



LABELSTAR OFFICE

Referenzhandbuch

Version 6.50 Build 1010
März 2019

Inhaltsverzeichnis

Erste Schritte	4
Verwenden des Application-Objekts	5
Verwenden des Label-Objekts	6
Verwenden des Printer-Objekts	8
Erstellen Sie ihre erste Anwendung	10
Schritt 1: Erstellen eines Windows Forms-Anwendungsprojekts	11
Schritt 2: Plattformziel ändern	12
Schritt 3: API in das Projekt einbinden	13
Schritt 4: Application-Objekt initialisieren	14
Schritt 5: Etikett öffnen	16
Schritt 6: Etikett drucken	19
Schritt 7: Programm ausführen	23
Programmierschnittstelle	24
Application-Objekt	25
ActivePrinter-Eigenschaft	26
GetPrinter-Methode	27
HasError-Eigenschaft	28
Import-Methode	29
Info-Eigenschaft	30
Initialize-Methode	31
IsInitialized-Eigenschaft	32
LabelDir-Eigenschaft	33
LastError-Eigenschaft	34
License-Eigenschaft	36
OpenLabel-Methode	37
OpenPrintStack-Methode	38
Error-Objekt	39
Details-Eigenschaft	40
ErrorCode-Eigenschaft	41
ErrorType Property	42
Message Property	43
Field-Objekt	44
FieldName-Eigenschaft	45
GetContent-Methode	46
GetPropertyValue Method	47
Eigenschaftennamen	48
Locked-Eigenschaft	49
Printable-Eigenschaft	50
SetContent-Methode	51
SetPropertyValue-Methode	52
Label-Objekt	53
ActivePrinter-Eigenschaft	54
Copies-Eigenschaft	55
CurrentRecord-Eigenschaft	56
FieldCount-Eigenschaft	57
FieldNames-Eigenschaft	58
GetFieldByIndex-Methode	59
GetFieldByName-Methode	60
GetPreview-Methode	61

GetPropertyValue Method	62
Eigenschaftennamen	63
IsDataAvailable-Eigenschaft	64
LabelPath-Eigenschaft	65
MaxRecord-Eigenschaft.....	66
Modified-Eigenschaft.....	67
PageName-Eigenschaft	68
Print-Methode.....	69
Print-Methode (int).....	70
Print-Methode (int, PrintOptions).....	71
PrintOptions-Eigenschaft	72
PrintToFile-Methode (string)	73
PrintToFile-Methode (string, int)	74
PrintToFile-Methode (string, int, PrintOptions)	75
Save-Methode.....	76
SaveAs-Methode.....	77
SavePreview-Methode (string)	78
SavePreview-Methode (string, ImageFormat).....	79
SelectRecord-Methode (string).....	80
Filtersyntax.....	83
Filterfunktionen.....	87
SelectRecord-Methode (string, string).....	89
SetPropertyValue Method	90
LicenseInfo-Objekt	91
IsTrialVersion-Eigenschaft.....	92
LicenseKey-Eigenschaft.....	93
Printer-Objekt	94
Dispose-Methode	95
Encoding-Eigenschaft	96
IsCVPrinter-Eigenschaft.....	97
Execute-Methode (PrintCommand)	98
Execute-Methode (string)	99
GetStatus-Methode	100
GetStatus-Methode (out string)	101
PrinterName-Eigenschaft.....	102
SendBytes-Methode.....	103
PrintStack-Objekt	104
FilePath-Eigenschaft	105
Print-Methode.....	106
VersionInfo-Objekt.....	107
CompanyName-Eigenschaft	108
CompiledVersion-Eigenschaft.....	109
Copyright-Eigenschaft.....	110
DisplayVersion-Eigenschaft	111
ProductName-Eigenschaft	112
Fehlercodes	113
Kontakte	114
Systemanforderungen	115
Impressum	116

Erste Schritte

Labelstar Office ist mit einer Automation-Schnittstelle, früher auch als OLE-Automation bezeichnet, ausgestattet. Diese Programmierschnittstelle (API = Application Programming Interface) ermöglicht es externen Anwendungen die Funktionalität von **Labelstar Office** in ihre eigenen Programme einzubinden, um z.B. **Labelstar Office** Etiketten zu öffnen und zu drucken.

Hinweis

Die Codebeispiele sind in C# und VB.Net geschrieben. Allerdings kann auch jede andere Programmiersprache verwendet werden, die Automation unterstützt, z.B. Visual Basic oder C++.

Um die Automation einsetzen zu können muss eine Kopie von **Labelstar Office** auf ihrem System installiert sein.

Verwenden des Application-Objekts

Ein [Application](#)-Objekt repräsentiert die gesamte **Labelstar Office** Anwendung und ist das oberste Objekt der Objekthierarchie, von dem aus alle anderen Objekte abgeleitet werden. Sie können die [Application](#)-Eigenschaften und -Methoden verwenden, um Etiketten zu öffnen, den Etiketteninhalt zu ändern und anschließend zu drucken und zu speichern.

Beispiel (C#)

```
LSOffice.Application objApp;

// Initialize application
objApp = new LSOffice.Application ();
objApp.Initialize ();

if (objApp.HasError)
{
    MessageBox.Show (objApp.LastError.Message, "Error");
}
```

Beispiel (VB.Net)

```
Dim objApp As LSOffice.Application

'Initialize application
objApp = New LSOffice.Application
objApp.Initialize()

If (objApp.HasError) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
End If
```

Verwenden des Label-Objekts

Mit Hilfe der Methode [Application.OpenLabel](#) können Sie ein Etikett öffnen. Das zurückgegebene [Label](#)-Objekt repräsentiert das **Labelstar Office**-Etikett (*.lbex) das gerade geöffnet worden ist und bietet Zugriff auf eine breite Palette von Eigenschaften und Methoden.

Das [Label](#)-Objekt ermöglicht es Ihnen das Etikett zu speichern und zu drucken, sowie Text-, Barcode- und Bilddaten zu ändern. Es erlaubt Ihnen nicht das Etikettenlayout zu ändern oder ein neues Etikett zu erstellen.

Beispiel (C#)

```
LSOffice.Application objApp;

// Initialize application
objApp = new LSOffice.Application ();
objApp.Initialize ();

if (objApp.HasError)
{
    MessageBox.Show (objApp.LastError.Message, "Error");
    return;
}

// Open label
LSOffice.Label objLabel = objApp.OpenLabel ("C:\\Label1.lbex");

if (objLabel == null)
{
    MessageBox.Show (objApp.LastError.Message, "Error");
    return;
}

// Change field content
LSOffice.Field objField = objLabel.GetFieldByName ("Text1");

if (objField == null)
{
    MessageBox.Show (objApp.LastError.Message, "Error");
    return;
}

objField.SetContent ("12345");

// Print label
objLabel.Copies = 5;
objLabel.Print ();
```

Beispiel (VB.Net)

```
Dim objApp As LSOffice.Application
Dim objLabel As LSOffice.Label
Dim objField As LSOffice.Field

'Initialize application
objApp = New LSOffice.Application
objApp.Initialize()

If (objApp.HasError) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
    Return
End If

'Open label
objLabel = objApp.OpenLabel("C:\\Label1.lbex")

If (objLabel Is Nothing) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
    Return
End If

'Change field content
objField = objLabel.GetFieldByName("Text1")

If (objField Is Nothing) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
    Return
End If

objField.SetContent("12345")

'Print label
objLabel.Copies = 5
objLabel.Print()
```

Verwenden des Printer-Objekts

Mit Hilfe der Methode [Application.GetPrinter](#) können Sie ein [Printer](#)-Objekt erstellen, das eine Drucker repräsentiert, der unter Windows installiert ist.

Das [Printer](#)-Objekt erlaubt es Ihnen dem Druckerstatus abzufragen und Daten direkt an den Drucker zu übertragen. Ist der Drucker ein Carl Valentin-Etikettendrucker können spezielle Druckbefehle an den Drucker übertragen werden um z.B. den aktuellen Druckauftrag anzuhalten, fortzusetzen oder abzubrechen.

Beispiel (C#)

```
LSOffice.Application objApp;

// Initialize application
objApp = new LSOffice.Application ();
objApp.Initialize ();

if (objApp.HasError)
{
    MessageBox.Show (objApp.LastError.Message, "Error");
    return;
}

// Get printer
using (LSOffice.Printer objPrinter = objApp.GetPrinter ("LabelPrt1"))
{
    if (objPrinter == null)
    {
        MessageBox.Show (objApp.LastError.Message, "Error");
        return;
    }

    // Check printer status
    string message;
    LSOffice.PrinterStatus status = objPrinter.GetStatus (out message);

    if (status != LSOffice.PrinterStatus.Idle)
    {
        MessageBox.Show (message, "Printer Status");
        return;
    }

    // Does the selected printer supports print commands?
    if (!objPrinter.IsCVPrinter)
    {
        MessageBox.Show ("Invalid label printer.", "Error");
        return;
    }

    // Label feed
    objPrinter.Execute (LSOffice.PrintCommand.LabelFeed);

    // Test print
    objPrinter.Execute ("(SOH)FF----r(ETB)");
}
```

Beispiel (VB.Net)

```
Dim objApp As LSOoffice.Application
Dim objPrinter As LSOoffice.Printer

'Initialize application
objApp = New LSOoffice.Application
objApp.Initialize()

If (objApp.HasError) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
    Return
End If

'Get printer
objPrinter = objApp.GetPrinter("LabelPrt1")

If (objPrinter Is Nothing) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
    Return
End If

'Check printer status
Dim message As String = ""
Dim status = objPrinter.GetStatus(message)

If (status <> LSOoffice.PrinterStatus.Idle) Then
    MsgBox(message, vbOKOnly, "Printer Status")
    objPrinter.Dispose()
    Return
End If

'Does the selected printer supports print commands?
If (Not objPrinter.IsCVPrinter) Then
    MsgBox("Invalid label printer.", vbOKOnly, "Error")
    objPrinter.Dispose()
    Return
End If

'Label feed
objPrinter.Execute(LSOoffice.PrintCommand.LabelFeed)

'Test print
objPrinter.Execute("(SOH)FF----r(ETB)")
objPrinter.Dispose()
```

Erstellen Sie ihre erste Anwendung

In diesem Abschnitt erstellen Sie eine einfache Anwendung um Etiketten zu öffnen und zu drucken.

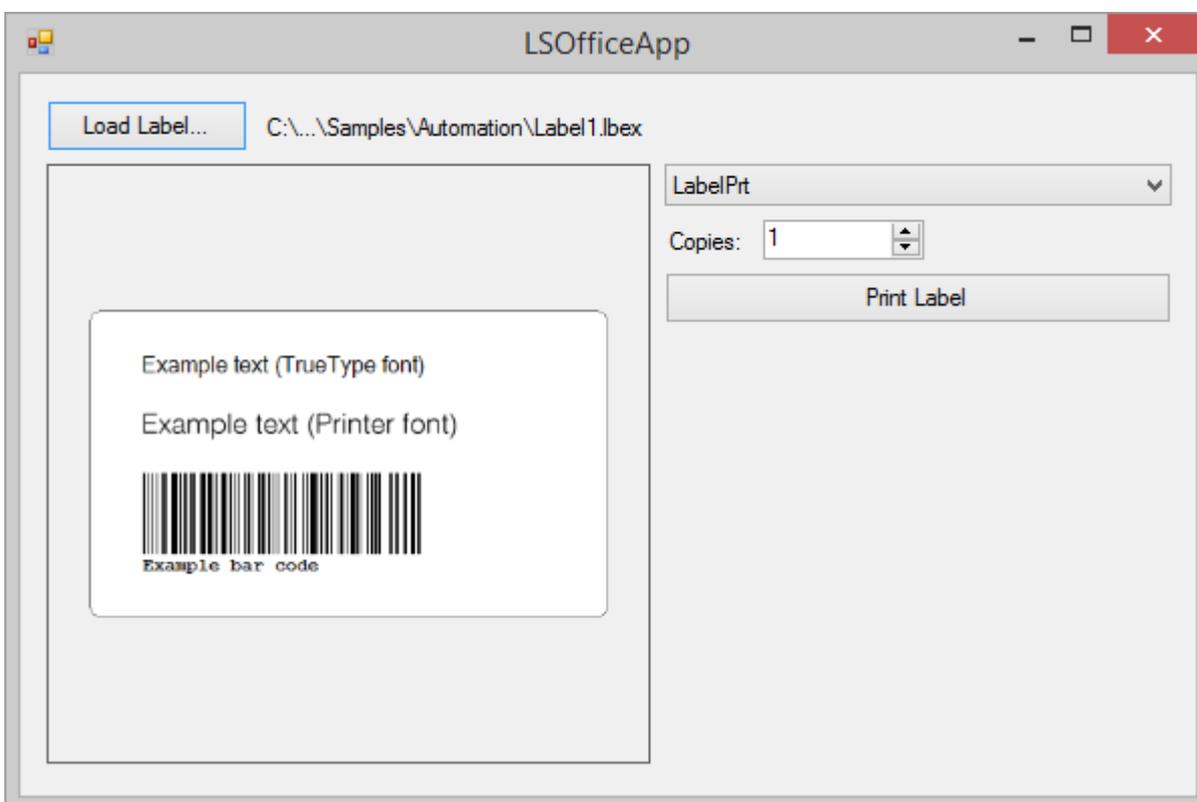
Hinweis

Die Anwendung wird mit Microsoft Visual Studio 2015 erstellt und eine Kopie von **Labelstar Office** muss auf Ihrem System installiert sein.

Sie lernen folgendes:

- Erstellen eines neuen Projekts
- Einfügen der LSOoffice API in ihr Projekt
- [Application](#)-Objekt initialisieren
- Etikett öffnen und drucken

Am Ende sieht das Programm so aus wie die folgende Abbildung.



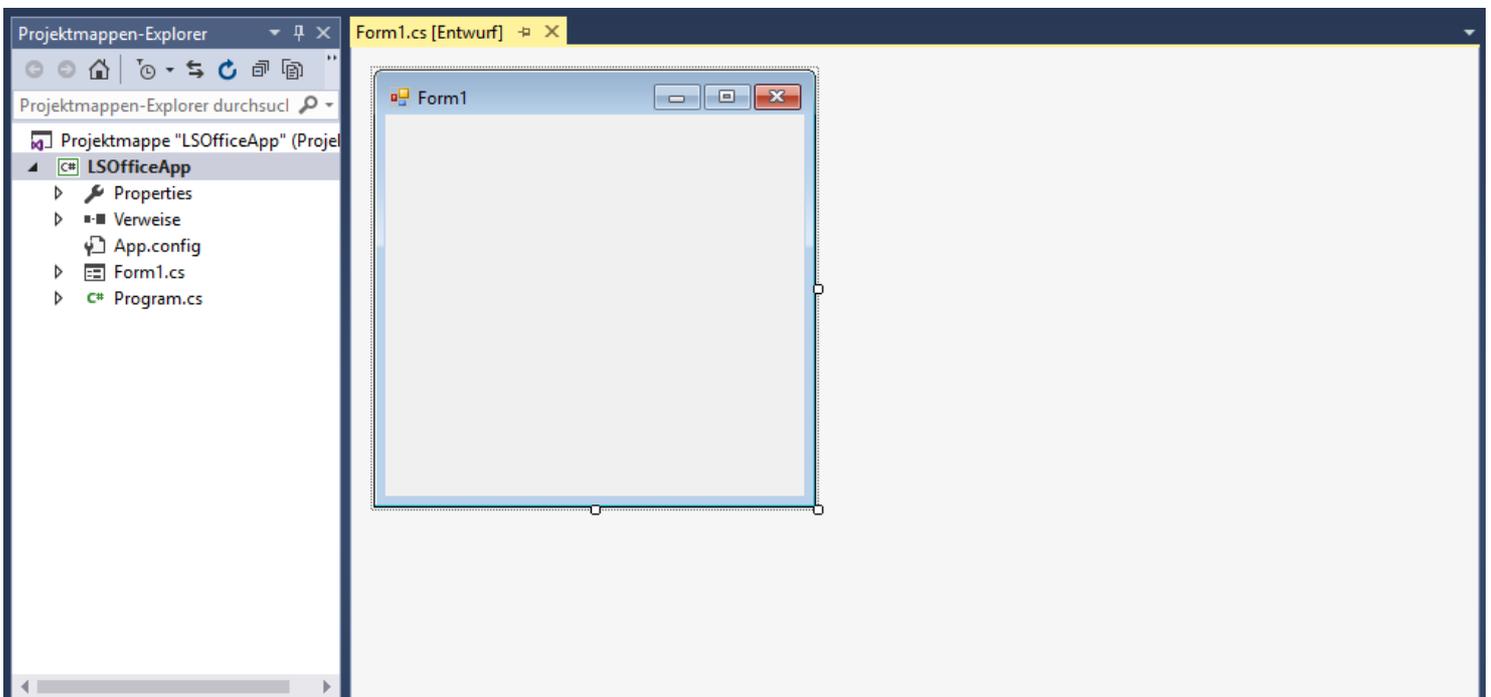
Schritt 1: Erstellen eines Windows Forms-Anwendungsprojekts

Um eine neue Anwendung zu erstellen müssen Sie zuerst ein Windows Forms-Anwendungsprojekt erstellen.

So erstellen Sie ein Windows Forms-Anwendungsprojekt

1. Öffnen Sie **Visual Studio**.
2. Wählen Sie in der Menüleiste **Datei, Neu, Projekt** aus.
Das Dialogfeld **Neues Projekt** wird geöffnet.
3. Wählen Sie als Zielframework **.Net Framework 4.6** aus.
4. Wählen Sie in der Liste **Installierte Vorlagen** entweder **Visual C#** oder **Visual Basic** aus.
5. Wählen Sie in der Liste der Vorlagen das Symbol **Windows Forms-Anwendung** aus.
6. Tragen Sie im Eingabefeld **Name** den Projektnamen (im Beispiel LSOfficeApp) ein.
7. Klicken Sie auf die Schaltfläche **OK**.

Visual Studio erstellt eine Projektmappe für das Programm. Die Projektmappe fungiert als Container für alle Projekte und Dateien, die vom Programm benötigt werden.



Schritt 2: Plattformziel ändern

Die Automationschnittstelle von **Labelstar Office** ist eine echte 32-Bit-Schnittstelle. Bei 64-Bit-Systemen muss das Programm immer als 32-Bit-Anwendung generiert werden.

So aktivieren Sie x86 als CPU-Plattformziel (C#)

1. Klicken Sie im **Projektmappen-Explorer** mit der rechten Maustaste auf den Projektknoten **LOfficeApp** und dann auf **Eigenschaften**.
2. Wählen Sie die Registerkarte **Erstellen** aus.
3. Legen Sie die Eigenschaft **Plattformziel** auf **x86** fest.

So aktivieren Sie x86 als CPU-Plattformziel (Visual Basic)

1. Klicken Sie im Projektmappen-Explorer mit der rechten Maustaste auf dem Projektknoten **LOfficeApp** und dann auf **Eigenschaften**.
2. Wählen Sie die Registerkarte **Kompilieren** aus.
3. Legen Sie die Eigenschaft **Ziel-CPU** auf **x86** fest.

Hinweis

Wenn Sie Ihre Anwendung mit dem falschen Plattformziel erstellen, wird folgende Fehlermeldung angezeigt:

Konflikt zwischen Prozessorarchitektur des Projekts "MSIL", das erstellt wird, und der Prozessorarchitektur des Verweises, "LOffice", "x86". Dieser Konflikt kann zu Laufzeitfehlern führen. Ändern Sie ggf. mithilfe des Konfigurations-Managers die als Ziel angegebene Prozessorarchitektur Ihres Projekts so, dass die Prozessorarchitekturen zwischen Ihrem Projekt und den Verweisen ausgerichtet werden, oder wählen Sie eine Abhängigkeit von Verweisen mit einer Prozessorarchitektur, die der als Ziel angegebene Prozessorarchitektur Ihres Projekts entspricht.

Schritt 3: API in das Projekt einbinden

Im nächsten Schritt wird ein Verweis auf die LSOoffice Typbibliothek zu ihrem Entwicklungsprojekt hinzugefügt. Danach können Sie die LSOoffice-Objekte in ihrer Entwicklungsumgebung verwenden.

So fügen Sie einen Verweis auf LSOoffice.dll hinzu

1. Klicken Sie im **Projektmappen-Explorer** mit der rechten Maustaste auf den Knoten **Verweise** und dann auf **Verweis hinzufügen**.
2. Klicken Sie im **Verweis-Manager** auf die Schaltfläche **Durchsuchen**.
3. Wählen Sie im Installationsverzeichnis von **Labelstar Office** die Datei **LSOoffice.dll** aus.
4. Klicken Sie auf die Schaltfläche **OK**.

Schritt 4: Application-Objekt initialisieren

Nun können Sie ein [Application](#)-Objekt erstellen und initialisieren.

Fügen Sie der *Form1*-Coddatei (Form1.cs or Form1.vb) den folgenden Code hinzu:

C#

```
LSOffice.Application objApp;

protected override void OnLoad (EventArgs e)
{
    // Initialize application
    objApp = new LSOffice.Application ();

    objApp.Initialize ();
    DisplayLastError ();

    if (objApp.License.IsTrialVersion)
        Text += " - Trial Version";

    base.OnLoad (e);
}

private bool DisplayLastError ()
{
    if (objApp.LastError.ErrorType == LSOffice.ErrorType.Success)
        return false;

    string title = "Error";

    if (objApp.LastError.ErrorType == LSOffice.ErrorType.Warning)
        title = "Warning";

    MessageBox.Show (objApp.LastError.Message, title);
    return true;
}
```

VB.Net

```
Dim objApp As LSOffice.Application

Protected Overrides Sub OnLoad(ByVal e As EventArgs)

    'Initialize application
    objApp = New LSOffice.Application

    objApp.Initialize()
    DisplayLastError()

    If (objApp.License.IsTrialVersion) Then
        Text += " - Trial Version"
    End If

    MyBase.OnLoad(e)

End Sub

Private Function DisplayLastError() As Boolean

    If (objApp.LastError.ErrorType = LSOffice.ErrorType.Success) Then
        Return False
    End If

    Dim title As String = "Error"

    If (objApp.LastError.ErrorType = LSOffice.ErrorType.Warning) Then
        title = "Warning"
    End If

    MsgBox(objApp.LastError.Message, vbOKOnly, title)
    Return True

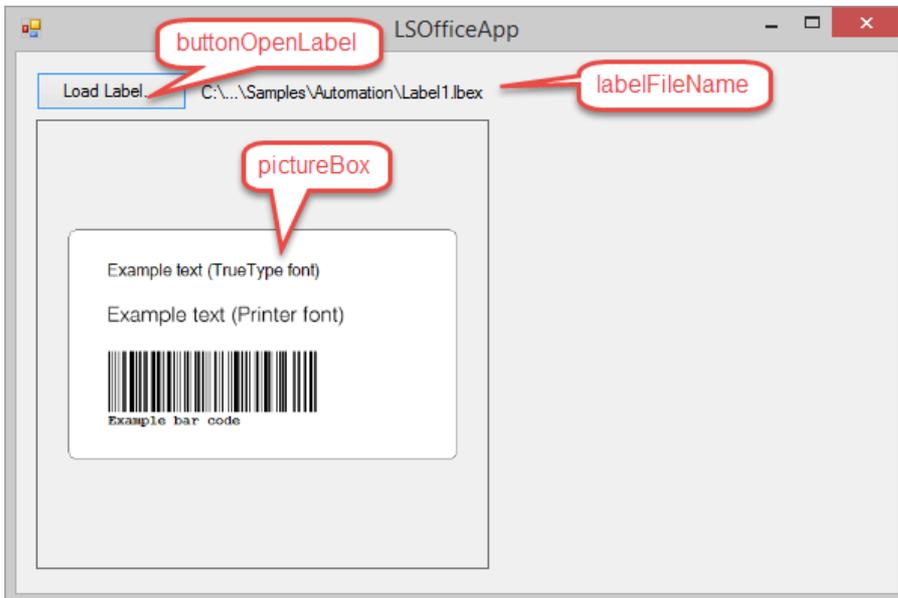
End Function
```

Schritt 5: Etikett öffnen

In diesem Schritt wird gezeigt, wie Code geschrieben wird, damit die Schaltfläche **Load Label** wie folgt funktioniert:

- Wenn der Benutzer die Schaltfläche auswählt, öffnet das Programm das Dialogfeld **Datei öffnen**.
- Wenn der Benutzer ein Etikett öffnet, zeigt das Programm ein Vorschaubild im PictureBox-Steuerelement an.

Fügen Sie eine Schaltfläche, ein Label- und ein PictureBox-Steuerelement zu dem Fenster hinzu.



Doppelklicken Sie auf die Schaltfläche **Load Label**, um den Standard-Click-Ereignishandler zu erstellen, und fügen Sie den folgenden Code hinzu:

C#

```
LSOffice.Label objLabel;

private void buttonOpenLabel_Click (object sender, EventArgs e)
{
    objLabel = null;

    OpenFileDialog dialog = new OpenFileDialog ();
    dialog.Filter = "Labels|*.lbex";

    if (dialog.ShowDialog () == DialogResult.OK)
        objLabel = objApp.OpenLabel (dialog.FileName);

    if (objLabel == null)
    {
        labelFileName.Text = "No label opened.";
        pictureBox.Image = null;
    }

    else
    {
        labelFileName.Text = objLabel.LabelPath;
        pictureBox.Image = objLabel.GetPreview ();
    }

    DisplayLastError ();
}
```

VB.Net

```
Dim objLabel As LSOffice.Label

Private Sub ButtonOpenLabel_Click(sender As Object, e As EventArgs) Handles ButtonOpenLabel.Click

    objLabel = Nothing

    Dim dialog As OpenFileDialog = New OpenFileDialog
    dialog.Filter = "Labels|*.lbex"

    If (dialog.ShowDialog() = DialogResult.OK) Then
        objLabel = objApp.OpenLabel(dialog.FileName)
    End If

    If (objLabel Is Nothing) Then
        LabelFileName.Text = "No label opened."
        PictureBox.Image = Nothing
    Else
        LabelFileName.Text = objLabel.LabelPath
        PictureBox.Image = objLabel.GetPreview
    End If

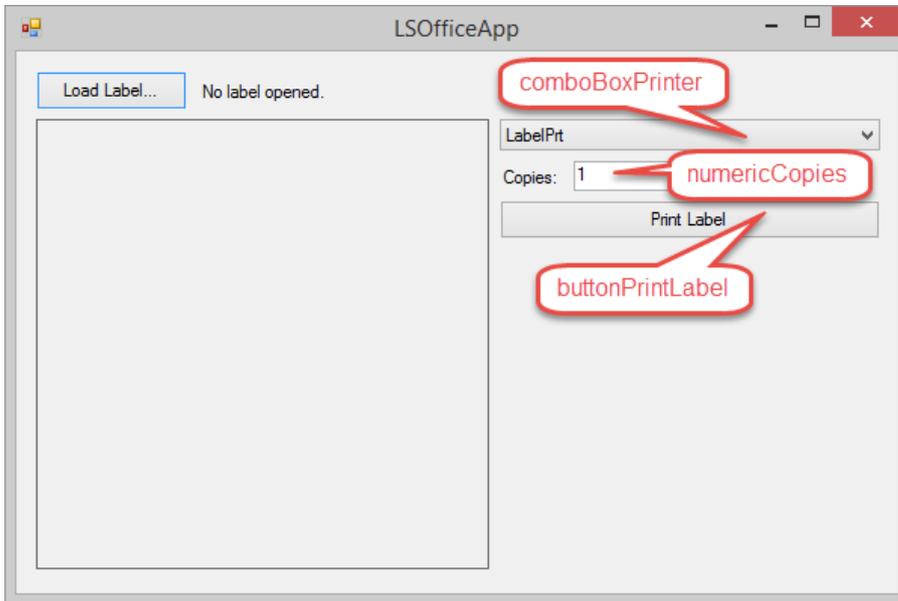
    DisplayLastError()

End Sub
```

Schritt 6: Etikett drucken

In diesem Schritt wird gezeigt, wie Code geschrieben wird, um das geöffnete Etikett auf dem ausgewählten Drucker zu drucken.

Fügen Sie eine Schaltfläche, ein ComboBox- und ein NumericUpDown-Steuerelement zu dem Fenster hinzu.



Um die in Windows installierten Drucker anzuzeigen, fügen Sie folgenden Code hinzu:

C#

```
protected override void OnLoad (EventArgs e)
{
    // Initialize application

    ...

    // Insert installed printers
    foreach (string installedPrinter in System.Drawing.Printing.PrinterSettings.InstalledPrinters)
        comboBoxPrinter.Items.Insert (0, installedPrinter);

    // Set the combo box to the active printer
    comboBoxPrinter.SelectedItem = objApp.ActivePrinter;

    base.OnLoad (e);
}
```

VB.Net

```
Protected Overrides Sub OnLoad(ByVal e As EventArgs)

    'Initialize application

    ...

    'Insert installed printers
    For Each installedPrinter In Printing.PrinterSettings.InstalledPrinters
        ComboBoxPrinter.Items.Insert(0, installedPrinter)
    Next installedPrinter

    'Set the combo box to the active printer
    ComboBoxPrinter.SelectedItem = objApp.ActivePrinter

    MyBase.OnLoad(e)

End Sub
```

Doppelklicken Sie auf die Auswahlliste, um den Standard-SelectedIndexChanged-Ereignishandler zu erstellen, und fügen Sie den folgenden Code hinzu:

C#

```
private void comboBoxPrinter_SelectedIndexChanged (object sender, EventArgs e)
{
    if (objLabel == null)
        return;

    objLabel.ActivePrinter = (string)comboBoxPrinter.SelectedItem;
    pictureBox.Image = objLabel.GetPreview ();

    DisplayLastError ();
}
```

VB.Net

```
Private Sub ComboBoxPrinter_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
ComboBoxPrinter.SelectedIndexChanged

    If (objLabel Is Nothing) Then
        Return
    End If

    objLabel.ActivePrinter = ComboBoxPrinter.SelectedItem
    PictureBox.Image = objLabel.GetPreview

    DisplayLastError()

End Sub
```

Doppelklicken Sie auf die Schaltfläche **Print Label**, um den Standard-Click-Ereignishandler zu erstellen, und fügen Sie den folgenden Code hinzu:

C#

```
private void buttonPrintLabel_Click (object sender, EventArgs e)
{
    if (objLabel == null)
    {
        MessageBox.Show ("No label opened.");
        return;
    }

    objLabel.Copies = (int)numericCopies.Value;
    objLabel.Print ();
}
```

VB.Net

```
Private Sub ButtonPrintLabel_Click(sender As Object, e As EventArgs) Handles ButtonPrintLabel.Click

    If (objLabel Is Nothing) Then
        MsgBox("No label opened.")
        Return
    End If

    objLabel.Copies = NumericCopies.Value
    objLabel.Print()

End Sub
```

Schritt 7: Programm ausführen

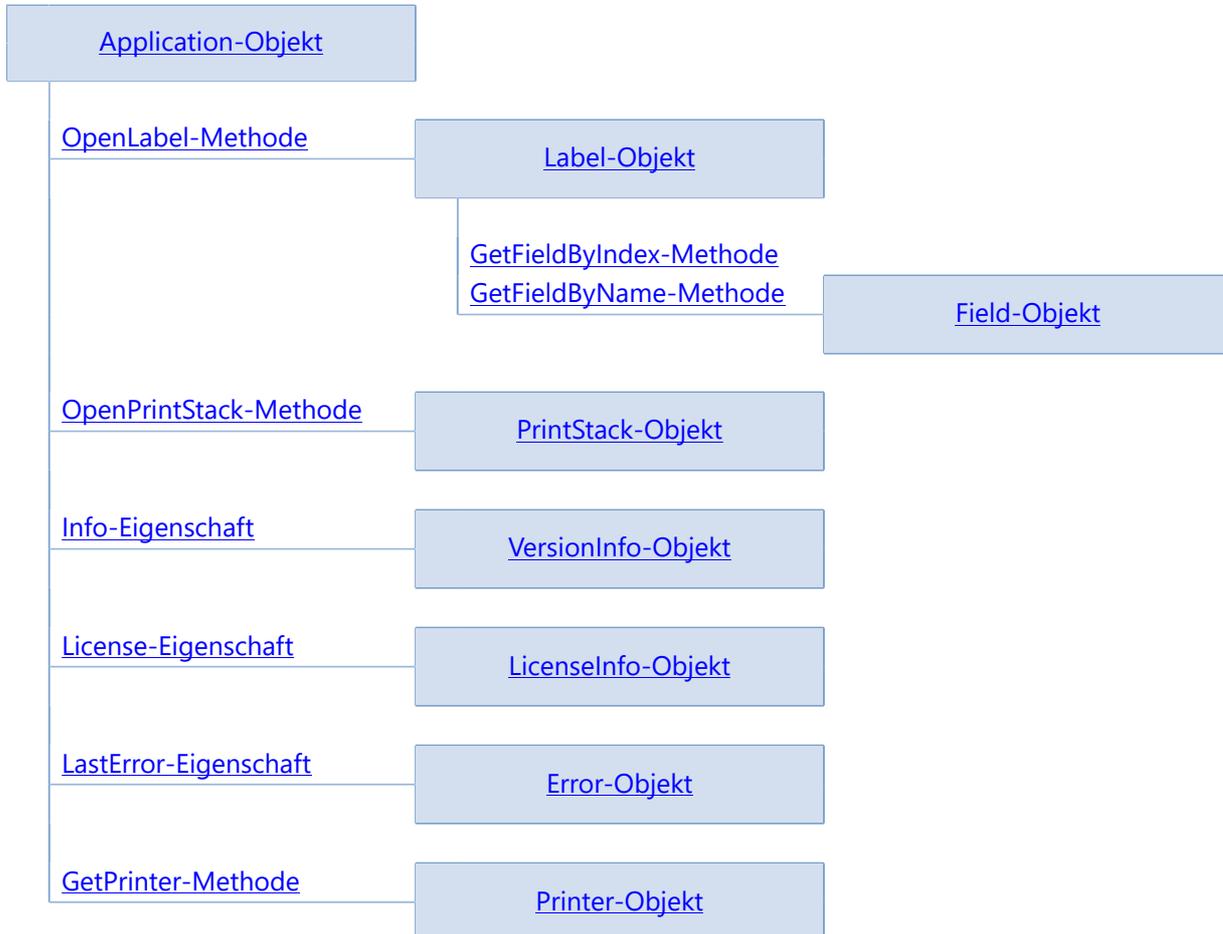
Das Programm ist fertig und bereit zur Ausführung. Um den Lerneffekt zu erhöhen, können Sie das Programm verbessern, indem Sie dem Benutzer die Möglichkeit geben den Inhalt der Felder auf dem Etikett zu ändern oder eine Datensatzauswahl für Datenbanketiketten implementieren.

Die fertige Version dieses Beispiels und weitere Beispielprojekte finden Sie im Verzeichnis *<Installationsverzeichnis>* \Automation. Bevor Sie Änderungen an den Beispielprojekten vornehmen, sollten Sie den Ordner (und dessen gesamten Inhalt) kopieren.

Programmierschnittstelle

In diesem Abschnitt erhalten Sie eine kurze Übersicht über die Funktionen der Programmierschnittstelle von **Labelstar Office**, die in LSOoffice.dll enthalten sind. Diese Eigenschaften und Methoden können in Kundenanwendungen verwendet werden um den Etikettendruck zu steuern.

Die folgende Abbildung zeigt, wie die einzelnen Objekte zusammenhängen:



Application-Objekt

Ein **Application**-Objekt repräsentiert die gesamte **Labelstar Office** Anwendung und ist das oberste Objekt der Objekthierarchie, von dem aus alle anderen Objekte abgeleitet werden. Das **Application**-Objekt ermöglicht das Öffnen, Speichern und Drucken von Etiketten, sowie das Ändern des Etiketteninhalts.

Siehe auch, [Verwenden des Application-Objekts](#).

Eigenschaften

	Name	Beschreibung
	ActivePrinter	Gibt den Namen des aktiven Druckers zurück.
	HasError	Ruft einen Wert ab, der angibt, ob während des letzten Aufrufs einer Methode oder Eigenschaft ein Fehler aufgetreten ist.
	Info	Ruft ein VersionInfo -Objekt ab, das Informationen über die API bereitstellt, z.B. Versionsnummer, Copyright usw.
	IsInitialized	Ruft einen Wert ab, der angibt, ob die Initialize -Methode bereits aufgerufen worden ist oder nicht.
	LabelDir	Gibt den Pfad des aktuellen Etikettenverzeichnisses zurück.
	LastError	Ruft den ein Error -Objekt ab, das Informationen über den zuletzt aufgetretenen Fehler enthält.
	License	Ruft ein LicenseInfo -Objekt ab, das Eigenschaften zum Abrufen der Lizenzinformationen bereitstellt.

Methoden

	Name	Beschreibung
	GetPrinter (string)	Gibt ein Printer -Objekt zurück.
	Import (string, string)	Importiert ein Labelstar Plus Etikett.
	Initialize ()	Initialisiert das Application -Objekt auf Grundlage der aktuellen Programmeinstellungen.
	OpenLabel (string)	Öffnet das angegebene Etikett und gibt ein Label -Objekt zurück.
	OpenPrintStack (string)	Öffnet den angegebenen Druckstapel und gibt ein PrintStack -Objekt zurück.

ActivePrinter-Eigenschaft

Gibt den Namen des aktiven Druckers zurück. Nur-Lese-Eigenschaft.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.10.1010

Syntax

Application.ActivePrinter

Typ

String

Beispiel (C#)

```
LSOoffice.Application objApp;

// Initialize application
objApp = new LSOoffice.Application ();
objApp.Initialize ();

if (objApp.HasError)
    MessageBox.Show (objApp.LastError.Message, "Error");
else
    MessageBox.Show ($"The name of the active printer is {objApp.ActivePrinter}", "Demo");
```

Beispiel (VB.Net)

```
Dim objApp As LSOoffice.Application

'Initialize application
objApp = New LSOoffice.Application
objApp.Initialize ()

If (objApp.HasError) Then
    MsgBox (objApp.LastError.Message, vbOKOnly, "Error")
Else
    MsgBox ("The name of the active printer is " & objApp.ActivePrinter, vbOKOnly, "Demo")
End If
```

See also

➤ [Application Object](#)

GetPrinter-Methode

Gibt ein [Printer](#)-Objekt zurück.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 6.20.1010

Syntax

```
Application.GetPrinter (printerName)
```

Parameter

printerName

Type: String

Druckername.

Rückgabewert

Typ: [LSOoffice.Printer](#)

Gibt ein [Printer](#)-Objekt oder **null** zurück, wenn der Druckername ungültig ist. Um weitere Fehlerinformationen zu erhalten, überprüfen Sie den Wert der Eigenschaft [LastError](#).

Hinweis

Wenn Sie ein [Printer](#)-Objekt erzeugt haben und nicht mehr benötigen, müssen Sie die Methode [Printer.Dispose](#) aufrufen.

Siehe auch

- [Verwenden des Printer-Objekts](#)
- [Application Object](#)

HasError-Eigenschaft

Ruft einen Wert ab, der angibt, ob während des letzten Aufrufs einer Methode oder Eigenschaft ein Fehler aufgetreten ist.
Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

Application.HasError

Typ

Boolean

Hinweis

Codebeispiele, die die Verwendung der **HasError**-Eigenschaft veranschaulichen, finden Sie [hier](#).

Siehe auch

- › [Verwenden des Application-Objekts](#)
- › [Application-Objekt](#)

Import-Methode

Importiert ein Labelstar Plus Etikett.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.50.1010

Syntax

```
Application.Import (sourceFileName, destFileName)
```

Parameter

sourceFileName

Type: String
Der Pfadname der Datei, die importiert werden soll.

destFileName

Type: String
Der neue Pfadname der importierten Datei.

Hinweis

Überprüfen Sie die Eigenschaft [LastError](#) um festzustellen, ob die Funktion erfolgreich abgeschlossen worden ist.

Info-Eigenschaft

Ruft ein [VersionInfo](#)-Objekt ab, das Informationen über die API bereitstellt, z.B. Versionsnummer, Copyright usw. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Application.Info

Typ

[LSOoffice.VersionInfo](#)

Siehe auch

➤ [Application-Objekt](#)

Initialize-Methode

Initialisiert das **Application**-Objekt auf Grundlage der aktuellen Programmeinstellungen.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

```
Application.Initialize ()
```

Hinweis

Bevor das **Application**-Objekt verwendet werden kann muss es mit dieser Methode initialisiert werden. Mit Hilfe der Eigenschaft [IsInitialized](#) können Sie feststellen, ob die Funktion bereits aufgerufen worden ist oder nicht.

Überprüfen Sie die Eigenschaft [LastError](#) um festzustellen, ob die Funktion erfolgreich abgeschlossen worden ist.

Siehe auch

- › [Verwenden des Application-Objekts](#)
- › [Application-Objekt](#)

IsInitialized-Eigenschaft

Ruft einen Wert ab, der angibt, ob die [Initialize](#)-Methode bereits aufgerufen worden ist oder nicht. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

```
Application.IsInitialized
```

Typ

Boolean

Hinweis

Verwenden Sie diese Eigenschaft, um zu überprüfen ob das [Application](#)-Objekt bereits initialisiert worden ist oder nicht. Die [Initialize](#)-Methode sollte nur einmal aufgerufen werden.

Siehe auch

➤ [Application-Objekt](#)

LabelDir-Eigenschaft

Gibt den Pfad des aktuellen Etikettenverzeichnisses zurück. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Application.LabelDir

Typ

String

Siehe auch

➤ [Application-Objekt](#)

LastError-Eigenschaft

Ruft den ein [Error](#)-Objekt ab, das Informationen über den zuletzt aufgetretenen Fehler enthält. Nur-Lese-Eigenschaft.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.10.1010

Syntax

Application.LastError

Typ

[LSOoffice.Error](#)

Hinweis

Nach jedem Aufruf einer Methode oder Eigenschaft eines Objekts wird diese Eigenschaft mit den Rückgabeinformationen des Aufrufs aktualisiert. Um festzustellen, ob ein Aufruf erfolgreich war oder nicht, überprüfen Sie die Eigenschaft [HasError](#).

Weitere Informationen finden Sie unter [Fehlercodes](#).

Beispiel (C#)

```
LSOoffice.Application objApp;

// Initialize application
objApp = new LSOoffice.Application ();
objApp.Initialize ();

if (objApp.HasError)
{
    MessageBox.Show (objApp.LastError.Message, "Error");
}

else
{
    // Open label
    LSOoffice.Label label = objApp.OpenLabel ("C:\\\\Label1.lbx");

    if (label == null)
        MessageBox.Show (objApp.LastError.Message, "Error");
}
```

Beispiel (VB.Net)

```
Dim objApp As LSOffice.Application
Dim objLabel As LSOffice.Label

'Initialize application
objApp = New LSOffice.Application
objApp.Initialize()

If (objApp.HasError) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
Else
    'Open label
    objLabel = objApp.OpenLabel("C:\\Label1.lbx")

    If (objLabel Is Nothing) Then
        MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
    End If
End If
```

See also

- [Using the Application Object](#)
- [Application Object](#)

License-Eigenschaft

Ruft ein [LicenseInfo](#)-Objekt ab, das Eigenschaften zum Abrufen der Lizenzinformationen bereitstellt. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Application.License

Typ

[LSOoffice.LicenseInfo](#)

Hinweis

In der Testversion wird zu jeder Grafik ein Wasserzeichen hinzugefügt und es werden alle 'e' durch 'x' und alle '5' durch '0' ersetzt.

Siehe auch

➤ [Application-Objekt](#)

OpenLabel-Methode

Öffnet das angegebene Etikett und gibt ein [Label](#)-Objekt zurück.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Application.OpenLabel (path)

Parameter

path
Type: String
Etikettenpfad

Rückgabewert

Typ: [LSOoffice.Label](#)

Gibt ein [Label](#)-Objekt zurück, wenn das Etikett erfolgreich geöffnet worden ist, sonst **null**. Um weitere Fehlerinformationen zu erhalten, überprüfen Sie den Wert der Eigenschaft [LastError](#).

Siehe auch

- [Verwenden des Label-Objekts](#)
- [Application-Objekt](#)

OpenPrintStack-Methode

Öffnet den angegebenen Druckauftrag und gibt ein [PrintStack](#)-Objekt zurück.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.50.1010

Syntax

Application.OpenPrintJob (path)

Parameter

path
Type: String
Dateipfad

Rückgabewert

Typ: [LSOoffice.PrintStack](#)
Gibt ein [PrintStack](#)-Objekt zurück, wenn die Datei erfolgreich geöffnet worden ist, sonst **null**. Um weitere Fehlerinformationen zu erhalten, überprüfen Sie den Wert der Eigenschaft [LastError](#).

Error-Objekt

Das **Error**-Objekt stellt Eigenschaften zum Abrufen des aktuellen Fehlerstatus bereit. Verwenden Sie die Eigenschaft [Application.LastError](#) , um eine Referenz auf das **Error**-Objekt zu erhalten.

Eigenschaften

	Name	Beschreibung
	Details	Detaillierte Fehlerbeschreibung des zuletzt aufgetretenen Fehlers.
	ErrorCode	Fehlercode des zuletzt aufgetretenen Fehlers.
	ErrorType	Fehlertyp des zuletzt aufgetretenen Fehlers.
	Message	Fehlerbeschreibung des zuletzt aufgetretenen Fehlers.

Details-Eigenschaft

Detaillierte Fehlerbeschreibung des zuletzt aufgetretenen Fehlers. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Error.Details

Typ

String

Hinweis

Gibt eine detaillierte Fehlermeldung (incl. Stack Trace) zurück, die die Ursache des zuletzt aufgetretenen Fehlers erklärt, oder eine leere Zeichenfolge (""), wenn kein Fehler aufgetreten ist.

Siehe auch

➤ [Error-Objekt](#)

ErrorCode-Eigenschaft

Fehlercode des zuletzt aufgetretenen Fehlers. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Error.ErrorCode

Typ

Integer

Hinweis

Ist diese Eigenschaft gleich 0, so bedeutet das, dass während des letzte Aufrufs einer Methode oder Eigenschaft kein Fehler aufgetreten ist. Weitere Informationen finden Sie unter [Fehlercodes](#).

Siehe auch

➤ [Error-Objekt](#)

ErrorType Property

Fehlertyp des zuletzt aufgetretenen Fehlers. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Error.ErrorType

Typ

LSOoffice.ErrorType

	Name	Wert	Beschreibung
	Success	0 (0x00)	Es ist kein Fehler aufgetreten.
	Warning	1 (0x01)	Der Aufruf war erfolgreich, aber es gibt ein mögliches Problem.
	Error	2 (0x02)	Der Aufruf ist fehlgeschlagen.

Siehe auch

- [Verwenden des Application-Objekts](#)
- [Error-Objekt](#)

Message Property

Fehlerbeschreibung des zuletzt aufgetretenen Fehlers. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Error.Message

Typ

String

Hinweis

Gibt eine Fehlermeldung zurück, die die Ursache des zuletzt aufgetretenen Fehlers erklärt, oder eine leere Zeichenfolge (""), wenn kein Fehler aufgetreten ist. Weiter Informationen finden Sie unter [Fehlercodes](#).

Siehe auch

- [Verwenden des Application-Objekts](#)
- [Error-Objekt](#)

Field-Objekt

Ein **Field**-Objekt stellt ein Feld auf einem Etikett dar. Es kann verwendet werden, um den Feldinhalt bzw. -eigenschaften auszulesen und zu ändern. Sie können eine Referenz auf ein **Field**-Objekt über die Methode [Label.GetFieldByIndex](#) oder [Label.GetFieldByName](#) erstellen.

Siehe auch, [Verwenden des Label-Objekts](#).

Eigenschaften

	Name	Beschreibung
	FieldName	Gibt den Feldnamen zurück.
	Locked	Gibt einen Wert zurück der angibt, ob das Feld geändert werden kann oder nicht.
	Printable	Gibt einen Wert zurück der angibt, ob das Feld gedruckt wird oder nicht.

Methoden

	Name	Beschreibung
	GetContent ()	Gibt den aktuellen Feldinhalt zurück.
	GetPropertyValue (string)	Gibt den Wert der Eigenschaft zurück.
	SetContent (string)	Setzt den aktuellen Feldinhalt.
	SetPropertyValue (string, object)	Setzt den Wert der Eigenschaft.

FieldName-Eigenschaft

Gibt den Feldnamen zurück. Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

Field.FieldName

Typ

String

Siehe auch

➤ [Field-Objekt](#)

GetContent-Methode

Gibt den aktuellen Feldinhalt zurück.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

Field.GetContent ()

Rückgabewert

Typ: String

Gibt den aktuellen Feldinhalt zurück, wenn die Methode erfolgreich abgeschlossen wurde, sonst eine leere Zeichenfolge (""). Überprüfen Sie die Eigenschaft [Application.LastError](#) um Informationen über mögliche Fehler zu erhalten.

Siehe auch

➤ [Field-Objekt](#)

GetPropertyValue Method

Gibt den Wert der Eigenschaft zurück.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.10.1010

Syntax

Field.GetPropertyValue (propertyName)

Parameter

propertyName

Typ: String

Name der Eigenschaft deren Wert ausgelesen werden soll. Eine Liste der möglichen Eigenschaftennamen finden Sie [hier](#).

Rückgabewert

Typ: Object

Gibt den Wert der Eigenschaft zurück, wenn die Methode erfolgreich abgeschlossen wurde, sonst **null**. Überprüfen Sie die Eigenschaft [Application.LastError](#) um Informationen über mögliche Fehler zu erhalten.

Siehe auch

➤ [Feld-Objekt](#)

Eigenschaftennamen

Die folgende Tabelle beschreibt einige möglich Eigenschaftennamen, die im Zusammenhang mit Feldern verwendet werden können.

Eigenschaftename	Beschreibung
HumanReadable	Nur Barcodes Typ: Boolean false oder 0: Klartextzeile nicht anzeigen true oder 1: Klartextzeile anzeigen
Locked	Typ: Boolean false oder 0: Das Feld ist nicht gesperrt true oder 1: Das Feld ist gesperrt Siehe auch, Locked-Eigenschaft
Printable	Typ: Boolean false oder 0: Das Feld wird nicht gedruckt true oder 1: Das Feld wird gedruckt Siehe auch, Printable-Eigenschaft
TextAlignment	Nur Texte/Barcodes Typ: Integer 0: Links 1: Zentriert 2: Rechts 3: Blocksatz

Locked-Eigenschaft

Gibt einen Wert zurück der angibt, ob das Feld geändert werden kann oder nicht. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Field.Locked

Typ

Boolean

Siehe auch

➤ [Field-Objekt](#)

Printable-Eigenschaft

Gibt einen Wert zurück der angibt, ob das Feld gedruckt wird oder nicht.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Field.Printable

Typ

Boolean

Siehe auch

➤ [Field-Objekt](#)

SetContent-Methode

Setzt den aktuellen Feldinhalt.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Field.SetContent (content)

Parameter

content
Type: String
Neuer Feldinhalt.

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um Informationen über mögliche Fehler zu erhalten.

Siehe auch

- [Verwenden des Label-Objekts](#)
- [Field-Objekt](#)

SetPropertyValue-Methode

Setzt den Wert der Eigenschaft.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.10.1010

Syntax

```
Field.SetPropertyValue (propertyName, value)
```

Parameter

propertyName

Typ: String

Name der Eigenschaft deren Wert geändert werden soll. Eine Liste der möglichen Eigenschaftennamen finden Sie [hier](#).

value

Typ: Object

Neuer Eigenschaftswert.

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um Informationen über mögliche Fehler zu erhalten.

Siehe auch

➤ [Field-Objekt](#)

Label-Objekt

Ein **Label**-Objekt repräsentiert ein geöffnetes Etikett. Sie erhalten eine Referenz für ein **Label**-Objekt wenn Sie die Methode [Application.OpenLabel](#) aufrufen.

Siehe auch, [Verwenden des Label-Objekts](#).

Eigenschaften

	Name	Beschreibung
	ActivePrinter	Gibt den Namen des aktiven Druckers zurück oder legt ihn fest.
	Copies	Gibt die Druckanzahl zurück oder legt sie fest.
	CurrentRecord	Gibt die Nummer (beginnend bei eins) des Datensatzes zurück, der gedruckt werden soll, oder legt sie fest.
🔒	FieldCount	Gibt die Anzahl der Felder zurück, die auf dem Etikett definiert sind.
🔒	FieldNames	Gibt eine Liste der Feldnamen zurück, die auf dem Etikett definiert sind.
🔒	IsDataAvailable	Ruft einen Wert ab, der angibt, ob Datenbankfelder auf dem Etikett definiert sind oder nicht.
🔒	LabelPath	Gibt den Pfad des geöffneten Etiketts zurück.
🔒	MaxRecord	Gibt die maximale Anzahl von Datensätzen zurück.
🔒	Modified	Ruf einen Wert ab, der angibt, ob sich der Etiketteninhalt geändert hat oder nicht.
	PageName	Gibt den Seitennamen zurück oder legt ihn fest.
	PrintOptions	Gibt die Druckoptionen zurück oder legt sie fest.

Methoden

	Name	Beschreibung
	GetFieldByIndex (int)	Sucht das Feld mit dem angegebenen Index.
	GetFieldByName (string)	Sucht das Feld mit dem angegebenen Namen.
	GetPreview ()	Erstellt ein Vorschaubild des aktuellen Etiketteninhalts.
	GetPropertyValue (string)	Gibt den Wert der Eigenschaft zurück.
	Print ()	Druckt das Etikett unter Verwendung der aktuellen Druckeinstellungen.
	Print (int)	Druckt das Etikett mit der angegebenen Druckanzahl.
	Print (int, PrintOptions)	Druckt das Etikett mit der angegebenen Druckanzahl und -optionen.
	PrintToFile (string)	Erstellt eine Druckdatei unter Verwendung der aktuellen Druckeinstellungen.
	PrintToFile (string, int)	Erstellt eine Druckdatei mit der angegebenen Druckanzahl.
	PrintToFile (string, int, PrintOptions)	Erstellt eine Druckdatei mit der angegebenen Druckanzahl und -optionen.
	Save ()	Speichert das Etikett.
	SaveAs (string)	Speichert das Etikett unter einem anderen Namen.
	SavePreview (string)	Speichert ein Vorschaubild des aktuellen Etiketteninhalts als Bitmap-Datei.
	SavePreview (string, ImageFormat)	Speichert ein Vorschaubild des aktuellen Etiketteninhalts.
	SelectRecord (string)	Sucht den angegebenen Datensatz.
	SelectRecord (string, string)	Sucht den angegebenen Datensatz in der ausgewählten Datenbank.
	SetPropertyValue (string, object)	Setzt den Wert der Eigenschaft.

ActivePrinter-Eigenschaft

Gibt den Namen des aktiven Druckers zurück oder legt ihn fest.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Label.ActivePrinter

Typ

String

Siehe auch

➤ [Label-Objekt](#)

Copies-Eigenschaft

Gibt die Druckanzahl zurück oder legt sie fest.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

Label.Copies

Typ

Integer

Hinweis

Ist die Druckanzahl > 0 wird das Dialogfeld **Drucken** nicht angezeigt.

Siehe auch

- [Verwenden des Label-Objekts](#)
- [Label-Objekt](#)

CurrentRecord-Eigenschaft

Gibt die Nummer (beginnend bei eins) des Datensatzes zurück, der gedruckt werden soll, oder legt sie fest.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Label1.CurrentRecord

Typ

Integer

Hinweis

Mit der Eigenschaft [IsDataAvailable](#) können Sie überprüfen, ob Datenbankfelder auf dem Etikett definiert sind oder nicht.

Siehe auch

➤ [Label-Objekt](#)

FieldCount-Eigenschaft

Gibt die Anzahl der Felder zurück, die auf dem Etikett definiert sind. Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

Label.FieldCount

Typ

Integer

Siehe auch

➤ [Label-Objekt](#)

FieldNames-Eigenschaft

Gibt eine Liste der Feldnamen zurück, die auf dem Etikett definiert sind. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Label.FieldNames

Typ

String[]

Siehe auch

➤ [Label-Objekt](#)

GetFieldByIndex-Methode

Sucht das Feld mit dem angegebenen Index.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Label.GetField (index)

Parameter

index

Typ: Integer
Feldindex (beginnend mit Null)

Rückgabewert

Typ: [LSOoffice.Field](#)

Gibt eine Referenz auf ein [Field](#)-Objekt zurück, oder **null**, falls kein Feld gefunden worden ist. Um weitere Fehlerinformationen zu erhalten, überprüfen Sie den Wert der Eigenschaft [Application.LastError](#).

Siehe auch

➤ [Label-Objekt](#)

GetFieldByName-Methode

Sucht das Feld mit dem angegebenen Namen.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

```
Label.GetField (fieldName)
```

Parameter

fieldName
Typ: String
Feldname

Rückgabewert

Typ: [LSOoffice.Field](#)

Gibt eine Referenz auf ein [Field](#)-Objekt zurück, oder **null**, falls kein Feld gefunden worden ist. Um weitere Fehlerinformationen zu erhalten, überprüfen Sie den Wert der Eigenschaft [Application.LastError](#).

Siehe auch

- [Verwenden des Label-Objekts](#)
- [Label-Objekt](#)

GetPreview-Methode

Erstellt ein Vorschaubild des aktuellen Etiketteninhalts.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

Label.GetPreview ()

Rückgabewert

Typ: Bitmap

Gibt ein **Bitmap**-Objekt zurück, wenn ein gültiges Etikett geöffnet ist, sonst **null**.

Siehe auch

➤ [Label-Objekt](#)

GetPropertyValue Method

Gibt den Wert der Eigenschaft zurück.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.20.1040

Syntax

```
Label.GetPropertyValue (propertyName)
```

Parameter

propertyName

Typ: String

Name der Eigenschaft deren Wert ausgelesen werden soll. Eine Liste der möglichen Eigenschaftennamen finden Sie [hier](#).

Rückgabewert

Typ: Object

Gibt den Wert der Eigenschaft zurück, wenn die Methode erfolgreich abgeschlossen wurde, sonst **null**. Überprüfen Sie die Eigenschaft [Application.LastError](#) um Informationen über mögliche Fehler zu erhalten.

Siehe auch

➤ [Label-Objekt](#)

Eigenschaftennamen

Die folgende Tabelle beschreibt einige möglich Eigenschaftennamen, die im Zusammenhang mit Etiketten verwendet werden können.

Eigenschaftename	Beschreibung
LabelRotation	Typ: Integer 0, 90, 180, 270
LabelType	Typ: Integer 0: Haftetiketten 1: Endlosetiketten
PrinterName	Typ: String

IsDataAvailable-Eigenschaft

Ruft einen Wert ab, der angibt, ob Datenbankfelder auf dem Etikett definiert sind oder nicht. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Label.IsDataAvailable

Typ

Boolean

Siehe auch

➤ [Label-Objekt](#)

LabelPath-Eigenschaft

Gibt den Pfad des geöffneten Etiketts zurück. Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

Label.LabelPath

Typ

String

Siehe auch

➤ [Label-Objekt](#)

MaxRecord-Eigenschaft

Gibt die maximale Anzahl von Datensätzen zurück. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Label.MaxRecord

Typ

Integer

Hinweis

Gibt die maximale Anzahl von Datensätzen in der Datenbank zurück, oder 0, wenn keine Datenbankfelder auf dem Etikett definiert sind. Mit der Eigenschaft [IsDataAvailable](#) können Sie überprüfen, ob Datenbankfelder auf dem Etikett definiert sind oder nicht.

Siehe auch

➤ [Label-Objekt](#)

Modified-Eigenschaft

Ruf einen Wert ab, der angibt, ob sich der Etiketteninhalt geändert hat oder nicht. Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

Label.Modified

Typ

Boolean

Siehe auch

➤ [Label-Objekt](#)

PageName-Eigenschaft

Gibt den Seitennamen zurück oder legt ihn fest.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.20.1040

Syntax

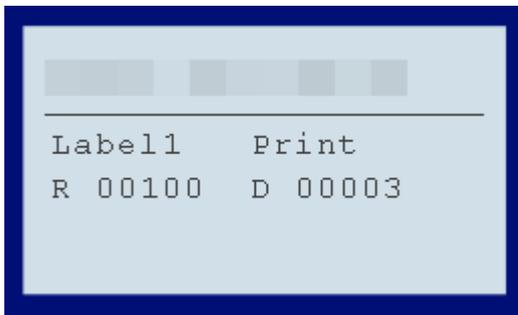
Label1.PageName

Typ

String

Hinweis

Der Seitenname wird während des Drucks im Druckerdisplay angezeigt.



Siehe auch

➤ [Label-Objekt](#)

Print-Methode

Druckt das Etikett unter Verwendung der aktuellen Druckeinstellungen.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

Label.Print ()

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- › [Print-Methode \(int\)](#)
- › [Print-Methode \(int, PrintOptions\)](#)
- › [Verwenden des Label-Objekts](#)
- › [Label-Objekt](#)

Print-Methode (int)

Druckt das Etikett mit der angegebenen Druckanzahl.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.40.1010

Syntax

Label.Print (copies)

Parameter

copies

Typ: Integer

Druckanzahl. Ist *copies* > 0 wird das Dialogfeld **Drucken** nicht angezeigt.

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- [Print-Methode](#)
- [Print-Methode \(int, PrintOptions\)](#)
- [Label-Objekt](#)

Print-Methode (int, PrintOptions)

Druckt das Etikett mit der angegebenen Druckanzahl und -optionen.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.10.1010

Syntax

Label.Print (copies, options)

Parameter

copies

Typ: Integer

Druckanzahl. Ist *copies* > 0 wird das Dialogfeld **Drucken** nicht angezeigt.

options

Typ: LSOoffice.PrintOptions

Druckoptionen. Eine detaillierte Beschreibung der Optionen finden Sie [hier](#).

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- [Print-Methode](#)
- [Print-Methode \(int\)](#)
- [Label-Objekt](#)

PrintOptions-Eigenschaft

Gibt die Druckoptionen zurück oder legt sie fest.

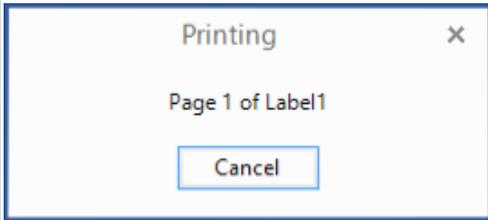
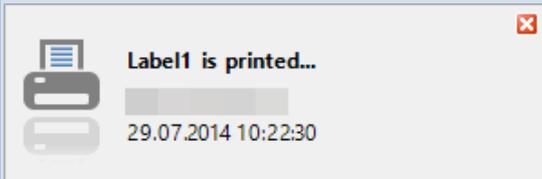
Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 6.20.1010

Syntax

Label.PrintOptions

Typ

LSOOffice.PrintOptions

Name	Wert	Beschreibung
Default	0 (0x00)	Standardeinstellungen
PrintCurrentRecord	1 (0x01)	Druckt den aktuell ausgewählten Datensatz. Das Dialogfeld Datensatz auswählen wird nicht angezeigt.
PrintAllRecords	2 (0x02)	Druckt alle Datensätze. Das Dialogfeld Datensatz auswählen wird nicht angezeigt.
ShowPrintingDialog	4 (0x04)	Zeigt einen Druckdialog an, solange Etiketten an den Drucker gesendet werden. 
ShowNotificationMessage	8 (0x08)	Zeigt eine Benachrichtigung im unteren rechten Bildschirmbereich an, wenn das Etikett gedruckt wird. 

Hinweis

Die einzelnen Werte der Aufzählung können bitweise verknüpft werden.

In den Methoden [Print](#) und [PrintToFile](#) wird diese Aufzählung ebenfalls verwendet.

Codebeispiele, die die Verwendung der **PrintOptions**-Eigenschaft veranschaulichen, finden Sie [hier](#).

Siehe auch

➤ [Label-Objekt](#)

PrintToFile-Methode (string)

Erstellt eine Druckdatei unter Verwendung der aktuellen Druckeinstellungen.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

```
Label.PrintToFile (fileName)
```

Parameter

fileName
Typ: String
Dateiname

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- [PrintToFile-Methode \(string, int\)](#)
- [PrintToFile-Methode \(string, int, PrintOptions\)](#)
- [Label-Objekt](#)

PrintToFile-Methode (string, int)

Erstellt eine Druckdatei mit der angegebenen Druckanzahl.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.40.1010

Syntax

```
Label.PrintToFile (fileName, copies)
```

Parameter

fileName

Typ: String

Dateiname

copies

Typ: Integer

Druckanzahl. Ist *copies* > 0 wird das Dialogfeld **Drucken** nicht angezeigt.

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- [PrintToFile-Methode \(string\)](#)
- [PrintToFile-Methode \(string, int, PrintOptions\)](#)
- [Label-Objekt](#)

PrintToFile-Methode (string, int, PrintOptions)

Erstellt eine Druckdatei mit der angegebenen Druckanzahl und -optionen.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.20.1040

Syntax

Label.PrintToFile (fileName, copies, options)

Parameter

fileName

Typ: String
Dateiname

copies

Typ: Integer
Druckanzahl. Ist *copies* > 0 wird das Dialogfeld **Drucken** nicht angezeigt.

options

Typ: LSOoffice.PrintOptions
Druckoptionen. Eine detaillierte Beschreibung der Optionen finden Sie [hier](#).

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- [PrintToFile-Methode \(string\)](#)
- [PrintToFile-Methode \(string, int\)](#)
- [Label-Objekt](#)

Save-Methode

Speichert das Etikett.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Label1.Save ()

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- › [SaveAs-Methode](#)
- › [Label-Objekt](#)

SaveAs-Methode

Speichert das Etikett unter einem anderen Namen.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

Label1.SaveAs (fileName)

Parameter

fileName
Typ: String
Dateiname

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- [Save-Methode](#)
- [Label-Objekt](#)

SavePreview-Methode (string)

Speichert ein Vorschaubild des aktuellen Etiketteninhalts als Bitmap-Datei.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.40.1010

Syntax

Label.SavePreview (fileName)

Parameter

fileName

Typ: String

Dateiname

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- [SavePreview-Methode \(string, ImageFormat\)](#)
- [Label Object](#)

SavePreview-Methode (string, ImageFormat)

Speichert ein Vorschaubild des aktuellen Etiketteninhalts.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 4.20.1040

Syntax

Label.SavePreview (fileName, format)

Parameter

fileName

Typ: String

Dateiname

format

Typ: LSOoffice.ImageFormat

Dateiformat

	Name	Value	Description
	Bmp	0 (0x00)	Speichert die Grafik im BMP-Format.
	Gif	1 (0x01)	Speichert die Grafik im GIF-Format.
	Jpeg	2 (0x02)	Speichert die Grafik im JPEG-Format.
	Png	3 (0x03)	Speichert die Grafik im PNG-Format.

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

- [SavePreview-Methode \(string\)](#)
- [Label-Objekt](#)

SelectRecord-Methode (string)

Sucht den angegebenen Datensatz.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

Label.SelectRecord (filterExpression)

Parameter

filterExpression

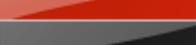
Typ: String

Filterkriterium. Beispiele dazu, wie Datensätze gefiltert werden, finden Sie unter [Filtersyntax](#).

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Beispieldatenbank (Europe.accdb)

ID	Name	Capital	Area	Population	NativeName	Flag
1	Albania	Tirana	28748	3560000	Shqipëria	
2	Andorra	Andorra la Vella	464	72300	Andorra	
3	Belgium	Brussels	30518	1030000	Belgique	
4	Bosnia and Her...	Sarajevo	51129	3800000	Босна и Херце...	
5	Bulgaria	Sofia	110994	7810000	България	
6	Denmark	Copenhagen	43093	5410000	Danmark	
7	Germany	Berlin	357114	82220000	Deutschland	
8	Estonia	Tallinn	45227	1350000	Eesti	
9	Finland	Helsinki	338145	5220000	Suomi	
10	France	Paris	547026	60660000	France	
11	Greece	Athens	131957	10670000	Ελλάς	
12	United Kingdom	London	244820	60440000	United Kingdom	
13	Ireland	Dublin	70284	4040000	Éire	

Beispiel (C#)

```
LSOffice.Application objApp;

// Initialize application
objApp = new LSOffice.Application ();
objApp.Initialize ();

if (objApp.HasError)
{
    MessageBox.Show (objApp.LastError.Message, "Error");
    return;
}

// Open label
LSOffice.Label objLabel = objApp.OpenLabel ("C:\\Label1.lbx");

if (objLabel == null)
{
    MessageBox.Show (objApp.LastError.Message, "Error");
    return;
}

objLabel.PrintOptions = LSOffice.PrintOptions.PrintCurrentRecord;
objLabel.Copies = 1;

// Print original label
objLabel.Print ();

// Select record
objLabel.CurrentRecord = 10;

if (objApp.HasError)
    MessageBox.Show (objApp.LastError.Message, "Error");
else
    objLabel.Print ();

// Select record
string selectionString = "Deutschland";
objLabel.SelectRecord ($"NativeName LIKE '{selectionString}'");

if (objApp.HasError)
    MessageBox.Show (objApp.LastError.Message, "Error");
else
    objLabel.Print ();
```

Beispiel (VB.NET)

```
Dim objApp As LSOffice.Application
Dim objLabel As LSOffice.Label

'Initialize application
objApp = New LSOffice.Application
objApp.Initialize()

If (objApp.HasError) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
    Return
End If

'Open label
objLabel = objApp.OpenLabel("C:\\Label1.lbx")

If (objLabel Is Nothing) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
    Return
End If

objLabel.PrintOptions = LSOffice.PrintOptions.PrintCurrentRecord
objLabel.Copies = 1

'Print original label
objLabel.Print()

'Select record
objLabel.CurrentRecord = 10

If (objApp.HasError) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
Else
    objLabel.Print()
End If

'Select record
Dim selectionString = "Deutschland"
objLabel.SelectRecord("NativeName LIKE'" & selectionString & "'")

If (objApp.HasError) Then
    MsgBox(objApp.LastError.Message, vbOKOnly, "Error")
Else
    objLabel.Print()
End If
```

Siehe auch

➤ [Label-Objekt](#)

Filtersyntax

Ein Filter wird mit der Filtersyntaxsprache generiert. Damit können Sie einfache, fortgeschrittene oder erweiterte Filter erzeugen.

Spaltennamen

Enthält ein Spaltennamen eines von diesen Sonderzeichen ~ () # \ / = > < + - * % & | ^ ' " [], so muss der Spaltenname in eckige Klammern [] eingeschlossen werden. Enthält der Spaltenname ein] oder \, so muss ein Backslash vor die Zeichen gesetzt werden (\] oder \\).

Beispiel	Beschreibung
"id = 10"	Kein Sonderzeichen im Spaltenname "id".
"\$id = 10"	Kein Sonderzeichen im Spaltenname "\$id".
"[#id] = 10"	Sonderzeichen "#" in Spaltenname "#id".
"[[id\]] = 10"	Sonderzeichen in Spaltenname "[id]".

Konstanten

Eine Konstante, gelegentlich auch als Literal- oder Skalarwert bezeichnet, ist ein Symbol, das einen bestimmten Datenwert repräsentiert. Das Format einer Konstante ist abhängig vom Datentyp des Werts, den sie repräsentiert.

Zeichenfolgenkonstanten werden in einfache Anführungszeichen ' ' eingeschlossen und enthalten alphanumerische Zeichen (a-z, A-Z und 0-9) sowie Sonderzeichen, wie z. B. Ausrufezeichen (!), @-Zeichen und Nummernzeichen (#). Enthält eine in einfache Anführungszeichen eingeschlossene Zeichenfolge ein eingeschlossenes Anführungszeichen, muss das eingeschlossene Anführungszeichen durch zwei einfache Anführungszeichen ersetzt werden.

Beispiel	Beschreibung
"Name = 'John'"	Zeichenfolgenkonstante
"Name = 'John "A"'"	Zeichenfolgenkonstante mit eingeschlossenem Anführungszeichen "John 'A'".

Zahlenwerte werden durch eine Zeichenfolge von Zahlen dargestellt, die nicht in Anführungszeichen eingeschlossen sind (englische Formatierung verwenden).

Beispiel	Beschreibung
"Year = 2008"	Ganzzahliger Wert
"Price = 1199.9"	Float-Wert

Datumskonstanten werden in # # eingeschlossen (englische Formatierung verwenden).

Beispiel	Beschreibung
"Date = #12/31/2008#"	Datum (Uhrzeit ist 00:00:00).
"Date = #2008-12-31#"	Auch dieses Format wird unterstützt.
"Date = #12/31/2008 16:44:58#"	Datum/Uhrzeit

Alternativ können auch alle Werte in einfache Anführungszeichen ' ' eingeschlossen werden. Das bedeutet, dass Textzeichenfolgen für Zahlen und Datum/Uhrzeit verwendet werden können. In diesem Fall wird die aktuelle Ländereinstellung verwendet um die Zeichenfolge in den zugehörigen Wert umzuwandeln.

Beispiel	Beschreibung
"Date = '12/31/2008 16:44:58'"	Ländereinstellung Englisch
"Date = '31.12.2008 16:44:58'"	Ländereinstellung Deutsch
"Price = '1199.90'"	Ländereinstellung Englisch
"Price = '1199,90'"	Ländereinstellung Deutsch

Vergleichsoperatoren

Vergleichsoperatoren testen, ob zwei Ausdrücke gleichwertig sind. **Vergleichsoperatoren** sind = (gleich), <> (ungleich), < (kleiner), <= (kleiner oder gleich), > (größer) oder >= (größer und gleich).

Hinweis: Beim Vergleich von Zeichenfolgen wird die Groß- und Kleinschreibung beachtet.

Beispiel	Beschreibung
"Num = 10"	Nummer ist gleich 10.
"Date < #1/1/2008#"	Datum vor dem 01.01.2008.
"Name <> 'John'"	Name ungleich John.
"Name >= 'Jo'"	Zeichenfolgenvergleich

Der **IN-Operator** ermittelt, ob ein angegebener Wert mit einem Wert aus einer Unterabfrage oder Liste übereinstimmt.

Beispiel	Beschreibung
"Id IN (1, 2, 3)"	Ganzzahlige Werte
"Price IN (1.0, 9.9, 11.5)"	Float-Werte
"Name IN ('John', 'Jim', 'Tom')"	Textzeichenfolgen
"Date IN (#12/31/2008#, #1/1/2009#)"	Datum/Uhrzeit
"Id NOT IN (1, 2, 3)"	Werte nicht in der Liste

Der **LIKE-Operator** bestimmt, ob eine bestimmte Zeichenfolge mit einem angegebenen Muster übereinstimmt. Ein Muster kann normale Zeichen und Platzhalterzeichen enthalten. Bei einem Mustervergleich müssen normale Zeichen exakt mit den angegebenen Zeichen in der Zeichenfolge übereinstimmen. Platzhalterzeichen können jedoch mit beliebigen Teilen der Zeichenfolge übereinstimmen. Das Verwenden der Vergleichsoperatoren für Zeichenfolgen = und <> ist nicht so flexibel wie das Verwenden von Platzhalterzeichen mit dem **LIKE-Operator**.

Platzhalterzeichen sind * oder %. Platzhalterzeichen können am Anfang '*value', am Ende 'value*', oder am Anfang und Ende '*value*' eines Musters eingefügt werden. Platzhalterzeichen in der Mitte eines Musters 'va*lue' sind **nicht erlaubt**.

Beispiel	Beschreibung
"Name LIKE 'j*'"	Alle Werte fangen mit 'j' an.
"Name LIKE '%jo%'"	Alle Werte enthalten 'jo'.
"Name NOT LIKE 'j*'"	Alle Werte fangen nicht mit 'j' an.

Enthält ein Muster eines von diesen Zeichen * % [], so müssen die Zeichen in eckige Klammern gesetzt werden, z.B. [*], [%], [[] or []].

Beispiel	Beschreibung
"Name LIKE '[*]*'"	Alle Werte fangen mit '*' an.
"Name LIKE '['*'"	Alle Werte fangen mit '[' an.

Logische Operatoren

Logische Operatoren sind **AND**, **OR** und **NOT** und testen den Wahrheitswert einer Bedingung. Der Operator NOT hat Vorrang vor AND und AND hat Vorrang vor OR.

Beispiel	Beschreibung
"City = 'Tokyo' AND (Age < 20 OR Age > 60)"	Alle Menschen die in Tokyo leben und zwischen 20 und 60 Jahre alt sind.
"City <> 'Tokyo' AND City <> 'Paris'; "NOT City = 'Tokyo' AND NOT City = 'Paris'; "NOT (City = 'Tokyo' OR City = 'Paris'); "City NOT IN ('Tokyo', 'Paris');"	Alle Menschen die nicht in Tokyo oder Paris leben.

Arithmetische und Zeichenfolgenoperatoren

Arithmetische Operatoren sind + (Addition), - (Subtraktion), * (Multiplikation), / (Division) und % (Modulo).

Beispiel	Beschreibung
"MotherAge - Age < 20"	Alle Menschen mit einer jungen Mutter.
"Age % 10 = 0"	Alle Menschen mit zehnjährigem Geburtstag.

Zusätzlich gibt es noch einen **Zeichenfolgenoperator**, + (Verkettung).

Parent-Child-Beziehung referenzieren

In einem Ausdruck kann auf eine **übergeordnete Tabelle (parent table)** referenziert werden, indem man dem Spaltenname das Prefix `Parent.` hinzufügt. Auf eine Spalte in einer **untergeordneten Tabelle (child table)** kann zugegriffen werden, wenn man dem Spaltennamen das Prefix `Child.` hinzufügt.

Der Zugriff auf die untergeordnete Spalte (child column) muss in einer Aggregatfunktion erfolgen, da untergeordnete Beziehungen (child relationships) mehrere Spalten zurückliefern können. Beispiel: `SUM(Child.Price)` liefert die Summe aller Preise in der untergeordneten Tabelle (child table) zurück, die in Zusammenhang mit der Zeile der übergeordneten Tabelle (parent table) stehen.

Hat eine Tabelle mehr als eine untergeordnete Tabelle (child relation), so muss das Prefix den Tabellennamen enthalten. Beispiel: `Child(OrdersToItemsRelation).Price` referenziert auf die Spalte `Price` in der Tabelle `OrdersToItemsRelation`.

Aggregatfunktionen

Zu den Aggregatfunktionen gehören folgende Funktionen: **SUM**, **COUNT**, **MIN**, **MAX**, **AVG** (berechnet den arithmetischen Mittelwert), **STDEV** (ermittelt die Standardabweichung bezogen auf eine Stichprobe) and **VAR** (berechnet die Varianz bezogen auf eine Stichprobe).

Beispiel	Beschreibung
"Salary > AVG(Salary)"	Alle Menschen mit überdurchschnittlichem Gehalt.
"COUNT(Child.IdOrder) > 5"	Alle Aufträge mit mehr als fünf Artikeln.
"SUM(Child.Price) >= 500"	Alle Aufträge mit einem Gesamtpreis (Summe aller Artikelpreise) größer oder gleich 500€.

Funktionen

Folgende Funktionen werden unterstützt. Weitere Informationen finden Sie unter [Filterfunktionen](#).

- **CONVERT** – Wandelt einen Ausdruck in einen anderen Datentyp um.
- **LEN** – Gibt die Anzahl von Zeichen in einer Zeichenfolge zurück.

- ISNULL – Ersetzt **NULL** durch den angegebenen Ersatzwert.
 - IIF – Gibt einen von zwei Werten zurück, abhängig davon, ob der boolesche Ausdruck **true** oder **false** ergibt.
 - TRIM – Entfernt alle führende und nachfolgende Leerzeichen aus einer Zeichenfolge.
 - SUBSTRING – Gibt eine Zeichenfolge aus der Mitte einer Textzeichenfolge zurück, und zwar ausgehend von einer angegebenen Startposition und Länge.
-

Siehe auch

- [Filterfunktionen](#)
- [SelectRecord-Methode](#)
- [Label-Objekt](#)

Filterfunktionen

Folgende Funktionen werden unterstützt:

CONVERT

Beschreibung	Wandelt einen Ausdruck in einen anderen Datentyp um.
Syntax	CONVERT (<i>expression</i> , <i>type</i>)
Argumente	<i>expression</i> -- Ausdruck, der umgewandelt werden soll. <i>type</i> -- Datentyp, in den der Ausdruck umgewandelt werden soll.
Beispiel	Expression = "Convert (total, 'System.Int32')"

Alle Konvertierungen sind möglich, bis auf die folgenden Ausnahmen: **Boolean** kann nur in **Byte**, **SByte**, **Int16**, **Int32**, **Int64**, **UInt16**, **UInt32**, **UInt64**, **String** und sich selbst umgewandelt werden. **Char** kann nur in **Int32**, **UInt32**, **String**, und sich selbst umgewandelt werden. **DateTime** kann nur in **String** und sich selbst umgewandelt werden. **TimeSpan** kann nur in **String** und sich selbst umgewandelt werden.

LEN

Beschreibung	Gibt die Anzahl von Zeichen in einer Zeichenfolge zurück.
Syntax	LEN (<i>expression</i>)
Argumente	<i>expression</i> -- Zeichenfolge
Beispiel	Expression = "Len (item)"

ISNULL

Beschreibung	Ersetzt NULL durch den angegebenen Ersatzwert.
Syntax	ISNULL (<i>expression</i> , <i>replacementvalue</i>)
Argumente	<i>expression</i> -- Ausdruck, der auf NULL überprüft werden soll. <i>replacementvalue</i> -- Der Ausdruck, der zurückgegeben werden soll, wenn <i>expression</i> NULL ist.
Beispiel	Expression = "IsNull (price, -1)"

IIF

Beschreibung	Gibt einen von zwei Werten zurück, abhängig davon, ob der boolesche Ausdruck true oder false ergibt.
Syntax	IIF (<i>expression</i> , <i>truepart</i> , <i>falsepart</i>)
Argumente	<i>expression</i> -- Boolescher Ausdruck, der ausgewertet werden soll. <i>truepart</i> -- Dieser Wert wird zurückgegeben, wenn <i>expression</i> true ergibt. <i>falsepart</i> -- Dieser Wert wird zurückgegeben, wenn <i>expression</i> false ergibt.
Beispiel	Expression = "IIF (total > 1000, 'expensive', 'dear')"

TRIM

Beschreibung	Entfernt alle führende und nachfolgende Leerzeichen aus einer Zeichenfolge.
Syntax	TRIM (<i>expression</i>)
Argumente	<i>expression</i> -- Zeichenfolge, aus dem die Leerzeichen entfernt werden sollen.

SUBSTRING

Beschreibung	Gibt eine Zeichenfolge aus der Mitte einer Textzeichenfolge zurück, und zwar ausgehend von einer angegebenen Startposition und Länge.
Syntax	SUBSTRING (<i>expression</i> , <i>start</i> , <i>length</i>)
Argumente	<i>expression</i> -- Die Textzeichenfolge, aus der die Zeichen extrahiert werden sollen <i>start</i> -- Die Position des ersten Zeichens, das Sie extrahieren möchten. <i>length</i> -- Die Anzahl der zurückzugebenden Zeichen.
Beispiel	Expression = "SubString (phone, 7, 8)"

Siehe auch

- [Filtersyntax](#)
- [SelectRecord-Methode](#)
- [Label-Objekt](#)

SelectRecord-Methode (string, string)

Sucht den angegebenen Datensatz in der ausgewählten Datenbank.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 6.50.1010

Syntax

Label.SelectRecord (databaseName, filterExpression)

Parameter

databaseName

Typ: String

Name der Datenverbindung

filterExpression

Typ: String

Filterkriterium. Beispiele dazu, wie Datensätze gefiltert werden, finden Sie unter [Filtersyntax](#).

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

➤ [Label-Objekt](#)

SetPropertyValue Method

Setzt den Wert der Eigenschaft.

Namespace: LSOOffice

Assembly: LSOOffice.dll

Version: 4.20.1040

Syntax

```
Label.SetPropertyValue (propertyName, value)
```

Parameter

propertyName

Type: String

Name der Eigenschaft deren Wert geändert werden soll. Eine Liste der möglichen Eigenschaftennamen finden Sie [hier](#).

value

Type: Object

Neuer Eigenschaftswert.

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um Informationen über mögliche Fehler zu erhalten.

Siehe auch

➤ [Label-Objekt](#)

LicenseInfo-Objekt

Ein **LicenseInfo**-Objekt stellt Eigenschaften zum Abrufen der Lizenzinformationen bereit. Verwenden Sie die Eigenschaft [Application.License](#) , um eine Referenz auf das **LicenseInfo**-Objekt zu erhalten.

Eigenschaften

	Name	Beschreibung
	IsTrialVersion	Ruft einen Wert ab, der angibt, ob die Anwendung im Demo-Modus gestartet worden ist.
	LicenseKey	Gibt den Lizenzschlüssel zurück, mit dem Labelstar Office aktiviert worden ist.

IsTrialVersion-Eigenschaft

Ruft einen Wert ab, der angibt, ob die Anwendung im Demo-Modus gestartet worden ist. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

LicenseInfo.IsTrialVersion

Typ

Boolean

Hinweis

In der Testversion wird zu jeder Grafik ein Wasserzeichen hinzugefügt und es werden alle 'e' durch 'x' und alle '5' durch '0' ersetzt.

Siehe auch

➤ [LicenseInfo-Objekt](#)

LicenseKey-Eigenschaft

Gibt den Lizenzschlüssel zurück, mit dem **Labelstar Office** aktiviert worden ist. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

LicenseInfo.LicenseKey

Typ

String

Siehe auch

➤ [LicenseInfo-Objekt](#)

Printer-Objekt

Ein **Printer**-Objekt repräsentiert einen Drucker, der unter Windows installiert ist. Sie erhalten eine Referenz für ein **Printer**-Objekt wenn Sie die Methode [Application.GetPrinter](#) aufrufen.

Siehe auch, [Verwenden des Printer-Objekts](#).

Eigenschaften

	Name	Beschreibung
	Encoding	Gibt die Zeichencodierung des Druckers zurück.
	IsCVPrinter	Ruft einen Wert ab, der angibt, ob der ausgewählte Drucker eine gültiger Carl Valentin-Etikettendrucker ist oder nicht.
	PrinterName	Gibt den Druckernamen zurück.

Methoden

	Name	Beschreibung
	Dispose ()	Gibt den durch das Printer -Objekt belegten Speicher frei.
	Execute (PrintCommand)	Führt einen vordefinierten Druckfehl aus.
	Execute (string)	Führt einen benutzerdefinierten Druckbefehl aus.
	GetStatus ()	Ruft den aktuellen Druckerstatus ab.
	GetStatus (out string)	Ruft den aktuellen Druckstatus sowie zusätzliche Informationen ab.
	SendBytes (byte[])	Sendet Daten an den Drucker.

Dispose-Methode

Gibt den durch das **Printer**-Objekt belegten Speicher frei.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

```
Printer.Dispose ()
```

Hinweis

Wenn Sie ein **Printer**-Objekt nicht mehr benötigen rufen Sie die Methode **Dispose** auf. Andersfalls wird der von dem **Printer**-Objekt belegte Speicher nicht freigegeben.

Siehe auch

- › [Verwenden des Printer-Objekts](#)
- › [Printer-Objekt](#)

Encoding-Eigenschaft

Gibt die Zeichencodierung des Druckers zurück. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

Printer.Encoding

Typ

System.Text.Encoding

Siehe auch

➤ [Printer-Objekt](#)

IsCVPrinter-Eigenschaft

Ruft einen Wert ab, der angibt, ob der ausgewählte Drucker eine gültiger Carl Valentin-Etikettendrucker ist oder nicht. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

Printer.IsCVPrinter

Typ

Boolean

Hinweis

Ist der Drucker ein gültiger Carl Valentin-Etikettendrucker können spezielle Druckbefehle an den Drucker übertragen werden, um z.B. einen Druckauftrag anzuhalten, fortzusetzen oder abubrechen.

Siehe auch

- [Execute-Methode \(PrintCommand\)](#)
- [Execute-Methode \(string\)](#)
- [Verwenden des Printer-Objekts](#)
- [Printer-Objekt](#)

Execute-Methode (PrintCommand)

Führt einen vordefinierten Druckfehl aus. Nur Carl Valentin-Etikettendrucker.

Namespace: LSOoffice

Assembly: LSOoffice.dll

Version: 6.20.1010

Syntax

Printer.Execute (command)

Parameter

command

Typ: LSOoffice.PrintCommand

	Name	Value	Description
	LabelFeed	1 (0x01)	Führt einen Etikettenvorschub aus.
	TestPrint	2 (0x02)	Führt einen Testdruck aus.
	StatusPrint	3 (0x03)	Druckt den Statusbericht aus.
	MeasureLabel	4 (0x04)	Etiketten messen.
	StopPrinting	5 (0x05)	Druckauftrag anhalten.
	ContinuePrinting	6 (0x06)	Druckauftrag fortsetzen.
	CancelPrinting	7 (0x07)	Druckauftrag abbrechen.

Rückgabewert

Typ: Boolean

Gibt **true** zurück wenn der Befehl erfolgreich an den Drucker übertragen worden ist, sonst **false**. Um weitere Fehlerinformationen zu erhalten, überprüfen Sie den Wert der Eigenschaft [Application.LastError](#).

Siehe auch

- › [Execute-Methode \(string\)](#)
- › [Verwenden des Printer-Objekts](#)
- › [Printer-Objekt](#)

Execute-Methode (string)

Führt einen benutzerdefinierten Druckbefehl aus. Nur Carl Valentin-Etikettendrucker.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

Printer.Execute (command)

Parameter

command

Typ: String

Benutzerdefinierter Druckbefehl.

Beispiel: "(SOH)FF----r(ETB)"

Jeder Befehl beginnt mit einem Startzeichen (SOH), gefolgt von den zu übertragenden Zeichen und am Schluß ein Stoppzeichen (ETB).

Bitte wenden Sie sich an Ihren Händler, um eine vollständige Dokumentation der unterstützten Druckbefehle zu erhalten.

Rückgabewert

Typ: Boolean

Gibt **true** zurück wenn der Befehl erfolgreich an den Drucker übertragen worden ist, sonst **false**. Um weitere Fehlerinformationen zu erhalten, überprüfen Sie den Wert der Eigenschaft [Application.LastError](#).

Siehe auch

- [Execute-Methode \(PrintCommand\)](#)
- [Verwenden des Printer-Objekts](#)
- [Printer-Objekt](#)

GetStatus-Methode

Ruft den aktuellen Druckerstatus ab.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

Printer.GetStatus ()

Rückgabewert

Type: LSOoffice.PrinterStatus

	Name	Wert	Beschreibung
	Other	1 (0x01)	
	Unknown	2 (0x02)	
	Idle	3 (0x03)	
	Printing	4 (0x04)	
	Warmup	5 (0x05)	
	Stopped	6 (0x06)	
	Offline	7 (0x07)	
	Error	8 (0x08)	

Siehe auch

- [Verwenden des Printer-Objekts](#)
- [Printer-Objekt](#)

GetStatus-Methode (out string)

Ruft den aktuellen Druckstatus sowie zusätzliche Informationen ab.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.20.1010

Syntax

Printer.GetStatus (out message)

Parameter

message [out]

Typ: String

Zusätzliche Statusinformationen, z.B. die Fehlermeldung, die im Druckerdisplay angezeigt wird.

Rückgabewert

Type: LSOoffice.PrinterStatus

	Name	Wert	Beschreibung
	Other	1 (0x01)	
	Unknown	2 (0x02)	
	Idle	3 (0x03)	
	Printing	4 (0x04)	
	Warmup	5 (0x05)	
	Stopped	6 (0x06)	
	Offline	7 (0x07)	
	Error	8 (0x08)	

Siehe auch

- › [Verwenden des Printer-Objekts](#)
- › [Printer-Objekt](#)

PrinterName-Eigenschaft

Gibt den Druckernamen zurück. Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 6.20.1010

Syntax

Printer.PrinterName

Typ

String

Siehe auch

➤ [Printer-Objekt](#)

SendBytes-Methode

Sendet Daten an den Drucker.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 6.20.1010

Syntax

Printer.SendBytes (bytes)

Parameter

bytes

Typ: byte[]

Byte-Array, der die Daten enthält, die an den Drucker gesendet werden sollen.

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

➤ [Printer-Objekt](#)

PrintStack-Objekt

Ein **PrintStack**-Objekt repräsentiert einen geöffneten Druckstapel. Sie erhalten eine Referenz für ein **PrintStack**-Objekt wenn Sie die Methode [Application.OpenPrintStack](#) aufrufen.

Eigenschaften

	Name	Beschreibung
	FilePath	Gibt den Pfad des geöffneten Druckstapels zurück.

Methoden

	Name	Beschreibung
	Print ()	Druckt den geöffneten Druckstapel.

FilePath-Eigenschaft

Gibt den Pfad des geöffneten Druckstapels zurück. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 6.50.1010

Syntax

PrintStack.FilePath

Typ

String

Siehe auch

➤ [PrintStack-Objekt](#)

Print-Methode

Druckt den geöffneten Druckstapel.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 6.50.1010

Syntax

PrintStack.Print ()

Hinweis

Überprüfen Sie die Eigenschaft [Application.LastError](#) um festzustellen, ob die Methode erfolgreich abgeschlossen werden konnte.

Siehe auch

➤ [PrintStack-Objekt](#)

VersionInfo-Objekt

Ein **VersionInfo**-Objekt stellt Eigenschaften zum Abrufen von Informationen über die API bereit, z.B. Versionsnummer, Copyright, usw. Verwenden Sie die Eigenschaft [Application.Info](#) , um eine Referenz auf das **VersionInfo**-Objekt zu erhalten.

Eigenschaften

	Name	Description
	CompanyName	Gibt den der API zugeordneten Firmennamen zurück.
	CompiledVersion	Interne Versionsnummer der API (dieses Feld ist nur für den internen Gebrauch).
	Copyright	Gibt den der API zugeordneten Copyrightvermerk zurück.
	DisplayVersion	Gibt die Versionsnummer der API zurück.
	ProductName	Gibt den der API zugeordneten Produktnamen zurück.

CompanyName-Eigenschaft

Gibt den der API zugeordneten Firmennamen zurück. Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

VersionInfo.CompanyName

Typ

String

Siehe auch

➤ [VersionInfo-Objekt](#)

CompiledVersion-Eigenschaft

Interne Versionsnummer der API (dieses Feld ist nur für den internen Gebrauch). Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

VersionInfo.CompiledVersion

Typ

String

Beispiel

Compiled Version: "4.10.1010"

Siehe auch

➤ [VersionInfo-Objekt](#)

Copyright-Eigenschaft

Gibt den der API zugeordneten Copyrightvermerk zurück. Nur-Lese-Eigenschaft.

Namespace: LSOoffice
Assembly: LSOoffice.dll
Version: 4.10.1010

Syntax

VersionInfo.Copyright

Typ

String

Siehe auch

➤ [VersionInfo-Objekt](#)

DisplayVersion-Eigenschaft

Name des Produkts. Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

VersionInfo.DisplayVersion

Typ

String

Beispiel

Display Version: "Version 4.10 Build 1010"

Siehe auch

➤ [VersionInfo-Objekt](#)

ProductName-Eigenschaft

Gibt den der API zugeordneten Produktnamen zurück. Nur-Lese-Eigenschaft.

Namespace: LSOOffice
Assembly: LSOOffice.dll
Version: 4.10.1010

Syntax

VersionInfo.ProductName

Typ

String

Siehe auch

➤ [VersionInfo-Objekt](#)

Fehlercodes

Fehlerbehandlung

Bei der Fehlerbehandlung ist es wichtig zu wissen was den Fehler ausgelöst hat. Wurde der Fehler von einer Automations-Methode oder -Eigenschaft ausgelöst können Sie detaillierte Fehlerinformationen erhalten indem Sie die Eigenschaft [Application.LastError](#) abfragen. Jeder Aufruf einer Automations-Methode oder -Eigenschaft (ausser LastError) setzt die Fehlerinformationen zurück, so dass [Application.LastError](#) immer die Fehlerinformationen für die zuletzt aufgerufene Methode oder Eigenschaft enthält.

Fehlercodes

Fehlercode	Beschreibung	Typ
1000	Windows Fehler.	Fehler
1001	Die Anwendung ist bereits initialisiert.	Warnung
1002	Ungültige Lizenznummer.	Warnung
1003	Ungültiger Lizenztyp; Professional Lizenz erwartet.	Warnung
1004	Die Anwendung ist nicht initialisiert; rufen Sie zuerst Initialize() auf.	Fehler
1005	Es ist keine Datei geöffnet.	Fehler
1006	Ungültiger Feldname.	Fehler
1007	Kein Feld ausgewählt.	Fehler
1008	Der Feldindex ist außerhalb des gültigen Bereichs.	Fehler
1009	Ungültiger Eigenschaftename.	Fehler
1010	Keine Datenbankfelder auf dem Etikett definiert.	Fehler
1011	Kein Datensatz gefunden.	Fehler
1012	Es sind mehrere Datenbankverbindungen auf dem Etikett definiert.	Fehler
1013	Ungültiger Druckertyp; Carl Valentin-Etikettendrucker erwartet.	Fehler
1014	Ungültiger Druckername.	Fehler
1015	Kein Drucker ausgewählt.	Fehler
1016	Generischer Fehler.	Fehler
1017	Labelstar Office ist auf diesem Computer nicht installiert.	Fehler
1018	Ungültiger Datenbankname.	Fehler

Kontakte

Produkt-Webseite

Zusätzliche Informationen zu **Labelstar Office** und die aktuellste Programmversion finden Sie auf unserer Webseite:
www.carl-valentin.de

E-Mail

Technischer Support: support@carl-valentin.de
Bestell- und Lizenzierungsanfragen: order@carl-valentin.de
Allgemeine Anfragen: info@carl-valentin.de

Systemanforderungen

Minimale Systemanforderungen

- Microsoft Windows 7 SP1/8/8.1/10 x86/x64
- .Net Framework 4.6 oder höher (Download unter <http://www.microsoft.com/net/>)
- Microsoft Visual C++ 2015 Redistributable (x86) (Download unter <https://www.microsoft.com/en-us/download/details.aspx?id=53587>)
- Microsoft Access Database Engine 2010 (x86)
- Empfohlene Druckertreiber: [Carl Valentin Druckertreiber](#) Version 2.4.1 oder höher

Hinweis

Einige Komponenten, wie z.B. das .NET Framework, sind standardmäßig nicht im Installationsprogramm enthalten. Während der Installation sucht das Programm nach den Komponenten, lädt sie ggf. aus dem Internet herunter und installiert sie. Falls Sie keinen Internetzugang haben finden Sie die erforderlichen Komponenten auf der Programm-CD im Ordner *Utilities*.

Impressum

Carl Valentin GmbH
Neckarstrasse 78-86 u. 94
78056 Villingen-Schwenningen

Telefon: +49 (0) 7720 9712 - 0
E-Mail: info@carl-valentin.de

Copyright © 2019 Carl Valentin GmbH

Alle Rechte vorbehalten. Kein Teil dieses Handbuchs darf in irgendeiner Form ohne ausdrückliche schriftliche Genehmigung reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Haftungsausschluss

Die Beschreibungen in diesem Handbuch stellen keine zugesicherten Eigenschaften im rechtlichen Sinne oder im Sinne der Produkthaftung dar. Die Autoren behalten sich das Recht vor, an der Software Änderungen vorzunehmen, ohne Verpflichtung diese Änderungen irgendeiner Person bekanntzugeben. Es wird keine Garantie für die Richtigkeit des Inhalts dieses Handbuchs übernommen. Da sich Fehler trotz aller Bemühungen nie vollständig vermeiden lassen, sind wir für Hinweise jederzeit dankbar.

Warenzeichenhinweise

Alle in diesem Handbuch erwähnten Produktbezeichnungen können Warenzeichen oder eingetragene Warenzeichen der entsprechenden Inhaber sein.