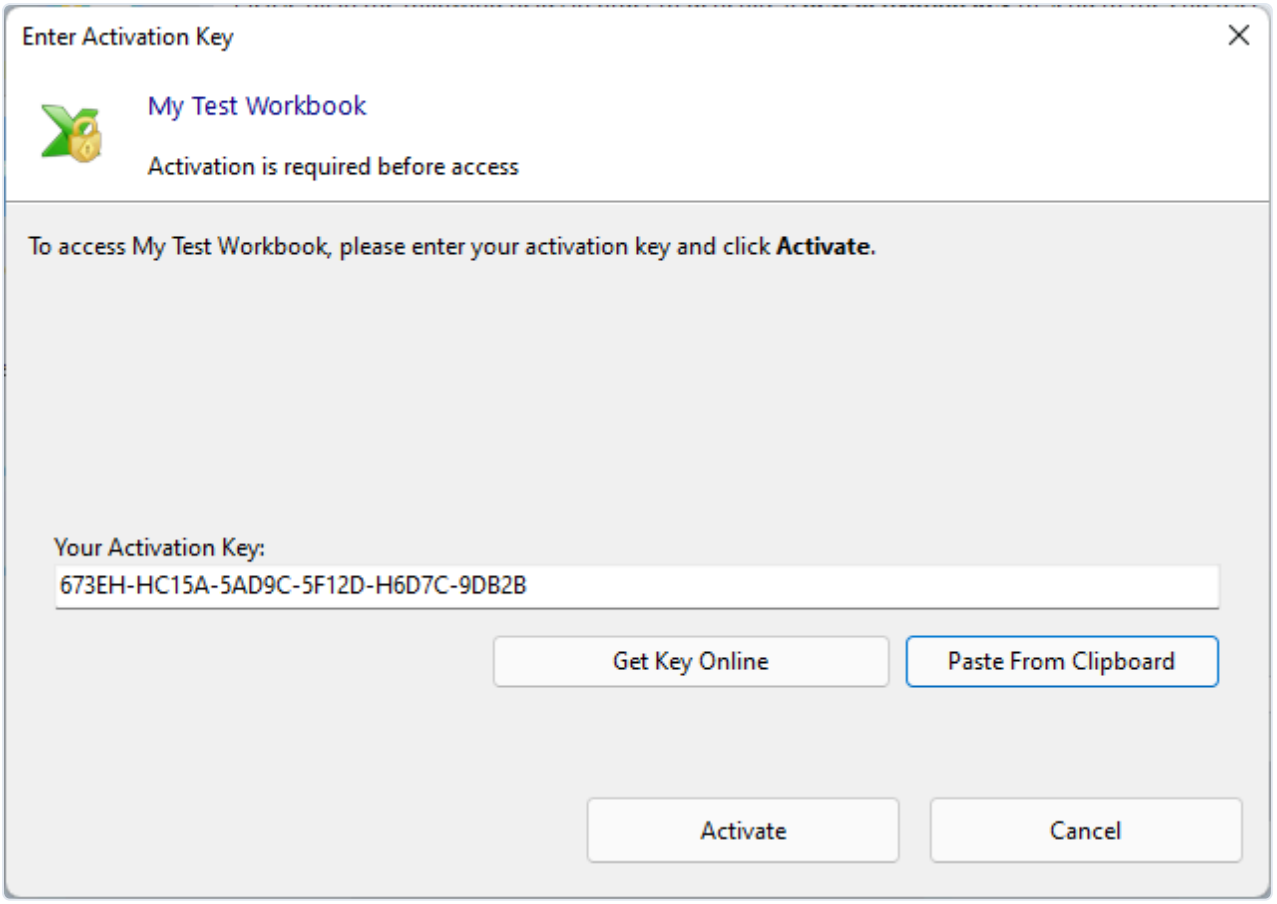


Um Kunden daran zu erinnern, dass sie eine Testversion verwenden, aktivieren Sie **"Display nag screen"** (Erinnerungsbildschirm anzeigen). Dadurch wird bei jedem Start der Anwendung ein Erinnerungsdialog angezeigt.

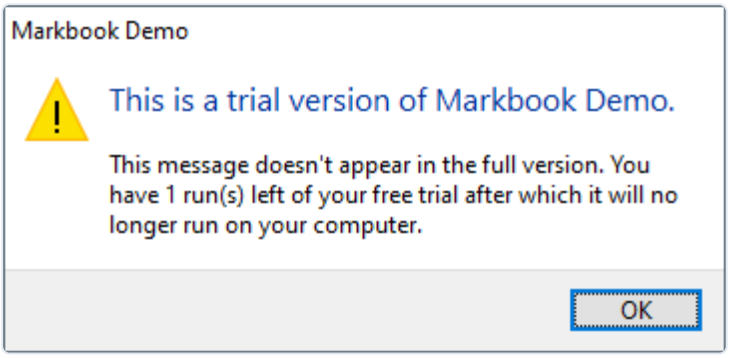
Klicken Sie auf **Generate** (Generieren), um den Testschlüssel zu erstellen, den Sie anschließend an Ihre Benutzer verteilen können.

So funktioniert die Anwendung im Testmodus

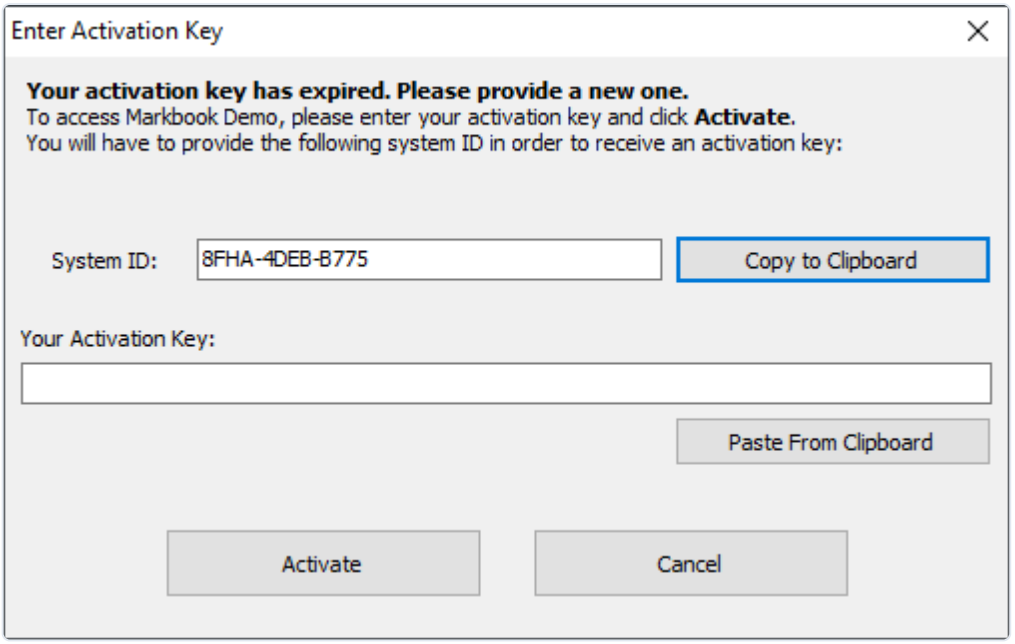
Ihre Kunden geben den Testschlüssel ein, um die Anwendung zu aktivieren.



Nach der Aktivierung wird die Arbeitsmappe geöffnet, der Erinnerungsbildschirm erscheint jedoch, um die verbleibende Testdauer anzuzeigen.



Wenn der Testzeitraum abgelaufen ist, benötigt die Anwendung einen neuen Aktivierungsschlüssel ohne Testbeschränkung, um weiterhin zu funktionieren.



Den Teststatus mit VBA prüfen

Sie können auch programmgesteuert prüfen, ob sich die kompilierte Arbeitsmappe im Teststatus befindet, indem Sie die VBA-API verwenden.

Nach einer bestimmten Zeit schließen

> **Wie kann ich verhindern, dass ein Benutzer eine Testversion unbegrenzt geöffnet lässt?**
>
> Ein Benutzer könnte einen Testschlüssel für einen Tag erhalten, die EXE-Datei öffnen und sie dauerhaft geöffnet lassen. Die Anwendung wird weiterhin funktionieren. Wie lässt sich das verhindern?

Lösung: Sie können VBA verwenden, um mit `Application.OnTime` einen Timer einzurichten, der die Arbeitsmappe nach einer festgelegten Dauer automatisch schließt. Dies ist eine wirksame Methode, um Testbeschränkungen durchzusetzen.

Kombinieren Sie dies mit den Sicherheitsoptionen von XLS Padlock, um den Zugriff auf den VBA-Editor zu verbieten, und kompilieren Sie Ihren VBA-Code, um zu verhindern, dass Benutzer den Timer deaktivieren.

Umsetzung

Fügen Sie den folgenden Code in das Modul **`ThisWorkbook`** Ihres VBA-Projekts ein.

```
' --- In ThisWorkbook module ---

' Variable to store the scheduled time for the timer
Private mScheduledTime As Date

' This procedure will be called by the timer to close the workbook
Public Sub CloseAndSave()
    ' Save any changes and close the workbook
    ThisWorkbook.Close SaveChanges:=True
End Sub

' This event runs when the workbook is opened, starting the timer
Private Sub Workbook_Open()
    ' Set the timer to run the "CloseAndSave" procedure in 8 hours.
    ' You can change the time value as needed.
    mScheduledTime = Now + TimeValue("08:00:00")
    Application.OnTime EarliestTime:=mScheduledTime, Procedure:="ThisWorkbook.CloseAndSave"

    ' Optional: Inform the user that the application will close automatically.
    ' MsgBox "This application will automatically close in 8 hours.", vbInformation
End Sub

' This event runs just before the workbook closes
Private Sub Workbook_BeforeClose(Cancel As Boolean)
    ' Cancel the scheduled OnTime event to prevent errors if the user
    ' closes the workbook manually before the timer runs.
    On Error Resume Next
    Application.OnTime EarliestTime:=mScheduledTime, Procedure:="ThisWorkbook.CloseAndSave", Sc
End Sub
```

So funktioniert es

1. **`Workbook_Open`**: Beim Start der Anwendung plant dieses Ereignis die Ausführung des Makros `CloseAndSave` nach 8 Stunden (`TimeValue("08:00:00")`). Sie können diese Dauer an Ihre Bedürfnisse anpassen (zum Beispiel `TimeValue("01:00:00")` für eine Stunde).
2. **`CloseAndSave`**: Dies ist das Makro, das die Aktion ausführt. Es speichert die Arbeitsmappe und schließt sie anschließend.
3. **`Workbook_BeforeClose`**: Dies ist ein entscheidender Bereinigungsschritt. Wenn der Benutzer die Arbeitsmappe manuell schließt, hebt dieser Code den ausstehenden Timer auf und verhindert so, dass Excel versucht, ein Makro für eine Arbeitsmappe auszuführen, die nicht mehr geöffnet ist, was zu einem Fehler führen würde.

Teststatus prüfen

XLS Padlock stellt eine VBA-API bereit, mit der Sie programmgesteuert ermitteln können, ob Ihre kompilierte Arbeitsmappe im Testmodus ausgeführt wird. Mit der folgenden Funktion können Sie bestimmte Funktionen aktivieren oder deaktivieren, Hinweise auf ein Upgrade anzeigen oder weitere benutzerdefinierte Einschränkungen für die Testversion abhängig vom Lizenzstatus des Benutzers erstellen...

👉 Fügen Sie die folgende Funktion in ein VBA-Modul ein:

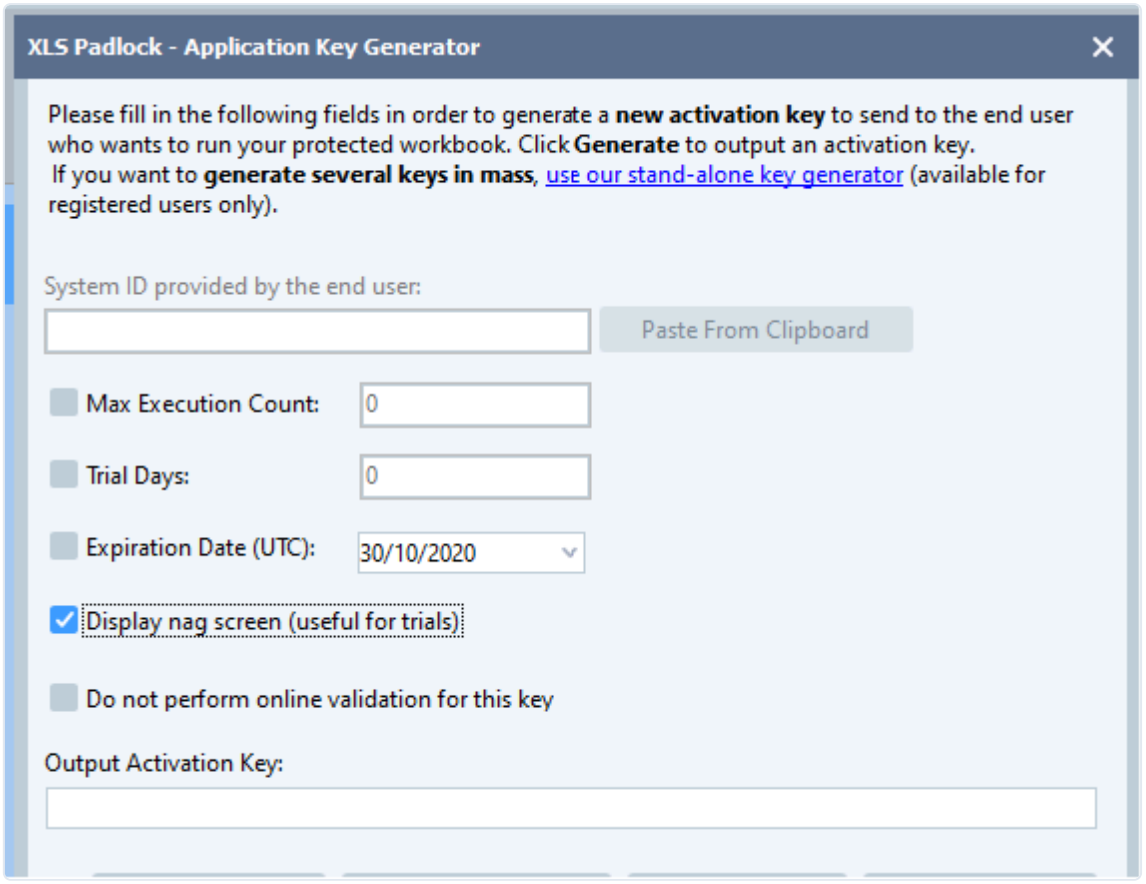
```
Public Function IsTrial()
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    IsTrial = XLSPadlock.PLEvalVar("IsTrial")
    Exit Function
Err:
    IsTrial = False
End Function
```

Anschließend können Sie die Funktion aufrufen:

```
Sub Test_IsTrial()
    If IsTrial() Then
       MsgBox "Trial"
    Else
       MsgBox "Registered"
    End If
End Sub
```

Warnung

Die Funktion gibt nur dann true zurück, wenn der Aktivierungsschlüssel das Kennzeichen "display nag screen" (Testversion) gesetzt hat:



👉 Siehe auch: Einschränkungen für Aktivierungsschlüssel festlegen

Verbleibende Testtage

Mit dieser VBA-Funktion können Sie **programmgesteuert die Anzahl der verbleibenden Tage oder Ausführungen** einer Arbeitsmappe in der Testversion abrufen. Sie funktioniert auch für registrierte Schlüssel mit einem Ablaufdatum oder einer begrenzten Anzahl von Ausführungen.

👉 Fügen Sie die folgende Funktion in ein VBA-Modul ein:

```
Public Function ReadTrialState()
    Dim XLSPadlock As Object
    On Error GoTo Err
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    ReadTrialState = XLSPadlock.PLEvalVar("TrialState")
    Exit Function
Err:
    ReadTrialState = ""
End Function
```

Anschließend können Sie die Funktion aufrufen:

```
Sub Test_Trial()
 rdays = ReadTrialState()
 Worksheets("Sheet1").Range("A1").Value = rdays
End Sub
```

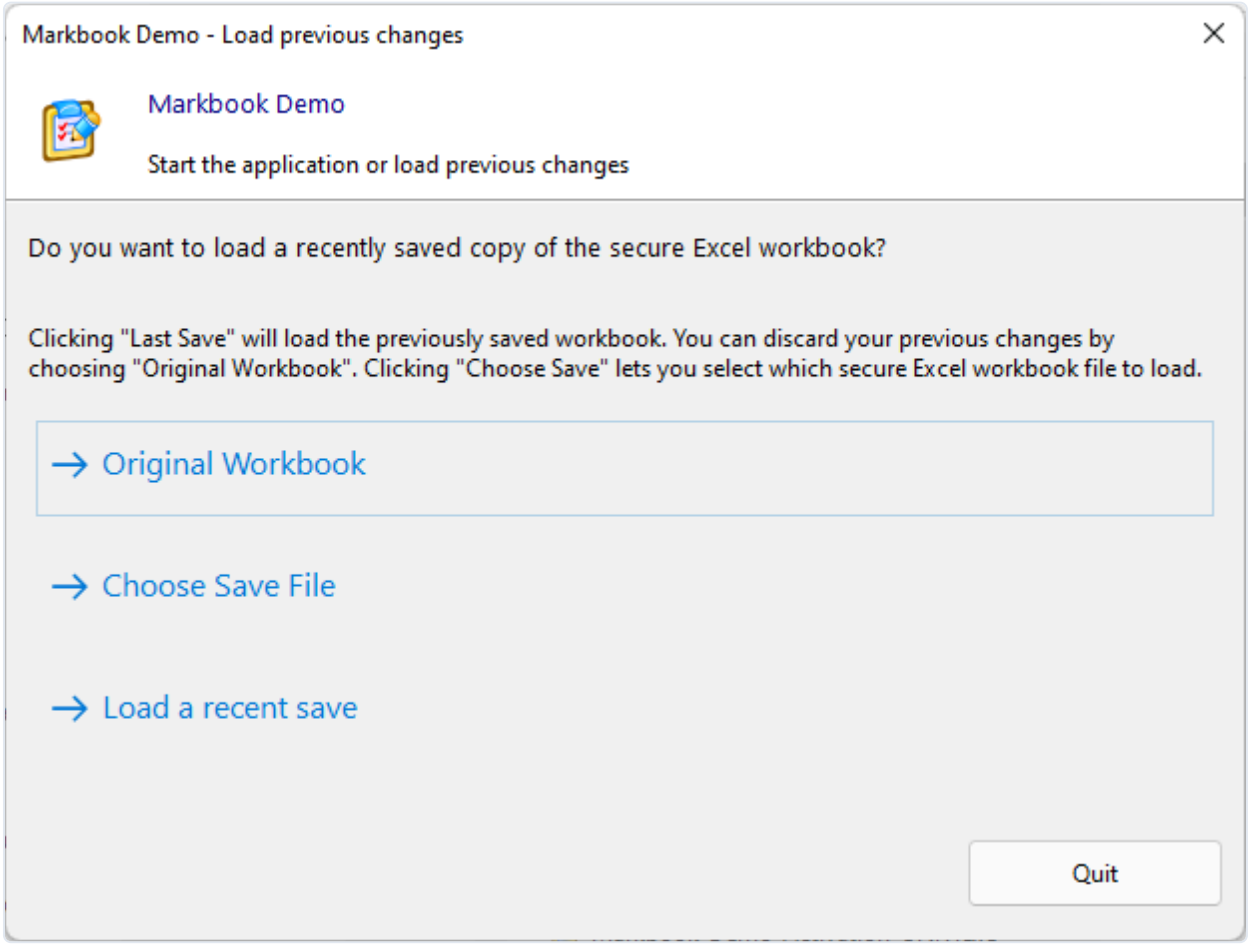
👉 Siehe auch: So erstellen Sie Arbeitsmappen in der Testversion

Speicheroptionen der Arbeitsmappe

Mit diesen Optionen können Sie genau festlegen, wie Änderungen an der Arbeitsmappe gespeichert und geladen werden, wenn Sie das Speichern aktivieren.

Always Show the "Load Previous Changes" Welcome Screen

Wenn keine Speicherdatei vorhanden ist, öffnet die Anwendung standardmäßig direkt die ursprüngliche Arbeitsmappe. Wenn diese Option aktiviert ist, wird zuerst immer das Willkommensdialogfeld angezeigt, sodass Benutzer entweder mit der ursprünglichen Arbeitsmappe beginnen oder eine vorhandene Speicherdatei laden können.



Dies ist nützlich, wenn Sie vorgefertigte Speicherdateien mit Ihrer Anwendung verteilen oder wenn Sie die EXE auf einen anderen Computer verschieben und eine vorhandene Speicherdatei laden möchten.

Save Changes Automatically and Load Them Without Prompt

Diese Option bietet einen nahtlosen, automatischen Speichermechanismus. Benutzeränderungen werden beim Schließen der Anwendung in einer Standarddatei gespeichert und beim nächsten Start automatisch wieder geladen. Es werden keine "Save As"-Dialoge oder Willkommensbildschirme angezeigt.

In diesem Modus können Benutzer nicht mehrere verschiedene Speicherdateien erstellen.

Save Files Can Only Be Opened on the Computer They Were Saved On (Hardware-Locking)

Diese Option erhöht die Sicherheit, indem Speicherdateien an den jeweiligen Computer gebunden werden, auf dem sie erstellt wurden. Eine auf einem PC erstellte Speicherdatei kann nicht auf einem anderen geöffnet werden, wodurch verhindert wird, dass Benutzer ihre gespeicherten Daten weitergeben.

➡️ Mehr über die Hardwarebindung von Speicherdateien erfahren.

Allow Save but Do Not Handle Loading/Saving

Aktivieren Sie diese Option, wenn Sie die integrierten Speicherdialoge von XLS Padlock deaktivieren und den gesamten Speicher- und Ladevorgang selbst mit VBA-Code verwalten möchten. Dies gibt Ihnen die volle Kontrolle, beispielsweise um eine sichere Kopie ohne jegliche Benutzeraufforderung zu speichern.

➡️ Erfahren Sie, wie Sie eine sichere Kopie der Arbeitsmappe ohne Aufforderung mit VBA speichern.

Do Not Display the "Original Workbook" Choice

Diese Option entfernt die Auswahl "Start with the original workbook" vom Willkommensbildschirm und zwingt die Benutzer, eine gespeicherte `.XLSC` - oder `.XLSCE` -Datei zu laden. Dies ist nützlich, wenn Ihre Anwendung immer mit bestimmten Datendateien verwendet werden soll, die Sie bereitstellen.

> **Automatisches Sicherheitsnetz**
>
> Wenn die vorherige Speicherdatei eines Benutzers nicht geladen werden kann, beispielsweise nachdem Sie den Secret Key für eine neue Version geändert oder den Speichermodus zwischen Versionen gewechselt haben, wird die Schaltfläche "Original Workbook" für diesen erneuten Versuch automatisch wieder auf dem Willkommensbildschirm aktiviert. Dies stellt sicher, dass der Benutzer die Anwendung immer wiederherstellen kann, anstatt in einer Schleife von Fehlermeldungen gefangen zu sein.

👉 Siehe auch

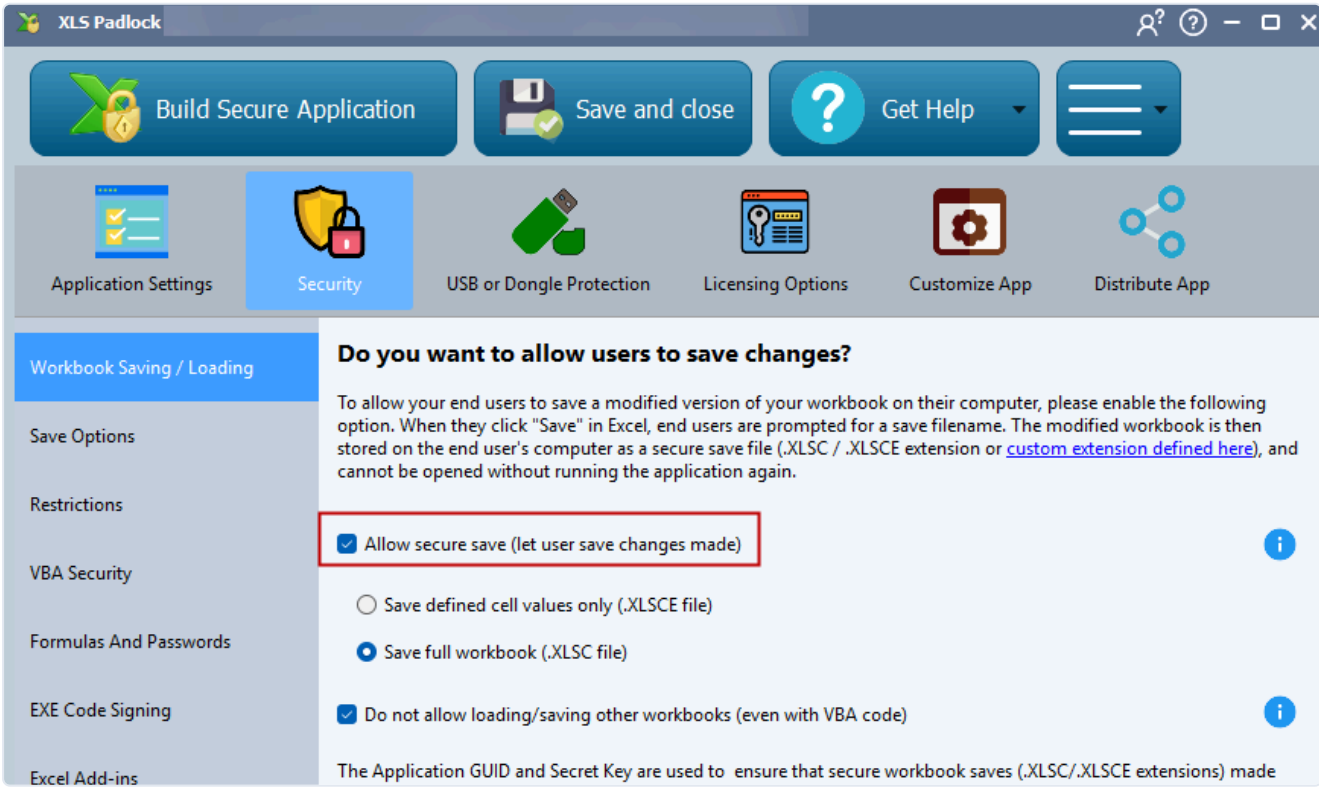
- Überblick über das Speichern und Laden von Arbeitsmappen
- Speichermodus: vollständige Arbeitsmappe vs. Zellwerte
- VBA-Rezepte zum Speichern von Arbeitsmappen

Speichern und Laden

Mit XLS Padlock haben Sie die volle Kontrolle darüber, ob Benutzer Änderungen an Ihrer Excel-Arbeitsmappe speichern können.

Wie XLS Padlock Benutzeränderungen speichert und lädt

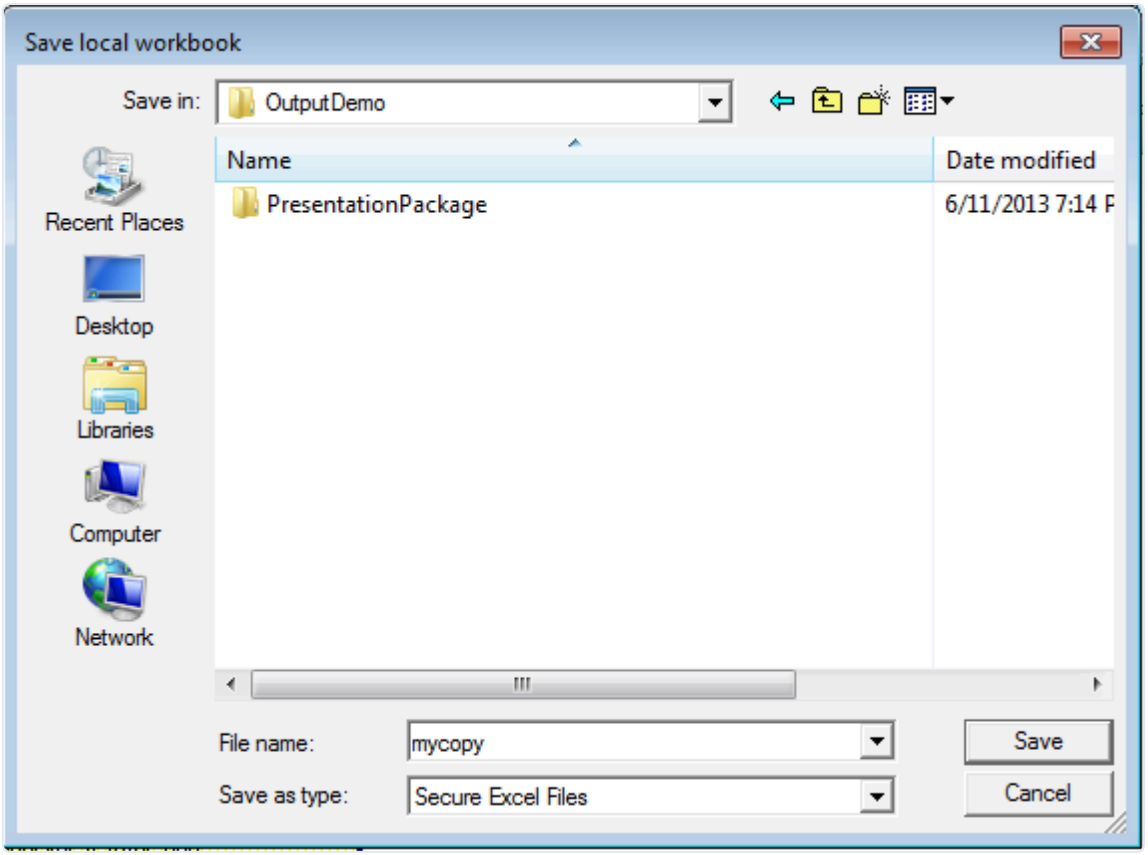
Um das Speichern zu aktivieren oder zu deaktivieren, aktivieren Sie die Option "**Allow secure save**" auf der Seite Security:



- **Wenn diese Option deaktiviert ist**, wird Ihre Anwendung niemals verändert. Alle Änderungen, die ein Benutzer vornimmt, werden verworfen, wenn er die Anwendung schließt. Beachten Sie, dass in einigen Excel-Versionen die Schaltfläche Speichern nicht deaktiviert ist und zu funktionieren scheint, aber tatsächlich keine Änderungen gespeichert werden.
- **Wenn diese Option aktiviert ist**, können Ihre Kunden ihre Änderungen speichern. Sie müssen dann einen Speichermodus auswählen.

Um Änderungen zu speichern, können Endbenutzer auf die standardmäßige Schaltfläche Speichern

in Excel klicken oder das Menü „Datei => Speichern“ verwenden. Das Dialogfeld „Speichern unter“ wird dann angezeigt und fragt sie, wo ihre sichere Kopie der Arbeitsmappe gespeichert werden soll:



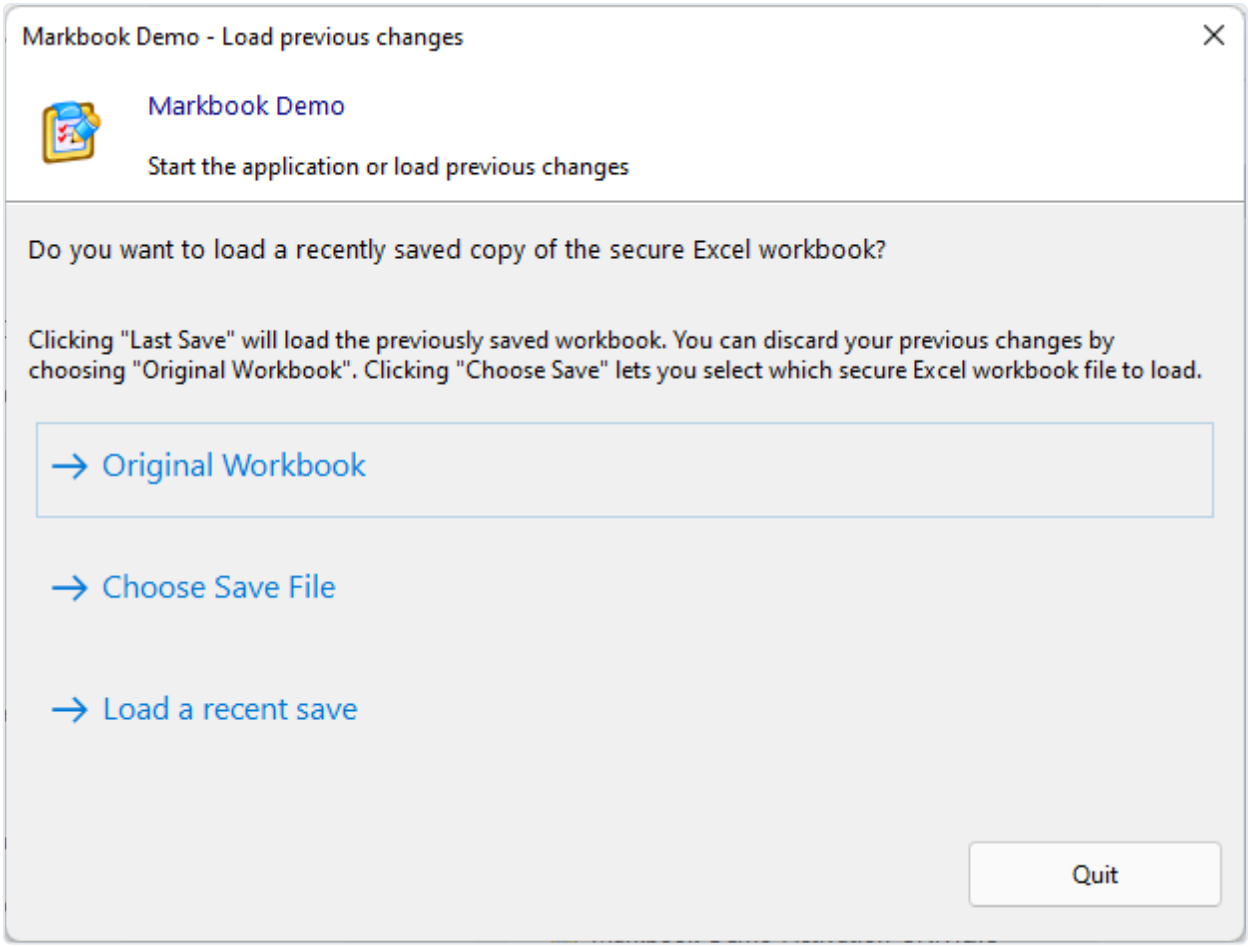
Diese Speicherdatei erhält die Erweiterung `.XLSC` (oder `.XLSCE`) und **kann nicht geöffnet werden, ohne die sichere Anwendung erneut auszuführen**.

Standardmäßig fragt das Dialogfeld die Endbenutzer, ob sie ihre Änderungen überschreiben möchten. Sie können dieses Verhalten in den Erweiterten Optionen deaktivieren.

> **Benutzerdefinierte Erweiterung für Speicherdateien**
>
> Sie können in den Erweiterten Optionen Ihre eigene benutzerdefinierte Erweiterung für Speicherdateien festlegen.

Wie Endbenutzer ihre Änderungen laden

Beim nächsten Ausführen Ihrer Anwendung haben die Kunden die Wahl, Ihre ursprüngliche Arbeitsmappe oder eine zuvor gespeicherte Version zu öffnen. Über den Willkommensbildschirm können sie "Choose Save File" auswählen oder eine Datei aus dem Menü "Recent Saves" laden.



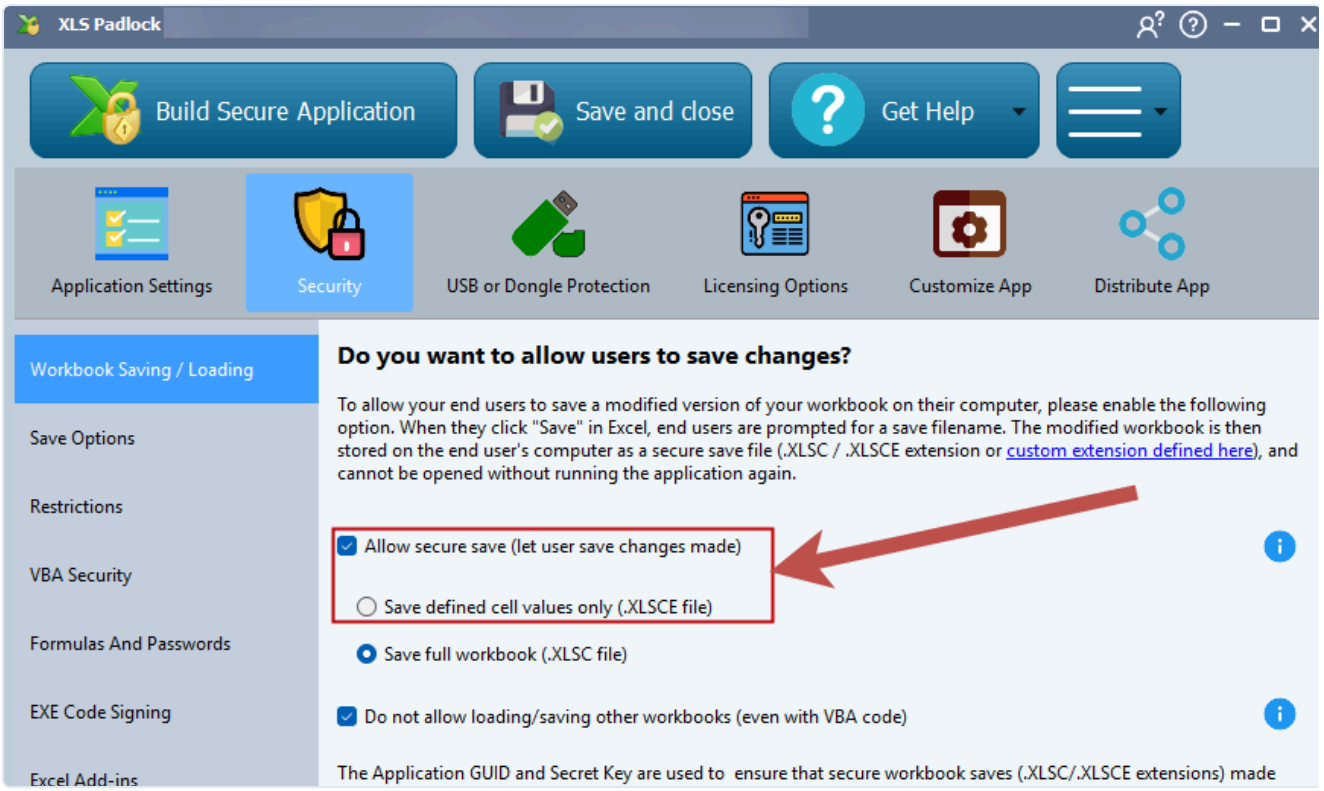
Standardmäßig wird das Dialogfeld „Speichern unter“ jedes Mal angezeigt, wenn ein Benutzer auf die Schaltfläche Speichern klickt. Die zuletzt gespeicherte Datei wird automatisch gemerkt. Beachten Sie, dass nur der Befehl `Save` funktioniert; `Save As` ist immer deaktiviert.

Siehe auch

- Wie Sie Benutzerdaten von einer früheren Version mit VBA migrieren
- Wählen Sie, wie Daten in der Arbeitsmappe gespeichert werden
- Benutzerdefinierte Werte programmgesteuert mit VBA-Code speichern und wiederherstellen

Speichermodus: vollständig oder Zellenwerte

Wenn Sie Benutzern erlauben, ihre Arbeit zu speichern, müssen Sie festlegen, wie XLS Padlock diese Änderungen speichert und lädt. XLS Padlock bietet zwei Speichermodi, die auf der Seite Workbook Saving and Loading ausgewählt werden können.



Vollständige Arbeitsmappe speichern (.XLSC-Datei)

In diesem Modus wird eine vollständige, verschlüsselte Kopie der Arbeitsmappe, einschließlich aller Benutzeränderungen, in einer sicheren `.XLSC`-Datei gespeichert. Diese Datei kann nur von Ihrer geschützten Anwendung geöffnet werden, wodurch die Arbeitsmappe sicher bleibt.

Der **Full Save mode** (vollständiger Speichermodus) verschlüsselt und speichert die gesamte Arbeitsmappe genau so, wie sie zum Zeitpunkt des Speicherns ist. Das bedeutet, dass Benutzer, die eine alte Speicherdatei öffnen, ihre zuvor gespeicherte Arbeit sehen und nicht Ihre neuen Aktualisierungen, falls Sie später eine aktualisierte EXE verteilen. Ihre Speicherdatei ist eine vollständige Momentaufnahme der Arbeitsmappe zu jenem Zeitpunkt.

Vollständige Speicherdateien können zwischen Benutzern weitergegeben werden, sofern Sie sie nicht an eine bestimmte Maschine binden. Sie können diese Speicherdateien auch selbst entschlüsseln, um Benutzerdaten wiederherzustellen.

Wann der Full Save mode zu verwenden ist

Der Full Save mode ist die Standardwahl für die meisten Benutzer und wird besonders für komplexe Arbeitsmappen empfohlen, bei denen Benutzer umfangreiche Änderungen über mehrere Tabellenblätter hinweg vornehmen.

Der Hauptnachteil besteht darin, dass Speicherdateien von Benutzern nicht automatisch mit aktualisierten Versionen Ihrer Anwendung kompatibel sind. Wenn Sie häufig Aktualisierungen der Logik oder des Designs Ihrer Arbeitsmappe veröffentlichen, sollten Sie stattdessen den Cell Values mode (Zellwerte-Modus) in Betracht ziehen.

👉 Siehe auch: So migrieren Sie Benutzerdaten aus einer früheren Version mit VBA

Nur definierte Zellwerte speichern (.XLSCE-Datei)

In diesem Modus werden nur die Werte bestimmter, vordefinierter Zellen gespeichert und wiederhergestellt. Dies ist die ideale Wahl, wenn Sie Ihre Quellarbeitsmappe häufig aktualisieren und neue Versionen verteilen.

Wenn ein Benutzer seine `.XLSCE`-Speicherdatei mit Ihrer neuen EXE lädt, werden seine gespeicherten Daten in Ihre aktualisierte Arbeitsmappenstruktur geladen. So kann er von Ihren Aktualisierungen profitieren, ohne seine Arbeit zu verlieren.

Der Hauptnachteil besteht darin, dass Sie vor dem Kompilieren Ihrer Anwendung manuell festlegen müssen, welche Zellen gespeichert werden sollen.

Wie `.XLSC`-Dateien sind auch `.XLSCE`-Dateien sicher verschlüsselt und können nur von Ihrer Anwendung geöffnet werden. Dies wird durch einen eindeutigen geheimen Schlüssel durchgesetzt, der spezifisch für Ihr Projekt ist.

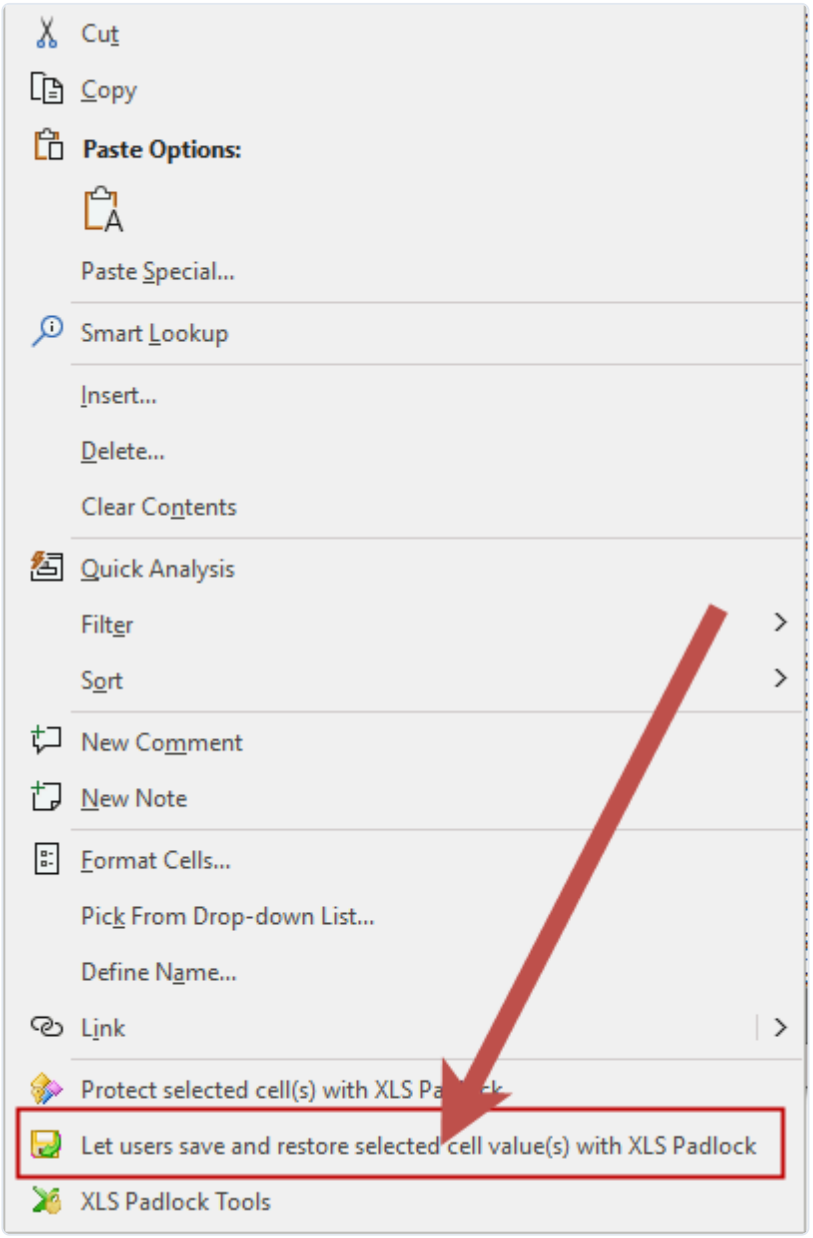
Wählen Sie Ihren Speichermodus von Anfang an

Die Formate Full Save (`.XLSC`) und Cell Values (`.XLSCE`) sind nicht austauschbar. Sobald Sie eine Anwendung mit einem Modus verteilt haben, bedeutet ein Wechsel zum anderen Modus in einer nachfolgenden Version, dass die vorhandenen Speicherdateien der Benutzer nicht geladen werden. XLS Padlock greift für diese Benutzer beim ersten Start der neuen Version stillschweigend auf die ursprünglich eingebettete Arbeitsmappe zurück, und alle Daten, die sie im vorherigen Modus gespeichert hatten, gehen verloren. Wählen Sie den Modus, der zu Ihrer Aktualisierungsstrategie passt, zu Beginn des Projekts und behalten Sie ihn bei.

Zu speichernde und wiederherzustellende Zellen festlegen

Wenn Sie den Modus **Save defined cell values only** verwenden, müssen Sie angeben, welche Zellen Ihre Anwendung speichern und wiederherstellen wird.

👉 Klicken Sie dazu einfach **mit der rechten Maustaste auf eine oder mehrere Zellen** und wählen Sie im Kontextmenü "**Let users save and restore selected cell value(s) with XLS Padlock**".

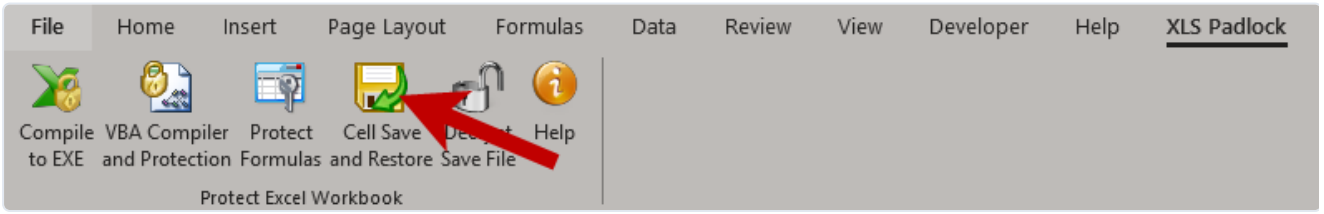


XLS Padlock bestätigt, dass die ausgewählten Zellen zum Speichern und Wiederherstellen markiert sind.

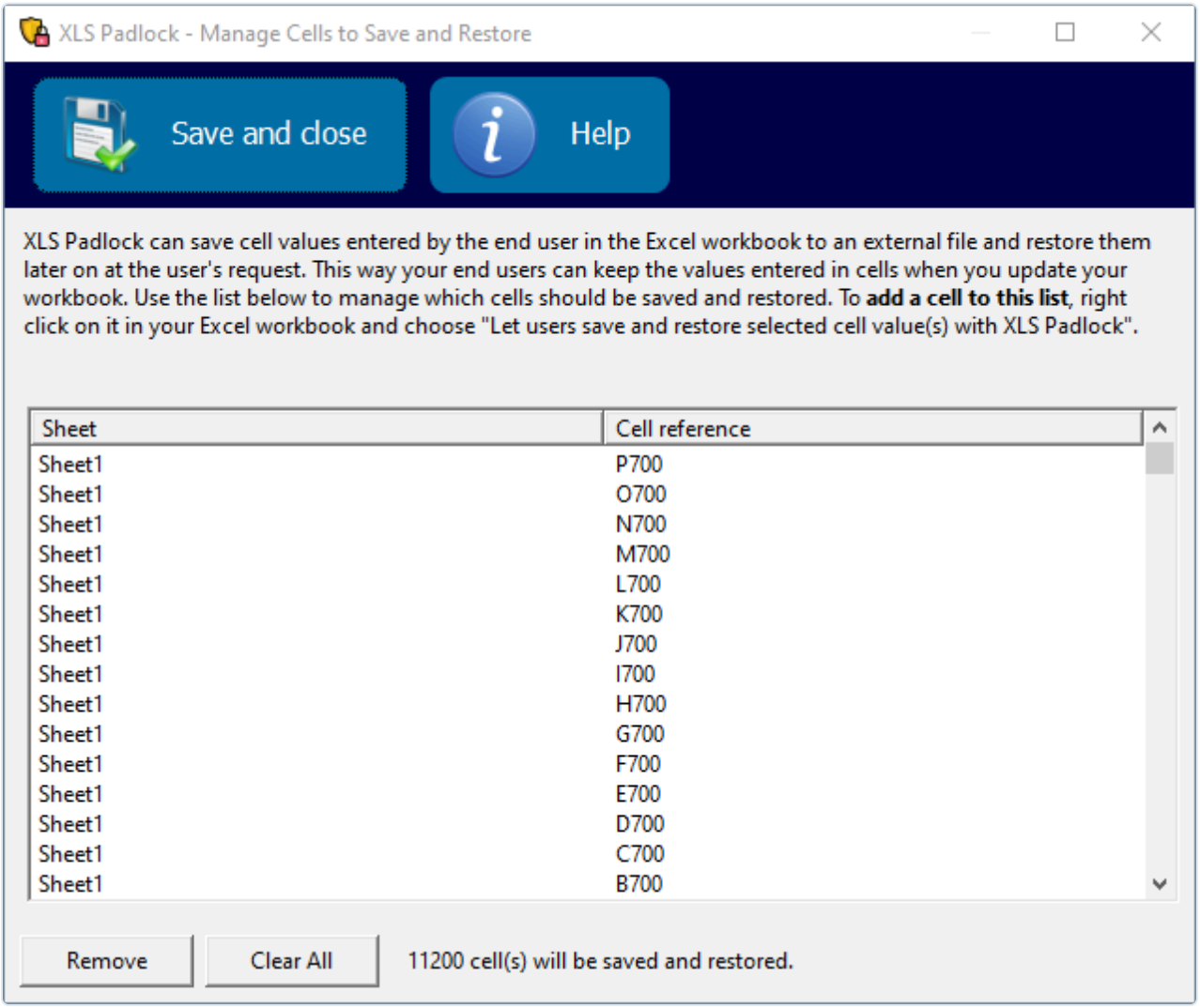
Sie können auch benutzerdefinierte Werte mit VBA-Code speichern und wiederherstellen.

Übersicht der zu speichernden Zellen

Um alle für das Speichern konfigurierten Zellen anzuzeigen, klicken Sie auf "**Cell Save And Restore**" in der XLS Padlock-Registerkarte im Menüband:



Dadurch wird ein Fenster geöffnet, das alle konfigurierten Zellen auflistet. Von hier aus können Sie einzelne Zellen entfernen oder die gesamte Liste löschen.



Mit VBA wiederherstellen und speichern

Im Modus **Save defined cell values only** können Sie mit XLS Padlock zusätzlich zu den vordefinierten Zellwerten auch benutzerdefinierte Werte speichern und wiederherstellen. Dies ist nützlich, um Variablen, Einstellungen oder Daten zu speichern, die nicht direkt in einer Zelle abgelegt sind.

Überblick

Diese Funktionalität beruht auf zwei VBA-Ereignissen und zwei VBA-API-Funktionen. Sie platzieren Code in den Ereignis-Unterroutinen, um Ihre benutzerdefinierten Daten zu lesen oder zu schreiben, wenn der Benutzer seine Arbeit lädt oder speichert.

Wichtigste Ereignisse und Funktionen

VBA-Ereignisse

Die folgenden beiden Ereignis-Unterroutinen müssen in einem Modul Ihrer Excel-Arbeitsmappe platziert werden. XLS Padlock ruft sie während des Speicher- und Ladevorgangs automatisch auf.

```
' Called when a user loads a save file.
Sub XLSPadlock_RestoreCustomValues()
    ' Your code to read values goes here.
    MsgBox ("Restoring custom values...")
End Sub

' Called when a user saves their work.
Sub XLSPadlock_SaveCustomValues()
    ' Your code to write values goes here.
    MsgBox ("Saving custom values...")
End Sub
```

VBA-API-Funktionen

- **WriteCustomCellValue(UniqueID, Value)**: Schreibt einen einzelnen Zeichenfolgenwert, der einer eindeutigen ID zugeordnet ist.
- **ReadCustomCellValue(UniqueID, DefaultValue)**: Liest einen einzelnen Zeichenfolgenwert für eine bestimmte ID. Wird die ID nicht gefunden, gibt die Funktion den `DefaultValue` zurück.

> **Wichtig**
>
> `WriteCustomCellValue` und `ReadCustomCellValue` funktionieren nur, wenn sie aus den Ereignissen `XLSPadlock_SaveCustomValues` bzw. `XLSPadlock_RestoreCustomValues` aufgerufen werden.

Einzelne Werte lesen und schreiben

So schreiben Sie einen einzelnen Wert:

```
Sub XLSPadlock_SaveCustomValues()
    Dim XLSPadlock1 As Object
    On Error Resume Next
    Set XLSPadlock1 = Application.COMAddIns("GXLS.GXLSPLock").Object
    XLSPadlock1.WriteCustomCellValue "MySetting", "MyValue"
End Sub
```

Und so lesen Sie ihn wieder ein:

```
Sub XLSPadlock_RestoreCustomValues()
    Dim XLSPadlock1 As Object
    On Error Resume Next
    Set XLSPadlock1 = Application.COMAddIns("GXLS.GXLSPLock").Object

    Dim MySettingValue As String
    MySettingValue = XLSPadlock1.ReadCustomCellValue("MySetting", "Default")
End Sub
```

Alle Werte als Dictionary lesen

Sie können auch alle gespeicherten benutzerdefinierten Werte auf einmal lesen, indem Sie eine leere Zeichenfolge als ID an `ReadCustomCellValue` übergeben. Dies gibt ein `Scripting.Dictionary`-Objekt zurück.

> **Verweis erforderlich**
>
> Um diese Funktion zu nutzen, müssen Sie in Ihrem VBA-Projekt einen Verweis auf **„Microsoft Scripting Runtime“** hinzufügen (im VBE-Menü zu Tools -> References gehen).

```vb Sub XLSPadlock_RestoreCustomValues() Dim XLSPadlock1 As Object On Error Resume Next Set XLSPadlock1 = Application.COMAddIns("GXLS.GXLSPLock").Object

```
Dim Dict As Object ' Scripting.Dictionary
Set Dict = XLSPadlock1.ReadCustomCellValue("", "")

If Not Dict Is Nothing Then
    For Each Key In Dict.Keys
        MsgBox "Key: " & Key & ", Value: " & Dict(Key)
    Next Key
End If
```

End Sub
```

````
## Vollständiges Beispiel: Speichern/Wiederherstellen einer Spalte

<div class="aside aside-note">

Der gesamte folgende Code muss in einem einzigen Modul platziert werden.

</div>

Diese Hilfsfunktion erzeugt aus den Werten eines bestimmten Zellbereichs eine durch Kommas getr

```vb
Function CsvRange(myRange As Range) As String
    Dim csvRangeOutput As String
    Dim entry As Variant
    For Each entry In myRange
        If Not IsEmpty(entry.Value) Then
            csvRangeOutput = csvRangeOutput & entry.Value & ","
        End If
    Next
    If Len(csvRangeOutput) > 0 Then
        CsvRange = Left(csvRangeOutput, Len(csvRangeOutput) - 1)
    End If
End Function
````

Dieses Ereignis wird aufgerufen, wenn der Benutzer seine Arbeit speichert. Es verwendet die Hilfsfunktion, um den gesamten verwendeten Bereich der Spalte A in eine einzige Zeichenfolge umzuwandeln und diese zu speichern.

```
Sub XLSPadlock_SaveCustomValues()
    Dim XLSPadlock1 As Object
    On Error Resume Next
    Set XLSPadlock1 = Application.COMAddIns("GXLS.GXLSPLock").Object

    Dim rng As Range
    Set rng = ThisWorkbook.Worksheets(2).Range("A1").CurrentRegion

    Dim myString As String
    myString = CsvRange(rng)

    XLSPadlock1.WriteCustomCellValue "MyEntireColumnA", myString
End Sub
```

Dieses Ereignis wird aufgerufen, wenn der Benutzer eine Speicherdatei lädt. Es liest die Zeichenfolge und stellt die Werte in Spalte A wieder her.
```

```
Sub XLSPadlock_RestoreCustomValues()
    Dim XLSPadlock1 As Object
    On Error Resume Next
    Set XLSPadlock1 = Application.COMAddIns("GXLS.GXLSPLock").Object

    Dim Val As String
    Val = XLSPadlock1.ReadCustomCellValue("MyEntireColumnA", "")

    If Val <> "" Then
        Dim r As Range, i As Long, ar
        Set r = ThisWorkbook.Worksheets(2).Range("A:A")
        r.ClearContents

        ar = Split(Val, ",")
        For i = 0 To UBound(ar)
         r.Cells(i + 1, 1).Value = ar(i)
        Next
    End If
End Sub

## Siehe auch

- [Wie Sie Benutzerdaten aus einer früheren Version migrieren](#chapter-excel-vba-migrate-user-
```

Zugriff auf gesicherte Arbeitsmappe und Begleitdateien

HINWEIS

Dieses Handbuch erklärt, wie Sie aus Ihrer geschützten Arbeitsmappe eine neue Excel-Instanz starten und ihr Zugriff auf die sichere Arbeitsmappe sowie ihre Begleitdateien gewähren.

Standardmäßig kann nur die von Ihrer kompilierten EXE gestartete Excel-Instanz auf die sichere Arbeitsmappe und ihre zugehörigen Dateien zugreifen. Wenn Sie mit VBA eine neue Excel-Instanz erstellen, hat diese keine Berechtigung, diese geschützten Dateien zu öffnen.

Um den Zugriff zu gewähren, müssen Sie die XLS Padlock API verwenden, um das Fensterhandle (HWND) der neuen Excel-Instanz an die Hauptanwendung zu übergeben. Dies autorisiert die neue Instanz, mit dem virtuellen Dateisystem zu arbeiten.

```
Sub RunInSeparateExcelInstance()
    Dim XLSPadlock As Object
    Dim appExcel As Object ' Excel.Application
    Dim wbExcel As Object  ' Excel.Workbook
    Dim companionFilePath As String

    ' Use LongPtr for 64-bit compatibility when getting the window handle.
    #If VBA7 Then
        Dim windowHandle As LongPtr
    #Else
        Dim windowHandle As Long
    #End If

    On Error GoTo Cleanup

    ' Get the path to a companion file located in the same folder as the EXE.
    companionFilePath = GetPathToFileInEXEFolder("data.xlsx")
    If companionFilePath = "" Then
        MsgBox "Companion file not found.", vbExclamation
        Exit Sub
    End If

    ' Get the XLS Padlock API object.
    Set XLSPadlock = Application.COMAddIns("GXLS.GXLSPLock").Object
    If XLSPadlock Is Nothing Then
        MsgBox "XLS Padlock API not available. Is the workbook protected?", vbCritical
        Exit Sub
    End If

    ' Start a new, invisible Excel instance.
    Set appExcel = CreateObject("Excel.Application")

    ' Get the window handle (HWND) of the new Excel instance.
    windowHandle = appExcel.Application.Hwnd

    ' Authorize the new instance by passing its HWND to the main application.
    ' Option "5" is used for this purpose.
    XLSPadlock.SetOption Option:="5", Value:=windowHandle

    ' The new Excel instance can now access the secure companion file.
    Set wbExcel = appExcel.Workbooks.Open(companionFilePath, False, True)

    ' Example: Read a value from the companion workbook.
    MsgBox "Value from companion file: " & wbExcel.Worksheets("Sheet1").Cells(1, 1).Value

Cleanup:
    If Err.Number <> 0 Then
        MsgBox "An error occurred: " & Err.Description, vbCritical
    End If

    ' Close the workbook and quit the new Excel instance.
    If Not wbExcel Is Nothing Then wbExcel.Close SaveChanges:=False
```

```
    If Not appExcel Is Nothing Then appExcel.Quit

    ' Release object variables.
    Set wbExcel = Nothing
    Set appExcel = Nothing
    Set XLSPadlock = Nothing
End Sub

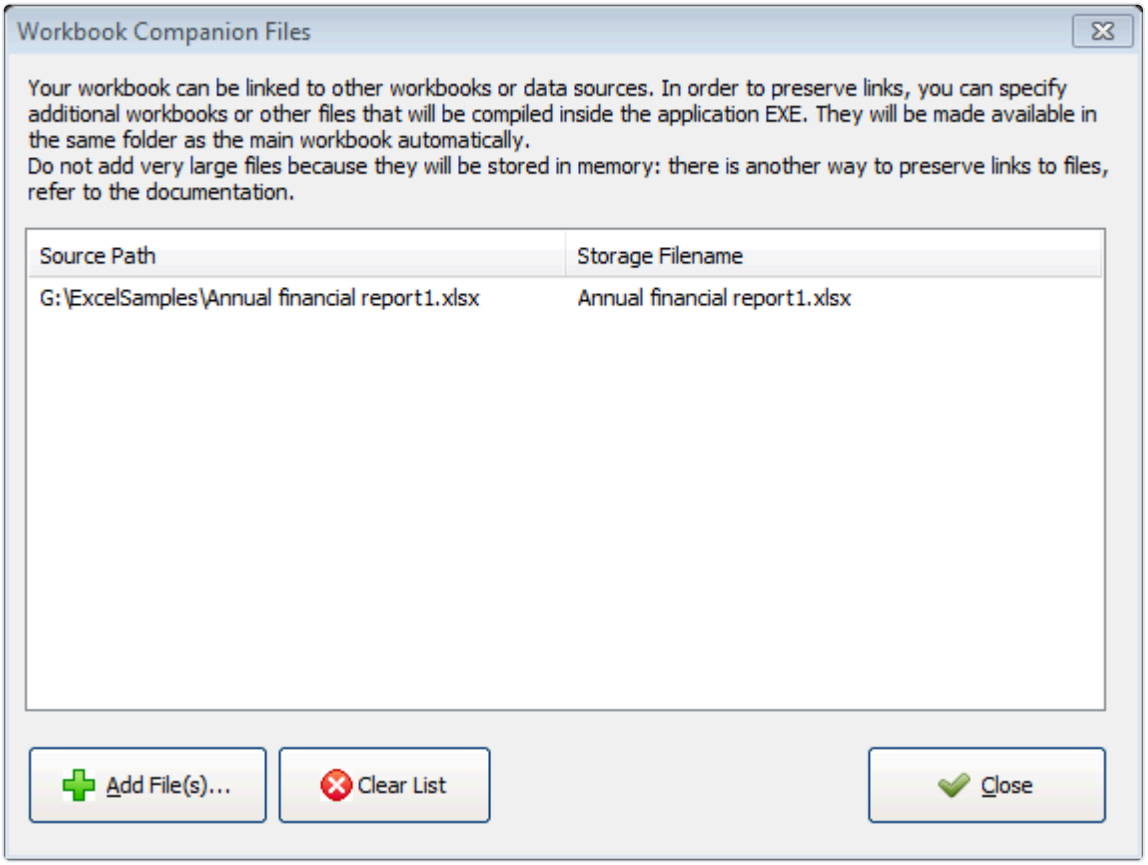
Public Function GetPathToFileInEXEFolder(ByVal Filename As String) As String
    ' Helper function to get the full path to a file in the EXE's directory.
    ' See: https://www.xlspadlock.com/doc/get-the-path-to-a-file-in-the-same-folder-as-the-comp
    Dim XLSPadlock As Object
    On Error Resume Next
    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    If Not XLSPadlock Is Nothing Then
        GetPathToFileInEXEFolder = Application.BuildPath(XLSPadlock.PLEvalVar("EXEPath"), Filer
    End If
End Function
```

Begleitdateien hinzufügen

Wenn Ihre Arbeitsmappe mit anderen Arbeitsmappen oder Datenquellen verknüpft ist, können Sie diese als "Begleitdateien" (companion files) hinzufügen. Dadurch werden sie direkt in die EXE der Hauptanwendung eingebettet, was die Verteilung erleichtert. Wenn die Anwendung ausgeführt wird, werden diese Dateien im selben virtuellen Ordner wie die Hauptarbeitsmappe verfügbar gemacht.

Sie können sogar XLL-Add-ins einbeziehen.

Wenn Ihre Arbeitsmappe beispielsweise eine externe Textdatei als Datenquelle verwendet, stellt das Hinzufügen als Begleitdatei sicher, dass sie immer gefunden wird, wenn die kompilierte Arbeitsmappe auf einem beliebigen Computer geöffnet wird.



Um Dateien hinzuzufügen, klicken Sie auf **Add Files** (Dateien hinzufügen) und wählen Sie sie aus. Sie erscheinen dann in der Liste. Die Spalte "Storage Filename" zeigt den Dateinamen an, der verwendet wird, wenn die Datei in die EXE kompiliert und zur Laufzeit wiederhergestellt wird.

Vermeiden Sie große Dateien

Fügen Sie keine sehr großen Dateien hinzu, da sie vollständig im Arbeitsspeicher abgelegt werden. Um auf andere große Arbeitsmappen oder Datendateien zu verweisen, ist es besser, sie im selben Ordner wie die EXE Ihrer Anwendung abzulegen und mit der Funktion `PLEvalVar("XLSPath")` zu referenzieren. Weitere Einzelheiten finden Sie unter Externe Referenzen und Hyperlinks verwenden.

Auf Begleitdateien mit VBA zugreifen

Um mit VBA auf Begleitdateien zuzugreifen, können Sie die folgende Funktion verwenden:

```
Public Function PathToCompiledFile(Filename As String) As String
    Dim XLSPadlock As Object
    On Error GoTo Err

    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    PathToCompiledFile = XLSPadlock.PLEvalVar("XLSPath") & Filename

    Exit Function
Err:
    PathToCompiledFile = ""
End Function
```

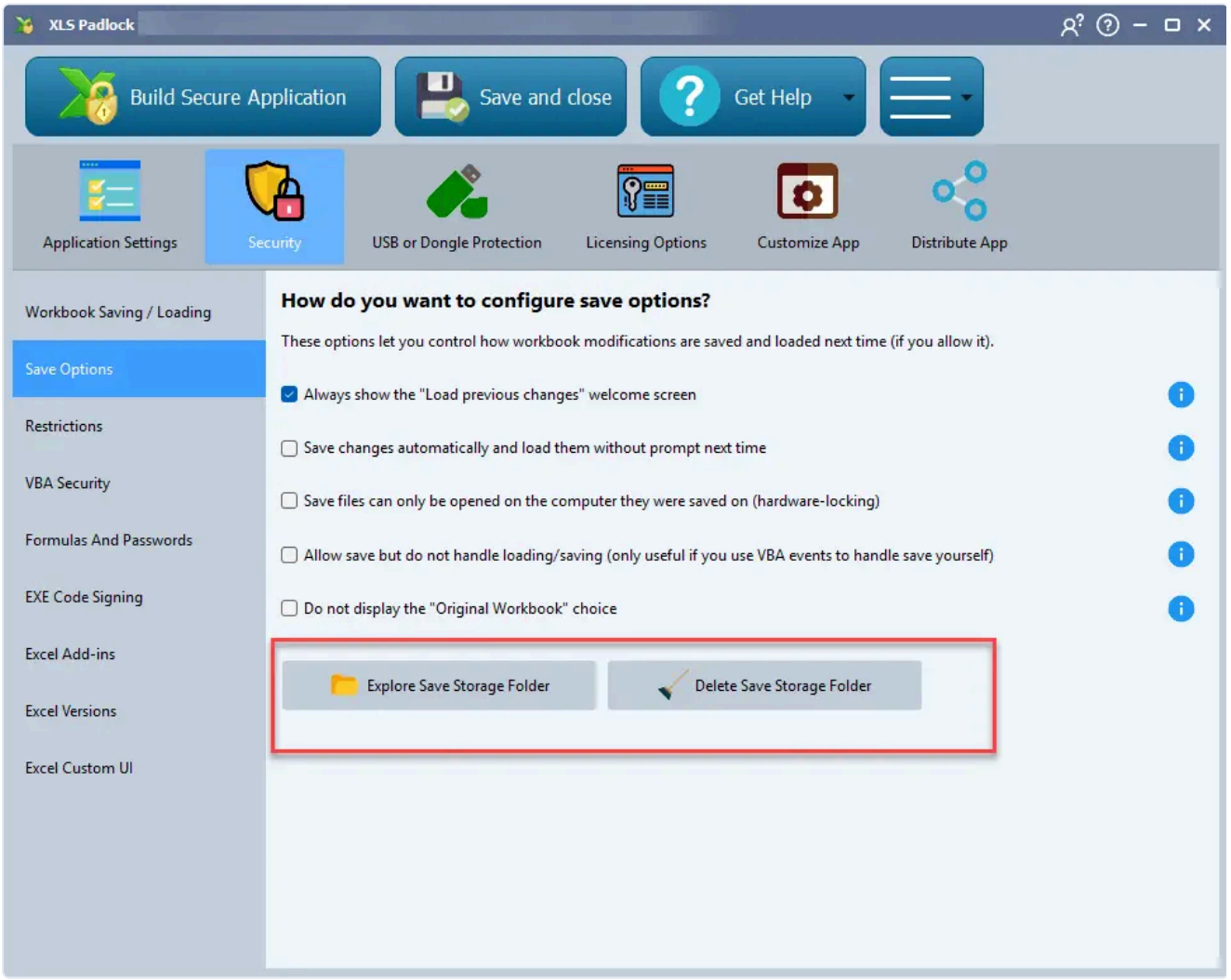
Anschließend können Sie diese Funktion verwenden, um eine Begleitdatei zu öffnen:

```
Sub ExampleUsage()
    Dim wk As Workbook
    Dim filePath As String

    filePath = PathToCompiledFile("Test File.xlsx")
    If filePath <> "" Then
        Set wk = Workbooks.Open(filePath, False, False)
        MsgBox wk.Sheets(1).Cells(1, 1).Value
        wk.Close
    Else
        MsgBox "Companion file not found!"
    End If
End Sub
```

Speicherordner für Speicherungen

Die Schaltflächen **Explore Save Storage Folder** und **Delete Save Storage Folder** bieten schnellen Zugriff auf den lokalen Ordner auf Ihrem PC, in dem Ihre geschützte Anwendung ihre Daten speichert.



Dieser Ordner enthält: automatisch gespeicherte Dateien (sofern diese Funktion aktiviert ist), die Verlaufsdatei der Sicherungen sowie die Lizenz- und Aktivierungsdaten.

Wenn der portable Modus aktiviert ist (in den Erweiterten Optionen), öffnen diese Schaltflächen denselben Ordner, der die EXE-Datei der Anwendung enthält. Aus Sicherheitsgründen ist die Schaltfläche "Delete" in diesem Modus deaktiviert.

Änderungen in der EXE speichern

Nein, Benutzer können Änderungen nicht direkt in der kompilierten `.exe`-Datei speichern. Das Ändern einer ausführbaren Datei nach der Kompilierung kann sie beschädigen und löst häufig Fehlalarme von Antivirensoftware aus.

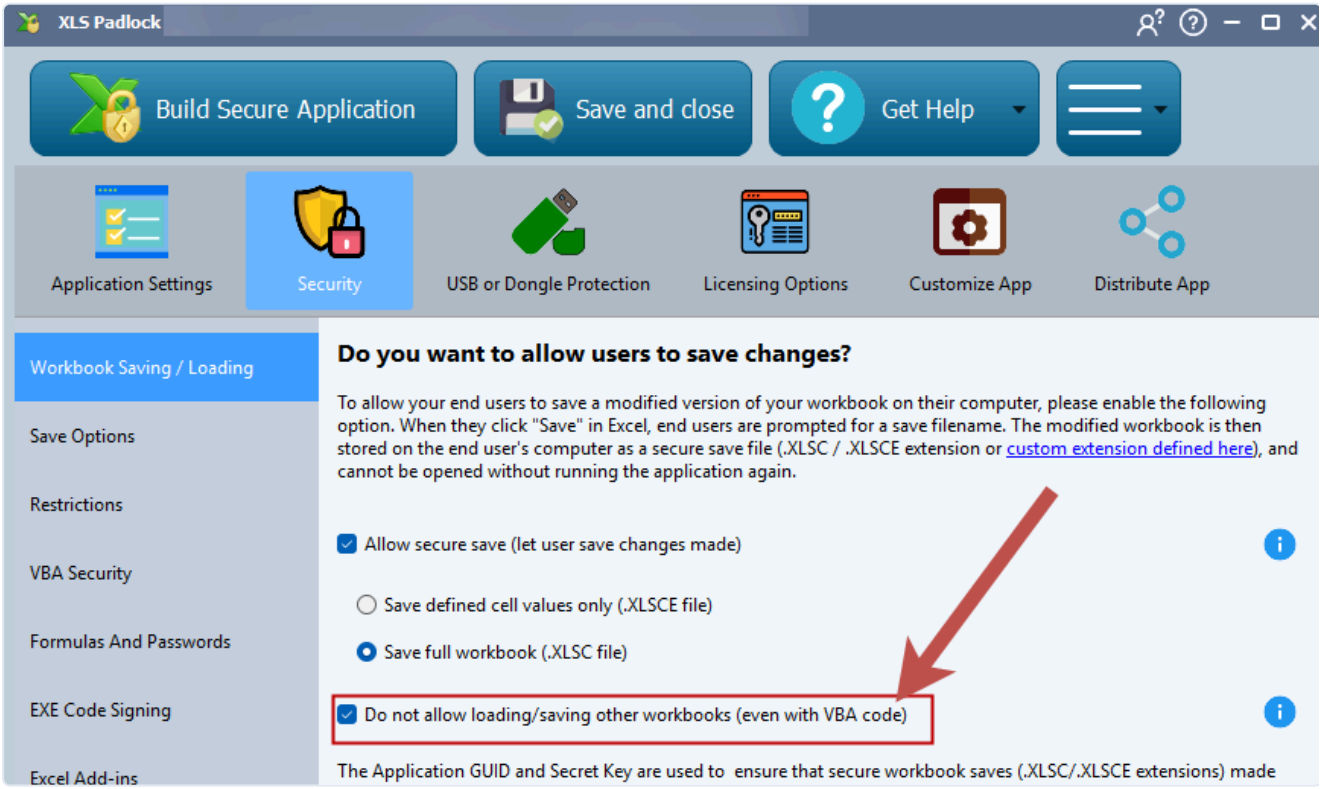
Aus diesem Grund ist XLS Padlock so konzipiert, dass Benutzerdaten in separaten verschlüsselten Speicherdateien (mit den Erweiterungen `.xlsc` oder `.xlsce`) abgelegt werden und nicht zurück in die Haupt-EXE der Anwendung.

Betrachten Sie Ihre kompilierte EXE als die Anwendung selbst (wie Excel.exe) und die Speicherdateien als die Dokumente, mit denen sie arbeitet.

Laden und Speichern einschränken

Eine mögliche Schwachstelle besteht darin, dass Excel-Arbeitsmappen mithilfe von VBA- oder OLE-Befehlen auf der Festplatte gespeichert werden können. Um diese Methode der Datenextraktion zu verhindern, enthält XLS Padlock eine leistungsstarke Sicherheitsoption mit dem Namen **"Do not allow loading/saving other workbooks"** (Das Laden/Speichern anderer Arbeitsmappen verbieten).

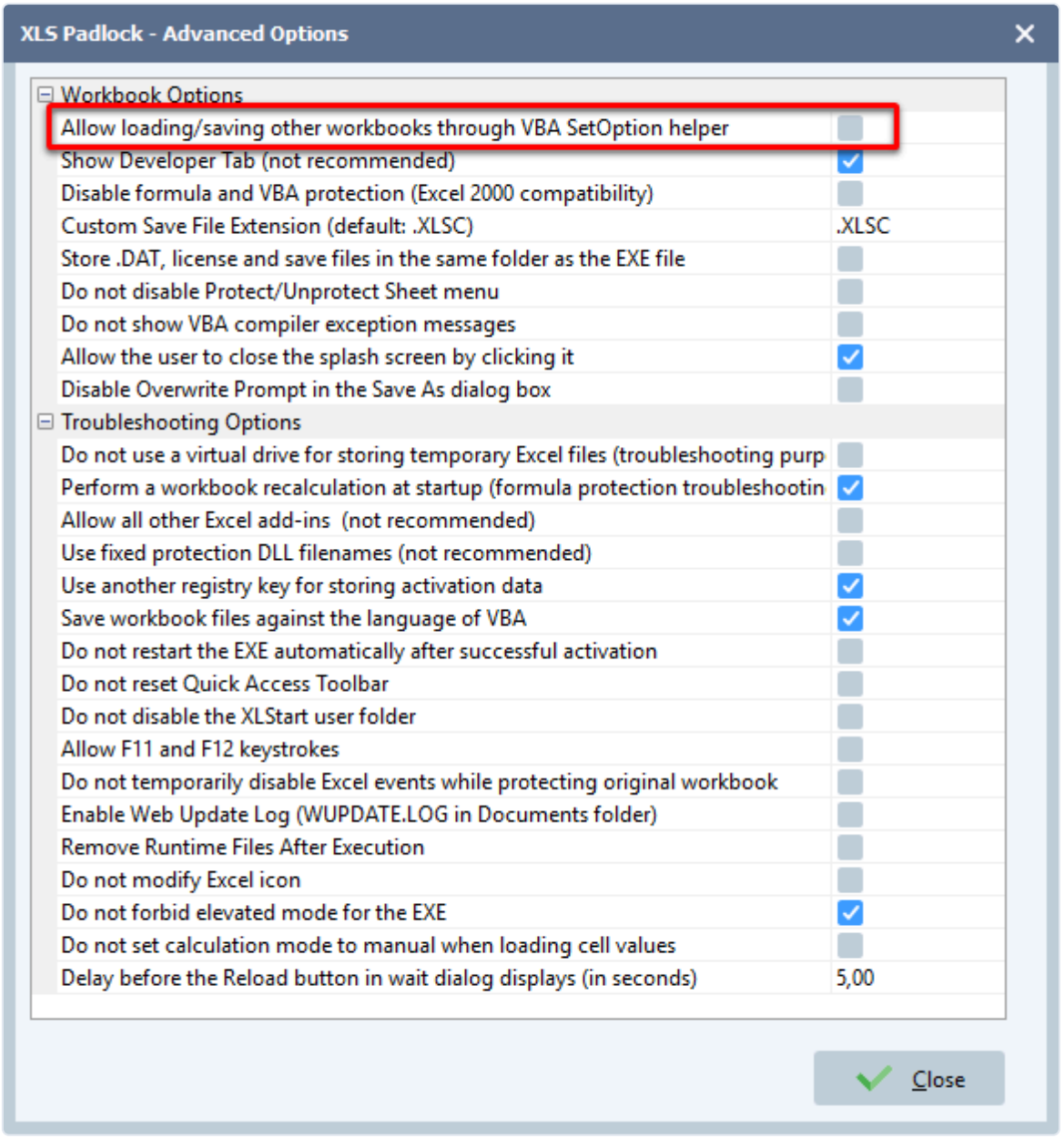
Wenn diese Funktion aktiviert ist, sorgt sie dafür, dass die Excel-Instanz, auf der Ihre Anwendung läuft, **keine andere Arbeitsmappendatei laden oder speichern kann**. Dies hindert Benutzer wirksam daran, andere Arbeitsmappen zu öffnen, und verhindert VBA-basierte Hacks, die versuchen, Daten in eine neue, ungeschützte Datei zu kopieren.



Was, wenn meine Anwendung andere Arbeitsmappen speichern muss? Die SetHelper-VBA-Lösung

Diese Sicherheitsfunktion blockiert auch Standard-VBA-Code zum Speichern oder Laden von Arbeitsmappen. Wenn Ihre Anwendung diese Aktionen rechtmäßig ausführen muss, steht eine Umgehungslösung zur Verfügung.

Sie können die Einschränkung in Ihrem VBA-Code vorübergehend deaktivieren. Dazu müssen Sie zunächst die erweiterte Option **"Allow loading/saving other workbooks through VBA SetOption helper"** (Das Laden/Speichern anderer Arbeitsmappen über den VBA-SetOption-Helper erlauben) hier aktivieren:



Anschließend können Sie den folgenden VBA-Code verwenden:

```
Dim XLSPadlock As Object
Set XLSPadlock = Application.COMAddIns("GXLS.GXLSPLock").Object

' Temporarily disable the security
XLSPadlock.SetOption Option:= "2", Value:= "0"
XLSPadlock.SetOption Option:= "1", Value:= "1" ' Also disable encrypted save prompt

' Your code to save a normal workbook
ActiveWorkbook.SaveAs "D:\My Documents\NormalWorkbook.xlsx"

' Re-enable the security
XLSPadlock.SetOption Option:= "2", Value:= "1"
XLSPadlock.SetOption Option:= "1", Value:= "0"
```

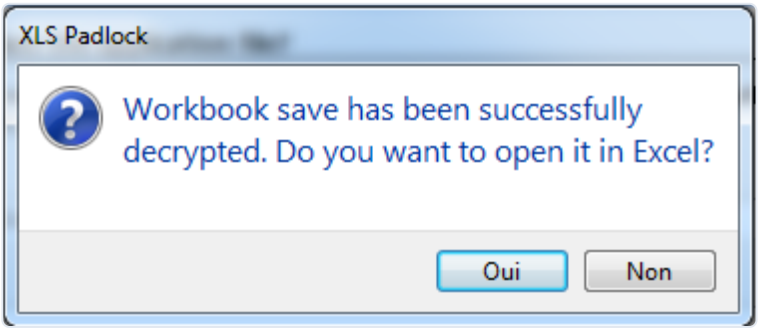
Speicherungen öffnen und entschlüsseln

Die von Ihrer Anwendung erstellten Speicherdateien sind verschlüsselt und können nicht direkt in Excel geöffnet werden. Als ursprünglicher Autor der Arbeitsmappe **können Sie jedoch jede von Ihrer Anwendung generierte Speicherdatei entschlüsseln**. Das ist nützlich, um Daten wiederherzustellen, wenn ein Kunde Ihnen seine Speicherdatei zusendet.

Um eine Speicherdatei (`.xlsc` oder `.xlsce`) zu öffnen, wählen Sie im XLS Padlock-Menüband in Excel den Befehl "**Decrypt Save File**" (Speicherdatei entschlüsseln):

Entschlüsselung von `.xlsc`-Dateien (Modus vollständige Speicherung)

Wenn Sie eine `.xlsc`-Datei auswählen, die mit dem Modus vollständige Speicherung erstellt wurde, entschlüsselt XLS Padlock die gesamte Arbeitsmappe und öffnet sie.



Einschränkungen der Entschlüsselung

- Mit XLS Padlock geschützte Formeln können beim Entschlüsseln einer Speicherdatei nicht wiederhergestellt werden. Die entschlüsselte Arbeitsmappe ist möglicherweise nicht vollständig funktionsfähig. Diese Funktion sollte nur verwendet werden, um vom Benutzer eingegebene Daten wiederherzustellen.
- Dieselbe Einschränkung gilt für das VBA-Projekt, wenn Sie es gesperrt oder kompiliert haben.
- Um die Arbeitsmappe mit allen Funktionen intakt zu verwenden, sollten Sie Ihre sichere Anwendung ausführen und die Speicherdatei normal laden.

Entschlüsselung von `.xlsce`-Dateien (Modus Zellwerte)

Wenn Sie eine `.xlsce`-Datei auswählen, die mit dem Speichermodus für Zellwerte erstellt wurde, fragt XLS Padlock, ob Sie die Zellen in Ihrer aktuell geöffneten Arbeitsmappe mit den Werten aus der Speicherdatei überschreiben möchten.

Dadurch werden Ihre Daten überschrieben

Wenn Sie diesen Vorgang bestätigen, werden die Zellwerte in Ihrer geöffneten Quellarbeitsmappe dauerhaft überschrieben.

Projektdatei erforderlich

Diese Entschlüsselungsfunktion erfordert die ursprüngliche XLS Padlock-Projektdatei (`.xplp`), die zum Erstellen der Anwendung verwendet wurde. XLS Padlock verwendet Ihre Projekteinstellungen (wie den Secret Key) zum Ver- und Entschlüsseln von Speicherdateien. Ohne die korrekte Projektdatei können Sie Speicherungen nicht entschlüsseln. **Geben Sie Ihre XLS Padlock-Projektdateien niemals weiter.**

Speicherungen an einen Rechner binden

Diese Option verhindert, dass Endbenutzer ihre Speicherdateien weitergeben. Wenn sie aktiviert ist, erstellt die Anwendung **hardwaregebundene Speicherdateien**, die nur auf dem Computer geöffnet werden können, auf dem sie erstellt wurden. Wenn ein Benutzer versucht, eine Speicherdatei auf einem anderen Computer zu öffnen, schlägt das Laden fehl.

Die Anwendung bettet die eindeutige System-ID des Computers in die Speicherdatei ein und überprüft sie beim Laden.

TIPP

Als Eigentümer der Anwendung können Sie jede hardwaregebundene Speicherdatei jederzeit selbst entschlüsseln, indem Sie die Funktion Decrypt Save File (Speicherdatei entschlüsseln) in XLS Padlock verwenden.

Warnung

Diese Funktion ist nur verfügbar, wenn Sie den Full Save mode (vollständiger Speichermodus) ausgewählt haben.

Externe Verweise und Hyperlinks

XLS Padlock schützt eine Arbeitsmappe pro EXE-Datei. Wenn Ihre Arbeitsmappe externe Referenzen verwendet oder zusätzliche Dateien benötigt, müssen Sie Ihre Pfade anpassen, damit sie nach der Kompilierung korrekt funktionieren.

XLS Padlock bietet zwei Hauptmöglichkeiten, um externe Dateien zu handhaben:

- Fügen Sie sie als Companion Files hinzu.
- Verwenden Sie dynamische Pfade in Hyperlinks oder VBA-Code.

Zur Verwaltung dynamischer Pfade stellt XLS Padlock eine Funktion namens `PLEvalVar` bereit, die direkt in Excel-Formeln verwendet oder aus VBA aufgerufen werden kann.

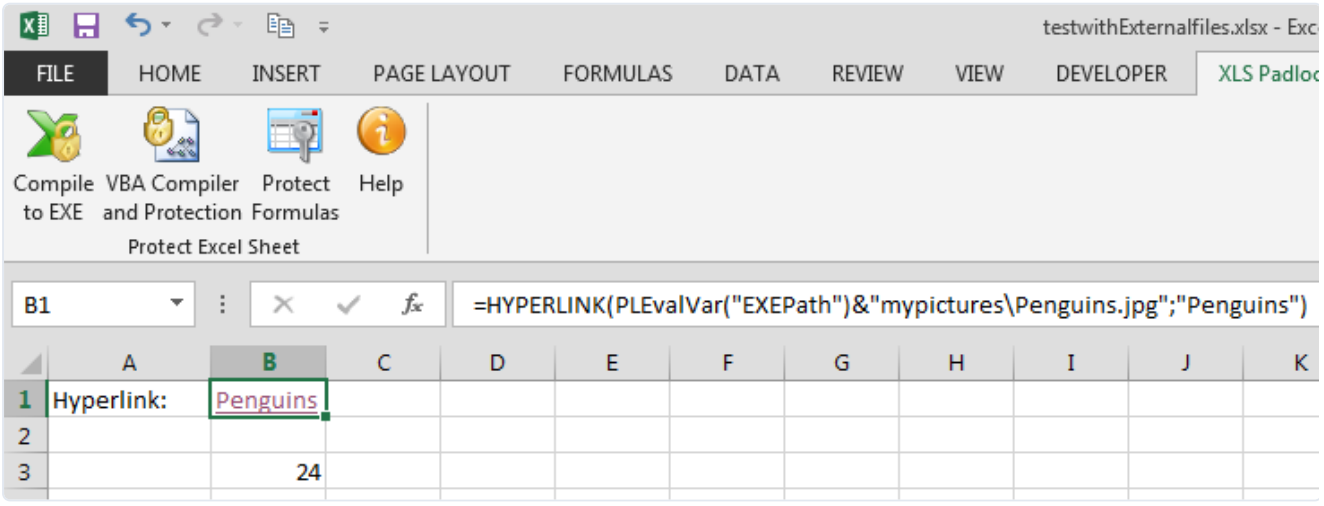
Diese Funktion nimmt ein Zeichenkettenargument entgegen:

- =PLEvalVar("EXEPath") gibt den vollständigen Pfad zu dem Ordner zurück, der die EXE-Datei der Anwendung enthält (einschließlich des abschließenden Backslashs).
- =PLEvalVar("XLSPath") gibt den vollständigen Pfad zu dem Ordner zurück, der die kompilierte Arbeitsmappe zur Laufzeit enthält (einschließlich des abschließenden Backslashs).

> **HINWEIS**
>
> Dieser Ordner ist ein virtueller Ordner, weshalb Sie keine echten Dateien darin ablegen können. Er ist nur nützlich, wenn Sie mit Companion files arbeiten (siehe Add Companion Files).

Beispiel 1

Sie haben Hyperlinks zu externen Bilddateien. Diese Bilddateien befinden sich im selben Ordner wie die XLS-Datei der Arbeitsmappe (oder in einem Unterordner).



Sie haben einen Hyperlink in einer Zelle, der wie folgt definiert ist:

```
=HYPERLINK("Penguins.jpg","Penguins")
```

Damit dies mit XLS Padlock funktioniert, müssen Sie alle externen Bilddateien in denselben Ordner wie die mit XLS Padlock erstellte EXE-Datei kopieren. Anschließend müssen Sie alle Hyperlinks ändern, um die Funktion PLEvalVar("EXEPath") einzufügen, die den Pfad zu diesem Ordner zurückgibt.

In unserem Fall wird daraus:

```
=HYPERLINK(PLEvalVar("EXEPath") & "Penguins.jpg", "Penguins")
```

> **Warnung**
>
> Externe Dateien müssen im selben Ordner wie die endgültige Anwendungs-EXE bereitgestellt werden. Es ist außerdem eine bewährte Vorgehensweise, Leerzeichen in Dateinamen zu vermeiden.

Dies funktioniert auch für Dateien in Unterordnern. Ein Link wie:

```
=HYPERLINK("My Pictures\\Penguins.jpg", "Penguins")
```

...sollte geändert werden in:

```
=HYPERLINK(PLEvalVar("EXEPath") & "mypictures\\Penguins.jpg", "Penguins")
```

Beispiel 2

Um mit VBA auf externe Dateien zuzugreifen, können Sie eine Hilfsfunktion verwenden, die den vollständigen Pfad zu einer Datei im selben Ordner wie die EXE erstellt.

👉 Eine ausführliche Erläuterung und einen wiederverwendbaren Code-Ausschnitt finden Sie in der Anleitung, wie Sie den Pfad zu einer Datei im Ordner der EXE ermitteln.

```
Public Function GetPathToFileInEXEFolder(ByVal Filename As String) As String
    Dim XLSPadlock As Object
    Dim exePath As String
    On Error GoTo Err

    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    exePath = XLSPadlock.PLEvalVar("EXEPath")
    GetPathToFileInEXEFolder = Application.BuildPath(exePath, Filename)
    Exit Function
Err:
    GetPathToFileInEXEFolder = ""
End Function
```

Pfad neben der kompilierten Arbeitsmappe ermitteln

Diese VBA-Funktion ermittelt den vollständigen Pfad zu einer Datei, die sich im selben Verzeichnis wie die EXE Ihrer Anwendung befindet. Sie ist besonders nützlich, um auf externe Ressourcen oder Begleitdateien zuzugreifen, die Sie zusammen mit Ihrer geschützten Arbeitsmappe verteilen.

👉 Fügen Sie die folgende Funktion in ein VBA-Modul ein:

```
Public Function PathToFile(ByVal Filename As String) As String
    Dim XLSPadlock As Object
    Dim exePath As String
    On Error GoTo Err

    Set XLSPadlock = Application.COMAddIns("GXLSForm.GXLSFormula").Object
    exePath = XLSPadlock.PLEvalVar("EXEPath")

    ' Use Application.BuildPath to correctly join the path and filename
    PathToFile = Application.BuildPath(exePath, Filename)

    Exit Function
Err:
    PathToFile = ""
End Function
```

Anschließend können Sie die Funktion aufrufen:

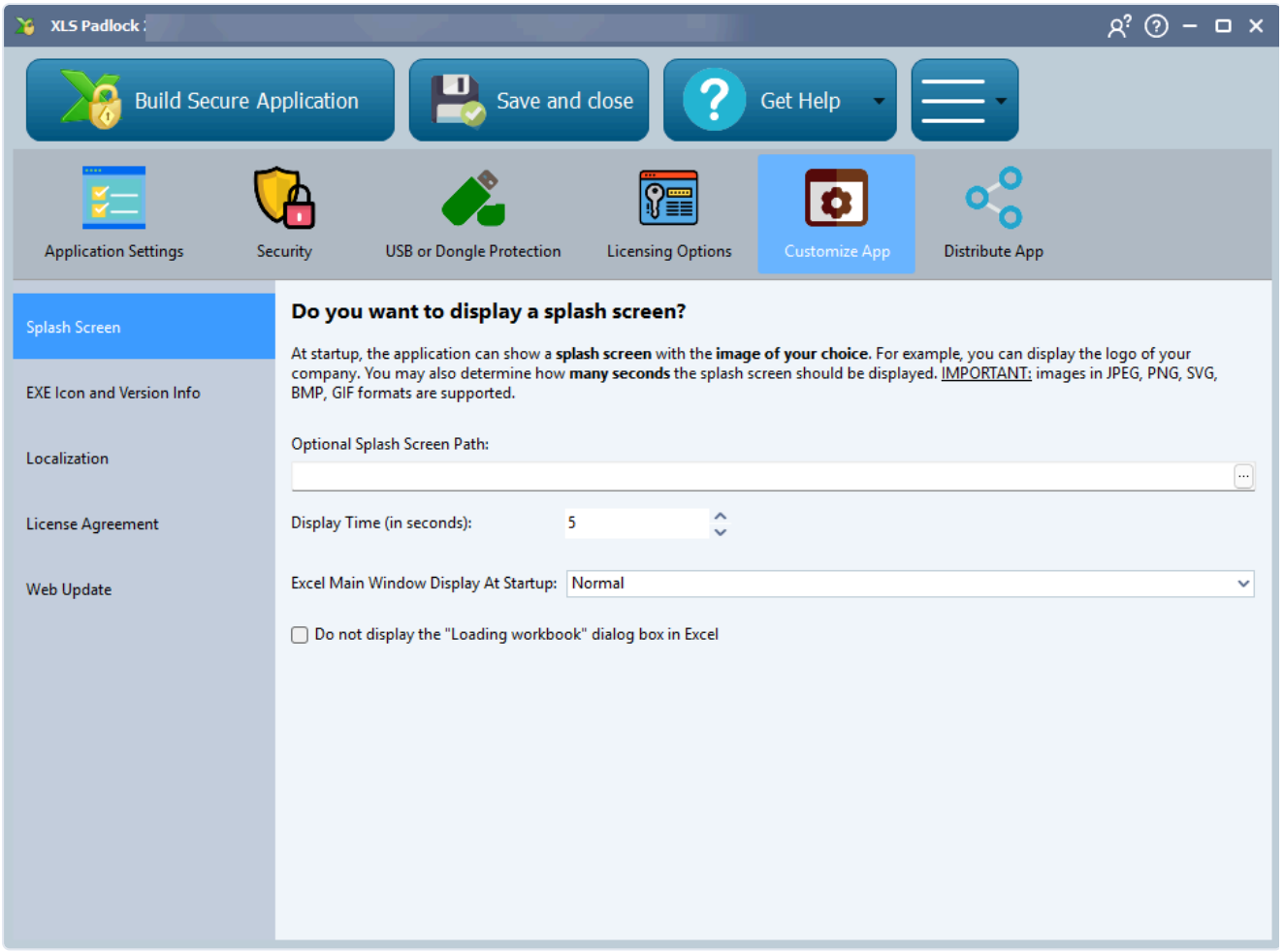
```
Sub Test_File()
 DoSomethingWith(PathToFile("data.xls"))
End Sub
```

👉 Siehe auch: VBA API Cookbook & Recipes

Anwendung anpassen

XLS Padlock bietet Ihnen mehrere Möglichkeiten, **Ihre kompilierte Excel-Anwendung anzupassen**, sodass Sie ihr Erscheinungsbild, ihre Meldungen und ihr Verhalten für die Endbenutzer personalisieren können.

Dieses Thema bietet einen Überblick über die Optionen, die unter "**Customize App**" in XLS Padlock verfügbar sind:



Begrüßungsbildschirm

Zeigen Sie beim Start Ihrer Anwendung einen benutzerdefinierten **Begrüßungsbildschirm** an, zum Beispiel das Logo Ihres Unternehmens oder ein Willkommensbild mit Ihrer Marke.
Sie können das Bildformat wählen (JPEG, PNG, BMP, SVG oder GIF), festlegen, wie viele Sekunden er sichtbar bleibt, und steuern, wie sich das Excel-Fenster während des Starts verhält.

➡️ Mehr über die Einstellungen des Begrüßungsbildschirms erfahren

EXE-Symbol und Versionsinformationen

Ersetzen Sie das Standardsymbol von XLS Padlock durch **Ihr eigenes Anwendungssymbol**.

Sie können auch Versionsinformationen angeben, etwa den **Produktnamen, den Firmennamen, das Copyright** und die **Dateibeschreibung**, die alle im Windows-Dialogfeld der Dateieigenschaften sichtbar sind.
Dies hilft Ihnen, Ihre EXE als professionelle eigenständige Anwendung mit Ihrer Marke zu versehen.

➡️ Mehr über das EXE-Symbol erfahren und Versionsinformationen

Lokalisierung

Passen Sie alle **integrierten Dialogfelder, Meldungen und Eingabeaufforderungen** an, die von Ihrer geschützten Arbeitsmappe angezeigt werden, oder übersetzen Sie sie.
XLS Padlock stellt ein flexibles Lokalisierungssystem bereit, sodass Sie Ihre Anwendung an verschiedene Sprachen anpassen oder vollständig personalisierte Benutzermeldungen erstellen können.

➡️ Mehr über die Lokalisierung erfahren

Lizenzvereinbarung

Fügen Sie eine **Lizenzvereinbarung** hinzu, die Benutzer akzeptieren müssen, bevor sie Ihre kompilierte Excel-Anwendung zum ersten Mal ausführen.
Sie können reinen Text einfügen oder eine externe RTF-/HTML-Datei laden.
Wenn Benutzer die Vereinbarung ablehnen, wird die Anwendung nicht weiter geöffnet.

➡️ Mehr über die Lizenzvereinbarung erfahren

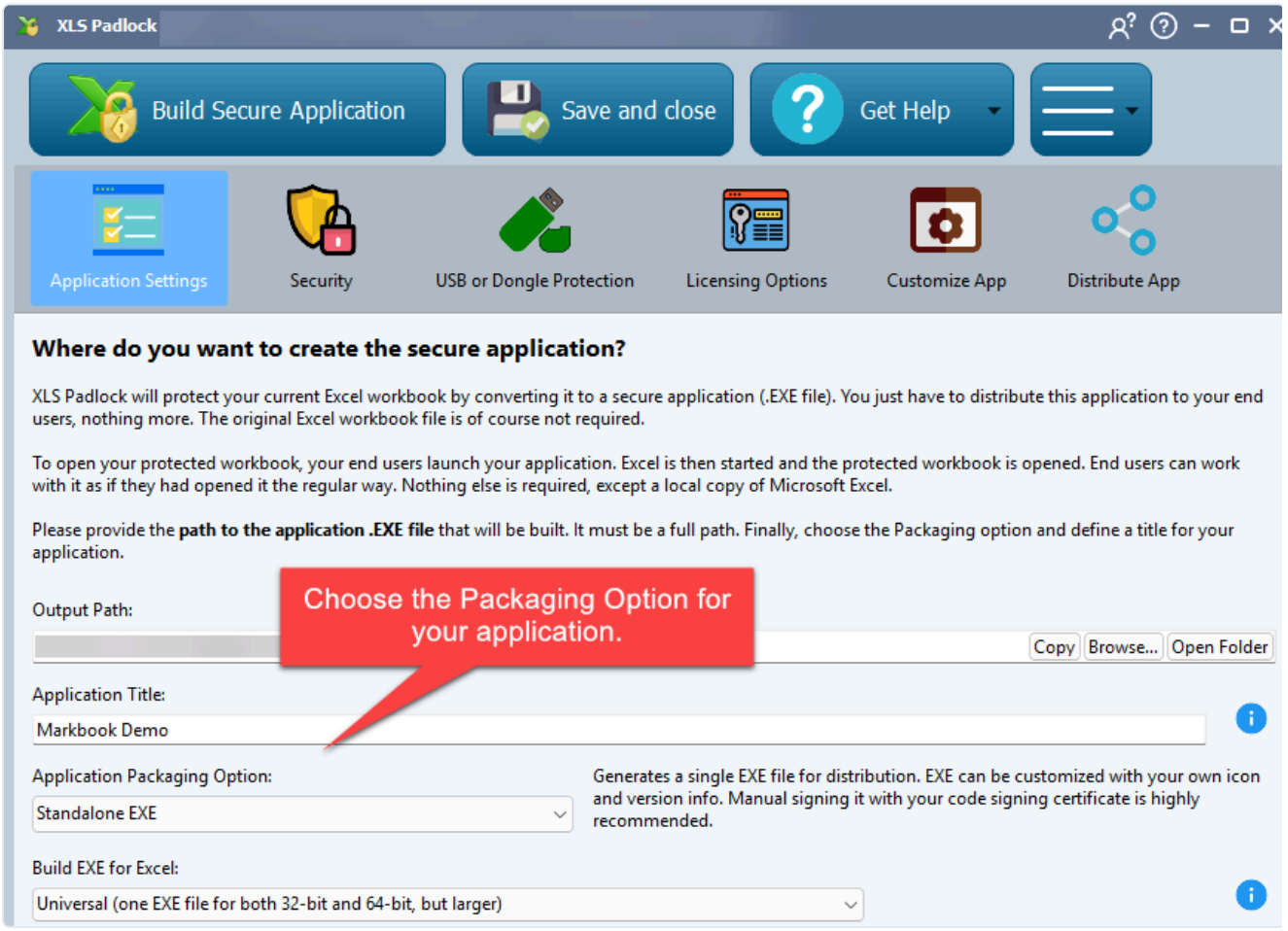
Web-Update

Aktivieren Sie **webbasierte Updates** für Ihre Excel-Anwendung.
Mit dieser Funktion können Sie Benutzer benachrichtigen, wenn eine neue Version Ihrer EXE verfügbar ist, und sie automatisch von Ihrer Website oder Ihrem Server herunterladen.
Das ist eine effiziente Möglichkeit, Ihre Benutzer mit minimalem Aufwand auf dem neuesten Stand zu halten.

➡️ Mehr über Web-Update erfahren

Anwendungspaketierung

Die Option **Application Packaging** bestimmt, wie Ihre geschützte Arbeitsmappe kompiliert wird, und bietet zwei unterschiedliche Formate.



Empfehlung

Wir empfehlen die Option **Standalone EXE**, wenn Sie über ein Zertifikat zur Codesignierung verfügen. Auch wenn dies nicht zwingend erforderlich ist, ist die Signierung Ihrer EXE die beste Methode, um "Unknown App"-Warnungen von Windows SmartScreen zu vermeiden und Fehlalarme von Antivirensoftware zu reduzieren.

Standalone EXE

Diese Option kompiliert Ihre Excel-Arbeitsmappe in eine einzige, eigenständige ausführbare Datei (`.exe`). Sie können diese einzelne Datei an Ihre Benutzer verteilen, und diese benötigen die ursprüngliche Excel-Arbeitsmappe nicht.

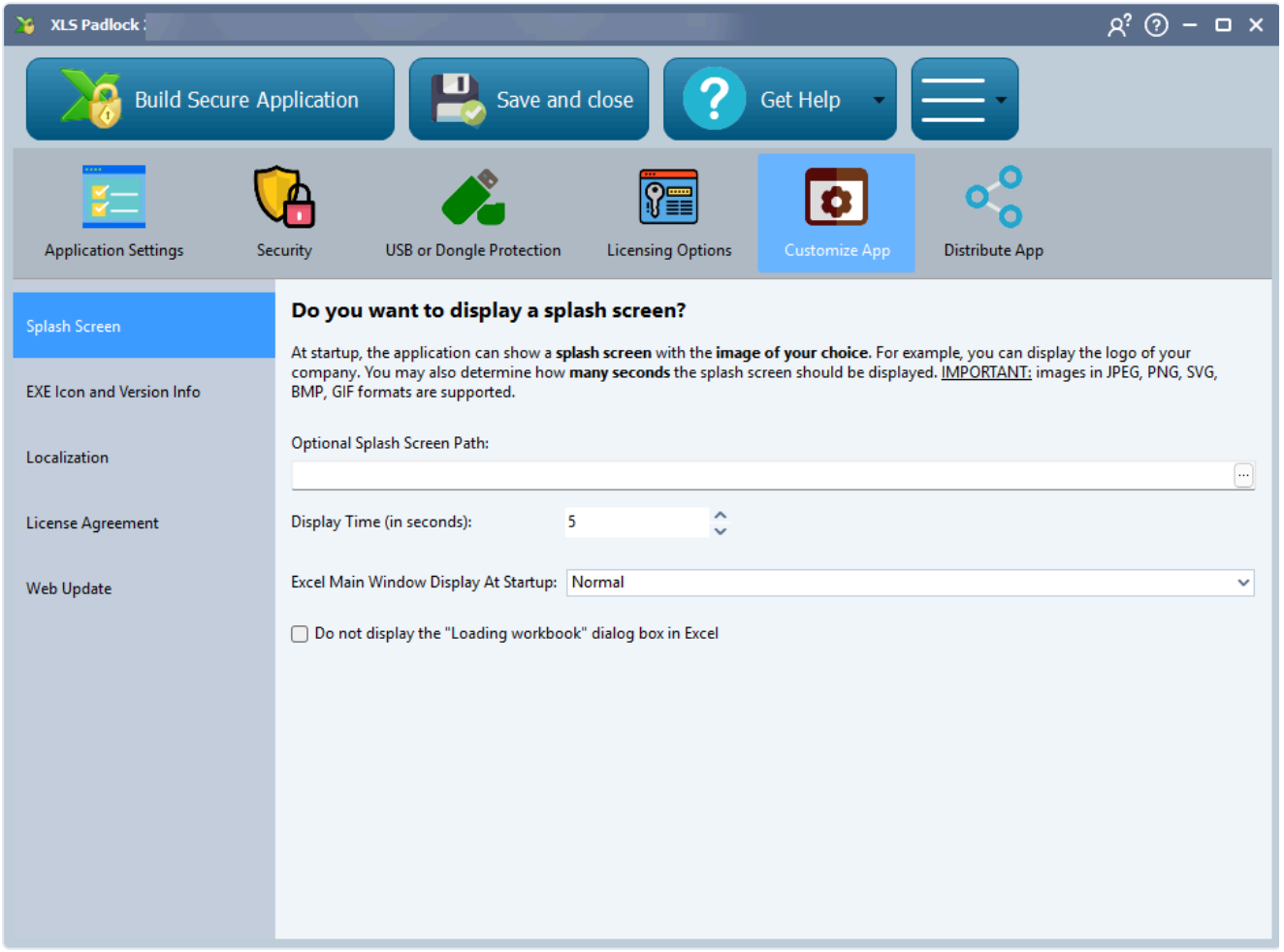
EXE + XPLAPP Application Bundle

Diese Option erstellt eine EXE-Datei zusammen mit einer Begleitdatei `.bin64` und einer Datendatei `.xplapp`. Die Haupt-EXE-Datei ist von unserem Unternehmen vorsigniert, was dazu beiträgt, dass sie von Windows SmartScreen und Antivirenlösungen stärker als vertrauenswürdig eingestuft wird.

- Diese Verteilungsmethode setzt voraus, dass die Dateien EXE, `.bin64` und `.xplapp` alle im selben Ordner aufbewahrt werden.
- Zur besseren Handhabung kann XLS Padlock diese Dateien in einem einzigen Zip-Archiv oder einem Installationsprogramm zusammenfassen.
- Die Verwendung dieser Methode verringert die Wahrscheinlichkeit erheblich, auf "Unknown Application"-Warnungen und Antiviren-Fehlalarme zu stoßen.
- Die Begleitdatei `.bin64` gewährleistet die Kompatibilität mit 64-Bit-Versionen von Excel.

Startbildschirm

Sie können einen **Begrüßungsbildschirm** anzeigen, ein Bild, das beim Start kurz eingeblendet wird, während Ihre Anwendung initialisiert wird. Dies ist eine hervorragende Möglichkeit, Ihr Firmenlogo anzuzeigen oder Ihre Anwendung mit Ihrer Marke zu versehen.



Einstellungen für den Begrüßungsbildschirm

Sie können ein beliebiges Bild im Format JPEG, PNG, BMP, SVG oder GIF verwenden.

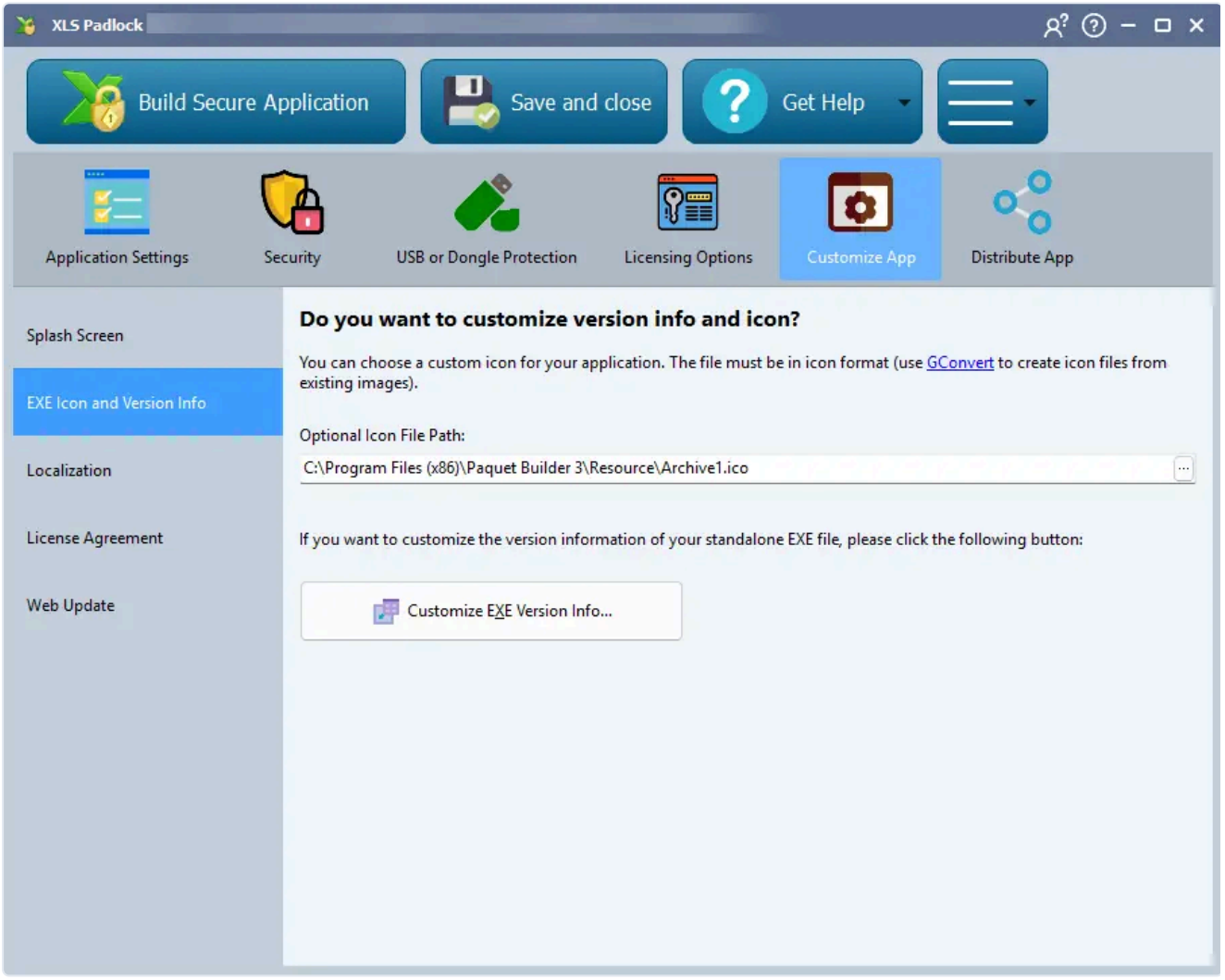
Sie können festlegen, wie lange der Begrüßungsbildschirm angezeigt wird. Standardmäßig können Benutzer ihn sofort schließen, indem sie darauf klicken. Dieses Verhalten lässt sich in den Erweiterten Optionen deaktivieren.

> **Verwenden Sie transparente PNGs für ein einzigartiges Aussehen**
>
> XLS Padlock unterstützt nicht rechteckige, halbtransparente Begrüßungsbildschirme. Verwenden Sie eine 32-Bit-PNG-Datei mit einem Alphakanal, um Ihrer Anwendung ein modernes, individuelles Erscheinungsbild zu verleihen.

👉 Siehe auch: [Das Dialogfeld "Loading workbook" in Excel nicht anzeigen](#chapter-do-not-display-the-loading-workbook-dialog-box).

EXE-Symbol ändern

Mit XLS Padlock können Sie das **standardmäßige Anwendungssymbol** durch ein eigenes ersetzen. Um das Erscheinungsbild Ihrer Anwendung anzupassen, geben Sie den **Pfad zu einer benutzerdefinierten Symboldatei** an, die eine `.ico`-Erweiterung haben muss:



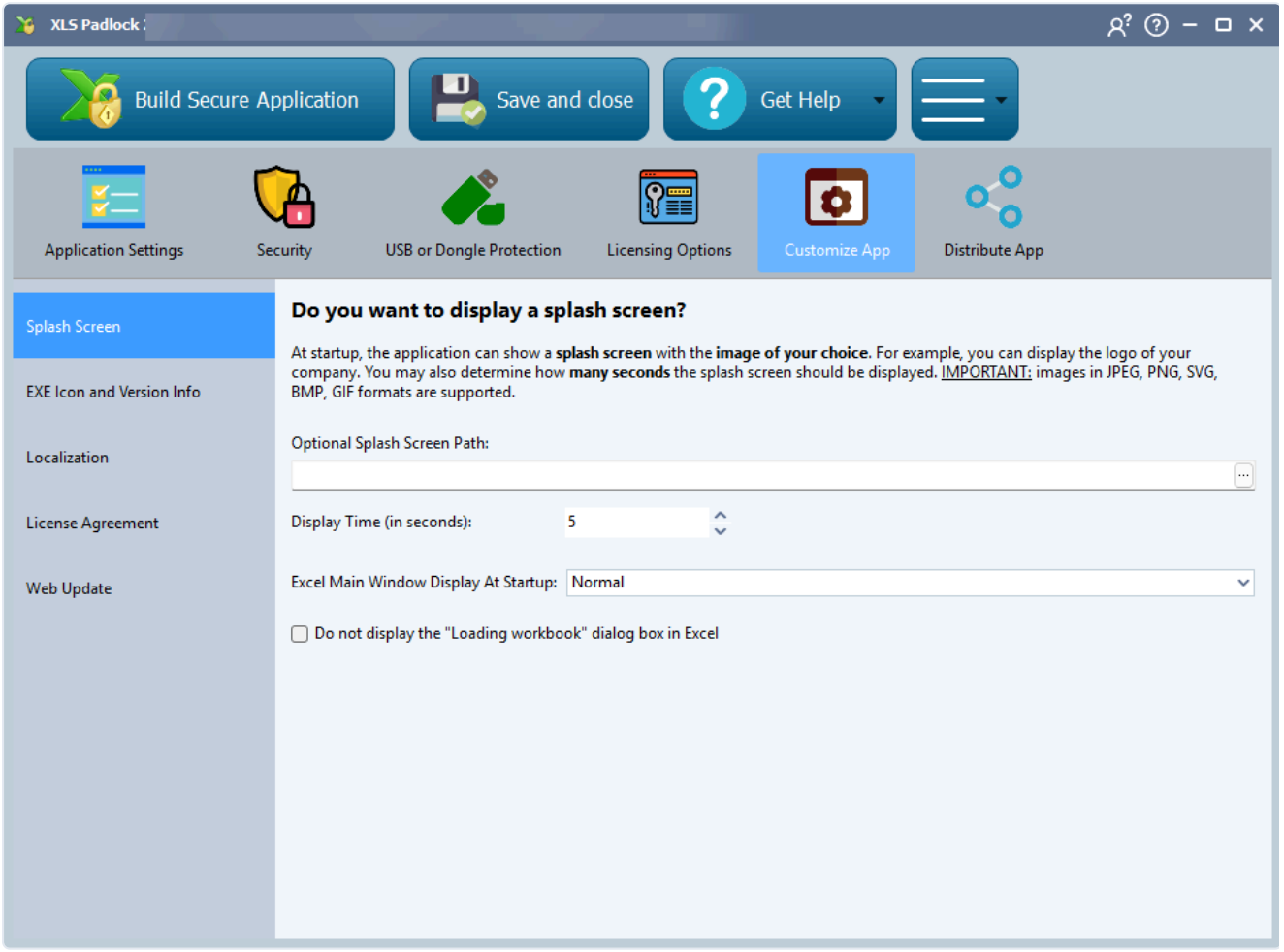
XLS Padlock unterstützt standardmäßige Symbolformate, einschließlich verschiedener Größen (z. B. 32x32, 48x48) und Farbtiefen. Wenn Sie ein Werkzeug zum Erstellen oder Extrahieren von Symbolen benötigen, empfehlen wir unser begleitendes Hilfsprogramm GConvert, das Bilder problemlos in Symbole umwandeln kann.

> **Warnung**
>
> Die Symboldatei muss während des Kompilierungsvorgangs als externe Datei auf Ihrem Computer verfügbar sein.

> **Info**
>
> XLS Padlock wendet dieses Symbol auch auf die EXE-Datei der eigenständigen Anwendung an und sorgt so für eine einheitliche Markenidentität.

Excel-Fenster beim Start

Mit der Option **Excel Main Window Display at Startup** (Anzeige des Excel-Hauptfensters beim Start) legen Sie fest, wie das Excel-Fenster beim Start Ihrer Anwendung dimensioniert wird. Diese Einstellung finden Sie auf der Seite Splash Screen.



Sie können zwischen drei Optionen wählen:

- **Normal**: Das Excel-Fenster wird mit seiner Standardgröße und -position geöffnet. Dies ist das Standardverhalten.
- **Minimized** (Minimiert): Die Anwendung startet minimiert in der Windows-Taskleiste. Dies ist nützlich, wenn Ihre Anwendung im Hintergrund läuft oder wenn Sie ein benutzerdefiniertes Formular anzeigen möchten, bevor das Excel-Hauptfenster eingeblendet wird.
- **Maximized** (Maximiert): Das Excel-Fenster wird maximiert geöffnet und füllt den gesamten Bildschirm aus.

Soll das Excel-Fenster vollständig ausgeblendet werden?

Wenn Ihre Anwendung ausschließlich durch VBA-Code und UserForms gesteuert wird und Sie möchten, dass das Excel-Fenster sowie sein Taskleisten-Symbol überhaupt nie erscheinen, lesen Sie Als reine VBA-Anwendung ausführen (Excel-Hauptfenster vollständig ausgeblendet). Wenn diese Option aktiviert ist, wird die obige Auswahl außer Kraft gesetzt und Excel startet immer ausgeblendet.

Siehe auch

- So konfigurieren Sie den Splash Screen
- So blenden Sie das Dialogfeld "Loading workbook" aus
- Als reine VBA-Anwendung ausführen (Excel-Hauptfenster vollständig ausgeblendet)

Als reine VBA-Anwendung ausführen

Wenn Ihre geschützte Arbeitsmappe vollständig durch **VBA-Code** und **UserForms** gesteuert wird und die Tabellenkalkulation selbst niemals für Ihre Endbenutzer sichtbar sein soll, können Sie sie als *reine VBA-App* ausliefern. Das Excel-Hauptfenster bleibt von Anfang bis Ende ausgeblendet, einschließlich seines Symbols in der Taskleiste, und nur Ihr UserForm erscheint auf dem Bildschirm. Ihre Anwendung sieht aus und verhält sich wie ein eigenständiges Windows-Programm, ohne sichtbare Spur von Excel.

Diese Option ist seit **XLS Padlock 2026.0** verfügbar.

Wann sollte sie verwendet werden

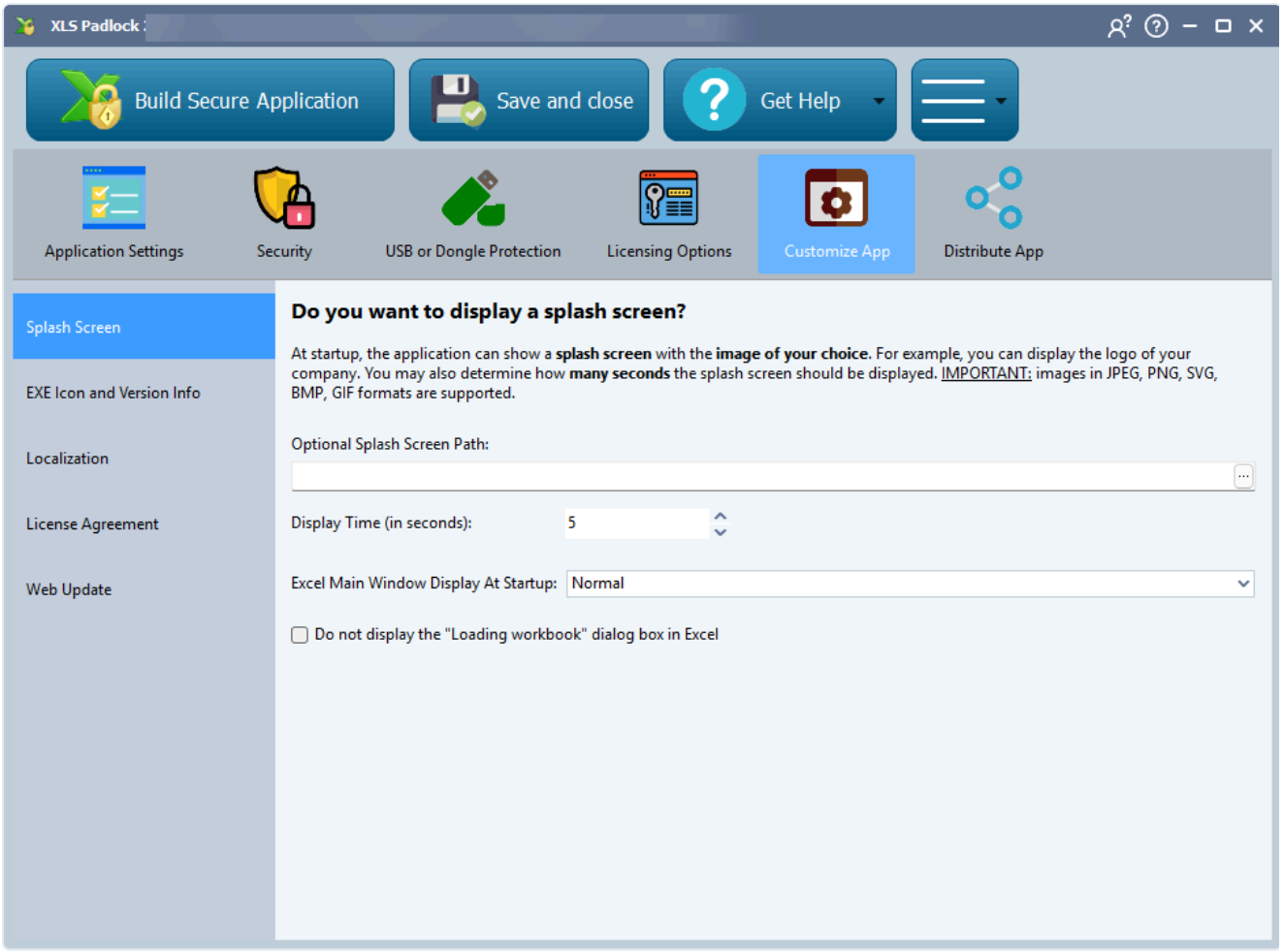
Verwenden Sie den reinen VBA-Modus, wenn **alle** folgenden Bedingungen zutreffen:

- Die Benutzeroberfläche Ihrer Anwendung ist mit **UserForms** aufgebaut, nicht mit Tabellenblättern.
- Sie möchten nicht, dass Endbenutzer die zugrunde liegende Tabellenkalkulation sehen, bearbeiten oder mit ihr interagieren.
- Sie wünschen sich beim Start ein sauberes, markenkonformes Erscheinungsbild, ohne Excel-Startbildschirm, ohne Excel-Eintrag in der Taskleiste.

Wenn Ihre Anwendung auf sichtbaren Tabellenblättern beruht (Dateneingaberaster, als Zellen dargestellte Dashboards usw.), aktivieren Sie diese Option **nicht**, denn die Tabellenblätter wären für Ihre Benutzer nicht sichtbar.

So aktivieren Sie sie

1. Öffnen Sie Ihr Projekt in XLS Padlock.
2. Wechseln Sie zur Seite **Splash Screen** (unter Application Customization).
3. Aktivieren Sie die Option **"Run as a VBA-only app (Excel main window fully hidden)"**.



Das ist alles auf der Seite von XLS Padlock. Nun müssen Sie Ihrer Arbeitsmappe ein kleines Stück VBA-Code hinzufügen, damit Ihr UserForm beim Start der Anwendung angezeigt wird.

Den VBA-Startcode hinzufügen

Öffnen Sie im VBA-Editor von Excel das Modul `ThisWorkbook` und fügen Sie ein:

```
Private Sub Workbook_Open()
    Application.Visible = False
    UserForm1.Show
End Sub
```

Ersetzen Sie `UserForm1` durch den Namen des Formulars, das Sie zuerst anzeigen möchten.

> **Warum beides, das Kontrollkästchen *und* der VBA-Code?**
>
> Das Kontrollkästchen von XLS Padlock garantiert, dass das Excel-Hauptfenster niemals sichtbar wird, **bevor** Ihr VBA-Code ausgeführt wird (kein Aufblitzen, kein Symbol in der Taskleiste). Die Zeile `Application.Visible = False` in `Workbook_Open` hält Excel **nach** Abschluss der Excel-Initialisierung ausgeblendet, denn Excel stellt seine Sichtbarkeit während des Starts intern wieder her, sodass VBA die Regel durchsetzen muss, sobald die Arbeitsmappe vollständig geladen ist. Die Verwendung beider Ebenen bietet Ihnen ein sauberes, flackerfreies Benutzererlebnis.

Was der Endbenutzer sieht

Mit aktiviertem Kontrollkästchen und dem obigen VBA-Ausschnitt:

- **Kein Excel-Startbildschirm** beim Start.
- **Kein Excel-Hauptfenster** zu irgendeinem Zeitpunkt während der Laufzeit der Anwendung.
- **Kein "Excel"-Eintrag in der Windows-Taskleiste**, nur Ihr UserForm.
- Ihre Anwendung verhält sich wie ein eigenständiges Windows-Programm, das um Ihr Formular herum aufgebaut ist.

Wenn Sie außerdem einen Splash Screen konfiguriert und das Dialogfeld "Loading workbook" ausgeblendet haben, bleibt die gesamte Startsequenz ab dem allerersten Frame innerhalb Ihrer markenkonformen Benutzeroberfläche.

Die Anwendung schließen

Da Excel ausgeblendet ist, steht Ihren Benutzern die standardmäßige Excel-Schaltfläche zum Schließen nicht zur Verfügung. Normalerweise schließen Sie die Anwendung von Ihrem UserForm aus mit:

```
Private Sub btnClose_Click()
    Application.Quit
End Sub
```

Oder, wenn Sie Excel weiterlaufen lassen, aber Ihre Benutzeroberfläche entladen möchten:

```
Unload Me
```

Wechselwirkung mit anderen Einstellungen

Einstellung	Verhalten, wenn "VBA-only app" aktiviert ist
Excel Main Window Display at Startup (Normal / Maximized / Minimized)	Außer Kraft gesetzt, Excel startet immer ausgeblendet.

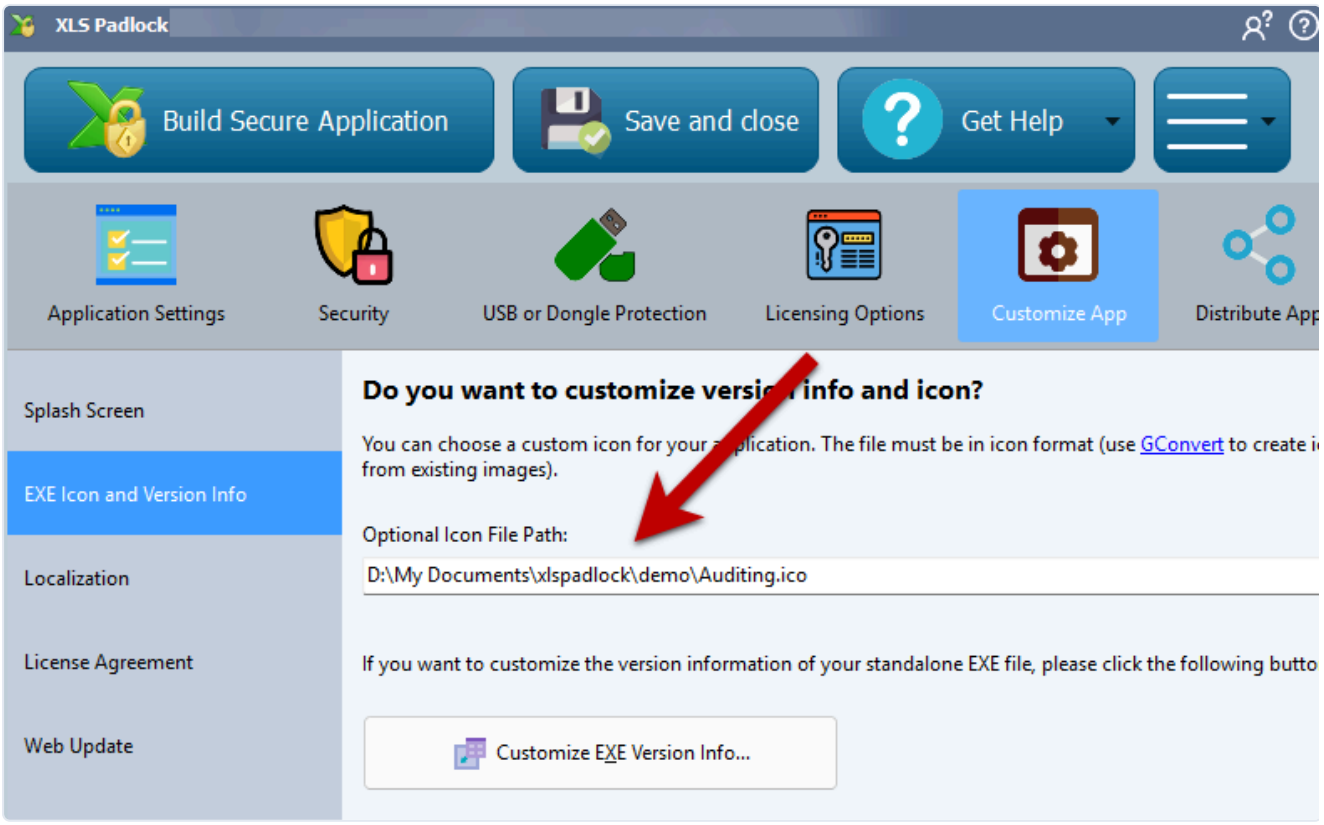
Einstellung	Verhalten, wenn "VBA-only app" aktiviert ist
Do not display the "Loading workbook" dialog box	Wird ebenfalls zur Aktivierung empfohlen, für einen vollständig stillen Start.
Splash Screen	Funktioniert wie konfiguriert. Wird angezeigt, bevor Excel überhaupt geladen wird.

Siehe auch

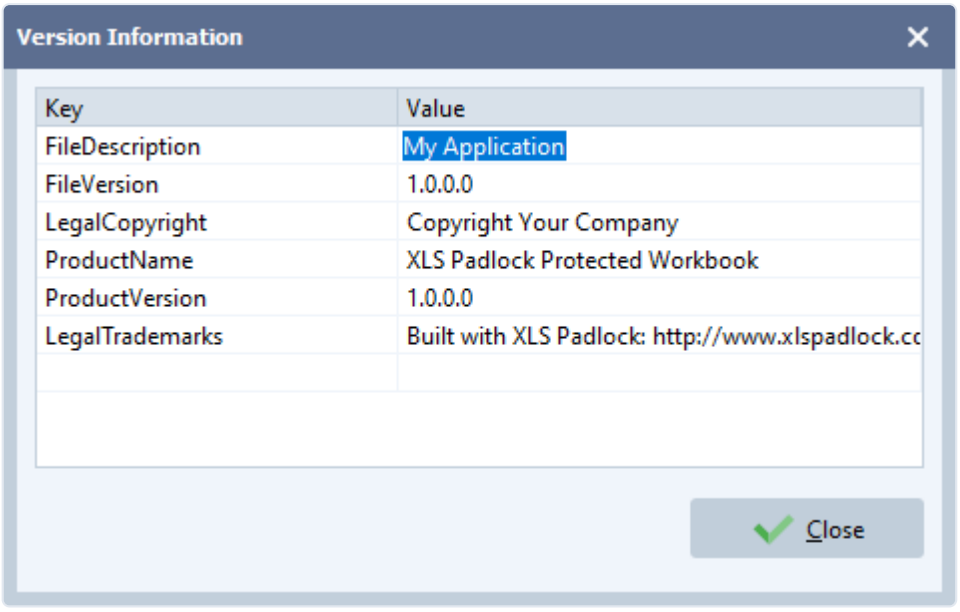
- So konfigurieren Sie den Splash Screen
- So legen Sie die Anzeige des Excel-Hauptfensters beim Start fest
- So blenden Sie das Dialogfeld "Loading workbook" aus
- VBA API Cookbook

EXE-Versionsinformationen

Die **Versionsinformationen** einer ausführbaren Datei bilden einen speziellen Ressourcenbereich mit Details wie der Versionsnummer der Datei, dem vorgesehenen Betriebssystem, dem ursprünglichen Dateinamen und den Copyright-Informationen. Diese Daten werden in die kompilierte EXE eingebettet. Wenn sie enthalten sind, können die Benutzer diese Informationen einsehen, indem sie mit der rechten Maustaste auf das Programmsymbol klicken, "Eigenschaften" auswählen und zur Registerkarte "Details" navigieren (oder indem sie im Explorer `Alt+Enter` auf der Datei drücken).



Mit XLS Padlock können Sie Ihre eigenen Versionsinformationen in die eigenständige EXE-Datei einbetten. Klicken Sie auf "**Customize EXE Version Info**" (EXE-Versionsinformationen anpassen), um ein Fenster mit den folgenden Feldern zu öffnen:



- **File Description**: Eine kurze Beschreibung des Inhalts Ihrer Anwendung.
- **Company Name**: Der Name Ihres Unternehmens.
- **File Version**: Die Release-Nummer Ihrer `.exe`-Datei im Format `X.X.X.X` (z. B. `1.20.34.45`). Dieser Wert wird auch von der Web-Update-Funktion verwendet, um nach neuen Anwendungsversionen zu suchen.
- **Legal Copyright**: Ihr Copyright-Hinweis, etwa "Copyright © [annee] Your Company. All rights reserved."
- **Product Name**: Der Name Ihres Produkts oder Ihrer Anwendung, der in der Regel mit dem Titel Ihrer Anwendung übereinstimmt.
- **Product Version**: Die Release-Nummer Ihres Produkts. Sie stimmt häufig mit der File Version überein.
- **Legal Trademarks**: Alle eingetragenen Marken, die Sie aufnehmen möchten.

> **Info**
>
> Die File Version wird von der Web-Update-Funktion verwendet.

Befehlszeilenschalter

Die EXE-Datei Ihrer kompilierten Anwendung unterstützt mehrere **Befehlszeilenschalter** (auch als Argumente bezeichnet), die beim Start bestimmte Aktionen automatisieren. Sie können diese in Verknüpfungen oder Skripten verwenden, um das Verhalten der Anwendung beim Start zu steuern.

> **TIPP**
>
> 👉 Diese Befehlszeilenargumente können zur Laufzeit auch über die VBA-API zum Lesen von Befehlszeilenparametern abgerufen werden.

Verfügbare Schalter

`-deact`

Startet den Deaktivierungsvorgang und ermöglicht es einem Benutzer, seine Lizenz auf einen anderen Computer zu übertragen.

```
MYAPP.EXE -deact
```

`-del`

Löscht alle sicheren Speicherdateien der Anwendung und lädt die ursprüngliche Arbeitsmappe. **Mit Vorsicht verwenden, da diese Aktion nicht rückgängig gemacht werden kann.**

```
MYAPP.EXE -del
```

`-enterkey`

Öffnet das Aktivierungsdialogfeld und ermöglicht es dem Benutzer, einen neuen Aktivierungsschlüssel einzugeben. Dies ist nützlich, um einen alten oder abgelaufenen Schlüssel zu ersetzen.

```
MYAPP.EXE -enterkey
```

`-load`

Öffnet ein Dateidialogfeld, das den Benutzer auffordert, eine zu ladende Speicherdatei auszuwählen.

```
MYAPP.EXE -load
```

`-reset`

Verwirft alle nicht gespeicherten Änderungen der letzten Sitzung und lädt die ursprüngliche, unveränderte Arbeitsmappe. Dabei werden keine `.xlsc`-Speicherdateien gelöscht.

```
MYAPP.EXE -reset
```

`-webupdate`

Zwingt die Anwendung, basierend auf Ihren "Web Update"-Einstellungen sofort online nach einer neuen Version zu suchen.

```
MYAPP.EXE -webupdate
```

Laden einer bestimmten Speicherdatei

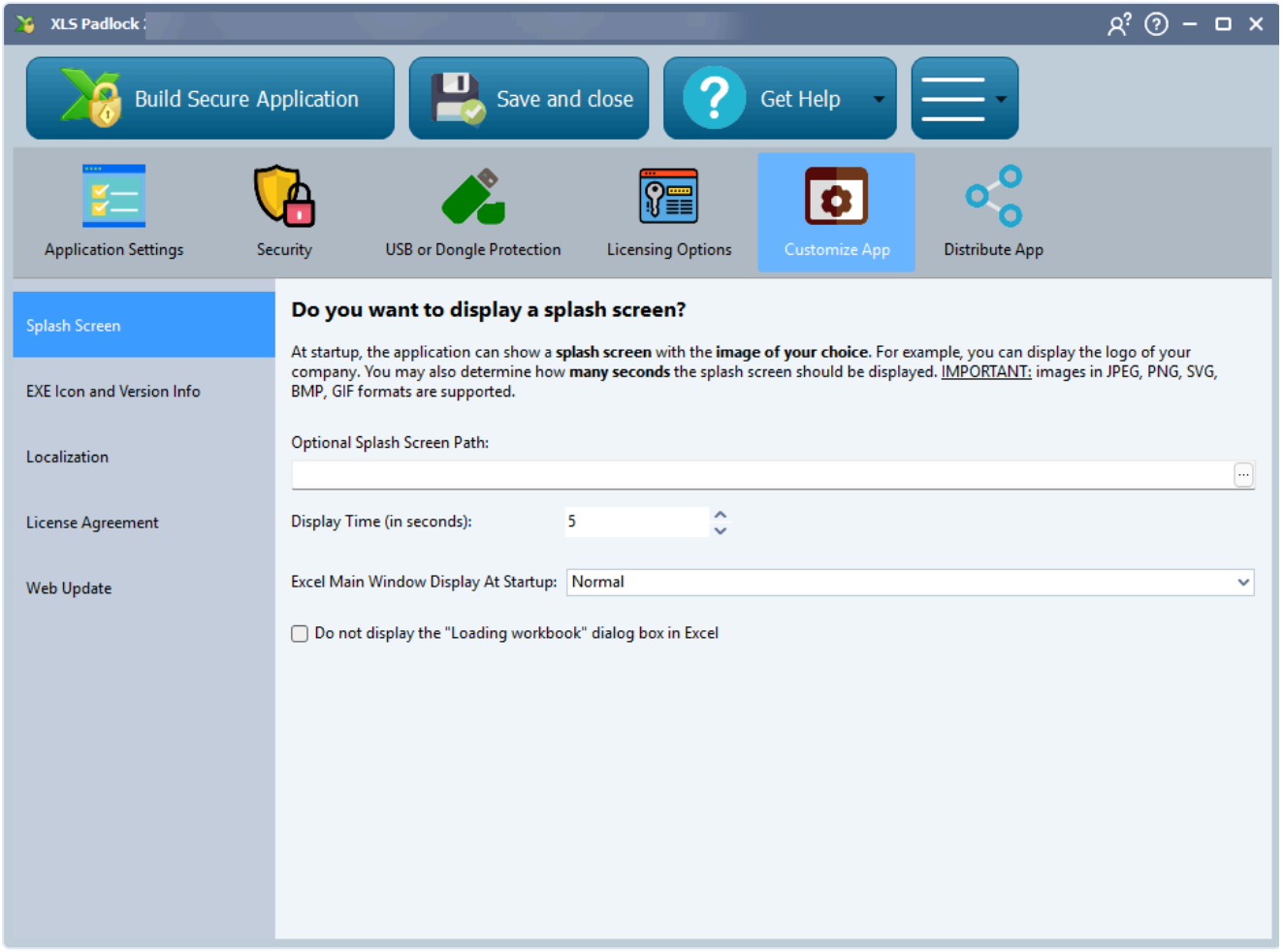
Sie können die Anwendung auch mit dem vollständigen Pfad zu einer sicheren Speicherdatei (`.xlsc` oder `.xlsce`) als Argument starten. Dadurch wird die angegebene Datei beim Start automatisch ohne weitere Eingabeaufforderungen geladen.

`MyApp.exe "D:\My Documents\123.xlsc"`

Ladedialog ausblenden

Standardmäßig zeigt Excel beim Start Ihrer geschützten Anwendung ein kleines Dialogfeld mit der Meldung „**Loading workbook, please wait…**“ an.

Sie können dieses Dialogfeld ausblenden, indem Sie die Option "**Do not display the "Loading workbook" dialog box in Excel**" (Das Dialogfeld "Loading workbook" in Excel nicht anzeigen) aktivieren, die sich auf der Seite Splash Screen befindet. Dies ist nützlich, um einen aufgeräumteren Startvorgang zu schaffen, insbesondere wenn Sie bereits einen benutzerdefinierten Begrüßungsbildschirm verwenden.



Es ist außerdem möglich, diese Meldung frühzeitig mithilfe von VBA-Code auszublenden, wodurch Sie mehr Kontrolle darüber haben, wann das Dialogfeld verschwindet.

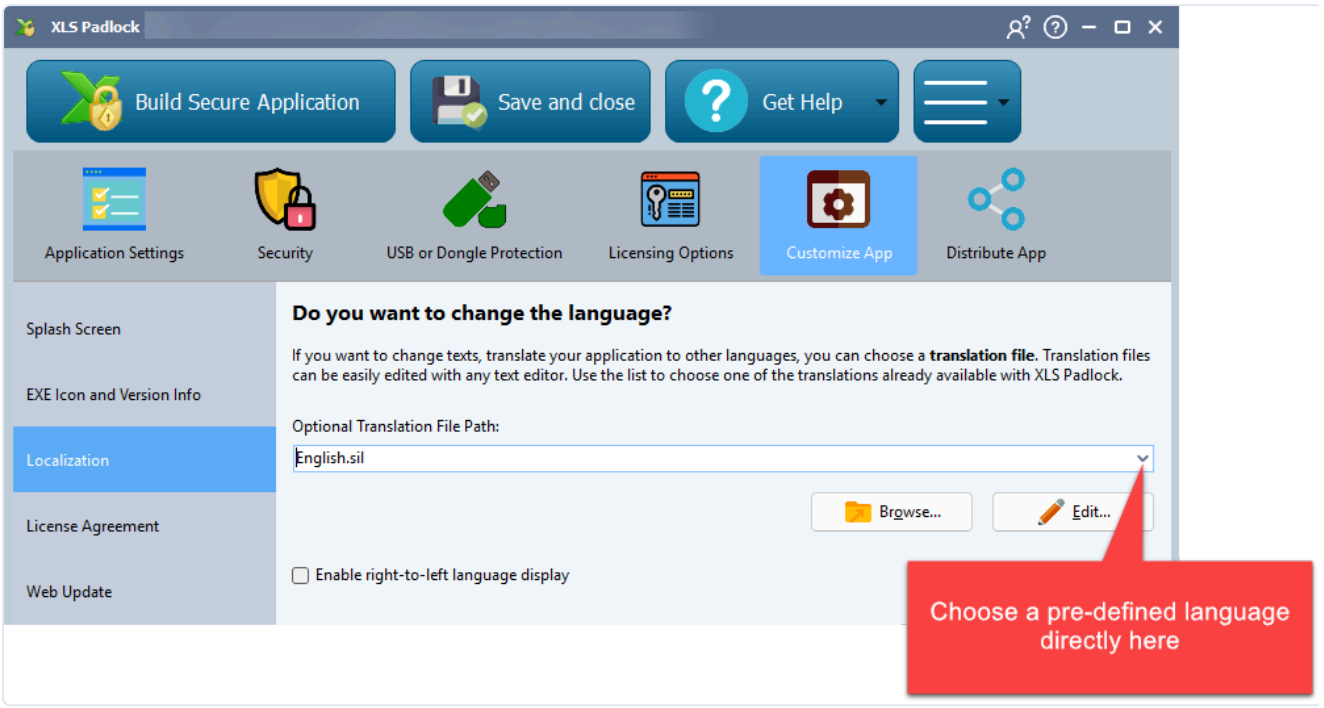
👉 Siehe auch

- Wie Sie den Splash Screen konfigurieren
- Wie Sie die Anzeige des Excel-Hauptfensters beim Start festlegen

Lokalisierung und Übersetzung

XLS Padlock bietet **Lokalisierungsunterstützung** für Ihre geschützten Arbeitsmappen, sodass Sie den gesamten Oberflächentext (Dialogtitel, Meldungen, Eingabeaufforderungen und Laufzeitwarnungen) in der Sprache Ihrer Wahl anzeigen können.

Sie können die Lokalisierung für Ihre geschützte Arbeitsmappen-App hier konfigurieren:



Format der Übersetzungsdatei

Seit XLS Padlock 2026 verwenden Übersetzungsdateien das **standardmäßige GNU-gettext**-Format mit der Erweiterung `.po` (und ihrem kompilierten binären Gegenstück `.mo`). Dies ist dasselbe Format, das von Linux-Distributionen, Python, PHP, WordPress und unzähligen anderen Anwendungen verwendet wird, was bedeutet, dass Sie Übersetzungen mit einem der zahlreichen ausgereiften gettext-Werkzeuge bearbeiten können, darunter der kostenlose plattformübergreifende Editor Poedit.

Die kompilierte Anwendung bettet zur Build-Zeit eine einzige `.mo`-Ressource ein, die zur Laufzeit vollständig aus dem Speicher geladen wird. Es werden keine temporären Dateien auf die Festplatte extrahiert.

XLS Padlock wird mit vorübersetzten `.po`-Dateien für **10 Sprachen** ausgeliefert, die im Unterordner `locale\` Ihres XLS Padlock-Installationsverzeichnisses verfügbar sind:

Datei	Sprache
`en.po`	Englisch
`fr.po`	Französisch
`es.po`	Spanisch

Datei	Sprache
`pt.po`	Portugiesisch (Brasilien)
`nl.po`	Niederländisch
`de.po`	Deutsch
`ar.po`	Arabisch (Layout von rechts nach links unterstützt)
`it.po`	Italienisch
`zh_CN.po`	Vereinfachtes Chinesisch
`zh_TW.po`	Traditionelles Chinesisch

Eine neue Sprache anfordern

Wenn Sie eine Sprache benötigen, die nicht in der Liste enthalten ist, kontaktieren Sie uns bitte. Sie können auch Ihre eigene `.po`-Datei ausgehend von einer der oben genannten als Ausgangspunkt erstellen. Siehe *Übersetzungsdateien bearbeiten* weiter unten.

So legen Sie eine Sprache für Ihre geschützte Arbeitsmappe fest

- Klicken Sie auf der Seite **Localization** auf **Browse** neben dem Feld *Translation file*. Der Dialog öffnet sich standardmäßig im Ordner `locale\` Ihrer XLS Padlock-Installation, in dem sich die 10 vorgefertigten `.po`-Dateien befinden. Wählen Sie die gewünschte Sprache aus oder navigieren Sie zu einer eigenen benutzerdefinierten `.po`-Datei.
- Sie können auch den vollständigen Pfad manuell in das Feld *Translation file* einfügen oder es leer lassen, um die in XLS Padlock eingebettete englische Standardübersetzung zu verwenden.

> **WICHTIG**
>
> Die ausgewählte Übersetzungsdatei muss unter ihrem angegebenen Pfad zugänglich sein, **wenn Sie Ihre Anwendung kompilieren**. XLS Padlock liest und kompiliert diese Datei während des Build-Vorgangs und bettet die resultierende `.mo`-Ressource direkt in die geschützte EXE ein. Zur Laufzeit besteht keine Abhängigkeit von der ursprünglichen `.po`-Datei.

Übersetzungsdateien bearbeiten

`.po`-Dateien sind einfache UTF-8-Textdateien. Sie können sie auf zwei Arten bearbeiten:

- **Mit Poedit** (poedit.net, kostenlos, plattformübergreifend), die empfohlene Wahl. Es zeigt Quell- und übersetzte Zeichenfolgen nebeneinander an, markiert unscharfe (fuzzy) und nicht übersetzte Einträge, validiert die Syntax beim Speichern und erzeugt eine saubere UTF-8-Ausgabe.
- **Mit einem beliebigen Texteditor** (Notepad++, VS Code, …). Speichern Sie als **UTF-8 ohne BOM** und behalten Sie ein `msgid` / `msgstr`-Paar pro Block bei. Zeilen, die mit `#` beginnen, sind Kommentare und werden dem Endbenutzer nicht angezeigt.

Wir empfehlen Ihnen, **eine der vorgefertigten `.po`-Dateien** an einen neuen Speicherort zu **kopieren** (zum Beispiel neben Ihre Arbeitsmappe), die Kopie zu ändern und XLS Padlock anschließend auf Ihre neue Datei zu verweisen. Das Ändern von Dateien innerhalb des XLS Padlock-Installationsordners erfordert in der Regel Administratorrechte und kann durch ein zukünftiges Update überschrieben werden.

Migration von älteren `.sil`-Übersetzungsdateien

Wenn Sie die Benutzeroberfläche Ihrer geschützten Arbeitsmappe unter XLS Padlock 2025.3 oder früher angepasst haben, wurde Ihre Übersetzung als `.sil`-Datei gespeichert (das ältere textbasierte Übersetzungsformat, das von früheren Versionen verwendet wurde). XLS Padlock 2026 liest `.sil`-Dateien zur Kompilierungszeit nicht mehr. Das Format wurde durch `.po` ersetzt.

Um zu vermeiden, dass Sie irgendetwas erneut eingeben müssen, enthält die XLS Padlock-Distribution einen kleinen **SIL → PO-Migrationshelfer** unter `tools\sil_to_po\` im Quellbaum:

```
python tools/sil_to_po/sil2po_customer.py your_translation.sil --lang fr --stats
```

Dies erzeugt `customer_fr.po` im aktuellen Ordner, automatisch zusammengeführt mit der aktuellen kanonischen `.po`-Datei, sodass:

- die Spalte, die Ihrer Zielsprache entspricht, übernommen wird,
- Zeichenfolgen, die seit der Erstellung Ihrer `.sil`-Datei hinzugefügt wurden, mit einem leeren `msgstr` erscheinen (bereit zum Ausfüllen in Poedit),
- Zeichenfolgen, deren englische Quelle zwischen Versionen leicht abgewichen ist, mit `#, fuzzy` zur Überprüfung gekennzeichnet werden,
- Zeichenfolgen, die in XLS Padlock nicht mehr existieren, am Ende als veraltete Einträge `#~` angehängt werden (oder mit `--drop-orphans` vollständig verworfen werden).

Um mehrere Sprachen in einem Durchlauf zu migrieren, trennen Sie sie durch Kommas:

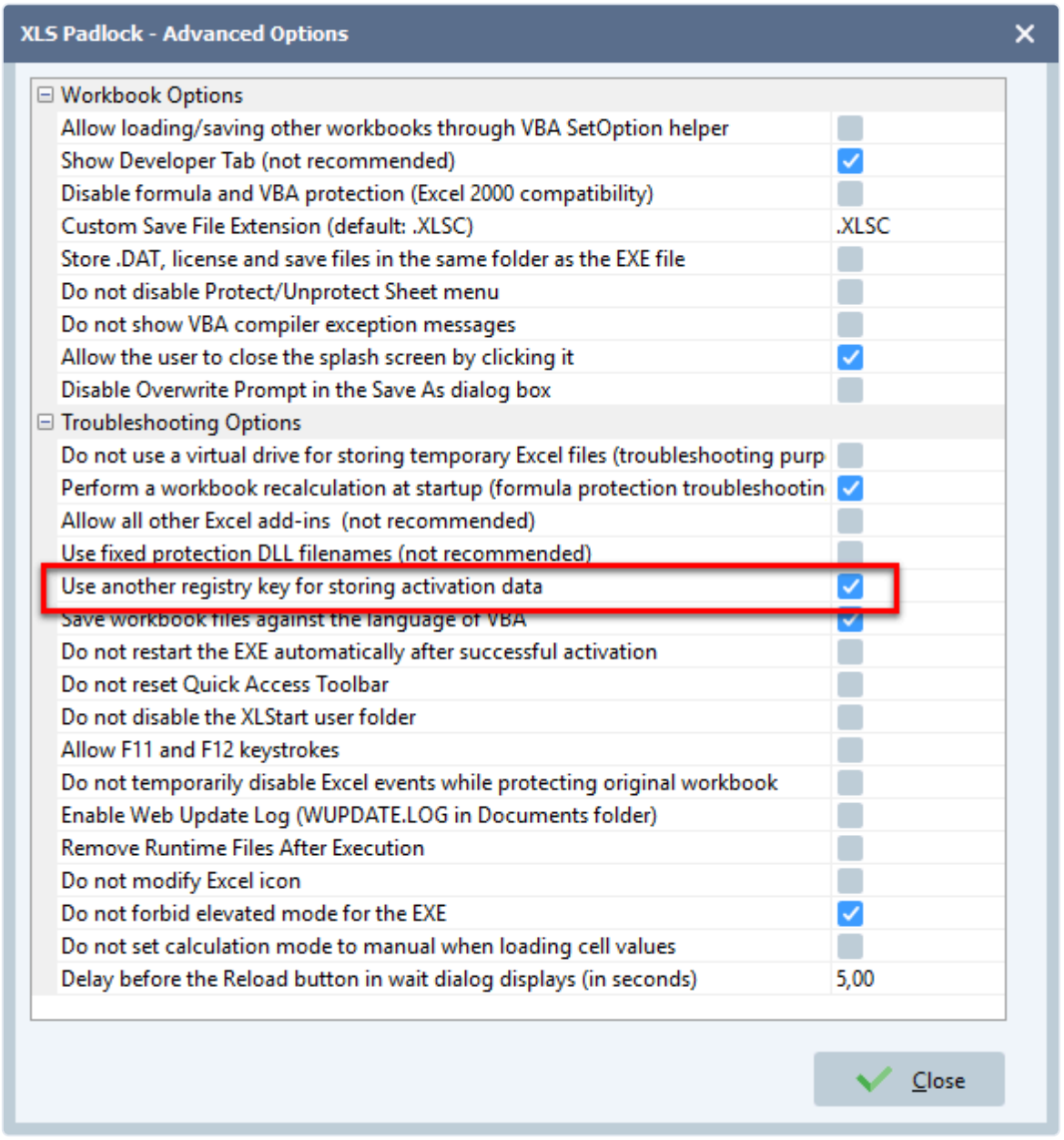
```
python tools/sil_to_po/sil2po_customer.py your_translation.sil --lang fr,es,de --stats
```

Die resultierenden `.po`-Dateien sind sofort über das Auswahlfeld *Translation file* im XLS Padlock Designer verwendbar. Das Helferskript und seine begleitende `README.md` (vollständige Optionsreferenz, Abgleichalgorithmus, durchgearbeitete Beispiele) werden im Unterordner `tools\sil_to_po\` des XLS Padlock-Installationsverzeichnisses ausgeliefert. Öffnen Sie `README.md` in einem beliebigen Texteditor, um die vollständige Dokumentation lokal zu lesen.

Python 3.8 oder höher ist erforderlich, um den Helfer auszuführen. Keine externen Python-Abhängigkeiten: Das Skript ist eigenständig.

Erweiterte Optionen

Klicken Sie in „Application Settings“ auf die Schaltfläche „Configure Advanced Options“, um das folgende Fenster anzuzeigen:



Nur für fortgeschrittene Benutzer

Diese Optionen sind für fortgeschrittene Benutzer bestimmt. Ändern Sie sie nur, wenn Sie deren Zweck verstehen oder wenn unser technischer Support Sie dazu anweist.

Verfügbare erweiterte Optionen

- **Allow loading/saving other workbooks through VBA SetOption helper**: eine Sicherheitsumgehung für bestimmte VBA-Operationen. Siehe Laden/Speichern von Arbeitsmappen über den VBA-SetOption-

Helfer.

- **Show Developer Tab**: macht die Excel-Registerkarte Entwicklertools sichtbar (aus Sicherheitsgründen nicht empfohlen).
- **Disable formula and VBA protection**: für die Kompatibilität mit Excel 2000.
- **Custom Save File Extension**: ändert die Standarderweiterung `.XLSC` für Speicherdateien.
- **Store .DAT, license and save files in the same folder as the EXE file**: aktiviert den portablen Modus.
- **Do not disable Protect/Unprotect Sheet menu**: hält die Menüeinträge für den Blattschutz aktiviert.
- **Do not show VBA compiler exception messages**: blendet die detaillierten Fehlermeldungen des VBA Compiler aus. Siehe Debug-Informationen deaktivieren.
- **Allow the user to close the splash screen by clicking it**: erlaubt Benutzern, den Begrüßungsbildschirm zu überspringen.
- **Disable Overwrite Prompt in the Save As dialog box**: verhindert die Meldung "do you want to overwrite?".
- **Do not use a virtual drive for storing temporary Excel files**: zu Zwecken der Fehlerbehebung.
- **Perform a workbook recalculation at startup**: zur Fehlerbehebung beim Formelschutz.
- **Allow all other Excel add-ins**: senkt die Sicherheit, indem alle Add-Ins aktiviert werden (nicht empfohlen).
- **Use fixed protection DLL filenames**: aus Sicherheitsgründen nicht empfohlen.
- **Use another registry key for storing activation data**: eine Problemumgehung für Registrierungskonflikte.
- **Save workbook files against the language of VBA**: behebt einige Lokalisierungsprobleme.
- **Do not restart the EXE automatically after successful activation**: verhindert den automatischen Neustart.
- **Do not reset Quick Access Toolbar**: bewahrt die Anpassungen des Benutzers an der Symbolleiste für den Schnellzugriff (QAT) auf.
- **Do not disable the XLStart user folder**: erlaubt die Ausführung von Add-Ins im XLStart-Ordner des Benutzers.
- **Allow F11 and F12 keystrokes**: aktiviert diese Funktionstasten.
- **Do not temporarily disable Excel events while protecting original workbook**: zu Zwecken der Fehlerbehebung.
- **Enable Web Update Log**: erstellt eine Datei `WUPDATE.LOG` im Dokumente-Ordner des Benutzers, um die Web-Update-Funktion zu debuggen.
- **Remove Runtime Files After Execution**: bereinigt temporäre Dateien.
- **Do not modify Excel icon**: verhindert, dass XLS Padlock das Symbol des Excel-Fensters ändert.
- **Do not forbid elevated mode for the EXE**: erlaubt der Anwendung, als Administrator ausgeführt zu werden.
- **Do not set calculation mode to manual when loading cell values**: zu Zwecken der Fehlerbehebung.
- **Delay before the Reload button in wait dialog displays (in seconds)**: passt den Wartedialog an.

Verarbeitungsfehler ignorieren

Die Option "Ignore errors when processing the workbook (internal protection)" (Fehler beim Verarbeiten der Arbeitsmappe ignorieren, interner Schutz) erlaubt es dem Kompilierungsvorgang, fortzufahren, auch wenn beim Öffnen Ihrer Excel-Arbeitsmappe Fehler auftreten. Diese Funktion wurde entwickelt, um den Schutz-Workflow zu vereinfachen, indem Fehlermeldungen unterdrückt werden, die den Vorgang andernfalls unterbrechen könnten.

Wenn diese Option aktiviert ist, hält XLS Padlock nicht an und zeigt keine Fehlermeldungen an, falls während der Öffnungsphase der Arbeitsmappe Probleme auftreten.

> **Diese Option behebt keine Fehler**
>
> Das Aktivieren dieser Option löst keine zugrunde liegenden Probleme innerhalb der Arbeitsmappe. Sie umgeht sie lediglich während der Kompilierung. Nachdem Sie Ihre Arbeitsmappe mit aktivierter Option geschützt haben, müssen Sie sie gründlich testen, um sicherzustellen, dass alle Funktionen wie erwartet arbeiten.

Falls Sie weiterhin Fehler in der kompilierten Arbeitsmappe feststellen, erwägen Sie, die alternative Option "[Excel-Automation für den Formelschutz verwenden](#chapter-formula-protection-method)" zu aktivieren, die Probleme beheben kann, die die standardmäßige interne Schutzmethode nicht löst.

Debug-Informationen deaktivieren

Standardmäßig zeigt der VBA Compiler eine detaillierte Fehlermeldung an, wenn er zur Laufzeit einen Fehler im kompilierten Code erkennt, um Ihnen beim Debuggen des Problems zu helfen.

Wenn Sie diese Informationen nicht für Ihre Endbenutzer anzeigen möchten, können Sie sie deaktivieren. Gehen Sie zu Configure Advanced Options und schalten Sie die Option „**Do not show VBA compiler exception messages**“ aus.

Benutzerdefinierte Excel-Oberfläche

Wie in der Dokumentation von Microsoft erläutert, können Sie die Excel-Benutzeroberfläche anpassen, indem Sie eine speziell erstellte `.officeUI`-Datei in einem bestimmten Benutzerordner ablegen.

XLS Padlock ermöglicht es Ihnen, Ihre eigene Excel.officeUI für die geschützte Anwendung anzugeben. Sie ersetzt automatisch die Standard-Benutzeroberfläche des Benutzers (sofern vorhanden).

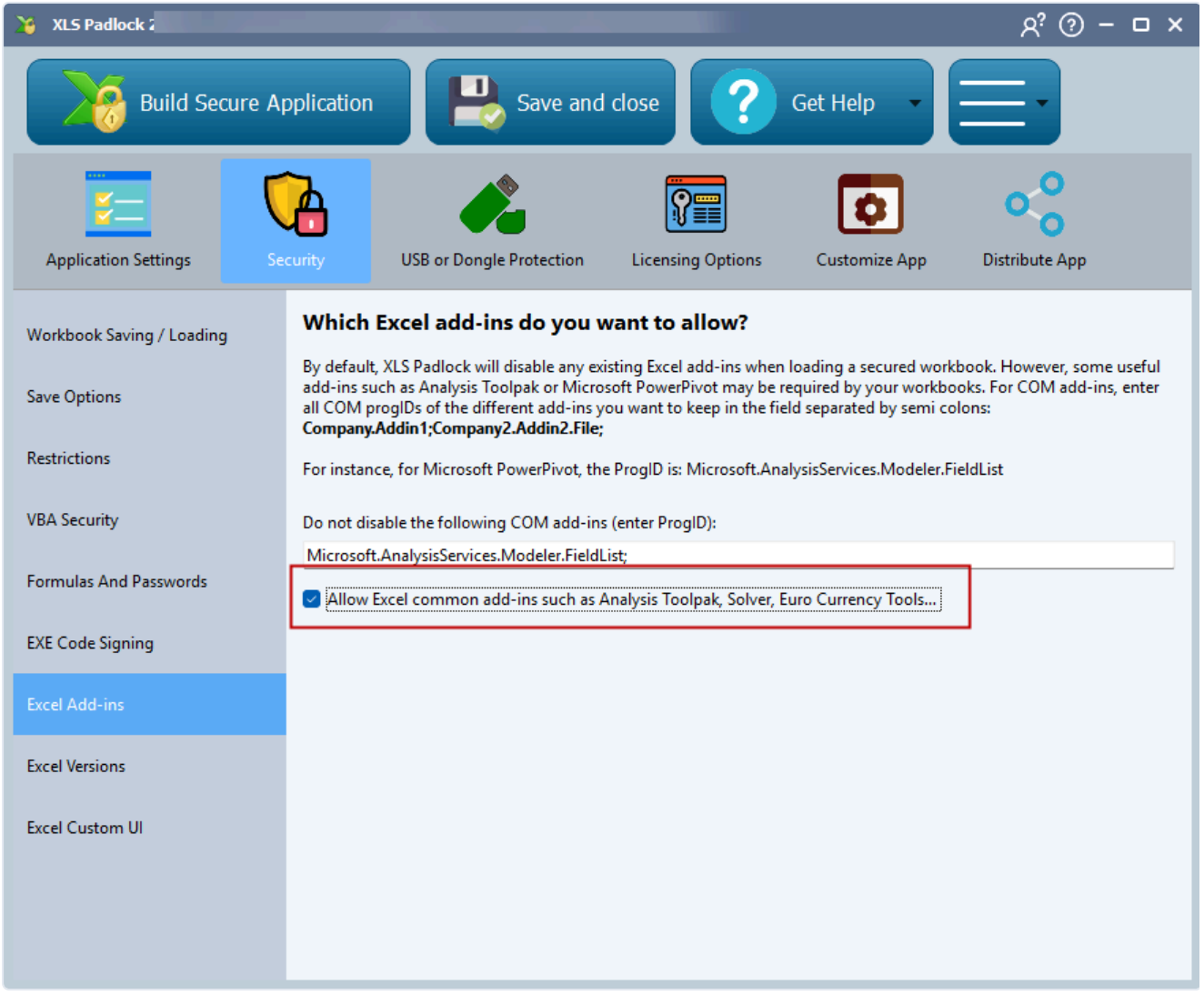
Sie müssen lediglich den vollständigen Pfad zu Ihrer `.officeUI`-Datei angeben. XLS Padlock kompiliert sie direkt in die endgültige EXE, sodass Sie die `.officeUI`-Datei nicht separat bereitstellen müssen.

Weitere Informationen zum Erstellen benutzerdefinierter UI-XML-Dateien finden Sie in der offiziellen Dokumentation von Microsoft Office.

Excel-Add-Ins

Standardmäßig deaktiviert XLS Padlock die meisten Excel-Add-Ins, wenn eine geschützte Arbeitsmappe geladen wird. Dies ist eine Sicherheitsmaßnahme, um eine kontrollierte Umgebung für Ihre Anwendung zu schaffen.

Wenn Ihre Arbeitsmappe jedoch auf bestimmte Add-Ins angewiesen ist, um korrekt zu funktionieren, können Sie diese gezielt wieder aktivieren. Diese Seite erläutert, wie Sie sowohl COM-Add-Ins als auch gängige integrierte Excel-Add-Ins zulassen.



Bestimmte COM-Add-Ins aktiviert lassen

Wenn Ihre Arbeitsmappe ein COM-Add-In benötigt (wie Microsoft PowerPivot), müssen Sie dessen `ProgID` angeben, um zu verhindern, dass XLS Padlock es deaktiviert.

Geben Sie im Feld "Do not disable the following COM add-ins" (Die folgenden COM-Add-Ins nicht deaktivieren) die `ProgID` für jedes Add-In ein, das Sie aktiviert lassen möchten. Wenn Sie mehrere haben, trennen Sie diese durch Semikolons (`;`).

Beispiel: Um Microsoft PowerPivot aktiviert zu lassen, würden Sie dessen ProgIDs eingeben:
`Microsoft.AnalysisServices.Modeler.FieldList;PowerPivotExcelClientAddIn.NativeEntry.1`

> **Eine ProgID finden**
>
> Die `ProgID` (programmatischer Bezeichner) ist ein eindeutiger Name, der in der Windows-Registrierung für eine COM-Komponente gespeichert ist. Sie finden ihn oft in der Dokumentation des Add-Ins oder durch Untersuchen der Registrierung. Microsoft stellt weitere Informationen zu diesem Thema auf seiner Support-Website bereit.

Gängige Excel-Add-Ins zulassen

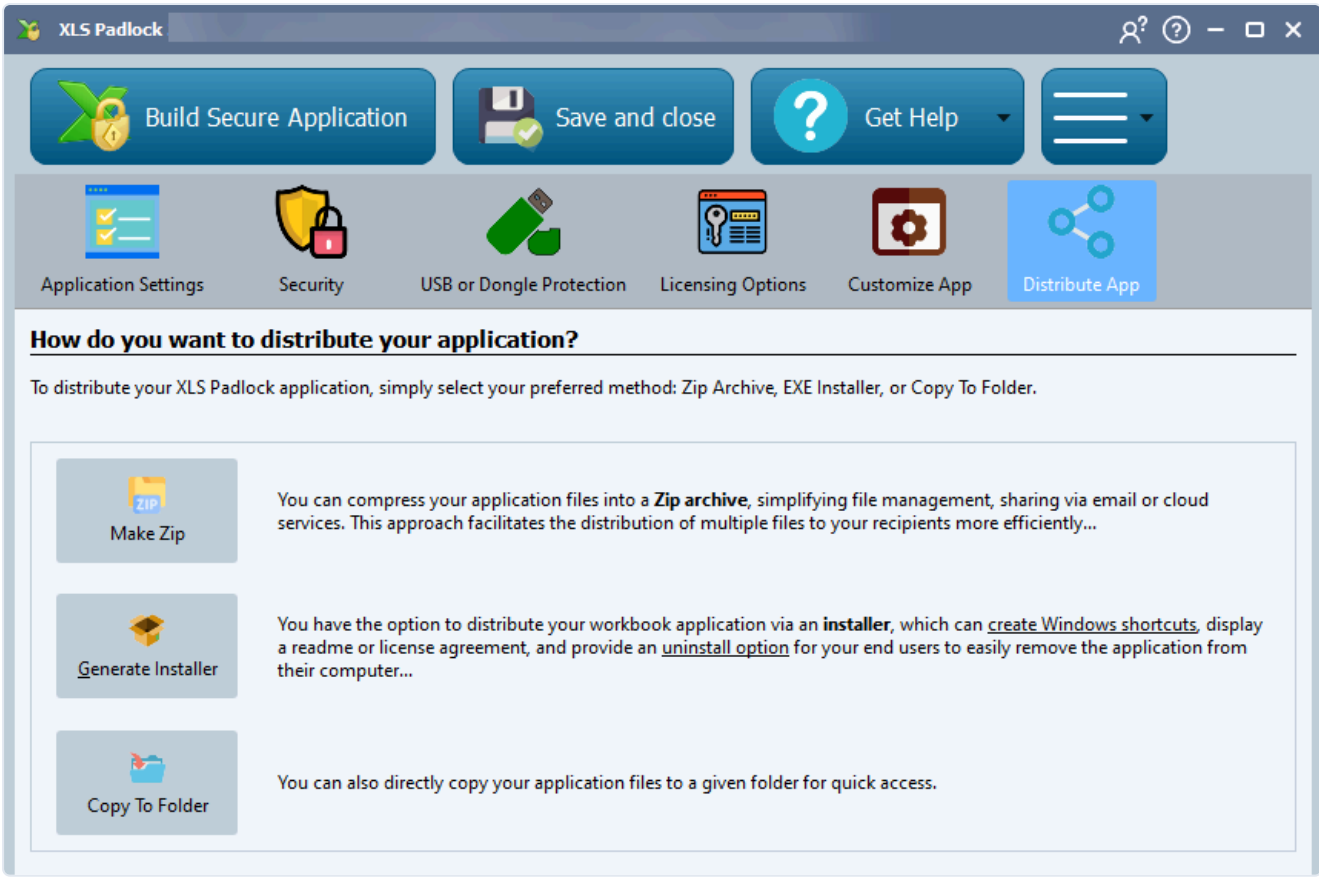
Excel enthält außerdem mehrere beliebte integrierte Add-Ins, wie das **Analysis ToolPak**, den **Solver** und die **Euro Currency Tools**.

Wenn Ihre Arbeitsmappe Funktionen eines dieser Add-Ins verwendet, aktivieren Sie die Option **"Allow Excel common add-ins..."** (Gängige Excel-Add-Ins zulassen). Dadurch wird sichergestellt, dass sie beim Start Ihrer geschützten Anwendung geladen und verfügbar sind.

Anwendung verteilen

Nachdem Sie Ihre Arbeitsmappe in eine ausführbare Datei umgewandelt haben (.EXE), besteht der nächste Schritt darin, sie an Ihre Benutzer zu verteilen.

XLS Padlock bietet mehrere Methoden, um diesen Vorgang zu vereinfachen: das Erstellen eines Zip-Archivs, das Erstellen eines EXE-Installationsprogramms oder das direkte Kopieren der Dateien in einen Ordner.



Zip-Archiv

Ein Zip-Archiv ist eine beliebte und unkomplizierte Methode zur Verteilung von Software. Das Komprimieren Ihrer Anwendung in eine einzige `.zip`-Datei vereinfacht die Dateiverwaltung für Ihre Benutzer. Diese Methode ist besonders nützlich, um Ihre Anwendung per E-Mail oder über Cloud-Dienste weiterzugeben, da sie die Dateigröße verringert und alle erforderlichen Komponenten in einem Paket bündelt.

Um ein Zip-Archiv Ihrer Anwendung zu erstellen, klicken Sie auf **Make Zip** (Zip erstellen) und wählen Sie aus, wo Sie die `.zip`-Datei speichern möchten.

EXE-Installationsprogramm

Für einen professionelleren Eindruck können Sie Ihre Anwendung mit einem EXE-Installationsprogramm verteilen. Diese Methode verbessert das Benutzererlebnis, indem sie einen vertrauten Installationsvorgang bietet, der in Windows-Umgebungen üblich ist. Ein Installationsprogramm kann Verknüpfungen auf dem Desktop und im Startmenü erstellen, eine Readme-Datei oder einen Lizenzvertrag anzeigen und einen Deinstaller bereitstellen, sodass Benutzer die Anwendung problemlos von ihrem Computer entfernen können.

➡️ Ausführliche Anweisungen finden Sie in unserer Anleitung dazu, wie Sie ein Installationsprogramm für Ihre Anwendung erstellen.

In einen Ordner kopieren

Schließlich ist das direkte Kopieren Ihrer Anwendungsdateien in einen Ordner die unkomplizierteste Verteilungsmethode. Sie umfasst zwei einfache Schritte:

- Kopieren Sie alle erforderlichen Anwendungsdateien in einen Ordner.
- Geben Sie diesen Ordner über ein physisches Medium (wie einen USB-Stick) oder per Netzwerkübertragung an Ihre Benutzer weiter.

Auch wenn diese Methode weniger ausgereift ist als die anderen, ist sie der schnellste Weg, um Ihre Anwendung auf einem anderen System zum Laufen zu bringen, da sie keine speziellen Zip-Programme oder Installationssoftware erfordert.

👉 Siehe auch:

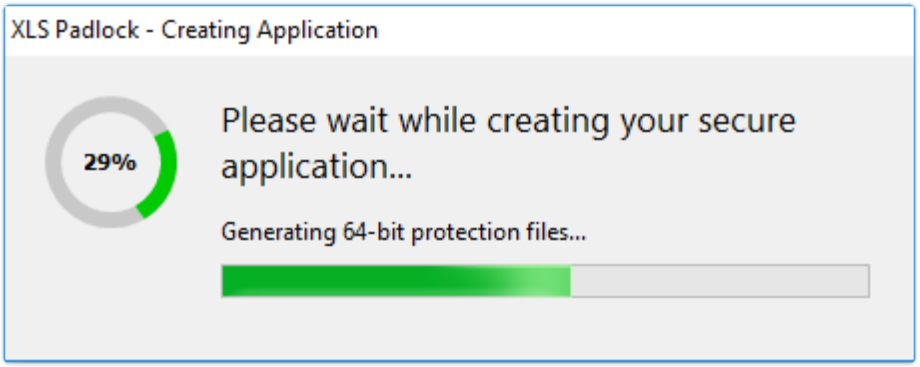
- Wie Sie Ihre Anwendung mit einer Codesignatur versehen

Geschützte Arbeitsmappe verteilen

Kompilieren Sie Ihre Excel-Arbeitsmappe zu einer EXE

Wenn Sie in XLS Padlock auf „**Build Secure Application**“ (Sichere Anwendung erstellen)

klicken, wird das Dialogfeld mit dem Fortschritt der Kompilierung angezeigt:



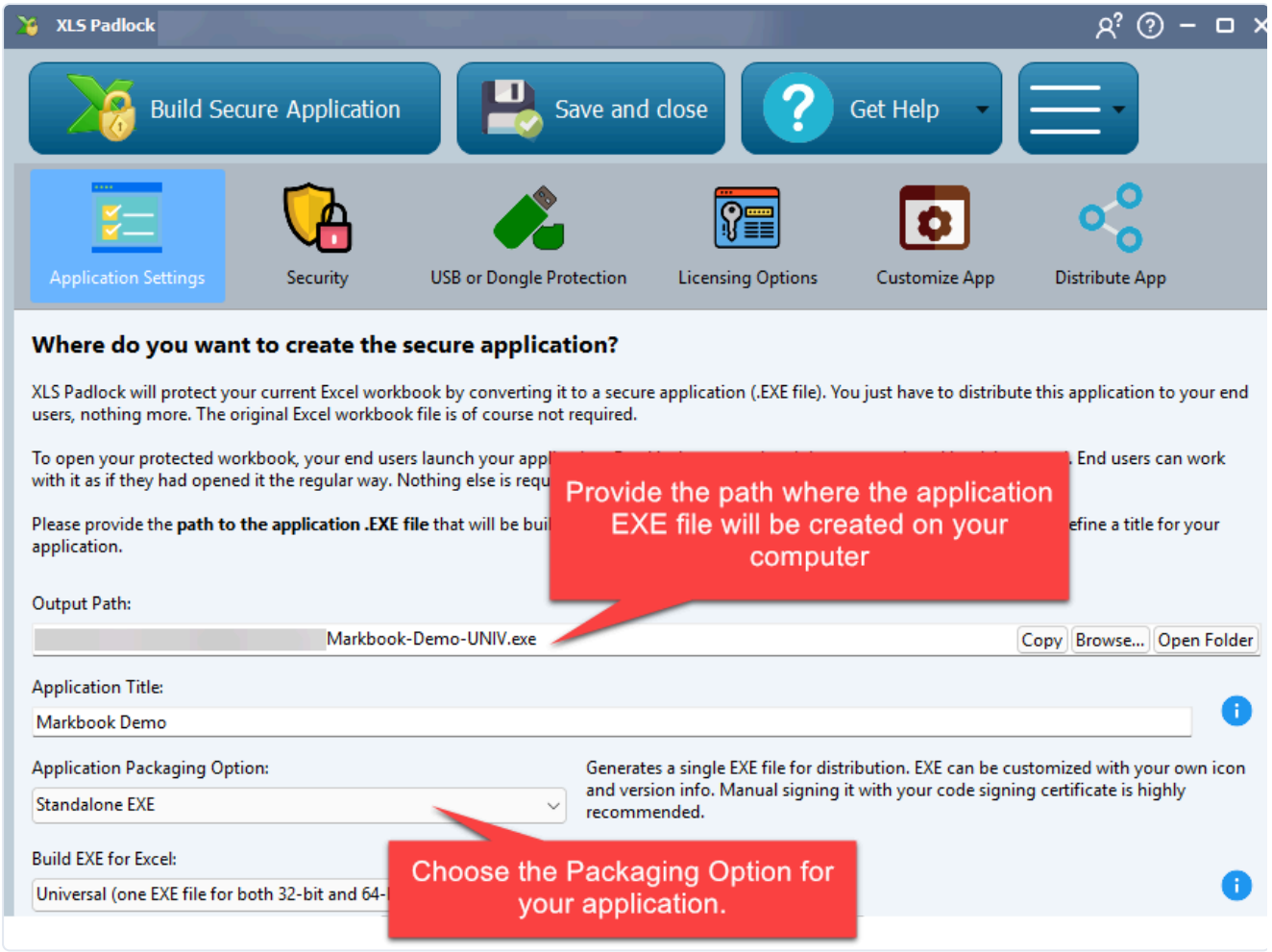
Sobald der Vorgang abgeschlossen ist, **ist Ihre sichere Anwendung bereit**. XLS Padlock hat aus Ihrer Excel-Arbeitsmappe eine **ausführbare Datei** (.EXE) erzeugt.

▶️ **Um nun auf die geschützte Arbeitsmappe zuzugreifen, starten die Endbenutzer diese .EXE-Datei. Eine lokale Kopie von Microsoft Excel ist die einzige weitere Voraussetzung.** Ihre ursprüngliche Arbeitsmappendatei wird **NICHT MEHR** benötigt.

- Beim Starten der .EXE-Datei startet Excel und die geschützte Arbeitsmappe wird geöffnet. Die Endbenutzer können damit interagieren, als ob sie sie auf die übliche Weise geöffnet hätten.
- Bestimmte Excel-Funktionen wie „New Workbook“, Open und Save (optional) sind aus Sicherheitsgründen deaktiviert.
- Wenn Excel auf dem Computer des Benutzers nicht gefunden wird, wird eine Fehlermeldung angezeigt. Sie können Ihre Anwendung so konfigurieren, dass bestimmte Excel-Versionen erforderlich sind.

Optionen für das Packaging der Anwendung

XLS Padlock bietet zwei Formate für Ihre sichere Anwendung an, was beeinflusst, wie Sie sie verteilen und wie Sie Sicherheitswarnungen wie Windows SmartScreen handhaben.



Eigenständige EXE

XLS Padlock erstellt eine einzige ausführbare Datei (.EXE) aus Ihrer Arbeitsmappe. Sie verteilen einfach diese einzelne Datei an Ihre Endbenutzer. Die ursprüngliche Excel-Arbeitsmappe wird nicht benötigt.

> **Code-Signierung wird dringend empfohlen**
>
> Wir raten Ihnen dringend, Ihre EXE-Dateien digital zu signieren. Die Verteilung einer unsignierten EXE-Datei führt wahrscheinlich dazu, dass Windows SmartScreen eine Warnung „Unknown Application“ anzeigt. Ihre EXE wird zwar trotzdem ausgeführt, aber eine digitale Signatur schafft Vertrauen und sorgt für eine bessere Benutzererfahrung.

Anwendungspaket EXE + XPLAPP

Diese Option erzeugt eine EXE-Datei, eine Begleitdatei `.bin64` und eine separate Datendatei `.xplapp`. Die Haupt-EXE ist von unserem Unternehmen vorab signiert, was die Erkennung durch Windows SmartScreen und Antivirensoftware erleichtert.

- Sie müssen alle drei Dateien zusammen verteilen (`.exe` , `.bin64` und `.xplapp`).
- XLS Padlock kann diese Dateien zur einfacheren Verteilung in einem einzigen Zip-Archiv oder einem Installationsprogramm bündeln.

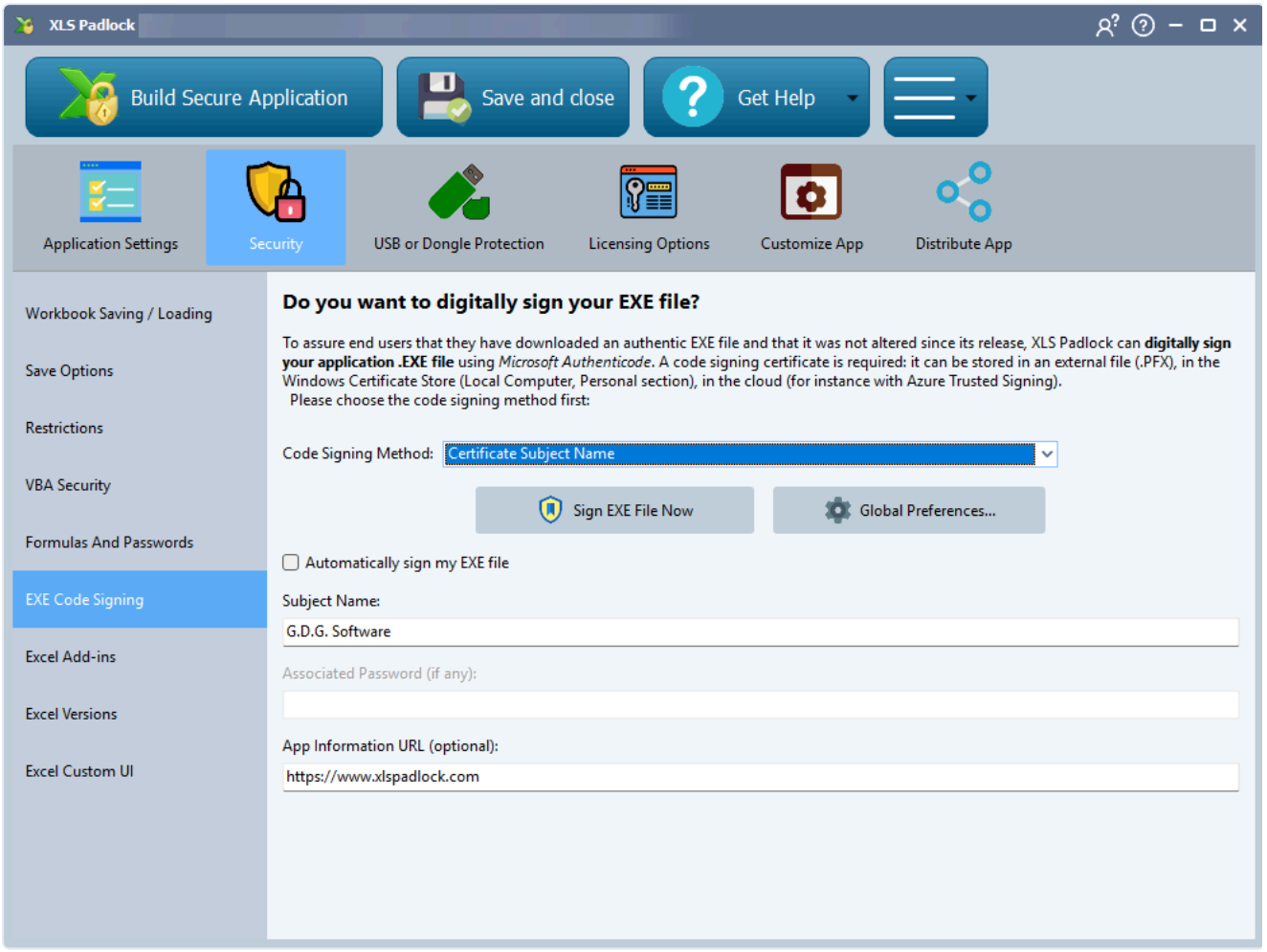
- Diese Option reduziert das Risiko von „Unknown Application“-Warnungen und Antiviren-Fehlalarmen erheblich.

Empfehlungen zur Verteilung

1. **Signieren Sie Ihre EXE-Dateien digital:** Dies ist der wichtigste Schritt, um Antiviren-Fehlalarme zu vermeiden und Vertrauen bei Ihren Benutzern aufzubauen. XLS Padlock kann diesen Vorgang automatisieren, wenn Sie über ein Code-Signing-Zertifikat verfügen.
2. **Deaktivieren Sie Ihr Antivirenprogramm vorübergehend:** Wenn Sie ein aktives Antivirenprogramm haben, sollten Sie es vor dem Kompilieren Ihrer Arbeitsmappen deaktivieren. Antivirensoftware kann die Erstellung neuer EXE-Dateien auf Ihrem Computer manchmal fälschlicherweise melden.
3. **Versenden Sie EXE-Dateien nicht per E-Mail:** Die meisten E-Mail-Anbieter blockieren `.exe`-Dateien. Laden Sie stattdessen die Dateien Ihrer Anwendung auf einen Filehosting-Dienst wie Dropbox, Google Drive oder Ihren eigenen Webserver hoch und teilen Sie den Download-Link. Sie können Ihre Anwendung auch in einem Zip-Archiv verpacken.

EXE digital signieren

Wenn Sie Ihre **eigenständige EXE-Datei der Arbeitsmappe digital signieren**, versichern Sie den Endbenutzern, dass die Datei authentisch ist und nicht manipuliert wurde. Dieser Vorgang, auch als Code Signing bezeichnet, verwendet die Technologie Microsoft Authenticode®, um zu überprüfen, dass der Code von einem vertrauenswürdigen Herausgeber stammt.



XLS Padlock vereinfacht den Code-Signing-Vorgang, indem es die erforderlichen Schritte intern abwickelt.

> **Warum Code Signing wichtig ist**
>
> Wenn Sie Ihre Anwendungen über das Internet verteilen möchten, wird Code Signing dringend empfohlen. Es hilft zu verhindern, dass Webbrowser und Windows Warnungen zu einem "Unidentified Publisher" anzeigen, und kann zudem die Wahrscheinlichkeit von Fehlalarmen durch Antivirensoftware verringern.

![Windows-Sicherheitswarnung, die für eine nicht signierte EXE-Datei einen Unidentified Publisher anzeigt](file:///D:/websites/xlspadlock/astro/src/content/docs/de/doc/code-sign-your-exe-file-digital-signature/images/image_97.png)

So erhalten Sie ein Code Signing Certificate

Um Ihre Anwendung zu signieren, benötigen Sie ein gültiges **Code Signing Certificate** von einer vertrauenswürdigen Zertifizierungsstelle (CA) wie Sectigo oder Digicert. Andere Zertifikatstypen wie SSL/TLS sind nicht kompatibel.

> **Cloud-basierte Zertifikate**
>
> Eine zunehmend beliebte und kostengünstige Alternative ist **Azure Trusted Signing**, ein cloudbasierter Dienst von Microsoft. Er macht physische USB-Token überflüssig und vereinfacht den Verwaltungsprozess. XLS Padlock unterstützt die Signierung mit Azure Trusted Signing vollständig.

Token-basierte Zertifikate (HSMs)

Seit dem 1. Juni 2023 müssen alle privaten Schlüssel neuer Code-Signing-Zertifikate auf sicherer Hardware gespeichert werden, beispielsweise auf einem nach FIPS 140-2 Level 2 zertifizierten USB-Token oder einem Hardware Security Module (HSM). Dies erhöht die Sicherheit, indem ein Diebstahl der Schlüssel verhindert wird. XLS Padlock arbeitet nahtlos mit token-basierten Zertifikaten zusammen. Stellen Sie lediglich sicher, dass das Token an Ihren Computer angeschlossen ist, wenn Sie Ihre Anwendung erstellen.

Code Signing in XLS Padlock konfigurieren

Navigieren Sie in der Oberfläche von XLS Padlock zur Registerkarte **Security -> EXE Code Signing** (Sicherheit, EXE-Code-Signierung). Um die Signierung zu aktivieren, wählen Sie die gewünschte **Code Signing Method** (Code-Signing-Methode):

- **PFX File:** Verwendet ein in einer `.pfx`-Datei gespeichertes Zertifikat. Dies ist eine ältere Methode für ältere Zertifikate.
- **Certificate Subject Name:** Sucht das Zertifikat im Windows-Zertifikatspeicher anhand seines Subject Name (Antragstellername). Dies ist eine gängige Methode für Zertifikate auf Hardware-Token.
- **Certificate Thumbprint:** Sucht das Zertifikat im Windows-Zertifikatspeicher anhand seines eindeutigen Thumbprint (Fingerabdruck, ein SHA-1-Hash). Dies ist oft die zuverlässigste Methode.
- **SignTool Commands:** Eine erweiterte Methode, mit der Sie benutzerdefinierte Befehle für das Microsoft-Dienstprogramm `SignTool.exe` bereitstellen können, was maximale Flexibilität bietet.
- **Azure Trusted Signing:** Signiert Ihre Anwendung mit dem cloudbasierten Dienst von Microsoft. Weitere Einzelheiten finden Sie in unserem Azure Trusted Signing Tutorial.

  Azure CLI erforderlich

  Um diese Methode zu verwenden, müssen Sie zunächst die Microsoft Azure CLI installieren und sich mit `az login` anmelden.

Durchführen der Signierung

- **Manuelle Signierung:** Klicken Sie auf **Sign EXE File Now**, um die zuletzt erstellte EXE-Datei sofort zu signieren.
- **Automatische Signierung:** Aktivieren Sie **Automatically sign my EXE file**, damit XLS Padlock die EXE-Datei bei jeder Erstellung Ihrer Anwendung signiert.

> **Fehlerbehebung**
>
> Wenn ein Signierungsfehler auftritt, prüfen Sie die detaillierten Meldungen im Kompilierungsprotokoll von XLS Padlock. Diese Protokolldatei heißt in der Regel `[Your Workbook Filename].xplcompil.log` und befindet sich im selben Verzeichnis wie Ihre Arbeitsmappe.

Installationsprogramm erstellen

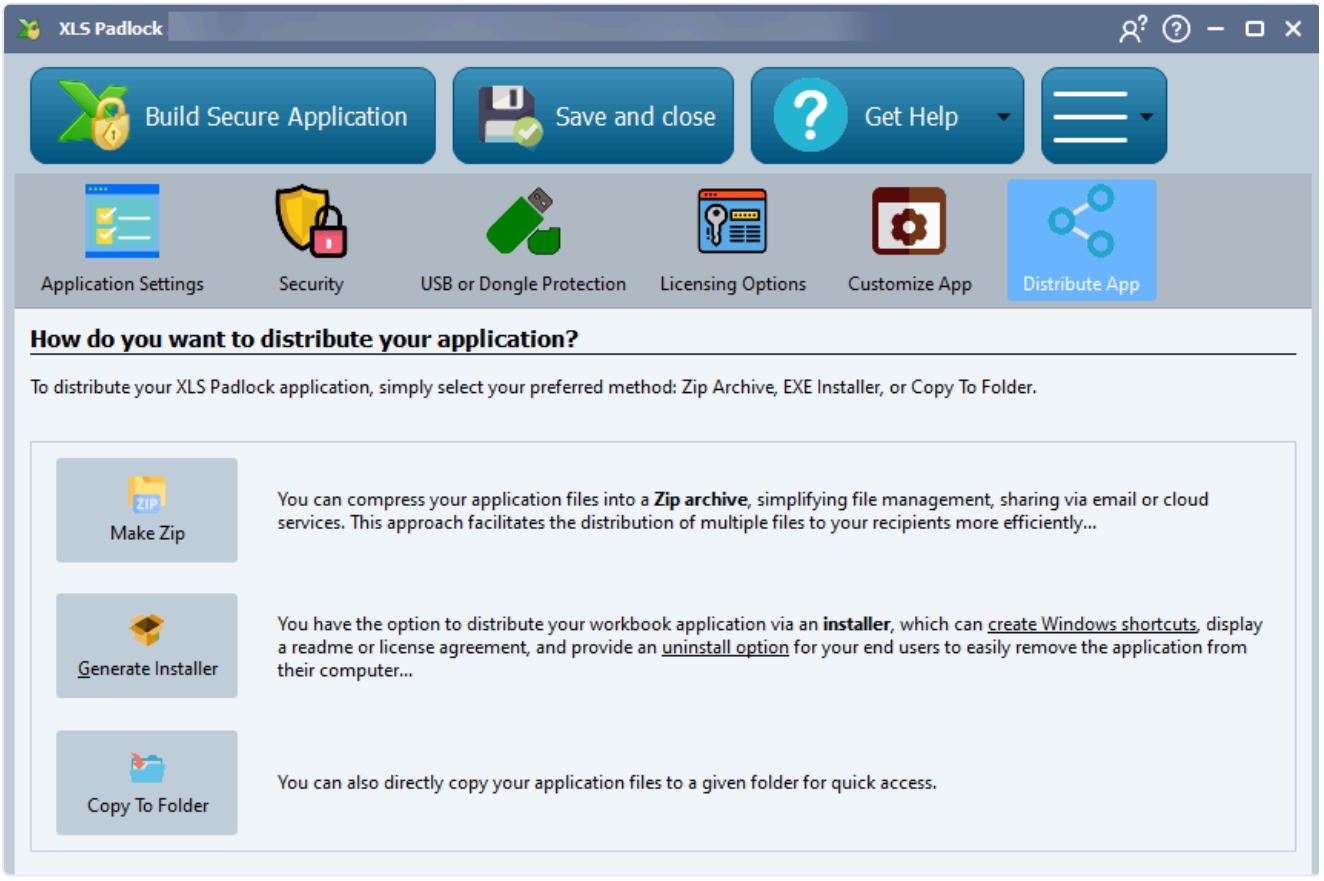
Das Verpacken Ihrer kompilierten Arbeitsmappe in ein professionelles Installationsprogramm (oder Setup-Programm) ist eine hervorragende Möglichkeit, Ihre Anwendung zu verteilen. Installationsprogramme bieten eine vertraute Benutzererfahrung und können Aufgaben wie das Installieren mehrerer Dateien (etwa einer Readme-Datei), das Anzeigen einer Lizenzvereinbarung, das Erstellen von Verknüpfungen und die Bereitstellung eines Deinstallationsprogramms für eine saubere Entfernung übernehmen.

XLS Padlock lässt sich in unsere Software Paquet Builder integrieren, um benutzerdefinierte und kompakte Installationsprogramme zu erzeugen. **Paquet Builder muss auf Ihrem Computer installiert sein, um diese Funktion nutzen zu können.**

Paquet Builder ist ein leistungsstarkes Werkzeug, das einen Ersteller selbstextrahierender 7-Zip-Archive mit einem Generator für Setup-Routinen kombiniert. Es ermöglicht Ihnen, flexible und kompakte Installationsprogramme für die professionelle Softwareverteilung zu erstellen, Dokumente und Programmdateien zu verpacken, mehrsprachige Installationspakete zu erstellen, Updates zu erzeugen und mehrere Dateien für eine einfache Online-Verteilung in einer einzigen `.exe` *zu bündeln.*

So erstellen Sie ein Installationsprogramm:

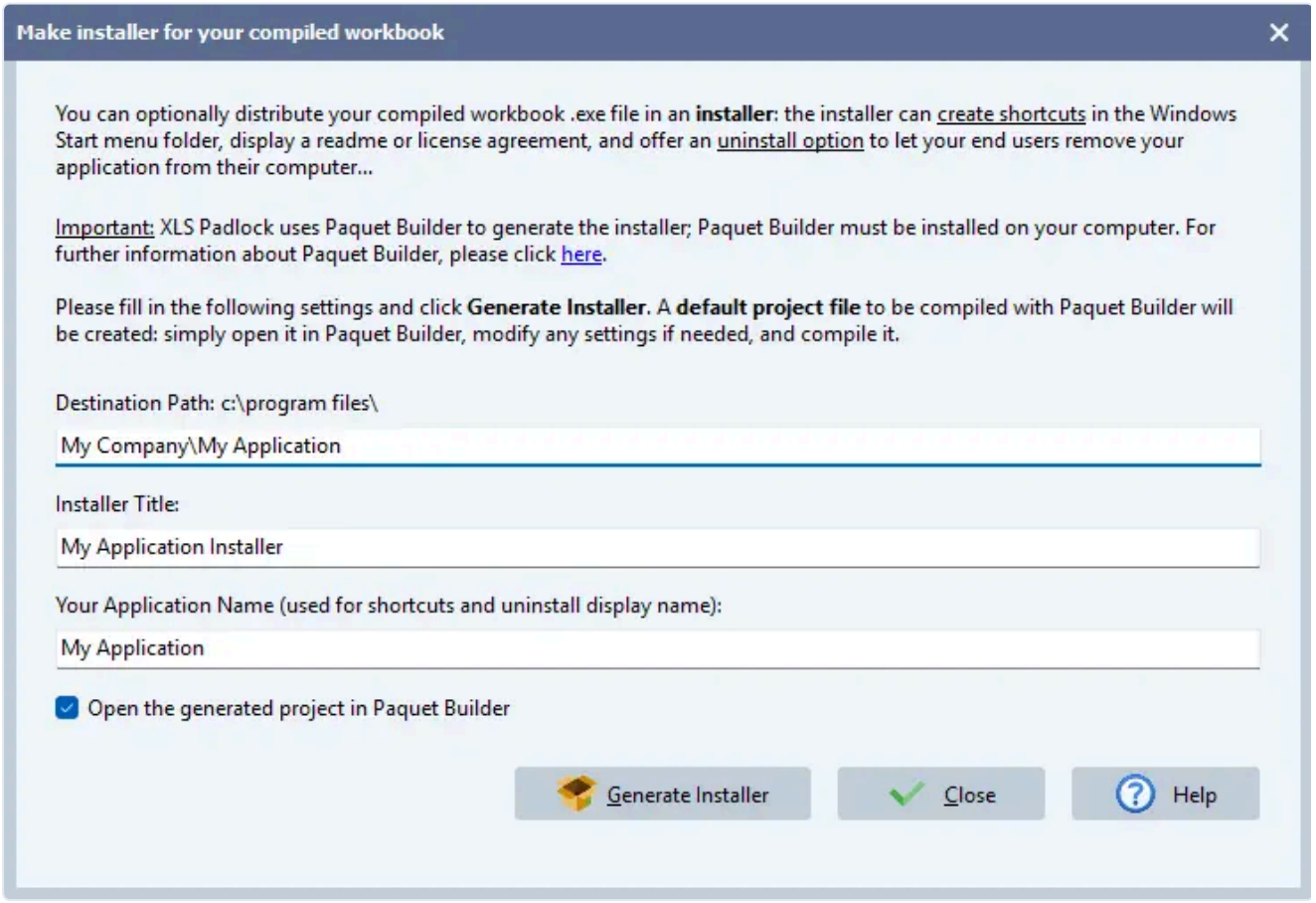
1. Klicken Sie im Reiter "Distribute" von XLS Padlock auf **Generate Installer** (Installationsprogramm erstellen). Dadurch wird das Fenster "Make Setup for your compiled workbook" geöffnet.



1. Füllen Sie die Felder "Destination path" (Standard-Installationsordner), "Setup title" (Fenstertitel des Installationsprogramms) und "Your Application Name" (für Verknüpfungen, Deinstallationsprogramm

usw.) aus.

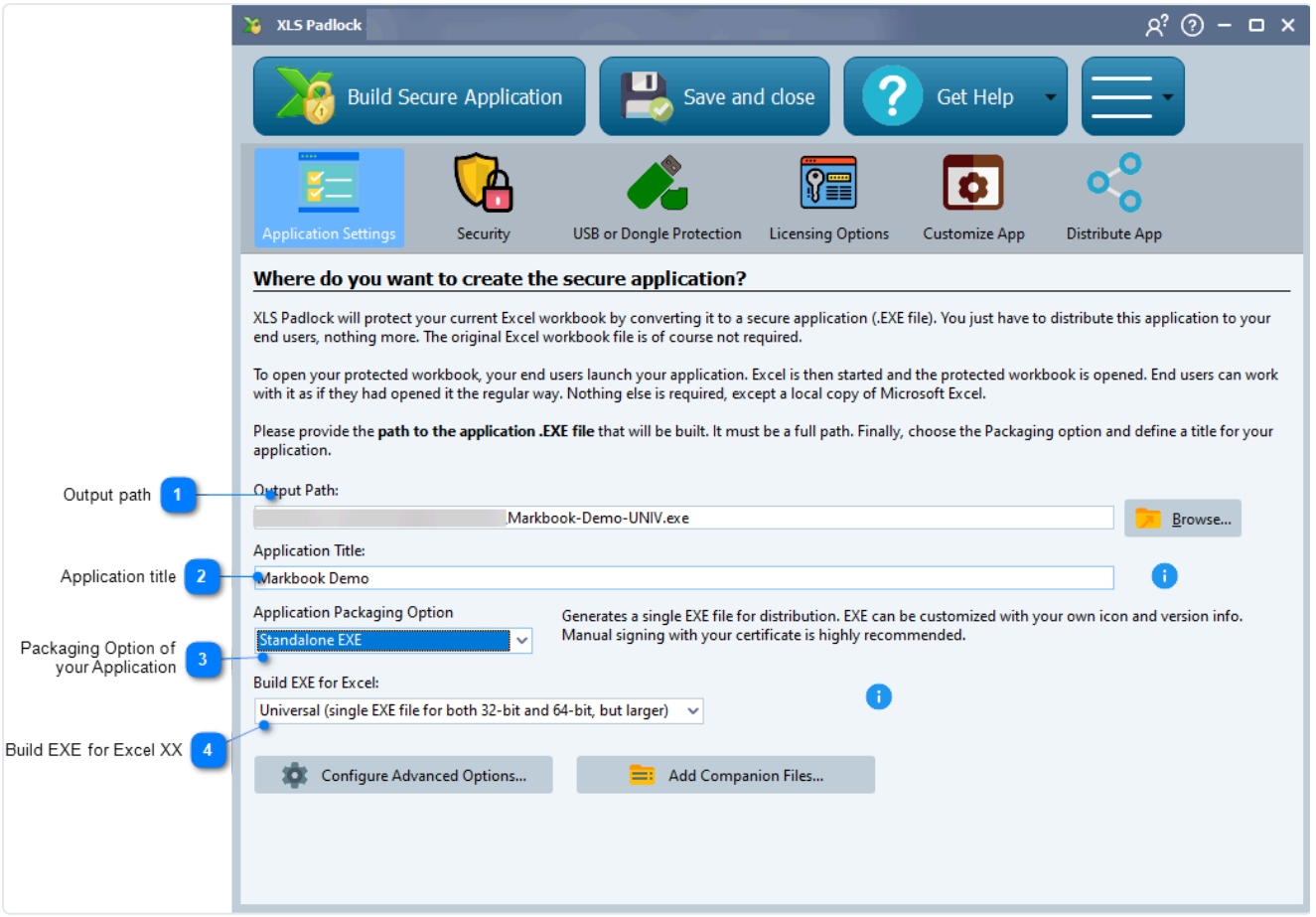
2. Drücken Sie **Generate Installer**, um das Paquet Builder-Projekt zu erstellen. XLS Padlock startet daraufhin Paquet Builder und ermöglicht Ihnen, das Projekt weiter anzupassen und das endgültige Installationsprogramm zu erstellen.



👉 Lesen Sie das vollständige Tutorial, wie Sie ein Installationsprogramm für Ihre Excel-Arbeitsmappe erstellen.

EXE für die Excel-Bitversion erstellen

Microsoft Excel ist in zwei Architekturen verfügbar: 32-Bit und 64-Bit. Da 64-Bit mittlerweile der Standard für moderne Office-Versionen ist, benötigen die meisten Ihrer Benutzer eine 64-Bit-kompatible Anwendung. Einige verwenden jedoch möglicherweise noch die 32-Bit-Version, insbesondere auf älteren Systemen. Mit XLS Padlock können Sie für beide Plattformen kompilieren.



Diese Option ist nur für eigenständige ausführbare Dateien verfügbar.

👉 Wählen Sie den Typ der EXE-Datei aus, den XLS Padlock generieren soll:

- **32-bit only**: Generiert eine EXE-Datei, die nur mit 32-Bit-Versionen von Excel kompatibel ist.
- **64-bit only**: Generiert eine EXE-Datei, die nur mit 64-Bit-Versionen von Excel kompatibel ist.
- **Universal**: Erstellt eine einzige EXE-Datei, die sowohl mit 32-Bit- als auch mit 64-Bit-Versionen von Excel funktioniert. Dies vereinfacht die Verteilung, aber die resultierende EXE-Datei ist etwa doppelt so groß.
- **32-bit and 64-bit**: Generiert gleichzeitig zwei separate EXE-Dateien. Um zu verhindern, dass sie sich gegenseitig überschreiben, fügt XLS Padlock den Dateinamen automatisch die Suffixe "32" und "64" hinzu (z. B. `MyApp32.exe` und `MyApp64.exe`).

Versionskonflikt

Wenn ein Endbenutzer eine EXE mit einer nicht übereinstimmenden Office-Architektur ausführt (z. B. eine 64-Bit-EXE auf einer 32-Bit-Office-Installation), erhält er eine Fehlermeldung mit der Aufforderung, Sie zwecks der korrekten Version zu kontaktieren.

Codesignierung für den Universal-Modus empfohlen

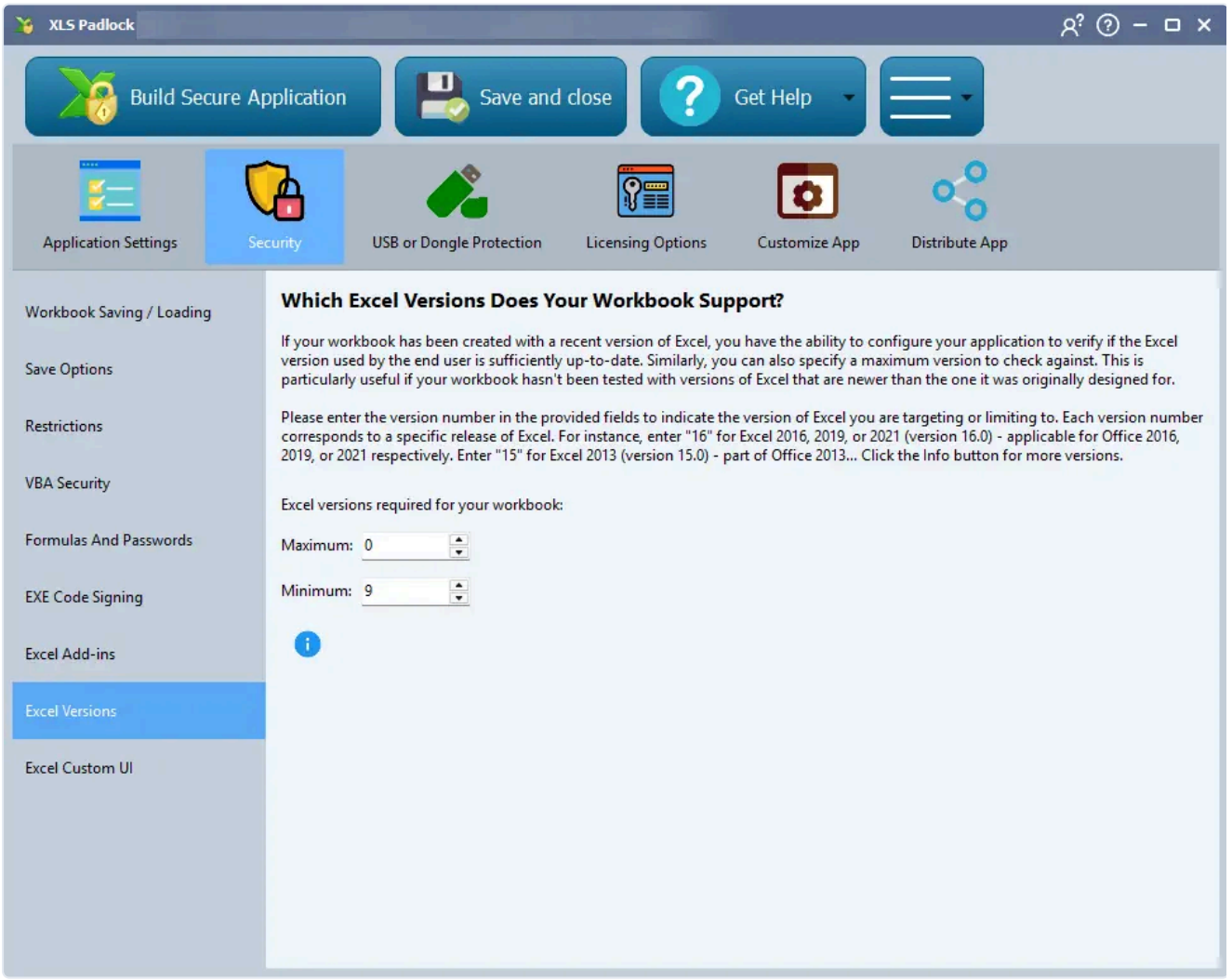
Wenn Sie den Universal-Modus verwenden, empfehlen wir Ihnen dringend, Ihre Anwendung mit einer Codesignatur zu versehen. Der Grund dafür ist, dass zur Laufzeit eine temporäre EXE-Datei erstellt wird, die manchmal von Antivirensoftware gemeldet werden kann. Eine digitale Signatur verringert das Risiko von Fehlalarmen erheblich.

Info

Standardmäßig sind die im Bundle-Format verteilten Anwendungen bereits universell.

Excel-Versionen

Wenn Ihre Arbeitsmappe Funktionen einer bestimmten Excel-Version benötigt, können Sie Ihre Anwendung so konfigurieren, dass sie die Excel-Version des Endbenutzers zur Laufzeit überprüft.



Sie können sowohl eine erforderliche Mindestversion als auch eine maximal zulässige Version festlegen. Dies ist nützlich, um die Kompatibilität sicherzustellen oder die Verwendung mit neueren, ungetesteten Excel-Versionen zu verhindern.

Geben Sie die entsprechende Versionsnummer in die Felder ein:

- **9**: Excel 2000 (Version 9.0)
- **10**: Excel 2002 (Version 10.0)
- **11**: Excel 2003 (Version 11.0)
- **12**: Excel 2007 (Version 12.0)
- **14**: Excel 2010 (Version 14.0)
- **15**: Excel 2013 (Version 15.0)
- **16**: Excel 2016, 2019, 2021 und 365 (Version 16.0)

Updates der Arbeitsmappe

Nachdem Sie Ihre Excel-Arbeitsmappen-Anwendung verteilt haben, müssen Sie früher oder später Aktualisierungen für Ihre Benutzer bereitstellen. XLS Padlock bietet mehrere Werkzeuge, um diesen Vorgang zu vereinfachen.

Ein wichtiger Aspekt bei Aktualisierungen ist die Art und Weise, wie Ihre Anwendung Benutzerdaten speichert. Sie sollten die verschiedenen von XLS Padlock angebotenen Speichermodi kennen, bevor Sie Ihre Anwendung erstellen.

Wenn Ihre Arbeitsmappe häufig aktualisiert wird und Ihre Benutzer nur einige wenige Zellwerte ändern müssen, ist der Modus **Save defined cell values only (.XLSCE file)** (nur definierte Zellwerte speichern) die beste Wahl. Mit diesem Modus können Sie die Hauptdatei der Arbeitsmappe aktualisieren, ohne die Benutzerdaten zu beeinträchtigen, da XLS Padlock nur die von Ihnen festgelegten Zellwerte speichert und wiederherstellt.

Wenn Sie Ihre Quellarbeitsmappe aktualisieren, besteht die direkteste Methode zur Verteilung der Aktualisierung darin, sie neu zu kompilieren und die neue EXE-Datei an Ihre Kunden zu senden.

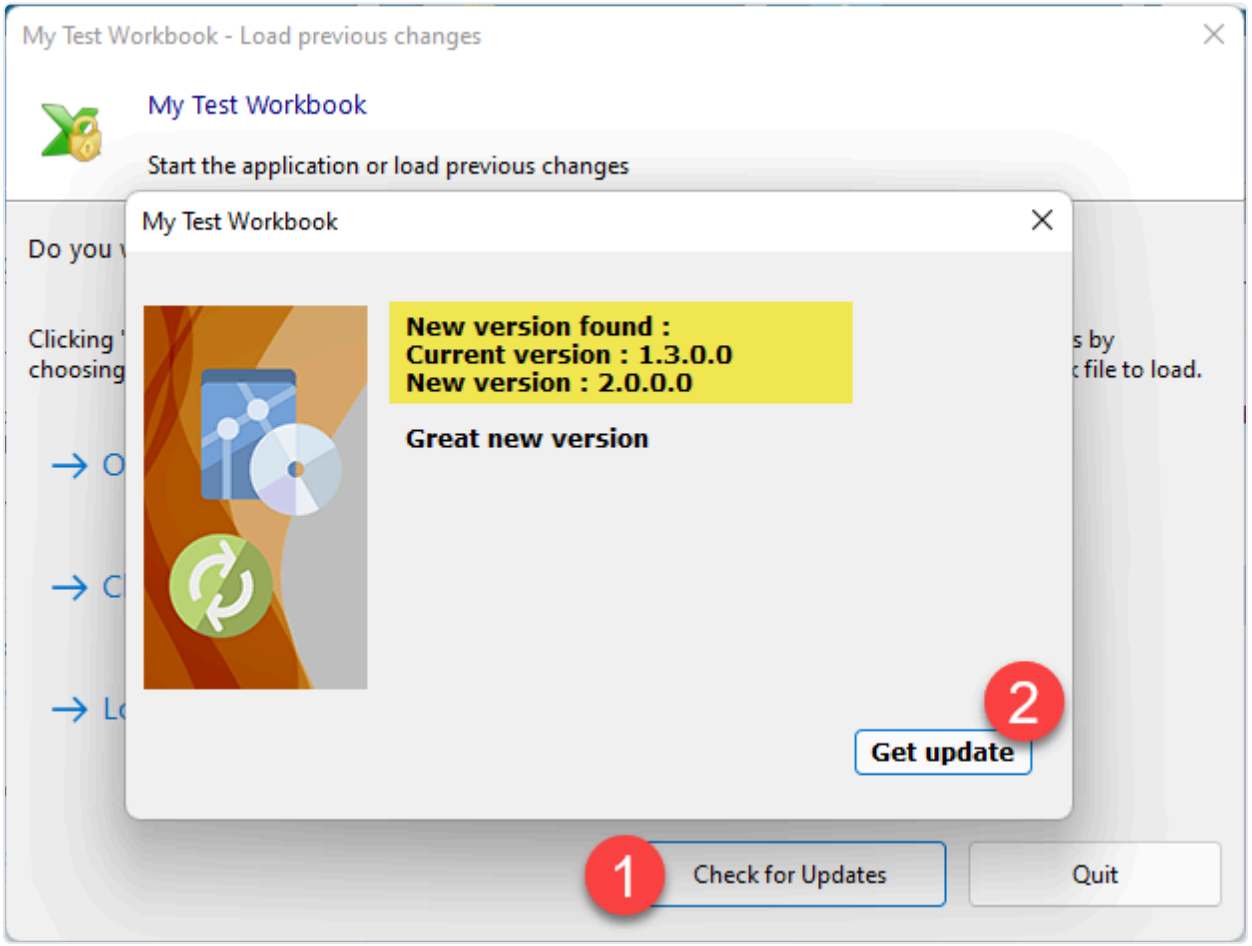
Empfohlener Aktualisierungsablauf

Das Verhalten auf der Benutzerseite hängt davon ab, was Sie zwischen den Versionen geändert haben:

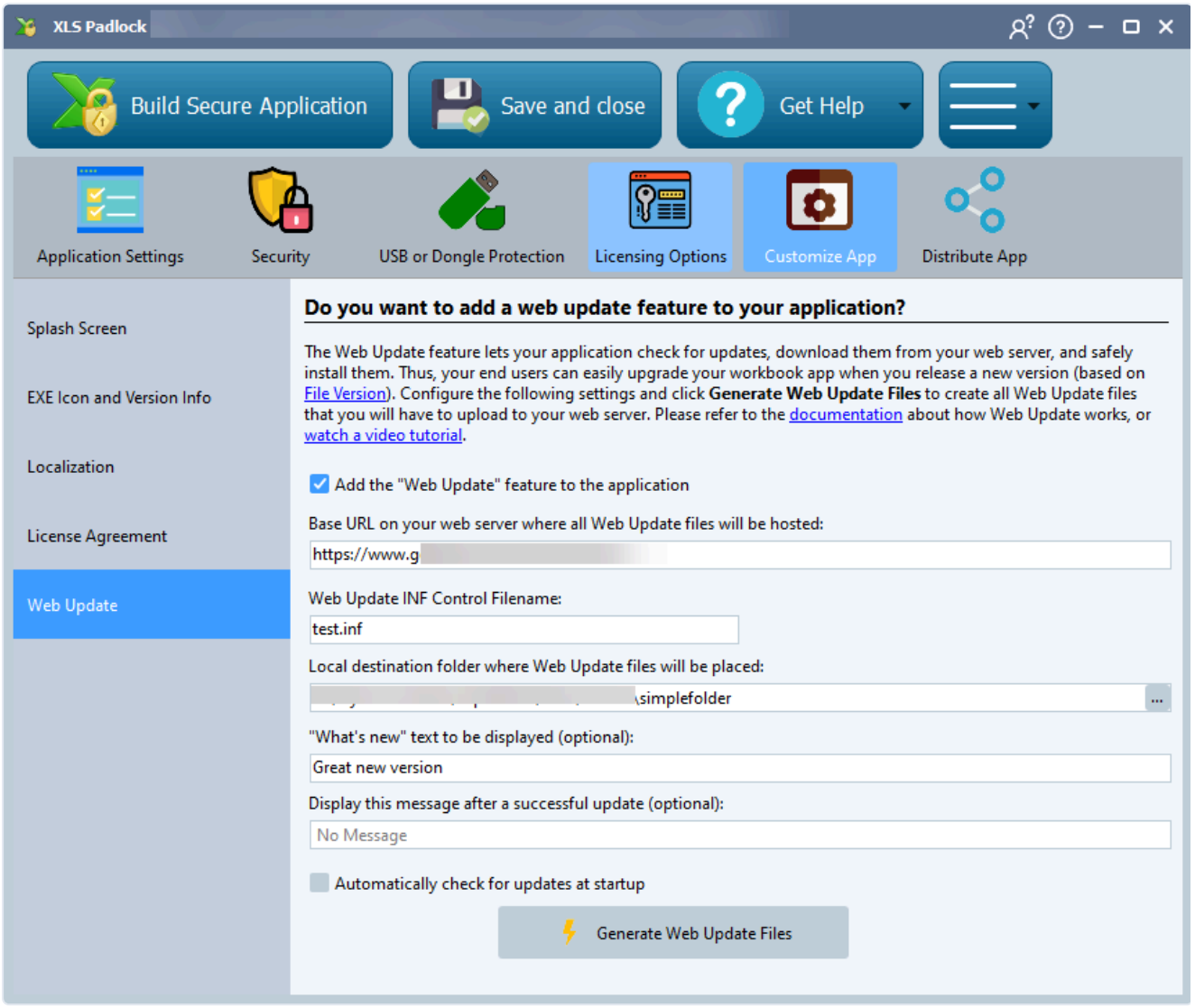
- **Nur Arbeitsmappeninhalt** (Formeln, Formatierung, Tabellenblätter, VBA-Logik): Behalten Sie die vorhandene **Application GUID** und den **Secret Key** bei. Im Modus `.XLSCE` sehen die Benutzer Ihre neue Logik mit ihren wiederhergestellten gespeicherten Zellwerten. Im Modus `.XLSC` sehen die Benutzer weiterhin ihre zuletzt gespeicherte Momentaufnahme, Ihre Änderungen werden nur angezeigt, wenn sie im Begrüßungsbildschirm manuell "Original Workbook" auswählen.
- **Sie möchten alte Benutzerspeicherdateien vollständig ungültig machen**: Erzeugen Sie in den Einstellungen unter Activation and Licensing einen neuen **Secret Key**, lassen Sie die **Application GUID** jedoch unverändert, damit vorhandene Aktivierungsschlüssel gültig bleiben. Beim ersten Start des Benutzers wird seine alte Speicherdatei als inkompatibel erkannt, und er wird über den Begrüßungsbildschirm aufgefordert, stattdessen die eingebettete Originalarbeitsmappe zu laden.
- **Wechseln Sie nicht zwischen den Versionen den Speichermodus** (`.XLSC` ↔ `.XLSCE`). Vorhandene Speicherdateien aus dem vorherigen Modus werden ignoriert, und alle Daten, die Ihre Benutzer noch nicht erneut eingegeben hatten, gehen verloren. Legen Sie zu Beginn des Projekts einen Speichermodus fest und behalten Sie ihn für die gesamte Lebensdauer der Anwendung bei.

> **TIPP**
>
> Die Schaltfläche Original Workbook auf dem Begrüßungsbildschirm dient dem Benutzer als Sicherheitsnetz, wenn eine alte Speicherdatei nicht geladen werden kann. Selbst wenn Sie sie ausgeblendet haben, aktiviert XLS Padlock sie für den erneuten Versuch automatisch wieder, sodass Benutzer niemals ausgesperrt werden.

👉 Das manuelle Herunterladen und Ersetzen der EXE-Datei kann für Endbenutzer jedoch mühsam sein. Um dies zu vereinfachen, bietet XLS Padlock eine automatische Web-Update-Funktion. Sie können Ihre Anwendung so konfigurieren, dass sie online nach Aktualisierungen sucht und die Benutzer benachrichtigt, wenn eine neue Version verfügbar ist:



Diese Funktion lässt sich schnell und direkt in XLS Padlock konfigurieren:



👉 Um Ihr eigenes Update-System einzurichten, benötigen Sie einen Webserver oder Speicherplatz auf einem Hosting, wo Sie Dateien zum direkten Download bereitstellen können. Folgen Sie anschließend den Anweisungen auf der Seite So richten Sie automatische Web-Updates ein.

> **Video-Tutorial**
>
> Sehen Sie sich unser Video-Tutorial dazu an, wie Sie automatische Web-Updates für Ihre Excel-Arbeitsmappen einrichten.

Automatische Web-Updates

XLS Padlock enthält eine Web-Update-Funktion, die neue Versionen Ihrer Anwendung automatisch herunterladen und installieren kann. So können Ihre Endanwender Ihre Arbeitsmappen-Anwendung mühelos aktualisieren, sobald Sie eine neue Version veröffentlichen.

Es ist keine Drittanbietersoftware erforderlich, aber **Sie benötigen einen Webserver oder Webspeicherplatz**, um die Update-Dateien zu hosten.

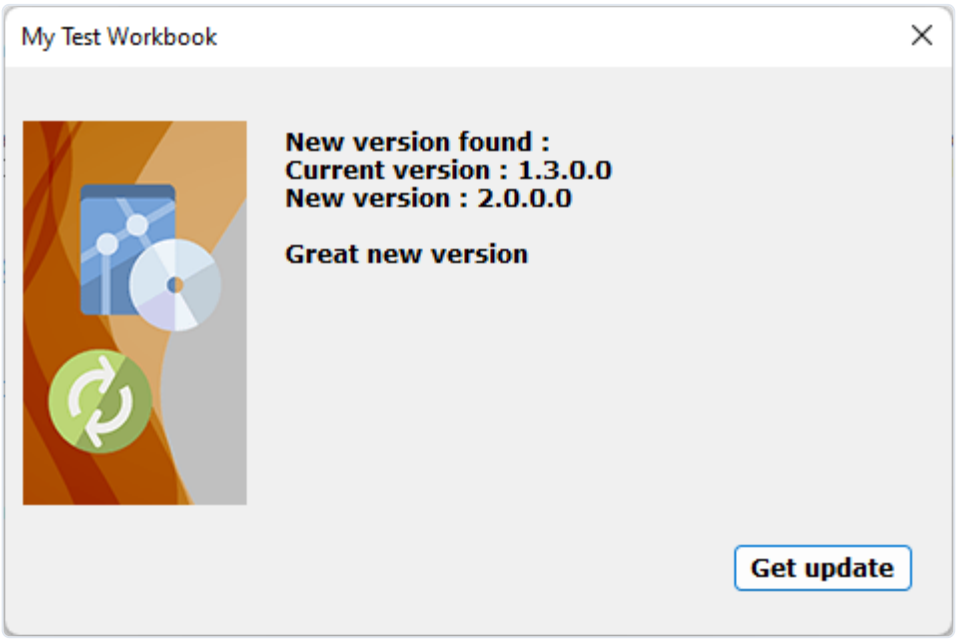
> **Video-Tutorial**
>
> Sehen Sie sich ein Video-Tutorial zur Einrichtung automatischer Web-Updates für Excel-Arbeitsmappen an.

So funktioniert das Web-Update

Die Anwendung lädt zunächst eine kleine Steuerdatei (`.inf`) von Ihrem Server herunter. Diese Datei teilt der Anwendung mit, ob eine neue Version verfügbar ist. Wenn das der Fall ist, lädt die Anwendung die erforderlichen Update-Dateien (`.cab`) herunter, schließt sich selbst, installiert das Update und startet neu.

Auf Updates prüfen

Sie können Ihre Anwendung so konfigurieren, dass sie **beim Start automatisch auf Updates prüft**, oder eine Schaltfläche "**Check for Updates**" (Auf Updates prüfen) zum Begrüßungsbildschirm hinzufügen. Wenn eine neue Version erkannt wird, werden die Anwender durch einen Web-Update-Assistenten geführt.



Anwender können ein Update auch über den Befehlszeilenschalter `-webupdate` starten.

Wie eine neue Version erkannt wird

Die Nummer der **File Version**, die auf der Seite EXE Version Info von XLS Padlock angegeben wird, bestimmt, ob ein Update erforderlich ist. Sie müssen diese Versionsnummer bei jeder Veröffentlichung eines Updates erhöhen.

Konfiguration

- **Base URL:** Die URL auf Ihrem Webserver, unter der die Web-Update-Dateien gehostet werden (z. B. `https://www.yourwebsite.com/myfolder`). HTTPS wird empfohlen.
- **Web Update INF Control Filename:** Der Name der Steuerdatei, die die Update-Informationen enthält.
- **Local Destination Folder:** Ein lokaler Ordner auf Ihrem Computer, in dem XLS Padlock die Update-Dateien erzeugt. Anschließend müssen Sie den gesamten Inhalt dieses Ordners auf Ihren Webserver hochladen.

  Warnung

  Der Zielordner sollte leer sein. XLS Padlock fragt nach, ob der Inhalt gelöscht werden soll, falls er nicht leer ist.
- **Text "What's new" (optional):** Text, der im Update-Assistenten angezeigt wird und die Anwender über die Änderungen in der neuen Version informiert.

Web-Update-Dateien erzeugen

Wenn Sie in XLS Padlock auf **Generate Web Update Files** (Web-Update-Dateien erzeugen) klicken, werden die erforderlichen Dateien (eine `.inf`-Steuerdatei und eine komprimierte `.cab`-Datei) in dem von Ihnen angegebenen lokalen Ordner erstellt. Sie können diese Dateien dann auf Ihren Webserver hochladen.

> **Administratorrechte**
>
> Wenn die Anwendung in einem geschützten Windows-Ordner wie `Program Files` installiert ist, sind zur Installation des Updates Administratorrechte erforderlich, was eine UAC-Eingabeaufforderung auslöst.

Fehlerbehebung

Um die Web-Update-Funktion zu untersuchen, aktivieren Sie die Option "**Enable WebUpdate Log**" (WebUpdate-Protokoll aktivieren) in den Erweiterten Optionen. Dadurch wird eine Datei `WUPDATE.LOG` im Dokumente-Verzeichnis des Anwenders erstellt, die detaillierte Informationen über den Update-Vorgang enthält.

Benutzerdaten über Updates hinweg migrieren

Wenn Sie Ihre geschützte Arbeitsmappe aktualisieren und eine neue EXE verteilen, kann es schwierig sein, die korrekte Übertragung der Daten Ihrer Benutzer sicherzustellen, insbesondere wenn Sie den Full Save-Modus verwenden. Dieser Modus erstellt eine **vollständige Momentaufnahme der Arbeitsmappe** zum Zeitpunkt des Speicherns. Wenn ein Benutzer also seine alte Speicherdatei (`.xlsc`) mit der neuen Anwendung öffnet, sieht er seine alten Daten in der alten Arbeitsmappenstruktur und profitiert nicht von Ihren neuesten Updates (etwa neuen Funktionen oder Fehlerbehebungen in Ihrem VBA-Code).

Dieses Thema erläutert, wie Sie ein VBA-basiertes Export-/Importsystem verwenden, um Ihren Benutzern zu helfen, ihre Daten von einer alten Version Ihrer Anwendung in eine neue zu migrieren.

> **Warnung**
>
> Die hier beschriebene Lösung hängt vom Speichermodus ab, den Sie für Ihr XLS Padlock-Projekt ausgewählt haben. Wenn Sie den Speichermodus **Cell Values** verwenden, werden die Benutzerdaten automatisch in die Struktur Ihrer aktualisierten Arbeitsmappe geladen, und dieser manuelle Vorgang ist nicht erforderlich.

> **Alternative: Speichermodus Cell Values**
>
> Wie oben erwähnt, ist der Speichermodus Cell Values die einfachste Methode, um Updates zu handhaben. Dieser Modus ist jedoch dafür konzipiert, nur die Werte bestimmter Zellen zu speichern. Wenn Sie andere Arten von Daten übertragen müssen, etwa den Zustand von VBA-Variablen oder -Objekten, müssen Sie eine benutzerdefinierte VBA-Lösung wie die in diesem Thema beschriebene verwenden.

Codebeispiel

Dieser Vorgang umfasst zwei Hauptschritte: das Exportieren der Daten aus der alten Version und das Importieren in die neue.

1. Benutzerdaten in eine Excel-Datei exportieren

Das nachstehende Makro erzeugt eine normale `.xlsx`-Excel-Datei mit den Daten des Benutzers. Sie sollten dieses Makro in Ihre alte Anwendung einbinden und mit einer Schaltfläche verknüpfen, damit Benutzer ihre Daten vor dem Upgrade exportieren können.

> **HINWEIS**
>
> Sie müssen die Tabellenblattnamen und Zellbereiche im nachstehenden Makro ersetzen, damit sie zur Struktur Ihrer Arbeitsmappe passen.

```
Sub GenerateData()
    Dim savePath As String
    'New workbook with 3 sheets
    Workbooks.Add xlWBATWorksheet
    ActiveSheet.Name = "SheetA"
    Sheets.Add(After:=Sheets(1)).Name = "SheetB"
    Sheets.Add(After:=Sheets(2)).Name = "SheetC"
    ActiveWorkbook.Sheets("SheetA").Range("A1:C3").Value = ThisWorkbook.Sheets("SheetA").Range(
    ActiveWorkbook.Sheets("SheetB").Range("B3").Value = ThisWorkbook.Sheets("SheetB").Range("B3
    ActiveWorkbook.Sheets("SheetC").Range("B1:C3").Value = ThisWorkbook.Sheets("SheetC").Range(
    savePath = Application.GetSaveAsFilename("", "Excel workbook (*.xlsx),*.xlsx", 1, "Export U
    If savePath <> "False" Then ActiveWorkbook.SaveAs savePath, FileFormat:=51
    ActiveWorkbook.Close False
End Sub
```

2. Die Daten in die neue EXE-Datei hochladen.

Der Benutzer muss die neue EXE öffnen und das nachstehende dritte Makro ausführen, um die Daten hochzuladen (verknüpfen Sie das Makro mit einer Schaltfläche). Sobald der Benutzer das Makro ausführt, wird er aufgefordert, die Datei auszuwählen (erstes Makro), und die Daten werden übertragen (zweites Makro). Das dritte Makro führt beide Makros aus, und genau dieses Makro muss mit der Schaltfläche verknüpft werden. Auch hier müssen Sie die Zellen und Tabellenblattnamen im zweiten Makro ändern, und Sie können außerdem den Titel im ersten Makro ändern:

```
************* 1st macro:
Sub Open_Workbook_Dialog()
Dim my_FileName As Variant
my_FileName = Application.GetOpenFilename( _
FileFilter:="Excel Files,.xl;.xm", _
FilterIndex:=3, _
Title:="Select the old version of your file, where you will pull the data from", _
MultiSelect:=False)
If my_FileName <> False Then
Workbooks.Open Filename:=my_FileName
End If
End Sub

*************** 2nd macro:
Sub TransferData()
If Workbooks.Count > 1 Then
Workbooks(1).Sheets("SheetA").Range("A1:C3").Value = Workbooks(2).Sheets("SheetA").Range("A1:C3
Workbooks(1).Sheets("SheetB").Range("B3").Value = Workbooks(2).Sheets("SheetB").Range("B3").Val
Workbooks(1).Sheets("SheetC").Range("B1:C3").Value = Workbooks(2).Sheets("SheetC").Range("B1:C3
Workbooks(2).Close savechanges:=False
Else
MsgBox "The data hasn't been transferred.", vbExclamation, "Error"
End If
End Sub

************ 3rd macro:
Sub TheTransfer()
Call Open_Workbook_Dialog
Call TransferData
End Sub
```

HINWEIS

Um normale Arbeitsmappen zu speichern, finden Sie weitere Informationen unter: Laden/Speichern von Arbeitsmappen über die VBA-Hilfsfunktion SetOption

👉 Siehe auch:

- Speichermodus, den Sie für Ihr XLS Padlock-Projekt ausgewählt haben

Einstellungen über Vorlagen speichern und wiederherstellen

Mit XLS Padlock können Sie die aktuellen Einstellungen Ihres Projekts in einer Vorlagendatei speichern. Dies ist eine leistungsstarke Funktion, die Ihnen erheblich Zeit sparen kann, insbesondere wenn Sie häufig neue Arbeitsmappenanwendungen mit ähnlichen Konfigurationen erstellen.

Projekteinstellungen in einer Vorlage speichern

Sobald Sie alle gewünschten Optionen für Ihr Projekt konfiguriert haben (z. B. Sicherheitseinstellungen, Anpassungen und Ausgabeoptionen), können Sie diese Konfiguration als Vorlage speichern.

Klicken Sie auf die Schaltfläche

App Menu und dann auf das Menü **Save Current Settings As Template**. Sie werden aufgefordert, einen Speicherort und einen Namen für Ihre Vorlagendatei zu wählen (die die Erweiterung `.XPLP` erhält).

> **Warnung**
>
> Wählen Sie nicht Ihre vorhandene XLS Padlock-Projektdatei aus.

Einstellungen aus einer Vorlage wiederherstellen

Wenn Sie ein neues Projekt starten oder eine Standardkonfiguration auf ein bestehendes Projekt anwenden möchten, können Sie Einstellungen ganz einfach aus einer zuvor gespeicherten Vorlage wiederherstellen.

Klicken Sie auf die Schaltfläche

App Menu, dann auf das Menü **Restore Project from Template**, navigieren Sie zu Ihrer `.XPLP`-Vorlagendatei und wählen Sie sie aus. Alle in dieser Vorlage gespeicherten Einstellungen werden sofort auf Ihr aktuelles Projekt angewendet.

HINWEIS

Der Ausgabepfad der Anwendung wird nicht in Vorlagen gespeichert.

Registrierungsfehler oder EREGISTRYEXCEPTION

Dieser Fehler weist häufig auf einen Konflikt mit einer Sicherheitssoftware oder einem Antivirenprogramm hin. Um ihn zu beheben, aktivieren Sie die Option „**Use another registry key for storing activation data**" (Einen anderen Registrierungsschlüssel zum Speichern der Aktivierungsdaten verwenden) in den Advanced Options.

Zugriffsverletzung an der Adresse

Dieser Fehler kann bei der Verwendung der Funktion „Lock VBA Project (simple VBA protection)" auftreten. Um ihn zu beheben, entfernen Sie das Kennwort aus Ihrem VBA-Projekt, bevor Sie die Arbeitsmappe mit XLS Padlock kompilieren. Das Kennwort wird nicht benötigt, da XLS Padlock das Projekt automatisch sperrt, ohne nach einem Kennwort zu fragen.

Alternativ können Sie ein kürzeres Kennwort verwenden oder zur Option Prevent access to VBA editor wechseln.

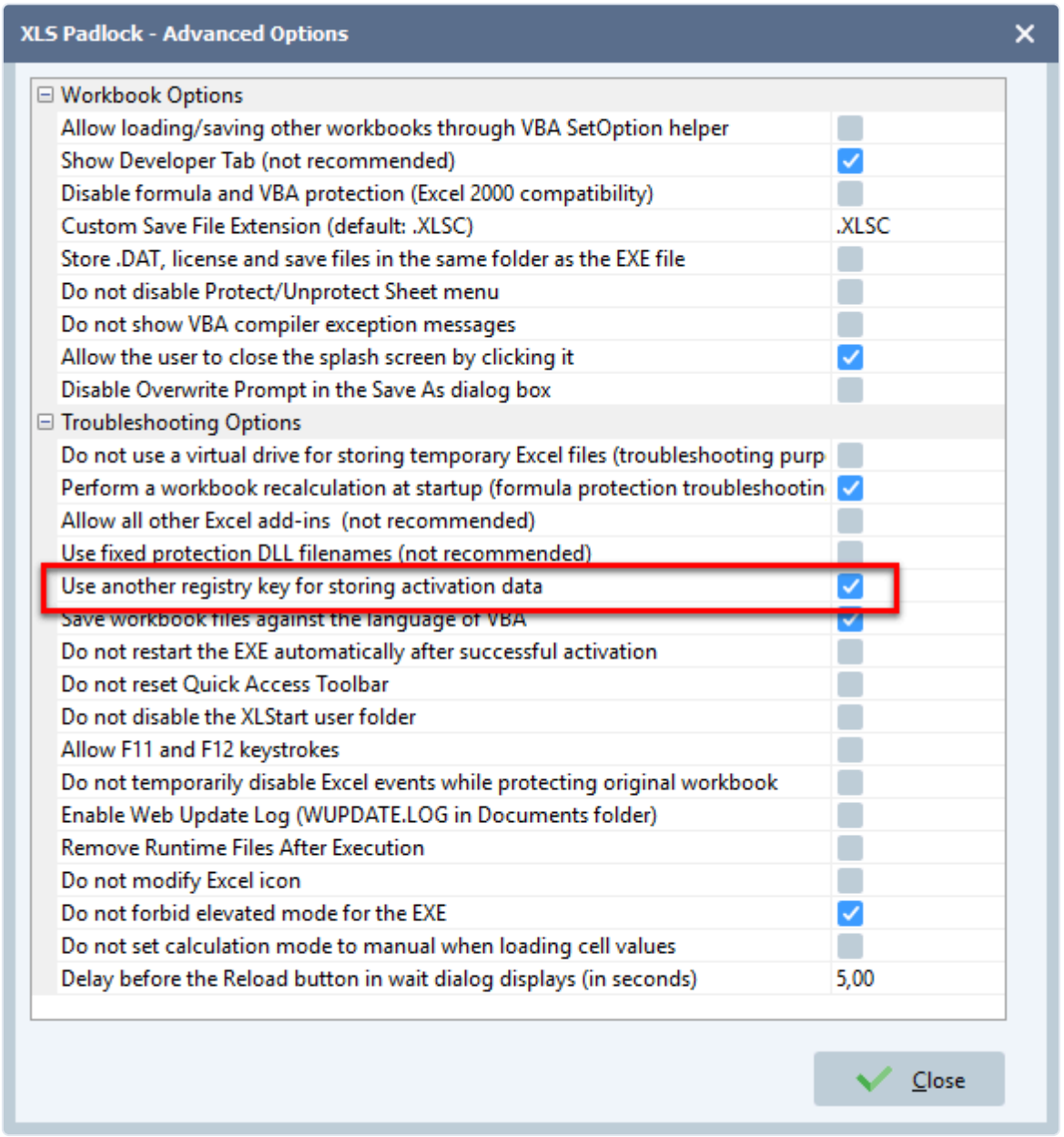
Fehler "Failed to set data for 'Data'"

Wenn Ihr Endbenutzer beim Eingeben eines Aktivierungsschlüssels die folgende Fehlermeldung erhält, weist dies in der Regel auf einen Konflikt mit einer Sicherheitssoftware auf seinem System hin.

Error setting data in registry, Failed to set data for 'Data'

Dieser Fehler tritt auf, weil ein anderes Programm, in der Regel ein Antivirenprogramm oder eine Sicherheitssuite, die Anwendung daran hindert, ihre Aktivierungsdaten in die Windows-Registry zu schreiben.

Um dieses Problem zu beheben, können Sie Ihr XLS Padlock-Projekt so konfigurieren, dass es einen anderen Registrierungsschlüssel verwendet. Aktivieren Sie die erweiterte Option **Use another registry key for storing activation data** (einen anderen Registrierungsschlüssel zum Speichern der Aktivierungsdaten verwenden), wie unten dargestellt, und kompilieren Sie Ihre Anwendung neu.



Warum ist die EXE so groß

Die kompilierte `.exe`-Datei ist deutlich größer als die ursprüngliche Arbeitsmappe, da sie eine eigenständige Anwendung ist, die mehrere wesentliche Komponenten bündelt:

- Einen sicheren Loader, der mit verschiedenen Versionen von Excel und Windows kompatibel ist.
- Eine vollständige Unicode-fähige Laufzeitumgebung.
- Den proprietären VBA Compiler und Interpreter.
- Mehrere Schichten von Anti-Piraterie- und Codeschutz.

So reduzieren Sie die Größe der EXE-Datei:

- **Use UPX Compression** (UPX-Komprimierung verwenden): UPX ist ein kostenloser, quelloffener EXE-Kompressor, der unter upx.github.io verfügbar ist. Das Komprimieren der EXE kann ihre Größe erheblich verringern. Wir empfehlen jedoch dringend, jede komprimierte EXE-Datei mit einer Codesignatur zu versehen, da einige Antivirenprogramme andernfalls Fehlalarme auslösen können.
- **Disable Formula Protection** (Formelschutz deaktivieren): Wenn Ihre Anwendung nicht auf sensiblen Formeln beruht, können Sie den Formelschutz deaktivieren, um die endgültige Größe der EXE-Datei geringfügig zu reduzieren.

XLS-Datei aus der EXE wiederherstellen

Nein, es ist nicht möglich, die ursprüngliche Excel-Datei aus einer kompilierten .EXE wiederherzustellen, wenn die Funktion "Save As" deaktiviert wurde.

Wenn Sie die Möglichkeit benötigen, die ursprüngliche Arbeitsmappe wiederherzustellen, müssen Sie die Funktion "Save As" während der Kompilierung aktivieren. Dadurch können Sie die Funktion zum Entschlüsseln gespeicherter Dateien verwenden.

Support-Links

Wenn Sie Fragen oder Probleme mit XLS Padlock haben, erreichen Sie uns über die folgenden Kanäle:

- **Benutzerforum**: https://www.gdgsoft.info (Wählen Sie die Kategorie "XLS Padlock")
- **E-Mail-Support**: info@xlspadlock.com
- **Für den Support des Enky SL/LC Dongles**: info@hs-securityware.com

Wenn Sie Support anfordern, geben Sie bitte Ihre Excel-Version (z. B. Excel 365, Excel 2019) und deren Architektur (32-Bit oder 64-Bit) an, damit wir Ihr Problem schneller lösen können.

Folgen Sie uns auf X: https://x.com/gdgsoft

Unsere weiteren Produkte finden Sie unter: https://www.gdgsoft.com